CREDIT CARD FRAUD DETECTION BY MACHINE LEARNING AND ANN

Deliverable III Execution and Interpretation

Modeling Overview & Feature Engineering

The project consists of a comparative study of Logistic Regression, Random forests and ANNs to understand the performance of each of them on credit card fraud detection. For this project, I implemented logistic regression, random forests, and convolutional neural networks. Logistic regression and random forests are implemented using the sklearn library and ANNs are implemented with Pytorch library. Convolutional neural network because CNN is the special family of ANN used to create the model. Besides, the model is highly scalable with the amount of data that is fed into the system. Besides, convolutional neural networks (CNN) are usually applied on 2-dimensional image data for feature extraction and prediction in deep learning, but a 1-dimensional variant of the same can be used in understanding trends in time series and financial data as the problem statement at hand.

Model Output Result Execution

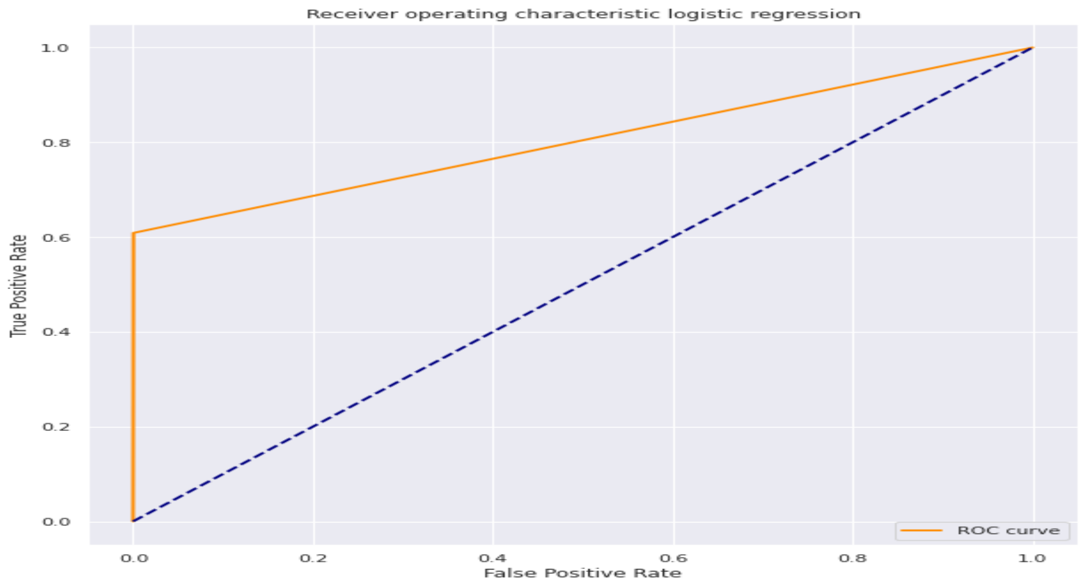
As part of my project proposal, I am very interested because it seems almost all output comes to the concrete result. Because all three models result in shows very high accuracy results during the mode implementations time, however, CNN took a long time to execute the final out accuracy result.

Logistic Regression, it saw that even though the precision of the model is high (~99%), the review score for false cases is entirely low (~ 62%). This is an immediate impact of the substantial class irregularity present in the informational collection that at long last shows as the powerlessness to comprehend the minority class.

Logistic regression

The classification report for logistic regression is given below:

```
          precision    recall  f1-score   support

           0       1.00      1.00      1.00      6229
           1       0.91      0.62      0.74        69

    accuracy                           1.00      6298
   macro avg       0.96      0.81      0.87      6298
weighted avg       0.99      1.00      0.99      6298
```



ROC curve for logistic regression is given above.

Random Forest

The Random forest algorithm pretty like Logistic Regression, it very well may be noticed that even though the accuracy of the model is high (~ 100%) the review score for deceitful cases is truly low (~ 68%). It performs better with many preparing information, likewise, the speed during testing and application is acceptable.

