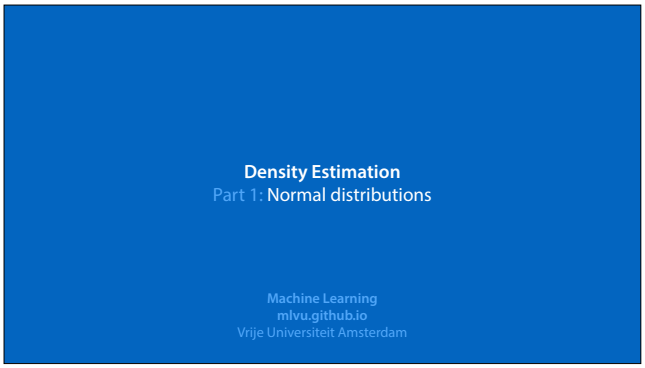
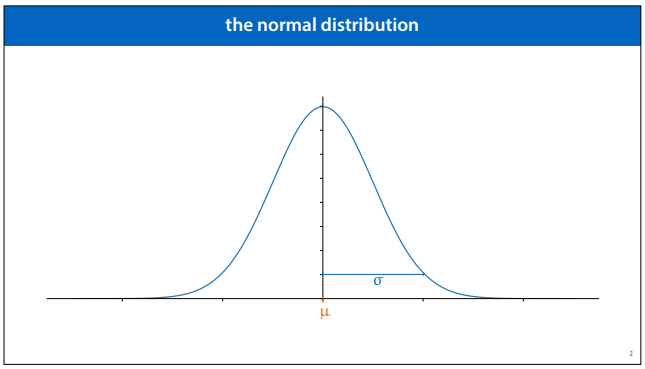


In a few videos so far, we made use of the Normal distribution, assuming that you'd seen it before, and that you know more or less what its properties are.

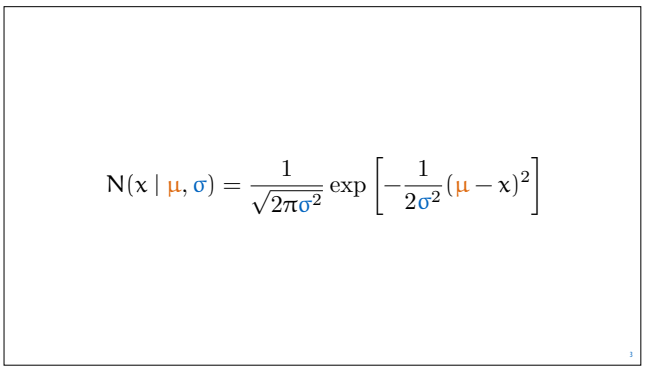
In this video, we'll take a step back and look at the normal distribution from first principles. It's an important tool in what is coming up in this lecture and the next, so we need to make ourselves eminently comfortable with the ins and outs.



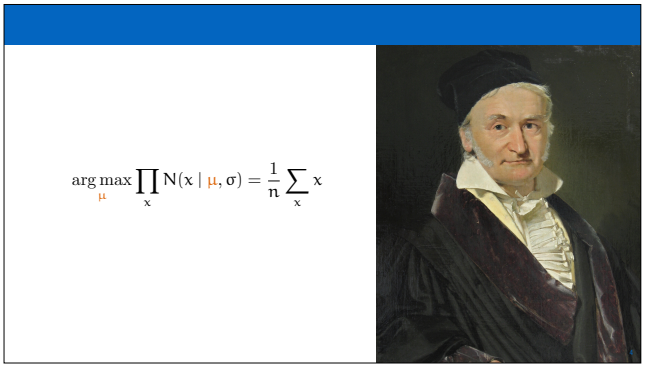
Here is the one dimensional normal distribution.

One of the reasons that the normal distribution is so popular is that it has a definite *scale*. If I look at something like income distribution, the possible values cover many orders of magnitude, from 0 to billions. This is not the case with normally distributed phenomena. Take height for instance: no matter how many people I check, I will never see a person that is 5 meters tall.

The normal distribution is a way of saying: I'm not sure about the value of x, and I can't definitely rule any value out, but I'm almost certain it's near this particular value.



This is the formula for the probability density function of the one-dimensional normal distribution. It looks very imposing, but if you know how to interpret it, it's actually not that complicated. Let's first see where it came from, and then try to figure out what all the different parts mean.



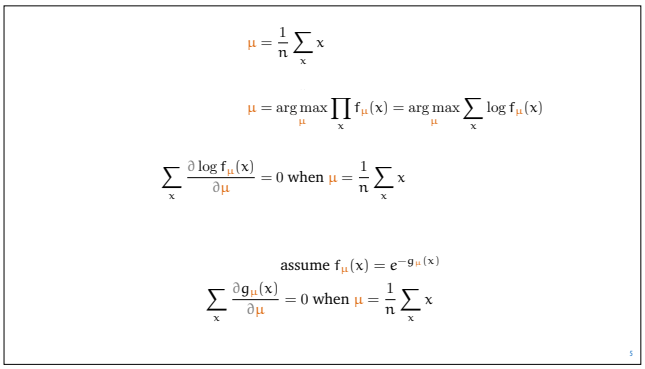
The Normal distribution was invented, or perhaps discovered is a better word, by Carl Friedrich Gauss, undisputably one of the three greatest mathematicians in history.

Gauss was working as an astronomer, and trying to estimate the positions and velocities from fallible measurements.

We've already seen that if we have a bunch of values, such as measuments of some quantity, and if they are normally distributed, then their maximum likelihood estimate works out as the mean. Of course, this is not the order in which things were worked out historically. Taking the means of a sequence of measurements has

been done since at least the third century BC, and the Normal distribution only emerged at the end of the 18th century.

For Gauss, the challenge was to make this derivation backwards. He knew the mean was an effective method of estimation. Taking the principle of maximum likelihood as a given, what kind of distribution would give rise to the means as an effective estimator?



Let's see if we can reconstruct some of his thought process. We may not have Gauss' genius, but we do have the benefit of having taking this particular walk before in the other direction.

We take the arithmetic mean as a given, together with the principle of maximum likelihood. We'll not assume all the finer details of probability, since which Gauss did not have access to them either. Let's just say that mu represents the truth, which we are trying to measure, and there is some function f of a measurement x, in which mu is a constant, that is larger for the measurements we are more likely to encounter.

Assume that we get some measurements, and choose the mu such that the product of these values over our series of measurements is maximal. What properties should f have for us to end up taking the mean?

We'll start, by taking the logarithm of f. This does not change the optimum, and turns our product into a sum. It's easier to work with and it already brings us closer to the computation of the mean.

We know, and Gauss knew, that the derivative of our objective function is zero at the optimum. So the derivative of our objective function should be zero when mu is the mean of our observations.

To get rid of the logarithm, let's make a further assumption: that f is simple some other function g, but exponentiated (and taking the negative). That simplifies things: the sum of the derivatives g for our observations x, should equal 0 when mu is equal to the mean. when f is large g is small so we can think of g as a metric of how unlikely (and hopefully bad) our measurement is.

$$\sum_x \frac{\partial g_\mu(x)}{\partial \mu} = 0 \text{ when } \mu = \frac{1}{n}\sum_x x$$

$$\mu = \frac{1}{n}\sum_x x$$

$$n\mu = \sum_x x$$

$$n\mu - \sum_x x = 0$$

$$\sum_x \mu - \sum_x x = 0$$

$$2\sum_x (x - \mu) = 0$$

$$g_\mu(x) = (x-\mu)^2$$

$$f_\mu(x) = e^{-(x-\mu)^2}$$

$$p_\mu(x) \propto e^{-(x-\mu)^2}$$

Our job is now to take the equation describing the mean and to rewrite it so that it aligns with the equation on the left, and we can read of what the derivative of g is.

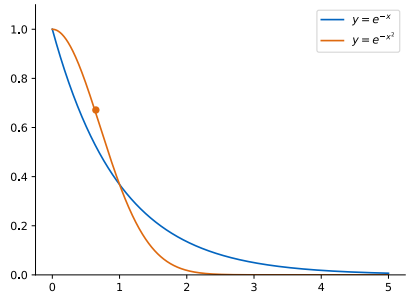
With a little rewriting, we see that this implies that the sum of the differences between mu and the various xs, the sum of our measurement errors, should equal one. If we make g the square of a measurement error, we see that its derivative leads to the desired question (we can freely add a constant, since we are setting the equation equal to zero.

This means that our f function is exp(x - mu)^2. And with that, we have the basic property of of our distribution. It's not a probability density, since it doesn't sum to one, but multiplying it by a constant won't change the minimum,so we can save that for later.

$$p_\mu(x) \propto e^{-(x-\mu)^2}$$

So, if we strip away the complexity, this is the only really important part of the normal distribution. A negative exponential for the squared distance to the mean.

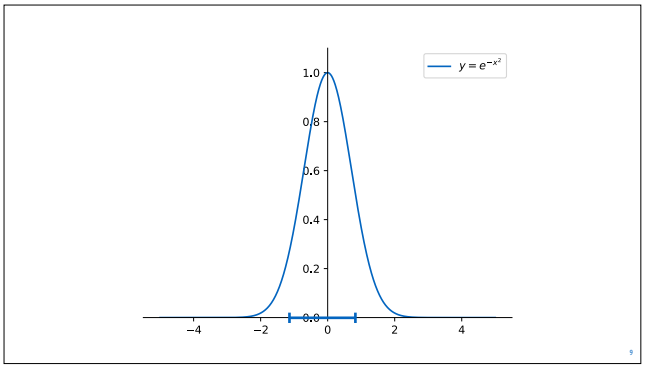
Everything else adding features, and making sure it sums to one.



What does this curve look like? To illustrate, we'll set the mean to zero.

Remember, we described the normal distribution as having a **definite scale**. This means that we first need to make outliers incredibly unlikely. An exponentially decaying function like e^{-x} gives us that propert. Each step of size 1 we take to the right more than halves the probability density. After seven steps it's one thousandth of where we started, afterfourteen steps one millionth, and after twenty-one steps one-billionth.

Taking the negative exponential of the square, as our function e^{-x^2} does, results in an even stronger decay, and it has two more benefits.

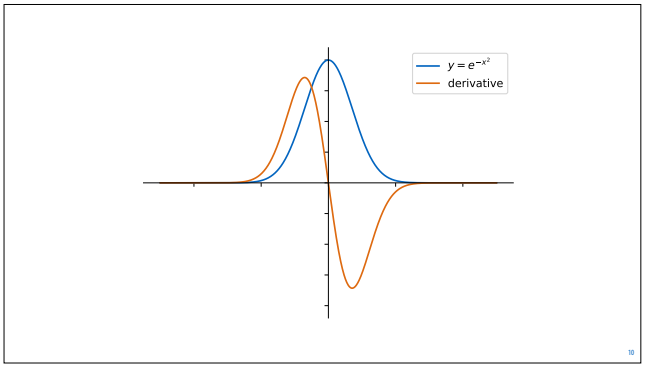


First, the function flattens out at the peak, giving us a nice bell-shaped curve, where exp(-x) has an ugly discontinuity at the top (if we make it symmetric).

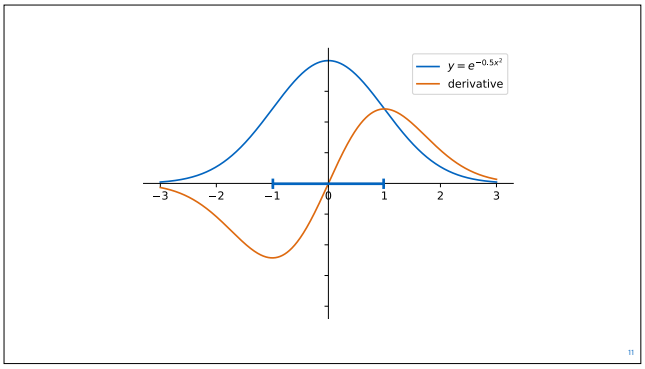
Furthermore, it has an **inflection point**: the point (around 0.7) where the curve moves from decaying with increasing speed to decaying with decreasing speed. We can take this as a point of reference on the curve: to the left of this point, the curve looks fundamentally different than to the right of it. With the exponential decay, the function keeps looking the same as we move from left to right, every seven steps we take, the density halves. With the negative exponential of the square, there is a place where the function keeps dropping ever more quickly, and a place where it starts dropping ever more slowly. We can use this to, as it were, decide where we are on the graph, which will help us determine a characteristic range of values for our distribution.

