Linear Regression

Module 1 : Introduction to Simple Linear Regression

Introduction:-

Welcome to the module on 'Linear Regression'.

In this module, you will start off with a quick introduction to three machine learning techniques: regression, classification and clustering. Then, you will go deeper into one of the most important regression models in machine learning: **linear regression**.

You will primarily learn about the following two types of linear regression models:

- Simple linear regression
- Multiple linear regression

In this session

You will get a quick introduction to machine learning models. You will learn about the basics of simple linear regression and understand how to find the best-fitted line for a model and all the various parameters associated with it.

This session covers the following topics:

- Introduction to machine learning
- Supervised and unsupervised learning methods
- The linear regression model
- Residuals
- Residual sum of squares (RSS) and R^2 (R-squared)

Introduction to Machine Learning:-

Till now you have learnt to explore and get insights out of data. Now we will be having a look at measuring relationships between variable through modeling.

Modeling uses machine learning algorithms, in which the machine learns from the data just like humans learn from their experiences.
Let's go deeper into the types of models that come under machine learning.

Machine learning models can be classified into the following three types based on the task performed and the nature of the output:

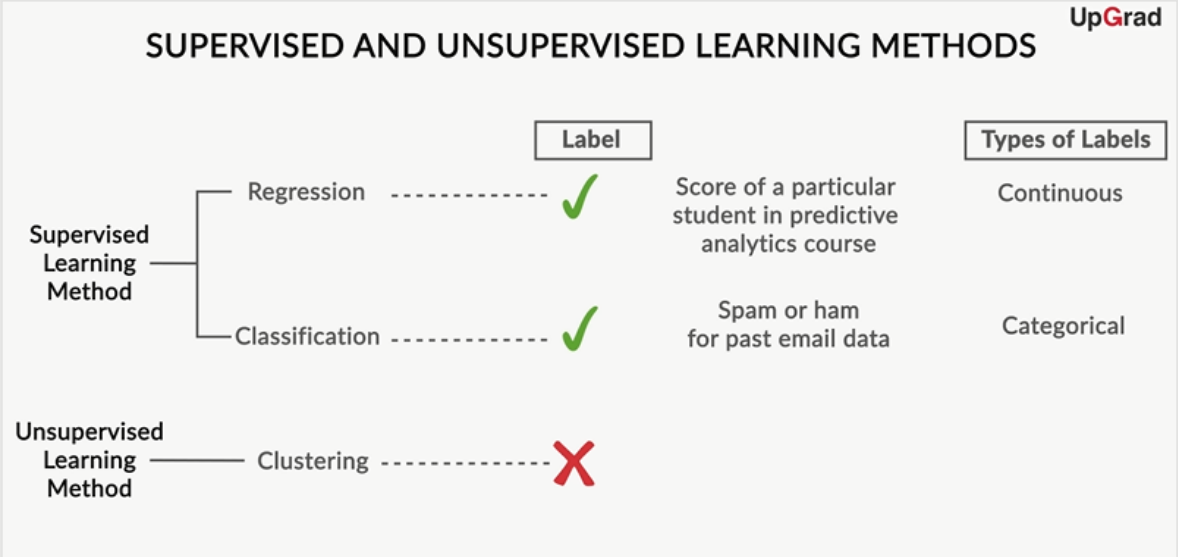
1. **Regression**: The output variable to be predicted is a **continuous**

variable, e.g., the score of a student on a subject.
2. **Classification**: The output variable to be predicted is a **categorical variable**, e.g., classifying incoming emails as spam or ham.
3. **Clustering**: No predefined notion of a label is allocated to the groups/ clusters formed, e.g., customer segmentation.

You will get to know about the different types of machine learning models in the coming modules.
Let's now learn about supervised and unsupervised learning methods.

So, you can classify machine learning models into two broad categories:

1. **Supervised learning methods**
    1. Past data with labels is used for building the model.
    2. **Regression** and **classification** algorithms fall under this category.
2. **Unsupervised learning methods**
    1. No predefined labels are assigned to past data.
    2. **Clustering** algorithms fall under this category.



Mentioned below are some of the problems that are addressed by either a supervised or an unsupervised learning algorithm. For each problem, identify if the task can be addressed by a supervised learning algorithm or an unsupervised learning algorithm. Assume that an appropriate data set is available for your algorithm to learn from.

In the next segment, you will learn about linear regression in a bit more detail. You will be introduced to the concept of a regression line.

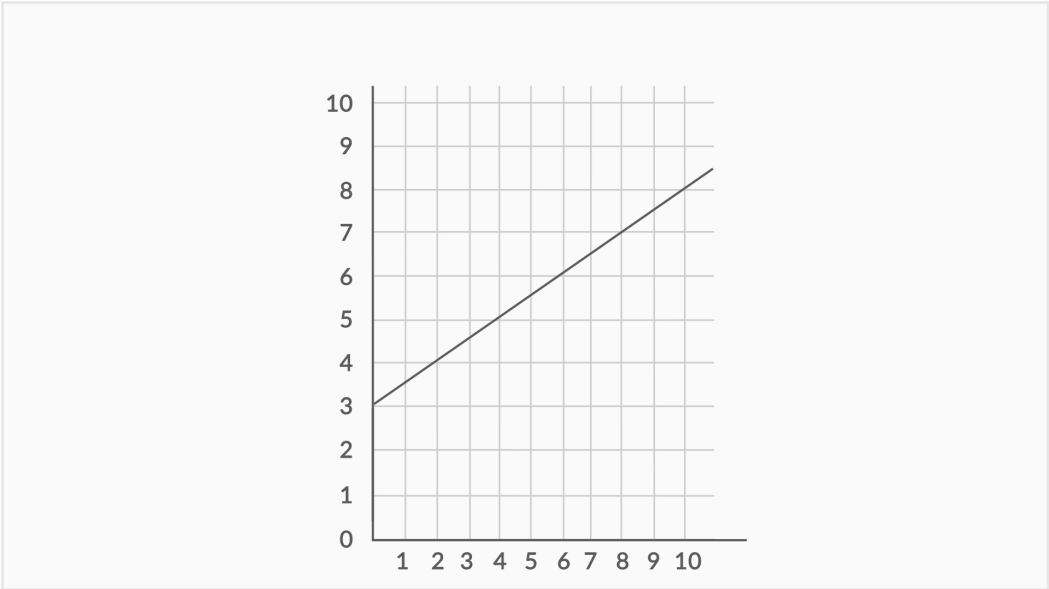
Regression Line:-

Let’s now delve deeper into linear regression. You will now learn to identify whether a given problem is a regression or a classification problem. Imagine that you need to predict the final score of a team before the end of its innings in a cricket match. You can build a regression model to be able to make a decent prediction.

Let's hear Prof. Dinesh talk about this and some other scenarios in a bit more detail in the next video.

Refer to this link to revise the physical significance of an equation of a straight line and also to understand how to find the slope and intercept of a straight line from its graph.

Answer the following questions in the context of the **straight-line plot** given below.



Since you now know what the equation of a straight line is, let’s look at the general equation of a straight line, which is fitted during **simple linear regression**.

A simple linear regression model attempts to explain the relationship between a dependent variable and an independent one using a straight line.

The independent variable is also known as the **predictor variable**, and the dependent variables are also known as the **output variables**.

Best Fit Line:-

In regression, a best fit line is a line that fits the given scatter plot in the best

way. Let's see how you can define the notion of a best fit line.

Let's reiterate what you have learnt so far:

1. You started with a scatter plot to check the relationship between sales and the marketing budget.
2. You found residuals and the RSS for any given line passing through the scatter plot.
3. Then you found the equation of the best fit line by minimising the RSS and also found the optimal values of β_0 and β_1.

So, you know that the best fit line is obtained by minimising a quantity called the residual sum of squares (RSS). You will now be introduced to the concept of a **cost function**.

Gradient descent
Gradient descent is an optimisation algorithm that optimises the objective function (cost function for linear regression) to reach the optimal solution.

You can learn about cost function and ways to optimise it (minimisation or maximisation) by referring to this link; the topic is covered in detail as part of the optional session. Even though the session is optional, we strongly recommend that you go through it, as gradient descent will also be used for logistic regression and even neural networks.

Let's now see a demonstration of obtaining a best fit line and finding out the RSS for the marketing spend vs sales example in Excel.

You can download the Excel file from below and look at the analysis carried out by the professor.

Strength of Simple Linear Regression:-

After determining the best fit line, there are a few critical questions that you need to answer, such as:

1. How well does the best fit line represent the scatter plot?
2. How well does the best fit line predict the new data?

Are the questions above answered well by the RSS? Attempt these two questions on your own before getting the answers from the professor in the following lecture.

You can play with the interactive graphic below and look at how the position of the regression line and the values of the RSS and the TSS change with a change in the values of β_0 and β_1.

Let's go back to our Excel demonstration and see how you can find out the TSS and then the R^2 for the same example for which we had performed regression.

