Fatoumata Barry

Rapport d’analyse de données : Factures impayées société X

Introduction :

Les factures impayées et leur récupération représentent un enjeu majeur pour toute entreprise. Une mauvaise gestion des paiements peut entraîner des problèmes de trésorerie, impactant la stabilité financière et la croissance de la société.

Plusieurs facteurs peuvent expliquer ces retards de paiement :

• La saisonnalité de l’activité,
• La solvabilité des clients,
• Le manque d’efficacité dans les relances, etc.

L’objectif d’une entreprise est donc de **minimiser le risque d’impayés** et d’**optimiser le recouvrement** des créances.

Dans cette étude, nous analyserons un ensemble de **100 factures** issues d’une société fictive appelée société X, comprenant 20 % de factures impayées et 80 % de factures payées.

Notre analyse se déroulera en deux étapes :

1. Étudier la répartition et les tendances des factures impayées et payées
2. Faire une prédiction des factures susceptibles d'être impayées

Ce rapport vise à démontrer l'impact des factures impayées sur la trésorerie des entreprises et à identifier les facteurs pouvant influencer les retards de paiement. À noter que pour une meilleure prédiction, il faudrait un panel de centaines de milliers de factures et non 100 factures. Le but étant de montrer ma compréhension du machine learning.

Code utilisé sur google colab :

```
import pandas as pd
import numpy as np
from datetime  import datetime, timedelta

np.random.seed(42)
n = 100

dates_emission = [datetime(2023, 1, 1) + timedelta(days=np.random.randint(0, 365)) for _ in range(n)]
delais = [np.random.randint(15, 60) for _ in range(n)]
dates_echeance = [dates_emission[i] + timedelta(days=delais[i]) for i in range(n)]
montants = np.random.randint(100, 5000, n)
status = np.random.choice(["Payée", "Impayée"], n, p=[0.8, 0.2])

df = pd.DataFrame({
    "Facture ID": range(1, n+1),
    "Montant": montants,
    "DateEmission": dates_emission,
    "DateEcheance": dates_echeance,
    "Statut": status,
    "Delai de Paiement": delais
    })

print(df.head())
```

```
   Facture ID  Montant DateEmission DateEcheance   Statut  Delai de Paiement
0           1     3681   2023-04-13   2023-06-10    Payée                 58
1           2     3557   2023-12-15   2024-01-06    Payée                 22
2           3     1736   2023-09-28   2023-11-05    Payée                 38
3           4     3796   2023-04-17   2023-05-12    Payée                 25
4           5     3099   2023-03-13   2023-04-13  Impayée                 31
```

Résultat dans google colab:

	Facture ID	Montant	DateEmission	DateEcheance	Statut	Delai de Paiement
0	1	3681	2023-04-13	2023-06-10	Payée	58
1	2	3557	2023-12-15	2024-01-06	Payée	22
2	3	1736	2023-09-28	2023-11-05	Payée	38
3	4	3796	2023-04-17	2023-05-12	Payée	25
4	5	3099	2023-03-13	2023-04-13	Impayée	31

Etape 1 : Analyse exploratoire des factures :

Avant d'aller plus loin dans notre étude, nous allons réaliser une analyse exploratoire des données dont le but est :

- d'identifier d'éventuelles anomalies
- mieux comprendre la répartition des factures

Nous allons faire plusieurs vérifications :

- Analyse statistique globale : avec les **moyennes** et **écarts types,** etc, pour détecter d'éventuelles disparités.
- Distribution des factures impayées et payées : identifier leur proportion et leur fréquence
- Qualité des données : vérifier la présence de valeurs manquantes, anomalies d'affichage, incohérences etc.