The classification report of random forests is given below:

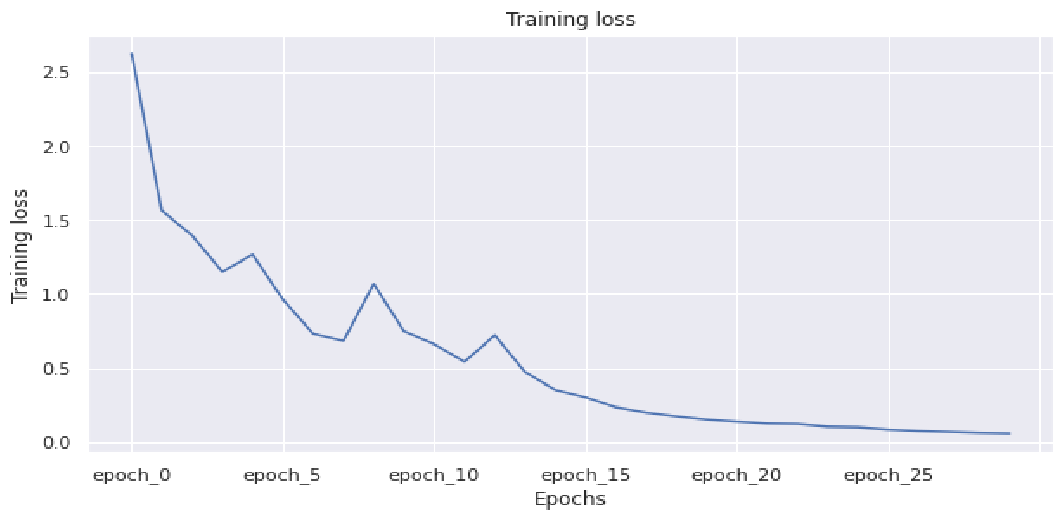
```
          precision    recall  f1-score   support

           0       1.00      1.00      1.00      6229
           1       0.96      0.68      0.80        69

    accuracy                           1.00      6298
   macro avg       0.98      0.84      0.90      6298
weighted avg       1.00      1.00      1.00      6298
```

Artificial neural networks

Neural networking, likewise, at last, arrives at the exactness of 95% on the dataset with an extremely high recall score of 90%, anyway it took an hour to get the outcome. In future work, I intend to improve the exactness, and handling season of the budgetary extortion process continuously joined with both AI-based procedure and profound fake neural systems.

```
TEST : epoch=25 total_loss: 0.359703 accuracy: 0.870599 roc_auc_score
0.934414: 100%|██████████| 33/33 [00:04<00:00,  6.82it/s]
TEST : epoch=25 total_loss: 2.642112 accuracy: 0.869300 roc_auc_score
0.923030: 100%|██████████| 33/33 [00:04<00:00,  6.93it/s]
TEST : epoch=25 total_loss: 0.715756 accuracy: 0.880533 roc_auc_score
0.939209: 100%|██████████| 33/33 [00:05<00:00,  6.60it/s]
TRAIN : epoch=26 total_loss: 0.847440 accuracy: 0.817163 roc_auc_score
0.904902: 100%|██████████| 886/886 [02:17<00:00,  6.43it/s]
TEST : epoch=26 total_loss: 0.289967 accuracy: 0.882446 roc_auc_score
0.940223: 100%|██████████| 33/33 [00:05<00:00,  6.54it/s]
TEST : epoch=26 total_loss: 1.676209 accuracy: 0.899138 roc_auc_score
0.938145: 100%|██████████| 33/33 [00:04<00:00,  6.66it/s]
TEST : epoch=26 total_loss: 0.603099 accuracy: 0.887756 roc_auc_score
0.942871: 100%|██████████| 33/33 [00:04<00:00,  6.60it/s]
TRAIN : epoch=27 total_loss: 0.434537 accuracy: 0.854498 roc_auc_score
0.925498: 100%|██████████| 886/886 [02:19<00:00,  6.35it/s]
TEST : epoch=27 total_loss: 0.244038 accuracy: 0.913584 roc_auc_score
0.941044: 100%|██████████| 33/33 [00:04<00:00,  6.80it/s]
TEST : epoch=27 total_loss: 2.924564 accuracy: 0.923528 roc_auc_score
0.950578: 100%|██████████| 33/33 [00:04<00:00,  6.78it/s]
TEST : epoch=27 total_loss: 0.480485 accuracy: 0.910846 roc_auc_score
0.954590: 100%|██████████| 33/33 [00:04<00:00,  6.72it/s]
TRAIN : epoch=28 total_loss: 0.300248 accuracy: 0.891691 roc_auc_score
0.944548: 100%|██████████| 886/886 [02:20<00:00,  6.33it/s]
TEST : epoch=28 total_loss: 0.174238 accuracy: 0.939626 roc_auc_score
0.954355: 100%|██████████| 33/33 [00:04<00:00,  6.85it/s]
TEST : epoch=28 total_loss: 1.556337 accuracy: 0.952884 roc_auc_score
0.965693: 100%|██████████| 33/33 [00:05<00:00,  6.52it/s]
TEST : epoch=28 total_loss: 0.404540 accuracy: 0.935587 roc_auc_score
0.967285: 100%|██████████| 33/33 [00:04<00:00,  6.61it/s]
```

```
TRAIN : epoch=29 total_loss: 0.242479 accuracy: 0.918960 roc_auc_score
0.958503: 100%|██████████| 886/886 [02:18<00:00,  6.38it/s]
TEST : epoch=29 total_loss: 0.171014 accuracy: 0.949096 roc_auc_score
0.959195: 100%|██████████| 33/33 [00:04<00:00,  6.69it/s]
TEST : epoch=29 total_loss: 1.662137 accuracy: 0.962827 roc_auc_score
0.970812: 100%|██████████| 33/33 [00:04<00:00,  6.79it/s]
TEST  :  epoch=29  total_loss:  0.369412  accuracy:  0.948019  roc_auc_score
0.973633: 100%|██████████| 33/33 [00:04<00:00,  6.68it/s]
```



