Loan Status Prediction

REPORT

Guided by:
Dr. Amir H. Gandomi

Akshay Kirolikar
Ankur Morbale
Gaurav Venkatraman

Fall 2018

TABLE OF CONTENTS

1. Abstract 3

2. Introduction 4

3. Problem Description 5

4. Dataset Description 6

5. Data Preprocessing 7

6. Dimensionality Reduction using PCA 10

7. Classification Methods 13

8. Conclusion 21

9. Future Scope 22

ABSTRACT

Home Loans and charging an interest on it from the borrower are one of the sources of revenue for every bank. A technique which can accurately predict potential customers for a loan would be a major asset to the bank to mitigate risk. One of the ways to do this is to build a predictive model on a historical data of loan applicants and use the trained model on new data. In this project we attempt to build a model using one such loan status data. Our goal is to predict the 'Loan_Status' variable as 'Yes' or 'No'. Any imbalance in the predicted variable is dealt with using sampling techniques. Dimension reduction techniques like PCA follows to select best components. Then, we have applied machine learning algorithms like Logistic Regression, Naïve_bayes, k-Nearest-Neighbors, Support Vector Machines to build a predictive model for loan status prediction. Our research and analysis show that the kNN model proves to be the most effective model giving the most optimal solution.

INTRODUCTION

Loan prediction problems are an extremely relevant topic in today's world with the focus and move towards automation and data driven solutions to meet business needs. Banks are also moving in this direction. This creates a demand for technology solutions for problems like which customers to give loan to. It would also help in identifying and acquiring new customers who would be candidates for a loan and would benefit from it. But, the recovery of the loan amount on time is uncertain and a big concern for every institution. This is where an accurate model for predicting loan status come in. We have attempted to build a binary classifier which categories recipients into two classes 'Yes' and 'No' which would work as a prescreening tool for banks to decide on whether an applicant should be given a loan or not.

PROBLEM STATEMENT

Dream Housing Finance company deals in all home loans. They have presence across all urban, semi urban and rural areas. Customer first apply for home loan after that company validates the customer eligibility for loan. Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set.

DATASET DESCRIPTION

This project deals with a binary classification problem. The data consists of 614 observations, 12 independent variables and 1 dependent variable. The goal is to predict the class of the dependent variable which in this case is *'Loan_Status'* as 'Y' or 'N'. The independent variables are Loan_ID, Gender, Married, Dependents, Education, Self-Employed Status, Applicant Income, Co-applicant income, Loan Amount, Loan Amount Term, Credit History and the Property Area

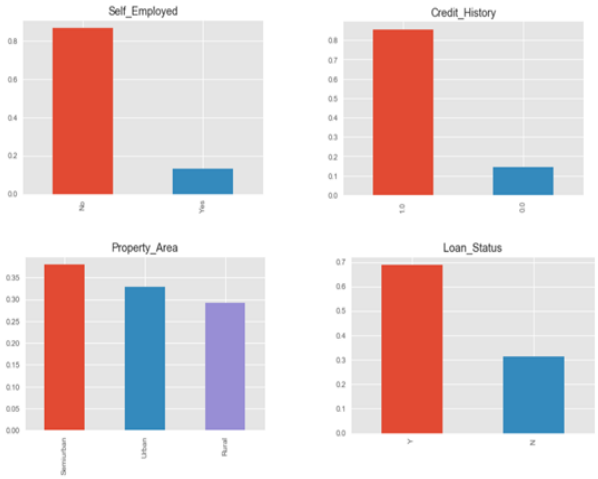
Independent variables=>

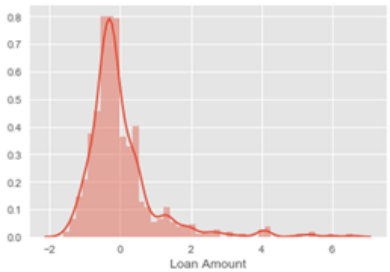
- Loan ID
- Gender
- Marital Status
- Dependents
- Highest Education Degree
- Is the person Self-employed
- The Income of the Applicant
- The Income of the Co-applicant
- Loan amount requested
- Loan amount term in days
- Credit history
- Property Area

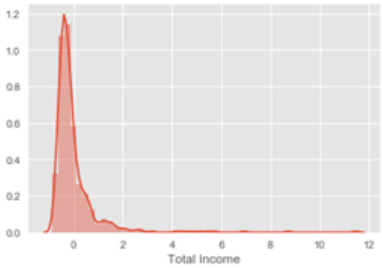
Dependent variable=>

- Loan request Status(Y/N)

For our analyses, we have dropped Loan ID from our Independent variable







DATA PREPROCESSING

With the tremendous explosion of raw data available from multiple sources, managing this data has become an ever-present concern for businesses across the globe. Preparing data is one of the most important part of analytics. The Steps for Data Preprocessing before analysis we performed are:

1. **Missing Values:** There were missing values in the dataset. We used the Median and the mode of the corresponding dataframe column for Imputation

   The Mode:

   The variables below being categorical values were replaced by the mode value.

