CAPSTONE PROJECT 1:
HOME LOAN PREDICTION

A Machine Leanring approach

Capstone Project 1 (Link)

Aditya Jakka
October 2018

Abstract

Dream Housing is a financial institution that grants loans to people for the purpose of acquiring property. There are a number of variables that are taken into account when granting a loan. Some of them are: the type of property being acquired (urban, rural, semi-urban), the applicant's income, the loan term, marital status, number of dependents, level of education, their credit history and various other variables.

I hope we can all agree that there is a certain degree of risk involved when granting loans to applicants. For example, it would definitely be a lot riskier to grant loans to people with no credit history. The company is also taking into account if the applicants are self-employed. Will they be able to continue making payments if they suffer a loss in business? Should applicants with incomes on the lower scale be granted large loan amounts? A 'risk analysis' is performed before a final decision is made.

This is clearly a complex and time consuming process which is also prone to human error (if done manually). Hence, with this project, a prediction model will be built which will make decisions on behalf of the company. That is, human intervention will not be necessary.

TABLE OF CONTENTS

INTRODUCTION

Financial institutions in recent years are trying to automate the process of loan approvals. One of the major motivating factors is that it eliminates the scope for human error. The loan approvals could be for a car, education, house or any other property. As discussed in the abstract, there are certain "**variables**" like **gender, income, loan amount, loan term, type of property** that are taken into consideration. These are all **independent variables.** These **independent variables** are taken into consideration when determining the outcome, which is our **dependent variable**, the **Loan_status.**

The key goal of this project is to design a prediction model that will make decisions (in this case predict the value of the **dependent variable**) in real time. For the design of this model, historical data will be used. This ***historical data*** is available in the csv file, '***train_data.csv'.*** The historical data contains information about the applicants (all the **independent variables)** and whether they were approved for a loan or not (i.e the **dependent variable**, the **Loan_status)**. The information of applicants for whom the loan approval decision is yet to be determined (**Loan_status)** is present in the ***test*** data file, '***test_data.csv***'.

The next part of the question is – "Who can benefit from this model?". A model of this nature could benefit any financial institutions that is in the business of granting loans. Depending on the type of loan granted, the dependent variables to be considered could vary. However, the crux of the concept and approach remains the same. This model is by no means a "perfect prediction model" and this project serves as a model upon which better prediction models can be built.

Proposed Approach

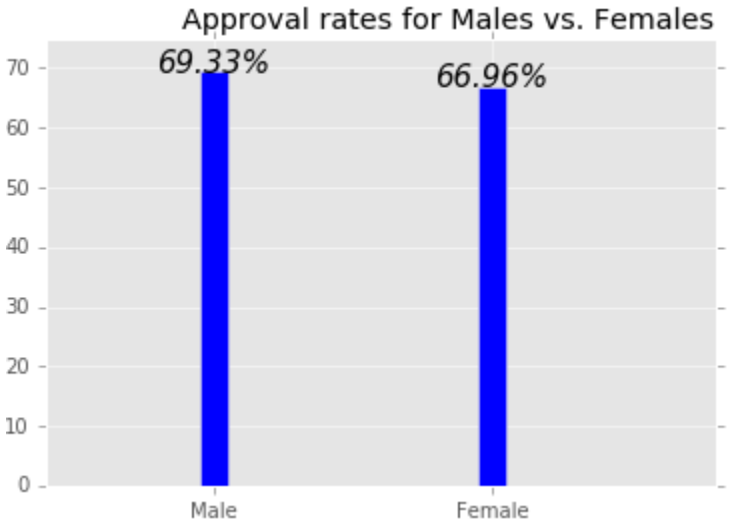
The dataset consists of a total of thirteen columns. The last column in the dataset, the **Loan_Status** is the **dependent variable**. Initially, the assumption is that all the other **independent variables** are critical in determining the loan approval. However, it is clear that the '**Loan_ID'** column plays no role in determining a decision. Hence, we will begin our analysis by excluding this feature altogether from our analysis.

For classification, we will apply a variety of machine learning classifiers. 'Logistic Regression', 'Support Vector Machines', 'Random forests', and 'Gradient boosting' will be used. The latter two are 'ensemble' methods, and with hyper parameter tuning, are expected to perform better. We will then evaluate our models by calculation various performance metrics and choose the best classifier.

Initial Hypotheses

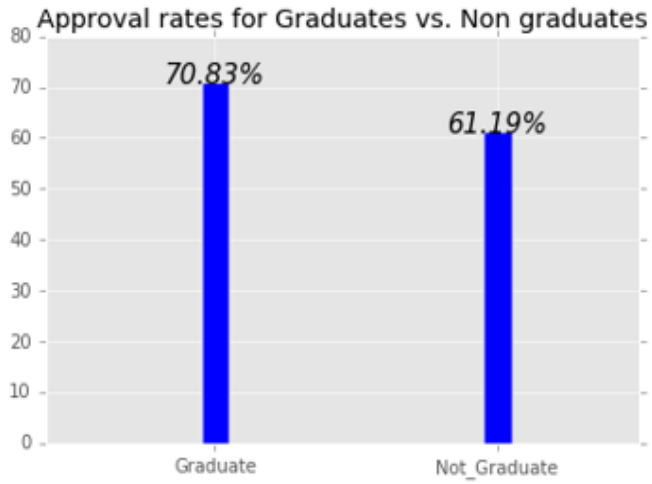
Before we clean the data and get into the nuts and bolts of predictive modeling, let us try and make a hypotheses and observe if the data echoes our assumptions.

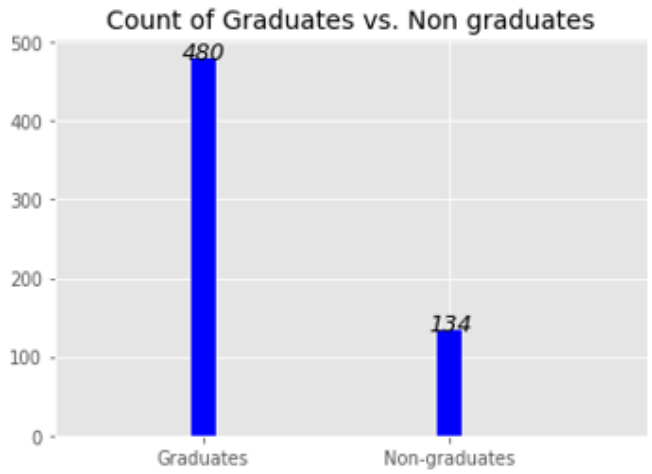
- Let us explore the data and see if males have a greater chance of approval. I computed the percentages of males vs. females approved for the loan.



From the bar plot above, we observe that indeed males do have a higher approval rating than females. However, the difference in percentage isn't significant enough to definitively confirm our hypothesis.