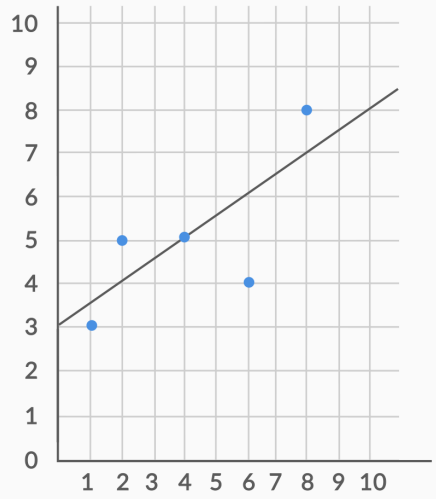
You can download the Excel sheet used in the demonstration for your reference.

Comprehension

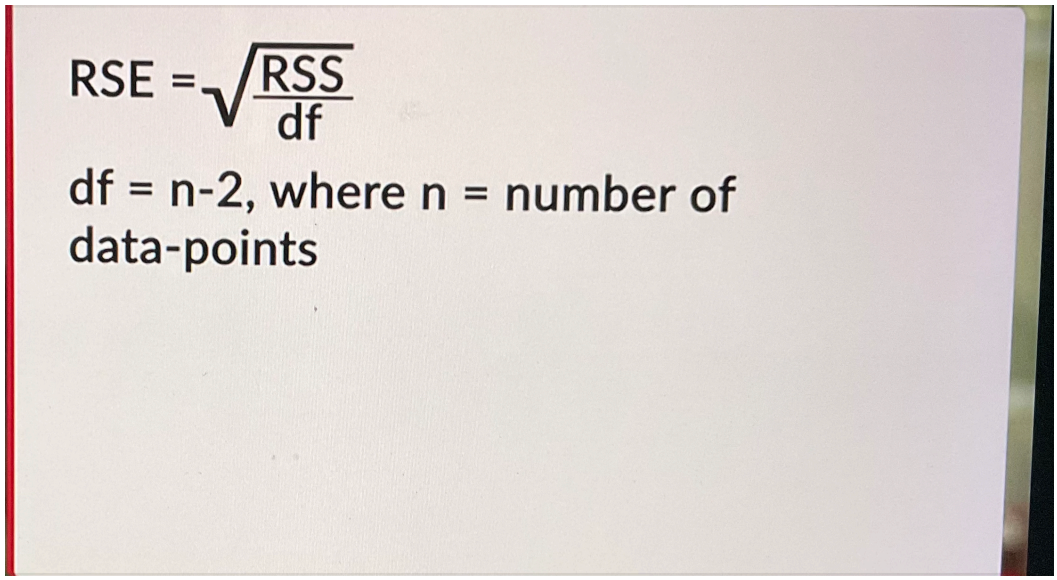
The plot below represents a scatter plot of two variables X and Y, with the Y variable being dependent on X. Let's assume that the line with the equation Y = X/2 + 3 plotted in the graph represents the best fit line. This is the same line that you found the equation of earlier.

You can find the value of the residual for each point, e.g., for x = 2, the residual would be 5 - 4 = 1.

Answer the following questions in order to consolidate your learning about RSS and R^2.



Apart from R^2, there is one more concept named RSE (residual squared error), which is linked to RSS. Let's see what that is.



Summary:-

Here's a brief summary of what you learnt in this session:

1. Machine learning models can be classified into the following two categories on the basis of the learning algorithm:
    - **Supervised learning method:** Past data with labels is available to build the model.
        - **Regression:** The output variable is continuous in nature.
        - **Classification:** The output variable is categorical in nature.
    - **Unsupervised learning method:** Past data with labels is not available.
        - **Clustering:** There is no predefined notion of labels.
2. Past dataset is divided into two parts in the supervised learning method:
    - **Training data** is used for the model to learn during modelling.
    - **Testing data** is used by the trained model for prediction and model evaluation.
3. Linear regression models can be classified into two types depending upon the number of independent variables:
    - **Simple linear regression:** This is used when the number of independent variables is 1.
    - **Multiple linear regression:** This is used when the number of independent variables is more than 1.
4. The equation of the best fit regression line $Y = \beta_0 + \beta_1 X$ can be found by minimising the cost function (RSS in this case, using the ordinary least squares method), which is done using the following two methods:
    - **Differentiation**
    - **Gradient descent**
5. The strength of a linear regression model is mainly explained

by R^2, where $R^2 = 1 - (RSS/TSS)$.
    - **RSS:** Residual sum of squares
    - **TSS:** Total sum of squares
6. RSE helps in measuring the lack of fit of a model on a given data. The closeness of the estimated regression coefficients to the true ones can be estimated using RSE. It is related to RSS by the formula:

$$RSE = \sqrt{\frac{RSS}{df}},$$

where $df = n - 2$ and n is the number of data points.

Module 2 : Simple Linear Regression in Python

Introduction:-

Welcome to the session on **Simple Linear Regression in Python**. So far, we have discussed the theory part of simple linear regression. Now, let's move on to building a simple linear regression model in Python.

In this session

You will learn about the generic steps that are required to build a simple linear regression model. You will first read and visualise the dataset. Next, you will split the dataset into train and test sets. After that, you will build the model on the training data and draw inferences. We have used the dataset and example from the ISLR book. You will use the advertising dataset given in ISLR and analyse the relationship between 'TV advertising' and 'sales' using a simple linear regression model. You will learn to make a linear model using two different libraries: **statsmodels** and **SKLearn**.

But before you move on to the Python code, let's do a quick recap of what you have learnt so far.

Assumptions of Simple Linear Regression:-

Before moving on to the Python code, we need to address an important aspect of linear regression: the assumptions of linear regression.

While building a linear model, you assume that the target variable and the input variables are linearly dependent. But do you need any assumptions other than this?

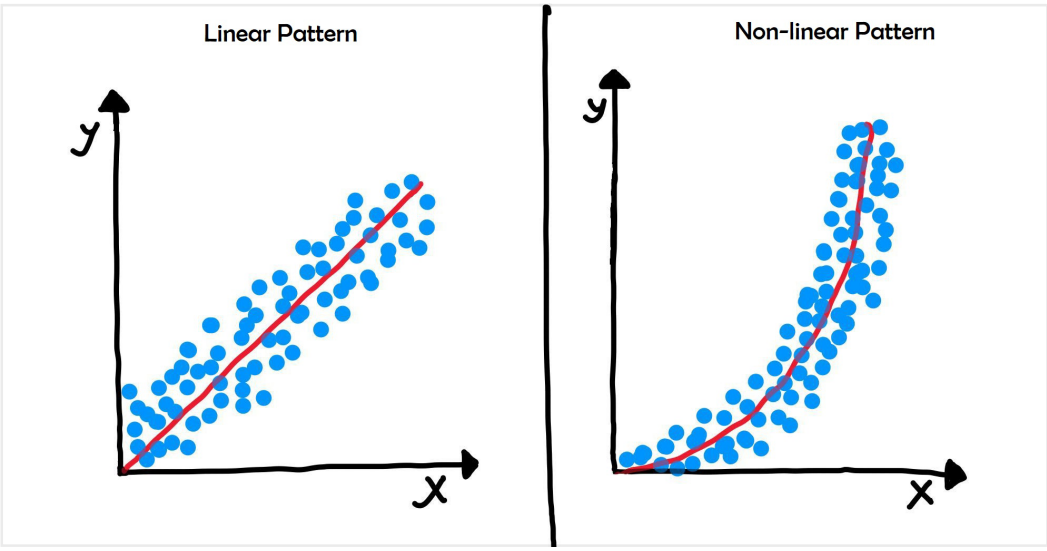
Let’s hear what Rahim has to say.

You are making inferences on the 'population' using a 'sample'. The assumption that variables are linearly dependent is not enough to generalise the results you obtain on a sample to the **population**, which is much larger in size than the sample. Thus, you need to have certain assumptions in place in order to make inferences.

Let's understand the importance of each assumption one by one:

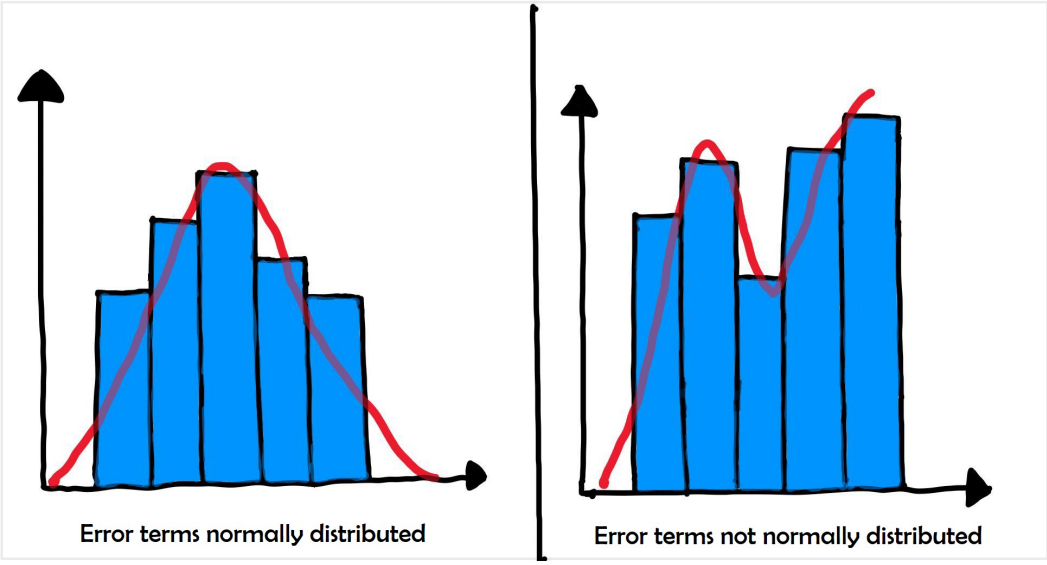
There is a *linear relationship* between X and Y:

- X and Y should display some sort of a linear relationship; otherwise, there is no use of fitting a linear model between them.



