LendingClub Loan repayment prediction using Decision Tree Classification and Random Forest Classification algorithms

Mini Project

Submitted by:

Swathi Mithanthaya – 4NM18CS200

Ritvik Vasanth Kumar – 4NM18CS135

for the course Artificial Intelligence

Project Guide

Dr. Aravinda C V

Associate Professor

Department of Computer Science and Engineering

in partial fulfillment of the requirements for the award of the Degree

Of Bachelor of Engineering in Computer Science and Engineering from

Visvesvaraya Technological University, Belgaum

NITTE EDUCATION TRUST

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC

☎: 08258 - 281039 - 281263, Fax: 08258 - 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

Dec 2020

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution affiliated to VTU, Belagavi)
(NBA Accredited, ISO 9001:2008 Certified)
Nitte – 574110, Karkala, Udupi District, Karnataka, India

Department of Computer Science and Engineering

CERTIFICATE

Certified that the mini project entitled

LendingClub Loan repayment prediction using Decision Tree Classification and Random Forest Classification algorithms

Is a bonafide work carried out by

Swathi Mithanthaya 4NM18CS200

Ritvik Vasanth Kumar 4NM18CS135

In partial fulfillment of requirements for the award of Bachelor Of Engineering Degree in computer science and engineering prescribed by Visvesvaraya Technological University, Belgaum during the year 2020-21.

It is certified that all corrections/suggestions indicated for internal assessment has been incorporated in the report.

The mini project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Name & Signature of Guide	**Name & Signature of HOD**
Dr. Aravinda C V	Dr. K R Uday Kumar Reddy
Associate Professor	Head of the Department
Department of CSE	Department of CSE

ACKNOWLEDGEMENT

It gives us immense pride and contentment on successful completion of the project. The success is incomplete without thanking the few people who are the foundation to our project.

Hence, we take this opportunity to thank our project guide, Dr Aravinda C V, Dept. of CSE for his continuous support and guidance.

Our sincere thanks to Dr K. R. Udaya Kumar Reddy, HOD-Dept. of Computer Science and Engineering, NMAMIT, Nitte for his generous support. We also acknowledge and express our sincere thanks to our beloved Dr Niranjan. N. Chiplunkar, Principal, NMAMIT, Nitte who is a source of inspiration to us.

We thank all the Teaching and Non-Teaching staff members of the department of CSE for providing resources for the completion of the project. A special thanks goes to our parents and friends for supporting and encouraging us in all ways thus making our project successful.

Finally, we thank all those who have contributed directly or indirectly in making this project a grand success.

Swathi Mithanthaya 4NM18CS200 Ritvik Vasanth Kumar 4NM18CS135

ABSTRACT

This Machine Learning code is developed to describe the data of the loan borrowers from the LendingClub and predict whether a borrower can repay the loan or not based on various criteria. This code will cover two machine learning models, decision tree classification algorithm and random forest classification algorithm. Both algorithms predict whether a person will repay loan or not. However, the accuracies are varied and have to be compared and has to be concluded about the best of the two algorithms.

Through this python code we will describe various attributes of the borrowers' datasets and understand the main features required to check the factors which affects the repayment of the loan. All attributes are taken into consideration like gender, married or not, education, working in a company or self-employed, applicant income, loan amount.

For implementation we have used machine learning algorithms and is coded using Python3. We will be classifying the datasets and produce a classification report for the same to verify the borrowers who can repay the loan and who cannot.

TABLE OF CONTENTS

Contents Page

LITERATURE SURVEY

Decision Tree algorithm

1. Information gain is an important aspect. It is defined by the formula

$$Gain(S,A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

where Entropy is:

$$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

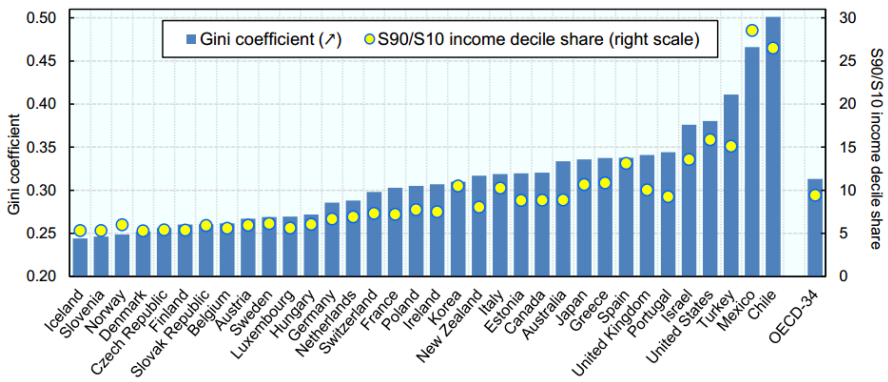
Note that *0*ln (0) is taken to be zero by convention.*

2. Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labelled if it was randomly labelled according to the distribution of labels in the subset.

$$G.I\ (Y) = 1 - \sum_{i=1}^{k} [p(Y)]^2$$

Random forest

This is the famous and most accepted case study done using random forest and Gini. Mexico has a population of 118 MM. Say, the algorithm Random forest picks up 10k observation with only one variable (for simplicity) to build each CART model. In total, we are looking at 5 CART model being built with different variables.



INTRODUCTION

Overview of Lending Club and loan datasets:

LendingClub is an American peer-to-peer lending company. It was the first peer-to-peer lender to register its offerings as securities with the Securities and Exchange Commission (SEC), and to offer loan trading on a secondary market. At its height, LendingClub was the world's largest peer-to-peer lending platform. The LendingClub dataset consists of attributes such as credit policy no, purpose of loan, interest rate, instalments, annual income, DTI, FICO etc.

Uses and Advantages:

LendingClub provides loans with less paperwork and less guarantors and less deposits due to its motto of networking with all banks and customer service. Hence, it requires a methodology to find out whether a person can repay the loan or not. This machine learning code based on the previous data can predict this so that they can gain more profits. Hence adds on to the advantages of LendingClub with this machine learning algorithm. To add on, other advantages of this code is that it takes 60:40 train to test datasets which creates balance in training and testing and hence improves accuracy. It is efficient and easy to predict as every data is described which enables deep analysis of the data by the LendingClub.

Methodology:

We have divided the project in different parts, namely, comparing the data, describing the data, analysis, model creation, splitting of data for training and testing. Furthermore, to run the same data onto the models of different algorithms to compare the accuracy.

Models:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

Random Forest is a robust machine learning algorithm that can be used for a variety of tasks including regression and classification. It is an ensemble method, meaning that a random forest model is made up of a large number of small decision trees, called estimators, which each produce their own predictions. The random forest model combines the predictions of the estimators to produce a more accurate prediction.

SOFTWARE SPECIFICATION

The implementation of the project requires Python3 installed in the system. Machine learning coding is done using Python3.

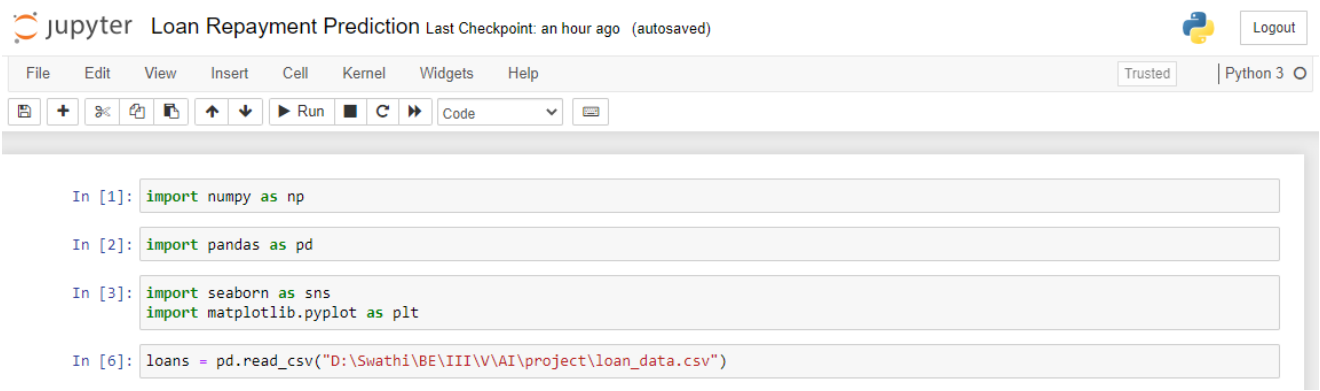
Anaconda navigator is required for using the python libraries. The other option is to install every library required separately in the command line. The libraries are pre-installed in anaconda prompt.

