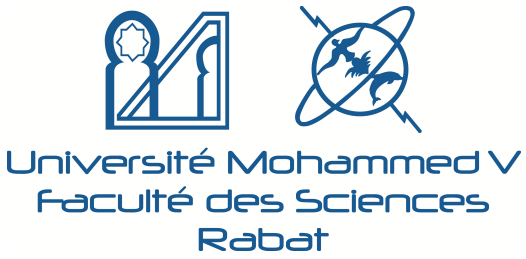
UNIVERSITÉ MOHAMMED V DE RABAT
Faculté des Sciences



Laboratoire (LRIT)
Master Informatique et Télécommunications

Projet en Machine learning et Deep Learning

---

le Feature Engineering en Machine Learning

Présentée par :
HAOUDI MARWA & SAIDI HANANE

Soutenu le 10 Avril 2025 devant le Jury

Pr. Abdelhak Mahmoudi Professeur à la Faculté des Sciences - Rabat

Année Universitaire 2024-2025

Table des matières

Introduction

Le Feature Engineering est un processus crucial dans le développement de modèles de Machine Learning (ML), car il transforme les données brutes en caractéristiques utiles et pertinentes, appelées "features", qui permettent aux modèles d'apprendre efficacement. La qualité des features utilisées a un impact direct sur la performance du modèle, en influençant sa capacité à généraliser, sa précision et sa vitesse d'entraînement. En d'autres termes, un bon Feature Engineering peut faire la différence entre un modèle performant et un modèle médiocre.

Le domaine du Machine Learning repose sur le principe fondamental que "les données sont le pétrole, mais les features en sont le moteur", ce qui souligne l'importance capitale de cette phase préparatoire. En effet, la transformation des données brutes en variables significatives n'est pas seulement une question de formatage, mais aussi d'extraction de relations complexes et de modèles sous-jacents, essentiels pour l'apprentissage automatique.

Les techniques de Feature Engineering incluent la sélection, l'extraction, la transformation des features, ainsi que la création de nouvelles variables. Ces méthodes sont essentielles pour rendre les données plus accessibles et plus compréhensibles pour les algorithmes d'apprentissage. Ce rapport explore les principales approches et techniques utilisées dans le Feature Engineering et analyse leur impact sur la performance des modèles de Machine Learning. Nous y abordons également les meilleures pratiques permettant d'optimiser l'utilisation des features pour maximiser la précision et l'efficacité des prédictions.

Chapitre 1

Cadre Théorique du Feature Engineering

1.1 Définition et Importance

Le Feature Engineering est une étape clé du Machine Learning qui consiste à transformer les données brutes en variables pertinentes pour les modèles. Il comprend des pratiques telles que la sélection, la transformation, la création et l'extraction de caractéristiques, dans le but d'optimiser les performances des algorithmes.

Les données initiales sont souvent désorganisées ou incomplètes, ce qui peut nuire à la qualité des prédictions. Grâce au Feature Engineering, on peut extraire les informations utiles, traiter les valeurs manquantes, créer de nouvelles variables révélant des relations importantes, et adapter les données au format attendu par les modèles.

Ce processus permet également de réduire la complexité des modèles, de limiter le risque de surapprentissage, et d'améliorer la lisibilité des résultats. Il est utilisé dans de nombreux domaines comme la finance, la santé, le marketing ou les sciences sociales, car il permet d'identifier des tendances, de révéler des patterns cachés et de rendre les données exploitables pour l'analyse prédictive.

1.2 Types de Feature Engineering

En Machine Learning, les features sont les variables d'entrée utilisées par les modèles pour effectuer des prédictions. Elles se divisent en plusieurs types : numériques, catégorielles, textuelles et temporelles.

- **Features numériques** : Les features numériques représentent des variables quantitatives continues ou discrètes. Elles sont exprimées sous forme de nombres et peuvent être utilisées directement par la plupart des algorithmes de Machine Learning.

- **Features catégorielles** : Les features catégorielles sont des variables discrètes qui prennent un nombre limité de valeurs. Elles doivent souvent être transformées en valeurs numériques à l'aide de techniques comme le One-Hot Encoding ou le Label Encoding pour être compatibles avec les modèles de Machine Learning.