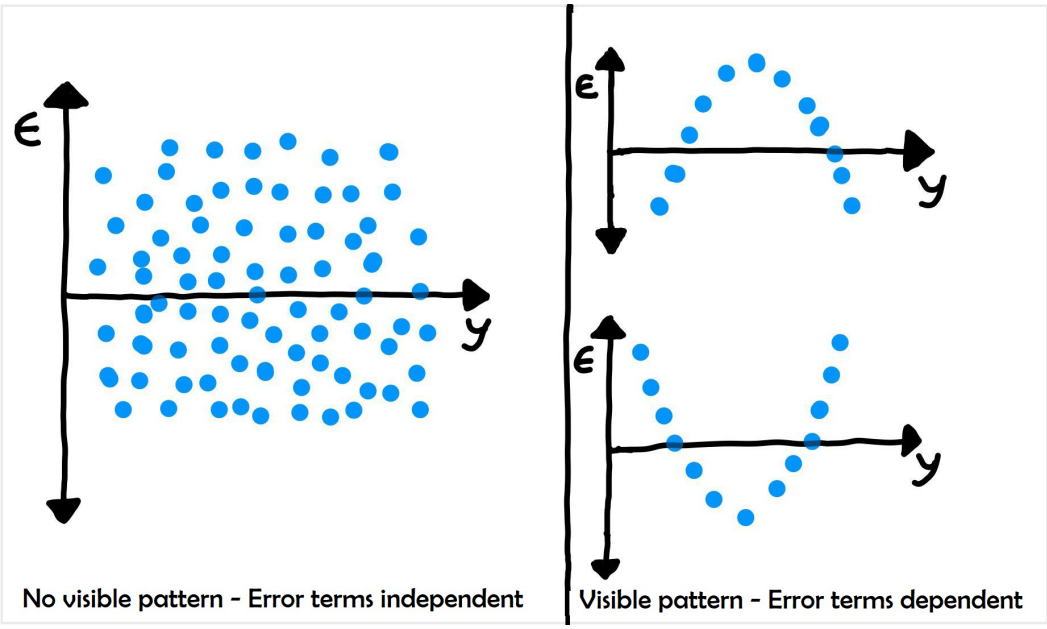
Error terms are *normally distributed* with mean zero(not X, Y):

- There is no problem if the error terms are not normally distributed if you just wish to fit a line and not make any further interpretations.
- But if you are willing to make some inferences on the model that you have built (you will see this in the coming segments), you need to have a notion of the distribution of the error terms. One particular repercussion of the error terms not being normally distributed is that the p-values obtained during the hypothesis test to determine the significance of the coefficients become unreliable. (You'll see this in a later segment)
- The assumption of normality is made, as it has been observed that the error terms generally follow a **normal distribution with mean equal to zero** in most cases.



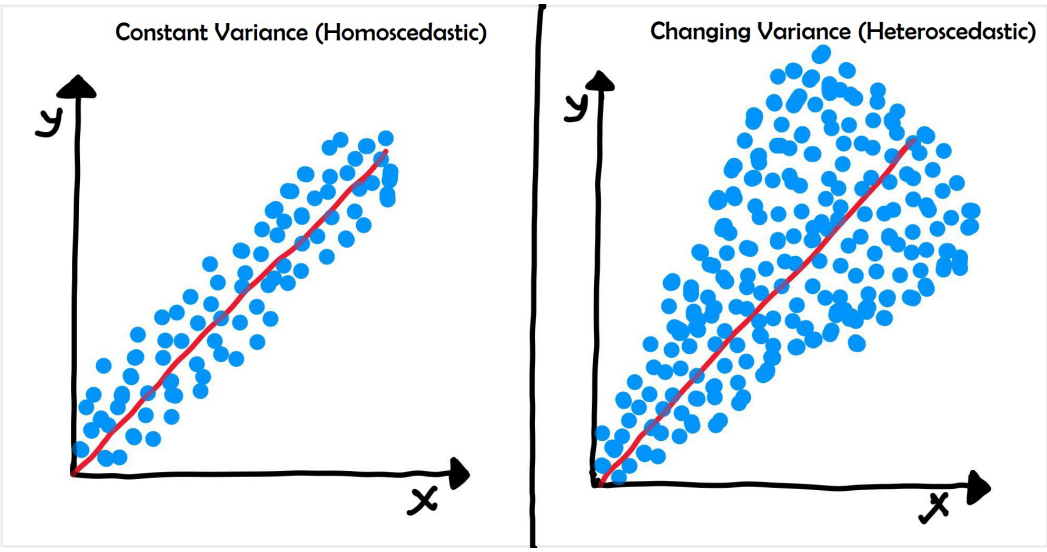
Error terms are *independent* of each other:

- The error terms should not be dependent on one another (like in a time-series data wherein the next value is dependent on the previous one).



Error terms have *constant variance* (homoscedasticity):

- The variance should not increase (or decrease) as the error values change.
- Also, the variance should not follow any pattern as the error terms change.



You will look at each of these assumptions in more detail later and validate these while building the model.

You can also go through the following link to see what happens when the assumptions are violated. But things will anyway get clearer once we keep moving ahead.

Reading and Understanding the Data:-

Before building a linear regression model, let's first understand and **visualise** the dataset.

Please download the following dataset and the python code and run it in the jupyter notebook as you watch the video.

You read the data and visualised it using '**seaborn**'. You also looked at the correlations between the target variable 'Sales' and the different predictor variables and saw that 'TV' has the strongest correlation with 'Sales'. Sales and TV are linearly correlated.

Coming up
Now that you have interpreted the data, let's start building a simple linear regression model on top of it, in the next segment.

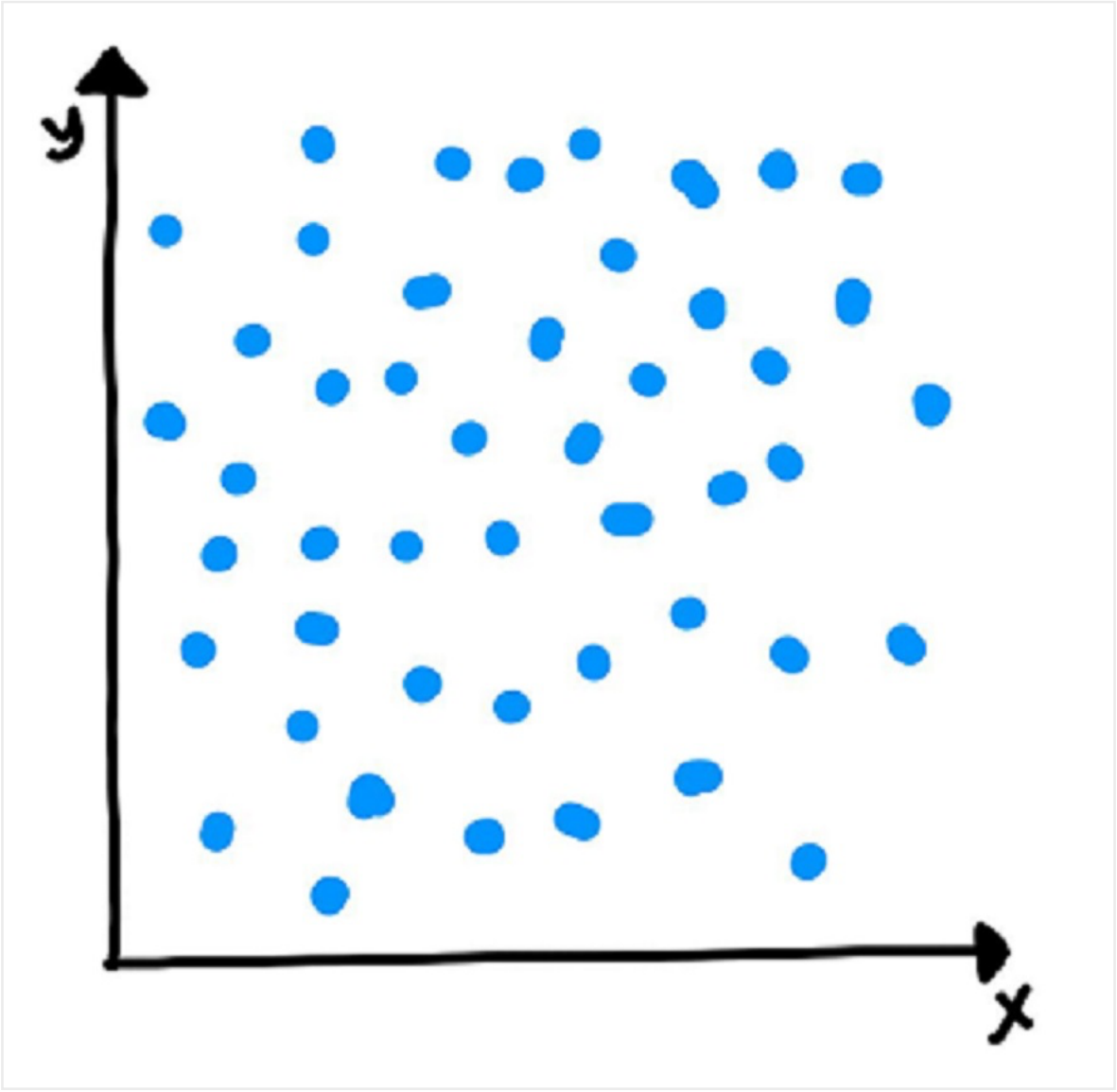
Hypothesis Testing in Linear Regression:-

