It's. That's okay. Are we. Are we good? Yes. You had ample time to discuss. Okay. He. I remember last week I started here and I said I will start this week on this side. I do remember back on this side place. What do you have for me? Advantages. Disadvantages. Anything interesting that came up? It's very easy to interpret. Yeah, absolutely. I mean, that's very similar to linear regression. We have interpretability. It's a big one. It's actually one of the main reasons why logistic regression is so popular in credit scoring. So in credit scoring for regulatory reasons, you have to be able to explain how a model decides whether someone is worthy of credit or isn't. So you can't just use a black box model. People often think we just throw our credit scoring data into a neural network and then something nice will come out. But you can't do that in a bank because if the regulator, if the regulatory bodies come around to call and tell, tell you, show me how you calculate that and you basically give them a piece of code and set the model told me to do it, That's not good enough. You can't make decisions like that. So, yes, interpretability easy to interpret. Very big one. Do we have something in the front? Maybe like the probability like that using the logistic regression. Usually it's used to analyse the historical data to predict something in the future. So I think only with a logistic regression that's possibility to. Predict the probability of. So, yes, it's connected to interpretability, isn't it? So there's this concept of probability and we do decide on a threshold. So we decide, for example, that probability threshold should be 0.5, but we still deal with calculated the probability of, for example, defaulting for different customers. And then we just use the sigmoid function to then transform that into a zero 0 or 1 outcome. So yes, the probability concept kind of makes sense from a, isn't it also interpretation ish? You know what I mean? I put a question mark and you'll remember my words instead. I'm surprisingly not good to put my words into writing. I know an academic should be, but I'm not good at it. Okay. Do we have something maybe here in the back? I guess one disadvantage is that it requires expert knowledge to be evaluated of the results because you have to choose what would be the best threshold to use. It's either you want to minimise the overall error rate or to minimise the individual. Yeah. So expert knowledge, obviously you need that for many of your models. But we did discuss that. Choosing the threshold and deciding what kind of error to make is worse depends on the application context. So we were talking about this kind of cost insurance for banks, and you have to be able to understand how how banks, what kind of costs do occur for you. So that makes it a bit more difficult to come in as an external and basically present them with something. So yes, very good. Someone last well, I can see you. As we discussed, can be an advantage and disadvantage. Is because it's a. Um, we don't see as many. Uh, you mean outliers in terms of the outcome? Very, yes. So it's a binary outcome. So, yes, you have less kind of less variation or less. Less variation in dependent variable. Yes, absolutely right. I mean, you still have your outliers, obviously, in your in your explanatory variables, but that does improve how easy it is to actually implement a model. Absolutely right. Some. Yes. I mean, I'm not sure if it's a disadvantage, but if we don't want, the binary outcome won't really be the model, right? Yeah, it is quite restricted. So I did mention that we have multiclass logistic regression and we have all the logistic regression as kind of special cases. But if we just talk about the model we discussed today, we only have a binary outcome. So there's just there's less variation. So there's both good and bad, there's less variation. So we have two classes, which is easy to interpret a model, but it's also much more restricted. So we're not, for example, reporting really the probability of default when we're talking about this concept of probability. Earlier we reporting a binary outcome. So we don't know if someone defaults, for example, whether it was just a marginal default and next time they wouldn't. We just know they defaulted full stop. There's nothing more we can do about that. So there's a bit of a restriction in that sense. Something in the middle? Yes. We? I think you. Also mentioned this. An advantage would be slow. So it's a relatively cheap model. That's one of the reasons why. Similarly to linear regression, it's often one of the first models we train and test on our data because it gives us a good threshold. So if you have a classification problem, if you have a binary classification problem, the first model you're probably trying to implement as a logistic regression and then you can use more advanced models depending on how large your data set is. Yes, I have another one which is closely related, but I'll see whether someone actually tells me without me having to tell you. So we have something else in the middle, maybe in the back. Everything was already set. I. So I say. The variables are. We assume that the. Yes and no. It can be a disadvantage. I said something specifically about that linear relationship. Do you remember what was? Yes. So the linear relationship is between your between your output and the log odds of your dependent. So it's not a direct linear relationship between the dependent variable values and the outcome variable. It's between the probability, the odds of that variable and the probability. Does it always happen or is it. Is it a usual thing that the log of the value and the dependent variable. I think it's an assumption we often make, at least in some sense. So what it basically tells us is that, for example, the probability of default. So I say the probability of default is linear, linearly relationship has a linear relationship