Project : Boston House Price Prediction Analysis

Final Project Presentation , In partial fulfillment of the final project of ML course (109449-MGT-665-MA1) *Submitted by* **Mohammad Nasim**

Intellectual Property - DeVos Graduate Sch.
Northwood University
4000 Whiting Drive Midland, Michigan, USA

Guide : Dr. Itauma, Itauma
Division Chair & Assistant Professor
Graduate Academic, Northwood University

Why House-Price Prediction?

Homebuyers:

Homebuyers use house price predictions to make informed decisions about purchasing a home. By knowing the potential future value of a property, they can make better decisions about when and where to buy a home.

Real Estate Investors:

Real estate investors use house price predictions to make informed decisions about buying or selling property. Predictions can help investors identify undervalued properties, as well as properties with potential for growth.

Real Estate Agents:

Real estate agents use house price predictions to advise clients on pricing their homes or properties for sale. They can also use predictions to advise clients on when to sell or buy property, and what type of property to invest in.

Mortgage Lenders: Mortgage lenders use house price predictions to determine the value of a property and assess the risk associated with a loan. This helps lenders determine the amount of financing they are willing to offer to potential buyers.

Government: Governments use house price predictions to inform policy decisions related to housing, including housing affordability and housing supply. They can also use predictions to monitor the health of the housing market and identify potential risks.

LITERATURE REVIEW

- Borde, S., Rane, A., Shende, G., & Shetty, S. (2017). Real estate investment advising using machine learning. International Research Journal of Engineering and Technology (IRJET), 4(3), 1821.
- Trawinski, B., Telec, Z., Krasnoborski, J., et al. (2017). Comparison of expert algorithms with machine learning models for real estate appraisal. In Proceedings of the 2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA).
- Kontrimas, V., & Verikas, A. (2011). The mass appraisal of the real estate by computational intelligence. Applied Soft Computing, 11(1), 443-448.
- Woźniak, M., Graña, M., & Corchado, E. (2014). A survey of multiple classifier systems as hybrid systems. Information Fusion, 16, 3-17.
- Barr, J. R., Ellis, E. A., Kassab, A., Redfearn, C. L., Srinivasan, N. N., & Voris, K. B. (2017). Home price index: A machine learning methodology. International Journal of Semantic Computing, 11(1), 111-133.
- McCluskey, W. J., McCord, M., Davis, P. T., Haran, M., & McIlhatton, D. (2013). Prediction accuracy in mass appraisal: A comparison of modern approaches. Journal of Property Research, 30(4), 239-265.
- Rosen, S. (1974). Hedonic prices and implicit markets: Product differentiation in pure competition. Journal of Political Economy, 82(1), 34-55.
- Lughofer, E., Trawiński, B., Trawiński, K., Kempa, O., & Lasota, T. (2011). On employing fuzzy modeling algorithms for the valuation of residential premises. Information Sciences, 181(23), 5123-5142.
- Kusan, H., Aytekin, O., & Özdemir, I. (2010). The use of fuzzy logic in predicting house selling price. Expert Systems with Applications, 37(3), 1808-1813.
- Bin, O. (2004). A prediction comparison of housing sales prices by parametric versus semi-parametric regressions. Journal of Housing Economics, 13(1), 68-84.

LITERATURE REVIEW ASSESSMENT

DEVOS GRADUATE SCHOOL
NORTHWOOD UNIVERSITY

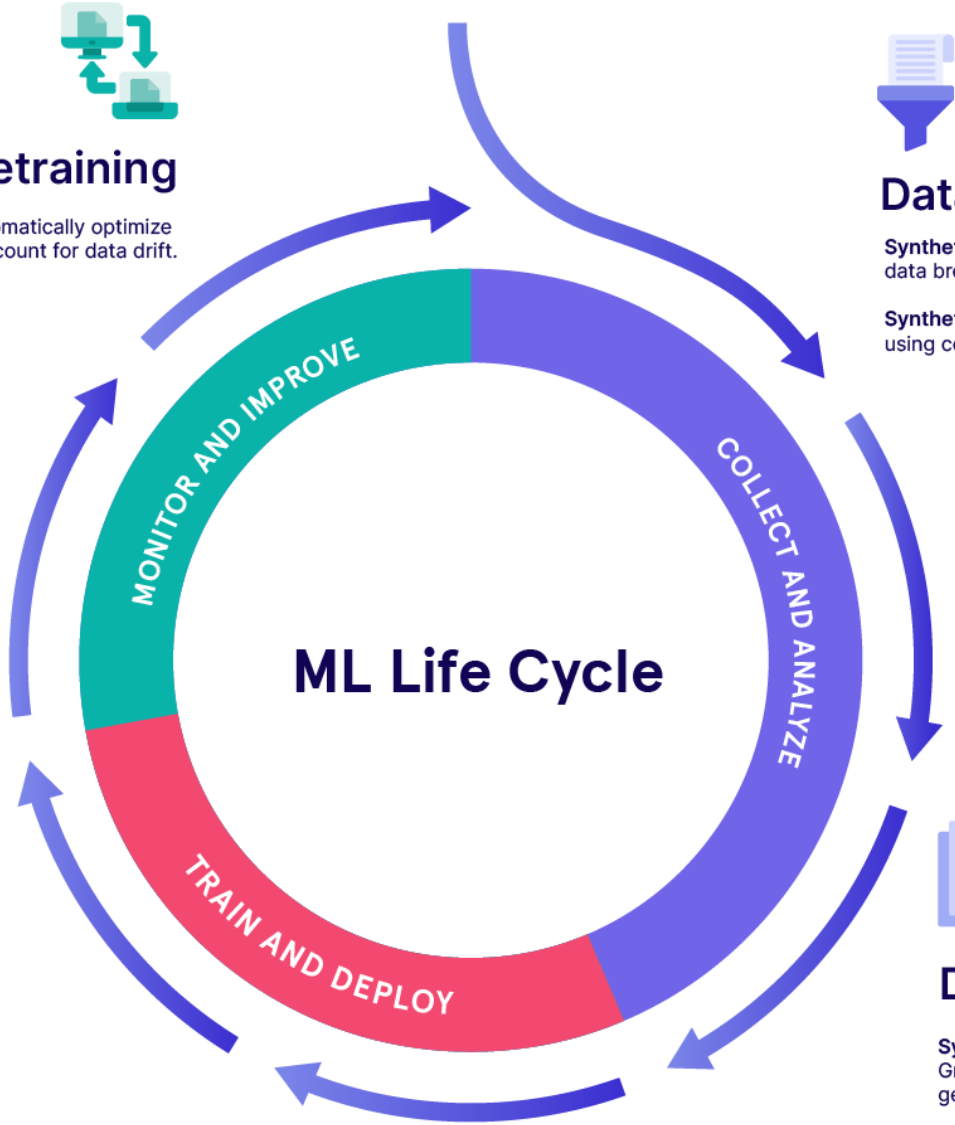
The mentioned ten **LITERATURES** related to real estate appraisal and prediction using machine learning, fuzzy logic, and regression analysis. The papers cover various topics, such as mass appraisal, prediction accuracy, hybrid systems, and valuation of residential premises. The authors employed several techniques to predict house selling prices, including fuzzy logic, machine learning, and parametric versus semi-parametric regressions.

PROBLEM STATEMENT

- The goal of this project is to predict the price of a house based on several features such as per capita crime rate ,proportion of residential land zoned, average number of rooms per dwelling, full-value property-tax rate etc. This is a regression problem where we are trying to predict a continuous value (house price) using machine learning.

LIFECYCLE TO BUILD A MODEL WITH MACHINE LEARNING

DEVOS GRADUATE SCHOOL
NORTHWOOD UNIVERSITY



Model Retraining

Synthetics + optimizer framework — automatically optimize your training sets to account for data drift.

Data Collection

Synthetics — anonymize data to mitigate the risk of data breaches and regulatory fines.

Synthetics — augment limited datasets and balance biased data using conditional data generation.

Model Monitoring

Synthetics — test model outcomes using synthetically-generated simulations of scenarios.

Synthetics — stress test your prediction pipeline power by amplifying samples a billion fold!

Exploratory Data Analysis

Evaluate — measure data quality and discover data correlations.
Console — safely share and collaborate on secure data.

Model Deployment

Synthetics — create demo data to test your model or application capabilities.

Data Annotation

Synthetics — boost training data examples using models such as Gretel GPT that are pre-trained on millions of documents to help generalize, fight data drift, and improve downstream accuracy.

