Week 11
Video Transcripts

Video 1 (6:42): Machine Learning - Rocks and Mines Dataset

We're going to learn a little bit about machine learning just to make things clear. This is a very very superficial introduction to machine learning, which is a very complex and deep topic. What we're going to do really is, focus on a few Python libraries that are available for machine learning, and actually a few parts of these libraries, and try to get a sense for what is it that is important when you're doing machine learning exercises. If you want to actually do some machine learning, then you should probably read up more on it.

As we go along, I'll point you to various resources and, but this should be a pretty, you know, quick, a pretty decent introduction. So, let's start by looking at machine learning, and our goal here is, so we look at 'regression', and our goal is to really walk through the steps in regression and see how we load the data, how we do the regression, and then how we can interpret the results so that it's not making sense to us. Because interpretation is really the key to understanding what the algorithm or the computer program has given to you. So, a general step is, of course, to read the data and then generate a few summary statistics and then try to understand what these things do. So, the data sets we're going to use for the machine learning exercise are mostly available online, and the first data set that we're going to focus on is the rocks versus mines data set.

This data set contains sonar soundings at different frequencies of a location underwater, so to speak, actually it's underwater, where there are rocks and there are mines, and the goal is to figure out whether we can learn - what is a rock and what is a mine from this...from this data. Obviously, the reason is that you don't want to accidentally step on a mine or swim into a mine. So, think of it like this, right, like, let's say this is, though this is underwater stuff but let's say, you're looking at a mine field in a war-torn... war-torn zone of some sort and you can see that there are rocks and there are mines and they look alike, they just look alike. But you send sonar sounds, soundwaves to the rocks and mines and you get different results back. So, the question now is that can we integrate these results to figure out which one is a rock, which one is a mine, and how—with what confidence can we walk through this minefield without blowing ourselves up, you know. That's the goal here, right! So, because the only real way to test it, of course, is to step on it, but that's not a good idea. So, that's—with that in mind, let's see how this analysis... this analysis will go, right! So, let's go to the data set first. The data set contains a bunch of independent variables, and these are sonar soundings at different frequencies, and the frequencies range increase slowly so you can expect that one frequency and the one imagery next to it are going to be highly correlated, right!

So, that would be an expectation here and the dependent variable is whether something is a rock or a mine. So, I guess, what... what we're saying is that we already have some data that tells us that for these frequency, for a certain rock, we send sonar sounds and we got a set of frequency, some set of data back, and that. So, we know for that data, there's a rock. And then, similarly, we know for some data,

there's a mine. We already know this, right! So, this is our—sample is available to us and we've done this over one field and now we want to expand this into other fields, so to speak, right! Remember, this is actually underwater but I'm going to be using the over land analogies, it's a lot simpler to understand. So...so that's our data. Typically, in a machine learning exercise that's called supervised learning. What happens is that you already have data that gives you both the values of the independent variables as well as that of dependent variable, and you want to train your algorithm or your machine to use that data to figure out whether, you know, in the future, when you get some new data arriving, whether that fits, in this case a rock or a mine, or whatever dependent variable you have.

So, this is an example of supervised learning, which is actually probably more common than anything else, but it is, that's where it is. So, the data we have is sitting in this URL over here, and you can just put the URL there and notice that 'read csv', pandas 'read csv' function over here, and all we need to do is, give it the URL and it'll get the data from the URL. So, csv data doesn't have to be sitting in your local computer but it can be anywhere on the internet. You give the URL and you get it back there. So, what do we do? Well, we import 'pandas', we import the 'dataFrame' but we don't have to, we can do 'pd' to 'dataFrame', and then we run this, and we get our data. So, this is what the data looks like, right! So, 'df' dot 'describe', remember, describes all the numerical columns in a data frame. So, we have a bunch of columns, so we can see the columns go from '0' all the way up to '60'—'59', right! So, there are '60' columns in the data set, and they contain, this is the data containing them. Okay, so this is the data that we have but these are all the numerical columns.

So, there's one additional column, which we will see in a second, which is the categorical variable, which tells us if something is a rock or a mine, that won't come up in describe. So, this tells us a lot of stuff. As you can see, we get the mean, the standard deviation, the minimum, and the quantiles—quartiles rather, of the data over here. So, we can...we can check that. And, just to show you a couple of things in pandas, you notice from '9' to '50', we have a dot dot dot over there. What this is telling us is that the data frame is too large to show on a screen. So, it's essentially trying to give us a sample of the data. Rather than showing all '60' columns, it's showing us a sample of the columns, '1' to '9', and then '50' to '59'. So, you can tell pandas that you want to see them all, and the way to do that is, to use what's called the 'options' command here, and the 'options' have many many things. You can look them up, you can display the maximum number of columns, the maximum of rows. We saw earlier that, when we had large data sets, that when you're viewing the data set, it puts dot dots in the middle. So, if you have like '10,000' lines, you'll see maybe '100', maybe '20' lines in the front and '20' lines in the back, but you can force it to show them all. So, what we're going to do is we say, show me all the columns. So, now when we look at this...this thing here, we get all the columns, okay! Not that it doesn't matters for this, it's just another thing that you should know.

Video 2 (9:44): Machine Learning - Understanding Data

So, let's take a look at one column, because when we, whenever we look at data, we want to really understand what the, what's going on inside our data before we do anything right! We've already seen this when we did our data cleaning but in this case, the data is clean, but we still want to understand

what is the nature of this data. So, let's examine the distribution of the data in 'column 4'. And if you look at 'column 4' here, you have the quantile data for 'column 4', or actually column 5, I guess, should be column 5. Okay, '0.67', oh, here, 'column 4', I'm sorry, '0.0067'. So, this tells us that— let me push this here. So, the minimum is '0.0067', that's this number there, and the maximum is '0.4010'. So, our data ranges from this to this. All we want to see is, whether the quartiles are evenly distributed or not, because that, you know, tells us how well distributed the data is. So, we can see here that 'quartile 1' ranges from '.0067' to '.038', which is about '.03' in range. This is '.03805' to '.0625', '.0625' to '.100275'. So, these three are reasonably similar, and the only difference that we have here is that the last quartile actually is humongous. It ranges from .1 to .4. It has a point three range. So, it's much larger than other quartiles.

