MachineLearning-Lecture12

Instructor (Andrew Ng):Okay. Good morning. I just have one quick announcement of sorts. So many of you know that it was about two years ago that Stanford submitted an entry to the DARPA Grand Challenge which was the competition to build a car to drive itself across the desert. So some of you may know that this weekend will be the next DARPA Grand Challenge phase, and so Stanford – the team that – one of my colleagues Sebastian Thrun has a team down in OA now and so they'll be racing another autonomous car.

So this is a car that incorporates many tools and AI machines and everything and so on, and it'll try to drive itself in midst of traffic and avoid other cars and carry out the sort of mission. So if you're free this weekend – if you're free on Saturday, watch TV or search online for Urban Challenge, which is the name of the competition. It should be a fun thing to watch, and it'll hopefully be a cool demo or instance of AI and machines in action.

Let's see. My laptop died a few seconds before class started so let me see if I can get that going again. If not, I'll show you the things I have on the blackboard instead. Okay. So good morning and welcome back. What I want to do today is actually begin a new chapter in 229 in which I'm gonna start to talk about [inaudible]. So [inaudible] today is I'm gonna just very briefly talk about clustering's, [inaudible] algorithm. [Inaudible] and a special case of the EM, Expectation Maximization, algorithm with a mixture of [inaudible] model to describe something called Jensen and Equality and then we'll use that derive a general form of something called the EM or the Expectation Maximization algorithm, which is a very useful algorithm. We sort of use it all over the place and different unsupervised machine or any application. So the cartoons that I used to draw for supervised learning was you'd be given the data set like this, right, and you'd use [inaudible] between the positive and negative crosses and we'd call it the supervised learning because you're sort of told what the right cross label is for every training example, and that was the supervision. In unsupervised learning, we'll study a different problem. You're given a data set that maybe just comprises a set of points. You're just given a data set with no labels and no indication of what the "right answers" or where the supervision is and it's the job of the algorithm to discover structure in the data.

So in this lecture and the next couple of weeks we'll talk about a variety of unsupervised learning algorithms that can look at data sets like these and discover there's different types of structure in it. In this particular cartoon that I've drawn – one has the structure that you and I can probably see as is that this data lives in two different crosses and so the first unsupervised learning algorithm that I'm just gonna talk about will be a clustering algorithm. It'll be an algorithm that looks for a data set like this and automatically breaks the data set into different smaller clusters. So let's see. When my laptop comes back up, I'll show you an example. So clustering algorithms like these have a variety of applications. Just to rattle off a few of the better-known ones I guess in biology application you often cross the different things here. You have [inaudible] genes and they cluster the different genes together in order to examine them and understand the

biological function better. Another common application of clustering is market research. So imagine you have a customer database of how your different customers behave. It's a very common practice to apply clustering algorithms to break your database of customers into different market segments so that you can target your products towards different market segments and target your sales pitches specifically to different market segments.

Something we'll do later today – I don't want to do this now, but you actually go to a website, like, use.google.com and that's an example of a website that uses a clustering algorithm to everyday group related news articles together to display to you so that you can see one of the thousand news articles today on whatever the top story of today is and all the 500 news articles on all the different websites on different story of the day. And a very solid [inaudible] actually talks about image segmentation, which the application of when you might take a picture and group together different subsets of the picture into coherent pieces of pixels to try to understand what's contained in the picture. So that's yet another application of clustering. The next idea is given a data set like this, given a set of points, can you automatically group the data sets into coherent clusters. Let's see. I'm still waiting for the laptop to come back so I can show you an example. You know what, why don't I just start to write out the specific clustering algorithm and then I'll show you the animation later. So this is the called the k-means clustering algorithm for finding clustering's near the inset. The input to the algorithm will be an unlabeled data set which I write as X1, X2, [inaudible] and because we're now talking about unsupervised learning, you see a lot of this as [inaudible] with just the Xs and no cross labels Y. So what a k-means algorithm does is the following.

