1. INTRODUCTION

1.1 MindMatrix

MindMatrix is designed to bring Industry and Academia together through close interaction between industry professionals and students. MindMatrix is an Industry aligned program for the Engineering students of colleges affiliated to State Universities. It is designed to bring Industry and Academia together through close interaction between industry professionals and students. These programs will be delivered to the students through the mobile based online platform. It includes Faculty Training, industry aligned course ware, Near Real life projects, competitions, Industry mentorship, live webinars etc.

Website
http://www.mindmatrixlabs.com/

Headquarters
Bangalore, Karnataka

Year Founded
2017

Company Type
Privately Held

Size
51-200 employees

1.2 MACHINE LEARNING

Machine learning (**ML**) is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering, and computer vision, where it is infeasible to develop an algorithm of specific instructions for performing the task. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning.

In its application across business problems, machine learning is also referred to as predictive analytics.

1.3 PROBLEM STATEMENT

‘Used Car Price Prediction in India’ is a project that tries to predict the best price for a used car based on some parameters of the vehicle.

The aim is to develop a Machine Learning Model that considers a data-set which consists a large number of records about used cars having the following attributes:

- Name
- Location
- Year purchased
- Kilometers Driven
- Fuel Type
- Transmission
- Owner Type
- Mileage
- Engine
- Power
- Number of Seats

Python Machine Learning Libraries and Frameworks will be used for the processing and visualization of data.

After proper analysis, the model will be designed selecting the best suited algorithm.

The model should be able to predict the price of an unknown car with as much accuracy as possible.

1.4 OBJECTIVES

The objective of our project is to be able to predict the price of a used car given various attributes (data) of that car. There is a saying that a car loses 10% of its value the moment you drive it off a lot. Given, that we would expect that one of the main predictors is the amount of miles driven in the car, since more driving wears down the car. Additionally, we would expect the brand (make) of the car to also be a factor in the price of a used car, since some brands of cars cost more and may be better made. We expect to encounter some issues with multicollinearity since some aspects of cars may be highly correlated. For example, larger cars will probably have larger engines and more doors. Larger engines are correlated with more cylinders.

1.5 SCOPE

(i) The outcomes of this project will be a result of the combined efforts of good data collection and efficient understanding and modelling.

(ii) The boundary of this project is that the data is limited to the Indian market but the approach aims to be as general as possible.

(iii) The biggest risks to the completion of the project may be the fact that existing methods may not be well suited to the problem and new methods may need to be formulated.

(iv) The project will be sought to carried out with as much efficiency as possible and in the given deadlines.

1.6 Timeline of Activities

Week 1: (04-06-19 to 12-06-19)

- A non-technical description of the project was presented.
- The prospects of the idea were discussed.

Week 2: (13-06-19 to 19-06-19)

- The environment setup for the project was done (Python Data Science tools).
- We also saved Jupyter Notebook programs (.ipython) files to python scripts (.py) and run them from console.
- Reviewed the basic theoretical concepts related to association analysis.
- Wrote programs for calculating support, confidence and lift.

Week 3: (20-06-19 to 26-06-19)

- The intended data set was taken up and some basic operations of loading and displaying was done.
- Preprocessing of the data was carried out.
- A basic model of Linear Regression was taken up and implemented with the dataset.

Week 4: (27-06-19 to 03-07-19)

- The documentation of the progress was done.
- The data was scaled up and the changes in the accuracies was observed.
- Wrote programs for taking inputs for filename and accepting inputs from command line arguments.
- Operations like loading the data from absolute/relative path and from URL directly.

Week 5: (04-07-19 to 10-07-19)

- The data from UCI repository is taken up and Linear Regression model was implemented on it.

Week 6: (11-07-19 to 17-07-19)

- As we compared our results with already worked model, we got almost same but less accuracy so we tried to understand what was the problem here by preprocessing the data.

Week 7: (18-07-19 to 24-07-19)

- Remaining coding and documentation.

