Hazwan’s project report:

Mohamad Hazwan Bin Mohamad Amin

PC02

2304983F

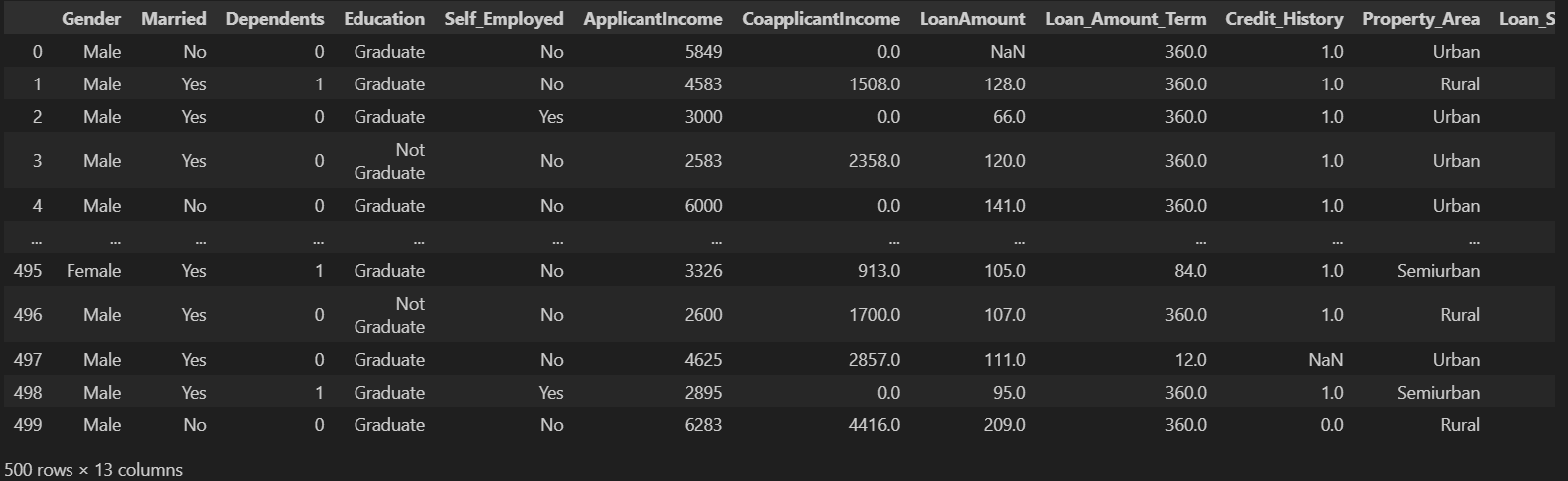
ii. Introduction – description of the topic and the dataset of choice.

|  |
| --- |
| The Topic is about loan acceptance prediction and Dataset is a set of data about past loan acceptance and its acceptance details. It has all the details of the applicants from where they stay, salary, loan amount, etc.  My data will predict the loan acceptance of applicants applying for a loan.  The model im making will be able to predict according from the dataset and based of it, will learn and will predict accordingly.  To learn more about my data, I used df, df.describe and df.nunique to get a sense of what I have. Seeing the rows, columns, the count, mean, std. Theres about 500 rows and 15 columns. This model im making with this dataset will be a classification one. The final output to be checked will be the loan status, whether approved or rejected. |

About the dataset, quoted from Kaggle:

|  |
| --- |
| Among all industries, Banking domain has the largest use of analytics & data science methods. This data set would provide you enough taste of working on data sets from insurance companies and banks, what challenges are faced, what strategies are used, etc. This is a classification problem. The data has 615 rows and 14 features to predict weather loan approved or not approved.  Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set. |

iii. Data Exploration and Pre-processing of data – description of the steps  
that you have taken to explore the datasets and the pre-processing of data



This image is the data before cleaning.