to the explaining variables. So it's kind of. Yes. Log odds of the. Event. To the dependent variables. So. So if we say the probability of that is related linearly to the combination of variables, that's something we can in many cases assume. Yes, it's it works surprisingly well. So it depends obviously on your application context. But credit scoring, for example, logistic regression is a really, really, really popular method. So you can probably assume that this is what is being used in practice because it's computationally cheap and it works surprisingly well. Something in the front or are you happy? I think what we can add is that it's really simple. The parameter we actually need to set the price for yourself. Yeah, I think that's also one of the reasons why it's also used in other neural networks. Yeah. So there's relatively few tuning parameters, for example, for you to actually care about and worry about. So there's some models are much more complex. If you think about neural networks, again, they are extremely complex in their tuning process compared to logistic regression, really easy to implement and easy to interpret as well. Something on the side or this time. Yes. So comparing it to linear regression, we could use categorical variables to predict here as we are using binary output as zero one. But while discussing it, we also realised that we could face the problem for multi collinearity more often than not than linear regression. Because if we are, if you are predicting categorical variables, we'll have a lot of dummy encoding and then we have to manage that too. So that can come in and disadvantage that managing categorical variables. You have to be more careful because for the auto encoding and multicollinearity, but we can predict categorical variables to yes. So predict categorical variables as in, for example, in this case a binary categorical yes, predict categorical outcome. And yes, you have to check for multicollinearity Multicollinearity. Oh, it's a terrible word, honestly. So yes. Yes, it basically means that some of the variables are linear combinations of the others, so they are correlated with each other. We were talking about I think a good example for that is we were talking about dummy variables. So if you have three colour options, for example, blue, red and yellow, then if you try to create a dummy variable for all three of those, your model will not work due to multicollinearity. And the reason for that is you can perfectly explain whether someone chooses yellow just for the information of whether they chose red or red or blue. So you can perfectly explain one of those variables through a combination of the other variables. Does that make sense? Does it mean that the variables cannot. Must be distinct from one another. Yes. Equivalent by multiplying. Exactly. Exactly that. So they have to be independent from each other. So you can't, for example, have income as one variable and then income divided by two as another variable. I don't know why you would do that, but you might. So if let's do a bit a better one. Let's say we have a variable called household size people in your household and you have one variable called income. How much income does your household generate? You can't have a variable which would tell you how much income is divided by its divided to each member of your household. So, for example, you can't have a variable which would describe value per member of your household because you could perfectly calculate that by dividing income through members of your household. So they have to be independent from each other. They can't be calculated from each other. Yes. So that's perfect. Multicollinearity. There can be. I mean, decimals. That was an example of a perfect linear collinearity. But there can be examples of imperfect collinearity and that can still present an issue in your data. So you do have to check for independence of your predictor variables. Yes, very important. And that's true for both linear regression and for logistic regression. That being said, logistic regression is a bit more flexible in terms of its assumptions made. So some of the assumptions that we make for linear regression are, for example, that our error terms have to be normally distributed and they have to have the same variance across all of our samples. So that's called homoscedasticity. I know another terrible word that statisticians forced upon me and so homoscedasticity of your error terms as well as their normal distribution. Both don't have to be fulfilled for logistic, but they do have to be fulfilled for linear regression. So logistic regression is a bit more flexible in that regard. But some assumptions like multicollinearity are true for both. Anything else on the side that you would like to add? Know everything covered by your colleagues. I would like to say it's probably not. Yes. Depends on what exactly you want to predict in your time series. So if you, for example, want to just predict the direction of your time series, that might be possible. So I'm thinking about a very simple model where you're trying to predict the direction of a stock. For example, it's a stock price. Whether it goes up or down would be a binary prediction. How valuable there would be, I don't know. So it's probably not very useful for Time series, I'll give you that. You could say about. This to be it or not, and then sum it up for a number and then use it as a time series, maybe like that. So yes. So I think there might be ways, but it's not natural fit for time series analysis. I do agree. Yes, but you could. So there's probably kind of limited applicability, maybe depending on your case due to the binary outcome and due to the linear relationship that we have with the log odds. Another thing. Yes. Why we use log odds for this. Wow, that has something to do because we wanted logistic transformation. So that's the reason that we want to transform the probabilities