So, this, what this does is, it tells us that, "Hey! maybe we have some outliers in our data." And, outliers can mess things up, because when you have outliers, it's harder to predict, right! You're trying to get may be a line or predict line or predict something in the future, but an outlier can mess stuff up. So let's go ahead and look for outliers in the data, and see what's, what's happening with them. We can use a quantile plot, which— to help us identify outliers, and here let's take a look at the quantile plot. So, what this does is, it'll show us a draw line, which is the straight line from our...our data, and then it plots all the data points as we go from decreasing order to increasing order in our dataset. So, if they were uniformly distributed or evenly distributed, I should say, then there would be on or close to this red line that we see in the plot. But we can see that we have a bunch of outliers, a few on the bottom, not a lot, but there's a lot on the top, right! On the top there's a whole bunch of outliers.

So, that's not so great, but, you know, we got to live with it, because that's our life for us. So...so, that's the first step and when you look at outliers, you have to decide whether you want to, you know, keep them or take them out or whatever. Typically, taking out outliers is a process that is best avoided because the bottom line is you want to get as good a result as possible, and if the outliers are, you know, fat enough, that means the enough outliers, then you, when you test your model in the real world, you're going to have a higher probability of getting blown up, which is not so great. On the other hand, if you are, if you believe that the cost associated with not correctly dealing with the outliers is low, then you can throw them out because in that case, you'll get a much, a far more robust result in the part that is reasonably following the red line in our plot here. In this plot, if you can, you know, for a large chunk of the data that's on or close to the red line, if you can get a reasonable estimate, and if the cost associated with...with missing outliers or not using, not dealing with outliers probably, properly is not high, then, you know, you can just ignore it. So, for now, and say, let's chuck them out. For now, let's keep them in because we want to see what we get from all this stuff here. So, next thing you want to look at is what are the unique values of the dependent variable, because we saw that we have 0 to 59 independent variables, and there's the 60^{th} column, and that's a dependent variable, and we find that's an array with 'R' and 'M'. 'R' is a rock, 'M' is a mine, okay!

So, that's the stuff there. So, we know that we have two dependent—two values for our dependent variables, so we've got, really looking forward, or looking at this stuff, we can see that we need to guess the categories of the data. As we get sonar soundings, we want to guess what category they're going to fall into, right! So, we have a categorical learning problem over here, and our categories are two. It's called a binomial classification problem. So now, let's look at the correlations between the dependent

variables, and this is also kind of crucial because if your dependent variable are highly correlated, then adding lots of dependent variables is not going to help you that much, right! So, typically in regression, you want to find relatively uncorrelated dependent variables, and, sorry, independent variables, and use them, rather than using a whole bunch of correlated ones. So, looking at this, we can just eyeball this and see a '0' and the '0' obviously is a '1', diagonal elements are all 1's, and '0' to '1' is '0.73', '0.57', '0.49', '0.34', and this makes intuitive sense to us because we know that the sonars— we're just increasing the sound, the frequency of the soundwaves slowly. So, the idea that '1' and '0' and '1' and '2' will give you pretty similar results is not, soundings is not, you know, it's very reasonable.

So, the correlation is going to be high and then it'll keep decreasing as we go by. If you look through this stuff here, we find that it keeps decreasing, if you look at the row '1', and this...this one, right, keeps decreasing, keeps decreasing, decreasing, and actually becomes negative around here, right! So, in the middle range, it's negative, and then as you go up there, it starts increasing again. So, oddly enough, we start getting higher correlations as our frequencies get really divergent, okay, which is kind of interesting but that's something to note here. So, there are various techniques in linear regression, in regression to find the set of uncorrelated variables that will explain your result more than, more. So, you want to try to look up those things if you want to, but we are going to, for our exercise, we're going to just use them all. So, we can, we have these correlations, it's kind of hard to read this whole thing. So what we'll do is, we'll draw a little color plot that shows our correlation, it's kind of a neat way of looking at correlations. So, we give it our correlation matrix, that's the dataframe really here, the dataframe of correlations that we compute, this one. And...and use the...the 'matplotlib' 'pyplot' function call 'plot' dot 'pcolor'. So, what pcolor is going to do is it plots these in color. Now, this is not very exciting looking but I can explain this to you here. So, what we get along the, oops, work along the diagonal over there the yellow one, it's very high correlations. As we go into the lighter yellow and then into green, the correlations are dropping.

As we go into the blues, the correlations are becoming negative, and then we go back into more positive territory, ending up with positive correlations again. So, if you look, for example, at the zeroth column, right, the very first column, '0', and watch it as it goes along. So, we see that '0' is initially, the correlation between '0' and 1, and maybe 2 is high, and then it starts dropping off, and then it goes on until we get to about 21 or 22, where it turns blue, right? So, that becomes negative, and that goes on till about 30 something, and then it starts going positive again, and finally at about '60', it's reasonably positive. Nowhere near, of course, the high correlations we saw earlier, but the correlation is out there. So, this tells us something about our data. That the correlations are, you know, tightly correlated across the various variables, but there's a lot of room to play in the mid area where there's negative correlations and low correlations in the...in the middle area. So, we can...we can use that to our advantage, but, for now, we'll just stick with this. So, let's look at the correlations of '1' with this, and this will really, intuitively, will show us what our, what we've been talking about so far. What this shows us is that the correlation of 0 to 0 is, you know, I had one, then 0 to 0 is point 9 7 8 9 something like that, and then it drops rapidly, right! As you go to 2, 3, 4, 5, it drops rapidly down.

So, probably around 5, it's already dropped all the way to '0.2'. And then, it keeps dropping, it's sort of juggles around a little bit there, it keeps dropping into negative territory, around 25, and then it rises, rises, rises, ending at about '0.2'. So clearly, our situation with... with this, with one anyway, and for

most of them this is the case, that the—for the two or three columns that are around a column, plus, minus 2 or 3 columns, the correlations are very high, but then it sort of tapers off. So, in general, high—highly correlated items are not good, low correlated items are good, and you want to actually try to pull out the ones that are highly correlated if you can, right! But, in our case, we're going to have a hard time doing that because...because of the way the data is, we, if we pull out one, we still have, you know, everything is correlated to everything nearby. So, we can't put all the nearby columns of everything out. So, we might have to do like one and five and 15 and 20, and maybe it's not a bad idea to try that. So, and, of course, if the correlation is high with a dependent variable, then that's...that's good, you know, that's something that you want to look at. So, our dependent variable here is categorical. So, we don't really have correlations. So, we skip that step.

