German_Credit_Risk

December 17, 2020

0.1 Introduction

This is a classification project that aims for predicting credit risk or in simpler terms will help the companies to predict bad loans. Predicting the credibility of a loan is very useful in the Banking business. Using machine learning will help to decide, whether to approve a loan or not by accounting for the risk value based on historical data.

The data used in this project is obtained from the Kaggle(https://www.kaggle.com/kabure/german-credit-data-with-risk), which gives detailed information about the historical data of loan approved by German banks. The data contains a feature column that provides whether the loan was good or bad.

So this Classification project aims to predict the quality of the loan approved by German banks using various Classification algorithms. The project is structured as follows: starts with the understanding of the data and an exploratory data analysis of it, then followed by feature engineering and preprocessing section, then a section which details the models applied to the data and outcome of various models. Finally concludes the reports by summarising the findings achieved from evaluation metrics of models.

0.2 Exploratory Data Analysis

The data we will be using is named 'German credit data', which contains details about loans offered by the German banks and which can be used to predict the credibility of the data. A brief glimpse of the data is shown below:

[768]:

	Unnamed: 0	Age	Sex	Job	Housing	Saving accounts	Checking account	\
0	0	67	male	2	own	NaN	little	
1	1	22	female	2	own	little	moderate	
2	2	49	male	1	own	little	NaN	
3	3	45	male	2	free	little	little	
4	4	53	male	2	free	little	little	

	Credit amount	Duration	Purpose	Risk
0	1169	6	radio/TV	good
1	5951	48	radio/TV	bad
2	2096	12	education	good
3	7882	42	furniture/equipment	good
4	4870	24	car	bad

The data contains 10 attributes which are:

1. Age (numeric)
2. Sex (text: male, female)
3. Job (numeric: 0 - unskilled and non-resident, 1 - unskilled and resident, 2 - skilled, 3 - highly skilled)
4. Housing (text: own, rent, or free)
5. Saving accounts (text - little, moderate, quite rich, rich)
6. Checking account (numeric, in DM - Deutsch Mark)
7. Credit amount (numeric, in DM)
8. Duration (numeric, in the month)
9. Purpose(text: car, furniture/equipment, radio/TV, domestic appliances, repairs, education, business, vacation/others
10. Risk (Value target - Good or Bad Risk)

So total data has 10 columns and 1000 rows, that means this data is not that of a big data. Here the Risk column is our target column, which classifies the loan into bad and good categories.

And the table below shows the types of our attributes. There are 6 categorical attributes including our target column and 5 numerical attributes. Since first column 'Unnamed:0' is just a column indicates number of each entry, we will delete this column.

	0
Age	int64
Sex	object
Job	int64
Housing	object
Saving accounts	object
Checking account	object
Credit amount	int64
Duration	int64
Purpose	object
Risk	object

A short description of numerical attributes is displayed below.

	count	mean	std	min	25%	50%	75%	max
Age	1000	35.546	11.3755	19	27	33	42	75
Job	1000	1.904	0.653614	0	2	2	2	3
Credit amount	1000	3271.26	2822.74	250	1365.5	2319.5	3972.25	18424

```
| Duration      |    1000 |   20.903 |   12.0588   |     4 |   12   |   18   |
24    |    72 |
```

From the short glimpse of our data, we can see that there were missing values for some attributes in our data. So let's check the missing values in the data.

```
|                  |   0 |
|:-----------------|----:|
| Age              |   0 |
| Sex              |   0 |
| Job              |   0 |
| Housing          |   0 |
| Saving accounts  | 183 |
| Checking account | 394 |
| Credit amount    |   0 |
| Duration         |   0 |
| Purpose          |   0 |
| Risk             |   0 |
```

So, the table above shows that there are 183 missing values for 'Savings account' and 394 for 'Checking account'. We will deal with these missing values in the preprocessing section, as for now, we can keep these values for exploratory data analysis.

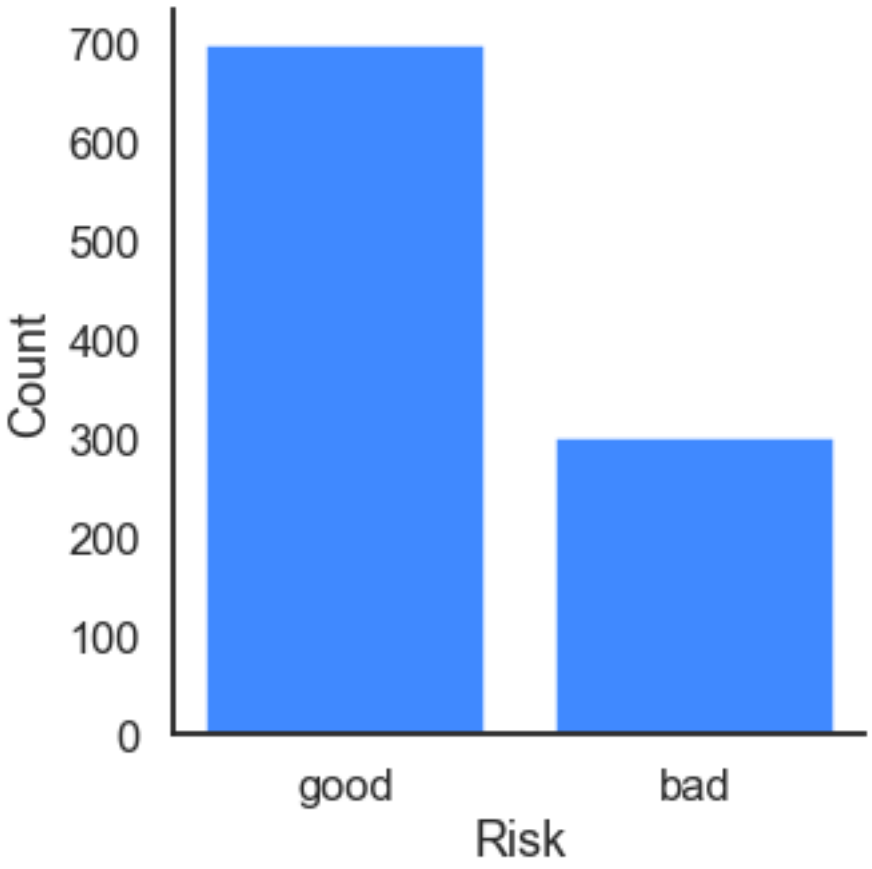
[774]: `['Sex', 'Housing', 'Saving accounts', 'Checking account', 'Purpose', 'Risk']`

First we will look into our target column that is 'Risk', it contains two catagorical value, bad and good loans. The propotion of two classes are 70% and 30% for good and bad respectively.

```
|      |   Risk |
|:-----|-------:|
| good |    700 |
| bad  |    300 |
```

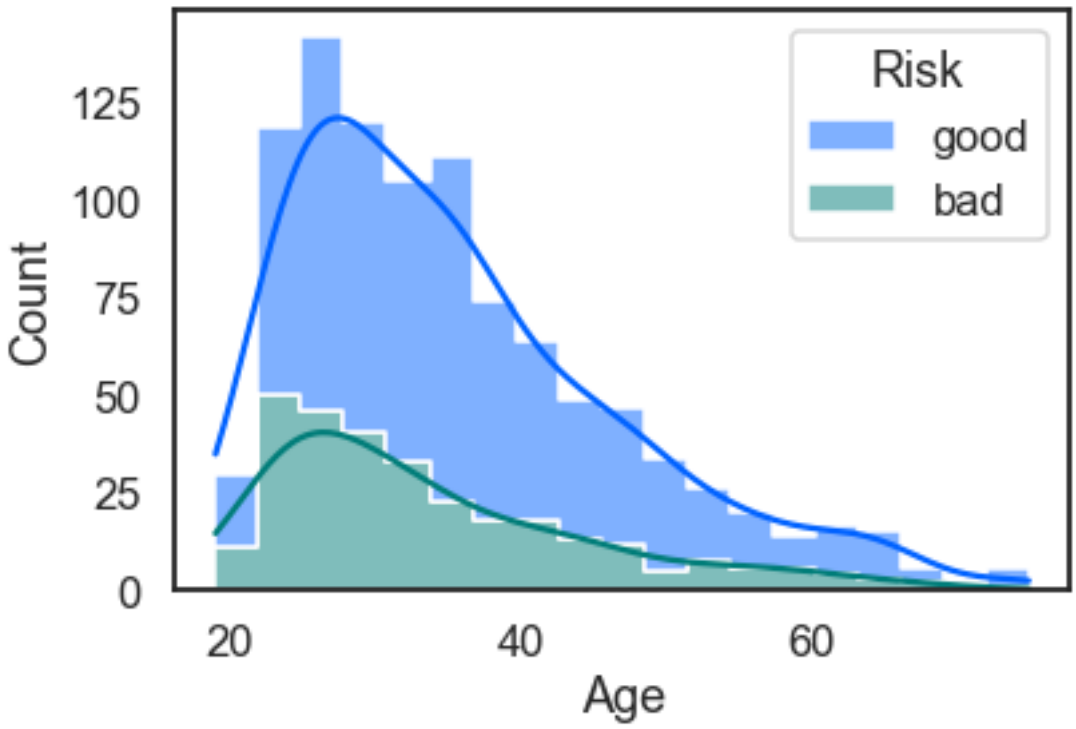
Below is the barplot of 'Risk' column which shows propotion of good and bad class. Eventhough the proption of classes are not balanced, I dont think it will have an notable influence on the predictions of different models, so dont need to balance the target column.

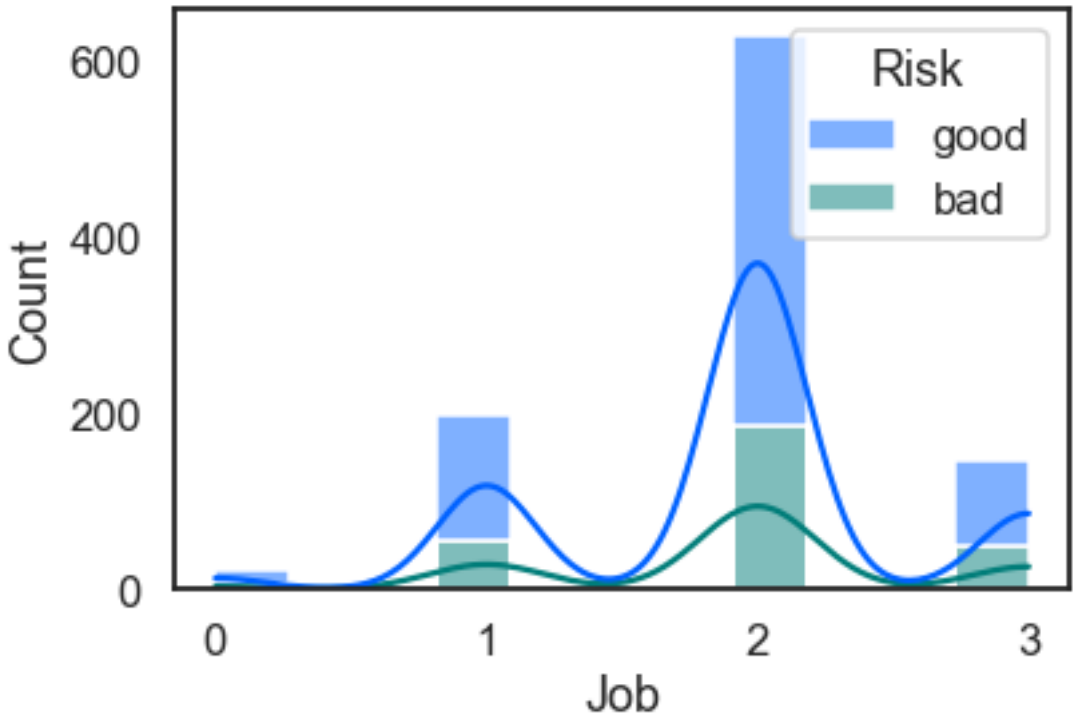
```
<Figure size 360x360 with 0 Axes>
```

