Prédiction de la solvabilité des demandeurs de crédit

Travail fait par :

-Aloulou Mouad

-Alouche Mustapha

-Chahid Anas

-Azioiz Anas

-Bouchentouf Oussama

Encadré par :

-Pr.Bouchentouf Toumi

-Pr.Haja Zakaria

I. Présentation du Problème :

Les prêts bancaires jouent un rôle crucial dans le développement des activités d'investissement des banques. De nos jours, il existe de nombreux problèmes liés aux risques associés aux prêts bancaires. Grâce à leurs systèmes d'informatisation, les banques sont devenues capables d'enregistrer les données des emprunteurs.
L'objectif de ce projet est de prédire est ce que la banque doit donner un crédit à un demandeur ou non à partir de ses données.
La prédiction de la solvabilité de la personne se résume à un problème de classification. Afin d'obtenir le meilleur modèle du point de vue d'une banque, nous avons passé par 5 étapes dans lesquelles nous avons employé les différents outils et techniques du Machine Learning abordées dans le cours.

II. Description des données :

a- Dataset:

https://www.kaggle.com/datasets/altruistdelhite04/loan-prediction-problem-dataset

b-Colonnes du Dataset :

- Loan_ID : Unique Loan ID
- Gender : Male/ Female
- Married : Applicant married (Y/N)
- Dependents : Number of dependents (nombre d'enfants).
- Education : Applicant Education (Graduate/ Under Graduate)
- Self_Employed : Self employed (Y/N)
- ApplicantIncome : Applicant income
- CoapplicantIncome : Coapplicant income (salaire du conjoint)
- LoanAmount : Loan amount in thousands of dollars
- Loan_Amount_Term : Term of loan in months
- Credit_History : credit history meets guidelines yes or no
- Property_Area : Urban/ Semi Urban/ Rural
- Loan_Status : Loan approved (Y/N) this is the target variable

III.Data preprocessing :

a-Valeurs manquantes :

```
Entrée [5]: # Voir les valeurs manquantes
            df.isnull().sum().sort_values(ascending=False)
```

```
Out[5]: Credit_History       50
        Self_Employed        32
        LoanAmount           22
        Dependents           15
        Loan_Amount_Term     14
        Gender               13
        Married               3
        Loan_ID               0
        Education             0
        ApplicantIncome       0
        CoapplicantIncome     0
        Property_Area         0
        Loan_Status           0
        dtype: int64
```

Il y a un nombre important de valeurs manquantes, il n'est pas intéressant de supprimer les lignes contenant ces valeurs.
On a séparé le dataset en 2 sous bases de données une contenant les valeurs numériques(num_data) et l'autre contenant les valeurs catégoriques(cat_data).

-Pour les valeurs numériques, on remplace la valeur manquante par celle qui la précède :

```
# Pour les variables numériques on va remplacer les valeurs manquantes par la valeur précedente de la meme colonne
num_data.fillna(method='bfill',inplace=True)
num_data.isnull().sum().any()
```

```
False
```

-Pour les valeurs catégoriques, on remplace par la valeur qui se répète le plus dans la même colonne :

```
#Pour les variables catégoriques on va remplacer par les valeurs qui se répetent le plus
cat_data=cat_data.apply(lambda x:x.fillna(x.value_counts().index[0]))
cat_data.isnull().sum().any()
```

b-Transformation des valeurs catégoriques en valeurs numériques :

Afin d'entraîner nos modèles, il est indispensable d'avoir des données de même type en particulier du type numérique.Pour cela, on a remplacé les colonnes qui ont seulement deux valeurs