Before you move on to the model-building part, there is still one theoretical aspect left to be addressed: the significance of the derived beta coefficient. When you fit a straight line through the data, you'll obviously get the two parameters of the straight line,i.e. the intercept (β0) and the slope (β1). Now,

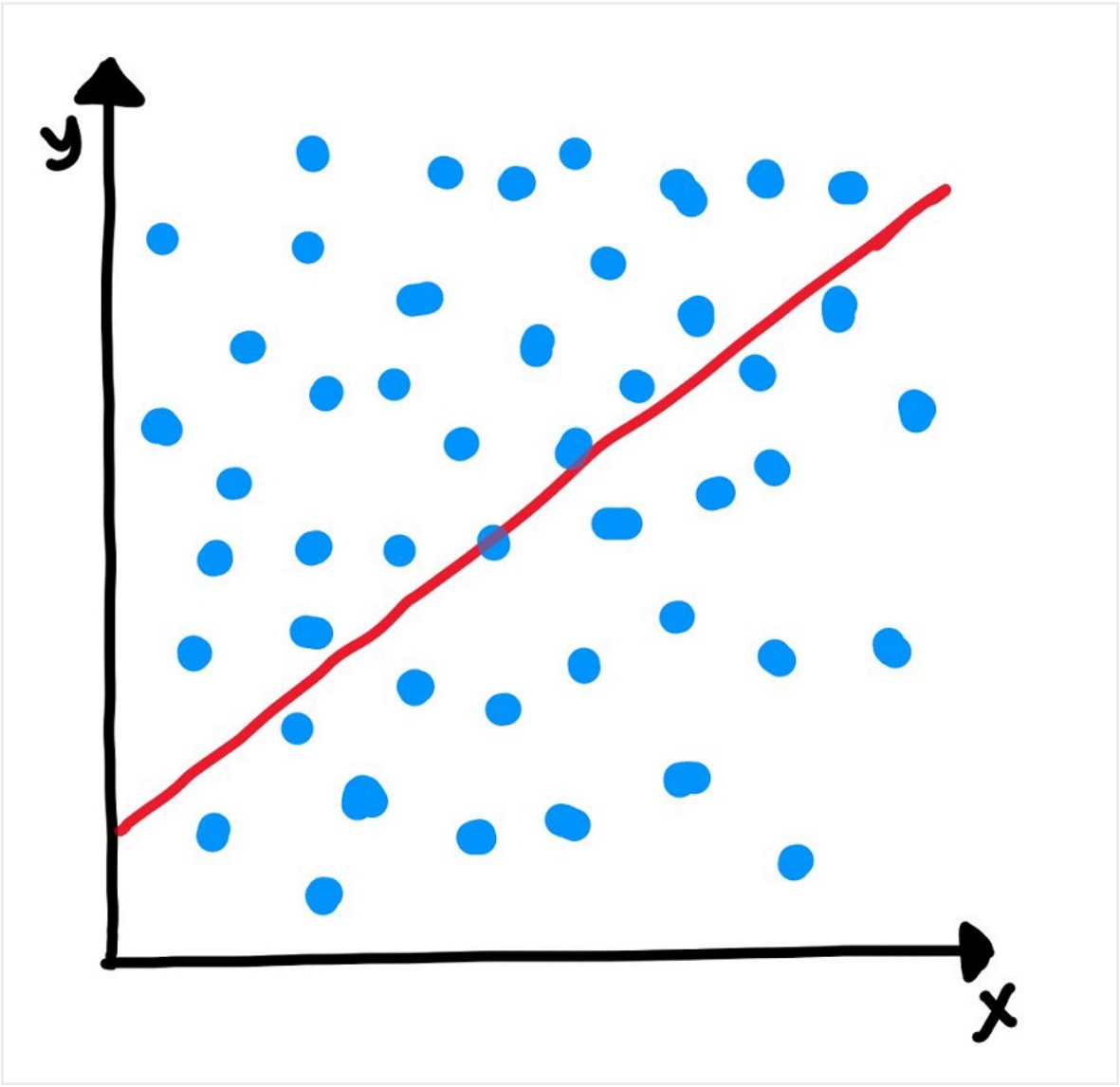
while β0 is not of much importance right now, but there are a few aspects surrounding that need to be checked and verified.

The first question we ask is, "Is the beta coefficient significant?" What does this mean?

Suppose you have a dataset for which the scatter plot looks like the following:



Now, if you run a linear regression on this dataset in Python, it will fit a line on the data which, say, looks like the following:



Now, you can clearly see that the data is randomly scattered and doesn't seem to follow a linear trend or any trend, in general. But Python will anyway fit a line through the data using the least squared method. But you can see that the fitted line is of no use in this case.

Hence, every time you perform a linear regression, you need to test whether the fitted line is a significant one or not or to simply put it,

you need to test whether β1 is significant or not. And in comes the idea of Hypothesis Testing on β1. **Please note** that the following text will assume the knowledge of hypothesis testing, which was covered in one of the earlier modules. Please revisit the module on hypothesis testing in case you need to brush up.

You start by saying that β1 is not significant, i.e. there is no relationship between X and y.
So in order to perform the hypothesis test, we first propose the null hypothesis that β1 is 0. And the alternative hypothesis thus becomes β1 is not zero.

- **Null Hypothesis** (H_0): $\beta_1 = 0$
- **Alternate Hypothesis** (H_A): $\beta_1 \neq 0$

Let's first discuss the implications of this hypothesis test. If you fail to reject the null hypothesis that would mean that β_1 is zero which would simply mean that β_1 is insignificant and of no use in the model. Similarly, if you reject the null hypothesis, it would mean that β_1 is not zero and the line fitted is a significant one.

Now, how do you perform the hypothesis test? Recall from your hypothesis testing module that you first used to compute the t-score (which is very similar to the Z-score) which is given by $\frac{X-\mu}{s/\sqrt{n}}$ where μ is the population mean and s is the sample standard deviation which when divided by $\sqrt{n}$ is also known as standard error.

Using this, the t-score for $\hat{\beta}_1$ comes out to be (since the null hypothesis is that β_1 is equal to zero):

$$\frac{\hat{\beta}_1 - 0}{SE(\hat{\beta}_1)}$$

Now, in order to perform the hypothesis test, you need to derive the p-value for the given beta. If you're hazy on what **p-value** is and how it is calculated, it is recommended that you revisit the segment on p-value. Please note that the formula of $SE(\beta_1)$ provided in the t-score above is out of scope of this course.

Let's do a quick recap of how do you calculate p-value anyway:

- Calculate the value of **t-score** for the mean point (in this case, zero, according to the Null hypothesis that we have stated) on the distribution
- Calculate the **p-value** from the cumulative probability for the given t-score using the t-table
- Make the decision on the basis of the p-value with respect to the given value of β
  (significance level)

Now, if the p-value turns out to be less than 0.05, you can reject the null hypothesis and state that β1 is indeed significant.

Please note that all of the above steps will be performed by Python automatically, which you'll learn in the very next segment.

Coming up
Now that you know how to determine whether your beta is significant or not, you'll start building the model in the next segment

Additional Reading
Why does the test statistic for β1 follow a t-distribution instead of a normal distribution? (here)

Building a Linear Model:-

Since 'TV' is very strongly correlated to 'Sales', let's first build a simple linear regression model with 'TV' as the predictor variable.

The first important step before building a model is to perform the test-train split. To split the model, you use the **train_test_split** function.

from sklearn.model_selection import train_test_split
X_train_lm, X_test_lm, y_train_lm, y_test_lm = train_test_split(X, y, train_size = **0.7**, test_size = **0.3**, random_state = **100**)

From now on, you will always use the SKLearn library to perform a test-train split before fitting a model on any data.

