In our previous video, we have understood the in-depth maths intuition behind linear regression and we found out that okay, we really need to find out the best fit line along with that. We also saw what is the equation of the best fit line. Instead of x is equal to theta zero plus theta one multiplied by x. This x is just my single independent feature. Suppose if I have many independent feature, it becomes a multiple linear regression. And with respect to this, this was the cost function that we use, right? The cost function was one by two. And summation of I is equal to one to m s of x of I minus y of I whole square. This s of x of I is basically my predicted point, and this is basically called as mean squared error. This is what is the cost function mean squared error. And we also found out that whenever we try to plot our coefficient with j of theta we are able to see something called as gradient descent. Our main aim is basically to come to this global minima. Okay. Now now we are going to understand some series of algorithm which is called as Ridge regression, lasso regression, elastic net regression. And we'll try to understand why do we use ridge regression, Lasso regression and elastic regression. In this video we are going to discuss just about ridge regression. And in the upcoming videos we'll cover all the other algorithms. Now let's start with something called as ridge regression. So here I'm just going to write it down which is called as ridge regression. Now, let's say, guys, uh, I, I have a single independent feature. Let's say I have something like x and Y, okay. And if I consider that I just have two specific data points in my training data set. Now, in this particular scenario, obviously if I apply linear regression, this is going to pass through the best fit line. So when it is passing through, uh, when I'm actually creating this best fit line, it passes through both this particular points. Right. So this is my best fit line. Now when this is my best fit line, obviously you can see that the error is zero. Right. Because there is no such difference right between the predicted and the real point. So in this particular scenario I can definitely say this model is overfitting. Now what does model overfit basically mean? Okay. Now with respect to this training data you can see that the accuracy is very high. The error is very very less right or error is almost zero. Right. But if I try to add some more new test data, let's say these are my new test data that I'm just going to add over here. Now when this new test data is basically getting added, you can see that now the error basically increases, right. The error with respect to this new test data is basically increasing. So this is basically a problem of overfitting. So here what is actually happening here. Overfitting is actually happening. Now what does overfitting basically mean. Because I have already covered this let's say with respect to our train data, my accuracy is very, very high. Let's say my accuracy is 99%, 100%. Okay. And whenever I talk about train data, at that point of time, we use something called as bias. In this scenario I will be saying low bias okay. Similarly with respect to the test data, when my accuracy is actually low, see, for the train data the accuracy is high. But for the test data over here the accuracy is low. And in this scenario, whenever we use test data here, I'm actually going to use something called as variance. And in this scenario it will be high variance okay. So this is the problem that is basically happening right. It is overfitting in this scenario right now. How do I reduce this overfitting. I should definitely find out a way to reduce this overfitting. And for that specific scenario we use called as ridge regression. Because in this scenario I know my best fit line is passing through both the points. Okay. And always remember guys, if you get an accuracy of 100% right, definitely consider it that the model is overfitting. You should never get the training accuracy as 100%. Just imagine this. That basically means the model has been trained very well on the training data right now. Let's understand what exactly ridge regression is all about. Okay? Ridge regression is also called as something called as L2 regularization L2 regularization. And this is used to reduce this overfitting. Suppose my linear regression creates some overfitting. Then in order to reduce that overfitting in the linear regression we use ridge regression okay. So we can just consider ridge regression as a new algorithm, which will actually help us to

