CMSE 492 Final Project Report :
Credit Fraud Classification by Alice Shirley

Background and Motivation:

There have been 52 million cases of credit fraud in the United States within the past year. The issue of credit fraud is more prevalent than ever, and with online banking as the default, credit fraud has become much more common in recent years due to phishing scams and the nature of internet security. In 2013, data was recorded containing transactions made on credit cards by European cardholders. It presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions.

The classification of fraudulent versus non-fraudulent transactions is more important than ever in order to protect cardholder's details and money. Fraud detection is a critical part of financial services and identifying fraud as early as possible can save businesses money and protect consumers. This problem is interesting, specifically with this dataset, because it is very imbalanced. This poses a challenge, as it means traditional measures of accuracy are not helpful in the same way as metrics like Area Under the Precision-Recall Curve are.

I will approach this problem using Support Vector Machines and Stochastic Gradient Descent, both of which are powerful when it comes to classification. This approach is new and interesting because SVMs are useful in maximizing the margin between classes, while distinguishing between fraudulent and non-fraudulent transactions even with a small number of fraud cases. SGDs are interesting because they are more flexible and can be used to minimize a loss function for classification tasks, while working significantly faster on large datasets than other models.

The desired outcome is to develop a fraud detection model that is capable of identifying fraud cases despite the imbalance in the dataset, with a high AUPRC. If these conditions are met, the model will be an excellent classifier as a high AUPRC indicates the model has good discriminatory ability. By improving the model's precision and recall, the aim is to ensure that the model can correctly identify fraud transactions while minimizing false positives, which are costly in terms of customer experience and business operations.

Machine Learning helps achieve this goal by learning complex patterns in the data that may not be obvious. It can also adapt and improve over time. SVMs and SGDs can handle high-dimensional data and work well with imbalanced datasets when properly tuned. Other approaches could be Logistic Regression or Random Forests. However, these methods rely heavily on predefined rules, which prevents the complexity of the data from being captured. Machine learning models, on the other hand, learn from the data and can be tuned for complex datasets.

ML Task and Objective:

The task is to build a supervised learning model for a binary classification problem that aims to differentiate fraudulent and non-fraudulent credit card transactions. The dataset consists of 284,807 transactions, with only 492 labeled as fraudulent, leading to a highly imbalanced class distribution where fraud accounts for just 0.172% of the data.

The problem presented is to detect fraudulent transactions effectively with machine learning techniques to address the challenges posed by data imbalance and the subtle differences between fraudulent and non-fraud transactions.

The primary goal is to create a model that maximizes the Area Under the Precision-Recall Curve (AUPRC) to effectively measure the model's ability to identify fraud cases while minimizing false positives. High AUPRC scores demonstrate the model's proficiency in handling imbalanced data.

The main evaluation metric is the AUPRC, which is more suitable than accuracy for imbalanced datasets. A secondary metric is the F1-score. A model can be considered excellent if it has an AUPRC score of .80 or higher. This would indicate the model has good discriminatory ability and that 80 percent of the time the model will correctly assign a fraudulent label to a randomly selected fraudulent transaction than to a randomly selected non-fraudulent transaction. The F1-score is considered good if it is .8 or higher as it ensures the model is both effective and reliable.

Metrics:

To evaluate the performance of the credit card fraud detection model, Area Under the Precision-Recall Curve (AUPRC) is the primary metric and F1-score is the secondary metric

AUPRC measures the trade-off between precision (the proportion of correctly identified fraudulent transactions out of all transactions labeled as fraud) and recall (the proportion of actual fraudulent transactions correctly identified by the model) across different thresholds. AUPRC is best suited for imbalanced datasets because it focuses specifically on the performance of the model for the positive class (fraudulent transactions). It avoids the misleadings results from a skewed dataset (non-fraud transactions). A model with an AUPRC score of 0.80 or higher is considered excellent. This indicates that the model can correctly assign a higher fraud risk to a randomly selected fraudulent transaction than to a randomly selected non-fraud transaction 80% of the time.