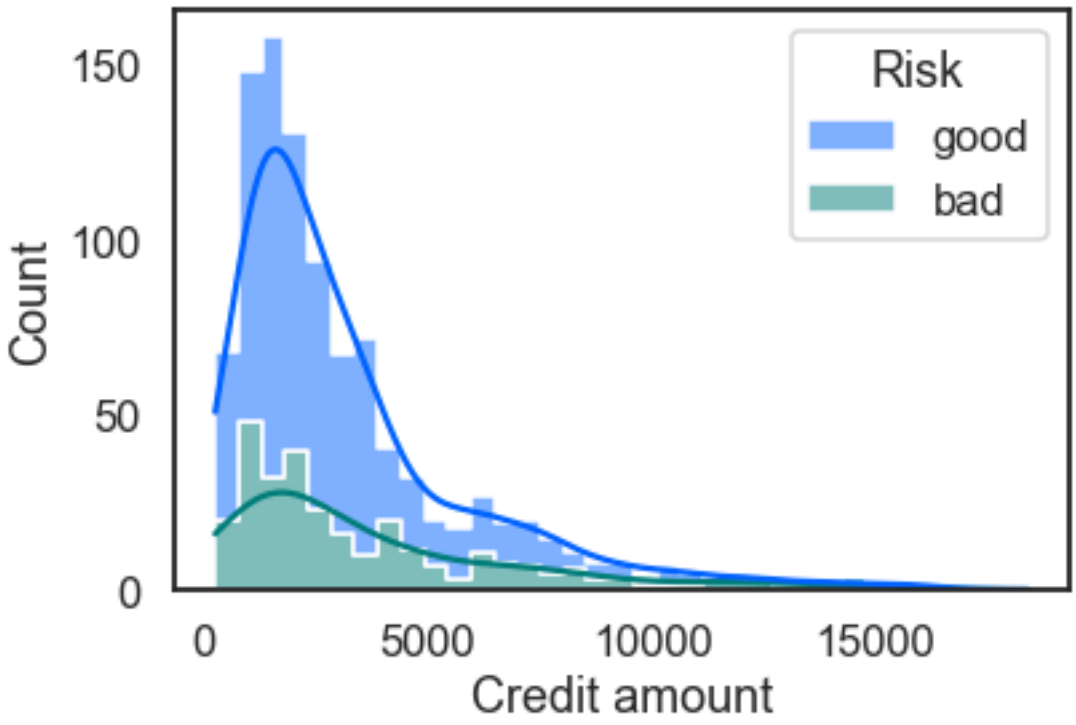


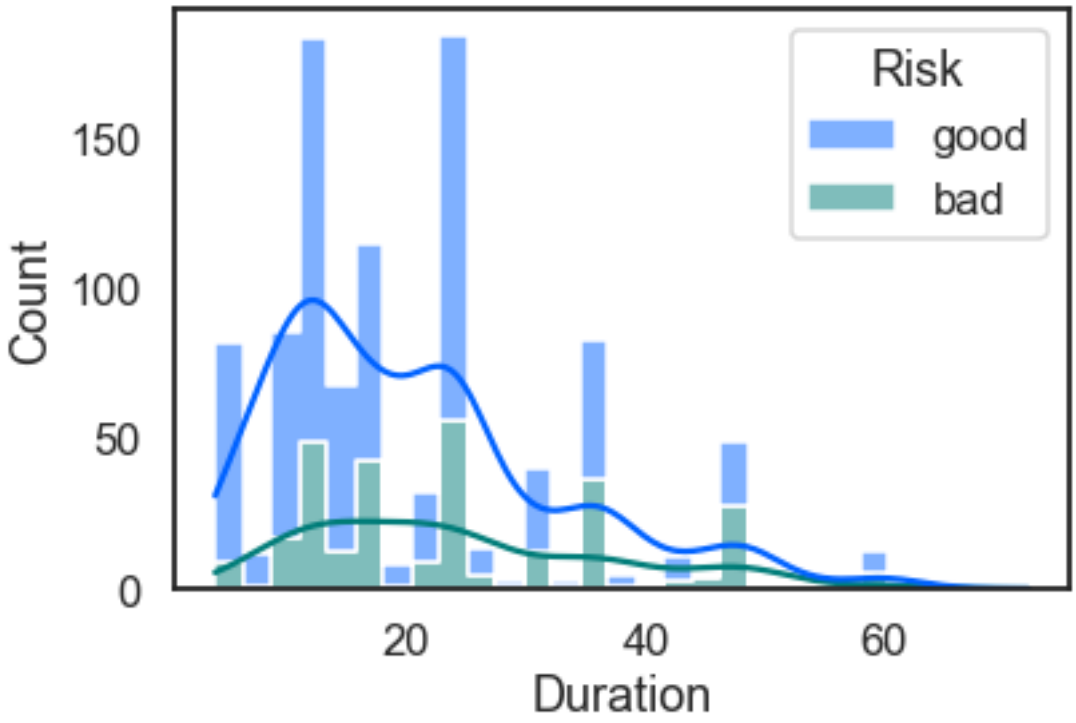
0.2.1 Numerical columns Analysis

In this part, we will look into our numerical type data columns and examine the distribution of each column. So first we plotted a histogram of each numerical column to visualize the distribution of values and separated the distribution for different Risk values to understand the correlation columns with the target columns. Below are the distribution plots for each numerical columns:


Risk
good
bad
125
100
75
50
25
0
Count
20
40
60
Age


Risk
good
bad
600
400
200
0
Count
0
1
2
3
Job

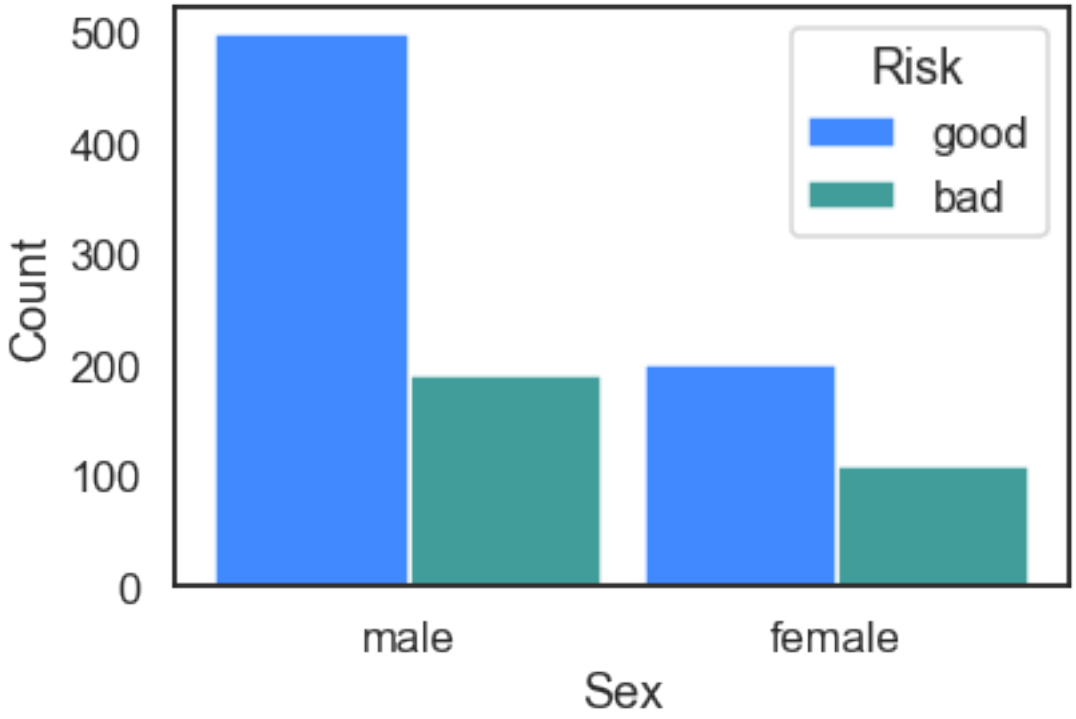

Risk
good
bad
Count
150
100
50
0
0
5000
10000
15000
Credit amount


Risk
good
bad
Count
150
100
50
0
20
40
60
Duration

The distribution of these columns are looks similar for both risk value. The distributions are not normal, they are right-skewed, so we will have to do log transformations on these distributions. Also, we can note that the job column looks like it has a categorical value, not continuous values. So we can include this column too in the categorical category.