hyperparameter tune the linear regression. Okay. Now let's go ahead and let's understand how does the cost function with respect to ridge regression look like. So here I'm just going to write down the cost function. Initially. If I consider the cost function of linear regression I hope everybody knows one by 2 a.m. summation of I is equal to one to m h theta of x of I minus y of I whole square. Right. So this is the cost function of the linear regression. Now in this particular case see when I create this best fit line it automatically, you know passes through both the points both the points in the my training data. Now when this is the scenario obviously my entire cost function, this will become zero. Right now when it is becoming zero. That basically means my model is overfitted because it is passing through all the points perfectly. Now in order to, you know, remove or add some parameters that we have to make sure that this never becomes zero. So what we do is that in ridge regression we add two parameters. One is lambda. And the second one that we try to add is summation of I is equal to one to m slope square okay. So this lambda that you see is a kind of hyper parameter. Okay. And I'll try to show you the relationship between Lambda and Slope and how this is basically reducing the overfitting that also will try to understand. Okay. Now see over here, if I'm trying to add lambda and slope square let's say lambda value is one okay. Now when this value becomes zero I have to make sure that this never becomes zero. Because if this becomes zero, that basically means this best fit line perfectly passes through all the points. And this should not happen. So my cost function should definitely know that this is not zero at this point of time. So what we do, we do we penalize this value by multiplying one lambda value. Lambda value I've just initialized as one okay. And then I try to show you the relationship between lambda and slope. So after this one let's say my slope my in this particular case my h theta of x is nothing but theta zero plus theta 1X1. Okay. Let's say I just have one input features. So summation of I is equal to one to m basically means how many number of coefficients or slopes are there. We are just going to do the summation of whole square like that. So here I'm just going to do theta one whole square. Now let's consider theta one any positive or negative value okay. So that basically means we are never going to get zero. We will be getting greater than zero right. In this particular scenario. What will happen is that my model training the best fit line will just not get stopping, will not just stop over here, then it will try to find out another best fit line like this, or it will try to find out another best fit line like this such that my cost function is minimal. Okay, so in this specific way, when I am specifically adding this two parameters, that is lambda and slope square, the situation is going to be in such a way that we are never going to get a best fit line that passes exactly through all the training data points. Okay, so this is how ridge regression is making sure that the overfitting will not happen. But one very important interview question that may come. What is the relationship between lambda and slope square okay. And for this let's draw that same curve that we discussed. You know with respect to the simple linear regression. And this specific curve is with respect to theta. That is my slope and cost function J of theta. Now in this particular case, if you know that we will be getting this kind of gradient descent right, let's say I'm just going to put some points over here like point, uh let's say this is zero. This is point two. This is point four. This is point six. This is point eight. This is 1.0 and this is minus point two. Okay. Let's say this I have just plotted with respect to theta and the cost function okay. With respect to this gradient descent. Now just understand when lambda is equal to zero. See when lambda is equal to zero. That basically means what I will just have the same linear regression cost function okay. If lambda is equal to zero this entire value will be zero. So that basically means I'm not applying ridge regression over here. Whenever I apply ridge regression I'm basically using this equation at the end. Right. So when lambda is equal to zero this I'm just having this linear regression over here the same cost function of the linear regression. This is fine. So this is basically my global minima right I hope this is my global minima. So let let me say that okay. This is my global

minima okay. And this is when when your lambda is equal to zero. Now let's say I increase my lambda to ten. Now what will happen is that I will get a new line which will look something like this. Okay. If you if you probably try to plot theta, I'm just trying to show you how the plotting, uh, how how the graph will look like when we increase our lambda. You can definitely check out by playing with different, different values. But once you increase the lambda value with respect to this theta, right, what will happen is that now this lambda value is basically getting assigned some value over here that is ten. And when we multiply with the slope square I will be getting a new gradient descent. And now you will be able to see that my global minima has changed. When my global minima has changed. So let me just draw this line properly so that you will be able to understand. So let's say my lambda is equal to ten. So what will happen is that when the lambda value increases I will be getting another line which looks like this. So during when lambda is equal to ten I'll be getting a different gradient descent. And now what is the difference between this gradient descent and this gradient descent? Here you can definitely see that my global minima has shifted. It has shifted from here to here right. Similarly my theta value has got reduced okay. It has basically shifted from here to here. Right. This particular is with respect to this theta value. Now similarly if I keep on increasing my lambda value let's say I increase my lambda value again. So with