Video 3 (7:04): Machine Learning - Wines Dataset

The second data set that we look at is the wine data. So, let's take a look at that as well. So, wine data has a bunch of 'independent variables' and these variables are the composition of wine. And the composition is really the chemical composition. What's the alcohol? What kind of sulphites? What kind of acidity it has? You know, that kind of stuff. So, it tells you, if you took your wine sample and then analyze it chemically what you would see. And, the dependent variable is interesting. It takes a panel of three wine tasters and they taste the wine. And they rate the wine quality between 0 and I think it's 10 but not 100 percent sure. So, 0 and say, let's say 10, 0 and something. So, they rate the wine and then we take the average of the three ratings, and we get a rating for the wine.
So, the goal here is to see if we can take a wine sample and chemically analyze it and then predict whether its quality is going to be rated high or low, by these three wine experts, anyway. But in general assuming that these three wine experts are reflective of the entire population of wine experts. So, let's get that data. That data's at this URL over here, okay! So, you can get that in and again we do a read CSV. We get that. We had a zero, and the separator in this data is not comma, which is a default for read CSV but it's a semicolon. So, we give this additional argument over here. Let's say separator equals semicolon, right! So, that's what read, it says read CSV, it can actually read any file structure as long as you know what the, what separates the data columns in that, data values in that file. So, let's take a look at this. So, we get a WDF and we look at this here. And we see, we have all these independent variable columns 'fixed acidity', 'volatile acidity', whatever. And it tells us, we have '1599' data points here, quite a bit. And the last column is the quality. And this is the average rating of the, the 3, 3, 5, 6, 8, 6. These are the quantiles. The average is 5.63. And the standard deviation is 0.807, okay! So, that's our variable over there.
So, that's...that's a different example. The first example had a, our rocks and mines data, had categorical variables and the wine data has continuous variables. So, let's take a look at the 'volatile acidity' column. Run that and this is the values we get over here. So, they're all nicely, this thing. And, we have lots of data here where we can take a look at the correlations, and do our plot. So, it's looking better and this tells us that the correlations from '0' to '0' is like one, which is big surprise but nothing else is really that tightly correlated, right! We have some, little bit correlations like between '6' and '5'. There's a little bit

of correlation. but there is, otherwise the correlations are pretty mixed. So, this is relatively uncorrelated data. And so that's good for us, right! We can also look at the correlation of one with something else.

So, we look at the correlation of one with each of the others. That is 'fixed acidity' with each of the others. So, that tells us that there is some correlation here like 0.5 something with 'citric acid' and 0.6 something with, I guess, 'sulfur dioxide'. but it's still all over the place. We can use a scatter matrix, a scatter matrix to visualize the correlation between features but though this is kind of not too much data. So, we can actually show this here quickly but scatter...scatter matrix is a fairly 'processor intensive' function. And, but it's interesting to see that. So, what we're going to do is, we are using the scatter matrix function here and we're giving it our alpha, if you know alpha is a density thing. So, the figure and we're giving it stuff here. So, we get this. So, the diagonal, this is a scatter matrix and what it tell us is that the, it looks as the... as the scatter plots of each pair of independent variables and of each variables in this case. So, we see, for example, let's take sulphates, right! So, because that's clearer and alcohol. So, we've got sulphates and alcohol and we look at it. And, we see it's relatively uncorrelated because the blobs are no distinct pattern. If we look on the other hand at alcohol and— I can't read these things very well. Let me see if I can find one that looks clear enough. Let's say the top, the third element from the top, on the top row. That is the...that is the top row is fixed acidity, and the third element, citric acid. That's the third element from the... from the left on the bottom row. Then we see that that's, you know, the scatter plot looks more increasing, as one increases, the other increases.

They're still pretty chunky. They're not that tightly correlated. Some of them are a lot more tightly correlated as we'll see but this one not that chunky, as we can see if we look at these graphs. So, scatter plots give us a good sense for how tightly correlated two variables are. The middle, the diagonal columns over here are the distribution of the variables themselves. So, we can see that, you know, for example, the very first top-left corner, we can see it has a fat tail on the right-hand side, on the higher side. And in fact, looking through these things, there are lots of fat tails and some of them are really well poorly behaved. Like citric acid is poorly behaved, right! Citric acid is the third from the top and the third from the left. So, that's poorly behaved. Something that is like this one over here, which is couple of them over here which are pH, and the one before that are reasonably well-behaved, right! They look kind of mostly normal. So, and if you look at the dependent variable wine quality, we find a problem because it has really got two so to speak midpoints, right! So, two...two highpoints in this curve here. So, it's not at all well-behaved at all. So, we're going to have a problem with that. We know that upfront, right! So, this probably won't work very well with regression. So, we won't use regression, we'll use something else for this. So, we've done that. We can also do quantile plots just like we did with the rocks and mines data. And, that again shows us some outliers over there, and, so that's really the first step. The first step is to take all your data and examine the distributions.

Examine the correlation between variables, look for outliers, and try to understand what the data looks like. What is the distribution of the data and you know whether you're going to get results that will make sense or not. So, looking at this, for example, the quality, the...the shape of the quality data. We can see that regression isn't going to work very well with this because it's not going to find a single, nice line that we're going to fit this thing here, right! So, you want to see what kind of technique will work

better with it as well. So, that's our data segment. We're going to move on to actually doing some regression with this thing here.

Video 4 (7:05): Setting Up Regression

So, now let's see if we can train a classifier on rocks versus mines. So, to do this, we're going to use a package in Python called 'sklearn', or sci-kit-learn, sci dash...sci dash kit dot learn, or something like that, anyway. We've got this here. So, we get sklearn. We import data sets, and we import the linear model. We're going to use the linear model here. And we will import—this I'll do again anyway, but these are the metrics we're going to focus on, the roc_curve, and the auc and I'll explain that as we go along and we can work with this. Yes, it should be imported before I forget. And, we also import NumPy and random because they're useful for building our training and testing samples. Then we read the data, of course, so we've got that. We've already done that before. And now, the first thing we want to do is we want to take our dependent variable, which are currently two characters, 'R' and 'M', and convert them into '0' and '1' because when you're doing a regression, our dependent variable has to be numerical. So, we convert that into '0', '1', and we're going to use our 'np' dot 'where', if you recall this one. This is like the Excel F. And what it says is wherever 'df[60]', which is the dependent variable column, equals 'R', you pop a '0', and otherwise, you put a '1'. And we know that from our earlier examination that unique values are only 'R' and 'M'. So, we won't have any NAs or anything else to deal with. So, we take our 'df[60]' and we convert it from 'R' and 'M' into '0' and '1'. The next thing we need to do is to divide the data set into training and test samples. Obviously we can take our entire data set and run it, and, you know, get the result, and then hope for the best when something new comes in. But that's not going to be really a great idea because what we want to do is, we want to see whether the results that we get are robust or not. If they're not robust, then we don't want to use them when we send our troops or people into the minefield. To test whether it's robust or not, typically what you do is, you take your data set and divide it into two parts, a training and a testing sample. You train the machine on the training sample, and then test it on the testing sample to see whether the results you got from the training sample actually make sense with some data that you haven't used in training, in...in trying to parameterize or train your model.

