Master Big data & à la décision S2

École Nationale des Sciences Appliquées de Khouribga

Decision Tree Regression

24 avril 2023

Encadrée par :

Pr : OURDOU Amal

Réaliser par :
ELAADERAOUI Yassine
KOUALIL Mohamed
MELLOUK Fatima zahrae

Table des matières

1 Introduction

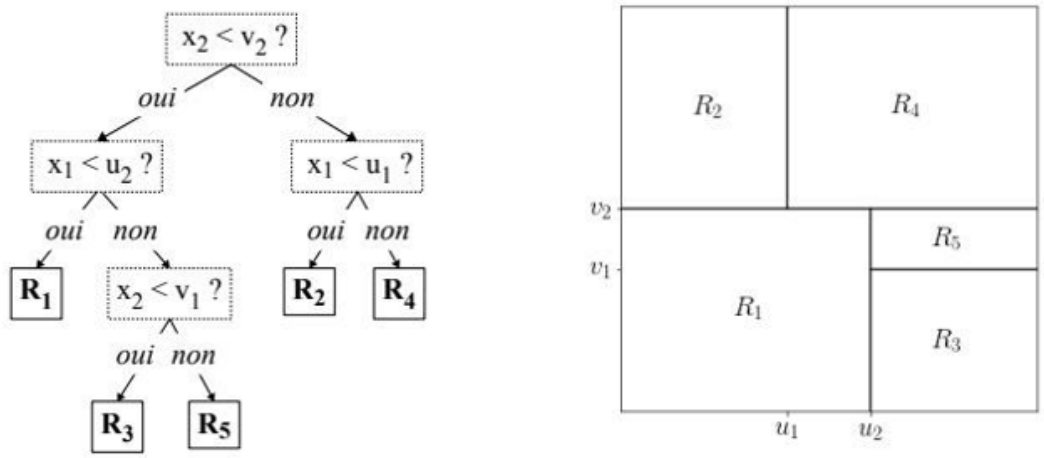
Les modèles que nous avons vus jusqu'à présent procèdent en une seule opération pour étiqueter une observation x. De tels modèles utilisent le même ensemble de variables pour toutes les classes, et leur accordent la même importance pour toutes les observations. Ces modèles ne sont pas bien adaptés aux cas ou les classes ont une distribution multi-modale, c'est-à-dire qu'elles sont déterminées par différentes variables dans différentes régions de l'espace, ou quand les variables sont discrètes. Par contraste, les arbres de décision sont des modèles hiérarchiques, qui se comportent comme une série successive de tests conditionnels, dans laquelle chaque test dépend de ses antécédents. Ils sont couramment utilisés en dehors du monde du machine Learning, par exemple pour décrire les étapes d'un diagnostic d'un choix de traitement pour un médecin, ou les chemins possibles dans un « livre dont vous êtes le héros ».

2 Régression arbre de décision

2.1 Définition

Regression Arbres de Decision est une puissante technique d'apprentissage automatique qui utilise des arbres de décision pour prédire le résultat d'un ensemble de données donné. Il est utilisé à la fois pour les problèmes de classification et de régression. L'objectif de l'arbre de décision est de créer un modèle qui prédit la valeur d'une variable cible en apprenant des règles de décision simples déduites des caractéristiques des données. L'arbre de décision est composé d'une série de nœuds, chaque nœud représentant une décision ou une prédiction. Le modèle est construit en divisant les données en sous-ensembles plus petits en fonction de la valeur d'un certain attribut. Le processus est répété jusqu'à ce que toutes les données soient divisées en groupes homogènes et que la variable cible soit prédite.

2.2 Principe



Comment faire pousser un arbre ?

Pour entraîner un arbre de décision ,il existe plusieurs techniques performantes ,mais puisque on travaille

sur la régression alors ,on va consacrer notre travail sur quelques techniques de ce type. La technique la plus connue c'est CART (Classification And Regression Tree).

Il s'agit d'un algorithme de partitionnement de l'espace par une approche gloutonne, récursive et divisive.

- Pour entraîner un arbre de décision ,il existe plusieurs techniques performantes ,
- La technique la plus connue pour la regression , c'est l'algorithme CART (Classification And Regression Tree).

- Il s'agit d'un algorithme de partitionnement de l'espace par une approche gloutonne, récursive et divisive.

L'algorithme CART nécessite 3 composants :

- Définir un critère pour sélectionner la meilleure partition .
- Une règle pour décider quand un nœud est terminal, c'est-à-dire qu'il devient un feuille.
- Tailler l'arbre pour éviter le sur-apprentissage.