   - Dependents
   - Credit History
   - Gender
   - Married
   - Education
   - Self_Employed
   - Loan_Amount_Term

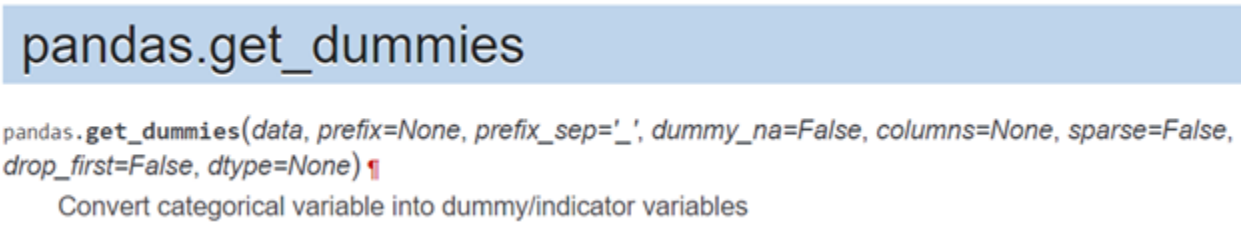
   The Median:

   - The Loan Amount term was replaced by the Median value because of its independence to any present outliers.
   - We created 2 subsets of Loan amount with Education variable that had 2 categorical variables.
   - Each of the Subset was replaced by the Loan amount was replaced by its median value.
   - We also Combined the two variables co-applicant income and applicant income to form a new column Total Income to replace the two in our dataset. This was done because the co-applicant income had some zero values, meaning that the co-applicant had no source of income.

2. **Categorical Variables:** The categorical variable was transformed to Dummy variables.

   - Property Area
   - Dependents
   - Gender
   - Married
   - Education
   - Self Employed
   - Credit History
   - Loan Status

This was done using pandas library in Python using get_dummies



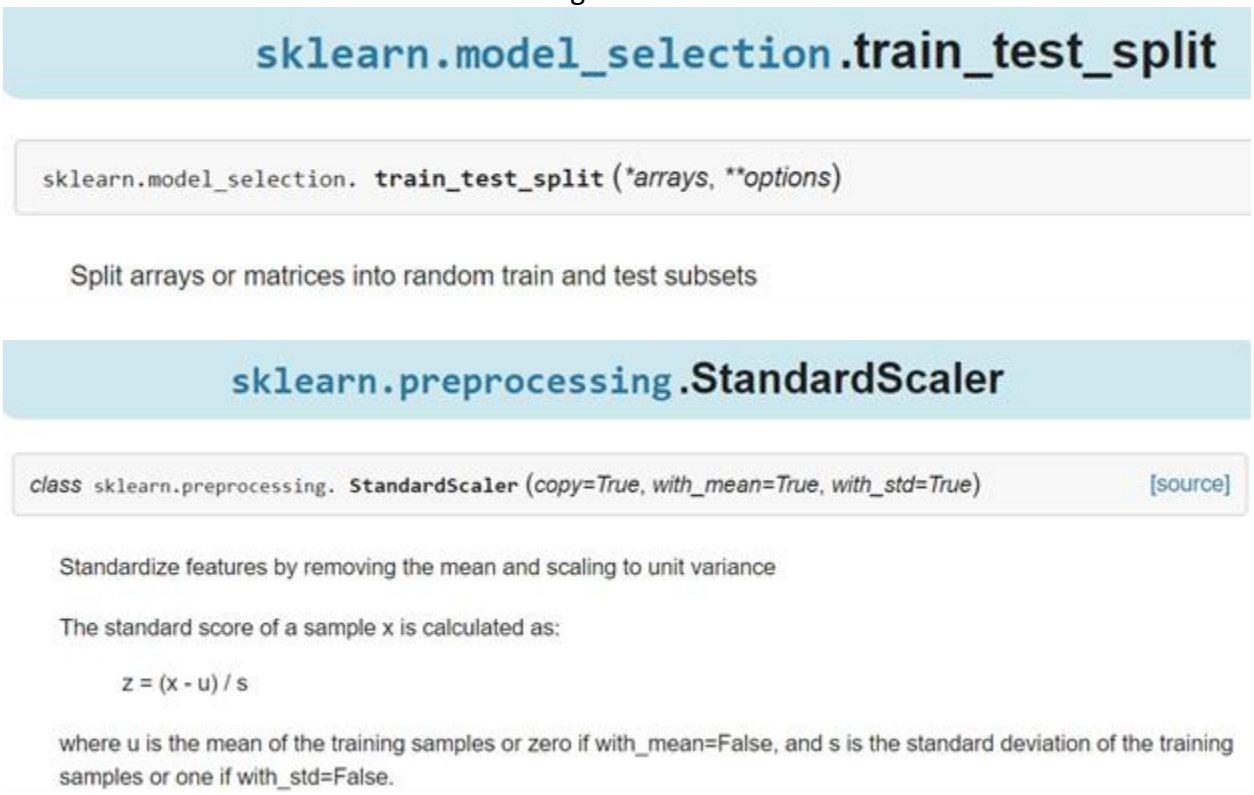
pandas.get_dummies

pandas.**get_dummies**(*data, prefix=None, prefix_sep='_', dummy_na=False, columns=None, sparse=False, drop_first=False, dtype=None*) ¶

Convert categorical variable into dummy/indicator variables

Fig: Description of Pandas.get_dummies

3. **Splitting Data into Training and Test sets:** The dataset is split into training and test sets using Python's Sklearn library 'train_test_split' in a 80:20 ratio respectively (Eremenko, de Ponteves, Machine Learning A-Z: Hands-On Python & R in Data Science, www.udemy.com). Accordingly, we get a training set consisting of 491 observations and a test set consisting of 123 observations.



sklearn.model_selection.train_test_split

sklearn.model_selection. **train_test_split** (**arrays, **options*)

Split arrays or matrices into random train and test subsets

sklearn.preprocessing.StandardScaler

class sklearn.preprocessing. **StandardScaler** (*copy=True, with_mean=True, with_std=True*) [source]

