

Introduction

22 September 2022 16:54

- Types of **regression** in **supervised** machine learning.
- We already know that, Regression is understanding the **relationship** Between Independent and dependent variable.
- Here, **Predictor** variable(independent/input variable) and **dependent** variable **related linearly** to each other.
- We try to find the **relation** between independent and dependent variables

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

Y = Dependent Variable

Bo = Population Y-intercept (Point where graph intercept with y-axis)

B1 = Population slope/ Coefficient

X = Independent variable

E = Error term (in the equation or residual)

- Dependent variable called as "source of truth"
- Linear regression also called as **Ordinary Least square** and **Linear least square**
- Is the real-work horse of the regression world

Note:

What is Least square method in statistics?

The least-square method states that the curve that best fits a given set of observations, is said to be a curve having a minimum sum of the squared residuals (or deviations or errors) from the given data points

β_0

Understanding Population Y-intercept:

- The intercept is the value of our target variable when all our features are zero and our function crosses the y-axis.
- **Practical Example:**





- Suppose that cooking a famous dish of mine takes me exactly 10 minutes for putting everything in place
 - and then 5 more minutes of finely chopping ingredients for every diner,
 - plus 10 more minutes for actually cooking the dish on the oven.
 - The slope, in this case, would be 5 and the intercept would be 20 - from the 10 fixed minutes of setting up everything and 10 minutes of cooking
 - Equation should be translated into **$Y = 5X + 20$**
- If we receiving 4 guests, Now we need to cook for 5 people in total,
- Equation should be changed into **$Y = 5 \cdot 5 + 20 = 45 \text{ min.}$**

If there are multiple variables such as,

- B1 - Time taken for finely chopping
- B2 - Time taken for extra shopping for dinner
- B3 - Extra time taken for cooking for additional guests
- Then,
- $Y = B_1x + B_2x + B_3x + \dots + B_nx + B_0$

β_1

Understanding the Co-efficient:

- The prediction of the machine is depend on the coefficient.
- coefficient indicates the direction of the relationship between a predictor variable and the response variable.
 - A positive sign indicates that as the predictor variable(y)increases, the response variable(X) also increases.

- A negative sign indicates that as the predictor variable(y) increases, the response variable(x) decreases.

Role of Co-efficient in Simple Linear Regression:

- $y = b + cx$
- .x and y are known to the machine
- .b as y-intercept we discussed earlier.
- .c as co-efficient or weight, determined by the hit and trial method by the machine
- It consider each row to compute the relation. Compare with all the data.

Example,

| Duration of self-study | Marks in Exam |
|------------------------|---------------|
| 2 | 20 |
| 3 | 30 |
| 4 | 40 |
| 7 | 70 |

- $20/2 = 10$
- $30/3 = 10$
- $40/4 = 10$
- $70/7 = 10$
- We can say, **10 is common relation** from which we can predict the value of y from the x.
- If a person study for 8hrs, then he/she will get $y = 10 \times 8 + 0 = 80$ Marks

Role of Co-efficient in Multi Linear Regression:

Example:

| Duration of Group Study | Duration of self-study | Marks in Exam |
|-------------------------|------------------------|---------------|
| 1 | 2 | 20 |
| 2 | 3 | 30 |
| 3 | 4 | 40 |
| 4 | 7 | 70 |

- In this example Duration of Group study (x1)and Duration of self-study (x2) both effects the prediction of marks(y). x1 and x2 have coefficient c1 and c2.

Hypothesis of Linear Regression

22 September 2022 17:34

The linear regression model can be represented by the following equation:

The diagram shows the linear regression equation $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p + \varepsilon$. Annotations include: a green arrow pointing to x_1 labeled 'predictor, 'x-variable', independent variable, explanatory variable'; an orange arrow pointing to β_2 labeled 'coefficient'; a blue bracket under the terms $\beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$ labeled 'linear predictor'; a red arrow pointing to Y labeled 'response, dependent variable, observation, 'y-variable''; and a purple arrow pointing to ε labeled 'random error, "noise"'.

Y is the actual value

β_0 is the bias term.

β_1, \dots, β_p are the model parameters

x_1, x_2, \dots, x_p are the feature values.

As β_0 is the kind of constant (C) we seen in equation $Y = MX + C$

Case:

If ML wants to calculate on sales of an electronic shop. Then,

Y = sales of an electronic shop

X_1 = sales of TV

$B_1 = 0.001$

X_2 = sales of Radio

$B_2 = 0.1$, (As both B_1 and B_2 calculated

by ML)

By this way, ML find out the individual sales of TV and Radio and forecast fir upcoming years for business

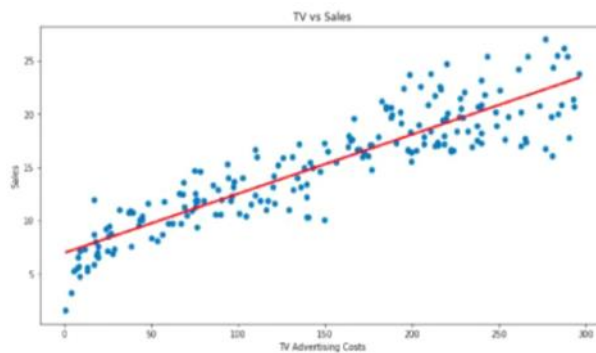
Case - ROI of TV ads

22 September 2022 17:38

Looking at the past investment on TV ads vs Sales done by company. Company wants to estimate return on investment (ROI) which can be calculated by estimating sales through past investment on TV ads

Goal of linear regression:

1. To plot the graph between Tv ads vs sales

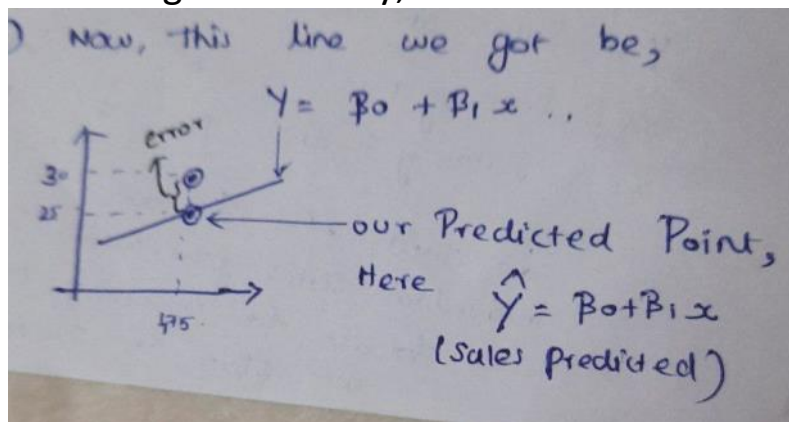


2. To create a **trend line or best fit line** based on the past data of investment on TV ads vs Sales.

Trend Line:

which can then be used to confirm or deny the relationship whether TV ads create impact on Sales and also predict Sales based on TV ads.

3. Now, Predict for our value.
 - i. Here we have only 300 tv ads
 - ii. Now we want to predict any value above that e.g. 400. How?
4. Predicting the sales by,



How we get
 y ?

→ Adding Error term ϵ

(ie) $y = \hat{y} + \epsilon$

$$y = \beta_0 + \beta_1 x + \epsilon$$

How β value
calculated?

can β can be
+ve / -ve? Yes.

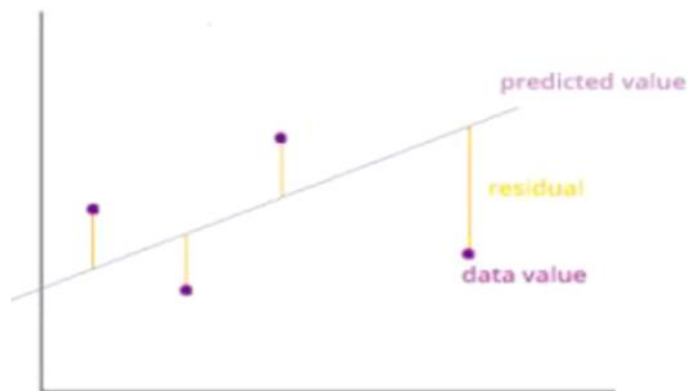
concept like
gradient descent,
comes to the
Picture,

The Best fit line

22 September 2022 18:06

How do we find the best fit line?

- Line for which the **error** between the predicted values and the observed values is minimum.
- The best fit line (or) **regression line**
- Errors are called as **Residuals**



$$\min(\text{SSE}) = \sum_{i=1}^n (\text{actual output} - \text{predicted output})^2$$

SSE : Sum square error

$$\text{SSE : } \sum_{i=1}^n (y - \hat{y})^2$$

How it works?

