Module 4: Clustering

Video Transcript

Video 4.1: Module Introduction: Clustering (2:02)

In this module, we'll start talking about specific descriptive and predictive algorithms and models, specifically on the very important class of clustering algorithms. At the high level, clustering algorithms attempt to divide a large group of objects that could correspond to customers, emails, athletes, financial transactions and so forth, each with its own individual characteristics and attributes into what we call similar subgroups. And we will define what similar means in a more precise way in just a bit. This module will be structured as follows. We will provide first a motivation about why clustering algorithms are so important and what kind of insights they can provide in different business environments. We will then discuss the output of clustering models and algorithms and talk specifically about their generic ingredients so you can get a general framework that they can then be specialized to understand specific algorithms. This will be followed by a more technical discussion of specific and very commonly used clustering algorithms and what are their respective pros and cons. Again, you will learn how to apply these algorithms using Python, but in this video we will primarily try to build your intuition about how they work so you can be a smart user. But first, I would like to provide a broader context and also set up some data modeling frameworks and sampling preliminaries that will be useful not only in this module, but throughout the course.

Video 4.2: Data Modeling Framework (6:24)

So, before we start talking about predictive and descriptive algorithms, it's very important to have some common language about how to think about data and how to model data. And what I would like to do next is to introduce some very commonly used framework and jargon. So, when we think about data, we can think about some raw data that can be extracted directly from some of the systems in your organization or like in the MGH example that we discussed in some previous modules. Think about some representation framework that you take the data and feed it into it. But when we actually think about predictive algorithms specifically, we typically think about the concept of observation.

For example, the objects that I talked about before, a financial transaction, an athlete, a customer, and so forth. Each one of them can be thought of as an observation that has some different data items. And we often tend to divide the different data items of a specific observation into two classes of features. We have what we call the descriptive features that describe the object, describe its attributes and characteristics. And then we also have what we call outcome features that are usually being determined by some performance measures of interest. So, for example, if I think about the customer, the customer descriptive features could be the age of the customer, where they live, their income, and so forth. And the outcome measure could be how much they spend with us over the last six months. Now when the outcome feature is a binary variable or something that can either get the values, zero, one, or maybe some integer value among a finite set of integer values. We often call this outcome feature a label or a classification label. So, outcome features can come either with continuous values or with a classification labels.

But in most cases, observations will consist of some descriptive features and some outcomes or labels. So, the modeling framework here is robust enough to capture different modeling needs and can change depending on our choices, depending on the setting and what we're trying to model. So again, the important thing to think about our observations that usually correspond to objects, and again that's a broad definition, and that's the power of this pro definition because it can be useful in basically any setting. And then we need to think about and define to ourselves what are the descriptive features in our particular setting and what are the outcome features or the classification labels. Now one thing that we will tend to assume and many of the models that you're going to learn about are going to assume is that the observations, both the descriptive features and the outcome features or labels come from some underlying stochastic process, and they're going to be very different assumptions depending on the model. What this stochastic process looks like and how much information we have about it? More to come about this in the later modules. But before we continue, I would like to emphasize that this notion of descriptive features and outcome features is very important also in the context of thinking broadly about what type of predictive algorithms they are. And one way to classify predictive algorithms is through the notion of what information we have about the labels and the outcome features. So, today we're going to talk about a situation, when we talk about clustering, where you actually don't have access to outcome features or labels. The only thing that we're going to work with is descriptive features that will describe the attributes of any object through a single observation. And this is called often the class of unsupervised learning algorithms.

When we say unsupervised, we mean that we don't have access to the outcome features or the labels. At the other extreme and we're going to talk about many more algorithms like that, like linear regression, logistic regression, and other algorithms. We also have a class of algorithms in which you actually have information about the outcome features and the labels, and these are called supervised learning algorithms. And we also sometimes have to deal with a middle point where when we have some partial information about the outcome features or the labels, but this information is relatively limited or partial. And these algorithms that apply in this setting will be called semi-supervised learning algorithms. So, the number of algorithms in each class is very broad and it's actually evolving over time because more and more algorithms are being developed. What you can see here is just a sample of some of the more common algorithms and on many of them you will be able to hear more details in coming modules in the course. But if you want to know more, we will provide some more information in the supporting material of the course. So, equipped with that notion. Hopefully now, you have a map of the types of predictive algorithms you're going to learn about throughout the course. And again, today we're going to focus on a very important class of algorithms, clustering algorithms that fall under the class of unsupervised learning algorithms. Again, we're going to assume that the only thing available to us are descriptive features. No information is available about the outcome features.

Video 4.3: Clustering Algorithms: Motivation and Use Cases (5:57)

So, recall that we talked about the fact that clustering algorithms aim to divide a group of objects or observations, if I use the data language are just introduced, into similar subgroups. And what I would like to do now is to talk about what are the benefits?

Why are we interested in doing so? And this is based on some modeling assumption that similar objects are assumed to behave in similar manners, to respond in a similar manner to different interventions. So, if you believe that's true and you find similar subgroups of objects, for example customers, you can then assume that you can apply the same marketing approaches to them, perhaps offer them the same types of products and services, perhaps give them the same medical treatment, and so forth.