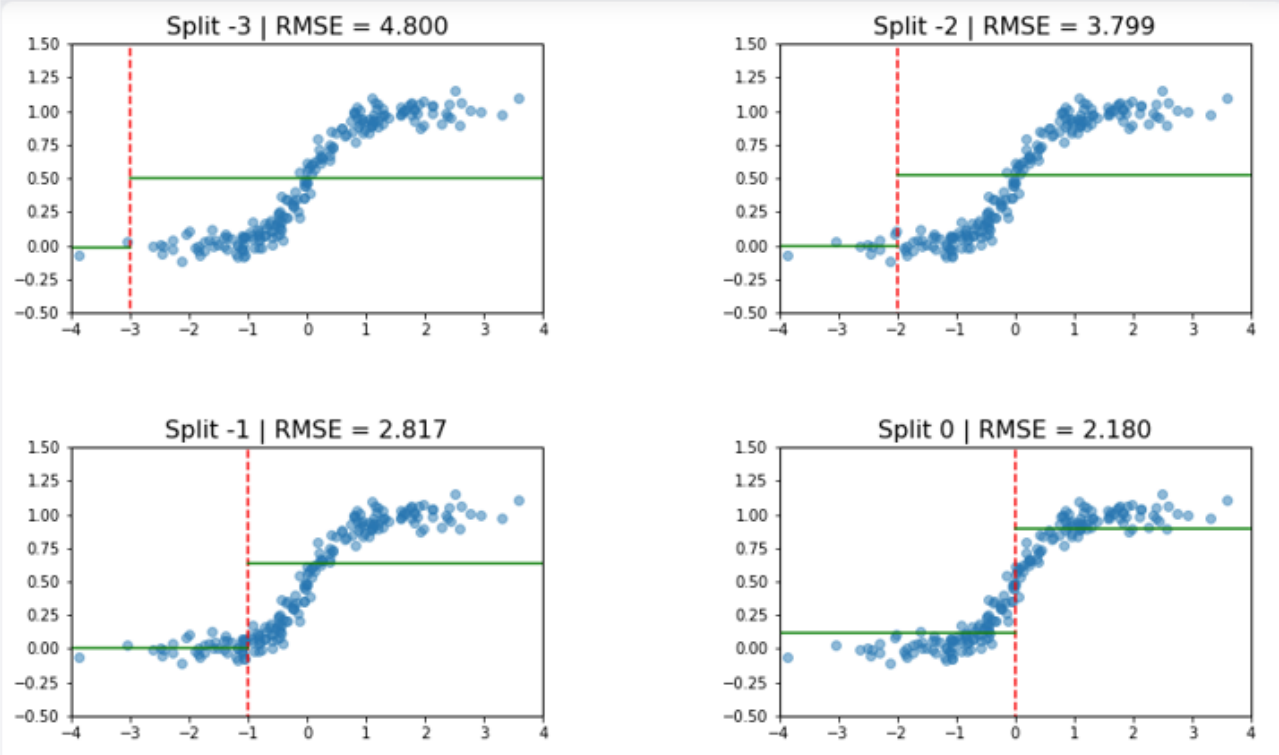
Sélection de la meilleure partition

On appellera le nombre split la valeur où l'on sépare en deux l'espace. Cette valeur est déterminée par CART en utilisant la MSE.c'est a dire on choisit la valeur qui minimise MSE.

La formule de la MSE est la suivante :

$$\sum_{j=1}^{j} \sum_{x_i \in R_m} (y_i - y_{Rj})^2 \tag{1}$$

y_{Rj} :la réponse moyenne pour les observations d'entraînement dans la jème région.



Remarque

Dans le cas d'une variable continue x_j, si l'on suppose les valeurs prises par cette varibale dans D ordonnées :

$$x_j^1 \leq x_j^2 \leq ..., \leq x_j^n$$

alors les valeurs possibles de s sont $\frac{x_j^{i+1} - x_j^i}{2}$ pour toutes les valeurs de i telles que $x_j^{i+1} \neq x_j^i$ **Règles d'arrêt** minsplit : pour éviter de créer des fractionnements qui petites feuilles, le nombre minimum d'observations qui doit exister dans un nœud pour qu'une scission soit tentée (minsplit = 20).

— Profondeur maximale

— Nombre minimal d'échantillons par nœud

— Nombre maximal de feuilles

Le sur-apprentissage :

Parmi Ces hyper-paramètres

— La profondeur maximale (max_depth)

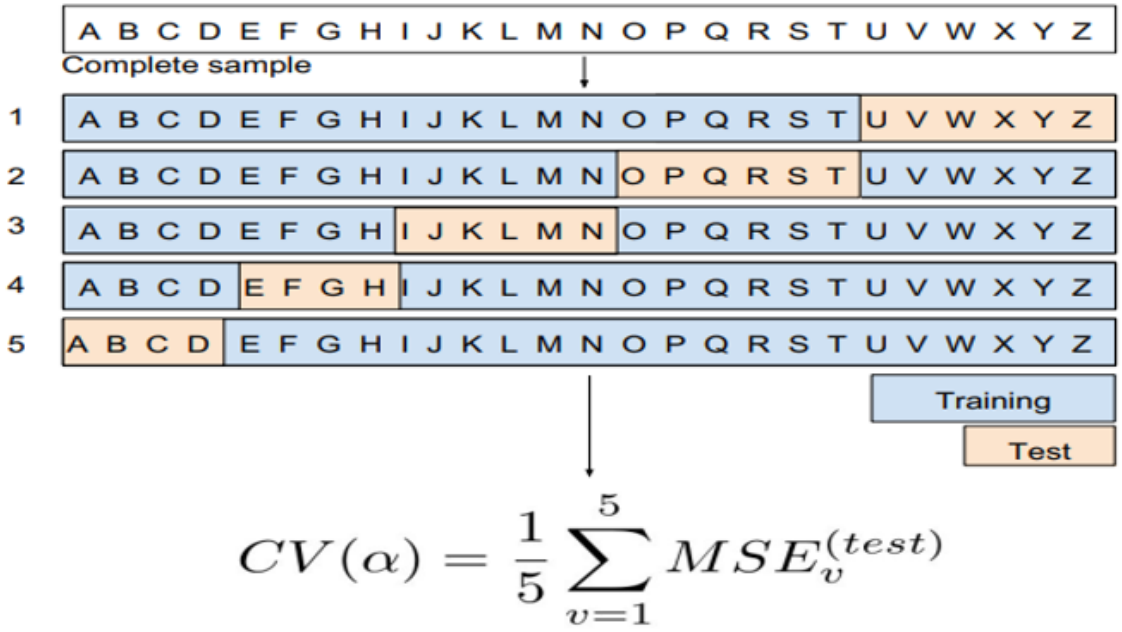
— Le nombre d'observations minimal dans un nœud pour effectuer un split.