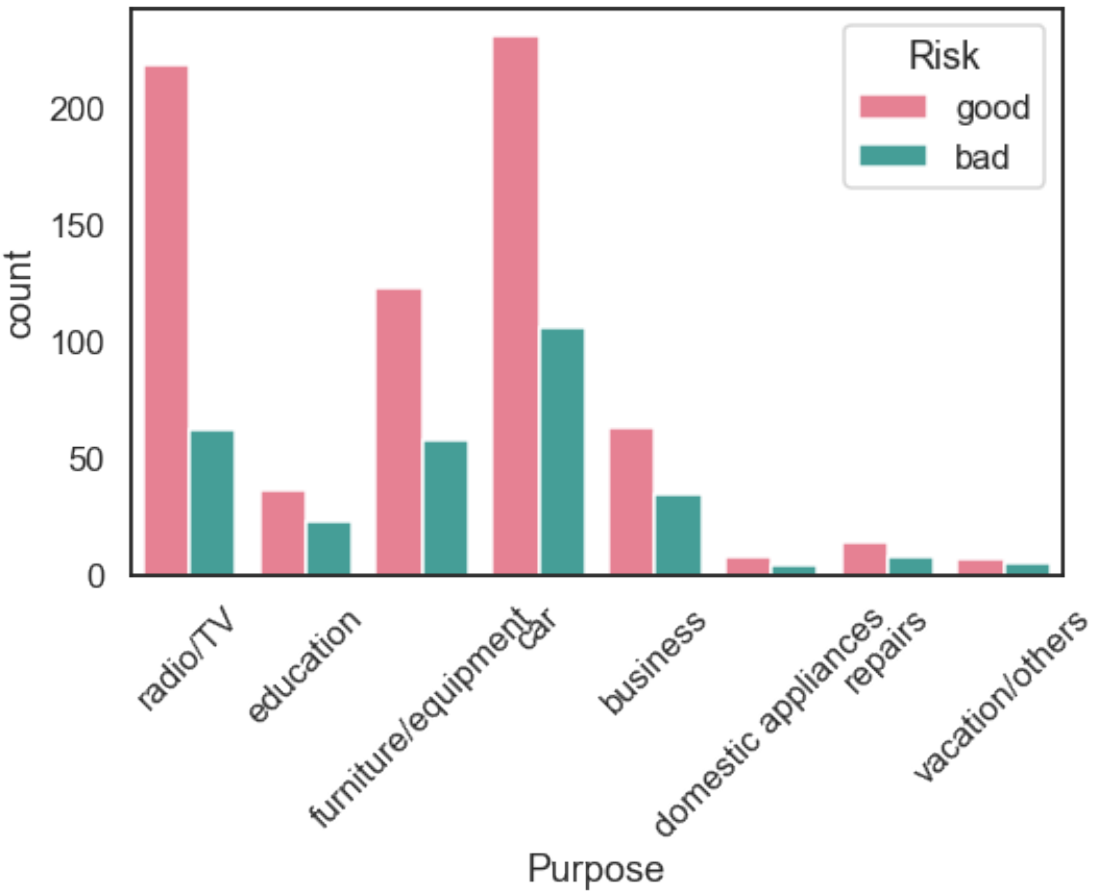
0.2.2 Catagorical columns

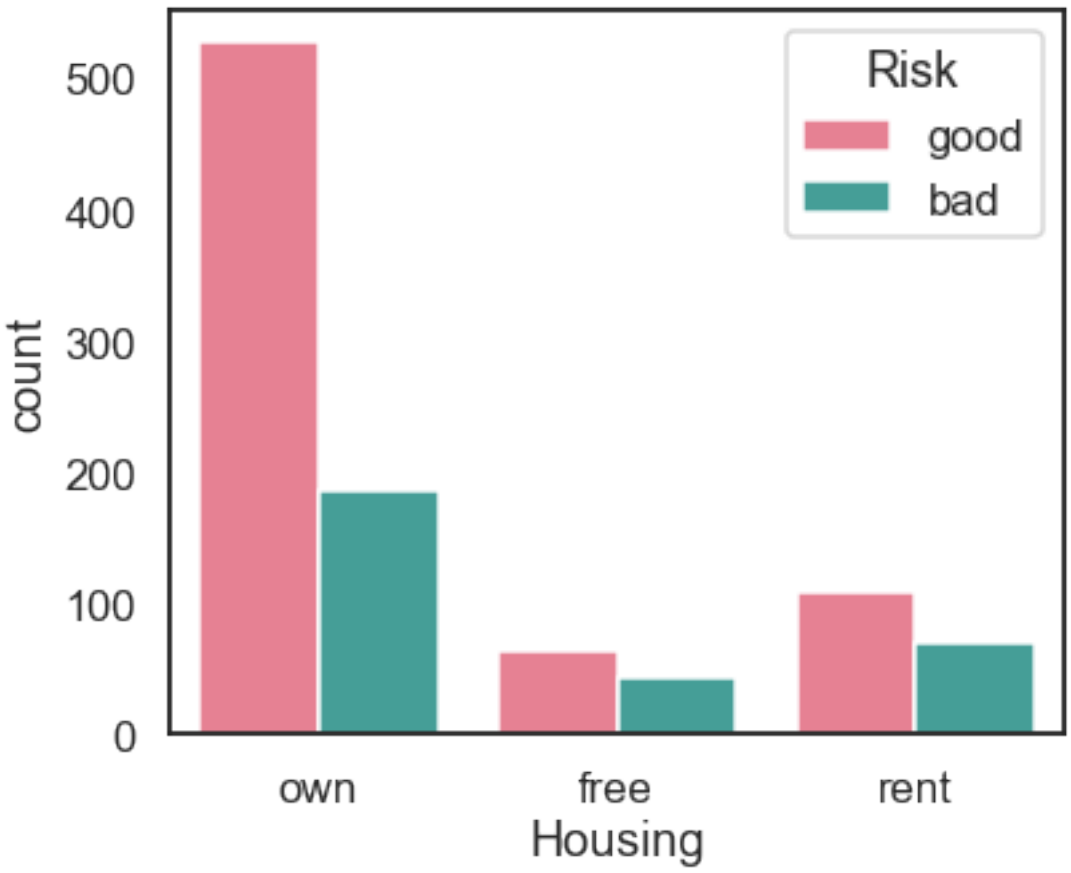
We will be plotting count plots for each categorical column with different target columns. So these plots will show how each categorical correlates with the target column.

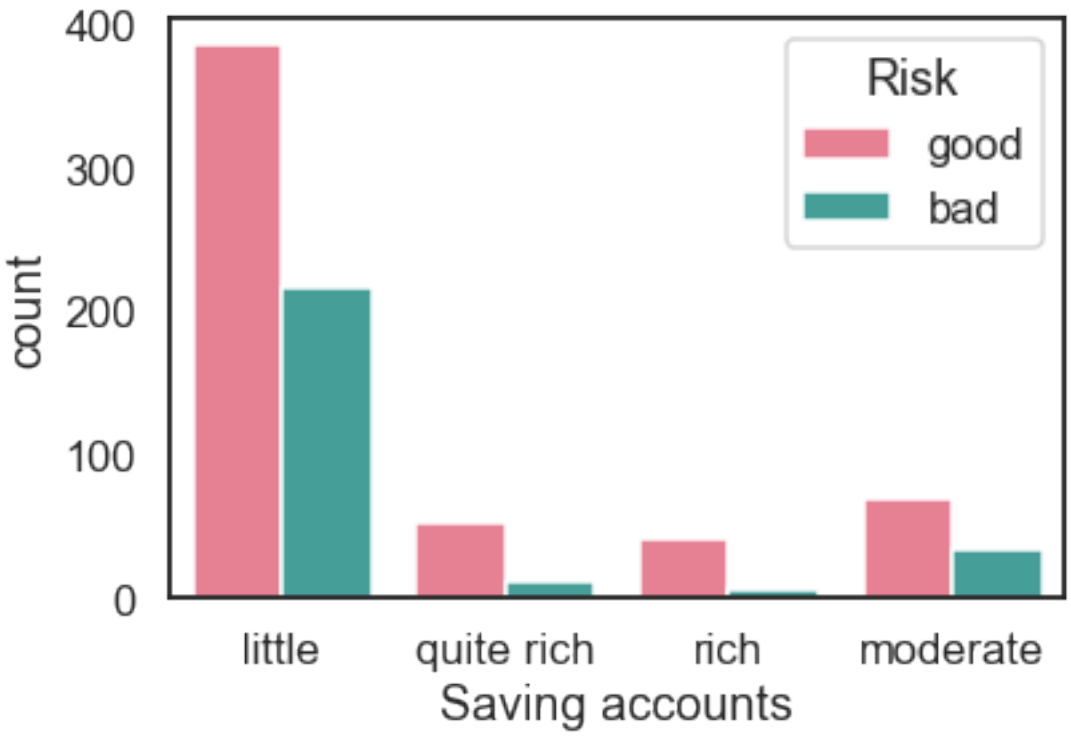


The count plot of sex shows that male customers are higher than female customers. The difference in the proportion of bad and good customers is very high for female customers compared to male customers.

Below is the count plot of other categorical value for both risk values:


Risk
good
bad
count
200
150
100
50
0
radio/TV
education
furniture/equipment
car
business
domestic appliances
repairs
vacation/others
Purpose


Risk
good
bad
500
400
300
200
100
0
count
own
free
rent
Housing



The count plot of the purpose of loan shows that to buy a car is most of the loan has been taken and it has a high number of a bad loan. This purpose doesn't have the highest proportion of good and bad loans. below is the table shows the proportion of different purposes:

	Purpose
vacation/others	0.714286
education	0.638889
repairs	0.571429
business	0.539683
domestic appliances	0.5
furniture/equipment	0.471545
car	0.458874
radio/TV	0.284404

Even though the car purpose has a high number of loans, it doesn't have many bad loans compared to good loans. Similarly, the proportion of risks for the other two categorical values, which are Housing and Saving accounts is given below:

	Housing
free	0.6875
rent	0.642202
own	0.352941

	Saving accounts
little	0.562176
moderate	0.492754
quite rich	0.211538
rich	0.142857

0.3 Preprocessing and Feature Engineering

In this section, we will look into the step taken to clean the data and, the process has taken to change the data into a more useful form.

0.3.1 Data Cleaning

As a first step, we will be taken care of the missing values in the Saving accounts and the Checking account. The number of missing values in these columns is once again shown below:

	0
Age	0
Sex	0
Job	0
Housing	0
Saving accounts	183
Checking account	394
Credit amount	0
Duration	0
Purpose	0
Risk	0

The Saving accounts have 183 missing values out of 1000 values, on the other hand, the checking account has 394 values. Because these columns are categorical values, we will be using the median to replace the missing values.

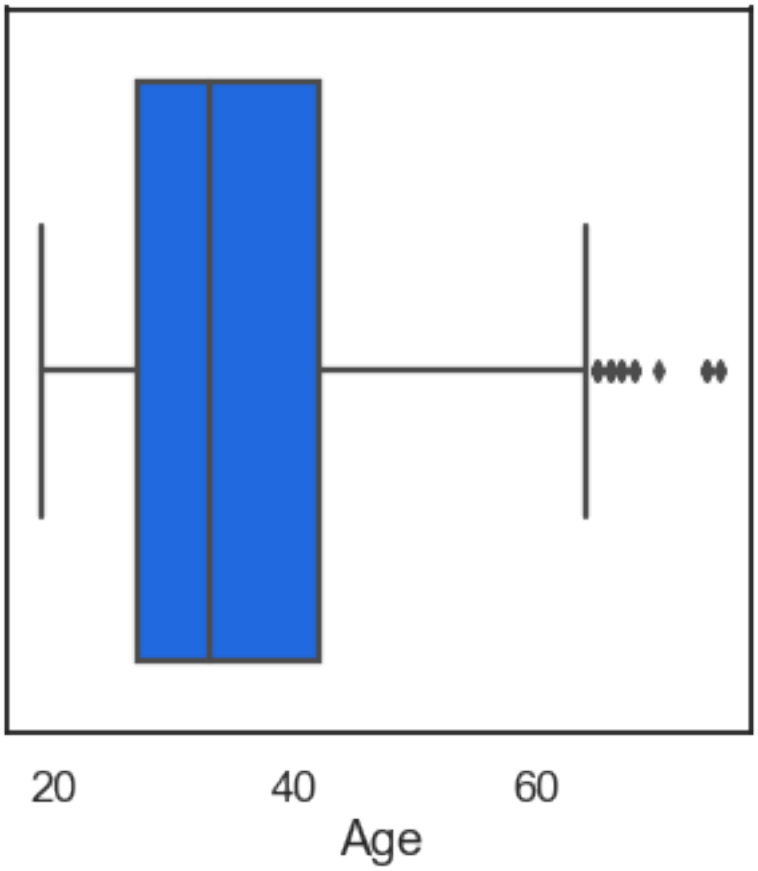
The number of missing values after the replacement is:

	0
Age	0
Sex	0
Job	0
Housing	0
Saving accounts	0
Checking account	0
Credit amount	0
Duration	0
Purpose	0
Risk	0

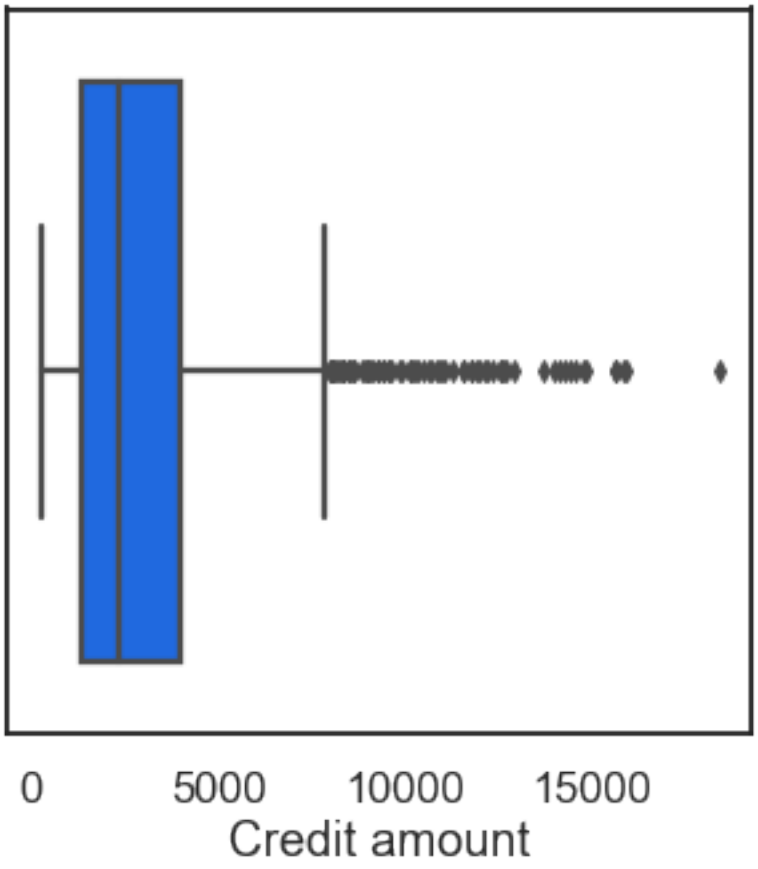
0.3.2 Outlier Analysis

To check the presence of outliers on any column, we checked the values which are higher than 1.5 times Inter quartile Range(IQR) from the upper quartile and lower than 1.5 times the inter quartile range from the lower quartile.