So, having some notion of similarity that is meaningful in the context of the interventions that you consider is very, very important. And this is one place where clustering algorithms can provide major insights. Another assumption about similar objects is that they might be working well together. So, that could be important, for example, when you want to decide how to form teams or decide which athletes will train with each other. If you're in the insurance business and you assume that similar customers, similar objects have the same risk, then that could guide what kind of insurance policies you would like to offer them, what kind of loans you might want to offer them, if you're a bank. So, again, thinking about similar customers as having similar level of risk or similar profile of risk could be very insightful. So, all of these examples correspond to what we call personalization. And we're going to talk more about this towards the end of the course. Now, clustering algorithms also have a very important role in distinguishing between what we call normal and abnormal states, that's often-called anomaly detection. So, let me give an example that will illustrate that. So, think about an IT system and think about your desire to be able to detect a cyber-attack. That's a very, very relevant threat to basically every business these days.

Now, one of the challenges here is that if you look on your IT system and all the transactions, and the data transmission, and the various data types that are being transmitted on your IT system, this is a very noisy environment that has very, very variable patterns. Moreover, in all likelihood, you don't have historical data on too many cyber-attacks, if any. So, what could be your approach in trying to be able to develop an alerting system that could maybe detect a cyber-attack in a proactive manner. So, one way to do that is to try and understand what are the normal patterns of behaviors, for example, in terms of the patterns of transactions or data transactions in your system. And then if you understand what is the notion of normal here, you could hopefully detect situations when suddenly these patterns seem to be out of order, seem to be abnormal. Again, this is often called anomaly detection, and a similar pattern or similar use case could be equally applicable in the context of detecting fraud in financial transactions, detecting spam emails. All of these use cases fall into the framework where you're trying to understand what normal behavior is in a very complex system and the hope is that that will allow you at real time detect when something abnormal is taking place. So, you can actually alert your system and intervene in a timely manner to hopefully prevent something undesirable. And again, clustering algorithms because in all likelihood you're not going to have any labeled data that tells you what is a cyber-attack, that tells you what is a fraud in financial transactions. You really have to rely mostly, if not entirely, on unsupervised algorithms. And clustering algorithms is the primary tool that you're going to apply. Clustering algorithm also can inform other models. For example, they can inform descriptive and representation models because they can give you a notion of personas. For example, if you talk about customers, they help you describe and understand the different types of objects that you have in your system and that can inform directly more advanced layers of descriptive models and representation models. But clustering algorithms can also inform predictive models. For example, they can be used to create attributes that can then be fed into more advanced predictive algorithms. For example, the membership of a given object in a cluster could become now a feature in a more advanced predictive algorithm. And clustering algorithms can also inform experimental design. If you have a good notion of what the various subgroups are in your system, and you have some assumptions about what will work for each one of these groups, that could directly guide some experiments that you can do in order to validate your assumptions.

Video 4.4: Clustering Algorithms: Ingredients (9:19)

So, we're going to talk in a second about how you create clusters, and how you create meaningful clusters by talking about specific clustering algorithms. But I wanted to start first by talking about what is the output of clustering algorithms, and how do we measure and assess the output. And that's going to provide us some intuition about what we are trying to accomplish. So, what you can see here is a typical output of a clustering algorithm. Specifically, you see four clusters, each with its assigned objects in blue. The first observation, I hope you can see, is that the number of objects in each cluster can actually vary. And that's a very typical situation in almost all clustering algorithms. The number of objects assigned to each one of the clusters might not be the same and could actually vary substantially. That's one of the things that we have to consider when we evaluate the quality of a clustering output. So, one thing we don't want to have is a cluster that has too few customers because that's not going to be really useful for us to really divide our collection of objects into very, very small groups.

So, the size of clusters is one consideration on how we think about the quality of clustering algorithms or the specific output that they provide. Now, one thing that is very important to interpreted clustering algorithms is what we call the representative of each cluster, and we're going to talk about some ways to create the representative of each cluster that typically, what we try to do here is to find a proxy of the average object in the cluster. And why is that important? Because remember, if we want to use that to provide insights into decision making, the whole point here is to take a very large collection of objects and try to essentially create a nice compact representation of them. And the clustering or the cluster representatives are a very effective way to create something like that. Now, there are other statistics that are important when we try to evaluate clusters. For example, in particular, we use the word similar for a while, and we still did not define what exactly do we mean by that. In a second, we'll do that.

But one of the things that we would like from a clustering algorithm and a good cluster output would be if the objects within each cluster are very similar to each other. And on the other hand, the objects across clusters are quite different from each other. That would be ideal clustering output because it basically tells us that we were able to put together all the objects that are similar to each other and really distinguish them from other objects that are not that similar to them. So, the notion of similarity and dissimilarity, both within each cluster and then across cluster is very, very important metric when you think about the quality of clustering algorithm outputs. Now, how would you define that? That's coming next when we're going to talk about the major ingredients of clustering algorithms. So, again, the idea here is, before we dive into specific algorithms, is to provide you a framework to think about clustering algorithms, and then you will be able to think about the different types of clustering algorithms and how they fit into this framework, and they would fit by basically tweaking and tuning different elements in this picture.