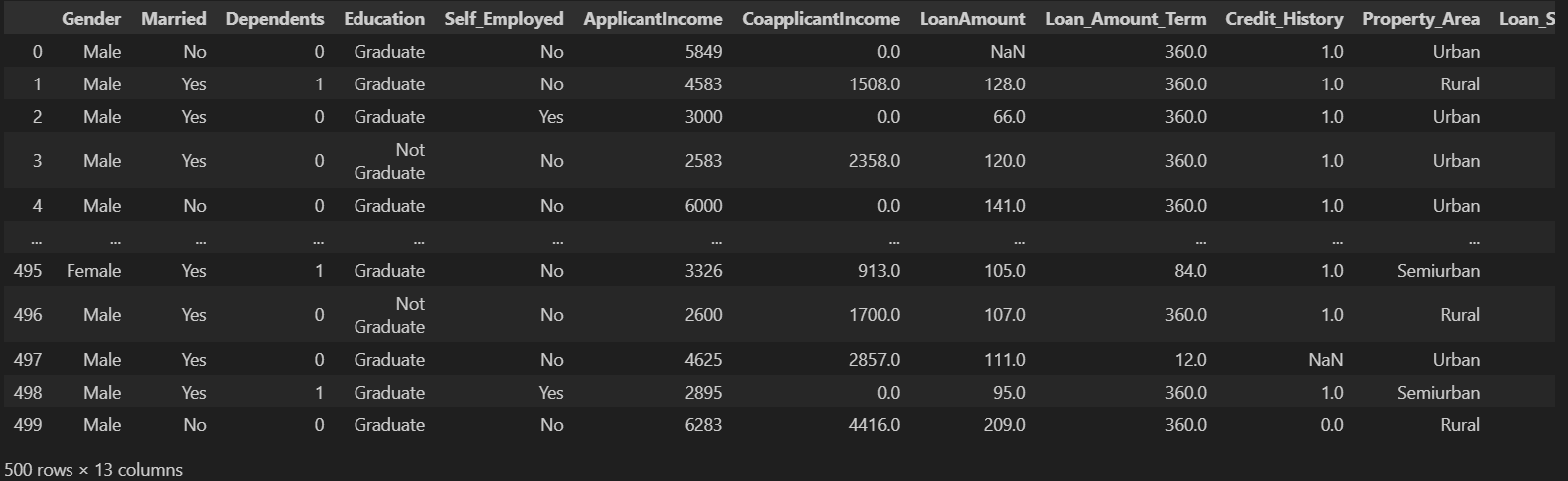
500 rows x 13 columns

Filled with null data, not compatible for models as some variables are in string.

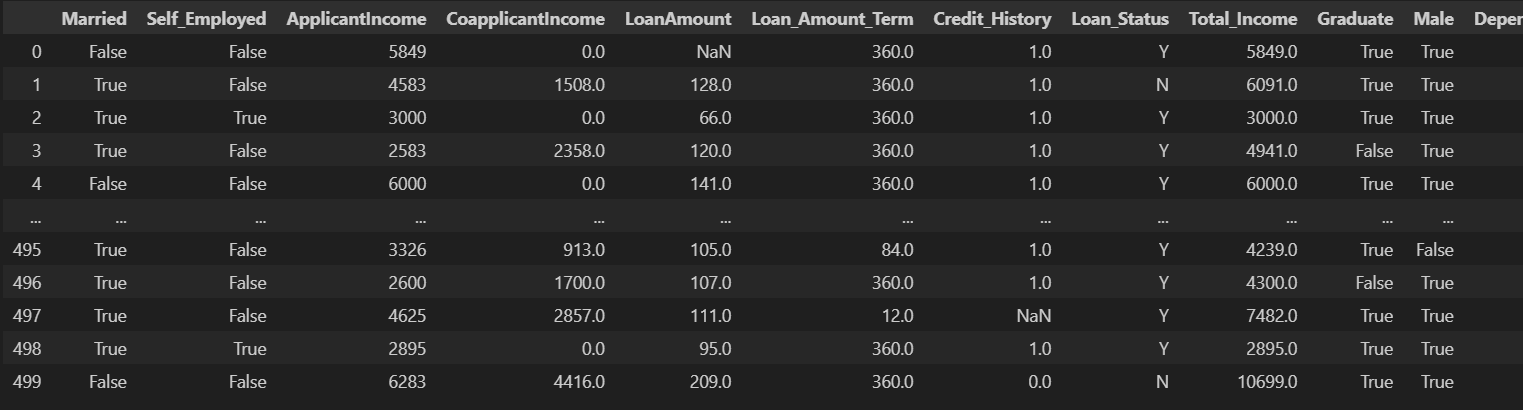
Changes:

1: Changing of data variable,

Before:



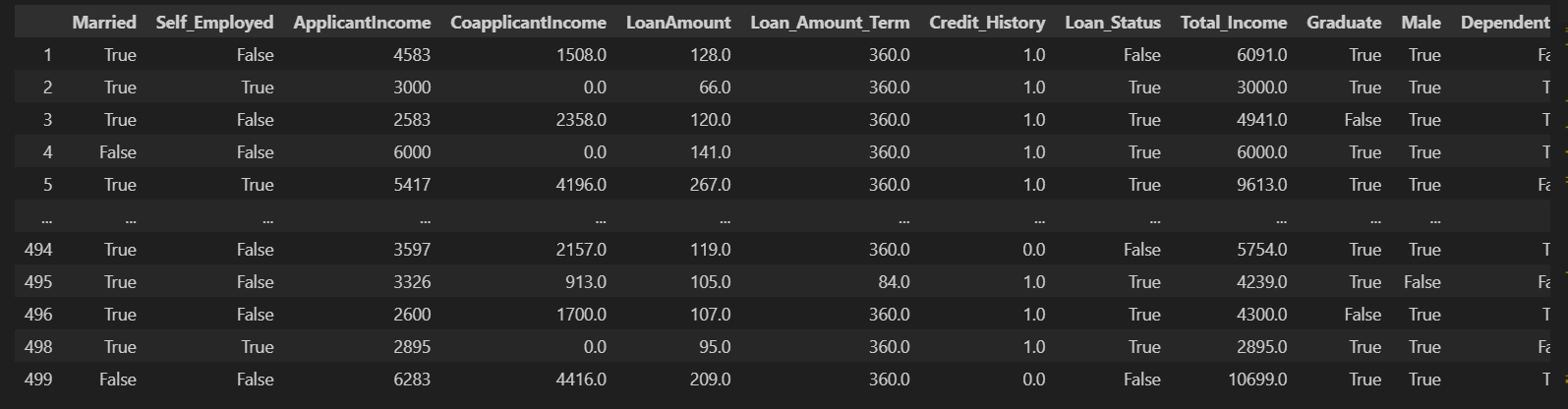
After:



Reason:

After viewing the data and its stats, the first thing I did was to simplify the variables. I changed variables with 2 types of answers to Trues and Falses for easier data sorting. This also makes it easier for conditional events as it is now in 0s and 1s. This also made new tables or replace current ones so that the data will make sense with the new changes

2:



Reason:

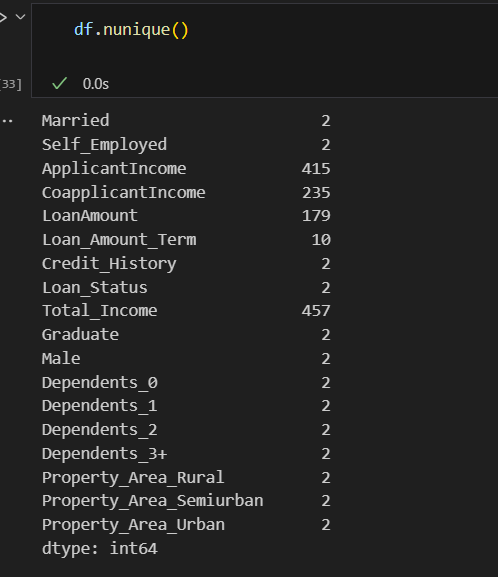
After step 1, certain data columns have been recreated and there a need to drop the redundant data. So to clean this up I run this step to ensure I only have what I need. This is done using df.drop.

3:

OHE



Reason:

Next is to run OHE on the data columns with multiple data variables in it. Doing this made them into true and falses based on their variables, turning them from categorical to Boolean. So for how many dependents a person has, it went from having answers from 0~, it now has columns for 0 dependent, 1 dependent and 2 dependent, having it written true and false according. With the same being done to Property\_Area as well, this is to make sure the it can be more compatible for the models it will be put into.  
  