— Nous cultivons d'abord le plus grand arbre possible Tmax puis le taillons retour afin d'obtenir un sous-arbre.

— Pour chaque valeur de , il existe un sous-arbre $T \subset Tmax$ qui minimise :
$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - y_{Rj})^2 + \alpha |T|$

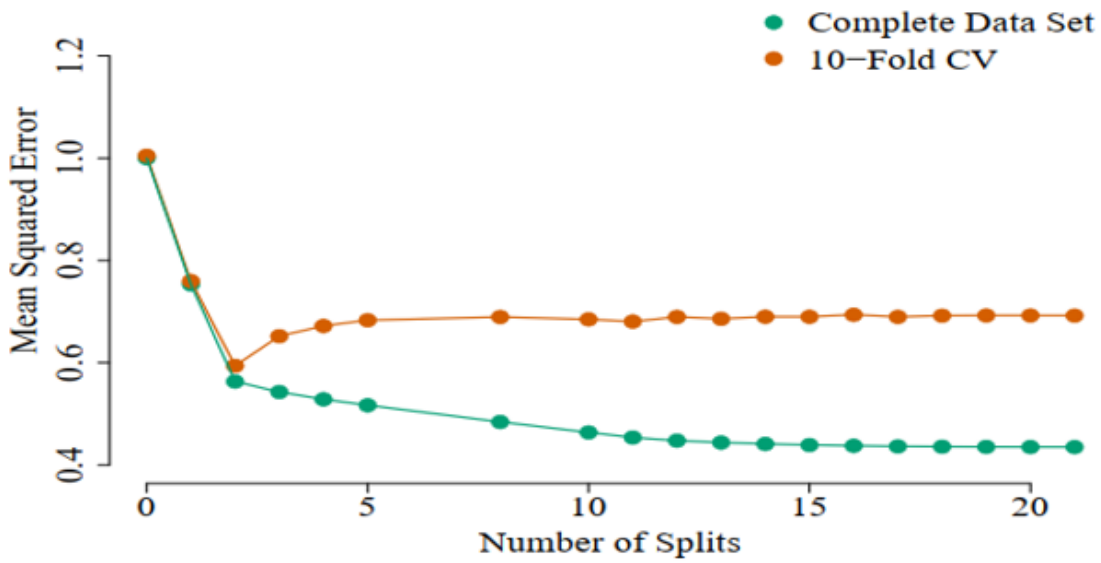
— |T| indique le nombre de nœuds terminaux de l'arbre T, Rm est le rectangle correspondant à la même feuille, et ŷRm est la réponse prédite associée à Rm.

— α est choisi en utilisant la validation croisée v-fold.



$$CV(\alpha) = \frac{1}{5} \sum_{v=1}^{5} MSE_v^{(test)}$$

Over-fitting :



3 Applications de la régression Arbres de décision

Les arbres de décision de régression peuvent être utilisés dans une variété d'applications. Il peut être utilisé pour l'analyse prédictive, comme la prévision de l'attrition des clients ou la prévision des cours des actions. Il peut également être utilisé pour des tâches de classification, telles que l'identification de transactions frauduleuses ou la classification d'images. Il peut également être utilisé pour les systèmes de recommandation, tels que la recommandation de films ou de produits. Les arbres de décision de régression peuvent également être utilisés pour des problèmes d'optimisation, tels que la recherche de l'itinéraire optimal entre deux points. Il peut également être utilisé pour la détection d'anomalies, comme l'identification de modèles inhabituels dans les données. Enfin, il peut être utilisé pour la sélection d'entités, comme la sélection des entités les plus importantes dans un jeu de données.

4 Exemple introductif

Prédiction des salaires des ligues majeures de baseball.

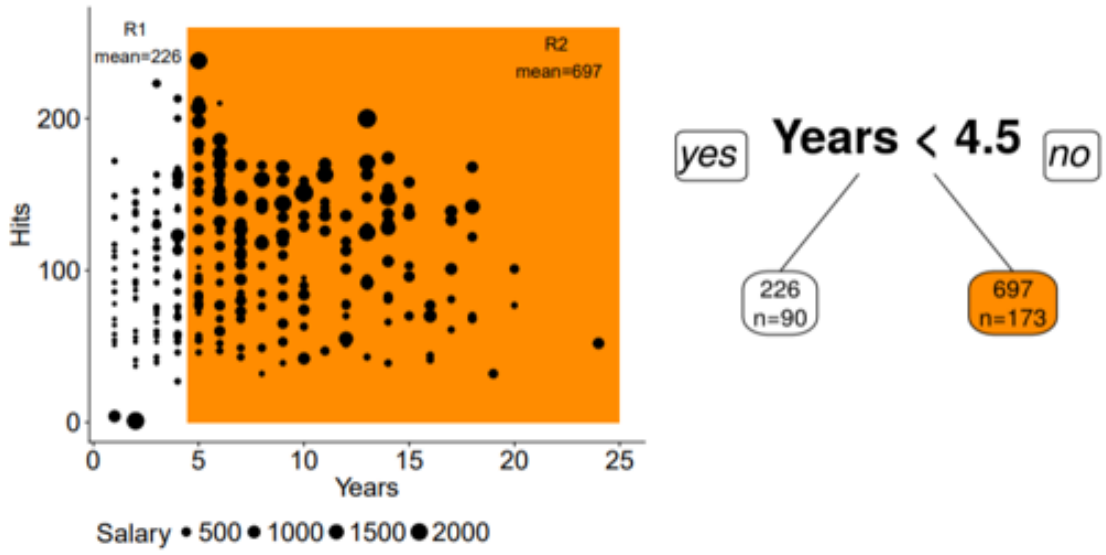
- variable cible
  - y : le salaire annuel en 1987 (variable cible y)
- Predictor variables :
  - X1 : le nombre d'années passées dans les ligues majeures
  - X2 : le nombre de coups (hits) réussis en 1986
- objectif prédire le salaire annuel au début de la saison de baseball de 1987 en utilisant les variables prédictives(years , hits).