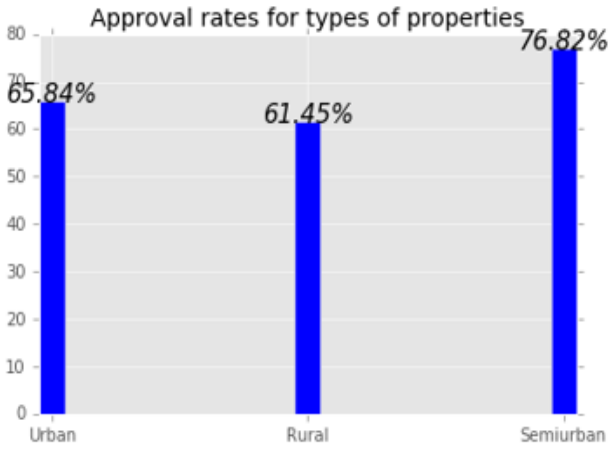
- Next, let us explore to see if Graduates have a higher chance of approval in comparison to non-graduates. From the bar graphs below, we see that graduates indeed have a ~10% higher approval chance. However, let's also see how many graduates and non-graduates were in the list. ***If there are an insignificant number of non-graduates in comparison to graduates or vice-versa, then we might have to consider rejecting the hypothesis as there aren't enough samples to support our hypothesis. [Note: This process will be followed for evaluating the other variables as well.]***

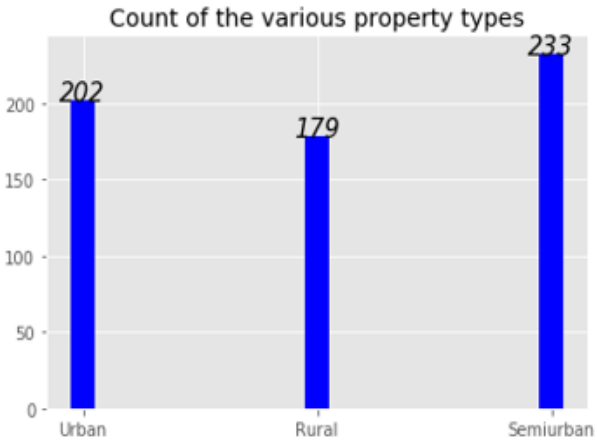




For a total of 614 applicants, 134 were non-graduates. Hence, we can argue that our hypotheses could be true.

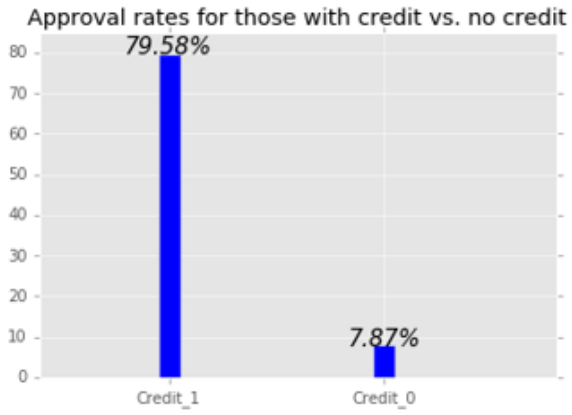
- Next, let us consider the property type and the percentage approvals. We see that semi-urban areas have a higher approval rate. They have a 15% higher approval rate than rural properties and more than 10% approval rate than urban areas. But let us also consider the count of the number of Urban, rural and semiurban properties in question. Two bar plots has been illustrated below.

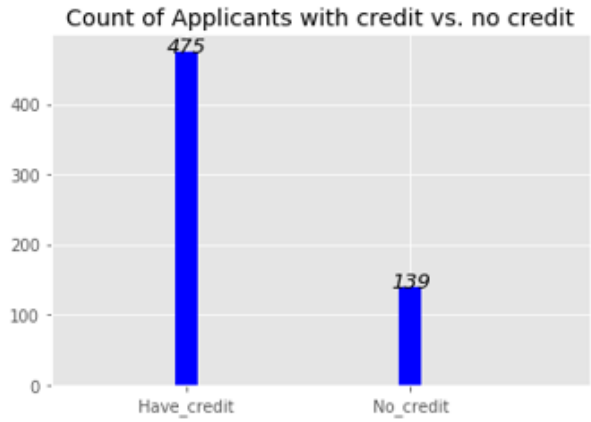




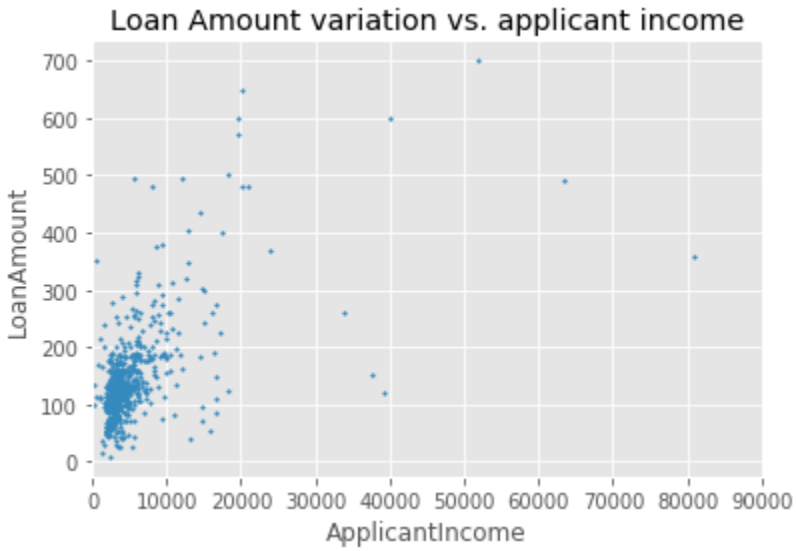
We see a significant number of loan applications for each of the property types. Hence, we could again argue that perhaps, our hypotheses holds true.

- Next, we will try to determine if people with a credit history has a higher chance of loan approval as opposed to those who don't. Let us explore the bar graph below. It would seem that applicants with a credit history are 10 times more likely to get approved. But before we confirm our hypothesis, let's investigate the number of people with credit vs no credit. Again, there are a significant number of samples for people with no credit. Hence, we could again argue that for now, our hypothesis holds good.

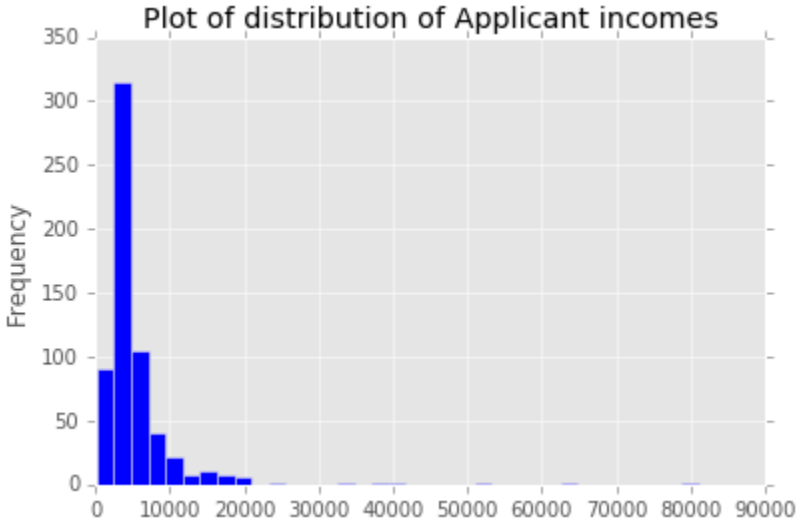




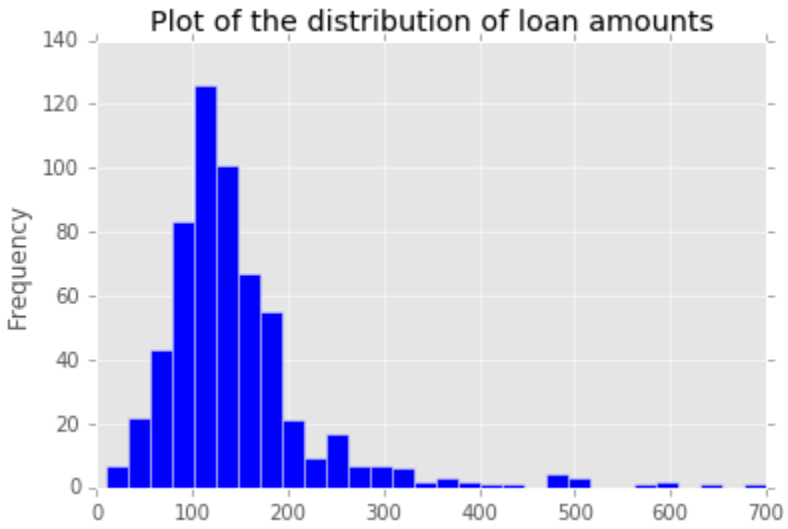
- Next, let us try to see if there is a relationship between the applicants' incomes and their loan amounts requested. While there is some indication that applicants with higher incomes requester larger loan amounts, the data points are diversely spread through the graph and we cannot confidently conclude that there is a solid pattern that supports our hypothesis.