2. IMPLEMENTATION

2.1 Technology used:

The task was to carry out a prediction of price of cars using various factors and attributes. For this purpose, various tools were required to load the data, find out the relationships between attributes, manipulate the data according to the requirements and use a suitable machine learning algorithm to make the predictions.

For our purpose, Python was chosen as the primary language and the following tools/libraries were used:

1. Jupyter Notebook: It is a web application that is used to create documents containing code, equations and visualizations. The entire code for the project was implemented in a Jupyter Notebook.

2. Numpy: It is a package for scientific computing in Python. Handling of arrays in the project was done using Numpy.

3. Pandas: It provides easy to use data structures and data analysis in Python. Loading, storing and manipulation of data was done using a Pandas dataframe.

4. Matplotlib: It facilitates 2D visualisations of data like plots, histograms, bar graphs etc. Data disrtibutions in attributes was plotted using Matplotlib.

5. Seaborn: It is a Python data visualization library based on Matlpotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. A heatmap of a correlation matrix was generated using Seaborn.

6. Sckit Learn: It provides simple and efficient tools for data mining and data analysis. Splitting of data, scaling of data, application of various algorithms were carried out with scikit-learn.

2.2 Code Snippets:

Important parts of the code are included as under:

1. Label Encoding the object attributes and cleaning the dataset:

```
#Encoding the dataset

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

for col in df.columns.values:

    if df[col].dtypes=='object':

        data = df[col]

        le.fit(data.values)

        df[col]=le.transform(df[col])

#clean the dataset; removing the NaN and infinity values

def clean_dataset(df):

    assert isinstance(df, pd.DataFrame), "df needs to be a pd.DataFrame"

    df.dropna(inplace=True)

    indices_to_keep = ~df.isin([np.nan, np.inf, -np.inf]).any(1)

    return df[indices_to_keep].astype(np.float64)

clean_dataset(df);
```

2. Obtaining the correleation matrix and visulaising it as a heatmap:

```
#obtain correlation matrix of the numeric dataset

cor = df_numeric.corr()

# figure size

plt.figure(figsize=(16,8))

cmap = sns.diverging_palette(220, 10, as_cmap=True)
```

```
# heatmap to check the relationship between different numerical attributes

sns.heatmap(cor, cmap=cmap, annot=True)

plt.show()
```

3. Implementing the Linear Regression Algorithm and calculating the r2 score and cross validation score:

```
#performing linear regression

from sklearn.linear_model import LinearRegression

linearRegressor = LinearRegression()

m1 = linearRegressor.fit(x_train,y_train)

#performing prediction

y_predict = linearRegressor.predict(x_test)

#import library for accuracy calculation

from sklearn.metrics import r2_score

from sklearn.model_selection import cross_val_score

print("Score = %0.2f" %(r2_score(y_test,y_predict)*100))

scores = cross_val_score(m1, x_test,y_test.values.ravel(), cv=10)

print ('Acuracy %0.2f' % abs((scores.mean()*100)))
```

2.3 Algorithms Used:

After the visualizations were made and the necessary preprocessing was done, the following algorithms were applied on the data for making predictions:

Some ideas were obtained in the process of analyzing a journal [1].

1. Linear Regression: Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One or more variables are considered to be an explanatory variable, and the other is considered to be a dependent variable.

2. Random Forest Regression: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that 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.