- ML works for multiple iterations
- In the end, minimum value of SSE is considered as best fit line

Why Squared?

- Predicted value can have positive or negative error.
- If we don't square the error, then the positive and negative points will cancel each other out.
- Absolute function($|\text{Actual} - \text{Pred}|$) can be one option but it is not differentiable at $\text{Actual} = \text{Pred}$

Predicted output

22 September 2022 18:19

- Linear regression equation **without error term** gives predicted output.

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \dots + \beta_p X_p$$

- By varying different parameter values (β_p), predicted value can be determined and value which gives minimum SSE will produce the best fit line

Least Square Estimators of Parameters

22 September 2022 18:25

The method of least squares chooses estimators of β that minimize SSE. To determine these estimators, we differentiate SSE first with respect to β_0 and then to β_1 as follows

$$\text{SSE} : \sum_{i=1}^n (y - \hat{y})^2$$

$$Y = \beta_0 + \beta_1 X + \epsilon$$

$$\frac{\partial SSE}{\partial \beta_0} = -2 \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)$$

$$\frac{\partial SSE}{\partial \beta_1} = -2 \sum_{i=1}^n x_i \cdot (y_i - \beta_0 - \beta_1 x_i)$$

Setting these partial derivatives equal to zero yields the following equations for the minimizing values β_0 and β_1 ,

$$\sum_{i=1}^n y_i = n\beta_0 + \beta_1 \sum_{i=1}^n x_i$$

$$\sum_{i=1}^n x_i y_i = \beta_0 \sum_{i=1}^n x_i + \beta_1 \sum_{i=1}^n x_i^2$$

Above equations are called as "**Normal Equation**"

The normal equation is a closed-form solution used to **find the value of β that minimizes the cost function**. Another way to describe the normal equation is as a one-step algorithm used to analytically find the coefficients that minimize the loss function.

Note:

What is the difference between cost function and loss function?

- There is no major difference.
- In other words, the loss function is to capture the difference between the actual and predicted values for a single record whereas cost functions aggregate the difference for the entire training dataset.

If we let

$$\bar{y} = \sum_i \frac{y_i}{n}$$

$$\bar{x} = \sum_i \frac{x_i}{n}$$

Now, the first normal equation as

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

Substituting this β_0 in the second normal equation,

$$\sum_i x_i y_i = (\bar{y} - \beta_1 \bar{x}) \cdot n \bar{x} + \beta_1 \sum_i x_i^2$$

Or

$$\beta_1 \left(\sum_i x_i^2 - n \bar{x}^2 \right) = \sum_i x_i y_i - n \cdot \bar{x} \bar{y}$$

$$\beta_1 = \frac{\left(\sum_i x_i y_i - n \cdot \bar{x} \cdot \bar{y} \right)}{\left(\sum_i x_i^2 - n \cdot (\bar{x})^2 \right)}$$

$$\beta_1 = \frac{n \sum_i x_i y_i - \sum_i x_i \sum_i y_i}{n \sum_i x_i^2 - \left(\sum_i x_i \right)^2}$$

Case:

Example: Sam found how many **hours of sunshine** vs how many **ice creams** were sold at the shop from Monday to Friday:



| "x" Hours of Sunshine | "y" Ice Creams Sold |
|-----------------------------|---------------------------|
| 2 | 4 |
| 3 | 5 |
| 5 | 7 |
| 7 | 10 |
| 9 | 15 |

Let us find the best **m** (slope) and **b** (y-intercept) that suits that data

$$y = mx + b$$

$$Y = MX + C$$

$$\text{similar to, } Y = B_0X + B_1$$

Applying the linear regression, finding the best fit line,

Step 1: For each (x,y) calculate x^2 and xy :

| x | y | x^2 | xy |
|---|----|-------|-----|
| 2 | 4 | 4 | 8 |
| 3 | 5 | 9 | 15 |
| 5 | 7 | 25 | 35 |
| 7 | 10 | 49 | 70 |
| 9 | 15 | 81 | 135 |

Step 2: Sum x, y, x^2 and xy (gives us Σx , Σy , Σx^2 and Σxy):

| x | y | x^2 | xy |
|----------------------------------|----------------------------------|-------------------------------------|------------------------------------|
| 2 | 4 | 4 | 8 |
| 3 | 5 | 9 | 15 |
| 5 | 7 | 25 | 35 |
| 7 | 10 | 49 | 70 |
| 9 | 15 | 81 | 135 |
| Σx: 26 | Σy: 41 | Σx^2: 168 | Σxy: 263 |

Also **N** (number of data values) = 5

Step 3: Calculate Slope **B_0**

$$\begin{aligned}
 B_0 &= \frac{N \Sigma(xy) - \Sigma x \Sigma y}{N \Sigma(x^2) - (\Sigma x)^2} \\
 &= \frac{5 \times 263 - 26 \times 41}{5 \times 168 - 26^2} \\
 &= \frac{1315 - 1066}{840 - 676} \\
 &= \frac{249}{164} = 1.5183...
 \end{aligned}$$

Step 4: Calculate Intercept **B_1**

$$\begin{aligned}
 B_1 &= \frac{\Sigma y - B_0 \Sigma x}{N} \\
 &= \frac{41 - 1.5183 \times 26}{5} \\
 &= 0.3049...
 \end{aligned}$$

Step 5: Assemble the equation of a line:

$$Y = B_0X + B_1$$

$$y = 1.518x + 0.305$$

In this case, we have only one independent variable/ feature.

What if we had many independent variables?

Ans: We had Gradient descent and we see it later.

Gradient descent

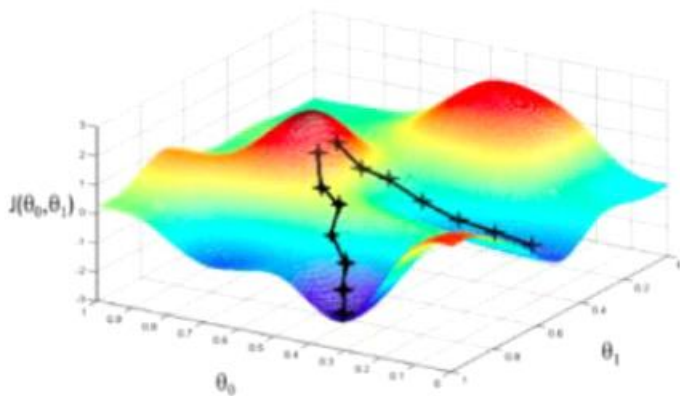
22 September 2022 18:51

With infinite possibilities of values which parameter can take, how do we arrive at the best possible β with minimum iteration?

Gradient Descent:

Gradient Descent is an optimization algorithm that helps machine learning models to find out paths to a minimum value using repeated steps. Gradient descent is used to minimize a function so that it gives the lowest output of that function. This function is called the **Loss Function**.

Relate the gradient descent with real-life analogy:



- Think of a valley you would like to descend when you are blind-folded.
- Any sane human will take a step and look for the slope of the valley, whether it goes up or down.
- Once you are sure of the downward slope you will follow that and repeat the step again and again until you have descended completely (or reached the minima)

This is exactly what happens in gradient descent,

- The inclined and/or irregular is the **cost function** when it is plotted.
- The role of gradient descent is to provide direction and the velocity (learning rate) of the movement in order to attain the minima of the function i.e. where the cost is minimum

How does the gradient descent works?

- primary goal minimize / maximize the cost function
- Minimizing/maximize cost functions will also **result in a lower error** between the predicted values and the actual values which also denotes that the algorithm has performed well in learning

The function which is used to **minimize for linear regression model** is the **mean squared error**.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2.$$

MSE is used instead of SSE to save space and avoid memory explosion with large # of observations(n)

Initialize

22 September 2022 19:25

Regression Equation : $\text{Sales}(y) = \beta_0 + \beta_1 * \text{TV ads}(X)$

$$\text{Loss function : } J(\beta_0, \beta_1) = \frac{1}{2n} \sum_{i=1}^n (y - (\beta_0 + \beta_1 X))^2$$

Let us now start by initializing β_0 and β_1 to any value, say 0 for both, and start from there. The algorithm is as follows:

$$\beta_j := \beta_j - \alpha \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_j} \quad (\text{For } j = 0 \text{ and } j = 1)$$

where α , alpha, is the **learning rate**, or how rapidly do we want to move towards the minimum. We can always overshoot if the value of α is too large.