respect to lambda is equal to uh let's say 30. Then I'll be getting another line which will look something like this. And now this will again get shifted. This, this global minima will get again shifted somewhere here. So in short what is happening. Theta values again getting reduced. And like this. It will keep on happening. But always remember in this particular scenario it will never be zero, okay? It will never, never be zero as I keep on increasing the lambda value. So lambda over here can be considered as a hyper parameter. Okay. So lambda can be basically considered as a hyper parameter. And if you really want to know the relationship between lambda and slope, you can definitely say when lambda is increasing, your slope is decreasing. How slope is decreasing because we are going on this left hand side right. You can see from point six it became .4.2 like this. It is going on the left hand side. So this is the exact relationship between lambda and slope. But why we are trying to understand this particular relation. Let's say I have a multi linear regression okay. Let's say I have a multi linear regression which is like this theta zero plus theta 1X1 plus theta 2X2 plus theta 3X3. Let's say I have three coefficients and three independent feature. And let's consider that theta zero is equal to zero. If I say theta zero is equal to zero then what does this basically indicate. This basically indicate that it is passing through the origin right. The specific line. This line is basically meeting the y axis in the origin. This I have already discussed. Now let's consider that my theta one value I'm assuming okay I got somewhere around 0.5 to x one. I'm just assuming theta one, theta two, theta three. Let's say over here I got probably 0.48. And here I got probably point uh two for x three. Right. Suppose I got this values. And obviously I have theta zero as zero. If I don't want to even take this let's consider some different theta zero value. So let's say this is 0.3 for my intercept is .34. Now in this particular scenario what will happen is that if I apply ridge regression on this cost function then this value will get reduced, this coefficient will get reduced, but it will never be zero. See what does this value basically indicate? .52 of x one. This basically indicates that with the unit movement in x one right with the unit movement in x one, what is the movement with y right. So here you can see 0.52 movement with respect to y. Similarly with the unit movement with x two what is the unit movement with respect to y. So here you can see that it is 0.48 unit movement with respect to Y. And that's a unit movement. But 0.48 movement with respect to y. That basically means if x one is moving by one, then y will move by 0.52. In this particular case, if x two is moving by one, then 0.48 y will basically be moving right. So this is what it basically indicates. Now if there is too much movement with respect to unit movement, if there is a unit movement, that basically means this

features are highly correlated with the output feature, right. Highly correlated with the output feature. But here you can see that with respect to x three we have a very small value .24. Right. Even though we have a very small value, what will happen is that if I try to make a movement unit movement with x three, there will be a small unit movement with respect to y, right? Somewhere around .24. But when I apply ridge regression to this equation, I may get something like this. See, I may get something like this where this value will get reduced, this value will get reduced. Let's say it will get reduced to point 40X1 it. This may be point 38X2. But when I see this value this value will also get reduced point 14X3. But here you can see that since this value has now a very small coefficient, this won't impact much change in the movement of this specific line. Okay. Because in this particular case I have three coefficients. So this will then become a plain. This may this may become a plain entirely right. The line that we are specifically using for division. Right. So here you can see over here, since this value is very very small, this won't impact impact the best fit line by a major movement. Only a smaller movement will basically happen. And always remember this value will never become zero. That is what ridge regression basically does and how it is reducing over fitting, because it is reducing the impact by the coefficient, by reducing the coefficient of the feature that are not directly related correlated with the output feature. Now here you can see x three is hardly correlated by .24. And it is trying to reduce this particular value also. And when it reduces this particular value, that basically means this will not have much impact on the best fit line. Right. So I hope you are able to understand. I hope you are able to understand the relationship between lambda and slope. So please, uh, just try to go through this video again. This all maths you really need to know guys. Then only you will be able to understand what things are actually happening. Now. In the next video we are basically going to discuss about something called as uh, lasso regression. So right now we have completed ridge regression. And we'll try to understand what is the difference between ridge and lasso. And then finally we'll go with elastic net. Thank you.

