# Rapport de Projet : Prédiction des Performances des Élèves


## Réalisé Par : Oumaima EL HARTI


# 1. Introduction

## 1.1 Objectif du Projet

L'objectif de ce projet est de prédire les scores en mathématiques des élèves en utilisant des techniques d'apprentissage automatique. Cette prédiction peut aider à identifier les élèves qui pourraient avoir besoin d'un soutien supplémentaire et à comprendre les facteurs influençant leurs performances académiques.

# 2. Source des Données

## 2.1 Description du Dataset

Le dataset utilisé pour ce projet est intitulé "Students Performance in Exams" et est disponible sur Kaggle. Il contient les scores de 1000 élèves dans trois matières : mathématiques, lecture et écriture, ainsi que des informations démographiques et des données sur leur éducation.

## 2.2 Variables du Dataset

Le dataset comprend les variables suivantes :

- **gender**: sexe de l'élève
- **race/ethnicity**: groupe ethnique de l'élève
- **parental level of education**: niveau d'éducation des parents
- **lunch**: type de déjeuner (standard ou gratuit/réduit)
- **test preparation course**: suivi ou non d'un cours de préparation aux tests
- **math score**: score en mathématiques
- **reading score**: score en lecture
- **writing score**: score en écriture

# 3. Analyse Exploratoire des Données

## 3.1 Analyse Descriptive

Nous avons commencé par une analyse descriptive des données pour comprendre les distributions et les relations entre les variables.

```
import pandas as pd
import matplotlib.pyplot as plt
```

```
import seaborn as sns

# Charger le dataset
data = pd.read_csv("StudentsPerformance.csv")
# Afficher les premières lignes du dataset
print(data.head())

# Statistiques descriptives
print(data.describe())

# Distribution des scores en mathématiques
plt.figure(figsize=(10, 6))
sns.histplot(data['math score'], kde=True, color='blue')
plt.title('Distribution des scores en mathématiques')
plt.xlabel('Score')
plt.ylabel('Fréquence')
plt.show()
```

## 3.2 Visualisation des Données

Nous avons utilisé des visualisations pour explorer les relations entre les variables.

```
# Matrice de corrélation
plt.figure(figsize=(12, 8))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm')
plt.title('Matrice de corrélation')
plt.show()

# Boxplot des scores en mathématiques par niveau d'éducation des parents
plt.figure(figsize=(12, 8))
sns.boxplot(x='parental level of education', y='math score', data=data)
plt.title('Scores en mathématiques par niveau d\'éducation des parents')
plt.xlabel('Niveau d\'éducation des parents')
plt.ylabel('Score en mathématiques')
plt.xticks(rotation=45)
plt.show()
```

# 4. Prétraitement des Données

## 4.1 Encodage des Variables Catégorielles

Nous avons encodé les variables catégorielles en variables numériques pour les utiliser dans les modèles d'apprentissage automatique.

```
# Encodage des variables catégorielles
data = pd.get_dummies(data, drop_first=True)

# Séparation des caractéristiques et de la cible
X = data.drop(['math score'], axis=1)
y = data['math score']
```

## 4.2 Division des Données en Ensembles d'Entraînement et de Test

Nous avons divisé les données en ensembles d'entraînement et de test pour évaluer les modèles de manière indépendante.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

### 4.3 Normalisation des Données

Nous avons normalisé les données pour améliorer les performances des modèles.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

## 5. Modèles d'Apprentissage Automatique

### 5.1 Régression Linéaire

Nous avons utilisé une régression linéaire pour prédire les scores en mathématiques

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Entraînement du modèle
model_lr = LinearRegression()
model_lr.fit(X_train, y_train)

# Prédiction
y_pred_lr = model_lr.predict(X_test)

# Évaluation
mse_lr = mean_squared_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)

print("Régression Linéaire - MSE:", mse_lr
print("Régression Linéaire - R2 Score:", r2_lr)
```

### 5.2 Forêt Aléatoire

Nous avons également utilisé un modèle de forêt aléatoire pour comparer les performances.

```
from sklearn.ensemble import RandomForestRegressor

# Entraînement du modèle
model_rf = RandomForestRegressor(n_estimators=100, random_state=42)
model_rf.fit(X_train, y_train)
```

```
# Prédiction
y_pred_rf = model_rf.predict(X_test)

# Évaluation
mse_rf = mean_squared_error(y_test, y_pred_rf)
r2_rf = r2_score(y_test, y_pred_rf)

print("Forêt Aléatoire - MSE:", mse_rf)
print("Forêt Aléatoire - R2 Score:", r2_rf)
```

# 6. Comparaison des Modèles

## 6.1 Résultats des Modèles

Nous avons comparé les performances des modèles en termes de Mean Squared Error (MSE) et de R2 Score.

| Modèle | Mean Squared Error (MSE) | R2 Score |
|---|---|---|
| Régression Linéaire | 79.19 | 0.87 |
| Forêt Aléatoire | 45.35 | 0.92 |

## 6.2 Conclusion

Le modèle de forêt aléatoire a montré de meilleures performances que la régression linéaire en termes de MSE et de R2 Score, indiquant une meilleure capacité à prédire les scores en mathématiques des élèves.

# 7. Conclusion et Perspectives

## 7.1 Résumé

Ce projet a démontré comment les techniques d'apprentissage automatique peuvent être utilisées pour prédire les performances des élèves en mathématiques en utilisant des données démographiques et éducatives. La forêt aléatoire a été le modèle le plus performant.

## 7.2 Améliorations Futures

Pour améliorer encore les performances, nous pourrions explorer d'autres modèles, comme les réseaux de neurones, et effectuer une recherche plus approfondie des hyperparamètres. De plus, l'intégration de nouvelles données, comme des informations sur les enseignants ou des détails sur les programmes scolaires, pourrait enrichir le modèle.