- Next, let us consider the distributions for applicant income. This has been illustrated with a histogram below. From the histogram plot below, it's evident that most of our applicants make $5000 a month or less. In fact, more than 50% of the applicants make between $2500 and $5000.



- Let us also consider the distribution for loan amounts requested. We see that most of the applicants request a loan amount of $200K or less.



Cleaning Steps performed

Before I get started, I will attach a snapshot of all the columns in the dataset.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
Loan_ID              614 non-null object
Gender               601 non-null object
Married              611 non-null object
Dependents           599 non-null object
Education            614 non-null object
Self_Employed        582 non-null object
ApplicantIncome      614 non-null int64
CoapplicantIncome    614 non-null float64
LoanAmount           592 non-null float64
Loan_Amount_Term     600 non-null float64
Credit_History       564 non-null float64
Property_Area        614 non-null object
Loan_Status          614 non-null object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.4+ KB
```

There are 614 observation in total. Some of the columns have missing values. These values had 'NaN' assigned for missing values.

Before we can an in-depth analysis of the cleaning steps performed, I will explain two concepts: Mode, median and Outliers

Mode - The **mode** of a set of data values is the value that appears most often. Let us consider a list, x = [1,2,2,3]. The mode of the list x is 2 since it is the most repeated value.

Median - The median is the value separating the higher half from the lower half of a data sample. Let's consider a list x = [3,4,5,6,7]. The median of x in this case is 5. For an even number of observations, the average of the two middle observations is computed as the median. This is represented by the *50th percentile* line in the box plot below.

Gender – The missing values of the gender column were filled with 'Other'.

Married – Missing values were filled with the mode value. The 'NaN' or missing values were then substituted with this value. It was '*Married*' for this column.

Self Employed – The mode value was computed once again.

LoanAmount – The mean of Loan amounts was computed and assigned to the NaN values.

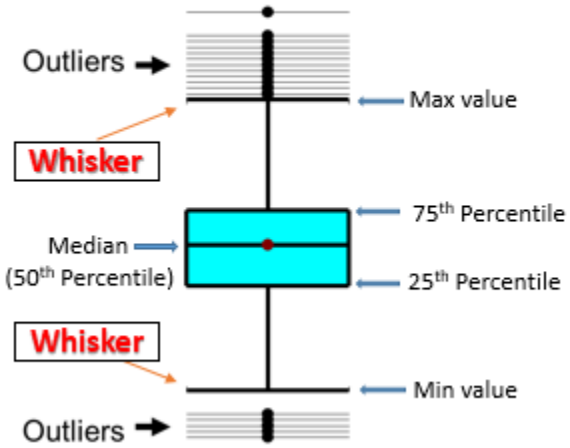
Loan_Amount_Term – The mode was computed and assigned to the missing values. It was 360.

Credit History – The NaN values for credit history were filled with 0.0.

Dealing with Outliers

Outliers - An **outlier** is an observation point that is distant from other observations. These data points could be a result of errors or wrongly recorded observations. These are often excluded from analysis as it could have an adverse effect on the reliability of our model.

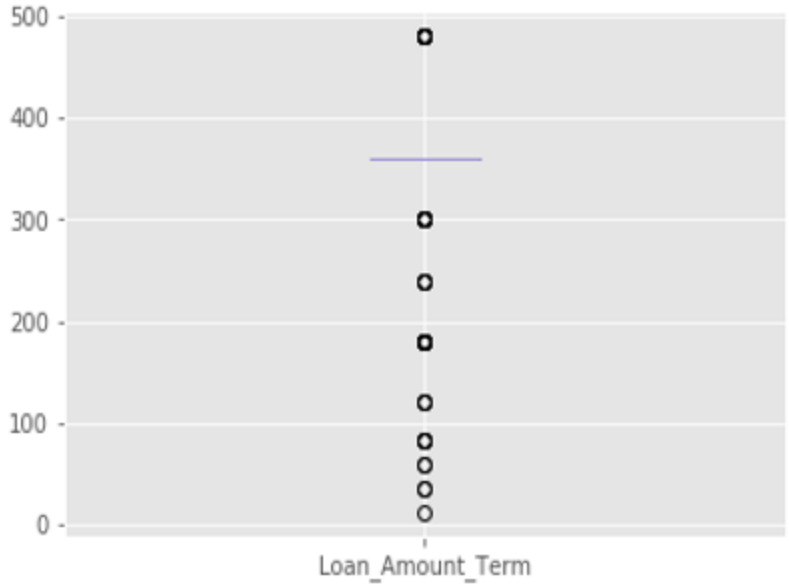
We will identify outliers with the help of box plots. A typical box plot with outliers is illustrated below:



The data points above and below the **whiskers** are our **outliers**. The min and max values in this plot have been plotted by excluding the outliers from the calculation.

I will now outline the steps I took to clean the data:

Loan_Amount_Term – From the box plot, there is one data point > 360. On pulling values from the dataframe for Loan_Amount_term > 360, I observed that there quite a few applicants for loan_amount term = 480 and considering the relatively small size of this dataset, these data points will be taken into consideration for analysis. However, there are a few data points below 100.



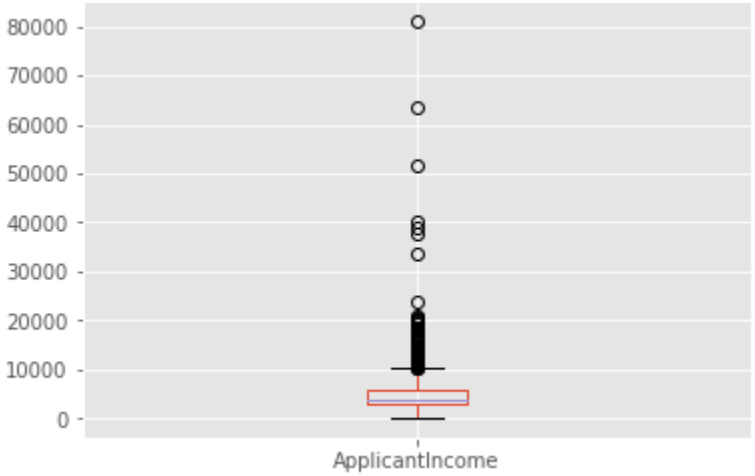
Loan Amount terms < 100

Loan_Amount_Term
60.0
60.0
36.0
84.0
84.0
12.0
36.0
84.0
84.0

So when I take a peek at that data, here's what I get: A total of 9 data points below 100. However, we can see that there are just 2 values below 50: 12 and 24.

Proposed course of action: While it seems highly unlikely that applicants would apply for a 12 or 36 term housing loan, these data points will still be considered for analysis.

ApplicantIncome - By observing the box plot for applicant incomes, we again see quite a few outliers.

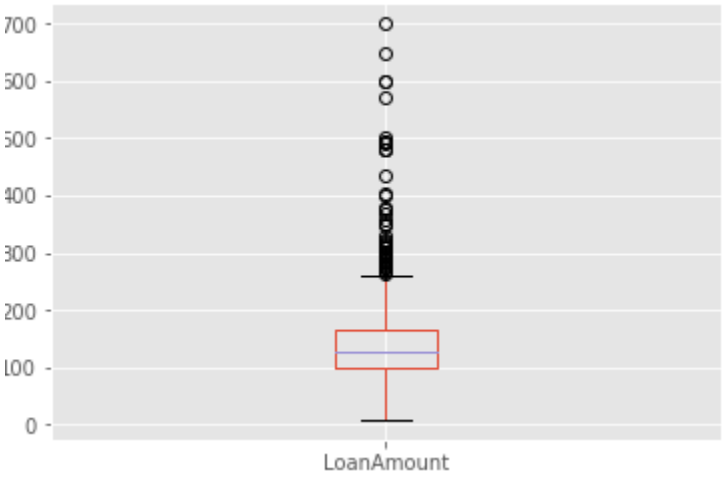


ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
51763	0.0	700.0	300.0	1.0	Urban	Y
63337	0.0	490.0	180.0	1.0	Urban	Y
81000	0.0	360.0	360.0	0.0	Rural	N

All the values in the boxplot from 50000 and above seem pretty far away and we see that there are indeed just three data points.

Proposed course of action: These rows will be considered for our analysis.

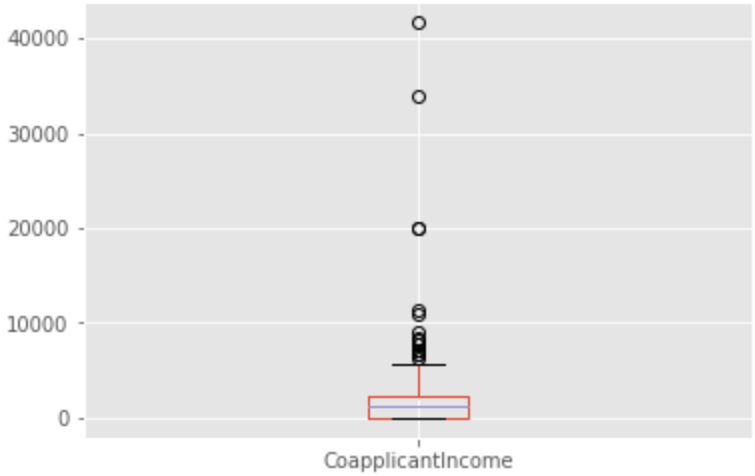