Pour cela nous utilisons les commandes suivantes:

```
print(df.describe())
print(df["Statut"].value_counts())
```

Résultat dans google colab :

```
       Facture ID      Montant         DateEmission         DateEcheance  \
count  100.000000   100.000000                  100                  100
mean    50.500000  2750.100000  2023-07-14 09:21:36  2023-08-22 14:52:48
min      1.000000   198.000000  2023-01-02 00:00:00  2023-02-13 00:00:00
25%     25.750000  1773.500000  2023-04-12 06:00:00  2023-05-31 06:00:00
50%     50.500000  3015.000000  2023-07-23 00:00:00  2023-08-23 00:00:00
75%     75.250000  3836.500000  2023-10-10 12:00:00  2023-11-24 12:00:00
max    100.000000  4837.000000  2023-12-30 00:00:00  2024-02-06 00:00:00
std     29.011492  1361.106847                  NaN                  NaN

       Delai de Paiement
count         100.000000
mean           39.230000
min            15.000000
25%            27.000000
50%            41.500000
75%            51.000000
max            59.000000
std            13.423825
Statut
Payée      81
Impayée    19
Name: count, dtype: int64
```

Nous allons également vérifier qu'il n'y a ni valeurs manquantes, ni d'erreur dans l'ordre de nos données.

Vérification des valeurs manquantes:

Nous avons compté le nombre de valeurs manquantes dans chaque colonne:

```
print(df.isnull().sum())
```

Résultat: Aucune colonne n'affiche de valeur manquante. Le dataset est donc complet.

Résultat dans google colab :

```
Facture ID          0
Montant             0
DateEmission        0
DateEcheance        0
Statut              0
Delai de Paiement   0
dtype: int64
```

Nous allons vérifier si les dates sont bien ordonnées : les dates d'échéance doivent être postérieures aux dates d'émission.

Pour cela nous avons compté le nombre de cas où DateEcheance < DateEmission:

```
print((df["DateEcheance"] < df["DateEmission"]).sum())
```

Le résultat est 0, ainsi les données sont bien ordonnées.

Nous avons maintenant toutes les conditions réunies pour calculer le délai de paiement, c'est-à-dire le nombre de jours entre la date d'émission et la date d'échéance de chaque facture.

Le délai de paiement est une information essentielle pour évaluer la gestion des créances d'une société et détecter les factures qui risquent d'être impayées.

```
df["DelaiPaiement"] = (df["DateEcheance"] - df["DateEmission"]).dt.days
print(df.head())
```

Interprétation : La nouvelle colonne DelaiPaiement nous permettra d'analyser la répartition des délais de paiement mais surtout d'identifier d'éventuels retards récurrents.

	Facture ID	Montant	DateEmission	DateEcheance	Statut	Delai de Paiement	DelaiPaiement
0	1	3681	2023-04-13	2023-06-10	Payée	58	58
1	2	3557	2023-12-15	2024-01-06	Payée	22	22
2	3	1736	2023-09-28	2023-11-05	Payée	38	38
3	4	3796	2023-04-17	2023-05-12	Payée	25	25
4	5	3099	2023-03-13	2023-04-13	Impayée	31	31

Dans la prochaine étape, nous utiliserons ces données pour visualiser la distribution de ces délais et mieux comprendre le comportement de paiement des clients.

Etape 2 : Analyse des données :

L'objectif est d'analyser la distribution des délais de paiement pour identifier les tendances et les périodes où les impayés sont les plus fréquents. Une analyse plus poussée aurait permis d'identifier les clients avec le plus de factures impayées.

Cela aurait pu aider à mieux cibler les actions de relance.

Cependant, le nom des clients n'est pas disponible dans ce dataset.

Néanmoins, si ces données étaient présentes, plusieurs actions préventives pourraient être mises en place comme :

- La mise en place de rappels automatisés avant la date d'échéance pour éviter les retards de paiement
- Des relances spécifiques adaptées aux profils des clients (des relances plus fréquentes pour les mauvais payeurs, etc.)

