Artificial Intelligence (AI) for Investments

Prof. Abhinava Tripathi
Department of Industrial and Management Engineering
Indian Institute of Technology, Kanpur

Lecture- 42

In this lesson, we will discuss the application of regression algorithm with R programming through a financial market case study. This includes modeling security price data with single and multiple features. First, we will train the algorithm using the training dataset and examine various goodness of fit measures. And next, we will test the algorithm on test dataset and examine its out of sample prediction efficacy. In this video, we will introduce ABC stock price case study which entails forecasting stock prices using regression algorithm. Please note that stock price prediction or stock return prediction is an attempt to determine the future value of a company based on analysis of factors which impact its price movement.

There are number of factors that help in predicting stock prices. These can be macroeconomic factors like state of country's economy, growth rate, inflation and so on. There are also other factors that are more specific to a stock like profit margin, debt to equity ratio, sales and so on. Let's examine the data provided in this case study.

Case Study: Sentiment Problem

So we are given the data for stock market price for ABC company, along with Nifty and Sensex (market indices). We are also given the data of dividend announcement and a sentiment index

Date	Price	ABC	Sensex	Dividend Announced	Sentiment	Nifty
03-01-2000	718.15	0.079925	0.073772	0	0.048936	0.095816
04-01-2000	712.9	-0.00731	0.021562	0	-0.05504	0.009706
05-01-2000	730	0.023987	-0.02441	0	0.019135	-0.03221
06-01-2000	788.35	0.079932	0.012046	0	0.080355	0.011205
07-01-2000	851.4	0.079977	-0.0013	0	0.094038	-0.0004
10-01-2000	919.5	0.079986	0.019191	1	0.015229	0.030168
11-01-2000	880	-0.04296	-0.04025	0	-0.07217	-0.04966
12-01-2000	893.75	0.015625	0.036799	0	0.01396	0.020999
13-01-2000	875	-0.02098	-0.00845	0	0.057518	-0.01164
14-01-2000	891	0.018286	0.004858	1	0.008828	0.020714
17-01-2000	819.75	-0.07997	-0.01228	0	-0.12395	-0.00962
.....	.....	.....	.....	.....	.....	.....
.....	.....	.....	.....	.....	.....	.....

So we are given data for market prices of a company ABC along with Nifty and Sensex. These are market indices that represent broad market wide movement of securities and we are also given data of dividend announcement and a sentiment index. Let's discuss

this data in more detail now. Think of a portfolio manager who has to build a model for a particular stock which is ABC in this question and under consideration. The manager wants to predict the ABC stock price return for this stock using regression model.

The data provided to us starts from 2000 and goes till 2019. So we have approximately 19 years of data. We can construct daily returns as provided here. So we have daily returns of ABC. These are provided in this column.

And next let's call it column B. Next we have daily return on Sensex. This is column C and daily return on Nifty in this column. Please note that Sensex and Nifty are two main stock indices used in India. They are the benchmark Indian stock market indices that represent the weighted average of the largest Indian companies.

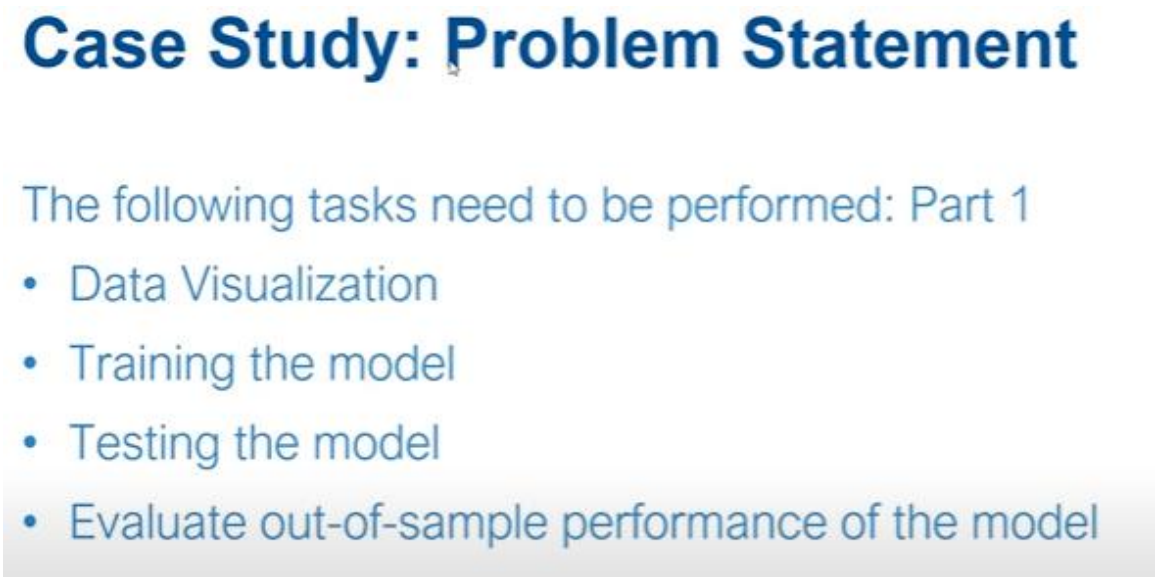
So Sensex represents average of 30 largest and most actively traded Indian companies. Similarly, Nifty represents a weighted average of 50 largest Indian companies. Another variable here is dividend announcement which is provided here in this column. This is 1. This variable attains a value of 1 if the company has announced a dividend on a particular date and 0 otherwise.

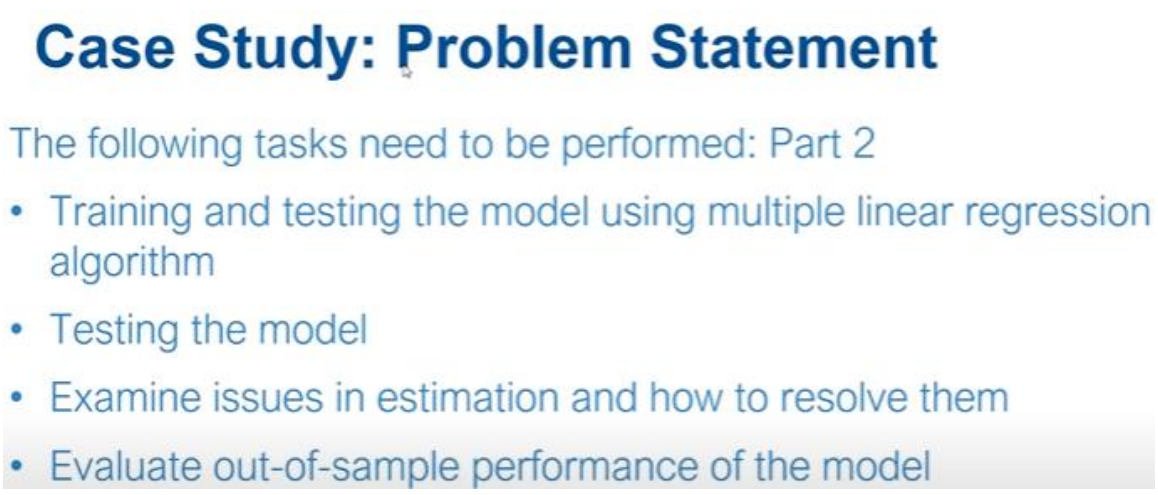
So for example, if the value is 1 on this given date of 10th January 2000 because the company ABC announced a dividend on this date and it is 0 for other days as we can see when the company did not announce any dividend. Please remember this is a dummy variable. Lastly, we have sentiment variable in this column. It is a sentiment score that quantifies how investors feel about ABC company. It can be based upon news analysis or upon option market analysis or based on some survey of investors.

We would not go into details of the construction of this score here and take it as given. A very high sentiment score represents bullish emotions of investors and vice versa. To summarize this video, in this video we introduced ABC case study problem with respect to stock price forecasting. We discussed the background of the case study, its relevance and implication for financial markets. We discussed the variables and data provided in the case study, the interpretation of each of the variables and its implication for the regression modeling.

In this video, we will introduce the problem statement with the ABC case study. The problem statement will be demonstrated and implemented with the R programming. In the first part of R problem implementation, we will start with the simple linear regression model. We will start the study with data visualization. We will plot the returns and cumulative returns for ABC stock and market indices like NIFTY and Sensex.

Then, we will segregate the data into test and train dataset. We will train and build the model using the simple linear regression algorithm, using market index as the independent variable and ABC returns as the dependent variable using the training dataset. So, we will train the model first. Then we will evaluate the model with various goodness of fit measures and through visualization. Thus, we will test the model and also as a part of testing the model, we will evaluate the out of sample model performance on test dataset using various quantitative measures and also with visualization.





