REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

FACULTÉ DES MATHEMATIQUES ET DE l'INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER

Spécialité : Genie Informtique

Par :
DROUI Aicha
SERARDI Amina

Sur le thème

Optimisation des Niveaux de Stock dans le Commerce de Détail grâce à l'Apprentissage Automatique et l'Analyse de Séries Temporelles.

Soutenu publiquement le

20/ 06 / 2024 à Tiaret devant le jury composé de :

Mr MOKHTARI Ahmed	MAA Université de Tiaret	Président
Mr MAASKRI Mustapha	MCB Université de Tiaret	Encadrant
Mr MOSTEFAOUI Kadda	MCA Université de Tiaret	Examinateur

2023-2024

Remerciements

Premièrement, nous remercions Dieu Tout-Puissant, qui nous a accordé la force, la détermination et la guidance nécessaires pour mener à bien ce travail.

Ensuite, nos sincères remerciements vont à notre encadreur **MAASKRI MUSTAPHA**, dont les conseils précieux, le soutien constant et la patience ont été essentiels à la réussite de ce projet. Merci de nous avoir guidés et encouragés tout au long de ce parcours.

Nous exprimons également notre profonde gratitude au Président du jury **MOKHTARI AHMED**, ainsi qu'à l'Examinateur **MOSTEFAOUI Kadda**. Vos remarques constructives et votre rigueur académique ont été d'une grande valeur pour nous. Merci pour le temps et l'attention que vous avez consacrés à l'évaluation de notre travail.

Nous tenons à exprimer notre gratitude à tous les professeurs qui nous ont accompagnés tout au long de notre parcours académique. Chacun de vous a contribué de manière significative à notre développement intellectuel et personnel. Merci pour vos enseignements, vos conseils et votre bienveillance.

Enfin, nous remercions chaleureusement nos familles, nos frères et sœurs, qui ont été un pilier de soutien tout au long de ce voyage. Votre amour, vos encouragements et votre soutien inconditionnel ont été une source de motivation constante. Merci de croire en nous et de nous accompagner à chaque étape.

Merci à tous ceux qui ont contribué à l'élaboration de ce travail de près ou de loin et qui méritent d'y trouver leur nom.

Dédicaces

♡ Je dédie ce travail ♡

À mes parents, pour leur amour inconditionnel, leur soutien indéfectible et leurs sacrifices sans fin.

À mes frères Amar , Hemida et Sadak qui ont toujours été là pour moi. À mes sœurs Zahira , Fatiha et Djadla, qui sont bien plus que des membres de ma famille.

À mes neveux , qui apportent une joie incommensurable à ma vie. Et à mes nièces, qui ajoutent une touche de douceur et d'amour à notre famille surtout Khadidja .

À mes chers amis, je vous remercie pour votre sincère amitié et vos encouragements.

Je n'oublie pas ceux qui nous ont quittés et qui ont laissé un vide dans ma vie, ceux dont j'avais toujours l'habitude de la présence et de l'aide dans les moindres détails : ma grand-mère, mon frère Jaloul, ma sœur Louisa, le mari de ma sœur Mohamed. Que Dieu ait pitié d'eux et les fasse entrer au paradis.

Et enfin, à Aicha, mon chère amie avant de devenir mon binôme, qui a partagé ce voyage avec moi. Merci pour votre confiance et votre coopération

SERARDI Amina

Dédicaces

Louange à Dieu tout puissant, qui m'a permis de voir ce jour tant attendu.

♡_____ Je dédie ce travail _____♡

À mon cher grand-père, qui a été un modèle et une source d'inspiration pour moi; Grâce à ta sagesse et à tes prières, j'ai toujours senti que j'étais sur la bonne voie.

À mes chers parents : Aucune dédicace ne saurait exprimer mon respect, mon amour éternel, ma reconnaissance et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être. Votre patience sans fin, votre compréhension et votre encouragement sont pour moi le soutien indispensable que vous avez toujours su m'apporter. Puisse Dieu, le Très Haut, vous accorder la santé, le bonheur et la longue vie et vous protège de tout mal.

À mon frère «Djilali» ; Merci pour ton soutien. Merci pour votre bonne humeur. Tu comptes énormément pour moi et je te souhaite tout le bonheur du monde.

À mes sœurs Fatima , Fadhila ,Houria ,Souhila, Afaf et Chaima, pour leurs soutient morale , pour leurs encouragements et les sacrifices qu'elles ont endurés dans les moments les plus ardus.

En témoignage de mon affection fraternelle, de ma profonde tendresse et reconnaissance, je vous souhaite une vie pleine de bonheur et de succès et que Dieu, le tout puissant, vous protège et vous garde. Je vous aime trop.

À mes neveux : « Abdelhamid » ET « Youcef»; qu'Allah vous

protège et vous accorde une bonne santé.

À toute ma famille Droui et Adim.

À Amina , ma chère amie qui a partagé ce voyage avec moi et est devenue mon binôme. Je vous suis reconnaissante pour la confiance que vous m'avez accordée et pour notre collaboration.

À tous mes amis, mes collègues.

À ceux qui sont entrain de lire ces lignes. Pour leur appui moral et leurs conseils.

DROUI Aicha

ملخص

اليوم، يعد تحسين مستويات المخزون في التجارة بالتجزئة أمرًا ضروريًا لتقليل التكاليف مع زيادة رضا العملاء إلى أقصى حد. يعتمد هذا التحسين على التنبؤ الدقيق بالطلب المستقبلي، مما يسمح بتجنب كل من الفائض في المخزون ونفاد المخزون. لتحقيق ذلك، من الضروري تحليل الاتجاهات الموسمية، واستخدام البيانات التاريخية، وأخذ الأحداث الخارجية في الاعتبار

في دراستنا، قمنا بتنفيذ تقنيات متقدمة للتعلم الآلي للتنبؤ بالسلاسل الزمنية. إستخدمنا مجموعة متنوعة من خوارزميات التعلم الآلي والتعلم العميق، مثل خوارزمية أقرب الجيران ك ، الغابة العشوائية ، الانحدار الخطي، التعزيز التدريجي المتطرف، والشبكات العصبية العميقة . سمحت لنا هذه الأساليب بمقارنة أداء هذة الخوارزميات المختلفة

الكلمات المفتاحية : نماذج التعلم الآلي، التنبؤ بالطلب، تقييم النماذج، تحليل السلاسل الزمنية، تكامل النماذج، التنبؤ المحسن

Abstract

Today, optimizing stock levels in retail is essential for minimizing costs while maximizing customer satisfaction. This optimization relies on accurately predicting future demand, allowing both overstock and stockouts to be avoided.
To achieve this, it is crucial to analyze seasonal trends, use historical data, and take external events into account.
In our study, we implemented advanced machine learning techniques for time series forecasting. We employed various machine learning and deep learning algorithms, such as k-nearest neighbors, Random Forest, linear regression, XGBoost (eXtreme Gradient Boosting), and deep neural networks (DNN). These approaches allowed us to compare the performance of these different algorithms.

Keywords: Machine Learning Models, Demand Forecasting, Model Evaluation, Time Series Analysis, Model Integration, Enhanced Forecasting.

Résumé

Aujourd'hui ,l'optimisation des niveaux de stock dans le commerce de détail est essentielle pour minimiser les coûts tout en maximisant la satisfaction client. Cette optimisation repose sur la prédiction précise de la demande future, permettant d'éviter à la fois les surstocks et les ruptures de stock.
Pour y parvenir, il est crucial d'analyser les tendances saisonnières, d'utiliser les données historiques et de prendre en compte les événements externes.
Dans notre étude, nous avons mis en œuvre des techniques avancées d'apprentissage automatique pour la prévision des séries temporelles. Nous avons employé divers algorithmes d'apprentissage automatique et d'apprentissage profond, tels que les k plus proches voisins , les forêts aléatoires (Random Forest), la régression linéaire, XGBoost (eXtreme Gradient Boosting) et les réseaux de neurones profonds (DNN). Ces approches nous ont permis de comparer les performances de ces différents algorithmes.

Mots clés: Modèles d'apprentissage automatique, Prévision de la demande, Évaluation des modèles, Analyse de séries temporelles, Intégration de modèles, Prévision améliorée.

Table des matières

Remerciements **2**

Dédicace **2**

Liste des figures **16**

Liste des Tableaux **17**

Liste d'abréviations **17**

Introduction générale **19**

1 État de l'art **22**
1.1 Introduction 22
1.2 Approvisionnement 22
1.3 Définition des stocks 22
1.3.1 Les coûts liés au stock 23
1.3.2 Les niveaux de stock 23
1.3.3 Les mouvements des stocks 26
1.4 Définition de la gestion des stocks 27
1.5 La relation de la gestion des stocks avec les autres fonctions de l'entreprise 27
1.6 Définition de la gestion des stocks du commerce de détail . . 28
1.7 L'importance de la gestion des stocks du commerce de détail 28
1.8 Différence entre la gestion des stocks de vente au détail et les autres types de gestion des stocks 29
1.9 Techniques de gestion des stocks du commerce de détail . . 29
1.9.1 L'évaluation des stocks 29
1.9.2 Le rapprochement régulier 29
1.9.3 Le comptage des stocks physiques 29
1.9.4 Les comptages périodiques 30
1.9.5 L'analyse ABC 30
1.9.6 Les contrôles ponctuels 30
1.10 Analyse et prévision des stocks pour les dé- taillants 30
1.10.1 La prévision qualitative 31
1.10.2 L'analyse de séries chronologiques 31

1.10.3 L'apprentissage automatique 31
1.11 Conclusion 31

2 L'intelligence artificielle et ses applications 32
2.1 Introduction 32
2.2 Définition de l'intelligence artificielle 32
2.2.1 Les disciplines qui influencent l'IA 33
2.2.2 Les domaines de recherche en IA 34
2.2.3 Applications de l'IA 35
2.3 Les avantages et les risques de l'intelligence artificielle . . . 36
2.3.1 Les avantages 36
2.3.2 Les inconvénients 36
2.4 Concepts et Sources de l'apprentissage automatique 37
2.5 L'apprentissage automatique 37
2.5.1 Modélisation 37
2.5.2 Les phases de l'apprentissage automatique 38
2.5.3 Les étapes de l'apprentissage automatique 38
2.5.4 Types d'apprentissage automatique 40
2.5.5 Les algorithmes utilisés 41
2.5.6 Les applications de l'apprentissage automatique . . . 41
2.6 Apprentissage automatiques supervisé 41
2.6.1 Les différents types d'apprentissage automatiques supervisé 42
2.6.2 La classification 42
2.6.3 La régression 44
2.6.4 Types de régression linéaire 44
2.7 L'apprentissage non-supervisé 45
2.7.1 Clustering 46
2.7.2 La réduction de la dimensionnalité 47
2.7.3 La détection des anomalies 47
2.7.4 L'apprentissage des règles d'association 47
2.8 L'apprentissage par renforcement 48
2.9 L'apprentissage par transfert 48
2.10 Récapitulatif des différents types d'apprentissage 49
2.11 Les modèles de l'apprentissage automatique 49
2.11.1 Algorithmes des K plus proches voisins 49
2.11.2 Arbre de décision 50
2.11.3 Réseaux de neurones artificiels 51
2.11.4 Machine à support de vecteur (SVM) 53
2.11.5 Modèle de mélange de gaussien 54
2.12 La relation entre machine-Learning et Deep Learning 54
2.13 L'apprentissage en profondeur 55
2.13.1 Domaine d'application 55
2.13.2 Fonctionnement du Deep Learning 55
2.14 les mesures de performance 56
2.14.1 La matrice de confusion 56
2.14.2 Accuracy 57
2.14.3 Précision 57

2.14.4 Le rappel (Recall) . . . 57
2.14.5 Le score F1 (F1-score) . . . 58
2.14.6 Taux d'erreur (Error Rate) . . . 58
2.14.7 MAE (Mean Absolute Error) . . . 58
2.14.8 MSE (Mean Squared Error) . . . 58
2.14.9 RMSE (Root Mean Squared Error) . . . 58
2.14.10 R2 (Coefficient de détermination) . . . 59
2.15 L'overfitting et l'underfitting . . . 59
2.15.1 Surapprentissage . . . 59
2.15.2 Sous-apprentissage . . . 60
2.15.3 Différence entre underfitting et overfitting . . . 60
2.15.4 Le right fitting . . . 61
2.16 Conclusion . . . 61

3 Les séries temporelles 62
3.1 Introduction . . . 62
3.2 Définition de série temporelle . . . 62
3.3 Les objectifs de l'analyse des séries temporelles . . . 62
3.4 Les composantes d'une série temporelle . . . 62
3.4.1 Tendance . . . 62
3.4.2 Saisonnalité . . . 63
3.4.3 Cycle . . . 63
3.4.4 Résidus . . . 63
3.4.5 Stationnarité . . . 64
3.5 Techniques de décomposition-recomposition . . . 64
3.5.1 Le schéma additif . . . 64
3.5.2 Le schéma multiplicatif . . . 65
3.5.3 Le schéma multiplicatif complet . . . 66
3.6 Intérêt des séries temporelles . . . 67
3.6.1 Description . . . 67
3.6.2 Explication . . . 67
3.6.3 Prévision . . . 68
3.7 Exemples de séries temporelles . . . 68
3.8 Calcule de prévision . . . 69
3.8.1 Méthode de Buys-Ballot . . . 69
3.8.2 Lissage exponentiel . . . 71
3.9 Analyse et caractérisation des séries temporelles . . . 71
3.9.1 Fonction d'auto-correlation . . . 71
3.9.2 Fonction d'autocorrélation partielle (PACF) . . . 72
3.10 Stationnarité des séries temporelles . . . 72
3.10.1 Stationnarité strict . . . 72
3.10.2 Stationnarité du second ordre . . . 73
3.11 Les modèles de prédiction . . . 73
3.11.1 Les modèles Auto-Régressifs . . . 73
3.12 Apprentissage automatique pour la prédiction temporelle . 75
3.12.1 Prédire avec la régression linéaire . . . 75
3.12.2 Prédire avec Random Forest . . . 75
3.12.3 Prédire avec XGBoost . . . 76

3.12.4 Prédire avec KNN 76
3.12.5 Prédire avec Les réseaux de neurones 77
3.13 conclution 77

4 Expérimentations et resultats obtenus 78
4.1 Introduction 78
4.2 Outlies de development: 78
4.3 Importation des bibliothèques 79
4.4 L'ensemble de données (Dataset) 80
4.4.1 Importation de la base de données 81
4.4.2 Description des fichiers de données 81
4.5 Gestion les valeurs manquantes dans l'ensemble de données des fonctionnalités 82
4.5.1 Résultat 83
4.6 Fusion de l'ensemble de données data avec l'ensemble de données stores 84
4.6.1 Résultat 85
4.7 La détection des valeurs aberrantes 85
4.7.1 Résultat 86
4.8 Fusion des données et suppression des valeurs manquantes 86
4.8.1 Résultat 86
4.9 la forme du DataFrame avant et après la suppression des valeurs aberrantes. 87
4.10 Les ventes hebdomadaires négatives (Negative Weekly Sales) 87
4.10.1 Résultat 88
4.11 Les ventes hebdomadaires 88
4.11.1 Résultat 89
4.12 Enregistre le DataFrame 89
4.13 Les visualisations de données 90
4.13.1 Résultat 90
4.14 Charger des données depuis un fichier CSV 91
4.14.1 Résultat 91
4.15 Nombre Total de Magasins 91
4.16 Nombre Total de Départements 92
4.17 Dates des Semaines de Collecte des Données 92
4.17.1 Résultat 93
4.18 Remplacer les dates par le numéro 94
4.18.1 Résultat 94
4.19 Différence des Ventes Moyennes pendant les Semaines de Vacances 95
4.19.1 Résultat 95
4.20 Ventes Mensuelles pour Chaque Année 96
4.21 Ventes Moyennes Hebdomadaires par Magasin 96
4.22 Ventes Moyennes par Département 97
4.23 Ventes vs Température 97
4.24 La distribution des vacances 98
4.25 La décomposition de la séries temporelle 98
4.26 Encodage One-hot plus Normalisation des Données 99

4.27 La corrélation entre les caractéristiques (features) 100
4.28 Recursive Feature Elimination (RFE) 100
4.29 modèle de régression de forêt aléatoire avec les paramètres spécifiés 101
4.29.1 Paramètre 101
4.30 L'importance de caractéristiques dans le modèle de forêt aléatoire 102
4.30.1 Résultat 102
4.31 Caractéristiques les plus importantes classées par le modèle de forêt aléatoire 103
4.31.1 Caractéristiques les plus importantes 103
4.32 division de DataFrame 104
4.33 Les modèles de machine learning 105
4.33.1 Modèle de régression linéaire 105
4.33.2 Modèle de régression par forêts aléatoires 108
4.33.3 Modèle de régression par KNN 109
4.33.4 Modèle de XGboost 111
4.33.5 Modèle de Réseau neuronal profond personnalisé DNN 113
4.33.6 Résultat 113
4.34 Comparaison des modèles 115
4.34.1 Accuracy 115
4.34.2 Correlation coefficient 116
4.34.3 F1 Score 117
4.35 Overfiting 118
4.35.1 Le modèle DNN 118
4.35.2 Analyse de résultats 119
4.36 Analyse de la Précision pour Chaque Modèle 120
4.37 Etude de cas(interprétation) 121
4.38 Exemple d'analyse des résultats des Ventes 121
4.38.1 Hypothèses 122
4.38.2 Stock de sécurité 122
4.38.3 Stock minimum : 122
4.38.4 Stock maximum: 122
4.38.5 Stock Moyen (SMO) 123
4.39 Conclusion 123

Table des figures

1.1 Les stocks, une régulation de flux [MAR13]. 23
1.2 représentation graphique du stock minimum ,maximum , de sécurité et d'alerté [Ass21]. 25
1.3 Gestion du stock et au cœur des autres fonctions de l'entreprise [Bet21]. 27

2.1 Schéma de décomposition du domaine de l'intelligence artificielle et de ces sous- domaines [RB22] 33
2.2 Panorama des domaines de l'IA [**?**]. 35
2.3 Schéma de modélisation d'une machine d'apprentissage . . 37
2.4 apprentissage supervisé & apprentissage non-supervisé [EN21]. 40
2.5 Schéma du fonctionnement de l'apprentissage supervisé [Mar20]. 42
2.6 La classification [EN21]. 42
2.7 La regression [EN21]. 44
2.8 Deux modèles de régression linéaire [DSSQ22]. 45
2.9 Schéma de fonctionnement de l'apprentissage non supervisé [Mar20]. 46
2.10 Clustering [BEL20]. 46
2.11 Détection d'anomalies [BEL20]. 47
2.12 Schéma du fonctionnement de l'apprentissage par renforcement [Mar20]. 48
2.13 Exemple de division d'un arbre de classification [LL20]. . . . 51
2.14 Schéma d'un exemple de réseau de neurones[Mar20] 51
2.15 L'hyperplan H optimale, vecteurs supports et marge maximale [LS20]. 53
2.16 Relation entre IA, machine-Learning et Deep Learning[Nes22]. 54
2.17 Différence entre un réseau de neurone simple et un réseau profond [EN21]. 56
2.18 Surapprentissage ,Ajustement Correct et Sous-apprentissage dans les modèles de classification et de régression [Mat24]. . 61

3.1 Représentation graphique d'un schéma additif [Sam20]. . . . 65
3.2 Représentation graphique d'un schéma multiplicatif [Sam20]. 66
3.3 Représentation graphique d'un schéma multiplicatif complet [Sam20]. 67

3.4 évolution du cours du Dow Jones entre 1928 et 2004 69
3.5 évolution de l'intensité du vent, en m/s, au large de la Bretagne entre 1979 et 2001 [Mon17]. 69

4.1 Importation des bibliothèques utilisées première partie . . . 79
4.2 Importation des bibliothèques utilisées deuxième partie . . . 80
4.3 Importation des bibliothèques utilisées troisième partie . . . 80
4.4 Code de lecture du fichier CSV. 81
4.5 Le jeu de données contient des données supplémentaires sur les magasins. 81
4.6 Le jeu de données contient les informations sur les magasins. 82
4.7 code de gestion les valeurs manquantes 82
4.8 Résultat de gestion des valeurs manquantes 83
4.9 Code de la fusion et le suppression de l'ensemble de données data avec l'ensemble de données stores. 84
4.10 Résultat suppression de la colonne 'IsHoliday-x'. 85
4.11 Code de la détection des valeurs aberrantes. 85
4.12 Code de La fusion des données et de la suppression des valeurs manquantes. 86
4.13 Résultat de la Fusion des données et la suppression des valeurs manquantes. 86
4.14 Code de la forme du DataFrame avant et après la suppression des valeurs aberrantes. 87
4.15 Code de ventes hebdomadaires négatives 87
4.16 Résultat de ventes hebdomadaires négatives. 88
4.17 Code de ventes hebdomadaires 88
4.18 Résultat de ventes hebdomadaires 89
4.19 Le code qui enregistre le DataFrame prétraité 89
4.20 Code de Ventes Moyennes Mensuelles 90
4.21 Résultat de Ventes Moyennes Mensuelles 90
4.22 Code qui charger des données depuis train.csv 91
4.23 Résultat de chargement des données depuis train.csv 91
4.24 Nombre Total de Magasins. 91
4.25 Nombre Total de Départements 92
4.26 Dates des Semaines de Collecte des Données 92
4.27 Résultat de dates des Semaines de Collecte des Données . . . 93
4.28 Code qui remplacer les dates par le numéro 94
4.29 Résultat de remplacement les dates par le numéro 94
4.30 Code qui comparer les ventes hebdomadaires 95
4.31 Résultat de comparaision les ventes hebdomadaires 95
4.32 Ventes Mensuelles pour Chaque Année 96
4.33 Les graphes de ventes Mensuelles pour Chaque Année . . . 96
4.34 Ventes Moyennes Hebdomadaires par Magasin 96
4.35 Ventes Moyennes par Département 97
4.36 Ventes vs Température 97
4.37 La distribution des vacances 98
4.38 La décomposition de la séries temporelle 99
4.39 Matrice de corrélation 100

4.40 Code de recursive feature elimination 100
4.41 Code qui recherche le meilleur nombre d'estimateurs 101
4.42 Code de régression de forêt aléatoire avec les paramètres spécifiés 101
4.43 Résultat de régression de forêt aléatoire avec les paramètres spécifiés 101
4.44 Code qui classe les caractéristiques 102
4.45 Résultat de classe les caractéristiques 102
4.46 Code de caractéristiques les plus importantes classées par le modèle de forêt aléatoire 103
4.47 Code de caractéristiques les plus importantes 103
4.48 Résultat de caractéristiques les plus importantes 104
4.49 Code de division de DataFrame 104
4.50 suite de code qui divise de DataFrame 104
4.51 Code de régression linéaire 105
4.52 Distribution des valeurs de ventes réelles 106
4.53 Prédiction du régression linéaire 107
4.54 graphique de prédiction du régression linéaire 107
4.55 Code de modèle de régression par forêts aléatoires 108
4.56 Prédiction du Random Forest 108
4.57 graphique de prédiction du Random Forest 109
4.58 Modèle de régression par KNN 109
4.59 Prédiction du KNN 110
4.60 graphique de prédiction du KNN 110
4.61 Modèle de XGboost 111
4.62 prédiction du XGboost. 112
4.63 graphique de prédiction du XGboost. 112
4.64 Code de modèle de DNN 113
4.65 Résultat de modèle de DNN 113
4.66 Prédiction du DNN 114
4.67 graphique de prédiction du DNN 114
4.68 Comparaison des modèles avec Accuracy. 115
4.69 graphique de omparaison des modèles avec Accuracy. . . . 115
4.70 Comparaison des modèles avec correlation coefficient 116
4.71 graphique de comparaison des modèles avec correlation coefficient. 116
4.72 Comparaison des modèles avec F1 Score 117
4.73 graphique de comparaison des modèles avec F1 Score. . . . 117
4.74 Test de overfiting et Underfitting pour Les 4 modèles 118
4.75 graphique de test de overfiting et Underfitting pour Les 4 modèles 118
4.76 Test de overfiting et Underfitting pour modèle DNN 118
4.77 graphique de Test de overfiting et Underfitting pour modèle DNN 119
4.78 Exemple de prédiction du XGboost. 121

Liste des tableaux

2.1 Récapitulatif des différents types d'apprentissage. 49
2.2 Matrice de confusion 57

3.1 Population totale du Sénégal 68
3.2 Exemple de constitution d'un tableau de Buys-Ballot pour des ventes trimestrielles 70
3.3 Classement des trimestres en fonction de leurs valeurs [TB98]. 70
3.4 Exemple de calcul d'une autocorrélation 72

4.1 Résultat de la détection des valeurs aberrantes. 86
4.2 Résultats obtenus avec LR 106
4.3 Résultats obtenus avec Random Forest 108
4.4 Résultats obtenus avec KNN 109
4.5 Résultats obtenus avec XGboost 111
4.6 Résultats obtenus avec DNN 113
4.7 L'entraînement et le test de chaque modèle 119
4.8 Test de underfitting et overfitting de chaque modéle 120

Liste d'abréviations

PMEs	Petites et Moyennes Entreprises
SI	Le stock initial
SF	Le stock final
SM	Le stock minimum
SS	Stock de sécurité
JAT	Le juste à temps
MRP	Material Requirements Planning
DDMRP	Demand Driven Material Requirements Planning
UPC	Universal Product Code
UCC	Uniform Code Council
PDV	Point de Vente
L'IRS	l'Internal Revenue Service
FIFO	First In First Out ((PEPS) Premier Entré, Premier Sorti)
LIFO	Last In First Out ((DEPS) Dernier Entré, Premier Sorti)
CUMP	coût unitaire moyen pondéré
ICP	Indicateurs Clés de Performance
NLP	natural language processing.
IA	L'intelligence artificielle
ML	Machine Learning (Apprentissage Automatique)
DL	Deep Learning (Apprentissage Profond)
SVM	Support Vector Machine (Machine à Vecteurs de Support)
LR	Linear Regression (Régression Linéaire)
NB	Naïve Bayes
DT	Decision Tree(arbre de décisions)
GMM	Modèle de mélange de gaussien
PCA	Analyse en composante principales
RF	Random forest (forêt aléatoire)
KNN	K-Nearest Neighbors(K-Plus-Porches-Voisins)
DNN	Deep Neural Networks
AR	Auto-Régressif
MA	Moving Average(Le processus moyenne mobile)
ARIMA	Auto Regressive lntegrated Moving Average
ACF	La fonction d'autocorrélation
PACF	Fonction d'autocorrélation partielle

Introduction générale

Dans le commerce de détail, la gestion des stocks est un enjeu majeur pour garantir la disponibilité des produits tout en minimisant les coûts de stockage et les pertes. Cela implique des politiques de réapprovisionnement efficaces et l'utilisation de technologies comme les systèmes de gestion des stocks et l'analyse prédictive. Avec l'essor des technologies de l'information, les approches traditionnelles de gestion des stocks cèdent la place à des méthodes plus sophistiquées basées sur l'apprentissage automatique et l'analyse de séries temporelles afin d'optimiser leurs niveaux de stock.

L'apprentissage automatique offre des solutions puissantes pour l'optimisation des niveaux de stock dans le commerce de détail .Elle peut analyser des données historiques de ventes pour prévoir la demande future avec une grande précision, surveiller en temps réel les niveaux de stock, les ventes et les livraisons et optimiser les quantités à commander en utilisant des algorithmes avancées de prédiction, de classification et de clustering.

À l'aide des séries temporelles l'optimisation des niveaux de stock dans le commerce de détail repose sur l'identification des tendances à long terme dans les données de vente ,l'impact des promotions passées sur les ventes et à prévoir l'effet des futures promotion ,les variations saisonnières dans les ventes et révéler des cycles plus courts dans les ventes en utilisant des techniques et des modèles tel que les modèles ARIMA .

Dans notre étude, nous avons expérimenté plusieurs algorithmes d'apprentissage automatique et d'apprentissage profond pour la prediction de series temporelles , tels que les k plus proches voisins (KNN), la régression linéaire (LR), la forêt aléatoire (RF), XGBoost (eXtreme Gradient Boosting) et les réseaux de neurones profonds (DNN).

Les résultats obtenus à l'aide de ces diverses méthodes se sont révélés satisfaisants. Comme observé à travers l'analyse exploratoire des données, la taille du magasin et les vacances ont une relation directe avec les ventes élevées.

Problématique

L'optimisation des niveaux de stock dans le commerce de détail est le processus stratégique consistant à gérer et à ajuster les quantités de produits stockés pour atteindre un équilibre optimal entre l'offre et la demande .

Cette gestion implique l'utilisation de techniques et d'outils avancés de prévision, de planification et de gestion pour s'assurer que les bons produits sont disponibles en quantités appropriées au bon moment et au bon endroit.

Objectifs de la recherche

L'objectif principal de ce mémoire était de prédire les ventes dans une entreprise de commerce de détail en se basant sur les données historiques disponibles, et de déterminer si des variables telles que la température, le taux de chômage, les prix du carburant, etc., ont un impact sur les ventes . De plus, cette recherche visait à analyser si les ventes sont relativement plus élevées pendant les périodes de vacances . L'objectif était ainsi de permettre aux magasins de concevoir des offres promotionnelles ciblées pour stimuler les ventes et générer des revenus accrus.
Notre ensemble de données contient une multitude d'informations essentielles sur les ventes de Walmart avec 421 570 lignes et cinq colonnes.

Organisation du mémoire:

Après cette introduction générale, notre travail est structuré de la manière suivante :

Le premier chapitre :

Nous aborderons les concepts généraux dans le domaine de la gestion des stocks dans le commerce de détail,leur ressources matérielles, financières et humaines.Nous présenterons les différents niveaux de stocks et de gestion de stock ,Les méthodes de gestion des stocks dans le commerce de détail .

Le deuxième chapitre :

Nous aborderons les fondamentaux du domaine de l'apprentissage automatique (machine Learning) et de l'apprentissage en profondeur (Deep Learning) .Nous discuterons des algorithmes spécifiques utilisées dans chaque domaine.

Le troisième chapitre :

Nous avons examiné les bases des séries temporelles, en abordant leurs composantes fondamentales, les techniques d'analyse, les modèles et les

algorithmes de l'apprentissage automatique et de l'apprentissage en profondeur pour la prédiction.

Le quatrième Chapitre:

Dans cette partie de notre travail, nous présenterons en détail le corpus utilisé pour nos recherches, les expérimentations réalisées, ainsi que les résultats obtenus.

CHAPITRE 1

État de l'art

1.1 Introduction

La gestion des stocks est une composante essentielle des opérations commerciales, notamment dans le commerce de détail et la production, car elle influence fortement leur trésorerie et leur pérennité. L'optimisation des niveaux de stock est cependant une tâche complexe.
Dans ce chapitre, nous aborderons la gestion des stocks dans le commerce de détail , leur niveaux et leur methodes .

1.2 Approvisionnement

L'approvisionnement désigne l'ensemble des activités et des processus visant à obtenir les biens et services nécessaires au fonctionnement d'une organisation ou d'une entreprise. Cela comprend la planification, la gestion des commandes, l'achat, la réception et le stockage des ressources nécessaires pour garantir la continuité des opérations [SOF14].

1.3 Définition des stocks

Le Larousse définit le stock comme étant l'ensemble des marchandises disponibles sur un marché ou dans un magasin. Le stock, propriété de l'entreprise, est un ensemble de marchandises en attente d'utilisation, assurant la continuité de l'activité. Il garantit la disponibilité des produits nécessaires à tout moment. Les matières premières, utilisées dans la fabrication, sont considérées comme des produits consommés dès qu'ils quittent le stock. Les stocks facilitent la coordination temporaire des activités d'achat et de vente de l'entreprise [MAR13].
Par exemple : On ne peut imaginer un constructeur d'automobile qui ne produit des voitures qu'à la demande et qui commande un à un les pneus nécessaires pour satisfaire la demande. D'un autre côté, un stock étant une ressource qui reste inutilisée pendant un certain temps, constitue un gaspillage, telle est la position des fondateurs de l'approche juste- à-temps qui visent le zéro stock.

Figure 1.1 : Les stocks, une régulation de flux [MAR13].

1.3.1 Les coûts liés au stock

Lorsque les entreprises acquièrent des stocks, elles bloquent des capitaux. Voici une liste des coûts associés aux stocks :

Les locaux

Les locaux désignent l'espace dédié au stockage des marchandises, nécessitant souvent des dépenses supplémentaires telles que l'éclairage, le chauffage, l'assurance et l'entretien des installations [Red20].

Le personnel

Le personnel englobe les individus impliqués dans la gestion quotidienne du magasin, comprenant les magasiniers, le personnel administratif et les gardiens. Les coûts associés varient en fonction de la taille et de la nature des marchandises stockées [Red20].

Le matériel

L'équipement comprend des éléments spécifiques adaptés aux besoins des produits, tels que des rayonnages et des chariots élévateurs, qui sont essentiels pour la manipulation et le stockage des marchandises [Red20].

La détérioration La détérioration désigne le risque potentiel d'endommager ou de détruire des articles fragiles lors de leur manipulation, tels que le verre ou les appareils de mesure. Elle peut également se produire pour les systèmes inutilisés en raison de phénomènes tels que la corrosion ou la déformation [Red20].

La péremption

La péremption concerne les articles qui deviennent périmés ou inutilisables en raison de modifications des normes ou du dépassement de la date d'utilisation, notamment dans le domaine alimentaire ou pharmaceutique [Red20].

1.3.2 Les niveaux de stock

Comprendre et maîtriser les divers niveaux de stock est fondamental pour toute entreprise cherchant à améliorer sa performance opérationnelle, à répondre efficacement aux besoins de ses clients et à maximiser sa rentabilité [Mec23].

Pour déterminer le niveau de stock optimal, il est indispensable de connaître les valeurs suivantes :

Stock minimum

Le stock minimum est la quantité nécessaire pour répondre à la demande des clients, évitant les ruptures de stock. Il est calculé en fonction des prévisions de ventes, des coûts, des quantités de commande, et des délais de livraison des fournisseurs [Mar22].