Les données :
Un exemple de ce à quoi ressemblent les données.

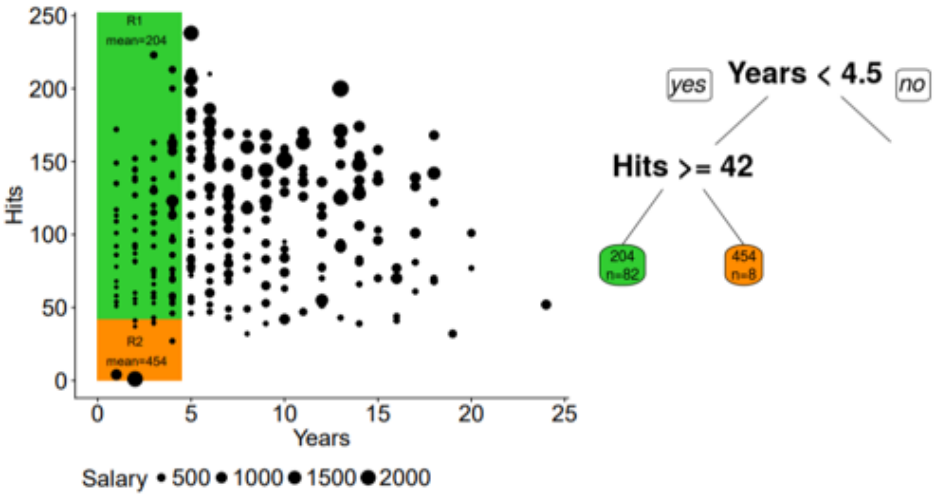
	Years	Hits	Salary
-Andre Dawson	11	141	500
-Andres Galarraga	2	87	92
-Barry Bonds	1	92	100
-Cal Ripken	6	177	1350
-Gary Carter	13	125	1926
-Joe Carter	4	200	250
-Ken Griffey	14	150	1000
-Mike Schmidt	2	1	2127
-Tony Gwynn	5	211	740

Une représentation visuelle des donnée

La Première division



Deuxième division



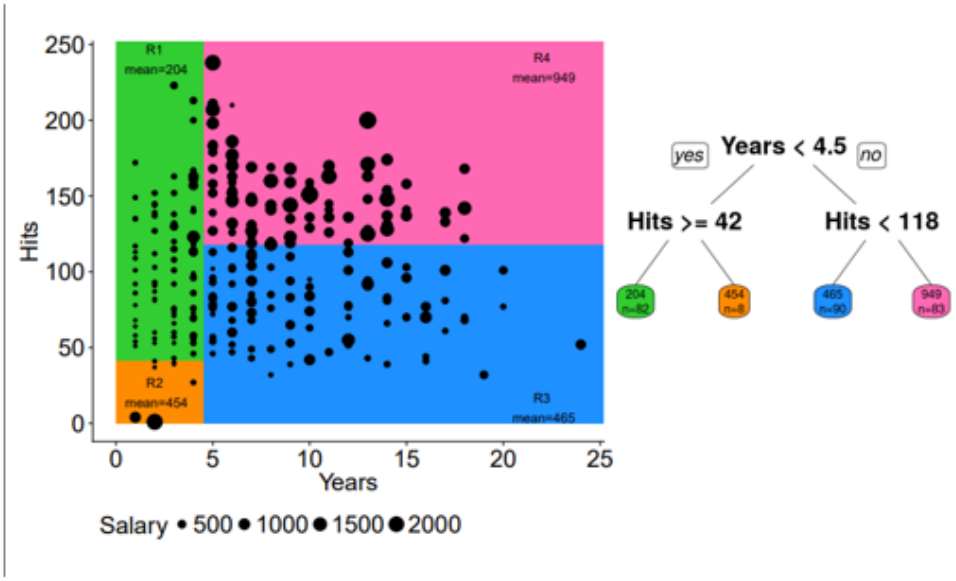
Une erreur dans les données :

Les données indiquent qu'il a joué seulement 2 ans et a eu 1 coup sûr en 1987, ce qui est incorrect.