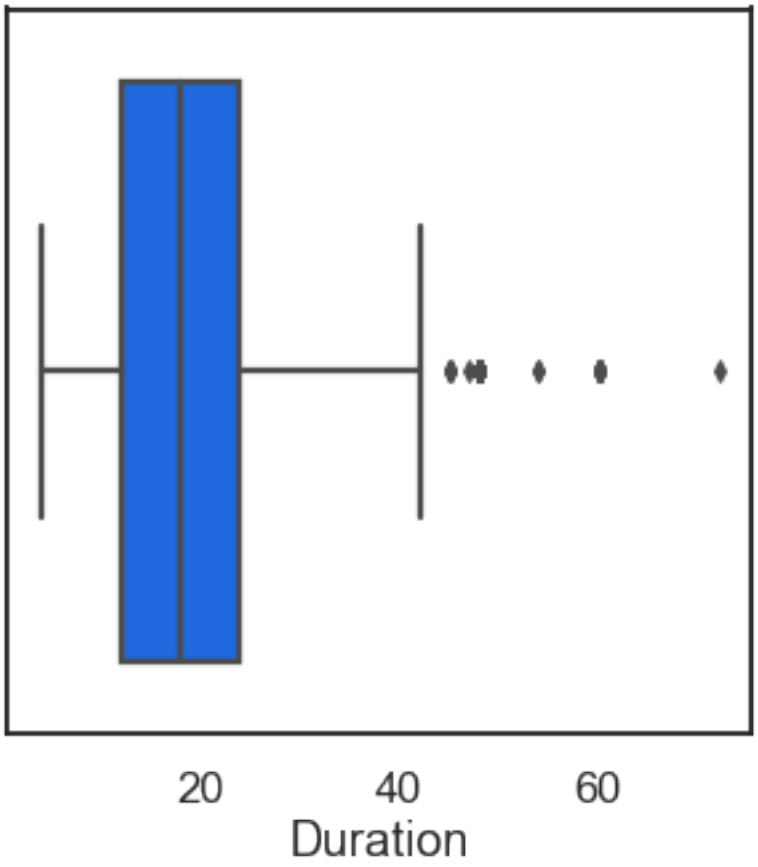
`Age : 23`



`Credit amount : 72`




```
Duration : 70
```




```
['Age', 'Credit amount', 'Duration']
```

It is found out that, there are some outliers in the Age, Credit amount, and Duration columns. The number of an outlier in each column and visualization of outliers on the box plots are given above. To correct these outliers, we will replace outliers that are lower than the lower limit(lower quartile - 1.5 times IQR) with a lower limit and higher than the outer limit(upper quartile + 1.5 times IQR) with the outer limit of the columns.

0.3.3 One hot encoding

As a next step, we hot encoded the categorical variables and converted them into a form that could be provided to ML algorithms to do a better job in prediction.

After one hot encoding the number of columns increases to 20 and new column names are shown below:

[794]:
```
['Age',
 'Sex',
 'Job',
 'Housing',
 'Saving accounts',
```

```
 'Checking account',
 'Credit amount',
 'Duration',
 'Purpose',
 'Risk',
 'Purpose_car',
 'Purpose_domestic appliances',
 'Purpose_education',
 'Purpose_furniture/equipment',
 'Purpose_radio/TV',
 'Purpose_repairs',
 'Purpose_vacation/others',
 'Sex_male',
 'Housing_own',
 'Housing_rent',
 'Savings_moderate',
 'Savings_quite rich',
 'Savings_rich',
 'Check_moderate',
 'Check_rich']
```

	count	mean	std	min	25%	50%	75%	max
Age	1000	35.4535	11.1063	19	27	33	42	64.5
Job	1000	1.904	0.653614	0	2	2	2	3
Credit amount	1000	3051.1	2187.14	250	1365.5	2319.5	3972.25	7882.38
Duration	1000	20.307	10.6152	4	12	18	24	42
Purpose_car	1000	0.337	0.472921	0	0	0	1	1
Purpose_domestic appliances	1000	0.012	0.10894	0	0	0	0	1
Purpose_education	1000	0.059	0.235743	0	0	0	0	1
Purpose_furniture/equipment	1000	0.181	0.385211	0	0	0	0	1
Purpose_radio/TV	1000	0.28	0.449224	0	0	0	1	1
Purpose_repairs	1000	0.022	0.146757	0	0	0	0	1
Purpose_vacation/others	1000	0.012	0.10894	0	0	0	0	1

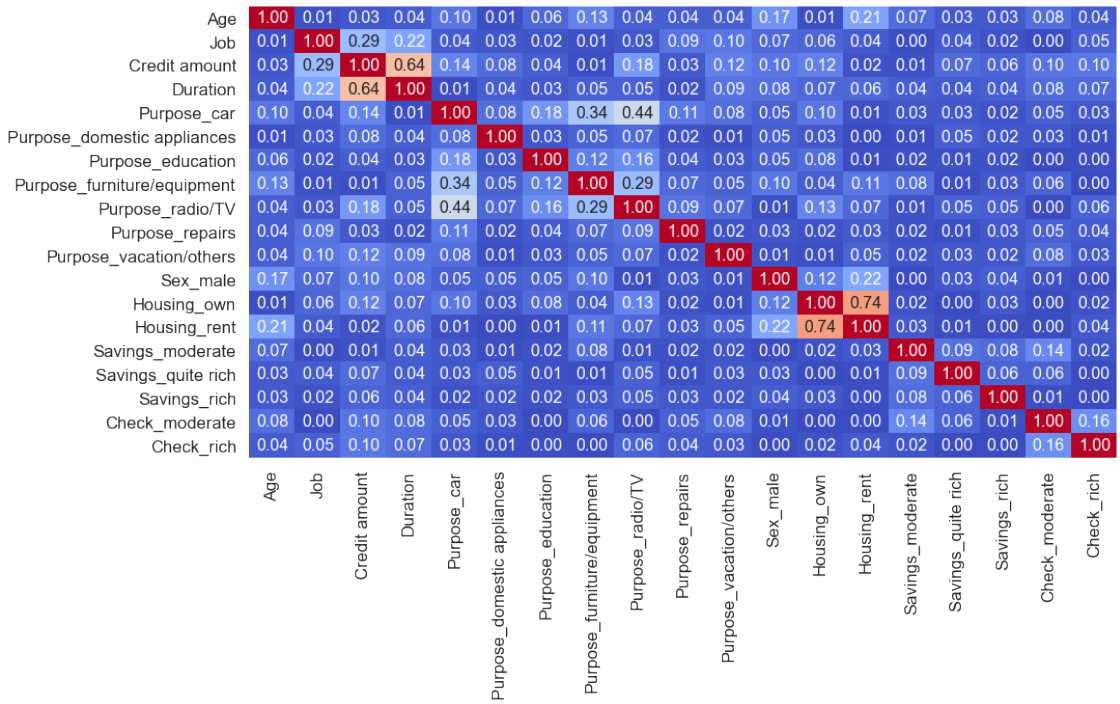
| Sex_male | 1000 | 0.69 | 0.462725 | 0 | 0
| 1 | 1 | 1 |
| Housing_own | 1000 | 0.713 | 0.452588 | 0 | 0
| 1 | 1 | 1 |
| Housing_rent | 1000 | 0.179 | 0.383544 | 0 | 0
| 0 | 0 | 1 |
| Savings_moderate | 1000 | 0.103 | 0.304111 | 0 | 0
| 0 | 0 | 1 |
| Savings_quite rich | 1000 | 0.063 | 0.243085 | 0 | 0
| 0 | 0 | 1 |
| Savings_rich | 1000 | 0.048 | 0.213873 | 0 | 0
| 0 | 0 | 1 |
| Check_moderate | 1000 | 0.269 | 0.443662 | 0 | 0
| 0 | 1 | 1 |
| Check_rich | 1000 | 0.063 | 0.243085 | 0 | 0
| 0 | 0 | 1 |

In the above description table of data columns, for the columns such as Age,Credit amount and Duration the difference between the minimum and maximum values are huge. So we will be doing minmaxsclaer on these column and scales the value between 0 and 1. The description box after the scaling is given below:

| | count | mean | std | min | 25%
| 50% | 75% | max |
|:---|---:|---:|---:|---:|---
:|---:|---:|---:|
| Age | 1000 | 0.361615 | 0.244095 | 0 | 0.175824
| 0.307692 | 0.505495 | 1 |
| Job | 1000 | 1.904 | 0.653614 | 0 | 2
| 2 | 2 | 3 |
| Credit amount | 1000 | 0.367003 | 0.286561 | 0 | 0.146154
| 0.271148 | 0.487692 | 1 |
| Duration | 1000 | 0.429132 | 0.279346 | 0 | 0.210526
| 0.368421 | 0.526316 | 1 |
| Purpose_car | 1000 | 0.337 | 0.472921 | 0 | 0
| 0 | 1 | 1 |
| Purpose_domestic appliances | 1000 | 0.012 | 0.10894 | 0 | 0
| 0 | 0 | 1 |
| Purpose_education | 1000 | 0.059 | 0.235743 | 0 | 0
| 0 | 0 | 1 |
| Purpose_furniture/equipment | 1000 | 0.181 | 0.385211 | 0 | 0
| 0 | 0 | 1 |
| Purpose_radio/TV | 1000 | 0.28 | 0.449224 | 0 | 0
| 0 | 1 | 1 |
| Purpose_repairs | 1000 | 0.022 | 0.146757 | 0 | 0
| 0 | 0 | 1 |
| Purpose_vacation/others | 1000 | 0.012 | 0.10894 | 0 | 0
| 0 | 0 | 1 |
| Sex_male | 1000 | 0.69 | 0.462725 | 0 | 0

1	1	1
Housing_own	1000	0.713	0.452588	0	0
1	1	1
Housing_rent	1000	0.179	0.383544	0	0
0	0	1
Savings_moderate	1000	0.103	0.304111	0	0
0	0	1
Savings_quite rich	1000	0.063	0.243085	0	0
0	0	1
Savings_rich	1000	0.048	0.213873	0	0
0	0	1
Check_moderate	1000	0.269	0.443662	0	0
0	1	1
Check_rich	1000	0.063	0.243085	0	0
0	0	1

After one hot encoding plotted the heatmap correlation between columns and target variable:



Since the Risk column is our target column, I comverted the column to binary values 1 and 0, 1 for good loans and 0 for bad loans. So this conversion will help to classify the risk of loans.