After you import the **statsmodel.api**, you can create a simple linear regression model in just few steps.

```
import statsmodels.api as sm
X_train_sm = sm.add_constant(X_train)
lr = sm.OLS(y_train, X_train_sm)
lr_model=lr.fit()
```

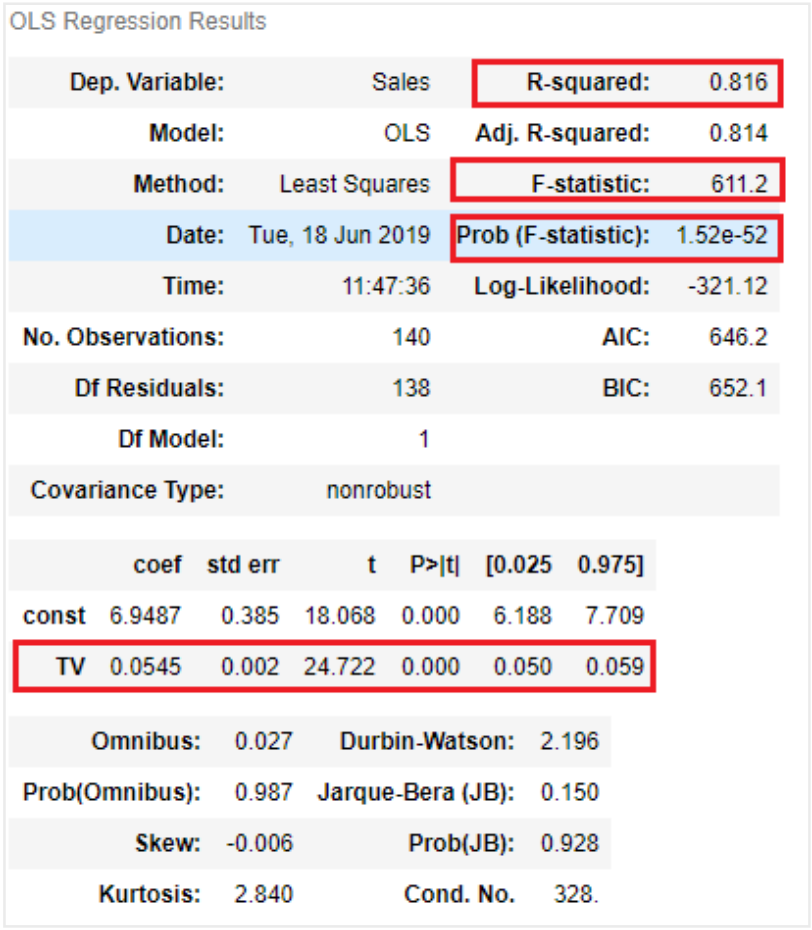
Here, **OLS stands for Ordinary Least Squares**, which is the method that 'statsmodels' use to fit the line. You use the command 'add_constant' so that statsmodels also fits an intercept. If you don't use this command, it will fit a line passing through the origin by default.

Now, let's take a look again at the summary statistics that was outputted by the model.

Summary Statistics

Now, let's take a look at the summary statistics that was outputted by the

model again.



OLS Regression Results

Dep. Variable:	Sales	R-squared:	0.816
Model:	OLS	Adj. R-squared:	0.814
Method:	Least Squares	F-statistic:	611.2
Date:	Tue, 18 Jun 2019	Prob (F-statistic):	1.52e-52
Time:	11:47:36	Log-Likelihood:	-321.12
No. Observations:	140	AIC:	646.2
Df Residuals:	138	BIC:	652.1
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	6.9487	0.385	18.068	0.000	6.188	7.709
TV	0.0545	0.002	24.722	0.000	0.050	0.059

Omnibus:	0.027	Durbin-Watson:	2.196
Prob(Omnibus):	0.987	Jarque-Bera (JB):	0.150
Skew:	-0.006	Prob(JB):	0.928
Kurtosis:	2.840	Cond. No.	328.

F-statistic

You were introduced to a new term named **F-statistic** and **Prob(F-statistic)**. Now, recall that in the last segment, you did a hypothesis test for beta to determine whether or not the coefficient β1 outputted by the model was significant or not. Now, F-statistic is similar in the sense that now instead of testing the significance of each of the betas, it tells you whether the overall model fit is significant or not. This parameter is examined because many a time it happens that even though all of your betas are significant, but your overall model fit might happen just by chance.

The heuristic is similar to what you learnt in the normal p-value calculation as well. If the '**Prob (F-statistic)**' is less than **0.05**, you can conclude that the overall model fit is significant. If it is greater than 0.05, you might need to review your model as the fit might be by chance, i.e. the line may have just luckily fit the data. In the image above, you can see that the p-value of the F-statistic is **1.52e-52** which is practically a zero value. This means that the model for which this was calculated is definitely significant since it is less than 0.05.

This will be more appreciable when you study multiple linear regression since there you have a lot of betas for the different predictor variables and thus it is

very helpful in determining if all the predictor variables together as a whole are significant or not or simply put, it tells you whether the model fit as a whole is significant or not.

R-squared
Like you studied earlier as well, R-squared value tells you exactly how much variance in the data has been explained by the model. In our case, the R-squared is about 0.816 which means that the model is able to explain 81.6% of the variance which is pretty good.

Coefficients and p-values:
The p-values of the coefficients (in this case just one coefficient for TV) tell you whether the coefficient is significant or not. In this case, the coefficient of TV came out to be **0.0545** with a standard error of about **0.002**. Thus, you got a t-value of **24.722** which lead to a practically **zero p-value**. Hence, you can say that your coefficient is indeed significant.

Apart from this, the summary statistics outputs a few more metrics which are not of any use as of now. But you'll learn about some more of them in multiple linear regression.

Let's see how the model actually looks by plotting it.

You visualised the predicted regression line on the scatter plot of the training data which is one of the things you should do as a part of model evaluation.

Coming Up
Now that you have fit the straight line, you will analyse the residuals and make predictions on the test set, in the next segment.

Additional Reading
The calculation of F-statistic is a complex task and is not required. Hence it is out of the scope of this course. But interested students can check out this link.

Residual Analysis and Predictions:-

Now that you have built the linear model, you can move ahead with the next steps. Now, building the model on the train set has two parts: fitting a line and validating the assumptions of regression.

Recall that one of the assumptions that you studied was that the error terms should be normally distributed with mean equal to 0. So once you have built the model, you'd need to verify if your model is not violating this assumption. And doing this is fairly simple: you just plot a **histogram of the error terms** to check whether they are normally distributed. And another assumption was that the error terms should be independent of each other. Again for this, you need to plot the error terms, this time with either of X or y to check for any patterns.

Let's see all of this in action in the following video.

The residuals are normally distributed, and there are no visible patterns in the error terms (except for the fact that the variance seems to be increasing a little for the higher values). So, this model fit looks good. Let's go ahead and make predictions on the test set.

You will use the '**predict()**' function present in 'statsmodels.api' for the same.

Coming up

So, overall, your model fit seems to be doing a good job. These were the steps for fitting a line using '**statsmodels**'. In the next segment, you will look at another library called '**SKLearn**', which can be used to perform linear regression as well.

Linear Regression using SKLearn:-

So far, you have worked with the '**statsmodels**' package. This is a great package if you want to fit a line and draw inferences as well. But many times, you may not be interested in the statistics part of linear regression. You might just want to fit a line through the data and make predictions. In such cases, you can use **'SKLearn'**, which involves lesser hassle than 'statsmodels'. Also, the industry standard as to what package should be used varies widely.
Some companies prefer statsmodels whereas some others prefer **SKLearn**, so it is better for you if you know about both of these packages.

SKLearn is a 'lighter' version of the linear regression packages. The two simple steps to fit a line using SKLearn are as follows:

```
from sklearn.linear_model import LinearRegression

lm = LinearRegression()   # Create a linear regression object
lm.fit(X_train, y_train)
```

Coming up
In the next segment, let's summarise your learnings from this session.

Summary:-

In this session, you built a simple linear regression model in Python using the advertising dataset. You also saw some more theoretical aspects in between. Here's a brief of what you learnt in this session.

1. A quick recap of simple linear regression
2. Assumptions of simple linear regression
    - Linear relationship between X and y.
    - Normal distribution of error terms.
    - Independence of error terms.
    - Constant variance of error terms.
3. Hypothesis testing in linear regression
    - To determine the significance of beta coefficients.
    - $H_0 : \beta_1 = 0; H_A : \beta_1 \neq 0.$
    - T-test on the beta coefficient.
    - $t\ score = \frac{\hat{\beta}_i}{SE(\hat{\beta}_i)}.$
4. Building a linear model
    - OLS (Ordinary Least Squares) method in statsmodels to fit a line.
    - Summary statistics
        - F-statistic, R-squared, coefficients and their p-values.

5. Residual Analysis

- Histogram of the error terms to check normality.
- Plot of the error terms with X or y to check independence.

6. Predictions

- Making predictions on the test set using the 'predict()' function.

7. Linear Regression using SKLearn

- A second package apart from statsmodels for linear regression.
- A more hassle-free package to just fit a line without any inferences.

Rahim has also answered some common doubts surrounding linear regression in the following additional segment. You can go through the segment here. This part has also been included in the notebook provided to you at the beginning of the session.

Coming Up
In the next session, you will move from simple linear regression to multiple linear regression wherein you will use multiple independent variables to explain a single dependent variable.

Module 3 : Gradient Descent-Linear Regression

Module 4 : Multiple Linear Regression

Introduction:-

Welcome to the session on '**Multiple Linear Regression**'. So far, we have discussed simple linear regression, where the model is built using one independent variable only. But what if you have **multiple independent variables**? How do you make a predictive model in such a case? Building a multiple linear regression on top of such data is one such solution.