to fit into this kind of binary outcome using our sigmoid function. So we take the logistic transformation to make it fit our curve. That's the very simple explanation, because we remember that this is the logistic function that we calculate that as the the exponential which is taken from our from our log values as they make sense. So you can if you we want to transform that in terms of our P to be able to plot it or be able to describe it in terms of a logistic function, does that make sense or not yet? You can say not yet. I will explain it again. Not yet. Very good. Thank you. So our goal is we want. Let's wrap it up from the back. So if this is our objective, we want to get this to describe our outcome because we're interested in whether something is a zero probability or probability of one. So we want to transform it into a binary outcome. And we say this the sigmoid function, this logistic function is the perfect fit for that. So let's say we want to be able to model our data in terms of a logistic function, and the logistic function looks like this at the top here. So we want to be able to predict, to model our probability, which is here on the Y axis in terms of a logistic function. Now, how do we derive this Logistic function is actually through these steps. So this is our. Logistic function. So we want to divide it if we go one step back in our calculations. This is our linear combination here. Let me show that on the screen. So on the other side, you can see as well. So this is our linear combination of intercept and explanatory variable here. And you can see here that we have the exponential value. That's the it's exponential is the right word for it, right? Because I have German and English, it's it's very difficult to do that in two languages. So if we want to be able to describe this just without the exponential. You have to take the lock of the other side of your equation. That's why we take the lock of the odds. Does it make more sense? Good. Glad. Yes, Yes. We can have a break now. Sorry about that. Do come back in, let's say five minutes a day. Is it okay? Good. Thank you. Okay. Okay, everyone settle back down. Settle back down. So we will continue with regularisation. I also want to talk about model tuning, data splitting and resampling. If we don't get into addressing imbalance today, I will move that to the next lecture. So don't worry, we will cover all material. But it might just be that I move things because apparently I talk too much or something like that. Um, before we set, before we start that, there's one more thing I wanted to add to the advantages and disadvantages part. We guarding low computational costs. So one of the advantages of logistic regression is actually that it works quite well on relatively small samples. So you don't need a huge sample for logistic regression or linear regression to work compared to more complex models. The reason for that is because the relationship we assume is relatively simple, So we don't need a lot of data to explain a simple relationship. So if you have small sample sizes, some models work better than others. So logistic linear regression of some of those models that work very well for small samples. And then we have things like random forests, neural networks and svms which work better for larger samples. So. Depending on your data situation that will affect obviously, which model you want to choose. So I wanted to add that as another advantage. Okay. Let's talk about regularisation. So we have a couple of things that we were talking about when we decided how do we actually fit our model? What do we care about while we're trying to fit our model? And in this case, when I say fit, I mean estimate our better, our better values in logistic regression, for example. So one of these factors is obviously prediction accuracy. So here it says a really important part. If N is not much larger than P and here I say N number of samples or observations and P is the number of predictors. So basically how many X values you have, there can be a lot of variability in the least. Squares fit obviously. So if our sample size small especially small compared to our number of predictors, then this might result in overfitting and poor predictions in our modelling later. And on the other hand, if the number of predictors actually larger than the number of observations and yes that can happen, then the variance of your least square fitting method is infinite. So you can't use least squares approach at all. If you have more predictors, more columns in your X, for example, then you have samples and that's actually a pretty big problem. And we will see later what you can do in these cases. Some of these approaches include, for example, choosing which variables you want to include, so forcing a smaller number of predictors mostly. So we can often try to reduce that variance by methods such as regularisation regularisation of our model. So the other thing is that in many cases we care about our interpretability. So this was one of the things that we've mentioned both for linear regression and logistic regression. Being able to interpret your results is really, really valuable in many cases. So if we have a lot of variables, a lot of predictors in our model that can help us to unnecessarily have a too complex model, which makes it much more difficult to explain what is actually happening in that relationship. So again, in that case, we want to remove a couple of those variables to allow us to be able to interpret our model more efficiently. So this kind of idea of feature selection or variable selection, the basic idea is we want to remove irrelevant variables, or at least those variables which don't add a lot of explanatory power to our regression model. And one way we can achieve that is for something called a lasso regression. A lasso regression is actually a relatively simple approach where we say we want to penalise the number