NAÏVE BAYES M.L. THEORY

note when most people want to learn

about **naive Bayes they want to learn**

**about the multinomial naive bayes**

**classifier** and that's what we talk about

in this video

however just know that there is another

commonly used version of naive Bayes called Gaussian naive Bayes

classification .

NAÏVE BAYES classification technique is based on Bayes theorem with assumption that features are independent of each other .

now imagine we received normal messages

from friends and family and we also

received spam unwanted messages that are

usually scams or unsolicited

advertisements and we wanted to filter out the spam messages so the first thing

we do is make a histogram of all the

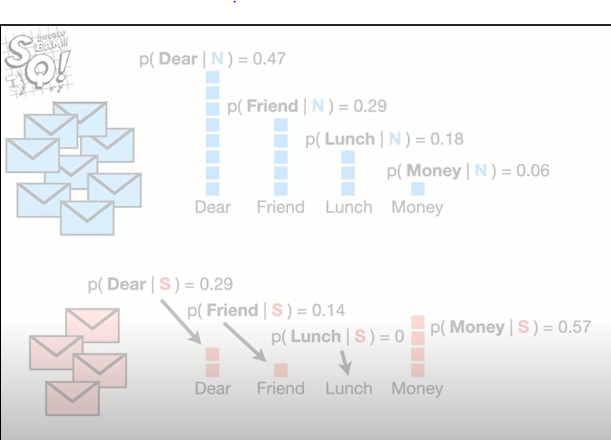
words that occur in the normal messages

from friends and family we can use the

histogram to calculate the probabilities

of seeing each word given that it was in

a normal message .



for example the

probability we see the word dear given

that we saw it in a normal message

is eight the total number of times deer

occurred in normal messages divided by

17 the total number of words in all of

the normal messages

and that gives us 0.47 so let's put that

over the word dear so we don't forget it

likewise the probability that we see the

word friend given that we saw it in a

normal message is 5 the total number of

times friend occurred in normal messages

divided by 17 the total number of words

in all of the normal messages and that

gives us zero point two nine so let's

put that over the word friend so we

don't forget it likewise the probability

that we see the word launch given that

it is in a normal message is 0.18

and the probability that we see the word

money given that it is in a normal

message is 0.06 now we make a histogram

of all the words that occur in the spam

and calculate the probability of seeing

the word dear given that we saw it in

the spam

and that is two the number of times we

saw deer in the spam divided by seven

the total number of words in the spam

and that gives us zero point two nine

likewise we calculate the probability of

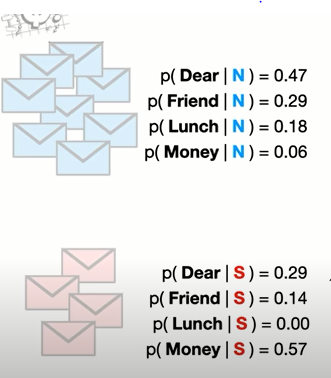
seeing the remaining words given that

they were in the spam BAM now because

these histograms are taking up a lot of

space let's get rid of them but keep the

Probabilities



Because we have

calculated the probabilities of discreet

individual words and not the probability

of something continuous like weight or

height these probabilities are also

called likelihoods

I mention this because some tutorials

say these are probabilities and others

say they are likelihoods in this case

the terms are interchangeable so don't

sweat it we'll talk more about

**probabilities versus likelihoods** when we

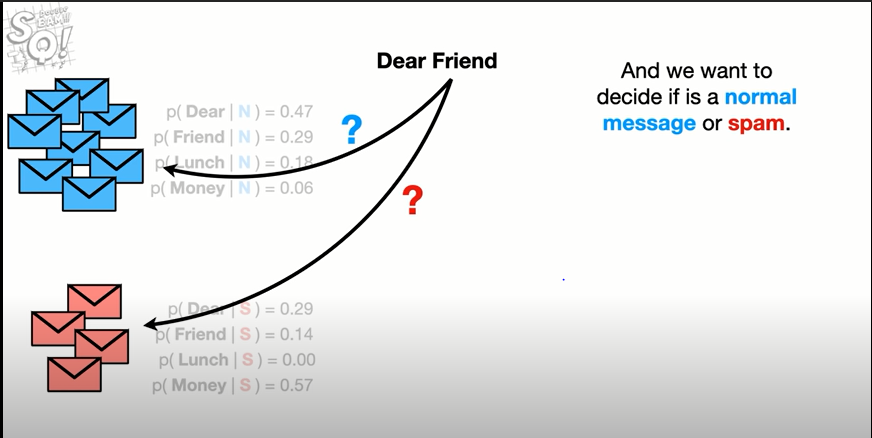
talk about Gaussian naive Bayes in the

follow-up lectures.