Stock maximum

Le stock maximum est défini comme la quantité maximale de marchandises qu'une entreprise peut stocker pour répondre à la demande de ses clients sans surstockage. Il est calculé en prenant en compte des facteurs tels que le modèle de consommation, les coûts, la nature des produits, et la saisonnalité [Roq22]
Remarque : Il est important de distinguer le niveau de stock optimal du stock moyen. Le niveau de stock optimal représente la quantité idéale de marchandises à maintenir pour répondre efficacement à la demande, tandis que le stock moyen correspond à la quantité moyenne de marchandises entreposées dans l'entrepôt sur une période déterminée [Mec23].

Le stock de sécurité

Le stock de sécurité également appelé stock tampon,stock couverture,il représente un niveau de stock nécessaire pour pallier aux aléas de la logistique et des problèmes d'approvisionnement. Il est calculé en fonction de différents critères, tels que l'écart type de la demande, le coefficient de sécurité, et la racine carrée du délai de réapprovisionnement [Mon20].

Le stock d'alerte

Il s'agit du point de commande, un niveau de stock prédéfini au-dessus du stock minimum, qui indique le moment où une nouvelle commande doit être passée. Une fois ce seuil atteint, il signale qu'une rupture de stock est imminente et déclenche automatiquement le processus de commande [MAH22].

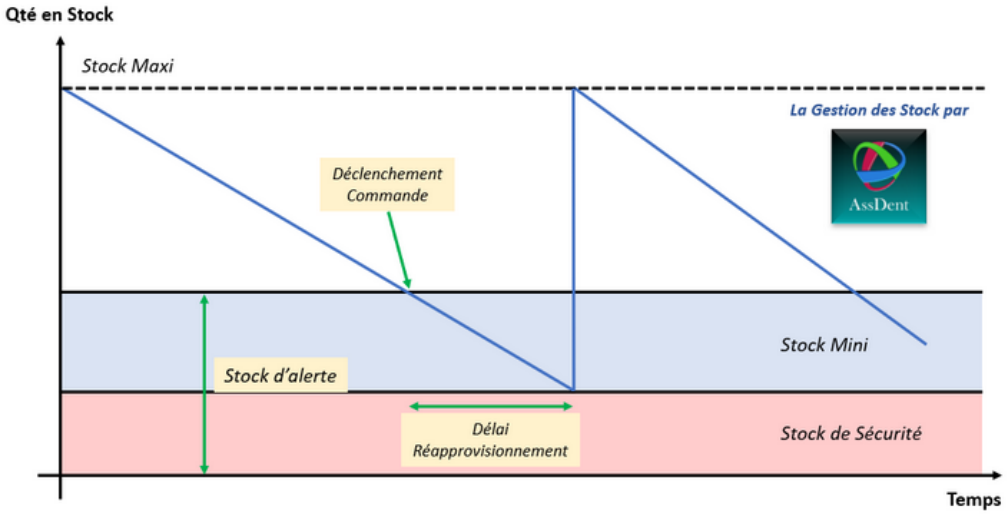

Figure 1.2 : représentation graphique du stock minimum ,maximum , de sécurité et d'alerté [Ass21].

Le stock moyen

Le stock moyen représente le niveau optimal de stock nécessaire pour couvrir les besoins habituels d'une entreprise, en fonction de son plan de charge. En d'autres termes, il permet de maintenir l'activité sans interruptions tout en assurant un approvisionnement régulier en fournitures de production ou de consommation, et ce au niveau le plus bas possible [MAH22].

Le stock-outil

Le stock-outil représente le niveau théorique du stock moyen, calculé et simulé par le gestionnaire de stocks en fonction des paramètres de gestion. En d'autres termes, il s'agit du niveau que le stock moyen devrait atteindre dans des conditions normales, sans perturbations du système [MAH22].
Stocks outil = (quantité de commandes / 2) + stock de sécurité.

Le stock mort

Le stock mort ou stock dormant désigne des produits qui ne se vendent pas ou très lentement. Cela peut résulter d'une mauvaise anticipation des envies des consommateurs, de ralentissements des ventes dus à des événements majeurs ou de changements de tendances. Les entreprises doivent limiter leurs stocks morts car ils entraînent des coûts significatifs en termes d'entretien, d'espace de stockage et d'immobilisation de capitaux. [Mon22].
La quantité en début de période = la quantité en fin de période.

Le stock disponible

Le stock disponible désigne la quantité de produit disponible en stock à un moment donné. Pour le déterminer, il est nécessaire de consulter les fiches de stock, qui fournissent une vue détaillée des articles présents dans l'inventaire [MAH22].

Le stock en commande

Le stock en commande correspond à la quantité de chaque produit qui a été commandée mais qui n'a pas encore été réceptionnée. Pour obtenir cette information, il est nécessaire de consulter les bons de commande passés auprès des fournisseurs [MAH22].

Le stock physique

Le stock physique correspond à la quantité réellement détenue par l'entreprise, c'est-à-dire la quantité de produits physiquement présente dans ses entrepôts ou ses locaux de stockage [MAH22].

1.3.3 Les mouvements des stocks

La surveillance minutieuse des mouvements de stocks est indispensable pour éviter les excès et les ruptures.Aussi l'utilisation d'outils tels que la fiche de stock et le cadencier, qui consignent les informations de manière exhaustive, de la commande chez le fournisseur jusqu'à la vente au client. Bien que ce suivi puisse être effectué manuellement, l'efficacité accrue offerte par les systèmes informatiques est préférée. L'objectif est d'assurer une gestion des stocks efficace, minimisant les immobilisations de fonds et les pertes liées à l'obsolescence, tout en préservant la satisfaction client et l'image de l'entreprise. Pour utiliser ces documents de manière appropriée, il est nécessaire de comprendre la signification de certains termes :

- **Stock initial (SI):** représente la quantité de produits disponibles au début d'une période.
- **Stock final (SF):** correspond à la quantité de produits restants à la fin de la période.
- **Tenue des stocks:** La tenue des stocks consiste à avoir une connaissance en temps réel de la quantité de produits physiquement présents, affectés ou disponibles.
- **Pilotage des stocks:** Le pilotage des stocks implique la définition d'une politique de gestion des stocks, l'évaluation de ses impacts financiers et la gestion des niveaux de risque, ainsi que l'établissement d'une politique d'approvisionnement.

- **L'optimisation des stocks:** L'optimisation des stocks consiste à trouver le compromis idéal entre le coût de stockage le plus bas possible et un taux de service maximal. Cela implique de déterminer le niveau de stock qui correspond à la fois au coût fixé et au niveau de service souhaité [YM15].

1.4 Définition de la gestion des stocks

La gestion des stocks est l'ensemble des procédures appliquées par une entreprise pour déterminer quand et combien de quantités à acheter, afin de répondre à la demande des clients tout en minimisant les coûts de stockage et en maximisant l'efficacité de la chaîne d'approvisionnement [DMl22].

1.5 La relation de la gestion des stocks avec les autres fonctions de l'entreprise

La gestion des stocks joue un rôle crucial dans la survie d'une entreprise, et cela se manifeste notamment par ses interactions avec les autres fonctions de l'entreprise.

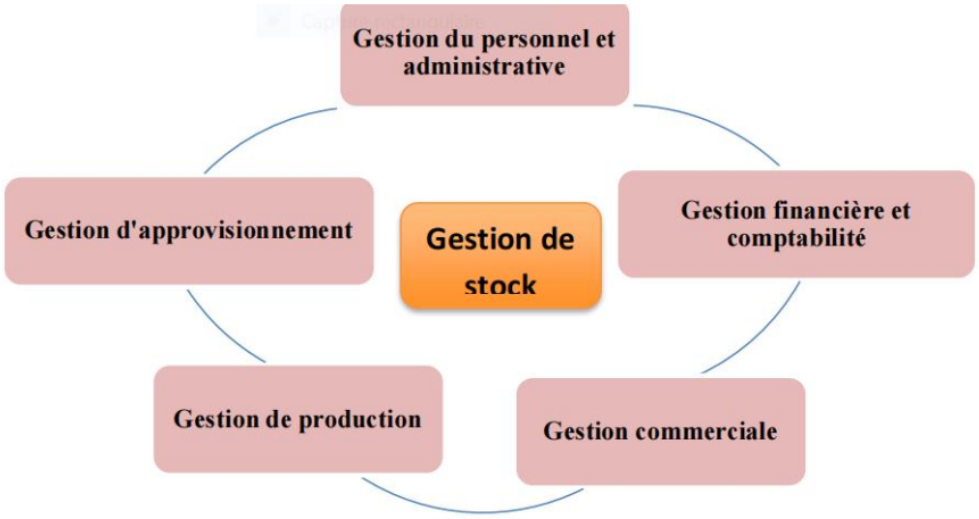

Figure 1.3 : Gestion du stock et au cœur des autres fonctions de l'entreprise [Bet21].

- **La relation de la gestion des stocks avec la gestion du personnel et administrative :** La gestion des stocks nécessite une bonne organisation et une planification efficace. Cela implique une coordination avec le personnel administratif pour gérer les stocks, les commandes et les livraisons. Une bonne gestion des stocks peut aider à réduire les coûts et à améliorer la satisfaction des clients, ce qui à son tour peut influencer positivement sur la gestion du personnel et administrative[jun22].

- **La relation de la gestion des stocks avec la fonction approvisionnement :**La fonction approvisionnement est chargée d'organiser l'exécution des demandes émises par la gestion des stocks. Elle assure également le suivi et la relance des fournisseurs afin d'éviter les retards et de garantir le respect des délais de livraison [MAH22].
- **La relation de la gestion des stocks avec la fonction achat:** La fonction achat est responsable de la sélection des fournisseurs en tenant compte du rapport qualité/prix. Elle gère également le règlement des factures, les compensations, ainsi que le suivi des fournisseurs. En outre, la fonction achat communique à la gestion des stocks toutes les modifications relatives au marché des fournisseurs, telles que les changements de délai de livraison ou les offres promotionnelles [MAH22].

1.6 Définition de la gestion des stocks du commerce de détail

La gestion des stocks dans le commerce de détail est un processus continu impliquant la planification, l'organisation et l'exécution du flux de marchandises, de la production aux points de vente via les entrepôts. Elle exige une surveillance constante des tendances et des prévisions de la demande, ainsi qu'une attention particulière aux produits populaires et à leurs lieux de vente. Les niveaux de stock doivent être ajustés en fonction de facteurs tels que les saisons, les promotions, le marketing et les tendances de l'industrie[VAL15].

1.7 L'importance de la gestion des stocks du commerce de détail

Les gestionnaires de ventes au détail comprennent les conséquences d'avoir un stock insuffisant en magasin, notamment les délais d'attente pour le réapprovisionnement et les risques de détérioration ou de perte des produits. C'est pourquoi il est crucial d'adopter des processus efficaces de gestion des stocks de vente au détail pour maintenir des opérations fluides. Pour les détaillants traitant des produits périssables, des précautions supplémentaires sont nécessaires. Par exemple, les épiceries risquent des amendes et des problèmes juridiques s'ils vendent des produits périmés, ce qui peut également ternir l'image et la réputation de l'entreprise. Pour surveiller la qualité et la quantité des produits périssables, les détaillants peuvent utiliser des accessoires tels que les capteurs SafetyCulture. Ces capteurs fournissent des données en temps réel sur des facteurs tels que la température, l'humidité et la qualité de l'air dans les zones de stockage des marchandises de détail [Saf24].

1.8 Différence entre la gestion des stocks de vente au détail et les autres types de gestion des stocks

La gestion des stocks de vente au détail se distingue des autres systèmes de gestion des stocks par le fait qu'elle se concentre uniquement sur les stocks de produits destinés à la vente directe aux clients, sans inclure les stocks en gros ni d'autres aspects de la chaîne d'approvisionnement.
En pratique, la gestion des stocks de vente au détail commence par un inventaire physique des produits disponibles. Ces stocks peuvent être entreposés dans des entrepôts ou directement dans les magasins. Les entreprises de plus grande taille ont souvent recours aux entrepôts, mais cela peut poser des défis en termes d'exactitude des stocks et de visibilité si la communication entre les responsables des magasins de détail et ceux des entrepôts n'est pas optimale. Pour surmonter ces défis, les détaillants peuvent mettre en place des processus de gestion des stocks d'entrepôt pour assurer une gestion globale efficace des stocks de détail [Saf24].

1.9 Techniques de gestion des stocks du commerce de détail

La gestion des stocks dans le commerce de détail englobe le suivi, le stockage et le réapprovisionnement des marchandises afin de garantir que les clients puissent trouver ce qu'ils recherchent. Voici six techniques couramment utilisées par les détaillants pour gérer efficacement leurs inventaires :

1.9.1 L'évaluation des stocks

L'évaluation des stocks consiste à attribuer une valeur monétaire aux stocks d'un détaillant. Cette valeur est déterminée en prenant en compte le coût d'acquisition des articles en stock, y compris les remises anticipées, les taxes et les frais de stockage. Cette évaluation est cruciale pour la comptabilité fiscale et la production de rapports financiers [Hic23].

1.9.2 Le rapprochement régulier

Le rapprochement régulier est une pratique courante chez les détaillants pour comparer les inventaires physiques enregistrés dans différents systèmes avec les inventaires physiques réels. Cette procédure garantit que les enregistrements de stocks sont précis et mis à jour [Hic23].

1.9.3 Le comptage des stocks physiques

Le comptage physique des stocks implique de dénombrer tous les articles stockés dans les entrepôts, les magasins et autres emplacements, puis

de comparer ces décomptes aux enregistrements comptables. Cette opération est chronophage et nécessite une main-d'œuvre importante, ce qui perturbe souvent les opérations. C'est pourquoi elle est généralement réalisée annuellement avant le début de l'exercice financier suivant, afin de régulariser les écarts entre les enregistrements [Hic23].

1.9.4 Les comptages périodiques

Les comptages périodiques consistent à réaliser des dénombrements d'échantillons ou de sections d'inventaire tout au long de l'année afin de garantir une comptabilité continue. Cette méthode implique souvent le comptage quotidien d'un petit pourcentage des stocks à différents moments, ce qui est moins perturbant que les comptages physiques globaux. Les comptages périodiques offrent l'avantage de détecter les problèmes au fil du temps plutôt que de les affronter tous simultanément, ce qui permet de résoudre les anomalies de manière plus efficace [Hic23].

1.9.5 L'analyse ABC

L'analyse ABC est une méthode de classification des produits en stock, de A à C, en fonction de leur importance financière pour l'entreprise. Les articles de la classe A sont considérés comme les plus précieux en termes de ventes, de risques, de demande et de coûts. Cette méthode aide les détaillants à déterminer quels articles doivent être priorisés pour le comptage périodique en fonction de leur valeur et de leur impact sur l'activité [Hic23].

1.9.6 Les contrôles ponctuels

Les contrôles ponctuels consistent à effectuer un comptage à petite échelle en sélectionnant au hasard des articles en stock et en les comparant aux enregistrements du système comptable. Cette méthode permet de détecter les écarts dans les enregistrements de stocks en identifiant les erreurs ou les divergences entre les quantités physiques et les données enregistrées [Hic23].

1.10 Analyse et prévision des stocks pour les détaillants

L'examen et la prédiction des stocks jouent un rôle important dans les opérations du commerce de détail. Ils permettent aux détaillants de surveiller leurs niveaux de stocks et d'anticiper la demande pour leurs produits. L'objectif principal est de garantir que les stocks sont suffisants pour satisfaire la demande des clients, tout en évitant les ruptures de stock et les coûts superflus associés à un excès de stockage. Gardez à l'esprit ces approches pour une gestion efficace des stocks [Hic23].

1.10.1 La prévision qualitative

Cette approche combine le jugement humain subjectif et des données tangibles pour anticiper la demande future à court terme. Les prévisions qualitatives sont précieuses lors de changements majeurs, comme une récession ou des perturbations de la chaîne d'approvisionnement, où l'analyse de données seule peut être insuffisante. Elles sont également utiles pour des événements imprévus, tels que la participation d'une célébrité à l'ouverture d'un magasin. Les détaillants peuvent enrichir leurs prévisions en consultant des experts externes, des groupes de clients et d'autres études de marché [Hic23].

1.10.2 L'analyse de séries chronologiques

L'analyse des séries chronologiques utilise les données de ventes passées pour repérer des schémas et des tendances. Elle emploie des techniques comme les moyennes mobiles, le lissage exponentiel et les modèles ARIMA. Basée sur des données quantitatives sur une période définie, elle permet de prédire les tendances futures. Cette méthode prend en compte la fréquence de vente de chaque détaillant et l'évolution d'autres variables au fil du temps. Par exemple, des variations dans des modèles saisonniers bien établis, comme un intervalle plus court entre Thanksgiving et Noël aux États-Unis, peuvent influencer le volume des ventes et les besoins en stock [Fas24].

1.10.3 L'apprentissage automatique

L'apprentissage automatique,à travers des algorithmes comme les forêts aléatoires, les réseaux de neurones et l'augmentation des gradients, permet d'affiner la précision des prévisions en assimilant des modèles complexes. Par exemple, un détaillant de vêtements peut exploiter une analyse de séries chronologiques pour anticiper la demande de manteaux d'hiver en se basant sur les ventes des années précédentes durant les mois les plus froids [Fas24].

1.11 Conclusion

Ce chapitre met en lumière l'importance des stocks pour toutes les entreprises, quel que soit leur domaine d'activité. Leur gestion, impliquant des ressources matérielles, financières et humaines, est essentielle pour garantir le bon déroulement des opérations commerciales. Une coordination efficace avec les autres fonctions de l'entreprise est nécessaire pour atteindre les objectifs fixés par la direction. En résumé, les stocks jouent un rôle vital dans la réalisation des objectifs de l'entreprise, nécessitant la mise en place d'un système de gestion efficace pour réguler les flux et garantir une distribution optimale des produits.

CHAPITRE 2

L'intelligence artificielle et ses applications

2.1 Introduction

L'intelligence artificielle(IA) vise à prendre des décisions autonomes et adaptatives dans la gestion de stock. Grâce à la combinaison de l'apprentissage automatique (AA)et d'autres algorithmes d'IA, les entreprises peuvent anticiper et prédire avec précision la demande future, optimiser les niveaux de stock, réduire les coûts, et même analyser les données historiques. Dans ce chapitre, nous aborderons l'intelligence artificielle dans ces domaines, les différents types d'AA, et les algorithmes associés.

2.2 Définition de l'intelligence artificielle

L'intelligence artificielle (IA) est une branche de l'informatique et des mathématiques qui regroupe un ensemble de techniques algorithmiques et de théories visant à créer des machines capables de simuler l'intelligence humaine. Son objectif est de reproduire cette capacité afin de résoudre des problèmes complexes. Pour ce faire, l'IA s'attache à modéliser l'intelligence humaine comme on le ferait dans des disciplines telles que la physique, la chimie ou la biologie. Ce domaine est en plein essor et trouve sa théorie et ses applications dans de nombreux domaines [DSSQ22] .

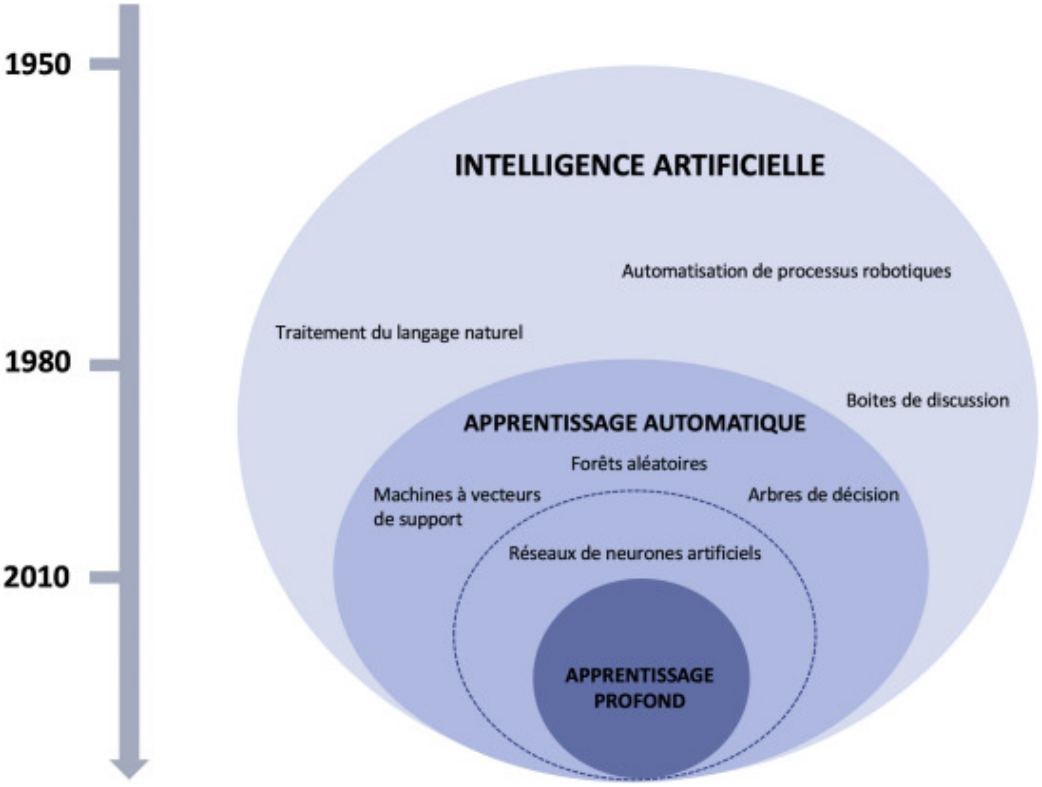

Figure 2.1 : Schéma de décomposition du domaine de l'intelligence artificielle et de ces sous- domaines [RB22] .

2.2.1 Les disciplines qui influencent l'IA

L'intelligence artificielle (IA) est un domaine interdisciplinaire qui s'appuie sur et contribue à plusieurs disciplines. Voici quelques-unes des principales disciplines qui influencent l'IA :

- Informatique et génie
- Mathématiques .
- Neurosciences .
- Psychologie cognitive .
- Économie et théorie de la décision .
- Linguistique .
- Philosophie .
- Éthique et droit .

Ces disciplines jouent un rôle crucial dans le développement, la compréhension et l'application de l'IA, contribuant à façonner son évolution et son impact sur notre monde.

2.2.2 Les domaines de recherche en IA

Les domaines de recherche en intelligence artificielle (IA) sont vastes et en constante évolution, englobant une multitude de sujets et de disciplines. Voici quelques-uns des principaux domaines de recherche en IA :

Apprentissage-machine

L'apprentissage automatique, souvent désigné par son terme anglais Machine Learning (ML), représente une sous-discipline de l'intelligence artificielle (IA) qui trouve application dans la résolution de problèmes de prédiction. En substance, l'apprentissage automatique implique la construction de modèles de prédiction, qu'il s'agisse de classification ou de prévision, en se basant sur un ensemble de données d'apprentissage composé d'exemples d'entrée accompagnés des sorties désirées correspondantes [LS20].

Reconaissance des formes

Les recherches dans ce domaine visent à automatiser la perception de situations typiques. Leurs méthodes ont de nombreuses applications, notamment dans la vision par ordinateur, la reconnaissance vocale, la lecture optique de documents et la synthèse d'images. Les avancées en reconnaissance d'images vidéo permettent déjà aux forces de l'ordre de repérer des cibles au sein de foules [Aï05].

Vie intelligence

La vie artificielle représente un domaine de recherche interdisciplinaire qui s'attache à développer des systèmes artificiels en s'inspirant des mécanismes observés dans les systèmes vivants. Ces systèmes peuvent prendre la forme de programmes informatiques ou de robots, et visent à reproduire des comportements et des caractéristiques propres aux organismes vivants [phi18].

Robotique

La robotique, un sous-domaine crucial de l'intelligence artificielle, se concentre sur la conception et la fabrication d'agents physiques capables d'interagir avec leur environnement [ft15] .

Indexation multimédia

Les ressources multimédia disponibles sur le web sont souvent nombreuses, volumineuses et parfois non pertinentes. Pour résoudre ce défi, l'intelligence artificielle propose des outils de fouille de données (Data mining) qui permettent d'extraire des connaissances synthétiques ou de découvrir des informations cachées dans les bases de données. Ces outils

peuvent être utilisés pour diagnostiquer des situations, aider à superviser la gestion de systèmes, ou encore filtrer et organiser efficacement les données multimédias [Aï05].

Le traitement naturel du langage

Le traitement naturel du langage, connu sous le nom de Natural Language Processing (NLP) en anglais, représente une technologie d'intelligence artificielle qui cherche à habiliter les ordinateurs à comprendre le langage humain [phi18].

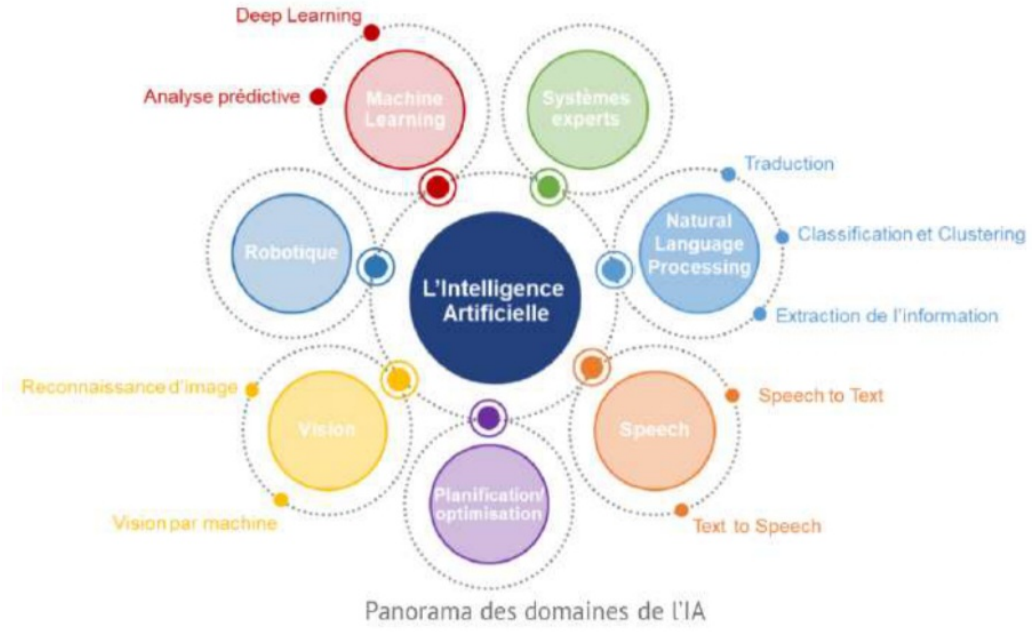

Figure 2.2 : Panorama des domaines de l'IA [**?**].

2.2.3 Applications de l'IA

Voici une liste d'applications de l'intelligence artificielle :

1. Diagnostic médical (thérapie, surveillance d'appareils).
2. Synthèse d'images (vision par ordinateur).
3. Classification(naturelle, biologie, minéralogie, etc.).
4. Planification de tâches (prédictions financières, etc.).
5. Architecture (conception assistée par ordinateur).
6. Détection de pannes (comme Sherlock pour les avions F16).
7. Éducation (systèmes tutoriels intelligents, e-Learning).
8. Génie (vérification des règles de conception).

9. Prospection géologique (gisements miniers).
10. Centrales nucléaires, feux de forêt (systèmes en temps réel).
11. Simulateurs de vols (CAE, Bombardier, etc.).
12. Jeux vidéos .

Ces applications démontrent la polyvalence de l'intelligence artificielle et son impact dans de nombreux domaines, allant de la médecine à l'industrie du divertissement en passant par l'ingénierie et la géologie [Aï05].

2.3 Les avantages et les risques de l'intelligence artificielle

l'intelligence artificielle (IA) est effectivement devenue un facteur clé dans de nombreux développements, mais comme toute technologie, elle comporte à la fois des avantages et des inconvénients.

2.3.1 Les avantages

Les principaux avantages de l'intelligence artificielle sont :

1. Rendre la vie plus facile .
2. Sauver des vies dans le domaine de la santé .
3. Digitaliser le monde et faciliter la communication entre les machines .
4. Rendre le monde très petit grâce aux réseaux sociaux .
5. Minimiser les coûts de main-d'œuvre dans les entreprises .

Ces avantages soulignent l'impact positif significatif que l'intelligence artificielle peut avoir sur nos vies et sur le fonctionnement des entreprises et de la société dans son ensemble [Nes22].

2.3.2 Les inconvénients

Les inconvénients importants de l'intelligence artificielle sont :

1. Risque de sécurité .
2. Risque de destruction de l'humanité .
3. Automatisation des armes .
4. Violation de la vie privée .

Ces inconvénients soulignent la nécessité de réglementer et de surveiller attentivement le développement et l'utilisation de l'intelligence artificielle afin de garantir qu'elle soit bénéfique pour la société dans son ensemble tout en minimisant les risques et les effets indésirables [Nes22].

2.4 Concepts et Sources de l'apprentissage automatique

L'apprentissage humain est un processus complexe et diversifié, difficile à décrire précisément. Les facultés d'apprentissage de l'homme ont joué un rôle vital dans son développement évolutif. Ces facultés incluent l'acquisition de compétences telles que la parole par observation, l'apprentissage de la lecture, de l'écriture, ainsi que des opérations arithmétiques et logiques avec l'aide d'un tuteur, et le développement d'habiletés motrices et sportives par la pratique [Nad13].

2.5 L'apprentissage automatique

L'apprentissage automatique, également connu sous le nom de machine learning, apprentissage artificiel ou apprentissage statistique, est un domaine de l'intelligence artificielle qui repose sur des approches mathématiques et statistiques pour permettre aux ordinateurs d'apprendre à partir de données [Rou21].
Il englobe le développement, l'analyse et la mise en œuvre de méthodes qui permettent à une machine d'évoluer à travers un processus d'apprentissage, lui permettant ainsi d'accomplir des tâches difficiles ou impossibles à réaliser par des moyens algorithmiques plus traditionnels [Ben10].

2.5.1 Modélisation

L'automatisation d'une machine implique toujours un ensemble de tâches concrètes, désignées comme T. Pour évaluer la performance de la machine dans l'exécution de ces tâches, on utilise une mesure de performance, notée P. La machine peut disposer à l'avance d'un ensemble d'expériences, noté E, ou bien elle peut enrichir cet ensemble au fil du temps [Nad13].

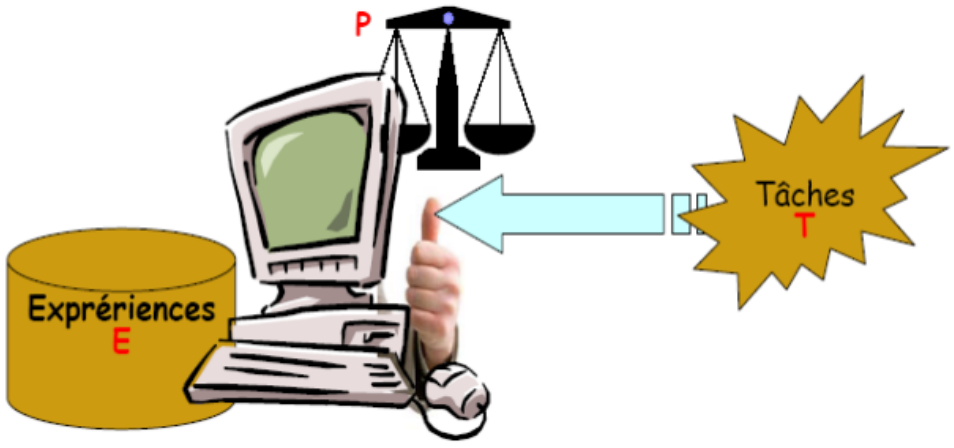

Figure 2.3 : Schéma de modélisation d'une machine d'apprentissage

Tout à fait, dans le contexte de l'apprentissage automatique, la machine utilise l'ensemble d'expériences E pour améliorer sa performance dans l'exécution des tâches T. En d'autres termes, elle apprend à partir de ces expériences afin de mieux accomplir les tâches qui lui sont assignées. Ce processus d'apprentissage peut se dérouler de manière supervisée, non supervisée ou semi-supervisée, en fonction des données disponibles et du type de tâches à effectuer. L'objectif est d'optimiser la performance de la machine sur les tâches données [Nad13].

2.5.2 Les phases de l'apprentissage automatique

les algorithmes d'apprentissage automatique suivent généralement deux phases, qui peuvent varier selon le type d'apprentissage et les données disponibles. Voici des phases couramment rencontrées:

Phase d'entraînement (ou d'apprentissage)

La phase d'entraînement dans l'apprentissage automatique est fondamentale. Le modèle, exposé à des données d'entraînement, apprend des règles implicites pour mieux comprendre le problème à résoudre. Cette étape permet au modèle d'ajuster ses paramètres pour minimiser les erreurs. Une fois l'entraînement terminé, le modèle est prêt pour la prédiction ou la classification. Certains systèmes peuvent même continuer à apprendre indéfiniment grâce à l'apprentissage en ligne, s'adaptant aux nouvelles données et aux changements dans l'environnement [DSSQ22].

Phase d'inférence

Après la phase d'entraînement, vient celle de l'inférence où le modèle entraîné est utilisé sur de nouvelles données. Le modèle peut ainsi effectuer des prédictions ou des classifications sur des données non vues auparavant. Cette capacité de généralisation du modèle lui permet d'appliquer les connaissances acquises à de nouvelles situations, assurant ainsi des résultats précis même sur des entrées inconnues. La phase d'inférence est importante dans de nombreuses applications d'apprentissage automatique, où le modèle doit fonctionner en temps réel et traiter de grandes quantités de données de manière efficace [DSSQ22].

2.5.3 Les étapes de l'apprentissage automatique

Pour développer et gérer un modèle d'apprentissage automatique adapté à une mise en production, il est généralement nécessaire de suivre les étapes suivantes :

Collecter des données

La collecte de données est la première étape essentielle le développement d'un modèle d'apprentissage automatique. La qualité et la quantité

des données collectées déterminent la qualité du modèle final. Les données peuvent être collectées à partir de diverses sources telles que des fichiers, des bases de données ou des capteurs. Cependant, les données collectées nécessitent souvent une préparation, car elles peuvent contenir des champs manquants, des valeurs aberrantes ou être non structurées. Ainsi, la préparation des données est essentielle pour rendre les données utilisables par le modèle [LS20].