The F1-score is the harmonic mean of precision and recall. It balances the two metrics with the following equation: *F1 = 2 x (Precision x Recall)/ (Precision + Recall)*. The F1-score is crucial in fraud detection as it ensures the model prevents disruptions to customers by reducing false positives, and that true fraud cases are caught to mitigate financial losses. An F1-score of 0.80 or higher is considered good, as this indicates the model is both effective in identifying fraud cases and reliable in minimizing false positives.

Initial and Exploratory Data Analysis:

The dataset contains anonymized credit card transactions by European cardholders in September 2013, available on Kaggle: Credit Card Fraud Detection Dataset. It includes 284,807 transactions, with only 492 fraud cases (0.172%).

The Raw data contains V1 to V28: PCA-transformed numerical features, Time: Seconds since the first transaction, Amount: Transaction value, useful for cost-sensitive analysis, and Class: Binary target variable (0: non-fraud 1: fraud). It poses challenges in that there is a class imbalance where fraud cases are only 0.172% of the data, and V1-V28 are anonymized. The data needs to be checked for missing values, and also addressed for imbalanced via either SMOTE, undersampling, or class weights

Models:

Baseline Model: Dummy Classifier

The initial baseline for comparison is a Dummy Classifier, utlizing both the Stratified and uniform strategy. The stratified strategy Generates predictions by respecting the training set's class distribution. This gives a reference point for performance in an imbalanced dataset. The Uniform strategy Generates random predictions with equal probabilities for both classes, providing a worst-case scenario baseline.

The Dummy Classifier will help establish whether the machine learning models provide meaningful improvement over trivial strategies.

Support Vector Models

SVMs are supervised machine learning models that aim to find the optimal hyperplane that separates classes in a high-dimensional space. For fraud detection: Since the dataset's features are PCA-transformed, SVMs are well-suited for handling complex, high-dimensional spaces and class imbalance. They also are good at capturing non-linear relationships

Stochastic Gradient Descent (SGD):

SGD is an optimization algorithm used to train machine learning models by minimizing the loss function through iterative updates to the model's parameters. It is efficient for large datasets. For Fraud detection: datasets can be large, and SGD is designed to handle large datasets, and different loss functions can be applied, allowing customization based on the problem’s complexity

Training Methodology:

Baseline Model

1. Data Preprocessing

- The dataset excludes the target variable Class and l non-essential columns like TransactionTime and Time.
- Target Variable (y):
  The Class column is used as the binary target, where 1 indicates fraudulent transactions and 0 indicates legitimate ones.
- Train-Test Split:
  - The data is split into 90% for training and 10% for testing using train_test_split.
  - Stratification ensures that the class distribution is maintained in both the training and testing sets, addressing the dataset's imbalance.

2. Model Training

Dummy Classifier:

- The DummyClassifier is trained on the training set (X_train, y_train) without any learning from data, providing a non-informative benchmark.
- Uniform DummyClassifer generates random predictions with equal probabilities
- Stratified DummyClassifer generates predictions with respect to the class distribution
-

SVM Models

1. Data Preprocessing

- Subsampling:
  The dataset is highly imbalanced, with frauds only 0.172% of all transactions. To address this, the majority class is downsampled to match 70 times the number of fraud instances. My computer was unable to process the full dataframe for SVM so subsampling was neccessary
- Feature Selection:
  The features Class, TransactionTime, and Time are excluded from the input data (X), as Class is the target variable and Time and TransactionTime are not directly used for training.