Now imagine we got a new message that

said dear friend and we want to decide

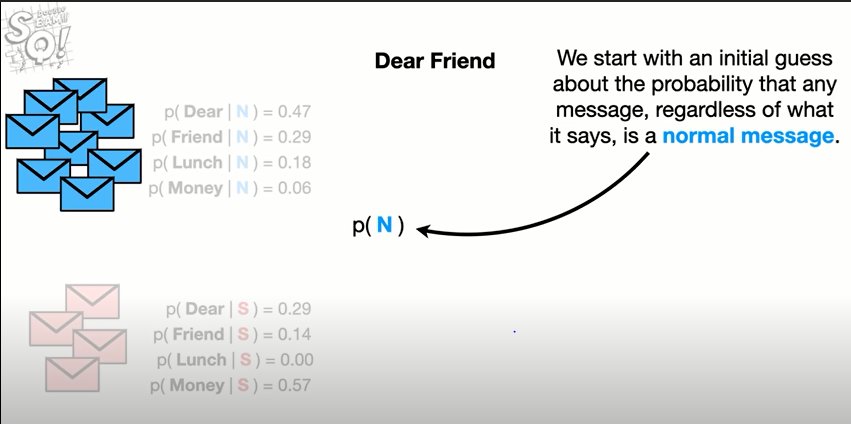
if it is a normal message or spam ????



we start with an initial guess about the

probability that any message regardless

of what it says is a normal message



this

guess can be any probability that we

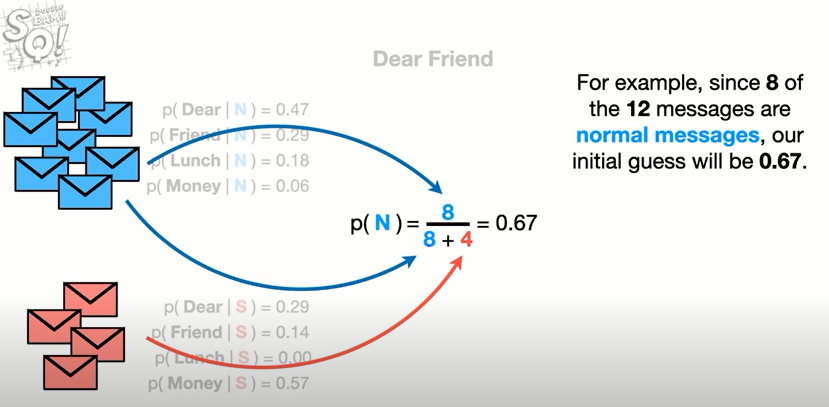
want but a common guess is estimated

from the training data for example since

8 of the 12 messages are normal messages

our initial guess will be 0.67. So let's

put that under the normal messages



so we

don't forget it

oh no it's another dreaded terminology

alert the initial guests that we observe

a normal message is called a **prior**

**probability**

now we multiply the initial guess by the

probability that the word dear occurs in

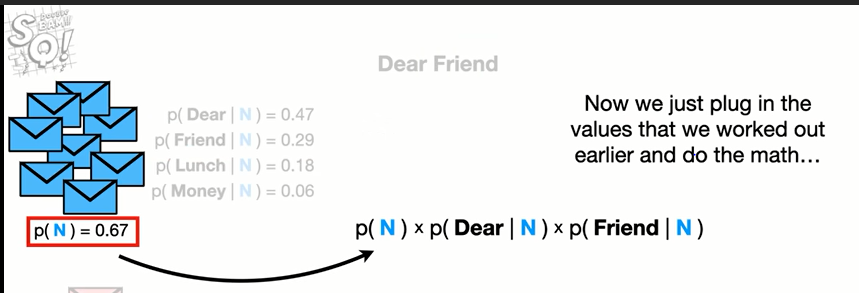
a normal message and the probability

that the word friend occurs in a normal

message

now we just plug in the values that

we've worked out earlier and do the math



it and we get 0.09.

**we can think of 0.09 as the score that**

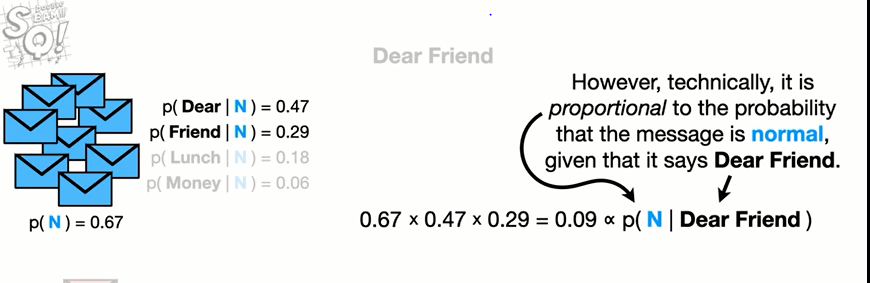
**dear friend gets if it is a normal**

**message . however technically it is**

**proportional to the probability that the**

**message is normal given that it says**

**‘dear friend’.**



so let's put that on top of the normal

messages .