Result after:  


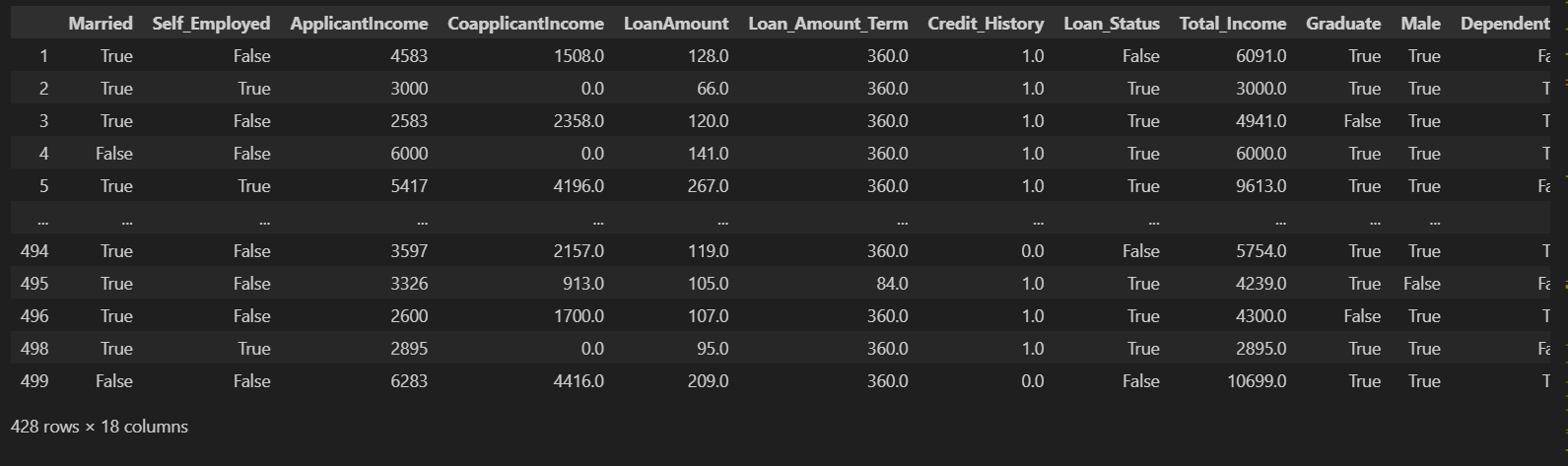
3:

Reason:

Next up, what I did was to clean the dataset of the null values, this can be found in the data columns LoanAmount, Credit\_History and Loan\_Amount\_Term. Using .notna(), I remove the rows with any null variables in them. Doing this as to not have any errors or issue with my dataset later on when using my models and to make a better and more accurate dataset that doesn’t have any holes of data.

The data after:

428 rows x 18 columns

No more null columns, categorical data turned into Booleans, (0/1,true/false)  


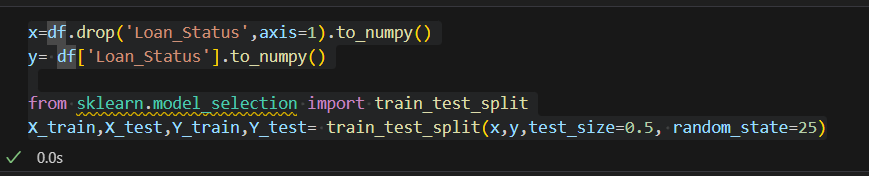
iv. Methods and Improvements-

The first step taken before any model change is to train\_test\_split my data to get my x test, y test, x train, y train.

Using the precision score to evaluate how many of the loan applicants that your model predicts as "approved" are actually good candidates for the loan and if they are actually worth of the loan.

Using Recall score to evaluate to check how creditworthy my model is in checking the applicants. If its low, it means the rejecting too much even if they are right and doing false negatives. So having a higher recall value would be better for the model.

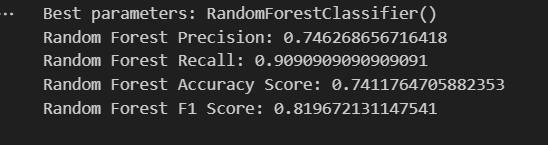
Finally im using an accuracy score as well per model to find an overall value of my models accuracy. If its high, it means that most of its prediction are correct both approved and rejected applicants.