2. Model Selection

Two different Support Vector Machine (SVM) models are used:

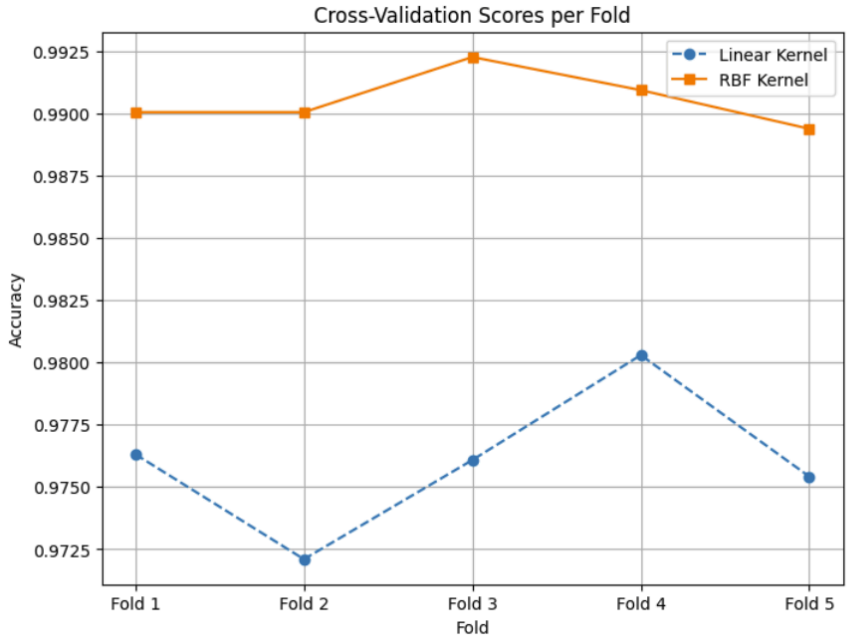
- Linear Kernel: This is a simpler SVM model that uses a linear decision boundary. It's effective when the data is linearly separable. :Even though fraud detection patterns are often complex, sometimes the features can exhibit linearly separable behavior when reduced to principal components.

- RBF (Radial Basis Function) Kernel: This model uses a non-linear kernel that can handle more complex decision boundaries, making it suitable for datasets with non-linear relationships between features

Both models use the class_weight='balanced' argument to handle class imbalance by adjusting weights inversely proportional to class frequencies.

3. Cross-Validation

- 5-Fold Cross-Validation:
  To evaluate the models, 5-fold cross-validation is used. The training data is split into 5 subsets (folds), and the model is trained on 4 folds and tested on the remaining fold. This process is repeated for each fold, and the average accuracy across all folds is calculated. Cross-validation provides a reliable estimate of model performance



4. Model Training

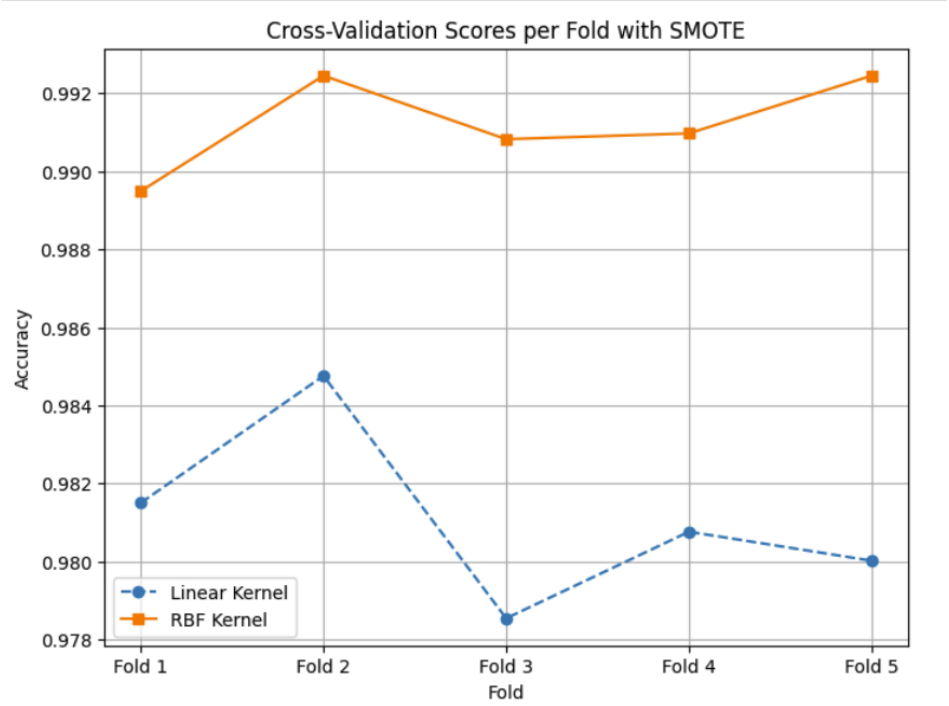
- After cross-validation, both the Linear and RBF SVM models are trained on the entire training set (X_train, y_train) using a Pipeline to ensure proper scaling and fitting.