So, what do we have here? On the left-hand side, you see the input into the algorithms, that as I said, it consists of objects or observations with their attributes, or with their descriptive features if I use the language data. Now, in addition, you might also provide as an input some things that we call hyperparameters. For example, in some algorithms, you will specify in advance the number of desired clusters. Now, the output of the algorithm, as we saw before, consists of a number of clusters, the objects assigned to each cluster and the representative of each cluster, and then some related metrics that again have to do with the similarity and the dissimilarity within clusters and across clusters. Now, what happens in the middle? What are the ingredients of the algorithms? So, each algorithm will have some initiation conditions. So, how you get it started, it will have the clustering formation mechanism; how it creates the clusters and many of these algorithms will create the clusters in an iterative manner, and then the additional ingredient is what we call the metric, the underlying metric. Remember that we talked about the notion of similarity.

We're going to use specific metrics to define the notion of similarity or dissimilarity, both within clusters and across clusters. So, the choice of what metric to use is a very critical design element in any clustering algorithm. And usually, what we need to have is a metric that will define the similarity between pair wise objects between any two objects and also between sets of objects. And that will allow us to to think about both the distance between specific objects, the similarity between specific objects, as well as the similarity and the dissimilarity within collection of objects within one cluster and across clusters. And let us just see a few examples of how you can make choices about the different ingredients that we just mentioned. So, remember that we talked about the importance of having a representative for each cluster that often corresponds to the notion of a centroid of a group of points. So, what do you have here are a few red points that corresponds to objects or observation. So, what do you see here? There's a collection of red points that correspond to objects or observations, and then the green point is what we're going to define the centroid. This is going to be the average point here or the average object. So, now how are we going to calculate this average object. One very, very natural way is to calculate it as the average of all the points. Specifically, if I think about points in the plane that each car is defined by a pair of coordinates, x_1, y_1, x_2, y_2 up to x_6, y_6, the centroid will be defined by averaging all the x coordinates and all the y coordinates. And that will give us the green point that we had before as the average object or the average observation. And that's often going to be a way of generating a representative of each cluster. Now, if you had a point that will not be in the plane, then we'll have more coordinates to describe it, you will apply the same approach just for each coordinate separately. So, basically, nothing is going to be different here. Now, we talked about metrics and what I would like to do is just to remind you a few metrics that are very useful and are commonly used within clustering algorithms.

And maybe the most commonly used one is the Euclidean metric in which when we take two points, a that is equal to a_1 and a_2. And again, I illustrate this in the plane in two dimensions, but very easy to generalize this to any number of dimensions. And b is the second point, b_1, b_2. Then the distance between these two points will be taking the square differences of each coordinate, a_1 - b_1 and a_2 - b_2, each one of them square, and then they take the square root of that. That's going to give us the Euclidean distance. But there are other metrics. For example, the Manhattan distance, the one norm is defined by taking the sum of the absolute value of the difference of each coordinate. So, if I have two points, x and y, I'm going to take x_i - y_i across all the coordinates, take the absolute value and sum it up. Another very commonly used metric is the maximum distance or the sup norm or the infinite norm in which when I take two points, I'm going to basically take the maximum different across all the coordinates. So again, these are just a few examples of commonly used metrics, but hopefully by now you have some sense of what are the typical ingredients of a clustering algorithm. And again, metric is one of the major and one of the central components of a clustering algorithm. And now we are ready to talk about some specifics.

Video 4.5: Method 1: K-Means Clustering (4:11)

So, we are ready to talk about specific clustering algorithm, and we're going to talk about the K-means algorithm. This is, perhaps, one of the most commonly used algorithms in analytics and machine learning. It's very simple, intuitive, and can scale and we're going to talk about some of its other advantages. But let me start by describing it at the high level. So, and I'm going to start with the input first. Remember, the input consists of the observations. And in this case, we are also going to have a hyper parameter which we'll call k, which will correspond to the desired number of clusters. So, this is an algorithm where we have to specify in advance how many clusters we would like. And we're going to talk later about how are we going to set that number or select that number. But given that the algorithm initiates itself by randomly selecting k centroids, or maybe not randomly, but somehow selecting k centroids, so each centroid will correspond to a location and another dummy observation in the dimension that we work at. In our case, we're going to work in two dimensions, but in general, it can work in any dimension. Be aware that the centroid doesn't need to be one of the observations. So, it can be any point in the plane, in our case or in the Euclidean space with higher dimension. Given that, we are going to take all of the observations and assign each one of them to one of the centroids to create some initial clustering. Specifically, we're going to use the Euclidean metric and assign each observation to the closest centroid to it. Now, we have a temporary assignment of objects into clusters. We're going to now repeat the following step. We're going to calculate for each one of these clusters the centroid and then we're going to reassign the points again to the new centroids, each point to the closest one.