3. Decision Trees Regression: Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data

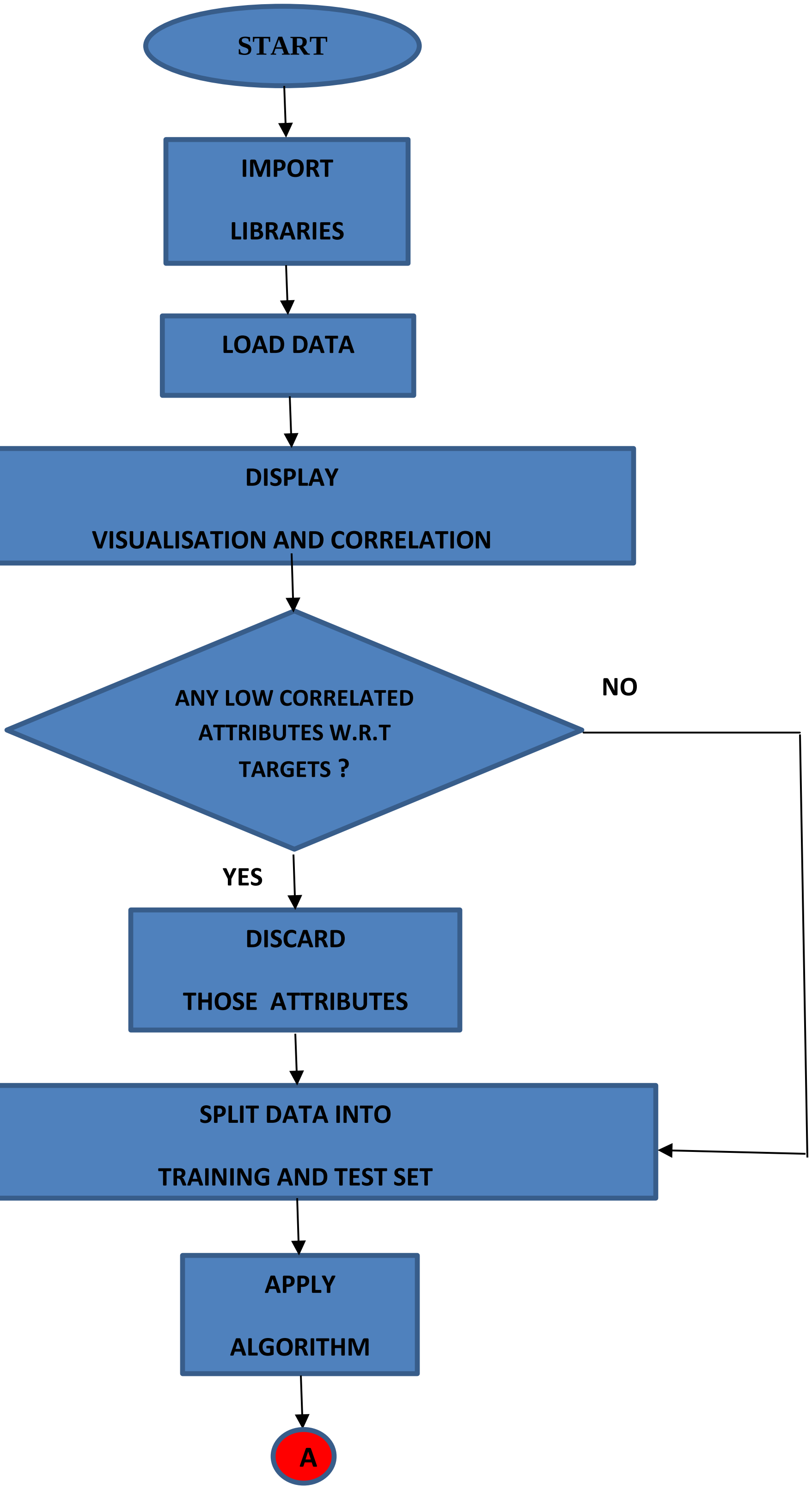
4. Support Vector Regression: "Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well

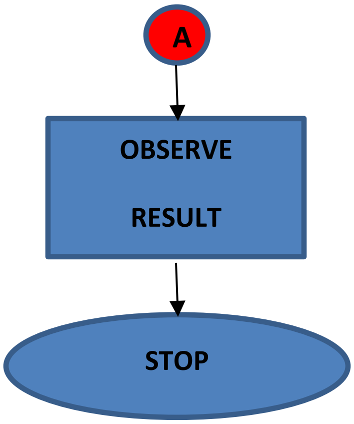
5. Neural Networks:

A neural network is a network or circuit of neurons, or in a modern sense, an artificial neural network, composed of artificial neurons or nodes. The algorithm used here is Mulilayer Perceptron (MLP) Regressor which considers the attributes in the input layer and processes it through multiple layers of perceptrons carrying out multiple iterations and produces an output.