We have already understood about ridge regression, which is also called as L2 regularization. And a super important interview question may come that why do you use ridge regression? You basically have to say about, uh, it reduces. It is basically used to reduce overfitting. Now in this video we are going to discuss about lasso regression. And Lasso regression is also called as something called as L1 regularization. And why do we specifically use. We specifically use for a very important. Work that is called as feature selection. Now how do we do feature selection on. Let's discuss about this. Now in Lasso regression we just need to focus more on the cost function. So here the cost function is nothing, but it is one by two m summation of I is equal to one to m. And here obviously you have something called as a sort of a state of x of I, which is my predicted point y of I. Along with this, now we are going to add two more parameters. One is lambda and the second one is something called as summation of I is equal to one to n, not slope square, but it should be magnitude of slope. Now, with the help of this specific equation, we will try to do some feature selection. Feature selection basically means the features that are not that important will automatically get deleted, and features that are very, very important it will be considered. Now, if I try to show you the relationship between lambda and this slope, okay, with respect to this specific cost function. So suppose if I try to draw this cost function over here in the right hand side with respect to theta and j of theta by using this cost function, obviously the first one, if your lambda value is zero, I will be getting a normal curve right. Normal gradient descent curve. So let's say this is my zero or. Let's say this is my I I'm just going to use

this over here zero. This is minus point two. And this is .2.4.6.81.0 okay. So if I try to draw this kind of curve, this curve is called as gradient descent with lambda is equal to zero. I'll be getting this kind of curve right. And this is obviously my global minima. Now what happens when my lambda value will increase. Then I will probably get my another curve which will look like this. Okay. And let me just draw this and then everything will make sense okay. And then I may probably also get one more curve which may look like this. And finally I may get one more curve which will look like this, which will look like this. Okay. Now what is the difference between this curve and the previous curve when my lambda value is increasing? Let's say this green line is basically when lambda is equal to ten. This is lambda is equal to 20. This is lambda is equal to 40. Let's see okay. Now in this scenario what happens is that obviously, uh, as my I increase my lambda my lap, my theta value is decreasing. But you will be seeing that after one point of time, my theta value will become zero. Okay. So this is nothing, but this is basically becoming zero. So in short my coefficient is actually becoming zero. Now what happens if my coefficient actually becomes zero? In short we are actually trying to remove that specific feature. Right. So let's say my h of x is this specific equation theta zero plus theta 1X1 plus theta 2X2 plus theta 3X3 plus theta 4X4. Let's consider that I have four independent features. Let's say my theta zero is 0.5 to my theta one that I have probably computed by creating the best fit line is somewhere around 0.6 5X1. This is 0.7 2X2 plus. This is 0.0 34X3. And this this is 0.1 to x four. Now in this particular scenario you can see that x four is the feature that is not at that is not much correlated with the output feature, that is with respect to h theta of x. Because here y we have a very small coefficient that is 0.12. So after we apply over here lasso regression. Since this feature is not that important, what it will do is that it will try to reduce this entire it will try to reduce this coefficient to something like zero. Right. So zero multiplied by x four will actually become zero. So this will entirely become zero. And this all value will get reduced by a smaller number phi 1X1. Because this is a this is directly correlated. Right. So this may be x two. This may be 0.1 4X3. Like this. So in this way you can see that this feature is being removed. And we have not used this particular feature. Or this coefficient has become zero because this feature is not that important. Right. So this is what basically indicates once we add this specific term in the cost function. Right. Once we add this specific term in the cost function, we can see that. What is the relationship between lambda and slope. So here obviously your theta value is becoming less. But at one point of time this will become zero. That is your theta value will become zero. So the same thing is basically happening over here, which all features are not that much correlated. And if it is not correlated then obviously it will be having a small coefficient value. So after we apply lasso regression this will actually become zero. And by this this entire feature is getting removed and remaining all features is basically used to find out the best fit line. Right. So this is the importance of lasso regression. Right. And when. We do, we use this. Suppose if I have hundreds and hundreds of features, right. I should definitely go ahead and use lasso regression, because automatically it will be able to find out features that are not that highly correlated. It will just try to remove it because it is going to make the coefficient to zero, right? Similarly, when should we use ridge regression? It is very simple. Whenever overfitting condition basically happens when your training data accuracy is very, very high and your test data accuracy is very, very low. This basically is an overfitting condition. So we should basically try to apply a ridge regression in this particular case right now coming to the next one which is super super important. What is elastic net right. Elastic net is nothing guys. It is the combination of both. So let me write this with respect to elastic net. It is a combination of both ridge and lasso. So here with the help of elastic net what all things we do. First of all we try to reduce overfitting. And the second one, uh, is that we try to, uh, along with overfitting what we do, we also do feature selection. So if I combine both ridge and lasso together, that actually becomes a elastic

net. So what will be the cost function that may look like over here now. So I may basically write my cost function which will look like this one by two m summation of I is equal to one to m, and this will be y of I minus y of. I will not say y of I, but I'll say h of x of I okay minus y of I whole square. Uh plus and we can basically use for reducing overfitting. What do we use. We use lambda one and this lambda one. We can basically multiply with summation of I is equal to one to n. And this will be my slope square okay. And plus lambda two summation of I is equal to one to I should not write n. Instead I can also write m okay because m basically indicates all the slope parts. Okay, so this will be slope. So what is basically happening. This reduces overfitting. Okay. And this reduces, uh, this actually helps us to perform feature selection. So this is for reducing overfitting. And this is for feature selection. Right. So by this we are trying to solve both the problems. Suppose if I have a model that is overfitting and it also have a lot of features, then I can directly use Elasticnet regression. Now why do we learn all the series like ridge, lasso and elastic? In short, we are hyperparameter tuning the linear regression. Okay, so hyperparameter tuning the. Linear regression. Right. So we are basically hyperparameter tuning it. So that is the main reason why we are specifically using it. Okay. So yes, uh, this was the series with respect to the regression linear. Uh, whenever you're using linear regression, you also need to think that when you should use ridge regression, lasso regression, elastic regression. And I've seen many people asking different kind of interview questions with respect to this. Most important is what is the relationship between alpha and slope with respect to, uh, Ridge and Lasso that both have actually shown it to you and how it performs feature selection and how it performs, uh, you know, uh, how it reduces overfitting. Both has been discussed. So, yes. Uh, let's continue. Uh, and uh, in the next video we are going to learn many more algorithms going ahead.

