For you. Because. You told me. Okay. Good morning everyone. How are we all holding up? Because it's it's getting very cold. And I do remember during the first week everyone was already complaining about the weather. So how are we doing? Don't. Yeah, okay. Don't worry. It's not getting that much colder in Scotland typically. So you can expect temperatures around zero throughout the winter. But it's not getting much into the minus degrees, so don't worry too much if this is already very cold for you, especially in the mornings. So wrap up warm. I see you already wearing your jacket in class. That's not a good sign, I hope. Is it very cold in here for you or is it okay? You're just cosy because it's early. Okay. Fair enough, fair enough. So I originally planned for this week to be a guest lecture, as you might have seen on our lesson plan. Unfortunately, I got an email last week by the guest speaker that he is unable to join us today, so he asked whether he could reschedule, which is why we're kind of swapping two lectures around. So this is originally Lecture Week seven. So we'll talk about clustering this week. And then in week seven, instead of talking about clustering we'll have the guest lecture. So it's just a swap. And next week is your reading week. So we won't have any lectures or any computer labs next week. So that will be your chance to catch up on any reading or your assessments or your assignments that you're doing anything like that. So please don't come to lecture next week because I won't be here and I think it will not be as much fun without me. I mean, I guess you can come in, but no one will be there, so it doesn't really make sense. Let me close the door for us. I say so because we're kind of swapping things around a little. We you will see that the computer lab that we're doing this week, we'll focus more on the data splitting, resampling steps that you should take. And then we will do the clustering computer lab in week seven when our guest speaker is talking, because we won't really have any new material to cover in the labs anyway. So that's a good ask. As good as a time as any to do our clustering in. So don't worry if you have a look at the computer lab notes. And that kind of only related to half of this lecture. Okay, so in our last lecture last week we talked about the evaluation of regression models. I hope you all remember. So we were talking about simple multiple logistic regression and how we evaluate them. There was this whole kind of matrices where we're looking at these different types of errors. And we were also looking at our C curves. And we briefly talked about regularisation. That was this idea of using lasso or which regression which allows you to actually do variable selection while fitting your model, which is very handy if you have a lot of variables that might be suitable. And then we also started talking about data splitting, resampling, specifically cross-validation. And we wanted to talk about class imbalance handling. So because I mismanaged time last week massively we did not manage to cover that. So I promise we will do that this week. So we will start with a recap of anything that was a bit rushed last week. So that means we'll talk briefly about data splitting and cross-validation again. And we'll also talk about class imbalance handling, which is really important in many applications, especially in credit scoring. So in many cases you only have a few cases of one particular class. So predictive model, if you just train it on this type of imbalanced imbalanced data, we won't be able to learn to predict both classes evenly. So what might happen in those cases is that the model just predicts the most frequent class, because in most cases that is the correct prediction to do. But it also means it's not very suitable of a model to use to predict both classes evenly. So in those cases, we use a couple of different methods to oversample or under sample our data so that it's more evenly spread. And then the big block today will be unsupervised machine learning. Yay! My favourite topic. So we'll talk about clustering algorithms. We'll focus on partitioning clustering and hierarchical clustering. I will do a brief mention of a couple of different other methods that might be interesting for you, but there will be further reading if that is something that interests you. So I have a couple of book recommendations as well, and recommended reading. This week is only the Witten and James. You will notice that our good friends Kuhn and Johnson don't have anything on clustering to say in their book, which is fair enough. You will see that clustering is a bit of a special method compared to the others, especially in predictive modelling. So that's why it's not really mentioned in all of the predictive analytics books that you will come across. The reason for that is that most of those focus more on supervised regression and classification methods. Okay. Let's start with this very important slide that we had last week on data splitting. So you will remember that in the past we've always trained our model on basically the whole data set. So we had our whole data set. We looked at all the variables that were available. And we had some kind of output value, a class label or a number, something like that. And then we try to build a model with the whole data and then predict the whole outcome. Now the idea that we're introducing now is that instead of building the model and the whole data set, we're actually splitting our data into two parts. One will be our training data. That's what we're actually building the model with. And the other is our testing data. And that's the part of the data that we're testing our model with. The reason why we do that is if you fit and tune your model on your whole of the data set, then sometimes your model can become very accustomed to the data. So it can only be able it will only be able to predict those specific cases that it's learning through your data set. But what you really curious about is how well does your model perform with new data that might be coming in the future, for example? So we want to see how well our model able to generalise from the rules that it learned through the training data onto unseen data, which is our test data. Now, depending on how complex your model is, you might do a three way split where you're training a model on training data, and then you're doing your actual parameter optimisation and tuning, which can be a lengthy process on validation data. And then you have your test data as a holdout sample at the very end that you are then just testing on. So the important thing that I mentioned in that context is that your test data should be unseen. What I mean by that is that you should try to avoid any of the information from your test data spilling into your training process, because as soon as that happens, you can't really evaluate how well your model is performing and unseen data because it's not unseen. It's kind of vaguely seen or parts of the information has leaked or spilled over. So that's why it's important that your test data should be held completely separately. So what that means, for example, is that in most cases we start by splitting our data, and then we do our pre-processing on both of these data halves separately. And what that means is, for example, if you do any transformations or class imbalance handling that we will talk about later. All of these steps should be done separately for your training and your test data. And that really ensures that there's absolutely no spill-over of any parameters or any kind of factors that you choose between the two. Now with k fold cross-validation, we're kind of taking this to an extreme level. So in this in this case we're not splitting into two sets. We are splitting two sets training and test data. But then we take our training data and split that even further. So we basically do the same thing again with our training data set. We ignore the test data that's kind of saved somewhere safe and you don't touch it. You take your training data, split it into k small sets K is a parameter chosen by. You mostly will depend on how good your computational power is, because the more k, the more expensive it will get. But you put you separate it into k sets of roughly equal size, and then you fit your model. So you train your model on all of those minus one. So for example, if you split it into five parts, then you train your model on four of those and you test that kind of fitted model on your fifth part, and then you do the same, you put it back, you kind of restart the whole process. You take another four sets of those and you test it on the fifth again, and you repeat that process until each of these parts had chance to be both in the training and in the test part of the data fitting. And then you repeat that process and you report typically the evaluation metrics of each of these rounds. So if you have for example k is equal to five you do five splits. Then you should produce five accuracy measures for example. And then the end result that you report would be the average across all of these five measures. What that looks like in pictures is like this. You have your whole data set. You split it into this case three parts. You build your model with two thirds of that and then predict on the on the last third, then the second round again you build on two thirds, predict on one of the holdouts and another third round two thirds and one holdout. Now, I was asked last week, what did we actually do with the test data, because you mentioned you shouldn't use the test data in your training part, but this is kind of doing that. But remember, we had this holdout test data at the very end and the very beginning that we didn't we didn't touch. So that was not part of this whole training and testing. The way I think about that is these kind of evaluation measures that you're producing within the k fold sample. It's kind of an interim result. So I think of that as a kind of ongoing interim evaluation measure of accuracy, for example. And in the variant you still have your completely holdout test set. So you can still then test the model that the actually. Fitted model on that, where we hold our sample at the end and report that real