All the algorithms were implemented using the modules of the sklearn library

2.4 Flowchart:




A
OBSERVE
RESULT
STOP

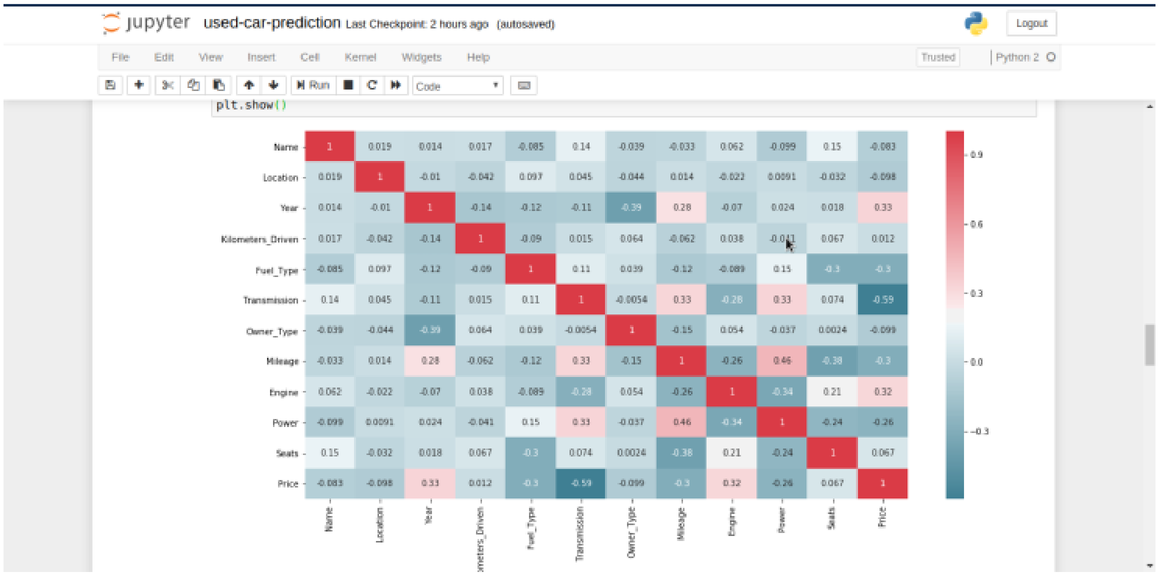
3. RESULTS

The dataset used was loaded and the following attributes were found to exist

```
Data columns (total 12 columns):
Name                non-null object
Location            non-null object
Year                non-null int64
Kilometers_Driven   non-null int64
Fuel_Type           non-null object
Transmission        non-null object
Owner_Type          non-null object
Mileage             non-null object
Engine              non-null object
Power               non-null object
Seats               non-null float64
Price               non-null float64
dtypes: float64(2), int64(2), object(8)
```

The attributes of type ‘object’ cannot be processed directly so they were encoded using a Label Encoder.

A correlation matrix was plotted to check the extent of correlation of the target attribute Price with all the other attributes. None of the attributes exhibited an excessively low correlation value so no attributes were dropped.