In the next set of implementation, we will employ multiple linear regression algorithm. We will start with training and building the model using multiple linear regression algorithm, using market indices, sentiment, dividends among other variables with the

training dataset. Next, we will evaluate the model with various goodness of fit measures and through visualization. There are several issues that we will deal with. These include multicollinearity, heteroscedasticity and autocorrelation.

We will examine these issues and we will try to find ways to resolve them. We will answer the question how to resolve these issues in the estimation and then we will evaluate out of sample performance on the test dataset using various quantitative measures and also with visualization. To summarize this video, we discussed the problem statement that will be implemented through R programming. In this video, we will start with our implementation of ABC case study regression problem. We start with the ABC case study problem.

We will conduct the following steps. First we will load the relevant packages. In this case study, we are going to use read excel package. This package will help us in reading excel files, data in the excel format. We will make use of lubrided package which will help us in date time manipulation.

Then we will use moments package for understanding various properties of data. Then we will load car package. Car package along with lmtest will help us in various regression related aspects. And lastly we have sandwich package that will help us in handling the estimated estimation related issues in the regression. So with this now we have loaded all the relevant packages.

As a next step, we will set our working directory. Please remember, please remember it is always a best practice to set the working directory. In order to set the working directory, we will go to this session button, set working directory, go choose working directory. And we have this particular file and folder if I keep put on open button. Notice a command appears on my console window.

This command suggests that now that this command is run, appropriate working directory has been set. So for reference, I will just copy paste this command on my code. So now that my working directory has been set, I can read the data file. So data, I will use read excel command.

The file name is abc.xlx. You can save it with any name. I have the file in with the name of abc.xlx. So now the file is saved. You can check a few initial elements of the data file by using head data command.

Notice there is a date variable, there is a price variable, return variable abc, sensex returns, dividend announcement dummy variable 01, sentiment variable and nifty as we

have discussed during the case study problem statement discussion. You can also check the dimensions of this data with this brim data command. It has 5153 rows that means 5153 daily observations for 7 variables just that we just saw. One important issue to be noted, you can check the class of various elements. For example, if I want to check the class of date variable, let us see.

So I put dollar symbol which means I am accessing that variable inside data. So it is already a date variable. Also please note there is a time element which is spurious kind of, spurious in nature. We need not handle this time element for now. We only need date which means month, the year.

So I want to remove this particular aspect. I can do that very easily. I can simply just run this date, data dollar date and I can change the convert type to date. Earlier it was in POSEXCT type of class which includes not only date but time element. Since we are making use of only date element, I simply convert it to the date file.

So now if I run this, notice the time element is eliminated, only date is there. I can run this class command again. So its class is date. So now it is not anymore the POSEXCT element that it was earlier.

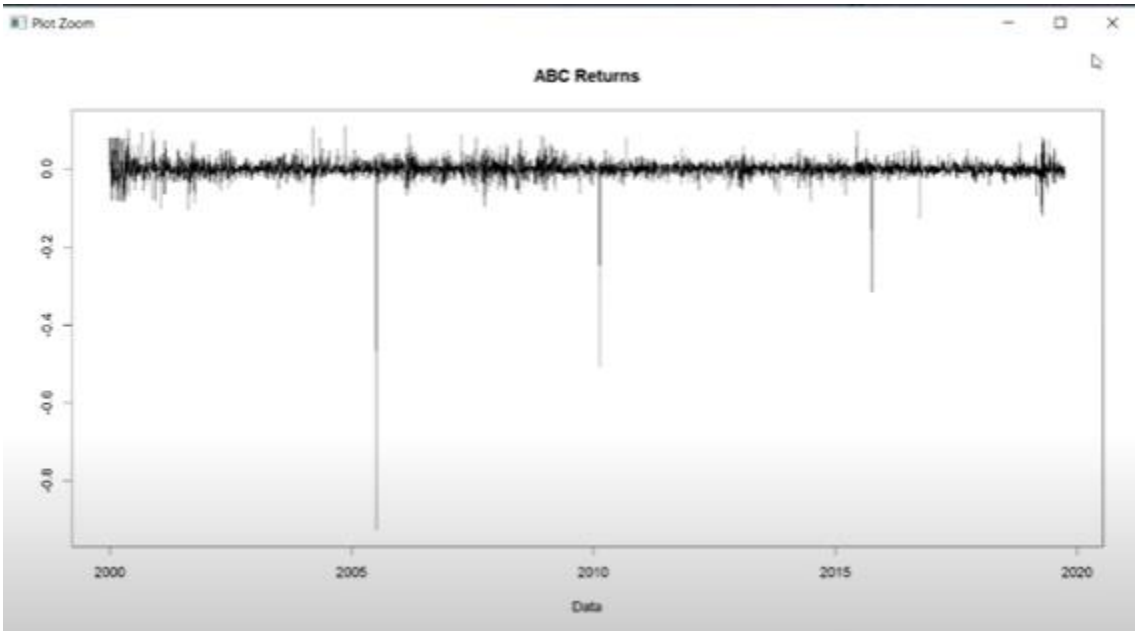
Now it is only purely a date element. So to summarize this video, we set the working directory. We load the relevant packages and then we set the working directory. We read the data file which is abc.lx in our data variable which is essentially a data frame.

We can check that also. For example, you can type the class command and check the data frame, it is a data frame. Then we check the dimensions of this data. We converted our date variable from POSEXCT date time object to purely a date object. From the next video, we will try to visualize the data with its relevant properties. In this video, we will conduct the visualization of data.

We will visualize the data. We will examine key variables, we will conduct the visualization of key variables including returns on abc and the market variable that is nifty. So, first let us plot the abc returns. So, we will use the plot command. On the x axis, we will have the date variable.

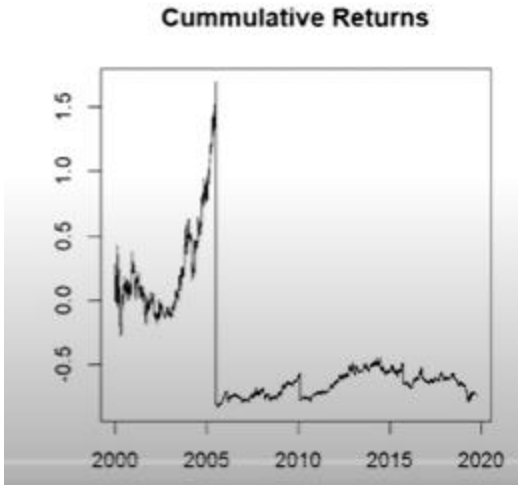
On the y axis, we have the return variable. We will give the x axis name using xlab command as date. So this will give us the x axis as date and then y axis label as, we will keep the y axis label empty but we will give them central heading. We will use central heading or main heading as abc returns. So, we will plot the abc returns along with prices and we would like to see the plotting format in the line form.

So we will put type as L. Now, as soon as I run this command, we will notice that a plot will appear on the plot window here. We can zoom it, you can see the plot. You can see there are some extreme falls here around 2005 and 2008 and 10, extreme down movements, increase in volatility. If you want, you can export this plot to save as image. You can use different image formats to save this plot.



For example, you have png, you have jpeg, tiff, various formats available to save. Also you can export it in the form of pdf, you can save it as pdf or you can copy it to clipboard and then paste it on word document and PowerPoint presentation and so on. So, this is the return variable, abc returns. I would like to plot the cumulative returns on abc. So how to compute cumulative returns? Let's name this variable as cumret.

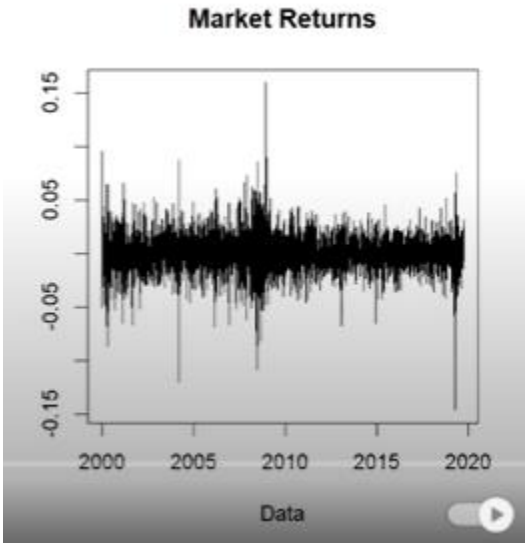
So we would like to plot this cumulative return variable. The simple way to compute cumulative returns are use data dollar price. This will use the price and divide it by the beginning price, opening price. So any price divided by opening, its opening value will give us the cumulative returns here. So I need to use the first value of price and then I subtract it from 1 to reflect the return aspect.