Now just like we did before we start

with an initial guess about the

probability that any message regardless

of what it says is spam

and just like before the guests can be

any probability we want but a common

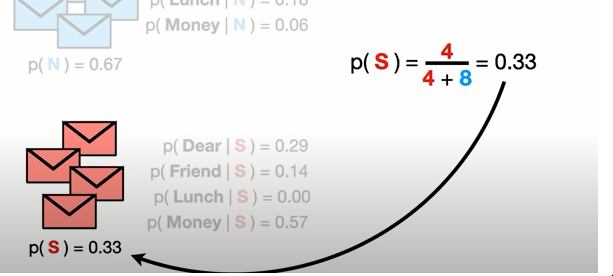
guess is estimated from the training

data

and since four of the twelve messages

are spam our initial guess will be 0.33

So let's put that under the spam .

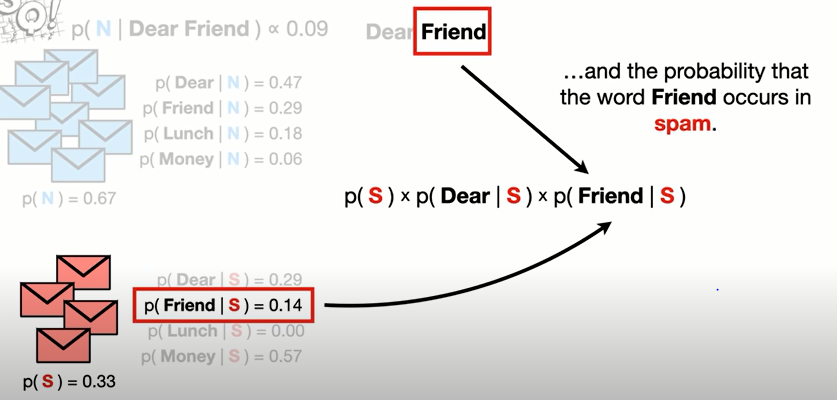


now we multiply that initial guess by

the probability that the word dear

occurs in spam and the probability that

the word friend occurs in spam



now we just plugged in the values that

we worked out earlier and do the math

and we get 0.01

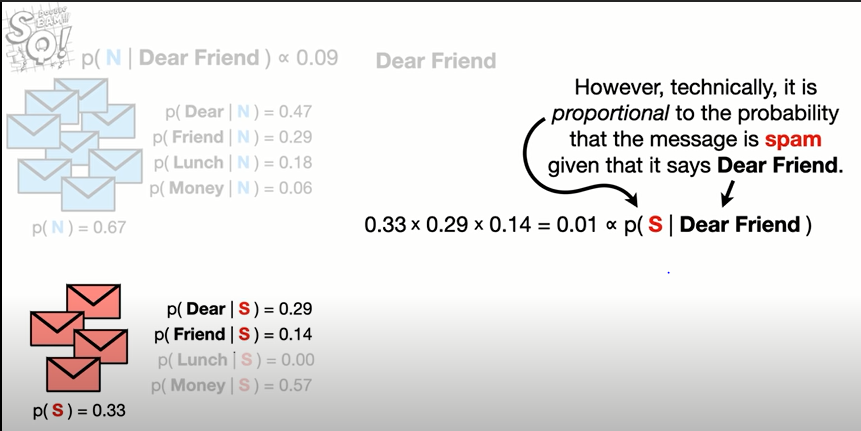
**Like before we can think of 0.01 as the**

**score “the dear friend gets if it is spam”**

**however, technically it is proportional**

**to the “probability that the message is**

**spam given that it says ‘dear friend’”.**

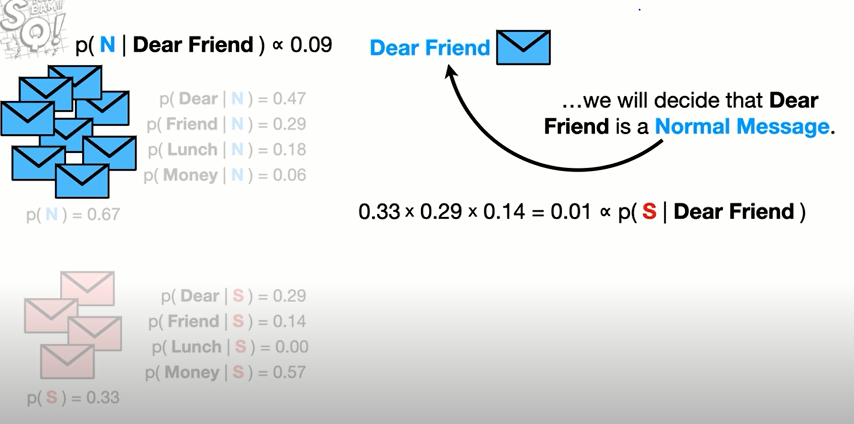


And because the score we got for normal

message 0.09 is greater than the score

we got for spam 0.01 we will decide that

dear friend is a normal message .