In this session
You will use the example of sales prediction using the TV marketing budget that you saw in the previous session to build a multiple linear regression model. But now, instead of just one variable, you will have three variables to deal with. The

marketing budget will be split into three marketing channels: **TV marketing**, **radio marketing** and **newspaper marketing**. You will see how adding more variables brings in many new problems and understand how to approach them. Finally, you will learn about feature selection and feature elimination to build the most optimal model.

This session is almost completely a theoretical session on multiple linear regression and its various aspects. So, don't worry much if you don't get everything in the first go; the concepts will become clearer when you see each of these aspects in action in the next session, which is a Python demonstration on multiple linear regression.

Motivation - When One Variable Is Not Enough:-

The term '**multiple**' in multiple linear regression is self-explanatory; it represents the relationship between two or more independent input variables and a response variable. Multiple linear regression is needed when one variable is not sufficient to create a good model and make accurate predictions.

Let's hear Rahim talk about it.

You saw that multiple linear regression proved to be useful in creating a better model, as there was a significant change in the value of the R-squared. Recall that the R-squared for simple linear regression using 'TV' as the input variable was 0.816. When you have two variables as input, namely 'Newspaper' and 'TV', the R-squared increases to 0.836. Using 'Radio' along with 'TV' increased its value to 0.910. So, it seems that adding a new variable helps explain the variance in the data better.

It is recommended that you check the R-squared after adding these variables to see how much the model has improved.

Let's now look at the formulation of multiple linear regression; it is just an extension of simple linear regression. Hence, the formulation is largely the same.

Most of the concepts in multiple linear regression are quite similar to those in simple linear regression. The formulation for predicting the response variable now becomes this:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \epsilon$$

However, some other aspects still remain the same:

1. The model now fits a hyperplane instead of a line.
2. Coefficients are still obtained by minimising the sum of squared errors, the least squares criteria.
3. For inference, the assumptions from simple linear regression still hold: zero mean, independent and normally distributed error terms with constant variance.

Moving from SLR to MLR - New Considerations:-

When moving from a simple linear regression model to a multiple linear regression model, you have to look at a few things.

Let's hear Rahim explain them in the following video.

The new aspects to consider when moving from simple to multiple linear regression are as follows:

1. **Overfitting**
    - As you keep adding variables, the model may become far too complex.
    - It may end up memorising the training data and, consequently, fail to generalise.
    - A model is generally said to overfit when the training accuracy is high while the test accuracy is very low.
2. **Multicollinearity**
    - This refers to associations between predictor variables, which you will study later.
3. **Feature selection**
    - Selecting an optimal set from a pool of given features, many of which might be redundant, becomes an important task.

Coming up
Rahim discussed overfitting in the video above. In the next segment, you will study multicollinearity using which you can reduce the number of variables in your model.

Multicollinearity:-

In the last segment, you learnt about the new considerations that are required to be made when moving to multiple linear regression. Rahim has already talked about overfitting. Let’s now look at the next aspect, i.e., **multicollinearity**.

Multicollinearity refers to the phenomenon of having related predictor (independent) variables in the input data set. In simple terms, in a model that has been built using several independent variables, some of these variables might be interrelated, due to which the presence of that variable in the model is

redundant. You drop some of these related independent variables as a way of dealing with multicollinearity.

Multicollinearity affects the following:

- **Interpretation**
  - Does “change in Y when all others are held constant” apply?
- **Inference**
  - Coefficients swing wildly, signs can invert.
  - Therefore, p-values are not reliable.

It is, thus, essential to detect and deal with the multicollinearity present in a model while interpreting it. Let's see how you can detect it.

You saw two basic ways of dealing with multicollinearity:

1. Looking at **pairwise correlations**
   - Looking at the correlation between different pairs of independent variables
2. Checking the **variance inflation factor** (VIF)
   - Sometimes, pairwise correlations are not enough.
   - Instead of just one variable, the independent variable may depend upon a combination of other variables.
   - VIF calculates how well one independent variable is explained by all the other independent variables combined.

The VIF is given by:

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

Here, '*i*' refers to the i-th variable, which is being represented as a linear combination of the rest of the independent variables. You will see VIF in action during the Python demonstration on multiple linear regression.

The common heuristic we follow for the VIF values is:
> 10: VIF value is definitely high, and the variable should be eliminated.
> 5: Can be okay, but it is worth inspecting.
< 5: Good VIF value. No need to eliminate this variable.

But once you have detected the multicollinearity present in the data set, how exactly do you deal with it? Rahim answers this question in the following video.

Some methods that can be used to deal with multicollinearity are

as follows:

1. **Dropping variables**
   - Drop the variable that is highly correlated with others.
   - Pick the business interpretable variable.
2. **Creating a new variable** using the interactions of the older variables
   - Add interaction features, i.e., features derived using some of the original features.
3. **Variable transformations**
   - Principal component analysis (covered in a later module)

Coming up

In the next segment, you will learn how to handle the categorical variables present in a data set.

Dealing with Categorical Variables:-

So far, you have worked with numerical variables. But many times, you will have non-numeric variables in the data sets. These variables are also known as categorical variables. Obviously, these variables cannot be used directly in the model, as they are non-numeric.

Let’s see how you can deal with these variables in the following video.

When you have a categorical variable with, say, 'n' levels, the idea of dummy variable creation is to build 'n-1' variables, indicating the levels. For a variable, say, 'Relationship' with three levels, namely, 'Single', 'In a relationship', and 'Married', you would create a dummy table like the following:

Relationship Status	Single	In a Relationship	Married
Single	1	0	0
In a Relationship	0	1	0
Married	0	0	1

As you can clearly see, there is no need to define **three** different levels. If you drop a level, say, 'Single', you will still be able to explain the three levels.

Let's drop the dummy variable 'Single' from the columns and see what the table looks like:

Relationship Status	In a Relationship	Married
Single	0	0

In a Relationship	1	0
Married	0	1

If both the dummy variables, i.e., 'In a relationship' and 'Married', are equal to zero, it means that the person is single. If 'In a relationship' is denoted by 1 and 'Married' by 0, it means that the person is in a relationship. Finally, if 'In a relationship' is denoted by 0 and 'Married' by 1, it means that the person is married.

Note that **scaling just affects the coefficients** and none of the other parameters, such as t-statistic, F-statistic, p-values and R-squared.

Two major methods are employed to scale the variables:
Standardisation and MinMax scaling.
Standardisation brings all the data into a standard normal distribution with mean 0 and standard deviation 1.
MinMax scaling, on the other hand, brings all the data in the range of 0-1. The formulae used in the background for each of these methods are as given below:

- Standardisation: $x = \frac{x - mean(x)}{sd(x)}$
- MinMax Scaling: $x = \frac{x - min(x)}{max(x) - min(x)}$

Coming up

In the next segment, you will learn how to assess and compare models. For example, you can penalise models which use a large number of variables.

Additional reading

- To know more about dummy variables (here)
- Why it's necessary to create dummy variables (here)
- When to normalise data and when to standardise (here)
- Various scaling techniques (here)

Model Assessment and Comparison:-

Once the model is built, you would want to assess it in terms of its predictive powers. For multiple linear regression, you may build more than one model with different combinations of the independent variables. In such a case, you would also need to compare these models with one another to check which one yields optimal results.

Let’s hear more on this from Rahim.

Now, for the assessment, you have a lot of new considerations to make. Besides, selecting the best model to obtain decent predictions is quite subjective. You need to maintain a balance between **keeping the model simple** and **explaining the highest variance** (which means that you would want to keep as many variables as possible). You can do this using the key idea that a model can be penalised for keeping a large number of predictor variables.

Hence, there are two new parameters that come into the picture:

$$Adjusted\ R^2 = 1 - \frac{(1-R^2)(N-1)}{N-p-1}$$

$$AIC = n \times log(\frac{RSS}{n}) + 2p$$

Here, n is the sample size, meaning the number of rows you would have in the data set, and p is the number of predictor variables.

Coming up

The adjusted R2 adjusts the value of R2 such that a model with a larger number of variables is penalised. In the next segment, Rahim will talk about feature selection.

Additional reading

- AIC
- BIC
- Mallows' CP

Feature Selection:-

The one crucial aspect of multiple linear regression that remains to be discussed is feature selection. When building a multiple linear regression model, you may have quite a few potential predictor variables; selecting just the right ones becomes an extremely important exercise.

Let’s see how you can **select the optimal features** for building a good model.

> Note that in the brute force approach, one of the combinations out of 2^p does not make any sense, namely the one which does not use any features at all. So, this means that we only need to try 2^p - 1 feature combinations.

To get the optimal model, you can always try all the possible combinations of independent variables and see which model fits best. But this method is time-consuming and infeasible. Hence, you need another method to get a decent model. This is where manual feature elimination comes in, wherein you:

1. Build the model with all the features,
2. Drop the features that are the least helpful in prediction (high p-value),
3. Drop the features that are redundant (using correlations and VIF),
4. Rebuild the model and repeat.