Standardize features by removing the mean and scaling to unit variance

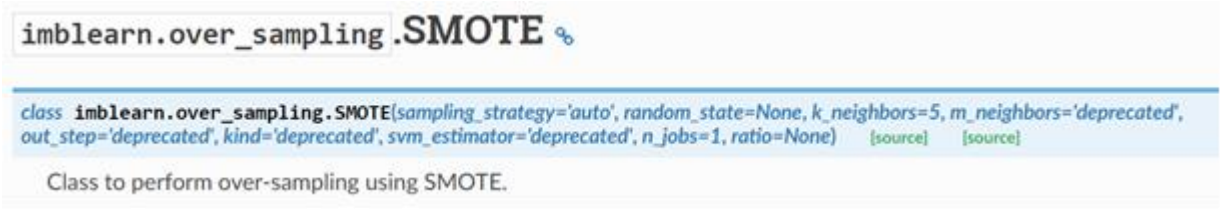
The standard score of a sample x is calculated as:

z = (x - u) / s

where u is the mean of the training samples or zero if with_mean=False, and s is the standard deviation of the training samples or one if with_std=False.

4. Standardizing Data: Data standardization is the critical process of bringing data into a common format that allows for collaborative research, large-scale analytics, and sharing of sophisticated tools and methodologies. A standardized variable (sometimes called a z-score or a standard score) is a variable that has been rescaled to have a mean of zero and a standard deviation of one. We standardized variables Loan Amount, Total Income and Loan Amount Term. The standardization was done using Python's Sklearn library 'StandardScaler'.

5. Handling Imbalanced Data by Over-sampling: Machine learning algorithms have trouble learning when one class dominates the other. In our case the Loan Status that our target variable was highly imbalanced. We used SMOTE (Synthetic Minority Oversampling Technique) algorithm to solve the Data Imbalance problem. This is a statistical technique for increasing the number of cases in your dataset in a balanced way. The module works by generating new instances from existing minority cases that you supply as input. This implementation of SMOTE does not change the number of majority cases. The new instances are not just copying existing minority cases; instead, the algorithm takes samples of the feature space for each target class and its nearest neighbors, and generates new examples that combine features of the target case with features of its neighbors. This approach increases the features available to each class and makes the samples more general. SMOTE takes the entire dataset as an input, but it increases the percentage of only the minority cases.



This was done using Python's Imblearn Library.

DIMENSIONALITY REDUCTION METHODS

In machine learning classification problems, there are at times a large number of variables or factors based on which the final classification is to be done. Most of these features are often corelated and hence, redundant and hence dimensionality reduction algorithms are used to reduce the number of features considered, by obtaining a set of principal variables (Uberoi, Introduction to Dimensionality Reduction, Geeks for Geeks A computer science portal for geeks, www.geeksforgeeks.org). We performed Principal Component Analysis on our Dataset.

Principal Component Analysis:

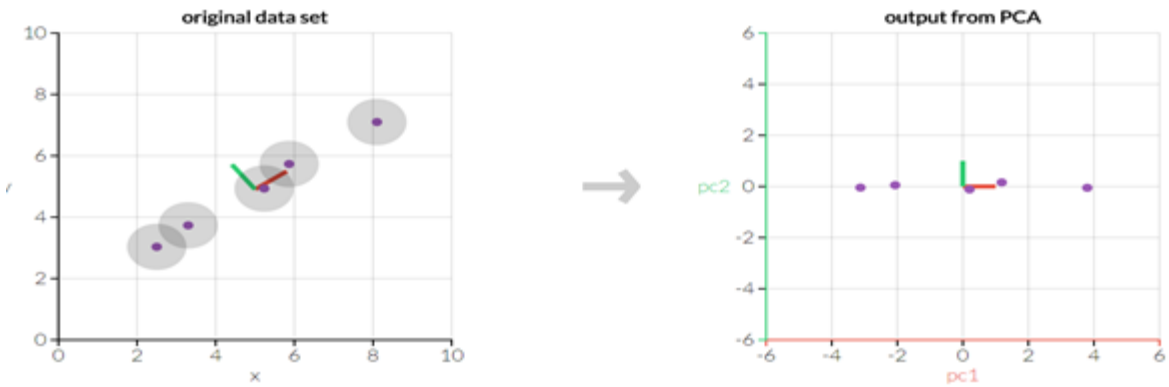
Principal Component Analysis (PCA) is used for getting simpler representation of a set of intercorrelated variables (Afifi et al. Practical Multivariate Analysis, Fifth Edition). Variables are not treated as dependent and independent variables, all variables are treated equally, and the original variables are transformed into new, uncorrelated variables called the Principal Components (Afifi et al. Practical Multivariate Analysis, Fifth Edition). These Principal Components are linear combinations of the original variables and each linear combination corresponds to a principal component.

Understanding PCA becomes very easy if we try to understand it in 2 dimensions or with 2 variables. The idea of PCA is to create 2 new components C_1 & C_2 as a linear combination of two variables x_1 and x_2 (Afifi et al. Practical Multivariate Analysis, Fifth Edition). The equation of principal components of only two variables can be written as seen in the first figure below. This concept can be better understood from a visual representation of the same idea that is shown below.

