ADMISSION PREDICTION

PROJECT 1

Submitted by :-

1. Shivam Suri (T.I.E.T)

2. Himanshu Mahajan (T.I.E.T)

3. Gowtham Devangam(SRIT)

4. Mudigonda Sri Harsha (VJIT)

Submitted to :-

Gurvansh Singh

M.Tech

Knowledge Solutions India

Knowledge Solutions India

Skill development | Certification | Placement prep

TABLE OF CONTENTS

TABLE OF GRAPHS

ABSTRACT

This project is about prediction of Graduate Admissions from an Indian perspective. We must predict the chances of a graduate student to take an admission in a college based on the various factors like GRE Score, TOFEL Score, etc. which were provided in the dataset.

We have used Multiple Linear Regressor and Random Forest Regressor to predict the chances of admission. We have predicted the chance of admission with MSE (Mean Squared Error), RMSE (Root Mean Squared Error) and r2_score.

We have also plotted the graphs for actual and predicted values for all four models. While using PCA, we plotted a graph between most significant features against the predicted values.

By applying the knowledge, we gained through the training and some inference from the various available resources, we collectively as a team made this project successful.

INTRODUCTION

In this project, we worked with Multiple Linear Regression and Random Forest Regression without Principle Component Analysis and with Principal Component Analysis.

Multiple Linear Regression, also known simply as multiple regression. The goal of multiple linear regression is to model the linear relationship between the independent variables and dependent variable. By taking the standard assumptions of Multiple Linear Regression, we worked out two models for MLR, with PCA and without PCA.

Random forest is a Supervised Learning algorithm which uses ensemble learning method for classification and regression. In this project we have worked with Random Forest Regression.

It is a bagging technique and not a boosting technique. The trees in random forests are run in parallel. There is no interaction between these trees while building the trees.

We have used libraries in python like Numpy for support in high-dimensional arrays and matrices. Pandas for handling the data provided, SkLearn for various operations like feature scaling, performing PCA, building and training models, calculation of metrics.

We visualized the data with the help of another library in python named Matplotlib. Using this, we have plotted different types of plots like scatter plot, correlation matrix. With the Seaborn library we have added additional features to the plots we have made.

Finally, we have concluded this project by saying which model is better for the given data, to predict the chance of admission of a graduate into a college from an Indian perspective.

SOFTWARE LIBRARIES

NUMPY:

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

It also has functions for working in domain of linear algebra, Fourier transform, and matrices.

NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.

NumPy stands for Numerical Python.

Why use Numpy:

In Python we have lists that serve the purpose of arrays, but they are slow to process.

NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

PANDAS:

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

Tabular data with heterogeneously typed columns, as in an SQL table or Excel spreadsheet

Ordered and unordered (not necessarily fixed frequency) time series data.

Why use Pandas:

- Easy handling of missing data (represented as NaN) in floating point as well as non-floating-point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects.

MATPLOTLIB:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Basic Plots:

- Line plot
- Bar plot
- Histogram
- Scatter Plot

SEABORN:

Seaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and closely integrated with pandas data structures.

Here is some of the functionality that seaborn offers:

- A dataset-oriented API for examining relationships between multiple variables
- Specialized support for using categorical variables to show observations or aggregate statistics
- Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data
- Automatic estimation and plotting of linear regression models for different kinds dependent variables
- Concise control over matplotlib figure styling with several built-in themes
- Tools for choosing color palettes that faithfully reveal patterns in your data

SKLEARN:

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis

Some popular groups of models provided by scikit-learn include:

- **Clustering**: for grouping unlabelled data such as KMeans.
- **Cross Validation**: for estimating the performance of supervised models on unseen data.
- **Datasets**: for test datasets and for generating datasets with specific properties for investigating model behaviour.
- **Dimensionality Reduction**: for reducing the number of attributes in data for summarization, visualization, and feature selection such as Principal component analysis.
- **Ensemble methods**: for combining the predictions of multiple supervised models.
- **Feature extraction**: for defining attributes in image and text data.
- **Feature selection**: for identifying meaningful attributes from which to create supervised models.
- **Parameter Tuning**: for getting the most out of supervised models.
- **Manifold Learning**: For summarizing and depicting complex multi-dimensional data.
- **Supervised Models**: a vast array not limited to generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

EDA

Checking Missing Values:

Pandas provides isnull() function to detect missing values, df.isna().sum() returns us the number of missing values in each column.

```
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
```

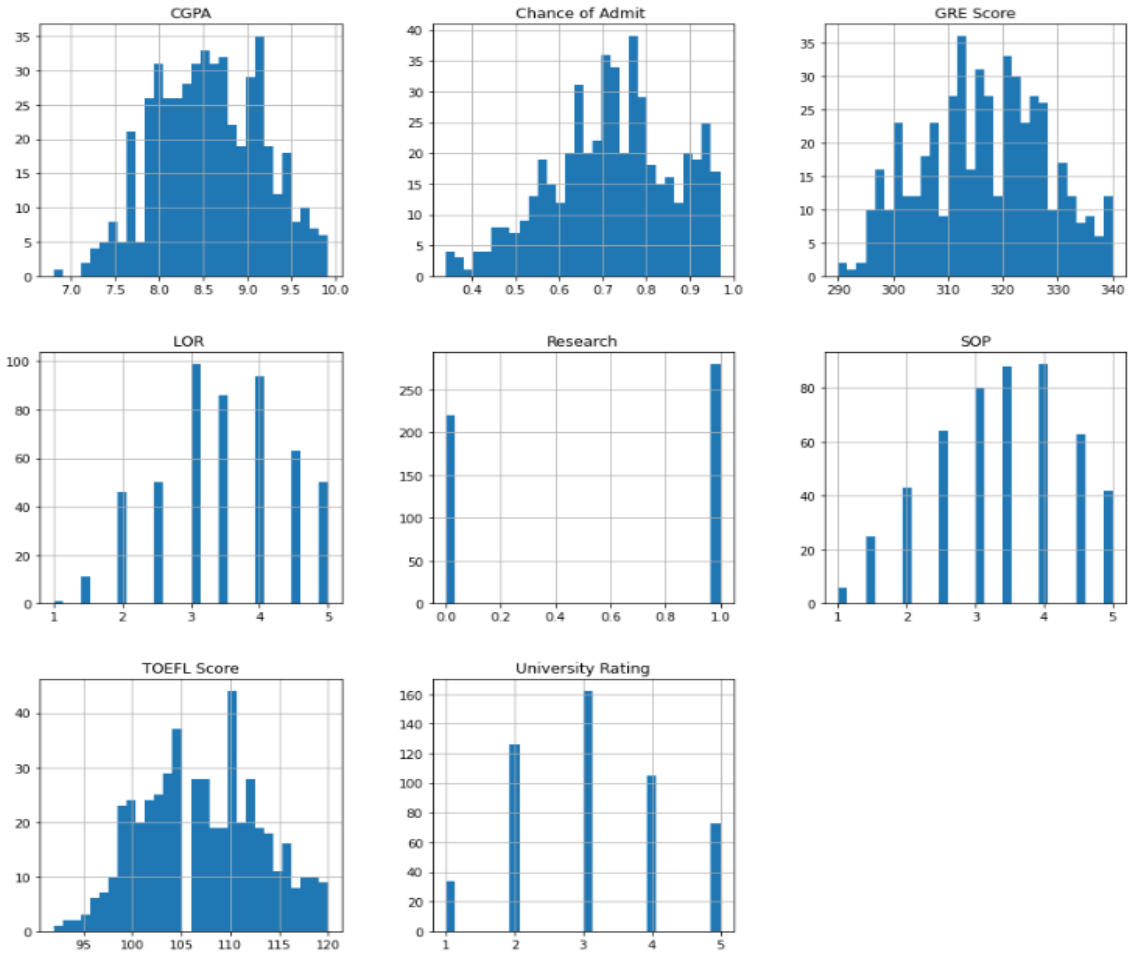
Descriptive Statistics:

The .describe() function is used to generate descriptive statistics that summarize the central tendency, dispersion and shape of a dataset's distribution, excluding NaN values.