- **Features textuelles** :

Les features textuelles contiennent du texte brut. Elles nécessitent souvent une transformation en représentations numériques exploitables, par exemple à l'aide de techniques de Traitement Automatique du Langage (NLP).

- **Features temporelless** :

Les features temporelles correspondent à des données associées à un point précis dans le temps. Elles nécessitent souvent des transformations spécifiques comme l'extraction du jour, du mois ou de l'année, ou encore l'utilisation de modèles adaptés aux séries temporelles.

1.3 Techniques de Feature Engineering

Le Feature Engineering regroupe plusieurs techniques permettant d'améliorer la qualité des données utilisées par les modèles de Machine Learning. Ces techniques se déclinent en plusieurs catégories, allant de la sélection des variables les plus pertinentes, à leur transformation, en passant par la création de nouvelles features.

1.3.1 Sélection des Features

La sélection des features vise à identifier les variables les plus pertinentes et à éliminer celles qui sont redondantes ou peu utiles, ce qui améliore la précision des modèles et réduit leur complexité. Parmi les méthodes courantes, on utilise l'évaluation de l'importance des features, comme avec les modèles Random Forest ou XGBoost, qui attribuent un score à chaque variable selon son impact sur les prédictions. D'autres techniques, comme les tests statistiques (chi-deux, ANOVA) et les méthodes de régularisation comme Lasso, permettent de filtrer les variables les moins significatives.

1.3.2 Extraction des Features

L'extraction des features est une technique utilisée pour réduire la dimensionnalité des données tout en conservant l'essentiel de l'information. Elle permet d'améliorer les performances des modèles de Machine Learning, en évitant notamment la malédiction de la dimensionnalité. Parmi les méthodes les plus courantes, on trouve le PCA (Analyse en Composantes Principales), qui transforme les variables initiales en nouvelles composantes non corrélées tout en maximisant la variance des données. À l'inverse, le LDA (Analyse Discriminante Linéaire) vise à maximiser la séparation entre les différentes classes, ce qui le rend particulièrement adapté aux problèmes de classification.

1.3.3 Transformation des Features

a transformation des features modifie les variables pour les rendre compatibles avec les modèles de Machine Learning. L'encodage catégoriel inclut le One-Hot Encoding, qui crée des colonnes binaires pour chaque catégorie, et le Label Encoding, qui attribue un entier à chaque catégorie. La normalisation (Min-Max Scaling) ajuste les valeurs dans un intervalle donné, tandis que la standardisation (Z-score) centre et réduit les données pour un écart-type de 1.

Chapitre 2

Approche Méthodologique

2.1 Outils et Technologies

La mise en œuvre du Feature Engineering nécessite l'utilisation d'un ensemble d'outils et de technologies adaptés afin de traiter, transformer, et exploiter efficacement les données en vue d'améliorer les performances des modèles d'apprentissage automatique. Les outils choisis dépendent des exigences spécifiques de chaque projet, incluant la complexité des données et la nature des transformations à appliquer.

- **Langages et Bibliothèques :**

— **Python** : Langage central du Machine Learning, apprécié pour sa simplicité et son riche écosystème.

— **pandas** : Outil essentiel pour la manipulation et le prétraitement des données sous forme de `DataFrames`.

— **scikit-learn** : Fournit des outils pour la sélection, l'encodage, la normalisation et la réduction de la dimensionnalité des features.

— **Featuretools** : Automatise la création et la transformation avancée des features à partir de données structurées.

— **TensorFlow** : Utilisé en deep learning, il optimise l'intégration des features dans des modèles prédictifs avancés.

- **Techniques et Algorithmes :**

— **PCA (Principal Component Analysis)** : Technique de réduction de la dimensionnalité transformant des variables corrélées en composantes principales indépendantes, tout en préservant un maximum de variance.

— **K-Means** : Algorithme de clustering regroupant les observations en clusters similaires. Il permet de créer de nouvelles features reflétant l'appartenance des données à un groupe spécifique.

— **Encodage des variables catégorielles** : Conversion des variables catégorielles en valeurs numériques via *One-Hot Encoding* (colonnes binaires par catégorie) ou *Label Encoding* (assignation d'un entier à chaque catégorie).