Delta

Delta value change in parameter is the derivative which refers to the slope of the function. It also gives us to know the direction (sign) in which the coefficient values should move so that they attain a lower cost on the following iteration.

$$\begin{aligned} \beta_0 : \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_0} &= -\frac{1}{n} \sum_{i=1}^n (y - (\beta_0 + \beta_1 X)) \\ \beta_1 : \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_1} &= -\frac{1}{n} \sum_{i=1}^n (y - (\beta_0 + \beta_1 X)) \cdot X \end{aligned}$$

Note:

$$\beta_0 = \frac{1}{2n} \sum_{i=1}^n (y - (\beta_0 + \beta_1 x))^2$$

$$\frac{\partial J_0}{\partial \beta_0} = \frac{1}{2n} \times 2 \sum_{i=1}^n (0 - 1 + 0)$$

$$= \frac{1}{n} \times 2 \times (-1)$$

$$= + \frac{1}{n} \times (-1)$$

$$= -1/n$$

$$\beta_0 : \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_0} = (-1) \left(\sum_{i=1}^n (y - (\beta_0 + \beta_1 x)) \right)$$

$$\beta_1 : \frac{\partial J(\beta_0, \beta_1)}{\partial \beta_1} = \left(\sum_{i=1}^n (y - (\beta_0 + \beta_1 x)) \right) \cdot (-x)$$

Repeat until Convergence

- Once we know the direction from the derivative
- we can update the coefficient values.
- Now you need to specify a learning rate parameter which will control how much the coefficients can change on each update.

coefficient = coefficient - (alpha * derivative value)

$$\beta_0 := \beta_0 - \alpha \cdot \frac{1}{n} \sum_{i=1}^n (y - (\beta_0 + \beta_1 X))$$

$$\beta_1 := \beta_1 - \alpha \cdot \frac{1}{n} \sum_{i=1}^n (y - (\beta_0 + \beta_1 X)) \cdot X$$

This particular process is repeated as long as the change in coefficients is 0.0 or close enough to zero.

Exploring β

22 September 2022

19:28

β_p

- If $\beta_p > 0$, then x (predictor) and y (target) have a positive relationship. That is an increase in x will increase y .
- If $\beta_p < 0$ then x (predictor) and y (target) have a negative relationship. That is an increase in x will decrease y

β_0

- The value of β_0 guarantees that the residual will have mean zero.
- If there is no β_0 term, then the regression will be forced to pass over the origin.
- Both the regression coefficient and prediction will be biased.

e.g. Sunshine vs ice-cream sales

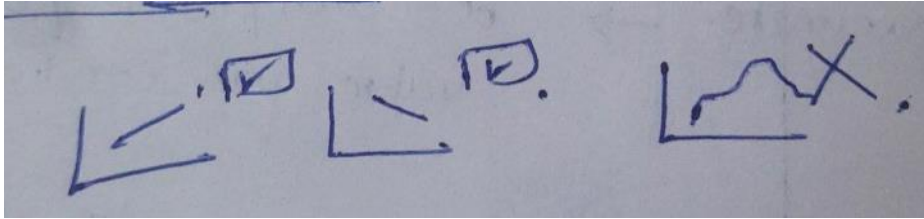
- As we seen the plot as linear regression model
- Increase in sunshine increase the sale of ice-cream
- But, it doesn't mean if there is no sunshine, no ice-cream will be sale.

Hence we using β_0 .

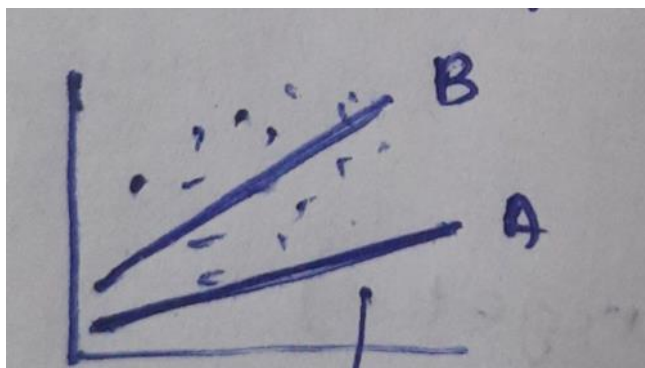
Assumption of regression line

22 September 2022 19:32

1. The relation between the dependent and independent variables should be almost linear.



2. Mean of residuals should be zero or close to 0 as much as possible. It is done to check whether our line is actually the line of "best fit".



3. There should be **homoscedasticity** or equal variance in a regression model. This assumption means that the variance around the regression line is the same for all values of the predictor variable (X).

4. There should **not be multicollinearity** in regression model. Multicollinearity generally occurs when there are high correlations between two or more independent variables.

eg:

☀ Sunshine (y)

correlate with

☹ Ice cream (x_1),

☔ Rain (x_2).

but x_1, x_2

correlated with each other

are not
Indp values.

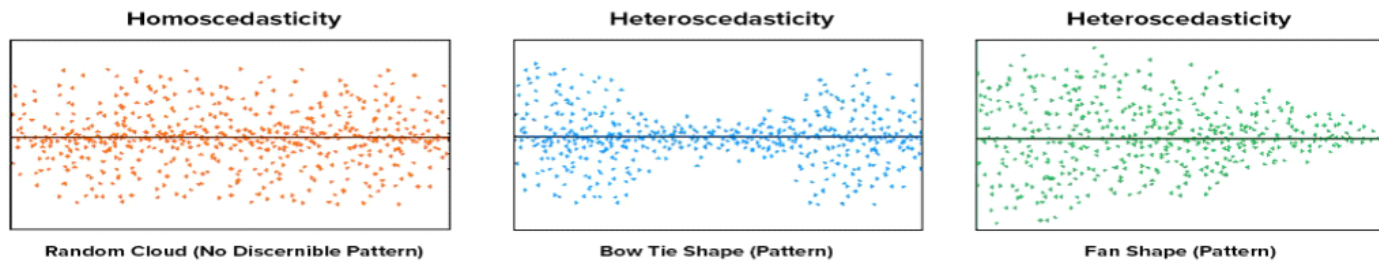
$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 \quad \left(\text{where } x_1, x_2, x_3 \text{ correlated to } y \checkmark \right)$$

If x_1 & x_2 & x_n & x_m co-related
multiple
then not Apply Linear Regression.

Homoscedasticity

22 September 2022 19:45

- Homoscedasticity (meaning “**same variance**”)
- Describes a situation in which the **error term is the same** across all values of the independent variables.
- **Heteroscedasticity** (the violation of homoscedasticity) is present when the size of the error term differs across values of an independent variable.
(Scatter plot : Residual vs Fitted value)



What if Heteroscedasticity?

- Heteroscedasticity **does not cause bias** in the **coefficient estimates**, it does make them less precise.
- Dealing with Heteroscedasticity:
 1. Log-transformation of features (Rescaling up the value)
 2. Outlier treatment
 3. Try polynomial fit

Multicollinearity

22 September 2022 19:52

- Multicollinearity occurs when independent variables in a regression model are correlated.
- This correlation is a problem because independent variables should be independent.
- If the degree of correlation between variables is high enough, it can cause problems when you fit the model and interpret the results

Detecting the Multicollinearity:

Variance inflation factor(VIF) detects multicollinearity:

$$VIF = \frac{1}{1 - R_i^2}$$

- VIF 1 = not correlated
- VIF between 1 and 5 = moderately correlated
- VIF greater than 5 = highly correlated

What if Multicollinearity?

e.g. value of VIF for

x1=1,

x2=2,

x3=5,

x4=7

x3 and x4 are correlated to each other.

and x3,x4 correlated to Y.

Note: How the calculation is done?

- While calculating the VIF of one independent variable (let it be x1)
- Then, x1 is consider as dependent variable
- Others x2,x3 and so on consider as independent variable
- This will be applicable to all x, when calculating VIF

Solution?

1. **Remove** some of the highly correlated independent variables.
(or)
2. Linearly combine the independent variables, such as **adding** them together.
(or)
3. Perform an **analysis** designed for highly correlated variables, such as principal components analysis.

Properties of Regression line

22 September 2022 20:26

1. Regression line passes through the mean of independent variable (x) as well as mean of the dependent variable (y).
2. β_1 explains the change in Y with a change in x by one unit.

In other words, if we increase the value of 'x' it will result in a change in value of Y.

3. The regression constant (β_0) is equal to the y intercept of the regression line.
4. The **line minimizes** the sum of squared differences between **observed values** (the y values) and **predicted values** (the \hat{y} values computed from the regression equation).

Note:

The least squares regression line is the only straight line that has all of these properties.