And then, we're going to recalculate centroids and we continue to do that iteratively until we get to a point, and we are guaranteed to get to that point in which we don't have to reassign any point. Let's just illustrate that on the specific example we discussed before. So, again we are in two dimensions here. We just selected randomly three centroids red, blue, and black. So, I take all of my observations and I now assign each one of them to one of the centroids, specifically to the one with the smallest distance. And that induces some clusters. Now, I'm going to take each one of these clusters and calculate its centroid. So, I have now new centroids and now I will reassign points and again I will reassign each point to the closest centroid among the new centroids that I have. And I'm going to continue to do that until I converge, as you can see and I basically don't have to assign any more point. So, again, this is conceptually very, very simple algorithm. It's guaranteed to converge and at the end of the day it will give you k cluster. And how do we control the number of clusters? We control it by the initial choice of k centroids that we maintain throughout the entire algorithm. What we're going to do next is to look on the results of applying this algorithm to the airline datasets we introduced before and talk about how to interpret the results.

Video 4.6: K-Means Clustering Applied to the Airline Customer Data (10:45)

So, we are ready now to discuss how to apply the K-means algorithm to the airline dataset that we saw before. Now, you might think to yourself, the algorithm that we saw before, and we introduced before might be very hard to apply and calculate manually. But you don't need to do any of that. This is something that Python can do for you in one command, and you will read about that and practice this after the module. So, what we did here, we really used the right command in Python, and we applied the K-means algorithm to the airline dataset we introduced before with request of eight clusters. And what you see here is the output of that algorithm applied to this dataset. And as you can see, there are eight clusters here and what we have here, there're numbers from 1-8 and for each cluster we represent here the centroid, the representative of the cluster with respect to the six features that we have. And this is a typical example of an output of a clustering algorithm. And what you get back is multiple numerical statistics, for example, the centroid averages.

You can get similarity metrics with respect to how the objects in a given clusters are similar to each other, how objects across clusters are similar to each other or dissimilar to each other. But this is just a set of technical metrics, numerical metrics. It doesn't wave you; it doesn't solve the problem of going about interpreting what these results mean to you from a business perspective. So, again, analytics is really about data and technical aspects, but also at the end of the day, applying your business intuition and business sense of what the results actually mean. So, looking at all of these clusters, one thing that I would like to first observe and draw your attention to is the fact that indeed the number of objects in each cluster are quite different. Some of the clusters are larger and some of them are smaller. And specifically, draw your attention to cluster number eight. It has only 14 customers in it. So, when you assess the output of this clustering, algorithm applies to apply to this dataset, this is something you have to ask yourself. "Does it make sense to have a cluster of 14 customers or not?"

And these are the questions that you have to ask yourself when you interpret a clustering algorithm. Now, let's just dive in and look on specific clusters to understand what they might be telling us and what kind of insights we can derive. And let's start with clusters two and five. And again, what you see here on the left is the normalized values of the centroids of each one of these clusters, and on the right-hand side, these are the same centroids with the original values. Remember, we normalized the data. Now, these clusters have about similar number of customers in each one of them, 1124 in cluster number two and 1107 in cluster number five. And when you look on all the values, all of them are quite negative. So, what does it tell us? Again, go back to the normalized data interpretation. This tells us that these are customers with relatively low engagement. They have lower than average balance, they have lower than average flying activities, and so forth and so forth. But there is one difference. If I look on these two clusters, there's one big difference. If I look on cluster number two and I look on their Days Since Enrollment, they have a relatively long tenure with the program.

They have already spent some time in our loyalty program, and nevertheless, they are very disengaged, if I may say. So, this might be the customer base that maybe already missed or there's not too much that you can do about them. But on the other hand, when I look on cluster number five, they have relatively low engagement, but they are actually relatively new to the program because the value there is - 0.88 as opposed to 0.95 for cluster two. So, if I'm thinking about what could be a potential insight here, I really need to focus on the customers in cluster five and try to make sure or make an effort that they don't become, in several months or a year or so, like the customers in cluster two. What can I do about that? Maybe I should actively engage with them, try to entice them to fly, try to provide them promotionals, try to offer them promotions, incentivize them for new purchases, and try to make sure again that they don't become like the customers in cluster two.

So, again, you saw here how you think about the interpretation of the numbers here, and it's very, very helpful to normalize the data and really see which customers behave with respect to what metric are compared to the baseline of the collection of all customers. So next, let's just look on cluster one and three. And again, you want to spot for things that are unusual, that are really, really unique. And one of the things that is really striking about cluster three, it has about 504 customers, is they seem to be really point addicts. Look on how many points they have. They have 1.7, standard deviations more Bonus Miles, the average across all the customers. That's very, very interesting. On the other hand, if you look on where their miles are coming from, they are not coming from flying. They actually slightly below the average of flying. It comes mostly from using miles in other transactions like credit cards and so forth. So, it's very characteristics of them. And when you actually look on cluster number one, one of the things that differentiate between cluster one and cluster three is the tenure. Again, the tenure in the program.

