HOUSE PRICE PREDICTION USING MACHINE LEARNING

Project

submitted in partial fulfillment of the

requirements for the award of the degree of

BACHELOR OF TECHNOLOGY (B. TECH)

Submitted by:

Rishabh Kumar	180060101085
Rishabh Tripathi	180060101086
Shivang Goel	180060101104
Shiva Kant Jha	180060101103

Under the supervision of

Mr. Himanshu Bartwal — *(Assistant Professor)*

Dr. Taresh Singh — *(HoD)*

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

COLLEGE OF ENGINEERING ROORKEE, ROORKEE

ROORKEE-247667 (UTTARAKHAND) INDIA

MAY, 2022

ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to **Mr. Himanshu Bartwal** Department of Computer Science Engineering for his generous guidance, help and useful suggestions. I express my sincere gratitude to **Dr. Taresh Singh, HoD** in Department of Computer Science Engineering for his stimulating guidance, continuous encouragement and supervision throughout the course of present work. I also wish to extend my thanks to **Mr. Himanshu Bartwal** for their insightful comments and constructive suggestions to improve the quality of this research work. I am extremely thankful to Department of Computer Science Engineering for providing me infrastructural facilities to work in, without which this work would not have been possible.

Rishabh Kumar **180060101085** _____

Rishabh Tripathi **180060101086** _____

Shivang Goel **180060101104** _____

Shiva Kant Jha **180060101103** _____

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the project report entitled "**House Price Prediction Using Machine Learning**" by "**Rishabh Kumar, Rishabh Tripathi, Shivang Goel, Shiva Kant Jha**" in partial fulfillment of requirements for the award of degree of B.Tech. submitted in the Department of Computer Science Engineering at "**COLLEGE OF ENGINEERING ROORKEE**" under **UTTARAKHAND TECHNICAL UNIVERSITY** is an authentic record of our own work carried out during a period from 15-03-2022 to 25-05-2022 under the supervision of **Mr. Himanshu Bartwal.**

Rishabh Kumar	**180060101085**	**_____**
Rishabh Tripathi	**180060101086**	**_____**
Shivang Goel	**180060101104**	**_____**
Shiva Kant Jha	**180060101103**	**_____**

CERTIFICATE OF APPROVAL

This is to certify the report entitled the "**House Price Prediction Using Machine Learning**" is record of bonafide work, carried out by **"Rishabh Kumar, Rishabh Tripathi, Shivang Goel, Shiva Kant Jha"** under my supervision and guidance.

In my opinion, the report in its present form is in partial fulfillment of all the requirement for the award of **Bachelor of Technology** (B. Tech), in Computer Science Engineering at College of Engineering Roorkee, is an authentic work carried by them under my supervision and guidance.

To the best of my knowledge, the matter embodied in the project has not been submitted for any other degree or diploma.

Mr. Himanshu Bartwal — Dr. Taresh Singh

(Assistant Professor) — (Hod)

The B. Tech Viva –Voce Examination of Rishabh Kumar, Rishabh Tripathi, Shivang Goel, Shiva Kant Jha has been held on 30-05-2022 and is accepted.

External Examiner

TABLE OF CONTENT

TABLE OF CONTENT

TABLE OF CONTENT

ABSTRACT

Housing prices are increasing every year, necessitating the creation of a long-term housing price strategy. Predicting a home's price will assist a developer in determining a home's purchase price, as well as a consumer in determining the best time to buy a home. The sale price of real estate in major cities depends on the specific circumstances. Housing prices are constantly changing from day to day and are sometimes fired rather than based on estimates. Predicting real estate prices by real factors is a key element as part of our analysis. We want to make our test dependent on all of the simple metrics that are taken into account when deciding the significance. In this project we use some machine learning algorithms like linear regression and random forest regressor and our results are not self-inflicted process rather is a weighted method of various techniques to give the most accurate results. There are thirteen features in the data collection. In this project, there has been an effort to build a forecasting model for determining the price based on the variables that influence the price. The results have proven to be effective lower error and higher accuracy than individual algorithms are used.

Keywords: Linear Regression, Random Forest Regressor, Forecasting Model

CHAPTER 1

INTRODUCTION

In this project we are going to do implementing a machine learning model for predicting the house price prediction using some of the regression techniques based of some of features in the dataset which is called Boston House Price Prediction. There are some of the processing techniques for creating a model.

1.1 MOTIVATION

Being extremely interested in everything having a relation with the Machine Learning, the group project was a great occasion to give us the time to learn and confirm our interest for this field. The fact that we can make estimations, predictions and give the ability for machines to learn by themselves is both powerful and limitless in term of application possibilities. We can use Machine Learning in Finance, Medicine, almost everywhere. That's why we decided to conduct my project around the Machine Learning.

1.2 IDEA

As a first experience, we wanted to make our project as much didactic as possible by approaching every different step of the machine learning process and trying to understand them deeply. So, we choose Boston Housing Price Prediction as our final year project. The goal was to predict the price of a given house according to the market prices taking into account different "features" that was available in the dataset.

1.3 SOURCE

Because we truly think that sharing sources and knowledges allow to help others but also ourselves, the source of the dataset and its description is available at the following link: [1] https://archive.ics.uci.edu/ml/machine-learning-databases/housing/

- Origin: This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University.
- Creator: [2] Harrison, D. and Rubinfeld, D.L. 'Hedonic prices and the demand for clean air', J. Environ. Economics & Management, vol.5, 81-102, 1978.
- Date: July 7, 1993

1.4 PAST USAGE

- Used in Belsley, Kuh & Welsch, 'Regression diagnostics ...', Wiley, 1980.
  N.B. Various transformations are used in the table on pages 244-261.
- Quinlan, R. (1993). Combining Instance-Based and Model-Based Learning.
  In Proceedings on the Tenth International Conference of Machine Learning, 236-243,
  University of Massachusetts, Amherst. Morgan Kaufmann.

CHAPTER 2

LITRATURE REVIEW

The field of Data Science is rather young, having taken form over the last half century as a discipline distinct from statistics. It is also rapidly growing with many interesting advancements in recent years, most notably within Machine Learning (ML). This has resulted in an increase in media attention as well as funding of AI related businesses and research projects.

In 50 years of Data Science [3] Donoho comments on the history of Data Science and questions whether it is really different from statistics. With regards to Machine Learning, he points to a study he conducted that compared a set of highly-cited and glamorous classifier methods such as Random Forests and k-Nearest neighbour to a simple linear classifier applied on the same problem. The study found that the simpler method did not only perform similarly, but had a lower worst-case regret. This suggests that when benchmarked, more advanced ML algorithms are not necessarily better when put in practice, which highlights the need for making algorithm comparisons.

A study using similar techniques was made on predicting the sales price of used cars. [4] This problem is similar to predicting house prices and arguably simpler because it is dealing with cars, commodities that aren't geographically fixed and are often highly standardized. The methods used were Multiple Linear Regression, k-Nearest Neighbours, Naïve Bayes and Decision Trees, including Random Forest. Cross validation was used for finding the optimal hyperparameters, such as the 'k' for k-NN.

The house pricing problem was approached by Bal dominos et al. [5] from the viewpoint of finding investment opportunities. They formulated the regression problem and used several Machine Learning algorithms such as k-Nearest neighbour, variations of neural networks and decision trees.

Another study by Oxenstierna [6] investigated it for the purposes of valuation of houses. The data set included 5000 entries. Again, the k-Nearest neighbour method was used as well as Artificial Neural Networks, to minimize the median absolute percentage error of the prediction. The methods performed similarly at around 8-9 % Median Absolute Percentage Error.

CHAPTER 3

METHODOLOGY

In order to answer the question about what machine learning method is better to use for the Boston House Price problem, the algorithms Linear Regression and Random Forest have been compared in terms of their prediction accuracy. Instead of implementing the algorithms from scratch for this study, algorithms from the scikit-learn library have been used. It is a state-of-the-art library part of the scikit suite of scientific toolkits for Python. We have also used the python data analysis library Pandas. Prior to comparing the algorithms, the data set has been pre-processed and cleaned in order for the algorithms to take the data as input. Moreover, a method for evaluating the data has been established and finally the machine learning algorithms have been executed with the cleaned data set and tested with different values for relevant hyperparameters for prediction.

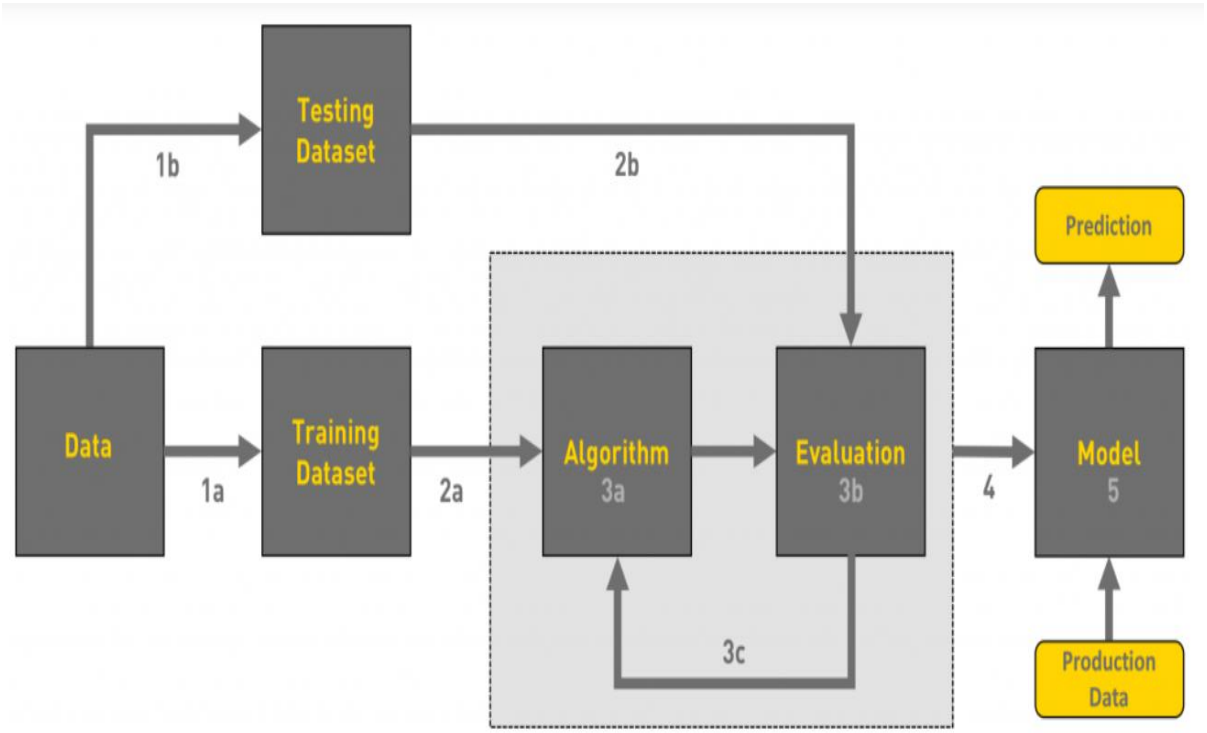


Fig 3.1 Flow Chart

3.1 ABOUT DATASET

Housing prices are an important reflection of the economy, and housing price ranges are of great interest for both buyers and sellers. In this project, house prices will be predicted given explanatory variables that cover many aspects of residential houses. The goal of this project is to create a regression model that is able to accurately estimate the price of the house given the features.

In this dataset made for predicting the Boston House Price Prediction. All of the feature for each house are given below:

- CRIM per capita crime rate by town
- ZN proportion of residential land zoned for lots over 25,000 sq. Ft.
- INDUS proportion of non-retail business acres per town
- CHAS Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX nitric oxides concentration (parts per 10 million)
- RM average number of rooms per dwelling
- AGE proportion of owner-occupied units built prior to 1940
- DIS weighted distances to five Boston employment centres
- RAD index of accessibility to radial highways
- TAX full-value property-tax rate per $10,000
- PTRATIO pupil-teacher ratio by town
- B 1000(Bk - 0.63) ^2 where Bk is the proportion of blacks by town
- LSTAT % lower status of the population
- MEDV Median value of owner-occupied homes in $1000's

3.2 DATA OVERVIEW

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33

3.3 DATA COLLECTION

The Dataset was collected from UCI Machine Learning Repository. This Dataset consist several features such as Number of Rooms, Crime Rate, and Tax and so on. As well we can also able to get the dataset from the sklearn datasets.

3.4 DATA PREPROCESSING

In this Boston Dataset we need not to clean the data. The dataset already cleaned when we download from the sklearn.

3.5 ALGORITHMS USED

The major aim of in this project is to predict the house prices based on the features using some of the regression techniques and its algorithms.

3.5.1 LINEAR REGRESSION

Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

This form of analysis estimates the coefficients of the linear equation, involving one or more independent variables that best predict the value of the dependent variable. Linear regression fits a straight line or surface that minimizes the discrepancies between predicted and actual output values. There are simple linear regression calculators that use a “least squares” method to discover the best-fit line for a set of paired data. You then estimate the value of X (dependent variable) from Y (independent variable).