LoanAmount - The box plot shows quite a few outliers. We see that some outliers over 500 (illustrated beside the box plot) that are pretty isolated. We find 3 data points in the 600s and one at 700.



LoanAmount
650.0
600.0
700.0
570.0
600.0

Proposed course of action: Consider all data points.

CoapplicantIncome - On observing the box plot, we again see quite a few outliers but most of them are located close to the box plot. However, there are three data points over the 20000 mark that appear to be pretty isolated.



Proposed course of action: There are only 3 data points above 20000. All rows will be considered for analysis

Data Assumptions and Limitations

The data set contains only several 100 rows which means that our training set is limited. Ideally, we would like to train our model on as many observations as possible. There are a total of 13 features including the Loan_ID, a feature which doesn't influence the outcome of the dependent variable, Loan_Status.

We are also unsure about how this data was originally obtained. That is, we do now know how recent the data is. Ideally, we would like to train our models on data as recent as possible. Was all the data obtained for a particular year? Or does this data contain a history of loan approval decisions spanning several years?

Perhaps, there are other "features of interest" that influence the loan approval process. This information could have helped better train our model perhaps. Some of these features could include information containing (but not limited to): does the applicant have a history of loan defaults, are the applicants currently making payments on their home/auto loans, how much do they spend on rent, or have they made any investments in the past few years.

For our model however, we will assume that the data is recent and that the features we currently have will help us make a good prediction model.

Statistical Analysis

Now that we have a clean dataset to being our analysis with, this section with deal with statistical analysis of the data set. Before the section digs deep in to the math/statistics, a few definitions will be explained that will help perceive the results of the analysis better.

Hypothesis testing and p-values

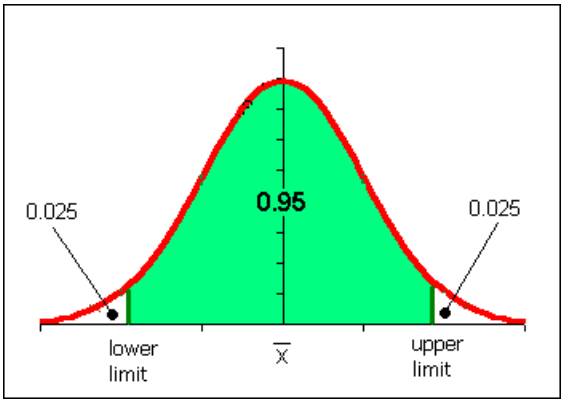
A statistical hypothesis, sometimes called confirmatory data analysis, is an hypothesis that is testable on the basis of observing a process that is modeled via a set of random variables. A statistical hypothesis test is a method of statistical inference. Commonly, two statistical data sets are compared, or a data set obtained by sampling is compared against a synthetic data set from an idealized model. A hypothesis is proposed for the statistical relationship between the two data sets, and this is compared as an alternative to an idealized null hypothesis that proposes no relationship between two data sets. The comparison is deemed statistically significant if the relationship between the data sets would be an unlikely realization of the null hypothesis according to a threshold probability—the significance level. Hypothesis tests are used in determining what outcomes of a study would lead to a rejection of the null hypothesis for a pre-specified level of significance. For more information, follow this link.

p-value

The p-value is the level of marginal significance within a statistical hypothesis test representing the probability of the occurrence of a given event. The p-value is used as an alternative to rejection points to provide the smallest level of significance at which the null hypothesis would be rejected. A smaller p-value means that there is stronger evidence in favor of the alternative hypothesis. For more information on p-values, refer this link.

Our p-value is set in advance i.e we before we begin our hypothesis testing. If we set a p-value of 5% (0.05), and we get a p-value less than 0.05, this would mean that this data point will not be within the 95% confidence interval. Let us try to visualize this.

(Refer to the illustration and the explanation in the following page.)



If we have a p-value greater than 0.05, then it lies in the green area. Else, it would lie in the white area (indicated by the area of the two tails "0.025", which accounts for the total probability of 5%

In layman terms, if we find the level of significance of our test to be less than 5% (or whatever hypothesis level we set), this would indicate a very rare event and one that didn't happen by random chance alone.

For our tests, let us set the level of significance to 0.05 or 5%.

2-Sample t-tests