Model Training & Evaluation

Synthetics + an optimization framework — generate optimized training datasets that enable maximum downstream machine learning task accuracy.

https://gretel.ai

METADATA (Information about Dataset)

- The dataset known as the Boston House Prices was gathered in 1978 and comprises 506 records that describe 14 features or characteristics of homes located in different suburbs of Boston.

METADATA (Featurenames and Its description)

DEVOS GRADUATE SCHOOL
NORTHWOOD UNIVERSITY

FeatureName	Description
CRIM	per capita crime rate by town
ZN	proportion of residential land zoned for lots over 25000 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 centers
RAD	index of accessibility to radial highways
TAX	full-value property-tax rate per $10000
PTRATIO	pupil-teacher ratio by town
BLACK	1000(Bk - 0.63)^2 where Black is the proportion of blacks by town
LSTAT	% lower status of the population
MEDV	Median value of owner-occupied homes in $1000's

Action Formula

S P D G

Select

- The variable we aim to predict is **MEDV**, which is equivalent to $1000 for a value of 1

Predict

Our task is to predict the outcome variable using 13 input **attributes** provided.

Data Analysis

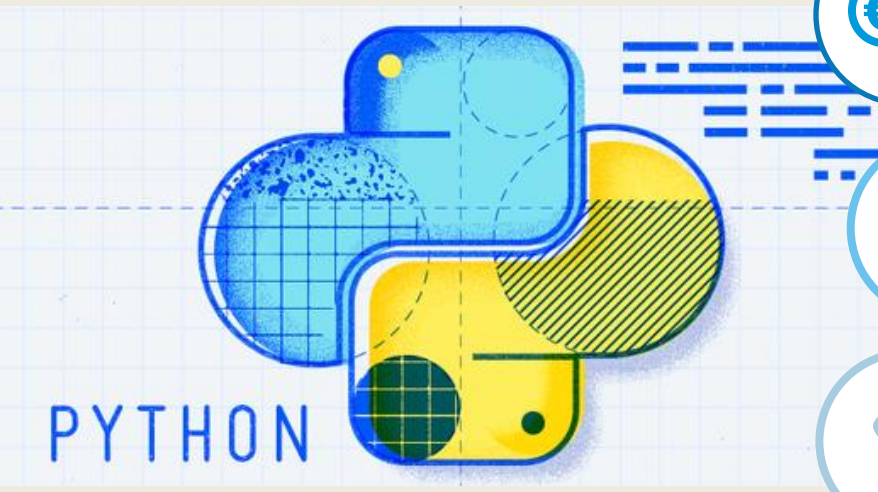
Discard

If necessary, we may discard certain attributes that do not contribute to the prediction of the target variable.

Generate

Furthermore, we may also generate new attributes from the given attributes.

Import Modules



- pandas - used to perform data manipulation and analysis
- numpy - used to perform a wide variety of mathematical operations on arrays
- matplotlib - used for data visualization and graphical plotting
- seaborn - built on top of matplotlib with similar functionalities
- %matplotlib - to enable the inline plotting

warnings - to manipulate warnings details

filterwarnings('ignore') is to ignore the warnings thrown by the modules (gives clean results)

Dataset Loading (Boston Dataset CSV File)

```
df = pd.read_csv("Boston Dataset.csv")
df.drop(columns=['Unnamed: 0'], axis=0, inplace=True)
df.head()
```

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	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

Conclusion : We have dropped the unnecessary column 'Unnamed : 0'.

Statistical Information

```
# statistical info
df.describe()
```

	crim	zn	indus	chas	nox	rm	age	dis	rad	
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.
75%	3.677082	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.

Conclusion :
There are no NULL values.
All other values are adequate.

Information of DataTypes

```
# datatype info
df.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    int64  
 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    int64  
 9   tax      506 non-null    int64  
 10  ptratio  506 non-null    float64
 11  black    506 non-null    float64
 12  lstat    506 non-null    float64
 13  medv     506 non-null    float64
dtypes: float64(11), int64(3)
memory usage: 55.5 KB
```

Conclusion :

- All the columns are in numerical datatype.
- We will create new categorical columns using the existing columns later

Dataset - Preprocessing

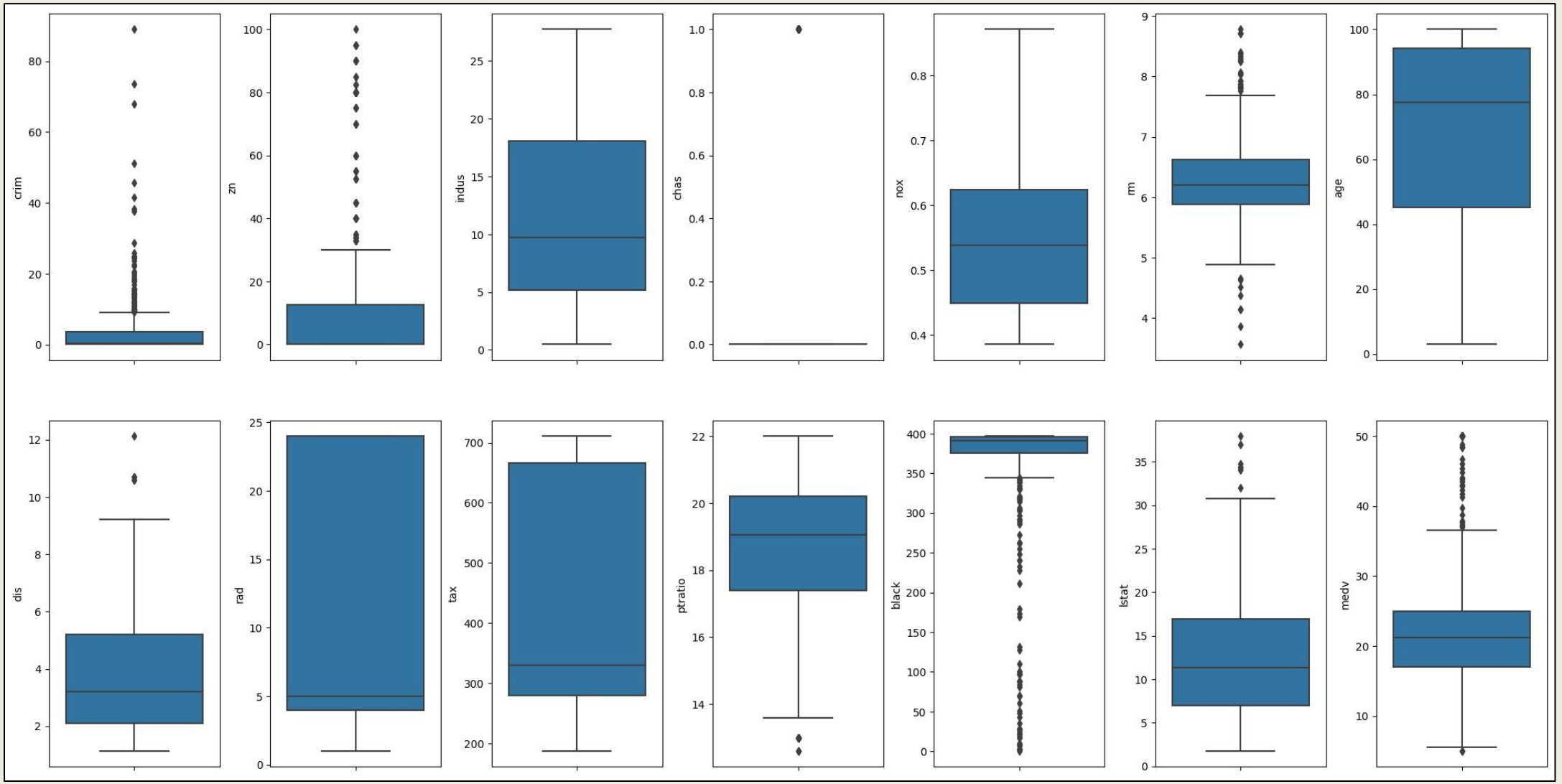
```
# check for null values
df.isnull().sum()
```

```
crim       0
zn         0
indus      0
chas       0
nox        0
rm         0
age        0
dis        0
rad        0
tax        0
ptratio    0
black      0
lstat      0
medv       0
dtype: int64
```

Conclusion : No NULL values were found.