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	500.000000	500.000000	500.000000	500.000000	500.00000	500.000000	500.000000	500.00000
mean	316.472000	107.192000	3.114000	3.374000	3.48400	8.576440	0.560000	0.72174
std	11.295148	6.081868	1.143512	0.991004	0.92545	0.604813	0.496884	0.14114
min	290.000000	92.000000	1.000000	1.000000	1.00000	6.800000	0.000000	0.34000
25%	308.000000	103.000000	2.000000	2.500000	3.00000	8.127500	0.000000	0.63000
50%	317.000000	107.000000	3.000000	3.500000	3.50000	8.560000	1.000000	0.72000
75%	325.000000	112.000000	4.000000	4.000000	4.00000	9.040000	1.000000	0.82000
max	340.000000	120.000000	5.000000	5.000000	5.00000	9.920000	1.000000	0.97000

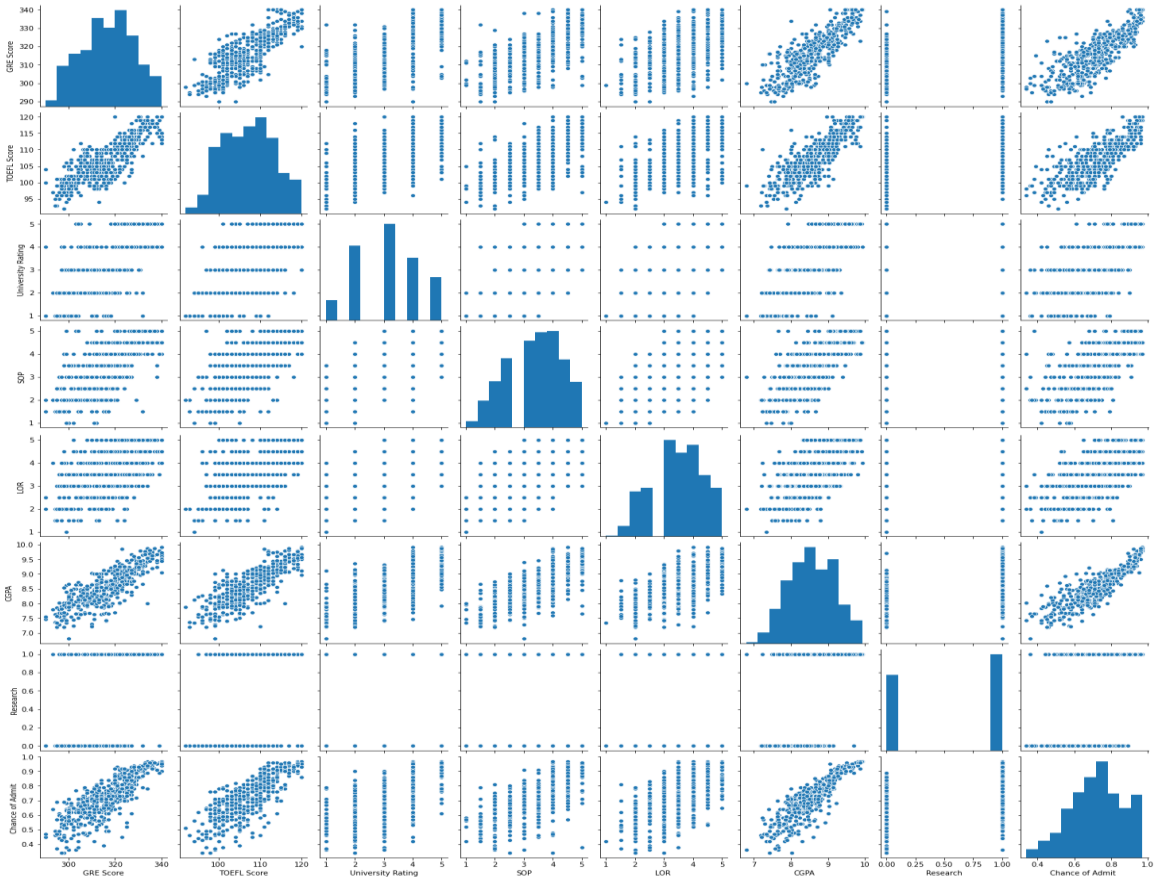
Visualizing Distribution of data:

Plotting a histogram is definitely a very convenient way to visualize the distribution of your data. The .hist() function plots a histogram for each column in your dataframe that has numerical values in it. It does the counting and automatically groups the data into bins.



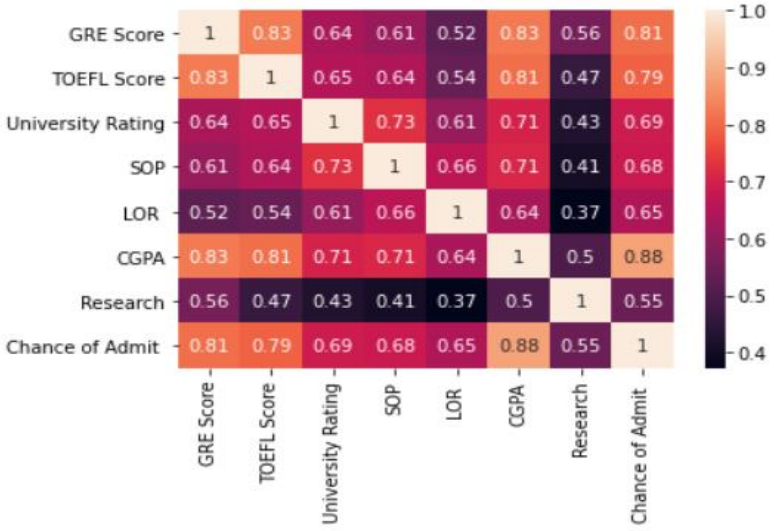
Correlation Graphs:

The .pairplot() function plots a pairwise relationships in a dataset. It creates a grid of Axes such that each variable in data will by shared in the y-axis across a single row and in the x-axis across a single column.



Correlation Matrix:

Seaborn provides .heatmap() function for creating correlation matrix. A Heatmap is a graphical representation of 2D data. Each data value represents in a matrix and it has a special color. The color of the matrix is dependent on value. Normally, low-value show in low-intensity color and high-value show in hight-intensity color format. The main intention of Seaborn heatmap is to visualize the correlation matrix of data for feature selection.



DATA PREPROCESSING

Feature Scaling:

Feature scaling is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.

Need:

Since the range of values of raw data may vary widely, in some machine learning algorithms, objective functions will not work properly without normalization. For example, many classifiers calculate the distance between two points by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance.