Prétraitement des données

Le prétraitement des données est essentiel pour nettoyer les données brutes et les rendre utilisables par les modèles d'apprentissage automatique[LS20]. Voici quelques techniques de prétraitement de base :

- **Conversion des données :** Les données catégorielles et ordinales sont converties en données numériques, car les modèles d'apprentissage automatique ne peuvent traiter que des fonctionnalités numériques.
- **Ignorer les valeurs manquantes :** Les lignes ou colonnes contenant des données manquantes peuvent être supprimées, bien que cette méthode ne doive pas être utilisée de manière excessive.
- **Remplissage des valeurs manquantes:** Les données manquantes peuvent être remplacées par la valeur moyenne, médiane ou la plus fréquente de la variable concernée.
- **Apprentissage automatique:** Les données manquantes peuvent être prédites en utilisant des techniques d'apprentissage automatique, permettant de prédire les valeurs manquantes en se basant sur les données existantes.
- **Détection des valeurs aberrantes:** Les valeurs aberrantes, qui s'écartent considérablement des autres observations, peuvent être identifiées et supprimées ou remplacées. Par exemple, un poids humain de 800 kg en raison d'une faute de frappe doit être corrigé .

Ces techniques de prétraitement des données contribuent à améliorer la qualité des données et à garantir que le modèle d'apprentissage automatique peut obtenir des résultats précis et fiables [LS20].

Recherche du modèle d'apprentissage

La recherche du modèle d'apprentissage automatique consiste à sélectionner un algorithme ou un modèle qui est capable de prédire une sortie pour une entrée donnée. Il existe une variété d'algorithmes disponibles, chacun étant adapté à un type spécifique d'apprentissage [LS20].

Formation et test du modèle

Pour former un modèle, les données sont divisées en trois sections : l'ensemble d'apprentissage (Training Set), l'ensemble de validation (Validation Set) et l'ensemble de test (Testing Set). Le classificateur est entraîné avec l'ensemble d'apprentissage, les paramètres sont ajustés avec l'ensemble de validation, puis les performances sont évaluées avec l'ensemble de test. L'ensemble de test ne doit pas être utilisé pendant l'entraînement du classificateur, étant réservé uniquement pour le test final [LS20].

L'évaluation

L'évaluation est une étape essentielle du processus de développement du modèle. Elle permet de déterminer quel modèle représente le mieux les données et à quel point ce modèle sélectionné fonctionnera efficacement à l'avenir [LS20].

2.5.4 Types d'apprentissage automatique

Les algorithmes d'apprentissage peuvent se catégoriser selon le type d'apprentissage qu'ils emploient :

- Apprentissage supervisé
- Apprentissage non-supervisé

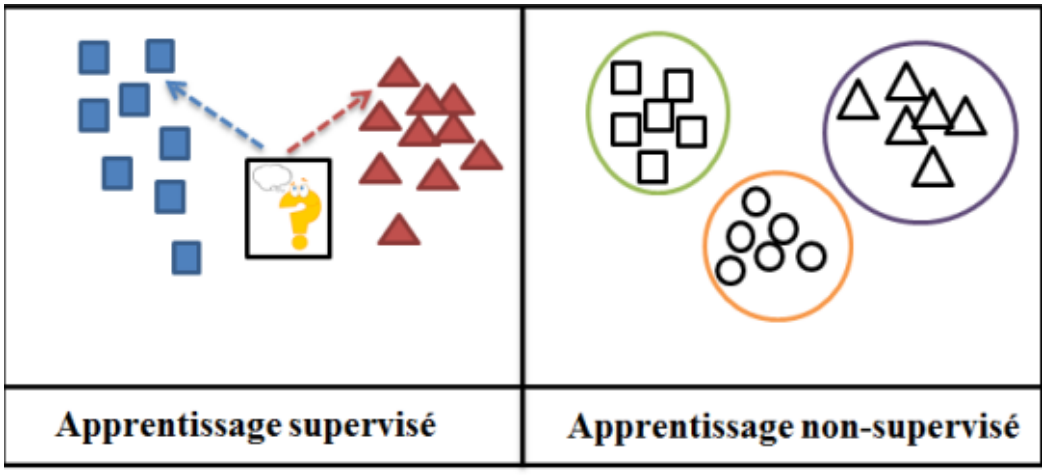

Figure 2.4 : apprentissage supervisé & apprentissage non-supervisé [EN21].

- Apprentissage semi-supervisé
- L'apprentissage partiellement supervisé (probabiliste ou non)
- Apprentissage par renforcement

2.5.5 Les algorithmes utilisés

Les méthodes algorithmiques utilisées incluent :

- Machines à vecteurs de support
- Boosting
- Réseaux de neurones pour un apprentissage supervisé ou non supervisé
- Méthode des k plus proches voisins pour un apprentissage supervisé
- Arbres de décision
- Régression logistique

Ces méthodes sont souvent combinées pour créer différentes variantes d'apprentissage. Le choix d'un algorithme dépend largement de la nature de la tâche à résoudre, que ce soit la classification, l'estimation de valeurs, ou autre [Nad13].

2.5.6 Les applications de l'apprentissage automatique

Les cas d'utilisation de l'apprentissage automatique sont variés et démontrent son utilité dans de nombreux domaines :

- La fouille de données (data mining) .
- La robotique .
- La transcription automatique de la parole (Chatbot) .
- La reconnaissance d'objets ou de personnes dans des images, ainsi que la reconnaissance du langage parlé [LS20].

2.6 Apprentissage automatiques supervisé

L'apprentissage automatique supervisé est une technique de machine learning où un modèle est formé sur un ensemble de données d'entraînement étiquetées. Les étiquettes sont des sorties correctes que le modèle doit apprendre à prédire à partir des entrées correspondantes [LL20].

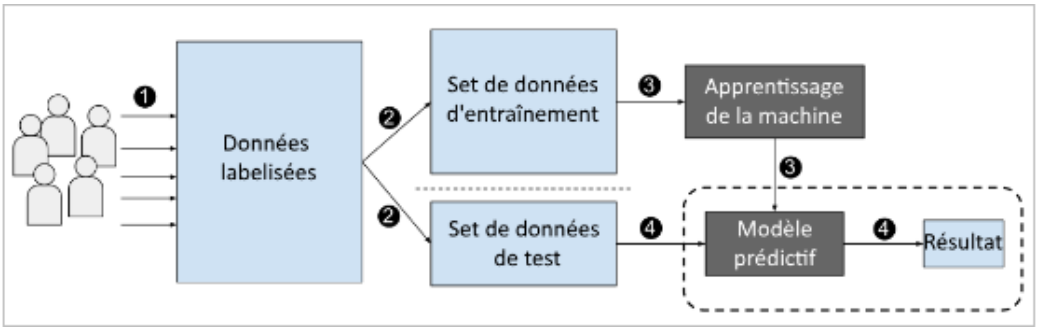

Figure 2.5 : Schéma du fonctionnement de l'apprentissage supervisé [Mar20].

2.6.1 Les différents types d'apprentissage automatiques supervisé

L'apprentissage automatique supervisé est une approche où le modèle est entraîné sur un ensemble de données étiquetées, c'est-à-dire que chaque exemple d'entraînement est associé à une étiquette ou à une sortie désirée. Les différents types d'apprentissage automatique supervisé comprennent:

2.6.2 La classification

La classification consiste à attribuer des étiquettes ou des catégories à des données en fonction de leurs caractéristiques. Les classificateurs prennent des entités en entrée et produisent une sortie correspondant à la classe de l'objet traité. Les algorithmes de classification, tels que SVM, K-NN et Naive Bayes, analysent les caractéristiques des données pour les associer à des classes prédéfinies, facilitant ainsi la classification de nouvelles données en fonction de ces caractéristiques [DSSQ22].

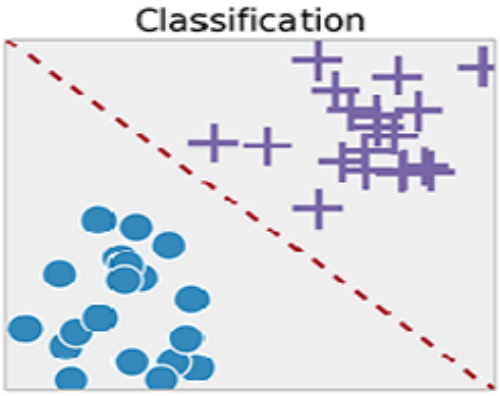

Figure 2.6 : La classification [EN21].

Les types de la classification

En classification, il existe plusieurs types de méthodes, chacune avec ses propres caractéristiques et approches. Voici quelques-uns des types de

classification les plus courants:

Classification binaire

Un problème d'apprentissage supervisé où l'espace des étiquettes est binaire, c'est-à-dire $Y = \{0, 1\}$, est appelé un problème de classification binaire. Voici quelques exemples de tels problèmes :

- Identifier si un email est un spam ou non.
- Identifier si un tableau a été peint par Picasso ou non.
- Identifier si une image contient ou non une girafe.
- Identifier si une molécule peut ou non traiter la dépression.
- Identifier si une transaction financière est frauduleuse ou non [Nes22].

Dans chacun de ces cas, le modèle de classification binaire cherche à attribuer une étiquette binaire (0 ou 1) à un exemple donné en fonction de ses caractéristiques, permettant ainsi de prendre une décision binaire concernant la nature de l'exemple.

Classification multi-classe

Un problème d'apprentissage supervisé où l'espace des étiquettes est discret et fini, c'est-à-dire $Y = \{1, 2, ..., C\}$, est appelé un problème de classification multi-classe. C représente le nombre de classes disponibles. Voici quelques exemples de tels problèmes :

- Identifier en quelle langue un texte est écrit.
- Identifier lequel des 10 chiffres arabes est un chiffre manuscrit.
- Identifier l'expression d'un visage parmi une liste prédéfinie de possibilités (colère, tristesse, joie, etc.).
- Identifier à quelle espèce appartient une plante.
- Identifier les objets présents sur une photographie [Aze22].

Dans ces exemples, le modèle de classification multi-classe cherche à attribuer une étiquette correspondant à l'une des classes possibles à un exemple donné en fonction de ses caractéristiques, permettant ainsi de prendre une décision parmi plusieurs options.

2.6.3 La régression

Un problème d'apprentissage supervisé dans lequel l'espace des étiquettes est Y = R est appelé un problème de régression [Aze22].
La régression est utilisée lorsque la sortie à prédire peut prendre des valeurs continues, c'est-à-dire qu'il s'agit d'une variable réelle. Par exemple, un algorithme prédisant la consommation électrique d'une installation ou le cours des actions en bourse [DSSQ22].
Voici quelques exemples de problèmes de régression :

- Prédire le nombre de clics sur un lien.
- Prédire le nombre d'utilisateurs d'un service en ligne à un moment donné.
- Prédire le prix d'une action en bourse.
- Prédire l'affinité de liaison entre deux molécules.
- Prédire le rendement d'un plant de maïs.

Dans ces exemples, le modèle de régression cherche à prédire une valeur continue en fonction des caractéristiques des données d'entrée, permettant ainsi de réaliser des prédictions sur des phénomènes qui varient de manière continue.

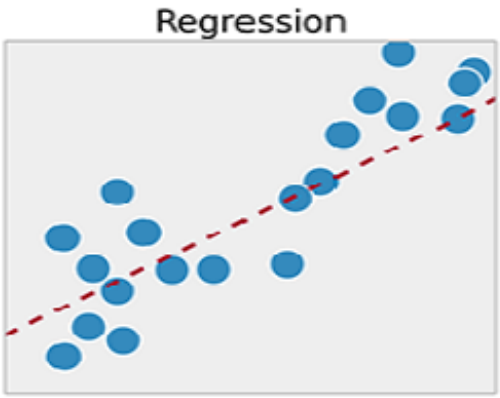

Figure 2.7 : La regression [EN21].

2.6.4 Types de régression linéaire

La régression linéaire, bien qu'elle ait un modèle fondamental, peut être utilisée de différentes manières selon le contexte et les objectifs spécifiques d'une analyse. Voici quelques types courants de régression linéaire :

- Régression linéaire simple.
- La régression linéaire multiple.
- Régression linéaire polynomiale

- Régression linéaire pondérée
- Régression linéaire régularisée
- Régression linéaire robuste

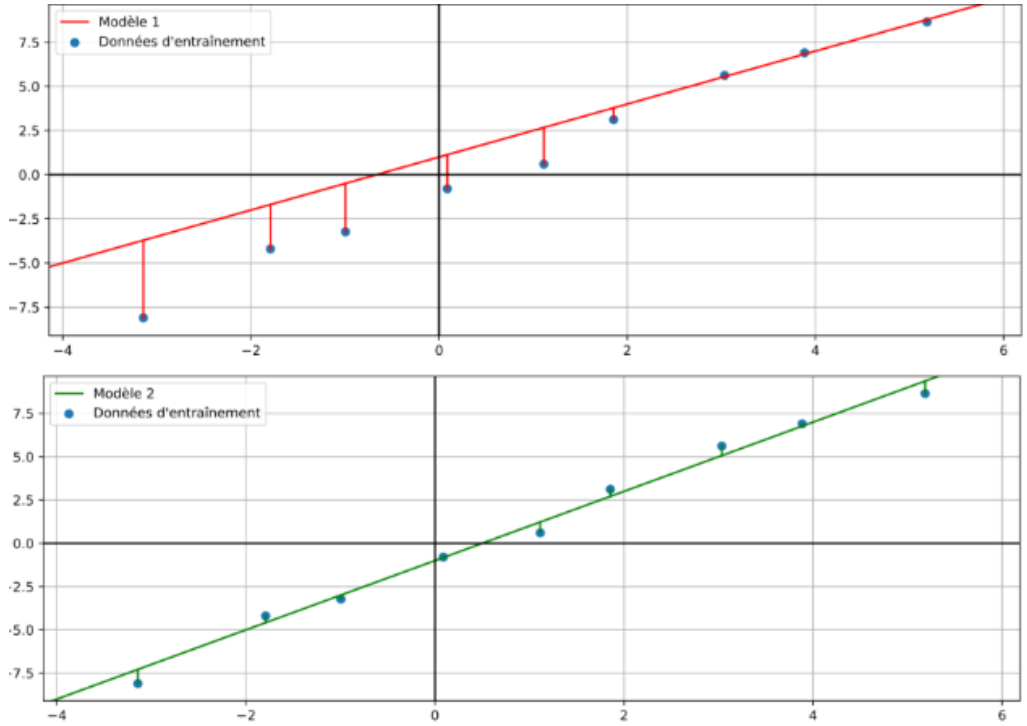

Figure 2.8 : Deux modèles de régression linéaire [DSSQ22].

Le premier modèle présente des écarts importants entre les valeurs prédites et attendues tandis que le second minimise les carrés de ces écarts .

2.7 L'apprentissage non-supervisé

L'apprentissage non-supervisé se compose principalement d'algorithmes de regroupement, qui cherchent à séparer les données d'entrée en un certain nombre de groupes. Chaque élément d'un groupe partage des caractéristiques similaires avec les autres membres du même groupe, tout en étant distinct des membres des autres groupes. Ces algorithmes regroupent ainsi les données en familles pour les catégoriser automatiquement. Par exemple, un algorithme de regroupement peut être utilisé pour regrouper des patients afin de prédire leurs réponses potentielles à certains traitements [DSSQ22].

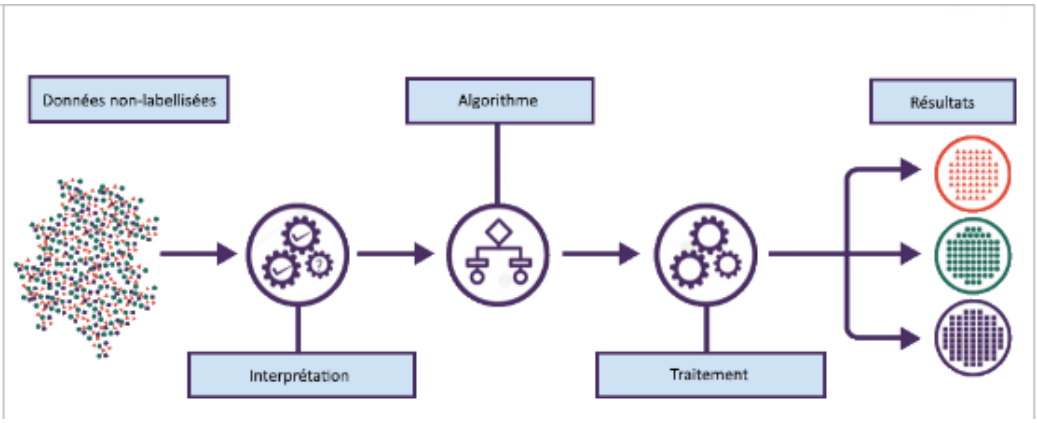

Figure 2.9 : Schéma de fonctionnement de l'apprentissage non supervisé [Mar20].

Voici quelques-uns des principaux algorithmes d'apprentissage non-supervisé :

2.7.1 Clustering

Dans le clustering, un algorithme analyse les données sans étiquettes pour détecter des groupes similaires. Par exemple, dans le contexte des visiteurs d'un blog, l'algorithme peut identifier des groupes tels que les amateurs de bandes dessinées ou les passionnés de science-fiction, en se basant sur des caractéristiques communes telles que les habitudes de consultation. Avec un algorithme hiérarchique, ces groupes peuvent être subdivisés en sous-groupes plus spécifiques, ce qui facilite le ciblage de messages adaptés à chaque segment [BEL20].
Concernant les algorithmes spécifiques de clustering :

- K-Means
- Analyse des clusters hiérarchiques (HCA)
- Maximisation des attentes

Dans le graphique suivant , nous montrons comment un ensemble de points peut être classé pour former trois sous-ensembles :

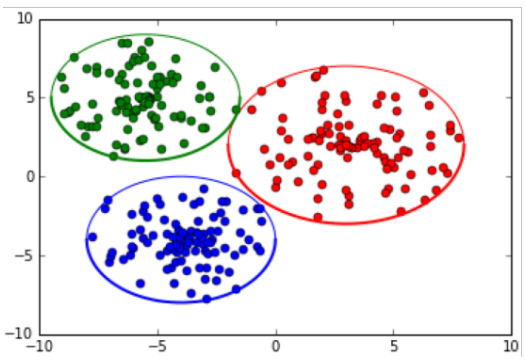

Figure 2.10 : Clustering [BEL20].

2.7.2 La réduction de la dimensionnalité

La réduction de la dimensionnalité vise à simplifier les données tout en préservant autant d'informations que possible. Une méthode pour y parvenir consiste à regrouper plusieurs caractéristiques corrélées en une seule. Par exemple, le kilométrage d'une voiture peut être fortement corrélé avec son âge, de sorte que l'algorithme de réduction de la dimensionnalité les fusionnera en une seule caractéristique représentant l'usure de la voiture. Cette technique est connue sous le nom d'extraction de caractéristiques [BEL20].

2.7.3 La détection des anomalies

Elle est un domaine fascinant qui cherche à repérer les éléments inhabituels qui s'écartent de la distribution générale des données. Identifier les valeurs aberrantes s'avère crucial dans de nombreuses applications, comme repérer les transactions suspectes sur une carte de crédit pour prévenir la fraude, détecter les défauts de fabrication ou filtrer automatiquement les valeurs aberrantes dans un ensemble de données avant de les soumettre à un autre algorithme d'apprentissage. Le système est entraîné sur des instances normales et, lorsqu'il rencontre une nouvelle instance, il peut déterminer si elle correspond à un comportement normal ou si elle est probablement une anomalie. comme le montre le graphe suivant:

Figure 2.11 : Détection d'anomalies [BEL20].

2.7.4 L'apprentissage des règles d'association

L'apprentissage des règles d'association vise à explorer de vastes ensembles de données pour découvrir des relations intéressantes entre les différents attributs. Par exemple, imaginons que nous soyons propriétaire d'un supermarché. En appliquant une règle d'association aux registres de vente, nous pourrions découvrir que les clients qui achètent de la sauce barbecue et des pommes de terre ont également tendance à acheter du

steak. Par conséquent, nous pourrions envisager de placer ces articles à proximité les uns des autres dans le magasin, car cela pourrait inciter les clients à acheter davantage de ces produits [BEL20].

2.8 L'apprentissage par renforcement

Dans le domaine de l'apprentissage par renforcement, le système d'apprentissage peut interagir avec son environnement en réalisant des actions. En retour de ces actions, il reçoit une récompense, qui peut être positive si l'action était bénéfique, ou négative si elle était préjudiciable. Il arrive parfois que la récompense ne soit obtenue qu'après une séquence d'actions étendue ; c'est notamment le cas dans l'apprentissage des jeux comme le go ou les échecs. Dans ce contexte, l'apprentissage consiste à définir une politique, c'est-à-dire une stratégie visant à obtenir la récompense la plus élevée possible de manière systématique.
Les applications principales de l'apprentissage par renforcement se retrouvent dans les jeux (comme les échecs ou le go) ainsi que dans la robotique. Ce sujet dépasse largement le cadre de cet ouvrage [Aze22].

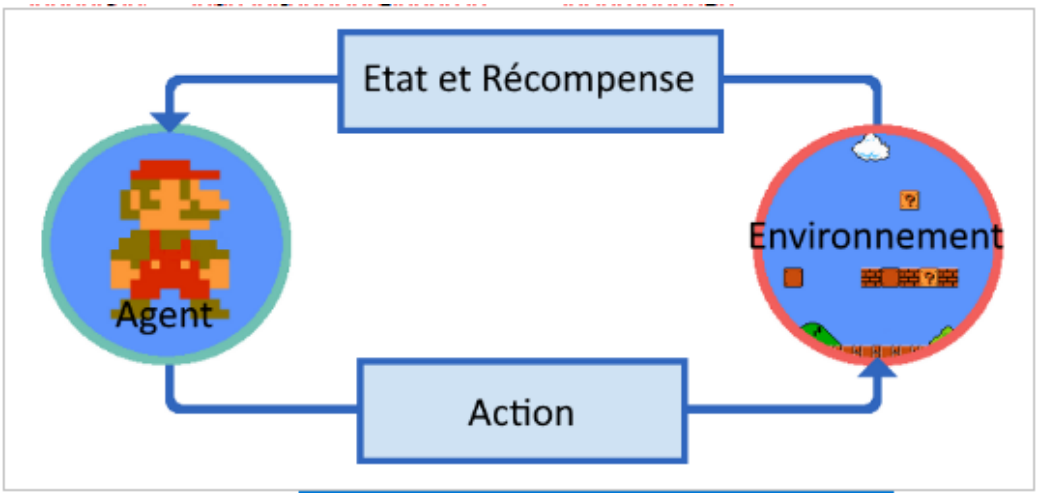

Figure 2.12 : Schéma du fonctionnement de l'apprentissage par renforcement [Mar20].

2.9 L'apprentissage par transfert

L'apprentissage par transfert est une technique de machine learning qui exploite les données et les connaissances déjà acquises pour résoudre une tâche similaire. En utilisant les informations extraites lors du traitement d'une tâche précédente, on peut aborder plus efficacement une nouvelle tâche similaire. Cette approche capitalise sur les connaissances déjà acquises pour accélérer et améliorer l'apprentissage sur une nouvelle tâche [LS20].

2.10 Récapitulatif des différents types d'apprentissage

Le tableau suivant résume les différents types d'apprentissage automatique:

	Supervisé	Non-supervisé	Renforcement
Utilise quelles données ?	Données labellisées	Données non-laballisées	Expériences de l'agent
Comment il fonctionne	Apprentissage et tests	Calcul de similarité /dissimilarité	Récompense /punition
Donne quels résultats ?	Labellisation	Affectation à un cluster	Optimisation

Table 2.1 : Récapitulatif des différents types d'apprentissage.

2.11 Les modèles de l'apprentissage automatique

Les modèles de l'apprentissage automatique sont des structures mathématiques ou statistiques qui sont utilisées pour représenter et résoudre des problèmes d'apprentissage automatique. Ces modèles peuvent être de différentes natures, selon le type de tâche à accomplir et les données disponibles. Voici quelques types de modèles couramment utilisés en apprentissage automatique :

2.11.1 Algorithmes des K plus proches voisins

L'algorithme des K plus proches voisins (KNN) est un modèle d'apprentissage supervisé, utilisé avec un ensemble de données $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, où x_i représente les caractéristiques des données et y_i leurs étiquettes. KNN est non paramétrique, ce qui signifie qu'il n'a pas de paramètres à optimiser lors de l'apprentissage. Contrairement aux modèles paramétriques tels que la régression linéaire, qui ont une structure prédéfinie avec un nombre fixe de paramètres, KNN ne contraint pas les données à une structure spécifique et ne forme pas explicitement un modèle. Il mémorise plutôt l'ensemble des données d'entraînement et les utilise pour les prédictions. KNN est principalement utilisé pour la classification, où les étiquettes y_i représentent des classes plutôt que des valeurs continues comme dans la régression. Pour classer un nouvel exemple, l'algorithme examine les étiquettes des K exemples les plus proches dans l'espace des caractéristiques, où K est un hyperparamètre spécifié par l'utilisateur [DSSQ22].

Avantages de la méthode des k plus proches voisins

Les avantages de la méthode des k plus proches voisins sont les suivants:

- Robustesse face aux données bruitées .

- Efficacité avec des données larges et incomplètes .
- Simplicité .

Ces avantages font de la méthode des k plus proches voisins un outil précieux dans de nombreuses applications, en particulier lorsque les données sont bruitées, larges ou incomplètes, et lorsqu'une approche simple et intuitive est préférée [ALI17].

Inconvénients de la méthode des k plus proches voisins

Les inconvénients de la méthode des k plus proches voisins sont les suivants :

- Dépendance du paramètre k .
- Temps de prédiction long.
- Utilisation importante de la mémoire.

Ces inconvénients doivent être pris en compte lors de la sélection de la méthode des k plus proches voisins pour un problème spécifique, et des techniques d'optimisation et des compromis peuvent être nécessaires pour atténuer ces limitations [ALI17].

2.11.2 Arbre de décision

Les arbres de décision, qu'ils soient utilisés pour la classification ou la régression, sont des méthodes de modélisation permettant de prendre des décisions en fonction des caractéristiques des données. Ils comprennent des nœuds racines représentant les entrées, des nœuds internes effectuant des opérations intermédiaires, et des feuilles représentant les valeurs de sortie.
Le fonctionnement de l'arbre repose sur une séquence de décisions du nœud racine aux feuilles. Chaque nœud décisionnel divise les données en sous-ensembles selon une caractéristique spécifique, se répétant de manière récursive jusqu'à ce que des conditions d'arrêt soient satisfaites.
Les arbres de décision sont construits de manière itérative en recherchant la meilleure décision à chaque nœud pour diviser les données. Cette méthode offre une bonne explicabilité, car le processus de décision est clair.
Avant l'avènement des réseaux de neurones, les arbres de décision étaient largement utilisés pour la modélisation des données. Pour des problèmes complexes, on peut utiliser plusieurs arbres formant une forêt aléatoire, améliorant ainsi les performances du modèle [DSSQ22].

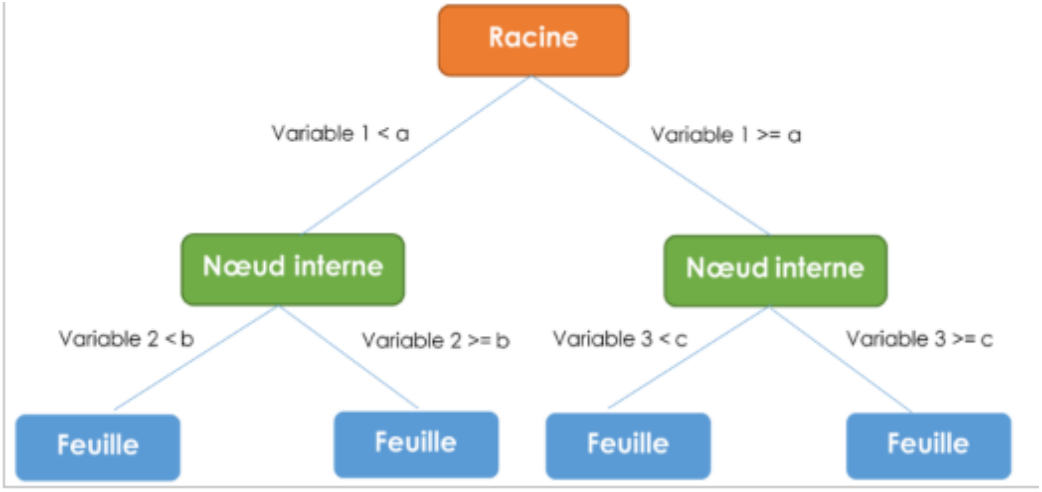

Figure 2.13 : Exemple de division d'un arbre de classification [LL20].

2.11.3 Réseaux de neurones artificiels

Les réseaux de neurones sont des modèles complexes utilisés en apprentissage supervisé pour effectuer à la fois de la régression et de la classification. Ces modèles sont détaillés dans une autre ressource de ce dossier. Leur utilisation s'étend de plus en plus dans de nombreux domaines en raison de leur capacité à modéliser des phénomènes complexes. Ils sont couramment utilisés en traitement d'images pour la reconnaissance de motifs, dans le traitement du langage et pour l'approximation de fonctions complexes nécessitant des calculs longs [DSSQ22].

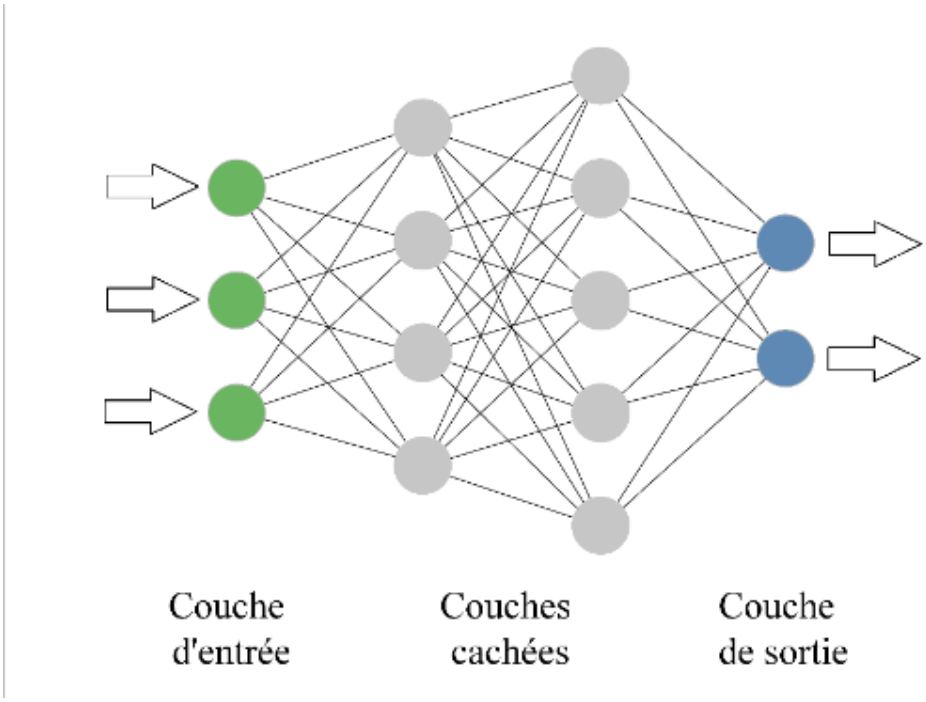

Figure 2.14 : Schéma d'un exemple de réseau de neurones[Mar20] .

Fonctionnement des neurones

Le fonctionnement des neurones peut être décrit comme suit :

1. **Réception de l'influx nerveux :** Les dendrites, les prolongements de la cellule nerveuse, reçoivent l'influx nerveux provenant d'autres neurones ou de stimuli externes.
2. **Évaluation de la stimulation :** Le neurone intègre l'ensemble des stimulations reçues au niveau de ses dendrites. Cette sommation des signaux électriques reçus détermine si le seuil d'excitation est atteint.
3. **Excitation :** Si la stimulation dépasse un seuil critique, le neurone est excité. Il génère alors un potentiel d'action, un signal électrique binaire (0/1), qui se propage le long de son axone.
4. **Propagation de l'excitation :** Le potentiel d'action se propage le long de l'axone du neurone et est transmis aux autres neurones via les synapses, les connexions spécialisées entre les neurones. Ce processus permet la transmission de l'information à travers le réseau neuronal [Ben10].

Avantages des réseaux de neurones

Les réseaux de neurones présentent plusieurs avantages :

- Flexibilité et généralité .
- Traitement des données non structurées .
- Performances dans des domaines complexes .
- Robustesse aux données incomplètes ou bruitées[ALI17].

Inconvénients des réseaux de neurones

Les réseaux de neurones présentent également certains inconvénients :

- Lenteur d'apprentissage.
- Difficulté de sélection des valeurs initiales des poids et d'adaptation du pas d'apprentissage.
- Apprentissage au détriment de la généralisation .
- Manque de transparence [ALI17].

2.11.4 Machine à support de vecteur (SVM)

Les Machines à Support de Vecteur (SVM) représentent une généralisation des classifieurs et des estimateurs linéaires, tels que les modèles de régression linéaire, pour des données de plus grande dimension. Le modèle prend en entrée un vecteur de paramètres X et lui associe une valeur de sortie Y à l'aide d'une fonction $h(X)$. En ce qui concerne la classification, des hyperplans séparateurs sont introduits pour servir de critère de décision. L'espace des valeurs de sortie Y est alors divisé en différentes zones, chacune correspondant à des prédictions. Par exemple, une classe sera prédite si la valeur de Y est positive et une autre si elle est négative [DSSQ22].

