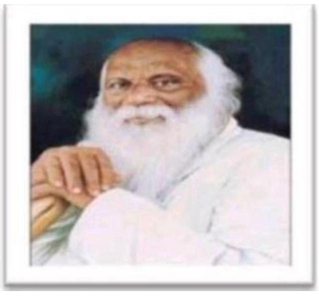
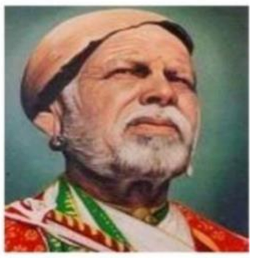




RAYAT SHIKSHAN SANSTHA'S,

SADGURU GADAGE MAHARAJ COLLEGE, KARAD.

(An Autonomous college)

Department of Statistics

Project report on

" A Statistical and Machine Learning Approach for Crop recommendations ”

Submitted by,
Mr. Digambar Kavitkar

M.Sc. II(Statistics)

Under the Guidance of

Miss. S.P. Patil

(2023-2024)

Index

CERTIFICATE

This is to certify that the Project report entitled “A Statistical and Machine Learning Approach for Crop recommendations” being submitted by Mr. Kavitkar Digambar. as partial fulfillment for the M.Sc. in Statistics of Sadguru Gadage Maharaj College, Karad. record of bonafide work carried out by them under supervision and guidance.

To the best of our knowledge and belief, the matter presented in this project report is original and has not been submitted elsewhere for any other purpose.

Place: Karad

Date:

Teacher in-charge	Examiner	P.G. Coordinator	Head
		(Department of Statistics)	(Department of Statistics)

ACKNOWLEDGEMENT

This year is an extremely informative journey to us. This journey of the study at the department and the project gave me "House Price Prediction". I am very thankful to Miss.S.P.Patil my internal guide for their incomparable affection during my project works. So we take opportunity to express my heartfelt thanks to all statistics department staff to encourage and support me to complete my documentation.

Also I would like to thank all the non-teaching staff of the department for there help and co-operation. I thank all my friends for their co-operation and help which I received from them during the work throughout.

The project have been the outcome ideas of combination of ideas of many people. My project is dedicated to all those people who support me and give guidance to me. I am indebted to my parents for their encouragement and patience throughout my study and the trust they have on me.

Yours' Sincerely

Mrs. Kavitkar Digambar

M.sc II (Statistics)

INTRODUCTION

It is essential to the success of global food production and to sustainable smart cities that farmers have the capacity to properly forecast crop type. Choices made promptly about imports and exports, based on accurate estimates, are essential to the nation’s food security. To generate better varieties, breeders need seed producers who can accurately predict how new hybrids will function in various environments. The ability to accurately forecast the type and yield empowers growers and farmers to make more informed decisions on management and finances

OBJECTIVES

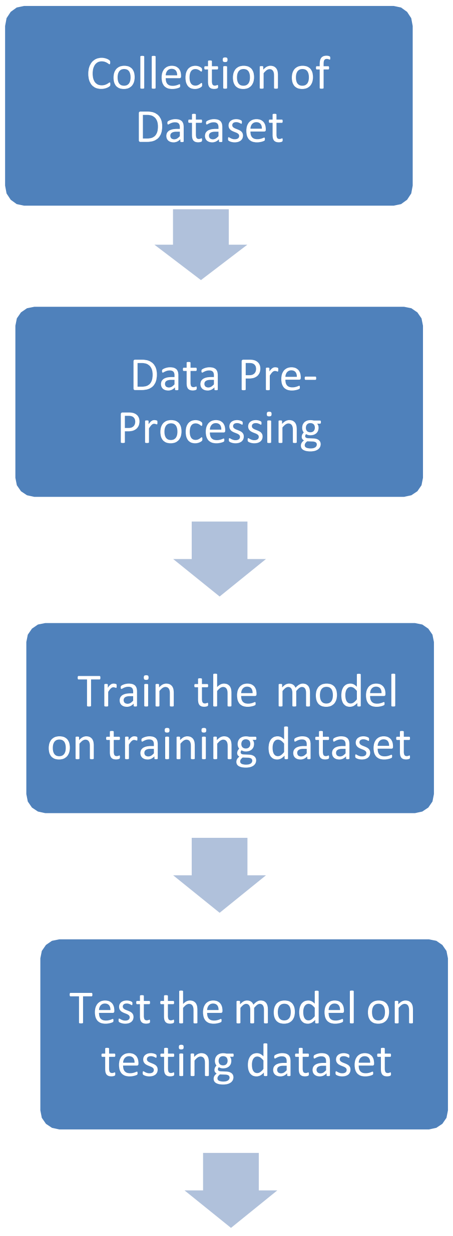
1. The solution will benefit farmers to maximize productivity in agriculture, reduce Soil degradation in cultivated fields, and reduce fertilizer use in crop production By recommending the right crop by considering various attributes.

2. This paper aims to recommend the most Suitable crop based on input parameters like Nitrogen (N), Phosphorous (P), Potassium (K), PH value of soil, Humidity, Temperature, and Rainfall.

3. The main objective is to obtain a better variety of crops that can be grown Over the season. The proposed system would help to minimize the Difficulties faced by farmers in choosing a crop and maximize the yield.

4. The model predicts the crop yield by studying factors such as rainfall, Temperature, soil type etc.

Methodology


Collection of Dataset
Data Pre-Processing
Train the model on training dataset
Test the model on testing dataset

➢ Tools and Techniques used :

- Techniques:

  Machine learning classification models:

  1. Decision Tree Classifier
  2. Random Forest
  3. Multiple Linear Regression

- Tools:

  Jupyter Notebook

ABSTRACT

House price forecasting is an important topic of real estate. To derive useful knowledge from historical data of property markets. Machine learning techniques are applied to analyze historical property transactions in India to discover useful models for house buyers and sellers. Revealed is the high discrepancy between house prices in the most expensive and most affordable suburbs in the city of boston .

This study demonstrates how different models of regression can forecast house price values. And we will compare the results of models, for example Linear Regression, Decision Tree Regressor, Random Forest Regressor.

Here RMSE of Linear Regression = 4.9452

Decision Tree Regressor = 1.5716

Random Forest Regressor =1.6369

This work offers the best approach to the Decision Tree Regressor.

About Data

The public datasets have been chosen because they are readily available and easily accessible. Kaggle is a popular platform. The dataset is used for crop prediction. It has features like N, P, K, rainfall, humidity, pH and crop. N, P, K stands for Nitrogen, Phosphorous and Potassium nutrients in soil. It has 4513 total records containing 16 unique crops. Data consists of the like crops: rice, maize, sugarcane etc.

Attribute Information :