Exploratory Data Analysis

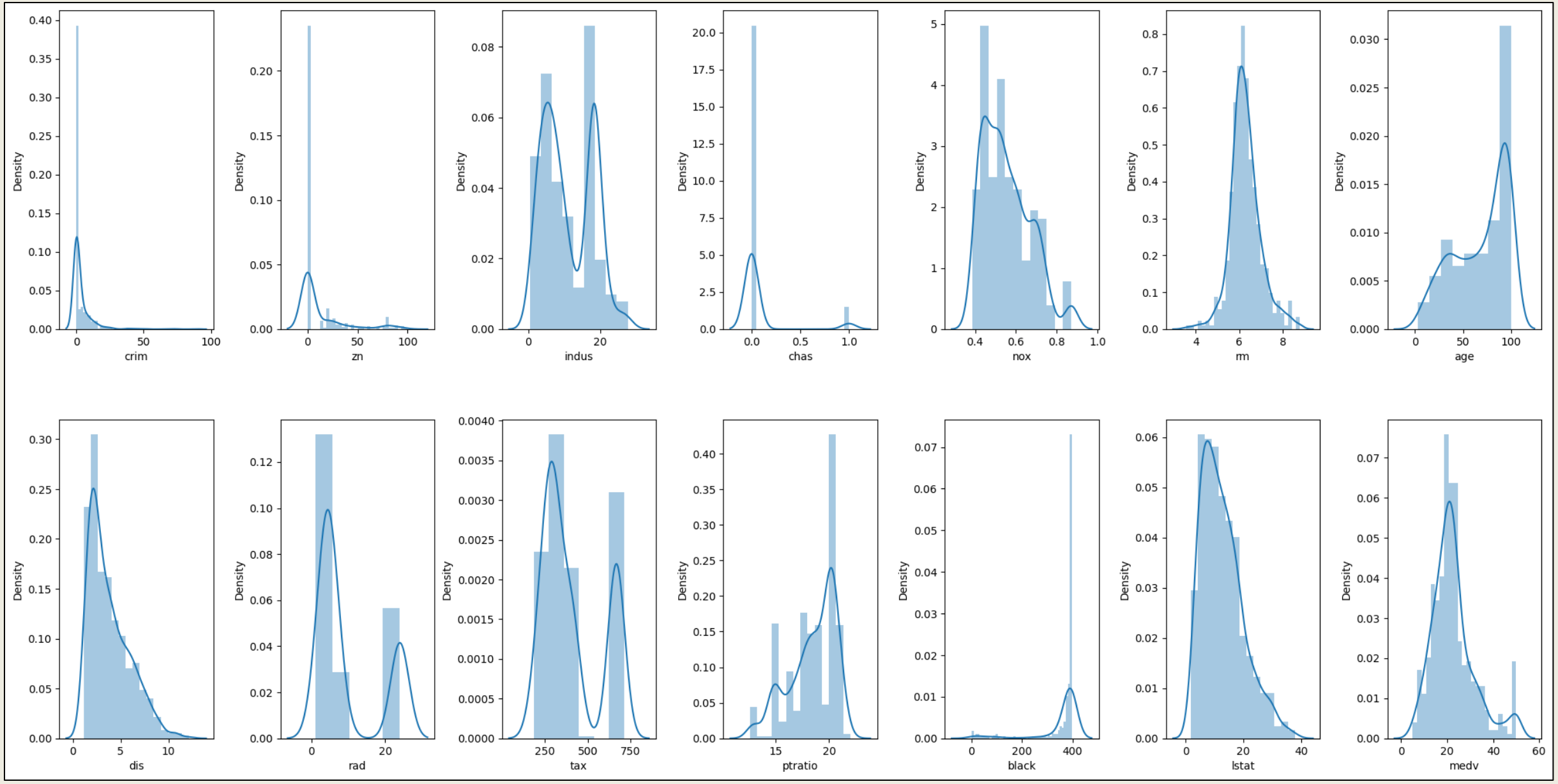
- Let us create box plots for all columns to identify the outliers.



Exploratory Data Analysis (EDA)

- In the graph, the dots represent the **outliers**.
- ‘Crim’, ‘zn’, ‘black’ has so many outliers and does not follow the normal distribution.
- The column containing many outliers does not follow the normal distribution.
- We can minimize outliers with log transformation or min max normalization
- We can also drop the column which contains outliers (or) we can delete the rows which contains the same.

Exploratory Data Analysis (EDA)



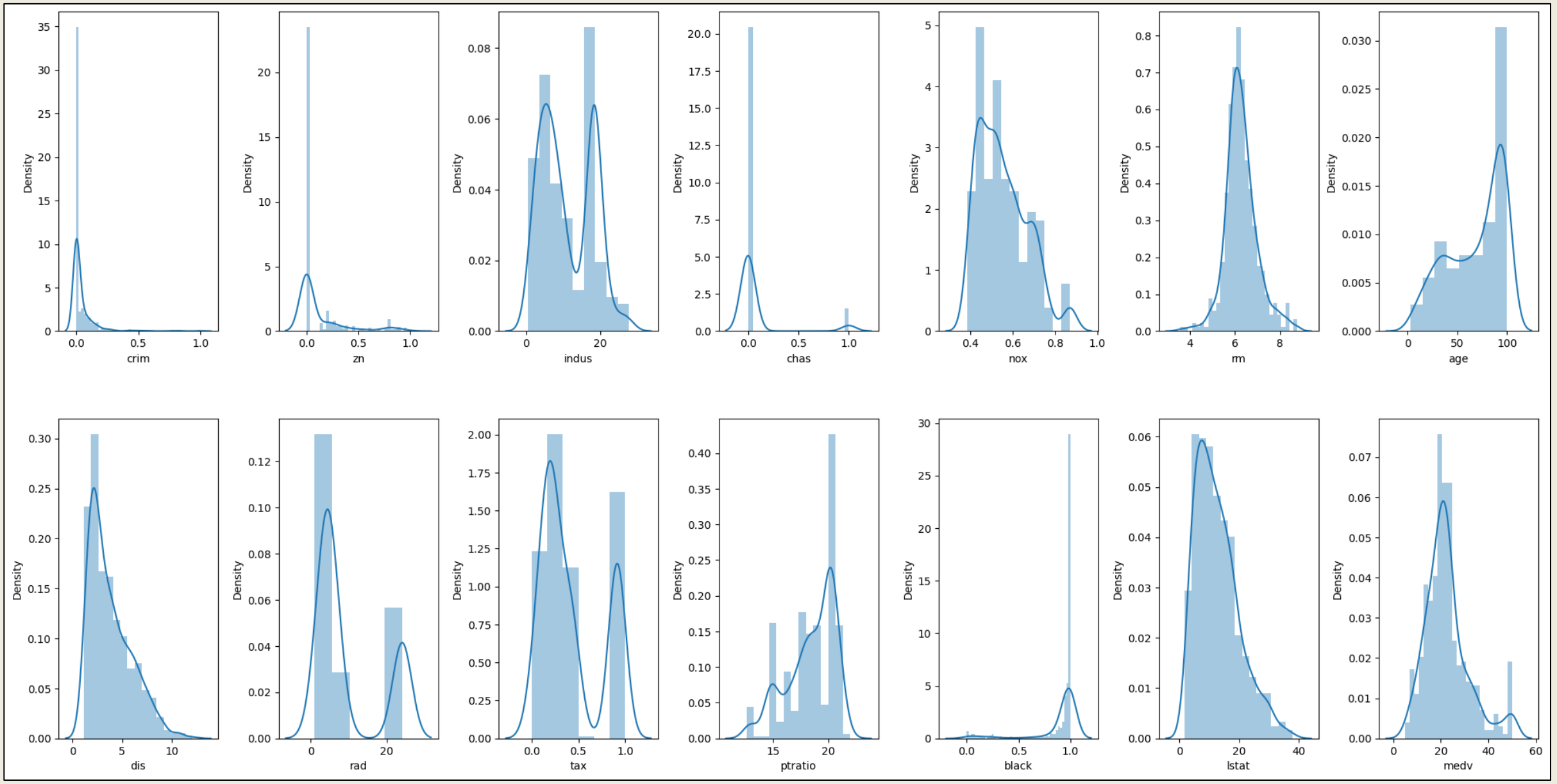
Exploratory Data Analysis (EDA) Min-Max Normalization.