So, cluster three customers have a relatively long time with our loyalty program. Customers in cluster one are relatively new, but if you look on their values with respect to Bonus Miles and Bonus Transactions, they seem to be on the rise. So, potentially again, the customers in cluster one are on the way to become the customers of cluster three in a couple of months or years. Now, what are the insights that you can derive? So first, with respect to the customers in cluster three, this should guide you in terms of both what to offer them, potentially what you want to target is to try and entice them to fly more because that generates for you more revenue. So, you can maybe use the fact that they have a lot of points. They have a very, very big balance, relatively speaking, and they care about points to try to entice them to fly with you. And then with the customers in cluster one, you might want to actually work on them and try to impact them at the early stage of their presence in your program. And because what you want to do is to shift more of their spending to flying with you. So again, hopefully this is another example of how to go about interpreting the results of the clustering algorithms.

And you have to really combine here the quantitative signals that you get from the data and the algorithm with your business intuition of what do they mean. And just to be honest, not all the time this is so obvious. A lot of the time you need to really think hard about what the output of clustering algorithms means to you. And the fact of the matter that sometimes you might be in a position where you don't have a clear insight. So, I don't want to elude you that life is always perfect, and this is always going to be very easy and straightforward. Maybe as a final example, let's just think about the cluster six and seven. And again, you see here on cluster seven, these are really, really engaged customers. A relatively small cluster, but these are the old guard. They're really long time in the program. They are really above the average on every possible dimension. And similarly, that's true for cluster number six. So, these are the customers that are most loyal to you. These are the ones that you want to preserve. These are the ones that you want to make feel special and put a lot of attention to them. So again, another example. And just to practice this, I'm going to leave you thinking about cluster four and eight as an assignment after the module. So, take cluster four and eight, and try to apply the same logic and see what kind of intuition you develop about the results of the algorithm. So, one thing that we still did not discuss was why did we choose eight clusters, and if that is the right number. That's coming next.

Video 4.7: Selecting the Number of K Clusters (6:44)

So, we mentioned that the K-Means algorithm takes as an input a hyperparameter K that corresponds to the desired number of clusters that we ask the algorithm to generate. And this is common in many other algorithms where you have a space of hyperparameters. And then there is an immediate question of how do you select the value that is optimal in respect to the hyperparameter? So, this is what we call a search. You want to search smartly and find the best value of the hyperparameter. And sometimes there are algorithms that have multiple hyper parameters, so the search might be more involved. So, what I would like to explain is how are we going to think about this in the context of the K-Means algorithm. So, in order to do that, we need to have some metric that will tell us something about the quality of the clusters that are generated. Again, this is going to be a numerical metric and it's not going to be necessarily the only metric that we want to look at. As I mentioned before, there are quantitative numerical metrics and there are some of them that you can consider. But most importantly, you always have to apply your own judgment based on the business context of what is the best choice with respect to the setting you are working in. But that said, one of the very common dissimilarity metrics that you can have for within cluster is calculated by taking the sum of the square distances of each observation assigned to the cluster from the cluster centroid. That's a proxy of how similar or dissimilar, depending on how you want to define this, the objects or the observations assigned to the clusters are from each other. If they are very similar to each other, then that similar, dissimilarity metrics should have a low value and vice versa. Now, what is the trade-off that we are thinking about when we decide how to select the number of clusters?

On one hand, if we choose very small number of clusters, then we are running the risks that we're going to get relatively dissimilar customers assigned to the same cluster. And that's going to, perhaps, go against the purpose of what we're trying to do if we have a cluster with two heterogeneous customers. The flip side is that if we choose a high number of clusters, so high value for K, then the clusters are likely to become too specifics. And again, the examples that we saw before of cluster number eight that had only 14 customers, you have to ask yourself, "Seriously, what does it even mean? Is it really practical to even think about 14 customers as a group by itself?" And again, there are no absolute answers to this or absolute truths. It's really about your judgment call. But again, this is the basic trade-off that you're capturing. And let me talk about how you would go about thinking about what could be reasonable K values in general and in this particular example specifically. So, what you do usually is you're trying to run the same K-Means algorithm with different values of K. So, you really do a search. And for each value right, you are going to look on the resulting clusters and you're going to calculate the total cumulative dissimilarity metrics across all the clusters that were generated. And what we see here is the plot of that. And again, this plot can be generated automatically for you by Python; you don't need to do that manually. But what we see here is on the Y axis, the dissimilarity metric. And on the X axis, this is the number of clusters. And naturally as you increase the number of clusters, the dissimilarity is going down.