7

If a relevant variable is omitted from a regression equation, the consequences would be that:



Your Answer

The standard errors would be biased

If the excluded variable is uncorrelated with all of the included variables, all of the slope coefficients will be inconsistent

If the excluded variable is uncorrelated with all of the included variables, the intercept coefficient will be inconsistent

Explanation:

Correct! If a relevant variable is omitted from a regression equation, then the standard conditions for OLS optimality will not apply. These conditions implicitly assumed that the model was correctly specified in the sense that it includes all of the relevant variables. If relevant variables (that is, variables that are in fact important determinants of y) are excluded from the model, the standard errors could be biased (thus (i) is true), and, the slope coefficients will be inconsistently estimated unless the excluded variable is (are) uncorrelated with all of the included explanatory variable(s) - thus (ii) is wrong. If this condition holds, the slope estimates will be consistent, unbiased and efficient (so (iv) is wrong), but the intercept estimator will still be inconsistent (so (iii) is correct).

5. In a simple linear regression model (One independent variable), If we change the input variable by 1 unit. **How much output variable will change?**

- a) by 1
- b) no change
- c) by intercept
- d) by its slope

View Answer

Answer: d

Explanation: For linear regression $Y = a + bx + \text{error}$. If neglect error then $Y = a + bx$. If x increases by 1, then $Y = a + b(x+1)$ which implies $Y = a + bx + b$. So Y increases by its slope.

Advantages of Linear Regression

22 September 2022 20:30

1. Linear regression is simple to implement and easier to interpret the output coefficients
2. When you know the **relationship** between the independent and dependent variable is linear, this algorithm is the best to use because it's less complex as compared to other algorithms
3. It works well irrespective of **data size**

4

The main purpose(s) of Linear Regression is/are (choose all that apply):



Your Answer

Predicting one variable on the basis of another

Exploring the relationship between one variable and another

Correct Answer

Predicting one variable on the basis of another

Explaining one variable in terms of another

Limitations of Linear Regression

22 September 2022 20:31

1. **Outliers** can have huge effect on the regression line.
2. Linear regression **assumes** a linear **relationship** between dependent and independent variables, which is not the case in most of the real world problems.
3. **Prone to underfitting** - Linear regression sometimes fails to capture the underneath pattern in data properly due to simplicity of the algorithm.

Data Preparation for Linear Regression

22 September 2022 20:34

1. Linear Assumption:

- Linear regression assumes that the relationship between your independent and dependent is linear.
- It does not support anything else.
- This may be obvious, but it is good to remember when you have a lot of attributes.
- You may need to transform data to make the relationship linear (e.g. log transform for an exponential relationship).

2. Remove Outlier:

- Linear regression assumes that your independent and dependent variables are not noisy.
- Consider using data cleaning operations that let you better expose and clarify the signal in your data.
- This is most important for the output variable and you want to remove outliers in the output variable (y) if possible.

3. Remove Collinearity:

- Linear regression will over-fit your data when you have highly correlated input variables.
- Consider calculating pairwise correlations for your input data and removing the most correlated.

4. Gaussian Distributions:

- Linear regression will make more reliable predictions if your independent and dependent variables have a Gaussian distribution.
- You may get some benefit using transforms (e.g. log or BoxCox) on your variables to make their distribution more Gaussian looking.
- Note: Always random variable, not biased variable.

5. Rescale Inputs:

- Linear regression will often make more reliable predictions if you rescale input variables using **standardization or normalization**.
Note:

Rescale Input

eg: Indp variable | Sales (dependen variable)

| | |
|-----------------------------|-------------|
| $x_1 \rightarrow$ Trade | 1M - 10M |
| $x_2 \rightarrow$ Radio | 100K - 200K |
| $x_3 \rightarrow$ Newspaper | 50L - 200L |

Normalisation

~~Standardisation~~

(most cases)

$$\frac{x - x_{\min}}{x_{\max} - x_{\min}} \Rightarrow \text{value only be (0-1)}$$

Standardisation

$$x = \frac{x - \mu}{\sigma}$$

Regression model evaluation metrics

23 September 2022 06:57

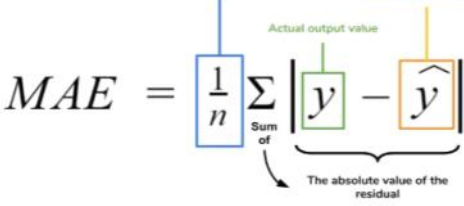
| | |
|----|--|
| Q: | How a model is built considered as a good model? |
| A: | By the difference(errors) in the values of the predicted and actual data is not much, and then used to make the future predictions |

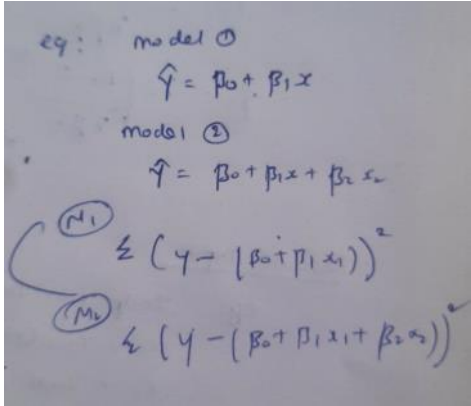
For this purpose of calculate errors in the model we have some metric tools,

1. MSE (Mean Squared Error)
2. RMSE (Root Mean Squared Error)
3. MAE (Mean Absolute Error)
4. MAPE (Mean Absolute Percentage Error)
5. R2 (R – Squared)
6. Adjusted R2

Note: These metrics can be used only to regression model where independent variables (x's) are only numerical

| | | |
|---------------------------------|--|--|
| Mean Squared Error: | <ul style="list-style-type: none"> • It is simply the average of the squared difference between the target value and the value predicted by the regression model. • MSE is optimized in <u>OLS(ordinary least square)</u> | $MSE = \frac{1}{n} \sum \underbrace{(y - \hat{y})^2}_{\text{The square of the difference between actual and predicted}}$ |
| Root Mean Squared Error (RMSE): | <ul style="list-style-type: none"> • Square root of MSE • Preferred in most cases • Because the errors are first squared before averaging which <u>poses a high penalty on large errors</u> | $RMSE = \sqrt{\frac{\sum_{i=1}^N (\text{Predicted}_i - \text{Actual}_i)^2}{N}}$ |
| Mean | <ul style="list-style-type: none"> • absolute | <div>Divide by the total number of data points</div> <div>Predicted output value</div> |

| | | |
|---|--|---|
| <p>Absolute Error(MAE) :</p> | <p>difference between the target value and the value predicted by the model.</p> <ul style="list-style-type: none"> • MAE <u>not</u> applicable when we pay more attention to <u>outliers</u> • MAE is a linear score which means all the individual differences are weighted equally. |  $MAE = \frac{1}{n} \sum y - \hat{y} $ |
| <p>Mean Absolute Percentage error /deviation (MAPE/MAPD) :</p> | <ul style="list-style-type: none"> • <u>measure of prediction accuracy</u> of a forecasting method in statistics <p>For example: In trend estimation, also used as a loss function for regression problems in machine learning.</p> | $MAPE = \frac{100\%}{N} \sum_{i=1}^N \left \frac{y_i - \hat{y}_i}{y_i} \right $ |
| <p>R²- R squared or Coefficient of Determination</p> | <ul style="list-style-type: none"> • The metric helps us to compare our <u>current</u> model with a constant <u>baseline</u> and tells us <u>how much our model is better.</u> • The constant baseline is chosen by taking the mean of the data and drawing | $R^2 = 1 - \frac{MSE(\text{model})}{MSE(\text{baseline})}$ <p>Properties of R²:</p> <ul style="list-style-type: none"> • R² ranges between 0* to 1. • R² of 0 means that there is no correlation between the dependent and the independent variable. • R² of 1 means the dependent variable can be predicted from the independent variable without any error. |

| | | |
|----------------|---|--|
| | <p>a line at the mean. Notably,</p> <ul style="list-style-type: none"> • R^2 is scale free • Values always ≤ 1 • When Model have <u>worse</u> than the baseline, Negative value of R^2 come. | <ul style="list-style-type: none"> • An R^2 of 0.20 means that 20% variance in Y is predictable from X; an R^2 of 0.40 means that 40% variance is predictable. <p>Note : R^2 score may range from $-\infty$ to 1 if OLS is not used to get the predictions.</p> |
| Adjusted R^2 | <p>Problem with R^2,</p> <ul style="list-style-type: none"> • R^2 suffers from the problem that <u>the scores improve on increasing terms</u> even though the model is not improving which may misguide the researcher. <p>Then what Adjustment it done?</p> <ul style="list-style-type: none"> • Adjusted R^2 is <u>always lower than R^2</u> as it adjusts for the increasing predictors and <u>only shows improvement if there is a real improvement.</u> | $R_a^2 = 1 - \left[\left(\frac{n-1}{n-k-1} \right) \times (1 - R^2) \right]$ <p>where: n = number of observations k = number of independent variables R_a^2 = adjusted R^2</p>  <p>In the above example, Model 1 will always produce higher R^2 value due to it has less independent variables than Model 2</p> |

Note:

Apart from the above Evaluation metrics "**F-statistics**" also used for linear regression.