- We can observe right skewed and left skewed graphs for 'crim', 'zn', 'tax', and 'black'.
- Therefore, we need to normalize these data.
- We will create the column list for the 4 columns and use Min-Max Normalization.
- The last line shows the formula for min-max normalization.

```
cols = ['crim', 'zn', 'tax', 'black']
for col in cols:
    # find minimum and maximum of that column
    minimum = min(df[col])
    maximum = max(df[col])
    df[col] = (df[col] - minimum) / (maximum - minimum)
```

- Now the range of these columns is between 0 to 1.
- Min-Max Normalization transformed the maximum value as '1' and the minimum value as '0'.

Min-Max Normalization



Exploratory Data Analysis (EDA) - Standardization

- Standardization uses mean and standard deviation. Here, preprocessing.StandardScaler( ) is the standardization function.

```
# standardization
from sklearn import preprocessing
scalar = preprocessing.StandardScaler()

# fit our data
scaled_cols = scalar.fit_transform(df[cols])
scaled_cols = pd.DataFrame(scaled_cols, columns=cols)
scaled_cols.head()
```

	crim	zn	tax	black
0	-0.419782	0.284830	-0.666608	0.441052
1	-0.417339	-0.487722	-0.987329	0.441052
2	-0.417342	-0.487722	-0.987329	0.396427
3	-0.416750	-0.487722	-1.106115	0.416163
4	-0.412482	-0.487722	-1.106115	0.441052

Standardized values.

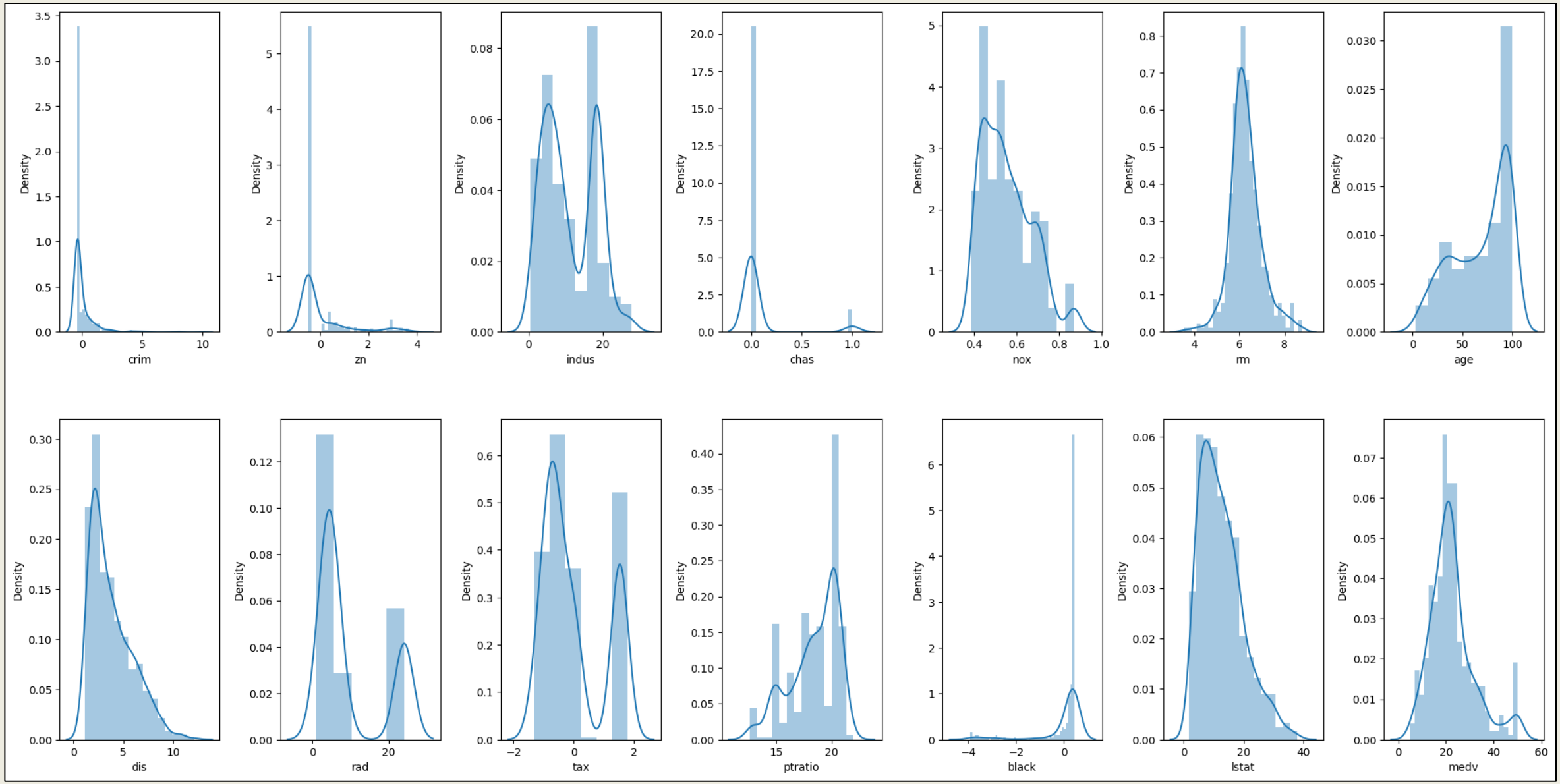
Let us get back to our Original database.

```
for col in cols: df[col] = scaled_cols[col]
```

•This code will assign the standardized value to the original data frame.

Even now the columns '**crim**', '**zn**', '**tax**', and '**black**' does not show a perfect normal distribution. However, the standardized value of these columns will slightly improve the model performance.