So this will compute the cumulative return in my data dollar cumulative return variable. Now I need to simply plot it. So I can use the plot. Again, I can simply copy paste the command that I used earlier, the same command, only that now instead of writing as abc, I can use this new variable, cumret, cumulative return variable. I can use that and then name the plot as cumulative returns.

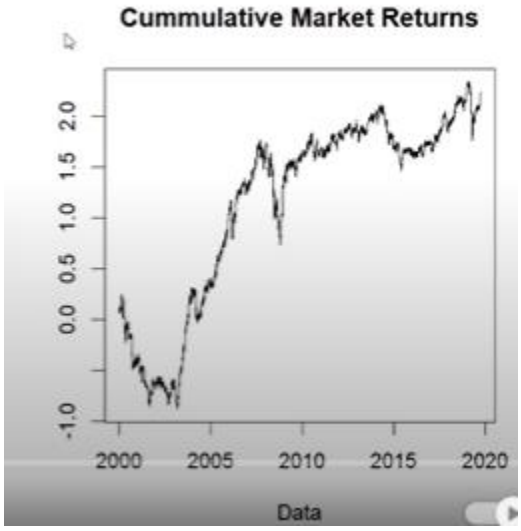
And if I run this, the cumulative returns, you can see the cumulative returns provided here. So it looks like this. There is a sharp fall around here, 2005. These are cumulative returns.

Now we will plot the market returns. So we will plot the nifty returns, market or nifty returns. Very simple. I do not have to do any extensive coding for that. I will simply use my plot command that I created earlier, I wrote earlier.



And instead of using abc, I will do a minor change here. I will write nifty. So this will give me the market return nifty, other things remain the same. I will use the instead of abc, I will change the central heading to market returns. And now if I run this command, notice these are my market returns.

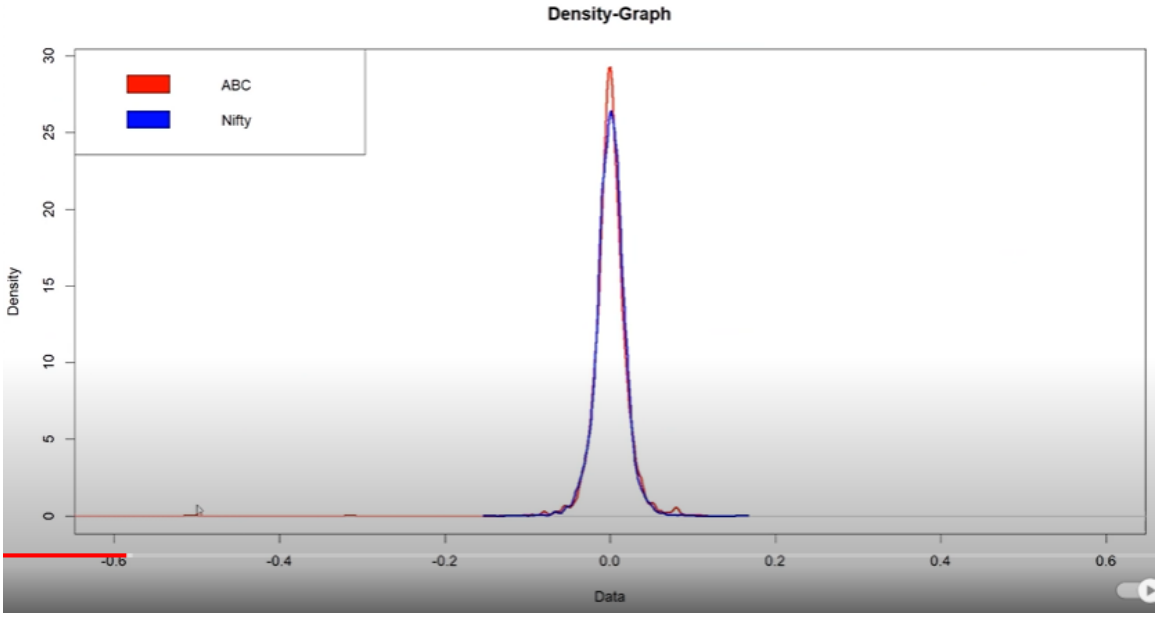
So these are simple market returns. Next we will plot the cumulative market returns, cumulative market returns. I have to just write a very simple command, which is data dollar cumulative. This time around we are computing nifty cumulative. So I will use this and the very simple cumsum command, cumulative sum command to compute the cumulative nifty returns.



This will provide me with the cumulative nifty returns. And I will not write the command again. So I will just simply use the same command that I used earlier. And I will just change here cumulative return instead of cumulative return, I will use cumulative nifty returns. And in the central heading, I will change the name to cumulative market returns. So here I can see the behavior of cumulative market returns easily.

This is the behavior of cumulative market returns. To summarize this video, in this video, we conducted basic visualization of return variables. This included abc returns and nifty returns. We also plotted the cumulative returns. In the next video, we will conduct the summarization of data, we will provide some summary measures. We will also have a look at the normality, skewness and kurtosis of the data and stationarity of the data as well.

In this video, we will discuss the basic properties of our data. We start with a very basic summary measure. So we will start by summarizing the data. For that we will use summary command, simply summary command. And for summary command for abc returns, we can see how abc returns are distributed, the minimum value which is quite high and the maximum value and first quantile which is 25 percent, median value which segregates the data between two halves, the mean of the distribution and 75 percentile of the data.



So these values are provided. Similarly, I can examine the summary of nifty market index which is this. So here also we can see extreme return which is minus 14 percent, maximum is around 15.97 which is 16 percent and same matrix that is first quantile, median, mean and third quantile. In the next step, we will conduct a very important operation.

We will try to see the probability density of our data. In R, the same can be implemented with this plot command and then density, data dollar abc. So this will plot that simply the density of abc but we want some aesthetic aspects also. So we will type main as, then we will type here as density. This is a density graph.

So we will type density graph and then we will name the x and y axis. So x axis we will just name simply the data. Also we would like to, since we are plotting the density for both market and abc, we will try to give them different colors. First we will plot the density of stock abc return, density in red. Let us give the line width of 2 so it will plot a slightly more solid line, line type as 1 so a more solid line will be there. And we

probably will, if you do not specify the limits can be extremely large so we will restrict the limits of x axis to minus 0.

6 to 0.6. Although these are, these will change depending upon your final result. You may modify and tinker with that a little bit here and there to improve the aesthetics. With this lines command, I do not want to plot a new graph. On the same graph, I want to plot nifty returns so I am using lines command nifty.

Then I would like to specify the color. Earlier I have used the red color to plot the abc returns so I am using the blue color to plot the nifty returns and line width again I will use the line width of 2. So this will be plotting the density of nifty and then I would like to add a nice legend on the, on one side. So in order to check this, let me plot the, okay so this is my plot of abc return then my density plot blue, in blue we have nifty. So now there is some empty space on probably the top left so I will use that space to put my legend here. So top left and I will use c abc then I will name the next legend as nifty because of this sequence since abc is in red so I will use the red here and nifty is in blue so I will use the blue here and now I will run the legend.

So if I run this legend a very nice legend will be added. Use fill equal to color and now the legend is added. Let us enlarge this plot a little bit. So notice on this plot abc is plotted in left and notice the behavior of abc it is extremely left skewed you can see the extreme left tail of abc while nifty is rather more balanced and symmetric in blue nifty returns are more symmetric over the sample period while abc which is in red are extremely skewed towards the left that means there are extreme negative returns which we also saw in the summary measures so it is sort of confirmation of that. To summarize this video we first summarize the nifty and abc returns using summary command and then we plotted their density distribution to understand the basic characteristics of our return data for abc stock and nifty.

In this video we will examine the normality and stationarity of the data. Now first as a part of normality examination we will start with the skewness of the data we will write skewness data dollar here so we will compute the skewness which is very highly negative as we already saw in the diagram the skewness of abc is extremely it is extremely left skewed then we will examine the skewness of nifty returns. So the nifty is also negatively skewed but the skew is not very high but we want a slightly more solid evidence for that so we will conduct basic against you know test of skewness so we will type data dollar abc and this will provide a more statistical measure and notice it provides you with a p value which suggests that the skewness is statistically significant in fact very significant for abc returns. Similarly we can compute the skewness of nifty returns so here we will write skewness data we computed the skewness of nifty returns

this is also in terms of magnitude not very high minus 0.177 but its significance is quite high so this is also slightly negatively skewed and low in terms of magnitude but reasonably higher in terms of statistical significance. Now skewness is a measure of asymmetry on the left side or right side and the remaining part we will discuss the kurtosis measure so we also examine kurtosis measure.