SVM Models with SMOTE

The methodology is the same as that above, except for the inclusion of SMOTE

- This means in addition to undersampling the original datatframe, SMOTE is applied to address the class imbalance by oversampling the minority class

- . The sampling strategy is set to 0.09, meaning the number of synthetic fraud examples will make up 9% of the majority class in the training data. This helps in balancing the



data

SGD Models

1. Data Preprocessing

- Feature Selection:
  The target variable Class is separated from the features, and non-relevant columns such as Time and TransactionTime are dropped. T
- Train-Test Split:
  The data is split into 90% training and 10% testing using train_test_split. Stratified sampling preservesthe class distribution and across both

2. Model Training with SGD Classifier

- Pipeline:
  A pipeline is created to first scale the features using StandardScaler and then apply the SGDClassifier.
- The pipeline is looped through with each possible loss function as the Stochastic Gradient Descent (SGD) classifier used to train the model can be trained with different loss functions, which impacts how the classifier learns to make predictions.

    - hinge (for linear SVM),
    - log_loss (for logistic regression),
    - modified_huber (a combination of hinge and log loss),
    - squared_hinge (a modification of hinge loss),
    - perceptron (for binary classification tasks).
- Class Weighting:
  The class_weight='balanced' parameter is used to handle the class imbalance.
- Hyperparameters:
    - max_iter=1000: Limits the maximum number of iterations for convergence.
    - tol=1e-3: Defines the tolerance for stopping the algorithm when the optimization stops improving.

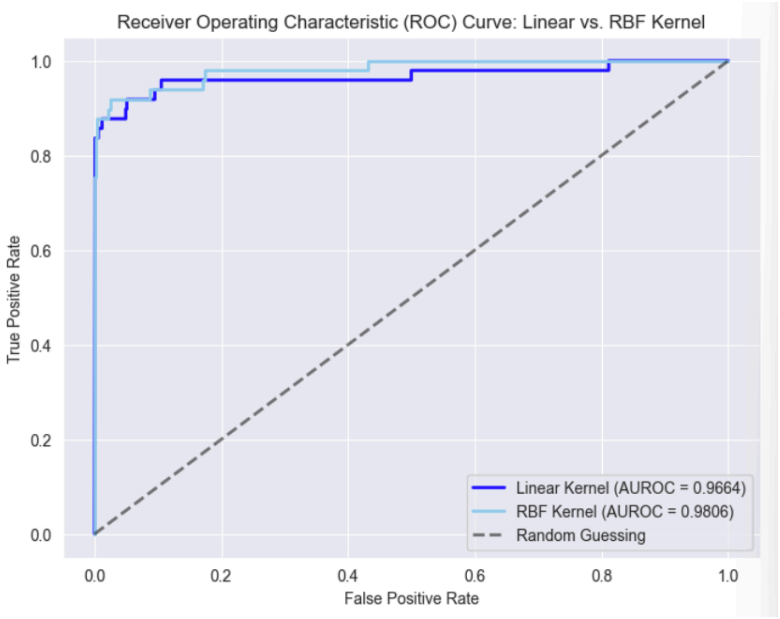
Results and Model Comparison

Linear SVM with SMOTE:

