4.0.0.0. Phase 4: Advanced Modeling and Feature Engineering

In phase 3 the best model was found and further tuned. Error analysis was also done. Only classical non deep-learning models were used - Logistic regression, Random forest and Lightgbm. In this phase, PCA shall be performed on test data and further analysis of correctly and wrongly predicted points will be done. New features shall be added to the original dataset and models will be trained on new dataset. Deep learning models shall also be trained. Results from the new models shall be analyzed.

The ipynb file corresponding to this phase is divided into sections. This phase 4 documentation has to be read along with the ipynb file for correlation of different terminologies and outputs. The description given here follows the same section wise approach as the ipynb file. All the relevant files can be accessed through the following link: https://drive.google.com/drive/folders/1evFZRwFWh4zkR9CiT46llB9PlaXFLfLA?usp=sharing

4.1.0.0. Common commands: The following actions are performed in this section:

4.1.1.0. Google Drive is mounted for accessing data files.

4.1.2.0. Relevant packages are installed.

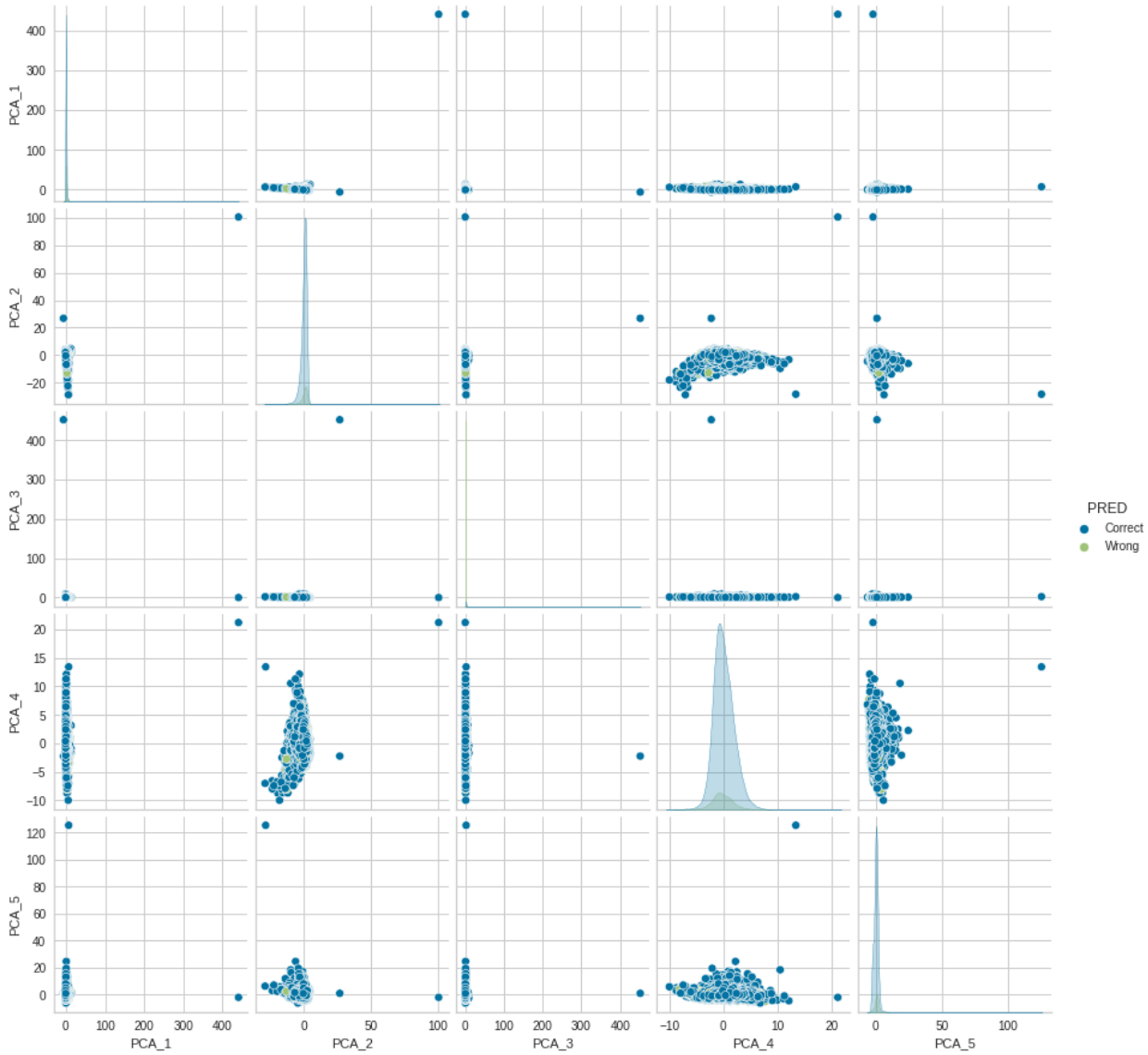
4.1.3.0. Relevant libraries are imported. It may happen that some packages and libraries are not used. However, they appear because they were used in the process of development of the final ipynb file.

