Report on approach to solving the classification problem

Firstly, we imported the dataframe under two names:

1. original_df
2. df

These two copies were kept just in case there occurred some error in the data cleaning process.

1. Missing Values
   There were plenty of missing values in the dataframe in multiple columns. The dataframe that had more than 4000 missing values which is equal to 90% of the data, it was better to drop the entirety of the column itself. The columns that were dropped during this process were **Gender.**
   However, there was another column named "**Home Ownership**" which had 4520 missing values as well but instead of dropping those values, it seemed plausible we could replace them with a new value named "None" since, this column was significant for the model building.
   The other columns that were missing values were the **Income** and the **Online** columns.
   The Income column was filled with the median because there were many outliers in the data which we saw visually through the boxplot.
   The Online Column was filled with the most frequent value (mode) since it was categorical column.

2. Out of Range Values:
   Upon inspecting the summary statistics of the data, we saw certain columns possessed the value that were not practical and just not possible to occur in real life like Negative Experience, Age of People over 100 etc. So upon inspection the columns like Experience that had values less than 0 were removed since the number of faulty data was less than the threshold (5%) of the data, so we dropped them. The rows with Age column that had the ages above 100 was also dropped because the correction of the data did not seem to work logically.

3. Making a new column “Has_Mortgage”

   The column “**Mortgage”** had the values starting from 0 to the actual values of Mortgages of the customer. The value 0 meant that there was no mortgage of the customer. So, a new column named Has_Mortgage was created that took values 0 if the customer had no mortgages that is 0, and 1 otherwise. This column was helpful in preparing the model, as it gave more information about the customer

4. Changing into categorical column

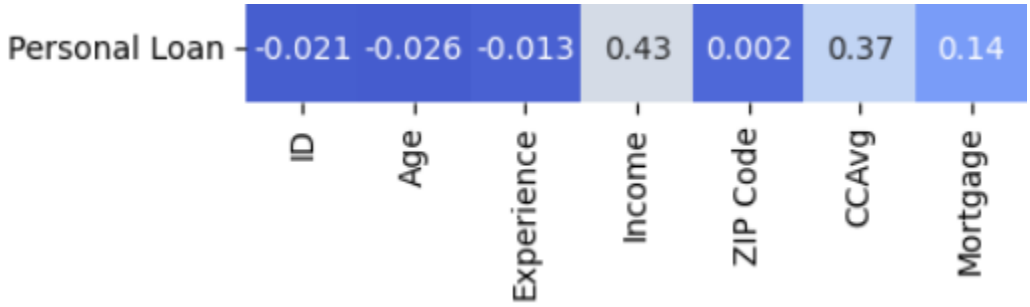
   The columns like Securities, CD Account, Online, CreditCard column were changed to categorical columns because there data was in categorical dtype and it would be helpful to use them as category during summary statistics presentation as well as graphical representation.

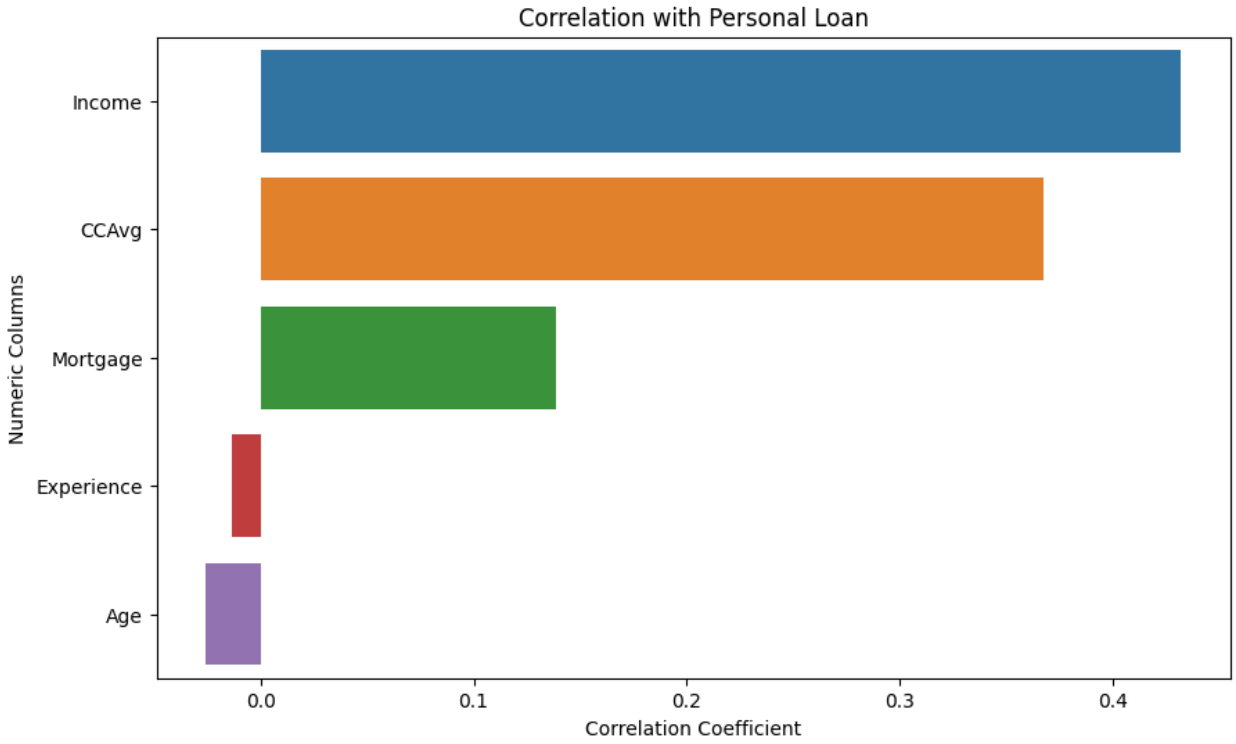
5. EDA

   We performed univariate analysis for both continuous and categorical variables. For continuous variables, we grouped the column by our target column “Personal Loan” and looked at the distribution using histogram and boxplot diagrams.

   The Categorical variables were viewed as countplot with bars arranged in descending order along with annotation on top of the bar for better view of data.

   We also created a heatmap between the numerical columns and the target column to view which columns affected more and the top columns that affected the Personal Loan column were:





The columns Income, CCAvg, Mortgage which is logically correct as they are related in real life as well.

6. Model Preparation

We chose to prepare our model using one of the ensemble learning methods known as RandomForestClassification since this was a binary classification problem. First we performed OneHotEncoding on the columns with categorical values, then we split the dataset into train and test in the ratio of 8:2, and performed GridSearchCV to find the best parameters. Our model with highest efficiency turned out to be:

```
rf_model = RandomForestClassifier(max_depth=None, min_samples_leaf=1,
min_samples_split=2, n_estimators=100, random_state=2)
```

We then fit the data to the model, and the precision scores and the accuracy scores were perfect which could be a hint to overfitting of data.
Precision Score of 1.0, Accuracy Score of 1.0 and Confusion Matrix of:

```
array([[908,   0],
       [  0,  81]], dtype=int64)
```

Was seen as the performance of the model.

The model was then exported in pickle format, which could be used in other files as:

```
import pickle

# Load the saved model from file
with open('rf_model.pkl', 'rb') as file:
    loaded_model = pickle.load(file)

# Use the loaded model for prediction
new_predictions = loaded_model.predict(new_data)
```

In this manner, we explored the dataset, performed feature engineering, data cleaning and preprocessing along with selection of best model and fitting. The detailed step by step execution and approach to the problem is in the notebook file provided with this report.