Two-sample hypothesis testing is statistical analysis designed to test if there is a difference between two means from two different populations. The two-sample t-test is one of the most commonly used hypothesis tests to compare whether the average difference between two groups is really significant or if it is due instead to random chance.

The null hypothesis here would be that there is no significant difference between the means of the populations and any difference observed is due to random chance. The alternative hypothesis would be that there is indeed a significant difference in means between the two distributions.

For more information on 2 sample t-tests, please refer to this video from Khan Academy.

Pearson Correlation Coefficient (r)

The Pearson correlation coefficient, r, can take a range of values from +1 to -1. A value of 0 indicates that there is no association between the two variables. A value greater than 0 indicates a positive association; that is, as the value of one variable increases, so does the value of the other variable. Conversely, a negative value indicates that if one of the variables increases in value, the other decreases. The formula is displayed below (*refer next page)*:

$$r = r_{xy} = \frac{\mathrm{Cov}(x, y)}{S_x \times S_y}$$

Cov(x,y) represents the covariance of x and y. S_x and S_y represent the standard deviations of x and y respectively. The pearson correlation, r is a dimensionless quantity.

For more on Pearson correlation coefficient, please follow this link.

A series of two sample t tests were carried out. Its results and explanations will now be explored in detail.

- **H_0 : There is no significant difference between males and females in terms of loan approvals`**

  H_a : There is a significant difference between males and females in terms of loan approvals

  The two sample t-test yielded the following results:

  t_statistic = 0.48608, **p_val** = 0.627

  Our p-value of 0.627 is much greater than the set level of 0.05. Hence, we cannot reject our hypothesis as our test doesn't indicate that gender is an influencing factor.

- **H_0: There is no significant difference between loan approvals for semi-urban and rural areas**

  H_a: There is a significant difference between loan approvals for semi-urban and rural areas.

  During our initial hypothesis, we found out that semi-urban areas had a higher loan approval rate in comparison to urban and rural areas. Let us do a significance test for the same. The two sample t-test yielded the following results:

  t_statistic = 3.419, **p_val** = 0.00069

  The small p-value above indicates that property type does matter when granting loan approvals. Hence, this wasn't due to random chance. A plot of the distributions of the various property areas has been plotted below.

We can notice that semi-urban property have a significantly higher percentage of approvals.

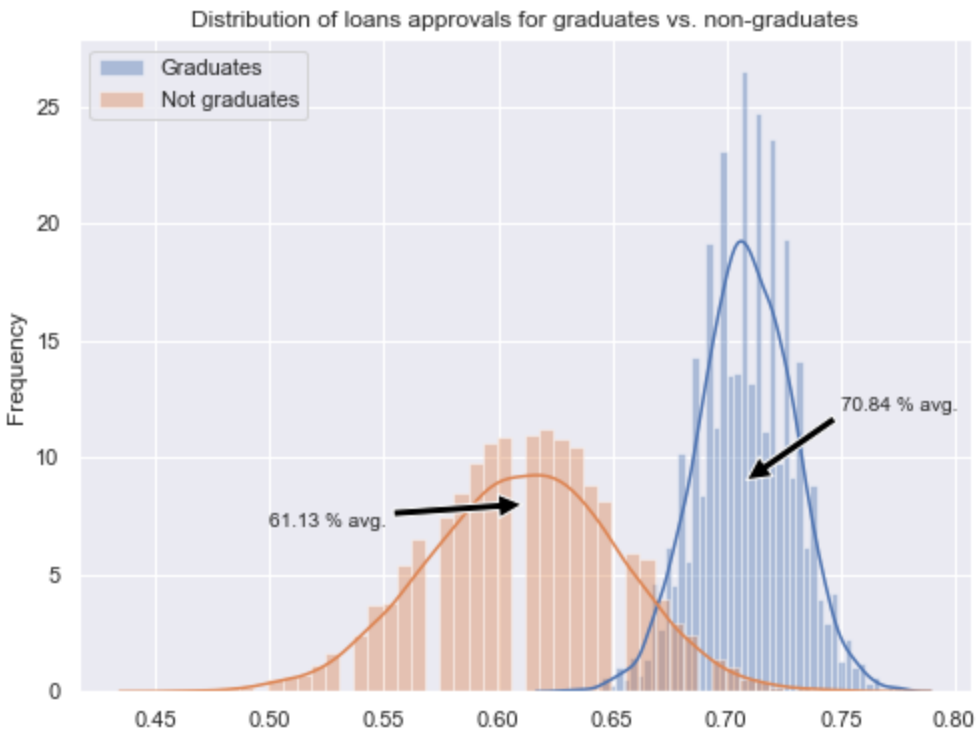
- **H_0: Graduates and non-graduates have an almost equal chance of having their loan approved.**

  H_a: Graduates and non-graduates have an almost equal chance of having their loan approved.

  A two sample test was carried out once again which yielded the results:

  t_statistic = 2.1325, **p_val** = 0.033

  The p-value obtained is < 0.05 which implies statistical significance. The plot below is a mean of the distributions for graduates vs. non graduates.



The significantly higher loan approval rates for graduates has been illustrated in the plot above.

- **H_0: The number of dependents of an applicant has no significance on the loan approval rates.**

  H_a: The number of dependents of an applicant has statistical significance on the loan approval rate.

  Various t-sample tests were carried out but none of them yielded a p-value less than 0.05. Hence, we cannot reject our null hypothesis. There were four categories here: 0 dependents, 1 dependent, 2 dependent, and 3 or more dependents. Let us visualize this result below by plotting the means of the distributions for various number of dependents. The p-value from the 2 sample t test for 2 dependents vs. 1 dependent yielded ~0.1 which is close to our significance value. Hence, we can see that these distributions are quite far away from each other. However, since we had our threshold set at 0.05, we didn't reject the null hypothesis.

- **H_0: There is no significant difference in loan approvals for people with credit history vs. those with no credit history.**

- **H_a: There is a significant difference in loan approvals for people with credit history vs. those with no credit history.**

  A two sample t-test was carried out for applicants with credit history vs. for those with no credit history. The results were:

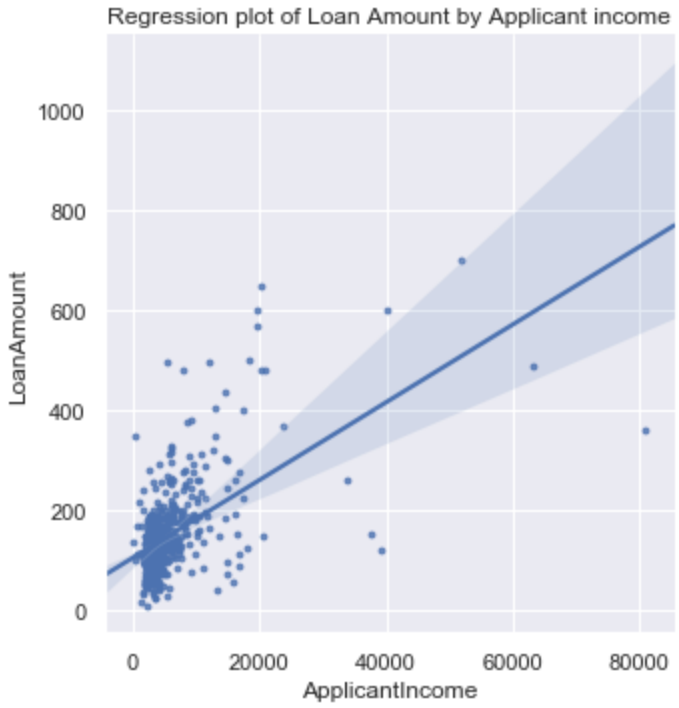
  t_statistic = 11.87, **p_val** = 2.1445*e^-29

  Our p-value is extremely small and hence, this signifies statistical significance. A visualization of the means of the distributions has been plotted below:

  It would seem that applicants with credit history have a much better overall chance of obtaining a loan.

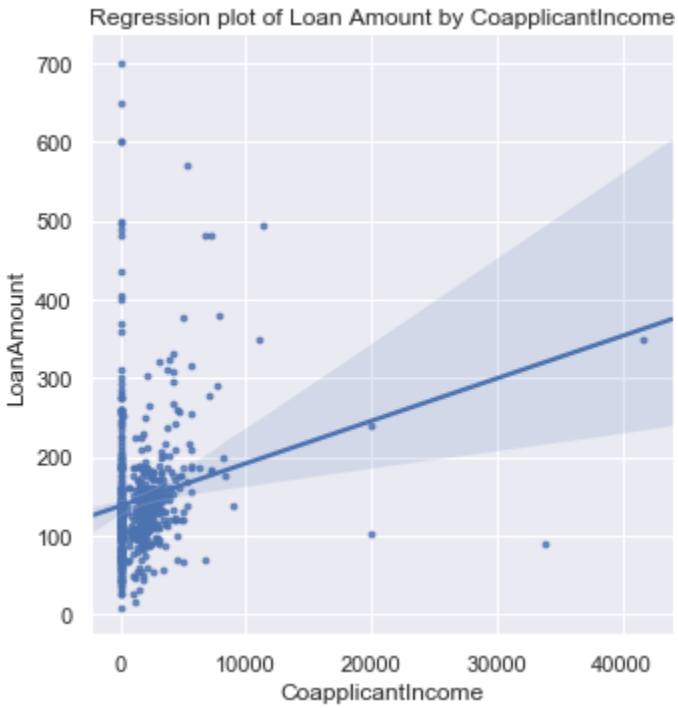
Now, let us try to explore the correlations between some of the independent variables. To achieve this, the Pearson correlation coefficients was computed. A series of the Pearson_r tests and their explanations will be explored below.

- From an intuitive sense, it would make sense for an applicant's income be somewhat proportional to the Loan amount requested. Let's see if this is the case. Computing the pearson_r value yielded 0.5656. This signifies a strong positive correlation. A linear regression plot between these two variables has been illustrated below.



- Next, let us observe the correlation between the Coapplicant income and the Loan Amount.

  The pearson_r value computed was 0.1878 which signifies a positive correlation albeit a weak one. Let us visualize this relation with a regression plot below.



- Let us now see what we can make of the relation between the Loan Amount requested and the loan amount term.

  The pearson_r value was computed to be 0.036. This is indeed a very weak positive correlation that is very close to 0.

Machine Learning for Classification

What is it?

Machine learning is a branch of Artificial Intelligence that provides systems with the capabilities to automate learning and improvise based on information in the training set without having to be explicitly programmed. One of the greatest uses of machine learning today is to automate decision making. Machine Learning can be used in a wide variety of industries like the financial sector, ecommerce industry, social networking and health industry.

Data preprocessing for training machine learning models

Most machine learning models understand data when it is represent by 1s, 0s or numerical data. In addition to the numerical variables (ApplicantIncome, CoApplicantIncome, Credit_History, Loan_Amount etc.), the data contains categorical variables like Gender, Education (Graduate or not), Property_Area etc.

These categorical variables have to be converted to generate dummy variables. This can be achieved by using pandas' get_dummies function. Let's take the education categorical variable. It either takes the values "Graduate" or "Not_graduate". Once the get_dummies function is applied, we will have two columns – "Education_Graduate" and "Education_Not_graduate". The Education_graduate column will contain 1s for applicants who are graduates and 0s for those who aren't. Similarly, "Education_Not_Graduate" will contain 1s for applicants who are not graduates and 0s for applicants who are graduates.

It is quite evident that these two newly created variables are strongly correlated to each other. That is, if one is 0, then the other is 1 and vice versa. As a rule of thumb, we do not include such highly correlated variables for our analysis. Moreover, this is redundant data. Hence, we only include one of these variables for our analysis. By including all dummy variables for our analysis, we would fall in to the dummy variable trap.

The dependent variable, "Loan_Status" is also a categorical variable with a "Yes" or "No". We create a separate column for this which reads a 1 for approved loans and 0 if not. The next step, which is our last step in the data preprocessing, is feature scaling.

Feature scaling

Once we have our dummy variables set up, we now need to perform feature scaling. In most data sets, we will have features with highly varying magnitudes, units and range. Features with greater magnitudes will carry a greater weight than those with low magnitudes. To nullify this effect, feature scaling is performed.

This article contains some useful information on feature scaling.

Various classification techniques were tried considering factors like accuracy, area under the ROC curve and area under the Precision vs. Recall curve.

Performance metrics explained

Accuracy – It is simple the ratio of the correctly predicted samples to the sum of correctly predicted samples and the wrongly predicted samples.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

True positives – The observations for which the algorithm predicted YES and the observed output (ground truth) was also YES.

False positives – The observations for which the algorithm predicted YES and the actual output was NO.