F-statistics:

f-statistics is a statistic used to test the significance of regression coefficients in linear regression models. f-statistics can be calculated as MSR/MSE where MSR represents the mean sum of squares regression and MSE represents the mean sum of squares error.

Note

23 September 2022 20:41

Difference between Standard scaler and MinMaxScaler

| Standard scaler | Normalization or MinMaxScaler |
|--|---|
| rescales a dataset to have a, <ul style="list-style-type: none">• mean of 0 and• a standard deviation of 1 | rescale a dataset so that each value fall between 0 and 1. (i.e. each value will be normalised by subtracting the mean and dividing by standard deviation.) |
| In both the cases useful when data has varying scales | "" |
| algorithm assumption about data having a gaussian distribution . | algorithm does not make assumptions about the distribution (good to use when we don't know about the distribution) |
| If there is Outliers , RobustScaler() . Alternatively you could remove the outliers and use either of the above 2 scalers (depends on distributions) | "" |

Note: If **scaler** is used before train_test_split, data leakage will happen. Do use scaler **after** train_test_split

4

We don't need feature selection if we have single digit columns.



Your Answer

False

Correct Answer

False

What is **Feature Selection**?

Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data.

It is the process of automatically choosing relevant features for your

machine learning model based on the type of problem you are trying to solve. We do this by including or excluding important features without changing them. It helps in cutting down the noise in our data and reducing the size of our input data.



Figure 3: Feature Selection

```
plt.scatter(df["Marketing Spend"], df["Profit"], alpha = 0.5)
```

What is **alpha in matplotlib**?

- Matplotlib allows you to regulate the transparency of a graph plot using the alpha attribute.
- By default, alpha=1. If you would like to form the graph plot more transparent, then you'll make alpha but 1, such as 0.5 or 0.25.

Play with Dataset

29 September 2022 13:27



The Boston Housing Dataset is derived from information collected by the U.S. Census Service concerning housing in the area of Boston MA. The following describes the dataset columns:

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 - $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town
LSTAT - % lower status of the population
MEDV - Median value of owner-occupied homes in \$1000's

Dependent Variables: "**MEDV**"

Step 01: "Importing Libraries"

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from scipy import stats
```

%matplotlib inline

Line-oriented Magic function

- Start with %

Cell-oriented Magic function

- Start with %%

What it does?

- Setup matplotlib to work effectively
- By activating the matplotlib interactive support
- "**inline**" enable to plot the graph just below the commands are written.
- It also ensures not to affect the previous plot when the code written after the output of the previous

There are few other magic functions available as listed below,

```
%matplotlib --list
```

Available matplotlib backends: ['tk', 'gtk', 'gtk3', 'wx', 'qt4', 'qt5', 'qt', 'osx', 'nbagg', 'notebook', 'agg', 'svg', 'pdf', 'ps', 'inline', 'ipympl', 'widget']

Step 02: "Importing drive"

```
[2] from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

```
[3] working_dir='/content/drive/MyDrive/Almabetter/Module 04 ML/housing.csv'
df=pd.read_csv(working_dir)
```

Step 03: "Understanding the dataset"

```
df.columns
```

```
Index([' 0.00632' 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00'], dtype='object')
```

```
df.head()
```

```
 0.00632 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00
0      0.02731 0.00 7.070 0 0.4690 6.4210 78...
1      0.02729 0.00 7.070 0 0.4690 7.1850 61...
2      0.03237 0.00 2.180 0 0.4580 6.9980 45...
3      0.06905 0.00 2.180 0 0.4580 7.1470 54...
4      0.02985 0.00 2.180 0 0.4580 6.4300 58...
```

We having only one column/ index

505 Rows with each row has the value in the form of string

```
df.describe()
```

```
 0.00632 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00
count                                          505
unique                                          505
top      51.13580 0.00 18.100 0 0.5970 5.7570 100...
freq                                          1
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 505 entries, 0 to 504
Data columns (total 1 columns):
 #   Column
----  -----
 0   0.00632 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00
dtypes: object(1)
memory usage: 4.1+ KB
```

| | Non-Null | Count | Dtype |
|--|----------|----------|--------|
| | 505 | non-null | object |

```
df.keys()
```

```
Index([' 0.00632' 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00'], dtype='object')
```

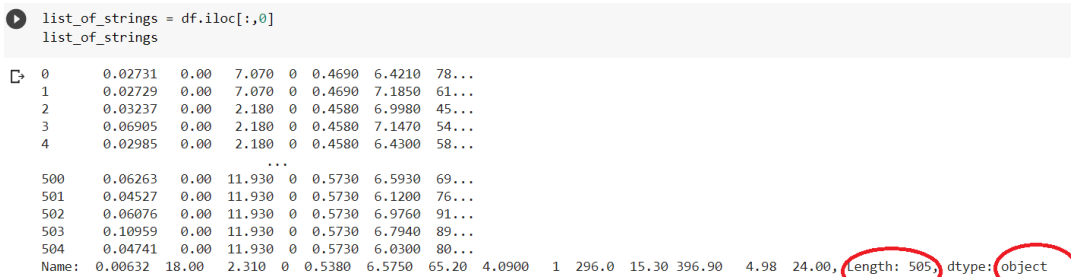
| | |
|----------------|--|
| DataFrame.keys | <ul style="list-style-type: none">• Pandas dataframe.keys() function returns the 'info axis' for the pandas object.• If the pandas object is series then it returns index.• If the pandas object is dataframe then it returns columns.• If the pandas object is panel then it returns major_axis. |
|----------------|--|

- For Supervised linear regression, we need a clean dataframe with rows and columns labelled.
- We have all the Row **values** in form of **list of strings** which we need to convert to **list of float** to make a new data set.

Step 04: "Data Cleaning"

- Creating the list of strings of the row values provided.

```
list_of_strings = df.iloc[:,0]
list_of_strings
```



0 0.02731 0.00 7.070 0 0.4690 6.4210 78...
 1 0.02729 0.00 7.070 0 0.4690 7.1850 61...
 2 0.03237 0.00 2.180 0 0.4580 6.9980 45...
 3 0.06905 0.00 2.180 0 0.4580 7.1470 54...
 4 0.02985 0.00 2.180 0 0.4580 6.4300 58...
 ...
 500 0.06263 0.00 11.930 0 0.5730 6.5930 69...
 501 0.04527 0.00 11.930 0 0.5730 6.1200 76...
 502 0.06076 0.00 11.930 0 0.5730 6.9760 91...
 503 0.10959 0.00 11.930 0 0.5730 6.7940 89...
 504 0.04741 0.00 11.930 0 0.5730 6.0300 80...
 Name: 0.00632 18.00 2.310 0 0.5380 6.5750 65.20 4.0900 1 296.0 15.30 396.90 4.98 24.00, Length: 505, dtype: object

We have 505 list with
 each list had one string
 or object init

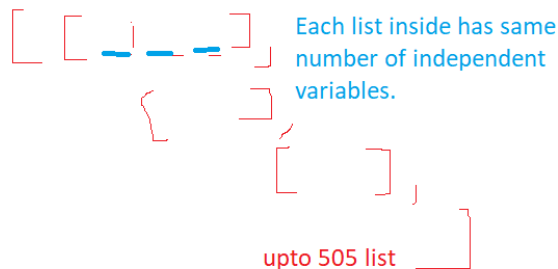
- This List of strings have to be converted into the list of floats
- Hence we defining a 2 functions,
 1. 1st to convert the list of string into the list of floats
 2. 2nd to append the new list of floats as nested list

```
def convert_to_list_of_float(k):
    s=[]
    for i in k:
        n=float(i)
        s.append(n)
    return (s)
```

```
# Nested list of Floats
a=[]
for i in range(len(list_of_strings)):
    a.append(convert_to_list_of_float(list_of_strings[i].split()))
print(a)
```

[[0.02731, 0.0, 7.07, 0.0, 0.469, 6.421, 78.9, 4.9671, 2.0, 242.0, 17.8, 396.9, 9.14, 21.6], [0.02729, 0.0, 7.07, 0.0, 0.469, 7.185, 61.1, 4.9671,