3.5.2 RANDOM FOREST REGRESSOR

The Decision Tree is an easily understood and interpreted algorithm and hence a single tree may not be enough for the model to learn the features from it. On the other hand, Random Forest is also a “Tree”-based algorithm that uses the qualities features of multiple Decision Trees for making decisions.

Therefore, it can be referred to as a ‘Forest’ of trees and hence the name “Random Forest”. The term ‘Random’ is due to the fact that this algorithm is a forest of ‘Randomly created Decision Trees’.

The Decision Tree algorithm has a major disadvantage in that it causes over-fitting. This problem can be limited by implementing the Random Forest Regression in place of the Decision Tree Regression. Additionally, the Random Forest algorithm is also very fast and robust than other regression models.

3.6 STATISITICS USING GRETL

Model 1: OLS, using observations 1-506

Dependent variable: Price

	Coefficient	*Std. Error*	*t-ratio*	*p-value*	
CRIM	−0.0928965	0.0344210	−2.699	0.0072	***
ZN	0.0487150	0.0144033	3.382	0.0008	***
INDUS	−0.00405998	0.0644397	−0.06300	0.9498	
CHAS	2.85400	0.903913	3.157	0.0017	***
NOX	−2.86844	3.35873	−0.8540	0.3935	
RM	5.92815	0.309109	19.18	<0.0001	***
AGE	−0.00726933	0.0138145	−0.5262	0.5990	
DIS	−0.968514	0.195630	−4.951	<0.0001	***
RAD	0.171151	0.0667524	2.564	0.0106	**
TAX	−0.00939622	0.00392309	−2.395	0.0170	**
PTRATIO	−0.392191	0.109869	−3.570	0.0004	***
B	0.0149056	0.00269654	5.528	<0.0001	***
LSTAT	−0.416304	0.0507862	−8.197	<0.0001	***

Mean dependent var	22.53281	S.D. dependent var	9.197104
Sum squared resid	12228.05	S.E. of regression	4.980295
Uncentered R-squared	0.959189	Cantered R-squared	0.713738
F (13, 493)	891.3139	P-value(F)	0.000000
Log-likelihood	−1523.775	Akaike criterion	3073.551
Schwarz criterion	3128.496	Hannan-Quinn	3095.100

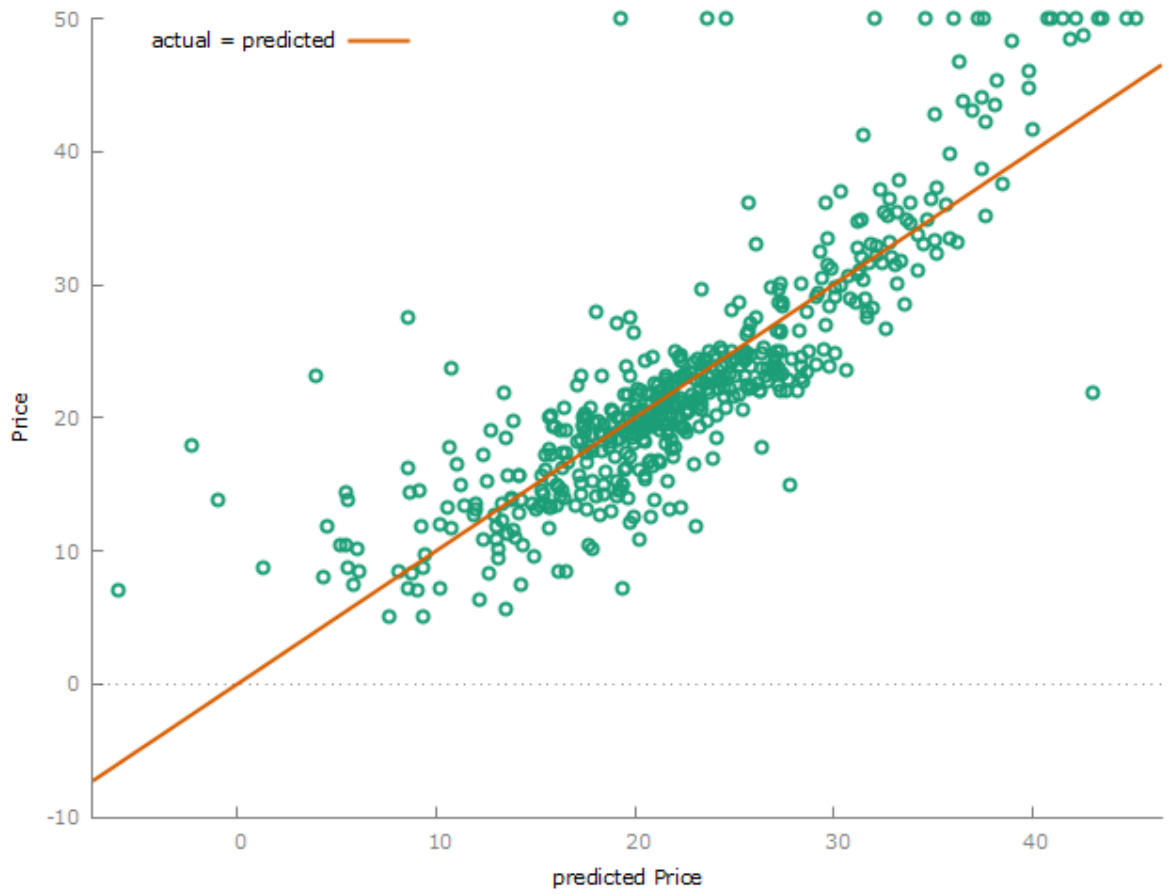

actual = predicted
Price
predicted Price
50
40
30
20
10
0
-10
0
10
20
30
40

Fig 3.6.1

CHAPTER 4

MODLE BUILDING

The steps for building a machine learning model include:

- Importing Libraries
- Importing Dataset
- Data Pre-processing
- Exploratory Data Analysis
- Model Fitting
- Model Evaluation
- Model Visualization
- Model Deployment

4.1 IMPORTING LIBRARIES

```
In [1]: import pandas as pd
        import numpy as np
        from sklearn import metrics
        import matplotlib.pyplot as plt
        import seaborn as sns
        from tkinter import *
        %matplotlib inline
```

Libraries used in this model are:

- Pandas
- NumPy
- Scikit Learn
- Matplotlib
- Seaborn
- Tkinter

4.2 IMPORTING DATASET

```
In [2]: from sklearn.datasets import load_boston
        boston = load_boston()

In [3]: data = pd.DataFrame(boston.data)

In [4]: data.head()
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94

4.3 DATA PREPROCESSING

In [10]:
```
data.nunique()
```

Out[10]:
```
CRIM       504
ZN          26
INDUS       76
CHAS         2
NOX         81
RM         446
AGE        356
DIS        412
RAD          9
TAX         66
PTRATIO     46
B          357
LSTAT      455
PRICE      229
dtype: int64
```

In [11]:
```
data.isnull().sum()
```

Out[11]:
```
CRIM       0
ZN         0
INDUS      0
CHAS       0
NOX        0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
B          0
LSTAT      0
PRICE      0
dtype: int64
```

In [7]:
```
data.shape
```

Out[7]:
```
(506, 14)
```

In [8]:
```
data.columns
```

Out[8]:
```
Index(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
       'PTRATIO', 'B', 'LSTAT', 'PRICE'],
      dtype='object')
```

In [9]:
```
data.dtypes
```

Out[9]:
```
CRIM       float64
ZN         float64
INDUS      float64
CHAS       float64
NOX        float64
RM         float64
AGE        float64
DIS        float64
RAD        float64
TAX        float64
PTRATIO    float64
B          float64
LSTAT      float64
PRICE      float64
dtype: object
```

In [12]:
```
data[data.isnull().any(axis=1)]
```

Out[12]:

CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE

4.4 Exploratory Data Analysis

In [13]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
 2   INDUS    506 non-null    float64
 3   CHAS     506 non-null    float64
 4   NOX      506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      506 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    float64
 9   TAX      506 non-null    float64
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    506 non-null    float64
 13  PRICE    506 non-null    float64
dtypes: float64(14)
memory usage: 55.5 KB
```

In [14]:
```
data.describe()
```

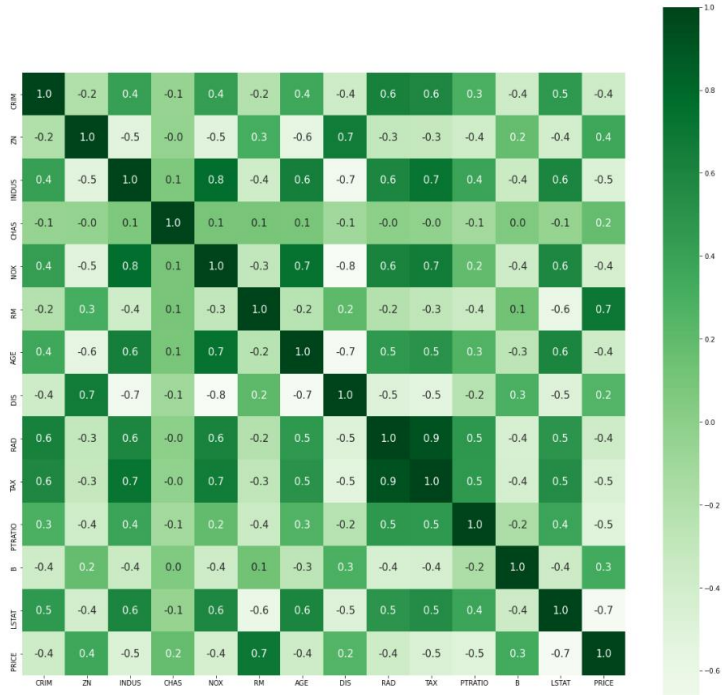
Out[14]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032	12.653063	22.532806
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864	7.141062	9.197104
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000	1.730000	5.000000
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500	6.950000	17.025000
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000	11.360000	21.200000
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000	16.955000	25.000000
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000	37.970000	50.000000

In [15]:
```
corr = data.corr()
corr.shape
```

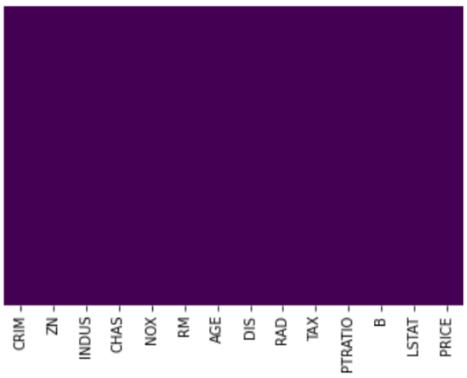
Out[15]: (14, 14)

In [16]:
```
# Plotting the heatmap of correlation between features
plt.figure(figsize=(20,20))
sns.heatmap(corr, cbar=True, square= True, fmt='.1f', annot=True, annot_kws={'size':15}, cmap='Greens')
```