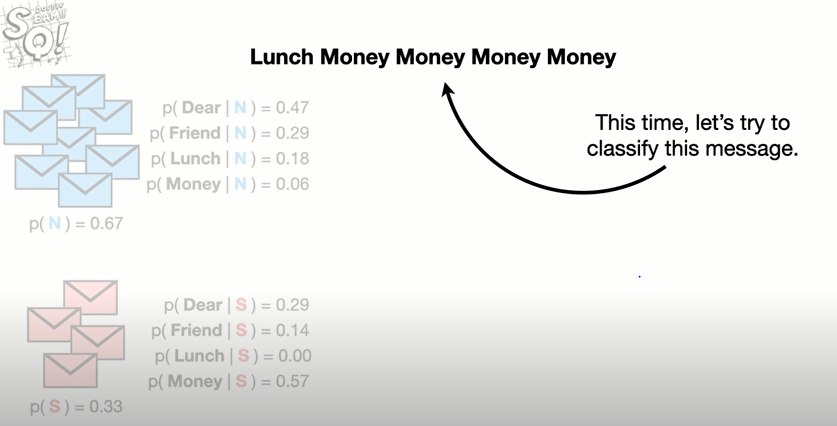
Example 2 :

Lets look at a slightly more complicated

example

this time let's try to classify this

Message “ **lunch money money money money** “



note : this message contains the word

money four times

and since the

probability of seeing the word money is

much higher in spam than in normal

messages then it seems reasonable to

predict that this message will end up

being spam so let's do the math

calculating the score for a normal

message works just like before we start

with the initial guess then we multiply

it by the probability we see lunch given

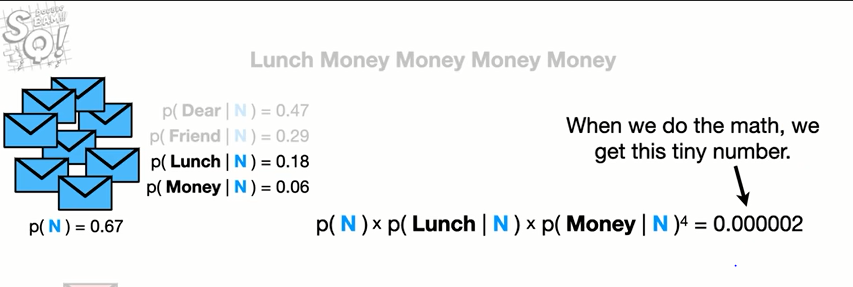
that it is in a normal message and the

probability we see money four times

given that it is in a normal message

when we do the math we get this tiny

Number = 0.000002



however when we do the same calculation

for spam we get zero

this is because the probability we see

lunch in spam is zero since it was not

in the training data and when we plug in

zero for the probability we see lunch

given that it was in spam then it

doesn't matter what value we picked for

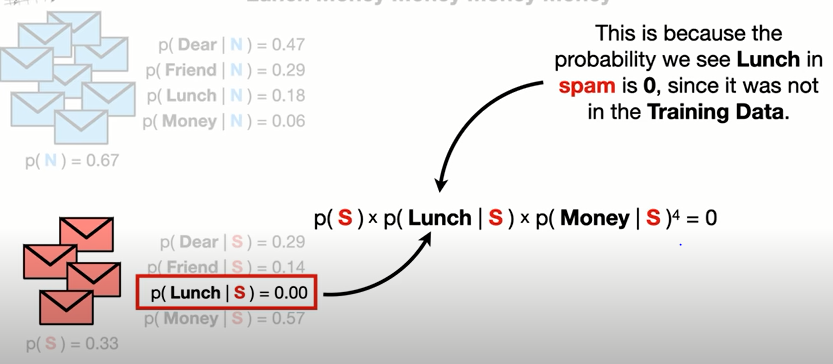
the initial guess that the message was

spam and it doesn't matter what the

probability is that we see money given

that the message was spam because

anything times zero is zero



in other words if a message contains the

word lunch it will not be classified as

spam and that means we will always

classify the messages with lunch in them

as normal no matter how many times we

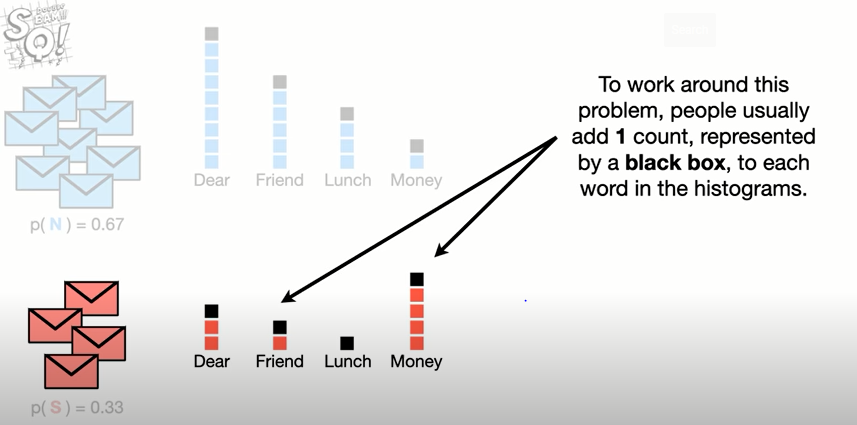
see the word money

**and that's a problem**

to work around this problem **people**

**usually add one count represented by a**

**black box to each word in the histograms**



note the number of counts we add to each

word is typically referred to with the

Greek letter ‘alpha’