of predictors that we have in our model. So it's a similar idea of penalising the number of predictors in your R-squared. You remember we talked about R-squared values and adjusted R-squared. Here we are also penalising the number of predictors that we have and while we are optimising our model. So the first part of this you will remember from your least squares estimation. So if you have a multiple linear regression, you'll remember we want to minimise the error that we're making which can be calculated as your actual value minus. And then this whole second part here is basically your estimation. So that's your beta hat. So this term would be your error. So your squared sum of errors. You remember that as what we want to optimise in least squares. And now we add this red term here at the end. And what that does is it counts your number of predictors basically, or it counts how strong or how many predictors you want to be able to include in your model. So the important term I want you to focus on here is this lambda. This lambda is the strength of your penalisation term. So the higher the lambda, the more each additional predictor actually increases the overall term. And if we want to minimise that quantity to optimise our model, you can see that each additional lambda would make that term worse in terms of minimisation. So each sorry, each additional better would make this would make this whole term worse if we're trying to minimise it for our optimisation. So you can choose this lambda based on, again your expert knowledge on how I know, I know how complex you want your model to be. So we were talking earlier, you if you have more predictors than observations in your in your model, then it doesn't really work. So in that case, you would want to force the number of predictors to be at least lower than your observations. So that's one way that you could choose your lambda. And so. The objective of a lasso regression is really that we want to build basically a perfect or a better multiple regression model, which only includes the most important predictors. So this is the objective of the lasso regression. So it's still a least sum of squares approach, but we still minimise, but we now also minimise the number of predictors that we have. So this can actually force some of our coefficient estimates to be zero if that lambda is sufficiently large. So if we have a large enough lambda then some of these betters that we're summing up here will have to be zero, which effectively removes a variable from your from your record. So why it is your number of of x and betas. You remember all your coefficients for each of them. So if you want to find the perfect set of betas while taking into consideration that each of these betas is making our result worse, then obviously some of these betas will be optimal at zero, and if you have a better of zero then the variable will be removed from the model. So yes, selecting a good value for that lambda is important. We usually use some kind of cross-validation, so we check multiple lambdas and then we choose basically the best model in terms of some kind of error measure, accuracy measure, etcetera, etcetera. So you're evaluating different different values of Lambda in terms of model performance at the end. But you also keep in mind that the number of predictors, for example, has to be low for interpretability or for being lower than your number of observations. Yes. It will eliminate several variables. Exactly. Yeah, exactly. So you can actually remove multiple variables at once as well. So depending on how how large their betters would be. So even if the better value is still very high, it wouldn't be removed from that model likely because it is still a very influential variable, but it would be kind of compressed. So you just I always imagine them to be kind of big, compressed a little bit, and some of these variables hold on strong because they are still really important and some of them are just compressed so far that they disappear. So I always think about Lasso as kind of a compression method. I think the name can also be thought of as this kind of lasso as a rope and you catch the most. Important. Variables if that's how you want to think about it. But yes, it can remove multiple variables from it. So, yes, an alternative formulation of that would be this term. So we are minimising this is still your error. So we are still minimising our squared errors in terms of by adjusting our batters subject to our sum of batters should be smaller than some kind of threshold that we set. So this is a similar formulation just with a with an S parameter instead of a lambda, but it has the same the same. Outcome. Yeah, the same thing happens. It's just an optimisation minimisation formulation to that. Oh yeah. And obviously you're better. Values may depend on the scaling of the different predictors that you have. So always apply lasso regression after standardising your predictors. That's a good rule of thumb for any kind of optimisation that you do. So do start typically with standardising all of your data because that helps the model to actually fit and find the optimal betters. There's an alternative formulation to that, which is rich regression. And so this actually penalises all of the coefficients towards zero, but it will not set any of them to exactly zero. So it's a bit of a softer approach. It's still kind of compresses everything, but none of the values actually disappear. So this is not suitable for variable selection or feature selection, but it can still be useful for interpretation of your model because it kind of still compresses some of those values down. And then elastic net is a combination of of the two. Um. Yeah. So I think lasso regression is basically a more prominent method. So this is also the one we will be implementing in the computer lab tomorrow. Related to that, let's talk briefly a bit about model tuning. So. We talked