- Now we have the data "a" as nested list of floats as,



- Now, New dataframe will be created using the pandas.

```
new_df = pd.DataFrame(data= a, columns=['CRIM','ZN','INDUS','CHAS','NOX','RM','AGE','DIS','RAD','TAX','PTRATIO','B','LSTAT','MEDV'])
```

new_df

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | PTRATIO | B | LSTAT | MEDV |
|-----|---------|-----|-------|------|-------|-------|------|--------|-----|-------|---------|--------|-------|------|
| 0 | 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 | 21.6 |
| 1 | 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 | 34.7 |
| 2 | 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 | 33.4 |
| 3 | 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 | 36.2 |
| 4 | 0.02985 | 0.0 | 2.18 | 0.0 | 0.458 | 6.430 | 58.7 | 6.0622 | 3.0 | 222.0 | 18.7 | 394.12 | 5.21 | 28.7 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 500 | 0.06263 | 0.0 | 11.93 | 0.0 | 0.573 | 6.593 | 69.1 | 2.4786 | 1.0 | 273.0 | 21.0 | 391.99 | 9.67 | 22.4 |
| 501 | 0.04527 | 0.0 | 11.93 | 0.0 | 0.573 | 6.120 | 76.7 | 2.2875 | 1.0 | 273.0 | 21.0 | 396.90 | 9.08 | 20.6 |
| 502 | 0.06076 | 0.0 | 11.93 | 0.0 | 0.573 | 6.976 | 91.0 | 2.1675 | 1.0 | 273.0 | 21.0 | 396.90 | 5.64 | 23.9 |
| 503 | 0.10959 | 0.0 | 11.93 | 0.0 | 0.573 | 6.794 | 89.3 | 2.3889 | 1.0 | 273.0 | 21.0 | 393.45 | 6.48 | 22.0 |
| 504 | 0.04741 | 0.0 | 11.93 | 0.0 | 0.573 | 6.030 | 80.8 | 2.5050 | 1.0 | 273.0 | 21.0 | 396.90 | 7.88 | 11.9 |

505 rows × 14 columns

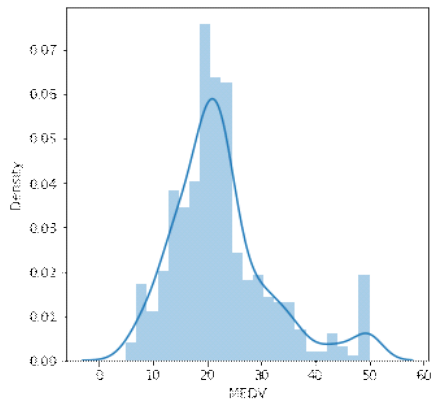
- The newly created dataset has "zero null values", and all the columns with the datatype "Float"

Step 04: "Understanding the distribution of each variables"

- A **Distplot** or distribution plot, depicts the variation in the data distribution.
- Using seaborn library we visualize the distribution of each variable data's.

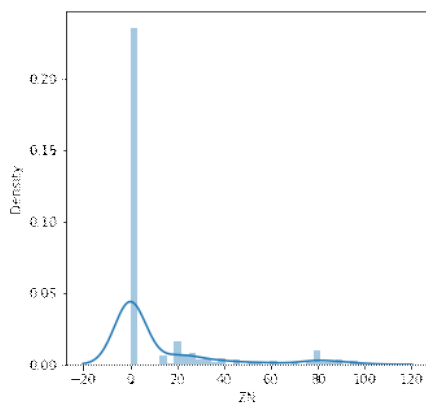
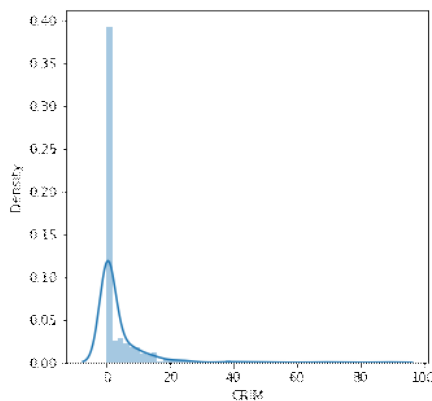
```
for i in new_df.columns:  
    plt.figure(figsize=(5,5))  
    sns.distplot(new_df[i])
```

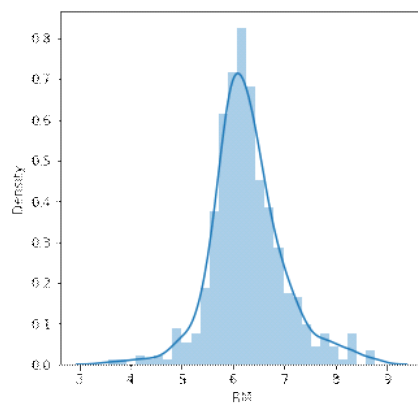
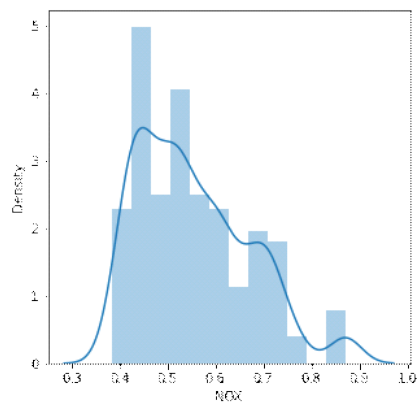
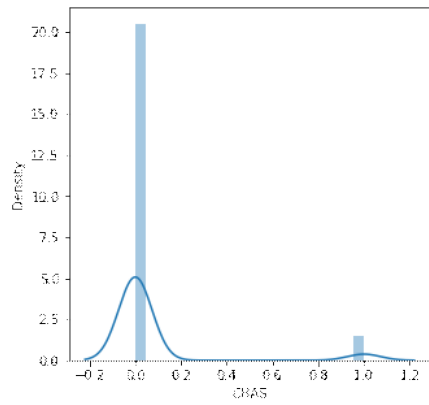
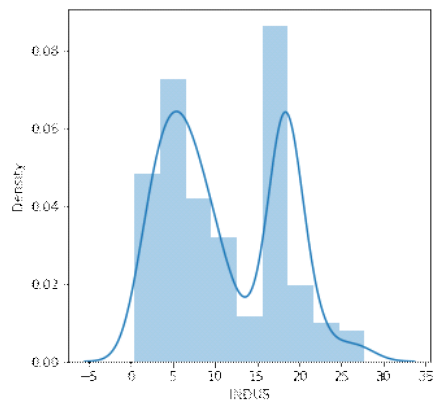
For Dependent variable "MEDV"

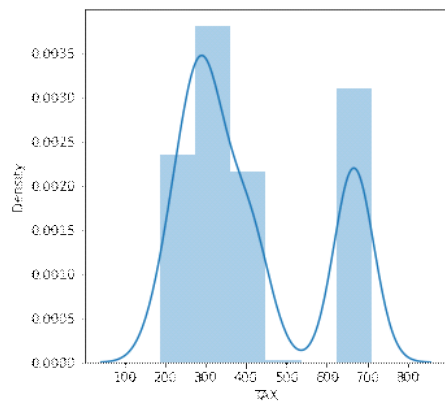
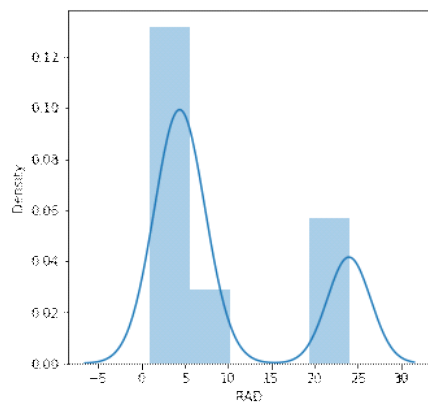
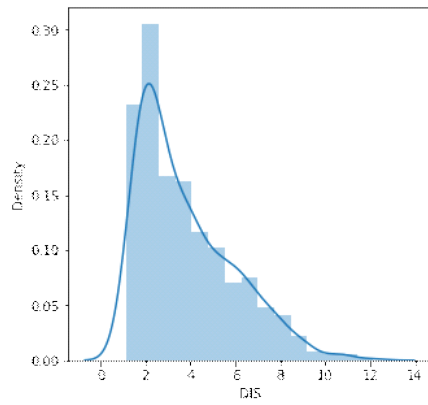
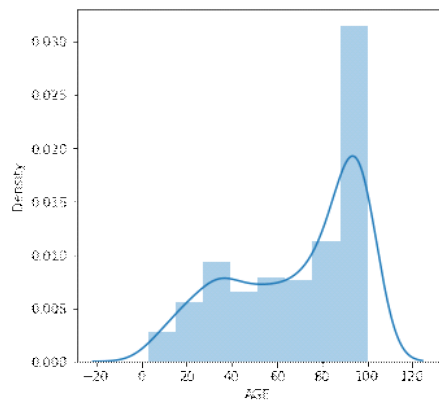