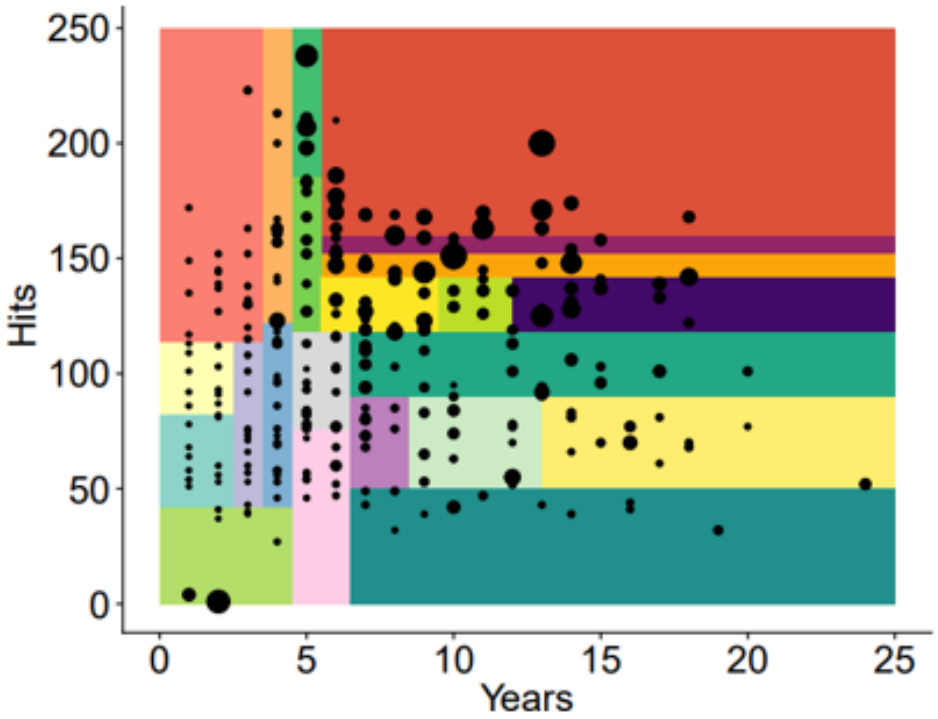
	Years	Hits	Salary
-Andre Dawson	11	141	500.00
-Andres Galarraga	2	87	91.50
-Barry Bonds	1	92	100.00
-Cal Ripken	6	177	1350.00
-Gary Carter	13	125	1925.57
-Joe Carter	4	200	250.00
-Ken Griffey	14	150	1000.00
-Mike Schmidt	2	1	2127.33
-Tony Gwynn	5	211	740.00

- Mike Schmidt started his career in 1972, and was inducted into the Baseball Hall of Fame in 1995.

Deuxième division



Et si nous continuons...



5 Avantages des arbres de décision de régression

Regression Arbres de Decision offre un certain nombre d'avantages par rapport aux autres techniques d'apprentissage automatique. Il est simple à comprendre et à interpréter et peut facilement gérer des données numériques et catégorielles. Il est également capable de gérer des relations non linéaires entre les variables et peut gérer de grands ensembles de données avec des valeurs manquantes. De plus, il résiste aux valeurs aberrantes et peut être utilisé pour la sélection des caractéristiques. Regression Arbres de Decision est également efficace sur le plan informatique et peut être implémenté dans un certain nombre de langages de programmation. Il est également capable de gérer des interactions complexes entre des variables et peut être utilisé pour identifier des caractéristiques importantes dans les données. Enfin, il est robuste au

surajustement et peut être utilisé pour créer des modèles puissants avec une grande précision.

6 Limites des arbres de décision de régression

Bien que Regression Arbres de Decision soit une puissante technique d'apprentissage automatique, elle présente également certaines limites. Il est sujet au surajustement et peut être difficile à interpréter. Il peut également être coûteux en calcul et peut être difficile à régler. De plus, il est sensible aux petits changements dans les données et peut être sujet à des biais si les données ne sont pas correctement équilibrées. Enfin, il n'est pas adapté à l'apprentissage en ligne et peut être difficile à mettre à jour avec de nouvelles données. Malgré ces limitations, Regression Arbres de Decision reste une puissante technique d'apprentissage automatique qui peut être utilisée pour créer des modèles puissants avec une grande précision.

7 Implementation

7.1 Exploration des donnees

7.1.1 Jeu de données

Après avoir expliqué le fonctionnement de cet algorithme (Arbre de décision), maintenant c'est le temps pour décrire l'implementation de cet algorithme, et pour cela on va utilise un jeu de données depuis kaggle de 2190 enregistrements et 4 columns, ces quatre columens représentés quelques paramètres de variations de températures. L'objectif d'après cette dataset c'est de predire le temperature(CPU Package Temperature) dans un tel jour si les autres parametres se fixent dans quelques valeurs.

```
import pandas as pd
df=pd.read_csv('temperature.csv')
```
✓ 20.2s

```
df
```

	CPU Total	CPU Package Temperature	Memory	Used Memory
0	3.528023	40	48.994663	1.918564
1	5.060095	40	48.971184	1.917644
2	10.937500	40	49.110200	1.923088
3	5.078125	40	49.125004	1.923668
4	2.343750	40	49.137670	1.924164
...	...	...	...	...
2185	8.203125	48	54.053402	2.116657
2186	23.437500	48	54.217260	2.123074
2187	17.968750	48	54.272884	2.125252
2188	11.742425	48	54.247360	2.124252
2189	13.709677	48	54.241320	2.124016

2190 rows × 4 columns

7.1.2 Description du jeu de donnees

```
df.info()
✓ 0.5s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2190 entries, 0 to 2189
Data columns (total 4 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CPU Total               2190 non-null   float64
 1   CPU Package Temperature 2190 non-null   int64
 2   Memory                  2190 non-null   float64
 3   Used Memory             2190 non-null   float64
dtypes: float64(3), int64(1)
memory usage: 68.6 KB
```

Voyons s'il y a des valeurs nulles à prendre en compte.

```
df.isna().sum().sum()
✓ 0.1s
0
```

Heureusement, pas de valeurs manquent.passons suprimons s'ils y'a des enregistrements dupliques.

```
df.drop_duplicates(inplace=True)
✓ 0.1s
```

7.1.3 Outliers

Ainsi, pour identifier les valeurs hors sujet ou illogiques dans une série de données numériques, on utilise généralement la technique de détection des "outliers". Il existe plusieurs méthodes pour repérer ces valeurs erronées, mais la plus simple et la plus couramment utilisée est la méthode de boxplot.