Formula:

The standard score of a sample X can be calculated as:

$$\mathbf{Z = (X - U) / S}$$

where U is the mean of the Sample and s is the standard deviation of the Sample.

Principal Component Analysis(PCA):

Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. It is an un-supervised learning algorithm.

Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analysing data much easier and faster for machine learning algorithms without extraneous variables to process.

So basic idea behind PCA is to reduce the number of variables of a data set, while preserving as much information as possible.

Pre-Requisite:

In order to perform PCA, data is always standardized. The reason why it is critical to perform standardization prior to PCA, is that the latter is quite sensitive regarding the variances of the initial variables. That is, if there are large differences between the ranges of initial variables, those variables with larger ranges will dominate over those with small ranges which will lead to biased results.

Mathematics behind PCA:

1.Take the whole dataset consisting of all x columns(let's say there are d dimensions).

2.Compute the covariance matrix of the whole dataset.

It can be calculated as :

$$cov(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n} (Xi - \bar{x})(Yi - \bar{y})$$

3.Compute eigenvectors and the corresponding eigenvalues.

Now, we can easily compute eigenvalue and eigenvectors from the covariance matrix that we have above.

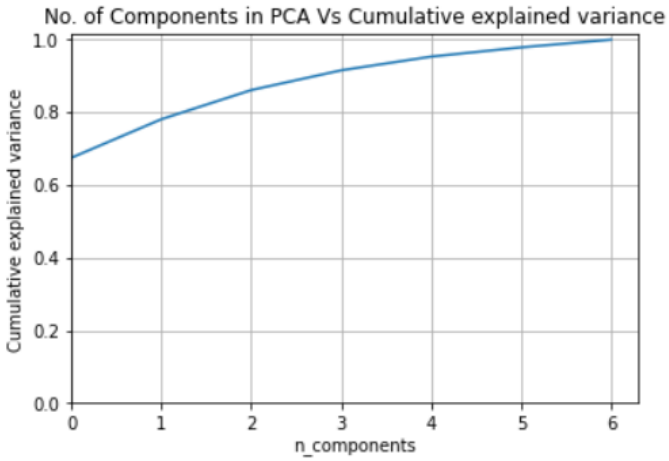
Let A be a square matrix, ν a vector and λ a scalar that satisfies Aν = λν, then λ is called eigenvalue associated with eigen vector ν of A. λ can be calculated using formula below:

$$\det(A - \lambda I) = 0$$

After finding λ, we can easily find eigen vector ν of A.

4.Sort the eigenvectors by decreasing eigenvalues and choose k eigenvectors with the largest eigenvalues to form a d × k dimensional matrix W.

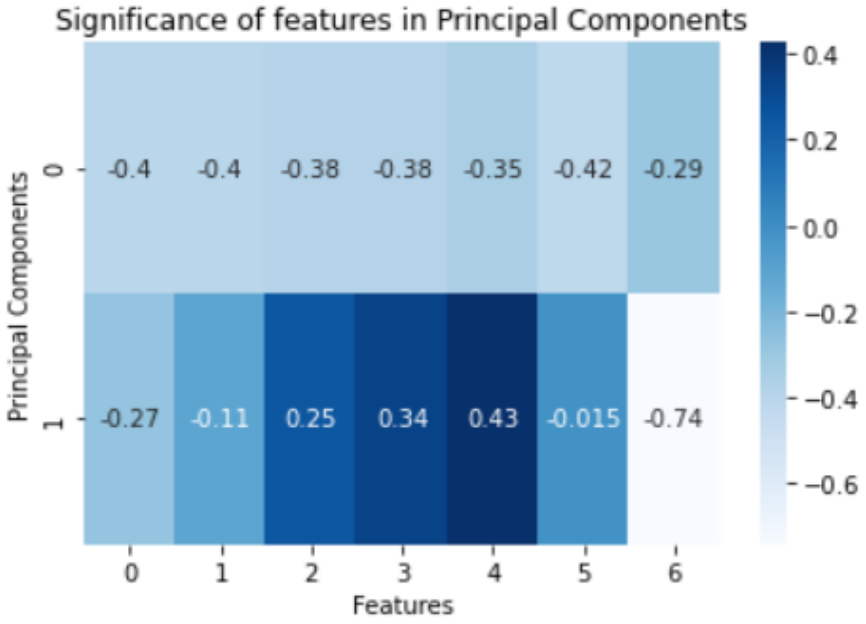
We can choose the number of components of PCA from the Cumulative explained variance graph. We can choose number of components which may explain sufficient amount of data.



From the above graph we can infer that index 1 of n_components i.e. n_components = 2 gives us about 78% of information of the dataset which is sufficient to train our model.

5.Use this d × k eigenvector matrix to transform the samples onto the new subspace

Significance of features in each Principal Component:



We can make a heat-plot to see how the features mixed up to create the components. Seaborn library is used to create such heatmaps. This heatmap shows the significance of various x features in components of PCA. Here negative values shows that their directions are reversed.

ALGORITHMS

MULTIPLE LINEAR REGRESSION:

Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression (MLR) is to model the linear relationship between the explanatory (independent) variables and response (dependent) variable.

Formula and Calculation of MLR:

$$y_i=\beta_0+\beta_1 x_{i1}+\beta_2 x_{i2}+...+\beta_p x_{ip}+\epsilon$$

where, for i=n observations:

y_i=dependent variable

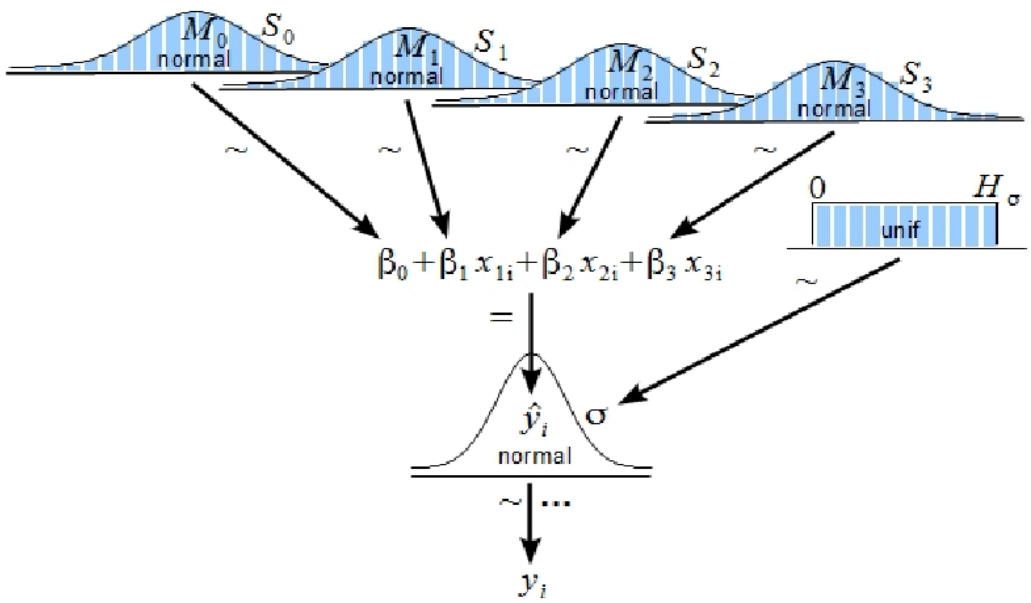
x_i=explanatory variables

β_0=y-intercept (constant term)

β_p =slope coefficients for each explanatory variable

ϵ=the model's error term (also known as the residuals)

Graphical Representation:



Assumptions:

The multiple regression model is based on the following assumptions:

- There is a linear relationship between the dependent variables and the independent variables.
- The independent variables are not too highly correlated with each other.
- y_i observations are selected independently and randomly from the population.
- Residuals should be normally distributed with a mean of 0 and variance σ.