In this video we are going to discuss about different types of cross-validation, and we'll also understand why cross-validation is actually used. Now, as you know that guys, you obviously have some data set. Let's consider that you have some data set over here and this specific data set initially whenever we train a machine learning model, first of all we divide this data set into two parts okay. The first part is something called as training data set. Right. And the second part, which we normally use or which we normally call is something called as test data set. Now two points I'm going to note it down over here. With the help of training data set we will be training our model. Right. So for training our model we will basically be using this training data set along with this. The second point that I really want to mention is that for hyper parameter tuning also we specifically use our training data set. Right now when I say how do I do a hyper parameter tuning? There are various ways like Gridsearchcv, Randomizedsearchcv and all. And you may be considering Chris, how training our model with the help of training, data set and even hyper parameter tuning over here. What we do is that we further divide this data into two more parts. And this is basically called as train and validation. Right. So we split this data set more into train and validation. Okay. And with the help of this validation data set, we basically, you know, try to. Hyperparameter tuned the model. So here I'm going to write it as validation data set. Okay so till here I hope everybody's clear. But once we divide from this train this training data set into train and validation, in short we are splitting this data set. So while splitting this uh, here specifically this training data set, my model will get trained and it will be validated with the help of this validation data set. And when we are validating it, in short, we are also hyperparameter tuning it. That basically means will play with multiple parameters. Now let's say initially I just have 1000

records okay. And let's consider that I'm using the 70% split and 30% split with as my train and test split. Uh, this data that is present will only be used by my model to check the performance. So to check. The performance of my model based on new data. Okay, based on new data, I will be using this test data. This test data will never be shown to the model at any point of time. Only after the model is trained completely validated completely, then only this new test data will be shown. And then we'll check the performance by seeing a lot of performance metrics let's say accuracy precision recall R square in case of R2 score. In the case of for regression problem statement mean absolute error, mean squared error. Many many different performance metrics which we have already discussed. Now let's go ahead and focus on this part where we split our data set from this training into train and validation. And for doing the split we can apply something called as cross validation. And why do we do this? Because understand, if I try to do a split off training into this two part, that is train and validation. There is an important parameter which is called as random state, and as we change this random state value, you'll be seeing that we will always be getting a separate training data and separate validation data. So let's say on one of the split I got 85% as my accuracy. In the other split, I may be getting 92% accuracy, or in other split I may also get my accuracy getting reduced like somewhere around 78 or 75. So this kind of scenario will always happen. And as a data scientist, we should be able to say that my model is probably giving an average accuracy of this much or of some specific value. And for that specific thing, we'll be using cross validation. Now there are different types of cross validation which we usually use. Or let's discuss about the first cross validation. The first cross validation uh, that I would like to discuss about is something called as leave one out. Cross validation, which we specifically say CV, and the short form is like, if I really want to use it, we will basically be writing it as L or CV. Okay, leave one out. Cross validation. Now let's understand this. In this cross validation what happens okay. You have to understand this word. Leave one out. Cross validation. So let's say currently in your training data. Now, when I say training data, I'm basically talking about this specific data, right. This part because this split is obviously required. We are never going to use the test data from this data. So if I go down this training data, let's say the number of records are 500. Okay, now further, I really need to split this data into training and validation data. So with the help of leave one out cross validation, what will happen is that we will be taking this 500 records and out of all these 500 records for the experiment, one when I say experiment one, that basically means let's say my cross validation. Um, right now I will not write cross validation. Let's say for my experiment one, the first record, one record, this will be put in my validation part, okay. And remaining will be my training data. Right. And my model will get trained on this training data and it will be validated by this validation data. So it is basically going to get some accuracy based on this particular validation data. Now similarly in experiment two. What will happen is that the next record that you will be having will be taken as your. Only one record, right? Only one record. This record will be your validation. Okay and remaining all will