Voici une explication détaillée de vos résultats, comme si je l'expliquais à un débutant :

**1. Comprendre les métriques d'évaluation**

**Précision (Precision)**

* **Définition** : Parmi tous les prêts que le modèle a prédit comme "refusés" (classe 0), combien étaient vraiment refusés ?
* **Exemple** :
  + Régression Logistique : 67% → Quand elle dit "refus", elle a raison 67% du temps
  + Random Forest : 65% → Un peu moins bonne

**Rappel (Recall)**

* **Définition** : Parmi tous les vrais prêts refusés, combien le modèle a-t-il correctement identifiés ?
* **Exemple** :
  + Régression Logistique : 67% → Elle détecte 67% des refus réels
  + Random Forest : 62% → Légèrement moins bon

**F1-Score**

* **Définition** : Moyenne harmonique entre précision et rappel (bon indicateur global)
* **Pour les refus (classe 0)** :
  + Régression Logistique : 0.67
  + Random Forest : 0.64

**Exactitude (Accuracy)**

* **Définition** : Pourcentage total de prédictions correctes
* **Résultats** :
  + Régression Logistique : 79%
  + Random Forest : 78%

**2. Analyse des Matrices de Confusion**

**Régression Logistique**

|  | **Prédit Refusé** | **Prédit Accordé** |
| --- | --- | --- |
| **Vrai Refus** | 39 | 19 |
| **Vrai Accord** | 19 | 108 |

* **39 vrais refus** correctement identifiés
* **19 faux refus** (le modèle a refusé à tort)
* **19 accords manqués** (le modèle a accordé à tort)

**Random Forest**

|  | **Prédit Refusé** | **Prédit Accordé** |
| --- | --- | --- |
| **Vrai Refus** | 36 | 22 |
| **Vrai Accord** | 19 | 108 |

* Performe légèrement moins bien sur les refus (36 vs 39)

**3. Comparaison des Modèles**

| **Métrique** | **Régression Logistique** | **Random Forest** |
| --- | --- | --- |
| Précision (N) | 67% | 65% |
| Rappel (N) | 67% | 62% |
| Exactitude | 79% | 78% |

**Pourquoi choisir la Régression Logistique ?**

1. **Meilleur équilibre** : Plus performante sur la classe minoritaire (refus)
2. **Interprétabilité** : On peut comprendre facilement pourquoi elle prend ses décisions
3. **Stabilité** : Moins sensible aux petites variations dans les données

**4. Points d'amélioration**

**Problème actuel**

* Le modèle a du mal avec les **refus de prêt** (classe minoritaire)
* Exemple : Il rate environ 33% des refus (recall = 67%)

**Solutions potentielles**

1. **Rééquilibrage des classes** :

python

 # Avec SMOTE (création de faux exemples de refus)

from imblearn.over\_sampling import SMOTE

smote = SMOTE()

X\_res, y\_res = smote.fit\_resample(X\_train, y\_train)

 **Pénalisation des erreurs** :

python

 # Donner plus d'importance aux erreurs sur les refus

log\_reg = LogisticRegression(class\_weight={0:2, 1:1}) # 2x plus important de bien classer les refus

 **Seuil de décision ajustable** :

python

1. # Au lieu de prédire 0/1 directement, obtenir les probabilités
2. probas = log\_reg.predict\_proba(X\_test)[:,0]
3. y\_pred\_adj = (probas > 0.4).astype(int) # Seuil à 40% au lieu de 50%

**5. Interprétation Business**

**Pour la banque :**

* **Faux Refus** (19 cas) : Client légitime refusé → Perte de revenus
* **Faux Accords** (19 cas) : Mauvais client accepté → Risque de défaut

**Recommandation opérationnelle :**

1. Utiliser la régression logistique comme premier filtre
2. Pour les cas limites (probabilité entre 40-60%), faire une vérification manuelle
3. Surveiller particulièrement les clients avec :
   * Credit\_History = 0
   * LoanAmount élevé
   * Total\_Income faible

**6. Code d'interprétation du modèle**

python

# Importance des variables

importance = pd.DataFrame({

'Feature': X.columns,

'Coefficient': log\_reg.coef\_[0]

}).sort\_values('Coefficient', ascending=False)

print(importance)

# Exemple de prédiction interactive

client\_exemple = X\_test.iloc[0:1].copy()

print("Probabilités :", log\_reg.predict\_proba(client\_exemple))

print("Décision :", "Accordé" if log\_reg.predict(client\_exemple)[0] == 1 else "Refusé")

**Conclusion**

Vos modèles fonctionnent bien (∼80% de justesse), mais peuvent être améliorés sur la détection des risques. La simplicité de la régression logistique en fait un bon choix initial, avec possibilité d'ajouter un Random Forest comme second avis pour les cas incertains.