The two inflection points are natural choices for the **range** bounding the "characteristic" scale of this distribution. The range of outcomes which we can reasonably expect. This is a little subjective: any outcome is possible, and the characteristic scale depends on what we're willing to call unlikely. But given the subjectivity, the inflection points are as good a choice as anything.

If you follow Gauss' logic for the *median* rather than the mean, you'll see that the corresponding distribution is one with simple exponential decay (the so called Laplace distribution). This fits our intuition that when our data doesn't have a definite scale (like income), we should use the median rather than the mean.

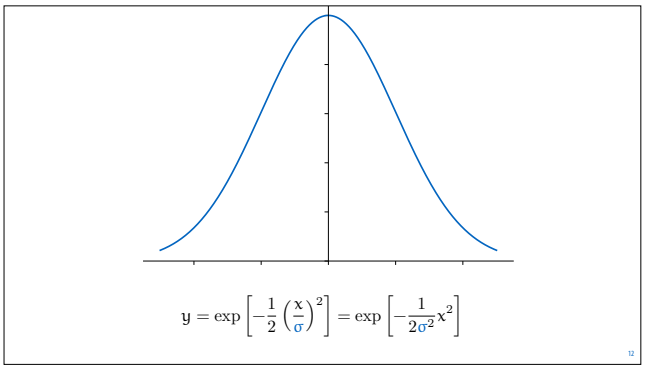


The inflection points are the peaks of the derivative.



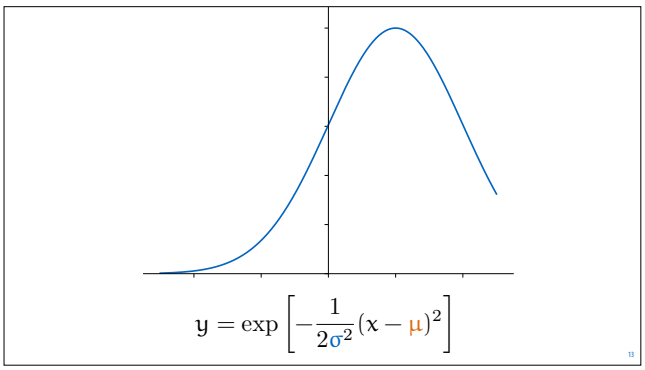
If we add a 0.5 multiplier to the inputs, the inflection points hit -1 and 1 exactly. This gives us a curve for which the characteristic scale is [-1, 1], which seems like a useful starting point (we can rescale this later to any range we require).

Additionally, when we now derive the mean, the exponent 2 will cancel out against this one half, which means we don't even need to introduce the constant 2 multiplier.



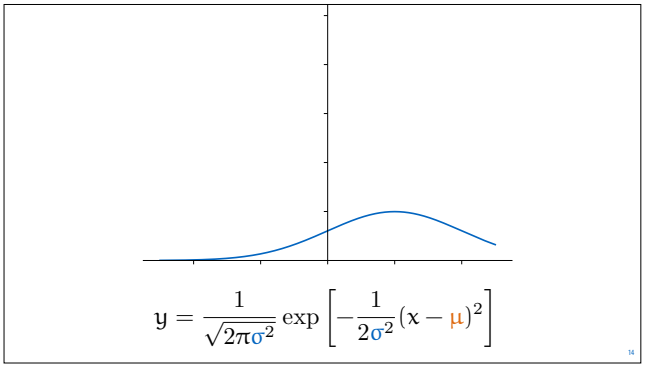
To change the scale, we add a parameter sigma. This will end up representing the the standard deviation, but for now, we can just think of it as a way to make the bell wider or narrower.

The square of the standard deviation is the variance. Either can be used as a parameter.



We can now add the mean back in, with parameter mu. This shifts the center of the bell forward or backward to coincide without the desired mean.

Note that shifting a curve forward by mu points is the same as shifting the coordinates *backward* by three points. Like wise, we can think of the mulitplication by sigma as keeping the curve the same, but just drawing the ticks on the horizontal axis closer together or further apart.



Finally, to make this a proper probability density function, we need to make sure the area under the curve sums to one.

This is done by integrating over the whole real number line. If the result is Z, we divide the function at every point by Z. This gives us a function that sums to 1 over the whole of its domain. For this function, it turns out that integrating results in an area equal to the square of two times pi times the standard

normalization

squared exponential decay

$$N(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2\sigma^2}(\mu - x)^2\right]$$

squared error

scale modifier

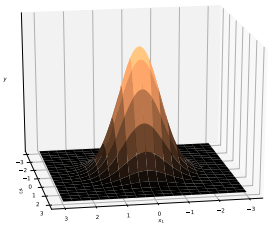
So that's what the different parts of the normal distribution do.

---

multivariate normal (MVN)

$$N(x \mid \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left[-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)\right]$$

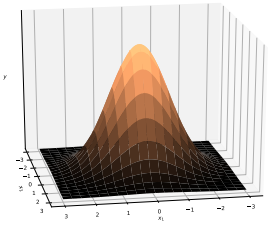
We can do the same thing in multiple dimensions. This gives us the **multivariate normal distribution**. We'll quickly run through how the different parts generalize to higher dimensions.



$$y = e^{-||\mathbf{x}||^2}$$

We start by defining a curve that decays squared-exponentially *in all directions*. Think of this as spinning our original function around the origin.

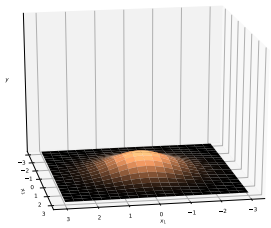
The inflection points now become a kind of "inflection circle", where the derivative peaks. Inside this circle lie the most likely outcomes for our distribution.



$$y = e^{-\frac{1}{2}\mathbf{x}^T\mathbf{x}}$$

To give the inflection circle radius 1, we rescale the exponent, as we did before before.

Note that the square of the norm is equal to the dot product of a vector with itself, so we write that instead.



$$y = \frac{1}{\sqrt{(2\pi)^d}} \exp\left[-\frac{1}{2}\mathbf{x}^T\mathbf{x}\right]$$

This time we'll normalize first, and then introduce the parameters.

This function is the probability density function of the ***standard*** **MVN** (zero mean, and variance one in every direction).

To define the density functions for other distributions we'll use a special trick. We'll start with this one, and apply a linear transformation. We'll see that the parameters of the linear transformation then become the parameters of the result multivariate normal.

---

introducing parameters

If

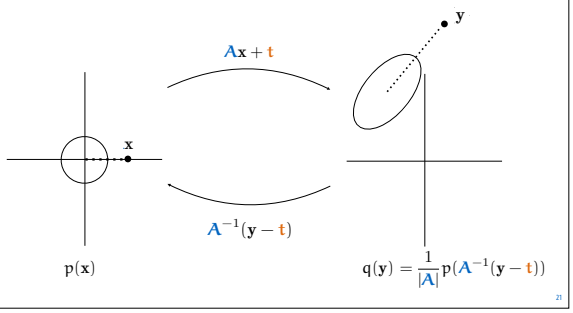
X ~ N(0, I)

Y = **A**X + **t**, and

p(x) is the density of X, then

what is the density q(y) of Y?

Here's the formal way of doing that. Imagine that we sample a point X from the standard normal distribution. We then transform that point by a linear transformation defined by matrix **A** and vector **t**, resulting in a vector Y. What then is the density function that defines our probability on Y?



So, if we transform a sample from x into y, we get a new distribution, with a new mean, and our inflection circle becomes an inflection ellipse. Say we pick a point y. What's the probability density for seeing that point after the transformation?

Consider, that the probability of ending up inside the inflection circle on the left must be the same as the probability of ending up inside the ellipse on the right. And this is true for any contour line we draw:we get a circle on the left, and an ellipse of the right, and the probabilities for both must be the same.

This suggests that if we pick a point y on the right, and we want to know its density, we can reverse the transformation, to give us the equivalent point x on the left. The density of that point under p(x), the standard normal distribution, must be related to the density of y under q(y). In fact, it turns out that q(y) is proportional to the density of the reverse-transformed point.

The only thing we need to correct for, is the fact that the matrix A shrinks of inflates the bell curve, so that the volume below it does not integrate to one anymore. The amount by which a matrix inflates space is its determinant. So, if we divide the resulting density by the determinant, we find a properly normalized density.

This trick is a simple case of integration by substitution **https://en.wikipedia.org/wiki/Integration_by_substitution#Application_in_probab**

ility In the context of probability it is also called the Change of Variable Theorem.

When dealing with normal distributions it can be very helpful to think of them as linear transformations of the standard normal distribution.

$$q(\mathbf{y}) = \frac{1}{|\mathbf{A}|} p(\mathbf{A}^{-1}(\mathbf{y} - \mathbf{t})) \qquad p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^d}} \exp\left[-\frac{1}{2}\mathbf{x}^T\mathbf{x}\right]$$

$$\begin{aligned} q(\mathbf{y}) &= \frac{1}{|\mathbf{A}|} \frac{1}{\sqrt{(2\pi)^d}} \exp\left[-\frac{1}{2}(\mathbf{A}^{-1}(\mathbf{y} - \mathbf{t}))^T(\mathbf{A}^{-1}(\mathbf{y} - \mathbf{t}))\right] \\ &= \frac{1}{\sqrt{|\mathbf{A}\mathbf{A}^T|}} \frac{1}{\sqrt{(2\pi)^d}} \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{t})^T\mathbf{A}^{-1^T}\mathbf{A}^{-1}(\mathbf{x} - \mathbf{t})\right] \\ &= \frac{1}{\sqrt{(2\pi)^d|\mathbf{A}\mathbf{A}^T|}} \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{t})^T(\mathbf{A}\mathbf{A}^T)^{-1}(\mathbf{x} - \mathbf{t})\right] \end{aligned}$$

$$\boldsymbol{\Sigma} = \mathbf{A}\mathbf{A}^T \quad \boldsymbol{\mu} = \mathbf{t}$$

22

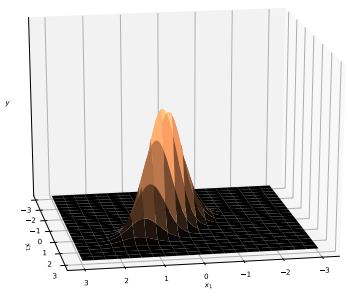
We fill in $\mathbf{A}^{-1}(\mathbf{x}\text{-}\mathbf{t})$ to transform our standard normal distribution pdf to the the pdf transformed by A and t. We set mu equal to t.

Using the basic properties of the determinant, the transpose and the inverse (you can look these up on wikipedia), we can rewrite the result to the pdf we expect.

$$N(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right]$$

23

Here is the final functional form in terms of the mean and the covariance matrix.



24

sampling

$N(0,1)$: see `numpy.random.randn`

$N(\mu, \sigma^2)$: $X\sigma + \mu$ with $X \sim N(0,1)$

$N^d(\mathbf{0}, \mathbf{1})$: $\begin{pmatrix} X_1 \\ \vdots \\ X_d \end{pmatrix}$ with $X_i \sim N(0,1)$

$N^d(\mu, \Sigma) = AX + \mu$

with $X \sim N^d(\mathbf{0}, \mathbf{I})$ and $\Sigma = AA^T$

One benefit of the transformation approach we used, is that it's now very easy to work out how to **sample from an MVN**. We can take the following approach.

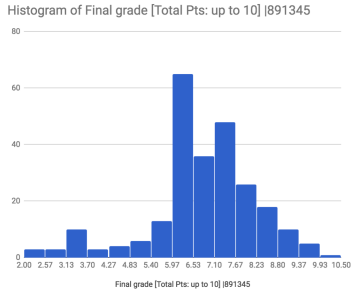
We'll take sampling form a univariate standard normal as read (it's usually done by an algorithm called the Box-Muller transform, if you're interested).

We can transform a sample from the standard normal distribution into a sample from a distribution with given mean and variance as shown above.

We can sample from the standard MVN by stacking d samples from the univariate normal in a vector.

We can then transform this to a sample from an MVN with any mean or covariance matrix by finding A and transforming as appropriate.

Gaussian mixture model



Here is the grade distribution from a few years ago. It doesn't look very normally distributed (unless you squint alot). The main reason it doesn;t normally distributed, it because it has multiple peaks, known as **modes**. This often happens when your population consists of a small number of clusters, each with their own distribution.