Note that the second and third steps go hand in hand, and the choice of which features to eliminate first is very subjective. You will see this during the hands-on demonstration of multiple linear regression in Python in the next session.

Now, manual feature elimination may work when you have a relatively low number of potential predictor variables, say, ten or even twenty. But it is not a practical approach when you have a large number of features, say 100. In such a case, you automate the feature selection (or elimination) process. Let's see how.

You need to combine the manual approach and the automated one in order to get an optimal model relevant to the business. Hence, you first do an automated elimination (coarse tuning), and when you have a small set of potential variables left to work with, you can use your expertise and subjectivity to eliminate a few other features (fine tuning).

Coming up
In the next segment, let’s summarise what you learnt in this session.

Summary:-

Here’s a brief summary of what you learnt in this session:

1. When one variable might not be enough
    - A lot of variance values are not explained by just one feature.
    - Predictions are inaccurate.
2. Formulation of MLR
3. New considerations to be made when moving from SLR to MLR
    - Overfitting
    - Multicollinearity
    - Feature selection

4. Dealing with categorical variables
    - Dummy variables for fewer levels
5. Feature scaling
    - Standardisation
    - MinMax scaling
    - Scaling for categorical variables
6. Model assessment and comparison
    - Adjusted R-squared
    - AIC, BIC
7. Feature selection
    - Manual feature selection
    - Automated feature selection
    - Finding a balance between the two

Module 5 : Multiple Linear Regression in Python

Introduction:-

Welcome to the session on '**Multiple Linear Regression in Python**'. In the last session, you learnt about the various theoretical aspects of multiple linear regression. Now, let's move on to building a multiple linear regression model in Python.

In this session
You will learn the generic steps that are required to build a multiple linear regression model. You will build this model for a **housing dataset** and predict the price of a house using the various potential predictor variables provided. You will first read and visualise your dataset and then prepare your data for building a linear model. This will include dealing with categorical variables, adding dummy variables, and scaling. You will then start building the model with a bottom-up approach, i.e., you will start with one variable and keep on adding more. Once all the variables have been added, you will perform a manual feature elimination and move on to the **residual analysis** and predictions, as usual. In the end, you will solve the same problem using **RFE**.

Reading and Understanding the Data:-

Linear regression is used in various fields such as real estate, telecom, e-commerce, etc. to build predictive models. Let's look at one such example from the real estate industry. Here, you will predict the price of a house on the basis of some predictor variables, such as floor area, number of bedrooms, parking space, etc.

Problem Statement:

Consider that a real estate company has the data of real estate prices in Delhi. The company wants to optimise the selling price of the properties, based on important factors such as area, bedrooms, parking, etc.

Essentially, the company wants:

- To identify the variables affecting house prices, e.g., area, number of rooms, bathrooms, etc.
- To create a linear model that quantitatively relates house prices with variables, such as the number of rooms, area, number of bathrooms, etc.
- To know the accuracy of the model, i.e. how well do these variables predict the house prices

Please download the dataset from below.
Please download the python code from below to practice along.

Now that you've read and inspected the data, let's move on to visualising it. This will help in interpreting the data well and identifying the variables that can turn out to be useful in building the model.

That was all about visualising the numerical variables. You might have noticed that there are a few categorical variables present in the dataset as well. Let's visualise them too, using **boxplots**.

Coming up
In the next segment, you will do the data preparation, which is an important step before model building.

Data Preparation:-

Now that you have a sense of what variables are important and that the data is well behaved with very few outliers, let's move on to preparing the data for multiple linear regression. This involves handling the categorical variables first and then performing dummy encoding.

You have successfully handled the categorical variables with two levels. But one of the columns – 'furnishingstatus' – has three levels. Here, you need to perform dummy encoding.

Coming up
Now that the data has been prepared, let's do the scaling in the next segment.

Initial Steps:-

Before model building, you first need to perform the **test-train split** and **scale the features**.

Scaling of variables is an important step because, as you may have noticed, the variable ‘area’ is on a different scale with respect to all other numerical variables, which take very small values. Also, the categorical variables that you encoded earlier take either 0 or 1 as their values. Hence, it is important to have everything on the same scale for the model to be easily interpretable.

In the next video, let's continue with the data preparation process before modelling it.

You have seen the two popular rescaling methods- Min-Max scaling and Standardisation (mean=0 and sigma=1). The advantage of Standardisation over the other is that it doesn't compress the data between a particular range as in Min-Max scaling. This is useful, especially if there is are extreme data point (outlier). Now, let's rescale and fit the data.

Now that you have prepared the data and are done with the test-train split, let’s prepare a heat map and take a look at the correlations between the variables.

Coming up
In the next segment, you will start building a multiple linear regression model to predict values based on multiple variables, instead of a single one.

Building the Model - I:-

Now that everything is in place, let's build the model. You will follow a bottom-up approach for this, i.e., you will start by building the model with just one variable. Hence, the choice of this variable becomes very crucial. Let’s see which variable turns out to be ‘The Chosen One’.

Even though ‘**area**’ is the most correlated variable, it could explain only 28% of the variance. After that, we added **'bathroom'** as it had the second-highest correlation with the target variable. Then the model was able to explain 50% of the variance.

Coming up
Adding two more variables has improved the model. From an adjusted R-squared value of 28%, it has moved to 50%. In the next segment, you will proceed with improving the model further.

Building the Model - II:-

The bottom-up approach was just to give you an idea of how the parameters change when the number of variables is increased. More generally, you first build a model using all and then try to improve the model by dropping some of them.

Let’s now build a model with all the available variables.

In the following video, you will use two main parameters to judge the insignificant variables, the p-values and the VIFs.

Coming up
Now that we seem to have a fair model, in the next segment, let's perform the final steps - **analyse the residual terms** and **predict** the price.

Residual Analysis and Predictions:-

Before making the predictions, you need to be certain that the model is reliable. To that end, you need to first perform a residual analysis of the error terms and then move on to making the predictions on the test set; and finally, evaluate the model based on the predictions. Let's see how Kshitij performs residual analysis for multiple linear regression.

Now that the model building is done, let’s go ahead and make inferences on the model.

Let's summarize what you have learned in this session.

Coming up
In the next segment, you will build a model using recursive feature elimination (RFE), which is an automated technique.

Variable Selection using RFE:-

Even though the housing dataset doesn’t have many variables and you can easily select the relevant features manually, it is important to learn to use certain automated techniques as well.

Let’s see Rahim rebuild another model for the housing dataset using RFE.

Before that please download the python notebook for model building using RFE given below. The dataset used in this case is the same dataset, i.e. the 'Housing' dataset that you had used earlier.

In order to use RFE, you need to use SKLearn instead of statsmodels. Let’s go ahead and create a linear model using SKLearn to perform RFE.

Now that the features have been selected and the model is built, let's do the residual analysis and make the final predictions.

Coming up
In the next segment, Rahim will summarise the concepts that you learnt for multiple linear regression.

Summary:-

In the following video, Rahim will sum up what you have learnt so far related to multiple linear regression.

Module 6 : Industry Relevance of Linear Regression

Introduction:-

Welcome to the session on 'Linear Regression: Industry Relevance'. So far, you have learnt the theoretical aspects of linear regression; it is more important to learn how to implement them in the industry.

In this module

You will gain an insight into the relevance and use of linear regression modelling in industries. Ujjyaini, our subject matter expert, will take you through a real-life example that will help you assess what you have understood so far, get an idea of the industrial applications of linear regression and get ready for the industry.

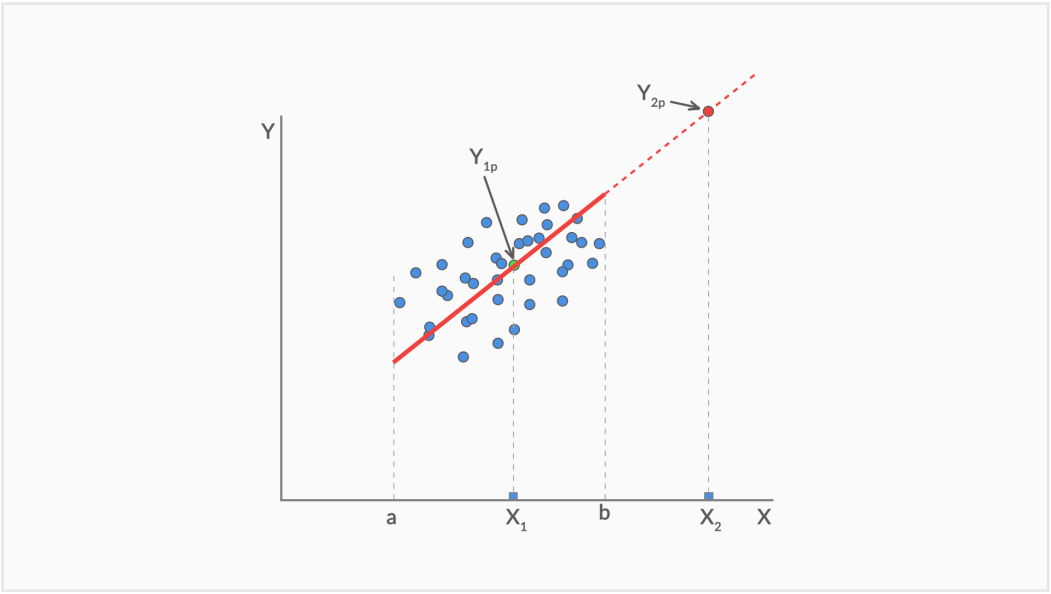
Linear Regression - Revision:-

You practised linear regression modelling in the previous sessions. Let's revise some of the concepts you learnt.