Exemple : {yes,no},{mâle,female},{graduate,not graduate} par 1 et 0

```
#Normaliser la BD en transformant les valeurs catégoriques en des valeurs numériques
cat_data.replace({"Graduate": 1,"Not Graduate":0}, inplace = True)
cat_data.replace({"Male": 1,"Female":0}, inplace = True)
cat_data.replace({"Yes": 1,"No":0}, inplace = True)

cat_data
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	Property_Area
0	LP001002	1	0	0	1	0	Urban
1	LP001003	1	1	1	1	0	Rural
2	LP001005	1	1	0	1	1	Urban
3	LP001006	1	1	0	0	0	Urban
4	LP001008	1	0	0	1	0	Urban
...	...	...	...	...	...	...	...
609	LP002978	0	0	0	1	0	Rural
610	LP002979	1	1	3+	1	0	Rural
611	LP002983	1	1	1	1	0	Urban
612	LP002984	1	1	2	1	0	Urban
613	LP002990	0	0	0	1	1	Semiurban

Puisque les colonnes Dependents et Property-Area peuvent avoir plus que deux valeurs, la solution est de procéder par la méthode **ONE-HOT-ENCODE**

```
# One Hot technique
Dependents = cat_data[["Dependents"]]
Dependents = pd.get_dummies(Dependents)

Property_Area = cat_data[["Property_Area"]]
Property_Area = pd.get_dummies(Property_Area)

# On concatène toutes ces donnés
cat_data = pd.concat([cat_data, Dependents, Property_Area], axis = 1)

# Supprime les colonnes du même nom que celles dans to_drop
to_drop = ['Dependents', 'Property_Area']
cat_data = cat_data.drop(columns=to_drop)