Kurtosis is a measure of piggness and tail so how peak or less peaked as compared to normal distribution and how tailed it is whether tails are fat or not. So first we will compute the kurtosis of abc returns. The kurtosis is quite high we will also measure the kurtosis of probably this can be accounted to the extremely fat tails of abc and peak nature as we saw in the density distribution. Nifty also has a very high kurtosis so we can see that it also has a very high kurtosis normal distribution has a kurtosis of 3 so as compared to normal distribution the kurtosis levels are very high. Now we would like to again have a slightly more statistically significant evidence so we will use the Anscombi test.

First we will compute it for abc returns $abc for abc so the kurtosis is not equal to in fact very significantly different from 3 for abc which we expected we will also see the kurtosis. Kurtosis is the measure of piggness and tail how fat tails are or how peak the distribution is as compared to normal distribution so the tails are fat and the piggness is very different from normal distribution. Here also kurtosis is very significantly different from normal distribution. As a final test we already know although that skewness and kurtosis are very different and as a combination skewness and kurtosis decide whether data is similar to a normal distribution the bell shaped curve that we discussed. As a final test we would like to have Harker Bearer test or it is called Harker Bearer test or Harker test which will give us a sense of its normality although we already know that data is not normal but it is still combined measure of normality which is very very high which is much greater so data is non-normal as you can see this is a combination of extreme skewness and peak nature of abc although in case of nifty this skewed nature was less it was not as skewed but still as compared to normal distribution it was skewed and the tails were fatter on both side as compared to normal distribution.

Same for nifty as well nifty we can see it was symmetric but still it is non-normal, normality is high we will see while working with the regression model we will see how to deal with these issues but we have seen the summary measures. Now just to summarize this video we computed the skewness, kurtosis and normality of the overall normality of the data. We found that abc was extremely left skewed and both of the distributions have fattailed as compared to normal distribution and higher peaks. This was statistically examined using Agastino's test and Anscombi test and using Harker Bearer test we also examined the overall normality of the data we found that both the series abc, return

series and NFT return series are non-normal in nature and the result is statistically significant as well. In this video we will discuss a very important property that is stationarity of the data.

Stationarity of the data refers to the fact that the mean and standard deviation or variance as we have already discussed the concepts these are changing with time and if these parameters such as mean and variance are changing with time estimation is fraught with various econometric issues. So, before estimating or estimating any kind of relationship it is important to know whether your data is stationary or not. There are three very useful tests for that and these are Augmented Decay Fuller test, Phillips Perot test and KPSS test for that these tests we need to load a very important package which is URCA unit root test. So, we will load this package and now we will start with our first we will conduct the Augmented Decay Fuller or often called ADF test. The command is quite simple we will write summary and ur dot df for Augmented Decay Fuller df data dollar abc is our price series.

So, we will use abc and notice the null hypothesis of this particular Augmented Decay Fuller test is the data is non-stationary. Since notice the test statistic it is minus 51.7 which is much lower and much higher in terms of magnitude as compared to 1%, 5% and 10%. Therefore, with a lot of confidence we reject the null and we say that data is stationary. Please note this is a non-stationarity test that means null hypothesis of this is that data is non-stationary.

Very similar syntax we will use to check the stationarity of the NIFTY returns. If the returns were non-stationary any model such as regression algorithm will be difficult to estimate properly. So, we will check the NIFTY returns also. NIFTY returns also notice that we reject the null of non-stationarity with a lot of confidence. So, again the data is stationary.

But while testing a stationarity it is very useful to test through a number of tests. So, for example, Augmented Decay Fuller test were test of non-stationarity. Here we will be using test such as PP test also. Look at the test such as ur dot pp and notice here again this is also a test of non-stationarity and we reject. Notice the test statistic which is very very large in magnitude very negative as compared to the standard statistic.

Using the large test statistic we can reject the null and again say that data is stationary. So, here also the null was of stationarity. So, we reject the null and we say that data is stationary. It is very high tested statistic. So, we are again and again saying that data is stationary.

Another very interesting test and generally it is required to test for stationarity test also. So, earlier both the tests earlier that we did Philips Perron Augmented Decay Fuller they were test of non-stationarity. Now, we will make use of KPSS test which is the test of stationarity. So, first we will run the KPSS for ABC returns and notice the test statistic is much lower than 1 percent, 5 percent and 10 percent level.

So, data is again we are not able to reject the null of stationarity. So, data is stationarity. Similar test we conduct for NIFTY returns also. KPSS test and here also we find the data is stationary. So, we have conducted the stationarity test. To summarize this video, we conducted test of stationarity and non-stationarity. Here Augmented Decay Fuller and Philips Perron test were test of non-stationarity where null hypothesis was of non-stationarity and rejection of null indicated stationarity while KPSS test was of stationarity and we failure to reject the null, we fail to reject the null which indicated again the stationarity of the data.

In the next set of videos, we will start with training and testing of regression algorithm and prediction of stock returns. In this video, we will segregate our main data which is ABC stock return data into training and rest data. Training data will be employed to train the linear regression algorithm while test data is employed to test the out of sample forecasting efficiency of the algorithm. So, we will start the analysis by segregating the data into training and test data. As a first step, we will create our train data segment which is equal to main data and we will tell R that the date should be less than equal to.

Now, we will filter the date with 2017 less than 2017 December. So, first December 2017 is our cut off point and this is the maximum and it should be great. Let us say we also filter observations before 2006. So, let us say we also want do not want to go too early. So, we filter those observations that are for which the year is greater than 2006.

So, we are filtering from 2007 to almost 2017. We are filtering all the observations. So, this is our train data and let us check this. So, we will check the head of train data head of train that that starts from 2007 as we can see here. Let us check the tail of the train data. It ends with 2017. As you can see here in the date segment, we can see on the console window it ends with 2017 and if you look at the dimension of the train data, it has 2850 observations.

Similarly, we will select our test data. Our test data is again in a similar manner. We will filter it. Data, dollar date should be greater than and now this time around we will use the exact same date criteria which is greater than as.date and we will precisely select the date that we selected earlier. We will select the same date in fact. So, all the observations with higher date number will be taken here.

So, now this test data will contain the remaining observations that are above 1st of December 2017. Let us examine this. So, the head of the test data is starts from 2017. In fact, 4th of December and look at the probably 2nd and 3rd are holidays and if you look at the tail of this, tail of the test data, the tail it ends with 2019 and if I examine the dimension of test data, so I have around 478 observations to test the algorithm. So, to summarize, in this video, we segregated our data into two segments. First the training data which starts from year 2007 and ends at 1st of December 2017 while the test data contains the remaining observations starting from 4th of December 2017 up till 2nd of October 2019.

In this video, we will train our simple linear regression model by regressing the ABC returns on NFT returns. So, we will model simple linear regression model by examining the ABC returns and NFT returns relationship. So, in R, this relationship examination is done with a very simple lm command. So, we will run this lm command which indicates some kind of linear model, linear regression model is about to take place and where we are regressing ABC returns. Here this notation suggests that ABC is the dependent return variable, ABC returns and the independent variable is NFT. Our data is, train data we are using to train this linear regression model and the output of this linear regression model, the train model output, the coefficients will be stored in SLR object, simple linear regression.

Let us see the summary of this object. We will assign, in fact, we will assign the summary to the model object, summary SLR. Let us have a look at the summary variable. I can simply extract this by printing model.

Notice the console window. On the console, we can see the printed output. In the output, the following objects to be noted. First, note the coefficient of NFT which is 0.397. This is also the measure of beta that we have discussed. Recall the discussion on beta that beta represents the sensitivity of ABC returns here that means if market moves by 1%, ABC stock returns move by 0.

39%. Notice, also notice the t value of the coefficient which is heavily significant. Remember, this t value is computed as the ratio between estimate which is the coefficient that is 0.397 here divided by the standard error.

Now, this t value 18.67 is very significant. The three star suggests that it is even significant at 1% level. Recall our discussion on the significance levels. Null hypothesis here is that this coefficient NFT has a value of 0 which means NFT does not affect, the movement in NFT does not affect, do not affect the returns of ABC stock. However, given this result, the significance, the very high significance of this result at 1%, we can

say definitely there is a positive impact of NFT that is market movement on ABC returns.