- AUROC: 0.9664
- F1Score: 0.6277
- Time: 36.56

SVM with RBF Kernel with SMOTE:

- AUROC: .9806
- F1 Score: .7414
- Time: 35:02

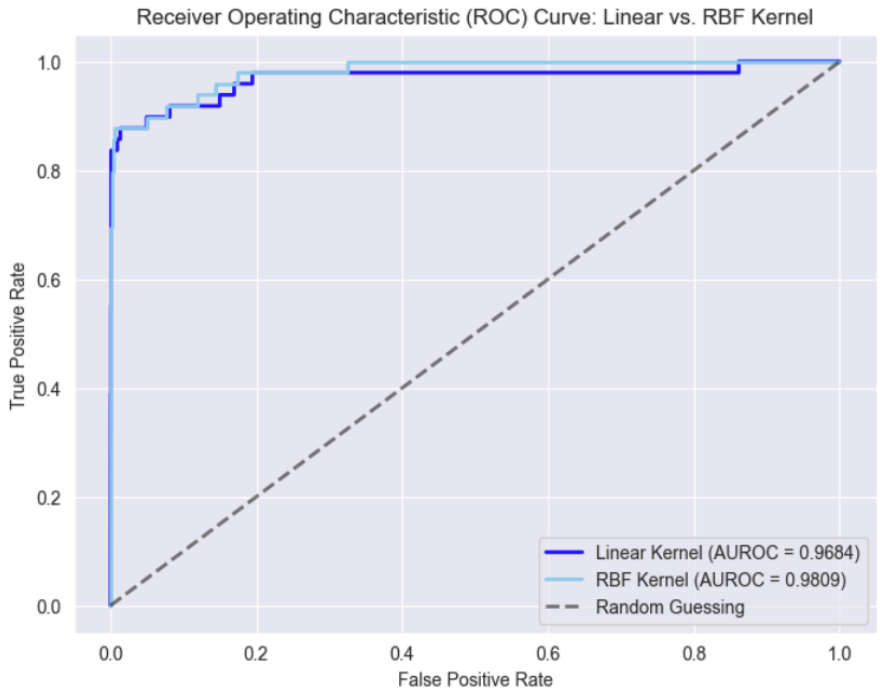


Linear SVM:

- AUROC: 0.9684
- F1 Score: .5119
- Time: 9:31

SVM with RBF Kernel::

- AUROC: 0.9809
- F1 Score: 0.7611
- Time: 7:14



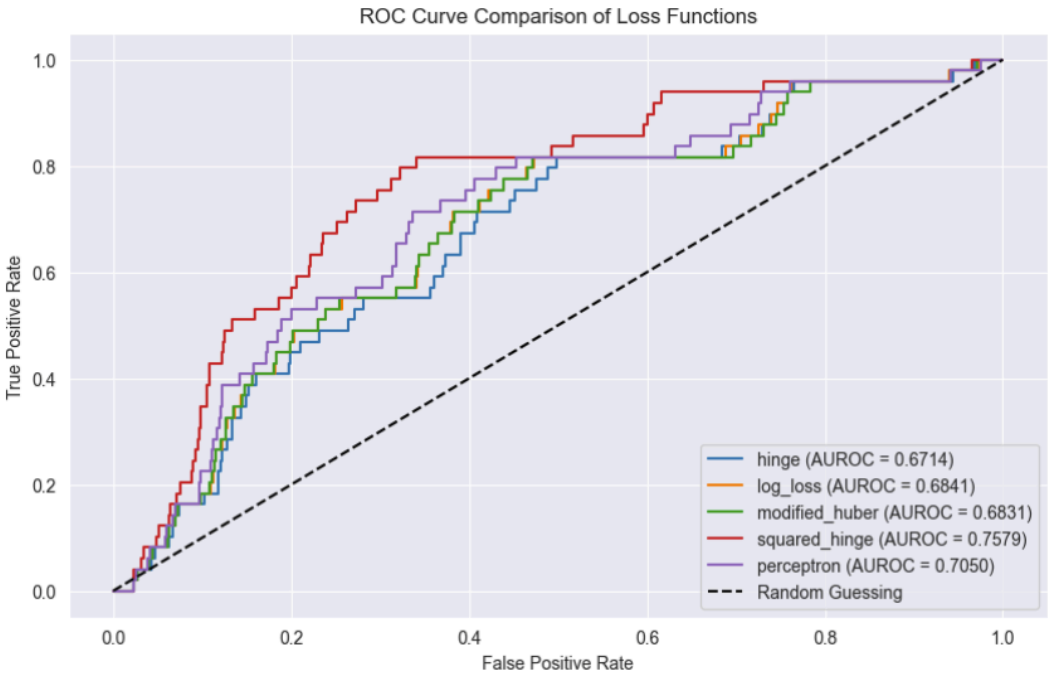
Dummy Classifier:

- AUROC: 0.5095
- F1 Score: 0.0206
- Time: 3.4 seconds

SGD Classifier:

- Loss: hinge - AUROC: 0.6714,, F1 Score: 0.0899
- Loss: log_loss - AUROC: 0.6841, F1 Score: 0.0970
- Loss: modified_huber - AUROC: 0.6831, F1 Score: 0.0978
- Loss: squared_hinge - AUROC: 0.7579, F1 Score: 0.0424
- Loss: perceptron - AUROC: 0.7050,, F1 Score: 0.0291

Time: 19 seconds



Best Algorithm: SVM with RBF Kernel and SMOTE

- The SVM with RBF kernel combined with SMOTE provides the best performance overall, with the highest AUROC (0.9806) and F1 score (0.7414). The ability to handle class imbalance through SMOTE, with the RBF kernel's ability to capture non-linear relationships, makes it the most effective choice.
- This easily clears the standard set earlier, of an AUPRC of .8 and f1-score of .8
- Why SMOTE and RBF SVM Perform Better:

- SMOTE improves model performance by balancing the class distribution and the RBF kernel captures more complex patterns in the data compared to a linear kernel

Trade-offs:

- The SVM with RBF Kernel,is more resource-intensive and time-consuming compared to the Linear SVM
- Adding SMOTE increases training time, but the performance boost justifies this additional cost.
- Better performing machinery may be able to streamline this process

4. Why Other Models Perform Worse

- Dummy Classifier: The Dummy Classifier serves as a baseline, and its poor performance is expected. It's not designed to learn from data.
- SGD Classifier: Despite its fast training time, SGD fails to capture the complex patterns in the fraud detection task, resulting in low AUROC and F1 scores across all loss functions.
- Linear SVM: While fast and computationally efficient, the Linear SVM fails to handle class imbalance effectively

Conclusion

The SVM with the RBF kernel and SMOTE gave the best results for detecting fraudulent transactions, with the highest AUROC and F1 score. SMOTE helped balance the dataset, and the RBF kernel was able to capture the complex patterns in the data. Although it took more time and resources to train, the performance improvement was worth it compared to models like the Linear SVM and SGD Classifier, which didn't perform as well. My algorithm did achieve the desired score.

I had challenges with the size of the data. I had to greatly undersample the data in order for my computer to process the data for SVM. If I did this project again, I would want to use a smaller sample. The SGD model underperformed given its potential. If I did this again, I would investigate the hyperparameters further to get a better model.

References

1. Glassbox Medicine. (2019, February 23). *Measuring Performance: AUC, AUROC*. Retrieved from https://glassboxmedicine.com/2019/02/23/measuring-performance-auc-auroc/

2. Brownlee, J. (n.d.). *SMOTE Oversampling for Imbalanced Classification*. Machine Learning Mastery. Retrieved from https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

3. Géron, A. (2022). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (3rd ed.). O'Reilly Media.