1. Distribution des délais de paiement :

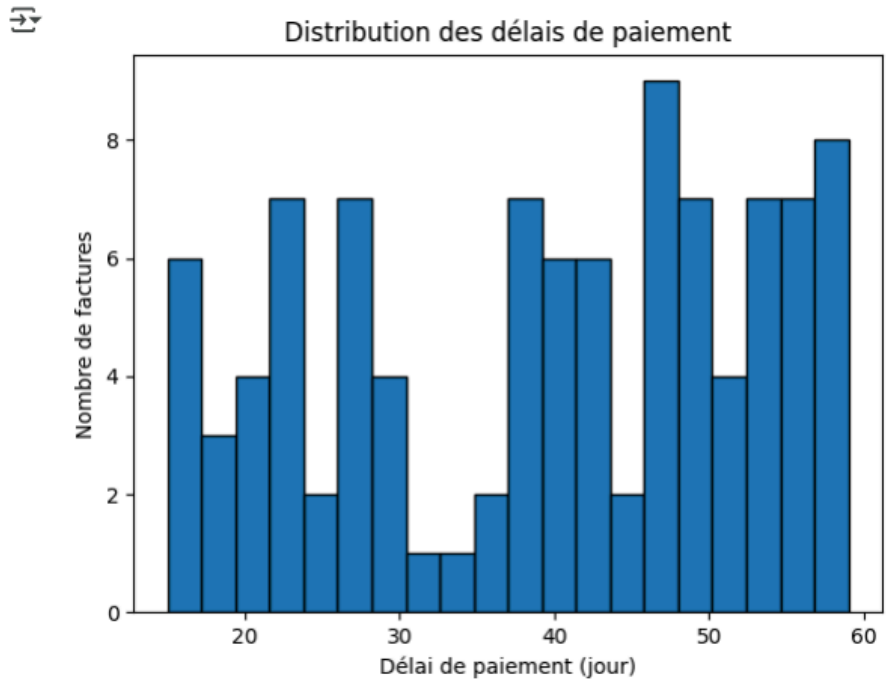
Nous allons visualiser la répartition des délais de paiement pour mieux comprendre le comportement des paiements et identifier d'éventuels retards chroniques :

Code utilisé sur google colab :

```
import matplotlib.pyplot as plt

plt.hist(df["DelaiPaiement"], bins=20, edgecolor="black")
plt.xlabel("Délai de paiement (jour)")
plt.ylabel("Nombre de factures")
plt.title("Distribution des délais de paiement")
plt.show()
```

Résultat :



Interprétation :

L'histogramme ci-dessus montre la distribution des délais de paiement pour toutes les factures (payées et impayées) :

- Pour mieux comprendre les retards, il serait pertinent d'analyser séparément les factures impayées et celles qui ont été réglées mais avec du retard.
- Cette tendance suggère tout de même des difficultés dans la gestion du recouvrement dans la société X et un impact potentiel sur sa trésorerie.

2. Analyse des factures impayées par mois :

L'objectif est d'identifier les mois où le nombre de factures impayées est le plus élevé afin de mieux comprendre les tendances et d'anticiper les périodes à risque.

Méthodologie : Nous avons créé une nouvelle colonne "MoisEmisison" qui correspond au mois d'émission de chaque facture.

Ensuite nous avons comptabilisé le nombre de factures impayées pour chaque mois.

Observation : on remarque que les mois d'août à novembre enregistrent un nombre important de factures impayées.

Une explication possible pourrait être la fermeture estivale de la société en août, ce qui ralentirait la gestion des paiements.

Une autre hypothèse serait la clôture budgétaire de certaines entreprises clientes en fin d'année, ce qui pourrait décaler les paiements vers l'année suivante.

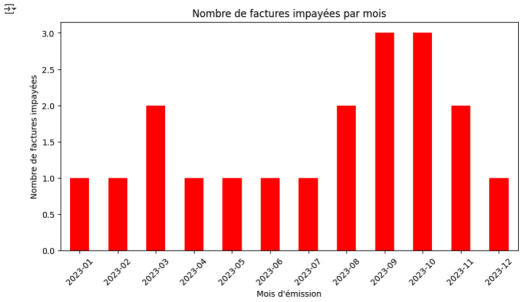
Code utilisé sur google colab:

```
df["MoisEmission"] = df["DateEmission"].dt.to_period("M")

impayes_par_mois = df[df["Statut"] == "Impayée"].groupby("MoisEmission")["Facture ID"].count()
impayes_par_mois.plot(kind="bar", color="red", figsize=(10, 5))

plt.xlabel("Mois d'émission")
plt.ylabel("Nombre de factures impayées")
plt.title("Nombre de factures impayées par mois")
plt.xticks(rotation=45)
plt.show()
```

Résultat :



Axes d'amélioration possibles :

- Analyse des tendances sur plusieurs années : Y a-t-il une récurrence annuelle des impayés sur ces périodes ?
- Comparaison avec le volume total des factures émises : Un pic de factures impayées est-il lié à une augmentation du nombre total de factures ?
- Examen des délais de paiement des clients : Certains clients prennent-ils plus de temps à payer en fin d'année ?

