Student: Allison Lu

Report: Exploring Feature Engineering and Model Selection for Credit Card Fraud Detection

Introduction

The objective of this report is to document the process of feature engineering and model selection for detecting fraudulent credit card transactions. The dataset consisted of 555,719 instances and 22 attributes, a mix of categorical and numerical data types. The primary focus was on enhancing feature representation and exploring various machine learning models to achieve optimal classification performance.

Feature Engineering

The initial phase of the project involved extensive feature engineering to extract meaningful insights from the dataset. One of the key observations I found was the presence of repeated transactions by certain users, which indicates potential fraudulent activity. was the presence of repeated transactions by certain users, indicating potential fraudulent activity. To capture this behavior, I calculated transaction frequency and transaction amount deviation, aiming to identify suspicious patterns in transactional behavior. In the end I found the most impact in transaction frequency count of cardholder and merchant, and transaction amount deviation didn't appear to have that much of a difference.

Next, I explored demographic information about users and merchants, such as age and geographical data related to the transaction locations. However, the inclusion of these features adversely impacted model performance, leading to a lower F1 score. Subsequently, I decided to retain the transaction frequency feature while excluding geographical information, resulting in improved performance compared to the initial feature set.

To further enhance feature representation, I employed log transformation and scaling techniques to address skewness and scale differences in numerical features. This is especially the case because I was using the KNN model, the model is very sensitive to the scale of features, so scaling was necessary to fight off skewness. Furthermore, binning or discretization of continuous features was applied to capture nonlinear relationships and improve model interpretability.

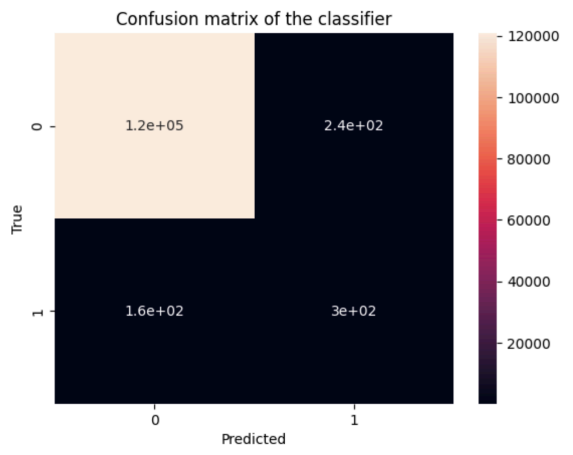
Model Selection

To evaluate the effectiveness of different machine learning algorithms, I experimented with several models, starting with Decision Trees, then KNN, to Random Forest, Gradient Boosting Classifier, and Gaussian Naive Bayes. Each model was trained and evaluated based on metrics such as accuracy and F1 score.

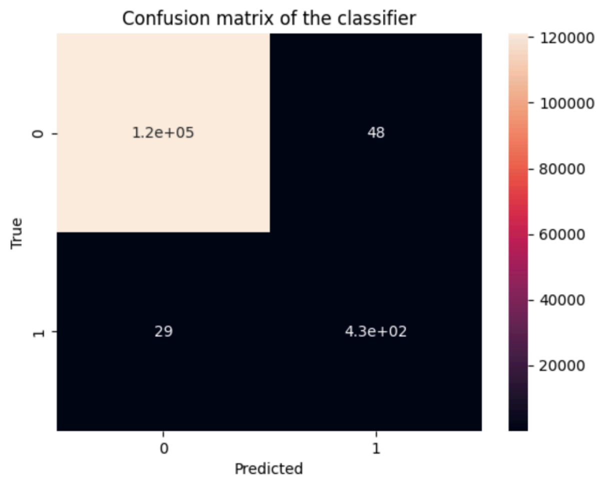
Early in the process, I added a random seed of 41. During model training and evaluation, it is essential to set a random seed for reproducibility. By fixing the random seed to 42, I ensured that the same random sequence of numbers would be generated each time the code was executed. This consistency is crucial for reproducible results and enables others to replicate the experiments and validate the findings.