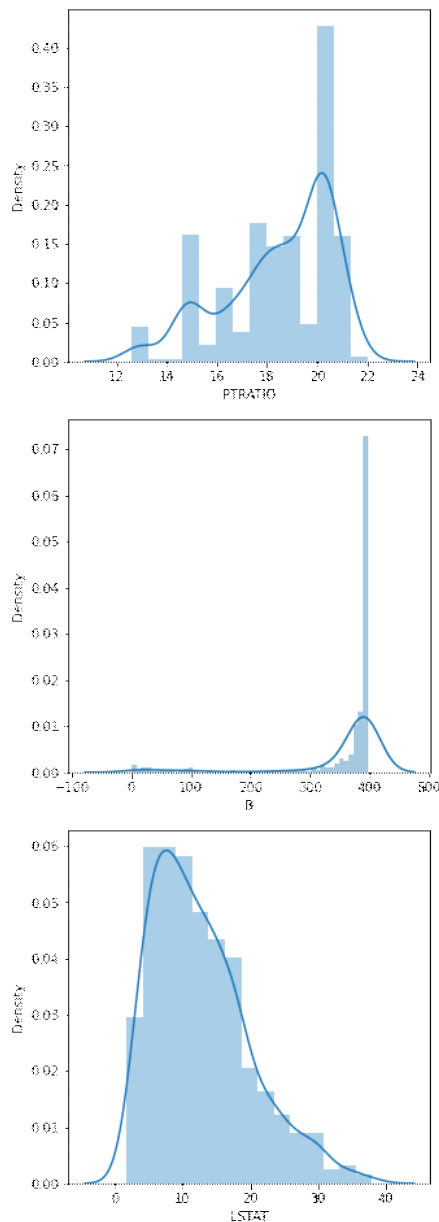
- It clearly states that data from the MEDV column are normal distributed with some outliers to the left.

For Independent variables,









Step 05: "Finding the Correlation between the dataset"

new_df.corr()

| | CRIM | ZN | INDUS | CHAS | NOX | RM | AGE | DIS | RAD | TAX | PTRATIO | B | LSTAT | MEDV |
|---------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| CRIM | 1.000000 | -0.200283 | 0.406251 | -0.056132 | 0.420934 | -0.218978 | 0.352701 | -0.379627 | 0.625396 | 0.582568 | 0.289394 | -0.384839 | 0.455329 | -0.388249 |
| ZN | -0.200283 | 1.000000 | -0.534022 | -0.042550 | -0.516574 | 0.311835 | -0.569524 | 0.664395 | -0.311717 | -0.314351 | -0.391713 | 0.175319 | -0.412894 | 0.360393 |
| INDUS | 0.406251 | -0.534022 | 1.000000 | 0.062350 | 0.764556 | -0.391330 | 0.645543 | -0.708848 | 0.594167 | 0.720561 | 0.380955 | -0.356506 | 0.602737 | -0.484126 |
| CHAS | -0.056132 | -0.042550 | 0.062350 | 1.000000 | 0.091134 | 0.091497 | 0.086461 | -0.099109 | -0.007907 | -0.035965 | -0.122570 | 0.049040 | -0.054576 | 0.175364 |
| NOX | 0.420934 | -0.516574 | 0.764556 | 0.091134 | 1.000000 | -0.302127 | 0.731461 | -0.769221 | 0.611758 | 0.668141 | 0.188918 | -0.380006 | 0.591262 | -0.427295 |
| RM | -0.218978 | 0.311835 | -0.391330 | 0.091497 | -0.302127 | 1.000000 | -0.240211 | 0.205170 | -0.209277 | -0.291680 | -0.355116 | 0.127754 | -0.613734 | 0.695365 |
| AGE | 0.352701 | -0.569524 | 0.645543 | 0.086461 | 0.731461 | -0.240211 | 1.000000 | -0.747872 | 0.456232 | 0.506527 | 0.261724 | -0.273486 | 0.602782 | -0.376932 |
| DIS | -0.379627 | 0.664395 | -0.708848 | -0.099109 | -0.769221 | 0.205170 | -0.747872 | 1.000000 | -0.494798 | -0.534492 | -0.232561 | 0.291451 | -0.497277 | 0.249896 |
| RAD | 0.625396 | -0.311717 | 0.594167 | -0.007907 | 0.611758 | -0.209277 | 0.456232 | -0.494798 | 1.000000 | 0.910202 | 0.463322 | -0.444065 | 0.487608 | -0.381690 |
| TAX | 0.582568 | -0.314351 | 0.720561 | -0.035965 | 0.668141 | -0.291680 | 0.506527 | -0.534492 | 0.910202 | 1.000000 | 0.460100 | -0.441505 | 0.543435 | -0.468543 |
| PTRATIO | 0.289394 | -0.391713 | 0.380955 | -0.122570 | 0.188918 | -0.355116 | 0.261724 | -0.232561 | 0.463322 | 0.460100 | 1.000000 | -0.176515 | 0.372148 | -0.508411 |
| B | -0.384839 | 0.175319 | -0.356506 | 0.049040 | -0.380006 | 0.127754 | -0.273486 | 0.291451 | -0.444065 | -0.441505 | -0.176515 | 1.000000 | -0.365637 | 0.333394 |
| LSTAT | 0.455329 | -0.412894 | 0.602737 | -0.054576 | 0.591262 | -0.613734 | 0.602782 | -0.497277 | 0.487608 | 0.543435 | 0.372148 | -0.365637 | 1.000000 | -0.738187 |
| MEDV | -0.388249 | 0.360393 | -0.484126 | 0.175364 | -0.427295 | 0.695365 | -0.376932 | 0.249896 | -0.381690 | -0.468543 | -0.508411 | 0.333394 | -0.738187 | 1.000000 |

For Better understanding, we visualizing the whole correlation using seaborn-heatmap,

```
plt.figure(figsize=(12,10))
sns.heatmap(new_df.corr(),annot= True, cmap='Spectral_r')
```




Observations from the correlations:

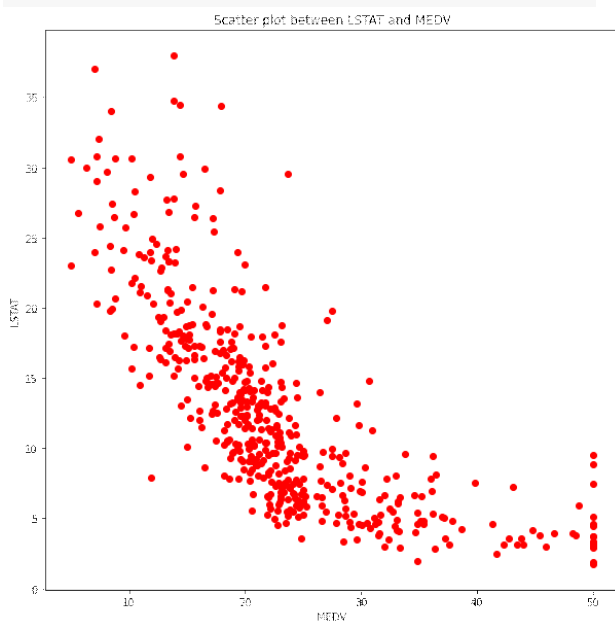
- It is better to avoid the highly positive/ negatively correlated independent variables such as INDUS-ZN, AGE-DIS, RAD-TAX
- We also observe that,
 - RM - Good correlation with Dependent variable
 - LSTAT - Highly correlated with the Dependent variable

From here, we can make some analysis using scatter plot before making any conclusion.

Step 06: "Scatter Plot" for Dependent variable vs selected Independent variable

scatter plot between LSTAT and MEDV.

```
plt.figure(figsize=(10,10))
x = new_df['MEDV']
y = new_df['LSTAT']
plt.scatter(x, y,color='red')
plt.title('Scatter plot between LSTAT and MEDV')
plt.xlabel('MEDV')
plt.ylabel('LSTAT')
```

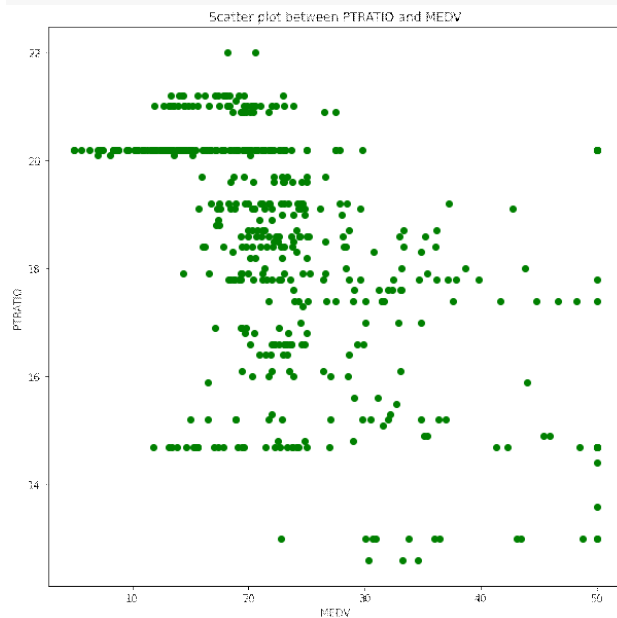


Observation:

- Suggest that, Relationship is Negative linear regression
- As states that, as "Percentage of lower status population" decreases, it increases the MEDV "no. of owner-occupied homes in price range of \$1000 would"
- It also proves why the correlation between LSTAT vs MEDV has highly negative.