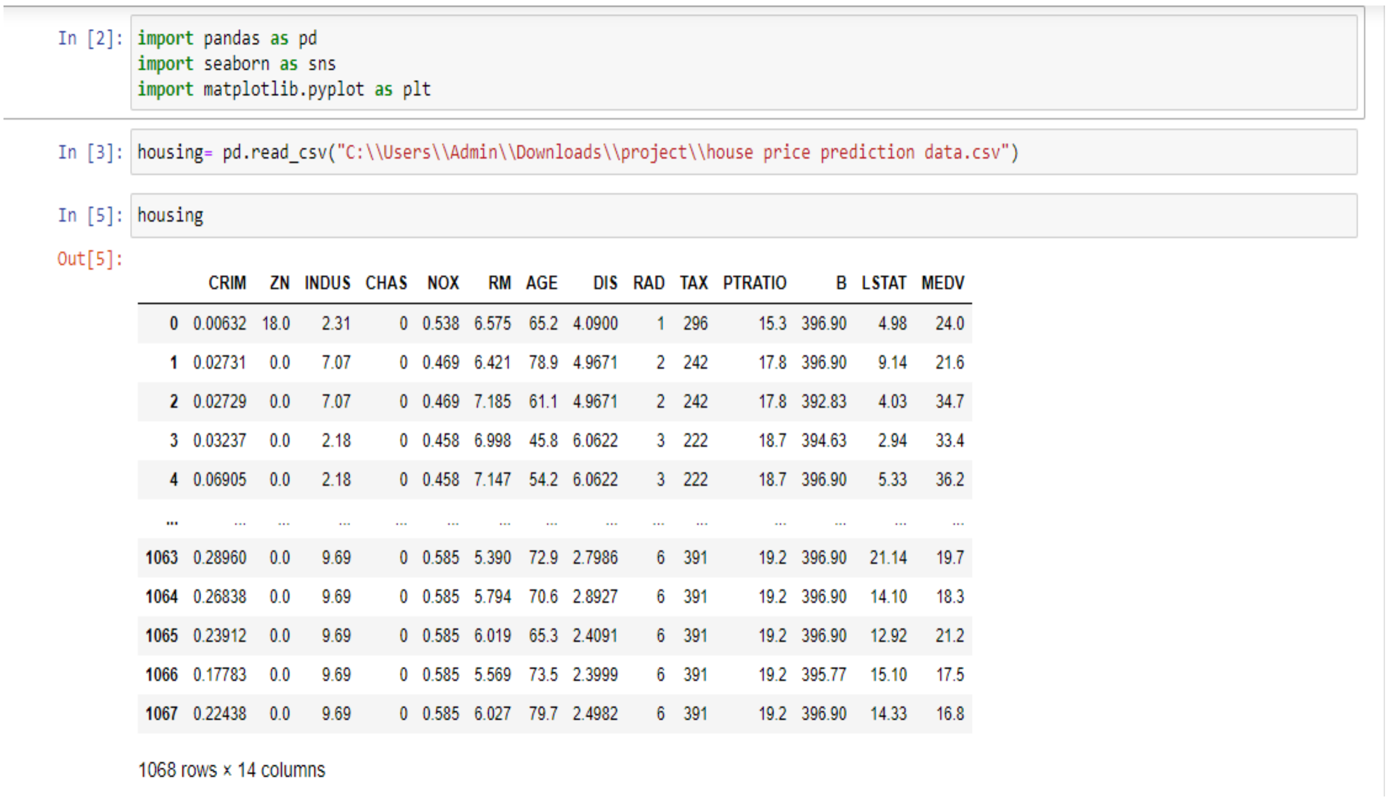
1. N : Nitrogan
2. P : Phosphorous
3. K : Potassium
4. Rainfall :
5. Humidity :
6. pH :
7. Temperature :
8. Crop :

9.	LSTAT	% lower status of the population
10.	MEDV	Median value of owner-occupied homes in $1000's

Discreptive Statistics

Data preparation:

We use python (Jupyter Notebook) to work on this dataset. Firstly import all the libraries which is required and then import the dataset.



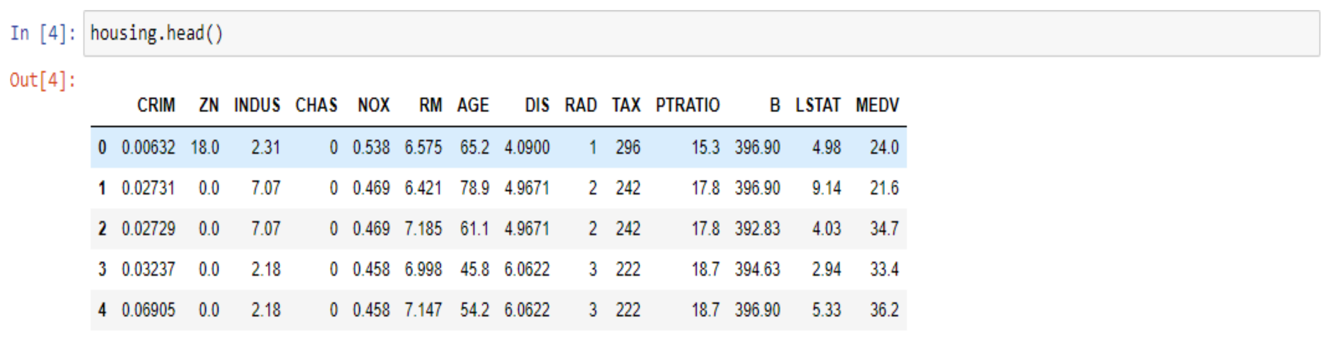
```
In [2]: import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
```

```
In [3]: housing= pd.read_csv("C:\\Users\\Admin\\Downloads\\project\\house price prediction data.csv")
```

```
In [5]: housing
```

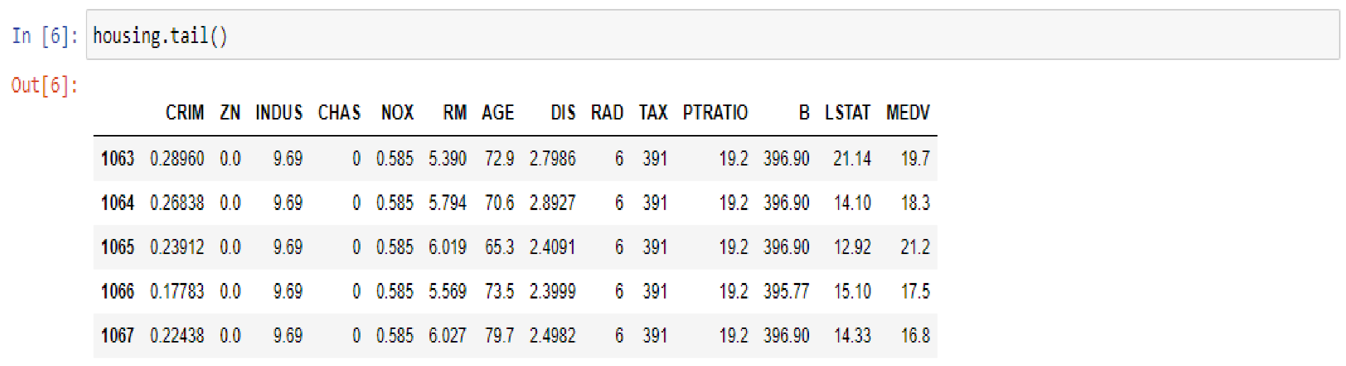
```
Out[5]:
          CRIM    ZN  INDUS  CHAS    NOX     RM   AGE     DIS  RAD  TAX  PTRATIO       B  LSTAT  MEDV
   0   0.00632  18.0   2.31     0  0.538  6.575  65.2  4.0900    1  296     15.3  396.90   4.98  24.0
   1   0.02731   0.0   7.07     0  0.469  6.421  78.9  4.9671    2  242     17.8  396.90   9.14  21.6
   2   0.02729   0.0   7.07     0  0.469  7.185  61.1  4.9671    2  242     17.8  392.83   4.03  34.7
   3   0.03237   0.0   2.18     0  0.458  6.998  45.8  6.0622    3  222     18.7  394.63   2.94  33.4
   4   0.06905   0.0   2.18     0  0.458  7.147  54.2  6.0622    3  222     18.7  396.90   5.33  36.2
 ...       ...   ...    ...   ...    ...    ...   ...     ...  ...  ...      ...     ...    ...   ...
1063   0.28960   0.0   9.69     0  0.585  5.390  72.9  2.7986    6  391     19.2  396.90  21.14  19.7
1064   0.26838   0.0   9.69     0  0.585  5.794  70.6  2.8927    6  391     19.2  396.90  14.10  18.3
1065   0.23912   0.0   9.69     0  0.585  6.019  65.3  2.4091    6  391     19.2  396.90  12.92  21.2
1066   0.17783   0.0   9.69     0  0.585  5.569  73.5  2.3999    6  391     19.2  395.77  15.10  17.5
1067   0.22438   0.0   9.69     0  0.585  6.027  79.7  2.4982    6  391     19.2  396.90  14.33  16.8

1068 rows × 14 columns
```