Étape 3 : Prévision des délais de paiement avec une régression linéaire

L'objectif est de créer un modèle de régression linéaire afin de prédire le délai de paiement d'une facture en fonction du mois d'émission.

L'idée est d'analyser s'il existe une relation linéaire entre le mois où la facture est émise et le délai moyen de paiement.

Méthodologie:

1. Transformation des données temporelles :
- Les dates ne peuvent pas être directement utilisées dans un modèle de machine learning.
- Nous transformons donc la date d'émission en une variable numérique : le mois d'émission (MoisEmission).
2. Séparation des données :
- X = MoisEmission (car nous cherchons à voir si le mois d'émission influence le délai de paiement).
- y = DelaiPaiement (c'est la variable que nous cherchons à prédire).
- Nous divisons les données en ensemble d'entraînement (80%) et ensemble de test (20%) pour évaluer la performance du modèle.

Code utilisé dans google colab :

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

df['MoisEmission'] = df['DateEmission'].dt.month

X = df[['MoisEmission']]
y = df['DelaiPaiement']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Etape 4 : Création et évaluation du modèle de régression linéaire:

1. Construction du modèle :

Nous avons choisi la régression linéaire afin d'analyser l'impact du mois d'émission sur le délai de paiement.

Le modèle est entraîné sur les données de l'échantillon d'apprentissage:

```
model = LinearRegression()
model.fit(X_train, y_train)
```

Ce processus permet au modèle d'identifier une tendance entre le mois d'émission d'une facture et son délai de paiement.

2. Prédiction et visualisation:

Une fois le modèle entraîné, nous réalisons des prédictions sur le jeu de test:

```
y_pred = model.predict(X_test)
```

Ce processus permet au modèle d'identifier une tendance entre le mois d'émission d'une facture et son délai de paiement.

Nous comparons ensuite ces prédictions aux données réelles grâce à une visualisation graphique:

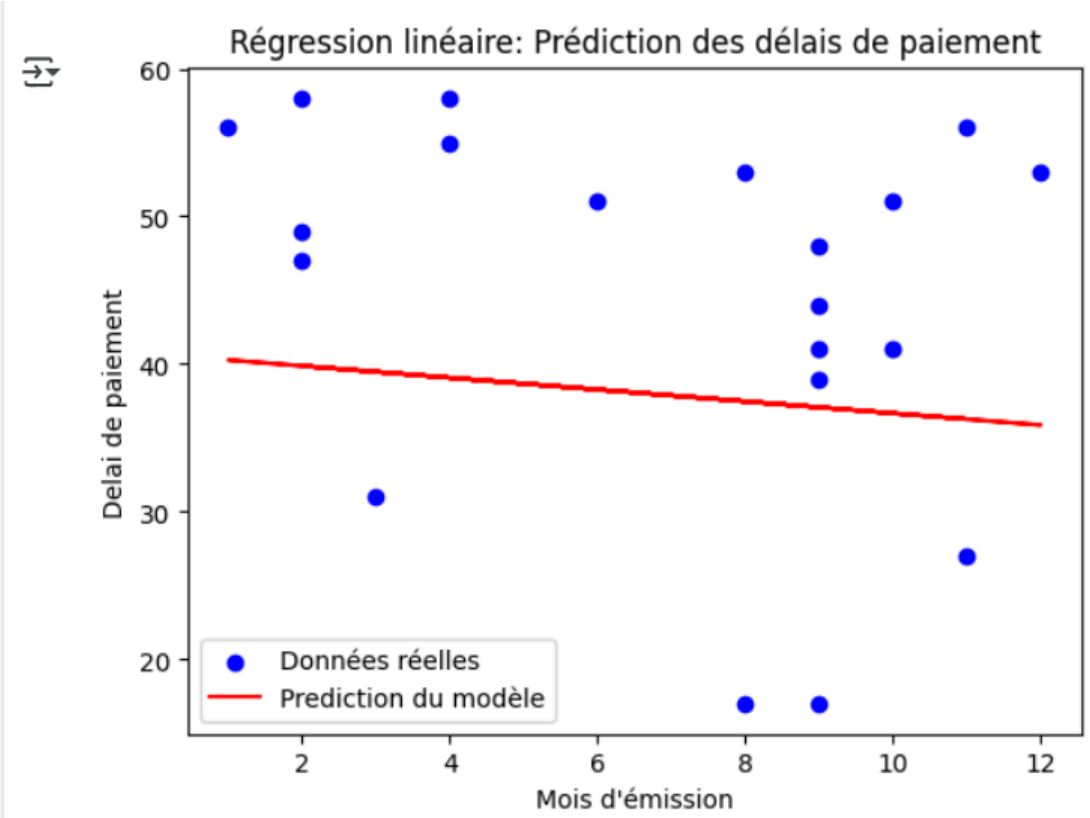
```
plt.scatter(X_test, y_test, color='blue', label='Données réelles')
plt.plot(X_test, y_pred, color='red', label='Prediction du modèle')
plt.xlabel('Mois d\'émission')
plt.ylabel('Delai de paiement')
plt.title('Régression linéaire: Prédiction des délais de paiement')
plt.legend()
plt.show()


mse = mean_squared_error(y_test, y_pred)
print(f"Erreur quadratique moyenne (MSE): {mse}")
```