about that last week actually, when we were talking about how some models can overemphasise patterns. So they are a bit too sensitive to different patterns in the data if you're not careful with your tuning. So with tuning, what what we mean by that is basically the selection of any parameter. So your lambda, for example, would be a tuning parameter in your lasso regression because it's a parameter that you can shift and change and try different values, which has an impact on your parameter and how on your model and how well your model fits and flexes around your data. So we will talk about that multiple times through the lecture series depending on what kind of model you have. You have more or less tuning parameters. So more or less opportunity for you to take influence on your model, for better or for worse. We also talked briefly last time about bias and variance as these type of two statistical concepts. So if we know that the true relationship between our independent variable is described as some kind of model, so F of X here can be, for example, your linear combination that we talked earlier in linear regression. So this is any type of model and any type of error for that to describe your bias and outcome, then we obviously want to find some kind of model. We want to estimate some kind of model F hat to approximate the true function F of x, which is a perfect description of the relationship. So we estimating the best possible model to represent to a relationship. This is actually really important concept to understand that models are only ever the best fit for the reality we're trying to describe. So no, no model will likely ever be able to perfectly describe a situation. We're just trying to find the best model possible given the data that we have and given the other restrictions like computational costs that we have. So you already know that we want to achieve that typically by optimising or minimising our error, which is just our deviation of our estimation from those two values that we have. Now something that we haven't talked about yet, but you might have heard, is that we typically compute our error first using training data. So we we split our data set into different parts. And part of that is used to, for example, estimate your better values for your linear regression. And then what we are really interested in is how well does this model then perform on other data that it hasn't seen so far. So when we're trying to predict something, obviously we can't train the model on something that happens in the future. So we're trying to know how well will the model perform on unseen test data. And that's why we calculate typically errors both on the training data. So how well does our model fit on the data that we know it's supposed to fit? Because that's what we're training it on. That's what we're showing to the model. But then we're also interested in our test error. How well does the model then fit a new situation which it hasn't seen before? And then typically we often calculate the average of those or we we report both values for both training and test data. So we are always interested in both errors of that because one is describing how well do we fit and the other is describing how well do we reform in the future. Yes, this is a bit of a complicated way of explaining what is bias and what is variance. So bias is capturing whether we are truly predicting what we say we are predicting. So how well does our model actually perform in finding what we are trying to explain relationship wise? And then the variance is how reliable is that model, for example, in the future or if we show it, if we apply it multiple times. So how much variation do we find in our prediction? And bias is basically how well do we actually find how what we're trying to predict variance is how likely are we to find the true value again and again and again if we want it multiple times? So both of these are really important because we are trying to obviously do both, right? So we are trying to actually predict, for example, defaulting customers. And we also want to be able to predict default and customers again and again and again with the same model. So we want low variance, low deviation from that true default and we want low, low bias. We actually predicting default. And obviously because nothing in life is ever good and free, it's typically impossible to do both equally well and equally perfect. So we typically trying to kind of balance between those two and that's what we call the bias variance Trade-off And so high variance learning methods, for example, that means they are able to represent a training set really well. So they are always kind of really pinpoint in the middle and exact, but then you show them something new that they haven't seen before and suddenly they are not able to represent that well because they are kind of too focussed to laser focussed on this and repeating the exact same action with precision again and again. And similarly, augmentation algorithms as high bias can produce simpler models because they are kind of able to understand the concept a little better and describe well what we're trying to do. But we might, but we might underfit for our training data. So we're kind of two, two vague in our in our modelling. So what I want you to take away from that is this idea of overfitting and underfitting to data is always related to this idea of bias and variance trade off. So whether we fit too closely to our training data and we're not able to then predict anything else in the future or whether we are two kind of too loose fitting and we're not able to predict and predict well any of those values that we have. And I think this is very similar to what we discussed last week when we're talking about finding the best fit line for your data. So you can see here on the left side, this is something that we would call under fitted model. So you can see the real relationship of