kind of hold out result. That being said, these interim measures that you do produce, because the model is built new in each round, it's still unseen data for that model round. So it's still a valid measure to report. So you can still report that. And it's not training. It's not kind of an in-sample result. It's still an out of sample result for that specific round of your cross-validation. We also briefly talked about time series data. This comes back to the idea of how do we split our data. You can do that randomly. In most cases you will do random k folds, but in some cases it makes sense to make sure that you preserve, for example, the time series structure when you build your samples. So I mentioned that time series data is special because the observations are dependent on each other. So they are not independent observations because each observation depends on the previous one or ones. So therefore if you then split the data, it doesn't really make sense to randomly sample observations. Instead, you should take blocks of that time series so that you preserve the temporal structure of that data. Similarly, if you have spatial data, in many cases it makes sense to choose specific locations and keep them together instead of kind of randomly sampling all over the place. Because that way you can preserve the dependency structure. Yes. So there is also a way of kind of choosing K in such a way. They are respectful of the groups that you have in your sample. So that becomes interesting if you think back to your principles of data analytics lecture when we were talking about stratified sampling. So if it's important for you that the model performs equally well on different subgroups of your data, then you will want to do a stratified sampling to form your K groups. In this case, and there's a number of alternative resampling techniques because we just have to do something all day. So we invent new, better techniques and methods that we think are better. And one of those is, for example, leave one out cross-validation. That's an extreme case of k fold cross-validation, where you k is equal to the number of samples that you have. So you would actually create. For example, if you have 1000 records, you would do 1000 k cross-validation where you train a 999 predict one train and 999 predict one. And repeat that. And you can imagine how expensive that is if you have a large data set. But you can also imagine how very accurate it is if you have a smaller data set. So then we have bootstrapping, repeated training test, splitting all of these kind of things that you can do where you kind of change the sampling process in your cross-validation. So bootstrapping for example, is where you sample with replacement. So you actually take kind of your sample for your k fold cross-validation. And then you put all your records back in and sample randomly again from that. So records can repeatedly be in the test or training part of each of your folds. And all of these are kind of very nicely summarised in your textbooks as well. So if you're curious to read more about them, have a look. There. And we will also implement k fold cross-validation during the computer lab as well. Yeah. So this is a bit of a brief. Image of this parameter tuning process, which also comes back to this idea of. Where does the test data come into our process? So typically we first define some set of values for our tuning parameters. This is arguably one of the most difficult steps because how do you know what is a reasonable assumption to make from where you want to start your tuning process. And it will also affect how efficient you are in your tuning and how well you actually achieve the optimal solution. The very common approach is to go into the literature and have a look at similar studies, what kind of parameters they found, and use those starting parameters because they're already relatively good estimates. If there are no similar studies, you have to go with a bit of a gut feeling and a bit of a bit of expert knowledge for what a reasonable diameter might be. In many cases, you would also choose just one small, one medium, one large, and then check how they perform. So they give you a bit of an idea where you might be looking for the optimal range of parameters, but it's a bit of voodoo up there, so it's very vague and you're not really sure what you're doing. So I think this is actually one of the most difficult steps. So we we then do is for example, we do our for each of these sets of parameters that we think might be a good fit. We would sample our data, fit our model, and then predict our holdout. So this is the main kind of repeated process that we're doing. For example, with our k fold cross-validation where we are repeatedly fitting and checking how our parameters are performing in that specific model. And then in the end, you would have some kind of performance profile. So that would be, for example, your different accuracy measures that you achieve with your k fold cross-validation. And then using that you can determine the final optimal tuning parameters. And I put them in kind of quotation marks because we will talk briefly about that tomorrow in the lecture, that finding optimal parameters can be very tricky. In many cases, all you will really achieve is finding a local optimum, not a global optimum. So for example, a lot of these optimisation parameters that we used, like gradient descent, are prone to only finding local optima in our parameter space. We will talk more about that during our lecture on neural networks as well where they become very important. So in the end, we hopefully have a set of reasonable parameters with which we can refit the model on the actual training set. In that case here, they propose to use the entire training set instead of all the k folds. And then we can use that to predict our test set in one final round, which would then give us our kind of final prediction value and our final accuracy value for that particular model. So this is the only time where you would actually use your test data. Addressing imbalance. As I mentioned, this is really important and it's mostly a problem in class prediction. So the frequency of your classes in your sample has an impact on how well your model is able to be trained on these different classes. So if you have one or more of these classes and they have very low proportion, so they're very rare in your training data, then this can affect how well your model is able to predict or predict those low occurrence classes. So be very careful if you encounter them. They are very frequent in actually a lot of applications in marketing. We often have them when we're trying to predict customer churn, for example. So in many cases our customer, our sample of customers has a lot of customers who are staying on and very few churn. But the churn is are actually the ones we're interested in. So we're trying to predict who might leave the company to go to a competitor. But if we don't have a lot of those data points, it's difficult to train the model on credit scoring. Similarly, you will have noticed that in your in your group coursework, for example, you are a defaulters are rarer class than your others. So training your model on that to to be able to predict that is quite tricky. So if you already know there is a class imbalance problem, then you can obviously select your data in such a way that this doesn't occur. So if you know that journals are rare, then you will try to, for example, do surveys more frequently with journals so that you have more data on those. And that class imbalance doesn't happen from the get go. That isn't always that isn't always possible. In some cases, these people just don't exist, so you can't ask more of them. And in some cases, you only really get to know that problem after you've already done your collection. So you can't go back to the data collection stage. In that case, you will do something called either downsampling or upsampling the data. So with upsampling, you basically add more data points, specifically more data points of more rare classes. And with downsampling, you reduce the more frequent class. The goal is just to even out between the two. So it doesn't really matter which one you choose, as long as the. In the end, your classes are relatively even. They don't have to be 100% even, but kind of relatively even, so that you give the model a good chance to be able to learn both equally. Now, which one you choose also depends on the data set size that you already have. If you have a very large data set, then creating more data points might make it too computationally expensive versus if you already have a small data set which using it by downsampling is probably not very clever either, because then you're just reducing how well you can train your model. So here are a couple of examples for what that would look like. You can see on the very left that would be our original data. You can see our two classes red and blue. So there's class one and class two. And you can see in our original data that we have a couple less blue points than we have red points. And they are quite a distinct group in that they are kind of in this space. So they are a group of interest. We want to be able to predict this class two. Now there's two ways we can do that. We can either downsample the red points. You can see that here. Now we have fewer points overall. And we have roughly equal numbers of red and blue. Or we can upsample the blue points. That's the next picture here where you can see we have the same number of red points as we originally had, but now we have more blue points. And they're kind of overlapping a little bit here. That's why they're darker. And then there's hybrid methods which are doing a little bit of both. So we're doing a bit of upsampling and a bit of downsampling. And we're kind of meeting in the middle. So one of the possible ways of oversampling is relatively popular. And it's the acronym is Smote, which stands for Synthetic Minority Oversampling Technique. Or you can just call it Smote because that's kind of it sounds a bit like like Lord of the rings, like the dragon smoke like smote. That's how I imagine it. So it's a data sampling procedure, which is kind of one of these hybrid methods. So