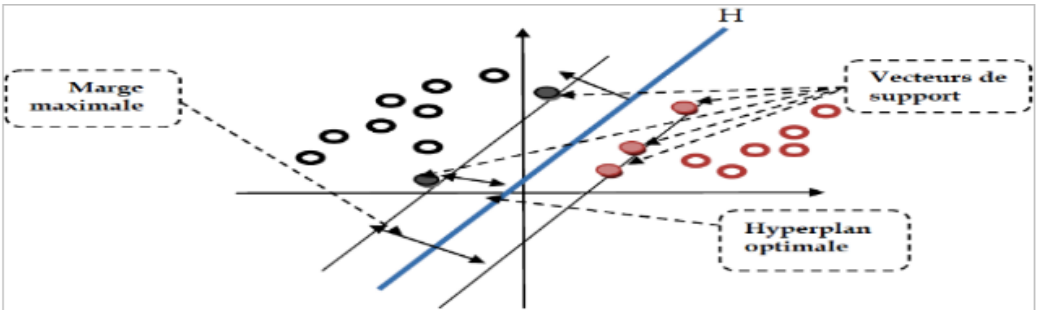

Figure 2.15 : L'hyperplan H optimale, vecteurs supports et marge maximale [LS20].

Avantages des SVM

Les SVM présentent plusieurs avantages significatifs :

- Base théorique solide. Efficacité dans les espaces de grande dimension .
- Flexibilité des fonctions noyau.

Ces avantages font des SVM une méthode populaire et largement utilisée dans de nombreux domaines, notamment la classification, la régression et la détection de motifs [ALI17].

Inconvénients des SVM

Les SVM présentent également certains inconvénients :

- Utilisation de fonctions mathématiques complexes.
- Temps de test élevé [ALI17].

2.11.5 Modèle de mélange de gaussien

Le modèle de mélange de gaussien (GMM) est une approche statistique utilisée pour estimer la distribution de variables aléatoires en les représentant comme une combinaison de distributions gaussiennes ayant des paramètres distincts. Pendant l'entraînement, le GMM cherche à déterminer les moyennes et les variances des différentes distributions gaussiennes en maximisant la vraisemblance des données observées. Ces modèles sont largement employés dans des tâches de classification telles que la reconnaissance d'écriture manuscrite et la segmentation d'images basée sur les histogrammes. Contrairement à l'algorithme des K moyennes, le modèle de mélange de Gauss permet de former des groupes de tailles variées [DSSQ22].

2.12 La relation entre machine-Learning et Deep Learning

Le deep learning, une branche du machine learning, qui à son tour fait partie de l'intelligence artificielle, utilise des réseaux de neurones pour traiter des données complexes telles que des images ou du texte. Il permet de réaliser des tâches nécessitant une compréhension avancée, similaire à celle du cerveau humain. En revanche, le machine learning englobe un ensemble plus large d'algorithmes, n'incluant pas nécessairement des réseaux de neurones complexes. Ainsi, tandis que le deep learning imite le fonctionnement du cerveau humain, le machine learning inclut également des techniques plus simples et directes pour l'analyse des données [Nes22]. Cette figure montre la relation entre l'intelligence artificielle, le machine Learning et le Deep Learning.

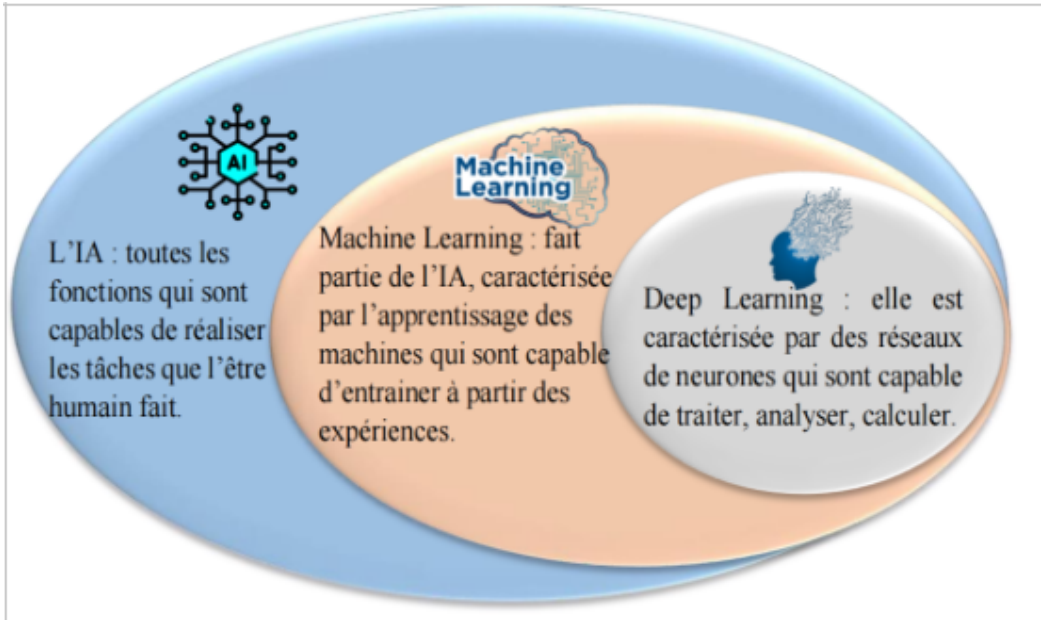

Figure 2.16 : Relation entre IA, machine-Learning et Deep Learning[Nes22].

2.13 L'apprentissage en profondeur

L'apprentissage en profondeur, ou deep learning, est une sous-discipline du machine learning qui implique l'utilisation de réseaux neuronaux composés de trois couches ou plus. Ces réseaux neuronaux tentent de modéliser le comportement du cerveau humain, bien qu'ils ne parviennent pas encore à atteindre ses capacités. Ils "apprennent" à partir de vastes ensembles de données. Alors qu'un réseau de neurones à une seule couche peut fournir des prédictions basiques, l'ajout de couches cachées permet d'optimiser et d'améliorer la précision des résultats. En somme, le deep learning vise à exploiter la structure hiérarchique des données pour extraire des caractéristiques pertinentes et effectuer des tâches complexes [IBM24].

2.13.1 Domaine d'application

Ces domaines d'application illustrent bien la diversité et la puissance du deep learning :

- Reconnaissance d'image .
- Diagnostic médical .
- Traduction automatique .
- Robots intelligents .

Dans chaque domaine, le deep learning permet d'exploiter efficacement de grandes quantités de données pour résoudre des problèmes complexes et réaliser des tâches qui étaient auparavant difficiles voire impossibles à accomplir de manière automatique [Nes22].

2.13.2 Fonctionnement du Deep Learning

Le Deep Learning utilise des réseaux de neurones artificiels pour imiter le fonctionnement du cerveau humain, en analysant des données pour reconnaître, classer et décrire des objets avec précision. Ces réseaux, composés de multiples couches interconnectées, progressent à travers des calculs en avant pour traiter les données d'entrée et produire des prédictions ou classifications à la sortie. La rétropropagation ajuste ensuite les poids et biais des fonctions en fonction des erreurs détectées, permettant ainsi au modèle de s'améliorer au fil du temps.
Bien que cette explication décrive un type de réseau de neurones profonds de manière simplifiée, il existe différentes architectures adaptées à des domaines spécifiques. Par exemple, les réseaux de neurones convolutifs (CNN) sont efficaces dans la vision par ordinateur, tandis que les réseaux neuronaux récurrents (RNN) excellent dans le traitement du langage naturel et de la parole grâce à leur capacité à gérer des données séquentielles ou chronologiques [IBM24].

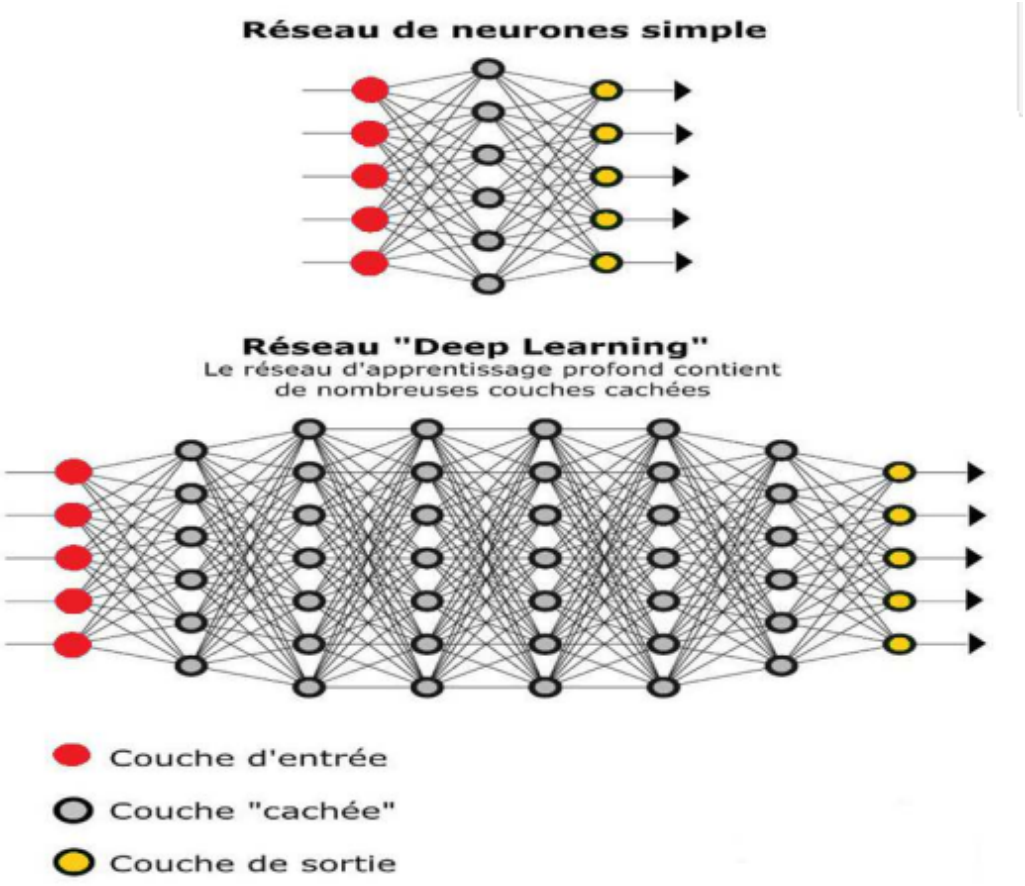

Figure 2.17 : Différence entre un réseau de neurone simple et un réseau profond [EN21].

2.14 les mesures de performance

En l'apprentissage automatique, les mesures de précision sont des métriques utilisées pour évaluer la performance des modèles. Voici quelques-unes des mesures de précision les plus couramment utilisées :

2.14.1 La matrice de confusion

La matrice de confusion est un outil important pour évaluer la performance d'un système de classification. Ses colonnes représentent la répartition des objets dans les classes réelles, tandis que ses lignes indiquent la répartition des points dans les classes estimées par l'algorithme de classification. Lorsqu'il s'agit de classer des pages Web dans un contexte multiclasses (c'est-à-dire lorsque le nombre de classes est supérieur à 2), une approche courante consiste à subdiviser le processus de classification en sous-problèmes. Chaque sous-problème se concentre sur une seule classe, et l'objectif est alors de déterminer si la nouvelle page Web appartient à cette classe spécifique ou non, par rapport aux autres classes disponibles [MAH18].

	Classe réelle : Positif	Classe réelle : Négatif
Prédiction : Positif	Vrais positifs (VP)	Faux positifs (FP)
Prédiction : Négatif	Faux négatifs (FN)	Vrais négatifs (VN)

Table 2.2 : Matrice de confusion

2.14.2 Accuracy

L'exactitude (Accuracy) est une mesure fondamentale et à la fois simple en apprentissage automatique. Elle représente le pourcentage de prédictions correctes par rapport au nombre total de prédictions (multiplié par 100). Un score élevé d'exactitude indique que le modèle effectue des prédictions précises, tandis qu'un score faible suggère une efficacité moindre du modèle. Pour une classification binaire, un score d'environ 50% indique que le modèle produit des résultats similaires à une attribution aléatoire des classes, ce qui signifie qu'il est inefficace [Ami22].

Accuracy=Nomb de prédictions correctes / Nomb tot d'échantillons [MO20].

2.14.3 Précision

La précision (Precision) est une mesure distincte de l'exactitude ("accuracy"), bien que les deux termes se traduisent par "précision" en français. Dans notre contexte, la mesure de "précision" représente le rapport du nombre de lignes correctement classées dans la catégorie "Normal", également appelées "True Positives", sur le nombre total de lignes classées dans la catégorie "Normal", qu'elles soient correctement ou incorrectement classées. En d'autres termes, elle évalue le pourcentage de lignes correctement classées dans la catégorie "Normal" par rapport à l'ensemble des lignes considérées comme normales. Un faible score de "précision" indique que le modèle a tendance à faire des erreurs en classifiant des cas "Anomaly" comme étant dans la catégorie "Normal" [Ami22].

Précision=Vrais positifs / Vrais positifs+Faux positifs [MO20].

2.14.4 Le rappel (Recall)

Le rappel (Recall) représente une mesure complémentaire à la précision. Contrairement à la précision, le rappel est le rapport du nombre de lignes correctement classées dans la catégorie "Normal" sur le nombre total de lignes qui auraient dû être classées dans la catégorie "Normal", qu'elles aient été correctement classées ou non. En d'autres termes, cette mesure évalue le pourcentage de lignes réellement dans la catégorie "Normal" qui ont été correctement identifiées. Un faible score de rappel indique que le modèle a tendance à mal classifier des cas "Normal" comme étant des "Anomaly".
Dans le cas d'un jeu de données déséquilibré, où une classe contient significativement plus d'exemples que l'autre, il est possible d'obtenir un score

élevé pour l'exactitude (accuracy), mais avec des scores très bas pour la précision et/ou le rappel [Ami22].

Rappel= Vrais positifs / Vrais positifs+Faux négatifs [MO20].

2.14.5 Le score F1 (F1-score)

Le score F1, une mesure de classification, évalue la capacité d'un modèle à prédire correctement les instances positives, en tenant compte à la fois de la précision et du rappel. Il représente la moyenne harmonique de ces deux indicateurs, nécessitant ainsi des valeurs élevées pour les deux pour que le score F1 soit élevé également [Ami22].

F1=2× (Précision×Rappel / Précision+Rappel) [MO20].

2.14.6 Taux d'erreur (Error Rate)

En général, le taux d'erreur est une mesure utilisée pour évaluer la performance d'un modèle. Il indique la proportion de prédictions incorrectes par rapport à l'ensemble des prédictions effectuées. C'est couramment utilisé dans des contextes de classification, mais il peut aussi s'appliquer à d'autres modèles.

2.14.7 MAE (Mean Absolute Error)

La moyenne des valeurs absolues des erreurs entre les valeurs prédites et les valeurs réelles.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

.

2.14.8 MSE (Mean Squared Error)

La moyenne des carrés des erreurs entre les valeurs prédites et les valeurs réelles.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

.

2.14.9 RMSE (Root Mean Squared Error)

La racine carrée de la moyenne des carrés des erreurs.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

2.14.10 R2 (Coefficient de détermination)

Une mesure statistique qui indique la proportion de la variance dans la variable dépendante qui est prévisible à partir des variables indépendantes.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

.

Ces métriques aident à évaluer et à comparer la précision et la performance des modèles de régression, permettant ainsi de choisir le modèle le plus approprié pour les prédictions.

2.15 L'overfitting et l'underfitting

L'overfitting (sur-apprentissage) et l'underfitting (sous-apprentissage) sont des concepts essentiels en apprentissage automatique. Ils sont les causes principales des mauvaises performances des modèles prédictifs générés par les algorithmes de apprentissage automatique.

2.15.1 Surapprentissage

Cela se produit lorsqu'un modèle mémorise les données d'entraînement au lieu d'apprendre à généraliser à de nouvelles données. En conséquence, il fonctionne bien sur les données d'entraînement, mais échoue sur les données de validation, car il ne peut pas faire de bonnes prédictions sur des données nouvelles et différentes. Pour y remédier, il est nécessaire d'améliorer la capacité de généralisation du modèle, par exemple en utilisant des techniques de régularisation ou en appliquant un arrêt précoce (early stopping) [Mau18].

Les méthodes pour éviter le surapprentissage

Il existe plusieurs méthodes pour éviter le surapprentissage d'un modèle prédictif. En voici quelques-unes :

1. **Augmenter la quantité de données d'entraînement:** Fournir plus de données au modèle durant la phase d'entraînement permet aux algorithmes de mieux généraliser, réduisant ainsi le risque qu'ils s'adaptent trop aux échantillons spécifiques.

2. **Limiter la complexité du modèle:** Un modèle trop complexe risque de mémoriser les données d'entraînement plutôt que d'apprendre des tendances générales. Garder le modèle aussi simple que possible aide à éviter cela.

3. **Utiliser la validation croisée (cross-validation) :** Cette technique consiste à diviser les données d'entraînement en plusieurs sous-groupes. Un des sous-groupes est utilisé pour le test, tandis que les autres sont utilisés pour l'apprentissage. Ce processus est répété

jusqu'à ce que chaque sous-groupe ait été utilisé pour le test. La validation croisée améliore la performance des algorithmes de machine learning en permettant une évaluation plus rigoureuse du modèle [Ora22].

2.15.2 Sous-apprentissage

L'underfitting, également appelé sous-ajustement ou sous-apprentissage, se produit lorsque le modèle souffre d'un grand biais. Cela survient lorsque seules des variables pertinentes sont utilisées ou lorsque le modèle est arrêté prématurément durant l'apprentissage, rendant le modèle trop simpliste et incapable de saisir la tendance générale.
Un modèle sous-ajusté s'adapte mal au jeu d'entraînement, produisant des prédictions biaisées et de nombreuses erreurs en phase d'apprentissage. En d'autres termes, il est trop généraliste et ne parvient pas à fournir des analyses prédictives précises.
Maîtriser ce concept est essentiel en machine learning avant de concevoir des modèles adéquats. Avant de voir comment éviter l'underfitting, il est également important d'étudier son contraire, l'overfitting [Kas22].

Techniques pour éviter l'underfitting

Pour éviter l'underfitting, plusieurs techniques peuvent être employées :

- Utiliser des modèles plus complexes ou ajouter des couches à un réseau de neurones.
- Ajouter des caractéristiques ou des variables explicatives pertinentes.
- Augmenter la quantité de données d'entraînement disponibles.
- Appliquer des techniques de prétraitement des données pour améliorer leur qualité.
- Changer l'algorithme d'apprentissage ou ajuster les hyperparamètres pour affiner le modèle [Rah23].

2.15.3 Différence entre underfitting et overfitting

À l'inverse de l'overfitting (surapprentissage), où le modèle prédictif s'ajuste trop précisément aux données d'entraînement, l'underfitting (sous-apprentissage) se produit lorsque le modèle de machine learning ne s'adapte pas suffisamment aux données d'entraînement.
Les algorithmes en situation de sous-apprentissage (underfitting) ne parviennent pas à généraliser avec des données nouvelles. Leurs performances sont insuffisantes, et leurs prédictions contiennent trop d'erreurs pour être viables [com23].

2.15.4 Le right fitting

Le right fitting consiste à ajuster les paramètres d'un modèle pour minimiser l'erreur de prédiction sur les données d'entraînement. L'objectif est de trouver un équilibre entre la complexité du modèle et sa capacité à généraliser aux nouvelles données. En effet, un modèle trop complexe risque de surapprendre les données d'entraînement, tandis qu'un modèle trop simple peut entraîner un sous-apprentissage [Rah23].

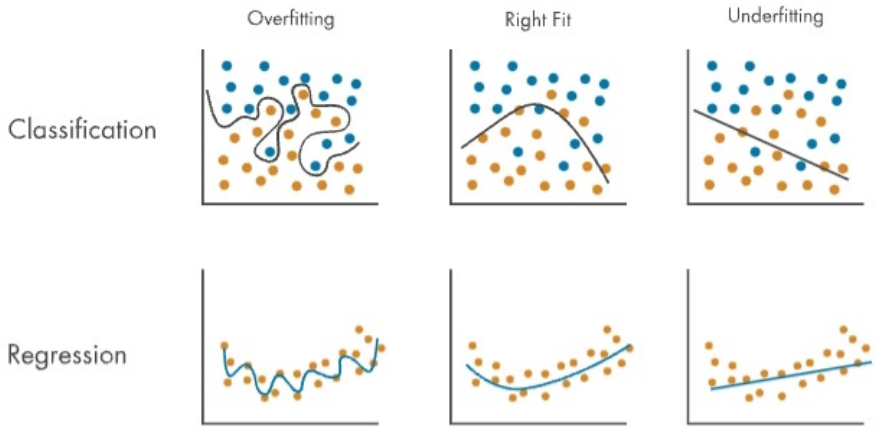

Figure 2.18 : Surapprentissage ,Ajustement Correct et Sous-apprentissage dans les modèles de classification et de régression [Mat24].

2.16 Conclusion

Dans ce chapitre, nous avons abordé les fondamentaux de l'apprentissage automatique, en examinant son fonctionnement et les algorithmes qui le sous-tendent. Nous avons également discuté des limitations inhérentes à ces algorithmes, ainsi que des divers domaines d'application de l'apprentissage automatique.

CHAPITRE 3

Les séries temporelles

3.1 Introduction

Dans le domaine de la prédiction temporelle, l'apprentissage automatique est utilisé pour anticiper les valeurs futures d'une série chronologique en se basant sur ses données antérieures. Ce chapitre abordera les composants des séries temporelles, leur importance et quelques algorithmes d'apprentissage automatique pour la prédiction temporelle.

3.2 Définition de série temporelle

Une série temporelle, également appelée série chronologique, est une séquence d'observations $x_1, x_2, \ldots, x_n$ indexée par le temps. On considère généralement qu'elle est une manifestation d'un processus X, représentant une suite de variables aléatoires $\{X_i\}$ [Mon17].

3.3 Les objectifs de l'analyse des séries temporelles

L'étude des séries temporelles vise principalement à comprendre l'évolution d'un phénomène au fil du temps, dans le but de prédire son comportement futur. Comprendre ce processus implique de trouver une structure qui explique le phénomène de manière intelligible. Cela implique souvent de détecter la présence de tendances et de saisons afin de réaliser des prévisions précises [TAL18].

3.4 Les composantes d'une série temporelle

Une série temporelle est généralement constituée de plusieurs éléments:

3.4.1 Tendance

Une "tendance" dans une série temporelle indique tout changement systématique à long terme dans son niveau. Cette évolution peut pré-

senter une direction et une pente constantes ou variables. Lorsque cette tendance est déterministe, c'est-à-dire qu'elle provient d'effets constants et déterministes, il est approprié de la modéliser avec une approche de régression. Cependant, des tendances stochastiques peuvent également se produire, résultant du mouvement aléatoire d'une variable dans le temps. Les tendances stochastiques sont souvent caractérisées par des changements imprévisibles dans leur pente et leur direction, et elles ne peuvent pas être expliquées par des facteurs externes. Essayer de modéliser ces tendances de manière déterministe entraîne des erreurs. La distinction entre ces deux types de tendances peut être réalisée par divers moyens, tels que la théorie, l'inspection visuelle ou des tests spécifiques [JT17].

3.4.2 Saisonnalité

La composante saisonnière, ou saisonnalité, se réfère à un phénomène qui se reproduit à intervalles réguliers dans le temps. Elle est souvent associée à des variations saisonnières[Far12]. Par exemple, en météorologie, les températures plus basses en hiver par rapport à l'été illustrent la saisonnalité. De même, en économie, la saisonnalité peut être influencée par les vacances, les périodes de fêtes, ou encore le climat, comme observé dans les chiffres d'affaires des magasins [Mon17].

3.4.3 Cycle

Une variation cyclique, notée $C(t)$, se produit lorsque la variable observée présente des fluctuations d'augmentation et de diminution à une fréquence irrégulière. En général, la durée moyenne d'un cycle est beaucoup plus longue que celle de la composante saisonnière. De plus, les mouvements pendant un cycle tendent à être plus variables que ceux de la composante saisonnière. Étant donné que nous disposons de seulement une année de données, il n'est pas possible d'observer de variation cyclique dans nos données [MEZ20].

3.4.4 Résidus

Dans l'analyse des séries temporelles, les résidus (ou erreurs) représentent la variation irrégulière, souvent assimilée au bruit statistique, similaire aux erreurs dans les modèles statistiques. Ce bruit aléatoire peut être corrélé dans le temps ou non. Lorsqu'il n'y a pas de corrélation, on parle de bruit blanc. Le bruit blanc revêt une importance fondamentale dans la théorie et la pratique de l'analyse des séries temporelles car il permet d'évaluer la qualité des modèles. Les résidus d'un modèle ajusté forment une série temporelle propre, où des résidus non corrélés et distribués normalement indiquent un bon ajustement du modèle aux données [JT17].

3.4.5 Stationnarité

Les séries temporelles sont considérées comme stationnaires lorsque leurs propriétés statistiques ne varient pas dans le temps, ce qui signifie qu'elles présentent une moyenne et un écart-type constants [No[U+FFFD]2].

3.5 Techniques de décomposition-recomposition

Les techniques de décomposition-recomposition sont des schémas utilisées pour analyser les séries temporelles .Voici les trois techniques de décomposition-recomposition:

3.5.1 Le schéma additif

Une méthode de décomposition fondamentale est la suivante : $X_t = m_t + s_t + U_t$, où :
- $(m_t)_t$ est une composante de tendance déterministe qui reflète le comportement à long terme de la variable observée, comme une croissance ou une décroissance linéaire ou quadratique. Cette composante peut également varier selon les périodes, par exemple en adoptant une forme affine par morceaux. Par exemple, dans le cas de la consommation d'électricité représentée dans la Figure 1.2, une tendance affine $m_t = a_t + b$ peut être observée. De manière plus générale, cette composante peut être considérée comme une fonction lisse du temps t.
- $(s_t)_t$ est une séquence périodique correspondant à une composante saisonnière, par exemple avec une période de 12 pour les séries de trafic de passagers et de consommation d'électricité, ou une période de 4 pour des séries trimestrielles, ou encore une période de 24 pour des séries horaires en météorologie. Dans certains cas, une somme de plusieurs séquences périodiques peut être pertinente, comme lors de l'observation de séries de températures horaires sur plusieurs années, où une périodicité quotidienne et annuelle doit être prise en compte.
- $(U_t)_t$ représente une composante irrégulière et aléatoire, généralement de faible amplitude par rapport à la composante saisonnière, mais significative en pratique car elle est souvent autocorrélée. Cela signifie que la covariance entre U_t et U_{t+h} est non nulle. Différents types de modèles peuvent être utilisés pour étudier cette composante [[U+FFFD]dlSedldl15].

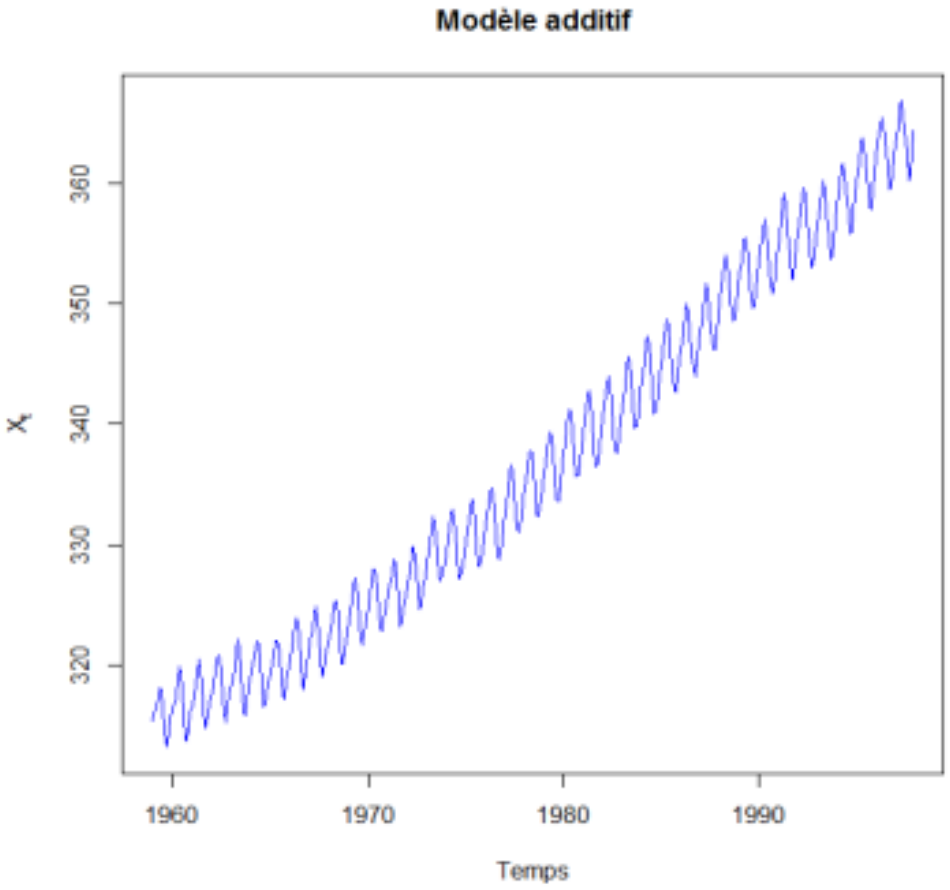

Figure 3.1 : Représentation graphique d'un schéma additif [Sam20].

3.5.2 Le schéma multiplicatif

Dans le schéma multiplicatif, chaque observation x_t est décomposée en une composante saisonnière (S) et une composante extra-saisonnier (E) avec une interaction entre elles, permettant une variation souple de l'amplitude saisonnière au fil du temps [TB98].

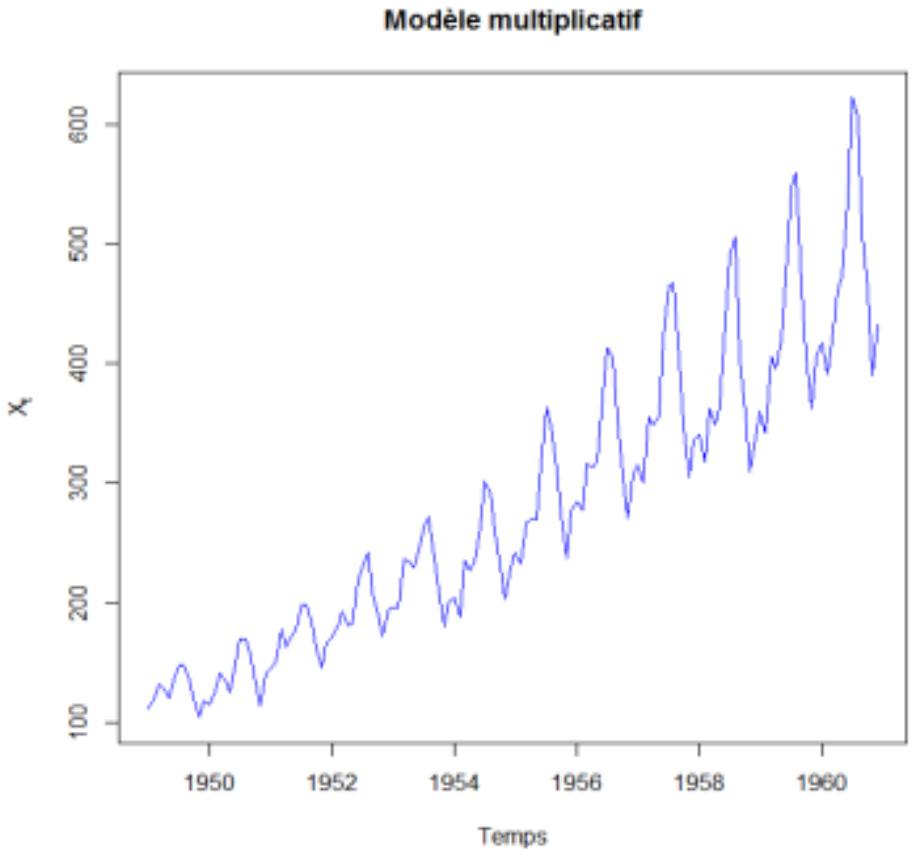

Figure 3.2 : Représentation graphique d'un schéma multiplicatif [Sam20].

3.5.3 Le schéma multiplicatif complet

Le schéma multiplicatif complet, où chaque observation x est le produit des composantes extra-saisonnier (E), saisonnière (S), et irrégulière (R), représente une interaction générale entre ces trois composantes. Il est largement adopté en économie en raison de sa commodité : le passage au schéma additif est facilité en prenant le logarithme de la série chronologique.[TB98]

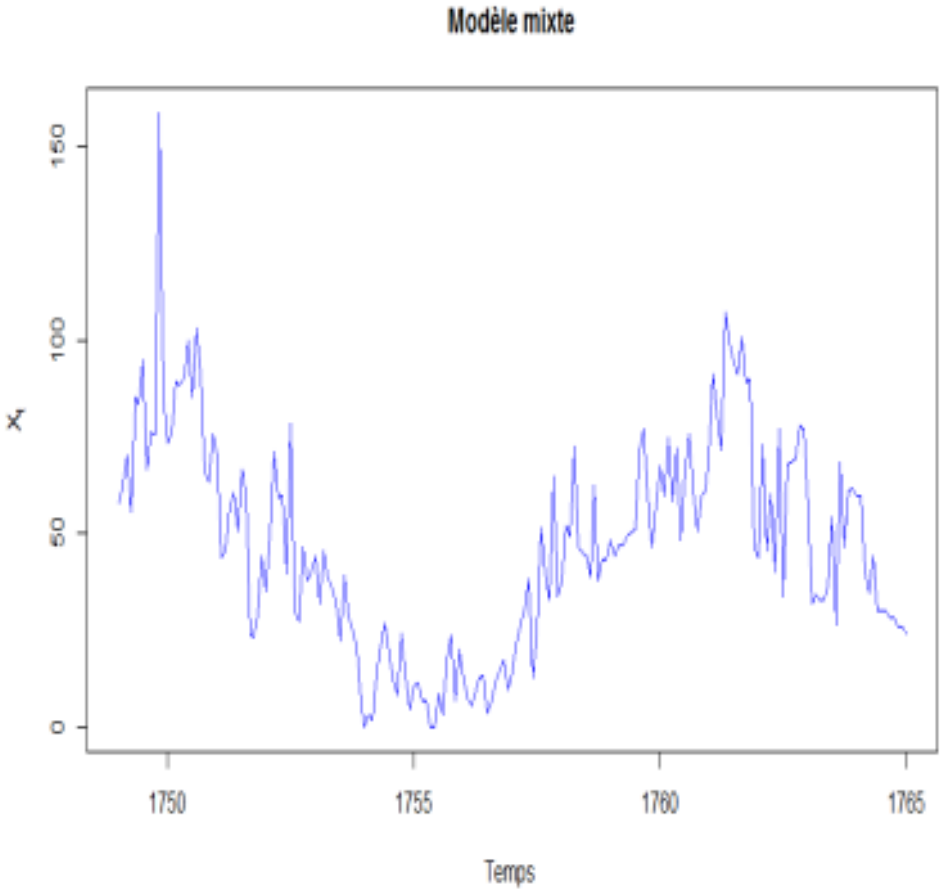

Figure 3.3 : Représentation graphique d'un schéma multiplicatif complet [Sam20].