2.2 Étapes de Mise en Œuvre

Le Feature Engineering est une étape clé dans le pipeline du Machine Learning, visant à optimiser la représentation des données afin d'améliorer la performance des modèles prédictifs. Ce processus se déroule en plusieurs étapes essentielles :

1) **Exploration des données (Exploratory Data Analysis - EDA)** Cette première phase consiste à analyser la structure des données, identifier les valeurs manquantes, détecter d'éventuelles anomalies et comprendre les corrélations entre les variables. Une exploration approfondie permet de dégager des insights clés pour orienter les choix de transformation et de sélection des features.

2) **Prétraitement des données** Cette étape inclut la gestion des valeurs manquantes, le traitement des valeurs aberrantes et le nettoyage général des données brutes. Des techniques comme l'imputation, l'élimination des valeurs extrêmes et la normalisation sont appliquées afin de garantir la cohérence et la qualité des données utilisées pour l'entraînement du modèle.

3) **Sélection et transformation des features** L'objectif ici est d'identifier les variables les plus pertinentes en fonction de leur impact sur les performances du modèle. Des techniques de sélection comme l'analyse de l'importance des features (via Random Forest ou Lasso) et des transformations adaptées (encodage catégoriel, standardisation, normalisation) sont appliquées pour maximiser l'efficacité du modèle.

4) **Évaluation et comparaison des performances** Une analyse comparative est menée entre les performances du modèle avant et après l'application du Feature Engineering. Cette validation permet de quantifier l'apport des transformations effectuées et d'affiner les stratégies employées pour obtenir des résultats optimaux. —

2.3 Études de Cas et Applications

-Cas 1 : Exploitation des Données Temporelles et Extraction du Jour de la Semaine

L'intégration des données temporelles dans le processus de Feature Engineering constitue un levier essentiel pour améliorer les performances des modèles prédictifs. L'une des approches les plus courantes consiste à extraire des informations temporelles pertinentes, telles que le jour de la semaine, afin de capturer des tendances cycliques influençant les phénomènes étudiés.

— **Contexe et Objectif :**
De nombreuses applications reposent sur des schémas temporels récurrents. Par exemple, dans les domaines du commerce, de la finance ou du transport, les comportements des utilisateurs varient en fonction des jours de la semaine. L'extraction du jour à partir d'un horodatage permet d'intégrer cette information dans un modèle prédictif, facilitant ainsi l'identification de patterns sous-jacents. L'objectif de cette étude de cas est d'évaluer l'impact de cette transformation sur la performance d'un modèle de Machine Learning.

— **Prétraitement des Données :**
Les données utilisées comportent une variable temporelle représentant, par exemple, la date d'une transaction ou d'un événement. La première étape consiste à convertir cette variable en un format exploitable, puis à en extraire le jour de la semaine sous forme numérique (0 pour lundi, 6 pour dimanche) ou catégorielle ("Lundi", "Mardi", etc.).
Extraction en Python :

```python
import pandas as pd

# Chargement des données avec une colonne de dates
df = pd.DataFrame({'date_transaction': ['2025-04-01', '2025-04-02', '2025-04-03']})

# Conversion en format datetime
df['date_transaction'] = pd.to_datetime(df['date_transaction'])

# Extraction du jour de la semaine (format texte et numérique)
df['jour_semaine'] = df['date_transaction'].dt.day_name()  # Exemple : "Mardi"
df['jour_semaine_num'] = df['date_transaction'].dt.weekday  # Exemple : 1 pour mardi

print(df)
```