True negatives – The observations for which the algorithm predicted NO and the actual output was NO.

False negatives – The cases in which we predicted NO and the actual output was YES.

True positive rate – It is the ratio of true positives to the sum of true positives and false negatives.

$$TruePositiveRate = \frac{TruePositive}{FalseNegative + TruePositive}$$

False positive rate – It is the ratio of False positives to the sum of False positives and true negatives.

$$FalsePositiveRate = \frac{FalsePositive}{FalsePositive + TrueNegative}$$

Precision – It is the ratio of the true positives to the sum of true positives and false positives

Recall – It is the ratio of the true positives to the sum of true positives and false negatives.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Next, in depth analysis of the performances of the various classifiers will be discussed.

Splitting the data

The dataset will be split into the training and test sets. The test set size has been chosen to be 25% of the original dataset. That is, the classification algorithm will train on 75% of the data and then, the model will be used to make predictions on the test set.

Random Forest Classifier

Why Random Forests over decision trees?

Decision tress train over a single training set only. Decision trees take into account each and every variable and every observation in the training set. While decision trees are very fast (in terms of computational speed), they generally overfit the data and perform poorly on test sets.

To solve this, we could generate bootstrap samples. That is, we generate samples with replacement from the original sample set. We now have multiple samples instead of one training sample. But these training samples are bound to have duplicate observations since we sampled with replacement. While this model does perform better, it does have access to all the features and most of the observations. To enhance the randomness, each sample is trained with a unique set of features. If we have "n" number of features, the algorithm uses sqrt(n) features for each model. What results is a series of "weak models". These weak models are then "aggregated" to give a complex but strong prediction model. Hence, the randomness comes from the bootstrap samples generated and the features involved for predicting each sample.

Since this is an ensemble approach which combines the best of different approaches, we will try the Random Forest classifier to predict our model with the hope of achieving "large initial gains".

The primary assumptions for this algorithm is that the randomly generated bootstrap samples with each sample set using a distinct set of features are independent of each other.

Performance of the model

The Random forest classifier with the default parameters achieved a cross validation score o5 0.69. The test set accuracy was also 0.69. Since the Random Forest Classifier is an ensemble method, hyper parameter tuning will be used to boost performance.

The hyper parameters that were tuned are:

n_estimtors: This represents the maximum number of trees to build before aggregating all of these trees to form a single predictor. Generally, higher number of trees give better results.

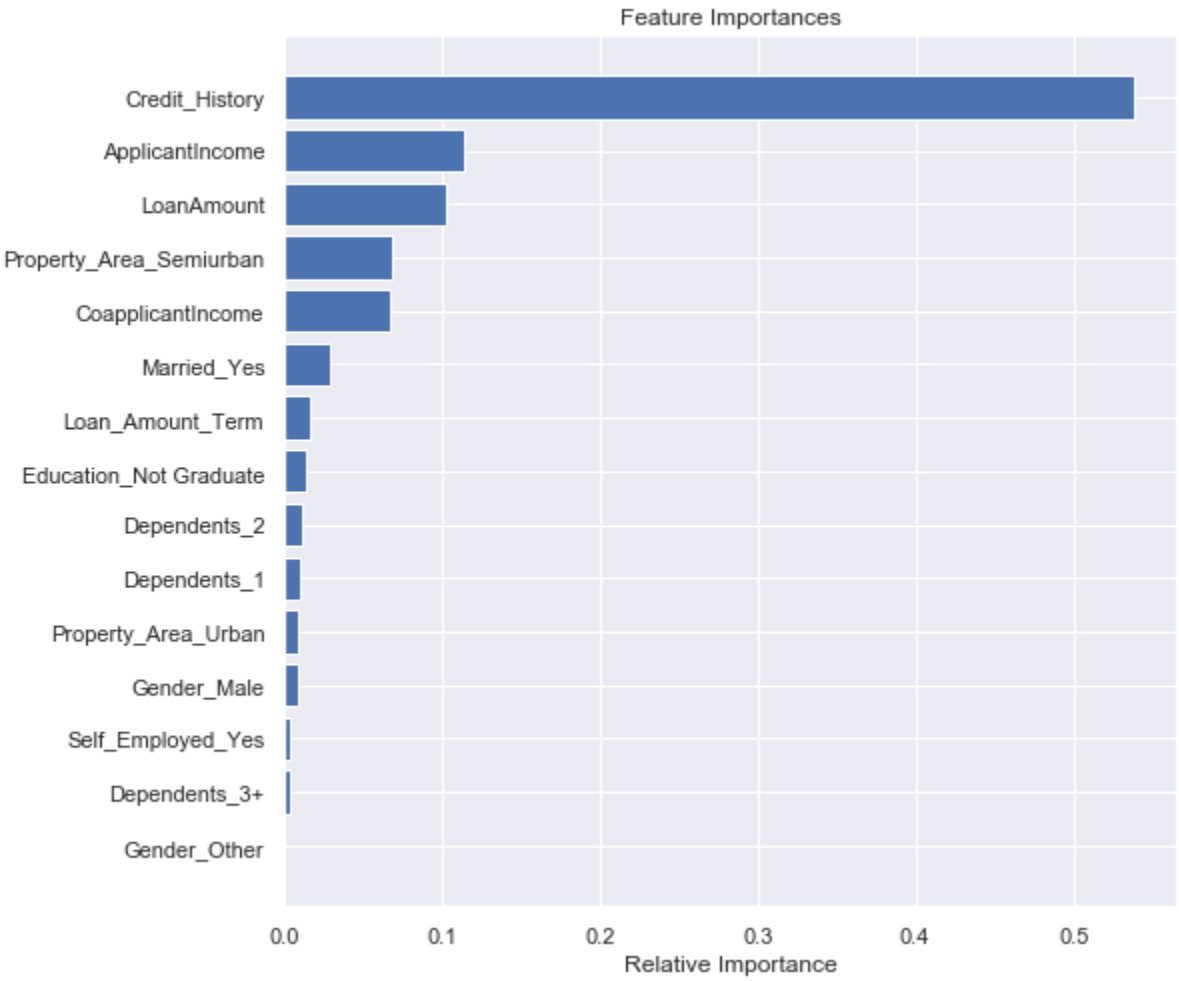
max_features: Represents the maximum number of features to consider for a particular tree. Usually, the square root of the total number of features is chosen.

Criterion: We choose between ‘gini’ and ‘entropy’.

Bootstrap: The default is True. If set to true(default), the random samples are generated by sampling with replacement.

Min_sample-leaf: A leaf represent the end node of a decision tree. Smaller leaves tend to produce ‘noisier’ models.

With the tuned hyper parameters, a cross validation score of 0.74 was achieved on the training set. The test set accuracy was found to be 71%. A feature importance plot has been illustrated below:



The most important features for the model were: Credit History, applicant income, loan amount and property type.

Support Vector Machines

How do they work?

SVMs are used to predict non-linear models. It is a supervised learning algorithm and a non-probabilistic classifier. Given a training set, the SVM algorithm separates the data points into various labelled classes using the ***kernel trick*** *. The kernel trick is used to map points in the high dimensional space essentially building the non-linear classification boundary. Moreover, the kernel trick drastically reduces computational time. SVMs perform better than linear models since most real world data is non-linear in nature.*