3.6 Intérêt des séries temporelles

Il est courant d'évaluer l'utilité des séries temporelles selon trois perspectives principales : descriptive, explicative et prévisionnelle.

3.6.1 Description

L'analyse temporelle offre un aperçu de la structure de la série de données examinée. Elle peut également servir à comparer une série avec d'autres séries, telles que la varicelle et les oreillons, par exemple [Mec13].

3.6.2 Explication

- Les variations d'une série peuvent être attribuées à une autre série, telles que l'exposition météorologique ou la pollution atmosphérique.
- Il est envisageable de modéliser l'impact d'une intervention externe grâce à l'analyse des séries temporelles.

- Ces analyses permettent également de créer des scénarios pour la période contemporaine : en manipulant une variable explicative, il est possible d'observer le comportement de la variable expliquée [Mec13].

3.6.3 Prévision

- La prévision a priori facilite la planification.
- La prévision a posteriori est utile pour évaluer l'impact d'une perturbation, comme le dépistage, sur la variable expliquée.
- Enfin, il est possible de créer des scénarios pour le futur [Mec13].

3.7 Exemples de séries temporelles

Exemples de séries temporelles dans différents domaines scientifiques et appliqués :

- **Météorologie :** données météorologiques telles que la température, la pression atmosphérique et la vitesse du vent.
- **Économie et finance :** variables économiques comme le PIB, les indices boursiers, les taux de change et les écarts.
- **Marketing :** activité commerciale et chiffre d'affaires [Bon13].
- **Médecine/biologie :** suivi des évolutions des maladies, analyse des électroencéphalogrammes et électrocardiogrammes.
- **Traitement du signal :** signaux de communication, de radar et de sonar, analyse de la parole.
- **Traitement des données :** séries de mesures successives de la position ou de la direction d'un objet mobile (trajectoire) [Mec13].

Voici un exemple de série temporelle annuelle : Considérons la série de la population du Sénégal de 2012 (P1) à 2020, notée (Pn) pour n appartenant à l'ensemble des entiers naturels non nuls.
Remarque : Les données peuvent être téléchargées depuis le dépôt GitHub de Ibrahima Tall, dans le dossier ENSAETraining-Courses, et sont disponibles sur la feuille Excel POPAN [TAL18].

Dates	2012	2013	2014	2015	2016	2017	2018	2019	2020
Population	13,4	13,7	14,1	14,5	14,9	15,4	15,8	16,2	16,7

Table 3.1 : Population totale du Sénégal

Economie:

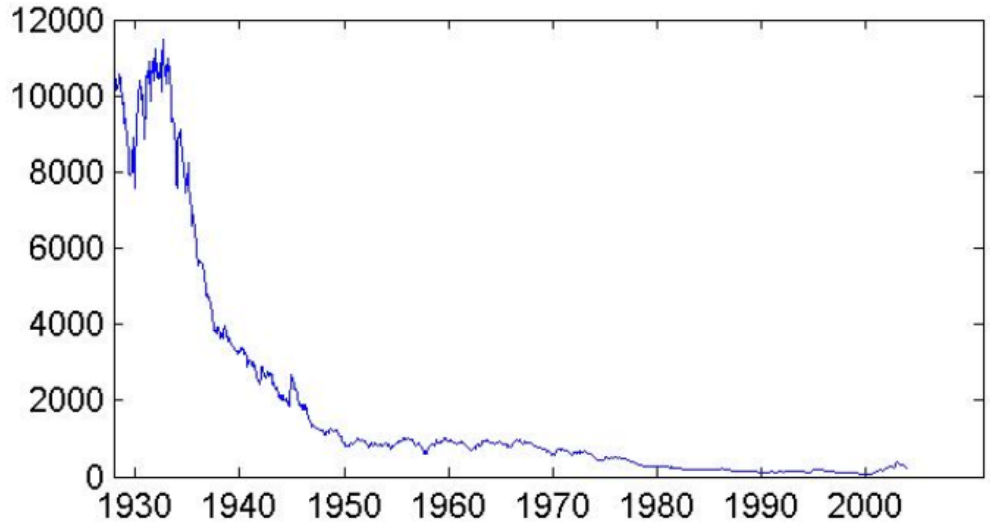

Figure 3.4 : évolution du cours du Dow Jones entre 1928 et 2004

Environement:

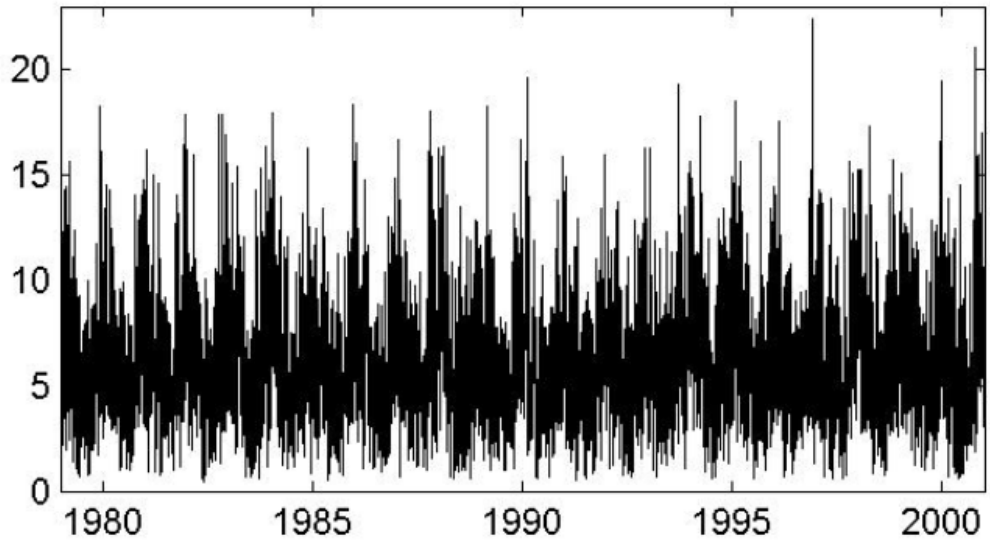

Figure 3.5 : évolution de l'intensité du vent, en m/s, au large de la Bretagne entre 1979 et 2001 [Mon17].

3.8 Calcule de prévision

Pour calculer les prévisions dans un modèle, vous pouvez utiliser les coefficients estimés du modèle pour prédire les valeurs futures de la série temporelle. Voici les méthodes générales pour calculer les prévisions :

3.8.1 Méthode de Buys-Ballot

La méthode de Buys-Ballot concerne les séries temporelles suivant un modèle additif avec une tendance linéaire. Nous supposons que le modèle

est donné par :

$$X_t = a + bt + \sum_{i=1}^{p} \beta_i S_i + \varepsilon_t$$

Cette méthode consiste à estimer les paramètres de ce modèle selon le critère des moindres carrés, puis à réaliser des prévisions. L'estimateur des paramètres $\hat{\theta} = (a, b, \beta_1, \beta_2, \ldots, \beta_p)^T$ est la solution du système :

$$\arg\min_{\hat{\theta}} \sum_{t=1}^{T} (X_t - a - bt - \sum_{j=1}^{p} \beta_j S_j)^2$$

Les formules des estimateurs sont donc :

$$\hat{b} = \frac{1}{2mN}\left(\frac{N(N^2-1)}{2}\right)\sum_{k=1}^{N} kX_k - \frac{N(N+1)}{2}X!,$$

$$\hat{a} = X - \hat{b}Nm + \frac{1}{2},$$

$$\hat{\beta}_j = X_j - X - \hat{b}\cdot\theta_j - m + \frac{1}{2},$$

avec les notations suivantes : - N : le nombre d'années. - m : le nombre de saisons pour chaque année. - posons : $t = j + m(k-1)$. - $X_{\bar{k}}$: moyenne de X_t relative à l'année $k, k = 1, N$. - $X_{\bar{j}}$: moyenne de X_t relative à la saison $j, j = 1, m$. - $X = \frac{1}{T}\sum_{t=1}^{T} X_t = \frac{1}{N}\sum_{k=1}^{N} X_{\bar{k}} = \frac{1}{m}\sum_{j=1}^{m} X_{\bar{j}}$: moyenne de l'ensemble des observations.

Dans ce cas, la prédiction d'une valeur future X_{T+h} à l'horizon h est donnée par :

$$\hat{X}_T(h) = \hat{X}_{T+h} = \hat{a} + \hat{b}(T+h) + \sum_{j=1}^{m} \hat{\beta}_j S_j$$

[Sam20]. **Exemple:**

Dates	T1	T2	T3	T4	Moyenne	Ecart type
1994	1248	1392	1057	3159	1714	842.69
1995	891	1065	1118	2934	1502	831.02
1996	1138	1456	1224	3090	1727	795.48
Moyenne	1092	1304	1133	3061	Moyenne générale	Ecart type général
Ecart type	149	171	69	94	1647.7	829.74

Table 3.2 : Exemple de constitution d'un tableau de Buys-Ballot pour des ventes trimestrielles

Années				
1994	T4	T2	T1	T3
1995	T4	T3	T2	T1
1996	T4	T2	T3	T1

Table 3.3 : Classement des trimestres en fonction de leurs valeurs [TB98].

Les données présentées dans les tableaux suivent une structure où les années sont disposées en colonnes et les trimestres sont les facteurs analysés. Pour chaque année, les valeurs des trimestres (T1, T2, T3, T4) sont enregistrées, ainsi que la moyenne et l'écart type calculés pour chaque année et pour l'ensemble des observations.
En utilisant ces informations, les trimestres sont classés pour chaque année en ordre décroissant en fonction de leurs valeurs, comme illustré dans le tableau 2. Cette classification permet de visualiser la tendance des trimestres au fil des années.
Dans cet exemple, on observe que le trimestre T4 persiste à se classer en première position quelle que soit l'année, tandis que le trimestre T1 occupe généralement une position de "creux". Cette constance dans le classement des trimestres suggère l'existence d'une saisonnalité dans les données, où le trimestre T4 tend à avoir des valeurs plus élevées que les autres trimestres, et le trimestre T1 des valeurs relativement basses.

3.8.2 Lissage exponentiel

L'objectif est de prédire les valeurs futures d'une série temporelle $X_1, X_2, \ldots, X_T$, représentées par $X_{T+1}, X_{T+2}, \ldots$. Cette prédiction est notée $\hat{X}_T(h)$ pour $h \in \mathbb{N}^*$, où h désigne l'horizon de la prévision. Aucun modèle statistique spécifique n'est présumé dans cette approche [[U+FFFD]dlSedldl15].

3.9 Analyse et caractérisation des séries temporelles

L'analyse et la caractérisation des séries temporelles sont essentielles pour comprendre l'évolution des données dans le temps et faire des prédictions éclairées. Les principales méthodes utilisées incluent:

3.9.1 Fonction d'auto-correlation

La fonction d'autocorrélation (ACF), souvent utilisée pour comparer plusieurs séries entre elles, mesure la corrélation entre une série et elle-même à différents intervalles de temps. Dans le cas d'un processus discret $X = \{X_1, \ldots, X_n\}$, la fonction d'autocorrélation $\rho(t, t+h)$ est définie comme la covariance corrigée de la variance :

$$\rho(t, t+h) = \frac{E[(X_t X_{t+h})] - E[X_t]E[X_{t+h}]}{Variance(X_t) \cdot Variance(X_{t+h})}$$

L'estimation de la fonction d'autocorrélation se fait par la fonction d'autocorrélation empirique :

$$\hat{\rho}(t, t+h) = \frac{\hat{C}(t, t+h)}{\hat{C}(t, t) \cdot \hat{C}(t+h, t+h)}$$

Si $\rho(t, t+h)$ ne dépend pas du temps (séries stationnaires d'ordre deux), alors on estime $\hat{\rho}(h) = \hat{\rho}(t, t+h)$ pour tout t. Pour une réalisation donnée,

les corrélations $\hat{\rho}(t, t+h)$ peuvent être visualisées en traçant des nuages de points de (x_i, x_{i+h}) pour $i = 1, \ldots, n-h$ [[U+FFFD]dlSedldl15]. **Exemple:**

	retards	autocorrélation
1 2 3 . . . t . . . n	0	r = 1
1 2 3 . . . t-1 . . . n-1	1	r1
1 2 3 . . . t-2 . . . n-2	2	r2
. . .	. . .	. . .
t-k . . . n-k	k	r_k
. . .	. . .	. . .
t-k . . .n-k	k	r_k

Table 3.4 : Exemple de calcul d'une autocorrélation

Dans ce tableau, nous avons une illustration du calcul des coefficients d'autocorrélation pour différents retards. Chaque ligne représente une observation de la série temporelle, allant de l'indice 1 à l'indice n. Les retards indiquent le décalage par rapport à l'observation courante, allant de 0 à k. Les coefficients d'autocorrélation correspondants sont notés r_0 à r_k.
Les coefficients d'autocorrélation sont calculés pour des ordres allant de 0 à k, où k représente le décalage maximum admissible. En général, ce décalage maximum est défini comme 6, 3 ou 5, en fonction de la taille de la série temporelle (n). Ceci garantit que les coefficients d'autocorrélation conservent leur signification statistique [TB98].

3.9.2 Fonction d'autocorrélation partielle (PACF)

La fonction d'autocorrélation partielle (PACF) évalue la corrélation entre $y(t)$ et son retard $y(t-k)$, en tenant compte de l'influence des valeurs $y(t-k-i)$, pour tout $i > k$. Le corrélogramme partiel, représentant le PACF pour différents ordres k, permet de visualiser cette relation[MEZ20].

3.10 Stationnarité des séries temporelles

La stationnarité dans les séries temporelles est une propriété importante qui indique que les caractéristiques statistiques de la série ne changent pas avec le temps. Il existe deux types de stationnarité :

3.10.1 Stationnarité strict

Une série chronologique X_t est dite strictement (ou fortement) stationnaire si, pour tout entier positif k et pour tout $(t_1, \ldots, t_k)$ dans T^k et h dans $\mathbb{Z}$, les distributions jointes de $(X_{t_1}, \ldots, X_{t_k})$ et $(X_{t_1+h}, \ldots, X_{t_k+h})$ sont identiques [Mer21].

3.10.2 Stationnarité du second ordre

Une série temporelle X_t est dite stationnaire du 2nd ordre (ou faiblement stationnaire), ou tout simplement stationnaire, si :

1. $E(X_t) = \mu$, pour tout $t \in T$,
2. $\text{var}(X_t) = \sigma^2 < \infty$, pour tout $t \in T$,
3. $\text{cov}(X_s, X_r) = \text{cov}(X_{s+t}, X_{r+t})$, pour tout $r, s, t \in T$ [Sam20].

La stationnarité revêt une importance cruciale dans l'analyse des séries temporelles, notamment dans le contexte de modèles comme les modèles ARIMA (AutoRegressive Integrated Moving Average). Ces modèles supposent que les données sont stationnaires pour être correctement appliqués. Pour évaluer la stationnarité, on peut recourir à des tests statistiques tels que le test de Dickey-Fuller ou bien examiner graphiquement la série temporelle afin de repérer toute tendance ou saisonnalité. Si la série ne présente pas de stationnarité, des techniques de transformation comme la différenciation peuvent être employées pour rendre la série stationnaire [MEZ20].

3.11 Les modèles de prédiction

Les modèles de prédiction sont des outils mathématiques ou statistiques utilisés pour estimer ou prévoir les valeurs futures d'une série de données.
Voici quelques-uns des types de modèles de prédiction les plus couramment utilisés :

3.11.1 Les modèles Auto-Régressifs

Les modèles auto-régressifs (AR) sont des modèles statistiques qui utilisent les valeurs passées d'une série temporelle pour prédire ses valeurs futures. en les classifiant en deux catégories, les modèles uni-variés et les modèles multi-variés.

Les modèles uni-variés

Les modèles uni-variés permettent de prédire une seule série temporelle, en regardent les dernières valeurs de la série pour estimer les valeurs futures [Hma18].

Le modèle AR : On dit que X_t est un modèle autorégressif d'ordre p (abrégé AR(p)), où $p \geq 1$, si :

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \epsilon_t = \sum_{j=1}^{p} \phi_j X_{tj} + \epsilon_t \quad (1.2)$$

où $\phi_1, \phi_2, \ldots, \phi_p$ sont des constantes ($\phi_p \neq 0$), et ϵ_t suit une distribution normale centrée avec une variance σ^2. On utilise couramment la notation suivante :

$$\Phi(L)X_t = \epsilon_t$$

avec $\Phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \cdots - \phi_p L^p$.

Dans ce contexte, un modèle AR(p) est considéré comme stationnaire si toutes les racines du polynôme caractéristique $\Phi(z)$ sont de module strictement supérieur à 1 [Mer21].

Le modèle MA : Le processus moyenne mobile (MA) est un modèle de série temporelle défini par un polynôme caractéristique $B(z)$ de degré q avec des racines dont le module est strictement supérieur à 1, ainsi qu'un bruit blanc centré $(\epsilon_t)_{t\in\mathbb{Z}}$. Dans un MA(q), la série est la somme pondérée du bruit blanc courant et de ses q valeurs précédentes, où les poids sont définis par les coefficients du polynôme $B(z)$.

Un MA(q) est stationnaire et centré, et son autocorrélogramme montre des valeurs significatives seulement pour les h premiers retards, où $h \leq q$. Contrairement aux modèles autorégressifs, l'autocorrélation partielle d'un processus MA décroît de manière exponentielle. En outre, un MA(q) peut être représenté comme un processus autorégressif d'ordre 1 (AR(1)), simplifiant ainsi son interprétation et son estimation [Mon13].

Le modèle ARIMA: Le modèle ARIMA, ou modèle autorégressif intégré à moyenne mobile, est une extension du modèle ARMA (AutoRegressive Moving Average). Avant de discuter de l'ARIMA, il est important de comprendre l'ARMA. Le modèle ARMA (p, q), développé par Box en 2013, combine deux processus : l'AR(p) (autorégressif) et le MA(q) (moyenne mobile). Il prend en compte à la fois les termes d'erreurs et les valeurs précédentes de la série temporelle, et est défini comme suit :

$$Y(t) = y_0 + \sum_{i=1}^{p} \alpha_i Y(t-i) + \sum_{i=1}^{q} \theta_i U(t-i) + U(t)$$

Ici, y_0, α_0, ..., α_p, et θ_0, ..., θ_q sont les paramètres du modèle. Pour les séries temporelles non-stationnaires, le modèle ARIMA (p, d, q) est plus approprié. Il consiste à appliquer le modèle ARMA(p, q) sur une série temporelle transformée par différenciation d'ordre d. Cela signifie calculer d fois les différences entre les observations consécutives [Hma18].

Les modèles multi-variés

Les modèles multi-variés sont motivés par la volonté d'améliorer la qualité des prédictions des séries temporelles en utilisant des données supplémentaires par rapport aux modèles uni-variés. Ils visent à exploiter d'autres sources d'informations tout en suivant la même logique que les modèles uni-variés. L'hypothèse principale de ces modèles est que les valeurs futures d'une série cible dépendent non seulement de ses propres valeurs précédentes, mais aussi d'autres séries qui peuvent influencer la série cible [Hma18].

3.12 Apprentissage automatique pour la prédiction temporelle

La prévision des séries chronologiques est cruciale dans de nombreux domaines scientifiques, industriels et économiques, comme la météorologie, les télécommunications et la finance. Elle implique souvent la prédiction de multiples étapes dans le futur, nécessitant des prévisions à court, moyen et long terme. Alors que les approches statistiques traditionnelles comme les modèles ARIMA ont dominé ce domaine, les séries chronologiques réelles montrent souvent des comportements non linéaires, stimulant ainsi l'émergence de modèles d'apprentissage automatique. Ces modèles, tels que les réseaux neuronaux, ont gagné en popularité ces dernières décennies, surpassant parfois les méthodes statistiques classiques dans diverses applications. Malgré cela, la recherche sur l'intersection entre la prévision des séries chronologiques et l'apprentissage automatique reste relativement limitée. Les compétitions de prévision et les études comparatives ont montré des résultats mitigés, soulignant le besoin d'explorer davantage l'utilisation de l'apprentissage automatique dans ce domaine. En outre, le rôle potentiel de l'apprentissage automatique dans le traitement des séries chronologiques pour la gestion de vastes ensembles de données reste un domaine de recherche important et prometteur [Tai14].

3.12.1 Prédire avec la régression linéaire

Une fois que l'équation d'une régression linéaire est établie, elle peut être utilisée pour prédire les valeurs de la variable dépendante en fonction de nouvelles valeurs des variables indépendantes. Ces valeurs peuvent être réelles ou hypothétiques, dérivées de scénarios futurs. Par exemple, les revenus peuvent être prédits en se basant sur des scénarios hypothétiques impliquant des valeurs spécifiques de cours d'actions ou d'investissements marketing.
Pour obtenir des prédictions à partir d'une régression dans XLSTAT, vous pouvez utiliser l'onglet "Prédiction" de la boîte de dialogue de régression linéaire. Cette fonctionnalité permet d'appliquer l'équation de régression obtenue à de nouvelles données afin de générer des prédictions pour la variable dépendante [Lum23].

3.12.2 Prédire avec Random Forest

Les arbres de décision (la forêt aléatoire) sont des outils d'exploration de données largement utilisés en Data Science. Ils permettent de modéliser une hiérarchie de tests pour prédire un résultat dans le cadre de l'apprentissage supervisé, en particulier pour la classification de données.
Le fonctionnement des arbres de décision repose sur des règles de logique simples. Les différentes décisions possibles sont représentées sur les feuilles de l'arbre, situées aux extrémités des branches. Ces décisions sont modifiées en fonction des décisions prises à chaque nœud, formant ainsi une séquence de règles. Les arbres de décision sont des modèles non

paramétriques et nécessitent peu de prétraitement des données. Ils sont également facilement interprétables et entraînables.
Dans le cadre de l'analyse, les arbres de décision utilisent des variables explicatives pour expliquer une variable cible, représentées par une matrice X avec n variables et m observations associées à un vecteur Z. Pour établir une relation entre X et Z, les arbres de décision partitionnent les données en groupes d'individus similaires, en se basant sur la variable à prédire. Le résultat obtenu met en évidence les relations hiérarchiques entre les variables.
Il est crucial de prendre en compte les préférences du décideur lors de l'identification du résultat souhaitable sur l'arbre de décision. Certains Data Scientists sont prêts à prendre des risques élevés pour des gains potentiels, tandis que d'autres privilégient les options à faible risque [Emi24].

3.12.3 Prédire avec XGBoost

XGBoost est reconnu comme un modèle extrêmement puissant et polyvalent, ayant démontré son efficacité dans un large éventail d'applications en classification et en régression en machine learning. L'un de ses atouts majeurs réside dans sa capacité à exploiter des caractéristiques temporelles telles que les décalages, les fréquences, les coefficients d'ondelettes et les périodes. En alimentant XGBoost avec ces caractéristiques temporelles, il est capable de fournir des prédictions très précises.
Cependant, malgré ses performances impressionnantes, XGBoost présente une lacune importante dans le contexte des séries temporelles. Pour comprendre cette lacune, il est nécessaire d'examiner les bases mathématiques sous-jacentes à ce modèle. En effet, XGBoost excelle dans l'identification de modèles dans les données, mais il lui manque une caractéristique essentielle pour être considéré comme un modèle de prévision des séries temporelles de qualité.
Dans le contexte des séries temporelles, cette caractéristique manquante est la capacité à prendre en compte la dépendance temporelle entre les observations successives. Contrairement à d'autres modèles spécifiquement conçus pour les séries temporelles, XGBoost ne dispose pas d'un mécanisme intégré pour modéliser cette dépendance temporelle, ce qui peut limiter sa performance dans la prévision des séries chronologiques[Sau20].

3.12.4 Prédire avec KNN

Le k-NN, ou k plus proches voisins, est l'un des algorithmes fondamentaux en apprentissage automatique. À l'origine utilisé pour la classification, il fonctionne en trouvant les k exemples les plus similaires à un exemple non étiqueté parmi un ensemble d'exemples étiquetés. Cette similarité est généralement mesurée par la distance euclidienne entre les vecteurs de caractéristiques. En se basant sur les classes majoritaires parmi ces k voisins les plus proches, l'algorithme prédit la classe de l'exemple non étiqueté.
Le k-NN peut également être adapté à la régression pour des variables cibles numériques. Dans ce cas, la valeur prédite est déterminée par la

moyenne ou la médiane des valeurs cibles des k voisins les plus proches. Cette approche peut également être appliquée à des problèmes de séries temporelles, où les caractéristiques sont définies par des valeurs retardées des observations temporelles. Ainsi, le k-NN peut être utilisé pour prédire des valeurs futures en se basant sur des observations passées, offrant ainsi une méthode simple et intuitive pour la modélisation des séries chronologiques [TWB+21].

3.12.5 Prédire avec Les réseaux de neurones

Les réseaux de neurones, également connus sous le nom de réseaux de neurones artificiels, sont des outils reconnus pour leur puissance dans la prédiction temporelle. Ils constituent un sous-ensemble crucial de l'apprentissage automatique et sont au cœur des algorithmes de deep learning. Inspirés du fonctionnement du cerveau humain, où les neurones biologiques se transmettent des signaux les uns aux autres, les réseaux de neurones artificiels sont conçus pour imiter ce processus.
Un réseau de neurones artificiels se compose de plusieurs couches de neurones, comprenant une couche d'entrée, une ou plusieurs couches cachées, et une couche de sortie. Cette structure permet au réseau d'apprendre des modèles complexes à partir des données en ajustant les poids des connexions entre les neurones.
À titre de comparaison, le cerveau humain est caractérisé par un vaste réseau de neurones divers, estimé à environ 100 milliards. Ces cellules neuronales se distinguent par leur capacité à échanger des impulsions électriques et à communiquer à travers un réseau de fibres nerveuses interconnectées. Chaque neurone est connecté à de nombreux autres neurones et possède la capacité unique de stocker, classer et traiter les informations de manière complexe [FA23].

3.13 conclution

Dans ce chapitre, nous avons exploré les fondamentaux des séries temporelles, en couvrant leurs éléments constitutifs, les méthodes d'analyse, les modèles et les algorithmes de prédiction. Nous avons étudié des concepts clés tels que la stationnarité, les techniques de lissage et l'autocorrélation, dans le but de projeter les valeurs futures en se basant sur les données passées.

CHAPITRE 4

Expérimentations et resultats obtenus

4.1 Introduction

Dans ce chapitre, nous décrirons le corpus utilisé pour nos recherches, les expérimentations menées, ainsi que les résultats obtenus. Notre étude se concentre sur l'optimisation des niveaux de stock dans le commerce de détail en utilisant l'apprentissage automatique et l'analyse de séries temporelles. Nous avons testé cinq algorithmes d'apprentissage automatique pour la prédiction de séries temporelles : les k plus proches voisins (KNN), la régression linéaire (LR), la forêt aléatoire (RF), XGBoost (eXtreme Gradient Boosting) et les réseaux de neurones profonds (DNN). Nous présenterons en détail les résultats obtenus, en soulignant les performances de chaque algorithme.

4.2 Outlies de development:

Python

Python a été choisi comme langage de programmation en raison de sa robustesse et de sa capacité à être un langage multi plateforme (Windows, Unix). De plus, c'est un outil open source. Il est particulièrement large en apprentissage automatique car il fournit Flexibilité de modulation et capacités existantes dépendantes du domaine.

Anaconda

Est une plateforme de distribution Python avec plus de 20 millions d'annonces Utilisateurs mondiaux, basés sur un écosystème entièrement open source. Anaconda inclut Jupyter, qui est essentiellement un IDE (Integrated Development Environment) et un serveur pour exécuter votre ordinateur portable. Jupyter prend actuellement en charge plus de 40 langages informatiques. Ces fameux cahiers sont très prisés par la communauté des data scientists, Contient du code et des éléments de présentation .

4.3 Importation des bibliothèques

Voici l'importation des bibliothèques que nous avons utilisé dans notre travail de recherche :

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import statsmodels.api as sm

from sklearn.preprocessing import MinMaxScaler

import pickle
from os import path
from sklearn import metrics
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from xgboost import XGBRegressor
```

Figure 4.1 : Importation des bibliothèques utilisées première partie

Ce script effectue les étapes suivantes :

1. Charge les ensembles de données.
2. Fusionne les ensembles d'entraînement et de test avec les informations sur les caractéristiques et les magasins.
3. Crée des fonctionnalités supplémentaires liées au temps.
4. Gère les valeurs manquantes en les remplaçant par 0.
5. Encode les variables catégorielles à l'aide de l'encodage one-hot.
6. Définit les fonctionnalités et la variable cible pour l'ensemble d'entraînement.
7. Divise l'ensemble d'entraînement en ensembles d'entraînement et de validation.
8. Initialise et entraîne plusieurs modèles d'apprentissage automatique.
9. Évalue les modèles en utilisant la RMSE.
10. Utilise le meilleur modèle pour faire des prédictions sur l'ensemble de test.

11. Enregistre les prédictions de l'ensemble de test dans un fichier CSV.
12. Enregistre le meilleur modèle dans un fichier pickle pour une utilisation ultérieure.

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from keras.models import Sequential
from keras.layers import Dense
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.base import BaseEstimator
```

Figure 4.2 : Importation des bibliothèques utilisées deuxième partie

Ces importations permettent de mettre en place un flux de travail complet pour la construction, l'entraînement et l'évaluation de modèles d'apprentissage automatique, incluant à la fois des modèles traditionnels comme les forêts aléatoires et des modèles de réseaux de neurones avec Keras.

```
import math
import numpy as np
import pandas as pd
from sklearn.model_selection import StratifiedShuffleSplit
from sklearn.model_selection import train_test_split
from sklearn.base import BaseEstimator, TransformerMixin
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
import category_encoders as ce
from category_encoders.binary import BinaryEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import FeatureUnion
import csv
```

Figure 4.3 : Importation des bibliothèques utilisées troisième partie

Ces importations rassemblent les bibliothèques et outils nécessaires pour le prétraitement, la transformation, et l'encodage des données avant l'entraînement des modèles de machine learning.

4.4 L'ensemble de données (Dataset)

Nous avons utilisé un ensemble de données historiques des ventes de Walmart pour prédire les ventes des magasins .L'ensemble de données (Da-

taset) peut être trouvé en suivant ce lien : http://www.kaggle.com/comprtitions/walmart-recruiting-store-sales-forecasting .Les données contiennent 421 570 lignes, il contient des informations historiques sur les ventes hebdomadaires de 45 magasins Walmart situés dans différentes régions du monde, ainsi que des informations par département pour ces magasins.

4.4.1 Importation de la base de données

```
data = pd.read_csv('train.csv')
stores = pd.read_csv('stores.csv')
features = pd.read_csv('features.csv')
```

Figure 4.4 : Code de lecture du fichier CSV.

4.4.2 Description des fichiers de données

Dataset containing additional data of Stores

```
features.shape
```

```
(8190, 12)
```

```
features.tail()
```

	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment	IsHoliday
8185	45	2013-06-28	76.05	3.639	4842.29	975.03	3.00	2449.97	3169.69	NaN	NaN	False
8186	45	2013-07-05	77.50	3.614	9090.48	2268.58	582.74	5797.47	1514.93	NaN	NaN	False
8187	45	2013-07-12	79.37	3.614	3789.94	1827.31	85.72	744.84	2150.36	NaN	NaN	False
8188	45	2013-07-19	82.84	3.737	2961.49	1047.07	204.19	363.00	1059.46	NaN	NaN	False
8189	45	2013-07-26	76.06	3.804	212.02	851.73	2.06	10.88	1864.57	NaN	NaN	False

Figure 4.5 : Le jeu de données contient des données supplémentaires sur les magasins.

Dataset containing info of Stores

```
stores.shape
```

```
(45, 3)
```

```
stores.tail()
```

	Store	Type	Size
40	41	A	196321
41	42	C	39690
42	43	C	41062
43	44	C	39910
44	45	B	118221

Figure 4.6 : Le jeu de données contient les informations sur les magasins.

4.5 Gestion les valeurs manquantes dans l'ensemble de données des fonctionnalités

La gestion des valeurs manquantes dans un ensemble de données de fonctionnalités peut être cruciale pour garantir des résultats précis dans notre modèle .

```
features["CPI"].fillna(features["CPI"].median(),inplace=True)
features["Unemployment"].fillna(features["Unemployment"].median(),inplace=True)
```

```
for i in range(1,6):
  features["MarkDown"+str(i)] = features["MarkDown"+str(i)].apply(lambda x: 0 if x < 0 else x)
  features["MarkDown"+str(i)].fillna(value=0,inplace=True)
```

Figure 4.7 : code de gestion les valeurs manquantes .

Ce code itère sur les colonnes "MarkDown1" à "MarkDown5" dans le DataFrame "features". Pour chaque colonne, il remplace les valeurs négatives par 0 à l'aide de la fonction lambda dans la méthode apply(). Ensuite, il remplace les valeurs manquantes par 0 à l'aide de la méthode fillna() avec l'argument value=0 et inplace=True.

4.5.1 Résultat

```
features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8190 entries, 0 to 8189
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Store         8190 non-null   int64
 1   Date          8190 non-null   object
 2   Temperature   8190 non-null   float64
 3   Fuel_Price    8190 non-null   float64
 4   MarkDown1     8190 non-null   float64
 5   MarkDown2     8190 non-null   float64
 6   MarkDown3     8190 non-null   float64
 7   MarkDown4     8190 non-null   float64
 8   MarkDown5     8190 non-null   float64
 9   CPI           8190 non-null   float64
 10  Unemployment  8190 non-null   float64
 11  IsHoliday     8190 non-null   bool
dtypes: bool(1), float64(9), int64(1), object(1)
memory usage: 712.0+ KB
```

Figure 4.8 : Résultat de gestion des valeurs manquantes .