This is often called script plot where at some point the curve is starting to flat. And this is the area where it starts to flat. We call it the script because it corresponds to really the situation of where the stones are going to be stuck if you think about the analog. And essentially, the intuition is that you really don't want to choose K too small where the graph is very steep or still going down in a very steep manner. And you also don't want to choose it way on the right when the graph is very flat. So, where you want to focus, really, is on this area where it goes from being very steep to being very flat. That's kind of the knee of the curve, sometimes it's called. And within that regime, that's really a matter of choice. And that choice might be guided by really your business interpretation. So, you will probably have to consider multiple solutions and make some arbitrary, relatively arbitrary choice about what do you like the best. Right? So, what this gives you is what not to choose right? It says, "Hey, don't choose very small values in which the curve is very steep. Don't choose very large values when the curve is very flat. Focus on these possible values and then make a judgment call." And this is very typical situations for other settings when you search over hyperparameters. You really want to balance between the complexity of the model and the experimental power of the model. So, this is all about the K-Means algorithm. Again, you can apply it using Python very easily. Python will generate for you the script plot that we just saw and will really help you with all the quantitative metrics, the numerical metrics. But what you still have to do is to do the work of interpreting what the output is and make choices based on your business understanding of the situation. So, next we're going to be talking about yet another algorithm that is very commonly used.

Video 4.8: Method 2: Hierarchical Clustering (AHC) (7:54)

So, the second clustering algorithm we're going to talk about is called a agglomerative hierarchical clustering. Again, this is a very commonly used algorithm, and I would like similarly to the K-Means algorithm to talk about the high-level framework that we're talking about here. So, unlike the K-Means algorithm, we are not going to have any hyperparameter. So, the only input is going to be the observations or the objects like we discussed before, and the initiation will be that each one of the observations will be its own cluster. So, in our case, we're going to start with 3.999 clusters. Now, how are we going to advance and form the clusters?

In each iteration, we're going to reduce the number of clusters by one by merging the two clusters that are closest are most similar to each other. Again, we're going to measure similarity based on the distance between the centroids of each one of the clusters. Since each iteration we're going to decrease it by one, we are guaranteed to converge and we're going to end up with a range of options to choose our clusters. But these options are going to be nested hierarchical and I'm going to show that in just a bit. And before I dive into the details, what I would like to mention that there is a version of these algorithms that unlike this algorithm that goes bottom up, it starts with all the points being one cluster than going up to bottom in a rather similar fashion. So, we're going to focus on the bottom-up version of this algorithm. So, let's just again apply this to the 20 points that we have in our mini dataset. So, I'm going to run you through the algorithm and what you will see on the left are the points, the observations that we have, the 20 observations and how they slowly become consolidated into one cluster. We start again with all each one of the points being its own cluster. What you're going to see on the right hand side is a representation of the hierarchical clustering that we're going to do and what I would like to sort of pay your, you know, just be be clear about that the points here on the right hand side are ordered in a somewhat arbitrary manner that I chose because I know what the output of the algorithm will be and I wanted to be very clear and create a very clear picture. But again, you don't need to worry about all of this in practice because you can have Python do it for you automatically.

So, look at this all of these points, the first iteration would be finding the two points that are closest to each other. These are 2 and 7. And what we're going to do, we're going to merge them. And what you see on the right-hand side, I'm now representing the fact that I emerge 2 and 7 by having a U, flipped U connecting it the height of the U corresponds to the new met dissimilarity metric of the newly merged or newly formed cluster. So, next I'm going to emerge two more points and again, they're going to be merged together and going to see them on the right-hand side. The U connecting them and again, the height of the U. Again, it's a flipped U. The height of the U will correspond to the dissimilarity metric of this newly formed clusters. So, on the Y axis of the right-hand side graph, you're always going to see the level of dissimilarity calculated for each one of the newly formed clusters. So, we continue like that. We create against small clusters by emerging every time the two points that are closest to each other. And again, you see the values of this dissimilarity.

So, that will correspond to the height of the flipped U that we have here. Now we continue. And at some point, we're going to start connecting higher level of clusters. So, now we are not connecting single points. We are connecting already clusters that were formed before. Now, we are further merging them into a larger cluster, and we continue to do that in a greedy manner. And create bigger and bigger clusters until the point where all points are included in one cluster. And again, every time we merge two clusters that corresponds to afflict you with height corresponding to the dissimilarity metric of that particular new reform cluster. So, when you look on the graph on the right-hand side, it has these three shapes, and it has a special name. It's called then dendrogram. And it basically represents to us the algorithm iterations from the very bottom when we merged single points throughout all the clusters that were formed up to the highest cluster, that includes all the points where again, the different heights of the flipped use that you see here corresponds to the dissimilarity of the respective clusters. Now, that still doesn't solve the problem for us. How do we use this to create a certain number of clusters? So, like the K-Means right, we can actually create multiple number of clusters. So, we can create different clustering outputs here. But unlike the K-Means where we have to specify the number of clusters upfront before we run the algorithm, here we can specify the number of clusters post running the algorithm. And how we're going to do that, we're going to do that by basically selecting the level of the similarity we would like to have and each one of them will correspond to a horizontal line. So, you see here multiple lines. And when you select the level of dissimilarity, you basically select the place by which you're going to break this graph into pieces. So, if you take the very top here, the red line, you're going to break the graph into two clusters. If you go more below, you're going to get already three clusters. And as more you take this line down, you create more and more clusters. So, that gives you again a range of choices depending on what the similarity level you would like to consider. You're going to break the graph on the right-hand side into a certain number of clusters and you're going to get a different clustering output. But what is going to be the form here is that these clusters as you move the line up and down are going to be nested in each other. So, one way to think about it that they're going to be either more refined or less refined, more coarse or less coarse depending on the dissimilarity level that you would like to prescribe. Okay. So, now again, the next question is how to select the number of clusters and also what kind of insights do we get when we apply this specific algorithm to the airline dataset that we had?