The dataset is downloaded from the LendingClub website for using it to train the model. The dataset should be in csv or excel format. Otherwise, some attributes of data such as annual income must be converted to numpy array and stored for easier and accurate analysis.

The fico values must be encoded to see the relationship of fico with other values in various graphs for analysis.

IMPLEMENTATION

Importing Python Libraries and the dataset



The libraries required are: numpy, pandas, seaborn and matplotlib. *NumPy* contains a multi-dimensional array and matrix data structures. It can be utilised to perform a number of mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.

Pandas objects rely heavily on NumPy objects. *Pandas* is the most popular python library that is used for data analysis. It provides highly optimized performance with back-end source code is purely written in *C* or *Python*. We can analyse series and data frames using pandas.

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. The dataset is in csv format and is read using the function read_csv imported from pandas' library.

Pre-processing

jupyter Loan Repayment Prediction Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

```
In [7]: loans.head()
```

Out[7]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2yrs	pub.rec	nc
0	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	52.1	0	0	0	
1	1	credit_card	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	76.7	0	0	0	
2	1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	4710.000000	3511	25.6	1	0	0	
3	1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	2699.958333	33667	73.2	1	0	0	
4	1	credit_card	0.1426	102.92	11.299732	14.97	667	4066.000000	4740	39.5	0	1	0	

```
In [8]: loans.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9578 entries, 0 to 9577
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   credit.policy      9578 non-null   int64
 1   purpose            9578 non-null   object
 2   int.rate           9578 non-null   float64
 3   installment        9578 non-null   float64
 4   log.annual.inc     9578 non-null   float64
 5   dti                9578 non-null   float64
 6   fico               9578 non-null   int64
 7   days.with.cr.line  9578 non-null   float64
 8   revol.bal          9578 non-null   int64
 9   revol.util         9578 non-null   float64
 10  inq.last.6mths     9578 non-null   int64
 11  delinq.2yrs        9578 non-null   int64
 12  pub.rec            9578 non-null   int64
 13  not.fully.paid     9578 non-null   int64
dtypes: float64(6), int64(7), object(1)
memory usage: 1.0+ MB
```

```
In [9]: loans.describe()
```

```
In [9]: loans.describe()
```

Out[9]:

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2y
count	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9.578000e+03	9578.000000	9578.000000	9578.0000
mean	0.804970	0.122640	319.089413	10.932117	12.606679	710.846314	4560.767197	1.691396e+04	46.799236	1.577469	0.1637
std	0.396245	0.026847	207.071301	0.614813	6.883970	37.970537	2496.930377	3.375619e+04	29.014417	2.200245	0.5462
min	0.000000	0.060000	15.670000	7.547502	0.000000	612.000000	178.958333	0.000000e+00	0.000000	0.000000	0.0000
25%	1.000000	0.103900	163.770000	10.558414	7.212500	682.000000	2820.000000	3.187000e+03	22.600000	0.000000	0.0000
50%	1.000000	0.122100	268.950000	10.928884	12.665000	707.000000	4139.958333	8.596000e+03	46.300000	1.000000	0.0000
75%	1.000000	0.140700	432.762500	11.291293	17.950000	737.000000	5730.000000	1.824950e+04	70.900000	2.000000	0.0000
max	1.000000	0.216400	940.140000	14.528354	29.960000	827.000000	17639.958330	1.207359e+06	119.000000	33.000000	13.0000

```
In [24]: loans['purpose'].value_counts()
```

```
Out[24]: debt_consolidation    3957
         all_other             2331
         credit_card           1262
         home_improvement       629
         small_business         619
         major_purchase         437
         educational            343
         Name: purpose, dtype: int64
```

```
In [25]: purpose_categories = pd.get_dummies(loans['purpose'],drop_first=True)

In [26]: loans_final = pd.concat([loans,purpose_categories],axis=1)

In [27]: loans_final.drop(['purpose'],axis=1,inplace=True)

In [28]: from sklearn.model_selection import train_test_split
```

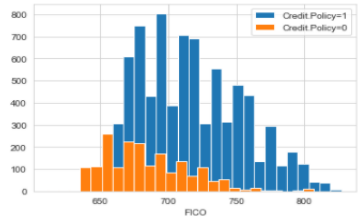
This is done to check for missing values, null columns or rows or unnecessary data etc. in the datasets. The data is not clean so pre-processing must be done. The possible ways are by entering the mean value in the null cells or deleting the column or row which has very few data and rest are empty or normalisation method.