it can do a bit of up sampling and a bit of down sampling depending on what you what you can, what you need. So if you implement that you will see that you can specify how much you want to do, the up sampling, how much you want to do the down sampling. And very important you have to specify the numbers of neighbours that are used to impute new cases. So with down sampling is easy. You basically delete a couple of cases up. Sampling is more tricky because you want to create artificial cases of the rare class, which are still representative and accurate for that class. So what we typically do is we use something like k and n k nearest neighbours where we use the existing cases and then create create new cases which are close to them or similar to them in some way. So in this case, for example, here we have new synthetic data is a random combination of the predictors of randomly selected data points and its neighbours. What that means is you have your blue points. So you know what kind of parameters these blue points typically have. So in what space they are located. And then you can create more by using a combination of these predictors or these parameters of your blue cases. And create a couple more that are similar and close by to them. So we call them neighbours because in a plot, in a scatter plot, they would be neighbouring points because they are so close. And yes, Smote obviously can also downsample cases from the majority from the majority class. And it does so by random sampling. So let's just select randomly a couple of the points and deletes them basically. Okay, so that was our quick look into data splitting, sampling and imbalance handling. Are there any questions about that part? Yes. Synthesising new cases. So it's like adding data points. Exactly. Predicting kind of isn't that? It basically is. Yes. We'll actually look at Kane as one of the techniques. So K and here is used to create these new data plots. So we will talk about that as one of the prediction techniques in a couple of lectures. Yes. What we're trying to use is data to study a prediction model. And we're predicting the data. What's the point of studying the model. Yeah. So I mean, we are basically building a model for parts of the data. So we only really creating it for this kind of blue blue part. But if you for example, what you're not creating at the same point, at the same case for the red. Yeah. So that's, that's why it's kind of predicting for parts of the model and then predicting for the whole of the data set. Yeah, like I suppose there is some bias if we use. A technique for the upper. Yeah. And then. Yes obviously. So suppose our data is. Uh, that's not so smart. Yeah, that's a good example. So you can already see this a little bit here where the data points are overlapping. So a lot of them are not really new data points. They are just the same data point of almost the same data point kind of created in the same space for the predictive model. That doesn't really matter that much. So the problem is that. You are not really creating new information per se. You're kind of replicating the information that you already have, but for the model, it doesn't really matter because it's still kind of taking those cases as if they were real. And so you're not really introducing too much bias that way because you're not creating false information. You're kind of recreating existing information that you know is real. But you're right. This is a big problem if you have a very, very small class. Because in those cases you don't really know which of those should be an outlier. For example, if you only have five records, you don't know which of those are real, which one of those are outliers. So you might accidentally oversample then outlier classes, which might bias your data. Yeah. So as you mentioned, with the nearest neighbour approach, there is a chance of error. Oh, yeah. Is there something that we can do to minimise that or. Not really, because you don't know where the data should be realistically. So the only thing you can really do is use kind of. Just justifications, critical thinking and expert knowledge to think, is this a realistic data set that might occur? But there's not really much you can do apart from looking into more data that you might or might not have. And also one question is nearest neighbour more appropriate for clustering data. Because you look at the nearest points to. So, but it's also as effective for predicting because it is the same thing, right? Yeah. So data points to figure out. Um, where the data should be? Basically, yeah. Yeah, exactly. So that's why we talk about K and N, I think in the next lecture, which should be week eight because of the shift. So you're right that it is basically. It's usually used for classification purposes, so that's its main purpose. The reason why we use it here is because it's quite powerful in creating unknowing, where more data should be based on the existing data. So these kind of additional blue points that we're creating here, we're just basing them on our known neighbours that we have a big problem that you'll see with K and N, it's actually that it's called k. And the k is a parameter that you get to choose. So it's another kind of researcher chosen parameter that you're adding to your model. And with every choice that you make, every parameter that you choose, you're introducing a little bit of subjectivity into your model. So depending on what k you choose, how many neighbours do you consider in your creation of new data points that will affect your model as well? So it's yeah, it's a bit tricky, but sometimes it's the only way that you really can achieve the data set that you need for this case. Yes, it's about a k for cross-validation. So how we choose k value in different. Yeah, exactly. So that's UK again. It's always tricky. Okay. How many folds should you do? So the more folds you do, the more accurate it will be. But it will also be computationally more expensive. So depending on how big your data set is you might only do three, five, ten. If you have a smaller data set, you might do 50, 20, 70. So there's not really a rule that you can choose. It will depend on your data set and will depend on how powerful your computer is, basically. So you can actually test that during the lab if you want to, because you can choose like three folds, have a look at the accuracies and then do 20, have a look at the accuracies and test whether your computer crashes and whether you managed to bring down the whole server of the building, I don't know. And so you can test that. That's not really a fixed rule. It depends on kind of your application. If you have very varied data. So there's a lot of kind of nuances in your data you want to use more because then it kind of catches all of those. But if you have a large data set, you have to choose less because it's too expensive. Okay. Get. Let's have a look at clustering. So so far in this course all we've really talked about are supervised methods. You will remember back in the very, very first lecture, we talked about the difference between supervised learning and unsupervised learning. So supervised techniques and what we've learned so far uses some kind of training set. We build and tune a model. So we estimated parameters. So for example of your batters and your regression models. And you use that tuned model to predict some kind of outcome a class or a numeric value something. So you have some kind of input data. You have some kind of output data. And you use an algorithm to learn kind of the mapping function. So how is x connected to Y for example through a linear combination like your logistic or linear regression. Now when we talk today unsupervised techniques they don't use separate training and test data. And the reason for that is we don't have an output Y. We don't really have any class or value that we're trying to predict. All we really have is one data set. You can think of it as one x. So one whole data set. And we're looking for structures within that. And that's kind of a vague term. So I always think about that visually I'm a very visual person. So I always think about this kind of data scatter plot that I have. And I'm interested in kind of these, these clumps, these lumps and these structures of data. So where is the data coming together? Where is it more separated. Where are kind of the holes in our scatter plot? How is the data distributed over the whole space. So we're looking for patterns. And that's why unsupervised learning is really within the area of machine learning called pattern recognition. So we're looking for patterns in the data that occur without really trying to predict any kind of label. So you might wonder, okay, what does that have to do with predictive modelling? Because we're trying to predict something don't we? But sometimes what you're actually trying to predict is this kind of structure. So you're trying to understand the groups because that tells you something about the data in the long term, which is all that prediction really is. You remember we were talking about what kind of prediction. It's using information to make decisions about the future. So we can do that with clustering. So here's a couple of different examples that we have. So we mainly use it to identify similar data points. Similar data points are these lumps in the data for example. So in customer segmentation clustering is very often used to identify groups of customers who are in some way similar. So they should then be treated similarly in the future by the company. And you can also use that to, for example, predict the behaviour of the future. If you have, for example, your customer segmentation, you have your different groups of customers. And, you know, one of these groups kind of has a high chance of churning, then this is already predicting their behaviour into the future. So you already know what does a churn in customer look like. Because it's this group and this kind of selection of parameters of that group, which seems to determine churning behaviour for the future. You can also use it in spatial data. So what I do in my own research, for example, is I often try to identify subpopulations in different regions of a country who are behaving similarly in terms