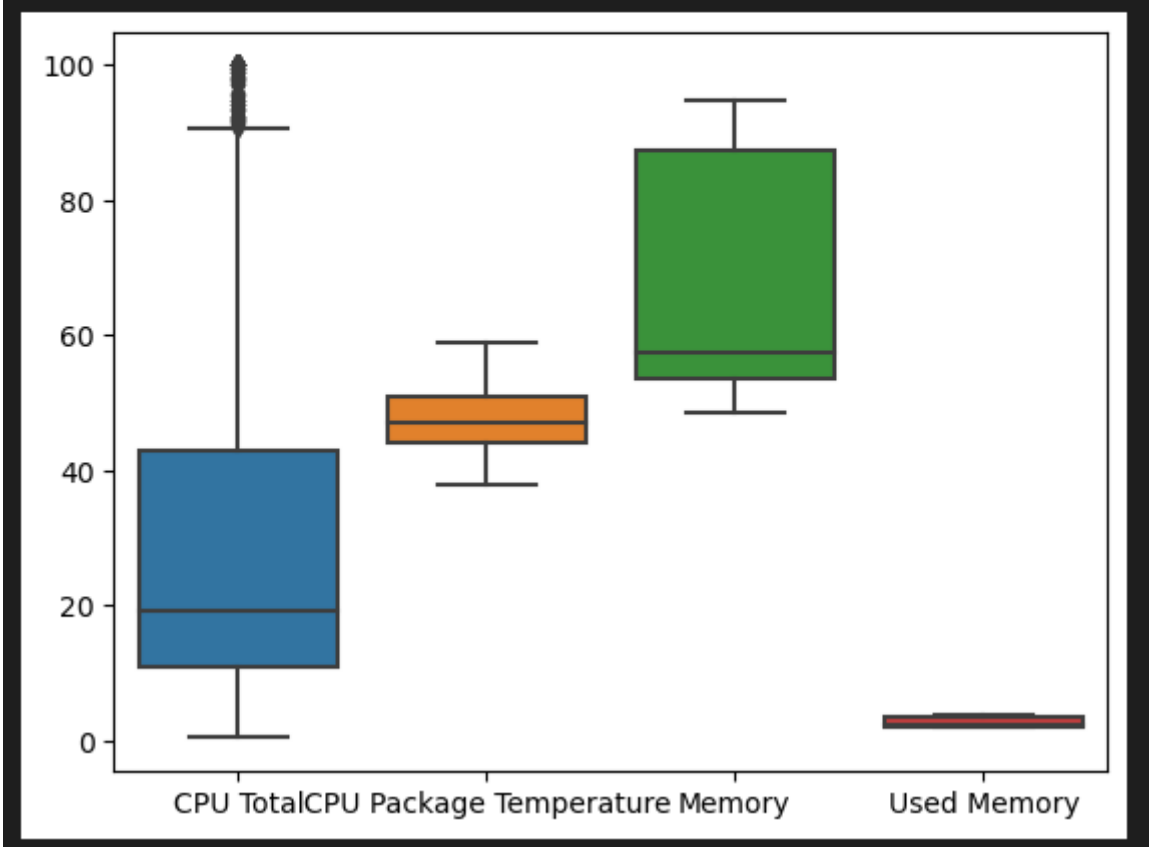
Le boxplot permet de visualiser graphiquement la distribution des données en représentant les quartiles de la distribution sous la forme d'un rectangle (la "boîte"), avec une ligne indiquant la médiane. Les "moustaches" qui s'étendent de chaque côté de la boîte représentent la dispersion des données, en excluant les valeurs considérées comme des outliers. Ces outliers sont représentés par des points ou des astérisques situés à l'extérieur des moustaches.

Pour déterminer les outliers, une règle empirique est souvent utilisée : toutes les valeurs situées à plus de 1,5 fois l'écart interquartile (c'est-à-dire la distance entre le premier et le troisième quartile) de la boîte sont considérées comme des outliers.

En somme, la méthode de boxplot est une technique simple et visuelle pour repérer rapidement les outliers dans une série de données numériques. Elle permet d'avoir une vue d'ensemble de la distribution des données et de détecter des valeurs potentiellement inhabituelles, qui pourraient fausser les résultats d'une analyse statistique.

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.boxplot(data=df)
plt.show()
```

✓ 0.7s



Comme on peut le voir, il y a plusieurs valeurs aberrantes qu'il convient de traiter. Pour cela, il existe plusieurs techniques, mais afin de ne pas perdre trop de temps, nous allons simplement les supprimer.