Analysis

Here, comparing fico values and credit policy of people shows that as fico value increasing credit policy in decreasing with few exceptions for some fico values are noted. This histogram function is imported from seaborn library.

```
In [10]: sns.set_style('whitegrid')
         plt.hist(loans['fico'].loc[loans['credit.policy']==1], bins=25, label='Credit.Policy=1')
         plt.hist(loans['fico'].loc[loans['credit.policy']==0], bins=25, label='Credit.Policy=0')
         plt.legend()
         plt.xlabel('FICO')

Out[10]: Text(0.5, 0, 'FICO')
```



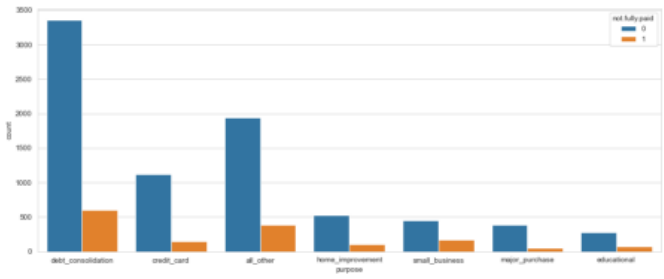
Using matplotlib library we can plot histogram for borrowers who have not fully paid their loans according to fico values. Lenders use borrowers' **FICO scores** along with other details on borrowers' credit reports to assess credit risk and determine whether to extend credit.

```
In [49]: plt.figure(figsize=(10,6))
         loans[loans['not.fully.paid']==1]['fico'].hist(bins=30, alpha=0.5, color='black', label='not.fully.paid=1')
         loans[loans['not.fully.paid']==0]['fico'].hist(bins=30, alpha=0.5, color='red', label='not.fully.paid=0')
         plt.legend()
         plt.xlabel('FICO')

Out[49]: Text(0.5, 0, 'FICO')
```

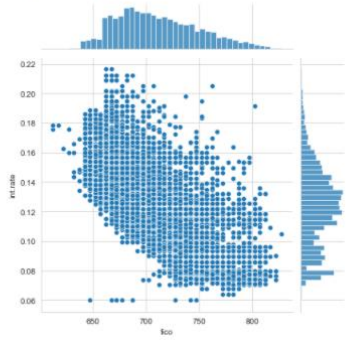

```
In [12]: plt.figure(figsize=(15,6))
         sns.countplot(data=loans, x='purpose', hue='not.fully.paid')

Out[12]: <AxesSubplot:xlabel='purpose', ylabel='count'>
```



The countplot shows the number of people in each category of purpose of borrowing loans. People who have given purpose as debt_consolidation are the highest number of people who are unable to repay the loan.

```
In [13]: plt.figure(figsize=(15,20))
         sns.jointplot(x='fico', y='int.rate', data=loans)
Out[13]: <seaborn.axisgrid.JointGrid at 0x18f37924430>
         <Figure size 1080x1440 with 0 Axes>
```




```
In [14]: sns.lmplot(data=loans, x='fico', y='int.rate', hue='credit.policy', col='not.fully.paid', palette='Accent_r')
Out[14]: <seaborn.axisgrid.FacetGrid at 0x18f37a76820>
```