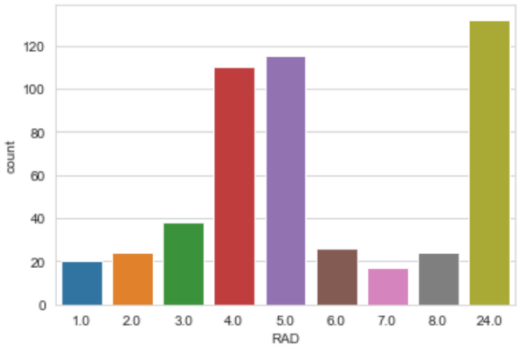
In [17]:
```
# Checking the null values using heatmap
sns.heatmap(data.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[17]: <AxesSubplot:>



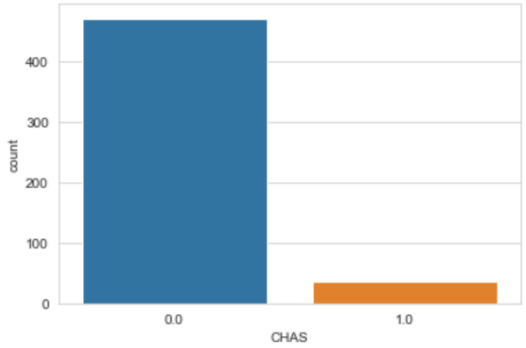
In [18]:
```
sns.set_style('whitegrid')
sns.countplot(x='RAD',data=data)
```

Out[18]: <AxesSubplot:xlabel='RAD', ylabel='count'>



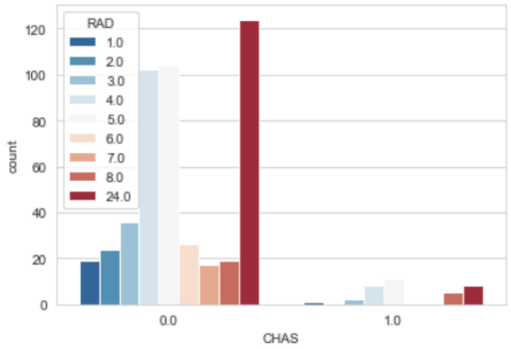
In [19]:
```
sns.set_style('whitegrid')
sns.countplot(x='CHAS',data=data)
```

Out[19]: <AxesSubplot:xlabel='CHAS', ylabel='count'>



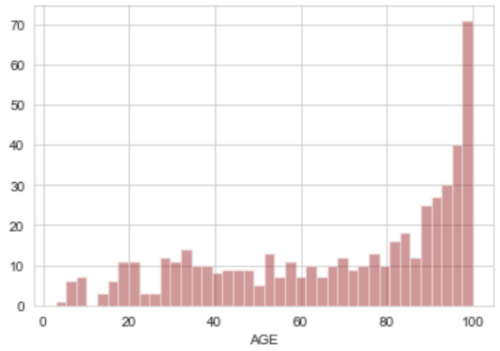
In [20]:
```
sns.set_style('whitegrid')
sns.countplot(x='CHAS',hue='RAD',data=data,palette='RdBu_r')
```

Out[20]: <AxesSubplot:xlabel='CHAS', ylabel='count'>



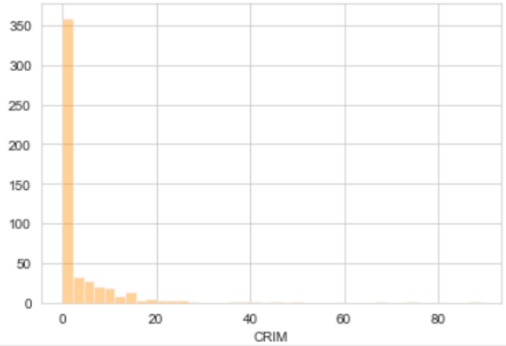
In [21]:
```
sns.distplot(data['AGE'].dropna(),kde=False,color='darkred',bins=40)
```

Out[21]: <AxesSubplot:xlabel='AGE'>



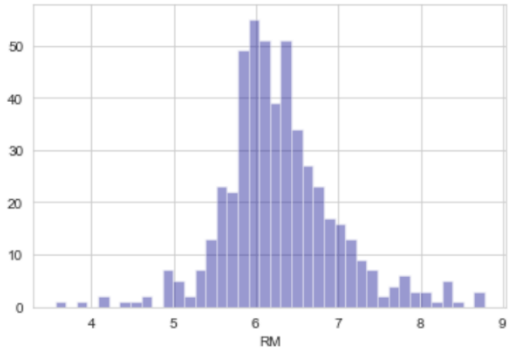
In [22]:
```
sns.distplot(data['CRIM'].dropna(),kde=False,color='darkorange',bins=40)
```

Out[22]: <AxesSubplot:xlabel='CRIM'>



In [23]:
```
sns.distplot(data['RM'].dropna(),kde=False,color='darkblue',bins=40)
```

Out[23]: <AxesSubplot:xlabel='RM'>



4.5 MODEL FITTING

4.5.1 LINEAR REGRESSION

In [24]:
```
# Spliting target variable and independent variables
X = data.drop(['PRICE'], axis = 1)
y = data['PRICE']
```

In [25]:
```
# Splitting to training and testing data

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.3, random_state = 4)
```

In [26]:
```
# Import library for Linear Regression
from sklearn.linear_model import LinearRegression

# Create a Linear regressor
lm = LinearRegression()

# Train the model using the training sets
lm.fit(X_train, y_train)
```

Out[26]:
```
LinearRegression()
```

In [27]:
```
# Value of y intercept
lm.intercept_
```

Out[27]:
```
36.357041376595205
```

In [28]:
```
#Converting the coefficient values to a dataframe
coeffcients = pd.DataFrame([X_train.columns,lm.coef_]).T
coeffcients = coeffcients.rename(columns={0: 'Attribute', 1: 'Coefficients'})
coeffcients
```

Out[28]:

	Attribute	Coefficients		Attribute	Coefficients
0	CRIM	-0.12257	7	DIS	-1.552144
1	ZN	0.055678	8	RAD	0.32625
2	INDUS	-0.008834	9	TAX	-0.014067
3	CHAS	4.693448	10	PTRATIO	-0.803275
4	NOX	-14.435783	11	B	0.009354
5	RM	3.28008	12	LSTAT	-0.523478
6	AGE	-0.003448			

4.5.2 RANDOM FOREST REGRESSOR

In [36]:
```
from sklearn.ensemble import RandomForestRegressor
reg = RandomForestRegressor()
reg.fit(X_train, y_train)
```

Out[36]:
```
RandomForestRegressor()
```

In [37]:
```
y_pred = reg.predict(X_train)
```

4.6 MODEL EVALUATION

4.6.1 LINEAR REGRESSION

Training Data

In [29]:
```
# Model prediction on train data
y_pred = lm.predict(X_train)
```

In [30]:
```
# Model Evaluation
print('R^2:',metrics.r2_score(y_train, y_pred))
print('Adjusted R^2:',1 - (1-metrics.r2_score(y_train, y_pred))*(len(y_train)-1)/(len(y_train)-X_train.
print('MAE:',metrics.mean_absolute_error(y_train, y_pred))
print('MSE:',metrics.mean_squared_error(y_train, y_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_train, y_pred)))
```

```
R^2: 0.7465991966746854
Adjusted R^2: 0.736910342429894
MAE: 3.08986109497113
MSE: 19.07368870346903
RMSE: 4.367343437774162
```

Testing Data

In [31]:
```
# Predicting Test data with the model
y_test_pred = lm.predict(X_test)
```

In [32]:
```
# Model Evaluation
acc_linreg = metrics.r2_score(y_test, y_test_pred)
print('R^2:', acc_linreg)
print('Adjusted R^2:',1 - (1-metrics.r2_score(y_test, y_test_pred))*(len(y_test)-1)/(len(y_test)-X_test
print('MAE:',metrics.mean_absolute_error(y_test, y_test_pred))
print('MSE:',metrics.mean_squared_error(y_test, y_test_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test, y_test_pred)))
```

```
R^2: 0.7121818377409195
Adjusted R^2: 0.6850685326005713
MAE: 3.8590055923707407
MSE: 30.053993307124127
RMSE: 5.482152251362974
```

4.6.2 RANDOM FOREST REGRESSOR

Training Data

In [38]:
```
# Model Evaluation
print('R^2:',metrics.r2_score(y_train, y_pred))
print('Adjusted R^2:',1 - (1-metrics.r2_score(y_train, y_pred))*(len(y_train)-1)/(len(y_train)-X_train.
print('MAE:',metrics.mean_absolute_error(y_train, y_pred))
print('MSE:',metrics.mean_squared_error(y_train, y_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_train, y_pred)))
```

```
R^2: 0.9775118374899715
Adjusted R^2: 0.9766519959822351
MAE: 0.8474689265536711
MSE: 1.6927026497175144
RMSE: 1.3010390654079202
```

Testing Data

In [39]:
```
# Predicting Test data with the model
y_test_pred = reg.predict(X_test)
```

In [40]:
```
# Model Evaluation
acc_rf = metrics.r2_score(y_test, y_test_pred)
print('R^2:', acc_rf)
print('Adjusted R^2:',1 - (1-metrics.r2_score(y_test, y_test_pred))*(len(y_test)-1)/(len(y_test)-X_test
print('MAE:',metrics.mean_absolute_error(y_test, y_test_pred))
print('MSE:',metrics.mean_squared_error(y_test, y_test_pred))
print('RMSE:',np.sqrt(metrics.mean_squared_error(y_test, y_test_pred)))
```

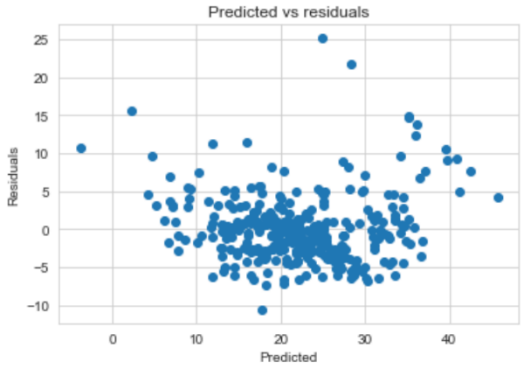
```
R^2: 0.8334269669497434
Adjusted R^2: 0.8177353044160236
MAE: 2.541190789473683
MSE: 17.393568151315783
RMSE: 4.170559692812918
```

4.7 MOEDL VISUALIZATION

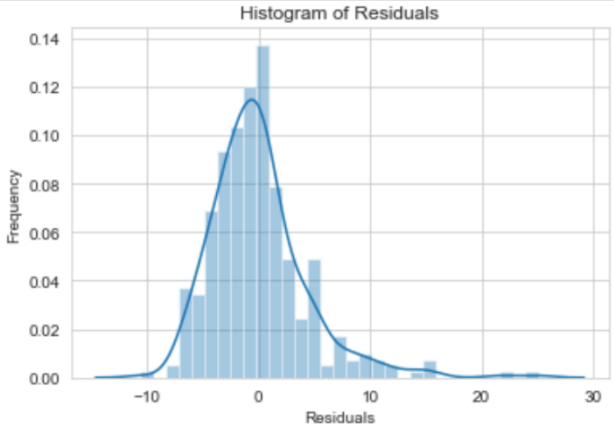
4.7.1 LINEAR REGRESSION

In [33]:
```
# Visualizing the differences between actual prices and predicted values
plt.scatter(y_train, y_pred)
plt.xlabel("Prices")
plt.ylabel("Predicted prices")
plt.title("Prices vs Predicted prices")
plt.show()
```

In [34]:
```
# Checking residuals
plt.scatter(y_pred,y_train-y_pred)
plt.title("Predicted vs residuals")
plt.xlabel("Predicted")
plt.ylabel("Residuals")
plt.show()
```



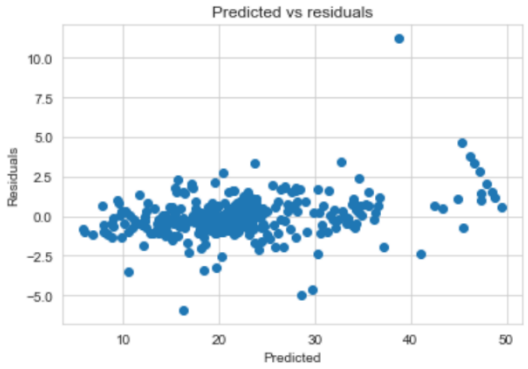
In [35]:
```
# Checking Normality of errors
sns.distplot(y_train-y_pred)
plt.title("Histogram of Residuals")
plt.xlabel("Residuals")
plt.ylabel("Frequency")
plt.show()
```



4.7.2 RANDOM FOREST REGRESSOR

In [41]:
```
# Visualizing the differences between actual prices and predicted values
plt.scatter(y_train, y_pred)
plt.xlabel("Prices")
plt.ylabel("Predicted prices")
plt.title("Prices vs Predicted prices")
plt.show()
```

In [42]:
```
# Checking residuals
plt.scatter(y_pred,y_train-y_pred)
plt.title("Predicted vs residuals")
plt.xlabel("Predicted")
plt.ylabel("Residuals")
plt.show()
```



4.8 MODEL DEPLOYMENT(GUI)

In [43]:

```
def Predict(e1,e2,e3,e4,e5,e6,e7,e8,e9,e10,e11,e12,e13,lbl_result1):
    f1=float(e1.get())

    f2=float(e2.get())

    f3=float(e3.get())

    f4=float(e4.get())

    f5=float(e5.get())

    f6=float(e6.get())

    f7=float(e7.get())

    f8=float(e8.get())

    f9=float(e9.get())

    f10=float(e10.get())

    f11=float(e11.get())

    f12=float(e12.get())

    f13=float(e13.get())

    res=np.array([[f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13]])

    pred_price="$"+str(reg.predict(res)*1000)
    np.set_printoptions(precision=2)

    lbl_result1.configure(text=pred_price,font=('Lucida Handwriting',25,'bold'))
```

In [80]:

```
def Help():
    global pop
    pop=Toplevel(root)
    pop.title("HELP")
    pop.config(bg="#91b3bd")

    pop_label1=Label(pop,text="""
CRIM    ->  per capita crime rate by town \n
ZN      ->  proportion of residential land zoned for lots over 25,000 sq.ft.\n
INDUS   ->  proportion of non-retail business acres per town\n
CHAS    ->  Charles River dummy variable \n
NOX     ->  nitric oxides concentration (parts per 10 million)\n
RM      ->  average number of rooms per dwelling\n
AGE     ->  proportion of owner-occupied units built prior to 1940\n
DIS     ->  weighted distances to five Boston employment centres\n
RAD     ->  index of accessibility to radial highways\n
TAX     -> full-value property-tax rate per $10,000\n
PTRATIO->  pupil-teacher ratio by town\n
B       ->  1000(Bk - 0.63) ^2 where Bk is the proportion of blacks by town\n
LSTAT   ->  % lower status of the population\n
""",fg="black",bg="#91b3bd",font=("helvetica",12),justify= LEFT)
    pop_label1.pack()
```

In [81]:

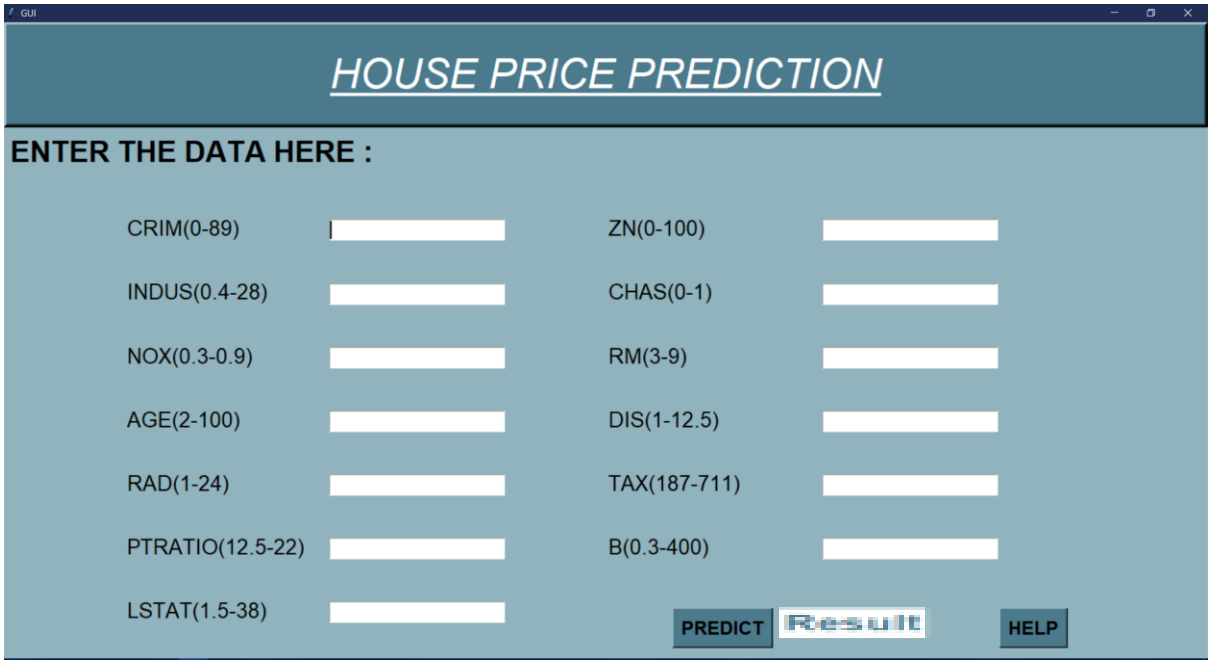
```
root =Tk()
root.title("GUI")
root.state('zoomed')
root.resizable(width = False,height = True)
root.configure(bg = "White")

def home_screen():
    MainFrame = Frame(root, bg = "#91b3bd")
    MainFrame.pack()
    root_lbl = Label(MainFrame,text = "HOUSE PRICE PREDICTION",bg = "#4A7A8C", fg = "White",font = ('',
    root_lbl.pack(anchor = N,expand = True)

    q_frame = Frame(MainFrame,bg = '#91b3bd', width = 1400, height=800, relief='flat', borderwidth=5)
    q_frame.pack(side = LEFT,anchor = W,expand = True)

    br_file = Label(q_frame,text = "ENTER THE DATA HERE :",bg = "#91b3bd",font = ('',28,'bold'),relief
    br_file.place(relx = 0.001,rely = 0.001)


    Label(MainFrame, text='CRIM(0-89)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0.3)
    e1 = Entry(MainFrame,font=('bold',15))
    e1.place(relx = 0.27, rely = 0.31)
    e1.focus()

    Label(MainFrame, text='ZN(0-100)',font=('bold',20),bg = "#91b3bd").place(relx = 0.5,rely = 0.3)
    e2= Entry(MainFrame,font=('bold',15))
    e2.place(relx = 0.68, rely = 0.31)
    e2.focus()


    Label(MainFrame, text='INDUS(0.4-28)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0.4)
    e3 = Entry(MainFrame,font=('bold',15))
    e3.place(relx = 0.27, rely = 0.41)
    e3.focus()



    Label(MainFrame, text='CHAS(0-1)',font=('bold',20),bg = "#91b3bd").place(relx = 0.5,rely = 0.4)
    e4 = Entry(MainFrame,font=('bold',15))
    e4.place(relx = 0.68,rely = 0.41)
    e4.focus()


    Label(MainFrame, text='NOX(0.3-0.9)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0.5)
    e5 = Entry(MainFrame,font=('bold',15))
    e5.place(relx = 0.27, rely = 0.51)
    e5.focus()

    Label(MainFrame, text='RM(3-9)',font=('bold',20),bg = "#91b3bd").place(relx = 0.5,rely = 0.5)
    e6 = Entry(MainFrame,font=('bold',15))
    e6.place(relx = 0.68, rely = 0.51)
    e6.focus()


    Label(MainFrame, text='AGE(2-100)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0.6)
    e7 = Entry(MainFrame,font=('bold',15))
    e7.place(relx = 0.27, rely = 0.61)
    e7.focus()



    Label(MainFrame, text='DIS(1-12.5)',font=('bold',20),bg = "#91b3bd").place(relx = 0.5,rely = 0.6)
    e8 = Entry(MainFrame,font=('bold',15))
    e8.place(relx = 0.68, rely = 0.61)
    e8.focus()



    Label(MainFrame, text='RAD(1-24)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0.7)
    e9 = Entry(MainFrame,font=('bold',15))
    e9.place(relx = 0.27, rely = 0.71)
    e9.focus()
```

```
    Label(MainFrame, text='TAX(187-711)',font=('bold',20),bg = "#91b3bd").place(relx = 0.5,rely = 0.7)
    e10 = Entry(MainFrame,font=('bold',15))
    e10.place(relx = 0.68, rely = 0.71)
    e10.focus()

    Label(MainFrame, text='PTRATIO(12.5-22)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0
    e11 = Entry(MainFrame,font=('bold',15))
    e11.place(relx = 0.27, rely = 0.81)
    e11.focus()

    Label(MainFrame, text='B(0.3-400)',font=('bold',20),bg = "#91b3bd").place(relx = 0.5,rely = 0.8)
    e12 = Entry(MainFrame,font=('bold',15))
    e12.place(relx = 0.68, rely = 0.81)
    e12.focus()

    Label(MainFrame, text='LSTAT(1.5-38)',font=('bold',20),bg = "#91b3bd").place(relx = 0.1,rely = 0.9)
    e13 = Entry(MainFrame,font=('bold',15))
    e13.place(relx = 0.27, rely = 0.91)
    e13.focus()

    predict_btn = Button(q_frame,text = "PREDICT",command=lambda: Predict(e1,e2,e3,e4,e5,e6,e7,e8,e9,e1
    predict_btn.place(relx = 0.61,rely = 0.91)

    lbl_result1=Label(q_frame,text='Price in $1000',font=('',20,'bold'),bg='White',fg='#4A7A8C')
    lbl_result1.place(relx=0.71,rely=0.91)

    help_btn = Button(q_frame,text = "HELP",command = lambda:Help(),bg = "#4A7A8C",font = ('',18,'bold'
    help_btn.place(relx = 0.91,rely = 0.91)