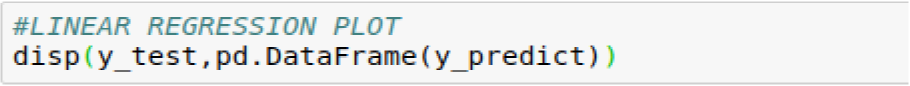
```
plt.show()
```

	Name	Location	Year	ometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price
Name	1	0.019	0.014	0.017	-0.085	0.14	-0.039	-0.033	0.062	-0.099	0.15	-0.083
Location	0.019	1	-0.01	-0.042	0.097	0.045	-0.044	0.014	-0.022	0.0091	-0.032	-0.098
Year	0.014	-0.01	1	-0.14	-0.12	-0.11	-0.39	0.28	-0.07	0.024	0.018	0.33
Kilometers_Driven	0.017	-0.042	-0.14	1	-0.09	0.015	0.064	-0.062	0.038	-0.041	0.067	0.012
Fuel_Type	-0.085	0.097	-0.12	-0.09	1	0.11	0.039	-0.12	-0.089	0.15	-0.3	-0.3
Transmission	0.14	0.045	-0.11	0.015	0.11	1	-0.0054	0.33	-0.28	0.33	0.074	-0.59
Owner_Type	-0.039	-0.044	-0.39	0.064	0.039	-0.0054	1	-0.15	0.054	-0.037	0.0024	-0.099
Mileage	-0.033	0.014	0.28	-0.062	-0.12	0.33	-0.15	1	-0.26	0.46	-0.38	-0.3
Engine	0.062	-0.022	-0.07	0.038	-0.089	-0.28	0.054	-0.26	1	-0.34	0.21	0.32
Power	-0.099	0.0091	0.024	-0.041	0.15	0.33	-0.037	0.46	-0.34	1	-0.24	-0.26
Seats	0.15	-0.032	0.018	0.067	-0.3	0.074	0.0024	-0.38	0.21	-0.24	1	0.067
Price	-0.083	-0.098	0.33	0.012	-0.3	-0.59	-0.099	-0.3	0.32	-0.26	0.067	1

Further procedures included the application of the intended algorithms over a varying number of records and the results of the r2_score and cross_validation_score were observed.

	2000		4000		6000	
	r2_score	cv_score	r2_score	cv_score	r2_score	cv_score
Linear Regression	55.67	49.16	58.53	58.35	47.17	48.24
Random Forest Regression	56.79	59.99	68.57	67.36	48.10	56.15
Random Forest Regression	71.01	54.44	71.83	61.50	72.38	68.84
Support Vector Regression	59.01	40.35	66.99	55.96	59.32	49.87
Neural Networks	84.47	56.07	84.11	76.85	78.75	60.66

Taking into account both the types of scores it can be inferred that for the given dataset, the Neural Networks (MLP Regressor) algorithm works better when trained and tested on a set of 4000 records.



```
#LINEAR REGRESSION PLOT
disp(y_test,pd.DataFrame(y_predict))
```

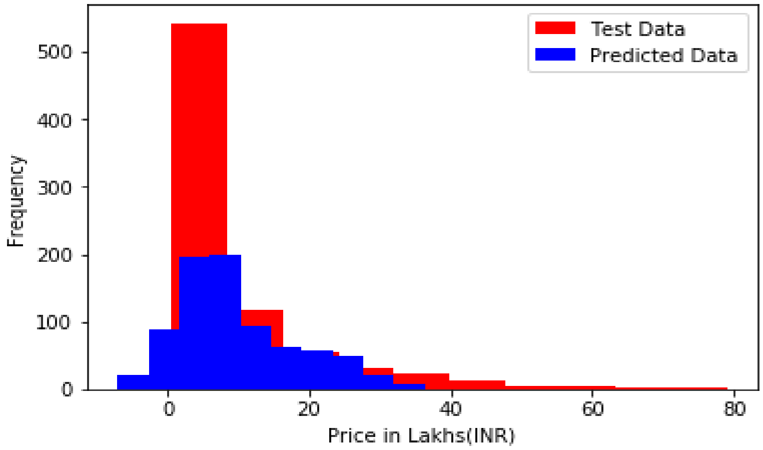
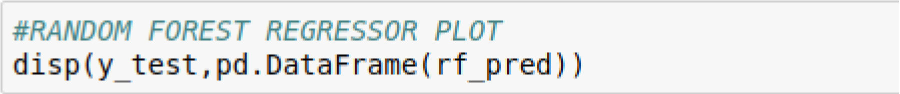