This will all make a bit more sense when I show you the animation on my laptop. To initialize a set of points, called the cluster centroids, [inaudible] randomly and so if you're [inaudible] of training data are [inaudible] then your cluster centroids, these muse, will also be vectors and [inaudible] and then you repeat until convergence the following two steps. So the cluster centroids will be your guesses for where the centers of each of the clusters are and so in one of those steps you look at each point, XI and you look at which cluster centroid J is closest to it and then this step is called assigning your point XI to cluster J. So looking at each point and picking the cluster centroid that's closest to it and the other step is you update the cluster centroids to be the median of all the points that have been assigned to it. Okay. Let's see. Could you please bring down the display for the laptop? Excuse me. Okay. Okay. There we go. Okay. So here's an example of the k-means algorithm and hope the animation will make more sense. This is an inch chopped off. This is basically an example I got from Michael Jordan in Berkley. So these points in green are my data points and so I'm going to randomly initialize a pair of cluster centroids. So the [inaudible] blue crosses to note the positions of New1 and New2 say if I'm going to guess that there's two clusters in this data. Sets of k-means algorithms as follow. I'm going to repeatedly go to all of the points in my data set and I'm going to associate each of the green dots with the closer of the two cluster centroids, so visually, I'm gonna denote that by painting each of the dots either blue or red depending on which is the closer cluster centroid. Okay.

So all the points closer to the blue cross points are painted blue and so on. The other step is updating the cluster centroids and so I'm going to repeatedly look at all the points that I've painted blue and compute the average of all of the blue dots, and I'll look at all the red dots and compute the average of all the red dots and then I move the cluster centroids as follows to the average of the respective locations. So this is now [inaudible] of k-means on here, and now I'll repeat the same process. I look at all the points and assign all the points closer to the blue cross to the color blue and similarly red. And so now I have that assignments of points to the cluster centroids and finally, I'll again compute the average of all the blue points and compute the average of all the red points and update the cluster centroids again as follows and now k-means is actually [inaudible]. If you keep running these two sets of k-means over and over, the cluster centroids and the assignment of the points closest to the cluster centroids will actually remain the same. Yeah.

Student:[Inaudible]

Instructor (Andrew Ng):Yeah, I'll assign that in a second. Yeah. Okay. So [inaudible]. Take a second to look at this again and make sure you understand how the algorithm I wrote out maps onto the animation that we just saw. Do you have a question?

Student:[Inaudible]

Instructor (Andrew Ng):I see. Okay. Let me answer on that in a second. Okay. So these are the two steps. This step 2.1 was assigning the points to the closest centroid and 2.2 was shifting the cluster centroid to be the mean of all the points assigned to that cluster centroid. Okay. Okay. [Inaudible] questions that we just had, one is, does the algorithm converge? The answer is yes, k-means is guaranteed to converge in a certain sense. In particular, if you define the distortion function to be J of C [inaudible] squared. You can define the distortion function to be a function of the cluster assignments, and the cluster centroids and [inaudible] square distances, which mean the points and the cluster centroids that they're assigned to, then you can show – I won't really prove this here but you can show that k-means is called [inaudible] on the function J. In particular, who remembers, it's called in a sense as an authorization algorithm, I don't know, maybe about two weeks ago, so called in a sense is the algorithm that we'll repeatedly [inaudible] with respect to C. Okay. So that's called [inaudible]. And so what you can prove is that k-means – the two steps of k-means, are exactly optimizing this function with respect to C and will respect a new alternately. And therefore, this function, J of C, new, must be decreasing monotonically on every other variation and so the sense in which k-means converges is that this function, J of C, new, can only go down and therefore, this function will actually eventually converge in the sense that it will stop going down.

Okay. It's actually possible that there may be several clustering's they give the same value of J of C, new and so k-means may actually switch back and forth between different clustering's that they [inaudible] in the extremely unlikely case, if there's multiple clustering's, they give exactly the same value for this objective function. K-means may also be [inaudible] it'll just never happen. That even if that happens, this

function J of C, new will converge. Another question was how do you choose the number of clusters? So it turns out that in the vast majority of time when people apply k-means, you still just randomly pick a number of clusters or you randomly try a few different numbers of clusters and pick the one that seems to work best. The number of clusters in this algorithm instead of just one parameters, so usually I think it's not very hard to choose automatically. There are some automatic ways of choosing the number of clusters, but I'm not gonna talk about them. When I do this, I usually just pick of the number of clusters randomly. And the reason is I think for many clustering problems the "true" number of clusters is actually ambiguous so for example if you have a data set that looks like this, some of you may see four clusters, right, and some of you may see two clusters, and so the right answer for the actual number of clusters is sort of ambiguous. Yeah.