home_screen()
root.mainloop()
```

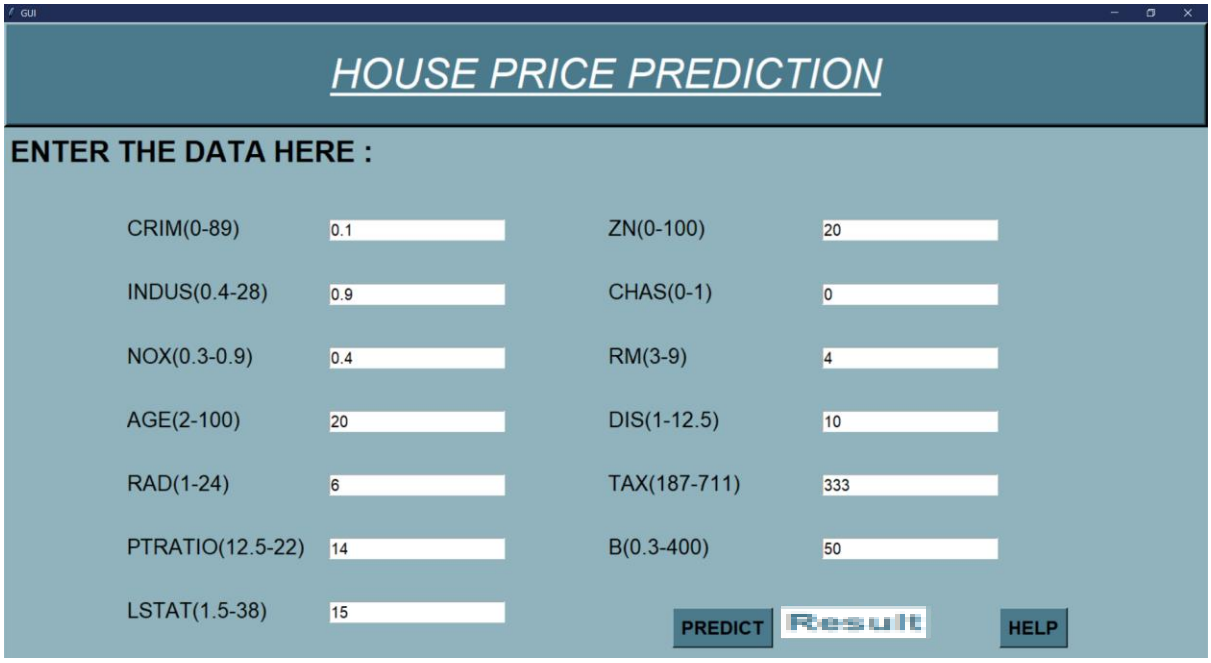
CHAPTER 5

OUTPUT AND RESULT

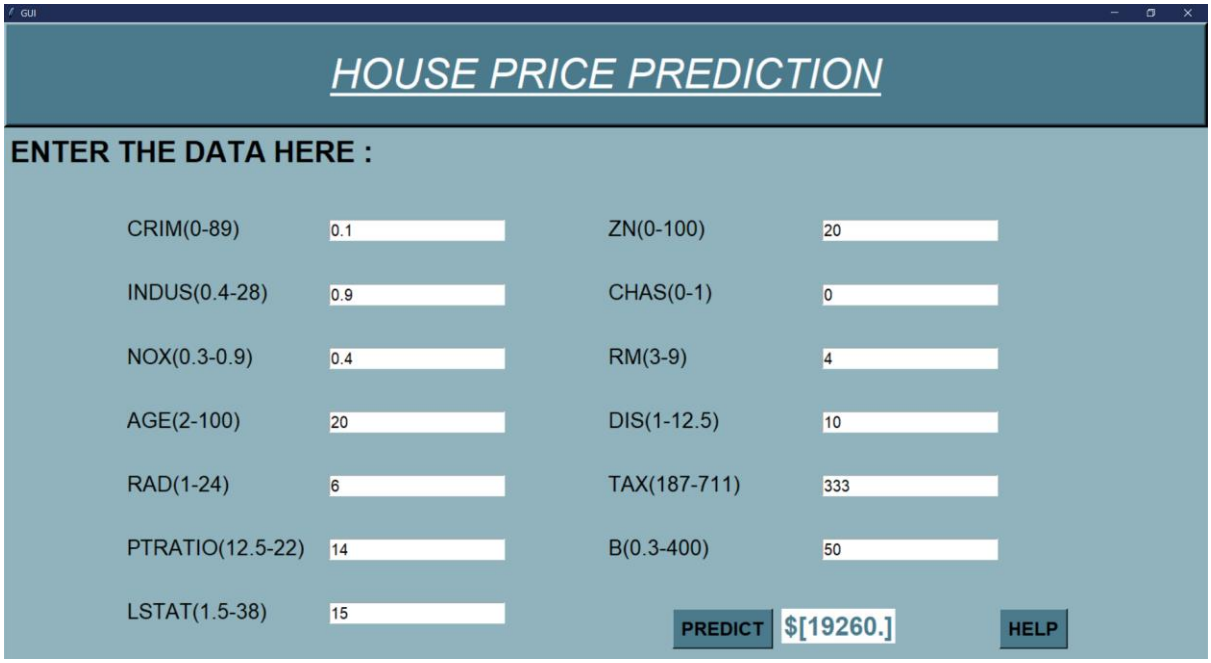
5.1 GUI



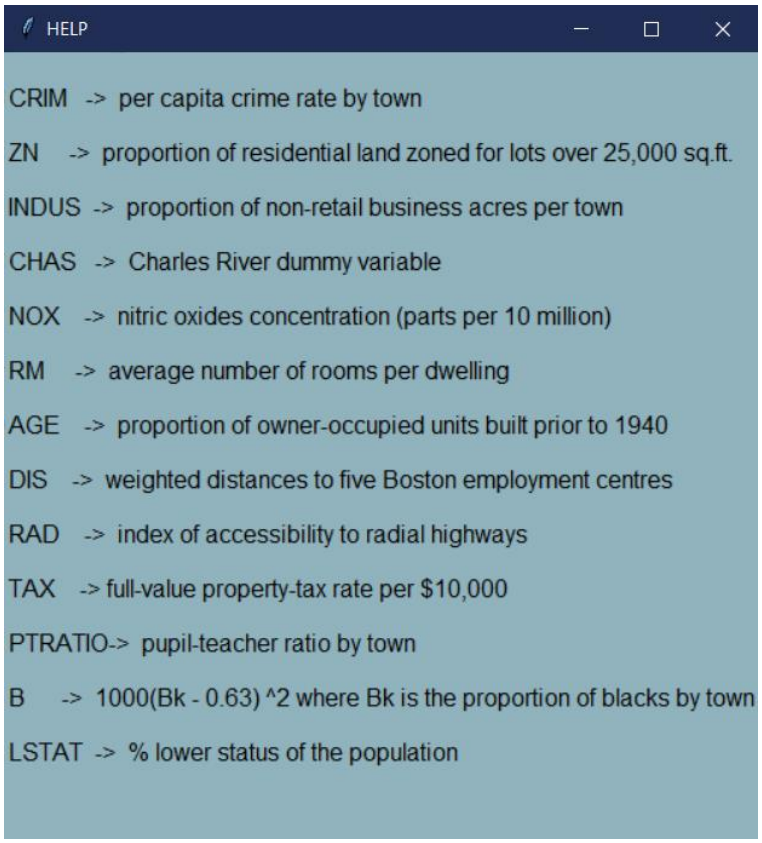
5.2 ENTERING THE DATA



5.3 PREDICTING THE PRICE OF HOUSE



5.4 HELP BUTTON



5.5 RESULT

	Model	Linear Regression	Random Forest Regressor
Training Data	**R Squared**	0.7465	0.9775
	Adjusted R Squared	0.7369	0.9766
	MAE	3.0898	0.8474
	MSE	19.073	1.6927
	RMSE	4.3673	1.3010
Testing Data	**R Squared**	0.7121	0.8334
	Adjusted R Squared	0.6850	0.8177
	MAE	3.8590	2.5411
	MSE	30.0539	17.3935
	RMSE	5.4821	4.1705

Fig 5.5.1 Final Result Table

CHAPTER 6

DATA VISUALIZATION USING TABLEAU

All the features are visualized against the target i.e., Price in $1000 using Cluster and Trend Line to understand the data and its trend.

6.1 AGE vs PRICE IN $1000

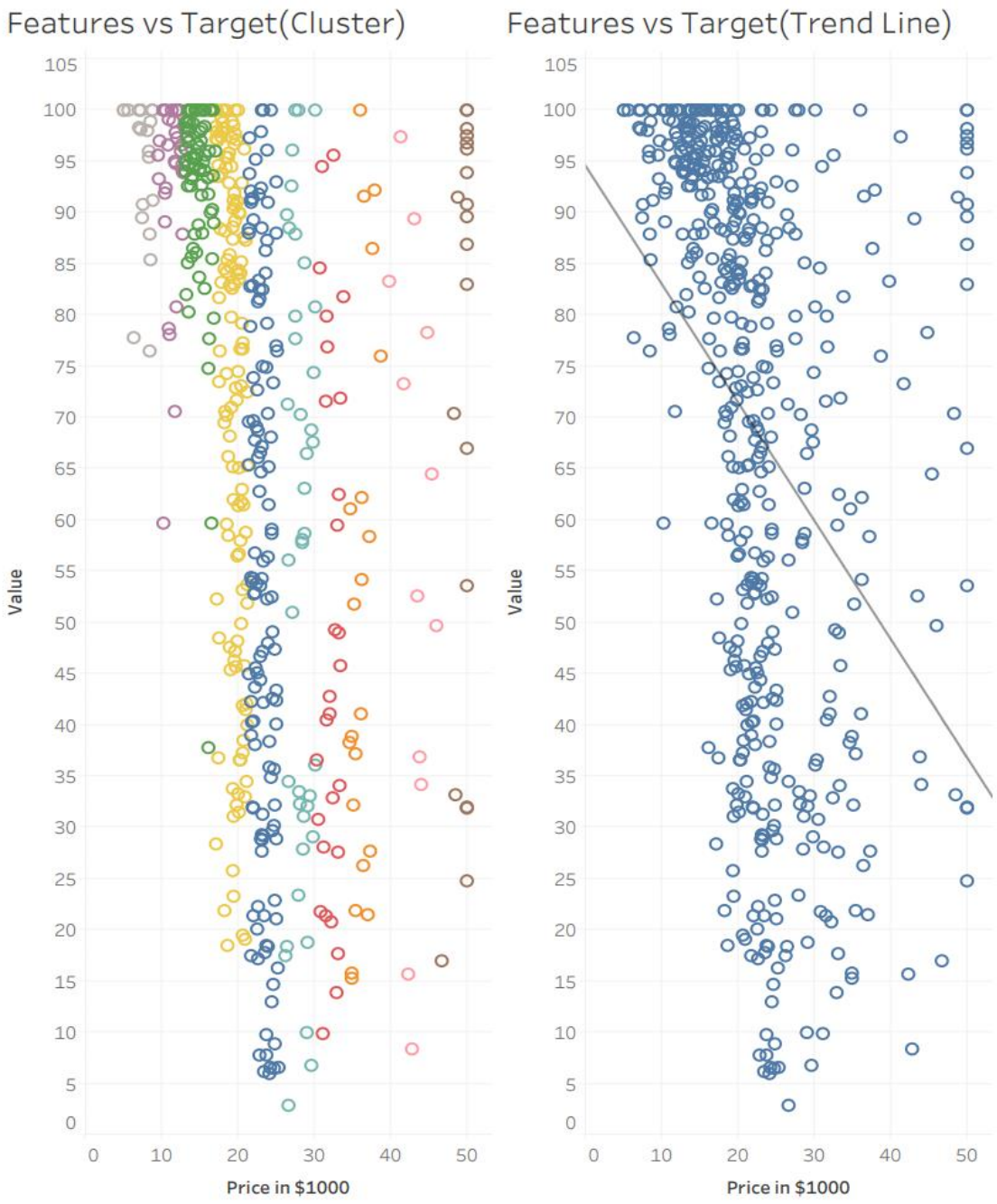


Fig 6.1.1

6.2 B vs PRICE IN $1000

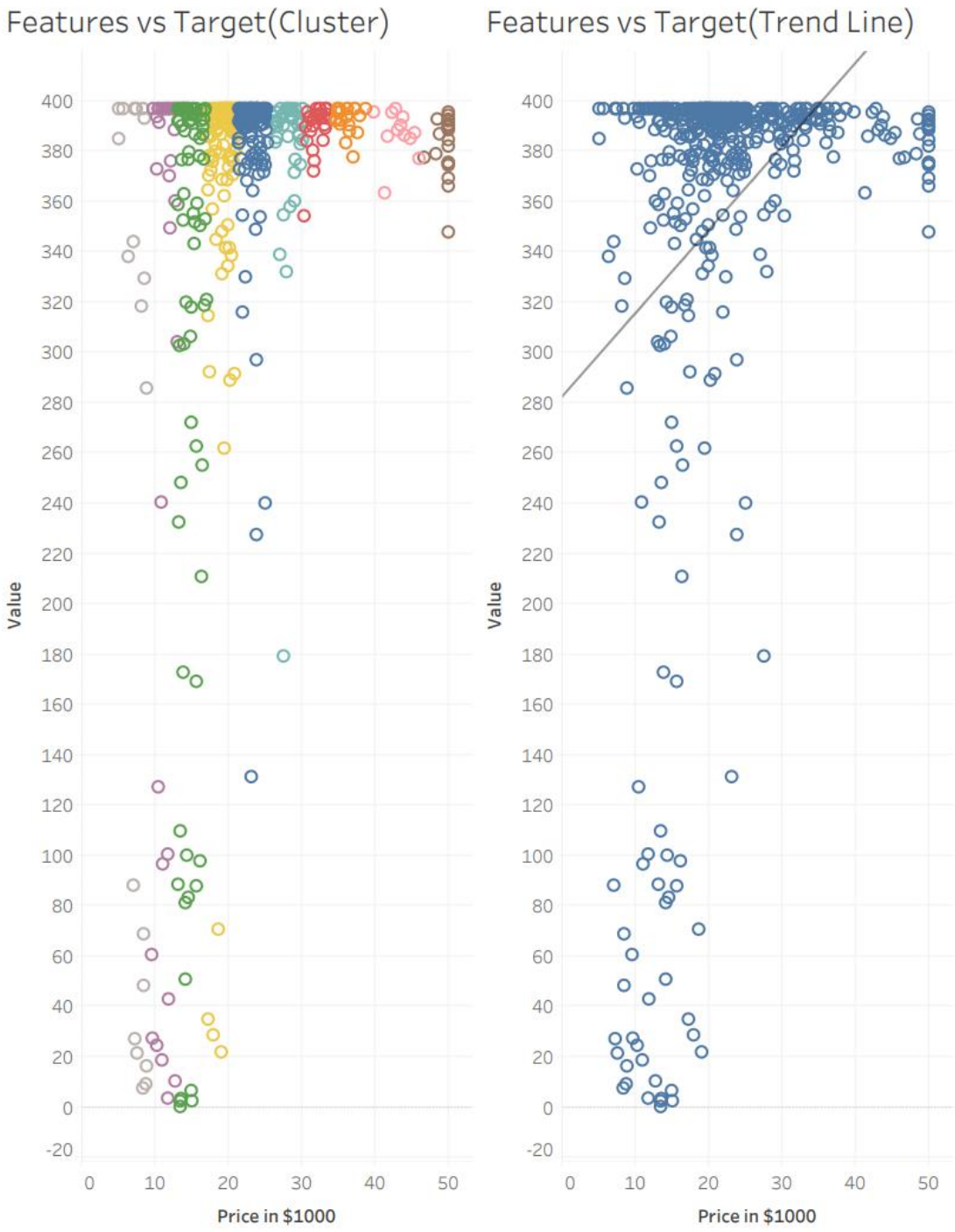


Fig 6.2.1

6.3 CHAS vs PRICE IN $1000

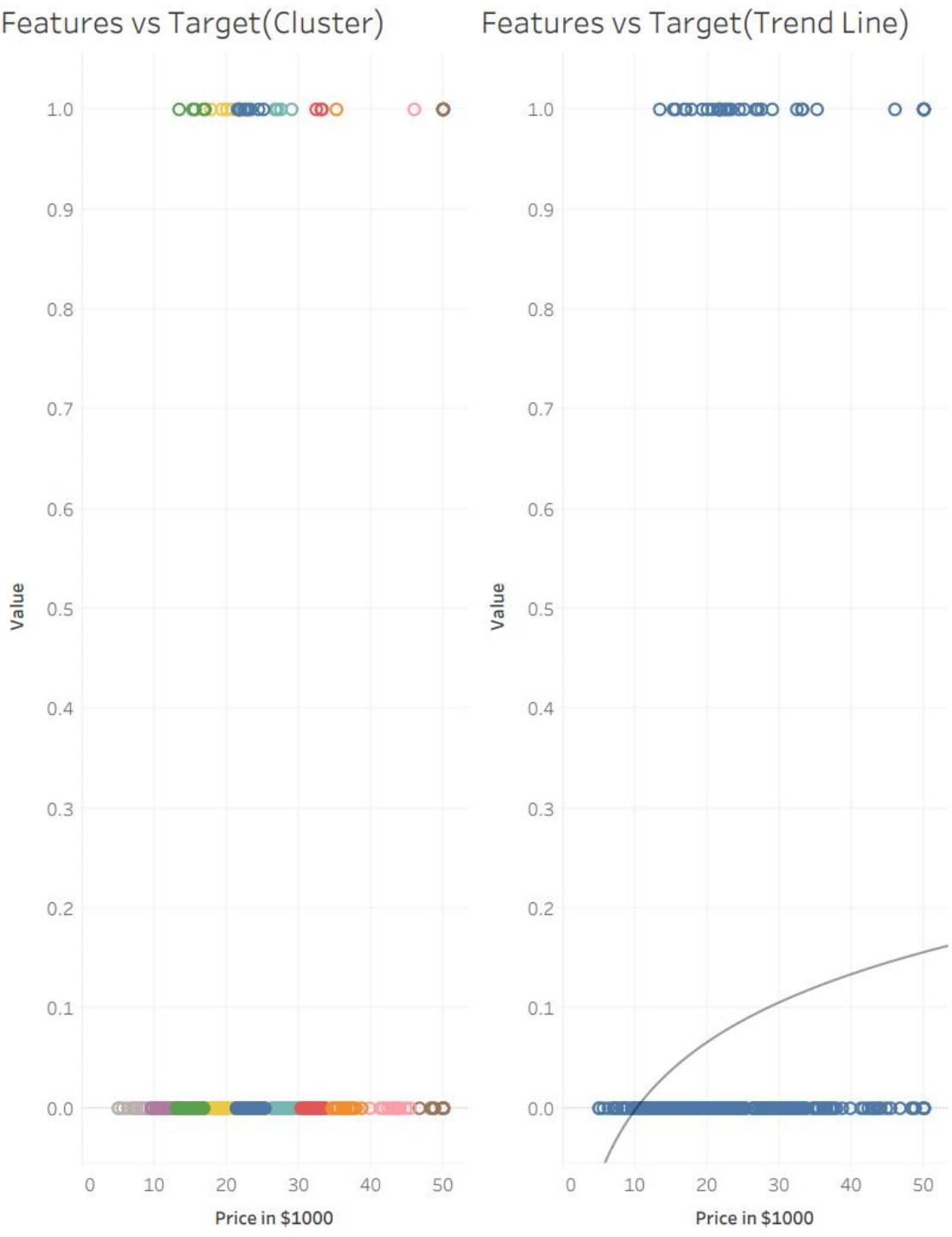


Fig 6.3.1

6.4 CRIM vs PRICE IN $1000

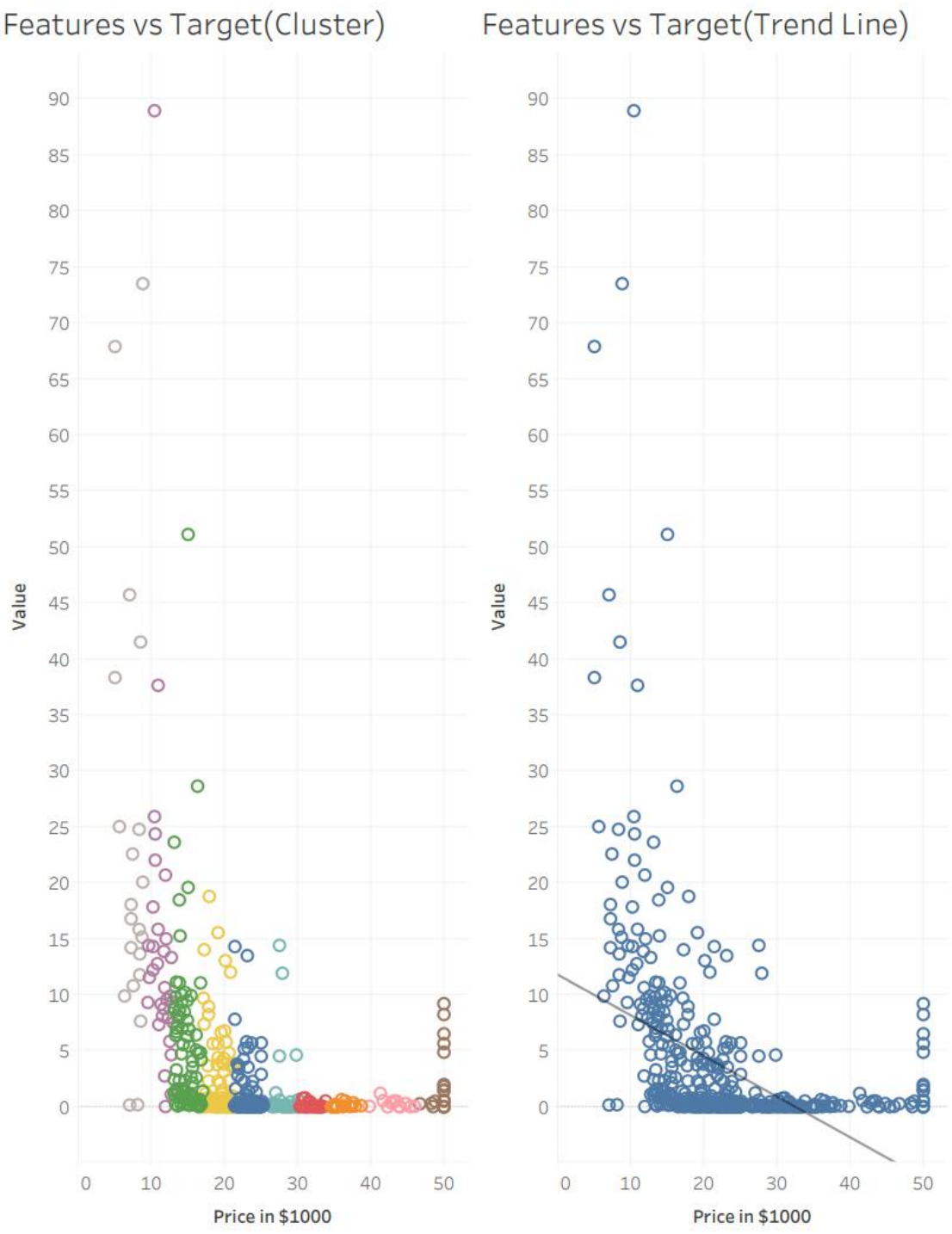


Fig 6.4.1

6.5 DIS vs PRICE IN $1000

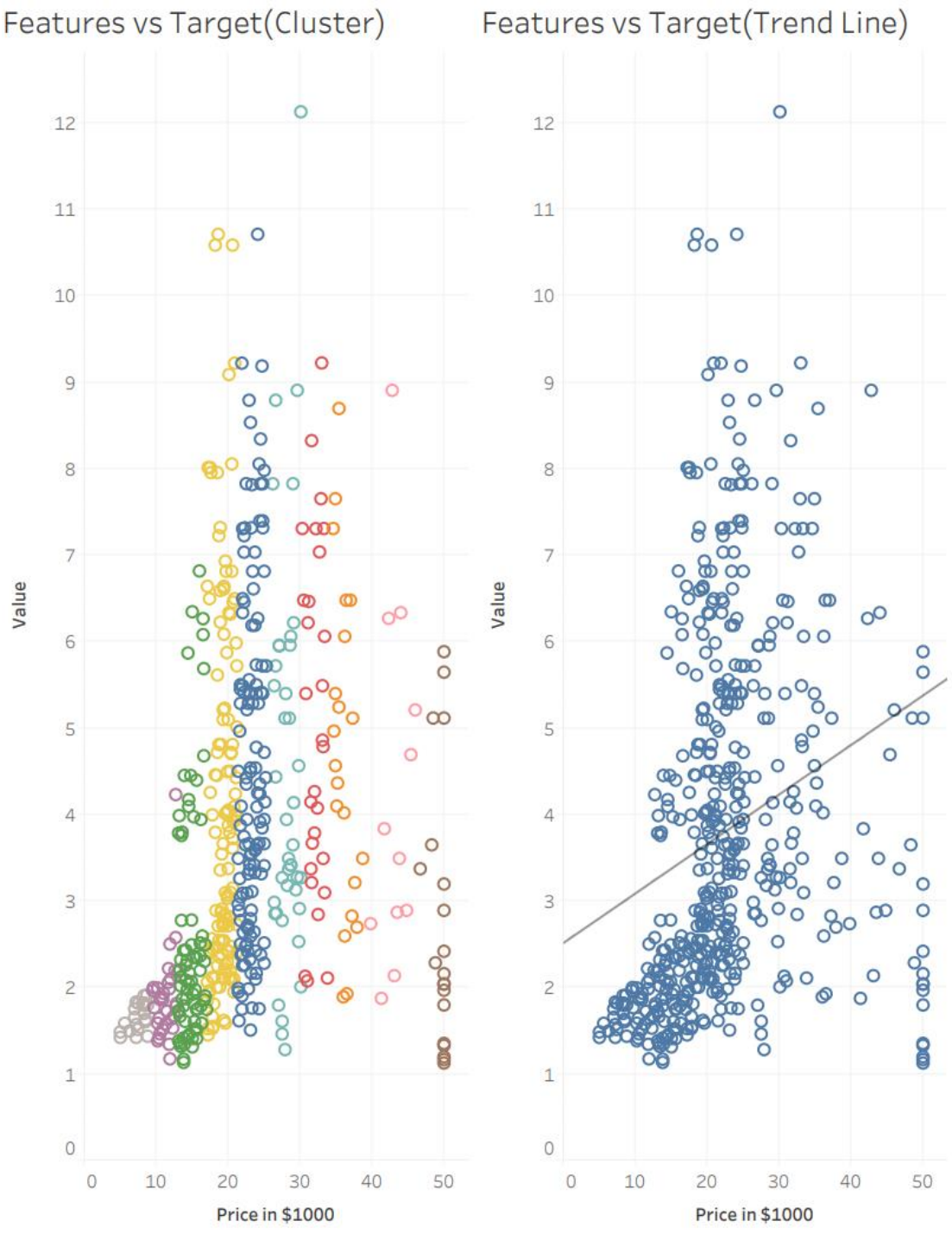


Fig 6.5.1

6.6 INDUS vs PRICE IN $1000

Fig 6.6.1

6.7 LSTAT vs PRICE IN $1000

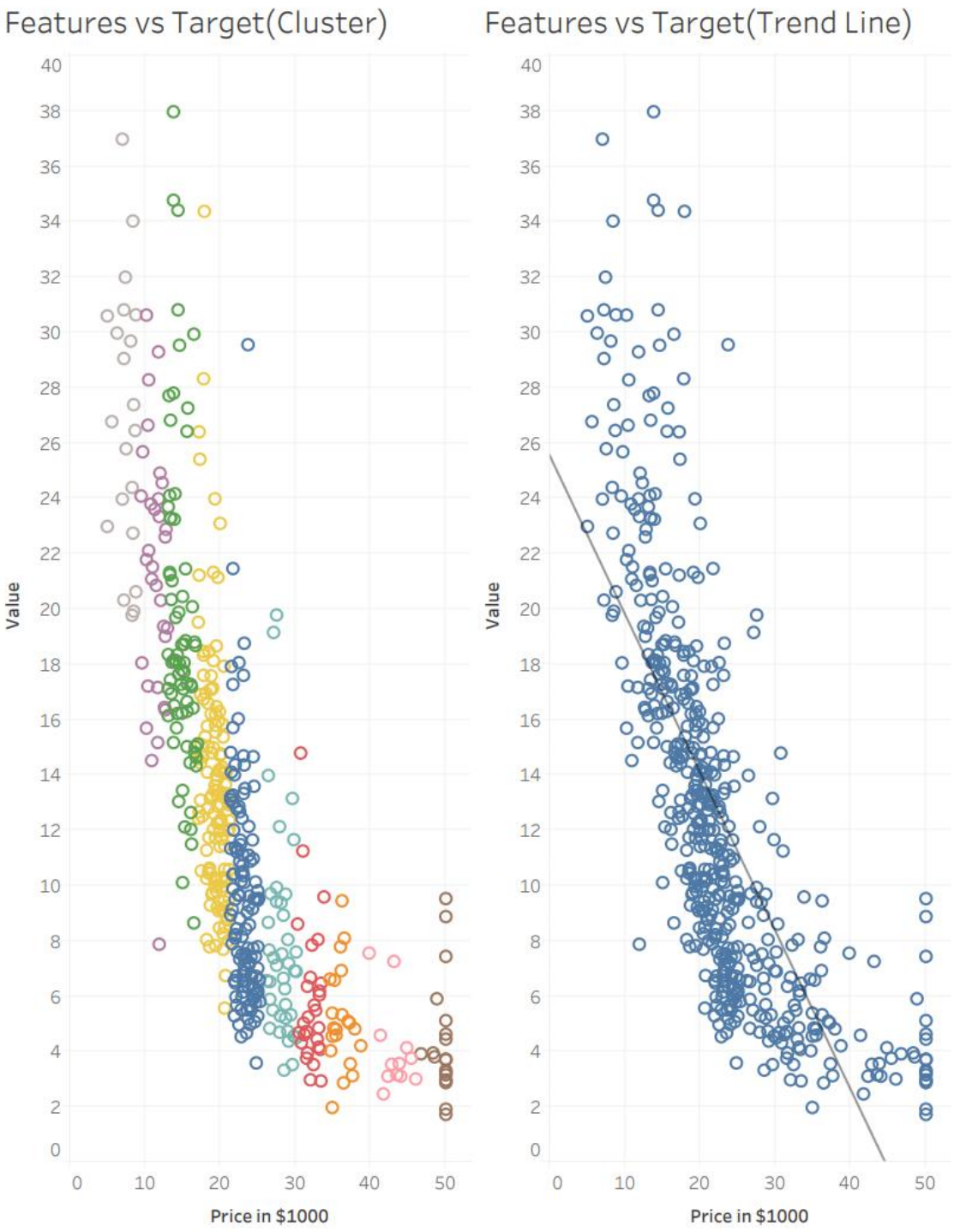


Fig 6.7.1

6.8 NOX vs PRICE IN $1000

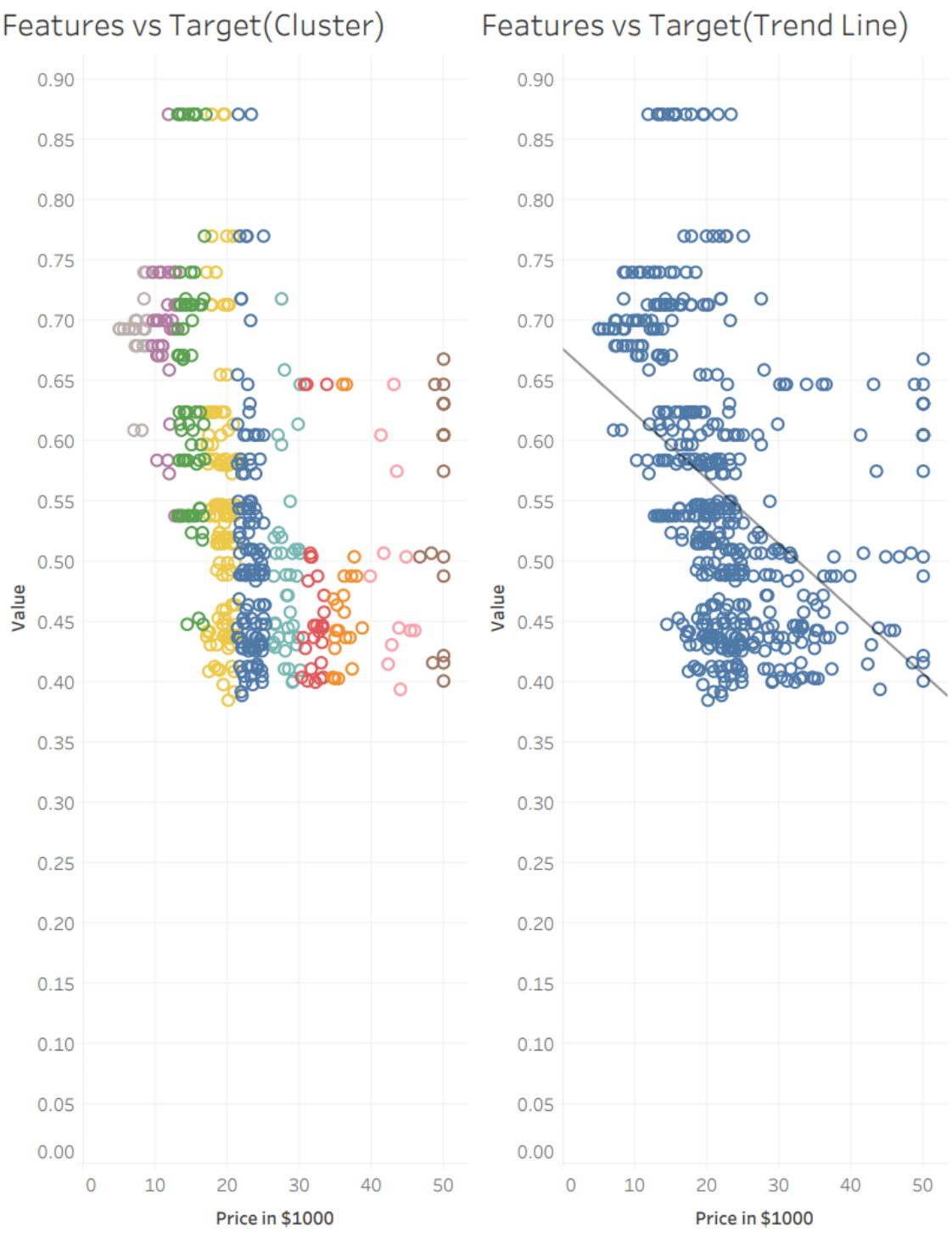


Fig 6.8.1

6.9 PTRATIO vs PRICE IN $1000

Fig 6.9.1

6.10 RAD vs PRICE IN $1000

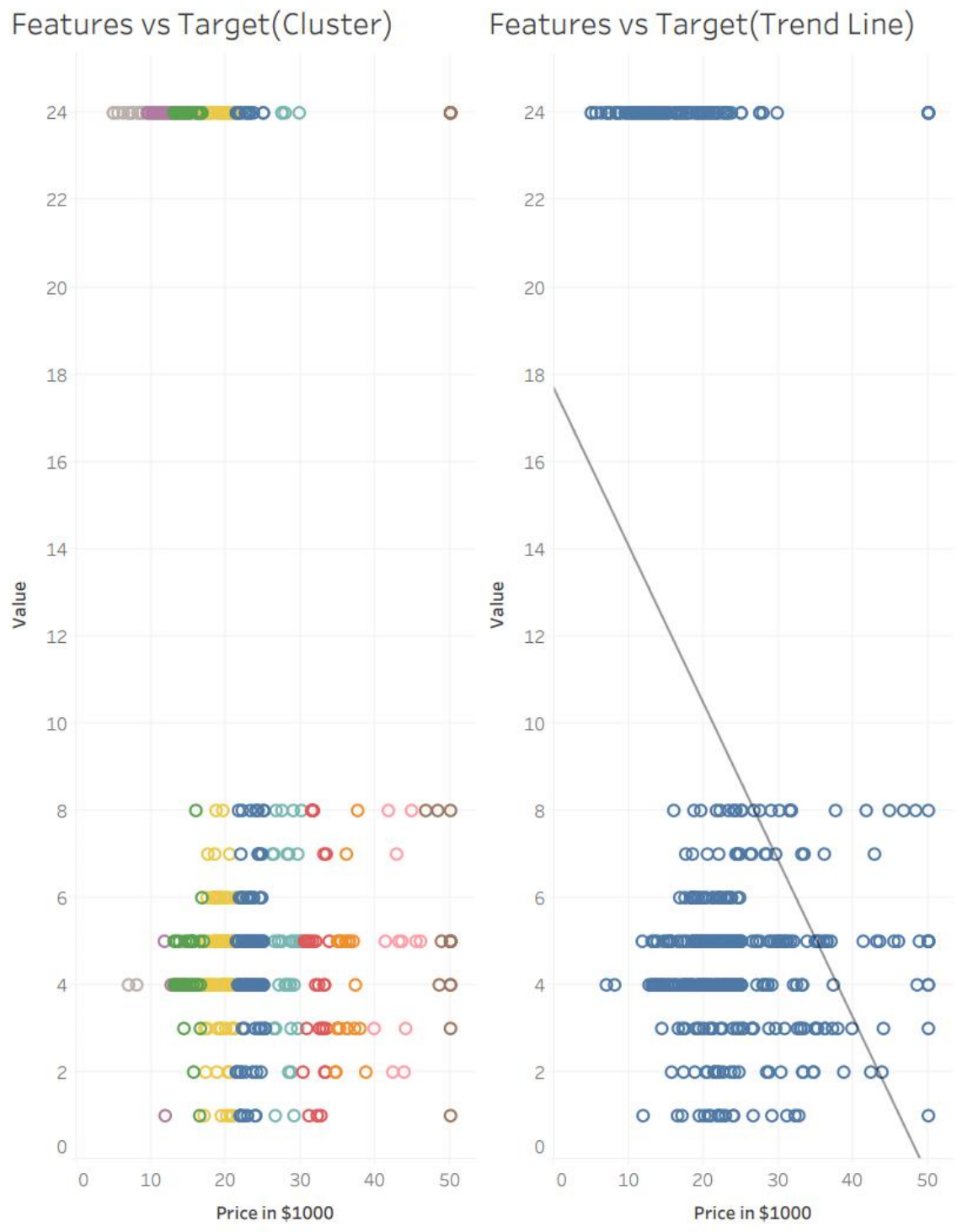


Fig 6.10.1

6.11 RM vs PRICE IN $1000

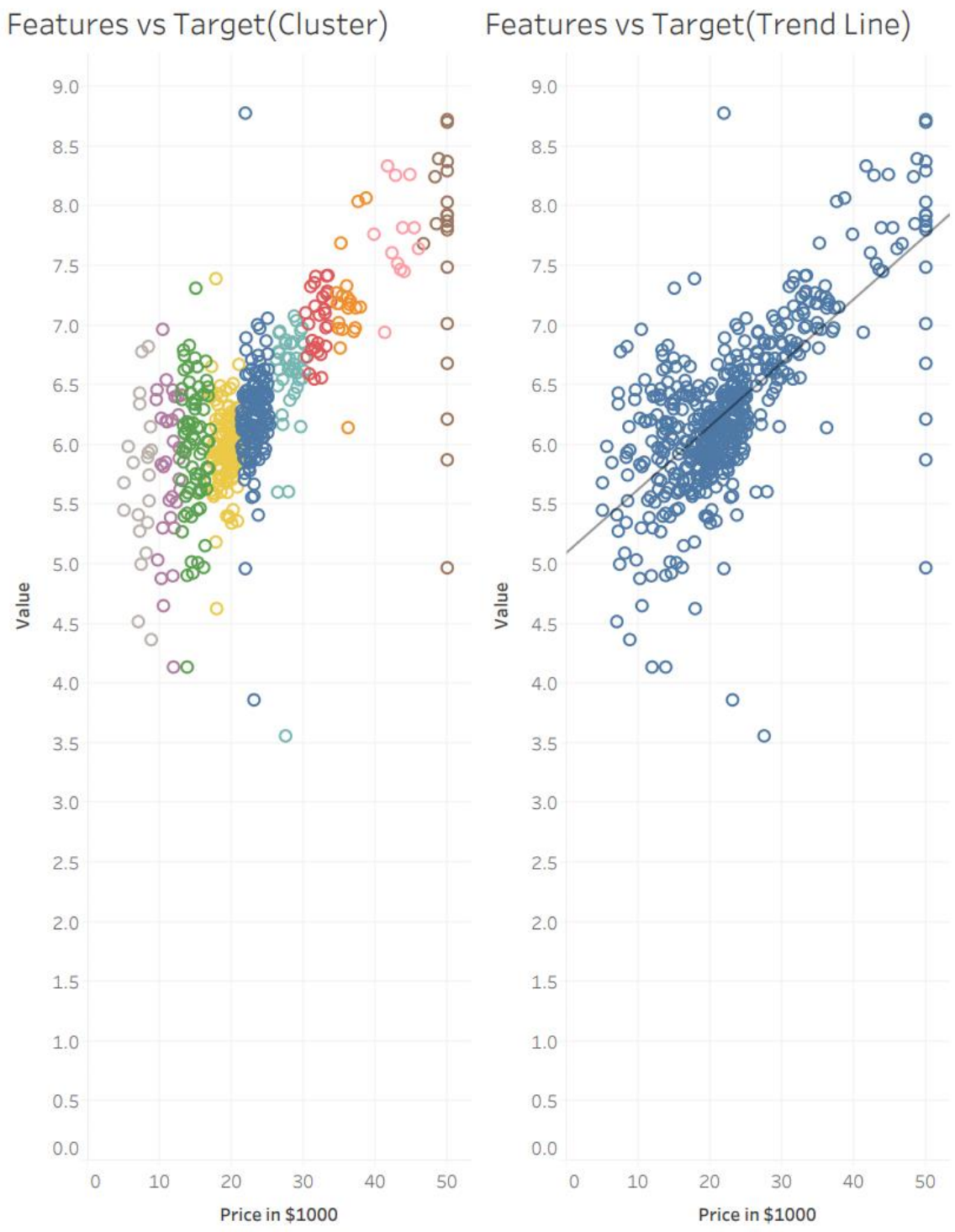


Fig 6.11.1

6.12 TAX vs PRICE IN $1000

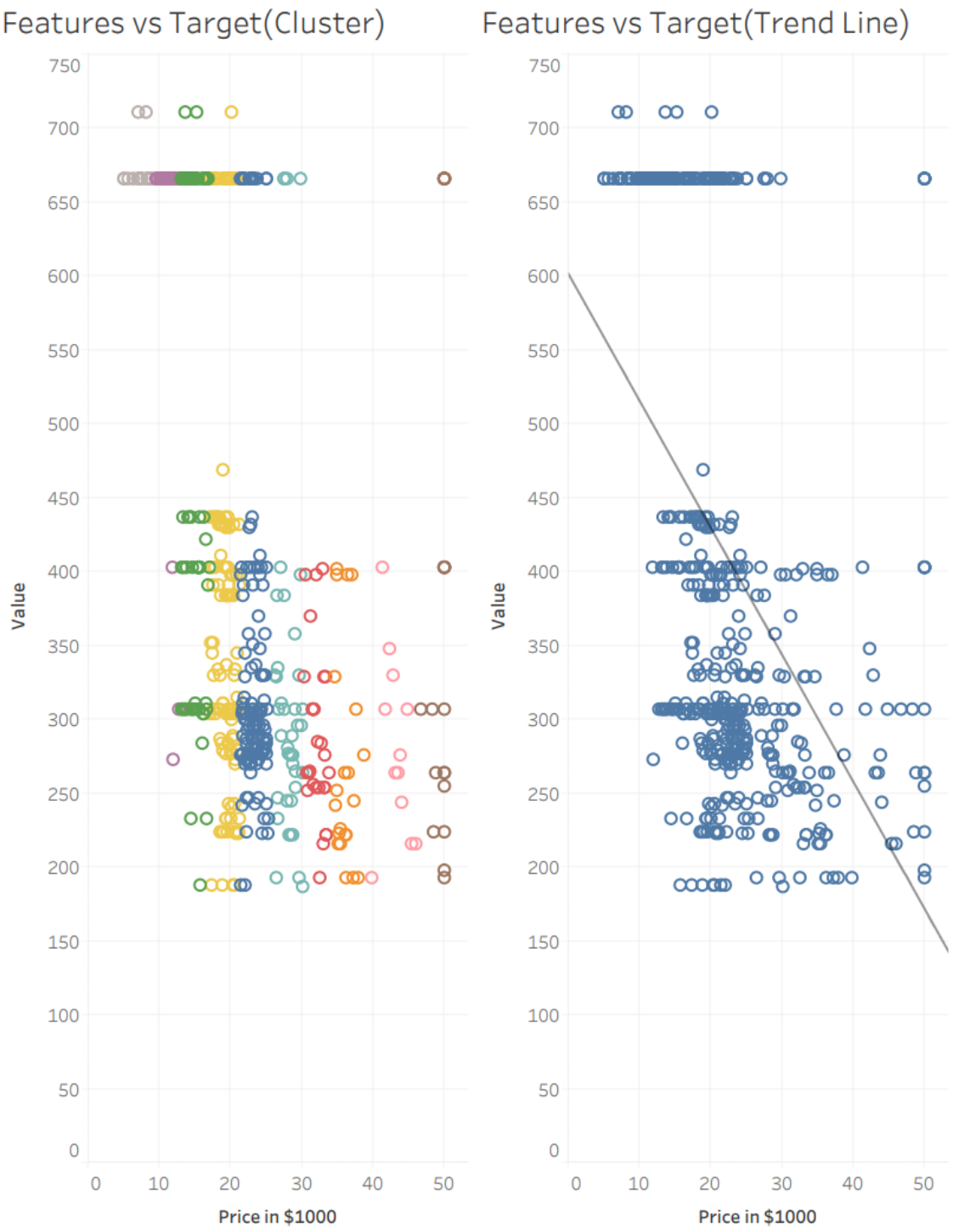


Fig 6.12.1

6.13 ZN vs PRICE IN $1000

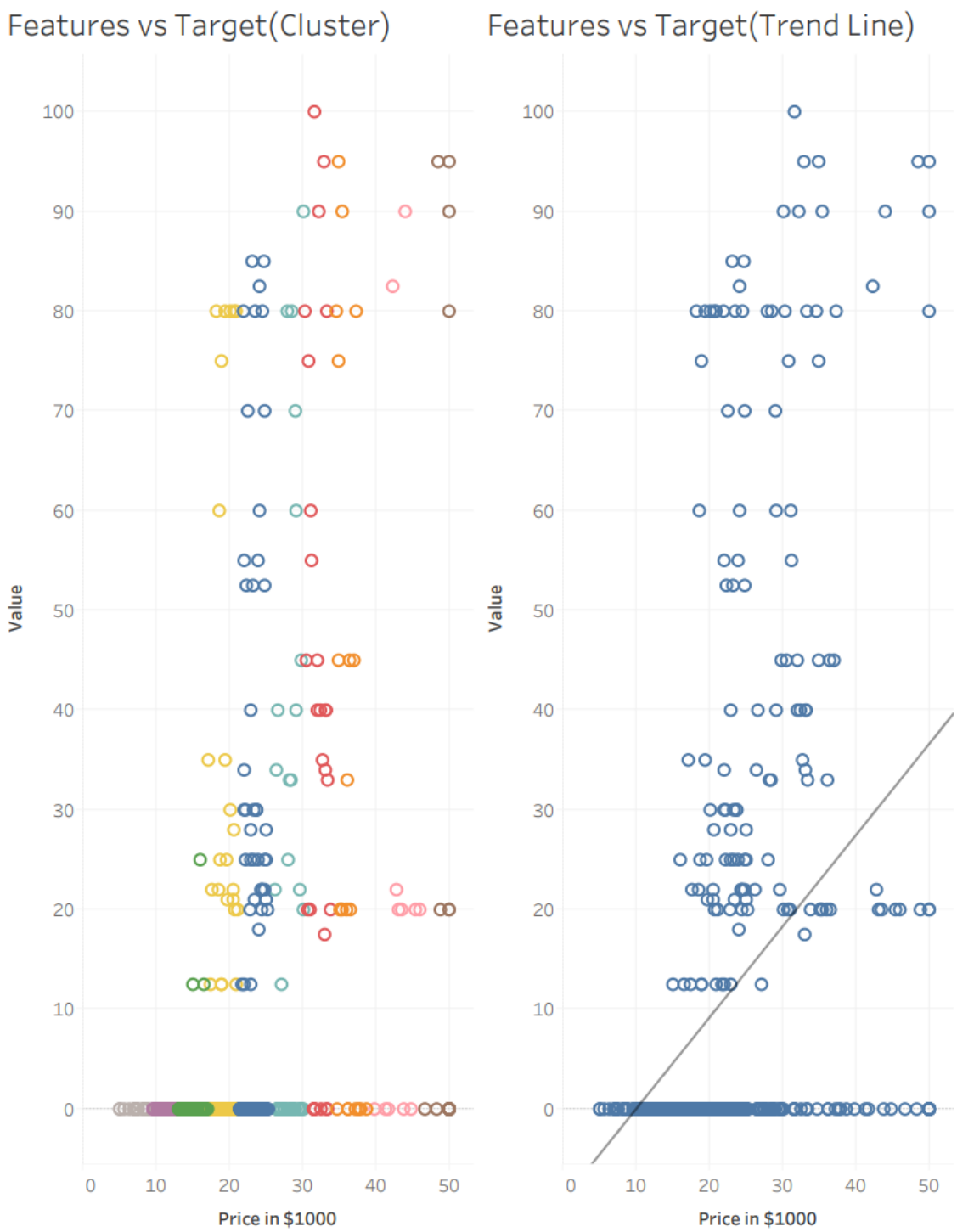


Fig 6.13.1

CHAPTER 7

ANALYSIS

According to the visualizations in chapter 6, here are some results based on every feature trend.

Age vs Price in $1000	**Downtrend**
B vs Price in $1000	**Uptrend**
Chas vs Price in $1000	**Categorical**
Crim vs Price in $1000	**Downtrend**
Dis vs Price in $1000	**Uptrend**
Indus vs Price in $1000	**Downtrend**
Lstat vs Price in $1000	**Downtrend**
Nox vs Price in $1000	**Downtrend**
Ptratio vs Price in $1000	**Downtrend**
Rad vs Price in $1000	**Downtrend**
Rm vs Price in $1000	**Uptrend**
Tax vs Price in $1000	**Downtrend**
Zn vs Price in $1000	**Uptrend**

Fig. 7 Analysis Table

CHAPTER 8

8.1 TOOLS AND TECHNOLOGY

The tools and technology used in this project are:

- **Anaconda :** Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012.