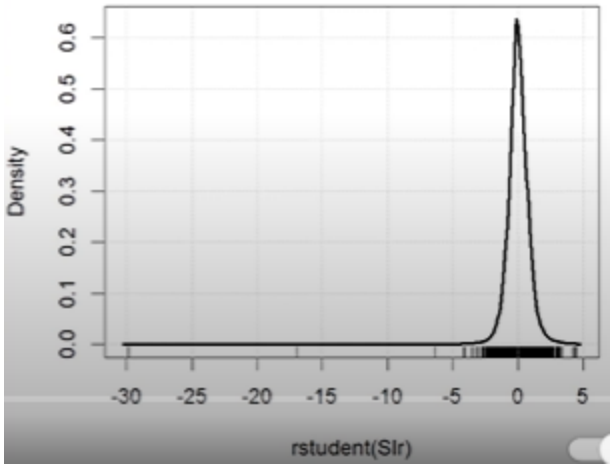
Here it appears if market moves by 1%, ABC stock moves by 0.39%. And this relationship is statistically significant at 1% level. So, we are very confident. In fact, we are more than 99% confident that there is a relationship between NFT returns and ABC stock returns. Moreover, notice this adjusted R square here and the multiple R square.

In fact, we will focus on adjusted R square which is a more improved version. It is 10.87%. This suggests that out of the total variation, out of the total variation in ABC return, 10.87% variation is explained by NFT which means and recall our discussion on the market risk and systematic risk that almost 10.87% part of the overall risk, total risk, total risk is 100%, 10.87% part of the risk is market risk or systematic risk driven by the market movements.

In this case, market is proxied here by NFT. The remaining approximately 89%, 89.21%, around 89% is the idiosyncratic risk which is specific to ABC stock. The remaining 10.87% is driven by market and explained by market. So, this is our interpretation of the overall regression. To summarize this video, we examined the relationship between ABC returns and NFT returns by regressing ABC returns on NFT returns.

We also examined the output of this regression model. We discussed the coefficient, its significance, the idiosyncratic and systematic part of the risk associated with ABC stock return. And we have now trained our linear regression algorithm. In fact, this is a simple linear regression algorithm because we are using only one variable which is NFT. So, now that we have trained our simple linear regression algorithm on the training dataset, now we will examine, we will examine the efficiency of this model in subsequent videos.

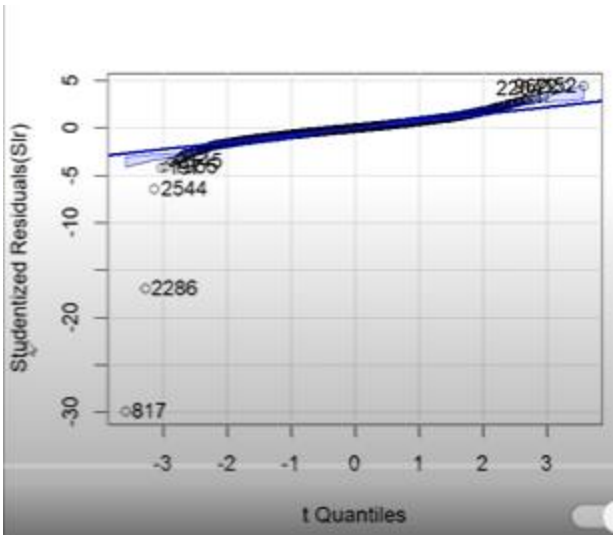
We will examine some of the econometric issues that this model may face and how to deal with them. In this video, we start with the residual diagnostics of the model. First, we look at the density plot of the error terms. So, we will use density plot. Recall that we have already stored our trained model in the model variable or model object.



So, we will simply use the model residuals. So, we will plot the density of this. Let us examine the density plot of residuals. Notice that while the shape of this density is relatively similar to normal distribution, however, its left tail is extremely long, which suggests that it is extremely skewed. We can also corroborate this by plotting the density plot of studentized residuals. For that, the command is simple. We can just write the same density plot and we type rstudent.

But this time, we will pass on the original model as LSR object, which is the model object that we created. So, again, it is also very similar. So, the density plot of residuals and as well as studentized residuals is extremely left skewed. Studentized, student's distribution is essentially t-distribution.

So, these are standardized errors. And as we can see, it is extremely left skewed. Another test to examine the behavior of the model and its residual is qqplot, which checks the normality. Let us examine the qqplot. So, we pass on the fitted object SLR, which we already fitted. We also would like to know some of the extreme observations that may create problems for us.



So, we will give the number as n equal to 10. So, this will identify 10 extreme observations for us. Let us see. So, we will run the qqplot here. Also, we would like to see some of the extreme observations. So, I type id equal to list n equal to 10.

So, we would like to highlight 10 extreme observations. I run this command and the model is plotted here. So, we can see the quantile wise distribution of studentized residuals. So, the residuals are plotted. And we can see this number of observations that are extreme, observation number 72, 152 and 2544.

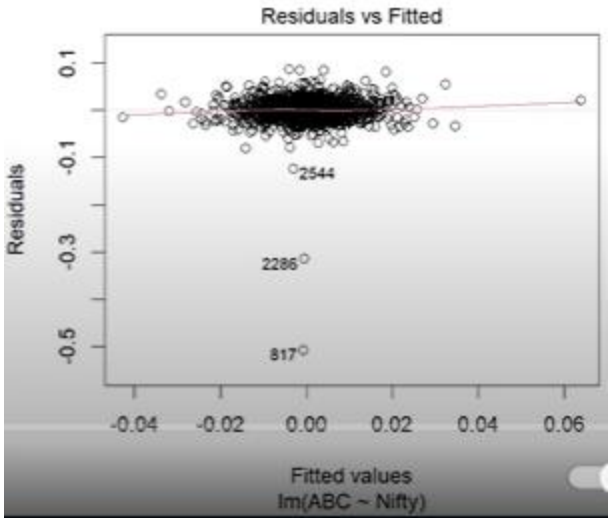
These are some of the extreme observations that depart from the normality. On the fitted model, these observations are plotted. Along the fitted model, we can see the observation numbers. The studentized residuals are plotted here. We can see some of the observations plotted here. So, this is, we can see that most of the time, the observations, the residuals are along the normal line, the straight line indicates the normal line, the observations that deviate from the straight line indicate deviation from normality. And we can see some of the fitted observations on the extreme left and right end, these some observations deviate from the normal distribution.

Normal distribution here is represented by the straight blue line. Lastly, we can also conduct the outlier test. Now, there is a basic outlier test and it will also adjust for Bonferroni values, adjusted p values. Let us see them. So, if I run this, notice, unadjusted p, there are lot of very significant unadjusted p values, but one, once there is Bonferroni correction, one can see the significance is lit, but still we have, there are six extreme values and these are appear to be, these appear to be very similar to the earlier ones. So, these are our extreme values. If you want to know more about these functions and their theoretical underpinnings, you can simply type question mark, qqplot and see the background, the entire background of these commands.

For example, this is for quantile comparison plot to check the normality and the outlier test also, you can check that the theory behind these commands by running the question mark, you can read more and more about these commands. Lastly, if you feel that these observations are vitiating, these extreme observations are vitiating your estimation model, you can remove them from their model. You already have the observation numbers, you can remove them from the model and then run the model again to improve its estimates.

To summarize this video, we created density plot of residuals. We also have examined the studentized residuals on the density plot. Next, we check the deviations from normality with the help of qqplot and we also conducted the outlier test to see some of the outlier observations. We found that there are number of outlying, outlier observations that may vitiate the estimation. As a future course of action, one can remove these observations from the model. But please note, sometimes these extreme observations also carry useful information probably that may help us model extreme values representing extreme events.

So sometimes it is not advisable to remove these values. So a subjective judgment call may have to be taken here. In this video, we will examine a very important issue that econometric issue that afflicts estimation which is heterocyticity, which is the non-constant variance. So as an intuition, let me plot the fitted model SLR and I will plot the residuals with this plot command. Notice the behavior of residuals with the fitted values. Notice the variance as we go ahead with values fitted values. Notice the variance is not constant, sometimes it is high, sometimes it is low and the way residuals are scattered around the fitted line, their variance seems to be not constant, which indicates that model may be afflicted by the issue of heterostasticity.



To check it more formally, there is NCB test. If you want to know more about it, you can type NCB test and all the details will be provided to you. However, now for the non-constant variance, error variance, we will simply run the command. So we will type NCB test and this is part of car package, CR package that we installed earlier. We need to supply it with the model object that we created earlier.