So, that's...that's really an important step. If you don't do that, you know, you get into trouble. You have to understand, you have to make sure that your results are going to be reasonably robust, otherwise, you're going to be in trouble. So, training and testing and in fact, often people use a third, holdout sample, completely unseen sample. And then after they have trained the model, training and testing, and they have tested it out and looked at different ways of training it, then they finally take a third sample that...that hasn't been used at all in the process and see whether the results hold over there. For example, you might train your model using one classification technique, and then try another classification technique and try a third classification technique and then you pick the one that looks the best. But at this point, you've only trained on one set of data, tested on another set of data, and then you used that to pick a model.

So, now you want to see whether that process of training and testing has sort of biased your results to whatever data was in the training and testing samples. So, now you pick a third one and test it out there, and hopefully it will work out well. We'll...we'll use only a training and testing sample and 'sklearn' has a...a...a model selection package, which contains a function called 'train_test_split'. What that does is, you give it the fraction of data that you want in your test sample. I think test size equals '0.3' over here. You give it a dataframe and it's going to split the thing into two dataframes, a training and testing one, and then we've got that. So, once we have a training and testing, we're going to pull out from our dataframe, the independent variables. That's column 0, sorry, row '0' onwards to '0:60', which is from the training. We have two things here, training and testing. Let me run this and we can see this for a second, okay! 'Insert', cell above. So, here if I look at train, we see it's a dataframe, and it has row '6', row '193', row '179', row '139', row '94', row '24' in it. So, it's randomly picked from a certain set of rules from our dataframe, and that goes in the training and the other rows have gone into testing, the ones that are not in...in this thing here. So, that's our 'x_train' and from the training on this dataframe, that is our training dataframe, the one that we just saw here, we pick the columns '0' through '60', that is '0' through 59, remember the last one that I have included. So, these are independent variable columns, and they go into the 'x_train', which is our data, our training data set, the independent variables in our training data set. And the 60^{th} column, that is actually 61^{st}, but that goes into 'y_train' which is our dependent variable, and similarly we do the same for 'x_test' and 'y_test' for the testing sample. And notice, we use the 'iloc' here because we're using row indices to pick data and we use row indices to use 'iloc'. If you use the row numbers, zero, one, two, three, four, then you use 'iloc'. If you use the actual index, we don't have an index here but if you add an index, and you use an actual index, you could use dot loc instead.

So, this gives us our thing here and we see that y training contains '6', '193', '179', which pretty much corresponds to the '6', '193', '179', '139', '94'. So, they...they are aligned, you know. We want aligned data in our training and testing samples. So, once we have that, then we build our model and fit the training data and this is the standard procedure regardless of which particular model you're using for machine learning from the sklearn package, you're going to use the same two steps. You're going to first start by building a model itself. So...so, you...you get this and we're going to use linear regression over here. I should point out that for our data. Logistic regression is another option, which is probably slightly better but because I want to explain how to, you know, the various methods for evaluating results, we'll use linear regression here. To use logistic regression, you just replace linear regression by logistic regression, and then, you know, most of what we are do is, probably going to work equally well with that. And, but anyway, so we have here, we're going from the linear models in our, in sklearn, we're going to be picking the linear regression package and recreate this model here. And then what we want to do is, we want to fit the model to our training data. And to fit the model, you tell it what your training sample, independent variable values are, and what the corresponding dependent variables are, and that will fit the model for us.

So, this is pretty much what we will do regardless of which particular model we happen to be using. Once we've done that, we fit it in our model, and now we can start thinking about how to interpret the results.

Video 5 (11:32): Classification Metrics (Part 1)

'Interpreting' the 'categorical prediction results'. We have a categorical prediction problem here, classification problem. We—there are a bunch of statistics that we can use to figure out how good our results are. And a lot depends upon what we want to do with the resulting model. So when you think about this, we're going to try to use our rocks and mines example. And keep that in mind as we work with it. But before we do that, we'll, I'll digress a little bit, and we'll take a dummy example and work through some of these various prediction results that we can get. So the topics here are pre 'Precision', 'Recall', 'True Positive Rate', 'False Positive Rate', Precision recall curve', 'ROC curve', F curve, and the area under the curves. So for this, let's move to our slides and take a look at that. So let's see what the class...what classification metrics we can use in our results here. So here's an example that I'm going to work with. The example has a predict— too has a dependent variable with two categories. And the category are represented by ones and zeros. So this—the column one in our example here, this column is the actual data, right! So this...this is actually what the—what we know. We know that case one had an actual of '1'. Case two, an actual of '1'. And the last case, for example, had an actual of '0'. And then column 2 is what our model has predicted.

So these are the predictions, right! So for example, if we look at the... if we look at the first row. It tells us that the actual was '1' and the predictor was '1'. So that was an accurate prediction. The second row tells us actual was '1'— the second case and the prediction was '0'. So this was an incorrect prediction. So the first thing we do when we are looking at categorical variables and trying to see—a category classification prediction problem is— and trying to see how well we've done our prediction, or the machine has learned is that we...we create what's called a confusion matrix. A confusion matrix, it is sort of a...a 2 by 2 in this case, but it could be, depending on the number of categories you have, could be larger. But it'll tell you how...how many cases you got right and you got wrong along several dimensions. So, the first thing it tells us is that if the— so a general idea here is that '1' is a positive prediction. So, you could say a mine, for example, and '0' is a negative prediction or a rock, okay! And the positive and negative, though we use those terms are the connotations are not strictly accurate. For example, I might decide that I'm more interested in finding the rocks and not the mines.