Student:[Inaudible]. [Inaudible] clusters [inaudible] far away from the data point [inaudible] points and the same cluster?

Instructor (Andrew Ng):I see. Right. So yes. K-means is susceptible to [inaudible] so this function, J of C, new is not a convex function and so k-means, sort of called in a sense on the non-convex function is not guaranteed to converge the [inaudible]. So k-means is susceptible to local optimal and [inaudible]. One thing you can do is try multiple random initializations and then run clustering a bunch of times and pick the solution that ended up with the lowest value for the distortion function. Yeah.

Student:[Inaudible]

Instructor (Andrew Ng):Yeah, let's see. Right. So what if one cluster centroid has no points assigned to it, again, one thing you could do is just eliminate it exactly the same. Another thing you can is you can just reinitialize randomly if you really [inaudible]. More questions. Yeah.

Student:[Inaudible] as a norm or can you [inaudible] or infinity norm or –

Instructor (Andrew Ng):I see. Right. Is it usually two norms? Let's see. For the vast majority of applications I've seen for k-means, you do take two norms when you have data [inaudible]. I'm sure there are others who have taken infinity norm and one norm as well. I personally haven't seen that very often, but there are other variations on this algorithm that use different norms, but the one I described is probably the most commonly used there is. Okay. So that was k-means clustering. What I want to do next and this will take longer to describe is actually talk about a closely related problem. In particular, what I wanted to do was talk about density estimation. As another k-means example, this is a problem that I know some guys that worked on. Let's say you have aircraft engine building off an assembly. Let's say you work for an aircraft company, you're building aircraft engines off the assembly line and as the aircraft engines roll off the assembly line, you test these aircraft engines and measure various different properties of it and to use [inaudible] example I'm gonna write these properties as heat and vibrations. Right.

In reality, you'd measure different vibrations, different frequencies and so on. We'll just write the amount of heat produced and vibrations produced. Let's say that maybe it looks like that and what you would like to do is estimate the density of these [inaudible] of the joint distribution, the amount of heat produced and the amount of vibrations because you would like to detect [inaudible] so that as a new aircraft engine rolls off the assembly line, you can then measure the same heat and vibration properties. If you get a point there, you can then ask, "How likely is it that there was an undetected flaw in this aircraft engine that it needs to go undergo further inspections?" And so if we look at the typical distribution of features we get, and we build a model for P of X and then if P of X is very small for some new aircraft engine then that would raise a red flag and we'll say there's an anomaly aircraft engine and we should subject it to further inspections before we let someone fly with the engine. So this problem I just described is an instance of what is called anomaly detection and so a common way of doing anomaly detection is to take your training set and from this data set, build a model, P of X of the density of the typical data you're saying and if you ever then see an example with very low probability under P of X, then you may flag that as an anomaly example.

Okay? So anomaly detection is also used in security applications. If many, very unusual transactions to start to appear on my credit card, that's a sign to me that someone has stolen my credit card. And what I want to do now is talk about specific algorithm for density estimation, and in particular, one that works with data sets like these, that, you know, this distribution like this doesn't really fall into any of the standard text book distributions. This is not really, like, a Gaussian or a [inaudible] explanation or anything. So can we come up with a model to estimate densities that may look like these somewhat unusual shapes? Okay. So to describe the algorithm a bit a more I'm also going to use a one dimensional example rather than a two D example, and in the example that I'm going to describe I'm going to say that let's imagine maybe a data set that looks like this where the horizontal access here is the X axis and these dots represent the positions of the data set that I have. Okay. So this data set looks like it's maybe coming from a density that looks like that as if this was the sum of two Gaussian distributions and so the specific model I'm gonna describe will be what's called a mixture of Gaussian's model.

And just be clear that the picture I have is that when visioning that maybe there were two separate Gaussian's that generated this data set, and if only I knew what the two Gaussian's were, then I could put a Gaussian to my crosses, put a Gaussian to the Os and then sum those up to get the overall density for the two, but the problem is I don't actually have access to these labels. I don't actually know which of the two Gaussian's each of my data points came from and so what I'd like to do is come up with an algorithm to fit this mixture of Gaussian's model even when I don't know which of the two Gaussian's each of my data points came from. Okay. So here's the idea. In this model, I'm going to imagine there's a latent random variable, latent is just synonymous with hidden or unobserved, okay. So we're gonna imagine there's a latent random variable Z and XI, ZI have a joint distribution that is given as follows. We have that P of X, ZI by the chamber of probability, this is always like that. This is always true. And moreover, our [inaudible] is given by the following ZI is distributed multinomial with parameters I. And in the special case where I have just to make sure that two Gaussian's

and ZI will be [inaudible], and so these parameter [inaudible] are the parameters of a multinomial distribution.