We can examine the data using pandas library function head( ) and tail( )



```
In [4]: housing.head()
Out[4]:
```

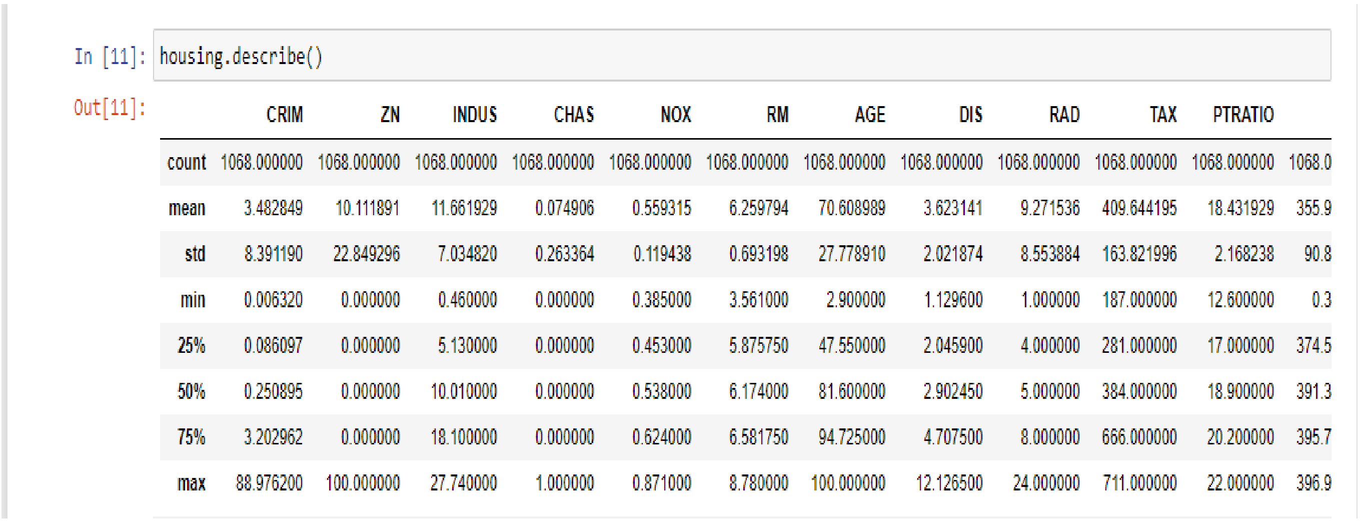
	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2



```
In [6]: housing.tail()
Out[6]:
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
1063	0.28960	0.0	9.69	0	0.585	5.390	72.9	2.7986	6	391	19.2	396.90	21.14	19.7
1064	0.26838	0.0	9.69	0	0.585	5.794	70.6	2.8927	6	391	19.2	396.90	14.10	18.3
1065	0.23912	0.0	9.69	0	0.585	6.019	65.3	2.4091	6	391	19.2	396.90	12.92	21.2
1066	0.17783	0.0	9.69	0	0.585	5.569	73.5	2.3999	6	391	19.2	395.77	15.10	17.5
1067	0.22438	0.0	9.69	0	0.585	6.027	79.7	2.4982	6	391	19.2	396.90	14.33	16.8

From above we observe that the dataset is consist of 1068 samples i.e. rows with 14 variables i.e. columns.