So, maybe you know then a rock would be the positive and the mine would be the negative, right! So, that's the idea there. So, but...but the goal here is say okay, for our example, a '1' is positive and a '0' is negative. So, the first thing we do is we look for true positives. True positives are cases like this one here. Where the actual is one and the prediction is one. So, we predicted something as being positive. And it actually is a positive. So, that becomes a true positive. So, we can count the true positives and the true positives we can see are this is a true positive. That's a true positive. That's a true positive. That's a true positive, and that's a true positive. So, we count them, we get five of them. So, we get five true positives. And we mark a five over there. So, that's our first data point in that cell in the confusion matrix. So, you can think, it's called confusion matrix. It tells you how confused your model is really, right! If it was perfectly unconfused, then all cases would fall either in this cell or in this cell, right! The true negative cell. True negatives are what, true negatives are where a model predicts a '0' and it actually is a '0'.

So, if you look at true negatives, that's a true negative. Oops. That's a true negative. That's a true negative, and that's a true negative. We get four true negatives, right! So, essentially our model has done a good job on this five and four, nine cases in our sample data set on this fake data set. Then on the other hand, we also have cases where the model, where the actual value is 1, and the model predicts a 0. So, the model is predicting a negative. So, that's, it's... it's incorrectly or falsely predicting a negative. So, we call that a false negative. And a false negative is where the actual is a one or positive. And the model is predicting a 0 or negative. Or falsely predicting a negative. So, it looks like I made a mistake in the, in this column over here.

So, I'm just changing for the purpose of this, changing the last column to, from a 0-1 to a 1-0. Got to interchange over there so the numbers match here. So we get a false negative in these three cases. This, this, and this, and that's where the actual is a '1' and the prediction is '0'. And then finally you want to look for the false positives that is where it, the model predicts that something is a '1', but actually it's a '0'. So it's predicting a positive but in actuality it's a negative. So for that, we want a '0' in the column one. And a '1' in column two. So that becomes here, false positive and false positive. And we get two of those so that becomes our confusion matrix. So this tells us how confused we are. Generally, like I said again, if you have a really good model, then you would hopefully get all your cases falling in these two cells. So now we can look at the race metrics. So the metrics are, the first metric is precision. And precision tells us what proportion of the cases that the model said were '1' were actually '1'. So this is, if...if the model is saying '1', that means that our predictor, our machine learning model is saying it's a '1', and it actually turns out it's a '1', then and it does it in all the cases, then our model is very precise. In other words, it's recognized every... or every mine that it recognizes the mine is actually a mine. That doesn't mean it recognizes all mines. But it finds...it finds a mine, it's a mine. Okay, that's the, if it's 100 percent, right! Then that's a mine. That's what it's trying to tell us. So generally you know if for example, let me flip it around a little bit and say if rocks were a '1'. Right, and mines were a '0'. And then we identified all the, everything the model identified as a rock was actually a rock. Then we could sort of safely across the field by stepping only on things that the model recognizes as a rock. Because we know it's going to be a rock if it's at 100 percent right! So that's what... that's what precision is really telling us. So that's the first thing. And that is defined by the total of true positives, divided by the true positives plus false positives, right!

So true positives plus false positives are the number of positives identified by the model. And if the true positives equal the true positives plus false positives. Which means the false positive is zero, then the...then we get 100 percent precision. In our case, we get five true positives and two false positives. So we get '71.4 percent' as our precision. Which is, you know, telling us that in our case, 71.4 percent, of the time if the model recognizes a mine, it's actually a mine, okay! So that's what that tells us. The second thing is recall. And what recall says is what proportion of the cases that were actually one or positive were identified as one by the model. So what this is saying is that if you look at our data set, and we look at all the actual mines in our data set, then we want to find see of, how... how many of these were recognized as mines by the model itself. For example, or just to put it in perspective. If our model exercise— finds all the mines and identifies them correctly as mines, that's a great thing. Because that means that it's, you know, everything that it hasn't identified as...as a rock, as a mine is not a mine,

right! That's...that's what that means. So if you look at this here, we get, for that to be case, the true, the total number of actual mines are the true positives plus the false negatives.
Why? Because if you look at this here, true positives are, this is true positives here, and what we're looking at the total of actual one which is the true positives plus the false negatives. Five plus three, right! So that's the total number of actual mines in our...in our dataset. So that's our denominator, true positives plus false negatives, and the numerator is a true positive that is in the proportion of the number of mines that it correctly recognizes as mines. Or the ones that it correctly recognizes as ones. So, if the—which means that if we get 100 percent, then we have no false negatives, right! And we...we know that we identified all the mines correctly. Even if we actually have some additional recognize—let's say our, we could have some false positives in this. Like some, you know we could have let's say there are 100 mines and 100 rocks, and we identify 120 of the, our set as mines. Then we still have 100 percent recall if all 100 of the mines are actually in that 120 set. We have 20 false negatives or false positives, sorry. But those 20 false positives we don't care about. Because we know that, you know because we got the mine's correctly, that's the idea there. So, in our case, we go to 62.5 percent recall rate. And then there's a thing called an F-Score which sort of balances these two out. And note that, you know in reality, depending on what you want to do with the model, you might want to focus more on precision or more on recall. And there's a tradeoff between the two. So what the F-Score does is, it sort of balances them out and tries to find if you don't care, you know whether you're more precise or more better at recall, and it does that by using this formula here. Which is sort of a balancing formula. And we find that our precision recall score is 67 percent. The other two metrics that we focus on are called true positive rate which is what proportion of the cases that were actually one were identified as one. Which is the same as recall. So that same 62.5 percent. It tells us again you know how many mines were, how many of the actual mines were correctly identified as mines. Basically. And the false positive rate, which is the really the difference— different from the others but says what proportion of the cases that the model said were '1' were actually '0'. That means that what proportion were the model identified as mines, but they were actually rocks. Okay, so again, if you flip it around a little bit and we say if you're looking for the rocks, we want to make sure, for example, that—you know ideally what we would like in our minefield is to correctly identify the rocks. If we correctly identify the rocks, then we are in good shape. Because we want to cross the minefield. So if you can step only on rocks, we are safe, right! So we are not as interested so to speak in finding all the mines correctly. We are more interested in finding all the rocks correctly. If that makes kind of sense. So for us again, we could trade off these true positive rate and false positive rate, and figure out, and there—you know they are, you're going to have to trade them off. And figure out where they land in this stuff here.