And the distribution of XI conditioned on ZI being equal to J so it's P of XI given ZI is equal to J. That's going to be a Gaussian distribution with [inaudible] and covariant sigler. Okay. So this should actually look extremely familiar to you. What I've written down are pretty much the same equations that I wrote down for the Gaussian Discriminant Analysis algorithm that we saw way back, right, except that the differences – instead of, I guess supervised learning where we were given the cross labels Y, I've now replaced Y in Gaussian Discriminant Analysis with these latent random variables or these unobserved random variables Z, and we don't actually know what the values of Z are. Okay. So just to make the link to the Gaussian Discriminant Analysis even a little more explicit – if we knew what the Zs were, which was actually don't, but suppose for the sake of argument that we actually knew which of, say the two Gaussian's, each of our data points came from, then you can use [inaudible] estimation – you can write down the likelihood the parameters which would be that and you can then use [inaudible] estimation and you get exactly the same formula as in Gaussian Discriminant Analysis. Okay. So if you knew the value of the Z, you can write down the law of likelihood and do maximum likeliness this way, and you can then estimate all the parameters of your model. Does this make sense? Raise your hand if this makes sense. Cool. Some of you have questions? Some of you didn't raise your hands. Yeah.

Student: So this ZI is just a label, like, an X or an O?

Instructor (Andrew Ng):Yes. Basically. Any other questions? Okay. So if you knew the values of Z, the Z playing a similar role to the cross labels in Gaussian's Discriminant Analysis, then you could use maximum likeliness estimation parameters. But in reality, we don't actually know the values of the Zs. All we're given is this unlabeled data set and so let me write down the specific bootstrap procedure in which the idea is that we're going to use our model to try and guess what the values of Z is. We don't know our Z, but we'll just take a guess at the values of Z and we'll then use some of the values of Z that we guessed to fit the parameters of the rest of the model and then we'll actually iterate. And now that we have a better estimate for the parameters for the rest of the model, we'll then take another guess for what the values of Z are. And then we'll sort of use something like the maximum likeliness estimation to set even parameters of the model. So the algorithm I'm gonna write down is called the EM Algorithm and it proceeds as follows. Repeat until convergence and the E set, we're going to guess the values of the unknown ZIs and in particular, I'm going to set WIJ. Okay. So I'm going to compute the probability that ZI is equal to J. So I'm going to use the rest of the parameters in my model and then I'm gonna compute the probability that point XI came from Gaussian number J. And just to be sort of concrete about what I mean by this, this means that I'm going to compute P of XI.

This step is sort of [inaudible], I guess. And again, just to be completely concrete about what I mean about this, the [inaudible] rate of P of XI given ZI equals J, you know, well that's the Gaussian density. Right? That's one over E to the – [inaudible] and then

divided by sum from O equals 1 to K of [inaudible] of essentially the same thing, but with J replaced by L. Okay. [Inaudible] for the Gaussian and the numerator and the sum of the similar terms of the denominator. Excuse me. This is the sum from O equals 1 through K in the denominator. Okay. Let's see. The maximization step where you would then update your estimates of the parameters. So I'll just lay down the formulas here. When you see these, you should compare them to the formulas we had for maximum likelihood estimation. And so these two formulas on top are very similar to what you saw for Gaussian Discriminant Analysis except that now, we have these [inaudible] so WIJ is – you remember was the probability that we computed that point I came from Gaussian's. I don't want to call it cluster J, but that's what – point I came from Gaussian J, rather than an indicator for where the point I came from Gaussian J. Okay. And the one slight difference between this and the formulas who have a Gaussian's Discriminant Analysis is that in the mixture of Gaussian's, we more commonly use different covariant [inaudible] for the different Gaussian's.