Ln: 13, Col: 10

Run | Share | $ | Command Line Arguments

```
  date_transaction jour_semaine  jour_semaine_num
0       2025-04-01      Tuesday                 1
1       2025-04-02    Wednesday                 2
2       2025-04-03     Thursday                 3


** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Cette transformation permet au modèle d'intégrer des informations temporelles pertinentes et d'exploiter la périodicité des données.

— **Impact sur la Modélisation :**

L'ajout de cette feature joue un rôle clé dans la capacité des algorithmes à mieux comprendre les tendances et comportements cycliques. Par exemple, dans une problématique de prévision des ventes, la prise en compte du jour de la semaine permet de mieux modéliser les variations de demande.
Dans un cadre supervisé, cette variable peut être traitée de différentes manières :
- Encodage catégoriel : One-Hot Encoding pour transformer chaque jour en une variable binaire distincte.
-Encodage ordinal : Conservation du jour sous sa forme numérique (0 à 6).
-Périodicité sinusoïdale : Transformation en variables sinusoïdales et cosinusoïdales pour mieux capturer les cycles temporels.

— **Évaluation des Performances :** Afin de mesurer l'impact de cette transformation, une comparaison est effectuée entre un modèle de base et un modèle intégrant la nouvelle feature. L'évaluation repose sur des métriques adaptées au problème étudié, telles que :
- Pour un modèle de régression : RMSE (Root Mean Squared Error), MAE (Mean Absolute Error).
- Pour un modèle de classification : Accuracy, F1-score.

Chapitre 3

Évaluation et Impact du Feature Engineering

Le Feature Engineering joue un rôle déterminant dans l'optimisation des modèles de Machine Learning. Son impact est évalué à travers des critères de performance, de complexité et de généralisation du modèle. Une analyse rigoureuse permet d'identifier les transformations les plus efficaces pour améliorer la précision du modèle tout en réduisant le risque de sur-apprentissage et en optimisant la vitesse d'entraînement.

3.1 Comparaison des Performances

L'évaluation des performances après le Feature Engineering se fait à l'aide de métriques adaptées au type de problème. Pour la classification, on utilise des scores comme Accuracy et F1-Score, tandis que pour la régression, le RMSE est couramment utilisé. Une amélioration de ces scores indique une meilleure performance du modèle. Le Feature Engineering contribue également à réduire le sur-apprentissage en améliorant la généralisation, ce qui peut être vérifié par la validation croisée. Enfin, l'utilisation de techniques comme le PCA ou la suppression de features inutiles permet d'accélérer l'entraînement et de réduire la consommation de mémoire.

3.2 Recommandations et Meilleures Pratiques

Une approche rigoureuse du Feature Engineering repose sur une expérimentation méthodique et des bonnes pratiques visant à garantir l'efficacité et la robustesse du modèle.

Tester plusieurs transformations : Appliquer différentes transformations de features (encodage, normalisation) et évaluer leur impact sur les performances du modèle.

Validation croisée : Utiliser des techniques de validation croisée pour garantir que le modèle se généralise bien et pour éviter le sur-ajustement. -

Éviter l'introduction de biais : Exclure les features trop spécifiques et éviter les données susceptibles de causer une fuite d'information, ce qui risquerait de fausser les résultats.

Conclusion

Le Feature Engineering constitue une étape fondamentale dans le développement de modèles de Machine Learning performants. En transformant les données brutes en variables exploitables, cette phase permet non seulement d'améliorer la précision des prédictions, mais aussi de renforcer la capacité des modèles à généraliser sur des données inconnues, tout en réduisant les risques de sur-apprentissage.

À travers ce projet, nous avons détaillé les différentes approches et techniques de Feature Engineering, telles que la sélection, l'extraction et la transformation des features, en soulignant leur impact direct sur les performances des modèles. Les résultats obtenus démontrent clairement que l'application judicieuse de ces méthodes peut entraîner des améliorations substantielles en termes de précision, de vitesse d'entraînement et d'efficacité computationnelle, tout en simplifiant la complexité des modèles.

Il est cependant crucial de souligner l'importance de suivre des pratiques rigoureuses, telles que l'expérimentation de plusieurs transformations de features, la validation croisée et la gestion des biais, afin d'assurer la robustesse et la fiabilité des modèles. En évitant les pièges courants, comme l'introduction de features trop spécifiques ou la fuite d'information, il est possible de maximiser la performance du modèle tout en garantissant sa capacité à s'adapter à de nouvelles données.

En conclusion, le Feature Engineering est une discipline incontournable pour tirer parti du potentiel des modèles de Machine Learning. Lorsqu'il est appliqué de manière stratégique et systématique, il permet non seulement d'améliorer la qualité des prédictions, mais aussi d'optimiser le processus global de modélisation, faisant du Feature Engineering un facteur clé de succès dans toute application de Machine Learning.