of their health or their financial well-being, their buying behaviour, the purchasing behaviour, all of these kinds of structures. And the way I do that is to then create policy recommendations for these regions of a country, because I can tell policymakers these regions are likely to experience in the future more financial hardship than others, for example, or other people are using it to create maps of locations affected by events like flooding or wildfires. And we can also use that for time series, where we look at how time series of different stocks, for example, are behaving similarly. So we're kind of creating these groups, these reference groups of specific stocks. And then we that way we can then use to predict the stock behaviour of some stocks because they are similar to others in that reference group. So we can kind of use that in prediction terms as well. So let's talk about cluster analysis because that is the big the big topic within unsupervised learning. So it's often treated synonymously with unsupervised machine learning. That's not particularly accurate. There are other unsupervised learning methods. Strictly speaking principal component analysis which we'll talk about soon, is also unsupervised machine learning. But most people, if they say unsupervised they mean clustering. It's a big one. So we will cover today two main groups of cluster analysis algorithms. And they are partitioning approaches. So you will be introduced to K-means and K mediate. And we'll also look at hierarchical clustering. Those are divisive and agglomerative methods. There are other approaches which we won't cover today due to time I'm conscious of time. This time I will not go too far. So we will not talk about density based approaches, graph theory based approaches, and probability based approaches. All of these are really exciting. So if you want to talk to me about them, please do, because I'm very excited about those. Especially density based approaches are very interesting because they use this kind of concept of neighbourhood again, which is very similar to the k and n thoughts that we had earlier. So they are really interesting as well. They also have a bunch of advantages because they able to identify outliers. So you will see that other methods will see today K means hierarchical methods then is really able to identify outliers while doing the clustering. They have to be captured doing pre-processing DBscan for example. This density based approach here is able to identify and label outliers automatically doing the clustering, which is really convenient. So it kind of reduces your preprocessing time. Okay. So I mentioned this idea of similarity and closeness and neighbours and all of that multiple times. This idea of distance and dissimilarity is key in clustering. If we want to identify groups of points or time series or areas, anything like that. Which are similar. We have to define what does similar mean in our context. So some people use similarity. Some people use distance. In this context they are basically synonymous. And which one you use depends a bit on your application context. I typically say similarity unless I'm talking about spatial contexts and then I'm talking about distance. So we'll talk about the similarity measures for numeric data and specifically continuous numeric data. There are two big ones that you typically encounter. I hope you're all familiar with Euclidean distance, because that is the most basic distance measure that you can think of. And we are just for example, if we calculate the distance between these two vectors that we have here. So you can imagine them as two columns of your data, or they could be two rows of your data depending on how you switch it. And they are basically two observations. Two people, for example. And you're trying to determine how similar are these two people depending on these different factors that we have to describe them. So for example, if we have their income or we have their what are other numeric values that we might have, they're a pension fund or anything like that. So we have a couple of numeric values to describe different people. And we want to know how similar they are. Basically what we're doing is we create this dissimilarity factor, the Euclidean distance from each other by. Basically just looking at how for each of these values, how dissimilar are they? So we take the elf value of our of our member x I minus the elf value of our member x j. We square that whole thing. We sum all of that up over all of these different factors, and we take the square root of that. And that's the Euclidean distance between these two people. There are other adaptations to this measure which are very similar, such as the Manhattan distance, which instead of taking the square root of the square of that, takes the absolute distance or the absolute dissimilarity between these two, and also sums those up for different factors. So we have all different factors. Now, obviously you won't only encounter numeric data. It would all be very easy if everything was numeric. You might also encounter binary data. In that case. One way of measuring the dissimilarity is looking at how many of them, how many of these factors are the same, and how many of them are not the same for any two people. And then counting basically, how many overlaps do these two people have in their categories? Binary data is very important because remember, if you think back to our idea of dummy encoding, this is one of your primary data types. If you have if you have categorical data, which is very common in the social sciences. So this would be your dissimilarity measure if you had all binary data. So all categorical data for two individuals, you would count up all of these overlaps that they have. That's N11 the simultaneous number of presences of a specific factor. And divide that by N11 plus w times. And these are basically all of these not similar. So not the same or not overlapping factors between two people. And this W is a factor that you can set to anything that you choose really. It's kind of a weighting factor. So how important are these non overlapping non simultaneous factors. How much weight do you want to give them. A common weight is just one. And in that case we would call this the coefficient. So this is one of these words you might encounter every now and then. Now I mentioned categorical data. There is another way that you can actually handle that. And that would be the matching coefficient. And it's very similar to the distance that we just looked at. So we still have this kind of value either it's zero if they do not match in this particular feature or it's w if they match in that feature. So this is the kind of slight change we're not doing. We're not counting the zeros and ones. And then summing them up, we actually kind of giving a weight immediately to each of these individual non non-overlapping these individual matching features that we have. So typically it's one but it can be set to any weight. But it's a very similar measure that you can choose also for categorical categorical data. Now you will notice here that sometimes I use the largest s and sometimes I use the larger d. So here for example I say d for the Hamming distance or d for distance or dissimilarity. And you will notice I do one minus some kind of factor or some kind of term that we have in the back. If I took the one minus away and just looked at the at the back here, that would be the similarity. So how similar are they? And if we take one minus the similarity we get their dissimilarity. So the reason why I do it, sometimes with a dissimilarity and sometimes with a similarity, is that the way they are defined in textbooks. So the matching criterion is specifically made of measuring similarity. There's not really a formal definition to the the this matching criterion or anything like that. So I have to define that as dissimilarity. But if you are interested in this similarity instead you can just take one minus the matching criterion. And that's your dissimilarity. So very commonly what you will encounter is not just categorical, just numeric data. You will have a mix. And the social science, this is extremely common. You will have some kind of survey that you've done. You will have numeric data like the income or something. You will have categorical categorical data like the education. And you have to find a dissimilarity between those people while taking into consideration all of these different dimensions. There's different ways of doing that. What I've done in the past is actually I created dissimilarity measures or matrices specifically for each factor categories separately and then combined them. So there are ways of kind of overcoming that and kind of combining them and then using the combined matrix for your clustering, for example. But you can also use a dissimilarity measure which is already created for mixed data. So these are so these two people here at the top specifically, if you read anything about clustering their names will pop up. I don't know what they've done for the careers. It's incredible. They've written like every book and every piece of paper about clustering I've ever written and I've ever read. So my whole thesis was basically about these two guys. It's incredible. So you will see them again and again. And they base this and on the Galois distance, which you might be familiar with and proposed the following in their book. And it's still kind of similar to what we've, what we've already seen. So we have the similarity measure here. That's your s for your record I and record J in the L feature. And we also have a coefficient, a binary coefficient delta here which defines whether that measure is missing or not. And that way they can actually sum up across all measures in one go. So each of these is either is either a missing similarity or not depending on whether the data is available or not for that particular combination. So then they create two different ways of calculating the similarity, either for discrete or binary variables. They are basically doing a simple matching coefficient here. So either they are matching, then it's one or they're not matching. Then it's zero. And for continuous and ordinal variables they propose