In this year, the student population was mainly made up of two programs. We can imagine that students from one program found the course more more difficult than students from the other program, and that the peak around 3.5 was that of students who only partially finished the course. This gives us three sub-populations, each with their own normal distribution. The problem is, we observe only the grades, and we can't tell which program a student is in.

We can describe this distribution with a *mixture* of normal distributions.

(These days the student population consists of many more programs and background, so the grade distribution looks more normal).

GMM (with three components)

three components:

$$\mathrm{N}(\mu_1, \Sigma_1), \mathrm{N}(\mu_2, \Sigma_2), \mathrm{N}(\mu_3, \Sigma_3)$$

three weights

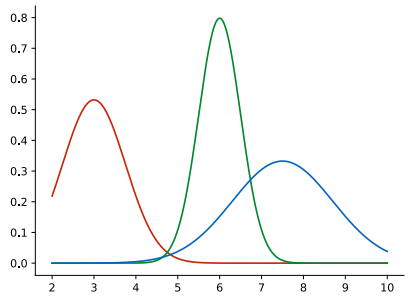
$$w_1, w_2, w_3 \quad \text{with} \quad \sum w_i = 1$$

27

Here is how to define a mixture model. We define three separate normal distributions, each with their own parameters. We'll call these **components**.

In addition, we also define **three weights**, which we require to sum to one. These indicate the relative contributions of the components to the total. In our example, these would be the sizes of the three subpopulations of students, relative to the total.

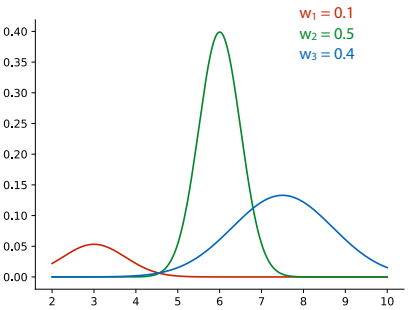
To sample from this distribution, we pick one of the components according to the weights, and then sample a point from that component.



28

We'll mostly look at the model in 1D, but it works the same for any dimensionality.

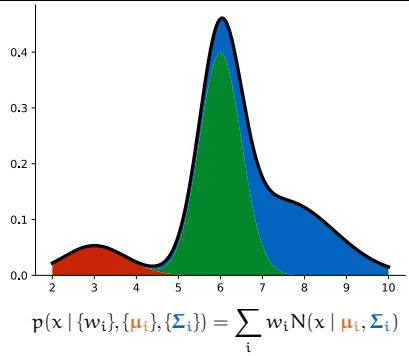
Here's three components that might broadly correspond to what we saw in the grade histogram.



29

We scale each by their component weight. Since the areas under these curves each were one before we multiplied by the weights, they are now 0.1, 0.5 and 0.4 respectively.

That means that if we sum these functions, the result is combined function with an area of one: a new probability density.



30

That looks like this. For each x we observe, each component could be responsible for producing that x, but the different components have different probabilities of being **responsible** for each x.

$$\arg\max_{\theta} \ln p(X \mid \theta)$$

In the next video, we'll take another look at how to fit some of these distributions to data.

Density Estimation
Part 2: Maximum likelihood estimators

Machine Learning
mlvu.github.io
Vrije Universiteit Amsterdam

Now that we have a better understanding of why the normal distribution looks the way it does, let's have another look at fitting one to our data.

---

notation

$$N(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2\sigma^2}(x-\mu)^2\right]$$

$$X \sim N(\mu, \sigma)$$

---

maximum likelihood for the mean

$$\frac{\partial \ln p(X \mid \theta)}{\partial \mu} = \sum_x \frac{\partial \left[\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(x-\mu)^2\right]}{\partial \mu}$$

$$\mu = \frac{1}{n}\sum_x x$$

We've seen the maximum likehood estimator mu already. It's the artihmetic mean. In fact, as we saw in the last video, this estimator was broadly used for thousands of years before Gauss worked out the distribution for which is a maximum likelihood estimator.

maximum likelihood for the variance

$$
\begin{aligned}
&\arg\max_{\sigma} \sum_x \ln N(x \mid \mu, \sigma) \\
&= \arg\max_{\sigma} \sum_x \ln \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2\sigma^2}(x-\mu)^2\right] \\
&= \arg\max_{\sigma} \sum_x \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(x-\mu)^2 \\
&= \arg\max_{\sigma} -\sum_x \ln\sqrt{2\pi} + \ln\sigma - \frac{1}{2\sigma^2}(x-\mu)^2 \\
&= \arg\max_{\sigma} -n\ln\sigma - \frac{1}{2\sigma^2}\sum_x (x-\mu)^2
\end{aligned}
$$

For the sake of completeness, let's work out the maximum likelihood estimator for the variance/ standard deviation

---

maximum likelihood for the variance

$$
\begin{aligned}
\frac{\partial -n\ln\sigma - \frac{1}{2\sigma^2}\sum_x (x-\mu)^2}{\partial\sigma} &= 0 \\
-\frac{n}{\sigma} + \frac{1}{\sigma^3}\sum_x (x-\mu)^2 &= 0 \\
-n + \frac{1}{\sigma^2}\sum_x (x-\mu)^2 &= 0 \\
\frac{1}{\sigma^2} &= \frac{n}{\sum_x (x-\mu)^2} \\
\sigma^2 &= \frac{1}{n}\sum_x (x-\mu)^2
\end{aligned}
$$

warning: biased

This is the maximum likelihood estimator for the variance. Taking the square on both sides gives us the estimator for the standard deviation.

Note that it turns out that this estimator is biased: if we repeatedly sammple a dataset and compute the variance, our average error in the estimate doesn't go to zero.

For an unbiased estimator, we need to divide by n-1 instead. For large data, the difference has minimal impact.

---

estimators for *weighted* data

$$\arg\max_{\theta} \sum_x \omega_i \ln p(\mathbf{x}_i \mid \theta)$$

$$\mu = \frac{1}{n_\omega}\sum_i \omega_i x \qquad \sigma = \frac{1}{n_\omega}\sum_i \omega_i (x-\mu)^2$$

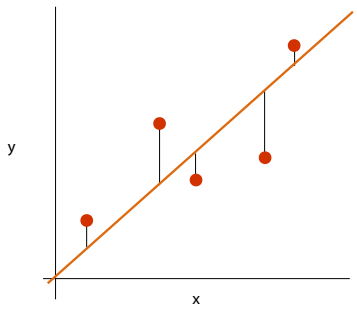
$$n_\omega = \sum_i \omega_i$$

Sometimes we have a **weighted** dataset. For instance, we might trust some measurements more than others, and so downweight the ones we distrust in order to get a more appropriate model. We'll see dataset weights crop up later in this lecture, and also, in the next lecture.

For such cases, we can easily define a weighted maximum likelihood objective. We minimize the log likelihood as before, but we assing each term (that is, each instance) a positive weight and maximize the weighted sum.

For the normal distributions, the weighted maximum likelihood estimators are what you'd expect: the same as for the unweighted case, except the sum becomes a weighted sum, and we divide by the sum of the weights, instead of by n.

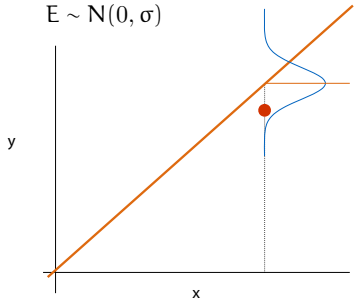
least squares regression



We first encountered the principle of least squeares, not in the context of descriptive statistics like the mean and the standard deviation, but in the context of regression.

You may ask whether this also leads to a normal distribution somewhere in the regression problem.

$$Y = X^T w + b + E$$
$$E \sim N(0, \sigma)$$



And indeed it does. When we fit a line using the least squares loss, we are implicitly assuming a model with noise. That noise, we are assuming to be normally distributed.

for a linear model, it works like this: we assume that our features were generated by some random process, which we don't know the details of. Somehow a random variable X was sampled. This sample was then transformed by a linear function, parametrized by (w, b), and to the result of that, a scalar E of normally distributed random noise was added (zero mean, with some variance).

We don't know the distribution that generated X and we don't know the variance on the noise distribution. As it turns out, we can estimate w and b without knowing these.

---

maximum likelihood for w and b

$$\begin{aligned}
&\arg\max_{w,b} p(Y \mid X, w, b) \\
&= \arg\max_{w,b} \ln \prod_i N(y_i \mid x_i^T w + b, \sigma) \\
&= \arg\max_{w,b} \sum_i \ln \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2\sigma^2}(x_i^T w + b - y_i)^2\right] \\
&= \arg\max_{w,b} -\sum_i \frac{1}{2\sigma^2}(x_i^T w + b - y_i)^2 \\
&= \arg\max_{w,b} -\frac{1}{2}\sum_i (x_i^T w + b - y_i)^2 = \arg\min_{w,b} \frac{1}{2}\sum_i (x_i^T w + b - y_i)^2
\end{aligned}$$

As we can see here, all elements from the normal distribution disappear except the square difference between the predicted output and the actual output, and the objective reduces to least squares.

maximum likelihood estimators for the MVN

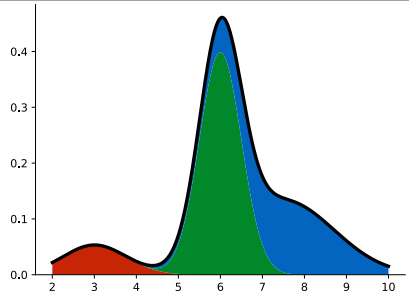
$$\mu = \frac{1}{n}\sum_{\mathbf{x}} \mathbf{x} \qquad \Sigma = \frac{1}{n}\sum_{\mathbf{x}} (\mathbf{x}-\mu)(\mathbf{x}-\mu)^T$$

$$\mu = \frac{1}{n_\omega}\sum_{i} \omega_i \mathbf{x}_i \qquad \Sigma = \frac{1}{n_\omega}\sum_{i} \omega_i (\mathbf{x}_i-\mu)(\mathbf{x}_i-\mu)^T$$

For the multivariate normal distribution, these are the maximim likelihood estimators.

The same things we said for the univariate case hold here. The estimator for the covariance requires a correction if you need unbiased estimates. And we can

---



$$p(x \mid \{w_i\},\{\mu_i\},\{\Sigma_i\}) = \sum_i w_i N(x \mid \mu_i, \Sigma_i)$$

Finally, let's look at the last of our modelsfrom the previous video: the Gaussian mixture model. What happens when we try to define the maximum likelihood objective for this model?

---

maximum likelihood for the GMM

$$\arg\max_{\{w_i\},\{\mu_i\},\{\Sigma_i\}} \sum_{\mathbf{x}} \ln \sum_i w_i N(\mathbf{x} \mid \mu_i, \Sigma_i)$$

Here we face a problem: there's a sum inside a logarithm. We can't work the sum out of the logarithm, which means we won't get a nice formulation of the derivative. We can do it anyway, and solve by gradient descent, we can even use backpropagation, so we only have to work out local derivatives, but what we'll never get,is a functional form for the derivative that we can set equal to zero and solve analytically.

After the break we'll discuss the **EM algorithm**, which does't give us an analytical solution, but it does allow us to use the tricks we've seen in this video, to help us fit a model.

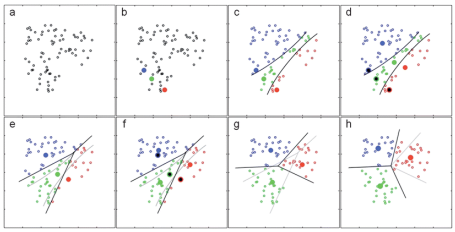
Density Estimation
Part 3: Expectation-maximization

Machine Learning
mlvu.github.io
Vrije Universiteit Amsterdam

alternating optimization

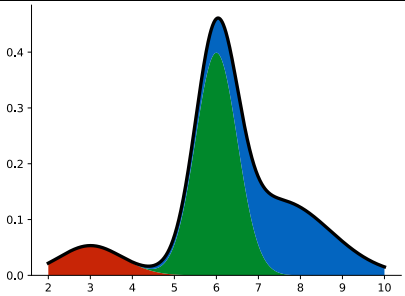
If your problem has two unknowns: fix one and solve for the other, then fix the other and solve for the first. Repeat.

k-means



We've seen one example of alternating optimization already, in the first lecture: the k-Means algorithm. Here, the two unknowns are where the centers of our clusters are, and which cluster each point belongs to. If we knew which cluster each point belonged to, it would be easy to work out the centers of each cluster. If we knew the cluster centers, it would be easy to work out which cluster each point belongs to.