cat_data.head()
```

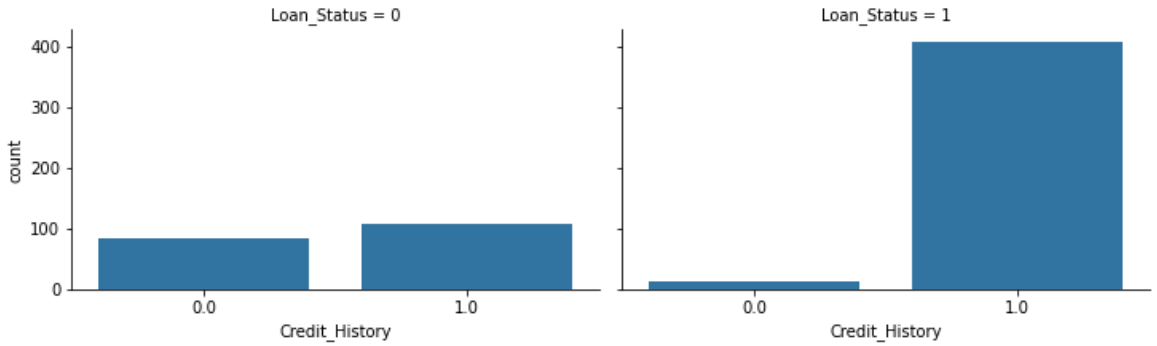
	Loan_ID	Gender	Married	Education	Self_Employed	Dependents_0	Dependents_1	Dependents_2	Dependents_3+	Property_Area_Rural	Property_Area_S
0	LP001002	1	0	1	0	1	0	0	0	0	
1	LP001003	1	1	1	0	0	1	0	0	1	
2	LP001005	1	1	1	1	1	0	0	0	0	
3	LP001006	1	1	0	0	1	0	0	0	0	
4	LP001008	1	0	1	0	1	0	0	0	0	

Après avoir rassembler les deux sous bases de données et supprimer la target(Loan-status) notre base de données principale est prête pour faire une analyse exploratoire.

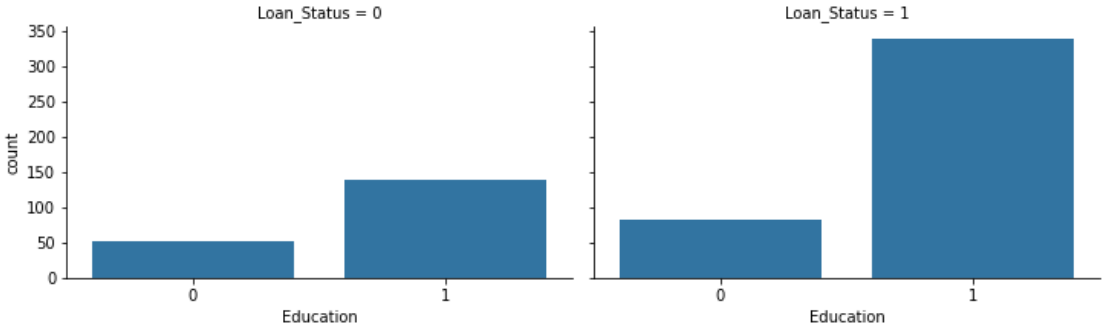
IV. Analyse Exploratoire :

Dans cette phase, nous avons essayé de visualiser l'impact des différentes variables sur la target.

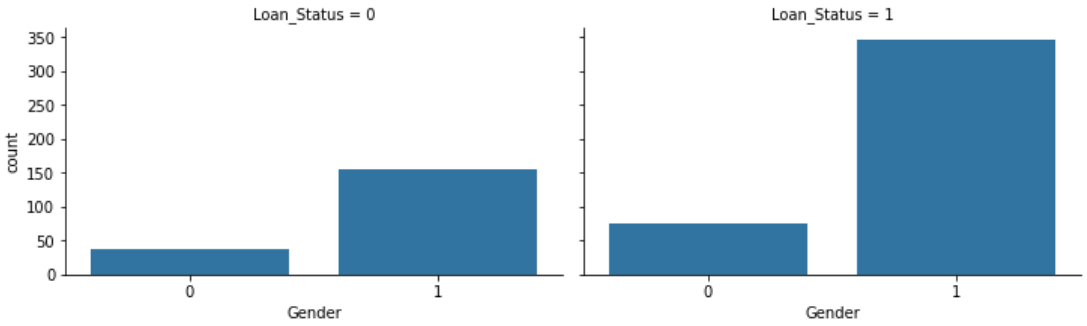
a-Credit History :



b-Education :



c-Gender :



---

d-Remarque :

On remarque qu'on le plus souvent un crédit aux personnes de sexe Masculin ainsi que ceux qui ont déjà eu un crédit et un diplôme.

V. Modèles ML et leurs performances :

Nous avons appliqué trois algorithmes de machine learning (RegressionLogistic, K-Nearest Neighbor(KNN),Decision,tree).En Analysant les matrices de confusion des 3 algorithmes, on a choisi 2 métriques pour mesurer leur performances.

Accuracy : le quotient entre le nombre de prédictions correctes et le nombre de total de prédictions permet d'évaluer d'une manière globale la performance del'algorithme.

Précision : Elle permet de bien connaitre le pourcentage de prédictions positives bien effectuées. Cette métrique reflète le pourcentage des personnes qui ont eu le crédit et que l'algorithme à réussi à prédire ce résultat.

Lorsqu'on a utilisé toutes les variables de X :

```
LogisticRegression :
accuracy-score:0.8292682926829268
precision_score:0.826530612244898
------------------------------
KNeighborsClassifier :
accuracy-score:0.6504065040650406
precision_score:0.71875
------------------------------
DecisionTreeClassifier :
accuracy-score:0.8455284552845529
precision_score:0.83
------------------------------
```

Lorsqu'on a utilisé les variables qu'on a jugées avoir plus d'impact sur la target : X2=X[['Credit_History','Education','CoapplicantIncome','Married']]

```
LogisticRegression :
accuracy-score:0.8536585365853658
precision_score:0.8383838383838383
------------------------------
KNeighborsClassifier :
accuracy-score:0.6991869918699187
precision_score:0.7307692307692307
------------------------------
DecisionTreeClassifier :
accuracy-score:0.8455284552845529
precision_score:0.83
------------------------------
```

Conclusion :

D'après ces résultats,les deux Algorithmes Logistic Regression et Decision Tree ont donné de meilleurs résultats.D'ailleurs en utilisant la base de données X2 les scores ont légèrement augmenté.