- **Jupyter Notebook :** Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating notebook documents. A Jupyter Notebook document is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media. Underneath the interface, a notebook is a JSON document, following a versioned schema, usually ending with the ". Ipynb" extension.

- **Python :** Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

- **Tableau:** Tableau Software is an American interactive data visualization software company focused on intelligence. Tableau products query relational databases, online analytical processing cubes, cloud databases, and spreadsheets to generate graph-type data visualizations. The software can also extract, store, and retrieve data from an in-memory data engine.

- **Machine Learning:** Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn Machine learning is actively being used today, perhaps in many more places than one would expect.

- **GretL:** GretL is a cross-platform software package for econometric analysis, written in the C programming language. It is free, open-source software. It has both a graphical user interface (GUI) and a command-line interface. It is written in C, uses GTK+ as widget toolkit for creating its GUI, and calls gnuplot for generating graphs. The native scripting language of GretL is known as hansl. It can also be used together with TRAMO/SEATS, R, Stata, Python, Octave, Ox and Julia.

8.2 LIBRARIES

The libraries used in this project are:

- **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. NumPy offers comprehensive mathematical functions, random number generators, linear algebra routines, Fourier transforms, and more.

- **Pandas :** pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is a Python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python.

- **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.

- **Seaborn:** Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and colour palettes to make statistical plots more attractive. It is built on the top of matplotlib library and also closely integrated to the data structures from pandas. Seaborn aims to make visualization the central part of exploring and understanding data. It provides dataset-oriented APIs, so that we can switch between different visual representations for same variables for better understanding of dataset.

- **Sklearn :** Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

- **Linear Regression:** Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.

- **Random Forest :** Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

CHAPTER 9

9.1 CONCLUSION