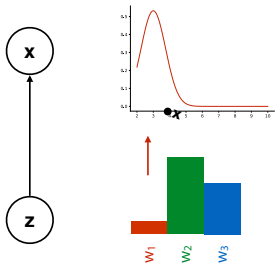
Since we know neither, we set one of them (the cluster centers) to an arbitrary value, and then assign the points to the obvious clusters. Then we fix the cluster memberships and recompute the cluster centers. If we repeat this process, we end up converging to a good solution.



$$p(x \mid \{w_i\}, \{\mu_i\}, \{\Sigma_i\}) = \sum_i w_i N(x \mid \mu_i, \Sigma_i)$$

With this idea in the back of our mind, we can return to the GMM. To fit this to data we can use what is probably the most famous alternating optimization method that exists: the **expectation maximization** algorithm.

hidden variable model



The GMM is an example of a **hidden variable model**: the data is produced by picking a component, and then sampling a point from the component.

We'll indicate which component we've picked by a variable z. This is a discrete variable, which for a three-component model can take the values 1, 2 or 3.

The problem is that when we see the data, we don't know z. All we see is the sampled data, but not which component it came from. For this reason we call z a **hidden variable**.

"completing" the data

Can we just marginalise **z** out?

We'd have to sum over all possible assignments of components to instances.

For two components, and just 30 points, this sum has a billion terms!

---

two unknowns

1. Which component generated each $\mathbf{x}_i$ in our data: $\{z_i\}$
2. What the parameters of the components are: $\{w_k\}, \{\mu_k\}, \{\Sigma_k\}$

Instead, we'll apply the philosophy of alternating optimization. First we state our two unknowns.

Clearly, if we knew which component generated each point, we could easily estimate the parameters. Just partition the data by component, and use the maximum likelihood estimators on each component. We'll see later what the MLE is for the component weights.

The other way around seems reasonable as well. Given the components, and their relative weight, it shouldn't be too tricky to work out how likely each component is to be responsible for any given instance.

EM: key insight

We can't optimise for θ and **z** together, but:

- Given some θ, we can compute p(**z** | **x**)
- Given **z**, we can optimise θ

EM (intuitive)

initialise components randomly

loop:

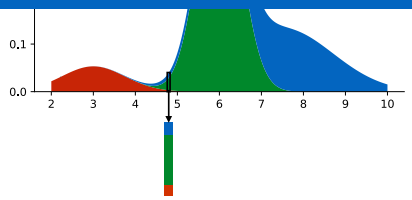
- expectation: assign soft *responsibilities* to each point
- maximization: fit the components to the data, weighted by responsibility.

52

The EM algorithm (for GMMs) expands on k-means by replacing the clusters with Gaussians, and by allowing points to "belong" to each Gaussian "to some degree". In other words, each Gaussian takes a certain *responsibility* for each point.

---

responsibilities (given the model parameters)



$$r_x^2 = \frac{N(x \mid \mu_2, \sigma_2)w_2}{N(x \mid \mu_1, \sigma_1)w_1 + N(x \mid \mu_2, \sigma_2)w_2 + N(x \mid \mu_3, \sigma_3)w_3}$$

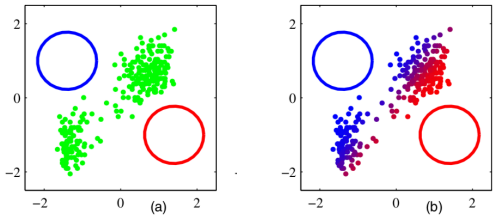
53

Here is how we define the **responsibility** taken for some point x by a particular component when the parameters of the components are given. We simply look at the three weighted densities for point x. The sum of these defines the total density for x under the GMM. The proportion of this sum that component 2 claims, is the responsibility that we assign to component 2 for producing x.

If you allow subjective probability, this is just Bayes' rule in action, the probability of component 2 being responsible given that we've observed x. If you want a purely frequentist interpretation of the EM algorithm, you have to be strict in calling these responsibilities and not probabilities. We cannot express a probability over which component generated x, since it is a hidden true value, which is not subject to chance.

For now, we'll just take this as a pretty intuitive way to work out responsibility, and see what it gets us. In the next video, we'll see a more rigorous derivation that even a frequentist can get behind.

In this case, the green component takes most of the responsibility for point x.



source: Machine Learning and Pattern Recognition, Christopher Bishop.

54

We can now take the first step in our EM algorithm. Here it is in two dimensions. We have some data and we will try to fit a two-component GMM to it. The number of components is a hyperparameter that we have to choose ourselves, based on intuition or basic hyperparameter optimization methods.

We start with two arbitrary components. Given these components, we then assign responsibilities to each of our points. for points that are mostly blue, the blue component claims the most responsibility and for points that are mostly red, the red component claims the most responsibility. For the purple points each component claims some responsibility.

model parameters (given the responsibilities)

$$\mu_i = \frac{1}{n_i} \sum_x r_x^i \mathbf{x} \qquad n_i = \sum_x r_x^i$$

$$\Sigma_i = \frac{1}{n_i} \sum_x r_x^i (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T$$

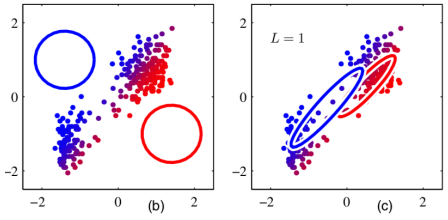
$$w_i = \frac{n_i}{n}$$

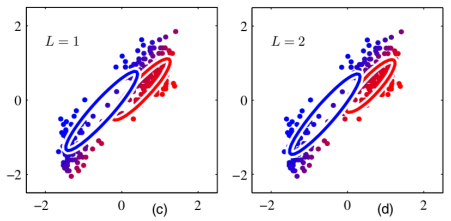
For each component i, we now discard the parameters mu and sigma, and recompute them to fit the subset of the data that the component has taken responsibility for.

Note that unlike the k-means algorithm, we never strictly partition the data. Every point belongs to the component *to some extent*. We do however get a weighting of the dataset which means that some instances should be much more important in where we put the mean and the variance than others. Here, we can use our weighted MLEs that we defined in the previous video.

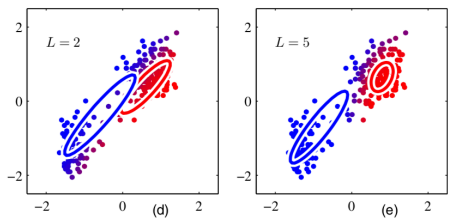
Our model isn't just the parameters of the components, we also need to work out the component *weights*. For now, we'll appeal to intuition, and say that it seems pretty logical to use the total amount of responsibility claimed by the component over the whole data. In the next video, we'll be a bit more rigorous.

With this, we have the two steps of our alternating optimization worked out: given components, we can assign responsibilities, and given responsibilities, we can fit components.

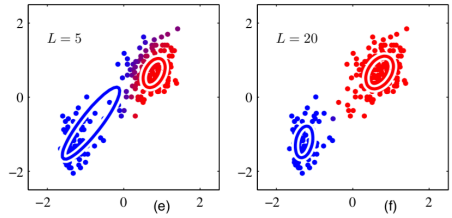



L = 1
L = 2
(c)
(d)

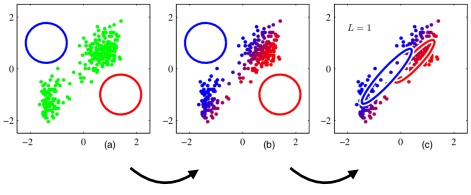
57


L = 2
L = 5
(d)
(e)

58


L = 5
L = 20
(e)
(f)

59


L = 1
(a)
(b)
(c)
Expectation step
Maximization step

60

What's the point?

61

clustering, density estimation

Fraud detection, targeting treatment, customer segmentation

- All unsupervised methods. Mostly useful for *exploratory* analysis.

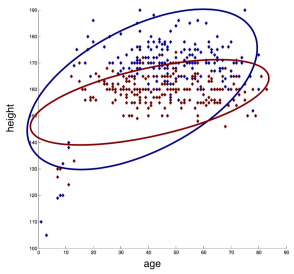
62

classification

Build a Bayes classifier. Fit an GMM model to the points for each class, and classify by maximum

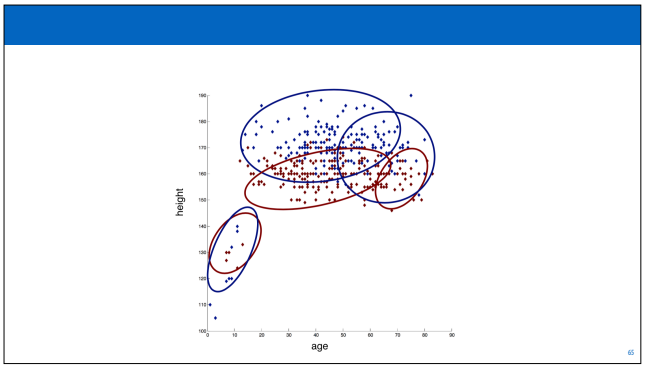
- This is a non-naive Bayes classifier. The features are not independent at all.

63



64

Here's what that looks like with a single Gaussian per class

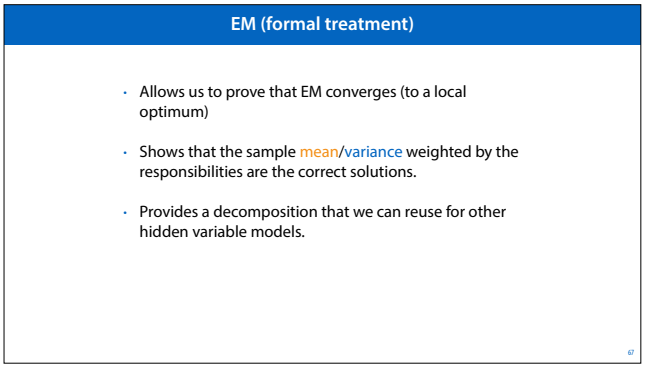


With multiple Gaussians, we can fit the shape of the data more naturally.

Density Estimation
Part 4: A formal analysis of EM

Machine Learning
mlvu.github.io
Vrije Universiteit Amsterdam

In the last video, we explained how EM works to fit a GMM model. We took a pretty informal approach, and appealed to intuition for most of the decisions we made. This is most helpful to get a comfortable understanding of the algorithm, but as it happens, we can derive all of these steps formally, as approximations to the maximum likelihood estimator fo the GMM model.



What do we get out of this, since we already have the EM algorithm and is seems to work pretty well?

First, we can prove that EM converges to a local optimum.

Second, we can derive the responsibilities and weighted mean and variance as the correct solutions. In the last video, if we had come up with five other ways of doing it that also seem pretty intuitive, we would have to implement and test all of them to see which worked best. With a little more theory we can save ourselves the experimentation. This is a good principle to remember: the better your grasp of theory, the smaller your search space.

And finally, the decomposition we will use here will help us in other settings as well, starting next week we we apply the principle to deep neural networks.

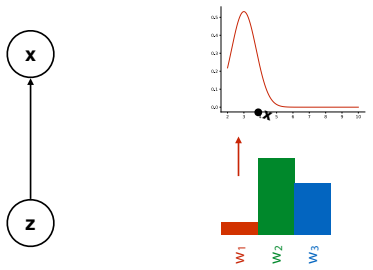
$$\arg\max_\theta \sum_x \ln p(x \mid \theta) = \arg\max_\theta \sum_x \ln \sum_z p(x, z \mid \theta)$$

$$= \arg\max_\theta \sum_x \ln \sum_z p(x \mid z, \theta) p(z \mid \theta)$$

$$= \arg\max_{\{\mu_k\},\{\Sigma_k\},\{w_k\}} \sum_x \ln \sum_k N(x \mid \mu_k, \Sigma_k) w_k$$

Let's start by going back to the objective that we actually want to solve: the maximum likelihood objective

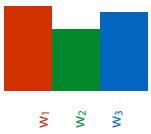
hidden variable model



The problem, as we saw before, is the hidden, or latent variable. The fact that we have to estimate both the parameters and the responsibilities of each component for each point together is

assume a distribution on z

$$q(z \mid x)$$
$$p(z \mid x, \theta)$$



Our first step is to assume some arbitrary function which gives us a distribution on z for x. This could be a very accurate distribution or a terrible one. We'll work out some properties first that hold for any q.