.**In this case alpha**

**equals one but we could have said it to**

**Anything.**

anyway now when we calculate the

probabilities of observing each word

we never get 0 for example the

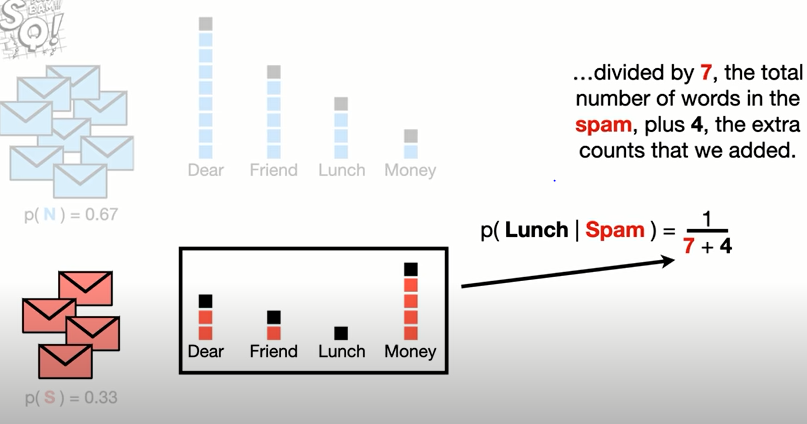
probability of seeing lunch given that

it is in spam is 1/7 the total number of

words in spam plus for the extra counts

that we added

and that gives us 0.09 .