The goal of this report was to determine the machine learning model that give the best accuracy. Various statistical techniques were used to choose the algorithm from linear regression and random forest regressor, where random forest is more accurate with the accuracy of both training and testing of 98% and 84% respectively. In examining the final model, one finds after aggregating the analysis of Tableau and GretL that house prices are higher in areas with lower crime and lower pupil-teacher ratios. House prices also tend to be higher closer to the Charles River, and houses with more rooms are pricier. This report is interested in the neighbourhood attributes of houses, so the number of rooms is not an important predictor. The most interesting factors to consider are nitrogen oxide levels and distance to the main employment centers.

On the one hand, people would want to live close to their place of employment. Yet it is reasonable to suggest that pollution levels are higher as one moves closer to these main employment centers. Most importantly, when talking of pollution, it is not just nitrogen oxide levels that are higher, but also noise pollution levels. The regression model that was fitted shows that higher levels of pollution decrease house prices to a greater extent than distance to employment centers. This suggests that people would prefer to live further away from their place of employment if it meant lower levels of pollution, which is an interesting point to consider. On a concluding note, it is important to note that the data for this report was collected several decades ago. In the years since, there is no doubt that pollution levels have risen and it would be interesting to examine the ways in which that affects house pricing in Boston today.

9.2 FUTURE SCOPE

The supplementary feature that can be added to our proposed system is to avail users of a full-fledged user interface so there can be multiple functionalities for users to use with the ML model for numerous locations. XG Boost can be used for little better accuracy of the model. Also, an Amazon EC2 connection will take the system even further and increase the ease of use. Lastly, developing a well-integrated web application that can predict prices whenever users want it to will complete the project.

REFERENCES

[1] Harrison, D. and Rubinfeld, D.L. ‘Hedonic prices and the demand for clean air’, J. Environ. Economics & Management, vol.5, 81-102, 1978.

[2] Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[3] David Donoho. “50 Years of Data Science”. In: Journal of Computational and Graphical Statistics 26.4 (2017), pp. 745–766. url: https: //doi.org/10.1080/10618600.2017.1384734.

[4] Sameer Chand Pudaruth. “Predicting the Price of Used Cars using Machine Learning Techniques”. In: International Journal of Information & Computation Technology 4 (Jan. 2014).