Video 4.9: Hierarchical Clustering Applied to the Airline Customer Data (2:46)

So, what we see here is the output of applying the algorithm we've just discussed to the airline dataset. And specifically, you see the dendrogram that was created. And again, the question is how do you select the right number of clusters? And we're going to apply exactly the same approach we've seen before for the K-Means algorithm where we're going to try to shift this line down and consider different dissimilarity values, and get in return the number of clusters that are created, and create this tree-like plot. And again, the choice is going to likely be around the knee of this curve. And when you think about the level of the similarity being 32 and you would like to consider only clusters with dissimilarity metric 32 or smaller, that would yield seven final clusters. And again, you see that the numbers here are quite different than what we've seen before. We don't have any very small cluster.

And again, you will need to go and try and interpret it what each one of these clusters actually is telling you. And typically, in real life, what you would try to do is to run different algorithms with different parameters and get different slices and cuts of your data to develop the intuition about the different insights that each algorithm provides and also allow yourself to consider different options before you decide what is the best output that is best fit your business context. What I would like to do next is to talk about some of the pros and cons of these clustering algorithms, but also to highlight to you the fact that we only talked about two examples of two very commonly used the clustering algorithm. And in fact, even for these two algorithms, the K-Means and the AHC, there are many variants that we still did not discuss, but you will be able to read about in the supporting material of the course. And more broadly speaking, there are many other clustering algorithms that have many different ingredients or different choices of ingredients, and this is a very rich class of algorithms.

Video 4.10: Pros and Cons of K-Means and AHC (11:31)

What I would like to do next is discuss some of the pros and cons of the two clustering algorithms that we discussed and use that to give you some intuition when they are likely to work well, when they are maybe likely to have some problems. And also use this opportunity to perhaps hint about a few extensions and variants of these algorithms that we did not discuss in detail, but again, you will be able to read about in the supporting material of the course. So, one thing that is very attractive about these two algorithms that they are relatively very simple to implement. Both conceptually but also computationally when you think about implementing them on a computer, they're really computationally tractable and they scale very well as you have larger and larger number of points. And the other thing that is very attractive about these two algorithms that both of them are guaranteed to converge. And in fact when you think about the k-means algorithm, that remember, it's being initiated by some selection of k descent rates. We talked about the case in which you select descent rates randomly, but you can actually optimize the selection and do what we call a warm start and that could enhance the performance of the algorithm. The other thing that is attractive about them that they can easily be adapted to new points. So, often what you want to have is, "Okay, I created my clusters based on historical data. Now I have new points, how easy this is to assign them to clusters. How easy is it to potentially update my clustering?" So, that's another advantage that they have. They also have some very intuitive variants or extensions or generalizations. For example, the k-means algorithm has a very attractive generalization that is called the k-means gaussian mixture. Again, you can read about this in the supporting material. But these algorithms are not perfect, and I would like to highlight some of the aspects that are relevant here because again, applying analytics is not just about plug and play of algorithms, you need to develop the intuition of when it's appropriate to apply different algorithms. And moreover, when algorithms are likely to perform well versus maybe not that well. So, one thing that is common to both the k-means and the AHC algorithms is that they best perform when the actual clusters have the following attributes that they are circular or spherical-shaped.

Essentially, the points within the observation within each one of the clusters are uncorrelated. They have uniform spread. And when the different clusters have different variants or dispersion, pardon. Or sometimes it's called the weed that basically their distance from each other and their distance from their descent rate is relatively similar across the different clusters that were created. Unfortunately, oftentimes these properties do not hold. For example, many times we have correlations between the variables. We talked about correlations. In which case points that are correlated will be arranged around some elliptical shapes. And that's going to create problems for the algorithms that we discussed. And another thing that is typically happening is that they do have different types of variants. So, the ellipse shapes might actually look quite differently. So, in these situations the k-means algorithm and the AHC algorithms are not likely to perform very well. And what I would like to do is to develop some geometric intuition about these different scenarios.

I'm going to start with a good scenario. So, on the left-hand side in black, you see the actual points and it's very natural for us in two dimensions to identify these two clusters. And indeed, when you apply the clustering algorithms, you're going to get back something that exactly matches the intuition that all of us would have if you looked on the left hand side. So, the right-hand side illustrates the output of the clustering algorithm that seems to be very, very intuitive and really captures what is going on. So, let's just look on the following example. Well, what do we see here, again, if we look at it together and think for a second, we see clearly three clusters. And if you remember our discussion about correlation, each one of these clusters consists of points that seems to be highly correlated with respect to the two variables that we are considering here. They seem to be very negatively correlated, linearly negatively correlated if I want to go back to the terminology that we introduced when we talked about correlation. So again, what you see here on the left-hand side is the original points. On the right-hand side, you see the clustering that you're going to get when you apply these algorithms, and this is where you see that they don't really capture this behavior. They create clusters that do not correspond to what we've seen before or to the intuition that we had when we looked on the original points.