Video 6 (6:18): Classification Metrics (Part 2)

Once we've got that, we can fix a threshold. To fix the threshold in and to study the efficacy of a model, in classification problems we plot two kinds of curves. The first curve is something called an ROC Curve, and the ROC curve plots the false positive rate against the true positive rate. So what we get is a curve that looks like this. The true positive rate on the Y axis, and the false positive rate on the X axis, and for

each threshold point we plot the various TPRs and FPRs. Right, that's the goal there. And the precision recall curve plots precision against recall in exactly the same way. So we get here precision and recall, and again, for every threshold point we plot the precision and recall values that we get. So the goal with this is twofold. I actually have an example here. Let me see that here. And then once we've got that, it's twofold really. So when we look at the ROC curve here, for example, so this is a case where the ROC curve is this one. So this is the tradeoff between the true positive rate and the false positive rate. And in—and this is, you know, for different for increasing threshold values.

So this is our classifier, and we take...take this thing, and we plot it against the random classifier. The random classifier would randomly pick and, assuming they are equally weighted, would randomly pick one category or the other. And you would get a curve that looks something like this for each threshold value. So generally you want your ROC curve to be above this random classifier. If it's below that, then it's doing worse than a random classifier. If it's at that, it's doing the same as a random classifier. And if it's above, it's doing better than a random classifier. That's the idea there. And the goal really here is to look at the curve and see whether it's stable or not. So what you don't want is, you don't want a curve that looks something like this. A curve that looks like this. I mean not that you're going to get that, but you know something like this. You know, what this tells you is that we have a curve that it will vary depending on the threshold value. I should clarify what is the threshold value. So we will understand...we will see that in a second. Okay. The threshold value in our rocks and mines example. So what happens here is that when you are doing a linear regression, we have a dependent variable that is either 0 or 1, right! So we've got a dependent variable that is 0 or 1. Now with a dependent of 0 or 1 on the right-hand side, we're doing, you know, regression. The regression is not going to predict 0 and 1. It's going to do '0.2', '0.4', '0.3', '0.9', '0.6', etc. Right! So we need to decide at what— But what we want to predict is a 0 and 1. So what we do then is we say, "All right, we're going to put a threshold value." And the threshold value would be, say, '0.3'. So anything that is above '0.3' becomes a one, and anything that's below '0.3' becomes a zero. That's...that's the way we cover the thresholds. So that's...that's how we convert the continuous dependent variable prediction into a categorical dependent variable value. And so the threshold becomes really important. So typically you would say, "All right, the threshold should be 0.5." And if you do that, for example, the logistic regression model in SK learn, then that's going to use 0.5 in the threshold. So what we want to do instead is we want to say, '"Hey", maybe 0.5 is not a good threshold."

Maybe '0.2' is a better threshold. Maybe 0.8 is a better threshold. So that's where this ROC curve and PRC curves come in to the picture. Right! So this tells us for different threshold values, what do...what is the value that we get for the TPR and the FPR? And what we would like to see is a threshold value that is, that doesn't sort of fluctuate a lot depending on where you pick it. So, if you pick a threshold value here, for example, then we get a, you know, a nice point here at point which iswell above the random classifier. But just a little bit, move a little bit away from there, and we are down here. Right! That's not so great. So because we don't want to be picking a threshold value that sort of fits our data. Because if we...if our data looks like this, then if our results look like this, then picking a threshold value is going to be fitting to our data. And remember the goal here is that we don't know what the future is going to be like. We are looking at the past. Like we have data. We know what's happened. Right! We know all that stuff. But the new data that comes in is not necessarily going to be exactly the same as what we had

before. And if a small change in that data is going to just upend our results, then you know it's not so great. So what we're looking for is a model that's reasonably robust, and the robustness is best defined by an ROC curve that is sort of even like this. Right! It doesn't fluctuate wildly depending on threshold values. That's the goal there. So we can draw an ROC curve, and a PRC curve, and figure that out. And what we do also is we compute the area under these curves.

So we compute a AUROC and a AUPRC. We can compute these two, and I've just drawn one here, but the area under the curve tells us what is this area really. What is here underneath all this. That's the area under this curve. So any area under the curve of 0.5, if the area under the curve is greater than 0.5, then, you know, we have a classifier that is doing better than the random classifier. And the higher the area under the curve, the better your classifier is. So ideally we would look for an area under the curve that's somewhere here, maybe, you know, like this, if you can get a curve that looks like this you know, great. A curve that looks like this, even better. That's the...that's the rough way of looking at this stuff here. So that's a sort of digression in to understanding how we evaluate classification problems.

Video 7 (7:41): Classification - Rocks and Mines Dataset

The first thing we can do is generate in-sample error. So, we looking at a regression, so it's kind of straight forward to go and look at the predictions that our model does and look at the actual values and computer mean square error for example on that, right! So, we're looking at our training predictions, our actual y dependent variable values, and squaring them— the difference between the two and squaring them, and we get a mean square error. We can look for 'R-Square', that's typical in regression so we can look at an 'R-Square'. And we get an 'R-Square' of '0.65' on our training sample and '0.04' on our testing sample. That's pretty horrible when you think about it, right! Now look at this the training is '0.65', the testing is [inaudible] '0.04'. But the question is do we really care? The first thing we have to note is of course is that our problem has training predictions and Y. The predictions and actuals are not identical, right! In the sense that they're not in the same kind of data that we're looking at.

So, if you look at our two samples here, we look at 'training_predictions'. Oops. This is the values we get, right! Negative '0.101', '0.38', blah blah. And if I look at 'y_train', that's '0' and '1'. So, obviously the mean square error the R square are going to be messed up, right! Because there are we are comparing '0' and '1' by that kind of stuff. We haven't applied a threshold, basically. So, what we need to do is we need to start thinking a little bit more creatively and on how you are going to predict the results of our thing here. So, what we want to do is we want to— we're getting a continuous value, right, in our prediction. Which is why our stuff doesn't work. So, 'we need to convert these continuous values into categorical '1's and '0's.' And 'we can do this by fixing the threshold between 0 and 1'. And 'values greater than the threshold are mines' and 'values less than the threshold are rocks'. And we build our 'confusion matrix'.

So, in SK learn for regression, there is no 'confusion matrix' but for logistic regression you can actually build a— there's a function for that. So...so, what we'll do is we'll write our own function because that's easy to do. So, what this function does is it takes our 'predicted' values, the 'actual' values and takes the 'threshold', and tells us whether— how many cases it essentially constructs that matrix that we saw.

How many cases are true positives, how many are false negatives, how many are true negatives and how many are false positives, right? So, that's a very straight forward thing to do. So, I'm not going to walk through that. We write that function and we're going to create a 'testing_predictions' variable that is our predictions based on— I should also point out that once you've fitted the model, we can actually get predicted values by calling this function 'model' dot 'predict' and give it a set of independent variables and data frame containing our independent variables. And it'll produce the dependent— the actual predictions for that set of independent variables.