Equations for Principal Components of 2 variables

$$C_1 = a_{11}x_1 + a_{12}x_2$$

$$C_2 = a_{21}x_1 + a_{22}x_2$$



(Afifi et al. Practical Multivariate Analysis, Fifth Edition, Pg. 360) Visual 2D example for first Principal Component

(Powell, Lehe, Principal Component Analysis Explained Visually, setosa.io)

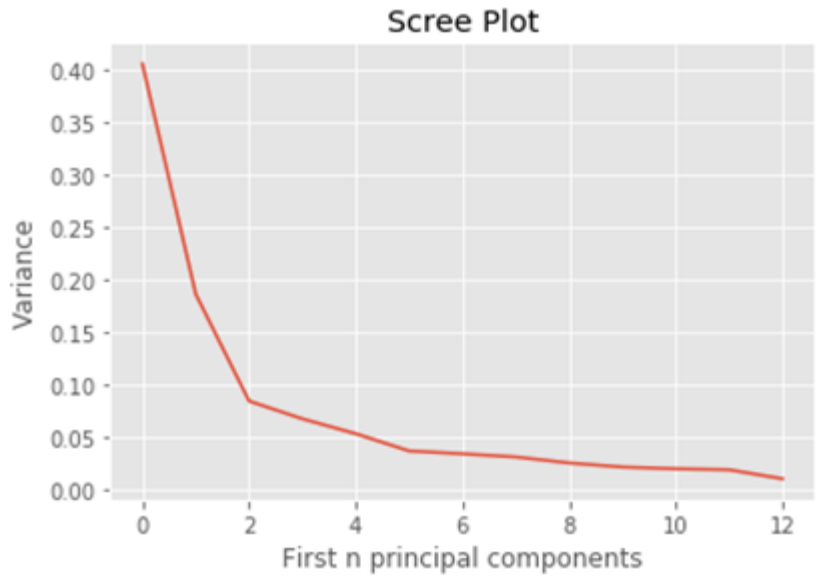
We have seen an example of PCA in two-dimensions above but, it is necessary to understand that PCA remains equally effective even if the dimensionality is increased greatly. The first linear component is the linear combination of variables that has the maximum variance among other principal components and it accounts for the most variance in the data (11.1 Principal Component Analysis (PCA) Procedure, STAT 505 Applied Multivariate Statistical Analysis, onlinecourses.science.psu.edu). The second principal component explains a slightly lesser variance as compared to the first principal component and subsequently the amount of variance explained by principal components goes on decreasing.

Standardization before performing PCA:

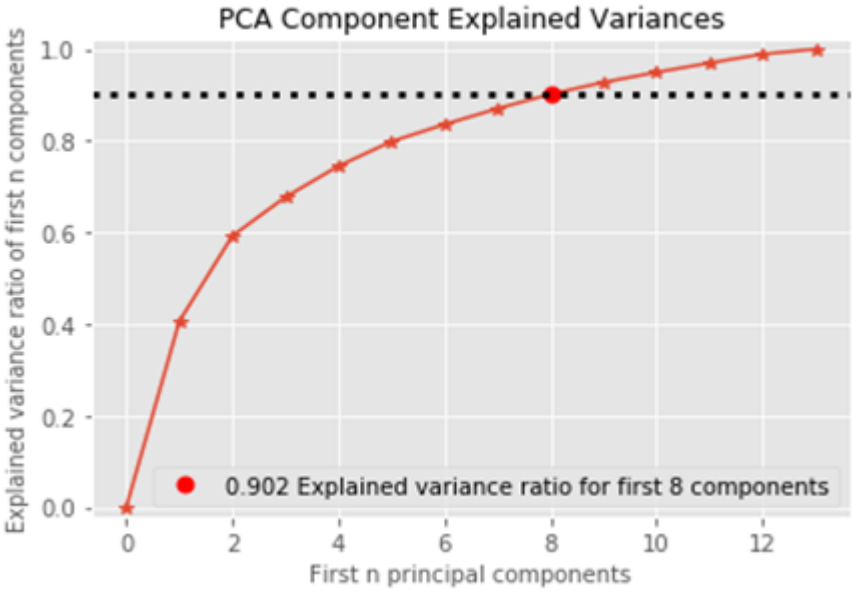
It is important to standardize the predictors before performing Principal Component Analysis to make sure that all the predictors are on the same scale as performing PCA on un-standardized variables will lead to large loadings for variables with large variance, thus creating dependence of Principal Components on the high variance variables (https://www.theanalysisfactor.com/tips-principal-component-analysis/). We have made sure that we have standardized all the predictors before performing Principal Component Analysis.

PCA Results and Analysis:

Dimension Reduction avoids over fitting and multicollinearity of data that could affect our classification models. Hence, we decided to perform PCA. After performing PCA we reduced our 13 variables to 8 principal components. Refer the figures below to understand the results that we obtained after performing PCA.



Scree plot of Proportion of variance vs Principal Components



Scree plot of Cumulative variances vs Principal Components

From the figures above which are the scree plots we obtained after performing PCA we interpreted that the traditional methods for selecting the Principal components with most variance like the 'Elbow Method' were not suitable for our results. Instead, we chose the first **8** Principal Components that cumulatively explained about **90.02%** of the total variance.

CLASSIFICATION METHODS

In Machine learning and statistics, classification methods are used for identifying which set of categories or populations does a new observation belong to and are considered as Supervised Learning techniques as a training set of correctly identified observations is available (Statistical Classification, Wikipedia, https://en.wikipedia.org/wiki/Statistical_classification). Classification algorithms are used when some decision or forecast is made on the basis of presently available information and these methods can be used repeatedly to make decisions in new situations (Sagar S. Nikam, A Comparative Study of Classification Techniques in Data Mining Algorithms, Oriental Journal of Classification Techniques in Data Mining Algorithms, www.computerscijournal.org). We have basically used four classification algorithms namely, Logistic Regression, Naïve Bayes algorithm, K-Nearest Neighbors algorithm and Support Vector Machine.