4.1.4.0. One custom function for dataframe optimisation is defined. Data sets created in the previous phase are imported and their shapes are printed.

4.2.0.0. PCA:

4.2.1.0. Data Preparation - We use the test dataset with predictions saved in phase 3. Some columns are dropped from the dataframe as they cannot be used to perform PCA.

4.2.2.0. PCA is performed with 5 components i.e., PCA outcome has 5 features. A data frame is formed with these 5 features. Target, Label and Score columns are added to this dataframe. Two new columns - Pred and Type - are added. Pred column indicates whether the data point is correctly predicted or wrongly predicted. Type column indicates prediction with correct and wrong label e.g., Correct_0 means the prediction is 0 and it is correctly predicted. We make pairplots from the 5 PCA features. Pair plots from the ipynb file is reproduced here.



From the pair plots (which are scatter plots considering 2 features at a time), it is observed that the spread of wrongly predicted data is lower compared to spread for correctly predicted data.

4.2.3.0. We also perform PCA with 2 features and make a scatterplot. From the scatter plot it is observed that the spread of wrongly predicted data is lower compared to spread for correctly predicted data.

4.3.0.0. <u>Train model with CONFIDENCE as output</u>

4.3.1.0. Light GBM with tuned parameters (from Phase 3) is trained on feature selected data (X_train_feature_selected). [Please note that the best model was saved in phase 3. However, using the saved model was throwing an error. So, the model had to be trained again with the already available tuned parameter values.] Pycaret is used for this training as was done in Phase 3. For this data setup is done and column names

of input data are changed to suit the acceptability of Pycaret. This trained model is named best_model. Prediction is made on test data and thus we obtain a dataframe with test data and TARGET, Label and Score columns. A new column is added which indicates the prediction and confidence. For example, Correct_High indicates that the datapoint is correctly predicted with high confidence. Similarly data points are labeled as Correct_Low, Wrong_High and Wrong_Low. This new column is added based on the Score column. The score column indicates the probability of predicted Label. A cut off point is selected as defining low or high confidence. This cut off value is 0.75. If the score is less than or equal to 0.75, the confidence is low otherwise high. Thus a correctly predicted point with Score more than 0.75 is labeled Correct_High. Similarly other points are classified. This new test data frame is named predict_test.

Prediction is also made on train data and confidence column is added. This new data frame is named predict_train.

4.3.2.0. New data frames are defined as train and test data. These are X_predict_train and X_predict_test respectively. These are obtained by dropping TARGET, Label and Score columns from predict_train and predict_test respectively. Data set up is done for Pycaret. Here, the target (i.e., y values or dependent variable) is the CONFIDENCE column. Thus we are going to train our model with the same data (as was used in the previous step to train the best_model) with the difference that now the target values are those in the CONFIDENCE column.

4.3.3.0. Lightgbm is trained and the model is called M2. Predictions are made on train and test data. As there are 4 different classes in the target column (CONFIDENCE column), 4 different score columns are generated corresponding to each class. New data frames are created by adding score columns to the original data set (i.e., X_train_feature_selected_with_target and X_test_feature_selected_with_target). New data sets are named X_train_data3 and X_test_data3.

4.4.0.0. <u>LDA and new feature</u>

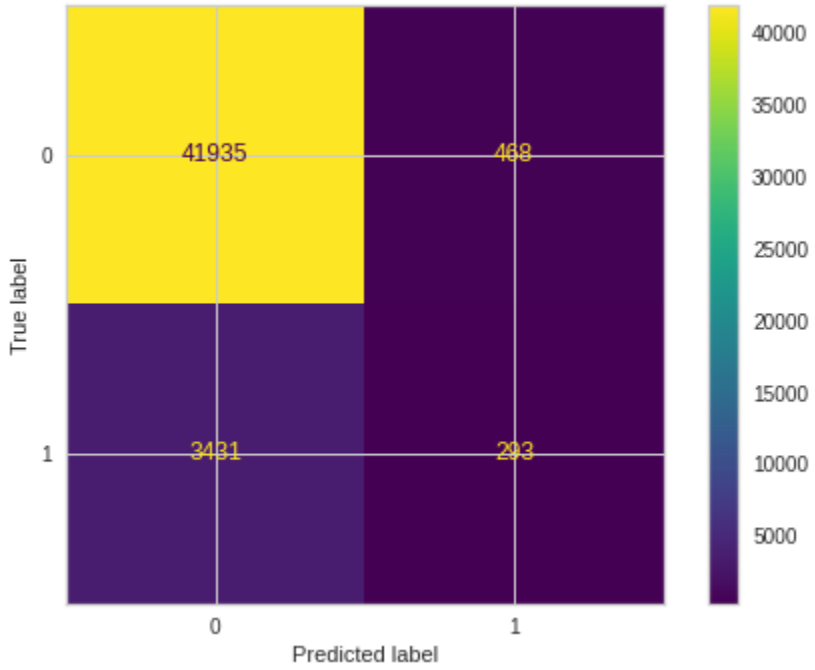
4.4.1.0. LDA is performed on X_train_data3 and X_test_data3. Based on LDA a new feature column named LDA is added to both train and test data (i.e., X_train_data3 and X_test_data3).