this measure here, which is looking at the absolute difference in the two values and divide that by the range of that value of of that variable. So this is kind of similar to your Manhattan distance that you were saw earlier. But it's scaled to a the range of that variable. And then you sum all of those up for each of these people, and you divide it by basically the number of coefficients that you have. Because remember, delta is zero if it's missing or one if it's not missing. And that gives you the overall similarity for mixed data. And if you take one minus the similarity for the data, you get the dissimilarity for these two records x x I and x j. So I know those were a lot of different measures. You don't have to remember all of the formulas. If you feel overwhelmed right now, it's important that you're aware that each of those exist. So it's very important to first check what kind of data you have before you decide how you want to create your dissimilarity matrices. A common mistake I often see is people just throw their data into K-means, and you can't just throw your data into K-means, because K-means specifically uses Euclidean distance, and Euclidean distance only makes sense for continuous variables. So if you have binary data, you throw it into K-means. You're. That's just not making any sense. So that's why it's so important to be aware of different dissimilarity measures for different types of data. Now there's a couple of a couple of rules that we have to go through, but I think we should probably take a ten minute break now and do that in the second half, because I know you're getting a bit tired. I need a glass of water and a quick break, so let's meet again in ten. Okay. Settle down. Everyone settle down! We are back. There is still much to talk about. I will also take this opportunity to mention something that one of your colleagues kindly reminded me of. So you can thank him for that. I because I got suddenly excited again about a project I've been doing. Listen, I was asked about kind of is there is there a connection between clustering and other methods like logistic regression? So what would be a combination? What kind of is a connection between the concept, or is that is there a relevance for both of them at the same time? I've actually used clustering in combination with logistic regression, where I was building a hybrid model, which first used clustering to segment the data into groups, and then build separate models for each of these groups to predict a specific outcome using logistic regression. Similarly, you can also use different variables that you find in each of these segments are more important for different model buildings. So you might find, for example, if you're trying to predict customer churn behaviour as a binary outcome, then you could first segment your data into groups of customers and then build separate logistic regression models for these customer groups using the factors that are important to them and their behaviour, instead of drawing a blank sheet over all of your data. So there are ways of kind of combining these models into hybrid approaches, which are quite interesting. So clustering can also be used not only as a standalone tool, as an explorative tool of kind of data discovery, pattern recognition, but also is a way of finding out information that can be fed into a predictive model in a second step. You can, for example, use the group label as part of your model. That might be very interesting as well. Okay. Quick detour. Let's talk about the similarity measures in that part of the lecture. So typically we will assume a couple of conditions are fulfilled for our dissimilarity measures. I already mentioned one of those earlier, which is that the similarity can be calculated as one minus the dissimilarity. So they are kind of complementary to each other. Some other things that we assume typically is the dissimilarity between two points I and j is symmetric. That means that the distance from I to j is the same as the distance from j to I. We also typically assume that these dissimilarities and distances are positive, or at least greater or equal to zero for all of our points. In this case, I and J. In some cases we also have more conditions which are fulfilled and if those are also fulfilled. In the addition to the above, we would call our dissimilarity measure a dissimilarity metric. So in this case, for example, there would be our triangle inequality where we have. If we have three points, then the distance between I and j is smaller or equal than the distance from A to k plus the distance from k to j. So that's kind of connecting points where you have point I, point k and point j. And then you can go through point k on your way from I to J if that makes sense. So this is actually if you think back, if you have ever had graph theory in your undergraduate studies, that is exactly that concept where if you go through a vertex that you can if you go through connecting vertices, that you can calculate the distance from vertex one to vertex three via vertex two if you sum up the edges of those connections. And yes, obviously reflexivity, if I and j at the same point, then the distance between the two should be zero. I mean, that also makes sense. If they are the same person then there is no dissimilarity between them. So something that I'm quite passionate about as a researcher is this idea of what does dissimilarity mean as a concept. So doing my doing my PhD. This is actually one of the questions I was asking myself repeatedly, because I was working with this kind of these existing dissimilarity measures where you just plug in your data and then you get some kind of number, and then you just accept that as a fact. So just accept that the dissimilarity between person A and B is 15 points away, and you don't really think what does 15.3 even mean? So what does that really kind of translate to is that it's that a unit attached to this distance, is there a way of kind of interpreting this dissimilarity? And also very importantly, does that dissimilarity change depending on our context? One of the projects that I was working on used different data sources to describe the distance, the perceived distance from different locations in the world to Edinburgh as a city. So the reason why we were asking ourselves that is we were interested in tourism recovery after Covid. So we wanted to know which countries are the most likely early visitors coming back to Edinburgh as a tourist location, which was extremely important for companies to know because those should be the regions that they should advertise to, again, because there are the people who want to visit soon. So we looked at three dimensions of this type of distance, and one of them was relatively simple, the interest in visiting. And we measured that by looking at Google Trends data over time. The other was we were interested in the affordability. So we were looking at economic indicators of how well people were doing in those countries economically. How willing would they be spend additional money on leisure, like travel to Edinburgh? And then the third was travel time, measured by flight times and flight connections to Edinburgh. So the reason for that is if you think about distance in a special way nowadays, it's kind of a very different. Different way of thinking about these things compared to 200 years ago. If you had to travel from Edinburgh to London 200 years ago, that meant a very long travel time in a coach. Very uncomfortable, not very safe. You wouldn't do that every now and then. It's just took too long. Nowadays you might go there by car or you might go there by train, or you might go there by by plane. And all of these different travel modes determine how long you are aware. So it determines kind of your distance travelled in terms of time. But it also it's also determined by the perceived distance that you have to that place. So for example, you can take a train from Edinburgh to London in like five hours, or you can take a plane which might take an hour. Now that might already look like a big difference. It's like 40 hours different difference, right? But then think about how that you have to be by the airport two hours early to check in your luggage. That's three hours. You might have to travel to the airport. That might be another half an hour or so. Then you might have to travel. If you have to go out of your plane, you might sit on the wall. Failed. London Heathrow is a nightmare. So you're sitting in your plane another half an hour waiting for baggage handling to take your luggage. You're waiting for your bag, your bag goes missing and you sit in London Heathrow, which is not London City. So you have to travel from London Heathrow to London city centre. So suddenly, five hours by train to London. Houston doesn't seem that long anymore, does it? So our kind of perceived distance can be very different from the actual distance as measured in kilometres. So that's kind of what I was thinking about throughout my thesis. And I'm still very kind of interested in that as a concept. So dissimilarity depends on context. And dissimilarity should be determined by context. For spatial data, if you have large distances you might want to consider the curvature of the Earth. Suddenly the distance is not Euclidean. It's not a straight line. It might follow flight routes or it might follow car routes, or it might follow the curvature of the Earth. It depends on what kind of distance you're measuring. For time series data, it's very important as well. I was mentioning our tourism project. We were interested in the interest but measured by Google Trends over time. So we were looking at these types of types of time series and comparing the time series of interest of different countries. So how does the interest in Edinburgh and Belgium change over time compared to the interest in visiting Edinburgh from the US? So there are different measures of dissimilar city for time series data specifically. So all of these measures that we talked about earlier are mostly for data, which is kind of describing, for example, an individual or location using numeric or categorical data. It's not really thinking about the shape of the data over time. So if you have time series, you can look