7.1 Logistic Regression algorithm and Results

Logistic regression can be used whenever an individual is to be classified into one of two populations. When there are more than two groups, what is called polychotomous or generalized logistic regression analysis can be used. In the past, most of the applications of logistic regression were in the medical field, but it is also frequently used in epidemiologic research. It has been used, for example, to calculate the risk of developing heart disease as a function of certain personal and behavioral characteristics such as age, weight, blood pressure, cholesterol level, and smoking history. (Afifi et al., Practical Multivariate Analysis, Fifth Edition)

Logistic regression analysis requires knowledge of both the dependent (or outcome) variable and the independent (or predictor) variables in the sample being analyzed. The results can be used in future classification when only the predictor variables are known, like the results in discriminant function analysis. (Afifi et al. Practical Multivariate Analysis, Fifth Edition)

The logistic function has the form as seen in Fig. 7.1:

$$Pz = \frac{e^{\Lambda(\alpha+\beta 1X1+\beta 2X2+\cdots+\beta pXp)}}{1+e^{\Lambda(\alpha+\beta 1X1+\beta 2X2+\cdots+\beta pXp)}}$$

Fig: Equation for Logistic Regression

(Afifi et al. Practical Multivariate Analysis, Fifth Edition, Pg. 272)

This equation is called the **Logistic Regression Equation**, where Z is the linear function α +β1X1 +···+βPXP. It may be transformed to produce a new interpretation. Specifically, we define the **Odds** as shown in the below figure:

$$Odds = Pz / 1-P$$

Fig: Odds in Equation for Logistic Regression

(Afifi et al. Practical Multivariate Analysis, Fifth Edition, Pg. 272)

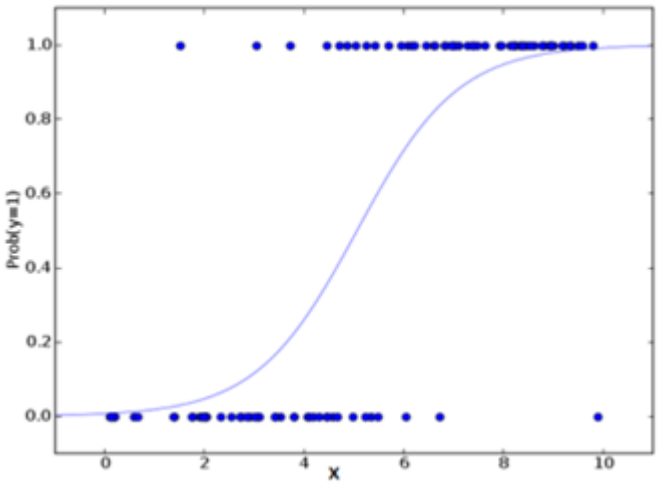
or in terms of **Pz**,

$$Pz = \frac{Odds}{1+ Odds}$$

Fig: Odds in terms of Pz Equation for Logistic Regression

(Afifi et al. Practical Multivariate Analysis, Fifth Edition, Pg. 272)

Logistic regression always produces probabilities that are more than 0 and less than 1. A typical model plot of logistic regression is shown below:

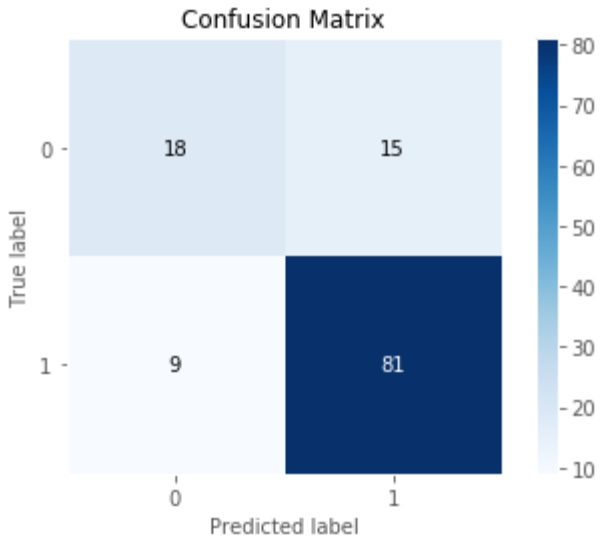


A typical logistic model plot

Source:https://www.analyticsvidhya.com/wp-content/uploads/2015/11/plot.png

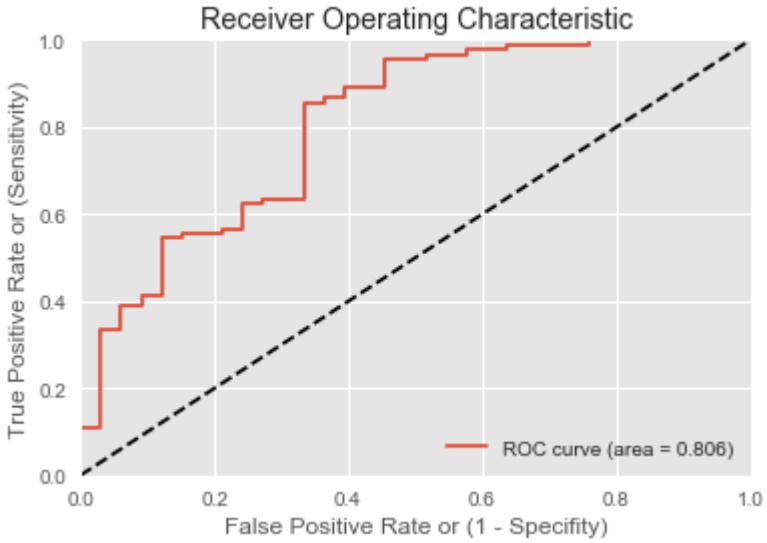
Results and Analysis:

Taking into account the 8 Principal Components we performed Logistic Regression. Upon performing Logistic regression, the accuracy achieved was 80.48%. The confusion matrix is as seen in Fig. 7.5:



Confusion Matrix of Logistic Regression

The ROC(receiver Operating Characteristic) curve we obtained after performing logistic regression gave us AOC(area under the curve) as 0.806



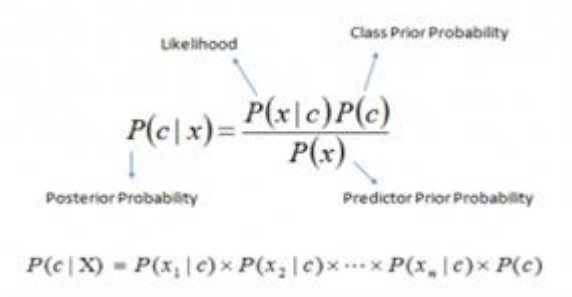
Curve plot of Logistic regression

7.2 Naïve Bayes Algorithm and Results

Naïve Bayes Algorithm is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a feature in a class is unrelated to the presence of any other feature. For example, a fruit may be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naïve'.

Source: Sunil Ray, September 11, 2017, 6 Easy Steps to Learn Naïve Bayes Algorithms (with codes in Python and R) https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

Naïve Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naïve Bayes is known to outperform even highly sophisticated classification methods. Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



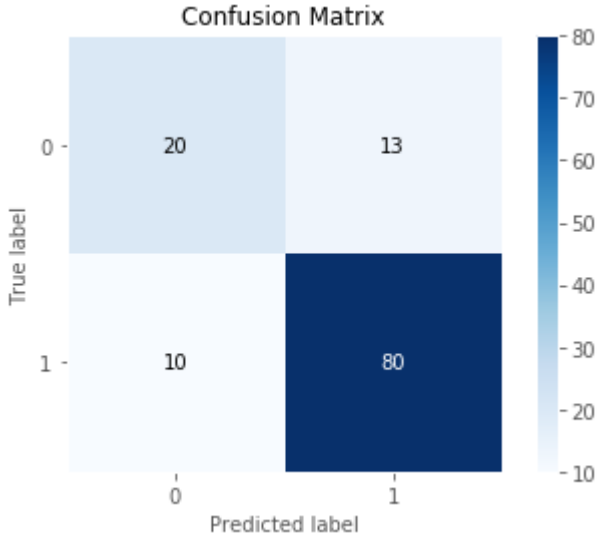
Naïve Bayes equation

- P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
- P(c) is the prior probability of class and P(x) is the prior probability of predictor.
- P(x|c) is the likelihood which is the probability of predictor given class.

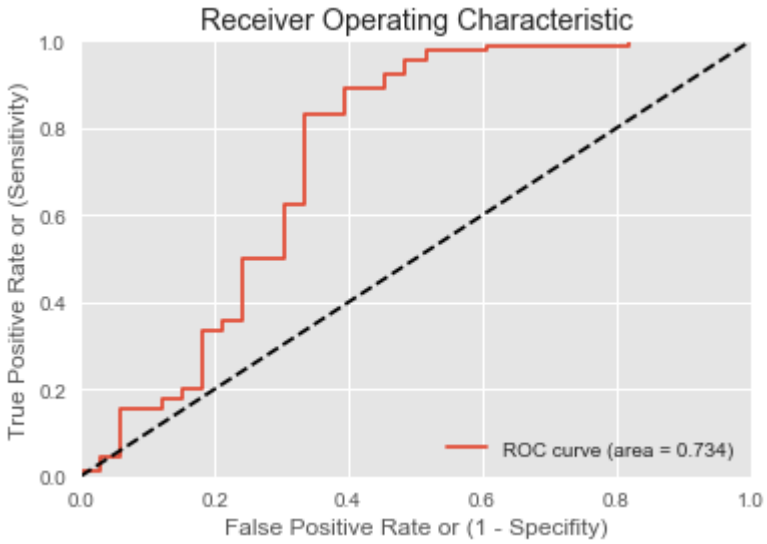
Source: Sunil Ray, September 11, 2017, 6 Easy Steps to Learn Naïve Bayes Algorithms (with codes in Python and R) https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

Results and Analysis:

Upon performing Naïve Bayes Algorithm for our dataset, we achived an accuracy of 81.30%. Notably, this accuracy percentage is quite close to what we achived using Logistic Regression Algorithm. The confusion matrix is as follows:



Confusion Matrix of Naïve Bayes Algorithm



The ROC(receiver Operating Characteristic) curve we obtained after performing Naïve Bayes gave us AOC(area under the curve) as 0.734

7.3 k-Nearest Neighbors and Results

The KNN or *k*-nearest neighbors algorithm is one of the simplest machine learning algorithms and is an example of instance-based learning, where new data are classified based on stored, labeled instances. More specifically, the distance between the stored data and the new instance is calculated by means of a similarity measure. This similarity measure is typically expressed by a distance measure such as the Euclidean distance, cosine similarity or the Manhattan distance.

Source: https://www.datacamp.com/community/tutorials/machine-learning-in-r

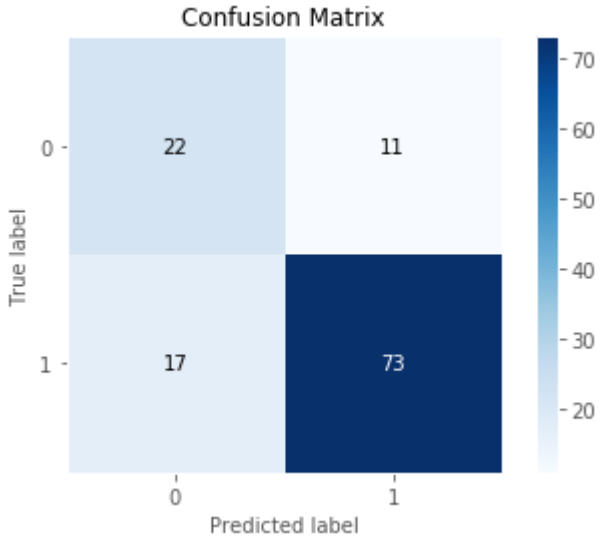
The *k*-nearest neighbor algorithm does that after the distance of the new point to all stored data points has been calculated, the distance values are sorted, and the *k*-nearest neighbors are determined. The labels of these neighbors are gathered, and a majority vote or weighted vote is used for classification or regression purposes.