4.5.0.0. <u>Train model with new features</u>

4.5.1.0. LGBM: Lightgbm is trained on the dataset with added features (X_train_data3) and further tuned. This tuned model is called lgbm_added_features. Predictions are made on test data (X_test_data3) and corresponding accuracy, AUC and confusion matrix are obtained. Following are the observations when compared to best_model (which is the best tuned model on original train data i.e., X_train_feature_selected_with_target. best_model can be referred to in section 3.1 of this phase):

- Overfitting is observed with substantial difference in accuracy & AUC values between predictions on train data and test data.
- Although AUC has increased compared to best_model, accuracy has decreased.

Based on the above observations, it is concluded that lgbm_added_features is not better than best_model. Confusion matrix from the ipynb file is reproduced here as a sample.



4.5.2.0. Stacked Model: Stacked model is trained using pycaret - Logistic regression, random forest & lightgbm are used as estimators; and lightgbm is used as predictor. This model is called stacker. Predictions are made on test data (X_test_data3) and corresponding accuracy, AUC and confusion matrix are obtained.Following are the observations when compared to best_model:

- Overfitting is observed with substantial difference in accuracy & AUC values between predictions on train data and test data.
- Accuracy and AUC have decreased compared to best_model.

Based on the above observations, it is concluded that stacker is not better than best_model.

4.5.3.0. Neural Network: Neural network is trained. This model is called NN_model. Following are the observations when compared to best_model:

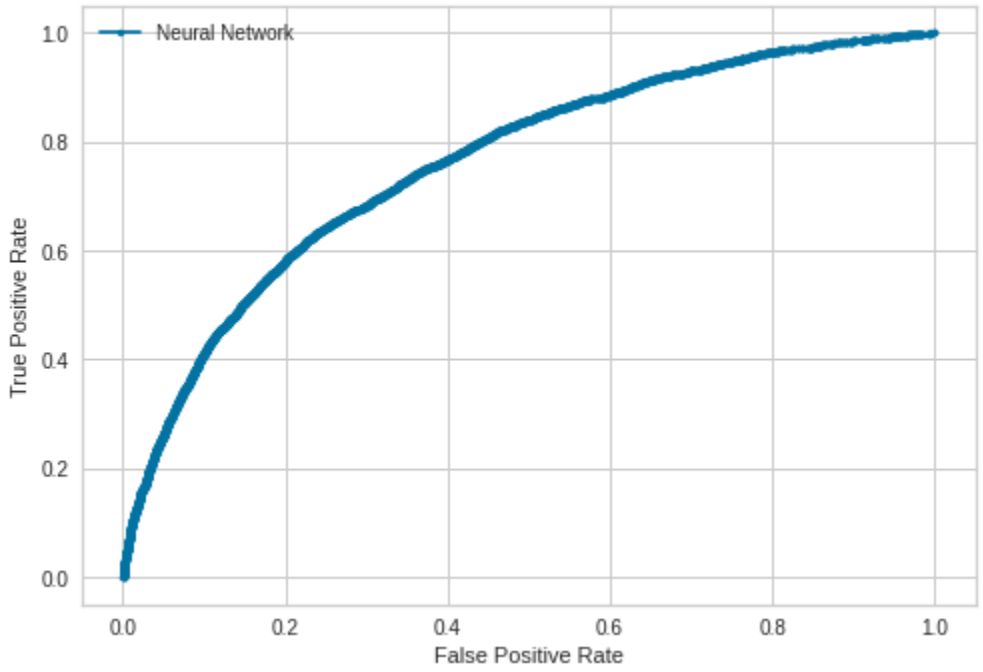
- Overfitting is observed although not major.
- Accuracy and AUC have decreased compared to best_model.

Based on the above observations, it is concluded that NN_model is not better than best_model.

4.5.4.0. Neural Network for original data: As a trial, Neural Network is trained on original data (data without additional features). This is done here as the Neural network was not tried for original data. Following are the observations when compared to best_model:

- Overfitting is not observed.
- Accuracy and AUC have decreased only slightly compared to best_model.
- The number of people who should have been rejected for loan but are predicted as eligible for loan is huge.

Based on the above observations, it is concluded that model_feature_selected is not better than best_model. AUC plot from ipynb file is reproduced here as a sample.



4.6.0.0. <u>Conclusion</u>

4.6.1.0. After trying so many models with so many different conditions (across phase 3 and phase 4), it is observed that tuned lightgbm is the best performer. Even deep learning models fare poorly as compared to tuned lightgbm. Also best results are obtained on feature selected data.

4.6.2.0. Many permutations and combinations are applied to reach this conclusion. However, more approaches can be tried which may or may not give better results. Such approaches can be tried with availability of time and resources.

4.6.3.0. This phase concludes the model training step. Further the trained model shall be deployed in the coming phase.