Starting off, I kept the starter code using the Decision Tree and adjusted the features - as mentioned above. Next, I tested the K-Nearest Neighbors (KNN) model, which consistently outperformed others in terms of classification performance. When I first switched to the KNN model, I noticed that the features that I added for Decision Tree didn't make lots of differences in the accuracy and F1 score. This could be due to decision trees relying more on features to make decisions and the structure, whereas KNN focuses more on the overall pattern of feature values, making it harder to see the impact of features in KNN.

Overall, the KNN model demonstrated robustness in capturing complex relationships within the dataset, making it the preferred choice for fraud detection in this context, and the model that I ended up using for my final submission. From the confusion matrix referenced below, can see for the KNN model that it drastically does a lot better, where another model such as Random Forest demonstrated a high number of false positives and false negatives.



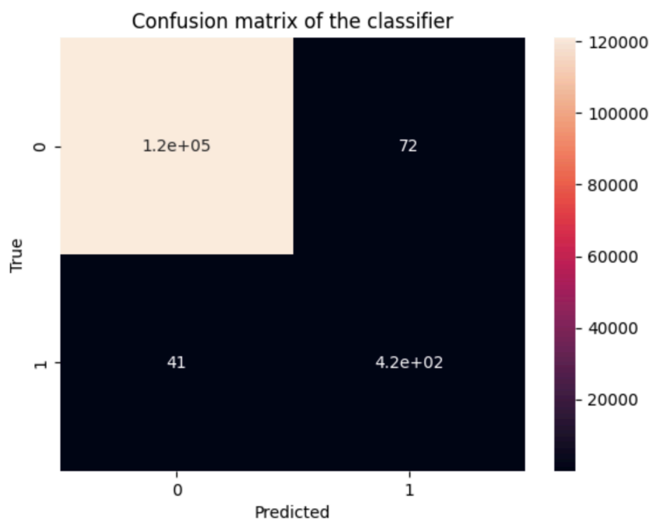
Random Forest Confusion Matrix



KNN Confusion Matrix

Now, specifically with the KNN model I used, it also went through multiple iterations to find the best possible combo of variables - this includes finding k neighbors, weights, and jobs. Originally, I set the number of neighbors to 5, but I wanted to explore whether adjusting this parameter could further the model's performance. Therefore, for the KNN model, I utilized grid search to find the best combination of hyperparameters. The grid search was performed over a range of k values ([3, 5, 7, 9, 11]), and the optimal value would be selected based on the F1 score. In the end, I got k=3 to be the most optimal number of k-nearest neighbors, which was then used to train the final KNN model.

As for class weights, during the exploration of model performance through the confusion matrix, I observed a notable class imbalance, with a disproportionate number of false negatives and false positives, with more false positives (as seen below, showing the before and after). To address this imbalance and ensure that the model appropriately accounts for both classes, I introduced class weights during model training.



Pre-weights, false positives = 72
false negatives = 41

Class weights assign higher weights to minority class samples (fraudulent transactions in this case) relative to majority class samples. This adjustment helps mitigate the impact of class imbalance on model training, resulting in a more balanced and accurate classification. By incorporating class weights, I aimed to improve the model's ability to detect fraudulent transactions while minimizing false positives and false negatives.

In addition to tuning hyperparameters and addressing class imbalance, I leveraged parallel processing capabilities by specifying the number of jobs during model training. By setting the number of jobs to a suitable value, I optimized computational efficiency and reduced training time, particularly for large datasets.

Conclusion

In conclusion, this project highlighted the importance of feature engineering and model selection in credit card fraud detection. Through iterative experimentation and evaluation, I identified key features that effectively capture fraudulent behavior while optimizing model performance. Furthermore, the results underscored the superiority of the KNN model for this specific task, emphasizing the significance of selecting the appropriate algorithm based on the dataset characteristics.

Moving forward, further research could explore additional feature engineering techniques and ensemble methods to enhance model performance and robustness in detecting fraudulent transactions. In order to achieve this, analyzing the data some more can help me identify more fraud patterns. Meanwhile, if possible, run the model twice on an updated set of data that filters through the fraud pattern.