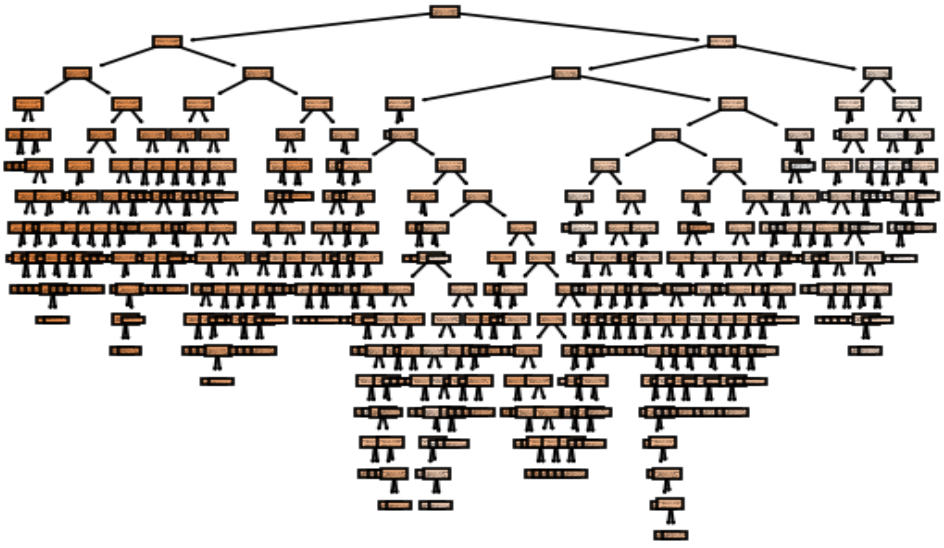
RANDOM FOREST REGRESSOR:

Random forest is a Supervised Learning algorithm which uses ensemble learning method for classification and regression.

Random forest is a bagging technique and not a boosting technique. The trees in random forests are run in parallel. There is no interaction between these trees while building the trees.It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

Graphical Representation:

A random forest is a meta-estimator (i.e. it combines the result of multiple predictions) which aggregates many decision trees. Ex. Below are the number of decision trees made having the index 9.



Advantages of Random Forest:

1. It is one of the most accurate learning algorithms available. For many data sets, it produces a **highly accurate classifier**.
2. It runs efficiently on large databases.
3. It can **handle thousands of input variables** without variable deletion.
4. It gives estimates of what variables that are important in the classification.
5. It generates an internal **unbiased estimate of the generalization error** as the forest building progresses.
6. It has an **effective method for estimating missing data** and maintains accuracy when a large proportion of the data are missing.

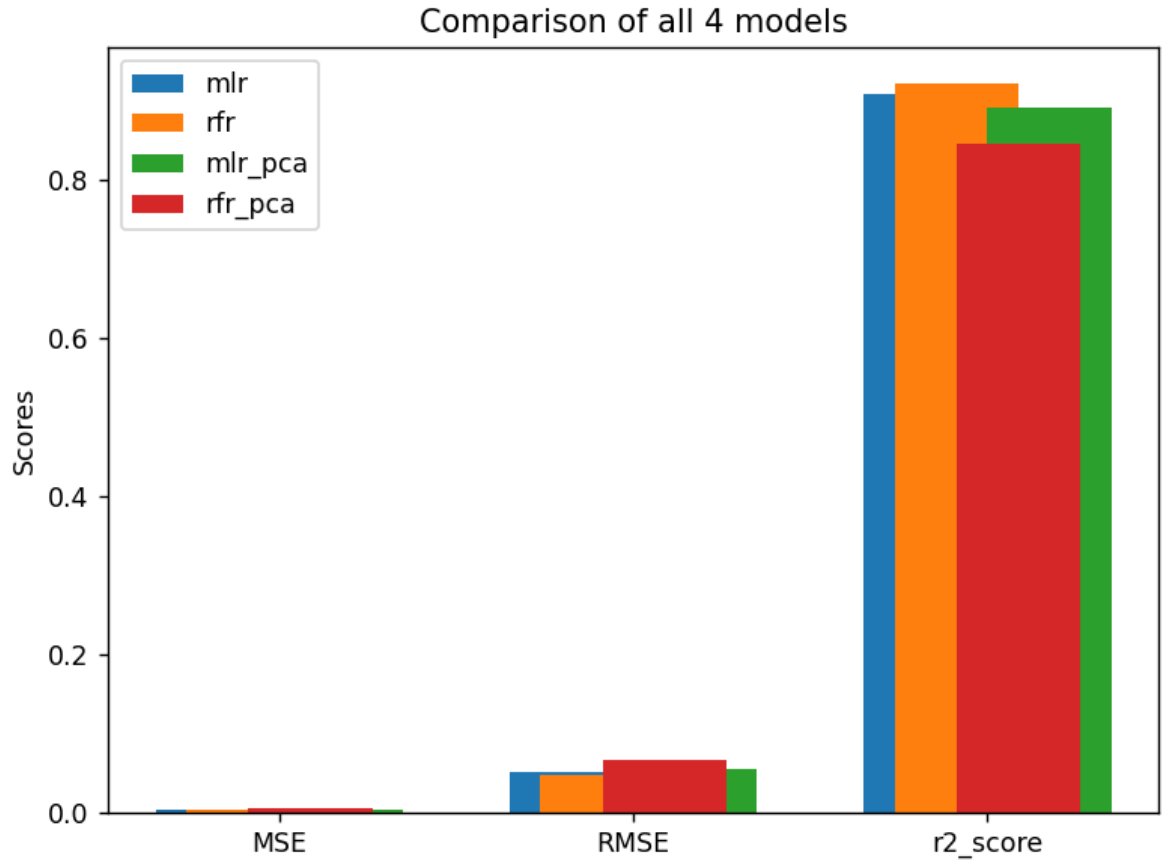
Disadvantages of Random Forest:

1. Random forests have been observed to **overfit for some datasets** with noisy classification/regression tasks.
2. For data including categorical variables with different number of levels, **random forests are biased in favour of those attributes with more levels**. Therefore, the variable importance scores from random forest are not reliable for this type of data.

CONCLUSION

This project is about prediction of Graduate Admissions from an Indian perspective. We must predict the chances of a graduate student to take an admission in a college based on the various factors like GRE Score, TOFEL Score, etc. which were provided in the dataset.

We have performed regression using Multiple Linear Regressor and Random Forest Regressor. We have also used PCA in the regressor models. To make the project more interesting and easily understandable, we have plotted various graphs like Correlation histograms, Correlation Matrix, to select n_components for PCA, significant features in PCA, actual values and predicted values.