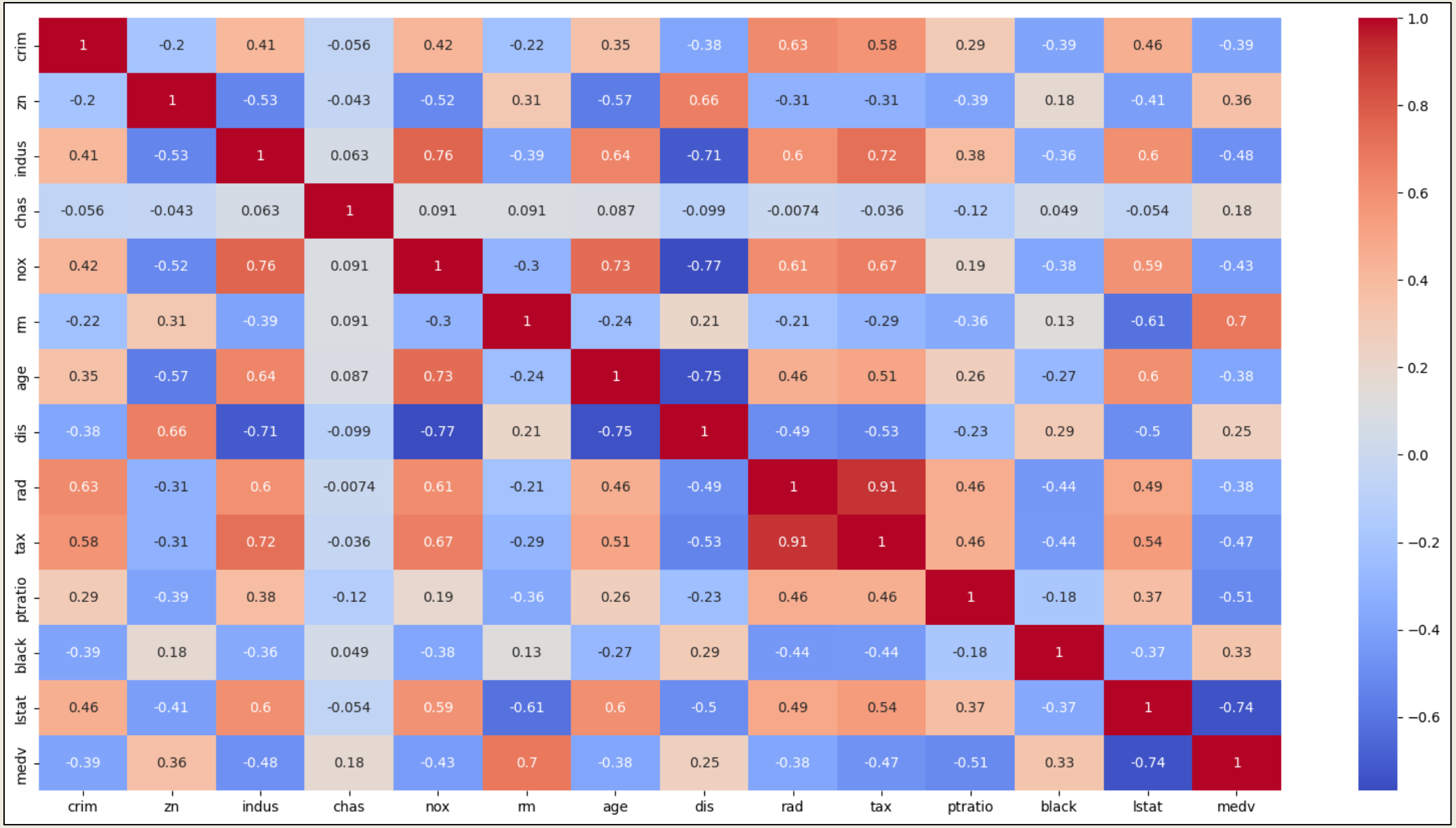
Standardization Output



Correlation Matrix

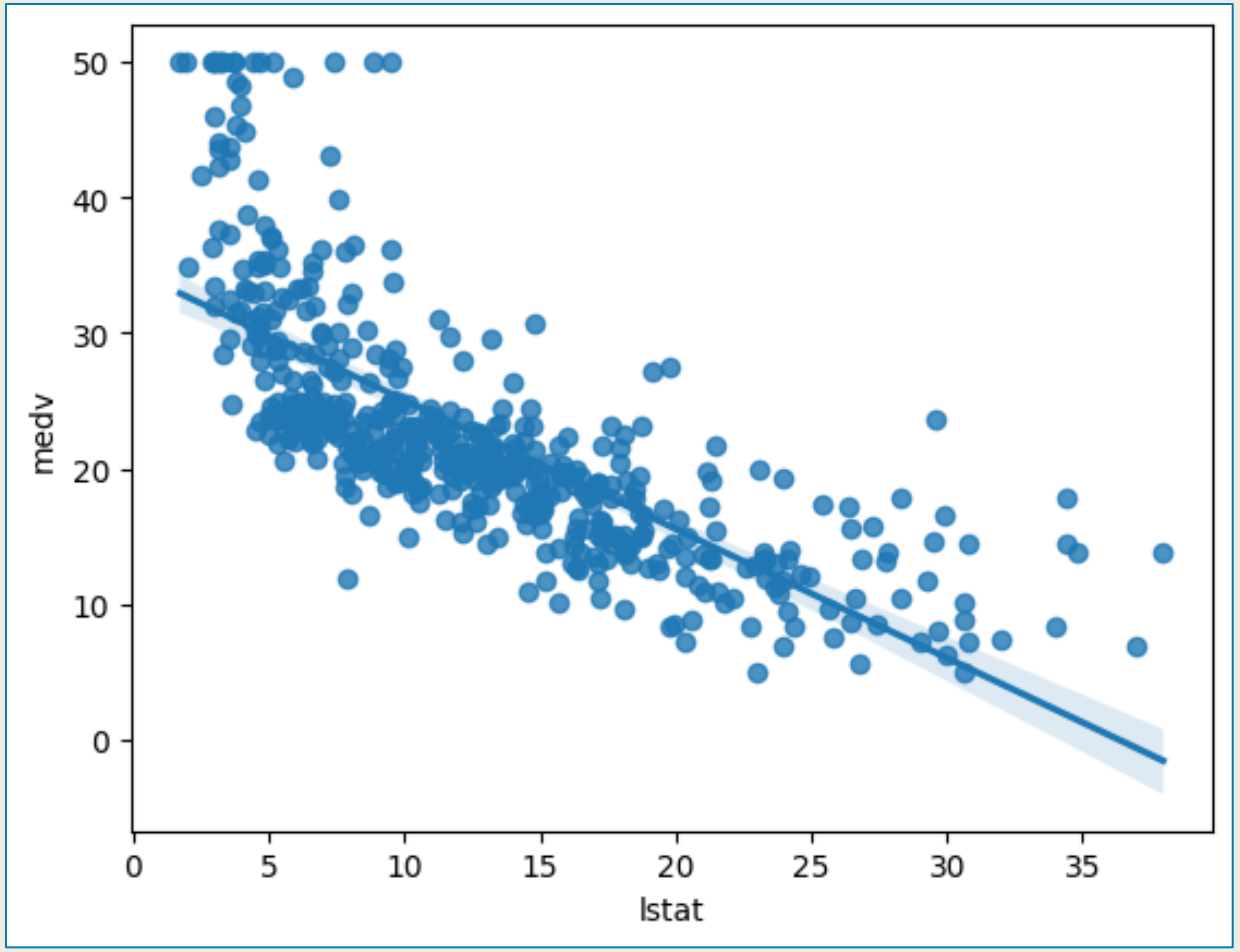
- We mostly focus on the target variable as this is a Regression problem.
- But we can also observe other highly correlated attributes by column **'tax'** and **'rad'**.
- We will later eliminate this correlation by ignoring any of the variables.
- Additionally, we will display **'lstat'** and **'rm'** to show their correlation with the target variable **'medv'**.

Correlation Matrix

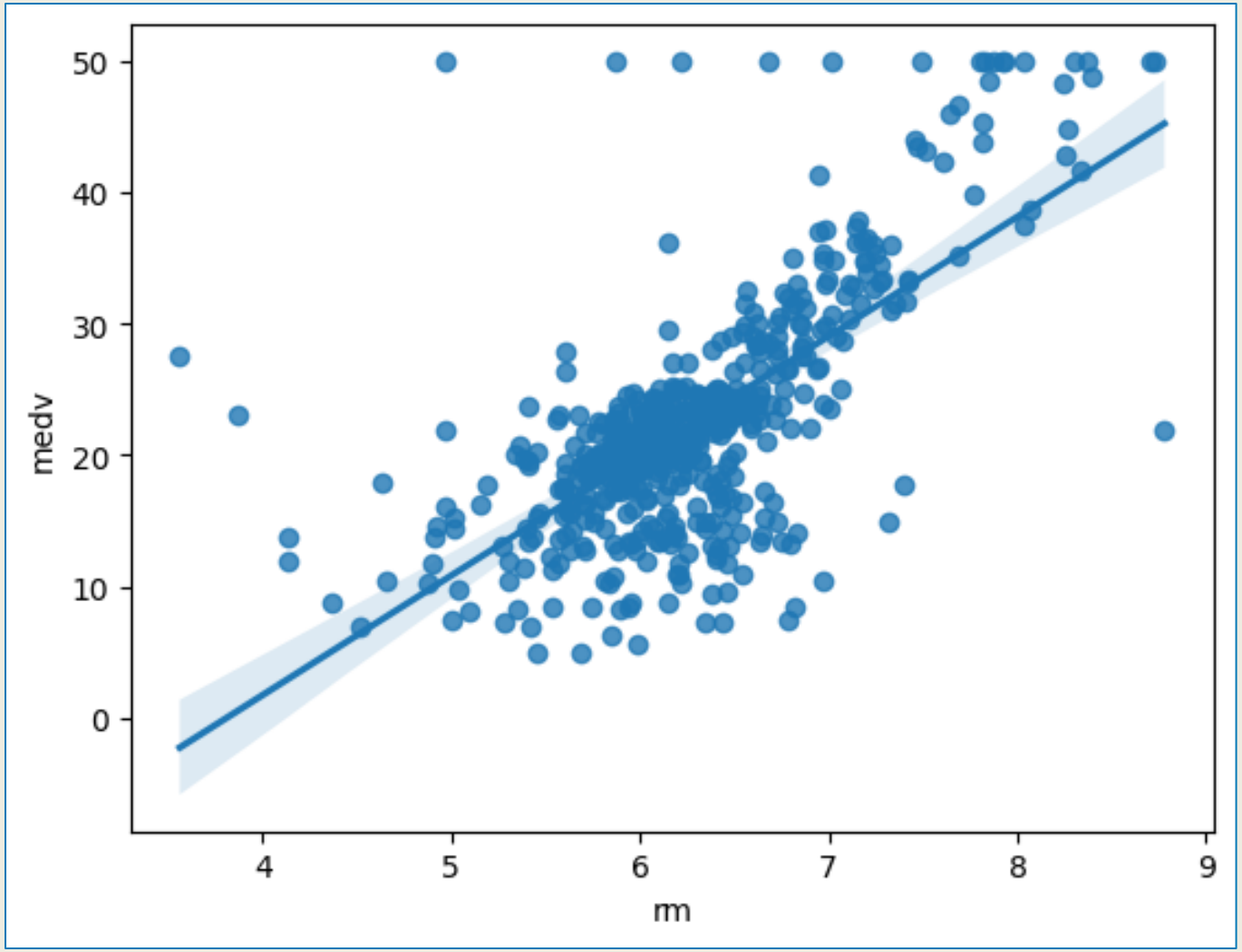


Relation between Target and Correlated Variables

- Here, the price of houses decreases with the increase in the **'lstat'**. Hence it is negatively correlated.