So model and first we will try with the studentized, without studentized errors and let us see. So let us see the output on the console. So I need to provide the fitted object actually. So instead of model, I need to provide the fitted object. Notice the p-value is very significant, which indicates and this and simple NCB test fits the values along the fitted values and it seems that when fitted along the fitted value, the error variance is not constant. It is varying very significantly which is indicated by this very high value of chi-square and a very high significance level of p-value.

So if you look at the p-value, it is quite high. We can also see if these residuals, they are varying with the nifty values. So we will specify the formula. This time around we do not want to see with fitted values, but we want to see with nifty. By default it runs with fitted values. This time we will model it with the nifty which is the nifty returns and along nifty returns also we can, we have similar evidence we can see, we can see that there is a very significant variance, residual variance that is not constant.

Another very interesting test of non-constant variance is BP test or Brouche-Pagrand test. If you want to know more about this, you can simply type question mark BP test and all the details are there. So this BP test is a very interesting test of where hydrostaticity, it simply checks whether error variance is changing. So let us run the model BP test and

we will supply the model object. Here we are supplying the model object and we will decide whether we want this error terms to be studentized.

So without studentized, again a very significant p-value, very similar to one we have earlier which indicates a very high level of hydrostaticity. If we studentize, let us see if we studentize the t-values, error values, this time we are not able to reject the null of homosidasticity. So when we are working with homosidastic, with studentized errors, probably it seems that the model is not afflicted that much with hydrostaticity. However, when we are running this model, this test, BP test, Brouche-Pagrand test without studentized error, then it gives us hydrostaticity. So probably the issue is there, the issue of hydrostaticity is slightly moderate in the sense that without student t-values, it is not affecting our estimation. So just to summarize, we tried to visually examine the issue of heterosidasticity, there appears to be, it appears to be the case that the residual variance is not constant.

However, when we conduct this using BP test with studentized errors, then our model seems to be not much afflicted by the issue of heterosidasticity. So there is issue of heterosidasticity, but it seems it is slightly moderate. In this video, we will discuss the issue of autocorrelation. Often financial market returns are severely correlated and that leads to issue of autocorrelation in error terms. A very simple test is Durbin-Watson test, which is, you can see the more detail about Durbin-Watson test with this dwtest? command as we have been doing earlier.

You can read more about it. Let us implement this dwtest command, Durbin-Watson test. So while implementing dwtest command, we need to supply it with the fitted object, which is SLR, we call the object SLR. If I run this command, notice that we get a true autocorrelation p-value as very high p-value, which means it is not significant and dwstatistic is close to 2. So we are not able to reject the null based on this p-value. That means, as per this test, we have zero autocorrelation. Now please note, if dwvalue statistic is close to 2, then there is no correlation. If it is close to 0, then there is a positive correlation. If it is much more than 2, then there is a negative autocorrelation. Here it seems it is very close to 2, which is also indicated by p-value. That means we are not able to reject the null that there is no autocorrelation.

However, there is another very interesting test which is Brugge-Gottfried test, for which you can see more details here with the BG test. So you can run the BG test and question mark and see all the details here. With Brugge-Gottfried test, let us see the issue of autocorrelation and how it affects our model. So we need to simply type bg test command and supply the model. By default, it will consider only 1 lag of error.

So with 1 lag notice, there is no autocorrelation as per p-value that the BG test statistic, which is essentially the LM test and notice the p-value is not significant. That means we are not able to reject the null. However, please remember many times the correlation is not because of the first lag, it can be also because of lags with the later periods, maybe second or third lag. To account for that one can simply put the model with higher lag, lag terms.

For example, I can specify the order as 4 and see if the error term we have. So we are able to catch some serial correlation with 4 lags. That means there is some autocorrelation or serial correlation with higher order lag terms. To resolve this issue, one simple solution with the model is to add lags of returns in such model when there is serial correlation, a very, in financial markets a very simple and easily available solution is to add the lags of returns, in this case maybe lags of market returns or lags of ABC returns and see whether then issue of serial correlation is resolved. To summarize, in this video, we conducted the Durbin-Watson test and BG test of autocorrelation. It seems with DW test and first lag Bruges-Gortree-BG test, we are not able to find any serial correlation, but when we examine the higher order lags with BG test, we find some evidence of serial autocorrelation. In practical situations to account for such serial autocorrelation, one can add lag of returns such as ABC stock returns or maybe market nifty returns lags to see the serial autocorrelation issue is resolved.

In this video, we will discuss the application of robust standard errors. Recall our previous discussion on heterosid-stg and autocorrelation. These issues afflict standard errors and standard errors go into the model t values with coefficient divided by standard errors which result in t values under or over, so under or overvalued standard errors can affect the estimation by resulting in lower or higher t values. One solution that has been developed is to create robust standard errors that account for issues such as heterosid-stg and autocorrelation. Now, with R, there are various flavors of these robust standard errors. We will discuss the four most prominent ones often employed in computation of standard error, the routines that are often employed. And please note, it will not affect the coefficient estimate, but it will provide us with the robust standard errors that will go into the computation of t values that are arguably robust against issues such as heterosid-stg and autocorrelation.

So one way to compute is to use HCCM. So we will simply use our fitted object SLR and then we will supply with the variance covariance matrix, which is HCCM. If you want to know what HCCM is, you can type question mark HCCM and it provides you with the heterosid-stg corrected covariance matrices. So if you run this, you will find the coefficients, coefficient will remain same, but the standard errors will be changed and

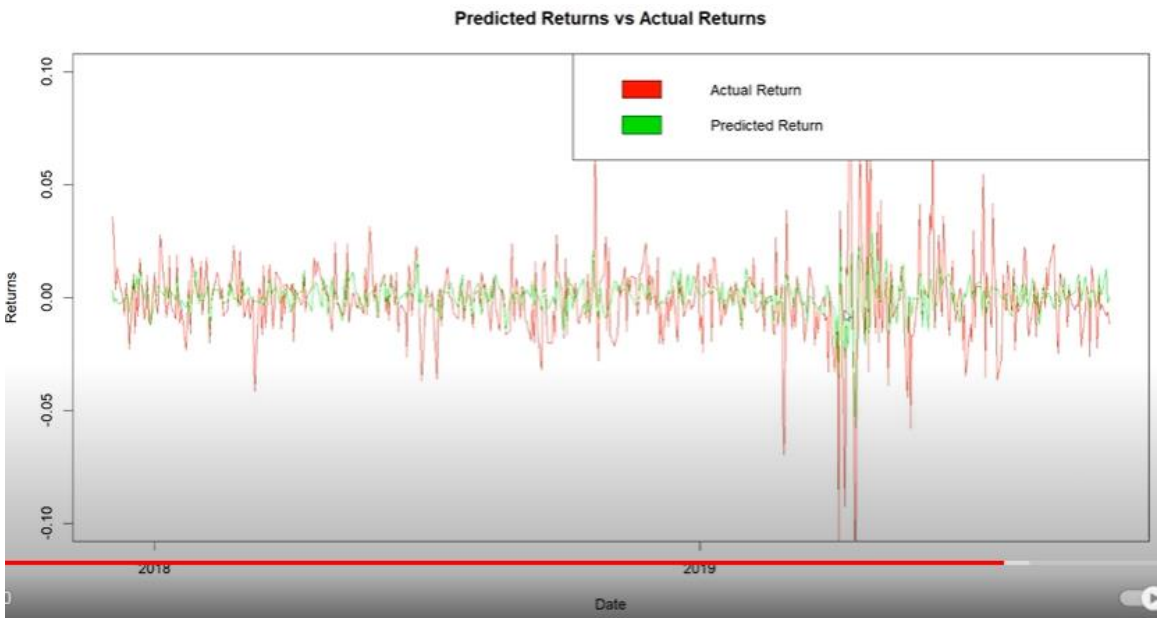
they will correct the t values for heterosid-stg corrected covariances. Similarly, we can also correct for heterosid-stg and autocorrelation correction through VCOVHAC.

This is for heterosid-stg and autocorrelation. These are part of sandwich matrix. So you will generate another coefficient matrix with its t values and p values. You can also correct simply for heterosid-stg depending upon the nature of whichever issue is affecting the model more, you can choose that.

So VCOVHC. This will correct for heterosid-stg only. And then there is new V wide correction. So lastly, we have new west wide correction. We simply type new west wide correction and if you run this, you will find the new west covariance matrix estimation, all the discussion is here. But for implementation, we will simply, we will use the same notation, only that instead of VCOVHAC, we will supply with the new west new west correction. And there we can obtain the corrected standard errors and the resulting t values.