be your training data, right? This will also be your train. This will also be your train. Right. And then again you may get some different accuracy. Now similarly based on the data size, you will be seeing that we have to perform that many number of experiments because at the end of the day, my validation data size is only one right now. In this particular case, this will basically be my training and this will basically be my validation data. And again, this will be my accuracy 500 right now obviously what are the disadvantage with respect to this. This will be my experiment 500 because I have 500 records. Now obviously what is the disadvantage of this if you are just taking one out cross validation at every every experiment that we perform, we will be taking only one record as a validation data and remaining all will be a training data. Now obviously for performing this much as the data set size increases, the complexity of training the model also increases right complexity of training. The

model also increases because if I have 5000 records, let's say, then I have to probably perform 1000 experiments. I have to make sure that each and every record is passed, and I'll be able to then calculate the average of all this accuracy. So obviously this is not a very good technique. And you know that this technique is no way used right now. Uh, in some of the cases we may use it. Okay. Now the second major disadvantage over here is that. This model obviously leads to overfitting. Now what does overfitting mean? Overfitting basically means with respect to our training data, since we are taking the entire training data, my accuracy is very high right here. The accuracy is very high since my test data is very small. All are not set as data here as a validation data, right? Since the validation data is very small. Right then the obviously the accuracy will keep on decreasing. Right. And when we test this same model with our new test data. Then my model performance will go down, right? Model will not perform well. So the accuracy here will go down. So this is a major, major disadvantage. Uh, usually we say overfitting wherein your training accuracy is high and your validation accuracy is, uh, less. That basically means it is perfectly fitted, the training data. And obviously because the train data size is quite huge. Okay. Now, similarly, the second type, I can name it over here as. Leave P out. Cross validation okay. So here instead of one you set some p value. Your p value can be ten. Your p value can be 20. Your p value can be 30. And this can be selected as a hyper parameter okay. The same process. Everything is same. Okay. Now let's go ahead and understand the third important type of cross validation. And this cross validation is basically called as k fold cross validation. So I will go ahead and basically write k fold. Cross validation. Okay. Now let's see if your data set is size is 500. Okay. Let's say the total data set that you are present in the training is 500. Now from this training I need to split this into train and validation. And I will say let my key value be five. Okay. The total size of the data set which is mentioned as n is 500. Now what will happen is that we will try to first of all find out cross validation how many different experiments we need to perform now in order to calculate. I will just divide 500 by five and here you will be able to see 100. Okay, so this should not be cross validation but this should be your test size okay. And when you calculate the test size how I'm calculating the test size the total number of records divided by five. Let me write it again so that this will be a super important point for you all. This. I am trying to calculate my test size and this is nothing but 500 divided by five. And here you will be seeing 100 records. So this will be 100 records. So what will happen is that in the experiment one which we also say cross validation equal to one in the first experiment, the first hundred records, let's say this is my this is my hundred records. First hundred records. This will basically become my validation data. And this will actually be my training data okay. And similarly with respect to experiment two. What will happen the next 500 records that you will be seeing? The next 500 records that you'll be seeing. This will basically become your validation data, and remaining all over here will basically become your training data. So again here you'll be getting an accuracy one. Similarly here also you'll be getting an accuracy two. Now just imagine how many number of experiments I have to do in order to traverse all the 500 records. Obviously it is very simple. We will go ahead with experimental experiment. How many see five experiments? Because five into 100 will cover all the data points. Right? So experiment five you will be able to see. The last hundred records. Will basically be your. The validation data point. And your this data will basically be your training data points. Okay, perfect. And finally here you'll be having your accuracy fine. And then I can probably calculate the average of all the accuracies. And as a data scientist, I can say that the max accuracy that I can get is the maximum one out of it, the minimum out of it. And what is the average accuracy? I can basically say with the help of cross validation dataset. Okay. Very very simple. Uh, we have discussed till k fold cross validation here. You can see leave one out cross validation. Leave p out cross validation. Now let's go ahead and discuss about one more technique uh, which we uh specifically say it as stratified k