- Here, the prices of houses increase with the increase in **'rm'**. Hence it is positively correlated



Model Training

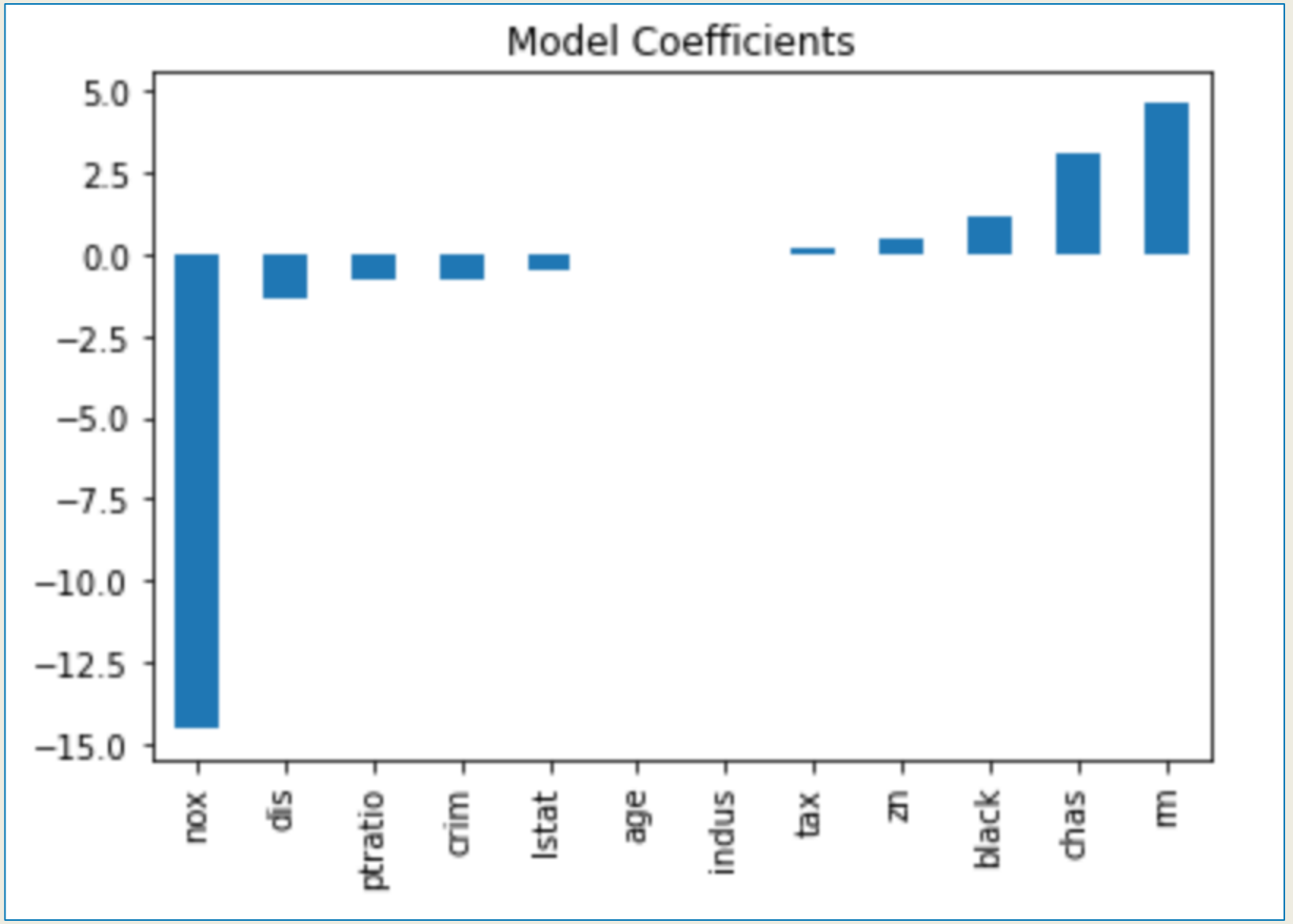
- **Now let's import functions to train models.**
- Instead of training the whole model, we will split the dataset for estimating the model performance.
- If you train and test the dataset completely, the results will be inaccurate. Hence, we will use '**train_test_split**'.
- We will add **random_state** with the attribute 42 to get the same split upon re-running.
- If you don't specify a random state, it will randomly split the data upon re-running giving inconsistent results.
- We use '**cross val score**' for better validation of the model.
- Here, **cv=5** means that the cross-validation will split the data into 5 parts.
- **np.abs** will convert the negative score to positive and **np.mean** will give the average value of 5 scores.

Linear Regression:

- Mean Squared Error is around 23 and Cross-Validation Score is around 35.

```
Model Report
MSE: 23.871005067364884
CV Score: 35.58136621076919
```

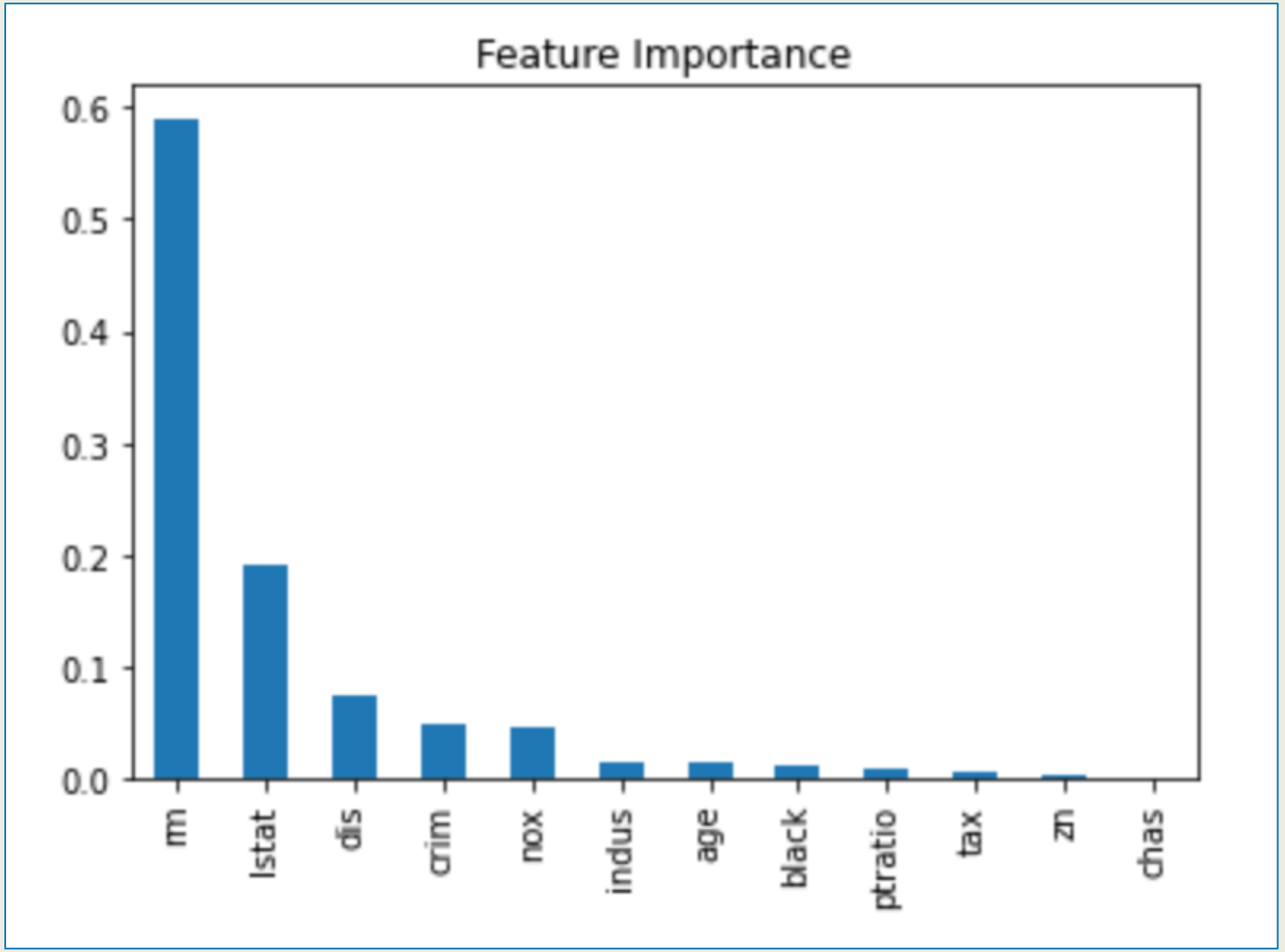
rm shows a high positive coefficient and **nox** shows a high negative coefficient.