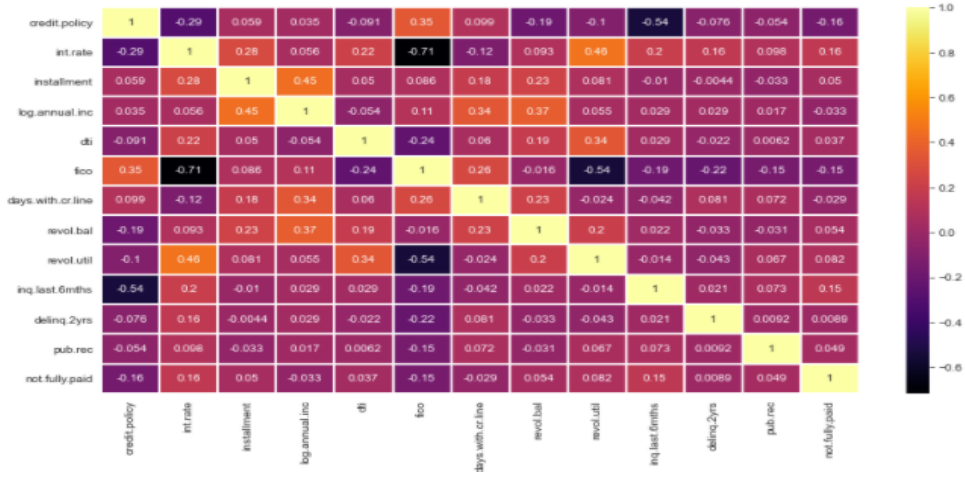

```
In [15]: purpose_c = pd.get_dummies(loans['purpose'], drop_first=True)
         loans_f = pd.concat([loans, purpose_c], axis=1).drop('purpose', axis=1)
         loans_f.head()
Out[15]:
```

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2yrs	pub.rec	not.fully.paid	credit_
0	1	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	52.1	0	0	0	0	
1	1	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	76.7	0	0	0	0	
2	1	0.1357	366.86	10.373491	11.63	682	4710.000000	3511	25.6	1	0	0	0	
3	1	0.1008	162.34	11.350407	8.10	712	2699.958333	33667	73.2	1	0	0	0	
4	1	0.1426	102.92	11.299732	14.97	667	4066.000000	4740	39.5	0	1	0	0	

The joint plot shows the relation between the interest rate and fico values. As the fico values increases the interest rate increased up to certain interest rate and later has decreased.

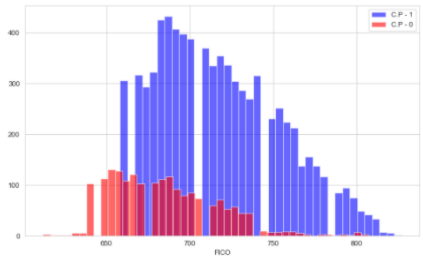
Best fit has been done among all the plotted data points in lm-plot. It shows that as overall interest rate has decreased as fico values have increased. The slope of the graph with data point of borrowers who have 1 credit policy is greater than with credit policy 0.

```
In [18]: plt.figure(figsize=(14,7))
         sns.heatmap(loans.corr(),annot=True,cmap='inferno',linewidths=1)
Out[18]: <AxesSubplot:>
```

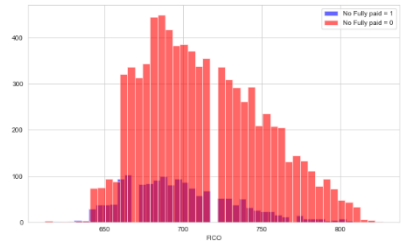


A **heat map** is a data visualization technique that shows magnitude of a phenomenon as colour in two dimensions. The variation in colour may be by hue or intensity, giving obvious visual cues to the reader about how the phenomenon is clustered or varies over space. This shows the overall relationship of all the attributes with each other. High positive relation or high negative values can be ignored as it might cause anomaly in the further analysis.

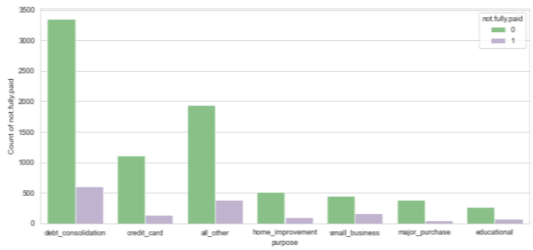
```
In [19]: plt.figure(figsize=(10,6))
         loans[loans['credit.policy']==1]['fico'].hist(bins=45,color='blue',label='C.P - 1',alpha=0.6)
         loans[loans['credit.policy']==0]['fico'].hist(bins=45,color='red',label='C.P - 0',alpha=0.6)
         plt.xlabel("FICO")
         plt.legend()
Out[19]: <matplotlib.legend.Legend at 0x18f38f8b4f0>
```