	Risk
0	1
1	0
2	1

3	1
4	0

0.4 Data Modelling

After analysis and cleaning of the data, we will try to predict the risk of transactions utilizing machine learning classification algorithms. So in this section, we will apply some algorithm such as Logistic regression with and without regularization, Decision Tree, Random Forest, Gradient boosting and Stacking and accurately predicts whether a loan is good or bad.

Before applying the model, we will split our dataset to train and test set, with 70 and 30 percentage. We will be using this split train and test set for our entire modeling. Here we used sklearn 'StratifiedShuffleSplit' to maintain the same ratio of predictor classes.

The two tables below show the percentage of two classes in the Risk column, and both are the same with 70% of 1 class which indicates the good class, and 30% of the 0 class which are bad class.

	Risk
1	0.7
0	0.3

	Risk
1	0.7
0	0.3

0.4.1 Logistic Regression

The first model we are trying to predict the class is Logistic Regression. To prevent we will be using regularization in our model. We will try both Ridge(L2) and Lasso(L1) regularization and uses evaluation metrics to compare which regularization provides better accuracy.

So we apply Logistic Regression with regularization, with l1 and l2 regularization. A short glimpse of predicted values and the probability of predicting three methods is given below:

Predicted values:

	lr	l1	l2
0	1	1	1
1	1	1	1
2	1	1	1
3	1	1	1
4	0	0	0

So from the table, we can see that, for the first four rows, all three methods predicted the same value, but for the 5th entry, regression without regularization predicted differently.

Prediction Probabilities:

	lr	l1	l2
0	0.794728	0.784209	0.776756

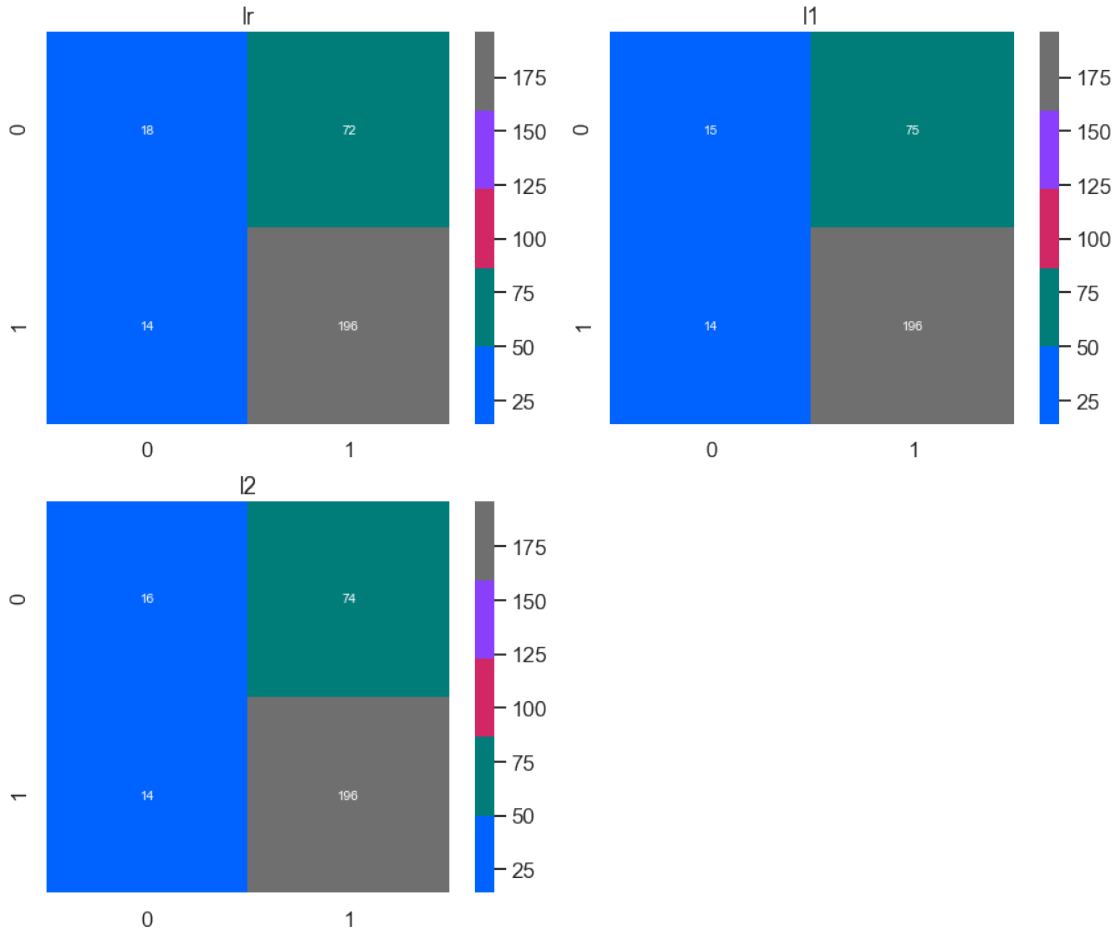
1	0.715493	0.751868	0.720443
2	0.67432	0.723267	0.68053
3	0.738843	0.720196	0.72302
4	0.581843	0.576225	0.59904

The table above with the prediction probability shows a drastic variation in probability. So we will examine the evaluation metrics for these models. Precision, recall, fscore, and AUC have been calculated and the values are given below:

	lr	l1	l2
precision	0.68069	0.661445	0.668148
recall	0.713333	0.703333	0.706667
fscore	0.662583	0.646108	0.651667
accuracy	0.713333	0.703333	0.706667
auc	0.566667	0.55	0.555556

The table above with the metrics clearly shows the difference in the evaluation metrics for different models. the logical regression without any regularization produces better prediction comparatively. But overall the accuracies are not good, so we will try out different classification algorithms.

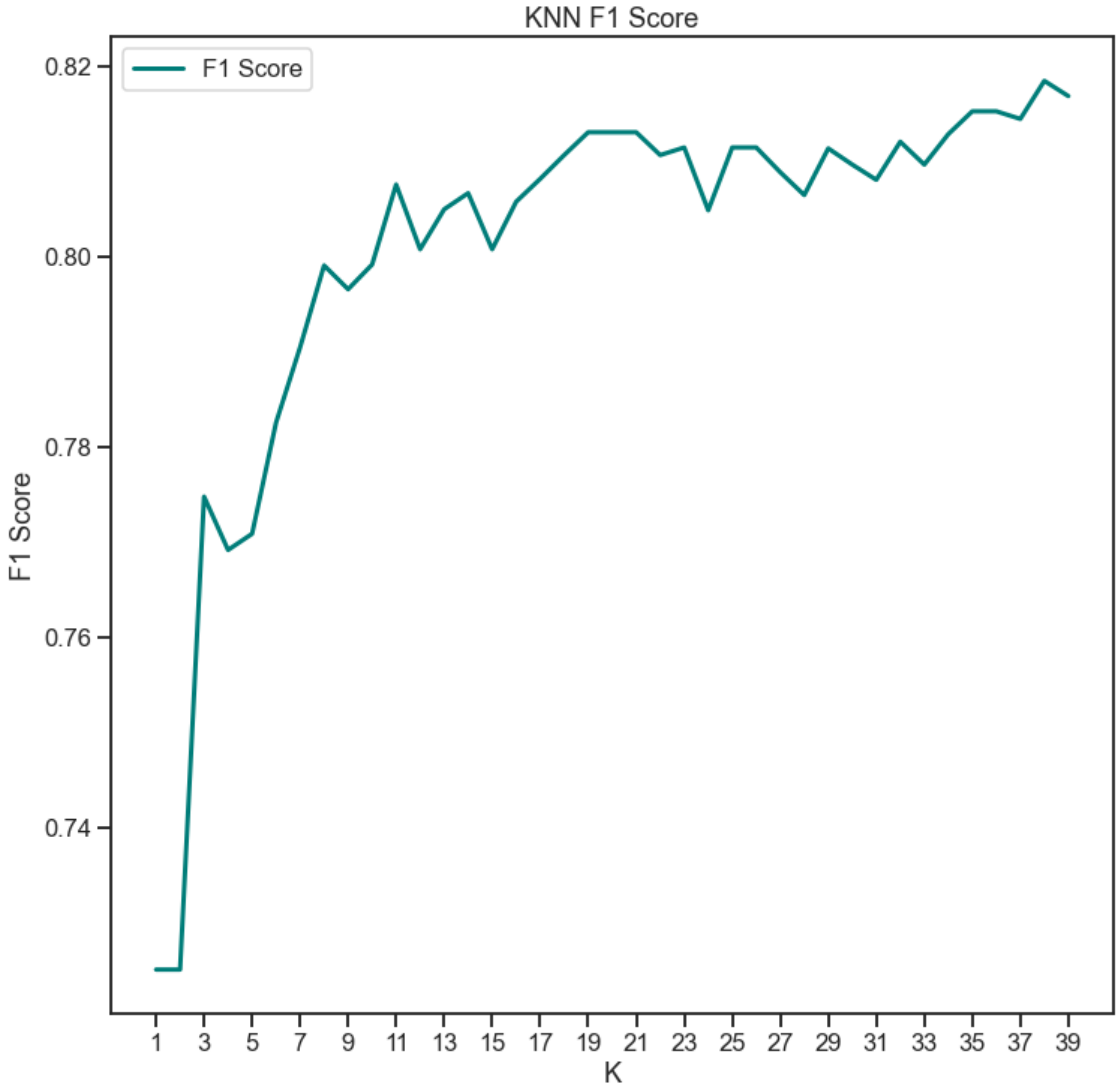
The Confusion matrix of regressions are given below:



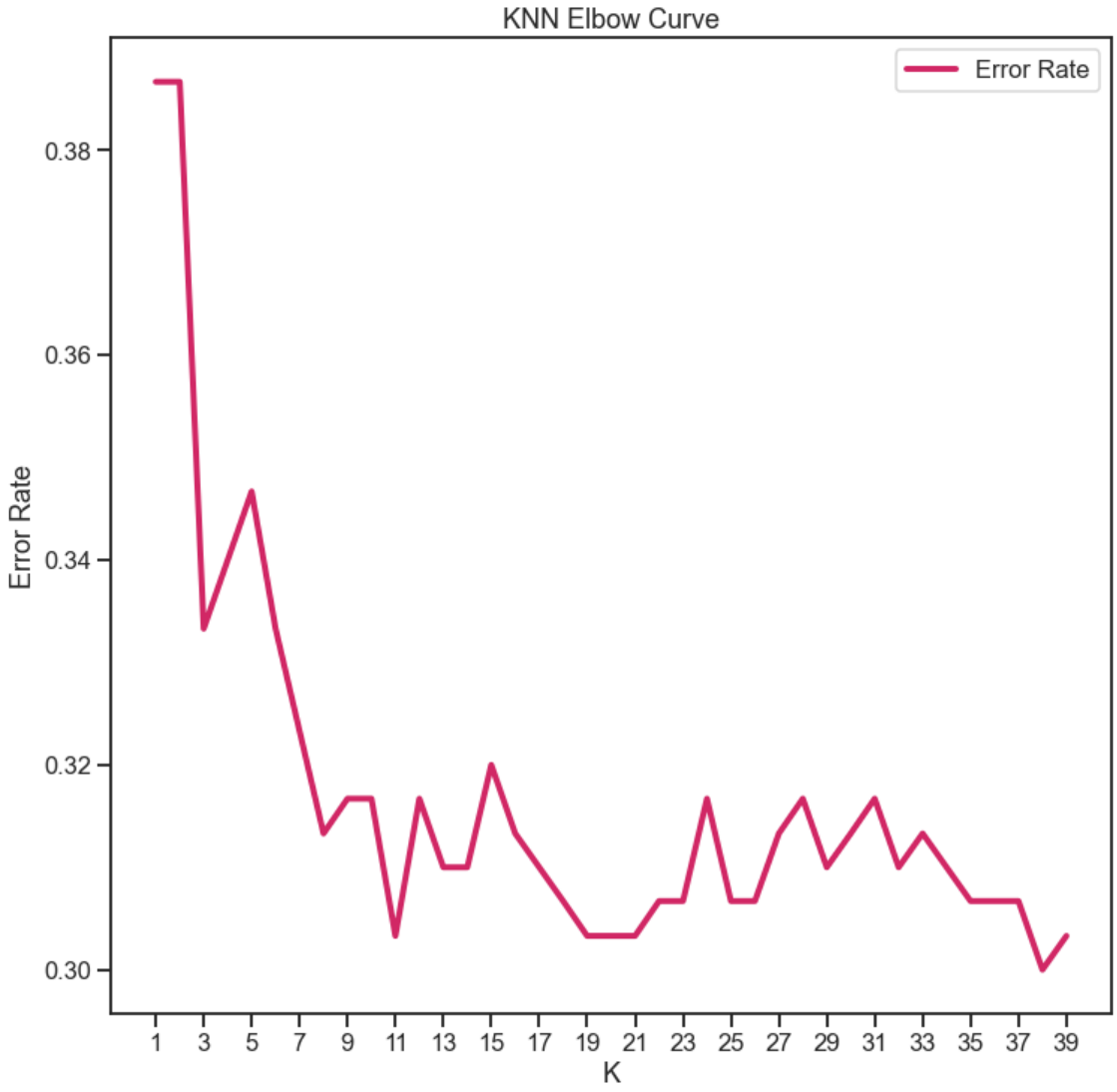
0.4.2 K nearest neighbors

The next algorithm we are going to deploy is the K nearest neighbor. To find out the best k for the prediction, we use cross-validation with K from 1 to 40 and using the Elbow method to determine the best K. We use the same train and test dataset that we used for logistic regression.

```
<Figure size 216x216 with 0 Axes>
```




```
<Figure size 216x216 with 0 Axes>
```



Above are the elbow plots, with Error Rate and F1 score and it shows that K around 32 is the optimum value, because that is the point in which after that there is not much difference in the error rate. So we will apply our KNN algorithm with K= 32 and predicts the loan class.

	precision	recall	f1-score	support
0	0.46	0.07	0.12	90
1	0.71	0.97	0.82	210
accuracy			0.70	300
macro avg	0.58	0.52	0.47	300
weighted avg	0.63	0.70	0.61	300

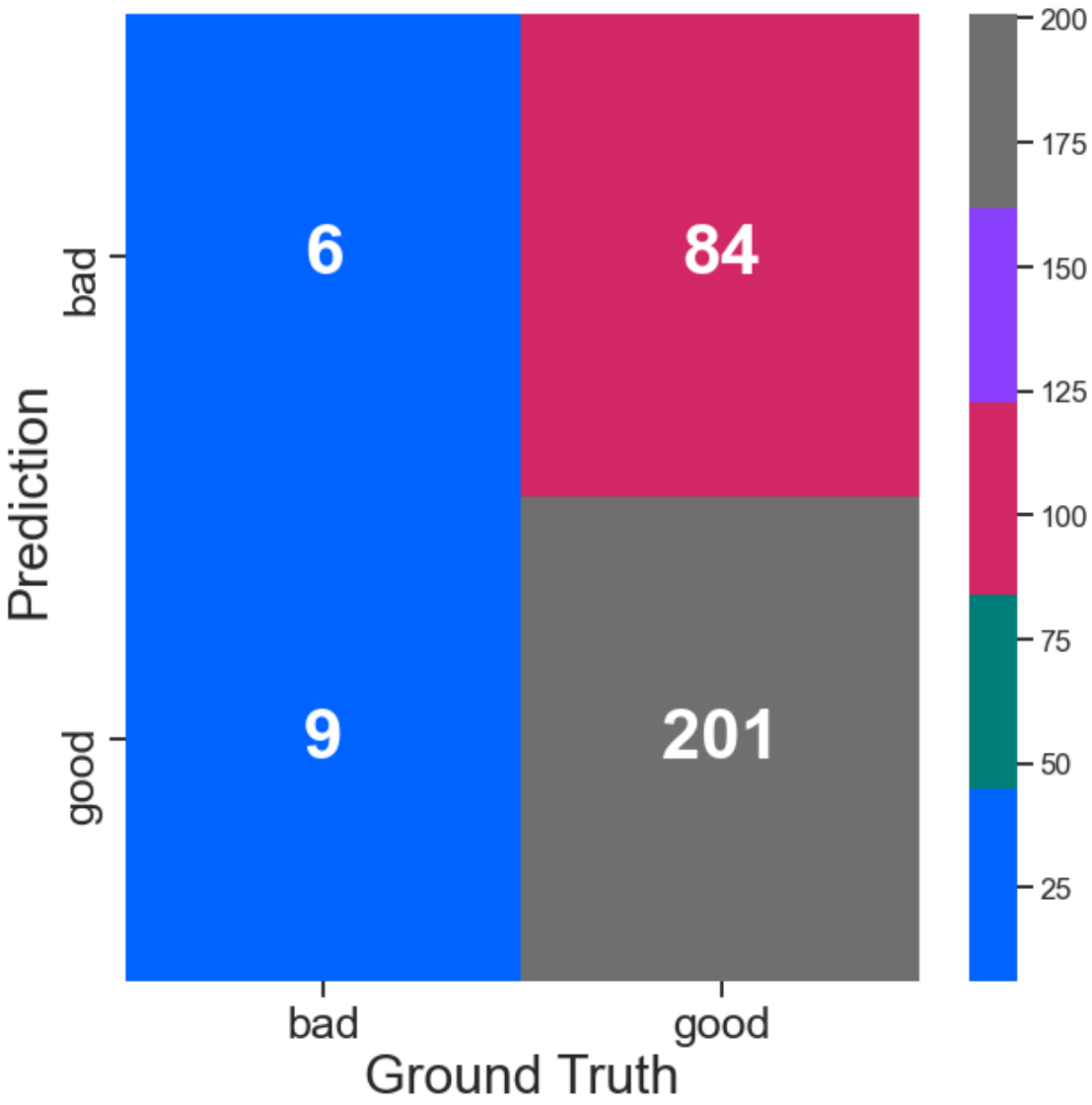
accuracy	precision	recall	f1	auc

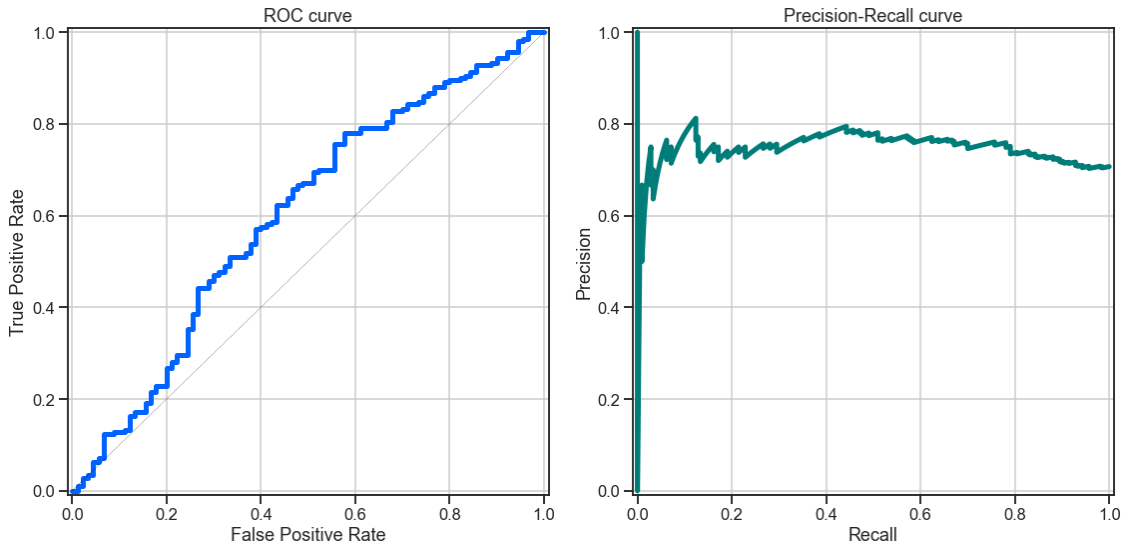
```
0       0.69   0.705263  0.957143  0.812121  0.511905
```

So the KNN algorithm predicted the loan value with 70% accuracy. The bad class with 67% precision and the good class with 72% precision. The recall of predicting bad class is very low, it is beacuse of the small size of the bad class data. Even though the accuracy we attained far better than the logistic regression, the accuracy is still not better.

Below is the confusion matrix of the prediction.

[821]: `Text(0.5, 58.5, 'Ground Truth')`

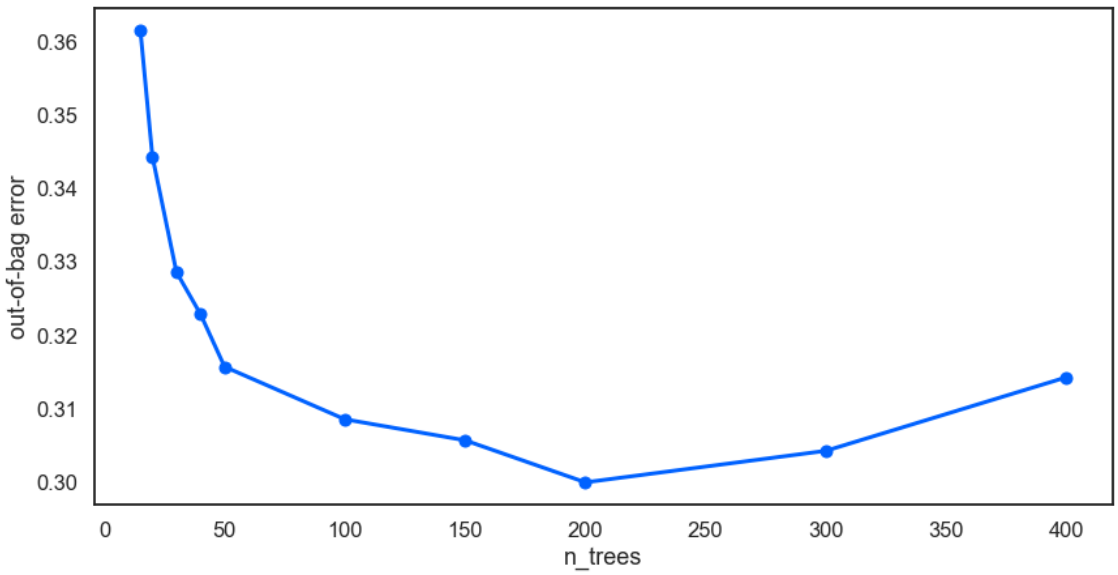




0.4.3 Random Forest

The next algorithm we will be deploying is the Random Forest. Here we try the different number of trees and find out the best parameter with the out-of-bag error. We used tree numbers from 15 to 400 and plotted the out-of-bag error with the tree number.

n_trees	oob
15	0.361429
20	0.344286
30	0.328571
40	0.322857
50	0.315714
100	0.308571
150	0.305714
200	0.3
300	0.304286
400	0.314286