Decision Tree:

- Here CV Score is higher than Linear Regression.

```
Model Report
MSE: 10.505354330708663
CV Score: 41.54329800038827
```

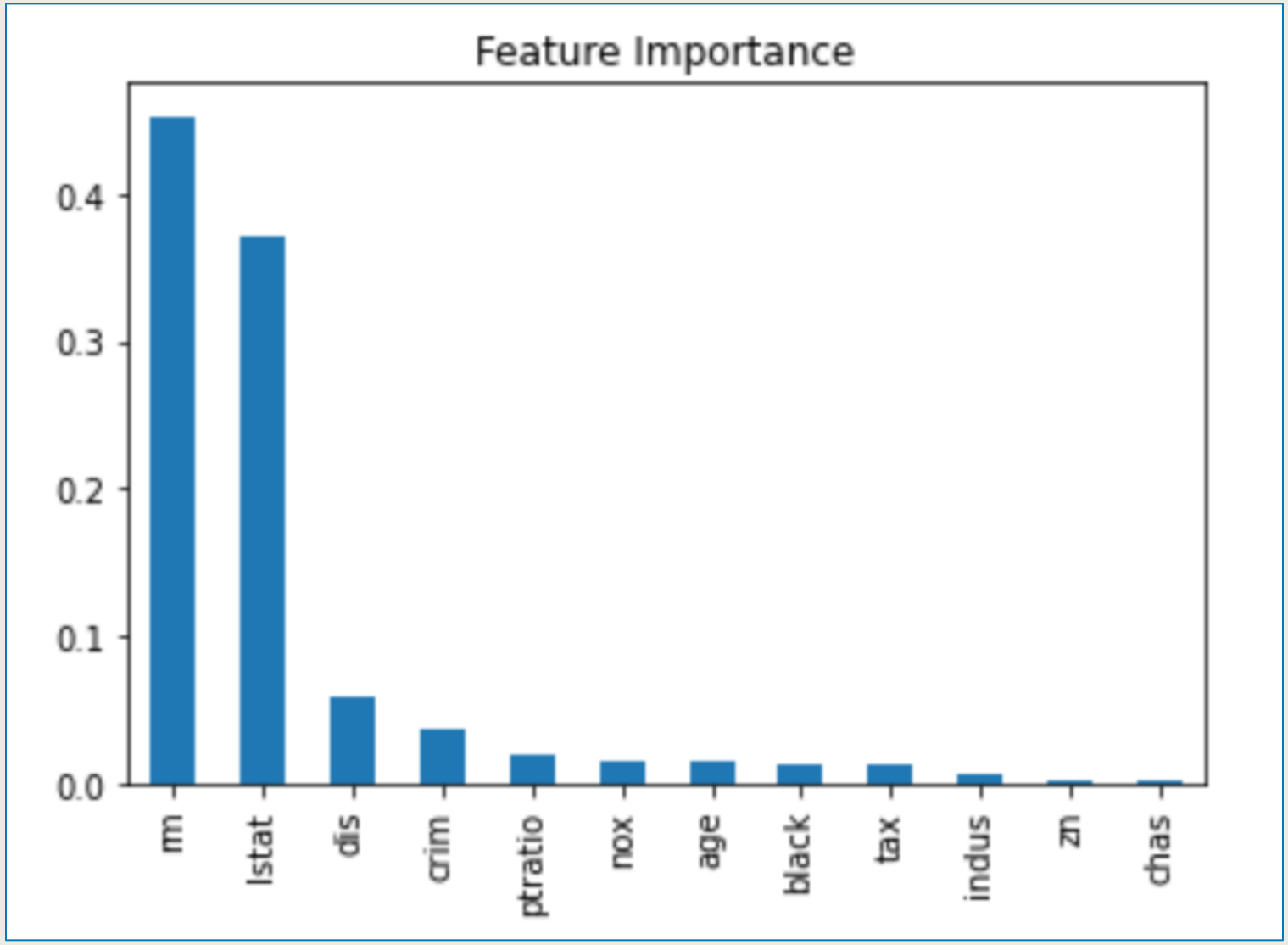


Random Forest:

We know that **'rm'** and **'lstat'** produces much correlation with the target variable. That is the reason behind its higher feature importance among other attributes.

```
Model Report
MSE: 10.72660739370079
CV Score: 21.943030949718498
```

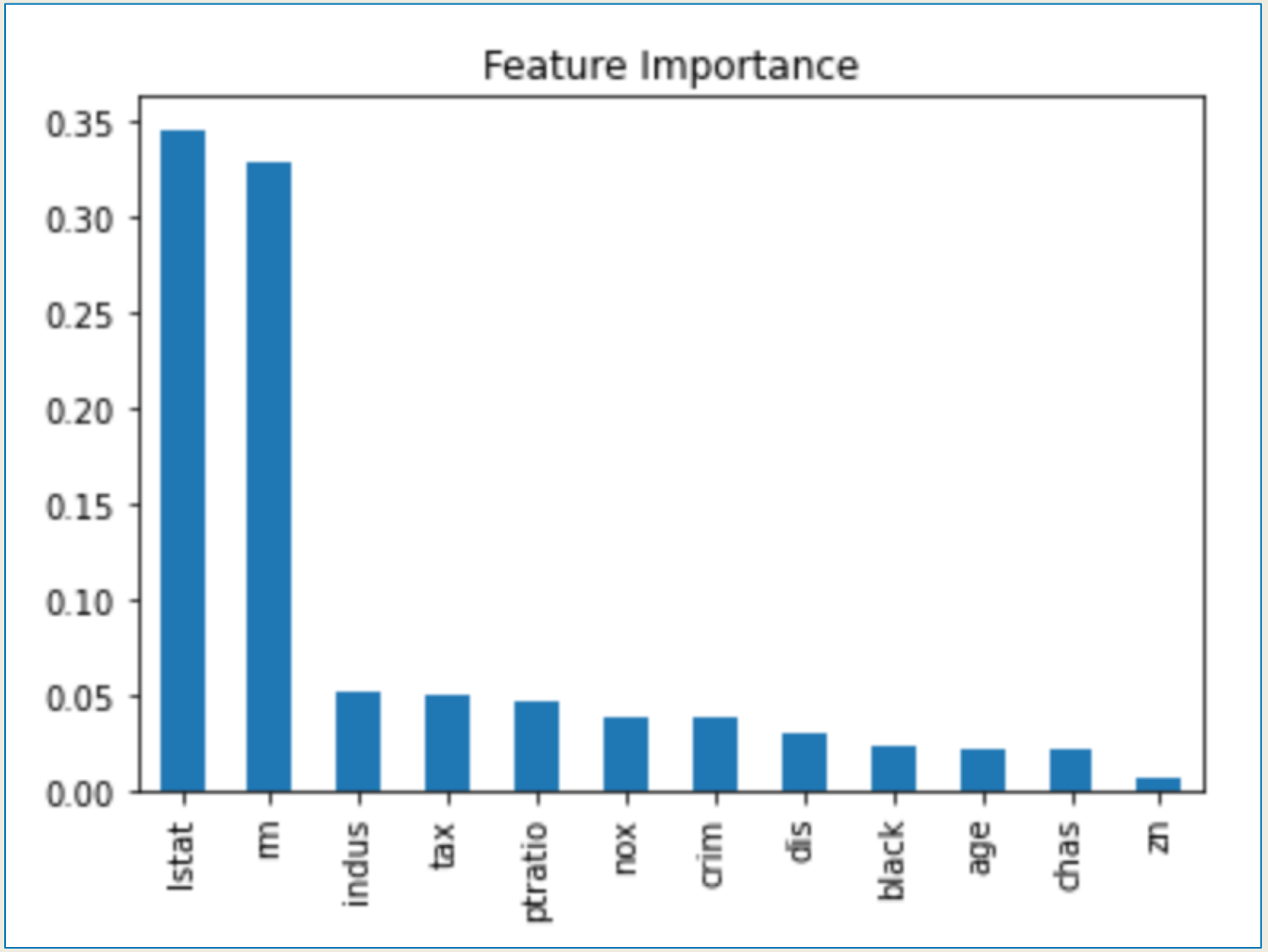
MSE is around 10 and CV score is around 21.