The SVM classifier performed decently with a cross validation score of 0.74 on the training set and 0.75 on the test set. The kernel used was 'rbf'. The other kernel option is 'linear'. The model wasn't trained using the linear kernel since our data is non-linear in nature.

The hyper parameter tuned was 'C'. This parameter accounts for the "slack". Lower values of 'C' imply more slack and vice versa. Usually, we would prefer to have small values of 'C' which results in a smoother decision boundary.

Using hyper parameter tuning, the best value for c was found to be 0.0001. This time, the accuracy on the training set was almost ~76%, which is a slight improvement over the default parameters.

Logistic Regression

This a traditional and basic approach to classification in supervised learning. It derives its name from the **logit function.**

The logistic regression model is performed with some assumptions: that data has no outliers, there are two classes to be predicted, and that no two independent variables are highly correlated to each other.

Using the default parameters, the test accuracy was around ~73%.

With hyper parameter tuning, a test set accuracy of ~75% was achieved. The hyper parameter tuned was 'C'. The 'C' here is the same as the one used for tuning SVM. Smaller values of C implement more slack and vice versa.

Gradient boosting

The last classifier used was the gradient boosting classifier. Gradient boosting, like the Random Forest Classifier, is an ensemble algorithm. In this algorithm, detection errors in earlier trials are prioritized and the model then tries to classify these points correctly. These "weak" learners to combines to form a complex but effective predictor. This article explains gradient boosting in detail.

Using the default parameters, the accuracy on the test set was ~74%. The following parameters were tuned for improving performance:

Loss**:** 'Deviance' and 'exponential' were chosen. The logistic loss computes deviance. The exponential loss is calculated using the exponential loss function.

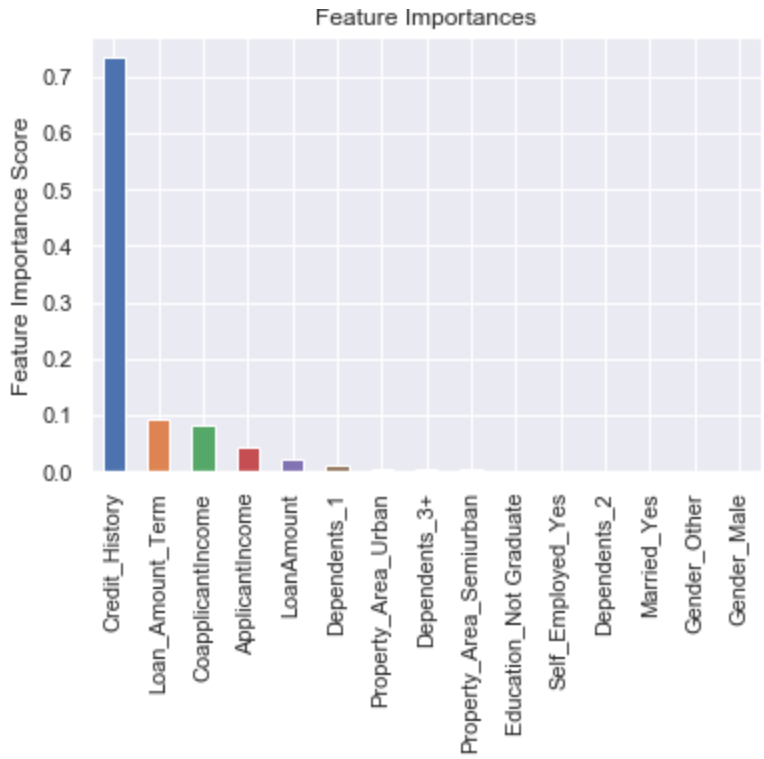
N_estimators: This is the sequential number of trees to be modeled.

Max_depth: It is the maximum depth of the tree. It is used to control overfitting.

Learning_rate: This determines the impact of each tree on the final outcome. Lower values are generally preferred as they make the model robust to the specific characteristics of tree and thus allowing it to generalize well.

Min_samples_split: The minimum number of samples in a terminal node or leaf.

Criterion: This parameter measures the quality of split. ''friedman_mse', 'mse', 'mae' are the opti ons. 'MSE' represents 'mean squared error' and 'mae' represents the 'mean absolute error'
By choosing the best hyper parameters, a test set accuracy of ~76% was obtained. In terms of accuracy, gradient boosting outperformed all the other models.
The feature importance plot for the gradient boosting classifier has been illustrated below. Credit history, loan amount term, co-applicant income and applicant income were the most important f eatures.

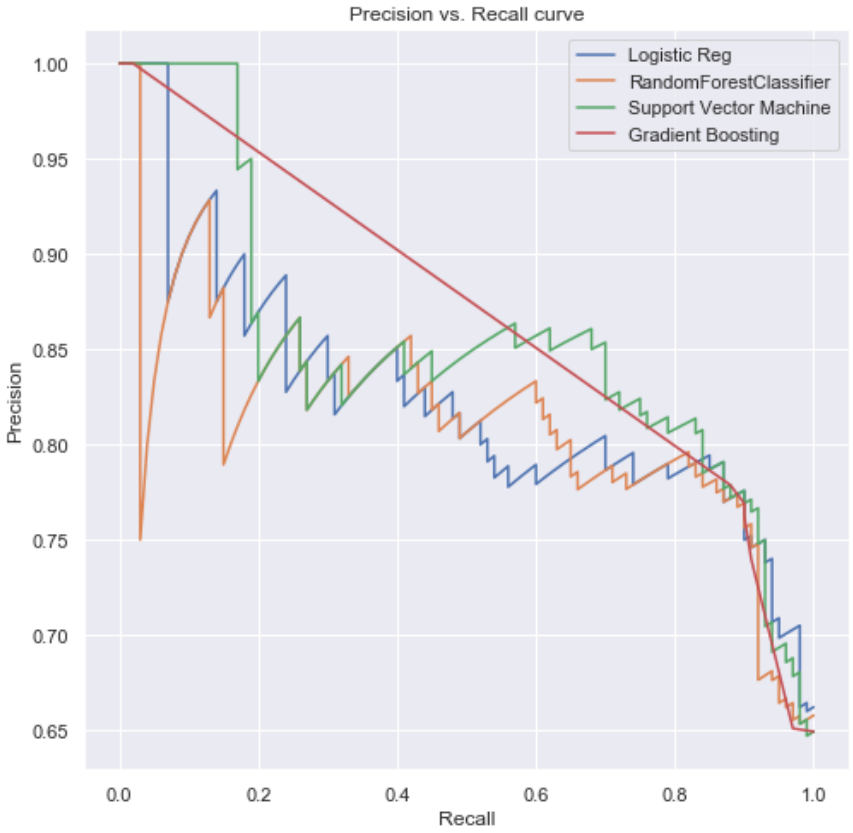


From the above plot, we see that the gradient boosting classifier nullified the effect of some of the features it deemed ‘irrelevant’ or noisy.

Choosing the best Classifier

Judging by the accuracy metric alone, the Gradient boosting classifier outperformed the rest of the classifiers. But is this metric alone important? What if our data set is imbalanced? This is, there is a significant difference between the number of observations in the positive and negative class. This metric alone wouldn't suffice.

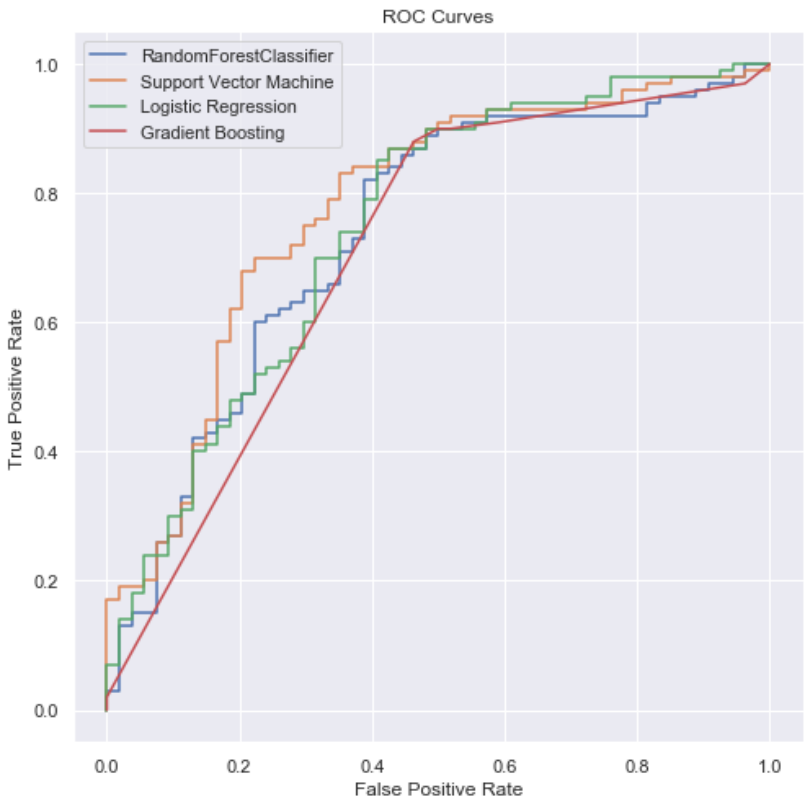
A plot of the precision vs. recall curves is illustrated below. Greater the area under the curve, the better the model.



Areas

Classifiers	
Gradient Boosting	0.869090
Support Vector Machine	0.852386
Logistic Regression Area	0.824748
RandomForestClassifier Area	0.812947

Once again, the gradient boosting classifier is the winner with the SVM coming in at a close second. Next, we will plot the ROC curves. These curves explain the tradeoff between the true positive rate and the false positive rate. Just as the Precision vs. Recall curves, greater areas under the curve represent better models. This plot has been illustrated in the following page:



Areas_ROC

Classifiers	
Support Vector Machine	0.777222
Logistic Regression Area	0.746296
RandomForestClassifier Area	0.740000
Gradient Boosting	0.709630

The result is quite unexpected. The Gradient boosting classifier performed worse than all the other classifiers.