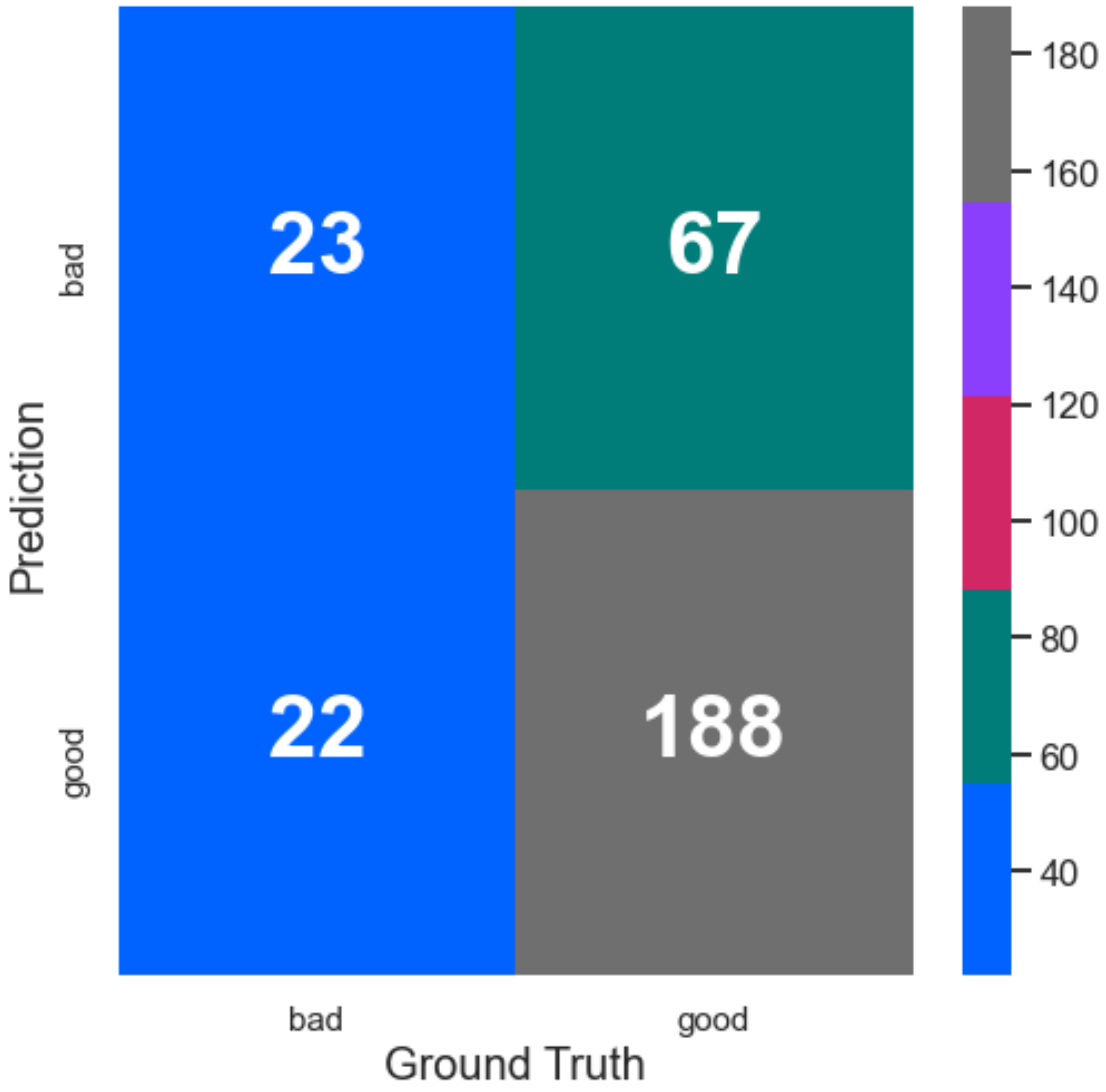
The error looks like to stabilizes around tree number 200. So we will apply a random forest with 200 trees for prediction.

```
              precision    recall  f1-score   support

           0       0.51      0.26      0.34        90
           1       0.74      0.90      0.81       210

    accuracy                           0.70       300
   macro avg       0.62      0.58      0.57       300
weighted avg       0.67      0.70      0.67       300

   accuracy  precision    recall        f1       auc
0  0.703333   0.737255  0.895238  0.808602  0.575397
```

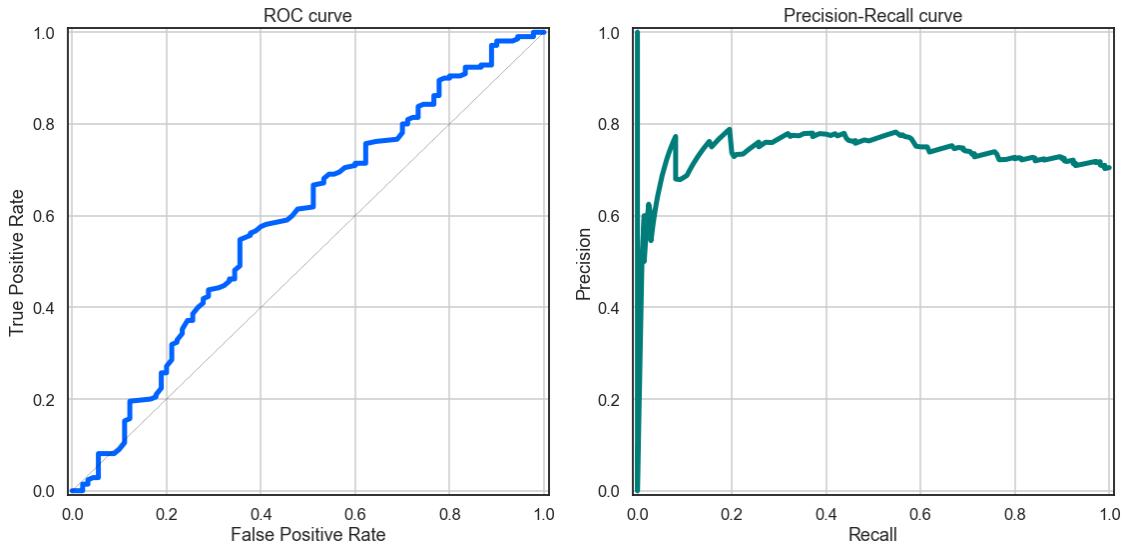
So the evaluation matrix for this model not better than the KNN model. The accuracy is still 70%.

The confusion matrix of the prediction is:

[828]: `Text(0.5, 40.5, 'Ground Truth')`



The ROC-AUC and precision-recall curves are shown below:

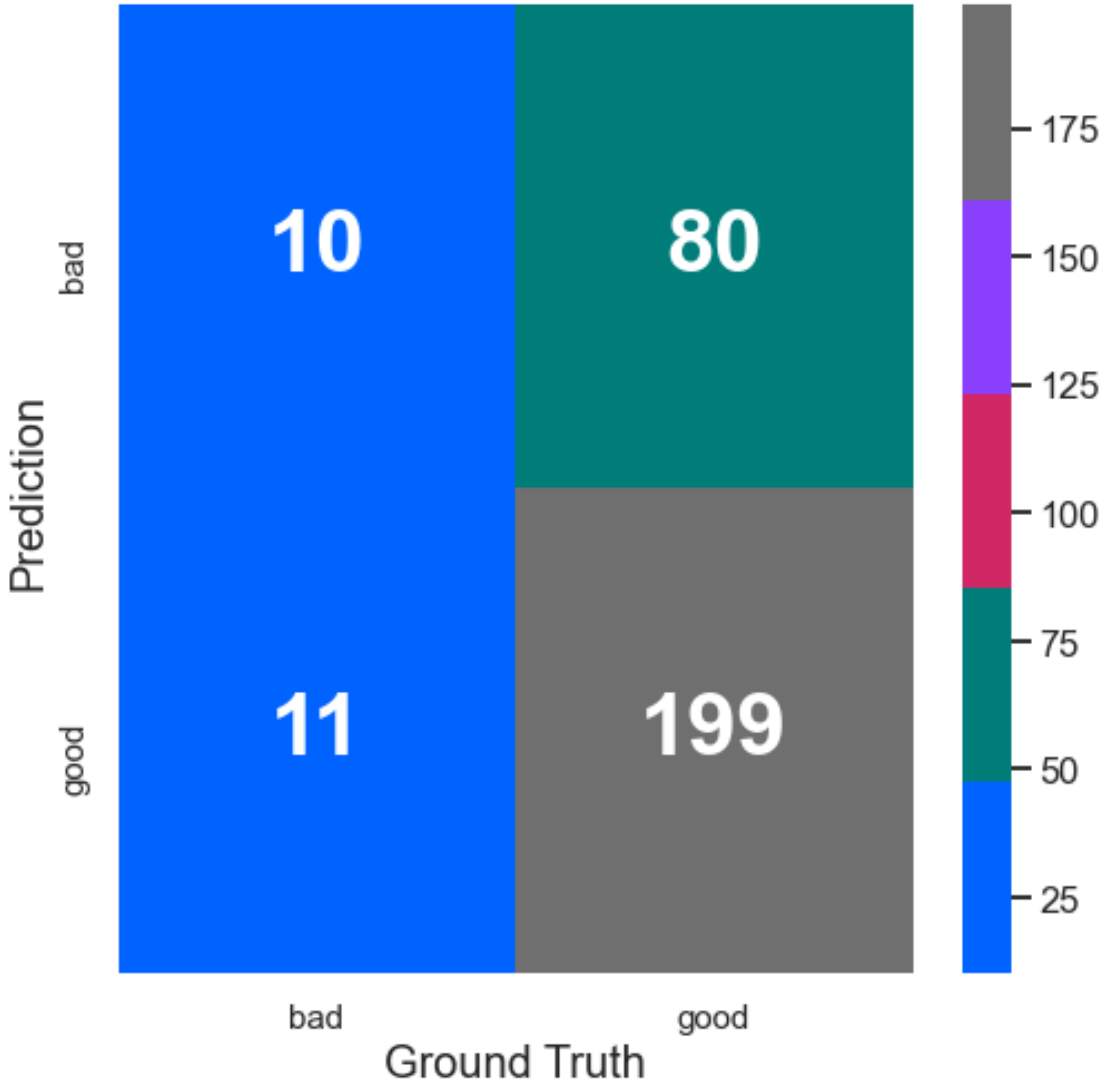


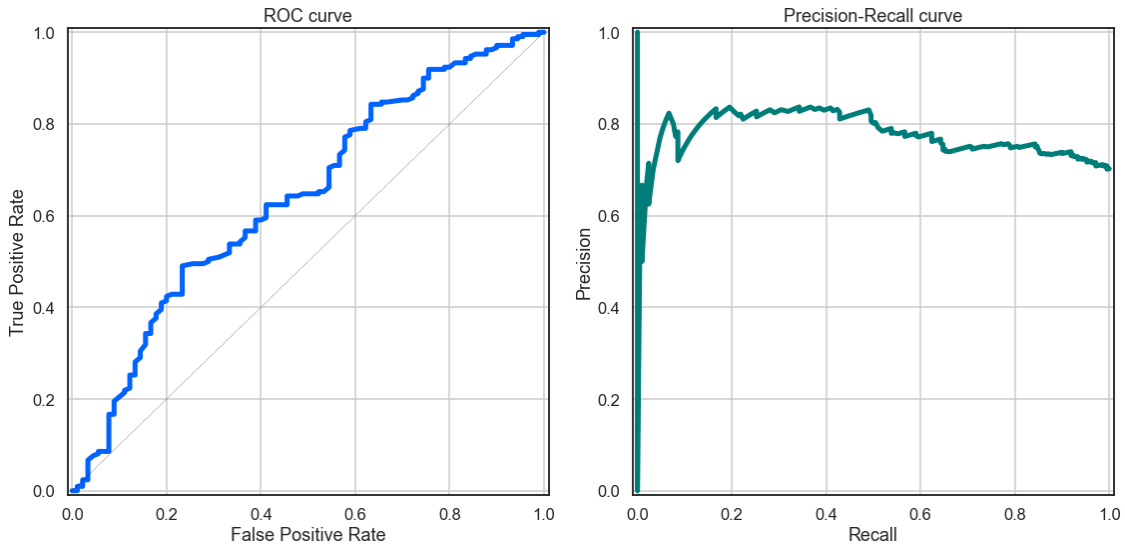
0.4.4 Gradient Boosting

Here we use Gradient Boosting Decision trees for prediction. Instead of using random values for hyperparameter, we use the GridSearchCV cross-validation method to find out the optimum hyperparameters.