Fig 01: Linear Regression Plot



```
#RANDOM FOREST REGRESSOR PLOT
disp(y_test,pd.DataFrame(rf_pred))
```

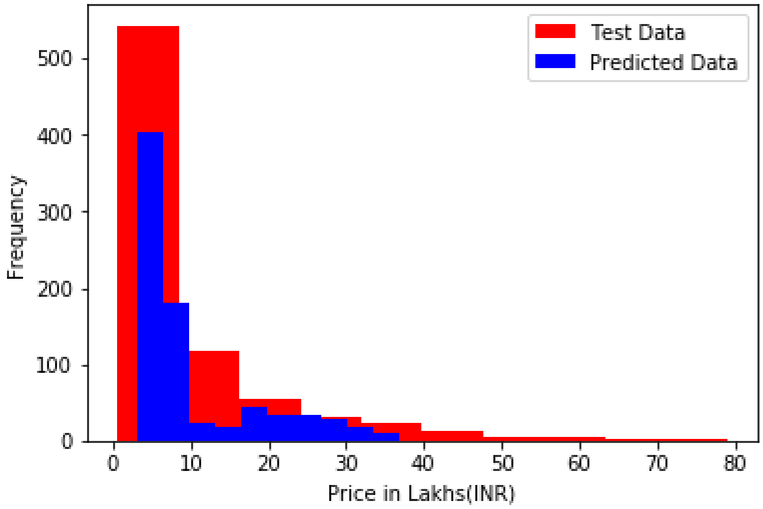
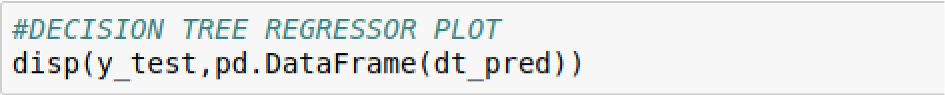


Fig 02: Random Forest Regressor Plot



```
#DECISION TREE REGRESSOR PLOT
disp(y_test,pd.DataFrame(dt_pred))
```

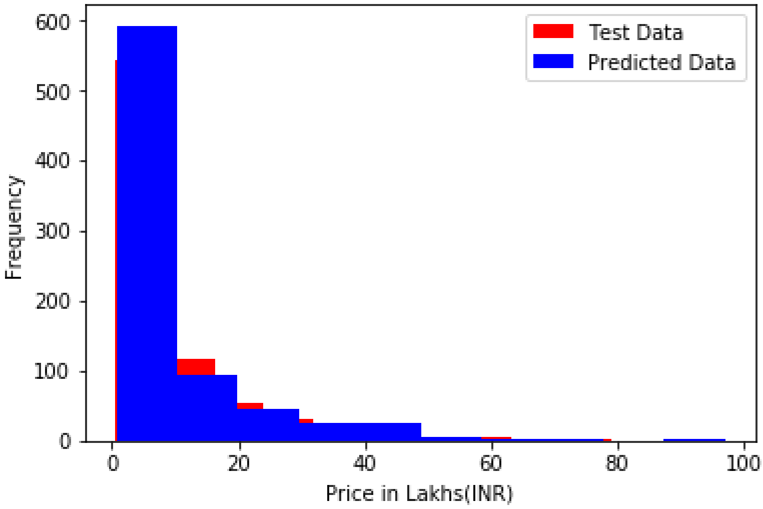
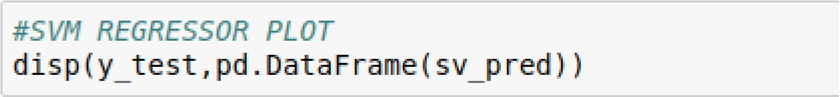


Fig 03: Decision Tree Regressor Plot



```
#SVM REGRESSOR PLOT
disp(y_test,pd.DataFrame(sv_pred))
```