obviously at each time step and just calculate the Euclidean distance between the value at each time. But that doesn't really take into account that there might be shifts in that time system, might be kind of warped peaks, for example, where you have a peak occurring in Belgium and the same peak occurring in France two weeks later. If you just look at each time separately, you're not capturing that. You actually have to look at the shape of the time series and see how far away and how dissimilar are they from each other. So there's something, for example, called pandemic time warping, which is a dissimilarity measure for specifically for time series, which takes into consideration the shape of the time series instead of just the values at each point of time. So spatial data and time series data. I say that again. Again, they are kind of spatial data types. So that's why there's a tricky to handle. There's dependency between observations. And they come with a lot of assumptions that we make about them. So coming back to your Principles of data analytics project, for example, I get asked how it can be used as time series data. And I tell you technically, yes, I really don't care what type of data you use because I care about you choosing the data in such a way that it interests you and is able to answer question. That being said, be careful with time series data because it's not the same kind of data that you might get from just a survey that you're looking at a cross-sectional survey with observations of David Dunn once about a thousand people. And that's it. So one of survey, it's very different from a time series that is being collected over time. Now there's overlaps obviously like cohort data, longitudinal data, all of those data types as well. So be careful with different data types and how you handle them. Now let's look into a couple of different algorithms for clustering. Now, after I've bought you to death with all of these extra steps and considerations. And it's not easy after all, is it? So one of the most common use clustering algorithms that you might have heard about before is K-means. So the idea behind K-means is relatively simple. You take your data, you divide it into k groups. There's our K again. So we have to choose another parameter. And then you use that as a partitioning algorithm which basically means it partitions. It kind of divides your data space into groups. So there are fuzzy approaches where you don't cut these groups completely, but you kind of allow for degrees of memberships instead of deciding group one, group two, group three, you might say, oh, this person is 50% group one and like 20% group two and 30% group three. So degrees of membership in fuzzy clustering. But let's focus on kind of the the cutting the partitioning part itself. So how Cummins operates is it's really interesting because we start randomly. We just have our data set, we select k different observations and there will be our cluster centroids, our centre points for each of our groups. Then we assign each of these observations to that cluster or that centroid which is the closest to it using your Euclidean distance. That's why your distance measure comes in. And then we recalculate the centroid by taking the new middle point, the new mean of all of those observations it was just assigned to. And we repeat steps two and three. So we assign each observation to its newest new centroid. That might or might not have changed. And then we recalculate the new centroid as a new middle point. And repeat that process until the centroids are not moving anymore. And at that point we say our solution is stable and we have our clusters, our k clusters in the end. Now it sounds a bit odd because it there's a lot of kind of random and then assigning and stuff and everything. So if I draw that it would look something like this. So we would have some kind of data space and we would have. Points somewhere over here. Let's do a bit of here and a bit of here. So I'm trying to kind of create a space where you can see naturally occurring clusters. Okay. So I hope you can see ish that there might be four clusters here and three points. And I don't know what I'm doing with. So what K means now does is basically sets, let's say for for some reason we know it's for four points. So, Len, come on, let me draw. Let's do one here and one here. One here. And then that's the one here. So K-means now will assign each of these points to its closest group. And you can see I. Set them. Randomly. So these will be for example assigned to that. This is the closest. This is probably the closest for this as well. So they will start kind of looking for all of these distances to that point. And then these will probably belong here. Etcetera, etcetera. So we now start assigning each of these group, each of these points to its closest point. And you can already see this is probably not the perfect solution. So there are a couple of points here which we think probably belong here. But they've been just split up by that so that there will be a bit of an odd movement here. The great thing about K-means is that hopefully, depending on how well you chose your initial clusters over time, these points will now move. So over. Over time, if we have here and here, let's try to replicate that. So if we now we calculate. We know on the top left, for example, these points here were just assigned to the same cluster that centroid. So these groups are now assigned cluster label one. And we calculate the mean of all of those points which were assigned to the same centroid. So the new centroid if we calculate that mean will probably wander up a little. So it might be now here. So this was it was originally here. It's now wandering up and slightly to the left because we're calculating the mean. Its position as the mean of all of these points. It was it was a devil assigned to it and a little bit of those, but not too many. So it's not drawing it too far to the right. So let's say it's here now. And then we repeat the process. And you can already see that for this centroid now becomes much easier. All of these points will likely be assigned to the centroid. So we already converging to some solution with it here. And we do the same process for all of these four points. So we calculate new positions for each of these centroids as the means of all of those points it was originally assigned to. And iteratively we arrive at some kind of solution, which would hopefully be that each of these groups are their own clusters. With their centroids in their middle. Now I say hopefully. The reason for that is the random selection of the initial centroids can be K-means downfall. It's computationally cheap, but it also means that. Depending on where exactly these initial points are selected, or set that a non optimal solution might be found where. For example, if you. If you select the the point in such a way that in the end K-means tends to split very large groups, for example. So we sometimes have these very large clusters and we might have a couple small ones here. Now the optimal solution that we want to see is that the one on the top is one large group. However, many cases K-means tends to split them. So it tends to look for these small spherical clusters instead. So it would split this into two distinct groups. And depending on your context on the application, that might or might not be a good thing. So in some cases we want small clusters in distinct groups. In other cases we actually want kind of to distinct to define these larger groups if they really exist. So K-means can be a bit tricky if this all seems a bit convoluted. I would recommend that you have a look at. Basically YouTube when you type in K-means algorithm all over time, and there are beautiful GIFs that were created in small videos where they show exactly how these points move and how it converges to a solution, which is hopefully optimal. But due to the random selection of initial point might not be optimal. So coming back to okay, because that is really determining what kind of solution we find. Earlier I was talking about this kind of solution where K-means would like to split. It likely then also depends on your K. If you select a k of four, it would definitely split it. If you select okay of three, might or might not split it depending on the initial centroids. So it might actually try for example to merge these this half with the bottom with the bottom here and merge this half with the bottom here. I've seen weird solutions by k means. So how do we choose which k is optimal? Earlier? I just told you it's for. Obviously. In some cases, it's as easy as that. In some cases, you will just plot your data as a scatter plot and you say, hey, this is very clearly for in most cases, like 99% of cases, that's not going to happen because your data won't be so easily divided and so pretty, and especially your data won't be in two dimensions. And as soon as you have more than two dimensions, it gets really tricky to visualise that. So there are a couple of different methods that we use to choose k. Three of the most common ones are the elbow criterion, the silhouette measure, and the gap statistic. So the elbow method is a very subjective plot based approach. It basically looks at the explained variation as a method or as a function of the number of clusters, and then chooses what it determines to be the optimal number of clusters as the number at which there's not much more improvement in variation. So you remember this concept of explained variation that we had earlier. When we're talking about linear and logistic regression, it comes back now. So this method is highly subjective. But in some cases the choice of k is very clear and very easy. I was talking about this kind of four plots earlier. Easy to see. So if you have something like that the elbow criterion can be perfectly fine to choose you can you wonder if only what that looks like and why we call it the elbow method. And the reason for that is that the plot. If you have your number of k here and you have, for example, your explained variation, let's do explain variation. There are different. Different ways of doing that. It can be explained variation on the y axis, or it can be the within within cluster sum of