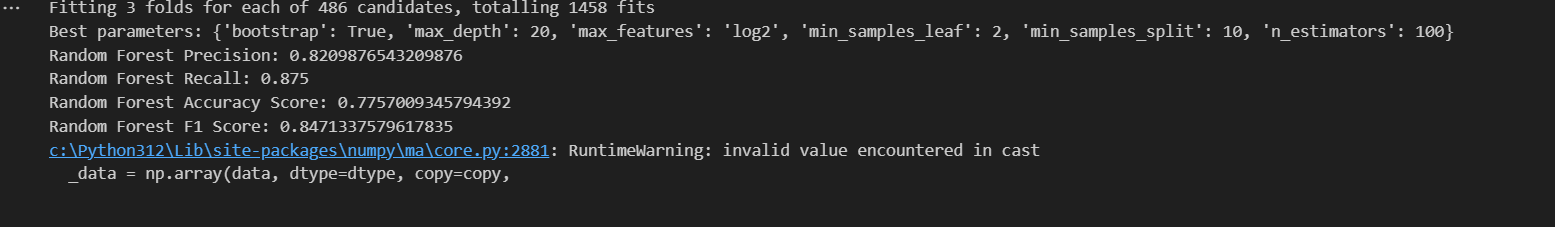
RandomForestClassifier:

This was the main model I chose to follow for my predictor. This model went through hypertuning in order as well to better and improve my models and find the best perimeters for my model as well as balance the underfitting and overfitting. I choose the n\_estimators to be [100,200,300] as typically increasing the number of trees will improve the models performance by reducing variances. But not too much that it will delay my computation timing. The values [100,200,300] balances performance and computation timing. As for my max\_depth, I set the option to be (10,20,none) as this sets the ability to test the shallow trees and while also controlling the overfitting. The value of none lets the trees to grow until all leaves are fewer than the min\_samples\_split. As for the Min\_samples\_split, is to combat overfitting, its set as (2,5,10), allows for exploration of strict splits with (2) and also more relax splits with (5,10). Max features will consider the best split. Using sqrt , log2 and none. Sqrt is the normal default for classification tasks. Log2 takes the logarithm of the number of features and reduces the subset. While none will consider all options, potentially making my model. Also doing gridsearch after to continue the tuning.

Before hypertuning:



After hypertuning:



Bias checks:

v. Results and Analysis

Results for LinearDiscriminantAnalysis:

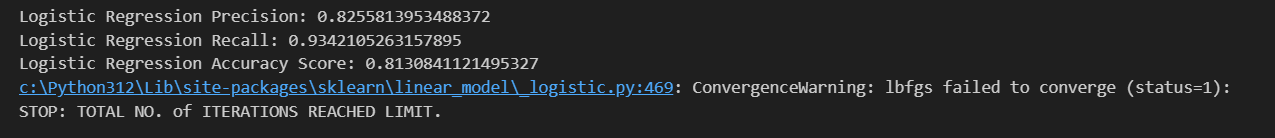
|  |  |  |
| --- | --- | --- |
| LinearDiscriminantAnalysis | Before hyper parameter tuning (3sf) | After hyper parameter tuning (3sf) |
| Precision score : | 0.820 | 0.710 |
| Recall score : | 0.961 | 1.000 |
| Accuracy score : | 0.822 | 0.710 |

The data came out quite slightly worse after hyper tuning with the accuracy score decreasing after and its precision dropping as well. The hyper tuning as well takes a very long time going up to 25 minutes. I proved this method inefficient and decided to drop it and try another model. Choosing to comment it out as to not let my wait time of the hyper tuning to delay my code.

Results for LogisticRegression:

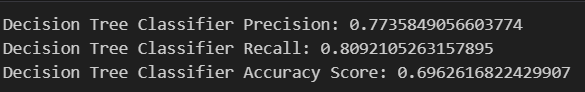
|  |  |
| --- | --- |
| LinearDiscriminantAnalysis | Before hyper parameter tuning (3sf) |
| Precision score : | 0.825 |
| Recall score : | 0.934 |
| Accuracy score : | 0.813 |

The accuracy score of the data from this model was quite good going to 0.813. However, I decided not to use this due to the iteration error from the data after trying this model. It seems to not be the best model for my data set and I decided to find another model that could fit it better. Even if the prediction was able to get it right. The error might make the data model itself not as reliable. So I chose not to do this model.