So, we can run that. And we have this, and now we are going to run our confusion matrix and let's try it with '0.5' and see what we get. So, we run it at '0.5', we find that we have '27' cases that are true positives, '9' cases that are false negatives, '5' cases that are false positives, and '22' cases that are true negatives. We can now actually compute all our true positive rate, false positive rate, precision recall, whatever we want using these numbers, right, for this threshold value, for a single threshold value. We can also compute what's called a 'misclassification rate'. Which is the number of false positives plus false negatives or the number of cases. Which tells us how many cases were incorrectly classified. So, we can see that in our testing sample, 22 cases— '22 percent' of the cases were incorrectly classified, right! So, that's—if you get of course 0 percent then that's great. That means everything was correctly classified. Essentially what we are saying is that we want our false positive and false negatives to be minimized. So we can in a very generic sense— because our actual model might require something different. But in a general sense we want our false positives and false negatives to be minimized. So, if we can minimize that number, that would be great. We can look at the 'Precision' and 'Recall'. And again we run our 'confusion matrix' on that and we get a '84 percent' confusion— 'precision', '75 percent' 'recall', and an 'f_score' of '0.79'. Which is not that bad. The question of course is what does it costs us. You know, in the sense that is it good enough. Good enough would mean it's very expensive to make mistakes, right! So we have, that's...that's where we always go with this stuff here. But for now we look at this and we say "Hey! That's not a bad result at all." It looks pretty good. '84 percent' 'precision'. '84 percent' of the mines that it recognized—that the model said were mines were actually mines, right! That's pretty good. It 'recall', '75 percent' of the mines that, the mines were actually were in our setup, '75 percent' of the time— '75 percent' of those were recognized as mines, right! So obviously some were—we have some...you know some mines that were classified as rocks. And that's why it's not—they're not equal. So, the, obviously the results that we get, all 'Precision' 'Recall', the numbers, 'Misclassification rate', all these kind of—these numbers here in the confusion matrix, all these depend upon what the threshold is. So, what we want to do is we want to start trading of these things. So, let's try for example a threshold of '0.9' and see what we get, okay! So, we saw that we had '84 percent', '75 percent' and '79 percent' for 'precision', 'recall', and 'f_score'. And if we run this with '0.9', we find that the 'precision' goes up to '88 percent'. The 'recall' drops to '41 percent' and the 'f_score' drops all the way down to '57 percent'. So, that's a pretty steep change. The question now is are we willing to give up so much on the recall, like all the way from '75' to '41', or '42' rather, just for a four percentish gain in the 'precision'. And again, that depends on what we plan to do. If you are looking at, let's say identifying all the mines— if you wanted to make sure that our model actually, when it says it's a mine that it actually is a mine, if that was our primary motivation, then we would probably say yes, you know, we don't care about it not really.

If it doesn't get all the mines we don't worry about it, as long as we know that with some confidence that when our model is saying a mine, it's actually a mine. For example, if you want to send your team in to diffuse a mine and it costs you money every time they try to diffuse a rock. So, what you want to do is you want to make sure that if you're sending them into the field to diffuse something that it actually is a mine. In which case you're more interested in precision. On the other hand, if you're more interested in getting safely through the mine, then, minefield, then you might be more interested in the recall, right! Because then you want to find more of the mines correctly. So, it's a question again of, you know, where...where your— what your model is really seeking to do in the real world, right! So, you have to really look at those things then. The next thing is something called 'Receiver Order Characteristics'. It's an...and that's an 'ROC curve' and this comes out of I think radio signals or something but it's got a history of its own. But what 'ROC curve' does is it 'shows the performance of a binary classifier as the threshold varies'.

Video 8 (11:26): Classification - Rocks and Mines Dataset (ROC Curve)

So in the receiver order characteristics we compute two series. A false positive rate series that is on different thresholds which we saw of course is the proportion of rocks that are identified as mines, things that are falsely recognized as mines when they're not, and that's false positive divided by true negatives plus false positives. And a true positive rate which tells us what proportion of actual mines this is seen as recall are identified as mines. So we compute these two, and then we plot them as we saw in our classification analysis digression, so to speak. We plot them in a curve, and we want to find out how good our model is by looking at these kinds of things here. So let's take a look at that. So...so first thing before we go to ROC curve, let's see if our...if our model has discriminated all or not. So what we can do is we can take our data, and look for wherever the thing was actually... wherever our case was actually a mine, we identify those ones. And say these are the ones that were identified as mines in— and if the prediction was also a mine. So, we have a mine that's actually a mine, and the prediction is a mine, then we store that in our record positives. And if the prediction was not a mine, then we store that, in whatever prediction we get, in the negatives. Right! So what we want to find out is whether things that are actually mines are recognized as mines, and things that are actually rocks were identified as rocks or not.
So, we do that, and we can draw a little curve over here, and that's what this is doing, which is sort of histogram which tells us that these here were mines that were actually recognized as mines. And then as we go further south, the they were recognized to the '0.06' values. These are our actual values, not our zeroes and ones, but the actual predictions, the continuous predictions between 0 and 1. And so this tells us that, yes, we have some kind of discrimination, but not perfect. Right! So in this, the greens are rocks, and the blues are mines. But our recognition, you know if you have a threshold, say if you put a threshold at '0.8', for example, then we pretty much have everything correctly recognized as a mine. If you put our threshold anywhere below that, things get a little bit messy. Right! So this is what this is showing us. So if the greens and the blues are perfectly separated, then our model has done a great job of separating. But here we have some, you know, rough idea about that. But more interesting...more