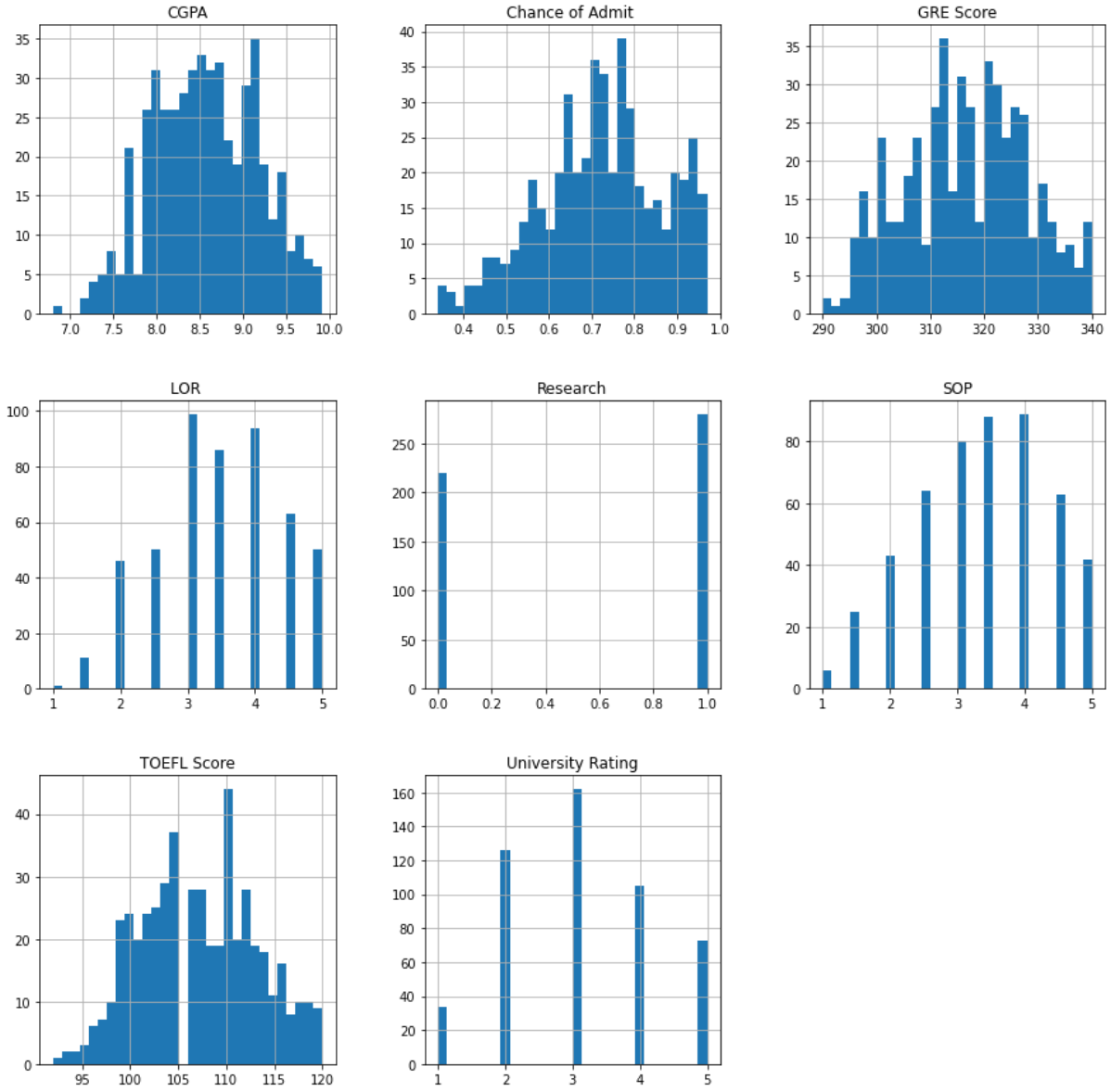


We observed that Random Forest Regressor without PCA outperforms all the other models based on the values obtained for MSE(Mean Squared Value), RMSE(Root Mean Squared Value), r2_Score of all the models, as it has minimum MSE and RMSE along with maximum r2_score.

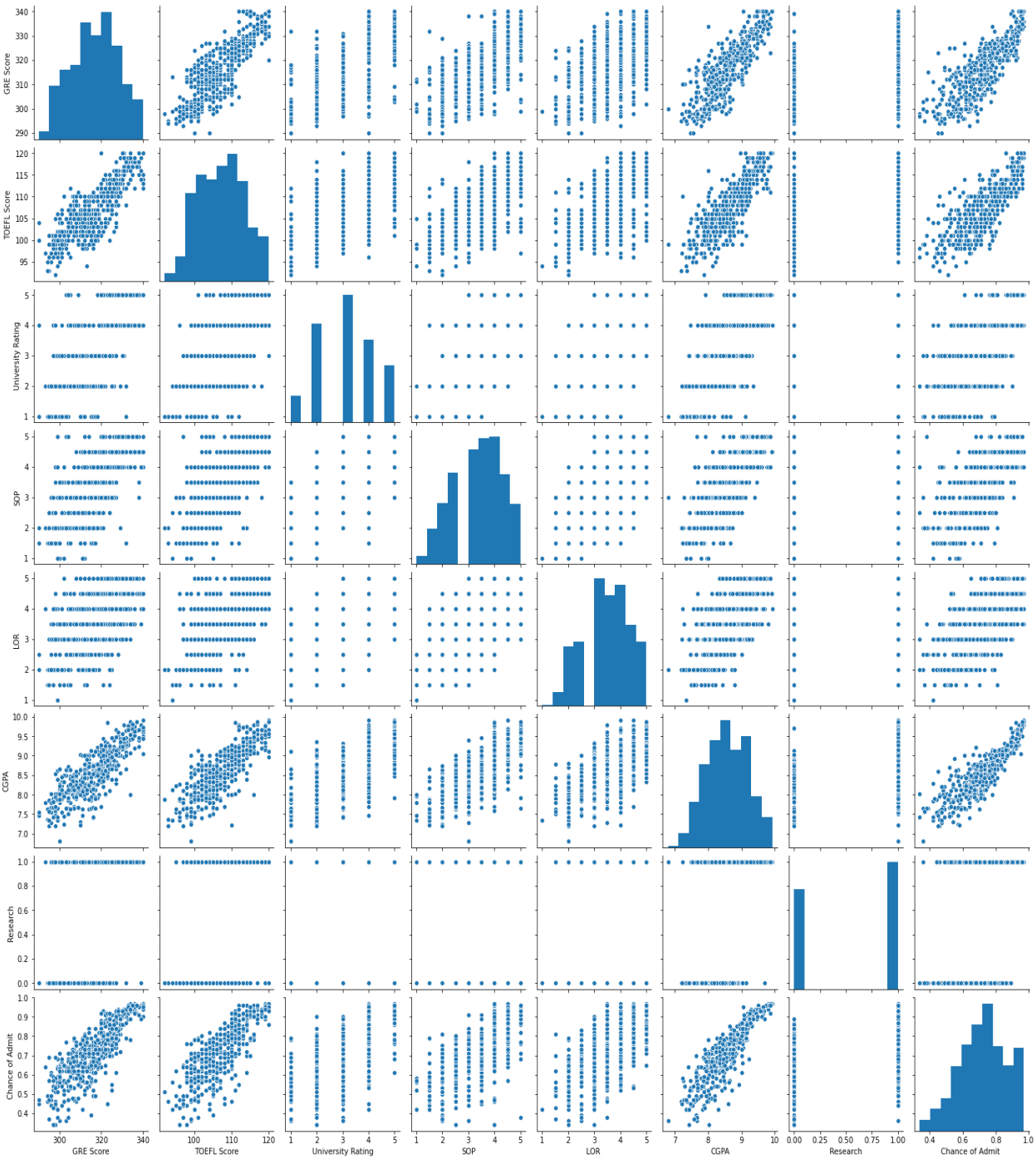
Hence, we collectively came to a conclusion that Random Forest Regressor without principal component analysis, is the best model to predict the chances of a graduate admission into a college based on the dataset provided.