squares. So how diverse or how big and weird are the clusters? But let's look at the explained variation. And then the elbow plot. Will. Look something like that. So it will go up and up and up and up. And at some point it will stop. And go relatively straight. Not perfectly. This would be perfect. It won't look like that, but relatively perfect. And then you would just choose. K is equal to three, because after three there is no more improvement in your explained variation that you can gain by having more clusters than that. And we call it the elbow because it's the elbow of the curve. That's what kind of changes direction. Now this is very pretty. It probably won't look like that. It will probably be much more smooth and that makes it much more difficult to choose. And it's also very subjective. So some people might say, yeah, okay. But realistically like how much improvement is this. This is not much improvement. Is that really worth it. Shouldn't we just use case two. And you can because it's subjective. So there's no one stopping you from choosing k two in this case. And in some cases it might be the right choice because your I don't know your application case might might make much more sense with case two because you have two ad campaigns to run. So you want two groups of people you don't want V because you don't know what to do with the third. So in that case you would choose k is two, even though the elbow plot might say something different. So because we just said the elbow plot is so subjective because we're just looking for this kind of change in the curve. We can use more objective measures as well. And the silhouette measure is strictly speaking, a measure of quality for the clustering solution. So this is something you would want post-hoc to have a look at how good your solution really is. But you can also use it by plotting it as a function of the number of clusters k to decide which of those k produces the best outcome. So it basically looks. At how. Similar are data points to points in the same cluster, and how dissimilar are they to points from other clusters. And that's what clustering is all about. You want to find points which are as similar as possible to other points of the same group, and as different as possible to points in other groups. So similar or different again is defined by your dissimilarity measure. And that can be for example Euclidean distance depending on your data. And it is a more objective measure. But it is more, more difficult to calculate like that. So there is more steps involved then hey, I'm just plotting this and I'm looking for like this little point in the curve and I'm happy about it. So suddenly we have to do a bit more calculations. Specifically, we want to know the dissimilarities within the cluster. We call that intra cluster dissimilarity or the within cluster dissimilarity. In this case, for example, we have two points I and j and then the same cluster C and we have the size of cluster C, which is the number of points in that cluster. And we want to know that the similarity between each of these points in the same cluster. So we calculate the dissimilarity could be Euclidean distance, could be any measure that you have. And then we sum up all of the distances in the of the points within the same cluster. So that's kind of your big within cluster sum of squares. You can it's also called that. And you divide it by the number of points in the cluster minus one minus one. Because we are always calculating that for each of our points. For example here for I. And then you do the same. Thing. For point I, but between the clusters. So what's the distance of point I to all points outside of its own cluster. So again calculating all the distances, summing all of those up you call them the between cluster dissimilarity or the Intel cluster dissimilarity. Not to be confused with the intra cluster similarity. That's why I prefer to use within between because my pronunciation sometimes kind of swallows both of those up. So it's the same calculation. We want to know all of the distances from point A to point j. Now j is in a different cluster. We divide that by the number of points in that other cluster. And crucially we look for the minimum. Why do we look for the minimum. Because we really just interested in the neighbouring cluster. We don't really care that much about the other clusters very far away. We want to know in our kind of space of a subspace of our data, how well are we separated from our neighbours? So how distinct are we from the cluster just next to us? And then the silhouette of this data point AI is calculated by taking the distance of point AI between cluster minus distance of point AI within their own cluster, and divide it by the maximum of those two terms. Now the silhouette measure or the silhouette coefficient. Here. Your name's again Kaufman. They're popping up again is then the maximum of all of these silhouette measures. So you calculate a silhouette of each data point, and then you take the maximum of that, the maximum mean of that. Um. And that's your kind of overall silhouette coefficient of the whole data space. So I can always see this is computationally more expensive. It gets especially more expensive with an increase in data size, because you have to calculate that for each of these data points. But it is a very robust measure because it basically looks at how distinct and how well separated are your clusters from each other. And the better separated your clusters are, the better your solution. So yes, the silhouette coefficient is a single value that you can then use to use to choose a specific k by namely calculating the silhouette coefficient for each value for each possible value of k that you can imagine, and then comparing them. Basically you look for intra versus between cluster dissimilarity. As I mentioned, for each possible solution that K would provide. Now you're probably wondering okay, that means I need a range of. What is that range? And no one can actually tell you that because you will have to try it all yourself. It depends on your data. So there's not really any rule of how many clusters you should check for, because the more varied your data is, the more possible clusters you can find. Good rule of thumb is go with what your application case realistically can handle. If you have a company and I want to do cluster clustering on their and their customers, and you tell them I divide these 364 different customer segments, they will tell you to get out and do that again, because they can't handle 364 different customer segments. The marketing department will throw their hands to the air. We don't like that. So they want a smaller number at the same time. If you have two small number, you're not really capturing the patterns. So you have to find something in between, something realistic which is big enough to capture everything and small enough to handle the solution for. So our third measure is the gap statistic. And the gap statistic is very similar. So it's very similar to the silhouette measure. And that it still looks at the within cluster. So the intra cluster dissimilarity. But now we compare it to a proposed baseline distribution. So we are looking for the gap between a clustered solution and an imaginary space in which your data would not be clustered because it's uniform across the space. So for example, if you have a two dimensional idea and you have a very nice and distinct kind of grouping of data points, each of these four corners, then you can compare it to what happens if the same data was uniformly distributed across that space. How big is the difference between the clustered solution versus our kind of uniform distribution of these points? So how far is our cluster solution from random data? The two hour cluster solution on our real data, and it kind of describes how separated or distinct these clusters are compared to uniform distribution. Okay. So these are kind of the three main measures that we would use to choose. Okay. Let's talk about K media. It's because I was complaining about people just using k means earlier when you can't do that. So k media is is the thing. It is the solution to that problem that I was mentioning earlier. K-means uses Euclidean distance. Euclidean distance only works for numeric data. So if you have, for example, a mixed data situation, or you have categorical data and binary and numeric and everything coming together, you can instead use k meteoroids. You already know who invented K meteoroids because it's the same people. Again, it's our good friends California and they invented it as the partitioning around meteoroids algorithm. Pan. So it's similar to K-means basically follows the same principle. You still have to choose K in advance. You still partition the data space according to similarity of data points. But in this case, the way you define similarity changes. So you can, for example, use the Jaccard similarity index. You remember that was the one for binary data earlier. You can use something for mixed data types. All of these similarity measures that we were talking about earlier can all be used with k meteorites. I use k meteorites myself. When I was doing the measure, the analysis I was talking about earlier where had time series data on customer interest in tourism. I had flight data on distances. So I had these different data types, different ideas of dissimilarity. So I combined them into one dissimilarity matrix. And because that was not using Euclidean distance, I couldn't use K-means. So I chose k meteorites instead because that allows me to self define my dissimilarity matrix and put that into the algorithm as an input. So if you choose k meteorites, you can see that you don't have to you don't have to define just the data. You can just give it a dissimilarity matrix that you defined yourself with different measures, such as those that we discussed earlier. So I was already talking about cluster evaluation earlier, for example, with the silhouette measure, which is your kind of evaluation metric. But we also talked about keeping in mind what clustering really does, which is we want to find clusters of the most similar data points. So we want to minimise the within cluster dissimilarity. So how dissimilar points