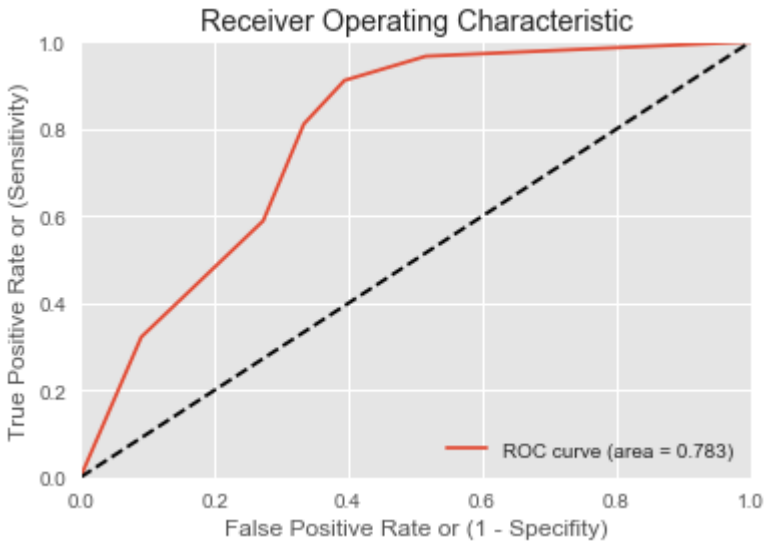
Source: https://www.datacamp.com/community/tutorials/machine-learning-in-r

Results and Analysis:

We performed k-nearest neighbors algorithm on our dataset and we calculated the accuracy of the algorithm using the confusion matrix seen in the figure below. The value of k was determined to be as 5. The accuracy achieved was 77.23%. Notably, once again the accuracy achieved here was quite close to the accuracies achieved using both Logistic regression and Naïve Bayes. The confusion matrix is as follows:

Confusion Matrix of k Nearest Neighbors for k = 5

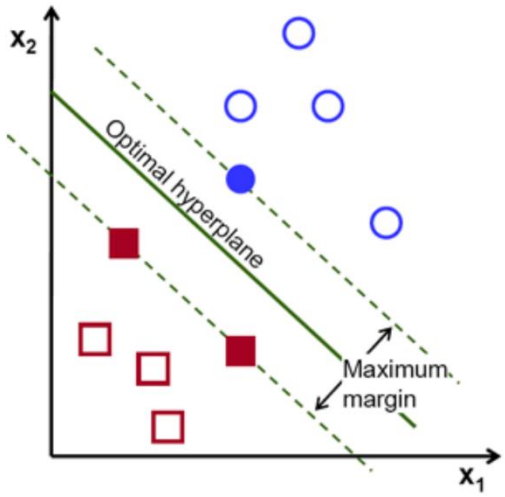




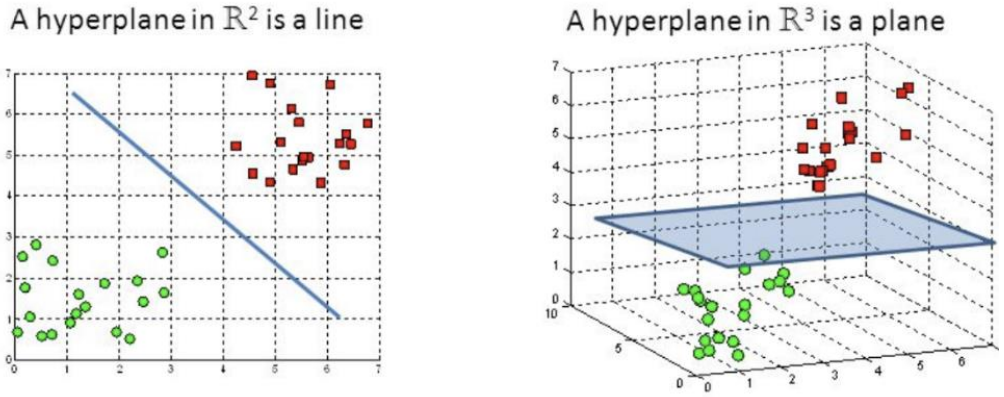
The ROC(receiver Operating Characteristic) curve we obtained after performing k-nearest neighbors gave us AUC(area under the curve) as 0.783

7.4 Support Vector Machine and Results

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimentional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.



To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.