[5] Alejandro Bal dominos et al. “Identifying Real Estate Opportunities Using Machine Learning”. In: MDPI Applied Sciences (Nov. 2018).

[6] Johan Oxenstierna. Predicting house prices using Ensemble Learning with Cluster Aggregations. 2017.

[7] Thuraiya Mohd, Suraya Masrom, Noraini Johari, "Machine Learning Housing Price Prediction in Petaling Jaya, Selangor, Malaysia ", International Journal of Recent Technology and Engineering (IJRTE), Volume-8, Issue-2S11, 2019.

[8] G. Naga Satish, Ch. V. Raghavendran, M.D. Sugnana Rao, Ch. Srinivasulu, "House Price Prediction Using Machine Learning”, International Journal of Innovative Technology and Exploring Engineering (IJITEE), Volume-8 Issue-9, 2019.

[9] Kuvalekar, Alisha and Manchewar, Shivani and Mahadik, Sidhika and Jawale, Shila, House Price Forecasting Using Machine Learning (April 8, 2020). Proceedings of the 3rd International Conference on Advances in Science & Technology (ICAST) 2020

[10] Neelam Shinde, Kiran Gawande, "Valuation of House Prices Using Predictive Techniques", International Journal of Advances in Electronics and Computer Science, Volume-5, Issue-6, 2018.

[11] Jingyi Mu, Fang Wu, and Aihua Zhang, " Housing Value Forecasting Based on Machine Learning Methods", Hindawi Publishing Corporation Abstract and Applied Analysis, Volume 2014.

[12] Sayan Putatunda, "PropTech for Proactive Pricing of Houses in Classified Advertisements in the Indian Real Estate Market".

[13] Atharva Chouthai, Mohammed Athar Rangila, Sanved Amate, Prayag Adhikari, Vijay Kukre, "House Price Prediction Using Machine Learning", International Research Journal of Engineering and Technology (IRJET), Vol:06 Issue: 03, 2019.

[14] B. Balakumar, P. Raviraj, S. Essakkiammal, "Predicting Housing Prices using Machine Learning Techniques".

[15] Akshay Babu, Dr. Anjana S Chandran, "Literature Review on Real Estate Value Prediction Using Machine Learning", International Journal of Computer Science and Mobile Applications, Vol: 7 Issue: 3, 2019.

[16] Mr. Rishikesh Naikare, Mr. Girish Girandole, Mr. Akash Dumbre, Mr. Kaushal Agrawal, Prof. Chaitanya Manka, "House Planning and Price Prediction System using Machine Learning", International Engineering Research Journal, Vol:3 Issue: 3, 2019.

[17] Aswin Sivam Ravikumar, Thibaut Lust, "Real Estate Price Prediction Using Machine Learning", 2016.

[18] Bindu Sivasankar, Arun P. Ashok, Gouri Madhu, Fousiya S, "House Price Prediction", International Journal of Computer Science and Engineering (IJCSE), Vol: 8 Issue: 7, 2020.

[19] M Thamarai, S P Malarvizhi, " House Price Prediction Modeling Using Machine Learning", International Journal of Information Engineering and Electronic Business (DJIEEB), VoL12, No.2, pp. 15- 20, 2020. DOI: 10.5815/ijieeb.2020.02.03