Les points bleus représentent les délais de paiement réels des factures du jeu de test.

La courbe rouge représente les prédictions du modèle.

Ainsi, plus la courbe suit la tendance des points bleus, plus le modèle est performant.



Erreur quadratique moyenne (MSE): 189.95391936317196

3. <u>Limites du modèle et évaluation des performances :</u>

Le modèle est évalué à l'aide de l'erreur quadratique moyenne (MSE-Mean squared error) qui mesure l'écart entre les valeurs prédites et les valeurs réelles :

```
mse = mean_squared_error(y_test, y_pred)
print(f"Erreur quadratique moyenne (MSE): {mse}")
```

Une MSE faible indique que le modèle est capable d'effectuer des prédictions précises.

Cependant, une MSE élevée signifie que la relation entre les variables est peu fiable.

Dans notre cas, le faible nombre de factures (100) constitue une limite importante, il faudrait un plus gros volume de factures pour un modèle prédictif plus fiable.

Conclusion :

Cette analyse des délais de paiement nous a permis d'identifier plusieurs tendances et d'apporter des pistes d'amélioration pour la gestion des factures impayées.

Tout d'abord, nous avons mis en évidence la distribution des délais de paiement, révélant que certaines factures présentent des retards importants, pouvant dépasser 50 jours.

Ensuite, l'étude des factures impayées par mois a montré des périodes critiques, notamment entre août et novembre, ce qui peut être lié à des facteurs saisonniers tels que la fermeture estivale ou la gestion des budgets en fin d'année.

Afin de mieux anticiper ces retards, nous avons testé un modèle de régression linéaire basé sur la date d'émission des factures.

Bien que cette approche ait permis d'observer une certaine tendance, la fiabilité des prédictions reste limitée en raison du faible volume de données et du nombre restreint de variables explicatives.

Recommandations et perspectives :

- Optimisation du suivi des paiements : mettre en place des rappels automatisés et des relances spécifiques selon le profil des clients.
- Affiner l'analyse prédictive : intégrer davantage de variables (historique des paiements clients, montant des factures, secteur d'activité, etc.) pour améliorer la performance du modèle.
- Augmentation du volume de données : collecter des informations sur une période plus longue afin d'obtenir des tendances plus représentatives et exploitables.

En conclusion, cette étude offre une première approche pour mieux comprendre les comportements de paiement et anticiper les retards.

Cependant, une analyse plus approfondie et un enrichissement des données seront nécessaires pour construire un modèle prédictif réellement performant.