Training loss across the different epochs are plotted in the figure above.

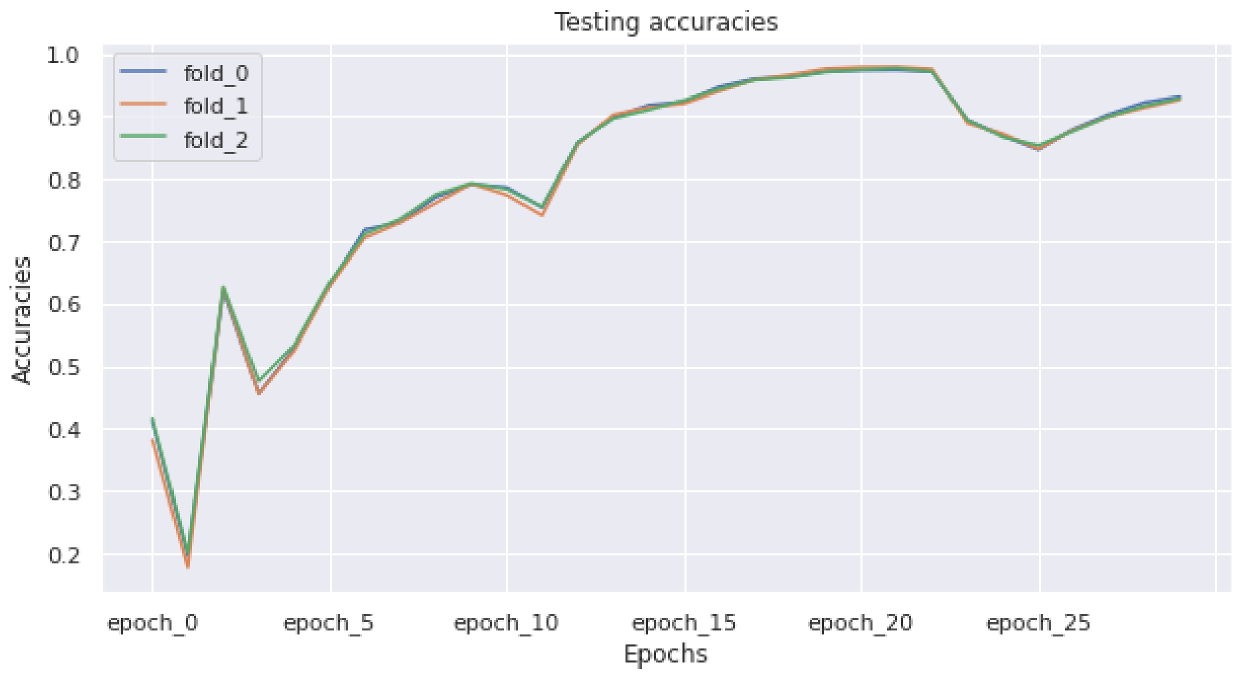


Figure show, the test accuracies for the different k-fold splits across different epochs are plotted.

Model Result Clarification Which Differently Done

The following improvements can be done on the existing project:

- ✓ Different loss functions for logistic regression can be implemented
- ✓ The number of decision trees can be increased to see the performance of random forests.
- ✓ The CNN architecture and learning strategy can be experimented with to provide further insights.
- ✓ A random up/down sampling of the complete dataset can be done to handle class imbalance before training.
- ✓ Logistic regression provides a very simple, fast, and highly interpretable model for predicting credit card fraud detection.
- ✓ Random forests provide a better model with medium interpretability and reasonable metrics (accuracy, precision, and recall) at a marginal computation resource.
- ✓ Artificial neural networks give a complicated model at par with random forests but can be fine-tuned to any extent with very low interpretability. Best recall scores but comes at a computational cost.

Conclusion

The result that has been concluded that for the implementation of Logistic regression provides a very simple, fast, and highly interpretable model for predicting credit card fraud detection. Random forests provide a better model than anyone with medium interpretability and reasonable metrics (accuracy, precision, and recall) at a marginal computation resource. The result that has been concluded that for the implementation of logistic regression, it observed that even though the accuracy of the model is high (~ 99%), the recall score for fraudulent cases is (~ 62%). This is a direct effect of the heavy class imbalance present in the dataset that finally manifests as the inability to understand the minority class. Random forest is very similar to logistic regression, it can be noted that even though the accuracy of the model is very high (~100%), the recall score for fraudulent cases is low (approx 68%). This is a direct effect of the heavy class imbalance present in the dataset, which finally manifests as the inability to understand the minority class. Besides.