```
In [11]: housing.describe()
```

```
Out[11]:
              CRIM           ZN        INDUS         CHAS          NOX           RM          AGE          DIS          RAD          TAX      PTRATIO
count  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.000000  1068.0
mean      3.482849    10.111891    11.661929     0.074906     0.559315     6.259794    70.608989     3.623141     9.271536   409.644195    18.431929   355.9
std       8.391190    22.849296     7.034820     0.263364     0.119438     0.693198    27.778910     2.021874     8.553884   163.821996     2.168238    90.8
min       0.006320     0.000000     0.460000     0.000000     0.385000     3.561000     2.900000     1.129600     1.000000   187.000000    12.600000     0.3
25%       0.086097     0.000000     5.130000     0.000000     0.453000     5.875750    47.550000     2.045900     4.000000   281.000000    17.000000   374.5
50%       0.250895     0.000000    10.010000     0.000000     0.538000     6.174000    81.600000     2.902450     5.000000   384.000000    18.900000   391.3
75%       3.202962     0.000000    18.100000     0.000000     0.624000     6.581750    94.725000     4.707500     8.000000   666.000000    20.200000   395.7
max      88.976200   100.000000    27.740000     1.000000     0.871000     8.780000   100.000000    12.126500    24.000000   711.000000    22.000000   396.9
```

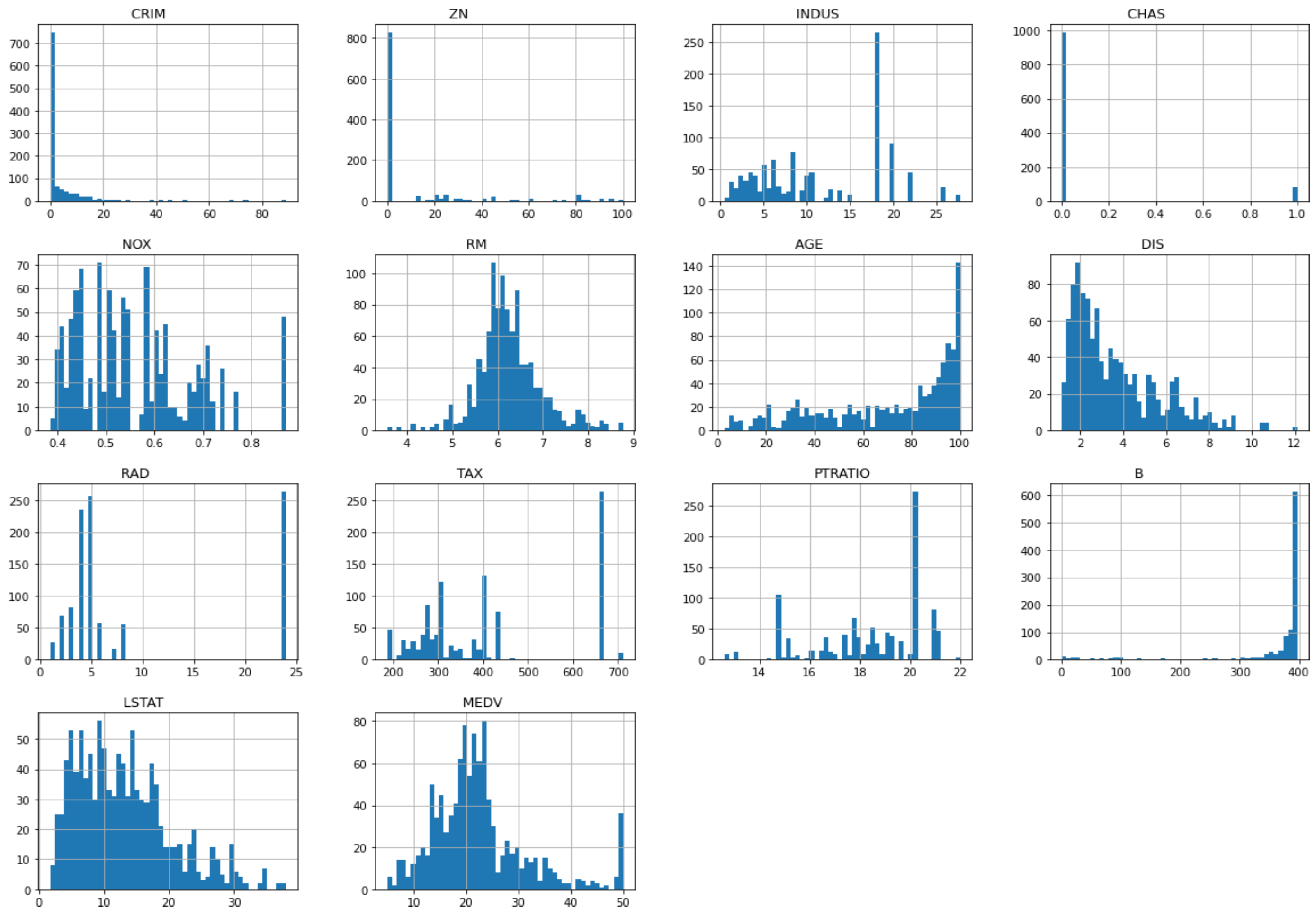
Interpretation:

Summary statistics provide mean, min, max, median and quartiles of all the elements present in dataset.

Data Visualization :

1) Visualize the data using Histogram to get the idea about distribution of the features.

Histogram for each attribute in the data set:



Interpretation:

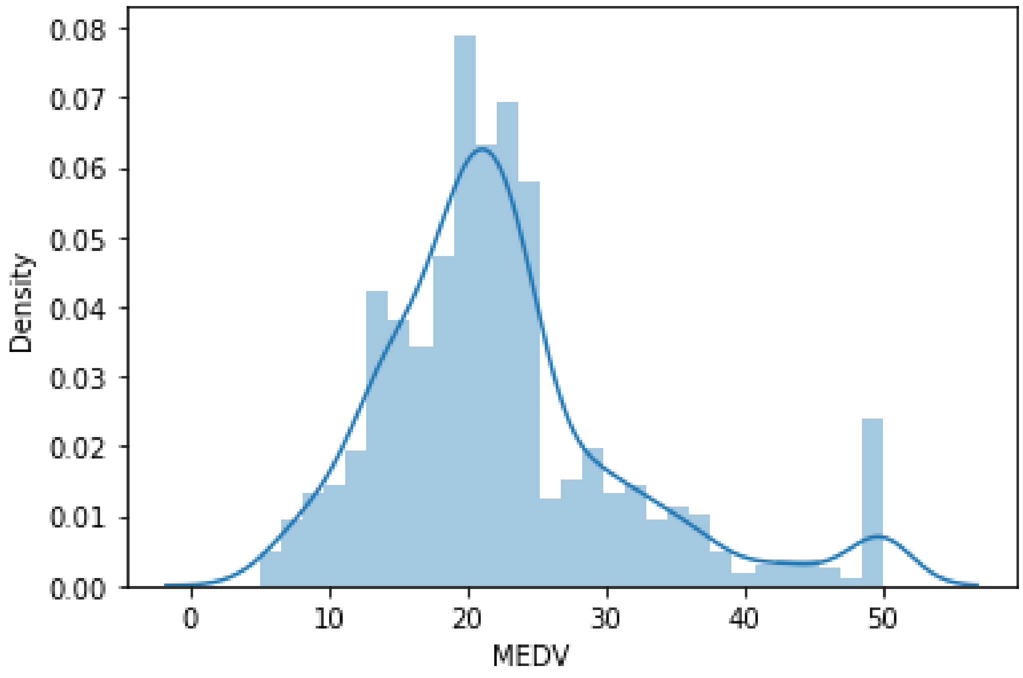
1) The median house value has a sudden peak around 500000, which is very different from others. It is recommended to ignore these data in training the model.

2) The Average number of rooms(RM) is centered around 6. Probably, 6 means $60,000.

3) Here crime rate is nearly about zero or it just goes from 0 to 20 and it is very rare that it goes up to 80.

2)Distribution of MEDV and AGE using distplot:

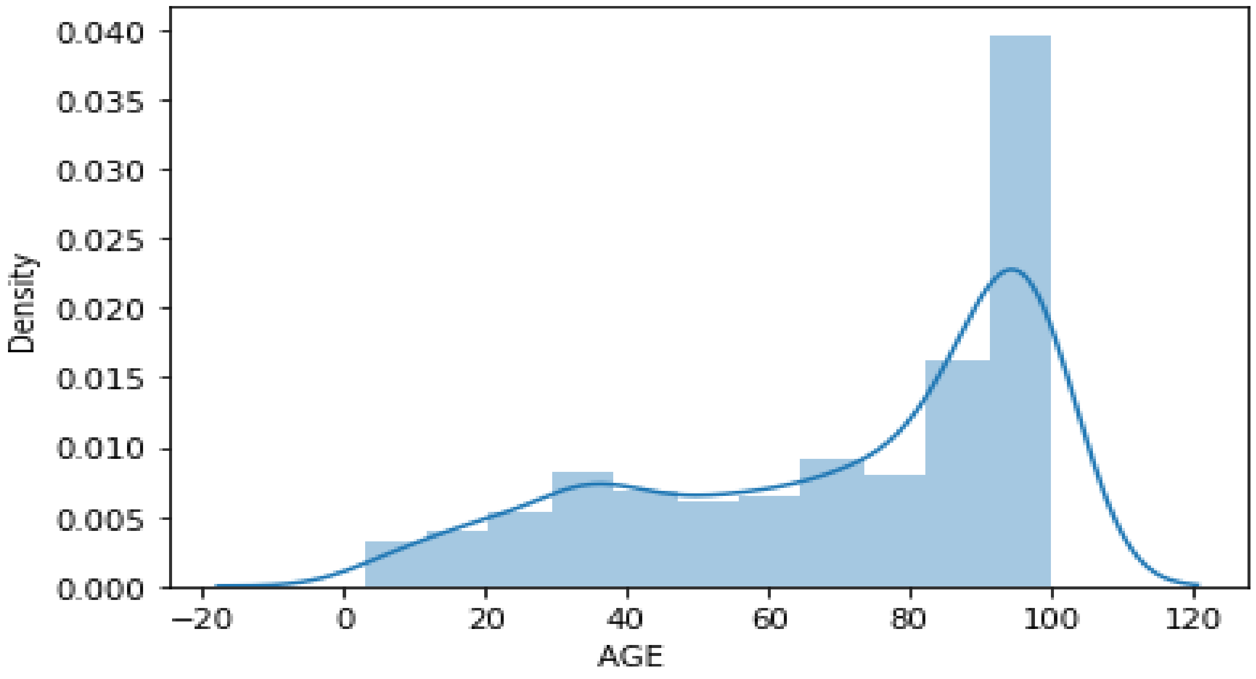
1) MEDV:Median value of owner-occupied homes in $1000's



Interpretation:

i. From above plot we see that there is a long tail of values on the positive side of the peak therefore the distribution is right skewed.
ii. Here skewness is positive. Hence the distribution is positively skewed.

3) AGE : proportion of owner-occupied units built prior to1940



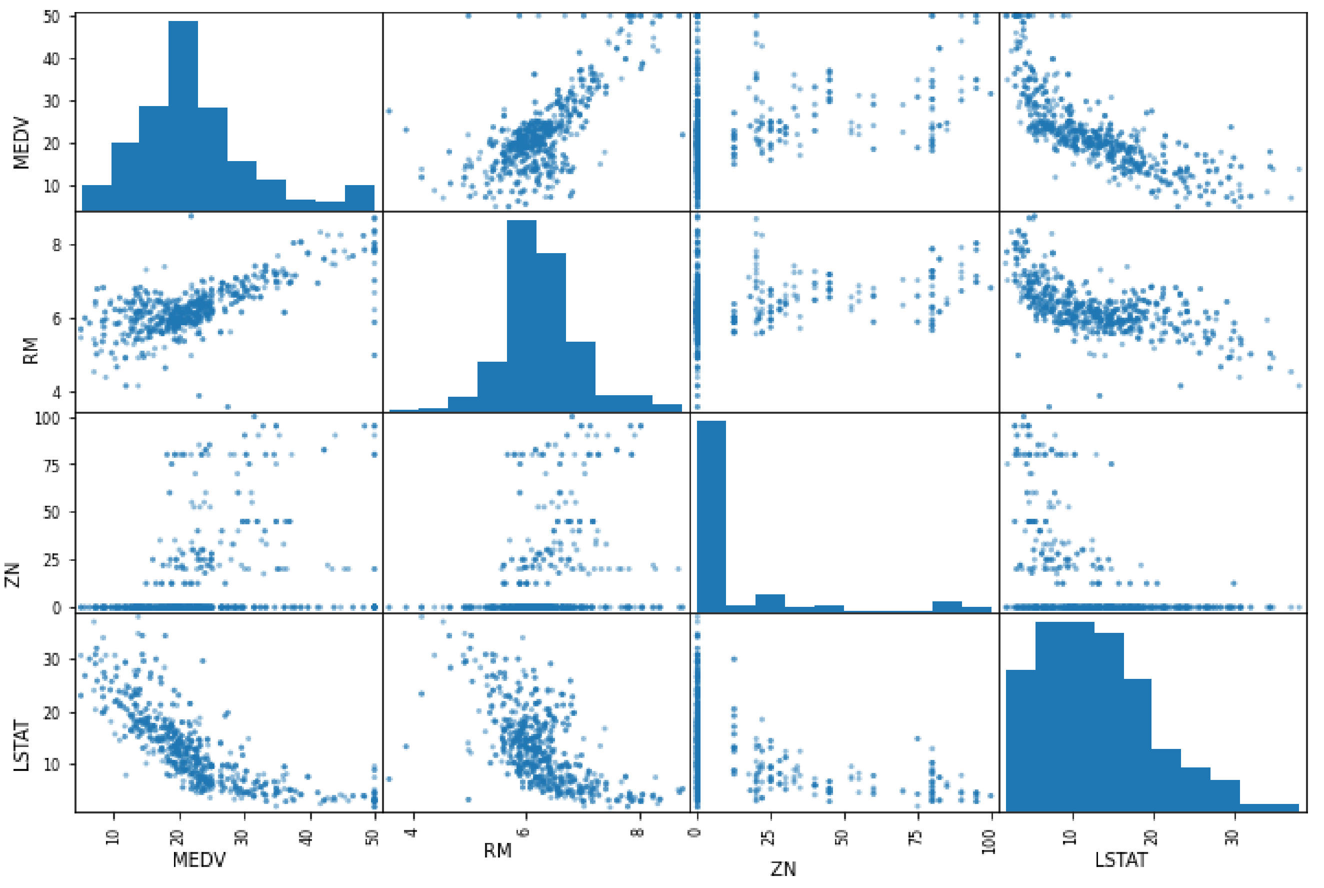
i. From above plot we see that there is a long tail of values on the negative side of the peak therefore the distribution is left skewed.

ii. Here skewness is negative. Hence the distribution is negatively skewed.

4) Pair-plot:

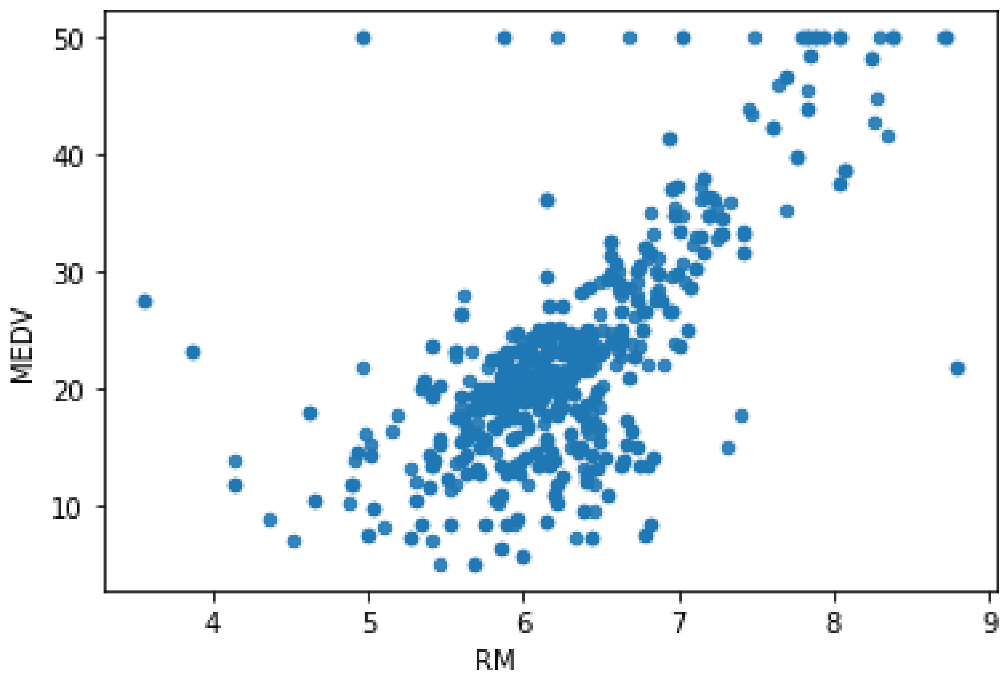
Pair-plot visualizes the given data to find the relationship between them where the variables can be continuous or categorical.

Pair-plots are also great way to immediately seen the correlation between all the variables but we see make with the only continuous columns from the data. Because with so many features it can be difficult to search one. so we make pair plot with only continuous variables.



Interpretation:

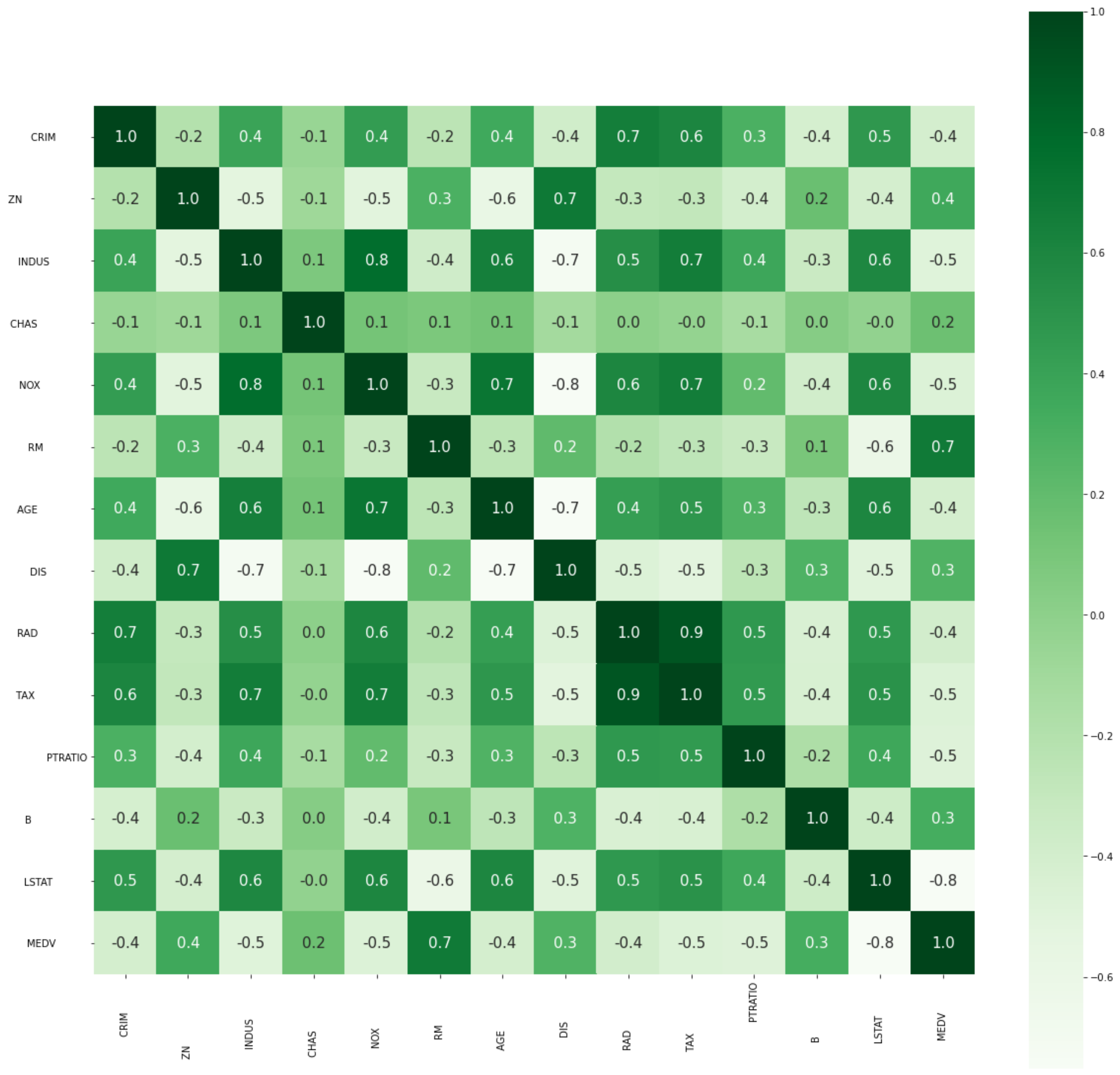
From above plot, we observe the variations in each plot. The main diagonal subplots are the univariate Histogram[Distribution] for each attribute. Median house value(MEDV) is highly correlated with average number of rooms(RM).



Interpretation:

Here the correlation between the Median house value(MEDV) and average number of rooms(RM) is high.

Heatmap:



	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
CRIM	1.0	-0.2	0.4	-0.1	0.4	-0.2	0.4	-0.4	0.7	0.6	0.3	-0.4	0.5	-0.4
ZN	-0.2	1.0	-0.5	-0.1	-0.5	0.3	-0.6	0.7	-0.3	-0.3	-0.4	0.2	-0.4	0.4
INDUS	0.4	-0.5	1.0	0.1	0.8	-0.4	0.6	-0.7	0.5	0.7	0.4	-0.3	0.6	-0.5
CHAS	-0.1	-0.1	0.1	1.0	0.1	0.1	0.1	-0.1	0.0	-0.0	-0.1	0.0	-0.0	0.2
NOX	0.4	-0.5	0.8	0.1	1.0	-0.3	0.7	-0.8	0.6	0.7	0.2	-0.4	0.6	-0.5
RM	-0.2	0.3	-0.4	0.1	-0.3	1.0	-0.3	0.2	-0.2	-0.3	-0.3	0.1	-0.6	0.7
AGE	0.4	-0.6	0.6	0.1	0.7	-0.3	1.0	-0.7	0.4	0.5	0.3	-0.3	0.6	-0.4
DIS	-0.4	0.7	-0.7	-0.1	-0.8	0.2	-0.7	1.0	-0.5	-0.5	-0.3	0.3	-0.5	0.3
RAD	0.7	-0.3	0.5	0.0	0.6	-0.2	0.4	-0.5	1.0	0.9	0.5	-0.4	0.5	-0.4
TAX	0.6	-0.3	0.7	-0.0	0.7	-0.3	0.5	-0.5	0.9	1.0	0.5	-0.4	0.5	-0.5
PTRATIO	0.3	-0.4	0.4	-0.1	0.2	-0.3	0.3	-0.3	0.5	0.5	1.0	-0.2	0.4	-0.5
B	-0.4	0.2	-0.3	0.0	-0.4	0.1	-0.3	0.3	-0.4	-0.4	-0.2	1.0	-0.4	0.3
LSTAT	0.5	-0.4	0.6	-0.0	0.6	-0.6	0.6	-0.5	0.5	0.5	0.4	-0.4	1.0	-0.8
MEDV	-0.4	0.4	-0.5	0.2	-0.5	0.7	-0.4	0.3	-0.4	-0.5	-0.5	0.3	-0.8	1.0