Since in our specific example, z can take one of k values, you should think of q(z|**x**) as a categorical distribution over the components in our model. For a particular x, it tells us which components are most likely. This is the same function as the responsibilities we defined earlier, and indeed we will see that q will become the responsibilities later, but right now, we are making no assumptions about how q is computed: it could be a completely arbitrary and incorrect function.

We can think of q(z|x) as an approximation to p(z|x, theta), the conditional distribution on the latent space.

Why do we introduce q, when we can actually compute p(z|x, theta)? Because q can be any function, which means it's not tied to a particular value of theta. q is not a function of theta, which means that in our optimization, it functions as a constant. As we shall see, this can help us a great deal in our analysis.

a very useful decomposition

$$\ln p(\mathbf{x} \mid \theta) = L(q, \theta) + KL(q, p)$$

with :

$p = p(\mathbf{z} \mid \mathbf{x}, \theta)$

$q(\mathbf{z} \mid \mathbf{x})$ any approximation to $p(\mathbf{z} \mid \mathbf{x})$

$KL(q, p)$ Kullback-Leibler divergence

$$L(q, \theta) = \mathbb{E}_q \ln \frac{p(\mathbf{x}, \mathbf{z} \mid \theta)}{q(\mathbf{z})}$$

71

Given some q, we can show that the likelihood p(x | theta) which we cannot easily optimize for, decomposes into the following two terms.

The KL divergence, as we saw in lecture 5, is a distance between two probability distributions. It tells us how good of an approximation q is for the distribution p we just compared it to. The worse the approximation, the greater the KL divergence.

L is just a relatively arbitrary function. The isn't much meaning that can be divined from it's definition, but we can prove that when we rewrite the log-likelihood of x into the KL divergence between p and q, L is what is "left over". L plus the KL divergence makes the log likelihood. This means that when q is perfect approximation, and the KL divergence becomes zero, L is equal to the likelihood. The worse the approximation, the lower L is, since the KL divergence is always zero or greater.

In our current case, z is just a scalar, but we'll treat it as a (boldface) vector to highlight that in general, this works for any kind of latent variable. We'll need that when we reuse this decomposition in later lectures.

$$\ln p(\mathbf{x} \mid \theta) = L(q, \theta) + KL(q, p)$$

$$\sum_{\mathbf{z}} q(\mathbf{z} \mid \mathbf{x}) \ln \frac{p(\mathbf{x}, \mathbf{z} \mid \theta)}{q(\mathbf{z} \mid \mathbf{x})} - \sum_{\mathbf{z}} q(\mathbf{z} \mid \mathbf{x}) \ln \frac{p(\mathbf{z} \mid \mathbf{x}, \theta)}{q(\mathbf{z} \mid \mathbf{x})}$$

$$= \mathbb{E}_q \ln \frac{p(\mathbf{x}, \mathbf{z} \mid \theta)}{q(\mathbf{z} \mid \mathbf{x})} - \mathbb{E}_q \ln \frac{p(\mathbf{z} \mid \mathbf{x}, \theta)}{q(\mathbf{z} \mid \mathbf{x})}$$

$$= \mathbb{E}_q \ln p(\mathbf{x}, \mathbf{z} \mid \theta) - \mathbb{E}_q \ln p(\mathbf{z} \mid \mathbf{x}, \theta)$$

$$= \mathbb{E}_q \ln \frac{p(\mathbf{x}, \mathbf{z} \mid \theta)}{p(\mathbf{z} \mid \mathbf{x}, \theta)} = \mathbb{E}_q \ln \frac{p(\mathbf{z} \mid \mathbf{x}, \theta) p(\mathbf{x} \mid \theta)}{p(\mathbf{z} \mid \mathbf{x}, \theta)}$$

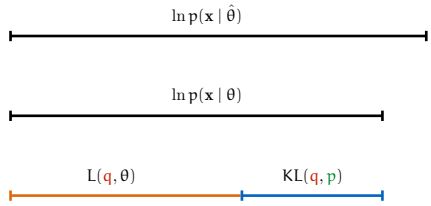
$$= \mathbb{E}_q \ln p(\mathbf{x} \mid \theta) = \ln p(\mathbf{x} \mid \theta)$$

72

Here is the proof that this decomposition holds. It's easiest to work backwards. We fill in our statement of the L and KL terms, and rewrite to show that they're equivalent to the log likelihood.

---

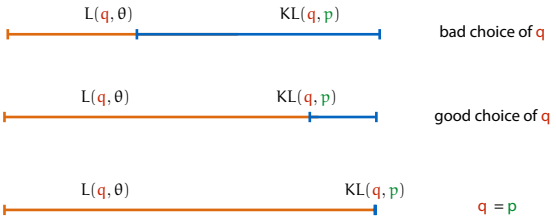
for *any* q



73

This is the picture that allthis rewriting buys us. We have the probability of our data under the optimal model (top line) and the probability of our data under our current model (middle line). And for any q, whether it's any good or not, the latter is composed of two terms.

$$\ln p(\mathbf{x} \mid \theta) = L(q, \theta) + KL(q, p)$$



74

Note that this is just a way of writing down the probability density of our data given the parameters (with the hidden variable z marginalized out). The sum of these two terms is always the same. The closer p is to q, the smaller the KL term gets.

In short L is a lower bound in the quantity that we're interested in. The KL term tells us how good of a lower bound this is.

EM

E: Choose q to minimize the KL term. Keep θ fixed.

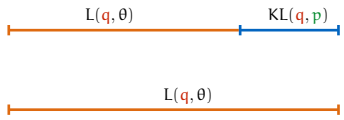
M: Choose θ to maximize the L term. Keep q fixed.

75

With this decomposition, it turns out that we can state the EM algorithm very simply.

EM

E: Choose q so that KL(q, p) = 0. Keep θ fixed.

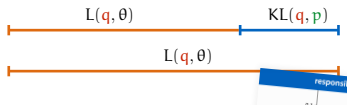


76

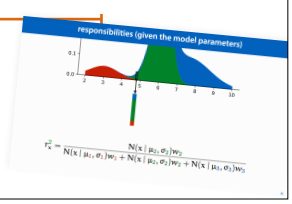
Here's what that looks like. Note that our objective is to maximize the sum of these two bars, so it may at first seem counter-intuitive to minimize something. However, the decomposition holds for any q, so if we re-select q, we don't change the total length. We just change the propotion that the q term takes.

E step

E: Choose q so that KL(q, p) = 0. Keep θ fixed.



q(z|p) = p(z|x, θ)



In our specific setting, the expectation step is easy to work out. The KL divergence is minimal when q is a perfect approximation to p. Since we keep theta as a constant, we can just work out the conditional probabilities on z given the parameters theta

The result is simply the responsibilites we already worked out in slide 55.

M: Choose θ to maximise L. keep q fixed.

$L(q, \theta)$

$L(q, \theta)$ $KL(q, p)$

convergence!

78

In the M step, we change the parameters theta. This means our q function is no longer a perfect approximation to the conditional probability on z under p, so the KL divergence is no longer zero. However, since we chose theta explicitly to increase L, we know that L increases, and so does the sum of L and the KL divergence.

Note that if we do this, we can be sure that the algorithm converges. The E step keeps the total length of the bar the same, and the M step increases it. In other words, the algorithm can only move uphill in the surface of log likelihood. There's no guarantee that it'll find the global optimum, but we know that it'll converge

---

M step

M: Choose θ to maximise L. keep q fixed.

$$\arg\max_{\theta} \sum_{\mathbf{x},\mathbf{z}} q(\mathbf{z} \mid \mathbf{x}) \ln \frac{p(\mathbf{x}, \mathbf{z} \mid \theta)}{q(\mathbf{z} \mid \mathbf{x})}$$

$$= \arg\max_{\theta} \sum_{\mathbf{x},\mathbf{z}} q(\mathbf{z} \mid \mathbf{x}) \ln p(\mathbf{x} \mid \mathbf{z}, \theta)\ p(\mathbf{z} \mid \theta) - \sum_{\mathbf{x},\mathbf{z}} q(\mathbf{z} \mid \mathbf{x}) \ln q(\mathbf{z} \mid \mathbf{x})$$

responsibilities likelihood

79

Let's look at what this step looks like for the one-dimensional GMM. If we take the division out of the logarihtm, it becomes a term that does not contain theta, so we can remove it from our objective.

The remainder is just a likelihood weighted by the responsibilities we've just computed.

Note that the sum is now outside the logarithm. That means we can work out an optimal solution for the model parameters given the current q.

---

mean covariance

$$\arg\max_{\theta} \sum_{\mathbf{z},\mathbf{x}} q(\mathbf{z})\ \ln p(\mathbf{x} \mid \mathbf{z}, \theta)\ p(\mathbf{z} \mid \theta)$$

$$= \arg\max_{\{\mu_k, \Sigma_k, w_k\}} \sum_{\mathbf{k},\mathbf{x}} q(Z = k \mid \mathbf{x})\ \ln N(x \mid \mu_k, \Sigma_k) w_k$$

$$= \arg\max_{\{\mu_k, \Sigma_k, w_k\}} \sum_{\mathbf{k},\mathbf{x}} r_{\mathbf{x}}^{k}\ \ln N(x \mid \mu_k, \Sigma_k) w_k$$

$$\arg\max_{\mu_2} \sum_{\mathbf{k},\mathbf{x}} r_{\mathbf{x}}^{k}\ \ln N(\mathbf{x} \mid \mu_k, \Sigma_k) w_k$$

$$= \arg\max_{\mu_2} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2}\ \ln N(x \mid \mu_2, \Sigma_2) w_2$$

$$= \arg\max_{\mu_2} -\frac{1}{2} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2} \|\mathbf{x} - \mu_2\|^2 + r_{\mathbf{x}}^{2} \ln w_2$$

$$\dots$$

$$= \frac{1}{n_2} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2} \mathbf{x}$$

$$\arg\max_{\Sigma_2} \sum_{\mathbf{k},\mathbf{x}} r_{\mathbf{x}}^{k}\ \ln N(\mathbf{x} \mid \mu_k, \Sigma_k) w_k$$

$$= \arg\max_{\Sigma_2} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2}\ \ln N(x \mid \mu_2, \Sigma_2) w_2$$

$$\dots$$

$$= \frac{1}{n_2} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2} (\mathbf{x} - \mu_2)(\mathbf{x} - \mu_2)^{\mathsf{T}}$$

80

If we take this criterion, and work out the maximum likelihood, we find that for the mean and covariance we we a weighted version of the maximum likelihood objective for the normal distribution, woking them out gives us the weighted versions of the maximum likelihood estimators of the mean and the covariance.

---

weights

$$\arg\max_{w_2} \sum_{\mathbf{k},\mathbf{x}} r_{\mathbf{x}}^{k}\ \ln N(\mathbf{x} \mid \mu_k, \Sigma_k) w_k \qquad \text{such that } \sum_{\mathbf{k}} w_k = 1$$

$$\arg\max_{w_2} \sum_{\mathbf{k},\mathbf{x}} r_{\mathbf{x}}^{k}\ \ln N(\mathbf{x} \mid \mu_k, \Sigma_k) + r_{\mathbf{x}}^{k} \ln w_k \qquad \text{such that } \sum_{\mathbf{k}} w_k = 1$$

$$\arg\max_{w_2} \sum_{\mathbf{k}} \ln w_k \sum_{\mathbf{x}} r_{\mathbf{x}}^{k} \qquad \text{such that } \sum_{\mathbf{k}} w_k = 1$$

$$L(w_1, w_2, \dots, w_k, \alpha) = \sum_{\mathbf{k}} \ln w_k \sum_{\mathbf{x}} r_{\mathbf{x}}^{k} - \alpha \left(\sum_{\mathbf{k}} w_k - 1 \right)$$

$$\frac{\partial L}{\partial w_2} = \frac{1}{w_2} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2} - \alpha \qquad \frac{\partial L}{\partial \alpha} = -\sum_{\mathbf{k}} w_k + 1$$

$$w_2 = \frac{1}{\alpha} \sum_{\mathbf{x}} r_{\mathbf{x}}^{2} \qquad \sum_{\mathbf{k}} w_k = 1$$

$$\sum_{\mathbf{k}} \frac{1}{\alpha} \sum_{\mathbf{x}} r_{\mathbf{x}}^{k} = 1$$

$$w_2 = \frac{\sum_{\mathbf{x}} r_{\mathbf{x}}^{2}}{\sum_{\mathbf{x},\mathbf{k}} r_{\mathbf{x}}^{k}} \qquad \sum_{\mathbf{x},\mathbf{k}} r_{\mathbf{x}}^{k} = \alpha$$

81

The weights should sum to one, so that part of our optimization is actually a constrained optimization problem. This gives us a good opportunity to practice our Lagrange multipliers.