the data is kind of going this way here, but we're just fitting a linear regression apparently to it, which is a straight line through and we're not really capturing any of that. Well, do we? It's like we're kind of capturing a little bit here, but it's not really a good fit model. So this is under fitted model on the right hand side, this is the other extreme. So we have a really overfitted model here. We're following every single small change in the data. If we try to fit this model to other data which has a similar behaviour but not really so kind of new points popping up here, for example, or down here, then that model would not be able to capture those because they are too far away from this over overfit. So what we're really looking for is kind of somewhere in between. It's not perfect, but it's pretty good. So we're getting the general kind of non-linearity of the relationship, and this is the perfect compromise between these kind of oversimplified models and these overcomplicated models. So as a general rule, more flexible models means that we have a higher variance and a lower bias. And then the other way round to strict models, then we would have a lower variance and a higher bias. And the relative change, the relative rate of change of these two quantities determines whether our test error increases or decreases. Because the reason for that is if we train the model on this type of data, what happens if we add new data to the model? Are we still able to predict new data points popping in or not? That is an odd plot. I don't remember putting that in there. Okay. So this is very similar to what I just said. So we are plotting in terms of flexibility of the model. So this is basically the first sentence. The higher flexibility we will have increase in variance. That's your orange curve and we have a decrease in bias. That's your blue curve. And then what we are basically looking for is kind of this optimal spot here. And this is what minimises our error on the test data, which is the red line. Okay. Um, yes, Let's talk a bit about data splitting in the last couple of minutes and then we will talk about sampling next time. So I was just talking about training and test data as two concepts. So given that trade off, we've seen how we typically achieve the best trade off between the two is we divide our data into part that we use to train the model to estimate our better parameters. And the part of the data that we use to test the model to see how well it performs with an unseen data, whether it's, for example, underfitting or overfitting. So training data is used to build and tune the model. And then we use a test data to estimate our model's performance. So a couple of modern, more modern approaches actually split the data multiple times. We will see that when we talk about things like cross-validation and resampling, because sometimes that helps us to fit better to both training and test data. Especially true if the data, for example, has some classes being underrepresented, or sometimes we have data which is quite which is quite split between locations. So if there's some kind of pattern or structure in the data, it usually improves your model. If you resample multiple times and split multiple times into training and test. So typically what we will do is we split data by 70% training and 30% test. That's kind of the rule of thumb most people follow. That is a rule of thumb. So you can pick any kind of value that you like. Some people use 6040, some do 5050, some do 8020. It doesn't really matter that much. The point is you want typically more training data than you want test data because you want to accurately train your model on that part and then you want to evaluate it on a smaller part of the data just to check and report the error of that. Some models also like to have an additional validation set in here. So you have training data for training your model, and then you have hyperparameters, for example, neural networks that you want to tune and optimise a bit more strongly. So you have another data set and then you test on your test data at the very, very end. And that's just a different approach to the same logic. The bottom is used for more complex models. So neural networks, for example, you typically use bottom. If you use linear logistic regression decision trees, it's usually the top. So you typically just have training and test data really, really important. And I never, ever want to see anyone do that. If you show your model test data, you're done. You're not doing any more tuning or kind of rerunning or experiments afterwards. The reason for that is as soon as test data is seen, it's information that's in the model and you can't call it an independent out of sample test data anymore. As soon as he has, you've used it for any step in your model. So as soon as you do some kind of tuning on the test data, it's not test data anymore. It becomes training data. Okay? So don't ever do any optimisation on your test data place. The simplest way to do that. We split randomly random sample into training and test sets here. Test validation sets. I would say test sets in this in this case, in some cases where some some classes have smaller frequencies, we do try to replicate that distribution, for example, by oversampling under represented classes. We will talk about that a bit later. And an interesting thing that we want to talk now is resampling. So the most famous approach to cross-validation and resampling techniques is k fold cross-validation. The idea here is that you randomly partition your data set into k sets of equal size. So if you have 100 data points, for example, and your K is set to ten, then you would have ten sets of ten data points each. And then you fit your model so that your training data to all of those except the last one. So nine out of those ten would become your training data. And the last kind of set of ten elements would be your test data.