Now let's look at the correlation between all the variables. We use the heatmap to visualize the correlation. Heatmap visualize the data through variation in coloring. The variables in darker colors that means correlation is more. Heatmap is great way to immediately see the correlationship between all variables but we see make with the only continuous columns from the data. Because with so many features it can be difficult to search one. From above we see that there is strong correlation of RM(Average number of rooms) with our median house value. But also there is negative correlation of variables with our median house value.

train-test-split:

The simplest method that we can use to evaluate the performance of a machine learning algorithm is to use different training and testing datasets. We can take our original dataset and split it into two parts. Train the algorithm on the first part, make predictions on the second part and evaluate the predictions against the expected results.
The size of the split can depend on the size and specifics of your dataset, although it is common to use 80% of the data for training and the remaining 20% for testing. This algorithm evaluation technique is very fast. It is ideal for large datasets (millions of records) where there is strong evidence that both splits of the data are representative of the underlying problem. Because of the speed, it is useful to use this approach when the algorithm you are investigating is slow to train.

Machine learning:

Machine learning is a method of data analysis which sends instructions (programmable code) to computers so that they can learn from data. Then, based on the learned data, they provide us the predicted results/patterns. With the help of Machine Learning, we can develop intelligent systems that are capable of taking decisions on an autonomous basis.

Supervised Learning:

Supervised learning: Supervised learning is a type of system in which both input and desired output data are provided. Input and output data are labelled for classification to provide a learning basis for future data processing. Supervised learning problems can be further grouped into Regression and Classification problems.
A regression problem is when the output variable is a real or continuous value, such as salary or weight.

We have different types of algorithms in Machine Learning:

1. Linear Regression
2. Decision Tree Regressor
3. Random Forest Regressor

1.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. Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression

The simple linear regression model is represente by:

$$y = \beta O+\beta 1x+\varepsilon$$

The linear regression model contains an error term that is represented by ε. The error term is used to account for the variability in y that cannot be explained by the linear relationship between x and y. Ife were not present, that would mean that knowing x would provide enough information to determine the value of y.

There also parameters that represent the population being studied. These parameters of the model are represented by β0 and β1.

The simple linear regression equation is graphed as a straight line, where: y is the predicted value of the dependent variable (y) for any given value of the independent variable (x).

BO is the intercept, the predicted value of y when the x is 0.
B1 is the regression coefficient-how much we expect y to change as x increases.
x is the independent variable (the variable we expect is influencing y).

e is the error of the estimate, or how much variation there is in our estimate of the regression coefficient.

Multiple linear regression is used to estimate the relationship between two or more independent variables and one dependent variable.

The formula for a multiple linear regression is:

$$Yi=\beta0+\beta lxil+\beta2xi2+...+\beta nxin+ \varepsilon$$

* y the predicted value of the dependent variable

* B0 = the y-intercept (value of y when all other parameters are set to 0)

*B1X1=the regression coefficient (B1) of the first independent variable (X1) (a.k.a, the effect that increasing the value of the independent variable has on the predicted y value)

BnXn= the regression coefficient of the last independent variable

e=model error (a.k.a. how much variation there is in our estimate of y)

Steps of linear regression classification:

Step 1: First we will split our data into 'X' array that contains the features and a 'y' array with the target variable.

Step 2: Next we will split our dataset into a training set and a testing set. We will train our model on the training set and then use the test set to evaluate the model(Predict 'y' variable).

Step 3: Here we pass 'X test to our model to predict the 'y_test'

Step 4:compare these 'prediction' results with actual results by plotting a graph.

2) Decision Tree Regressor:

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. A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. 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.