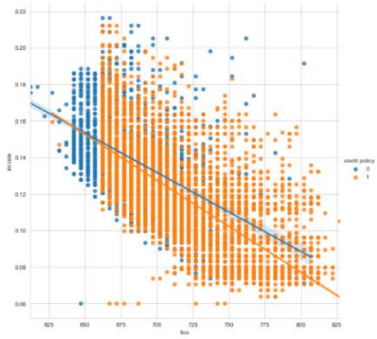

```
In [21]: plt.figure(figsize=(10,6))
         loans[loans['not.fully.paid']==1]['fico'].hist(bins=45,color='blue',label='No Fully paid = 1',alpha=0.6)
         loans[loans['not.fully.paid']==0]['fico'].hist(bins=45,color='red',label='No Fully paid = 0',alpha=0.6)
         plt.xlabel("FICO")
         plt.legend()
Out[21]: <matplotlib.legend.Legend at 0x18f394b1670>
```




```
In [22]: plt.figure(figsize=(12,5))
         sns.countplot(x='purpose',hue="not.fully.paid",data=loans,palette='Accent')
         plt.ylabel("Count of not.fully.paid")
Out[22]: Text(0, 0.5, 'Count of not.fully.paid')
```




```
In [23]: sns.lmplot(x='fico',y='int.rate',data=loans,hue='credit.policy',height=8)
Out[23]: <seaborn.axisgrid.FacetGrid at 0x18f393d7940>
```



First histogram shows relation between fico and credit policy. Fico values are high for values of credit policy=1 and less for credit policy=0. In second histogram, the fico values are high for borrowers who did not fully pay their debt. The countplot shows the number of people in each category of purpose of borrowing loans. People who have given purpose as debt_consolidation are the highest number of people who are unable to repay the loan. In the last lm-plot shows that, people who pay full instalments have low interest rates over a period of time and vice versa.

Model development – Decision Tree Classification Algorithm

```
In [28]: from sklearn.model_selection import train_test_split

In [29]: X = loans_final.drop('not.fully.paid',axis=1)
         y = loans_final['not.fully.paid']

         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.40, random_state=10)

         print(X_test)

      credit.policy  int.rate  installment  log.annual.inc    dti  fico  \
1460              1    0.1020       485.42       10.315597  12.87   752
3130              1    0.1284       605.13       11.205041   7.69   717
8717              0    0.0800       141.02       10.166006   1.43   767
3776              1    0.1347       203.54       10.341742  20.25   702
2542              1    0.1347       232.37       11.512925  11.62   687
...             ...       ...          ...             ...    ...   ...
6390              1    0.1287       168.17       11.407565  22.43   677
7369              1    0.1099       589.24       11.314133  22.82   717
9008              0    0.2011        65.14        9.903488   2.40   667
5146              1    0.0859       158.06       10.596635   7.62   802
5852              1    0.1322       507.01       11.775290   8.36   697

      days.with.cr.line  revol.bal  revol.util  inq.last.6mths  delinq.2yrs  \
1460        5789.958333      14857        31.3               2            1
3130        3217.000000       4724        78.7               0            0
8717         929.958333       1057        23.5               0            0
3776        1199.958333       2314        32.1               1            0
2542        3120.000000      10867        51.1               0            0
...                 ...        ...         ...             ...          ...
6390        4140.041667      44084        65.1               1            0
7369        3750.041667      18940        72.2               0            1
```

```
In [30]: from sklearn.tree import DecisionTreeClassifier

In [31]: dtc = DecisionTreeClassifier()

In [32]: dtc.fit(X_train,y_train)

Out[32]: DecisionTreeClassifier()

In [33]: predictions = dtc.predict(X_test)
         print(predictions)

         [0 0 0 ... 0 0 1]

In [35]: from sklearn.metrics import classification_report,confusion_matrix

In [36]: print(y_test)
         print("\t\t\t\tCLASSIFICATION REPORT:\n")
         print(classification_report(y_test,predictions))

         1460    0
         3130    0
         8717    0
         3776    0
         2542    0
                ..
         6390    0
         7369    0
         9008    0
         5146    0
         5852    0
```

```
                    CLASSIFICATION REPORT:

              precision    recall  f1-score   support

           0       0.86      0.83      0.84      3241
           1       0.20      0.24      0.22       591

    accuracy                           0.74      3832
   macro avg       0.53      0.53      0.53      3832
weighted avg       0.75      0.74      0.74      3832
```

```
In [37]: print("CONFUSION MATRIX:\n")
         print(confusion_matrix(y_test,predictions))

         CONFUSION MATRIX:

         [[2676  565]
          [ 450  141]]
```

The dataset is split into training and testing data in the ratio of 60:40. The training dataset is passed to the decision tree model for training and later the test data is passed to predict the values. The values appear as numpy array which are hot-encoded with 0 and 1. 0 means loan will be repaid and 1 means loan will not be repaid.