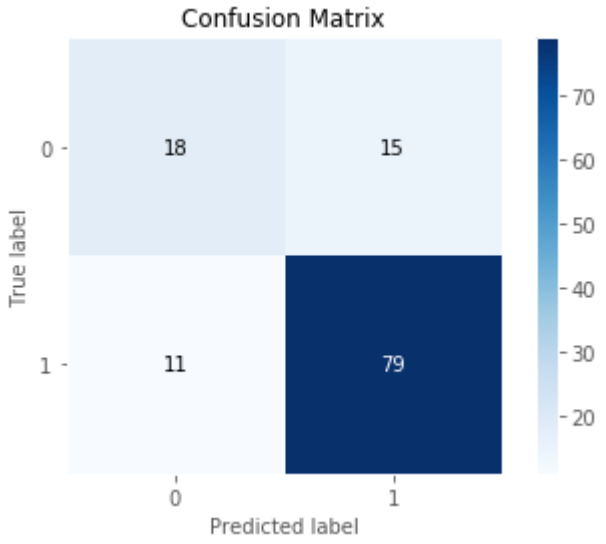
Hyperplanes in 2D and 3D feature space

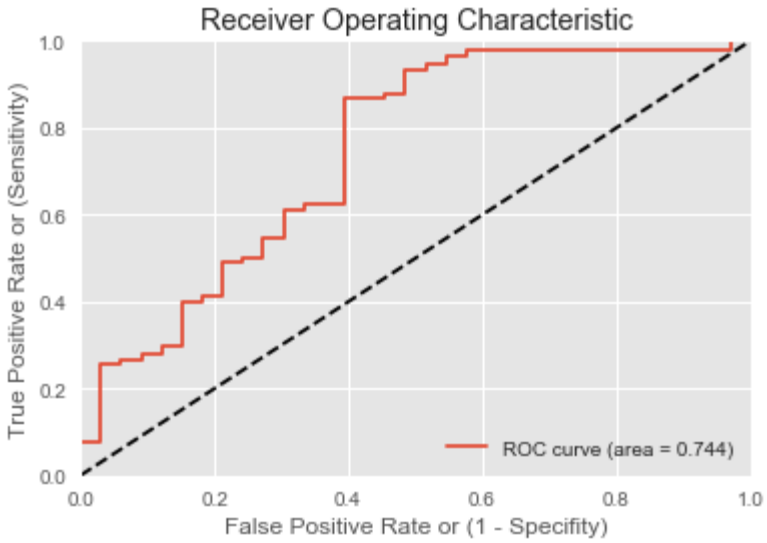
Hyperplanes are decision boundaries that help classify the data points. Data points falling on either side of the hyperplane can be attributed to different classes. Also, the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane.

Results and Analysis:

We performed Support Vector Machine algorithm on our dataset and we calculated the accuracy of the algorithm using the confusion matrix seen in the figure below. The accuracy we achived was 78.86%. Notably, once again the accuracy achieved here was quite close to the accuracies achieved using both Logistic regression, Naïve Bayes and K-nearest Neighbours. The confusion matrix is as follows

Confusion Matrix of SVM





The ROC(Receiver Operating Characteristic) curve we obtained after performing SVM gave us AOC(area under the curve) as 0.744

CONCLUSION

In this section we briefly present all the conclusions, learnings and insights that we have gained after performing all the analyses.
Following are the major conclusions and insights-

- Logistic Regression gives False Positives (FP) as 15, in Naive Bayes FP's are 13 and in kNN FP's are 11. From this, we can infer that although Naive Bayes displays higher accuracy, kNNs display a lower FP rate, very similar accuracy and a good AUC score. Our primary goal is to have reduced FP rates along with high accuracy so as to prevent the number of undeserved loans being sanctioned and hence from the banks perspective, kNN model proves to be the optimum solution here.

- The accuracy showcased by all the models lie in the similar bracket, i.e., 77-82%. According to our problem statement, our key goal is to reduced the FP rates so as to reduce the risk incurred by the bank.

- Moreover, the AUC score of kNN classifier is in the top 2 and very close to the AUC of Logistic Regression which is at the top. KNN has an AUC score of 0.783 which means that there is a 78.3% probability of correct classification of Yes and No which is very good.

FUTURE SCOPE

The current analysis stopped at using four classification techniques for prediction. Although we attained a high accuracy and AUC, there are other algorithms like Decision Trees and Random Forest which can also be applied in order to improve the classifier. Another point for future work would be to do a k-fold cross validation to decrease chances of bias and improve the model. Since this dataset was limited to 614 observations there wasn't enough data to create a generalized model which can be deployed to production. So, another scope for future analysis would be to source more applicant data and add it to the current dataset. This dataset can then be used to train a classifier using Ensemble Learning methods to obtain a better classifier. This model can then help the decision makers to reinforce their decision making through data driven insights.

REFERENCES

- Abdelmonem Afififi, Susanne May, Virginia A. Clark, Practical Multivariate Analysis, Fifth Edition
- Analytics Vidhya Content Team, Practical Guide to Principal Component Analysis (PCA) in R & Python, www.analyticsvidhya.com
- Victor Powell, Lewis Lehe, Principal Component Analysis Explained Visually, setosa.io, http://setosa.io/ev/principal-component-analysis/
- 11.1 Principal Component Analysis (PCA) Procedure, STAT 505 Applied Multivariate Statistical Analysis, onlinecourses.science.psu.edu, https://onlinecourses.science.psu.edu/stat505/node/51
- Anannya Uberoi, Introduction to Dimensionality Reduction, Geeks for Geeks A computer science portal for geeks, http://www.geeksforgeeks.org/dimensionality-reduction/
- Kirill Eremenko, Hadelin de Ponteves, Machine Learning A-Z: Hands-On Python & R in Data Science, www.udemy.com
- Statistical Classification, Wikipedia, https://en.wikipedia.org/wiki/Statistical_classification
- Sagar S. Nikam, A Comparative Study of Classification Techniques in Data Mining Algorithms, Oriental Journal of Classification Techniques in Data Mining Algorithms, http://www.computerscijournal.org/vol8no1/a-comparative-study-of-classification-techniques-in-data-mining-algorithms/
- Sunil Ray, September 11, 2017, 6 Easy Steps to Learn Naïve Bayes Algorithms (with codes in Python and R https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/
- https://stackoverflow.com/