```
def remove_outlier_IQR(df):
    Q1=df.quantile(0.25)
    Q3=df.quantile(0.75)
    IQR=Q3-Q1
    df_final=df[~((df<(Q1-1.5*IQR)) | (df>(Q3+1.5*IQR)))]
    return df_final
```

✓ 0.1s

```
df=remove_outlier_IQR(df)
```

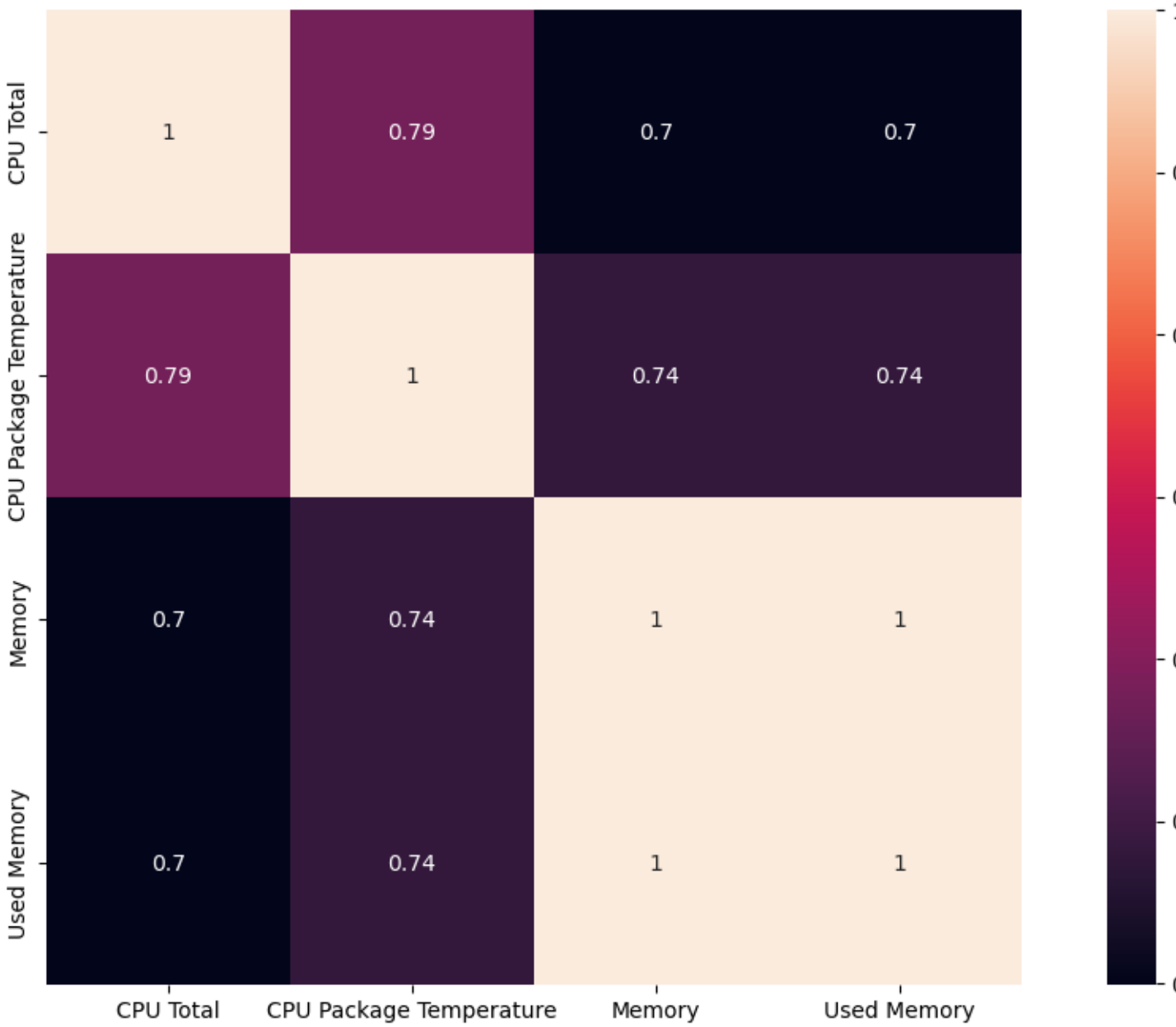
✓ 0.3s

7.2 Feature engineering

7.2.1 Correlation

```
corr=df.corr()
plt.figure(figsize = (14, 8))
sns.heatmap(corr, annot = True, square = True)
```

✓ 2.1s



En observant les caractéristiques, on remarque qu'elles sont en grande partie corrélées, ce qui suggère qu'un seul paramètre pourrait être efficace pour les prendre en compte.

7.3 Construction du model

Importation des bibliothèques et entraînement du modèle

```
from sklearn.metrics import accuracy_score,mean_absolute_error
from sklearn import tree

from sklearn.tree import DecisionTreeRegressor

regressor = DecisionTreeRegressor(random_state = 0)

regressor.fit(X_train, y_train)
```

✓ 4.9s

```
DecisionTreeRegressor
DecisionTreeRegressor(random_state=0)
```

7.4 Evaluation du modele

```
pred=regressor.predict(X_test)
pred_train=regressor.predict(X_train)
print("Train Error:", mean_absolute_error(y_train,pred_train))
print("Test Error:", mean_absolute_error(y_test,pred))
```

✓ 1.4s

