## HW2: Credit Risk Classification Report

Date: 09/29/2020

Binhui Xu(G01172196)

Username on miner2: WhitneyXu

LeadBoard Score: 0.66

# Approach:

1. **Read data from .csv files and encoding categorical variables:** Most machine learning models are unable to handle non-numeric columns. Therefore, I used One-Hot Encoding[1] to encode those object variables(F10, F11) to numbers.

Function: read\_data()

1. **Use Synthetic Minority Oversampling technique to deal with imbalanced classes:**  SMOTE (Synthetic Minority Oversampling Technique) works by randomly picking a point from the minority class and computing the k-nearest neighbors for this point. The synthetic points are added between the chosen point and its neighbors[2]. Here I used [imblearn.over\_sampling.SMOTE](https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over_sampling.SMOTE.html) to perform SMOTE.

Function: sampling(train2, credit)

1. **Experiment with various classification models:**

* **Decision tree classifier:** I used sklearn.tree.DecisionTreeClassifier to build the Decision Tree classifier. Besides, Bagging classifier was used to improve the accuracy.

Function: decision\_tree(tr\_bal, credict\_bal,X\_test,y\_test)

* **Naive bayes classifier:** I used sklearn.niave\_bayes.GuassianNB to build the Guassian naïve bayes classifier.

Function: naive\_bayes(tr\_bal,credit\_bal,X\_test,y\_test)

* **SVM classifier**: I used [sklearn.svm.svc](https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html) library to build the SVM classifier and set the kernel as linear.

Function: SVM(tr\_bal,credit\_bal,X\_test,y\_test)

* **Random-forest classifer:** I used [sklearn.ensemble.RandomForestClassifier](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html) library to build the model. And use cross validation and GridSearchCV to tune the parameters.

Function: random\_forest(tr\_bal,credit\_bal,X\_test,y\_test)

* **Adaboost classifier:** I used use sklearn.ensemble.AdaBoostClassifier to build the model, And use cross validation and RandomizedSearchCV to tune the parameters.

Function: adaboost(tr\_bal,credit\_bal,X\_test,y\_test)

1. **Classifier evaluation:** use cross validation to select one of the models that performed better, and the result(Fig.1) shows that Decision Tree and Adaboost classifier has better f1 score.

Function: cross\_validation(), get\_f1\_score(X\_test, predict)

**Chart, bar chart

Description automatically generated**

Fig.1

# F1 Score:

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall[10].

# Reference:

1. <https://medium.com/swlh/german-credit-risk-classification-modeling-and-metrics-19182b87f060>
2. <https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/>
3. <https://machinelearningmastery.com/adaboost-ensemble-in-python/>
4. <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html>
5. <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingClassifier.html>
6. <https://scikit-learn.org/stable/modules/svm.html>
7. <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html>
8. <https://stackoverflow.com/questions/31681373/making-svm-run-faster-in-python>
9. <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>
10. <https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/>