Here, Ujjyaini mentioned that regression guarantees interpolation of data and not extrapolation.

Interpolation means using the model to predict the value of a dependent variable on the independent values that lie within the range of data you already have. Extrapolation, on the other hand, means predicting the dependent variable on the independent values that lie outside the range of the data the model was built on.

To understand this better, look at the diagram below. The model is built on the values of x between a and b.



When you wish to predict the Y for X1, which lies between a and b, it is called interpolation. On the other hand, extrapolation would be extending the line to predict Y for X2, which lies outside the range on which the linear model was trained.

For now, you only need to understand what these terms mean. You will learn more about these in the upcoming lectures.

Ujjyaini also mentioned that linear regression is a parametric model.

Even though a detailed discussion on parametric and non-parametric models is beyond the scope of this module, a simple explanation is given below. You may also refer to the additional resources provided below.

In simple terms, a parametric model can be described using a finite number of parameters. For e.g., a linear regression model built using n independent variables will have exactly n 'parameters' (i.e., n coefficients). The entire model can be described using these n parameters.

In the upcoming modules, you will learn about some non-parametric models as well, such as decision trees.
They do not have a finite set of parameters that completely describe the model.

To read more on this topic further, here are some useful links:

- Parametric vs non-parametric models in brief
- A detailed explanation of parametric and non-parametric models

It is very crucial to understand when to apply linear regression modelling. Let's go through some business cases to understand where you can apply linear

regression modelling.

You saw various cases and learnt where linear regression modelling can be used and where it cannot. Answer the question below to test your understanding.

Prediction vs Projection:-

In the last lecture, you learnt when linear regression modelling can be applied.

You will often see the terms 'prediction' and 'projection' being used interchangeably; however, in terms of modelling, these are different applications. Let’s learn the difference between them.

In the previous segment, you got a basic idea of interpolation and extrapolation. Let's now learn more about these in detail.

In this lecture, you learnt that linear regression can be effectively used for interpolation, but extrapolation may not necessarily be effective/accurate.

In the next lecture, you will look at a case study for a media company.

Media Company Case Study:-

Recall the case of a media company from the business cases that we considered for the application of linear regression; this case is quite similar.

The problem statement is as follows: A digital media company (similar to Voot, Hotstar, Netflix, etc.) had launched a show. Initially, the show got a good response, but then, it started witnessing a decline in viewership. The company wants to figure out what went wrong.

You can download the data set, data dictionary and the IPython Notebook from below:

Let’s see how we can use linear regression to identify the root cause of the decline in viewership.

Let's now understand the data that we will use to derive our insights. You should open the Media Company data set that you downloaded above.

You looked at the digital media company data set. You understood the various attributes that could have led to a decline in the new show’s viewership. In the next lecture, you will look at the exploratory data analysis of the data set.

Additional readings:
Read to understand more about ad impressions here.

Exploratory Data Analysis:-

Let's try and see what we can learn from the data before going deeper into it.

[**Note** - The graph shown in the video above is actually a **line chart** since the individual points in the original scatter plot have been joined together to form a continuous graph]

You now have a good understanding of the viewership data. The next step is model building.

Model Building - I:-

To arrive at a data-driven solution, you would use a linear regression model with an aim to understand what the predictors / driver KPIs of viewership are. Then, based on the model, you will be able to determine the root cause of the viewership decline.

Let's now dive into model building. Our SME Ujjyaini will illustrate the model building process through the forward selection method, which you learnt about in the variable selection. You would have already downloaded the Python Notebook for the case study. If not, you can download the same from the link below.

Our subject matter expert first began the model with variables that seemed to be the most important at first instance. The model was updated with changes in the variables and the addition of new variables that help explain the outcome better.

The first question you tried to answer was whether the decline in the number of visitors to the platform was the main reason for the decline in the show viewership. You started with a weekday variable and then replaced it with a weekend variable to help explain the spikes in the graph better. Then you added the Character A variable, which shows the impact of the character's presence in the show viewership.

Intuitively, it appears that the number of views today must be related to the views yesterday, e.g., if you had viewers watching the show yesterday, a large proportion of them are also expected to watch the show today.

In the code provided, the lag variable has already been created. Run the model including the lag variable along with the Weekend, Visitor and Character A variables and answer the following question.

Model Building - II:-

Let's now look at the next steps in model building.

Upon the introduction of more variables, the model starts becoming complex. Some of the variables, which were significant earlier, now become insignificant. Variables that have better alternatives are replaced.
You brought back variables that were previously removed to test whether they fit the business requirement better. For example, you brought back Visitors as a predictor against Views to the platform because Visitors could be impacted through marketing actions.

In the latest version of the model (Model 6), Ujjyaini spotted two major problems:

1. The variable 'Visitors' has become insignificant with a large p-value, and
2. The variable 'character A' has changed its sign.

Before proceeding any further, answer the following questions:

Model Building - III:-

Let's continue the process of model building.

In the process of building the model, you checked for multiple variables and then arrived at the variables that best suited the model.

Based on your understanding of the models, answer the following questions:

Now that you have a model with a fairly high adjusted R-squared, it may seem that the task is done. But you still might be missing out on an important variable. Thus, you need to assess the model. In the following lectures, you will learn some important methods of assessing the model and confirming that the model you arrived at is the best you could have, given the data you have

Assessing the Model:-

In the last lecture, you arrived at a model that has a fairly high adjusted R-squared. This means that the model was able to explain the viewership pattern to a large extent.

Before we finalise the model, let's run some checks to assess whether the model we built was right.

You finally have a model that seems good enough to predict why the show

viewership fell. You saw that the actual and predicted views overlapped, thus indicating that the model is able to explain the change in viewership very well.

Then, you saw that the errors (the differences between the actual views and the views predicted by the model) were randomly distributed. What this essentially confirms is that there are no variables that could have helped explain the model better.

A non-random error pattern, on the other hand, would mean that the errors are capturing some patterns, thus indicating that the model could have been better. A non-random error pattern indicates that there are certain systematic unexplained aspects in the outcomes that are being captured in the error. This pattern in the errors could probably have been explained by some explanatory variable, which is missing in the model. So, the idea is that a model should explain everything that is possible such that only the random errors are left.

So, in total, you built eight models before you arrived at the final model. Let's check whether the previous models would have passed the checks.

You saw that the plot results for model 5 were clearly not as good as model 9. The views predicted by model 5 were not able to capture the highs and lows that the actual show views had. There was a seasonality pattern in the error plot. This pattern in the errors could probably have been explained by some explanatory variable, which is missing in model 5.

A good model tells a good story. It is not important for you to base your story entirely on a single model. Drawing insights from the current model (model 9) and previous models, Ujjyaini identified **Ad Impressions** and **Character A** as the driver variables that could be used to increase the viewership of the show. **Ad impressions are directly proportional to the marketing budget**. Thus, by increasing the marketing budget, better viewership could be achieved. Similarly, Character A's absence and presence create a significant change in show viewership. Character A's presence brings viewers to the show. Thus, these two variables can be acted upon to improve the show viewership.

In the next segment, let's build on the understanding that we have obtained so far. You will see how the marketing team can devise further steps of action to drive ad impressions.

Additional reading
Read more about the bootstrap method here.

Interpreting the Results:-

It is not always possible to change the script of the story at a moment's notice.

Character A’s dates might also not be available. Thus, the marketing team can focus on ad impressions instead.

Let's learn how the model can be used by the marketing team to drive ad impressions

In this lecture, you learnt how to interpret the model so that you can gain some actionable insights.

The marketing team wanted to know the additional investment required to increase the ad impressions and, thus, increase the viewership of the show.

A model equation was used to find the increase in ad impressions required to take the show viewership back to the peak levels attained in week 13. The ad impressions figure could then be used by the marketing team to calculate the required increase in the marketing budget.

Summary:-

Let’s summarise what you learnt:

1. It is important to understand if a linear regression modelling will be applicable to the problem you are trying to solve. For example, linear regression cannot help you decide if a customer will opt out of a subscription, as this is a classification problem.
2. Linear regression guarantees interpolation but not extrapolation.
3. While linear regression can be used for both projection and prediction, there is a difference between the two. In prediction, the goal is to identify the most important variables that explain the outcome in a simpler way. In projection, the goal is to accurately forecast the outcome, no matter how complex the model gets.
4. The business goal is crucial and will decide what path the modelling process should take.
5. In the industry, variables that are actionable are valued over others. For example, given two quite similar variables, “Views to the platform” and “Visitors to the platform”, the latter is more actionable, as it is easier to get viewers to the platform than forcing anybody to watch the shows.
6. You don’t end the modelling process until you are sure that no more significant variables could be added to explain the outcome. Thus, you check for randomness of errors, which could indicate whether any KPI that could have helped explain the outcome was left out.