GRAPHS

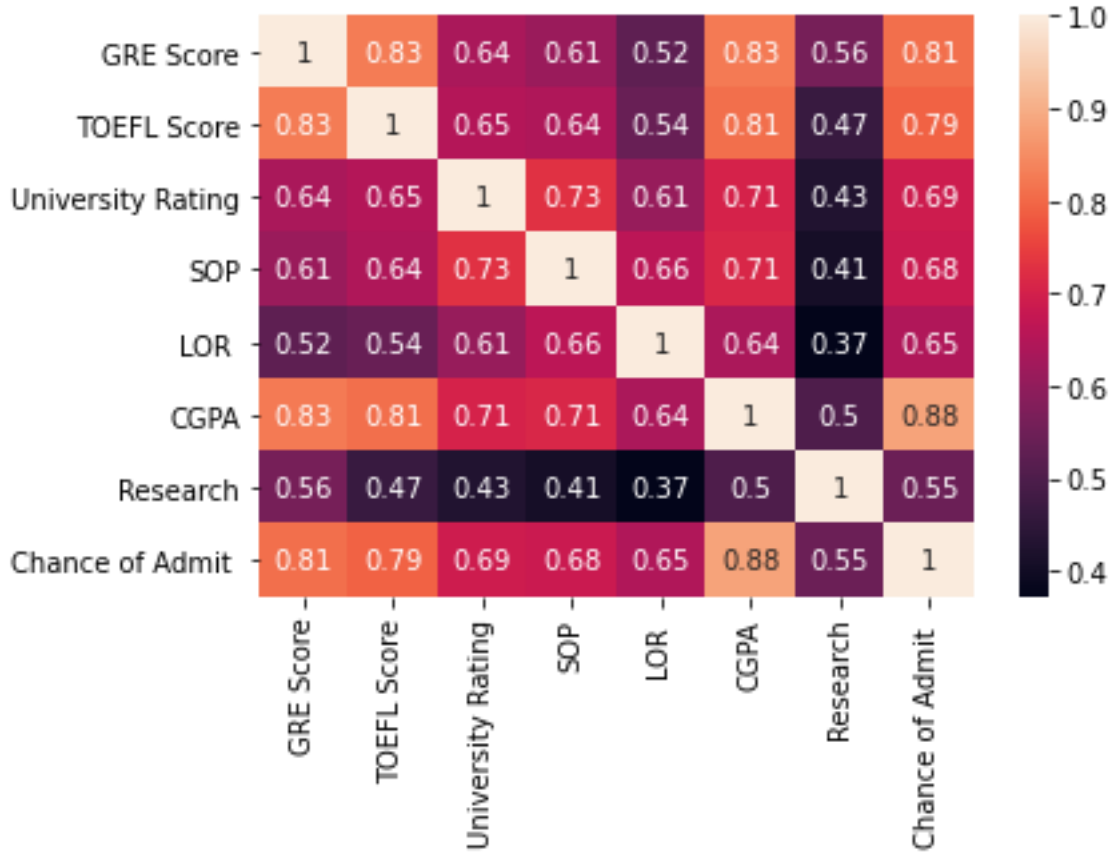
Histograms of all features in the dataset