to each other in their own group. And we separate them from the other data points that we want to maximise the between cluster dissimilarity. And there are other things to keep in mind that I already talked about throughout this lecture. How useful are those results really for our context? How computationally expensive is our result or our approach? And clustering doesn't really predict an outcome. So evaluating the outcome is very tricky because it's such an explorative solution. You can't really say the solution is optimal because you don't know the cluster memberships. It's something that you invent as a concept, so you can't really kind of test how well your categorisation of each data points is how the way you would do it in a classification situation, because you don't know the true class label, you invent the class label for basically nothing. So evaluating clustering solutions is a whole can of worms. It's very tricky. Typically we look at how well separate our clusters and how useful are they for our solution. Okay, so let's spend the last ten minutes or so on hierarchical clustering. And this is a different approach to K-means I was talking earlier. K-means is a partitioning approach. That means we partition our space into groups. And that's it. It's a one way go. So we divide each member. Each data point now has a cluster membership. We're done. That's it. I work in clustering does a bit differently. So instead of just dividing once we actually creating a hierarchical structure of possible clustering solutions over time. It's an iterative process where we keep going forward and dividing more or merging more. So instead of dividing deciding where to where to how many clusters you want to create. So instead of dividing, instead of choosing your K, we have to decide where to cut our tree and you will see what exactly that means. I would like to. So because it's kind of this iterative working. Hierarchical process. There are two ways that you can do that is either a top down process or a bottom up process. So this will come back to you for decision trees as well. So it's a very similar process that we are following here with a top down. That's also called a divisive hierarchical clustering. I think the most popular algorithm for that is called Diana. So in that case all observations start off in one group. So all points. So think about your two dimensional space. Everyone is in the same cluster. Cluster one or cluster zero whatever you want to do it. And that and then we kind of iterate over time divide that space into more and more smaller and smaller clusters. So we do one cut. We have two clusters do another cut. Now we have three clusters to another cut. We have four. Yes. So we go. We kind of keep going and divide the space more and more in each step until at the end each of these data points is in its own cluster. So we go all the way from everyone in one to everyone in their own cluster. Bottom up is the exact reverse of that process. Each data point is in its own cluster, and then we start merging those which are the closest step by step. So we start by merging two, and then we start merging another one in and we start merging another one in. We kind of start building a solution until at the end again all data points are in the same cluster. So it's the same process just just flipped. And I think the most popular agglomerative clustering algorithm is Agnes. So we have Diana and Agnes as our two algorithms in that space. So when I was talking about this cutting idea, what we mean is we get this type of tree on the top left and we call it a dendrogram. And Addenbrooke describes how the solution is kind of merged over time. So each of these leaves that you can see here are one of our nine observations. And you can see what these nine observations look like on the right side as well in our two dimensional data space. So as we move up the tree the leaves are fused. So in the beginning each of these leaves each of these nine points are their own cluster. But then we decide, hey, five and seven and one and six. They are really close. So we can see these points here one and six, five and seven. They're very close. We merge them. We merge them at this first step so we can see that now one cluster okay. And then we see this five seven cluster is really close to 0.8. We can see that here on the right as well. The five seven cluster is close to cluster eight. So we start merging this one as well. And then the next step we can see our one six cluster is close to four. So we merge that and then that's close to three. And we keep merging those. And you can see 0.9 is a bit of our outlier up here. Right. It's the furthest away. And that's why 0.9 is the last one to be actually merged into our final solution. So we kind of merge those over time. And the earlier this fusion work the earlier this fusion occurs the closer the observations are. So you can see five and seven and six and one. They were the closest. So they were merged. Earliest nine was the furthest away from everyone else. That's why we merge them in latest. So the concept of the. We mentioned that as a key concept in clustering. And we'll come back to it here again because we have to decide what does this mean. When do we merge our groups. So when do we decide we should merge them. And that's especially true if we have multiple points or within that cluster. So for example here we had five and seven together in a group. How do we decide whether eight is close to them or isn't. There are different ways of thinking about that. You can do the maximum maximal inter cluster dissimilarity. You can do the minimum inter cluster dissimilarity, the average of that or the centroid. So what does that mean. You basically look at if you have cluster eight here and cluster five seven here. What's the kind of closest distance between them. That would be five and eight. What's the furthest distance between then. That's seven and eight. And what's kind of the medium. So that would be eight. And whatever is the kind of middle point between 5 and 7. So you can see if I look at the maximum distance that we have that will be complete linkage. The minimal cluster definitely have the minimum distance that we have single linkage, the mean, which would be kind of the average linkage or the centroid, which is the dissimilarity between the centroids average and centroid are very similar. And the reason why we kind of separate the two between between the two is just the way we either calculate the mean inter class dissimilarity. So that's the the mean of the dissimilarities between the clusters entire cluster versus the dissimilarity between the two centroids. So this is a sum where we summing up all of the all of the distances and we calculate the mean. And this is just a dissimilarity between two centroids that we have. In many cases they're almost the same. But it depends on how the data is spread within a cluster, whether they exactly the mean or not. So you will also see that depending on which one you choose, and that will affect the shape of the clusters that are formed. So maximal in the cluster dissimilarity, we're looking really at the furthest possible points in deciding whether we want to link them. So these typically form very distinct and kind of small clusters which are far away from each other. Minimal inter cluster dissimilarity is easier. So it's much more likely to merge because if there are two points which are close to each other, that's enough, even if the other points are very far away in that same group. So in that case, we kind of build very elongated and very kind of large clusters very easily. And then the average, the centroid methods are kind of balancing between the two. So they are the most kind of balanced option. That is the conservative option. The more easygoing option and then the kind of average in between the two. Yes. So last but not least, let's talk about what do we do with that angiogram. Because it's a tree, it's not really one solution. It's a nested set of possible solutions that you might want to choose. So you can choose to keep the whole tree in some cases that make sense. In some cases, all you care about is kind of describing the structure and the patterns in your data. And in that case, you can just keep the whole tree and present it as basically to result to that, because it's describing the structure and the similarities of each of the points in that. So if your objective is to explore, you can keep the whole tree. In some cases you want to cut the tree as a point at some point. So we were looking at this kind of graph here. You might choose that at some point. This is the maximum dissimilarity that you want to accept. So for example, if you choose the value 1.0 this is where you would cut. And then you would keep two, nine and three in their own separate clusters and only accept the merging of these points here 164 and five seven, eight. So that would be a way of cutting the trees. At some point you just you decide, okay, this is as far as I want to go with dissimilarity in my solution. So typically we want to cut when the merging or dividing has no further significant effect on the dissimilarity. So it just comes back to this thought process or kind of silhouette measure where we're looking at okay. When does our dissimilarity kind of even off a little bit. And that is where we would cut the solution. And that's our our final solution that we accept in terms of clustering. But again it's a bit subjective where exactly you want to cut. So clustering is very a very subjective method in itself really because it's all about there is no right or wrong answer. It's explorative. So we exploring the data and finding possible patterns in it. So let's have a discussion. Maybe next time briefly about clustering because I see I'm managing very, very well with the time today. So I mentioned in the very beginning next week is reading week. So there will be no lectures on computer labs. Use that time to catch up on your reading. And we will do our computer lab this week where we look at cross-validation particular. And if you have any other questions about clustering, I will be here for another ten minutes or so unless someone else tries to throw me out of the room. Okay, see you tomorrow.