# Progress Report 2

# Fraudulent Transaction Detection System

# Yathartha Patankar

# yatharthapatankar@csu.fullerton.edu

1. **Project milestones**

**Status**: completed ongoing not completed

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Item #* | *Tasks* | *Due Date* | *Status* | *Comments* |
| 1 | Identifying Data Sources and Data Set for the Project. | 1/30 |  | Completed. There can be various sources and types of transactions, but fraudulent transactions mainly involve credit-card-based fraud, so focusing on credit-card-based datasets. Once the model is trained and tested, I will also try to include other related datasets. |
| 2 | Preprocessing and cleaning data in the dataset. | 2/15 |  | Currently ongoing. Identifying the relevant fields, cleaning the noise from data, and normalizing specific fields with larger scales.  Used RobustScaler to preprocess specific columns in the dataset. |
| 3 | Determine the correct tools and technologies to use in the project. | 2/15 |  | Currently using Google Colab for training and evaluating the model as it provides better GPU. Python 3.0, Django, Jupyter Notebook, JavaScript HTML and CSS, and NoSQL constitute the project's planned tech stack. |
| 4 | Define the appropriate ML model to train and use later in the project. | 2/22 |  | Ongoing. It is determined which model would be the best fit for the project. Classification will best fit this project as the dataset has labeled data and will predict the category of new data points(transactions).  Training of the model is ongoing. |
| 5 | Tackle the imbalance of the dataset and check the model performance. | 3/05 |  | The confusion matrix was skewed towards the majority class, and the ‘False Negative’ number was high in the trained model. Implemented SMOTE and under-sampling techniques to compare results. SMOTE gave better results, and the confusion matrix had fewer false negatives. |
| 6 | Implement an autoencoder for anomaly detection after balancing the data. | 3/12 |  | As suggested by the project advisor and as a distinctive feature of the IEEE paper, the autoencoder model was implemented to predict and flag fraudulent transactions. After using SMOTE, the Autoencoder model gave a significant accuracy and low ‘False Negative.’ The model also correctly predicted a high number of ‘True Positives.’ |
| 5 | Testing and evaluating the model performance. | 3/14 |  | An autoencoder model is ready. Currently, I am experimenting with different Optimizers(currently using ‘adam’) and Loss Functions(binary\_crossentropy) to get better accuracy and less ‘False Negative’ in the model. |
| 6 | Saving the model and using it integrate it with the front-end. | 3/28 |  | Integrating the model with the front end is not possible on Google Colab. The model will be saved and used later in VS code for development. |
| 7 | Designing and developing the UI and UX. | 4/11 |  | Incomplete. A comprehensive front-end user interface will allow users to see the statistics of transactions and validate new transactions.  Planning to add an API that sends a message every time a transaction is flagged. |
| 8 | Integrating the model and front-end and testing the overall working of the project. | 4/25 |  | Incomplete. This phase will involve combining the developed model with the user interface and subsequently evaluating the functionality and performance of the project through comprehensive testing procedures. |
| 9 | Deployment | 4/30 |  | After successful development and testing, the project will be strategically deployed utilizing an appropriate platform, such as GitHub or Netlify, considering the final project size and resource availability. |

1. **A summary of progress**

Progress since last report:

The credit card transaction dataset was loaded and preprocessed using pandas for efficient data handling, ensuring clarity in feature-label separation. Synthetic Minority Over-sampling Technique (SMOTE) was applied to mitigate class imbalance, facilitating practical model training. Data splitting with train\_test\_split ensured appropriate partitioning for training and evaluation. StandardScaler normalized both training and testing data for improved convergence. A supervised autoencoder model architecture was defined in TensorFlow and Keras, compiled with Adam optimizer and binary cross-entropy loss. The model was trained on balanced data, minimizing loss for optimal performance. Using a confusion matrix and classification report, evaluating original testing data provided a comprehensive assessment of binary classification efficacy.

Google Colab:<https://colab.research.google.com/drive/1GCO-8KmTIH1dT3OwBlyVhE3HCiWQ_ab7>

1. **Significant problems or changes**
2. No significant problems as of now. The problems that I encountered during progress report 1 have been resolved.
3. **Questions**
4. **None**
5. **Detailed descriptions of completed tasks and results (optional)**

Detailed descriptions of the completed tasks and results.