So with this, in this manner, we can apply the robust standard errors. There are various flavors available. We can check more about them with this coeftest command, coeftest. We can check more about them, different variants, different flavors, how to apply all the, all the flavors are available. We can read more about this. The four most prominent we have discussed, hCCM, heterosid-stg corrected, various heterosid-stg and autocorrelation corrected, new west correction and so on. So we can apply them to observe if there are significant changes in t values from our original values.

It seems still applying all the robust standard errors, our coefficient is still very, very significant. To summarize, in this video, we discussed the application of robust standard errors in correcting for issues such as heterosid-stg and autocorrelation. We discussed four very prominent flavors of variance covariance matrices that is hCCM, VCOVHAC, VCOVHC and new west routines for correcting the standard errors in the model. Now that we have trained our simple linear regression algorithm and tested it for various econometric issues, we will try to do some prediction using the test data. So we will do some prediction using the test data. As a first step, we will create a prediction object, let us call it predict, P-R-E-D and we will assign the predicted values, predicted values, we will use our fitted object SLR and we will supply the new data which is test data and assign the values to our predict object.



As a first step, let us visualize this along with the actual test data. So we will plot, along with the date, we will plot our actual returns that are there in the test data. So this is test data, ABC returns, we will be plotting them. So on x axis, we have date and on y axis, we will have ABC returns.

Or rather let us call returns. We'll use central object as predicted returns versus actual returns. Now we would like to restrict our graph for y limit. We would like to visualize from minus 0.2 to plus 0.2, which is a reasonable limit to visualize daily data.

You would not expect returns to vary too much from 20 percent, more than 20 percent. Then PCH 20 will ensure that our objects are plotted as solid dots. We want them to plot in red colors. Also, let us specify the type as line L, type L. So there will be a solid line in fact. Now, along with BC returns, we would also like to have our predicted object plotted along the date.

So again, on the x axis, we have date. And on y axis, we would like our predict object line width of one. Let us specify the line width of one here also. Although it's not necessarily needed, but just for information, you can change it to two or any other number if you want it more or different the width to be higher or something. So first, let us plot this.

Let us plot using this command. Let's see if we are given a let us plot using this command. Let's see if we are getting the object right or we are getting some error. So, so this object has worked. So this is the currently the in red, we have the actual returns in the test data. We'll add some lines, these lines.

We had some lines with this command, so we will run this lines command and these green ones are the predicted ones. This is our predict PRED object, predict object. Now, to make it more aesthetically clear, I'll also add the legend here.

So legend. Let's put the legend on the top right, I see a lot of empty space, so I'll put the legend on the top right. And notice, you can see that in our window, minus 0.2 to plus 0.2 sufficient amount of gap is there. So if I would have put it minus one to plus one or some higher number, it would have looked not so aesthetically good.

So right now it is decent. The windows decent. In fact, I can put it as minus 0.1 also. That would be okay as well. So if I run it for minus 0.1 to 0.1, that also looks better. Now I'll add the legend on the top right part of it.

So we have actual return. And we have predicted return. And we'd like to fill the, provide the fill. So here, the actual return is in red, as we noted earlier, and predicted one is in green. So I'll add the green here. And then we'll run our plot. And we have the complete diagram here. Let's examine the full diagram. Notice, it seems that where the green one is reasonably capturing, not in terms of amplitude, but in terms of fluctuation, where the fluctuations are higher in the red ones, fluctuations are also higher in the green ones.

So there is reasonable similarity in the movement. However, we need more statistical evidence to conclude that. And for that, we will use our correlation command, we'll use the core command to test whether our predicted returns and actually we see this.

So this is actually we see return. And this is our pre object. Let's examine the correlation. And the correlation is reasonably high. So this is 0.43. That's almost 43.78%. So we are able to capture the dynamics of the ABC returns to the extent of 43.78% using our regression simple linear regression algorithm. So to summarize this video, we created a predict object, then we plotted this predicted object along with our actual returns contained in the test data that we created. The dynamics appear to be reasonably similar, that means we are able to capture a lot of dynamics of actual return in our predicted object.

We statistically examined this with our correlation measure, we check the correlation between actual return and predicted returns, there's almost 43.8% correlation between these two entities. So it seems that our prediction algorithm has done a reasonably good job. In daily data context, this is a very high level of prediction efficiency. In this video, we'll discuss the computation and examination of out of sample fit or accuracy of our

prediction algorithm. Please note that you need some kind of cost or error function, cost or error function to compare the model across to compare across various competing models or algorithms.

Now there are various cost functions and error functions that can be employed, starting from very simple models such as MSE or RMSE errors to more complex functions. Now let us examine how to execute this in R and for that we need library matrix which provides us with a number of such error functions to compute. For all, we'll start, let's start with this MSE.

MSE computes the average squared difference between two numeric vectors. So if you want to compute MSE with matrix library, you can simply type MSE test $abc. So this is our actual return object. And along with the predicted object and you can compute compute the MSE measure. Now on its own on a standalone basis, it does not have any meaning you need some other competing model and you would like to compare this MSE that is the average squared difference between the vectors to compare. Another variant of this error could be RMSE. RMSE computes the root mean squared error between two vectors like we have here two returns, ABC return and actually BC returns in the test data and its predicted values.

Also, if you want to read more or know more about these measures, you have to simply type question mark RMSE and you'll get most of the information here. You can check on the internet also, but most of the information relevant here can be obtained from the help window. Now to implement that RMSE in R you simply need to type RMSE and then test $abc. And then you have predicted object you can compute this measure and then you can compare this measure with other competing algorithms or models. If you have you may have various other competing models, maybe multiple linear regression or some other variables you want to test and then you can put it against this value of RMSE.

Then another important measure you have RAE. RAE computes the relative absolute error between two numeric vectors. So again, to compute the RAE you have to just type RAE. And please remember these are cost or error functions that help you understand the out of sample fit. Why we are saying out of sample? Because we are using it on test data, not on the data where we print the algorithm. So again, we'll type test $abc, which is our actual data along with the predicted object, and we can compute this measure of error as well.

On their own standalone basis, they do not make much sense. You need to have a competing model or competing algorithm to compare with. Then you have MAE error

also. MAE computes the average absolute difference between two numeric vectors. Again, the computation method is same. The implementation is same. We use test $abc along with the predicted object and you get the number here in the output in the console window.

Similarly, you have MAPE, another very important and famous measure of error, which is MAPE computes the average absolute percent difference between two numeric vectors. If you want to compute the MAPE, you can simply type MAPE test $abc and you have predict object.

Thus, you can compute MAPE also. Then there are a number of measures. For example, you have S-MAPE. S-MAPE is basically, it computes the symmetric mean absolute percentage error between two vectors. So you can compute S-MAPE for test $abc data with predict data, you get that number.

In a similar manner, there are other measures. We can compute them. For example, you have MSLE, MSLE computes the average of squared log error between two numeric vectors. The computation method remains the same. Then you have RMSLE. RMSLE computes the root mean squared log error between two vectors.

Similarly, you have bias measure. Bias measure computes the average amount that is actual greater than predicted. You can compute bias in the percentage form also. You have RSC measure. RSC measure computes relative squared error between two vectors. For example, you can simply type RSC test $abc along with predict object.

The syntax remains the same and you can get the RSC object. You can also have RRC, which is the root relative squared between the two vectors and so on. There are many measures. These were the most important ones. You can pick a number of measures and then compare this matrix along with other metrics. You can create more complex objects. For example, you can give different weights to these different errors and then create a new object as per your requirement or some more complex expression.

You can create out of all these error objects and create a more complex error object to compare against computing models. So there are different flavors to it available with you. You can use them to check the out of sample accuracy of your model. To summarize this video, we discussed a number of error or cost measures to examine the out of sample forecasting or prediction accuracy of our trained algorithm. We also saw their implementation in R and we noted that we can create more complex forms of error or cost functions by assigning weights or some complex expressions to these readymade readily available error or cost functions.

In this video, we will start with multiple linear regression model. Most of the syntax to implement and execute the model will remain same. So let us call it MLR, multiple linear regression model. And as we did with the simple linear regression model, we will use LM object, ABC returns and let us add sensex returns, basis sensex, sentiment measure since this is a multiple linear regression model. We will be using a number of variables on the dependent side, nifty, dividend announce, dividend and we will use our train data.

Set to train the model. So now that we have put the expression, we have provided the expression, let us run it. Our object, MLR object saves the entire output. So let us print the output summary object MLR. Let us call it model. Again, we are using the same naming convention. Let us see this output. Let us examine the output of this model.