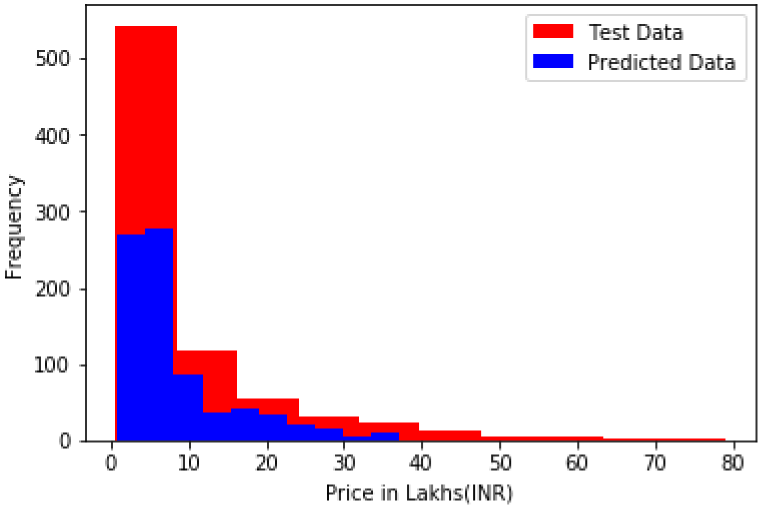
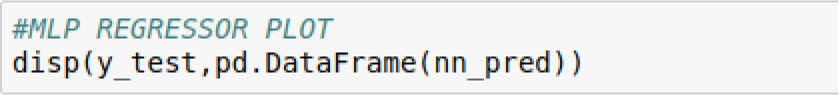


Fig 04: SVM Regresor Plot



```
#MLP REGRESSOR PLOT
disp(y_test,pd.DataFrame(nn_pred))
```

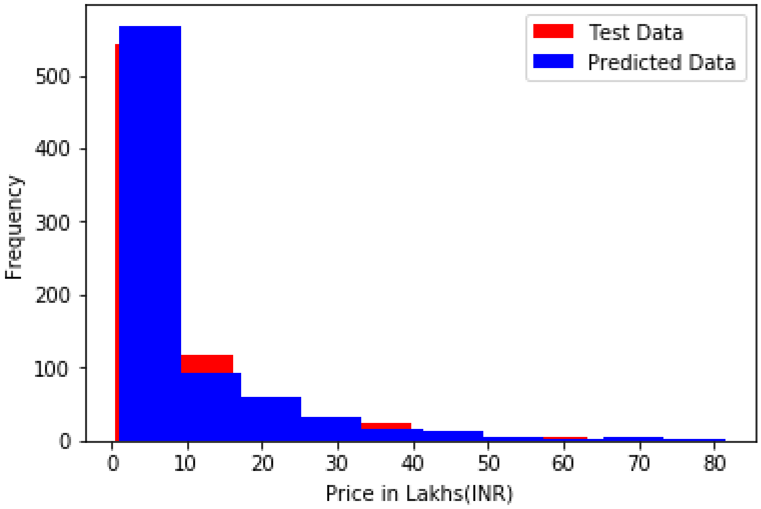


Fig 05: MLP Regressor Plot

4. CONCLUSION

The implementation of this project led to the understanding of modelling a price prediction system using real world data. The outcomes were interesting as the observations showed that for the same preprocessing steps, different algorithms can yield a better model for prediction as compared to others.

Undeniably there were a few challenges which we came across in this attempt. A few being understanding the data and knowing how to apply the right methods to process it.

Although some promising results were obtained, yet it can be said that there is a room for improvement and the work will definitely be continued towards identifying and implementing those improvements.

For future prospects, we hope to experiment with more methods and algorithms which would could possibly give us extremely better results. Also, we hope to deploy our work into an application and make it usable for real life purposes.

5. REFERENCES

[1]. Enis Gegic, Becir Isakovic, Dino Keco, Zerina Masetic, Jasmin Kevric (2019) ‘Car Price Prediction using Machine Learning Techniques’. *TEM Journal. Volume 8, Issue 1, Pages 113-118, ISSN 2217-8309, DOI: 10.18421/TEM81-16.*