Decision Tree Classifier:

|  |  |
| --- | --- |
| LinearDiscriminantAnalysis | Before hyper parameter tuning (3sf) |
| Precision score : | 0.773 |
| Recall score : | 0.809 |
| Accuracy score : | 0.696 |



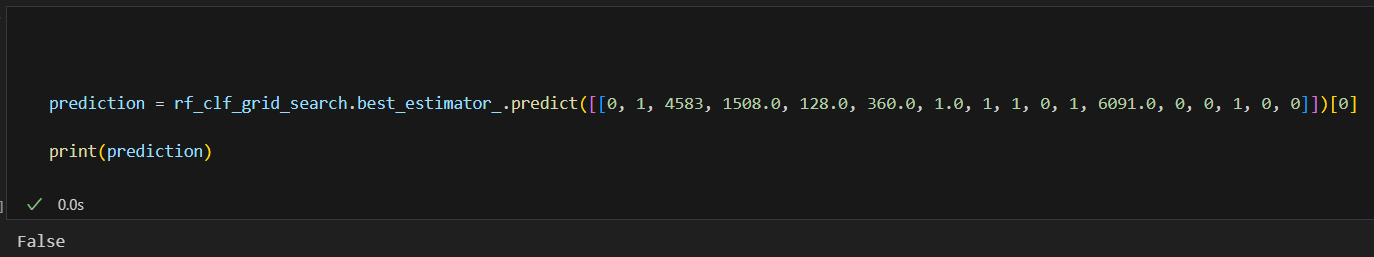
For decision tree classifier, I chose not to use it at all or proceed with this dataset the moment I saw the accuracy score, I didn’t want use it as it was way to low compared to others.

RandomForestClassifier:

Reason for picking this model was because of its accuracy score. Compared to the other models. its accuracy, precision and recall score is one of the highest. Even if linear discriminant analysis has a higher classifying score. I chose not to use it as it’s hyper perimeter tuning only makes it worst and not as good as random forest classifier.

when compared to before and after the hyper parameter tuning, it has improved my overall model. Accuracy score, f1 score, precision score increasing, however, the recall score did decrease between. This could because of some variables like gender having an imbalance in the amount between variable, male and female.

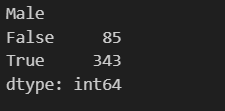
The model does work well in a prediction test as shown.



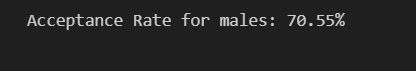
It is able to properly approve or reject an applicant when its similar to the data used from the original dataset.

However, though the model is working as from the data, one thing noticed is the possibility there is a bias. For example, with gender.

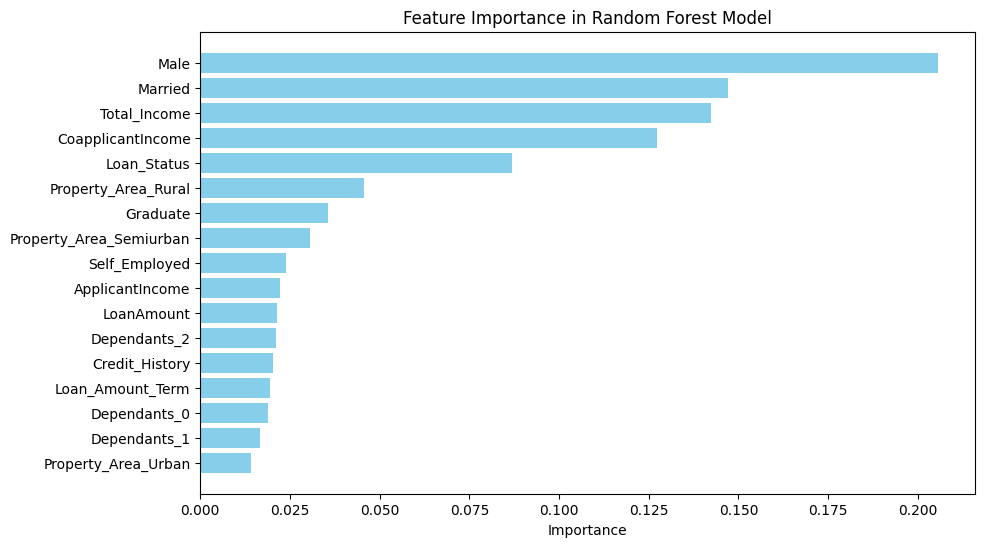
When checking the details of the ratio between male and female, theres a huge imbalance between the 2. Thinking about it made me think about whether gender makes a difference and whether it holds any bias on my model. To check, I also checked the difference in acceptance rate between male and females. At the moment, the acceptance rate between male and female is 70.55 and 62.55 respectively.