We define an L function that includes the constraints, take it's derivative wrt to all its parameters (including the multiplier alpha), and we set them equal to zero. The result for the weights is an expression including alpha, and the result for the lagrange multiplier recovers the constraint, as it always does. Filling the former into the latter shows us that alpha expresses the total sum of responsibility weights over all components and instances.

This means that the optimal weight for component 2 is the amount of responsibility assigned to component 2, as a proportion of the total.

M step

M: Choose θ to maximise L. keep q fixed.

$$\arg\max_{\theta} \sum_{z} q(\mathbf{z}) \ln p(\mathbf{x} \mid \mathbf{z}, \theta) p(\mathbf{z} \mid \theta)$$

$$n_i = \sum_x r_x^i$$

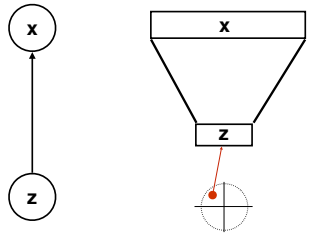
$$\mu_i = \frac{1}{n_i} \sum_x r_x^i \mathbf{x}$$

$$\Sigma_i = \frac{1}{n_i} \sum_x r_x^i (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T$$

$$w_i = \frac{n_i}{n}$$

82

Next lecture: hidden variable models with neural networks.



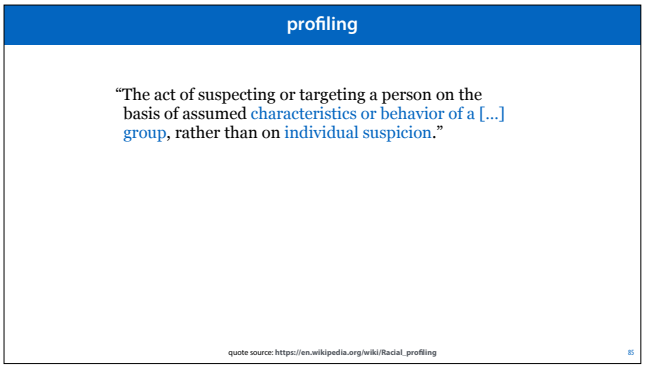
83

Density Estimation

Part 5: Social Impact 3

Machine Learning
mlvu.github.io
Vrije Universiteit Amsterdam

This week and the last, we've discussed a lot of probability theory. With these tools in hand, we can go back to our discussion on social impact, and try to make it more precise.



Specifically , in this video, we'll look at the problem of **profiling**.

When we suspect people of a crime or target them for investigation, based on their membership of a group rather than based on their individual actions, that's called **profiling**.

Probably the most common form is **racial profiling**; which is when the group in question is an ethnic or racial group. Examples include black people being more likely to be stopped by police, or Arabic people being more likely to be checked at airports.

Other forms of profiling, such as gender or sexual orientation profiling also exist in various contexts.

We saw an example of this in the first social impact video: a prediction system (essentially using machine learning) which predicted the risk of people in prison re-offending when let out. This system, built by a company called Northpointe, showed a strong racial bias.

As we saw then, it's not enough to just remove race as a feature. So long as race or ethnicity can be predicted from the features you do use, your model may be inferring from race.

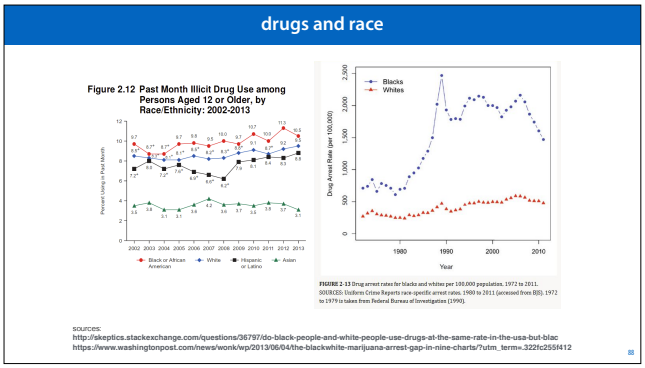
Profiling doesn;t just happen in automated systems. And lest you think this is a typically American problem, let's look a little closer to home.

A few years ago, a Dutch hip-hop artist called Typhoon was stopped by the police. The police admitted that the combination of his skin colour and the fact that he drove an expensive car played a part in the choice to stop him. This caused a small stir in the Dutch media and a nationwide discussion about racial profiling.

The main argument usually heard is "if it works, then it is worth it." That is, in some cases, we should accept a certain amount of racism in our criminal procedures, if it is in some way successful.

This statements hides a lot complexity: we're assuming that such practices are successful, and we're not defining what being successful means in this context. Our responsibility, as academics, is to unpack such statements, and to make it more precise what is actually being said. Let's see if we can do that here.

We'll focus on the supposed pros and cons of profiling and on what it means for a profiling method to be successful, regardless of whether it's an algorithm or a human doing the profiling.

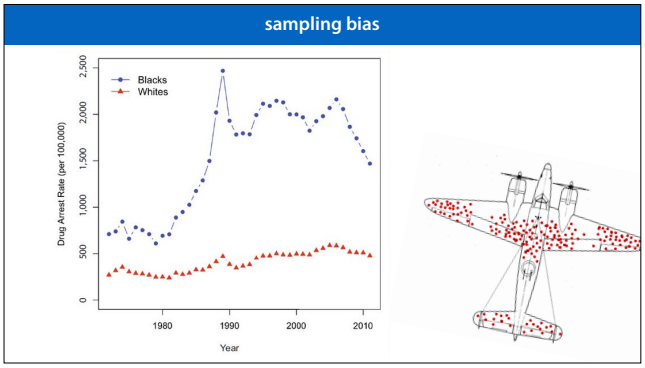


Since this is a sensitive subject, we'll try to make our case as precisely as possible, and focus on a specific instance, where we have all the necessary data available: illicit drug use in the US. The US has a system in place to record race and ethnicity in crime data. The categorization may be crude, but it'll suffice for our purposes.

From these graphs, we see on the left that black people engage in illicit drug use more than people of other ethnicities, and that they are also arrested for it more than people of other ethnicities. However, the rate of use is only marginally higher than that of white people, whereas the arrest rate can be as much as five times as high as that for white people,

This points to one potential problem: racial profiling may very easily lead to disproportionate effects like those seen on the right. Even if there's difference in the proportion with which black people and white people commit a particular crime, it's very difficult to ensure that the profiling somehow honors that proportion. But we shouldn't make the implicit assumption that that's the only problem. If the proportions of the two graphs matched, would profiling then be justified? Is the problem with profiling that that we're not doing it carefully enough, or is the problem that we're doing it at all?

We'll look at some of the most common mistakes made in reasoning about profiling, one by one.



One problem with an automated system like that of Northpointe is that there is a strong risk of data not being sampled uniformly. If we start out with the arrest rates that we see on the right, then a system that predicts illicit drug use will see a lot more black drug users than white ones. Given such a data distribution, it's not suprising that the system learns to associate being black with a higher rate of drug use.

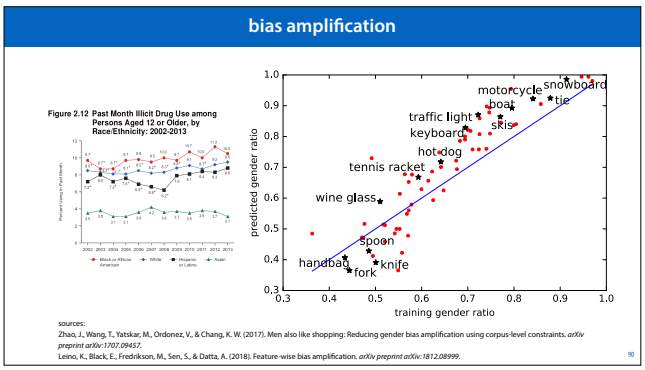
This is not because of any fundamental link between race and drug use, but purely because the data is not representative of the population. We have a **sampling bias**.

It's a bit like the example of the damaged planes in

WWII we saw at the start of the fourth lecture: if we assume a uniform distribution in the data, we will conclude the wrong thing. In that case we weren't seeing the planes that didn't come back. Here, we aren't seeing the white people that didn't get arrested.

Note that it's not just algrithms that suffer from this problem. For instance, if we leave individual police officers to decide when to stop and search somebody, they will likely rely on their own experience, and the experience of a police officer is not uniform. There are many factors affecting human decision making, but one is that if they already arrest far more black than white people, they are extremely likely to end up with the same bias an algorithm would end up with.

So let's imagine that this problem is somehow solved, ande we get a perfectly representative dataset, with no sampling bias. Are we *then* justified in racial profiling?



You'd be forgiven for thinking that if a bias is present in the data, that the model simply reproduces that bias. In that case, given a dataset without sampling bias, we would start with the minor discrepancies on the left, and simply reproduce those. Our model would be biased, but we could make the case that it is at least reproducing biases present in society.

However, it's a peculiar property of machine learning models that they may actually *amplify* biases present in the data. That means that even if we start with data seen on the left, we may still end up with a predictor that disproportionately predicts drug use for black people.

An example of this effect is seen on the right. For an image labeling tasks, the authors measured gender ratios in the training set, for subsets of particular nouns. For instances, for images containing both a wine glass and a person, we see that the probability of seeing a male or female person in the data is about 50/50, but in the predictions over a validation set, the ratio shifts to 60/40.

It's not entirely clear where this effect comes from. The second paper quoted shows that it's related to our choice of inductive bias, so it's a deep problem, that gets to the heart of the problem of induction. Even the Bayes' optimal classifier can suffer from this problem. For our current purposes it's enough to remember, that **even if our input has biases that are representative,**

there's no guarantee that our output will.

It appears that this is a problem that may be impossible to solve. But let's imagine, for the sake of arguments, that we somehow manage it. What if we get a perfectly representative dataset with no sampling bias, *and* we somehow ensure that our model doesn't amplify bias. Can we then do racial profiling?

Much of racial profiling falls into the trap of the **prosecutor's fallacy**. In this case the probability that a person uses illicit drugs, given that they're black is very slightly higher than the probability that they do so given that they are white, so the police feel that they are justified in using ethnicity as a feature for predicting drug use (it "works").

However, the probability that a person uses illicit drugs given that they are black is still very much *lower* than the probability of not using illicit drugs given that they they are black. This probability is never considered.

As we see in the previous slide the rates are around p(drugs|black) = 0.09 vs. p(~drugs|black) = 0.91. If the police blindly stop only black people, they are disadvantaging over 90% of the people they stop.

To help you understand, consider a more extreme example of the prosecutor's fallacy. Let's imagine that you're trying to find professional basketball players. The probability that somebody is tall given that they play professional basketball, p(tall| basketball) is almost precisely 1. Thus, if you're looking for professional basketball players, you are justified in only asking tall people. However, the probability of somebody playing professional basketball given that they're tall, is still extremely low. That means that if you go around asking tall people whether they are profesional basketball players, you'll end bothering a lot of people before you find your basketball player, and probably annoying quite a few of them.

What if

the data is a fair representation of the population

and

the model doesn't amplify bias

and

we've correctly used Bayes' rule?

92

So, have we now covered all our bases? We get a dataset that is a fair representation, our model doesn't amplify biases, and we correctly use Bayes' rule.

Can we then use the model to decide whether or not to stop black people in the street?

The answer is still no.