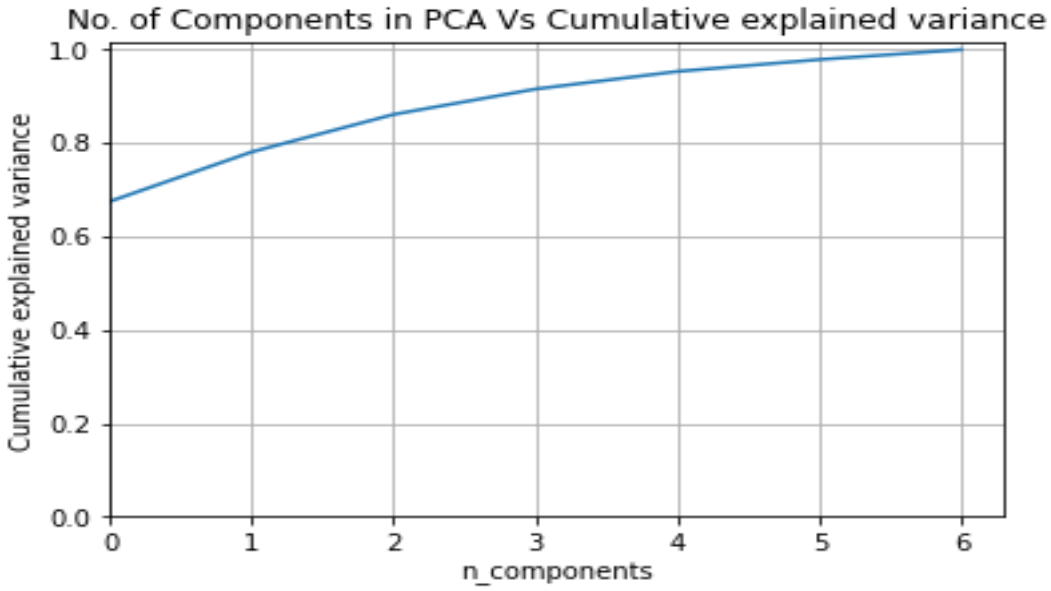
Pair plots


GRE Score
TOEFL Score
University Rating
SOP
LOR
CGPA
Research
Chance of Admit

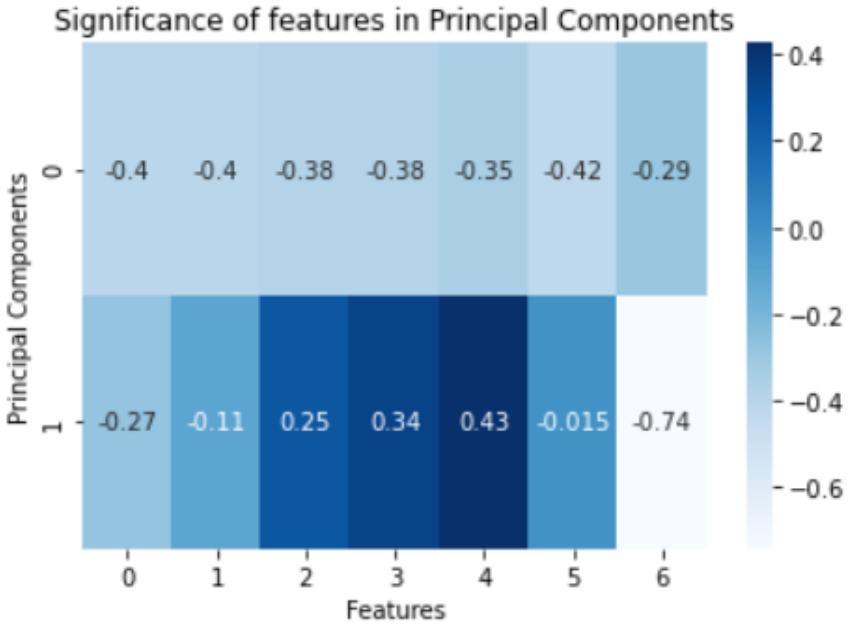
Correlation Matrix



Visualising the graph to predict sufficient no. of components in PCA

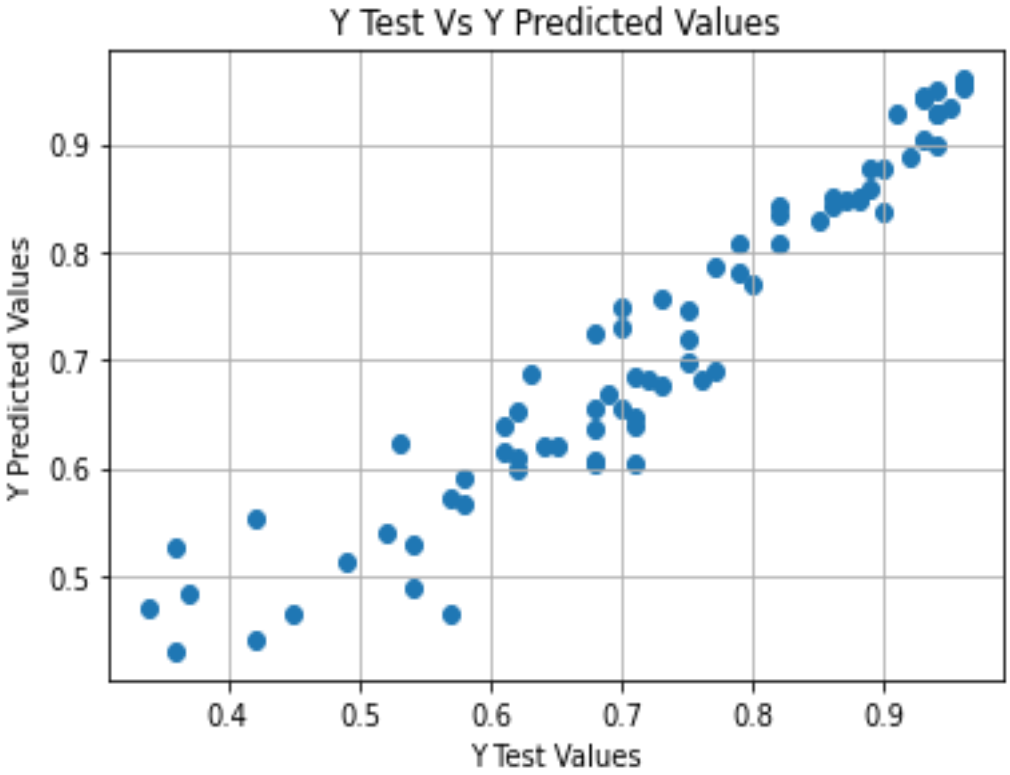


HeatMap showing significance of features in each Principal Component

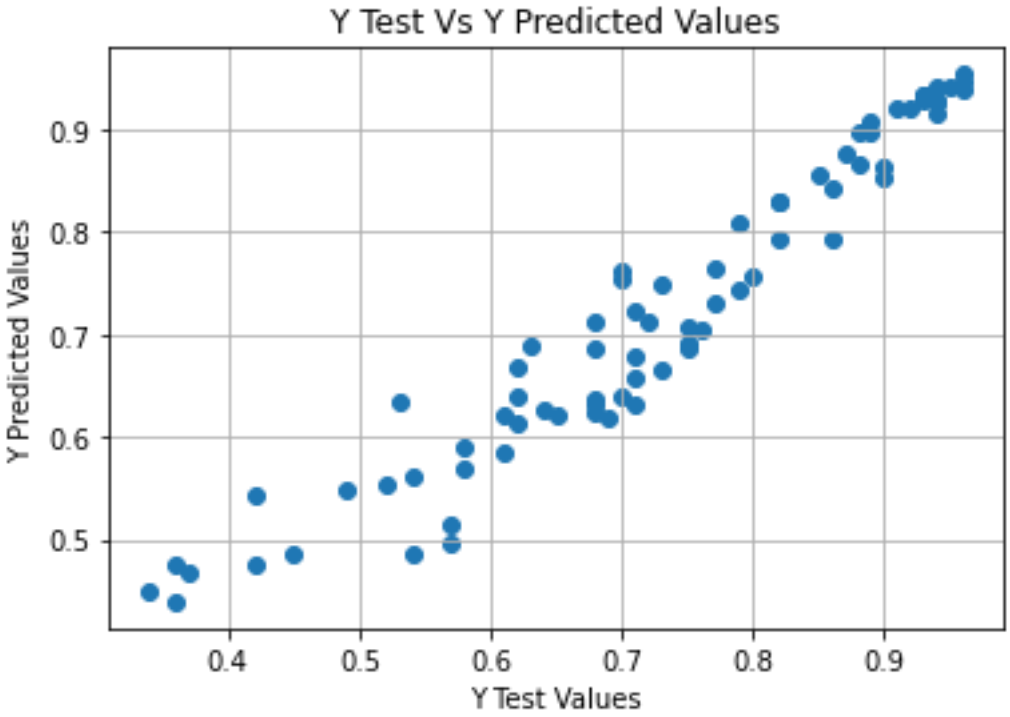


Graphs: y_test vs y_pred

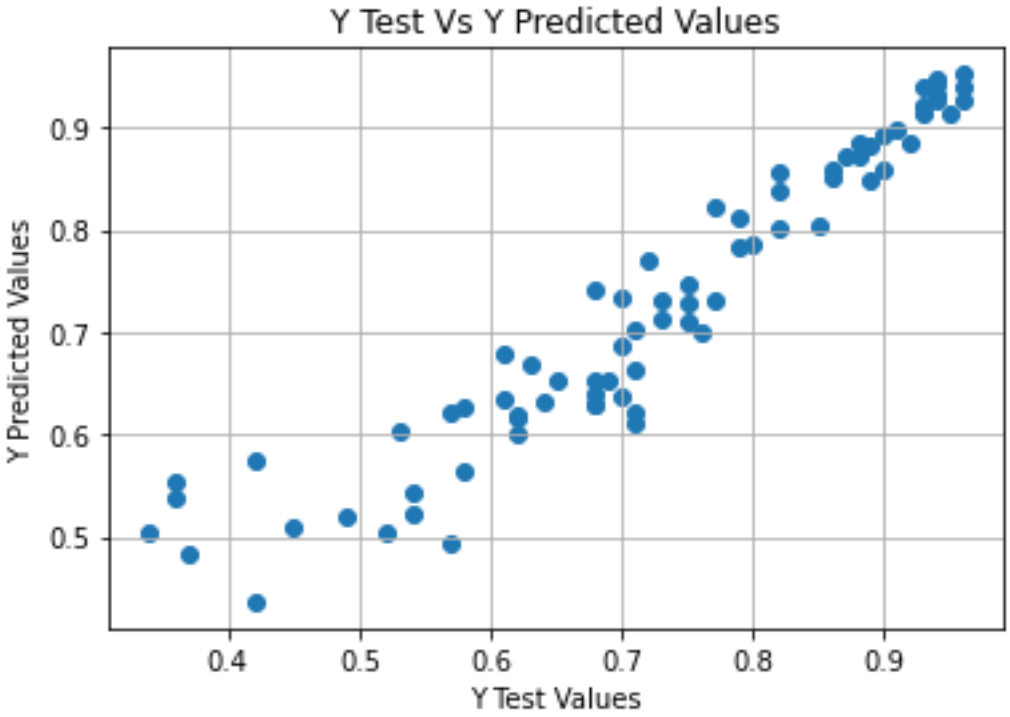
- **MLR without PCA**