In the classification report, the output and predictions for some values are displayed. The accuracy of the model is 74%. Precision, f1-score and recall are calculated with help of confusion matrix. Confusion matrix shows 4 cells in matrix format, namely, true positive, true negative, false negative and false positive. True positive and false positive means that actual and predicted values are same. True negative and false positive means wrong predictions. The number of such data are represented. The different ratios of the values of confusion matrix are shown in classification report with precision is ratio of predicted to actual values, recall is ratio of predicted to false actual, etc.

Model development – Random Forest Classification Algorithm

```
In [40]: from sklearn.ensemble import RandomForestClassifier

In [41]: rfc = RandomForestClassifier()

In [42]: rfc.fit(X_train,y_train)

Out[42]: RandomForestClassifier()

In [43]: predictions2 = rfc.predict(X_test)
         print(predictions2)

         [0 0 0 ... 0 0 0]

In [44]: print(y_test)
         print("\t\t\tCLASSIFICATION REPORT:\n")
         print(classification_report(y_test,predictions2))

         1460    0
         3130    0
         8717    0
         3776    0
         2542    0
                ..
         6390    0
         7369    0
         9008    0
         5146    0
         5852    0
         Name: not.fully.paid, Length: 3832, dtype: int64
                                 CLASSIFICATION REPORT:
```

```
         7369    0
         9008    0
         5146    0
         5852    0
         Name: not.fully.paid, Length: 3832, dtype: int64
                                 CLASSIFICATION REPORT:

                       precision    recall  f1-score   support

                    0       0.85      1.00      0.92      3241
                    1       0.50      0.02      0.04       591

             accuracy                           0.85      3832
            macro avg       0.67      0.51      0.48      3832
         weighted avg       0.79      0.85      0.78      3832

In [45]: print("CONFUSION MATRIX:\n")
         print(confusion_matrix(y_test,predictions))

         CONFUSION MATRIX:

         [[2676  565]
          [ 450  141]]

In [46]: from sklearn.metrics import accuracy_score
         accuracy_score(y_test,predictions)
```

In the classification report, the output and predictions for some values are displayed. The accuracy of the model is 85%. Precision, f1-score and recall are calculated with help of confusion matrix. Confusion matrix shows 4 cells in matrix format, namely, true positive, true negative, false negative and false positive. True positive and false positive means that actual and predicted values are same. True negative and false positive means wrong predictions. The number of such data are represented. The different ratios of the values of confusion matrix are shown in classification report with precision is ratio of predicted to actual values, recall is ratio of predicted to false actual, etc.

RESULTS

The output of the data pre-processing is that unnecessary data and data which may cause anomalies are removed for analysis. The analysis has resulted in showing the relationship between different attributes of the datasets so we can make decision about which are the factors affecting the prediction of loan repayment. According to the analysis fico and credit policy are highly interrelated with interest rates and instalments of the debt.

There were models created using two algorithms namely, decision tree and random forest algorithms. Standard decision tree classifiers have the disadvantage that they are prone to overfitting to the training set. The random forest's ensemble design allows the random forest to compensate for this and generalize well to unseen data, including data with missing values.

Hence, the accuracy of the model is better by 10% in random forest classification. However, it may depend on the number of data as well. Hence, the accuracy may again vary as we add the data in future.

CONCLUSION & FUTURE WORK

To conclude, for the available datasets, random forest has proven to be better for prediction of loan repayment. The factors affecting loan repayment is fico and credit policy. The missing values and average values may affect the data but here 97% of data was complete hence we can say that both the models were accurate enough to predict if a person would repay the loan or not.

In future, more data needs to be added to avoid overfitting and to come to best fit. We have to continue to evaluate both the algorithms as the datasets vary. We plan to make a user interface so that any common man without knowledge of machine learning can enter the attributes values and can receive a prediction of a borrower is likely to repay the loan or not. This will not only increase the profits for LendingClub but also help in expanding the business as they will know whom to provide the loan and whom not to.

REFERENCES:

- www.towardsdatascience.com
- www.Kaggle.com
- www.wikipedia.com
- www.lendingclub.com