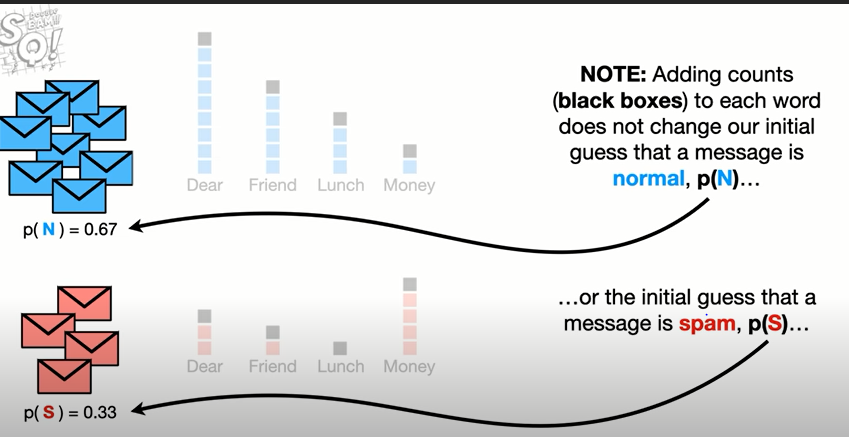
Note adding

counts to each word does not change our

initial guess that a message is normal

or the initial guess that the message is

spam ,

because adding a count to each word

did not change the number of messages in

the training data set that are normal or

the number of messages that are spam

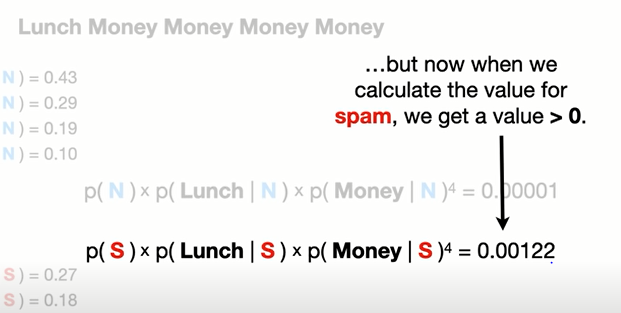
now when we calculate the scores for

this message we still get a small number

for the normal message

but now when we calculate the value for

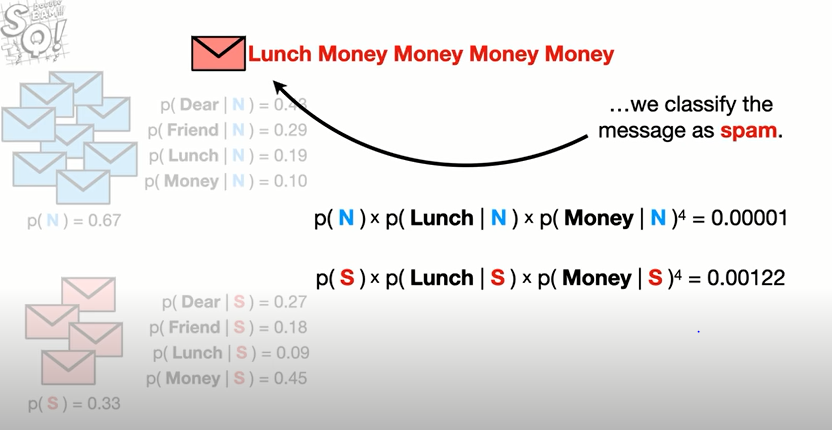
spam we get a value greater than zero



and since the value for spam is greater

than the one for a normal message we

classify the message as spam



Now let's talk about why naive Bayes is

Naïve. the thing that makes naive Bayes

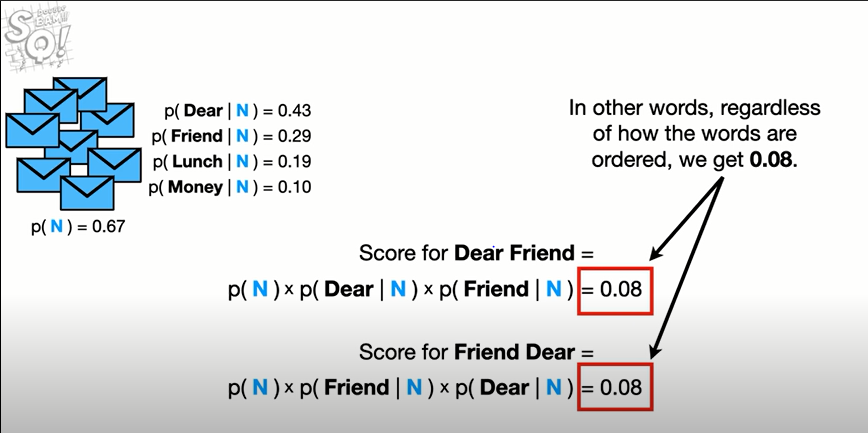
so naive is that it **treats all word**

**orders the same** for example the normal

message score for the phrase dear friend

is the exact same for the score for

friend dear.

In other words regardless of

how the words are ordered we get “ 0.08”

Treating all word orders equal is very

different from how you and I communicate

every language has grammar rules and

common phrases but naïve bayes ignores

all of that stuff

instead naivebayes treats language like

it is just a bag full of words and each

message is a random handful of them

naive bayes ignores all the rules

because keeping track of every single

reasonable phrase in a language would be

impossible that said even though naive

bayes is naive it tends to perform

surprisingly well when separating normal

messages from spam.

In machine learning lingo we'd say that

**by ignoring relationships among words**

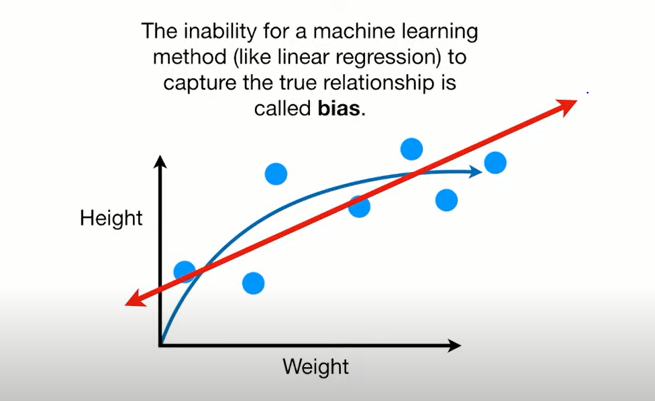
**naivebayes has high bias but because it**

**works well in practice naive Bayes has**

**low variance.**

**Bias and Variance explained :**

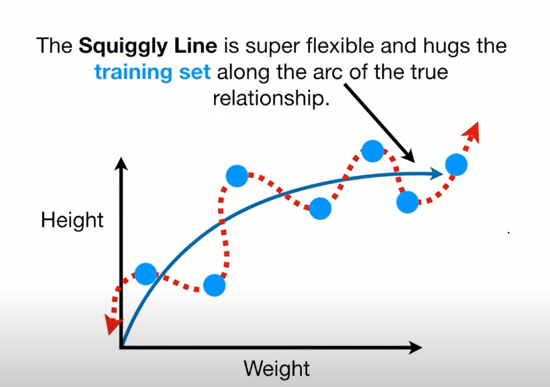
EXAMPLE :



Because the Straight Line can't be

Curved(SEE NEXT FIG) like the "true" relationship, it

has a relatively large amount of bias.

Because the Squiggly Line can handle the arc

in the true relationship between weight and

height, it has very little bias.

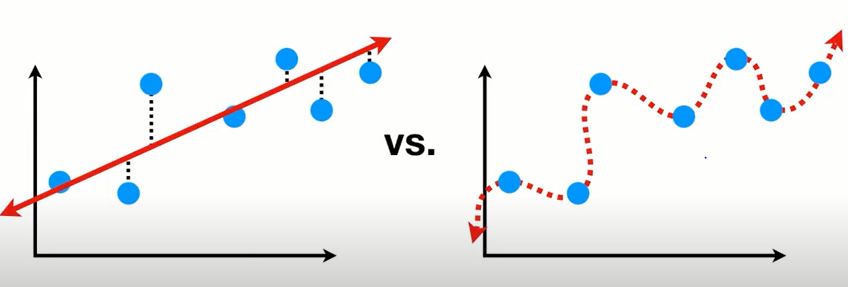
DEFINATION: we measure the distances from

the fit lines to the data, square them and add

them up to get bias. They are squared so that negative distances DO NOT cancel out each other while adding.