So, this is where this elliptical shape, the high correlation between the points in each cluster, are basically confusing the clustering algorithms that we are using and do not lead to an output that makes a lot of sense. Now, bear in mind this is maybe easy to identify when you look in two dimensions, when you can actually look and see pictures. This is maybe less obvious when you have higher dimensions. So, this is again why no algorithms, and no models are perfect, and you really need to scrutinize the outcomes of what you get from algorithms to understand whether they really capture what you're trying to model or perhaps you need to consider alternative approaches. And again, there is no rule for that. But hopefully, this kind of discussions develop your senses about what questions to ask yourself and how to go about making sure that you are not being food by an algorithm. Here's another example. So, when you look at this, again, I think all of us would agree that we see about three clusters. And again, the intuition would be that these clusters don't really look the same.

If I look on the cluster on the very right-hand side, it consists of observations and objects that are very, very similar to each other, they are very consolidated together. The left-hand side cluster maybe less so but still somewhat similar. And then in the middle we have this big cluster of relatively dispersed points. And again, this is where you have a lot of differences among the different class clusters. When you apply the clustering algorithms that we discussed, you don't get back what you actually hope to get. So, somewhat separating maybe better the left-hand side from the middle but having problem really separating the middle cluster from the right-hand side cluster. Another way to think about it is that the algorithms fail to understand what are the boundaries or detect what are the boundaries between the different clusters? So, they make mistakes around those boundaries. So, they capture most of the right-hand side points but not the points that lie between the cluster in the middle and the right-hand side cluster and similarly to the left-hand side cluster and the middle cluster. So, difficulty to identify the boundaries between clusters. And there are a few more cons to these algorithms that I would like to talk about. So, for the k-means algorithm, L=like you need to choose the number of clusters manually and in the hierarchical algorithm you need to choose them post algorithm.

You could view that as a disadvantage because it leaves some room for some arbitrary choices. Ideally you would like something that has some rigorous way to tell you what the number of clusters should be. Both of these algorithms are also very sensitive to outliers. So, this is why preprocessing of the data is so important. Because if you don't do that and you have too many outliers in your data, you could actually make the algorithm fail and not provide the right outcome. And while they scale very well with the number of observations, they actually not scale very well with the dimension of the data. So again, if you think about the mini dataset example that we discussed, the dimension of the data is two. Each point is described by two attributes, by two features. But more often than not even in the initial dataset that we had, we had six dimensions. And in general, you can actually have highly dimensional data where each observation is described by many, many features. This is where these clustering algorithms are going to likely struggle.

One intuition to about this would be that there are too many degrees of freedom. So, you might get highly unstable clusters depending on your initial points that you select descent rates and that's an undesirable attribute of an algorithm. And this is often called the curse of dimensionality and there are different ways you can go about mitigating this problem, for example, by applying some dimensionality reduction methods. Again, you will be able to read about that in the supporting material of the course. Finally, these algorithms, these clustering algorithms have difficulty to identify non-convex shapes of clusters. All of the clusters that we saw were having a continuous, rounded nice structure. But sometimes you can have clusters that may not consist of nice shapes. They might have very arbitrary boundaries. And this is something that these types of algorithms will struggle to identify. Again, no algorithm is perfect. You need to understand the pros and cons of different algorithms, so you can apply them strategically and also be able to understand the limits of their outputs when you take the outputs and try to interpret it and use it to actually make decisions.

Video 4.11: Module Summary (3:07)

So, let me talk about some of the takeaways of this module. We talked about clustering algorithms that are very commonly used and a very important approach to conduct exploratory and diagnostic analytics. Being an unsupervised method that does not require the outcome features, the labels, it is very handy to use them and to apply them in almost all situations and they can yield major insights that can inform decisions but also the development of additional algorithms. Clustering is a key approach in creating personalization and segmentation. That is a very important key business trends in many industries. Specifically, the ability to specialize your interventions could be marketing interventions, medical interventions, and other interventions into subsets of customers, subsets of patients. This is really really a key capability that clustering is enabling. Additionally, clustering algorithms serve as key enablers in creating capabilities of what we call anomaly detection. This is very important when you think about systems that try to alert against cyber-attacks, financial frauds and again they have broad and emerging applications in many industries.

As I mentioned, clustering algorithms can inform also the development of additional models like predictive models and additional descriptive models. There are many approaches to clustering, but all have similar ingredients. They all rely on a choice of a metric. They all have formation mechanisms and some of them have hyperparameters. Hopefully, this general framework will allow you to easily understand and grasp any clustering algorithm beyond what the specific examples that we just discussed. None of these algorithms is perfect. And often it is important to apply many approaches and then make choices about what works best in the setting that you're working in. And I cannot emphasize more the fact that you have to do very rigorous post analysis, post algorithm analysis to interpret it, what the algorithms is telling you, and what kind of insights you have to take and can take that can inform the business decisions that you're worried about. This is absolutely critical.