- It is noted that, even though LSTAT is high negative correlation, it is linear relationship with the dependent variable.
- all variables doesn't exhibit the same the properties
- Let we check with the independent variable "PTRATIO" which has second highest negative correlation with the Dependent variable "MEDV"

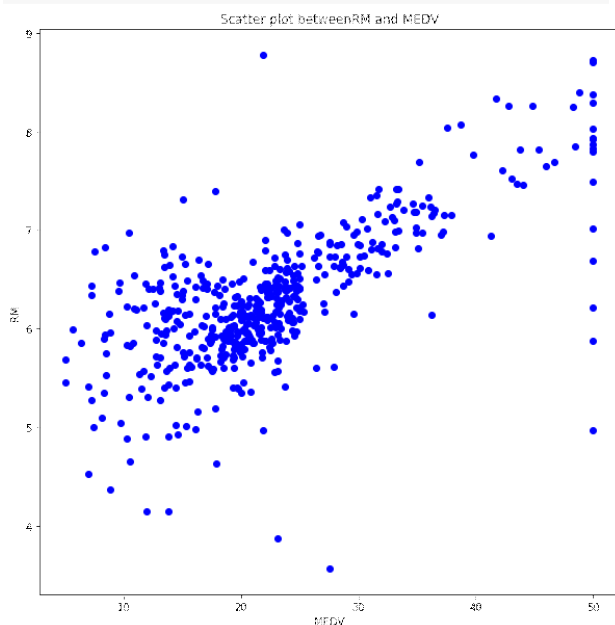
```
plt.figure(figsize=(10,10))
x = new_df['MEDV']
y = new_df['PTRATIO']
plt.scatter(x, y,color='green')
plt.title('Scatter plot between PTRATIO and MEDV')
plt.xlabel('MEDV')
plt.ylabel('PTRATIO')
```



- It is observed that, Relationship between PTRATIO and MEDV are not linear.

plot scatter plot between RM and MEDV.

```
plt.figure(figsize=(10,10))
x = new_df['MEDV']
y = new_df['RM']
plt.scatter(x, y,color='blue')
plt.title('Scatter plot betweenRM and MEDV')
plt.xlabel('MEDV')
plt.ylabel('RM')
```



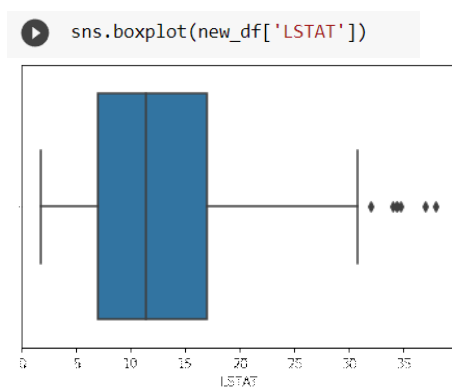
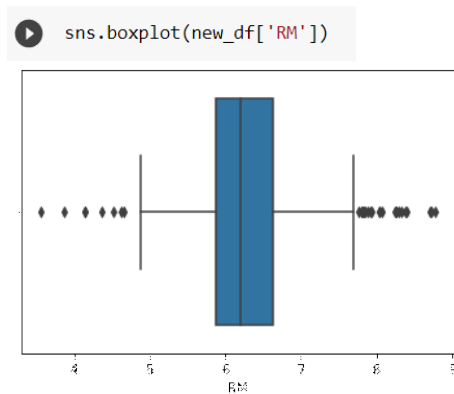
Observation:

- Positive linear regression

- It also proves why we found high positive correlation between the same

Hence, LSTAT and RM will be consider to make predictions after checking for outliers.

Step 07: "Checking for the outliers"



Conclusion from the observation,
LSTAT and RM do not consist a lot of outliers too therefore I would say we can easily consider these columns to make any predictions.

Step 08: "Conclusion about best possible independent variables"

To conclude, **LSTAT** and **RM** are the best possible independent variables to make any future predictions about our dependent variable MEDV.

Step 09: "Calculating the best fit for RM and MEDV"

Step 09-01: Creation of the dataframe with taken dependent and independent variable,

```
rmdf={'y':new_df['MEDV'],'x':new_df['RM']}
rmdf=pd.DataFrame(rmdf)
rmdf['x2']= rmdf['x']**2
rmdf['xy']= rmdf['x']*rmdf['y']
```

| | y | x | x2 | xy |
|---|------|-------|-----------|----------|
| 0 | 21.6 | 6.421 | 41.229241 | 138.6936 |
| 1 | 34.7 | 7.185 | 51.624225 | 249.3195 |
| 2 | 33.4 | 6.998 | 48.972004 | 233.7332 |
| 3 | 36.2 | 7.147 | 51.079609 | 258.7214 |
| 4 | 28.7 | 6.430 | 41.344900 | 184.5410 |

Step 09-02: Calculate the Population co-efficient Bo and B1

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\beta_1 = \frac{n \sum_i x_i y_i - \sum_i x_i \sum_i y_i}{n \sum_i x_i^2 - \left(\sum_i x_i \right)^2}$$

```
[ ] n= len(rmdf['y'])
    xys=rmdf['xy'].sum()
    xs=rmdf['x'].sum()
    ys=rmdf['y'].sum()
    s=rmdf['x']**2
    x2s=s.sum()
    b1= ((n*xys)-(xs*ys))/((n*x2s)-(xs))
    print(b1)
```

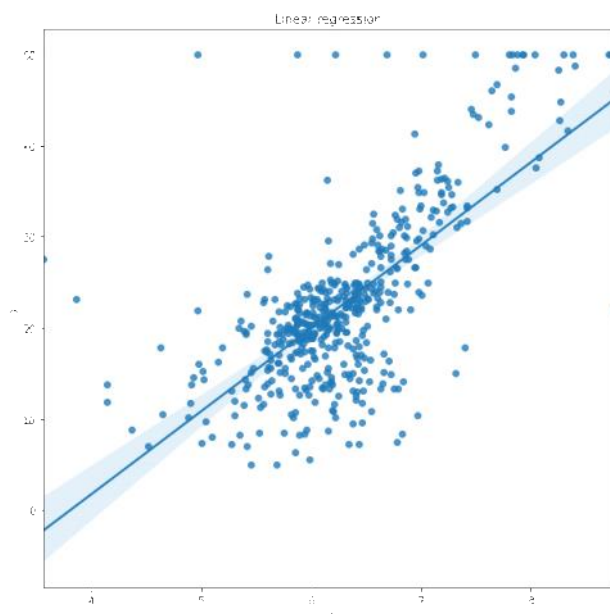
0.11239801390640114

```
[ ] ybar= ys/n
    xbar= xs/n
    b0=ybar-(b1*xbar)
    print(b0)
```

21.823585193601446

```
[ ] plt.figure(figsize=(10,10))
    plt.title('Linear regression.')
    plt.xlabel('RM')
    plt.ylabel('MEDV')
    sns.regplot(x=rmdf['x'],y=rmdf['y'])
```

"seaborn.regplot" Plot data and a linear regression model fit.



Step 10: "Calculating the best fit for LSTAT and MEDV"

```
[ ] ldf={'y':new_df['MEDV'],'x':new_df['LSTAT']}
ldf=pd.DataFrame(ldf)
ldf['x2']= ldf['x']**2
ldf['xy']= ldf['x']*ldf['y']
```

```
[ ] n= len(ldf['y'])
xys=ldf['xy'].sum()
xs=ldf['x'].sum()
ys=ldf['y'].sum()
s=ldf['x']**2
x2s=s.sum()
b1= ((n*xys)-(xs*ys))/((n*x2s)-(xs))
print(b1)
```

-0.22913670172915626

```
ybar= ys/n
xbar= xs/n
b0=ybar-(b1*xbar)
print(b0)
```

25.432663713289553

```
plt.figure(figsize=(10,10))
plt.title('Linear regression.')
plt.xlabel('LSTAT')
plt.ylabel('MEDV')
sns.regplot(x=ldf['x'],y=ldf['y'])
```