fold cross validation. Now before understanding why do we use stratified k fold cross validation. Let's go to the fourth one here I'm just going to write stratified. K fold. Cross validation. See, at the end of the day, cross validation is used for hyperparameter tuning along with to check, uh, my how my model performs with respect to the validation data set. Also in stratified k fold cross validation. First of all, we'll try to understand what is the problem with k fold. Let's say if we are solving a classification problem, there may be scenario in my test data or validation data. Since I'm selecting, you know, part by part, one by one, and trying to cover every data set, there may be high possibility that only one type of categories may come over here. Let's say if it is a binary classification problem. Then in my test data the scenario may be or in sorry. In my validation data, the scenario may be that all the outputs are ones or zeros. Right. We may not get the right combination of ones and zeros as my output. So that may be a problem because my data set is only giving one type of data. And obviously model will not be able to understand, you know, or it will not be able to train itself properly. Stratified k fold cross validation makes sure that whatever validation we are doing with respect to different, different k values, let's say if this is my k value is equal to five, obviously test size will be 100. Okay. Because uh, my 500 divided by five will be 100 over its test site. Right. Test data. Whenever it is taking the validation part. Whenever I say test data, I'm basically mentioning it as validation data. Okay, guys, don't get confused with respect to this, right? So this is nothing but validation data. Now when this validation data is selected okay. It will always make sure that the number of outputs will be evenly distributed over here. Let's say 6040 ratio. The number of outputs will be distributed, right? If I have 60 ones as my output or 40 zeros as my output. So this is also a balanced data set right. So in my validation data always it will make sure that equal amount of zeros and ones will be present in our training data set. We can have somewhat little bit in a different way. But it will always make sure that in our validation data set, it will probably provide you almost equal proportion. So let's say if this is my CV one, this is my experiment one then in the experiment two, it will try to select the next 100 records. And it will again make sure that my validation data set is proportion with respect to the number of outputs. Now this is an example with respect to binary classification. So again this will be my validation data set. This will be my train data set. This is the only difference between k fold and stratified k fold cross validation okay. At the end of the day, it is always making sure that the number of outputs in the validation data set is almost equal, so that my model will be able to perform remaining all whatever things are there in k fold. The same thing actually happens over here. Now coming to the fifth one, which is again super important. And uh, the fifth one is something related to time series data. So here I'm going to basically write time series. Cross validation. Now here what we are going to do. I think you have seen a lot of use cases where, let's say there is something called as product sentiment analysis or product comment sentiment analysis, like people usually write product reviews, right? I can basically write reviews. Okay. Sentiment analysis. Now let's say initially a product is created and later on company adds some more features in that specific product. So initially, let's say when your reviews are bad, you know, later on it can become good, right? So all those reviews that are coming that are those are based on time, right. And you can say as we go from January to December, right. Initially in the first quarter, we got bad comments for this product reviews or we got bad reviews for this product. Later on when we went to July, September, October, the reviews became better. So in this scenario, what happens is that in time series all the cross validation will happen based on days or based on some time. Let's say day one, day two. Day three. Day four. Right. And like this up to the end. Now in this we divide our data set based on days. Let's say this from day one to day four is the part of my training data set. Day four to day N is part of my validation data set. We cannot randomly pick up days and put it in the training data set. Our validation data set. We always have to make sure that whenever we are doing the split,

it happens based on days, right? The initial number of days, it can be any number of days, right? And this should only be the sequence. We cannot randomly pick up some part from here and put it over here. No, that will not be possible. And where do we apply all these scenarios? Usually in time series application you'll be seeing. We'll be doing a lot of projects with respect to time series. So in time series application we will be using this kind of time series cross validation okay. And this is the reason because reviews you know something that is some outcome that is related to time will always change. And that is the reason we always try to make this kind of split. Right. So this was the basic differences. And yes, as we go ahead, uh, we are going to understand all these types of cross validation. And we are going to implement with the help of Python and sklearn. And we'll see that how we can actually solve it. So yes, uh, this was it from my side. I will see you all in the next video. Thank you.