interesting than that is what happens in our hold out sample. So we do the same thing with the testing sample. And you find that's a lot more messy. Right! In the hold out sample we have a lot of greens that are intruding on to the blues.
So that's not so great. Right! Still, we can see that if we fix a threshold between, let's say, 0.625 and '0.75', somewhere there, we will get some kind of a separation. We're going to get a lot of false positives and false negatives, but there will be some kind of separation. So that's one way of looking at it, but we look at it with an ROC curve. And SK learn has its own ROC curve package. So we will...we will use that. And what ROC curve does is given a y, a actual y, that is 0s and 1s that were actually in the training sample here, and given our predictions in these, remember our continuous values, the ROC curve is going to run through the predictions at different thresholds, and decide whether they're 0s or 1s, and compute the false positive rate and the true positive rate. And then we can plot that in the curve. So just to show you what that is, I'm going to take this maybe I have done it here. I'm going to take this. Insert a cell above. And let's look at that. And look at the area. So this tells us the area under the curve is '0.98'. Which is like humongous. So, our prediction—this has a very stable kind of model, if nothing else, right! And we can see the actual values also.
So we have FPR, TPR thresholds. Look at them. So, this tells us these are the FPR rates. We see them vary '00.1014' all the way to '1'. And the—if true positive rate is 0 point 0 one, well that's not so good, but '0.74', '0.74', '0.85', '0.85', blah, blah, and the array that we—the thresholds that we get because our data actually, thresholds are— it's continuous. So, rather than being from between 0 and 1, they're reading from negative 0 point 4 7 to one point 4 4. And that's okay. So that's our, that's what the ROC curve is doing. It's just looking at different thresholds and plotting TPRs and FPRs for that. So we can plot these in a nice little graph and take a look at the curve. So we get a curve here that looks like this. So, this [inaudible] is our in sample ROC curve has an area under the curve of 0.98, and we find that it's, you know, reasonably stable. Right! So if we work anywhere in this middle area that is not at the edges of the orangish curve, assuming that's orange, but you know what I mean, then we should be in a fairly stable area where moving the threshold this way or that way is not going to affect our results too much. Okay, assuming that we don't know what the trade off between TRP and FPR is. Right? I mean that's the goal here. If you know what the trade off between TPR and FPR is, we can pick a threshold, but...but we want to be picking a threshold in the middle anyway because that's likely to be more stable when we get new results or new cases coming into our model. But of course that's our training sample. Right! Right? So what we really want to do is look at the out sample ROC curve.
So let's draw that. And we find that out sample ROC here doesn't look as great, but it's still not bad. '0.84' is still pretty good. And we still find that the area in the middle somewhere from a threshold of let's say point 2 5 to point 8 or so is a reasonably stable threshold area. Again, at the extremes it's not great. In the extremes, we find that a small change in threshold can cause a huge change in the curve, and that's not what we want. We want to be working inside the stable area. So, that's the way to look at this stuff, and so what the ROC and AUC curve give you is— and the AUC number the area under the curve gives you, is a good idea of how good your classifier is and how sensitive it is to changes in the threshold. Too sensitive is not good, and so you don't want to be too sensitive. And so, roughly what we want to do then is, we want to figure out what...what should our model do. I mean we've gone through all this analysis. We have precision. We have recall. We have all kinds of stuff but we don't know what is

it that we want to finally do, right! What threshold should we pick? So, generally what you want to do then is, you want to look back in to the domain and try to figure out what the cost of picking a particular threshold is going to be, you know. In other words, what is the cost of the true positive rate versus the false positive rate, precision versus recall, depending on your model, that's what is the most important thing here.

So, let's take an example. Let's say that everything that's classified as a rock needs to be checked with a hand scanner at dollars, '200' dollars a scan. And, everything that's classified as a mine needs to be defused at '1,000' dollars if it's a real mine or '300' dollars if it turns out to be a rock. So, we have cost numbers here and, you know, you can pick anything like the cost, for example, of human lives are involved, if the mines are small mines that will injure people or they're big mines that will blow up people. You can decide, you can try to put a cost number to it and I don't necessarily mean dollars, but it maybe a utility number, right! So, anything that measures whether something is good or bad essentially, what you're picking as the result, the positive— the negative outcome is good or bad number. So, here let's say we want to be with these numbers, what we want to do is, we want to first get a confusion matrix. We get a confusion matrix on our test sample, not on the training sample. And, let's say at this point, and you probably want to do this for more continuous numbers, but we will say, we will take two extremes, '.1' and '.9'. And say, "All right! what's the cost associated with the threshold of '.1'?" So, we get back our true positive rate, false positive rate, etc., and compute our cost over here.

So, the true positive rate is 'cm[0]', confusion matrix related to the true positive rate at cm[0]. And, we know that...that it's telling us that something's a mine, but it is actually a mine. So that's '1,000' dollars, and we're going to go and test it, right! So, there's going to be 1,000 dollars. Then, the next is the false positive rate which tells us that something's a mine, but it's not a mine. So that's going to cost us '300' dollars because we sent someone to do it, to test it, and it turned out to be a rock, and we spent '300' dollars trying to figure that out. The next is our false negative rate which is— it doesn't matter in this case, but this is what tells us it's a rock. If it's a rock, it costs us '200' dollars to check it and make sure that it is actually a rock. And, if a mine is classified as a rock, and we can check it, and assuming nobody gets blown up, then that costs us another '200' dollars, right! Am I right about this? Let me see. Yep. Yep, these are classified, so you know, it could be a different number here because if it's classified as a rock, and it turns out to be a mine. You might get blown up. It is just example, let's just say, we are safe. You know, we go and test it. It costs us '200' bucks, but it doesn't kill us, right! So, that gives us a cost for '.1' and then we can run the same thing for a threshold of '.9' and see what we get. So, we get a cost here that says at '.1', it costs us '41,100' dollars on our training sample, and at '.9' it costs us '24,800' dollars. So, let's take another example. And, we said everything that classified as a rock would be assumed a rock, and if it's wrong it will cost us 5,000 dollars in injuries, treatment of injuries, or whatever. So, whenever something is classified as a rock, and it turns out to be incorrectly classified, and it turns out to be a mine, it costs us 5,000 dollars, otherwise it doesn't cost us anything.

We are walking through the field. We're not testing anything at all. So, here in this case, the only thing that we care about is this one over here that is a false negative where it says it's a rock, but it turns out to be a mine. So we again take '.1', and we compute the cost. So, this is 5,000 times this, and everything else is 0. and we do the same thing for 0.9, it's 5,000 for this. And everything else is 0. And we run the cost, and we now get that '.1' threshold is going to cost us 10,000 dollars, but a '.9' threshold is going to

cost 105,000 dollars. So, we can see that in these two cases our results are different, depending upon our cost structure. If our cost structure corresponds to this case, then we will pick a '.1' threshold. Sorry, 1 '.9' threshold because that's better, right! 24,800 is lower than 41,100 but if it's this case, we would probably pick the 0.1 threshold because that's better, right! So, it's a question of how we want to use the results, what the cost structure in our real world domain is, and that is what controls the choice of threshold more than anything else in a classification problem. So, that's it for using regression for classification. I...I urge you to try logistic regression on your own, and see what kind of results you get and then compare that with the results we got for this. Thank you.