4.6 Fusion de l'ensemble de données data avec l'ensemble de données stores

Merging Training Dataset and merged stores-features Dataset

```
data = pd.merge(data,stores,on='Store',how='left')
```

```
data = pd.merge(data,features,on=['Store','Date'],how='left')
```

```
data['Date'] = pd.to_datetime(data['Date'])
```

```
data.sort_values(by=['Date'],inplace=True)
```

```
data.set_index(data.Date, inplace=True)
```

```
data['IsHoliday_x'].isin(data['IsHoliday_y']).all()
```

```
True
```

```
data.drop(columns='IsHoliday_x',inplace=True)
data.rename(columns={"IsHoliday_y" : "IsHoliday"}, inplace=True)
data.info()
```

Figure 4.9 : Code de la fusion et le suppression de l'ensemble de données data avec l'ensemble de données stores.

Ce code effectue les opérations suivantes :

1. Fusion de l'ensemble de données 'data' avec l'ensemble de données 'stores' sur la colonne 'Store', en utilisant une jointure à gauche ('how='left''). Cela ajoute les informations sur les magasins à l'ensemble de données 'data'.

2. Fusion de l'ensemble de données 'data' avec l'ensemble de données 'features' sur les colonnes 'Store' et 'Date', en utilisant une jointure à gauche ('how='left''). Cela ajoute les informations sur les caractéristiques à l'ensemble de données 'data'.

3. Conversion de la colonne 'Date' de l'ensemble de données 'data' en format de date et tri des données par date.

4. Définition de la colonne 'Date' comme index de l'ensemble de données 'data'.

5. Vérification si les valeurs de la colonne 'IsHoliday-x' de l'ensemble de données 'data' sont toutes présentes dans la colonne 'IsHoliday-y'. Le résultat est 'True', ce qui signifie que toutes les valeurs de la colonne 'IsHoliday-x' sont également présentes dans la colonne 'IsHoliday-y'.

6. Suppression de la colonne 'IsHoliday-x' de l'ensemble de données 'data' à l'aide de la méthode 'drop()'.

7. Renommage de la colonne 'IsHoliday-y' en 'IsHoliday' à l'aide de la méthode 'rename()'.

8. Affichage des informations sur l'ensemble de données 'data' à l'aide de la méthode 'info()'.

Cela garantit que les données sont prêtes pour l'analyse ou la modélisation ultérieure.

4.6.1 Résultat

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 421570 entries, 2010-02-05 to 2012-10-26
Data columns (total 16 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   Store         421570 non-null  int64
 1   Dept          421570 non-null  int64
 2   Date          421570 non-null  datetime64[ns]
 3   Weekly_Sales  421570 non-null  float64
 4   Type          421570 non-null  object
 5   Size          421570 non-null  int64
 6   Temperature   421570 non-null  float64
 7   Fuel_Price    421570 non-null  float64
 8   MarkDown1     421570 non-null  float64
 9   MarkDown2     421570 non-null  float64
 10  MarkDown3     421570 non-null  float64
 11  MarkDown4     421570 non-null  float64
 12  MarkDown5     421570 non-null  float64
 13  CPI           421570 non-null  float64
 14  Unemployment  421570 non-null  float64
 15  IsHoliday     421570 non-null  bool
dtypes: bool(1), datetime64[ns](1), float64(10), int64(3), object(1)
memory usage: 51.9+ MB
```

Figure 4.10 : Résultat suppression de la colonne 'IsHoliday-x'.

4.7 La détection des valeurs aberrantes

```
agg_data = data.groupby(['Store', 'Dept']).Weekly_Sales.agg(['max', 'min', 'mean', 'median', 'std']).reset_index()
agg_data.isnull().sum()
```

Figure 4.11 : Code de la détection des valeurs aberrantes.

Ce code calcule les statistiques agrégées (maximum , minimum , moyenne , médiane et écart-type) des ventes hebdomadaires ('Weekly-Sales') pour chaque combinaison de magasin ('Store') et département ('Dept').

4.7.1 Résultat

Store	0
Dept	0
Max	0
Min	0
Mean	0
Median	0
std	37
dtype:	int64

Table 4.1 : Résultat de la détection des valeurs aberrantes.

4.8 Fusion des données et suppression des valeurs manquantes

```
store_data = pd.merge(left=data,right=agg_data,on=['Store', 'Dept'],how ='left')
store_data.dropna(inplace=True)
data = store_data.copy()
del store_data
```

```
data['Date'] = pd.to_datetime(data['Date'])
data.sort_values(by=['Date'],inplace=True)
data.set_index(data.Date, inplace=True)
data.head()
```

Figure 4.12 : Code de La fusion des données et de la suppression des valeurs manquantes.

Ce code effectue plusieurs opérations pour fusionner les données, supprimer les valeurs manquantes et préparer l'ensemble de données pour une analyse ultérieure.

4.8.1 Résultat

Date	Store	Dept	Date	Weekly_Sales	Type	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	...	CPI	Unemployment	IsHoliday	Year	Mo
2010-02-05	1	1	2010-02-05	24924.50	A	151315	42.31	2.572	0.0	0.0	...	211.096358	8.106	False	2010	
2010-02-05	9	97	2010-02-05	668.48	B	125833	38.01	2.572	0.0	0.0	...	214.655459	6.415	False	2010	
2010-02-05	9	85	2010-02-05	693.87	B	125833	38.01	2.572	0.0	0.0	...	214.655459	6.415	False	2010	
2010-02-05	8	80	2010-02-05	8654.60	A	155078	34.14	2.572	0.0	0.0	...	214.471451	6.299	False	2010	
2010-02-05	9	55	2010-02-05	11123.56	B	125833	38.01	2.572	0.0	0.0	...	214.655459	6.415	False	2010	

5 rows × 23 columns

Figure 4.13 : Résultat de la Fusion des données et la suppression des valeurs manquantes.

4.9 la forme du DataFrame avant et après la suppression des valeurs aberrantes.

```
data['Total_MarkDown'] = data['MarkDown1']+data['MarkDown2']+data['MarkDown3']+data['MarkDown4']+data['MarkDown5']
data.drop(['MarkDown1','MarkDown2','MarkDown3','MarkDown4','MarkDown5'], axis = 1,inplace=True)
```

```
numeric_col = ['Weekly_Sales','Size','Temperature','Fuel_Price','CPI','Unemployment','Total_MarkDown']
data_numeric = data[numeric_col].copy()
```

```
data.shape
```

```
(421533, 19)
```

```
data = data[(np.abs(stats.zscore(data_numeric)) < 2.5).all(axis = 1)]
data.shape
```

```
(375438, 19)
```

Figure 4.14 : Code de la forme du DataFrame avant et après la suppression des valeurs aberrantes.

Le code commence par créer une nouvelle colonne 'Total-MarkDown' en additionnant plusieurs colonnes existantes et supprime ensuite ces colonnes originales. Ensuite, il sélectionne un sous-ensemble de colonnes numériques et utilise le score Z pour filtrer les valeurs aberrantes. Finalement, il affiche la forme du DataFrame avant et après la suppression des valeurs aberrantes.

4.10 Les ventes hebdomadaires négatives (Negative Weekly Sales)

```
y = data["Weekly_Sales"][ data.Weekly_Sales < 0]
sns.displot(y,height=6,aspect=2)
plt.title("Negative Weekly Sales", fontsize=5)

plt.show()
```

Figure 4.15 : Code de ventes hebdomadaires négatives .

Le code sélectionne toutes les ventes hebdomadaires négatives de DataFrame 'data', puis crée et affiche un graphique de distribution pour ces valeurs négatives. Cela permet de visualiser comment les ventes négatives sont réparties, ce qui peut être utile pour identifier des anomalies ou des problèmes dans les données.

4.10.1 Résultat

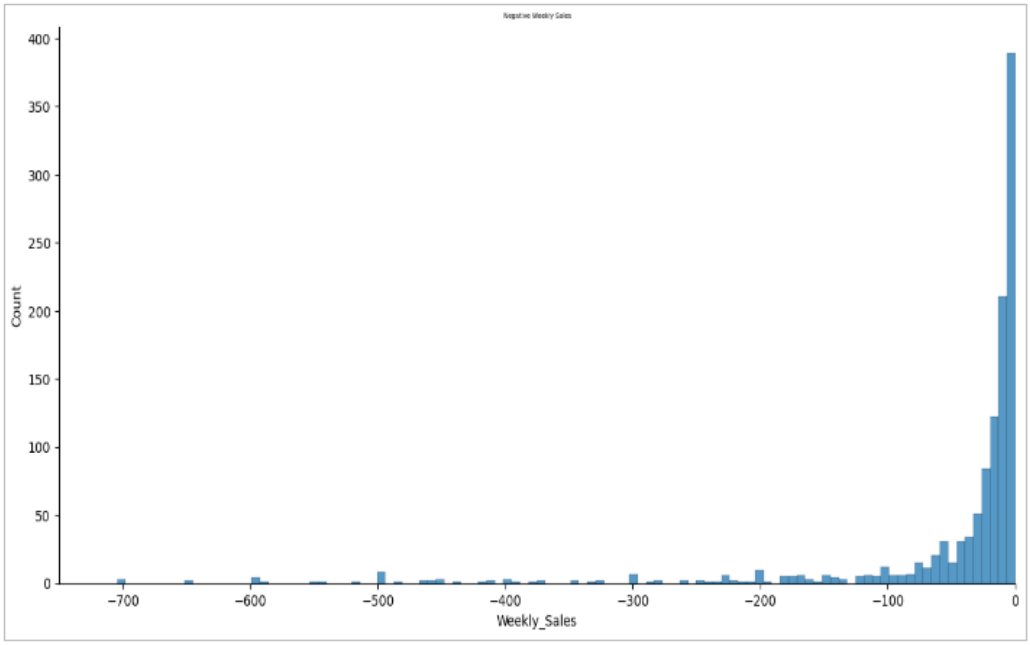

Figure 4.16 : Résultat de ventes hebdomadaires négatives.

4.11 Les ventes hebdomadaires

```
data=data[data['Weekly_Sales']>=0]
```

```
data.shape
```

```
(374247, 19)
```

```
data['IsHoliday'] = data['IsHoliday'].astype('int')
```

```
data
```

Figure 4.17 : Code de ventes hebdomadaires .

Le code commence par supprimer les ventes hebdomadaires négatives du DataFrame ‘data‘. Ensuite, il convertit la colonne ‘IsHoliday‘ en type entier. Finalement, il affiche le DataFrame mis à jour pour vérifier les modifications. Cela garantit que le DataFrame ne contient que des ventes hebdomadaires positives et que la colonne ‘IsHoliday‘ est de type entier, ce qui peut être plus approprié pour certaines analyses ou algorithmes de machine learning.

4.11.1 Résultat

Date	Store	Dept	Date	Weekly_Sales	Type	Size	Temperature	Fuel_Price	CPI	Unemployment	IsHoliday	Year	Month	max	min	
2010-02-05	1	1	2010-02-05	24924.50	A	151315	42.31	2.572	211.096358	8.106	0	2010	2	57592.12	14537.37	2251
2010-02-05	9	97	2010-02-05	668.48	B	125833	38.01	2.572	214.655459	6.415	0	2010	2	766.93	-9.92	37
2010-02-05	9	85	2010-02-05	693.87	B	125833	38.01	2.572	214.655459	6.415	0	2010	2	2512.14	110.56	87
2010-02-05	8	80	2010-02-05	8654.60	A	155078	34.14	2.572	214.471451	6.299	0	2010	2	11990.43	7414.43	918
2010-02-05	9	55	2010-02-05	11123.56	B	125833	38.01	2.572	214.655459	6.415	0	2010	2	29166.26	4791.74	860
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
2012-10-26	2	26	2012-10-26	9258.88	A	202307	69.79	3.506	223.078337	6.170	0	2012	10	19679.04	4179.99	934
2012-10-26	38	23	2012-10-26	53.12	C	39690	65.95	4.301	131.193097	10.199	0	2012	10	289.78	3.50	3
2012-10-26	27	6	2012-10-26	5339.65	A	204184	58.99	4.071	142.762411	8.000	0	2012	10	41005.33	2524.01	773
2012-10-26	36	40	2012-10-26	10216.27	A	39910	74.39	3.494	222.113657	6.228	0	2012	10	13605.53	8444.83	1054
2012-10-26	45	98	2012-10-26	1076.80	B	118221	58.85	3.882	192.308899	8.667	0	2012	10	1504.71	2.00	56

374247 rows × 19 columns

Figure 4.18 : Résultat de ventes hebdomadaires .

4.12 Enregistre le DataFrame

```
data.to_csv('preprocessed_walmart_dataset.csv')
```

Figure 4.19 : Le code qui enregistre le DataFrame prétraité .

Le code enregistre le DataFrame prétraité ‘data‘ dans un fichier CSV nommé ‘preprocessed-walmart-dataset.csv‘. Cela permet de sauvegarder les données prétraitées pour une utilisation ultérieure, par exemple pour des analyses supplémentaires ou pour l’entraînement de modèles de machine learning. Si vous souhaitez personnaliser davantage l’exportation, vous pouvez utiliser des options supplémentaires fournies par la méthode ‘to-csv‘.

4.13 Les visualisations de données

```
plt.figure(figsize=(14,8))
sns.barplot(x='Month',y='Weekly_Sales',data=data)
plt.ylabel('Sales',fontsize=14)
plt.xlabel('Months',fontsize=14)
plt.title('Average Monthly Sales',fontsize=16)

plt.grid()
```

Figure 4.20 : Code de Ventes Moyennes Mensuelles .

Ce code permet de visualiser clairement les ventes moyennes pour chaque mois de l'année, en offrant une représentation graphique utile pour l'analyse des tendances saisonnières.

4.13.1 Résultat

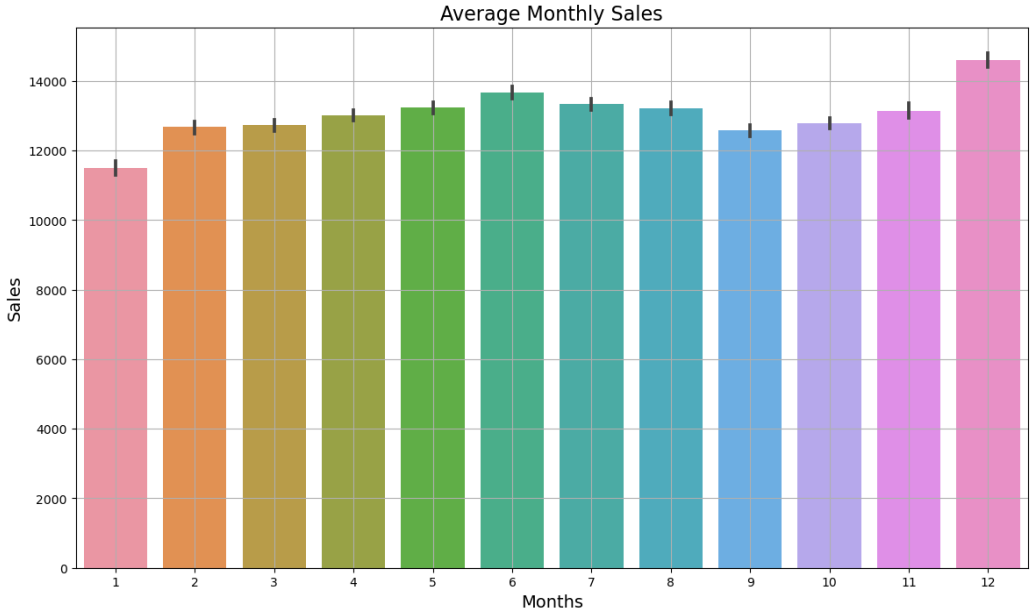

Figure 4.21 : Résultat de Ventes Moyennes Mensuelles .

4.14 Charger des données depuis un fichier CSV

```
def importData(feature = None):
    data = pd.read_csv("train.csv")
    train = test = 0
    if feature is None:
        train, test = train_test_split(data, test_size = 0.2, random_state = 21)
    else:
        split = StratifiedShuffleSplit(n_splits=1, test_size = 0.2, random_state = 21)
        for train_index, test_index in split.split(data, data[feature]):
            train = data.loc[train_index]
            test = data.loc[test_index]
    return [train,test]
```

```
train, test = importData("Date")
```

```
train.head(10)
```

Figure 4.22 : Code qui charger des données depuis train.csv .

Ce code assure que les données sont divisées correctement, en prenant en compte les caractéristiques spécifiées, et évite les erreurs liées à des colonnes inexistantes.

4.14.1 Résultat

	Store	Dept	Date	Weekly_Sales	IsHoliday
359179	38	81	2012-01-20	10766.50	False
19344	2	91	2011-01-07	82980.32	False
268367	28	10	2012-01-06	14014.90	False
366368	39	41	2012-04-27	365.00	False
392556	42	12	2011-04-22	36.94	False
393058	42	16	2012-09-14	799.70	False
134014	14	49	2011-01-21	12890.41	False
261783	27	37	2010-08-27	2923.50	False
252321	26	42	2010-04-23	2439.76	False
251827	26	36	2011-10-21	1115.30	False

Figure 4.23 : Résultat de chargement des données depuis train.csv .

4.15 Nombre Total de Magasins

```
print(train['Store'].unique())
print('Total number of stores = {}'.format(len(train['Store'].unique())))
```

```
[38  2 28 39 42 14 27 26 17 22 11 21 10 40  6 34 33  5 29  4 20 13 31  1
 41 19 23 30 37 32 25 16 24 43 12 15  9  8  3 36 35 18 45 44  7]
Total number of stores = 45
```

Figure 4.24 : Nombre Total de Magasins.

4.16 Nombre Total de Départements

```
print(train['Store'].unique())
print('Total number of stores = {}'.format(len(train['Store'].unique())))
```

```
[38  2 28 39 42 14 27 26 17 22 11 21 10 40  6 34 33  5 29  4 20 13 31  1
 41 19 23 30 37 32 25 16 24 43 12 15  9  8  3 36 35 18 45 44  7]
Total number of stores = 45
```

Figure 4.25 : Nombre Total de Départements .

4.17 Dates des Semaines de Collecte des Données

```
print('Dates of the weeks on which data was recorded :- ')
print(train['Date'].unique())
print('Total number of weeks = {}'.format(len(train['Date'].unique())))
year12 = []
year11 = []
year10 = []
date_list = train['Date'].unique()
for i in range(len(train['Date'].unique())):
    l = list(map(int, date_list[i].split('-')))
    if l[0] == 2012:
        year12.append(l[0])
    elif l[0] == 2011:
        year11.append(l[0])
    else:
        year10.append(l[0])
print('Number of Entries in Year 2010 = {}'.format(len(year10)))
print('Number of Entries in Year 2011 = {}'.format(len(year11)))
print('Number of Entries in Year 2012 = {}'.format(len(year12)))
```

Figure 4.26 : Dates des Semaines de Collecte des Données .

Ce code utilise efficacement les fonctionnalités pandas pour manipuler les dates et compter les occurrences, ce qui simplifie et optimise le processus.

4.17.1 Résultat

```
Dates of the weeks on which data was recorded :-
['2012-01-20' '2011-01-07' '2012-01-06' '2012-04-27' '2011-04-22'
 '2012-09-14' '2011-01-21' '2010-08-27' '2010-04-23' '2011-10-21'
 '2011-12-09' '2011-12-16' '2011-07-08' '2010-06-18' '2010-10-15'
 '2012-06-08' '2010-03-12' '2011-09-23' '2010-12-24' '2011-03-18'
 '2011-06-03' '2010-02-12' '2010-05-14' '2011-04-08' '2012-05-25'
 '2010-07-16' '2012-03-16' '2010-04-30' '2010-04-16' '2010-12-31'
 '2011-08-05' '2012-02-03' '2010-08-06' '2012-06-01' '2011-02-04'
 '2011-05-13' '2011-07-15' '2010-04-09' '2011-02-11' '2011-03-04'
 '2011-04-15' '2011-06-10' '2012-07-13' '2010-04-02' '2010-12-17'
 '2012-04-20' '2011-08-19' '2012-03-30' '2010-03-26' '2011-02-25'
 '2011-08-12' '2011-12-30' '2010-02-26' '2011-12-23' '2010-06-25'
 '2010-09-24' '2010-09-03' '2012-05-18' '2010-03-19' '2012-07-27'
 '2011-07-29' '2012-10-05' '2011-02-18' '2012-01-13' '2010-07-30'
 '2010-02-19' '2011-03-25' '2012-01-27' '2010-08-20' '2010-06-04'
 '2010-07-02' '2011-09-09' '2012-08-10' '2011-04-29' '2012-02-10'
 '2010-02-05' '2012-02-17' '2012-09-28' '2010-11-26' '2010-10-08'
 '2010-08-13' '2011-08-26' '2010-09-17' '2011-11-18' '2011-10-14'
 '2010-10-29' '2011-09-02' '2012-04-06' '2012-08-31' '2012-02-24'
 '2012-03-09' '2012-10-12' '2011-01-28' '2011-10-07' '2011-09-30'
 '2011-05-27' '2012-08-03' '2011-06-17' '2011-04-01' '2012-06-15'
 '2011-11-04' '2012-04-13' '2010-06-11' '2011-11-11' '2011-09-16'
 '2010-03-05' '2010-11-19' '2010-12-10' '2012-09-21' '2012-07-06'
 '2011-07-01' '2012-03-23' '2011-05-06' '2011-11-25' '2010-05-28'
 '2011-07-22' '2010-10-01' '2011-05-20' '2011-03-11' '2010-10-22'
 '2011-12-02' '2011-06-24' '2010-11-12' '2012-06-22' '2010-12-03'
 '2011-01-14' '2010-05-21' '2010-11-05' '2012-07-20' '2012-08-24'
 '2010-07-23' '2012-05-04' '2012-05-11' '2010-05-07' '2012-10-26'
 '2012-06-29' '2012-10-19' '2012-08-17' '2011-10-28' '2012-03-02'
 '2012-09-07' '2010-09-10' '2010-07-09']
Total number of weeks = 143
Number of Entries in Year 2010 = 48
Number of Entries in Year 2011 = 52
Number of Entries in Year 2012 = 43
```

Figure 4.27 : Résultat de dates des Semaines de Collecte des Données .

4.18 Remplacer les dates par le numéro

```
class replaceDateWithWeek(BaseEstimator, TransformerMixin):
    def __init__(self,arg=None):
        self.arg = arg
    def fit(self,X,y=None):
        return self
    def transform(self,data,y=None):
        #Finding the dates and converting them to int
        dates_list = []
        dates = data['Date'].unique()
        print("Number of weeks in the dataset = {}".format(len(dates)))
        for i in range(len(dates)):
            l = list(map(int, dates[i].split('-')))
            dates_list.append(l)

        #Sorting the dates
        weeks = []
        for i in range(2010,2013):
            for j in range(1,13):
                for k in range(1,32):
                    for date in dates_list:
                        if date[0] == i and date[1] == j and date[2] == k:
                            weeks.append(date)

        #Reconverting the dates back to string
        for i in range(len(weeks)):
            if weeks[i][1] >= 10 and weeks[i][2] >= 10:
                weeks[i] = str(weeks[i][0])+"-"+str(weeks[i][1])+"-"+str(weeks[i][2])
            elif weeks[i][1] >= 10 and weeks[i][2] < 10:
                weeks[i] = str(weeks[i][0])+"-"+str(weeks[i][1])+"-0"+str(weeks[i][2])
            elif weeks[i][1] < 10 and weeks[i][2] >= 10:
                weeks[i] = str(weeks[i][0])+"-0"+str(weeks[i][1])+"-"+str(weeks[i][2])
            elif weeks[i][1] < 10 and weeks[i][2] < 10:
                weeks[i] = str(weeks[i][0])+"-0"+str(weeks[i][1])+"-0"+str(weeks[i][2])

        #Replacing dates with week number
        week_num = []
        l = data['Date'].tolist()
        for i in range(len(l)):
            week_num.append(weeks.index(l[i]) + 1)

        data['Week'] = week_num
        data.drop(['Date'], axis = 1, inplace = True)
        return data

temp = replaceDateWithWeek()
temp.transform(train)
train.head(10)
```

Figure 4.28 : Code qui remplacer les dates par le numéro .

Cette classe personnalisée, 'replaceDateWithWeek', est conçue pour remplacer les dates par le numéro de semaine correspondant.

4.18.1 Résultat

```
Number of weeks in the dataset = 143
```

	Store	Dept	Weekly_Sales	IsHoliday	Week
359179	38	81	10766.50	False	103
19344	2	91	82980.32	False	49
268367	28	10	14014.90	False	101
366368	39	41	365.00	False	117
392556	42	12	36.94	False	64
393058	42	16	799.70	False	137
134014	14	49	12890.41	False	51
261783	27	37	2923.50	False	30
252321	26	42	2439.76	False	12
251827	26	36	1115.30	False	90

Figure 4.29 : Résultat de remplacement les dates par le numéro .

4.19 Différence des Ventes Moyennes pendant les Semaines de Vacances

```
y = []
x = ['Holiday', 'Normal']

y.append(sum(train[train['IsHoliday'] == True]['Weekly_Sales'])/len(train[train['IsHoliday'] == True]['Weekly_Sales']))
y.append(sum(train[train['IsHoliday'] == False]['Weekly_Sales'])/len(train[train['IsHoliday'] == False]['Weekly_Sales']))

plt.figure(figsize = (10,8))
plt.title('Difference in average sales on Holiday Weeks')
sns.barplot(x = x, y = y)
sns.set(style = 'darkgrid')
plt.xlabel("Type of week")
plt.ylabel("Average Sales (In $)")
plt.show()
```

Figure 4.30 : Code qui comparer les ventes hebdomadaires .

Ce code calcule la moyenne des ventes hebdomadaires pour les semaines de vacances et les semaines normales, puis trace un graphique à barres pour visualiser la différence dans les ventes moyennes entre ces deux types de semaines.

4.19.1 Résultat

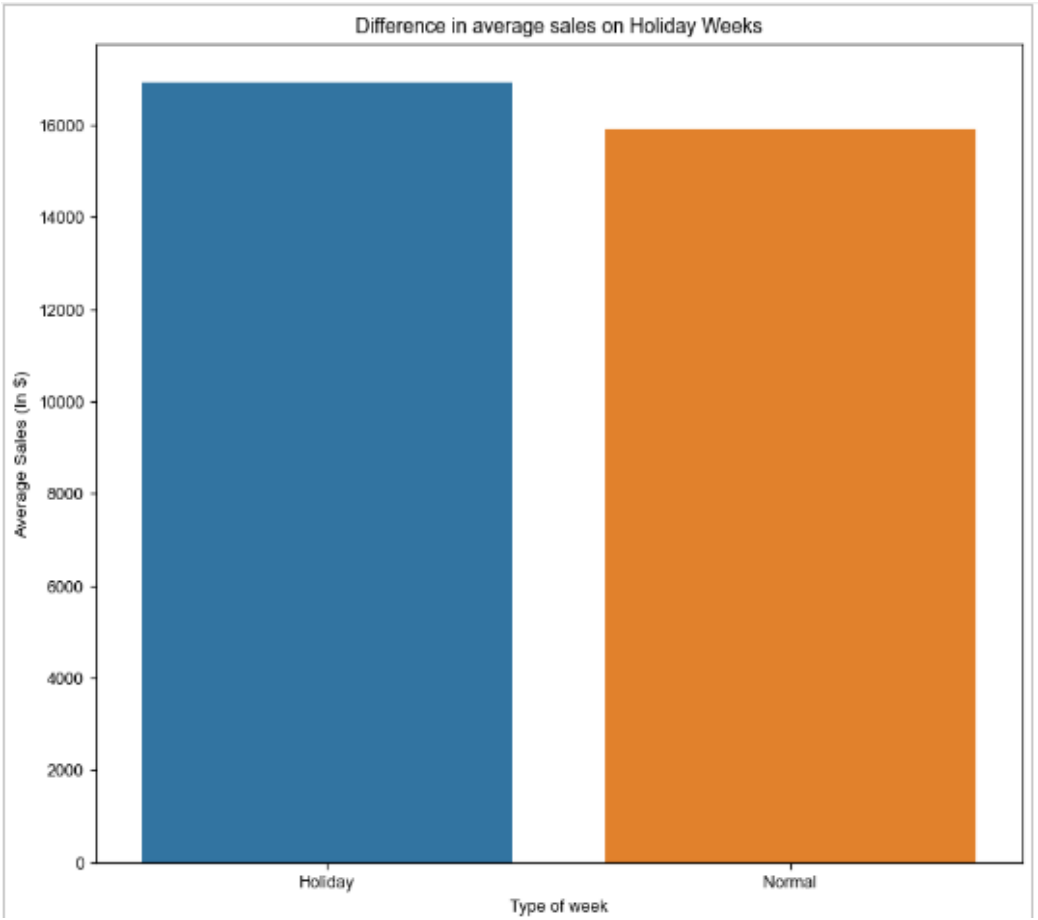

Figure 4.31 : Résultat de comparaision les ventes hebdomadaires .

4.20 Ventes Mensuelles pour Chaque Année