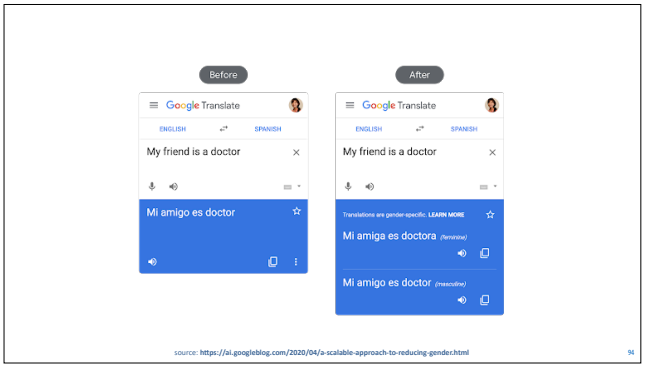
At this point, we may be certain that our **predictions** are accurate, and we have accurately estimated the probability accurately that a particular black person uses drugs illicitly.

However, the fact that those predictions are accurate tells us nothing about whether the action of then stopping the person will be **effective**, **justified**, or **fair**. That all depends on what we are trying to achieve, and what we consider a fair and just use of police power. The accuracy of our predictions cannot help us guarantee any of this.

Just because your **predictions** are accurate, doesn't make your **actions** sound.

This is an extremely important distinction in the responsible use of AI. There is a very fundamental difference between **making a prediction** and **taking an action** based on that prediction.

We can hammer away at our predictions until there's nothing left to improve about them, but none of that will tell us anything about whether taking a particular action is justified. How good a prediction is and how good an action is are two entirely different questions, answered in completely different ways.



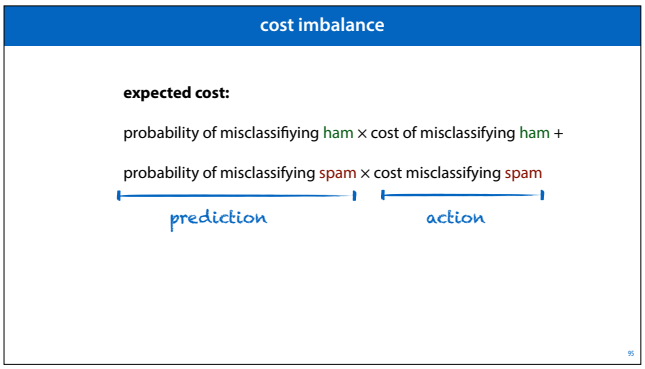
source: https://ai.googleblog.com/2020/04/a-scalable-approach-to-reducing-gender.html

94

Recall the Google translate example from the first lecture. Given a gender neutral sentence in English, we may get a prediction saying that with probability 70% the word doctor should be translated as male in Spanish and with probability 30% it should be translated as female. There are almost certainly biases in the data sampling, and there is likely to be some bias amplification in the model, but in this case we can at least define what it would mean for this probability to be accurate. For this sentence, there are true probabilities, whether frequentist or Bayesian, for how the sentence should be translated. And we can imagine an ideal model that gets those probabilities absolutely right.

However, that tells us nothing about what we should *do* with those probabilities. Getting a 70/30 probability doesn't mean we are justified in going for the highest probability, or in sampling by the probabilities the model suggests. Both of those options have positive consequences, such as a user getting an accurate translation, and negative consequences, such as a user getting an accurate translation and the system amplifying gender biases.

In this case, the best solution turned out to be a clever interface design choice, rather than blindly sticking with a single output.



This is related to the question of cost imbalance. We may get good probabilities on whether an email is ham or spam, but until we know the cost of misclassification we don't know which action to prefer (deleting the email or putting it in the inbox). The expected cost depends on how accurate our predictions are, but also on which actions we decide to connect to each of the predictions. This is an important point: cost imbalance is not a property of a classifier in isolation: it's a property of a classifier, inside a larger system that takes actions. The cost imbalance for a system that deletes spam is very different from the cost imbalance in a system that moves spam to a junk folder.

Here, we should always be on the lookout for creative solutions in how we use our predictions. Moving spam to a junk folder instead of deleting it, showing users multiple translations instead of just one, and so on. The best ways of minimizing cost don't come from improving the model performance, but from rethinking the system around it.

In questions of social impact, the cost of misclassification is usually extremely hard to quantify. If a hundred stop-and-searches lead to two cases of contraband found, how do we weigh the benefit of the contraband taken off the streets against the 98 stop-and-searches of innocent individuals. If the stop-and-search is done in a biased way, with all black people being searched at least once in their lifetime and most white people never being searched, then the stop-and-search policy can easily have a very damaging effect on how black people are view in society.

It's very easy, and very dangerous to think that we can easily quantify the cost of mistakes for systems like these.

correlation and causation

A and B are correlated: I can predict A from B (and vice versa)

A causes B: changing A causes a change in B (but *not* vice versa.

Correlation does not imply causation.

No correlation without causation.

96

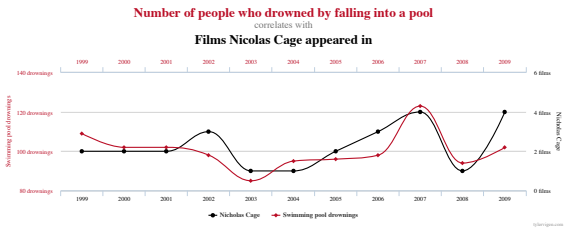
A large part of choosing the right action to take based on a prediction, is separating **correlation** and **causation**. A lot of social issues, in AI and elsewhere, stem from confusions over correlation and causation, so let's take a cerful look at these two concepts.

Two observables, for instance, being black and using illicit drugs are correlated, if knowing the value of one can be used to predict the value of the other. It doesn't have to be a good prediction, it just has to be better than it would be if we didn't know the value of the first.

This doesn't mean that the first **causes** the second. I can from the smoke in my kitchen that my toast has burned, and if somebody tells me that my toaster has been on for half an hour, I can guess that there's probably smoke in my kitchen. Only one of these causes the other. There are many technical definition of what constitutes causaility, but in general we say that A causes B if changing A causes a change in B. Turning off the toaster removes the smoke from my kitchen, but opening a window doesn't stop my toast burning.

---

spurious correlations

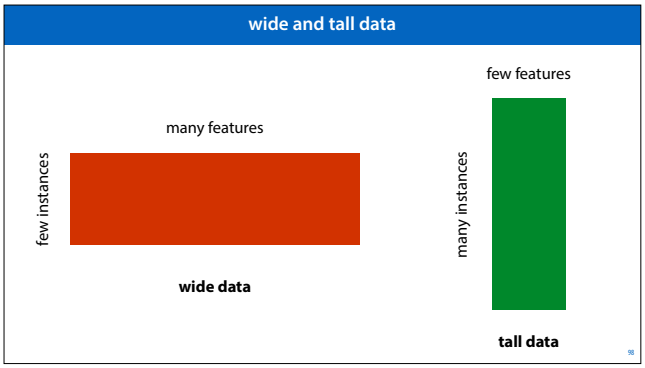


source: **https://www.tylervigen.com/spurious-correlations**

97

When talking correlation, the first thing we need to be on the lookout for is ***spurious* correlations**. According to this data here, if we know the number of films Nicolas Cage appeared in in a given year, we can predict how many people will die by drowning in swimming pools.

This is not because of any causal mechanism. Nicolas Cage is not driven by drowning deaths, and people do not decide to jump into their pools just because there are more Nicolas Cage movies (whatever you think of his recent career). It's a *spurious correlation*. It looks like a relation in the data, but because we have so few examples for each, it's possible to see such a relation by random chance (especially if you check many different potential relations).

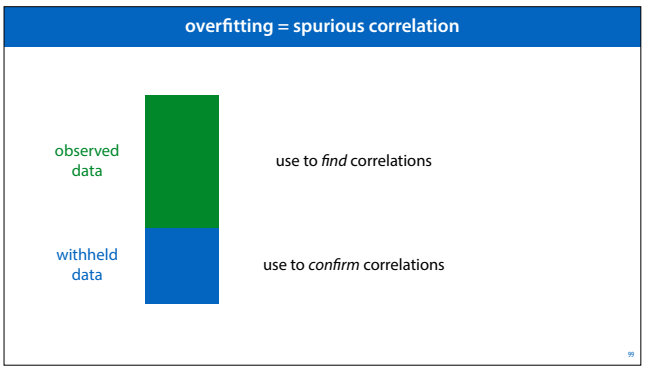


Gathering more data can hurt or help you here.

The more features you have, the more likely it is that one of them can be predicted from the other purely by chance, and you will observe a correlation when there isn't any. If we see the target label as another feature, this also tells us that using many features increases the probability of overfitting: observing good predictions on the training data without actually getting good performance. We can call this **wide data**.

Adding *instances* has the opposite effect. The more instances, the more sure we can be that observed correlations are true and not spurious. We can call this **tall data**.

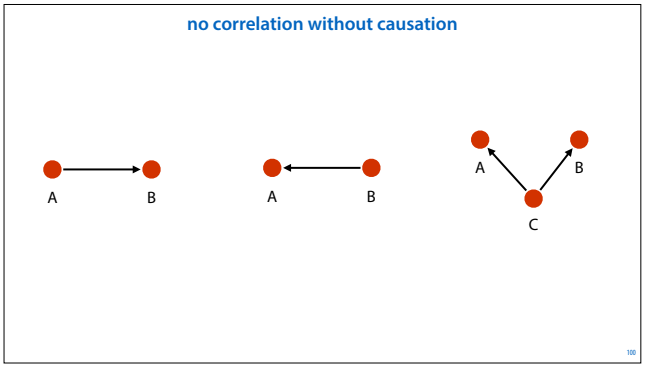
Thus, if we are conservative with our features, and liberal with our instances, we can be more confident that any observed correlations are correct. The litmus test is to state the correlations you think are true and then *to test them on new data*. In life sciences, this is done through replication studies, where more data is gathered and the stated hypothesis from an existing piece of research is evaluated be the exact same experiment.



In machine learning, we do this whenever we withhold a test set.

This is essentially a way of guarding against spurious correlations, or in other words, overfitting is just a spurious correlation. The definition of a spurious correlation is one that disappears when you gather more data, so if our correlation is spurious, it should not be present in the withheld data.

A good machine learning model finds only *true correlations* and no *spurious correlations*. How to make that distinction without access to the withheld data, is the problem of induction.



So if we rule out spurious correlations, what can we say that we have learned when we observe **a correlation**?

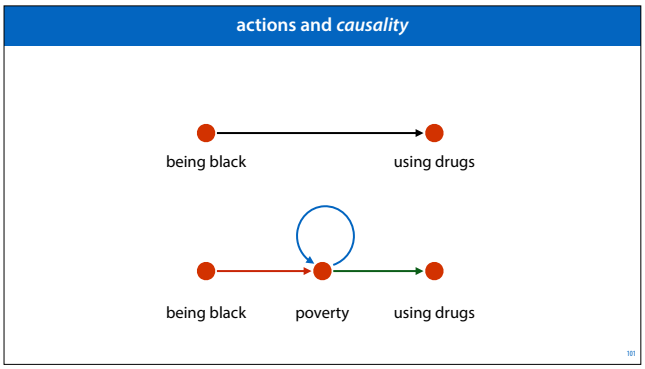
If I see you have a runny nose, I can guess you have a cold. That doesn't mean that having a runny nose causes colds. If I make the exam too difficult this year, it affects all grades, so somebody can predict from your failing grade that other students are also likely to have a failing grade. That doesn't mean that you caused your fellow student to fail. This is the cardinal rule of statistics: **correlation is not causation**. It is one that you've hopefully heard before.

There is another rule, that is just as important, and a

lot less famous. **No correlation without causation.** If we observe a correlation and we've proved that it isn't spurious, there must be a causation *somewhere*.

Simplifying things slightly, these are the ways a correlation can occur. If A and B are correlated then either A causes B, B causes A, or there is some other effect that causes both A and B (like me setting the difficulty of the exam). A cause like C is called a **confounder**.

It is important to note that C doesn't have to be a single simple cause. It could be a large network of many related causes. In fact, causal graphs like these are almost always simplifications of a more complex



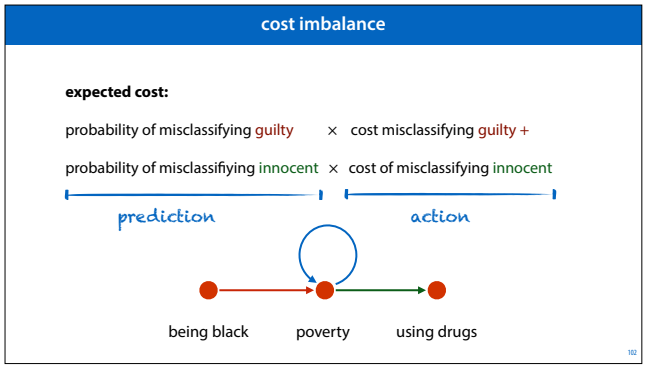
So let's return to our example of illicit drug use in America. We know that there's a small correlation between race and illicit drug use (even though there is a far greater discrepancy in arrests). What is the causal graph behind this correlation?