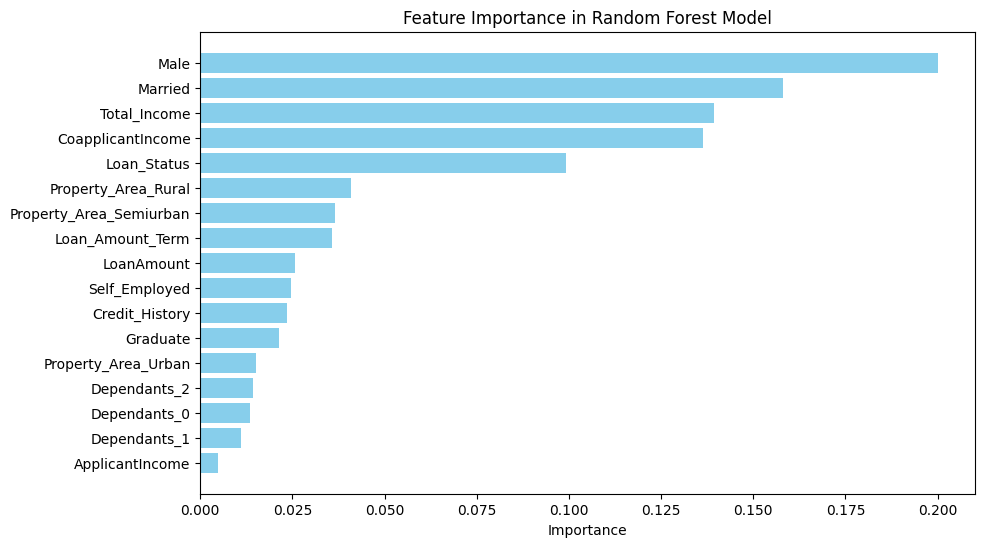


To further check, I also did a feature importance graph to have a more visual look and to see the importance.



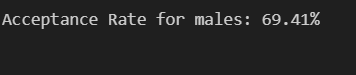
As of now, the importance of the gender, aka male, is currently the highest. To test if it’s a bias between gender. I will run a test with a new data set, I will do the same cleaning and train test. However, in this data set, I shall sample it to have an equal amount of male and female applicants. The importance of gender is 0.1933.

In the end, this is the result.

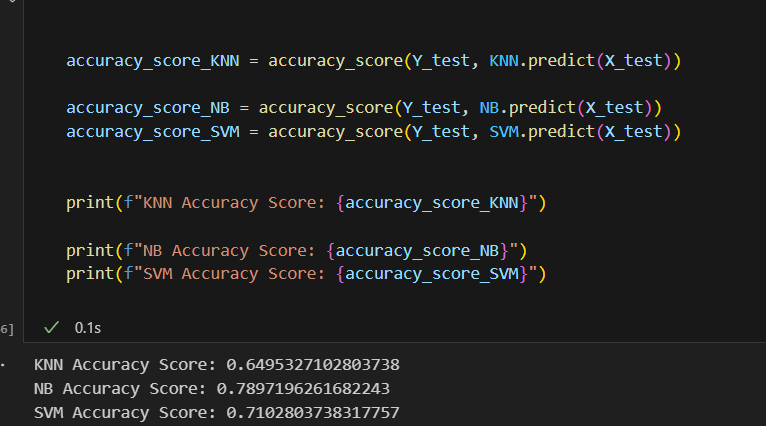


There is now a change in the feature importance graph. Now that its equal, the importance of gender has dropped slightly. The importance rate dropped by 0.0208 and went to 0.1725 the acceptance rate as well does not change for female but it did slight increase for male. This tells me there is a bias in the acceptance rate for males and females. Even when the numbers are equal between applicants. Therefore, model/dataset has a tendency to pick males over females, creating this bias. However, the difference between the acceptance rate is still not so bad, roughly, 8-10% that I deemed it not to problematic to the model/dataset even if gender is the most important feature.