[836]:
```
GradientBoostingClassifier(max_features=3, n_estimators=25, random_state=42)
```

```
              precision    recall  f1-score   support

           0       0.11      0.48      0.18        21
           1       0.95      0.71      0.81       279

    accuracy                           0.70       300
   macro avg       0.53      0.59      0.50       300
weighted avg       0.89      0.70      0.77       300

   accuracy  precision    recall        f1       auc
0  0.696667   0.713262  0.947619  0.813906  0.529365
```


10
80
11
199
bad
good
Prediction
bad
good
Ground Truth
175
150
125
100
75
50
25



The accuracy score for prediction with gradient boosting Decision tree is 69%, which is not better than other models.

0.4.5 Staking

In this model, we will try to fit Gradient boosting along with regression with regularization using Voting Classifier. Performance for the voting classifier should improve relative to either logistic regression or gradient boosted trees alone.

	precision	recall	f1-score	support
0	0.57	0.14	0.23	90
1	0.72	0.95	0.82	210
accuracy			0.71	300
macro avg	0.64	0.55	0.53	300
weighted avg	0.67	0.71	0.64	300

	accuracy	precision	recall	f1	auc
0	0.71	0.722022	0.952381	0.821355	0.548413

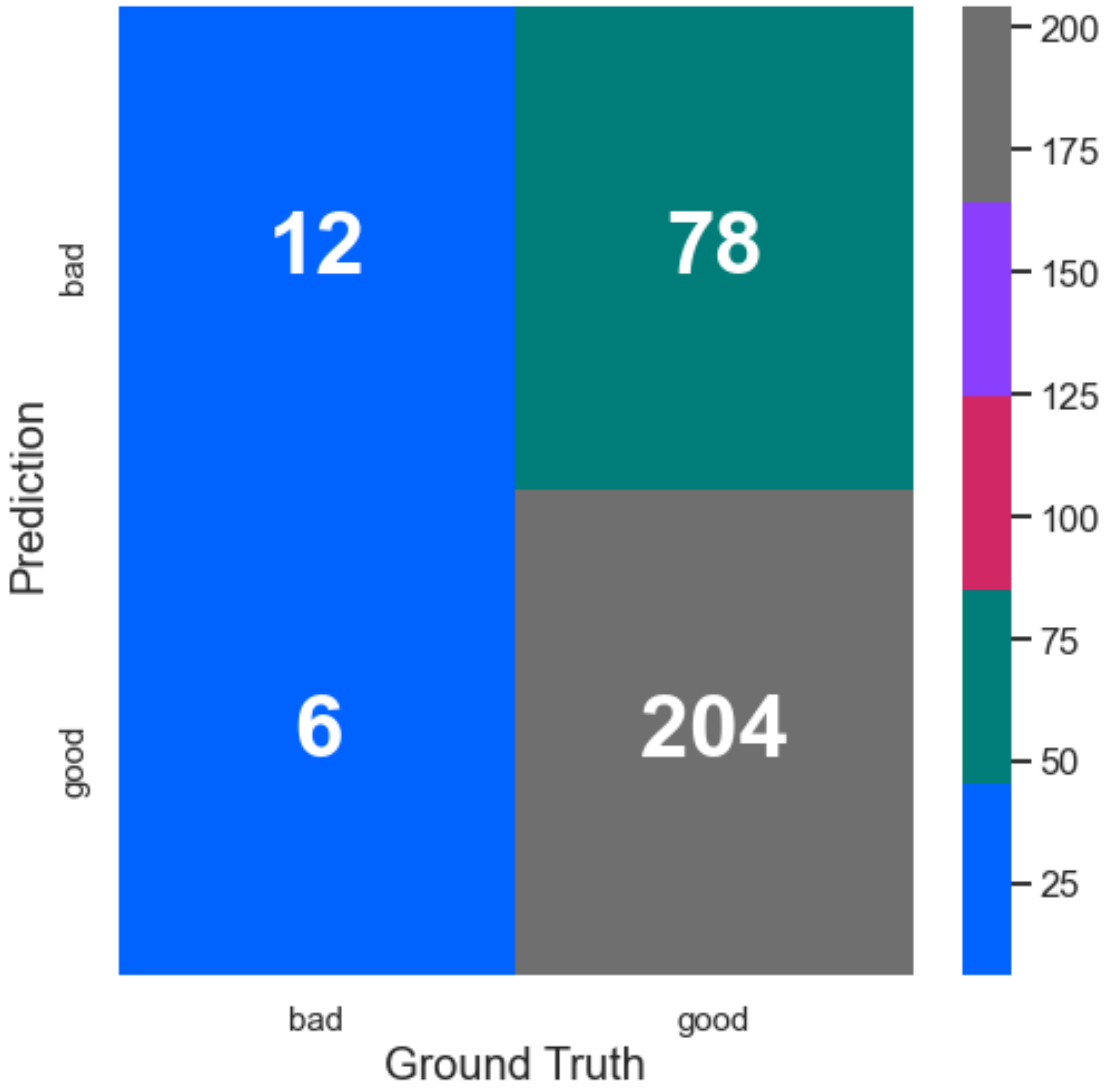
So when using the Voting classifier to stack logistic regression and Gradient Boosting the accuracy score got a little better with 71%. So as a next step I used KNN along with Gradient boosting because those are the two model which produced better accuracy score.

So evaluation matrics for stacking KNN and Gradient boosting is:

	precision	recall	f1-score	support
0	0.67	0.13	0.22	90
1	0.72	0.97	0.83	210

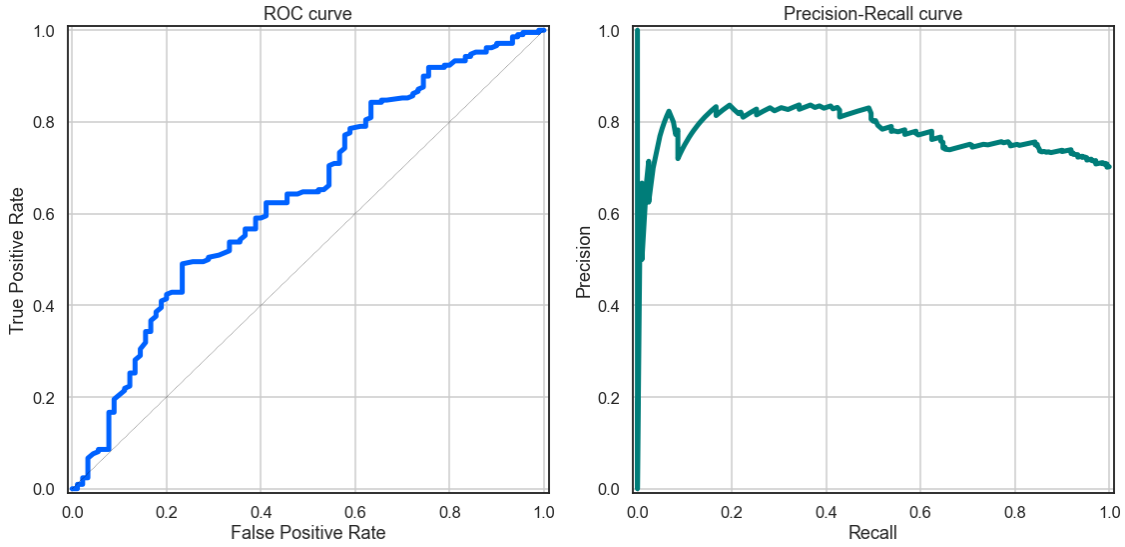
```
    accuracy                           0.72       300
   macro avg       0.70      0.55      0.53       300
weighted avg       0.71      0.72      0.65       300

   accuracy  precision    recall        f1       auc
0      0.72   0.723404  0.971429  0.829268  0.552381
```

As expected the accuracy score got better, increased to 72%, which is the best accuracy score we got so far. The confusion matrix for this model looks like this:



From the Confusion matrix, we can see that it correctly predicted 12 bad classes out of 18, and for a good class, 204 there is 204 correct prediction in a total of 282.

The ROC-AUC curve and Precision-Recall curve is given below. These curves visualize a better prediction compared to other models.



0.5 Result

This project aimed to predict the risk of the loan offered by German Banks. So we applied different model for the prediction, the summarised accuracy score of all the model we have used given below:

[841]:

	Model	Accuracy Score	Precision Score	Recall Score	F1 score	AUC Score
0	Linear Regression	0.716667	0.732342	0.938095	0.822547	0.569048
1	Linear Regression l1	0.706667	0.722628	0.942857	0.818182	0.549206
2	Linear Regression l2	0.710000	0.725275	0.942857	0.819876	0.554762
3	KNN	0.716667	0.719298	0.976190	0.828283	0.543651
4	Random Forest	0.696667	0.733333	0.890476	0.804301	0.567460
5	Gradient Boosting	0.696667	0.713262	0.947619	0.813906	0.529365
6	Stacking LogR & Boosting	0.710000	0.722022	0.952381	0.821355	0.548413
7	Stacking KNN & Boosting	0.720000	0.723404	0.971429	0.829268	0.552381

From the analysis of the above table, we can understand that the model with K nearest Neighbour and Gradient boosting, logical regression with l2 regularization, and surprisingly regression without any regularization performed well, compared to others. The precision score and Area under curve have the highest values in the Regression without regularization model, but Accuracy, Recall score

and F1 score is best with the stacking method.

Still, all these models couldn't predict the Risk factor of the Bank loan, with an accuracy above 75%. In a real-world case, an accurate prediction of at least 80% will be useful. This reduction of accuracy is coming from the imbalance of the data. For all the model, the recall score of class 0, that is a bad loan is very low compared to class 1. That is because of the small number of 0 class presence in the data.

0.6 Summary and Future Insights

The main goal of this project was to predict the Risk factor of loans approved by the German Banks. For this, we had historical data of loans provided by the banks, and we cleaned and preprocessed data to make it useful for our prediction. Then we deployed, the classification algorithm such as Logical Regression with(l1 & l2) and without Regression, K nearest Neighbour method, Random Forest, Gradient Boosting, and finally stacking of Gradient Boosting with Logical regression and KNN. From the analysis of evaluation matrics of these models, we found that logical regression and Stacking of KNN with Gradient Boosting performed well, with around 71% accuracy.

I believe that this model couldn't predict the class of bank loans with 80% or more accuracy is because of the imbalance in the data. The percentage of class 0 in the data is 30%. So as the next step to this project we could try to balance the data with a different technique like Synthetic Minority Oversampling Technique (SMOTE) and Adaptive Synthetic sampling (ADASYN). Another step we can include here to improve our accuracy score by adding additional features useful for prediction. These additional features will be the time taken to return the loan, Bank Balance of the customer. Then a model could predict these classes with more accuracy.