Notice the coefficient of sensex and nifty, both are broad market indices while sensex is heavily significant at three star that is significant even at 1% level. Nifty is turning out to be negative. Now this is slightly surprising for us because we already saw that nifty on with simple linear regression model that nifty also has a positive impact. Now this and here nifty is not significant. This suggests that there is some possibility of issue of multicollinearity as we will examine in the next video because sensex and nifty both are market wide variable.

So this provides us intuition of multicollinearity issue that because of that the coefficient and significance estimation of nifty is getting affected. We will examine that later. Also notice that sentiment variable is turning out to be a very significant, very very significant even at 1% level that means 99% confidence. Dividend announced is also significant. It is turning out to be at 95% confidence or 5% significance level as we can see here significance quotes.

So dividend announced is also very significant. Notice that just a dash square, the power of model is very high as compared to the around 10% that we saw with the simple linear regression model. This is for the fact that there are number of significant variables that have been added. So the overall power of model to explain the variation in returns is increased considerably to around 27.

84%. However, we still need to observe the issue of multicollinearity here to resolve that. To summarize in this video, we trained our multiple linear regression algorithm using the train data. We summarized the output and briefly reviewed the output. In the next video, we will jump to the resolving multicollinearity issue as we discussed in this video. We will skip residual diagnostic, issue of heterostasticity, autocorrelation and

robust standard errors because they are exactly identical to what we have already seen in the simple linear regression model.

Same commands with same notations will be employed and the examination will be done in an identical manner. So in the interest of time and brevity, we will skip that part. We can directly apply the same codes here, same exactly identical codes can be applied for the multiple linear regression model to obtain and examine the residual diagnostic, heterostasticity, autocorrelation and robust standard errors related discussion. For more detail, we can directly look to the video topics on heterostasticity, autocorrelation, residual diagnostics and robust standard errors video topics to get more information on this. In continuation with our last video, we noted that the issue of multicollinearity may afflict our estimation process and therefore in this video, we will examine the issue of multicollinearity in more detail. Please recall, we said that there are, there is a possibility that independent, some of the independent variables may be correlated and therefore that may give rise to the issue of multicollinearity.

So as a preliminary evidence or examination, we will try and compute the correlation in the train dataset for the independent variables. Our independent variables include ABC, returns, sensex and we will see which of these variables are heavily correlated. These are all independent variables that we employed in the multiple regression model.

These are the main variables. We are not, we are leaving the dividend nouns variable, which is a dummy variable. So chances of this variable being correlated are very low. This is, dividend nouns are just ones and zeros. So notice the correlation matrix that we have produced here in console and notice the very high correlation. So between nifty and sensex, the correlation is almost 80%, which is a clear indication that there is a possibility of multicollinearity because of this high correlation between sensex and nifty. All the other variables are though correlated, but the levels are even lower than 50%, much lower than 50%.

So they may not have much multicollinearity related effect. So let's also recall that we created our model object MLR in which we trained algorithm. So we'll use this, what is called VIF, variance inflation factor, what we call variance inflation factor VIF, which measures the extent of multicollinearity in a model. So we'll use VIF and MLR. If this number and it indicates variance inflation factor for different variables in the model, if the number is around one, then multicollinearity is almost negligible.

If it is higher than two, then it is moderate. If it is higher than five, then it is really, really high. Notice the multicollinearity with sensex and nifty variable and both of them are heavily correlated is 2.9, which is on the higher side, which indicates that we have to

drop one variable to remove the multicollinearity issue in the model. Now that we have seen our model contains the show multicollinearity because of the variable market variables sensex and nifty, we would like to remove one of these variables to improve our estimation.

And in the interest of that we'll again run train our model on the train data. But this time we'll remove one of the market variables and run our regression ABC on sensex. So we are removing the nifty. So we are only running on sensex sentiment plus dividend announced and we'll again use our train data. Let us run this model. And let's summarize this into a model object.

So we'll create a model object in which we'll put our summary of the model, we'll print this model object. And again, the adjusted R square numbers are same, but notice the sensex has much higher significance as compared to earlier.

The beta which is the coefficient of sensex is around 0.55. Sentiment variable is also very significant. And dividend announced is very significant. To interpret the sensex variable, for example, 1% increase in market or sensex leads to 0.55% increase in the ABC returns. Also, notice the dividend announced variable to interpret since it is a dummy variable.

On average, there is a difference in return when the dividends are now there is increase in return of about 0.0038% on that dividend announcement date. And the model is able to explain around 27.85% of the variation in the ABC returns. So we have improved our model and accounted for the issue of multicollinearity. To summarize this video, we noted that when we computed the correlation matrix, we found that the two market variables that is nifty and sensex were heavily correlated with a correlation number of almost 80% and indicated the presence of chances of multicollinearity which we further examined with our variance inflation factor, which is VIF. And VIF suggested that indeed sensex and nifty variables are affected by the issue of multicollinearity. So we removed the nifty variable and then again trained our model and the model output is reviewed in the discussion.

In the next video, we will create a predict object and we will compare the predicted and actual test value and we try to visualize them to see our out of sample forecasting efficiency. In this video, we will create a prediction object using our test data with our trained algorithm, multiple linear regression algorithm.

So we will create a prediction object with test data. The procedure is very similar to

what we did with simple linear regression model. We will create a predict object. We will assign the values that are predicted using MLR. This is MLR object that we have trained, already trained and in the new data we will assign the test data set. So we are assigning the prediction object, creating a prediction object with the test data. Now that we have created this object, we will try and compute the correlation between this predicted object and our actual ABC returns.

So we will use this predict object and we will compute the correlation between predict and test and there is a 57 percent correlation. Let us also try and visualize, make a comparison using visualization. So we will first plot our actual returns. We have already seen these commands with the simple linear regression model, but still we will run them again.

So test$date, test$abc. So with this xlab again, the same syntax date, ylab as returns, we will be plotting actual and predicted returns. So the central heading again here is actual versus predicted returns. Again, we would like to see the ylim here.

So we would like to set the y limits from minus 20 percent, which is a reasonable limit to plus 0.20. We would like to plot red color line as actual ABC returns in the test data and type as line. So this is our line type. We'll run this later with the complete thing. Next, we would like to have lines command with this, with the help of lines command, we'll add our predicted object.

So let's call it predicted object that we have just created and color will give it green. Again, the type will assign a type of line. So we are using line type. Other option is to plot in terms of points for this kind of graph.

Line type is more preferable to make a better comparison. Again, we'll add legend. So like we did last time, we'll use legend on the top right side. On the top right side, we have legend and we have actual returns. First, then we have predicted return.

We'll assign colors. So fill equal to we have red as actual and green as predicted, as we have noted. So now let's do some plotting here. So I'll enlarge the plotting window for better visualization.

First, I'll run the main plot command. So these are my actual returns in red. Then I'll add the lines. So lines command, I'll add the predicted returns. Then I'll add the lines command. And then I'll add the legend. Now let's zoom this graph and notice very nicely our object is capturing as we also saw in the correlation, there was a very high

correlation and the same is reflected here. Our predicted object in green is very nicely capturing the fluctuations in red actual returns, ABC returns.

So that means our multiple linear regression algorithm has done a very nice job of predicting the returns. And that is that was also reflected in our very heavy correlation of 57.7% between actual and predicted returns. To summarize, in this video, we created a predict object using our test data. In this predicted object, we computed the correlation with the actual test data, the correlation is 57.7%, which was very high. We also plotted and visualize the actual return and superimpose them with the predicted returns, which again confirmed that our prediction has been, the algorithm has been very efficient in predicting the objects.

Now, here we are not doing the out of sample forecasting errors that we did for simple linear regression model. We can go and review the video on out of sample regression forecasting video topic. In that video topic on out of sample error forecasting, we have discussed in great detail the R implementation and the interpretation of error functions. So, in the interest of time and brevity, we will not repeat there.

The command, execution and discussion remains identical. So, we can refer to that video topic on out of sample forecasting. To summarize this lesson, we model ABC stock prices using simple regression problem with market index variable. The model is trained using trained dataset and various goodness of fit measures are examined. The fitted model is examined visually as well. The model is tested using test dataset and various measures of out of sample fit are also examined.

Next, a multiple linear regression model is trained using trained dataset on multiple variables. The fitted model is visually examined and also various goodness of fit measures are examined as well. The model is evaluated on various issues related to multicollinearity, heterosoxicity and autocorrelation. Lastly, the model is examined on various parameters for its out of sample fit performance. Thank you.