3).Random Forest Regressor:

Random Forest Regression is a supervised learning algorithm that uses ensemble learning method for regression. Ensemble learning method is a technique that combines predictions from multiple machine learning algorithms to make a more accurate prediction than a single model. A Random Forest
Regression model is powerful and accurate. It usually performs great on many problems, including features with non-linear relationship.

Here we have given Machine Learning Algorithm Performance Regression
Metrics

Root Mean Square Error:

Taking the square root of the mean squared error converts the units back to the original units of the output variable and can be meaningful for description and presentation. This is called the Root Mean Squared Error (or RMSE).

* We have also given table of some predicted values with its corresponding actual values to get an ordinary and general idea of how values accurately predicted by respective model.

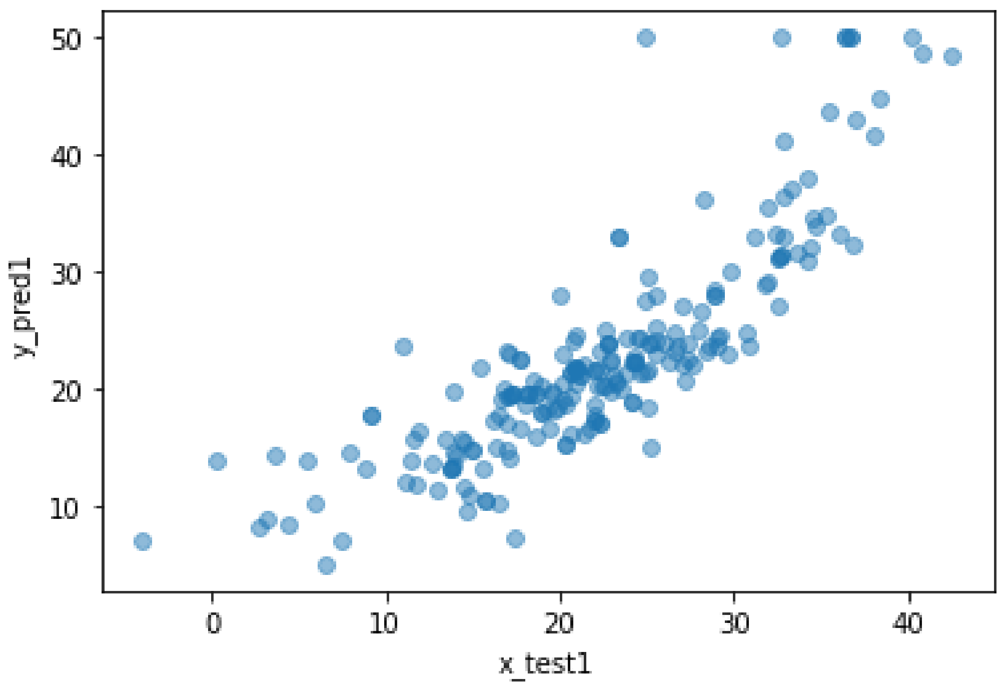
We have also given a scatter plot of actual and predicted values of each particular model

*Construction of Linear Regression Model

*Linear Regression:

This model yields accuracy, which is quite good.

Mean	4.6995
Standard deviation	0.8570
RMSE	4.9452

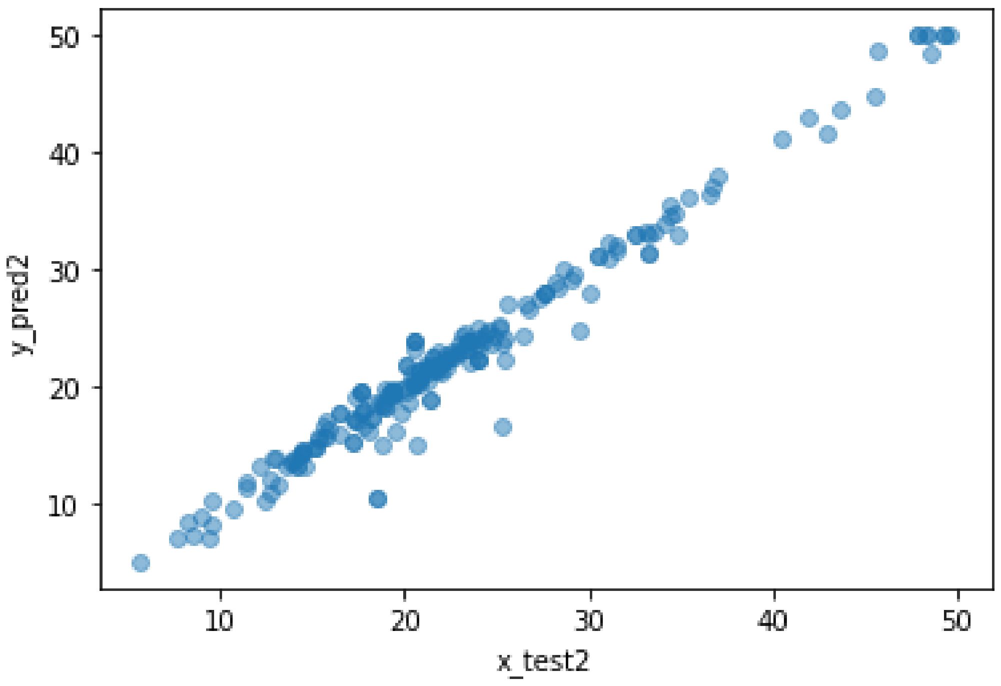


This plot gives the correlation between actual value & predicted value by linear regression model. The predictions are slightly close to the actual value.

*Random Forest:

This model yields accuracy, which is quite good.

Mean	1.9743
Standard deviation	**0.6593**
RMSE	1.6369

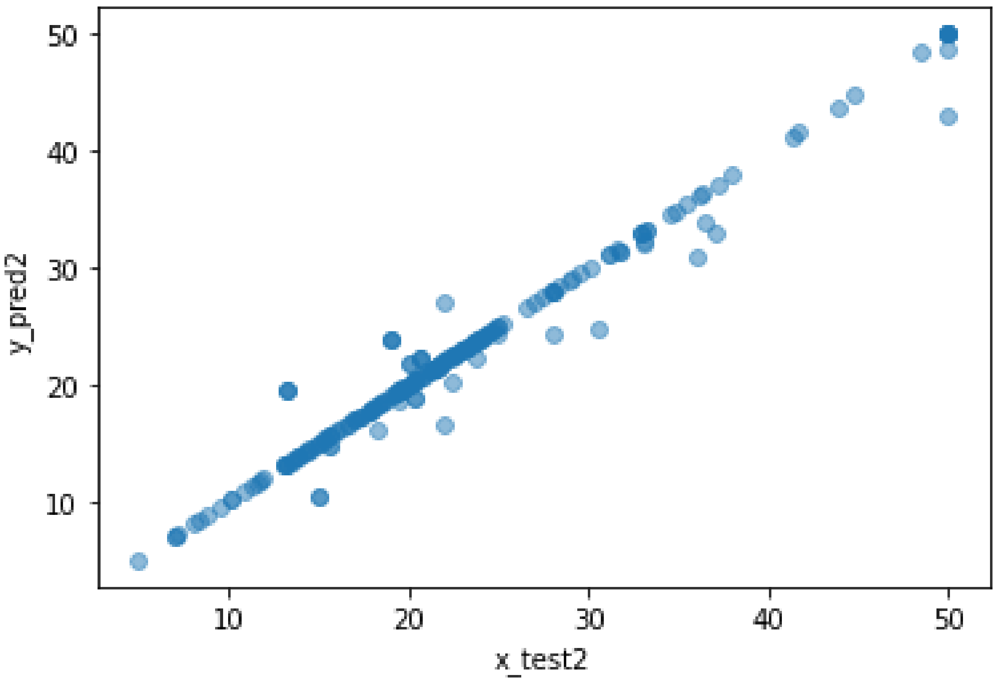


This plot gives the correlation between actual value & predicted value by random forest model. The predictions are slightly close to the actual value.

*Decision Tree:

This model yields high accuracy which is best among all the above models.

Mean	2.0124
Standard deviation	0.5278
RMSE	1.5716



This plot gives the correlation between actual value & predicted value by decision tree model. The predictions are very close to the actual value.

- **The performance of all regression model is compared in the following table**

Algorithm	Mean	Standard deviation	RMSE
Linear Regression	4.6995	0.8570	4.9452
Decision Tree	2.0124	0.5278	1.5716
Random Forest	1.9743	0.6593	1.6369

From the above that, we found that Decision Tree regression has most accuracy amongst 3 of them. We must confirm that all Standard deviation, RMSE should be low, therefore we select Decision Tree regression model for predicting median house value**.**

Results:

- The Median house value(MEDV) is centered around 20.
  Probably, 20 means $20,000.
- Distribution of MEDV (median house value) is positively skewed.
- Median house value is highly correlated with Average number of rooms(RM).
- decision tree Regressor model gave best performance with an RMSE score 1.5716

Major Findings:

i. Out of all regression model Decision Tree Regressor gives highest accuracy.

ii. The Decision tree Regression is the predictive model to predict the median house value.

SCOPE:

- House Price prediction, is important to drive Real Estate efficiency. As earlier, House prices were determined by calculating the acquiring and selling price in a locality.
- So, the House Price prediction model is very essential in filling the information gap and improve Real Estate efficiency.

LIMITATIONS:

- Tools applied to the data can change their performance if we.change the data.
- The results of statistical tools comes from this particular analysis are not valid universally.
- We develop model only with available variables but if add other important variables then we expect that our models gives better results.

REFRENCES:

- Jiawei Han, Micheline Kamber, Jian Pei (2011).Data Mining: Concepts and Techniques, Third Edition
- Python Machine Learning (by Seabstin Raschka & Vahid Mirajalili)
- https://www.researchgate.net/
- https://scikit-learn.org https://en.wikipedia.org
- https://machinelarningmastery.com

APENDIX

Import the libraries

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
housing= pd.read_csv("C:\\Users\\Admin\\Downloads\\project\\house price prediction
data.csv")
housing
housing.tail()
housing.info()
housing.notnull().sum
type(housing)
housing.columns
housing.rename(columns ={' CRIM          ':'CRIM','ZN                  ':'ZN','
INDUS                     ':'INDUS',' CHAS              ':'CHAS.',' NOX
':'NOX',' RM           ':'RM',' AGE              ':'AGE',' DIS            ':'DIS',' RAD
':'RAD',' TAX                ':'TAX'})
housing.columns
housing[' CHAS               '].value_counts()
housing.describe()
%matplotlib inline
import matplotlib.pyplot as plt
housing.hist(bins=50 , figsize=(20,15))
housing[' CRIM          '].hist()
```