At the top we see what we can call the *racist interpretation*. That is, *racist* in the literal sense: seeing race as the fundamental cause of differences in behaviour. Put simply, this interpretation assumes a fundamental, biological difference in black people that makes them more susceptible to drug addiction. Few people hold such views explicitly these days, and there is no scientific evidence for it. But it's important to remember that this kind of thinking was much more common not too long ago.

At the bottom, is a more modern view, backed by a large amount of scientific evidence. Being black makes you more likely to be poor, due to explicit or implicit racism in society, and being poor makes you more likely to come into contact with illicit drugs and makes you less likely to be able to escape addiction.

There is a third effect, which I think is often overlooked: poverty begets poverty. The less money your parents have, the lower your own chances are to escape poverty. Having to live in poverty means living from paycheck to paycheck, never building up savings, never building up resilience to sudden hardship, and never being able to invest in the long term. This means that on average, you are more likely to increase your poverty than to decrease it.

The reason all this is relevant, is that for interventions to be effective, they must be aligned to the underlying causes. In the world above, racial profiling may actually be effective (although it could still be unjust). However, in the picture below, racial profiling actually increases pressure on black people, pushing them further into poverty. Even though the police *feel* like they're arresting more drug users, they are most likely strengthening the blue feedback loop (or one similar to it).



If we ignore data bias, and assume a perfect predictor, we still have to deal with the cost of misclassification.

Misclassifying a guilty person can feed into this blue feedback loop. In the best case, it leads to embarrassment and loss of time for the person being searched. But there can also be more serious negative consequences.

One subtle example is being found out for another crime than the one you were suspected of, due to the search. For instance, imagine that the if the predictor classifies for driving a stolen car, and during the stop, marijuana is found. This may at first seem like a win: the more crimes caught, the better. However, the result of doing this *based on profiling* is again that we are feeding into the blue feedback loop.

There is a certain level of crime that we, as society allow to pass undetected, because detecting it would have too many negative consequences. It would cost too much to detect more crime, or infringe too much on the lives of the innocent.

This is true for any society anywhere, although every society makes the tradeoff differently. However, if we stop people because they are predicted, through profiling, to be guilty crime X, and then arrest them for crime Y, then we end up setting this level differently for black people than for white people. Essentially, by introducing a profiling algorithm for car theft, we are lowering the probability that people get away with marijuana possession, and we are lowering it further for black people than for white people.

Causality plays a large role in setting the rules for what is and isn't *fair*. In law this is described as **differentiation**, justly treating people differently based on their attributes and **discrimination**, unjustly treating people differently based on their attributes.

For instance, if we are hiring an actor to appear in in an ad for shaving cream, we have a sound reason for preferring a male actor over a female actor, all other qualifications being the same. There is a clear, common-sense causal connection between the attribute of being male and being suitable for the role.

If we are hiring somebody to teach machine learning at a university, preferring a male candidate over a female

one, all else being equal, is generally considered wrong, and indeed illegal.

That is, differentiation is usually allowed, if and only if there is an unambiguous causal link between the sensitive attribute and job suitability.

So what if we:

- Sample a representative dataset,
- Prevent bias amplification,
- Apply Bayesian reasoning correctly,
- Carefully design sensible actions,
- Only follow causal patterns?

Can we then permit ourselves some profiling?

104

Let's take one final look at our question, including everything we've learned.

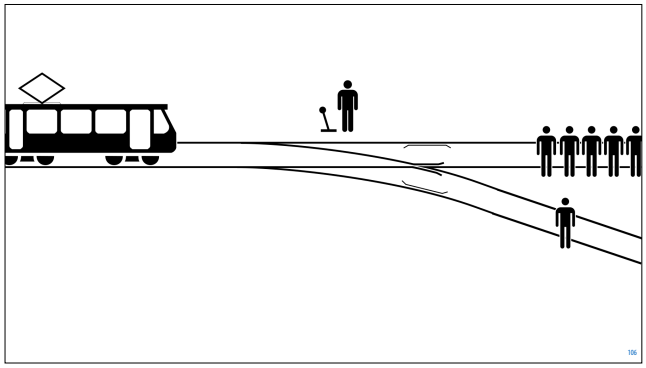
Say we somehow get a representative dataset, which is difficult. We somehow prevent bias amplification, which may be impossible. We apply Bayesian reasoning correctly, which is possible, we carefully design sensible actions based one some quantification of cost, which is very difficult. And we take care to consider all causal relations to avoid inadvertent costs and feedback loops, which is difficult at best.

Imagine a world where we can do all this, and get it right. Are we then justified in applying profiling?

Consequentialism: the consequences of our actions determine how ethical our actions are.

105

What we have taken so far is a purely **consequentialist** view. The consequences of our actions are what matters. The more positive those consequencesm the more ethical the system is, and vice versa.



106

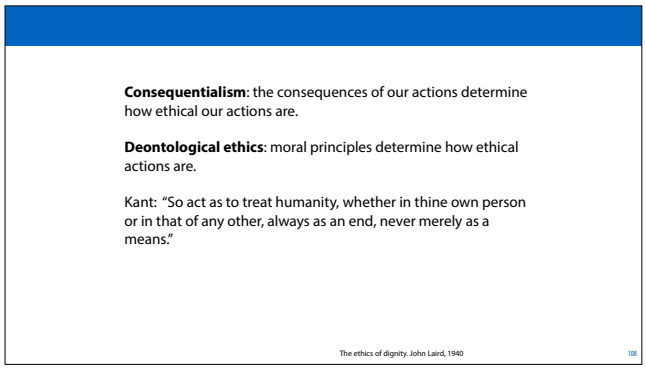
Consider the famous trolley problem: there is a an out of control trolley thundering down the tracks towards five people, and you can throw a switch to divert it to another track with one person on it. This illustrates some of the pitfalls of consequentialist thinking.

The consequentialist conclusion is that throwing the switch is the ethical choice. It saves five lives and sacrifices one.

Now imagine a maverick doctor who decides that he will kill one person, harvest their organs, and use them to save five terminally ill people in need of transplants. With two kidneys, two lungs and a heart he should easily be able to find the patients to save.

From a consequentialist perspective, this is exactly the same as the trolley problem. And yet, we can be certain that many of the people who considered throwing the switch in the trolley problem to be the ethical choice, would not be so certain now.

Without taking a position ourselves, what is that makes the difference between these two situations? Why is the second so much less agreeable to many people?



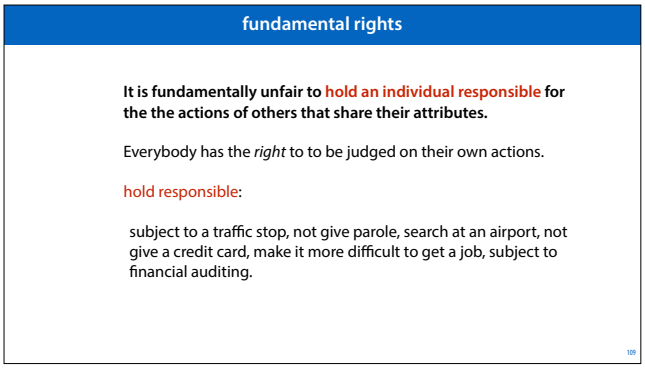
Without going into details, we can say that some actions are in themselves more morally disagreeable than others, regardless of the consequences. This quality, whatever it is, leads to **deontological ethics**. Ethical reasoning based on fundamental moral codes, regardless of consequences.

Such codes are often tied to religion and other aspects of culture, but not always. Kant's *categorical imperative* is an example of a rule that is not explicitly derived from some religious or cultural authority. Broadly, it states that to take an ethical action, you should only follow a rule if you would also accept it as a universal rule, applying to all.

One aspect that crops up in deontological ethics is that of **human dignity**. This may be an explanation for the discrepancy between the trolley and the doctor. Flipping the switch is a brief action made under time pressure. This is in contrast to the premeditated murder and organ harvesting of an innocent person. The latter seems somehow a deeper violation of the dignity of the person, and therefore a more serious violation of ethics.

Kant, again, considered this a foundational principle of basic morality, to treat another human being as a means to an end, rather than as an end in themselves is to violate their dignity.

Consider the difference between killing a human being in order to eat them and killing a human being to get revenge for adultery. From a consequentialist perspective, the first has perhaps the greater utility: in both cases, someone dies, but in one of them we get a meal out of it. From the deontological perspective of human dignity, the first is the greater sin. When we cannibalize someone, we treat them as a means to filling our stomach, without regard for their humanity. When we kill out of revenge, even though it may be wrong or disproportional, we treat the other as a human being and our action is directly related to one of theirs.



To bring this back to our example, we can now say that our analysis of racial profiling is entirely consequentialist. We have been judging the cost of our actions and trying to maximize it by building the correct kind of system. It is perhaps not surprising that a lot of AI ethics follows this kind of framework, since optimizing quantities is what we machine learning researchers do best.

The deontological view, specifically the one focused on human dignity, gives us a completely different perspective on the problem. One that makes the correctness and efficacy of the system almost entirely irrelevant. From this perspective it is **fundamentally** unjust to hold a person responsible for the actions of another. If we are to be judged, it should be on our own actions, rather than on the actions of another.

To prevent crime from being committed, or to make some reparations after a crime is committed, some people need to suffer negative consequences: this ranges from being subjected to traffic stops to paying a fine. A just system only subjects those people to these negative consequences, that committed or planned to commit the crime. From this perspective, racial profiling, even if we avoided all the myriad pitfalls, is still a fundamental violation of dignity. It treats the time and dignity of Black people as a means to an end, trading it off against some other desirable property, in this case, a reduction of crime.

While human dignity is often posed as hard constraint: something that should never be violated, in many cases this cannot be reasonably achieved. For instance, any justice system faces the possibility of convicting innocent people for the crimes of others. The only way to avoid this is to convict no one, removing the justice system entirely. So, we allow some violation of human dignity in order that we can punish the guilty.

However, if we do have to suffer a certain probability that our dignity will be violated, we can at least ask that such violations are doled out uniformly.

Most of my examples in this video were from a few years ago, and our community began seriously working on these probably around the time the ProPublica piece about the Northpointe system broke, almost five years ago now. You may expect, that after all that time, and so much scrutiny, we have learned our lesson, and that at least such gross mistakes as the Northpointe scandal won't be made again.

Less than a month ago as I record this, however, the Dutch government fell. In a parliamentary investigation at the end of last year, it was found that the tax service had wrongly accused an estimated 26 000 families of fraudulent claims for childcare benefits, often requiring them to pay back tens of thousand of euros, and driving

them into financial difficulty.

There were many factors at play, but an important problem that emerged was the use of what were called "self-learning systems." In other words, machine learning. One of these, the risk-indicator, candidate lists for people to be checked for fraud. The features for this classification included, among other things the nationality of the subject (Dutch/non-Dutch). The system was a complete black box, and investigators had no insight into why people were marked as high risk. People with a risk level above 0.8 were automatically investigated, making the decision to investigate an autonomous one, made by the system without human intervention.

https://www.groene.nl/artikel/opening-the-black-box

https://autoriteitpersoonsgegevens.nl/sites/default/files/atoms/files/onderzoek_belastingdienst_kinderopvangtoeslag.pdf

One of the biggest criticisms of the tax service in the child welfare scandal is how few of the people involved understood the use of algorithms in general, and the details of the algorithms they were using specifically.

This hopefully goes some way towards explaining why we've felt it necessary to discuss social impact in these lectures. We're teaching you how to build complex systems, and history has shown again and again that policy makers and project managers are happy to deploy these in critical settings without fully understanding the consequences. If those responsible for building them, that is you and me, don't have the insight and the ability required to communicate the potential harmful social impacts of these technologies, then what chance does anybody else have?

image source: ***https://www.trouw.nl/nieuws/ouders-bij-debat-toeslagenaffaire-mijn-leven-is-naar-de-klote~bc3f3e52/***