```
data_monthly = pd.crosstab(data["Year"], data["Month"], values=data["Weekly_Sales"],aggfunc='sum')
data_monthly
```

Month	1	2	3	4	5	6	7	8	9	10
Year										
2010	NaN	1.365986e+08	1.369976e+08	1.753251e+08	1.422677e+08	1.455770e+08	1.776954e+08	1.426176e+08	1.344151e+08	1.687452e+08
2011	1.170809e+08	1.320987e+08	1.341586e+08	1.710181e+08	1.382097e+08	1.441182e+08	1.762046e+08	1.398143e+08	1.668033e+08	1.375280e+08
2012	1.172222e+08	8.915290e+07	1.734250e+08	1.479222e+08	1.512280e+08	1.902330e+08	1.458221e+08	1.636039e+08	1.432228e+08	1.480537e+08

Figure 4.32 : Ventes Mensuelles pour Chaque Année .

Figure 4.33 : Les graphes de ventes Mensuelles pour Chaque Année .

4.21 Ventes Moyennes Hebdomadaires par Magasin

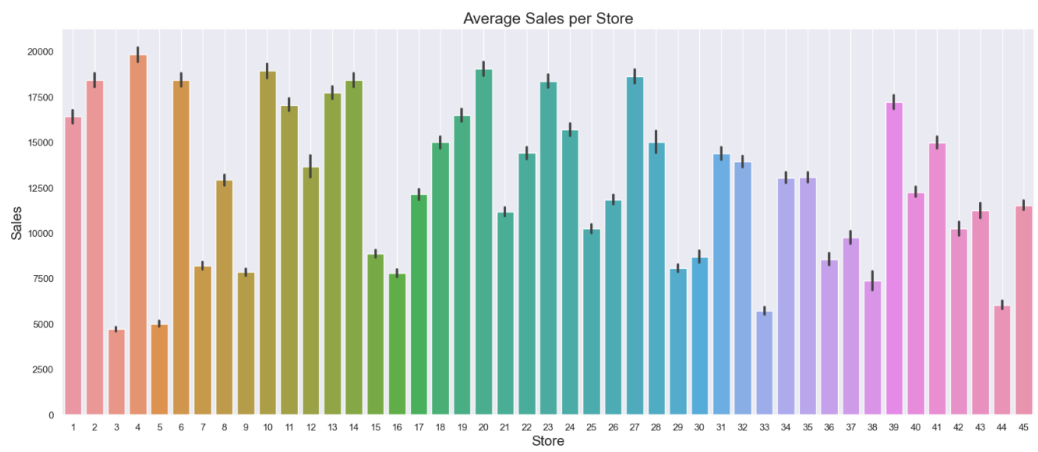

Figure 4.34 : Ventes Moyennes Hebdomadaires par Magasin .

4.22 Ventes Moyennes par Département

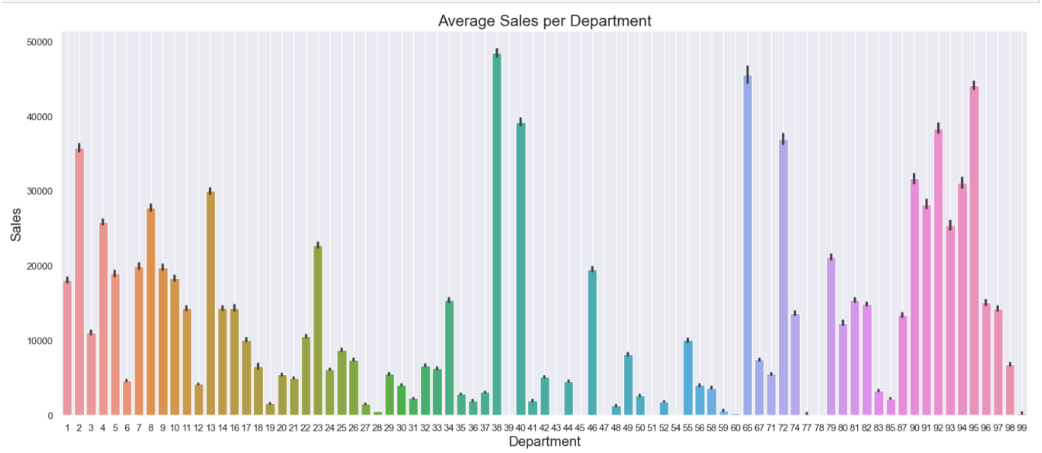

Figure 4.35 : Ventes Moyennes par Département .

4.23 Ventes vs Température

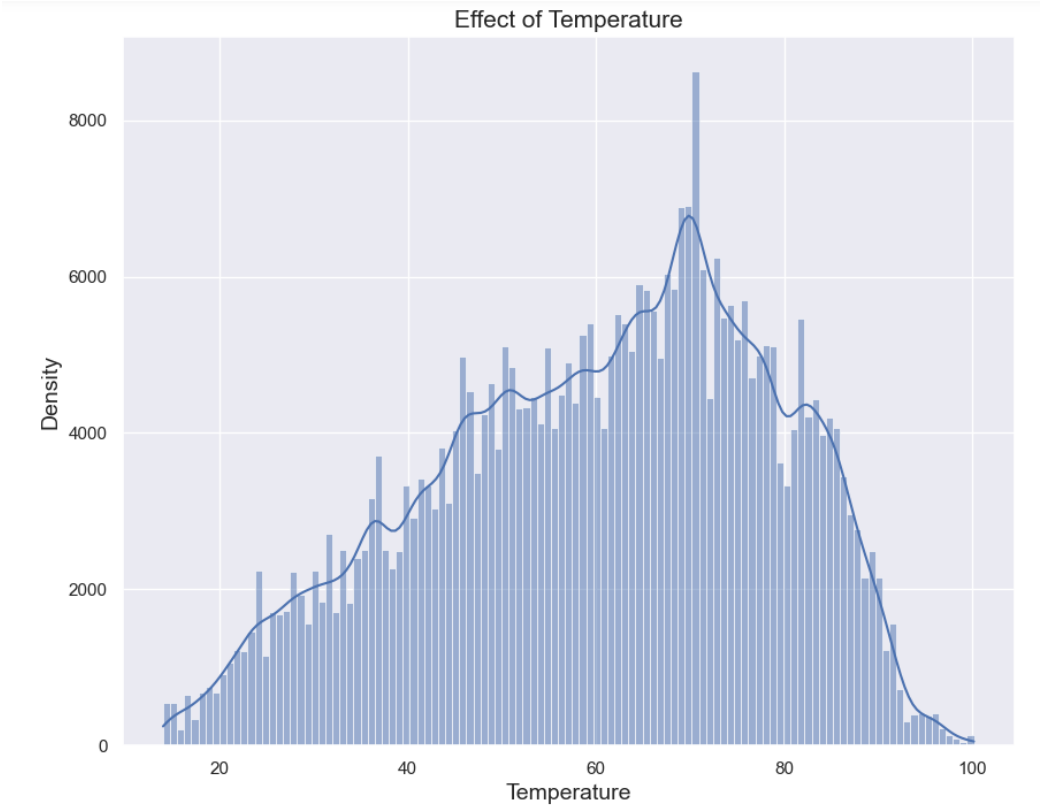

Figure 4.36 : Ventes vs Température .

Ce graphique permet d'observer la distribution de la variable 'Temperature' dans les données et de visualiser la densité de probabilité estimée.

4.24 La distribution des vacances

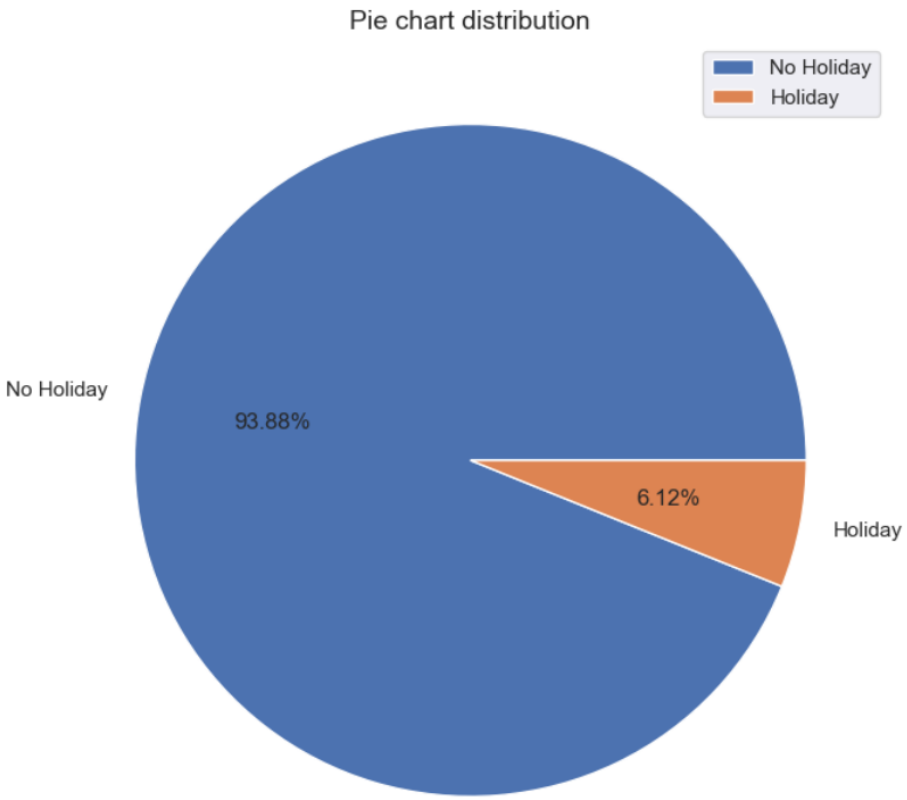

Figure 4.37 : La distribution des vacances .

Ce graphique en secteurs permet de visualiser rapidement la répartition des valeurs de la variable 'IsHoliday' et de déterminer la proportion de semaines de vacances par rapport aux semaines normales.

4.25 La décomposition de la séries temporelle

La décomposition saisonnière permet de visualiser les composantes saisonnières, de tendance et résiduelle de la série temporelle, ce qui peut aider à mieux comprendre les motifs et les tendances dans les ventes hebdomadaires.

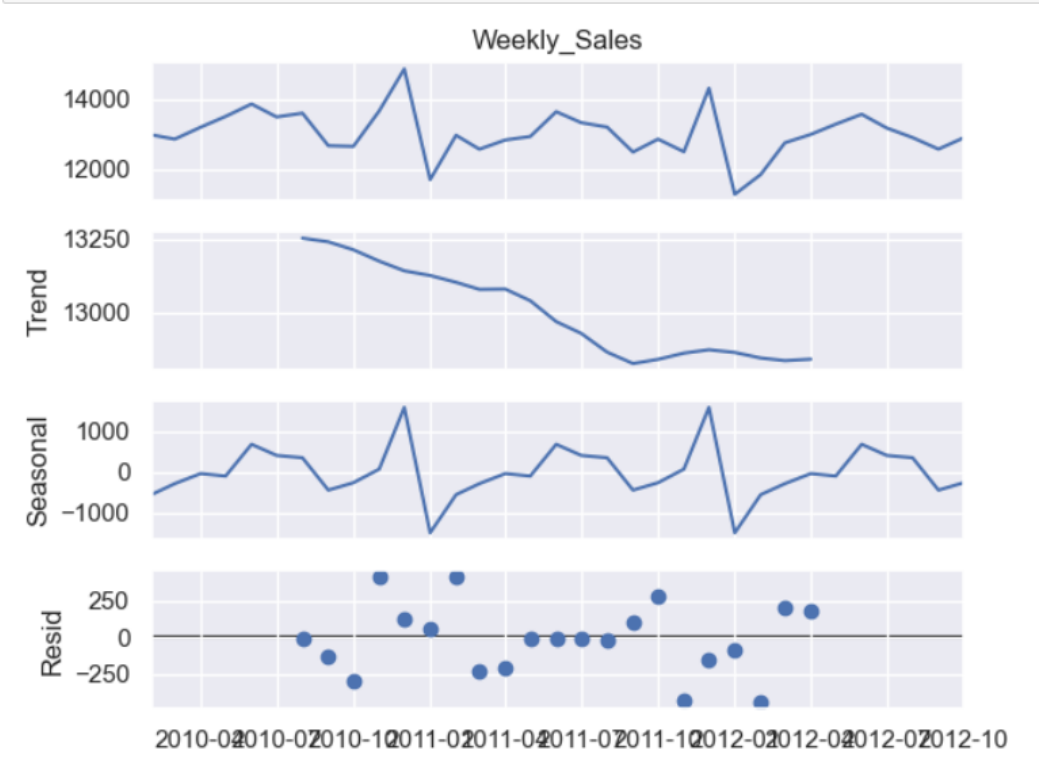

Figure 4.38 : La décomposition de la séries temporelle .

4.26 Encodage One-hot plus Normalisation des Données

L'encodage one-hot et la normalisation des données sont deux techniques couramment utilisées en science des données et en apprentissage automatique pour préparer les données avant de les utiliser dans des modèles.

4.27 La corrélation entre les caractéristiques (features)

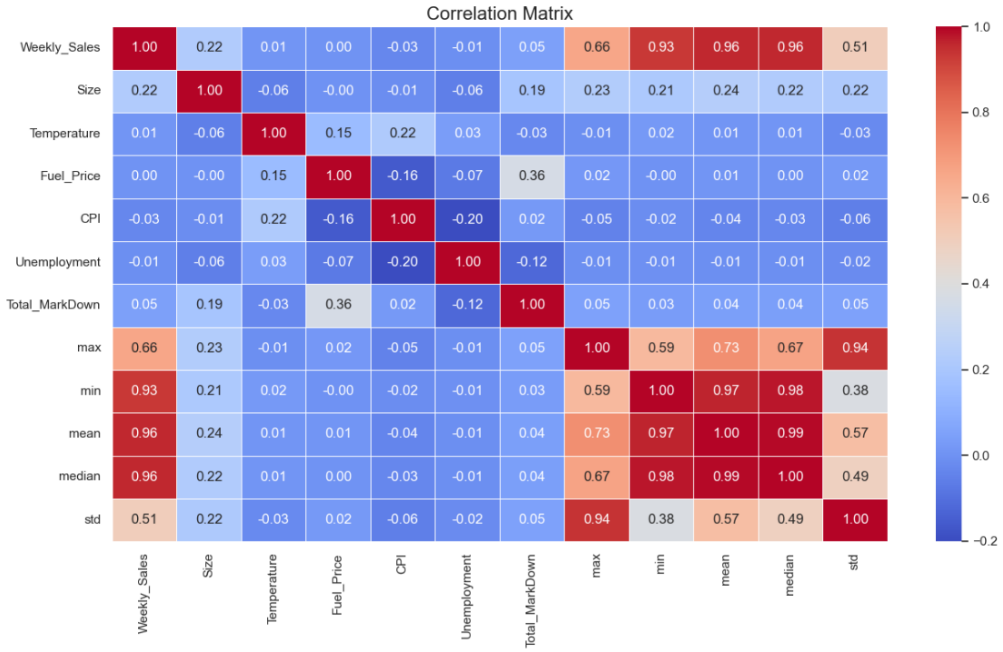

Figure 4.39 : Matrice de corrélation .

Ce heatmap de corrélation permet de visualiser les relations entre les caractéristiques numériques du jeu de données, en mettant en évidence les paires de caractéristiques qui sont fortement corrélées. Les valeurs de corrélation proches de 1 indiquent une forte corrélation positive, tandis que celles proches de -1 indiquent une forte corrélation négative. Les valeurs proches de 0 indiquent une faible corrélation .

4.28 Recursive Feature Elimination (RFE)

```
feature_col = data.columns.difference(['Weekly_Sales'])
feature_col
```

Figure 4.40 : Code de recursive feature elimination .

La méthode Recursive Feature Elimination (RFE) est une technique de sélection de caractéristiques qui fonctionne en récursivement éliminant les caractéristiques les moins importantes et en construisant le modèle avec les caractéristiques restantes jusqu'à ce que le nombre de caractéristiques désiré soit attein .

```
'''
param_grid={'n_estimators':np.arange(10,25)}
tree=GridSearchCV(RandomForestRegressor(oob_score=False,warm_start=True),param_grid,cv=5)
tree.fit(data_train[feature_col],data_train['Weekly_Sales'])
'''
```

```
"\nparam_grid={'n_estimators':np.arange(10,25)}\ntree=GridSearchCV(RandomForestRegressor(oob_score=False,warm_start=True),param
_grid,cv=5)\ntree.fit(data_train[feature_col],data_train['Weekly_Sales'])\n"
```

Figure 4.41 : Code qui recherche le meilleur nombre d'estimateurs .

Ce code recherche le meilleur nombre d'estimateurs (nombre d'arbres) pour le modèle de forêt aléatoire en utilisant une recherche par grille avec une validation croisée à 5 plis. Une fois que la recherche est terminée, le modèle avec les meilleurs hyperparamètres peut être utilisé pour faire des prédictions sur de nouvelles données.

4.29 modèle de régression de forêt aléatoire avec les paramètres spécifiés

```
radm_clf = RandomForestRegressor(oob_score=True,n_estimators=23)
radm_clf.fit(data[feature_col], data['Weekly_Sales'])
```

Figure 4.42 : Code de régression de forêt aléatoire avec les paramètres spécifiés .

4.29.1 Paramètre

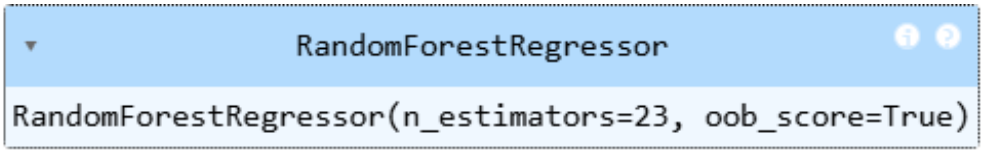

Figure 4.43 : Résultat de régression de forêt aléatoire avec les paramètres spécifiés .

Le modèle de forêt aléatoire est entraîné et prêt à être utilisé pour faire des prédictions sur de nouvelles données.

4.30 L'importance de caractéristiques dans le modèle de forêt aléatoire

```
indices = np.argsort(radm_clf.feature_importances_)[::-1]
feature_rank = pd.DataFrame(columns = ['rank', 'feature', 'importance'])

for f in range(data[feature_col].shape[1]):
    feature_rank.loc[f] = [f+1,
                           data[feature_col].columns[indices[f]],
                           radm_clf.feature_importances_[indices[f]]]

feature_rank
```

Figure 4.44 : Code qui classe les caractéristiques .

Ce code classe les caractéristiques (features) en fonction de leur importance dans le modèle de forêt aléatoire entraîné.

4.30.1 Résultat

	rank	feature	importance
0	1	median	5.244362e-01
1	2	mean	4.033255e-01
2	3	Month	1.369417e-02
3	4	Temperature	1.193349e-02
4	5	CPI	7.799685e-03
...	...	...	...
138	139	Dept_51	2.716471e-10
139	140	Dept_45	2.144224e-10
140	141	Dept_78	4.268046e-12
141	142	Dept_39	7.307891e-15
142	143	Dept_43	3.019127e-15

143 rows × 3 columns

Figure 4.45 : Résultat de classe les caractéristiques .

4.31 Caractéristiques les plus importantes classées par le modèle de forêt aléatoire

```
x=feature_rank.loc[0:22,['feature']]
x=x['feature'].tolist()
print(x)
```

```
['median', 'mean', 'Month', 'Temperature', 'CPI', 'Fuel_Price', 'max', 'min', 'Unemployment', 'std', 'Total_MarkDown', 'IsHoliday', 'Dept_16', 'Dept_18', 'Size', 'Dept_3', 'Dept_9', 'Year', 'Dept_11', 'Dept_1', 'Dept_5', 'Dept_56', 'Dept_55']
```

Figure 4.46 : Code de caractéristiques les plus importantes classées par le modèle de forêt aléatoire .

En fin de compte, la liste x contient les noms des 23 caractéristiques les plus importantes classées par le modèle de forêt aléatoire. Ces caractéristiques peuvent être utilisées pour une analyse plus approfondie ou pour construire des modèles plus simples avec un nombre réduit de caractéristiques.

4.31.1 Caractéristiques les plus importantes

```
X = data[x]
Y = data['Weekly_Sales']
```

```
data = pd.concat([X,Y],axis=1)
```

```
data
```

Date	median	mean	Month	Temperature	CPI	Fuel_Price	max	min	Unemployment	std	...	Size	Dept_3	Dept_9	Year	Dept_1
2010-02-05	0.173215	0.208157	2	0.328495	0.840500	0.050100	0.088635	0.253530	0.508787	0.138276	...	0.630267	False	False	2010	Fals
2010-02-05	0.004767	0.004499	2	0.278565	0.875680	0.050100	0.001180	0.064648	0.305248	0.004083	...	0.492338	False	False	2010	Fals
2010-02-05	0.008968	0.009135	2	0.278565	0.875680	0.050100	0.003866	0.066212	0.305248	0.004314	...	0.492338	False	False	2010	Fals
2010-02-05	0.086290	0.085594	2	0.233627	0.873861	0.050100	0.018453	0.161046	0.291286	0.010611	...	0.650636	False	False	2010	Fals
2010-02-05	0.071542	0.080242	2	0.278565	0.875680	0.050100	0.044887	0.126993	0.305248	0.054362	...	0.492338	False	False	2010	Fals

Figure 4.47 : Code de caractéristiques les plus importantes .

Ce code crée un nouveau DataFrame ‘data‘ en sélectionnant uniquement les caractéristiques les plus importantes (‘X‘) ainsi que la variable cible ‘Weekly-Sales‘ (‘Y‘), puis les concatène ensemble.

En fin de compte, le DataFrame ‘data‘ contient les caractéristiques les plus importantes ainsi que la variable cible, prêt à être utilisé pour l’analyse ou la construction de modèles.

Résultat

Date	median	mean	Month	Temperature	CPI	Fuel_Price	max	min	Unemployment	std	...	Size	Dept_3	Dept_9	Year	Dept_1
2010-02-05	0.173215	0.208157	2	0.328495	0.840500	0.050100	0.088635	0.253530	0.508787	0.138276	...	0.630267	False	False	2010	Fals
2010-02-05	0.004767	0.004499	2	0.278565	0.875680	0.050100	0.001180	0.064648	0.305248	0.004083	...	0.492338	False	False	2010	Fals
2010-02-05	0.008968	0.009135	2	0.278565	0.875680	0.050100	0.003866	0.066212	0.305248	0.004314	...	0.492338	False	False	2010	Fals
2010-02-05	0.086290	0.085594	2	0.233627	0.873861	0.050100	0.018453	0.161046	0.291286	0.010611	...	0.650636	False	False	2010	Fals
2010-02-05	0.071542	0.080242	2	0.278565	0.875680	0.050100	0.044887	0.126993	0.305248	0.054362	...	0.492338	False	False	2010	Fals
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
2012-10-26	0.082590	0.087055	10	0.647585	0.958935	0.518036	0.030286	0.119050	0.275758	0.039642	...	0.906277	False	False	2012	Fals
2012-10-26	0.001617	0.001419	10	0.602996	0.050698	0.916333	0.000446	0.064822	0.760713	0.000478	...	0.026063	False	False	2012	Fals
2012-10-26	0.064375	0.072181	10	0.522178	0.165055	0.801102	0.063107	0.097548	0.496028	0.072670	...	0.916437	False	False	2012	Fals
2012-10-26	0.097114	0.098037	10	0.700999	0.949400	0.512024	0.020939	0.174424	0.282740	0.014648	...	0.027253	False	False	2012	Fals
2012-10-26	0.007070	0.006234	10	0.520553	0.654796	0.706413	0.002316	0.064803	0.576312	0.005210	...	0.451136	False	False	2012	Fals

374247 rows × 24 columns

Figure 4.48 : Résultat de caractéristiques les plus importantes .

4.32 division de DataFrame

```
X = data.drop(['Weekly_Sales'],axis=1)
Y = data.Weekly_Sales
```

Figure 4.49 : Code de division de DataFrame .

Ces lignes de code divisent le DataFrame ‘data‘ en deux parties :

1. **X = data.drop([’Weekly-Sales’], axis=1) :** Cette ligne crée un DataFrame ‘X‘ en supprimant la colonne ‘’Weekly-Sales‘‘ du DataFrame ‘data‘. Cela signifie que ‘X‘ contient toutes les colonnes de ‘data‘ sauf la colonne ‘’Weekly-Sales‘‘, qui est la variable cible.
2. **Y = data.Weekly-Sales :** Cette ligne crée une Series ‘Y‘ qui contient uniquement la colonne ‘’Weekly-Sales‘‘ du DataFrame ‘data‘. C’est la variable cible que nous voulons prédire.

```
X_train,X_test,y_train,y_test = train_test_split(X,Y,test_size=0.20, random_state=50)
```

Figure 4.50 : suite de code qui divise de DataFrame .

Cette ligne de code utilise la fonction ‘train-test-split‘ de scikit-learn pour diviser les données en ensembles d’entraînement et de test. Voici ce que chaque partie de la ligne fait :

1. **'X' :** Les caractéristiques que nous voulons utiliser pour entraîner le modèle.
2. **'Y' :** La variable cible que nous voulons prédire.
3. **'test-size=0.20' :** Cela signifie que 20% des données seront utilisées pour le test et 80% pour l'entraînement.
4. **'random-state=50' :** Cela fixe la graine aléatoire pour garantir la reproductibilité des résultats.
5.

4.33 Les modèles de machine learning

4.33.1 Modèle de régression linéaire

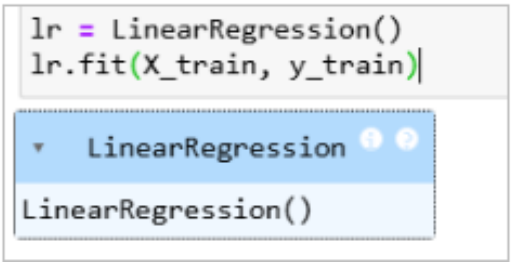

Figure 4.51 : Code de régression linéaire .

Ces lignes de code entraînent un modèle de régression linéaire sur les données d'entraînement. Voici ce que chaque ligne fait :

1. **lr = LinearRegression() :** Cette ligne crée un objet de modèle de régression linéaire à l'aide de la classe 'LinearRegression' de scikit-learn.
2. **lr.fit(X-train, y-train) :** Cette ligne ajuste (ou entraîne) le modèle de régression linéaire aux données d'entraînement. Les caractéristiques sont passées en tant que premier argument ('X-train') et les valeurs cibles correspondantes sont passées en tant que deuxième argument ('y-train').

Après l'exécution de ces lignes de code, le modèle de régression linéaire ('lr') sera entraîné et prêt à être utilisé pour faire des prédictions sur de nouvelles données.
Le tableau suivant montre les performances de l'algorithme de régression linéaire :

Accuracy	R-squared Score	Correlation Coefficient	Recall Score	F1 Score
0.9226	0.9226	0.9605	1.0	0.8928
Error Rate	MAE	MSE	RMSE	R2
0.0773	0.0300	0.003	0.0590	0.9226

Table 4.2 : Résultats obtenus avec LR .

Distribution des valeurs de ventes réelles

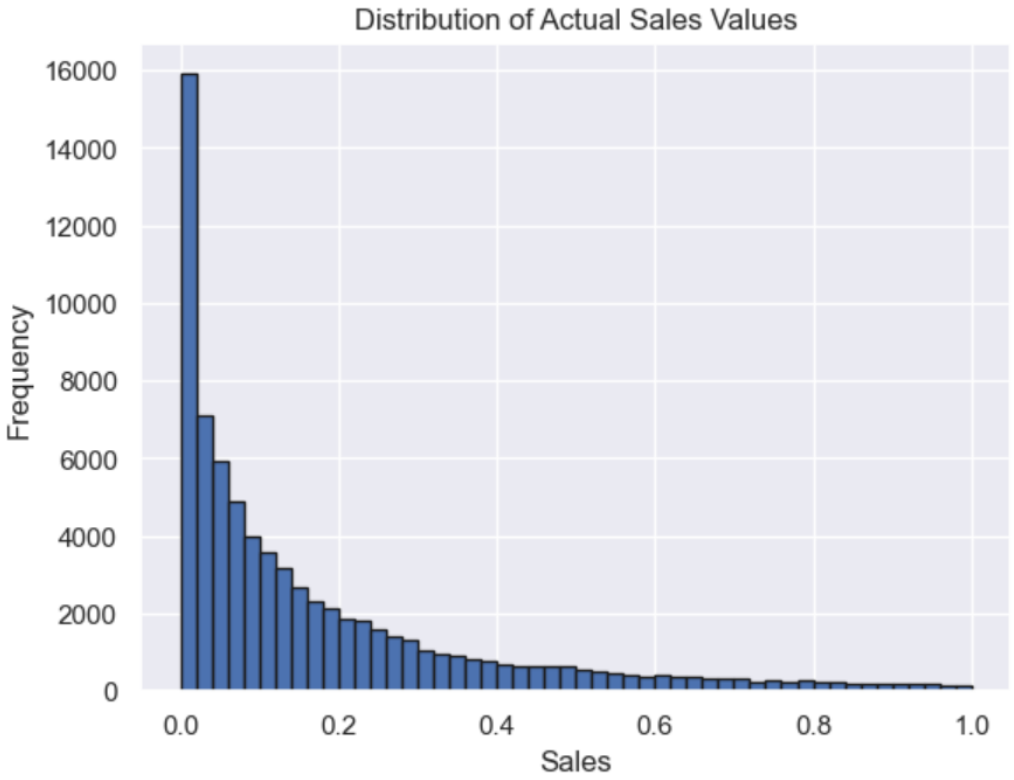

Figure 4.52 : Distribution des valeurs de ventes réelles .

Prédiction du régression linéaire

Date	Actual	Predicted
2011-08-05	0.161661	0.129306
2010-07-09	0.364278	0.279373
2011-07-01	0.005003	0.021291
2012-01-06	0.015856	0.012506
2011-08-26	0.000318	0.004835
...	...	...
2011-01-28	0.169068	0.239815
2010-08-20	0.252860	0.236692
2010-11-26	0.265617	0.325573
2010-03-12	0.008865	0.012404
2010-02-12	0.230510	0.234714

74850 rows × 2 columns

Figure 4.53 : Prédiction du régression linéaire .

Figure 4.54 : graphique de prédiction du régression linéaire .

Ce graphique permet de comparer visuellement les prédictions du modèle avec les valeurs réelles pour les 200 premières instances de l'ensemble de test.

4.33.2 Modèle de régression par forêts aléatoires

```
rf = RandomForestRegressor()
rf.fit(X_train, y_train)
```

```
RandomForestRegressor
RandomForestRegressor()
```

Figure 4.55 : Code de modèle de régression par forêts aléatoires .

Le tableau suivant montre les performances de l'algorithme de Random Forest :

Accuracy	R-squared Score	Correlation Coefficient	Recall Score	F1 Score
0.9728	0.9728	0.9863	0.9653	0.4493
Error Rate	MAE	MSE	RMSE	R2
0.0170	0.0171	0.0012	0.0349	0.9728

Table 4.3 : Résultats obtenus avec Random Forest .

Prédiction du Random Forest

Date	Actual	Predicted
2011-08-05	0.161661	0.130431
2010-07-09	0.364278	0.310731
2011-07-01	0.005003	0.013994
2012-01-06	0.015856	0.019743
2011-08-26	0.000318	0.000572
...	...	...
2011-01-28	0.169068	0.171988
2010-08-20	0.252860	0.273227
2010-11-26	0.265617	0.439510
2010-03-12	0.008865	0.015065
2010-02-12	0.230510	0.263194

74850 rows × 2 columns

Figure 4.56 : Prédiction du Random Forest .

Figure 4.57 : graphique de prédiction du Random Forest .

Ce graphique permet de comparer visuellement les prédictions du modèle avec les valeurs réelles pour les 200 premières instances de l'ensemble de test.

4.33.3 Modèle de régression par KNN

```
knn = KNeighborsRegressor(n_neighbors = 1,weights = 'uniform')
knn.fit(X_train,y_train)
```

```
KNeighborsRegressor
KNeighborsRegressor(n_neighbors=1)
```

Figure 4.58 : Modèle de régression par KNN .

Le tableau suivant montre les performances de l'algorithme de KNN :

Accuracy	R-squared Score	Correlation Coefficient	Recall Score	F1 Score
0.9422	0.9728	0.9709	0.8712	0.8732
Error Rate	MAE	MSE	RMSE	R2
0.0271	0.0271	0.0026	0.0510	0.9422

Table 4.4 : Résultats obtenus avec KNN .

Prédiction du KNN

Date	Actual	Predicted
2011-08-05	0.161661	0.112559
2010-07-09	0.364278	0.261874
2011-07-01	0.005003	0.011921
2012-01-06	0.015856	0.028551
2011-08-26	0.000318	0.001063
...	...	...
2011-01-28	0.169068	0.176391
2010-08-20	0.252860	0.252642
2010-11-26	0.265617	0.203904
2010-03-12	0.008865	0.001663
2010-02-12	0.230510	0.287258

74850 rows × 2 columns

Figure 4.59 : Prédiction du KNN

Figure 4.60 : graphique de prédiction du KNN .

4.33.4 Modèle de XGboost

```
xgbr = XGBRegressor()
xgbr.fit(X_train, y_train)
```

```
XGBRegressor

XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
```

Figure 4.61 : Modèle de XGboost .

Le tableau suivant montre les performances de l'algorithme de XGboost :

Accuracy	R-squared Score	Correlation Coefficient	Recall Score	F1 Score
0.9679	0.9679	0.9838	0.8712	0.9011
Error Rate	MAE	MSE	RMSE	R2
0.0320	0.0210	0.0014	0.0380	0.9679

Table 4.5 : Résultats obtenus avec XGboost .

Prédiction du XGboost

Date	Actual	Predicted
2011-08-05	0.161661	0.122643
2010-07-09	0.364278	0.283100
2011-07-01	0.005003	0.016122
2012-01-06	0.015856	0.019276
2011-08-26	0.000318	0.000338
...	...	...
2011-01-28	0.169068	0.216721
2010-08-20	0.252860	0.238558
2010-11-26	0.265617	0.378962
2010-03-12	0.008865	0.012178
2010-02-12	0.230510	0.257230

74850 rows × 2 columns

Figure 4.62 : prédiction du XGboost.

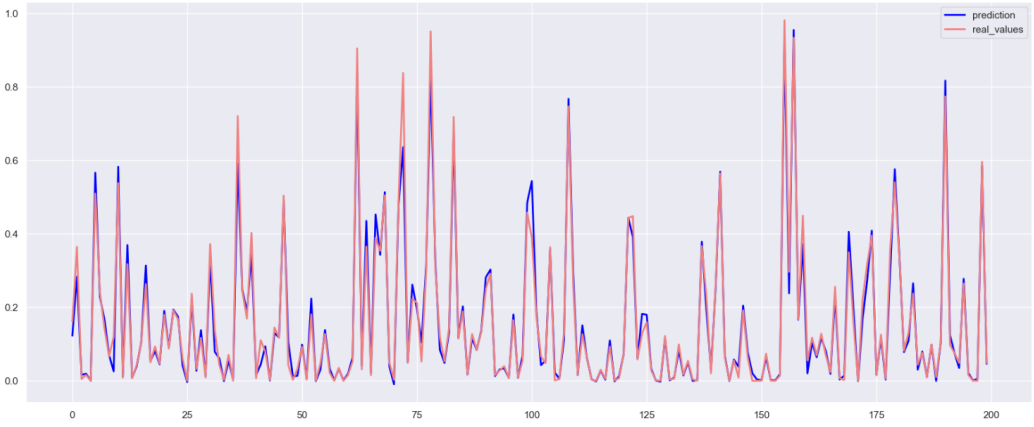

Figure 4.63 : graphique de prédiction du XGboost.

4.33.5 Modèle de Réseau neuronal profond personnalisé DNN