So in Gaussian's Discriminant Analysis, sort of by convention, you usually model all of the crosses to the same covariant matrix sigma. I just wrote down a lot of equations. Why don't you just take a second to look at this and make sure it all makes sense? Do you have questions about this? Raise your hand if this makes sense to you? [Inaudible]. Okay. Only some of you. Let's see. So let me try to explain that a little bit more. Some of you recall that in Gaussian's Discriminant Analysis, right, if we knew the values for the ZIs so let's see. Suppose I was to give you labeled data sets, suppose I was to tell you the values of the ZIs for each example, then I'd be giving you a data set that looks like this. Okay. So here's my 1 D data set. That's sort of a typical 1 D Gaussian's Discriminant Analysis. So for Gaussian's Discriminant Analysis we figured out the maximum likeliness estimation and the maximum likeliness estimate for the parameters of GDA, and one of the estimates for the parameters for GDA was [inaudible] which is the probability that ZI equals J. You would estimate that as sum of I equals sum of I from 1 to M indicator ZI equals J and divide by N. Okay. When we're deriving GDA, [inaudible]. If you knew the cross labels for every example you cross, then this was your maximum likeliness estimate for the chance that the labels came from the positive [inaudible] versus the negative [inaudible]. It's just a fraction of examples.

Your maximum likeliness estimate for probability of getting examples from cross J is just the fraction of examples in your training set that actually came from cross J. So this is the maximum likeliness estimation for Gaussian's Discriminant Analysis. Now, in the mixture of Gaussian's model and the EM problem we don't actually have these cross labels, right, we just have an unlabeled data set like this. We just have a set of dots. I'm trying to draw the same data set that I had above, but just with the cross labels. So now, it's as if you only get to observe the XIs, but the ZIs are unknown. Okay. So the cross label is unknown. So in the EM algorithm we're going to try to take a guess for the values of the ZIs, and specifically, in the E step we computed WIJ was our current best guess for the probability that ZI equals J given that data point. Okay. So this just means given my current hypothesis, the way the Gaussian's are, and given everything else, can I compute the [inaudible] probability – what was the [inaudible] probability that the point XI actually came from cross J? What is the probability that this point was a cross versus

O? What's the probability that this point was [inaudible]? And now in the M step, my formula of estimating for the parameters [inaudible] will be given by 1 over M sum from I equals 1 through M, sum of WIJ. So WIJ is right. The probability is my best guess for the probability that point I belongs to Gaussian or belongs to cross J, and [inaudible] using this formula instead of this one. Okay.

And similarly, this is my formula for the estimate for new J and it replaces the WIJs with these new indicator functions, you get back to the formula that you had in Gaussian's Discriminant Analysis. I'm trying to convey an intuitive sense of why these algorithm's make sense. Can you raise your hand if this makes sense now? Cool. Okay. So what I want to do now is actually present a broader view of the EM algorithm. What you just saw was a special case of the EM algorithm for specially to make sure of Gaussian's model, and in the remaining half hour I have today I'm going to describe a general description of the EM algorithm and everything you just saw will be devised, sort of there's a special case of this more general view that I'll present now. And as a pre-cursor to actually deriving this more general view of the EM algorithm, I'm gonna have to describe something called Jensen's and Equality that we use in the derivation.

So here's Jensen's and Equality. Just let F be a convex function. So a function is a convex of the second derivative, which I've written F prime prime to [inaudible]. The functions don't have to be differentiatable to be convex, but if it has a second derivative, then F prime prime should be creating a 0. And let X be a random variable then the F applied to the expectation of X is less than the equal of 2D expectation of F of F. Okay. And hopefully you remember I often drop the square back, so E of X is the [inaudible], I'll often drop the square brackets.

So let me draw a picture that would explain this and I think – Many of my friends and I often don't remember is less than or great than or whatever, and the way that many of us remember the sign of that in equality is by drawing the following picture. For this example, let's say, X is equal to 1 with a probability of one-half and X is equal to 6 worth probability 1 whole. So I'll illustrate this inequality with an example. So let's see. So X is 1 with probability one-half and X is 6 with probably with half and so the expected value of X is 3.5. It would be in the middle here. So that's the expected value of X. The horizontal axis here is the X axis. And so F of the expected value of X, you can read of as this point here. So this is F of the expected value of X. Where as in contrast, let's see. If X is equal to 1 then here's F of 1 and if X equaled a 6 then here's F of 6 and the expected value of F of X, it turns out, is now averaging on the vertical axis. We're 50 percent chance you get F of 1 with 50 percent chance you get F of 6 and so these expected value of F of X is the average of F of 1 and F of 6, which is going to be the value in the middle here. And so in this example you see that the expected value of F of X is greater than or equal to F of the expected value of X. Okay.