Train-test splitting

```
import numpy as np
```

```
 def split_train_test(data, test_ratio):
 np.random.seed(42)
 shuffled= np.random.permutation(len(data))
  print(shuffled)
 test_set_size =int(len(data) * test_ratio)
 test_indices= shuffled[:test_set_size]
 train_indices= shuffled[test_set_size:]
  return data.iloc[train_indices],data.iloc[test_indices]
 from sklearn.model_selection import train_test_split
 train_set, test_set =train_test_split(housing, test_size=0.2,random_state=42)
 print(f"Rows in train set:{len(train_set)}\nRows in test set:{len(test_set)}\n")
 from sklearn.model_selection import StratifiedShuffleSplit
 split= StratifiedShuffleSplit(n_splits=1,test_size=0.2,random_state=42)
 for train_index,test_index in split.split(housing,housing[' CHAS                ']):
strat_train_set=housing.loc[train_index]
strat_test_set=housing.loc[test_index]
strat_train_set
strat_test_set
strat_train_set.info()
strat_train_set[' CHAS                 '].value_counts()
strat_train_set[' CHAS                 '].value_counts()
 housing =strat_train_set.copy()
housingg =strat_test_set.copy()
```

Looking for correlations

```
corr_matrix = housing.corr()
corr_matrix['MEDV          '].sort_values(ascending=False)
from pandas.plotting import scatter_matrix
 attributes=['MEDV          ',' RM              ','ZN                    ','LSTAT          ']
scatter_matrix(housing[attributes], figsize = (12,8))
```

```
 housing.plot(kind="scatter", x=" RM              ", y="MEDV          ", alpha=0.9)
 corr= housing.corr()
 corr.shape
 plt.figure(figsize=(20,20))
sns.heatmap(corr,cbar=True,square=True,fmt='.1f',annot=True,annot_kws={'size':15},
cmap='Greens')
```