```
Train Error: 0.0
Test Error: 1.4622641509433962
```

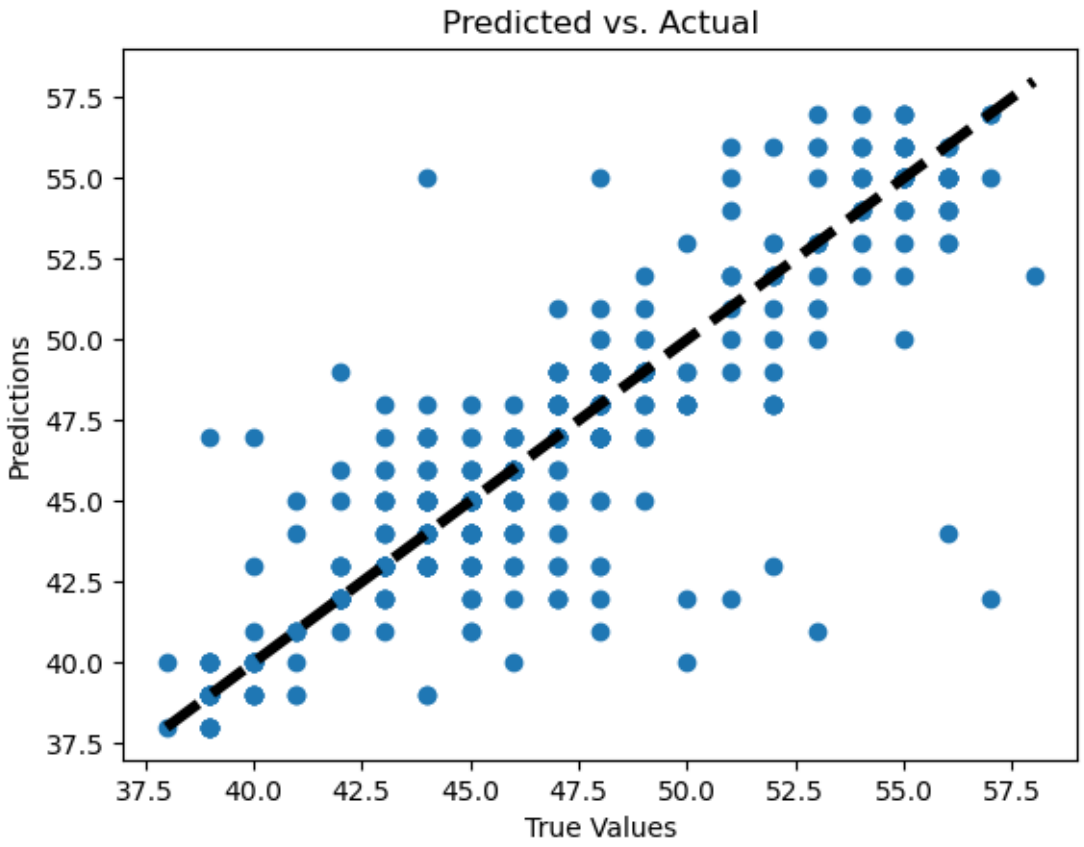
```
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

y_pred = regressor.predict(X_test)
rmse = mean_squared_error(y_test, y_pred, squared=False)
print("RMSE:", rmse)

plt.scatter(y_test, y_pred)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=4)
plt.xlabel('True Values')
plt.ylabel('Predictions')
plt.title('Predicted vs. Actual')
plt.show()
```

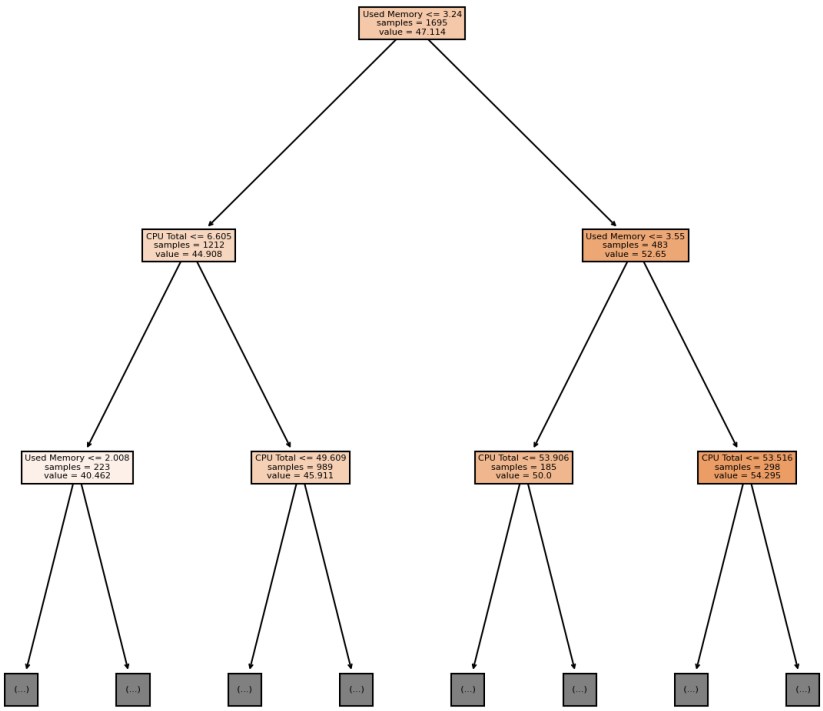
✓ 1.7s

```
RMSE: 2.4437058156021045
```



Le graphique montre que notre modèle est performant et capable de prédire la température avec une précision élevée.

7.5 Visualisation de l'arbre de decision



8 Conclusions

Regression Arbres de Decision est une puissante technique d'apprentissage automatique qui peut être utilisée à la fois pour les problèmes de classification et de régression. Il offre un certain nombre d'avantages, tels qu'être simple à comprendre et à interpréter, et être capable de gérer de grands ensembles de données avec des valeurs manquantes. Il peut être utilisé pour une variété d'applications, telles que l'analyse prédictive et les tâches de classification. Cependant, il présente également certaines limites, telles qu'une tendance au surajustement et une difficulté à interpréter. Malgré ces limitations, Regression Arbres de Decision reste une puissante technique d'apprentissage automatique qui peut être utilisée pour créer des modèles puissants avec une grande précision.