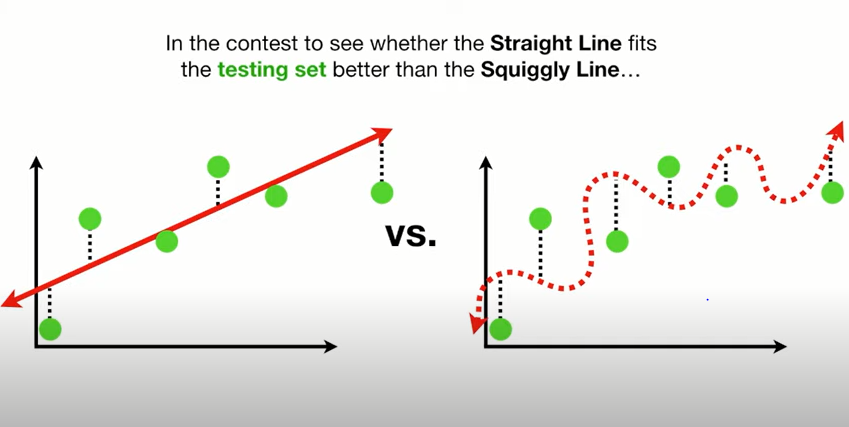
In the contest to see whether the Straight Line fits

the **training set** better than the Squiggly Line... the squiggly line WINS (see fig below).



BUT,

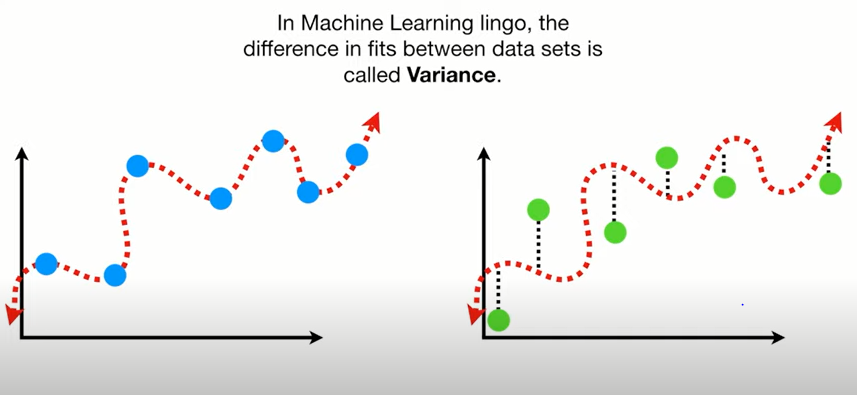
WE RUN THE TEST ON TESTING DATA ……………In the contest to see whether the Straight Line fits the **testing set** better than the Squiggly Line, the straight line WINS.(see fig below)



ln Machine Learning lingo, the

difference in fits between data sets(i.e difference in fits b/w training and testing sets) Is

called Variance.



The Squiggly Line has low bias, since it is

flexible and can adapt to the curve in the

relationship between weight and hight(see 1st part of above fig.)... ...but the Squiggly Line has high variability,

because it results in vastly different Sums of

Squares for different datasets(see 2nd part of above fig.)

GAUSSIAN NAÏVE BAYES

Recall: Gaussian plot is also called as Normal Distribution curve.