```
plt.figure(figsize=(8,4))
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Test Loss')
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epochs')
plt.legend(loc='upper right')
plt.show()
```

Figure 4.64 : Code de modèle de DNN .

4.33.6 Résultat

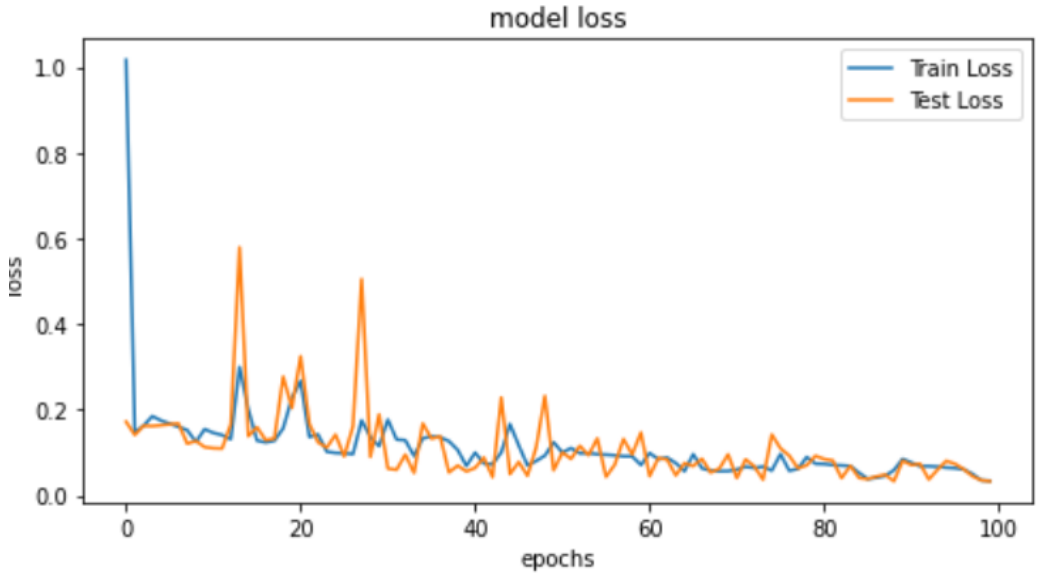

Figure 4.65 : Résultat de modèle de DNN .

Le tableau suivant montre les performances de l'algorithme de DNN :

Accuracy	R-squared Score	Correlation Coefficient	Recall Score	F1 Score
0.9664	0.9111	0.9573	0.7609	
Error Rate	MAE	MSE	RMSE	R2
0.0336	0.0366	0.0040	0.0633	0.9111

Table 4.6 : Résultats obtenus avec DNN .

Prédiction du DNN

Date	Actual	Predicted
2011-08-05	0.161661	0.126853
2010-07-09	0.364278	0.281885
2011-07-01	0.005003	0.043948
2012-01-06	0.015856	0.030796
2011-08-26	0.000318	0.026302
...	...	...
2011-01-28	0.169068	0.226628
2010-08-20	0.252860	0.233265
2010-11-26	0.265617	0.326623
2010-03-12	0.008865	0.029723
2010-02-12	0.230510	0.235078

74850 rows × 2 columns

Figure 4.66 : Prédiction du DNN .

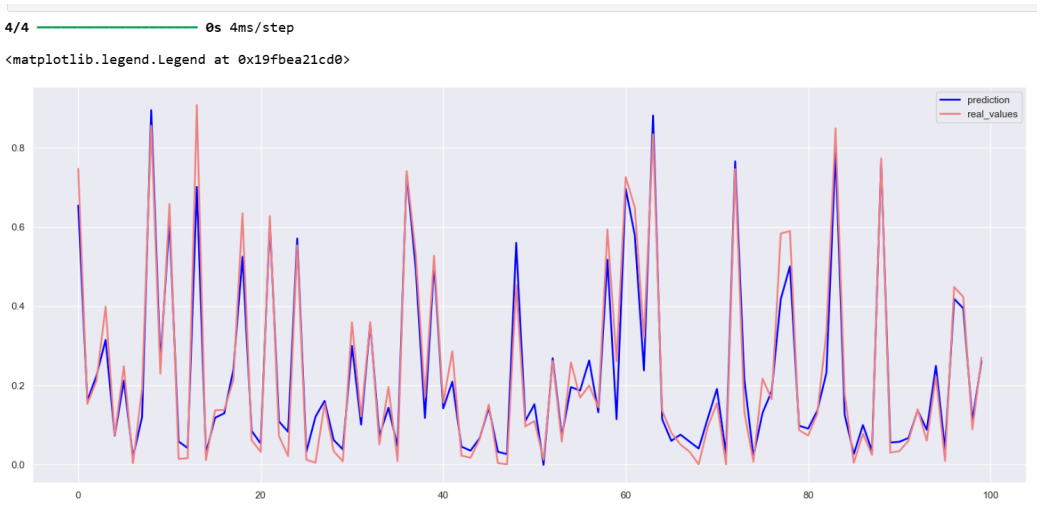

Figure 4.67 : graphique de prédiction du DNN .

4.34 Comparaison des modèles

4.34.1 Accuracy

	model	accuracy
0	lr_acc	92.269737
1	rf_acc	97.287326
2	knn_acc	94.223130
3	xgb_acc	96.791365
4	dnn_acc	96.649528

Figure 4.68 : Comparaison des modèles avec Accuracy.

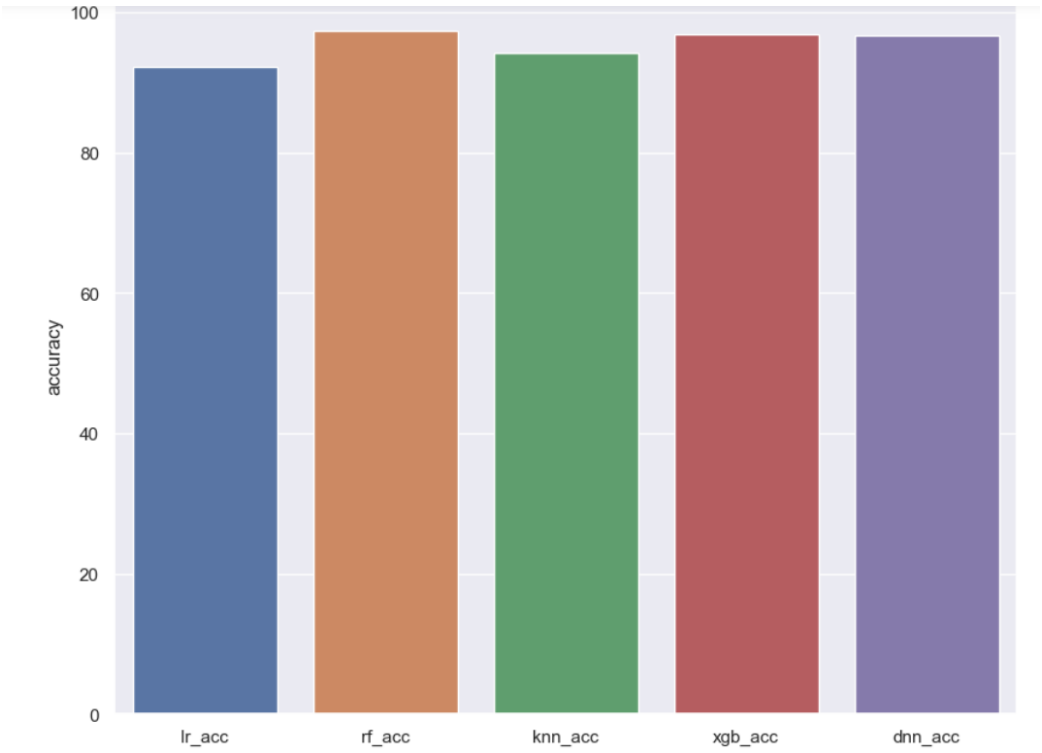

Figure 4.69 : graphique de omparaison des modèles avec Accuracy.

4.34.2 Correlation coefficient

	model	correlation_coefficient
0	knn_correlation_coefficient	0.970952
1	xgb_correlation_coefficient	0.983840
2	lr_correlation_coefficient	0.960573
3	rf_correlation_coefficient	0.986348
4	dnn_correlation_coefficient	0.957393

Figure 4.70 : Comparaison des modèles avec correlation coefficient .

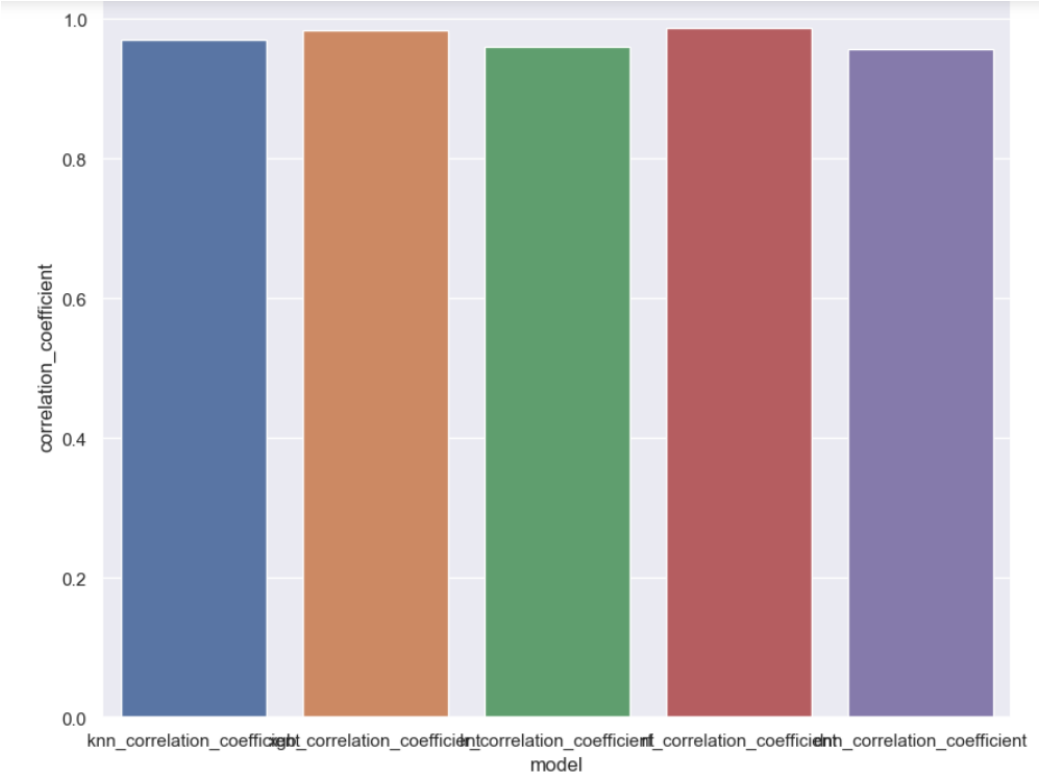

Figure 4.71 : graphique de comparaison des modèles avec correlation coefficient.

4.34.3 F1 Score

	model	F1_Score
0	knn_f1	0.872544
1	xgb_f1	0.903497
2	lr_f1	0.893054
3	rf_f1	0.448443

Figure 4.72 : Comparaison des modèles avec F1 Score .

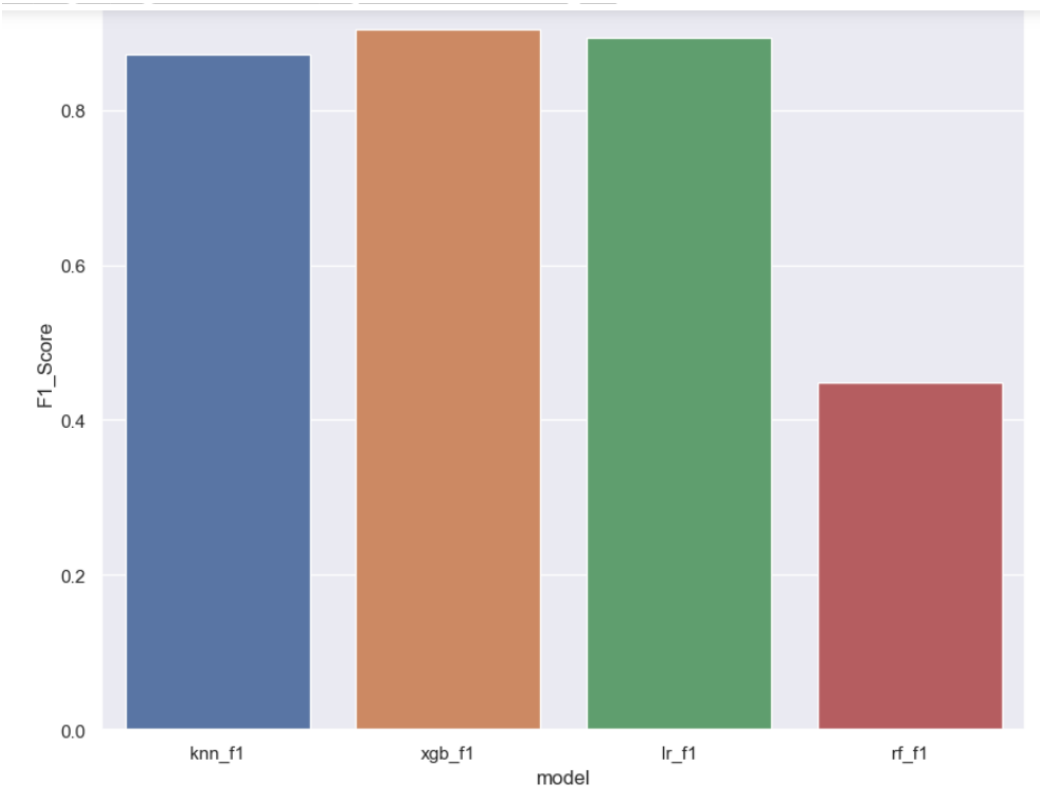

Figure 4.73 : graphique de comparaison des modèles avec F1 Score.

D'après ce schéma on remarque que le modèle de RF n'est pas vraiment performant et les meilleurs classificateurs est le XGB .

4.35 Overfiting

```
LR - Training Accuracy: 0.7651679876551869, Testing Accuracy: 0.7654108216432866, Difference: -0.0002428339880996555
RF - Training Accuracy: 0.9999799597190353, Testing Accuracy: 0.9294054776219105, Difference: 0.0705744820971248
KNN - Training Accuracy: 0.9340407552513886, Testing Accuracy: 0.9023647294589179, Difference: 0.03167602579247075
XGB - Training Accuracy: 0.9367061126196989, Testing Accuracy: 0.9242752171008684, Difference: 0.012430895518830587
```

Figure 4.74 : Test de overfiting et Underfitting pour Les 4 modèles .

Figure 4.75 : graphique de test de overfiting et Underfitting pour Les 4 modèles .

4.35.1 Le modèle DNN

```
Epoch 1/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 30s 3ms/step - accuracy: 0.8801 - loss: 0.3129
Epoch 2/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 42s 3ms/step - accuracy: 0.9049 - loss: 0.2319
Epoch 3/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 37s 4ms/step - accuracy: 0.9082 - loss: 0.2217
Epoch 4/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 51s 5ms/step - accuracy: 0.9104 - loss: 0.2162
Epoch 5/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 30s 3ms/step - accuracy: 0.9125 - loss: 0.2100
Epoch 6/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 26s 3ms/step - accuracy: 0.9128 - loss: 0.2089
Epoch 7/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 25s 3ms/step - accuracy: 0.9135 - loss: 0.2078
Epoch 8/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 31s 3ms/step - accuracy: 0.9142 - loss: 0.2048
Epoch 9/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 36s 4ms/step - accuracy: 0.9142 - loss: 0.2055
Epoch 10/10
9357/9357 ━━━━━━━━━━━━━━━━━━━━ 33s 4ms/step - accuracy: 0.9146 - loss: 0.2034
DNN Training Accuracy: 0.915810763835907
DNN Testing Accuracy: 0.913480281829834
```

Figure 4.76 : Test de overfiting et Underfitting pour modèle DNN .

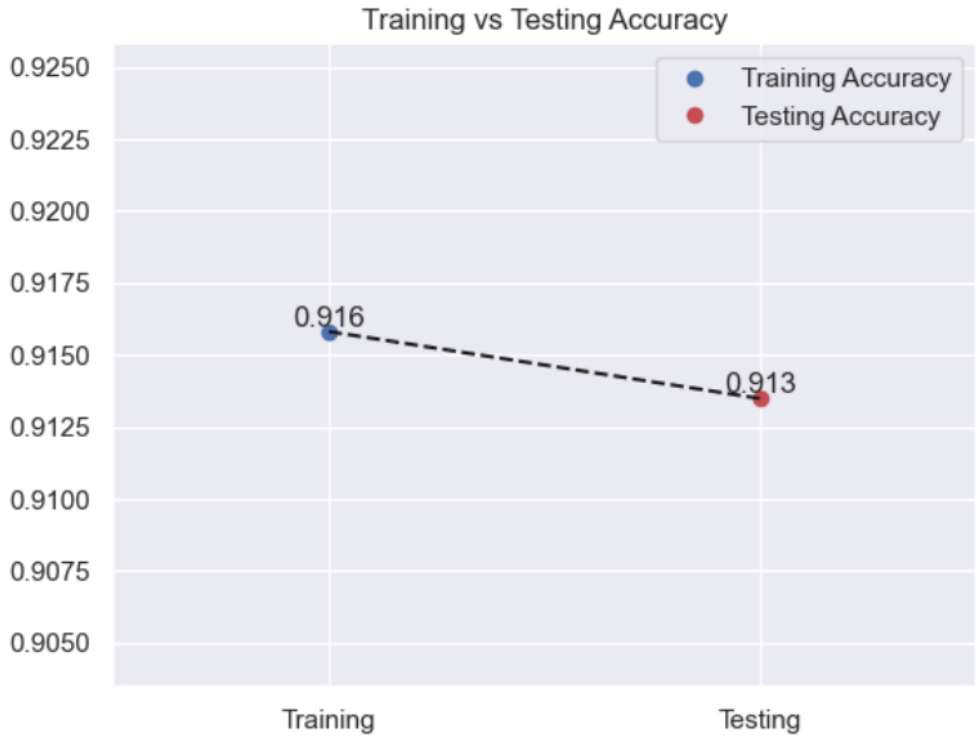

Figure 4.77 : graphique de Test de overfiting et Underfitting pour modèle DNN .

4.35.2 Analyse de résultats

Analysons les résultats des différentes précisions pour déterminer s'il y a un problème de surapprentissage (overfitting) et de sous-apprentissage (underfitting) . Le surapprentissage se produit lorsque le modèle fonctionne très bien sur les données d'entraînement (haute précision), mais nettement inférieure sur les données de test. Nous devons vérifier si les précisions de l'entraînement et du test sont toutes deux faibles, ce qui indiquerait que le modèle ne capture pas bien les motifs des données.Voici l'analyse des résultats :

Modèle	LR	RF	KNN	XGBoost	DNN
Précision de l'entraînement	0.7652	0.99998	0.9340	0.9367	0.9158
Précision du test	0.7654	0.9294	0.9024	0.9243	0.9135
Différence	-0.0002	0.0706	0.0317	0.0124	0.0023

Table 4.7 : L'entraînement et le test de chaque modèle .

- Régression Linéaire montre des signes de sous-apprentissage, avec des précisions relativement basses pour l'entraînement et le test.
- Régression Linéaire,XGBoost et DNN ne montrent pas de signes évidents de surapprentissage, car la différence entre la précision de l'entraînement et du test est très petite.

- Forêt Aléatoire montre des signes évidents de surapprentissage, car la différence entre la précision de l'entraînement et du test est grande.
- KNN pourrait avoir un certain surapprentissage, mais ce n'est pas très grave.

Pour améliorer un modèle de Random Forest qui souffre de surapprentissage, voici quelques techniques essentielles :

- Réduire la profondeur des arbres ('max-depth').
- Limiter le nombre de caractéristiques par split ('max-features').
- Augmenter le nombre minimum d'échantillons par feuille ('min-samples-leaf').
- Augmenter le nombre minimum d'échantillons pour diviser un nœud ('min-samples-split').
- Augmenter le nombre d'arbres('n-estimators'), en veillant à ne pas augmenter la complexité individuelle des arbres.
- Utiliser la validation croisée pour optimiser les hyperparamètres ('GridSearchCV' ou 'RandomizedSearchCV').

Pour améliorer un modèle qui souffre de sous-apprentissage, comme la Régression Logistique, vous pouvez essayer les techniques suivantes :

- Utiliser un modèle plus complexe ou non-linéaire.
- Ajouter des caractéristiques supplémentaires ou effectuer une ingénierie des caractéristiques pour mieux capturer les motifs des données.
- Augmenter la durée de l'entraînement ou ajuster les hyperparamètres pour permettre au modèle d'apprendre davantage des données.

Ces ajustements aident à contrôler la complexité du modèle et à améliorer sa généralisation.

4.36 Analyse de la Précision pour Chaque Modèle

Modèle	LR	RF	KNN	XGBoost	DNN
Underfitting	Oui	Non	Non	Non	Non
Overfitting	Non	Oui	Non	Non	Non

Table 4.8 : Test de underfitting et overfitting de chaque modéle .

- Régression Linéaire (LR) souffre de sous-ajustement, avec une précision relativement faible pour l'entraînement et le test.

- Forêt Aléatoire (RF) présente un surajustement, avec une différence importante entre la précision de l'entraînement et celle du test.
- K Plus Proches Voisins (KNN) ne montre ni sous-ajustement ni surajustement significatifs.
- XGBoost (XGB) et Réseau de Neurones Profonds (DNN) ne montrent ni sous-ajustement ni surajustement, avec des différences minimes entre la précision de l'entraînement et celle du test.

4.37 Etude de cas(interprétation)

Le modèle formé à l'aide de l'algorithme XGBoost a montré une performance exceptionnelle par rapport aux autres algorithmes. Le modèle a atteint un F1-score élevé, ce qui indique une grande précision des prévisions.

Des graphiques ont été utilisés pour analyser et interpréter les résultats. Les graphiques ont montré une bonne correspondance entre les valeurs réelles et les valeurs prévues des ventes.

4.38 Exemple d'analyse des résultats des Ventes

Le modèle de prévision des ventes du magasin utilisant l'algorithme XGBoost a été évalué pour ses performances en termes de précision des prévisions. Voici un exemple de résultat pour la date du 6 janvier 2012 :

- Ventes actuelles : $DA = 0.015856$
- Ventes prévues (XGBoost) : $DP = 0.019276$

Date	Actual	Predicted
2011-08-05	0.161661	0.122643
2010-07-09	0.364278	0.283100
2011-07-01	0.005003	0.016122
2012-01-06	0.015856	0.019276
2011-08-26	0.000318	0.000338
...	...	...

Figure 4.78 : Exemple de prédiction du XGboost.

Avec les calculs, les niveaux de stock pour la date du 6 janvier 2012 :

4.38.1 Hypothèses

- Délai de réapprovisionnement (T) : 10 jours
- Écart-type de la demande (σ) : 0.002
- Quantité de commande (QT) : 0.005
- Stock actuel (SA) : 0.02
- Niveau de service désiré : 95% (Z = 1.65)

4.38.2 Stock de sécurité

Les formules pour calculer le stock de sécurité sont les suivante :

Méthode de Z-score

$$SS = Z \times \sigma \times \sqrt{T}$$

où :

- Z est le score Z correspondant à un niveau de service spécifique.
- σ est l'écart-type de la demande.
- T est le délai de réapprovisionnement.

4.38.3 Stock minimum :

La formule pour calculer le stock minimum est la suivante :

$$SM = DC + SS$$

où :

- SM est le stock minimum.
- DC est la demande moyenne pendant le délai de réapprovisionnement.
- SS est le stock de sécurité.

4.38.4 Stock maximum:

La formule pour calculer le stock maximum est la suivante :

$$\text{Stock maximum} = \overline{D} \times (L + T) + \text{Stock de sécurité}$$

Où :

- $\overline{D}$ est la demande moyenne quotidienne.
- L est le délai de livraison.
- T est la période de réapprovisionnement.

4.38.5 Stock Moyen (SMO)

Le stock moyen peut être calculé en utilisant la formule suivante :

$$SMO = \frac{(SMX + SM)}{2}$$

où :

- SMO est le stock moyen.
- SMX est le stock maximum.
- SM est le stock minimum.

Stock d'Alerte (SA)

Le stock d'alerte peut être calculé comme suit :

$$SA = SMO \times \%SA$$

où :

- SA est le stock d'alerte.
- SMO est le stock moyen.
- $\%SA$ est le pourcentage du stock moyen ou du stock maximum que vous souhaitez utiliser comme stock d'alerte.

Résumé des résultats :

- **Stock Minimum (SM)** : 0.029676
- **Stock Maximum (SMX)** : 0.014676
- **Stock de Sécurité (SS)** : 0.0104
- **Stock Moyen (SMO)** : 0.022176
- **Stock d'Alerte (SA)** : 0.007338

4.39 Conclusion

Dans ce chapitre, nous avons donné les résultats obtenus à partir des différentes approches avec le Dataset qui contient 421 570 lignes. Nous avons utilisé cinq algorithmes d'apprentissage automatiques pour la prédiction temporelle. Ces algorithmes sont les suivants : les K plus proches voisins , la régression logistique , la forêt aléatoire, XGBoos et es réseaux de neurones profonds .

À travers l'analyse, il a été observé que le modèle de la XGBoost fournissait les prédictions de ventes les plus précises et des relations légères ont été observées entre des facteurs tels que la taille du magasin, les vacances, le chômage et les ventes hebdomadaires.

Conclusion générale

Au cours des dernières années,l'optimisation des niveaux de stock est devenue cruciale pour le succès et la rentabilité des entreprises de commerce de détail. Une gestion inefficace des stocks peut entraîner des ruptures, des surstocks et des coûts élevés de stockage, impactant négativement la rentabilité et la satisfaction des clients. Pour relever ce défi, nous avons utilisé des techniques avancées d'apprentissage automatique et d'apprentissage profond pour la prévision des séries temporelles.

Dans un premier temps, nous avons exploré la performance d'un sous-ensemble de magasins Walmart et prédit les ventes hebdomadaires futures pour ces magasins sur la base de plusieurs modèles, notamment les k plus proches voisins (KNN), la régression linéaire, la forêt aléatoire, XGBoost et les réseaux de neurones profonds (DNN).

De plus,une analyse exploratoire des données a été réalisée sur le jeu de données pour explorer les effets de différents facteurs tels que les vacances, le prix du carburant et la température sur les ventes hebdomadaires de Walmart.

À travers cette étude, il a été démontré que l'algorithme XGBoost est le meilleur pour prédire les ventes d'un magasin grâce à sa performance remarquable sur le critère du F1-score. Cependant, il est toujours possible de réduire davantage l'écart par une optimisation continue du modèle. Ces résultats justifient l'utilisation de XGBoost pour des applications de prévision de ventes dans des scénarios similaires.

Après avoir obtenu les résultats et entrepris les actions nécessaires pour améliorer les niveaux de stock, plusieurs perspectives peuvent être appliquées pour atteindre cet objectif efficacement. Cela comprend l'amélioration des processus de prévision et de gestion de la production, l'utilisation de la technologie pour améliorer la visibilité des stocks, l'optimisation de la chaîne d'approvisionnement, le développement de stratégies pour éliminer les stocks excédentaires, et la réduction des coûts liés au stockage. En appliquant ces perspectives, les entreprises peuvent améliorer l'efficacité de la gestion des stocks et atteindre leurs objectifs de manière plus efficace et efficiente.

Bibliographie

[ALI17] LABIAD ALI. SÉlection des mots clÉs basÉe sur la classification et l'extraction des rÈgles d'association, 6/ 2017.

[Ami22] BOUKERTOUTA Mohammed Amin. Detection des intrusions basée sur l'apprentissage automatique dans les systèmes ido (internet des objets), 6/ 2022.

[Ass21] AssDent. *https://fr.linkedin.com/pulse/la-gazette-n1-gestion-des-stocks-dans-les-cabinets-dentaires-*. 5/ 2021.

[Aze22] Chloé-Agathe Azencott. *Introduction au Machine Learning*, volume 263. 2100834762 edition, 2022.

[Aï05] Esma Aïmeur. Introduction à l'intelligence artificielle. 6/ 2005.

[BEL20] Hacene BELLAHMER. Implémentation et évaluation d'un modèle d'apprentissage automatique pour l'estimation de la valeur marchande de propriétés immobilières, 2020.

[Ben10] Benmammar. L'apprentissage automatique. Technical report, Campus de Chetouane - Université Abou Bekr Belkaid, 6/ 2010.

[Bet21] Rachid El Bettioui. Gestion de stock et rentabilité financière des coopératives : Etude quantitative inventory management and financial profitability of cooperatives: Quantitative study. *Journal of Academic Finance (J.o A.F.)*, 12, 2021.

[Bon13] Gianluca Bontempi. Machine learning strategies for time series prediction machine learning summer school (hammamet, 2013). Technical report, Machine Learning group, Computer Science Department Boulevard, 2013.

[com23] France compétences. *https://www.intelligence-artificielle-school.com/ecole/technologies/comprendre-l-underfitting/*. 5/ 2023.

[DMl22] Boudjemaa M a Delladj M l. L'importance du choix de la méthode de calcul stock de sécurité et son influence sur le coût de stockage, 6/ 2022.

[DSSQ22] Ludovic De, Matteis Steeven, and Janny Solal Nathan Wenqi Shu-Quartier. Introduction à l'apprentissage automatique. *Culture Sciences de l'Ingénieur*, 108, 5/ 2022.

[Emi24] Myriam Emilion. https://www.jedha.co/formation-ia/arbre-de-decision-random-forest, 5/ 2024.

[EN21] Samadi Taieb Essedik and Brahmi Mohamed Naime. Deep learning pour l'estimation et l'adaptation de la modulation dansles systèmes ofdm, 2021.

[FA23] Houamed Fatima and Sekhara Amina. Prédiction de séries chronologiques en utilisant un modèle de deep learning etude de cas : Sonelgaz, 2023.

[Far12] BENABBAS Farouk. *Méthodes Heuristiques pour la prédiction des séries temporelles Benabbas Farouk*. PhD thesis, 2012.

[Fas24] FasterCapital. https://fastercapital.com/fr/contenu/gestion-des-stocks—gestion-des-stocks-pour-les-detaillants—comment-gerer-et-optimiser-votre-inventaire.htmlpr-vision-des-stocks-et-planification-de-la-demande, 3/ 2024.

[ft15] fmi.univ tiaret.dz. Chapitre 1 : Naissance de l'ia. Technical report, Université Ibn khaldoun Tiaret, 3/ 2015.

[Hic23] Michael Hickins. https://www.oracle.com/ca-fr/retail/fashion/retail-inventory-management/, 4/ 2023.

[Hma18] Youssef Hmamouche. *Prédiction des Séries Temporelles Larges*. PhD thesis, 12/ 2018.

[IBM24] IBM. https://www.ibm.com/fr-fr/topics/deep-learning, 2024.

[JT17] Andrew T. Jebb and Louis Tay. Introduction to time series analysis for organizational research: Methods for longitudinal analyses. *Organizational Research Methods*, 20:61–94, 1/ 2017.

[jun22] jungheinrich. *https://www.jungheinrich-profishop.fr/fr/guide-pro/gestion-des-stocks/*. 2022.

[Kas22] Raphael Kassel. *https://datascientest.com/underfitting-tout-savoir*. 12/ 2022.

[LL20] Whitney Kate Chucya Lozano and Chucya Lozano. Développement d'un modèle de prédiction de retours de vêtements en fonction des caractéristiques des clients et des produits, 2020.

[LS20] AIBOUD Lila and LASKRI Samia. Appréciation de la qualité des leads dans le marketing numérique à l'aide de l'apprentissage profond, 11/ 2020.

[Lum23] Lumivero. https://help.xlstat.com/fr/6504-quel-outil-danalyse-de-series-temporelles-choisir, 2023.

[MAH18] FATIMA AIT MAHAMMED. Approches d'apprentissage automatique pour la détection du spam web : exploration de diverses caractéristiques, 3/ 2018.

[MAH22] MELBOUCI MAHFOUD. La gestion des approvisionnements et des stocks dans une entreprise industrielle. cas sarl saemo, 2022.

[MAR13] Lauréna MARQUIS. *MISE EN PLACE D'UNE GESTION DE STOCK AU SEIN D'UNE ENTREPRISE D'AMENAGEMENT PAYSAGER*. PhD thesis, 2013.

[Mar20] Léa Marmande. *Le machine learning dans le cadre de l'innovation technologique*. PhD thesis, 9/ 2020.

[Mar22] Sandra Martin. *https://supply-chain.net/stock-minimum-calcul/*. 2022.

[Mat24] The MathWorks. *https://www.mathworks.com/discovery/overfitting.html*. 2024.

[Mau18] Bastien Maurice. *https://deeplylearning.fr/cours-theoriques-deep-learning/comprendre-overfitting-et-underfitting/*. 9/ 2018.

[Mec13] Raihane Mechgoug. *La Prédiction des Séries Temporelles utilisant les Paradigmes de Soft Computing*. PhD thesis, 6/ 2013.

[Mec23] S.A. Mecalux. https://www.mecalux.fr/blog/niveaux-de-stock, 4/ 2023.

[Mer21] CHAOUCHKHOUANE Meriem. Méthode de box et jenkins, 6/ 2021.

[MEZ20] Ameni MEZNI. Approches d'apprentissage automatique pour la prédiction de la qualité de performance dans les réseaux optiques opérationnels, 9/ 2020.

[MO20] Hadjersi Mohamed and Benguergoura Oussama. Le machine learning pour l'evaluation automatique des réponses courtes : Application à la langue arabe., 2020.

[Mon13] V. Monbet. Modélisation de séries temporelles. Technical report, université de Rennes, 2013.

[Mon17] V Monbet. Modélisation des séries temporelles. Technical report, Université de Rennes, 2017.

[Mon20] Jennifer Montérémal. *https://www.pennylane.com/fr/fiches-pratiques/gestion-achats/stock-minimum-et-stock-maximum-definitions/*. 5/ 2020.

[Mon22] Jennifer Montérémal. *https://www.appvizer.fr/magazine/operations/gestion-de-stock/stock-dormant*. 11/ 2022.

[Nad13] Marref Nadia. Apprentissage incrémental machines à vecteurs supports, 12/ 2013.

[Nes22] DEGDEG Nesrine. Application de l'apprentissage automatique pour un problème de classification des machines de l'entreprise mega-papiers, 7/ 2022.

[No[U+FFFD]2] Valentin Noël. Séries temporelles et réseaux de neurones récurrents. *Culture Sciences de l'ingénieur*, 109, 2022.

[Ora22] Oracle. *https://blogs.oracle.com/oracle-france/post/qu-est-ce-que-le-surapprentissage*. 10/ 2022.

[phi18] philippe.gesset. L'intelligence artificielle. *La Technologie en Collège*, 5/ 2018.

[Rah23] Sakouhi Rahma. *https://prezi.com/p/skvwkigcpifh/fitting-overfitting-et-underfitting/*. 4/ 2023.

[RB22] E. Brasnu R. Bunod, E. Augstburger. Intelligence artificielle et glaucome : une revue de la littératureartificial intelligence and glaucoma: A literature review. *Journal Français d'Ophtalmologie*, 45, 2/ 2022.

[Red20] BBelkacemi Redha. Gestion des stocks. Technical report, Faculté de Médecine- Département de Pharmacie 5 éme année Pharmacie, 4/ 2020.

[Roq22] Alexandre Roquoplo. *https://www.appvizer.fr/magazine/operations/gestion-de-stock/stock-securite*. 3/ 2022.

[Rou21] L Rouvière. Introduction au machine learning. Technical report, université rennes , école d'ingénieur et de commerce, 6/ 2021.

[Saf24] SafetyCulture. https://safetyculture.com/fr/themes/gestion-des-stocks-de-vente-au-detail/, 1/ 2024.

[Sam20] DENDOUGA Samah. Séries temporelles : Théorie et application, 9/ 2020.

[Sau20] Guillaume Saupin. https://www.verteego.com/blog/xgboost-insuffisant-pour-lextrapolation-des-s

[SOF14] BECHIRI SOFIANE. Développement d'une application access sur la gestion du stock d'outillage au niveau du département de génie mecanique, 2014.

[Tai14] Souhaib Ben Taieb. *Machine learning strategies for multi-step-ahead time series forecasting*. PhD thesis, 9/ 2014.

[TAL18] Ibrahima TALL. Analyse descriptive des series temporelles. Technical report, Ecole nationale de la statistique et de l'analyse économique Pièrre Ndiaye, 3/ 2018.

[TB98] Michel Terraza and Régis Bourbonnais. *Analyse des séries temporelles en économie*, volume 108. Presses universitaires de France rédition numérique FeniXX, isbn 2 13 049370x edition, 1998.

[TWB+21] Samya Tajmouati, Bouazza El Wahbi, Adel Bedoui, Abdallah Abarda, and Mohamed Dakkoun. Applying k-nearest neighbors to time series forecasting : two new approaches. 3/ 2021.

[VAL15] VALPOLIS. *https://www.petite-entreprise.net/P-1334-136-G1-definition-de-la-gestion-des-stocks.html*. 4/ 2015.

[YM15] Djendel Youcef and Bouchen Melissa. Analyse critique de la gestion des stocks et la distribution etude de cas : Tchin-lait / candia, 2015.

[[U+FFFD]dlSedldl15] ENSAI École Nationale de la Statistique et de l'Analyse de l'Information. Séries temporelles 2a. Technical report, ENSAI École Nationale de la Statistique et de l'Analyse de l'Information, 12/ 2015.