Trying out attribute combination

```
housing["TAXRM"]= housing[' TAX                  ']/housing[' RM              ']
housing.head()
corr_matrix = housing.corr()
corr_matrix['MEDV           '].sort_values(ascending=False)
housing.plot(kind="scatter", x="TAXRM", y="MEDV           ", alpha=0.9)
housing= strat_train_set.drop("MEDV           ",axis=1)
housing_labels =strat_train_set["MEDV           "].copy()
housingg= strat_test_set.drop("MEDV           ",axis=1)
housingg_labelss =strat_test_set["MEDV           "].copy()
a =housing.dropna(subset=[' RM              '])
a.shape
housing.drop(' RM              ',axis=1).shape
median =housing[' RM              '].median()
housing[' RM              '].fillna(median)
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy= "median")
imputer.fit(housing)
imputer.statistics_
x= imputer.transform(housing)
housing_tr= pd.DataFrame(x,columns=housing.columns)
housing_tr.describe()
```

Creating a pipeline

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
my_pipeline=Pipeline([('imputer',SimpleImputer(strategy="median")),('std_scaler',
StandardScaler()),])
housing_num_tr = my_pipeline.fit_transform(housing_tr)
housing_num_tr
```

selecting a desired model

```
#from sklearn.linear_model import LinearRegression
#from sklearn.svm import SVR
#from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeRegressor
#from sklearn.ensemble import RandomForestRegressor
model=DecisionTreeRegressor()
#model=LinearRegression()
#model= RandomForestRegressor()
#model= LogisticRegression()
#model=SVR()
model.fit(housing_num_tr, housing_labels)
some_data= housing
some_labels=housingg_labelss
prepared_data= my_pipeline.transform(some_data)
model.predict(prepared_data)
```

```
 list(some_labels)
 x_test=model.predict(prepared_data)
 x_test
 x_pred=list(some_labels)
 x_pred
plt.scatter(x_test,x_pred,alpha=0.5)
plt.xlabel("x_test2")
plt.ylabel("y_pred2")
```

Evaluating the model

```
from sklearn.metrics import mean_squared_error
housing_predictions = model.predict(housing_num_tr)
mse= mean_squared_error(housing_labels,housing_predictions)
rmse=np.sqrt(mse)
rmse
```

Using better evaluation technique - Cross Validation

```
from sklearn.model_selection import cross_val_score
        scores=
 cross_val_score(model,housing_num_tr,housing_labels,scoring="neg_mean_sq
 uared_error",cv=10)
rmse_scores= np.sqrt(-scores)
rmse_scores
```

```
 def print_scores(scores):
print("scores:",scores)
print("Mean:",scores.mean())
print("standard deviation:",scores.std())
print_scores(rmse_scores)
```

Saving the model

```
  from joblib import dump, load
 dump(model,  'Dragon.joblib')
 x_test=strat_test_set.drop('MEDV          ',axis=1)
Y_test=strat_test_set['MEDV          '].copy()
x_test_prepared= my_pipeline.transform(x_test)
final_predictions = model.predict(x_test_prepared)
final_mse = mean_squared_error(Y_test, final_predictions)

final_rmse = np.sqrt(final_mse)
print(final_predictions)
final_rmse
```