Extra Trees:

```
Model Report
MSE: 10.913022118110243
CV Score: 20.266603599126373
```

Here MSE is similar to Random Forrest and the cross-validation score is decreased.

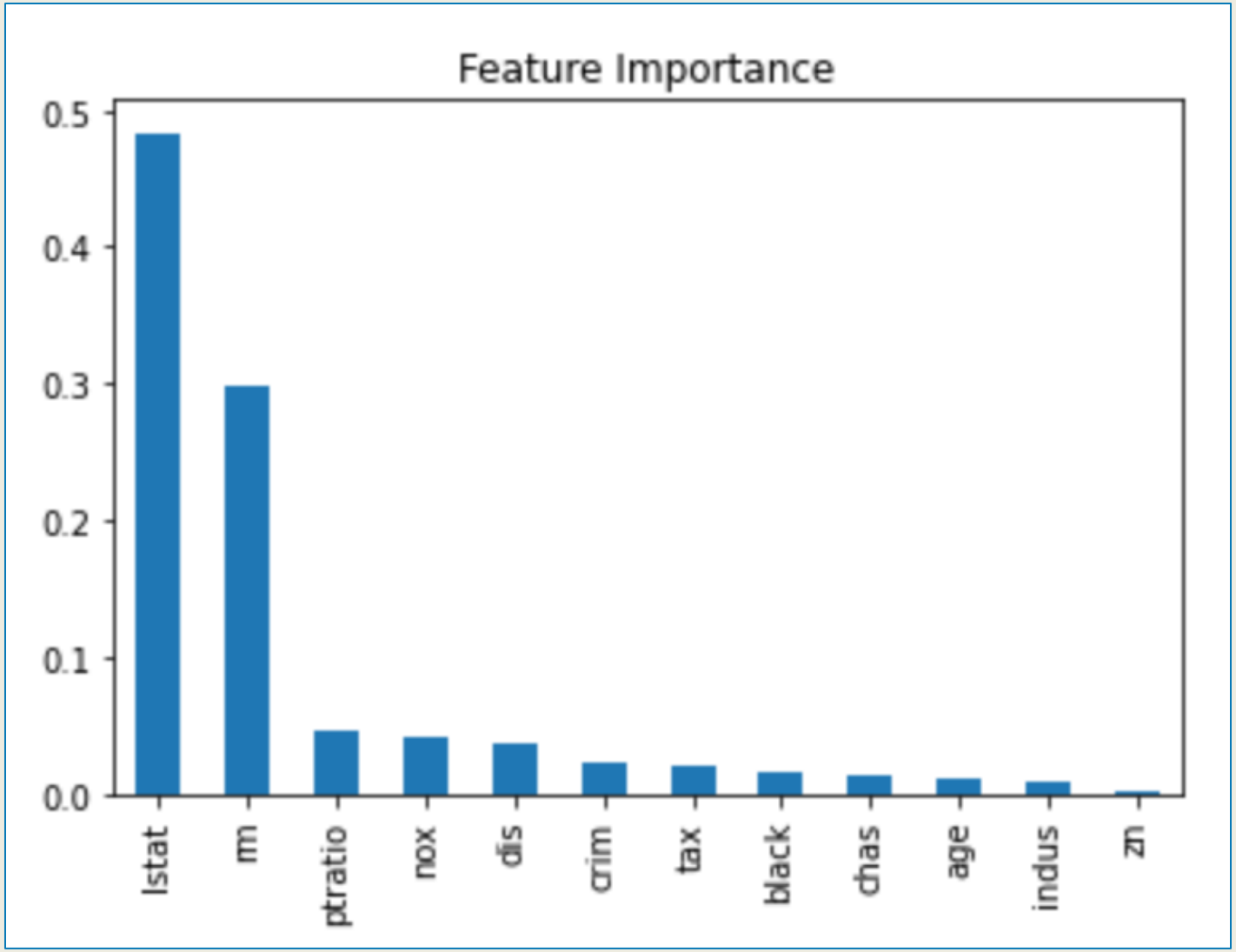


XGBoost:

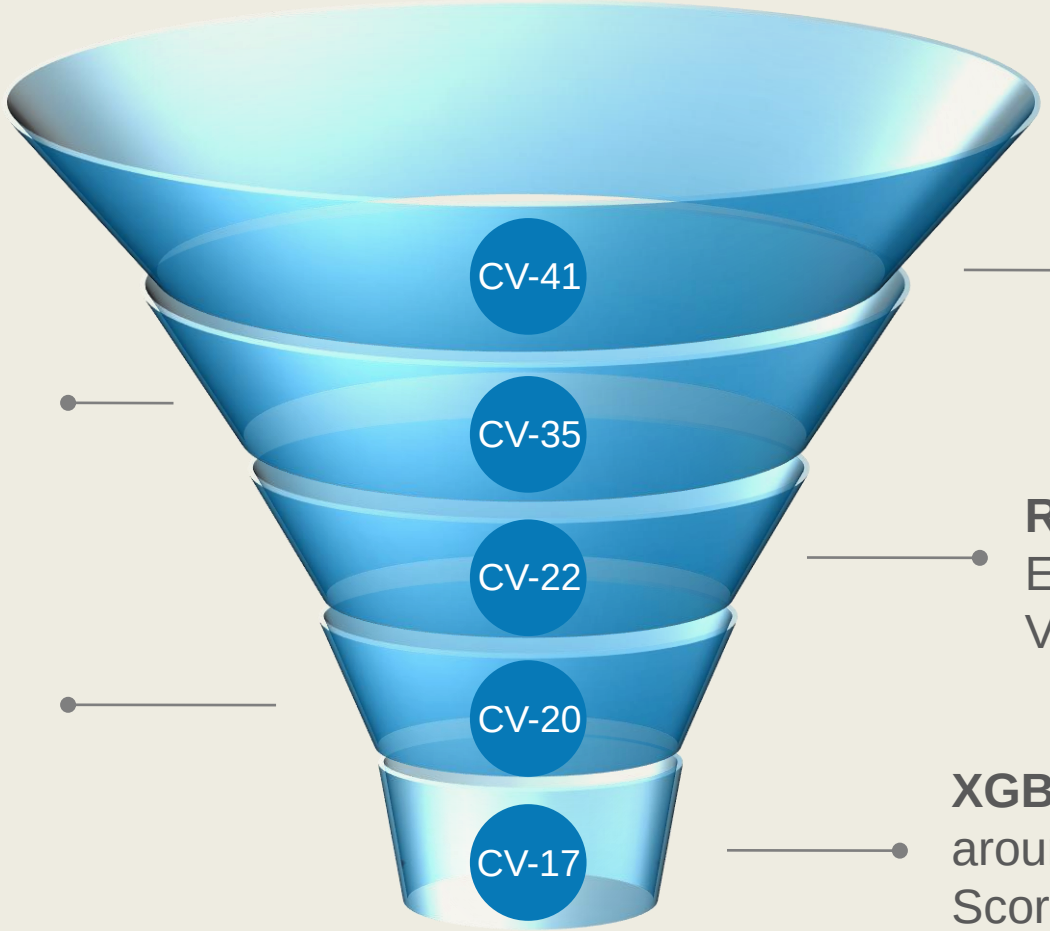
```
Model Report
MSE: 10.032316106542387
CV Score: 17.971291041858336
```

MSE, as well as CV Score, is the least among all other models.

The XGBoost shows the best result with minimal cross validation error.



Comparison Diagram and Summary :



Linear Regression: Mean Squared Error is around 23 and Cross-Validation Score is around 35.

Extra Trees: Mean Squared Error is around 10 and Cross-Validation Score is around 20.

Decision Tree: Mean Squared Error is around 10 and Cross-Validation Score is around 41.

Random Forest: Mean Squared Error is around 10 and Cross-Validation Score is around 22.

XGBoost : Mean Squared Error is around 10 and Cross-Validation Score is around 17.

- To summarize, XGBoost Regressor works best for this project.
- We can further improve the model by creating new attributes and performing hyperparameter tuning.
- We can create a new categorical attribute with the help of an existing numerical attributes.

Questions: ??