Transactions frauduleuses

Le but de ce projet est de trouver les transactions frauduleuses. Ce document va présenter la démarche de ce projet.

Le chargement des données et l'exploration

D'abord, j'ai chargé les données. J'ai regardé les premières et les dernières 10 lignes des données pour avoir la vue d'ensemble. La fonction describe est aussi utile parce qu'elle donne les informations statistiques du dataset.

Ensuite, j'ai vérifié s'il y a de valeurs manquantes dans le dataset. Notre dataset n'a pas la valeur manquante.

Après, j'ai regardé la corrélation entre les informations encodées (les variables de V1 à V28). J'ai trouvé qu'ils ne sont pas corrélées. J'ai regardé la distribution de ces variables, aussi. Elles sont centrées. Par contre, il y a des valeurs aberrantes (outliers) dans chaque variable. Pour l'instance, je n'ai pas su si les outliers sont important pour la classification. Je vais le vérifier après.

En plus, j'ai regardé les variable Time et Amount. J'ai normalisé ces 2 variables puisque qu'elles ont les valeurs beaucoup plus grandes que celles des variables de V1 à V28. Ces grandes valeurs vont prendre trop de l'importance dans la classification.

A la fin, j'ai regardé la variable Class. Plupart de données (plus de 99%) sont les transactions innocentes. Notre dataset est déséquilibré. J'ai appliqué l'algorithme SMOTE pour oversample les données des transactions frauduleuses. On peut undersample les données des transactions innocentes, aussi. Mais, undersampling va perdre trop des données. J'ai préféré oversampling.

J'ai séparé les données dans train set et test set, et j'ai faire oversampling sur train set. Après oversampling, j'ai trouvé plusieurs variables sont corrélées. J'ai pensé que le nombre des outliers sont augmenté à cause d'oversampling, et ils peut-être causent la corrélation. Alors, j'ai supprimé les outliers.

J'ai nettoyé les données. Ensuite, je vais utiliser ces données à créer le modèle.

La création du modèle

D'abord, j'ai choisi les algorithmes populaires pour la classification. J'ai entrainé le modèle sur oversampled train set, et évaluer le modèle sur test set. Pour gagner le temps, je n'ai pas fait Cross-Validation (CV) pour trouver les meilleures hyper paramètres. C'est mieux de le faire pour chaque algorithme.

J’ai trouvé LogisticRegression est le meilleur modèle. Les métriques j’ai regardé est la précision, le rappel et le F1 score pour la classe 1 (les transaction frauduleuses). La bonne classifier doit trouver les transactions frauduleuses, et ne prend pas le les transactions innocentes comme celles frauduleuses. LogisticRegression peut trouver presque toutes les transactions frauduleuses (le rappel est plus de 95%). Mais, la précision est trop bas. J’ai voulu trouver les meilleures hyper paramètres par CV pour augmenter la précision.

CV sur le dataset déséquilibré est différent que celui normal. Puisqu’on veut évaluer le modèle avec les données qui ne participent pas l’entrainement, il faut juste oversample sur les folders qui participent l’entrainement et laisser un folder comme il est pour l’évaluation dans CV. Pour réaliser ça, il faut oversample les données dans chaque itération de CV. J’ai créé une fonction pour faire CV sur le dataset déséquilibré.

J’ai trouvé les meilleures hyper paramètres, et obtenu le modèle. Ensuite, je vais évaluer ce modèle avec test set.

Evaluation du moèle

Les métriques j’ai regardé est la précision, le rappel et le F1 score pour la classe 1 (les transaction frauduleuses). Le modèle est entrainé avec le oversampled train set sans outliers. Le rappel de la classe 1 dans test set est haute (0.98). Mais, la précision de la classe 1 dans test set est trop bas (0.02). En revanche, celle dans train set est haute (0.78). J’ai pensé que les outliers sont importants pour la classification. Il faut les garder.

Donc, j’ai refaire CV utilisé le dataset avec outliers pour trouver les meilleures hyper paramètres, entraine le modèle utilisé le dataset avec outliers. Le rappel de la classe 1 dans test set reste haute (0.94). La précision de la classe 1 dans test set est un peu mieux (0.06), et celle dans train set augmente beaucoup (0.97). On peut résumer que les outliers sont importantes pour la classification, et le modèle est overfitted sur train set. J’ai pensé que SMOTE génére les données sont trop similaire que celles originales. Alors, le modèle est entrainé sur la même type de données. Peut-être, ça cause overfitting.

Conclusion

J’ai normalisé Time et Amount, oversample train set avec SMOTE. J’ai appliqué LogisticRegression pour créer la classifier. Le rappel de la classe 1 (les transaction frauduleuses) est haute (0.94), mais la précision est trop bas (0.06). D’ailleurs, le modèle est overfiitted sur train set. Pour résoudre ce problème, il faut trouver un autre moyen à oversample train set.