And it turns out further that if F double prime of X makes [inaudible] than Z row, if this happens, we say F is strictly convex then the inequality holds an equality or in other words, E of F of X equals F of EX, if and only if, X is a constant with probability 1. Well, another way of writing this is X equals EX. Okay. So in other words, if F is a strictly

convex function, then the only way for this inequality to hold its equality is if the random variable X always takes on the same value. Okay. Any questions about this? Yeah.

Student:

[Inaudible]

Instructor (Andrew Ng):Say that again?

Student: What is the strict [inaudible]?

Instructor (Andrew Ng):I still couldn't hear that. What is –

Student: What is the strictly convex [inaudible]?

Instructor (Andrew Ng):Oh, I see. If double prime of X is strictly greater than 0 that's my definition for strictly convex. If the second derivative of X is strictly greater than 0 then that's what it means for F to be strictly convex.

Student:

[Inaudible]

Instructor (Andrew Ng):I see. Sure. So for example, this is an example of a convex function that's not strictly convexed because there's part of this function is a straight line and so F double prime would be zero in this portion. Let's see. Yeah. It's just a less formal way of saying strictly convexed just means that you can't have a convex function within a straight line portion and then [inaudible]. Speaking very informally, think of this as meaning that there aren't any straight line portions. Okay. So here's the derivation for the general version of EM. The problem was face is as follows. We have some model for the joint distribution of X of Z, but we observe only X, and our goal is to maximize the law of likelihood of the parameters of model.

Right. So we have some models for the joint distribution for X and Z and our goal is to find the maximum likeliness estimate of the parameters data where the likelihood is defined as something equals 1 to M [inaudible] probably of our data as usual. And here X is parameterized by data is now given by a sum over all the values of ZI parameterized by data. Okay. So just by taking our model of the joint distribution of X and Z and marginalizing out ZI that we get P of XI parameterized by data. And so the EM algorithm will be a way of performing this maximum likeliness estimation problem, which is complicated by the fact that we have these ZIs in our model that are unobserved. Before I actually do the math, here's a useful picture to keep in mind. So the horizontal axis in this cartoon is the [inaudible] axis and there's some function, the law of likelihood of theta zero that we're trying to maximize, and usually maximizing our [inaudible] derivatives instead of the zero that would be very hard to do. What the EM algorithm will do is the following. Let's say it initializes some value of theta zero, what the EM algorithm will

end up doing is it will construct a lower bound for this law of likelihood function and this lower bound will be tight [inaudible] of equality after current guessing the parameters and they maximize this lower boundary with respect to theta so we'll end up with say that value. So that will be data 1. Okay.

And then EM algorithm look at theta 1 and they'll construct a new lower bound of theta and then we'll maximize that. So you jump here. So that's the next theta 2 and you do that again and you get the same 3, 4, and so on until you converge to local optimum on [inaudible] theta function. Okay. So this is a cartoon that displays what the EM algorithm will do. So let's actually make that formal now. So you want to maximize with respect to theta sum of [inaudible] – there's my theta, so this is sum over 1 [inaudible] sum over all values of Z. Okay. So what I'm going to do is multiply and divide by the same thing and I'm gonna write this as Q – okay. So I'm going to construct the probability distribution QI, that would be over the latent random variables ZI and so these QI would get distribution so each of the QI would bring in a 0 and sum over all the values of ZI of QI would be 1, so these Qs will be a probability distribution that I get to construct. Okay. And then I'll later go describe the specific choice of this distribution QI. So this QI is a probability distribution over the random variables of ZI so this is [inaudible]. Right. I see some frowns. Do you have questions about this? No. Okay.

So the log function looks like that and there's a concave function so that tells us that the log of E of X is greater than and equal to the expected value of log X by the other concave function form of Jensen's and Equality. And so continuing from the previous expression, this is a summary of a log and an expectation, that must therefore be greater than or equal to the expected value of the log of that. Okay. Using Jensen's and Equality. And lastly just to expand out this formula again. This is equal to that. Okay. Yeah.