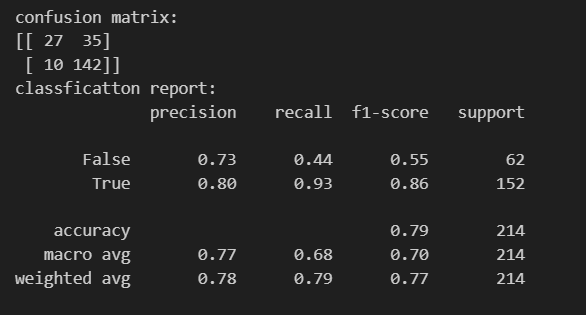




Others:



I used KneighbourClassifier, gausianNB and SVM models as an additional models just to test. However, For KNN and SVM, the low accuracy score made me not proceed further and chose not to use this model. However I tried to proceed more with NB. However, after doing the a confusion matrix and classification report. Even though the output was good, I chose not to. Mostly due to the fact I already went through with random forrest classification. But I will acknowledge that the data is really good and if I hadn’t use random forrest, I would have used this instead.



vi. Conclusion

In conclusion, I believe made the right choice on my model choice with random forest classifier. It was able to help me produce results that are pretty accurate according to the dataset. It is able to predict accurately when given data from the original dataset.

The other dataset either had really low accuracy rates or had other problems, whether with the hyper parameter tuning timing or its results.

The model Linear discriminant analysis had a good classification report output. However, due to its super slow hyper parameter tuning, I opted not to use it. It took close to 25-30mins just to get the tuning. And the tuning was not better either as it instead lowered the classification report output afterwards as well after the fitting and tuning.

When it came to good dataset, the GaussianNB dataset had really good classification reports. If I had done with this first I would have used this model instead. However, due to time and choice, I didn’t use it. Another dataset I would have I used as well if it wasn’t for the technical iterations error during classification testing, it would have been logicalregression.

Cleaning the dataset was also important. Dropping the “Load\_ID” as it wasn’t that important. Other than that, I also had to change certain categorical data into more Boolean/true-false type of data to allow for better integration into a model in order to not run into any string issues. Afterwards, also doing OHE to further sort the categorical data. Once cleaning, the data is better fitted for such usage. However, I had to troubles for abit. When converting the string type data, I had to learn how to do so. Using a new method thought by my lecturer, I was able to learn how to convert the data. Other than that, I also had trouble cleaning the data of null data. I was able to drop them after finally using isnull() on the data column that had null elements to finally clean my data. When I was done, my data dropped to 428 rows.

However, This dataset is not without its flaws. It has a bias towards the gender of the applicant and if even when the dataset is even, this bias stays. However, I believe it does not effect the results of the dataset as the difference between the male and female acceptance rate isn’t that badly apart and I chose to continue with this dataset. Using the data from the model, I used the feature importance graph to prove this. Showing the level of importance from gender before and after. The graph also compares it to other features and the difference between them visualized to show who holds the most influence in the dataset’s choice.

What I can do next time with this, is to perhaps drop gender itself. Gender shouldn’t really be important in the choosing of loan. Though in some countries it is. I believe for a fair choice, gender should not be part of the model. And perhaps a bigger dataset might be able to prove this.

Finally, when deploying the model, I chose to deploy with a local storage instead of on cloud via git was because I mostly followed the lab. However, even though I was given the choice, I preferred using local host instead as I felt it was abit faster and easier to handle. I didn’t have to keep updating my git and I just had to save my code on my end and rerun my app. However, it did come with its own issues like accidentally running it wrongly and messing up my model on the app giving it false reports. But overall, I preferred the local storage method of testing and deployment.

vii. References –

Dataset:  
<https://www.kaggle.com/datasets/vipin20/loan-application-data>

Learned about models from this:  
<https://scikit-learn.org/1.6/modules/generated/sklearn.ensemble.RandomForestClassifier.html>