Artificial neural networks give a complicated model at par with random forests but can be fine-tuned to any extent with very low interpretability. Best recall scores but comes at a computational cost, also finally reach an accuracy of 95% on the dataset with a very high recall score of 97%, however, it took an hour to get the result.

In future work, I aim to improve the accuracy and processing time of the financial fraud process in real-time combined with both machine learning-based process and deep artificial neural networks.

Reference

1,Machine Learning and Deep Learning with Python, scikit-
learn, and TensorFlow Third Edition – Includes Tensor-
low 2, GANs, and Reinforcement Learning Sebastian Raschka & Vahid Mirjalili

2,https://www.geeksforgeeks.org/ml-credit-card-fraud-detection/

3, References1. Credit Card Fraud Detection Based on Transaction Behavior -
by JohnRichard D. Kho, Larry A. Vea published by Proc. of the 2017 IEEERegion 10 Conferenc
e (TENCON), Malaysia, November 5-8, 2017

4,https://scikit-learn.org/stable/modules/generated/sklearn.

5,linear_model.LogisticRegression.html

6,https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.

7,RandomForestClassifier.html

8,https://scikit-learn.org/stable/modules/generated/sklearn.

9,model_selection.GridSearchCV.html

10,https://scikit-learn.org/stable/modules/generated/sklearn.tree.

11,DecisionTreeClassifier.html

12,https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.

13,VotingClassifier.html

14,https://www.datacamp.com/courses/fraud-detection-in-python

15,https://towardsdatascience.com/

16,how-dbscan-works-and-why-should-i-use-it-443b4a191c80

17,https://scikit-learn.org/stable/modules/generated/sklearn.cluster.
18,https://towardsdatascience.com/pytorch-layer-dimensions-what-sizes-should-they-be-and-why-4265a41e01fd