Student:

[Inaudible]

Instructor (Andrew Ng):

[Inaudible]

Student:

[Inaudible]. Yeah. Okay. So this has the [inaudible] so let's say

Random variable Z, right, and Z has some distribution. Let's denote it G. And let's say I have some function G of Z. Okay. Then by definition, the expected value of G of Z, by definition, that's equal to sum over all the values of Z, the probability of that value of Z times Z of G. Right. That's sort of the definition of a random variable. And so the way I got from this step to this step is by using that. So in particular, now, I've been using distribution QI to denote the distribution of Z, so this is, like, sum over Z of P of Z times

[inaudible]. And so this is just the expected value with respect to a random variable Z joined from the distribution Q of G of Z. Are there questions?

Student: So in general when you're doing maximum likelihood estimations, the likelihood of the data, but in this case you only say probability of X because you only have observed X whereas previously we said probability of X given the labels?

Instructor (Andrew Ng):Yes. Exactly. Right. Right. [Inaudible] we want to choose the parameters that maximizes the probability of the data, and in this case, our data comprises only the Xs because we don't reserve the Zs, and therefore, the likelihood of parameters is given by the probability of the data, which is [inaudible]. So this is all we've done, right, we wanted to maximize the law of likelihood of theta and what we've done, through these manipulations, we've know constructed a lower bound on the law of likelihood of data. Okay.

And in particular, this formula that we came up, we should think of this as a function of theta then, if you think about it, theta are the parameters of your model, right, if you think about this as a function of your parameters theta, what we've just shown is that the law of likelihood of your parameters theta is lower bounded by this thing. Okay. Remember that cartoon of repeatedly constructing a lower bound and optimizing the lower bound. So what we've just done is construct a lower bound for the law of likelihood for theta. Now, the last piece we want for this lower bound is actually we want this inequality to hold with equality for the current value for theta.

So just refrain back to the previous cartoon. If this was the law of likelihood for theta, we'd then construct some lower bound of it, some function of theta and if this is my current value for theta, then I want my lower bound to be tight. I want my lower bound to be equal to the law of likelihood of theta because that's what I need to guarantee that when I optimize my lower bound, then I'll actually do even better on the true objective function. Yeah.

Student:

How do [inaudible]

Instructor (Andrew Ng):Excuse me. Yeah. Great question. How do I know that function is concave? Yeah. I don't think I've shown it. It actually turns out to be true for all the models we work with. Do I know that the law of bound is a concave function of theta? I think you're right. In general, this may not be a concave function of theta. For many of the models we work with, this will turn out to be a concave function, but that's not always true. Okay. So let me go ahead and choose a value for Q. And I'll refer back to Jensen's and Equality. We said that this inequality will become an equality if the random variable inside is a constant. Right. If you're taking an expectation with respect to constant valued variables.

So the QI of ZIs must sum to 1 and so to compute it you should just take P of XI, ZI, parameterized by theta and just normalize the sum to one. There is a step that I'm skipping here to show that this is really the right thing to do. Hopefully, you'll just be convinced it's true. For the actual steps that I skipped, it's actually written out in the lecture notes. So you then have the denominator, by definition, is that and so by the definition of conditional probability QI of ZI is just equal to P of ZI given XI and parameterized by theta. Okay.

And so to summarize the algorithm, the EM algorithm has two steps. And the E step, we set, we choose the distributions QI, so QI of ZI will set to be equal to a P of ZI given [inaudible] by data. That's the formula we just worked out. And so by this step we've now created a lower bound on the law of likelihood function that is now tight at a current value of theta. And in the M step, we then optimize that lower bound with respect to our parameters theta and specifically to the [inaudible] of theta. Okay. And so that's the EM algorithm. I won't have time to do it today, but I'll probably show this in the next lecture, but the EM algorithm's that I wrote down for the mixtures of Gaussian's algorithm is actually a special case of this more general template where the E step and the M step responded. So pretty much exactly to this E step and this M step that I wrote down. The E step constructs this lower bound and makes sure that it is tight to the current value of theta. That's in my choice of Q, and then the M step optimizes the lower bound with respect to [inaudible] data. Okay. So lots more to say about this in the next lecture. Let's check if there's any questions before we close. No. Okay. Cool. So let's wrap up for today and we'll continue talking about this in the next session.

[End of Audio]

Duration: 76 minutes