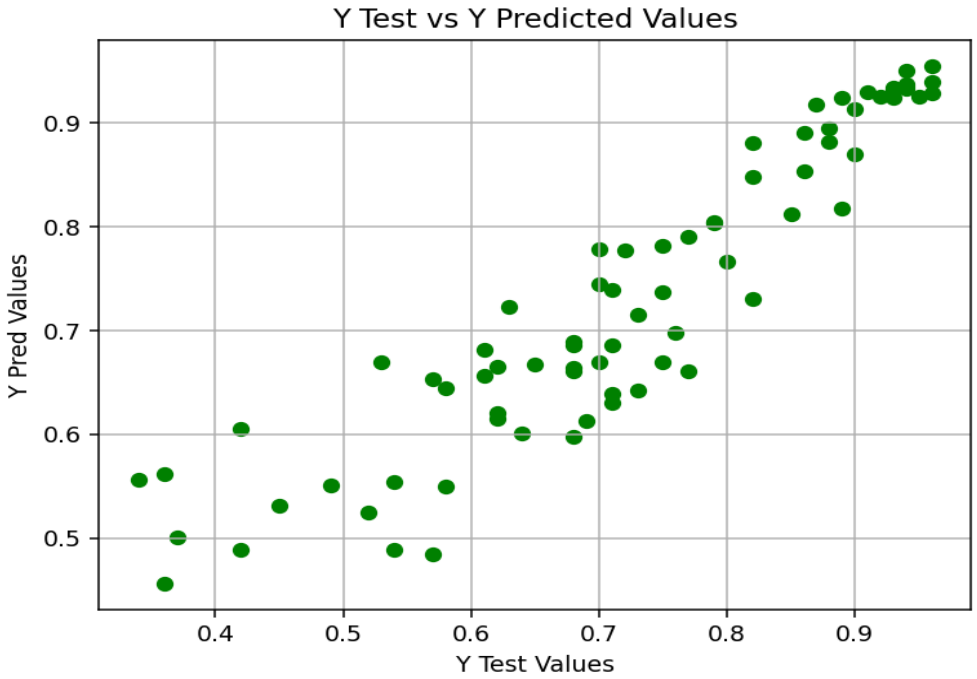
- **RFR without PCA**



- **MLR with PCA**

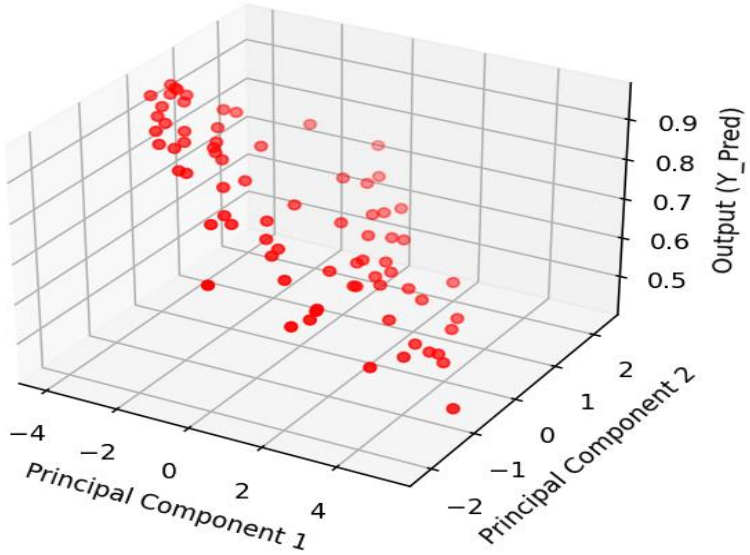


- **RFR with PCA**

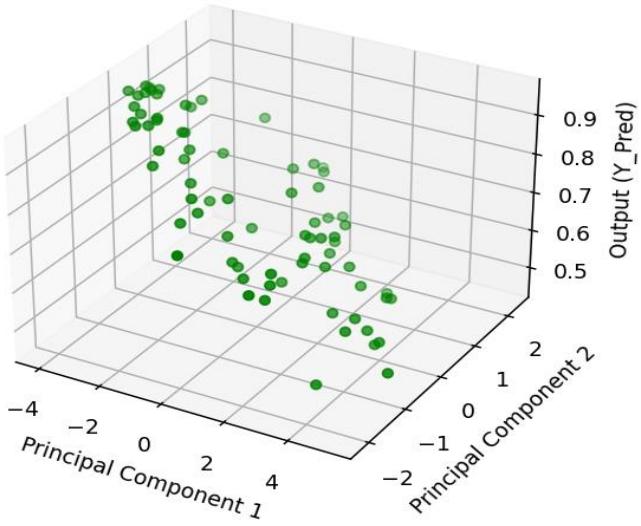


3D Graph between PCA Components and Output(Y_Pred)

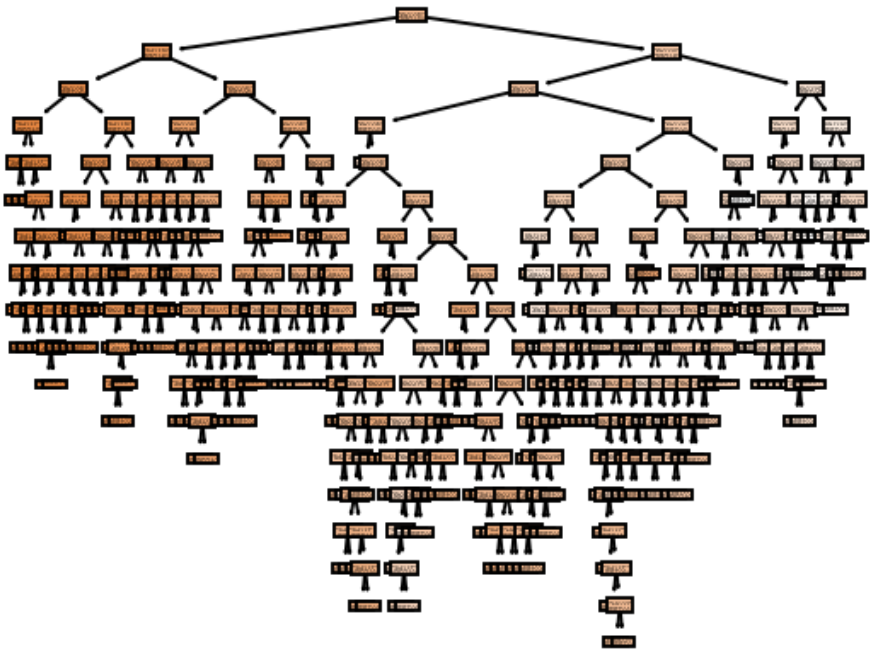
- **MLR with PCA**



- **RFR with PCA**



Visualizing Random Forest formed at index 9



Comparing all the four models on the basis of minimum MSE and RMSE and maximum r2_score

Creating Grouped bar chart to determine the best model