Imagine we wanted to predict if someone

would love the 1990 movie

troll 2 or not

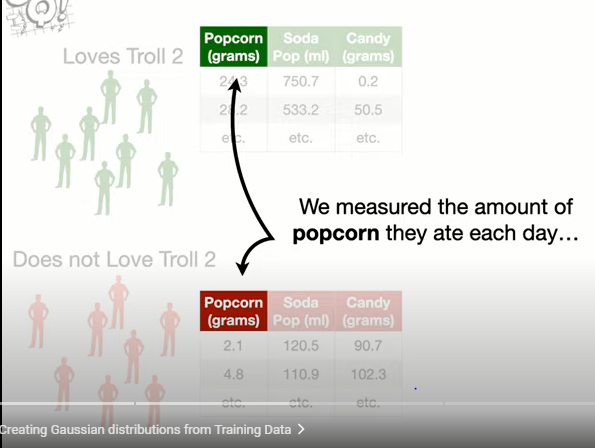
so we collected data from people that

love troll 2

and from people that do not love troll 2

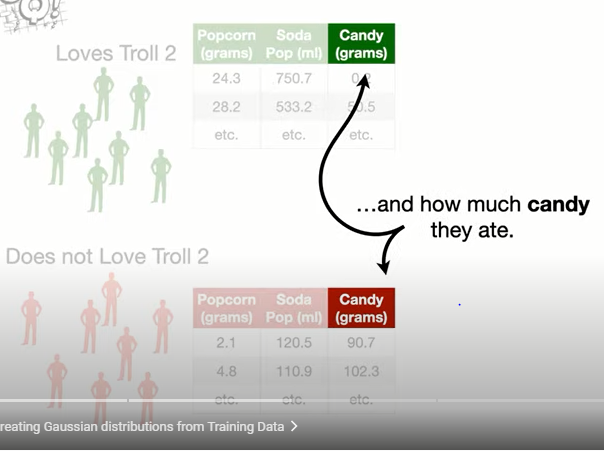
we measured the amount of popcorn they

ate each day



how much soda pop they drank

and how much candy they ate



the mean for popcorn for the people who

love troll 2

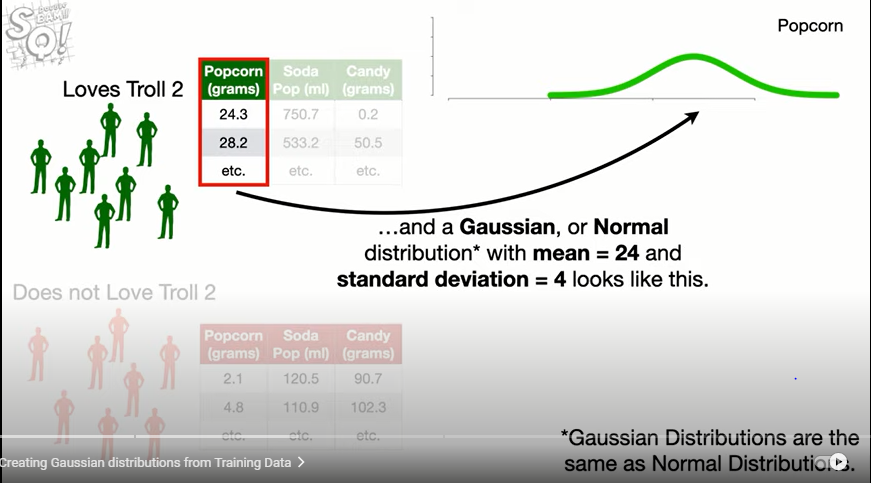
is 24 and the standard deviation

is 4 and a gaussian or

normal distribution with mean equals 24

and standard deviation equals 4 looks

like this



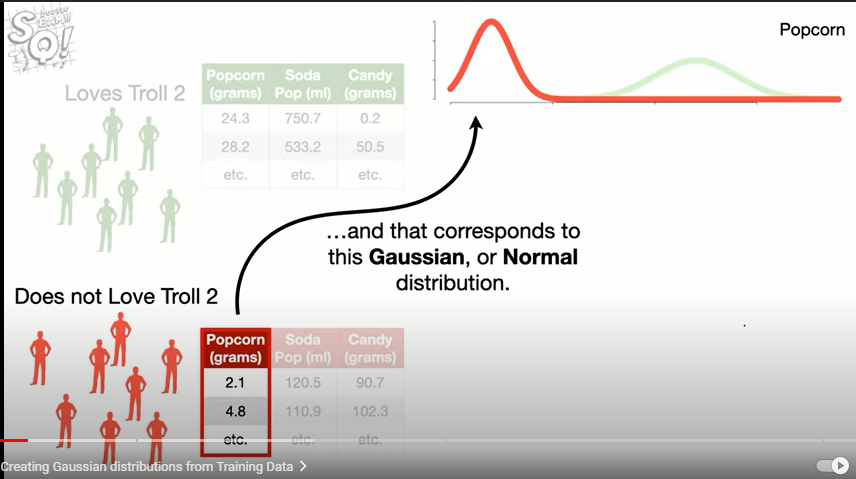
likewise the average amount of popcorn

for people who do not love troll 2

is 4. and the standard deviation

is 2 and that corresponds to this

gaussian or normal distribution



now we calculate the mean and standard

deviation for soda pop for people that

love troll 2

and draw the corresponding gaussian

distribution

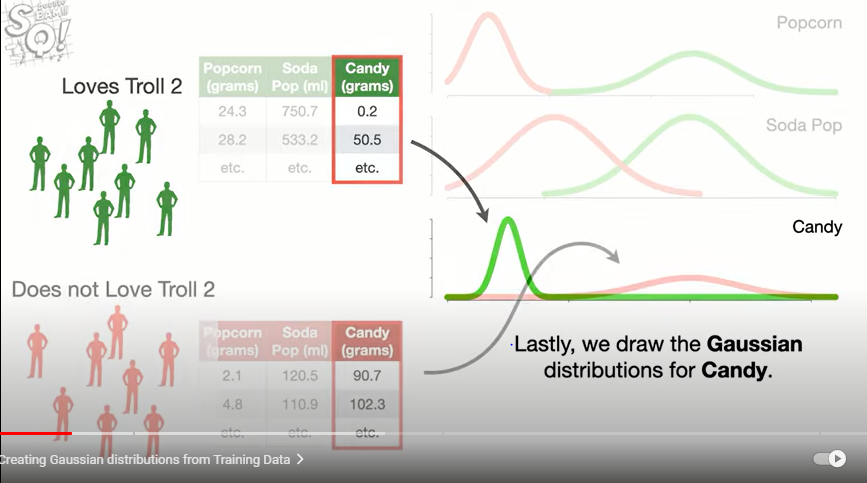
then we do the same thing for the people

that do not

love troll 2. lastly

we draw the gaussian distributions for

Candy(see below)



**Gaussian naive bayes is named after the**

**gaussian distributions that represent**

**the data**

**in the training data set.**

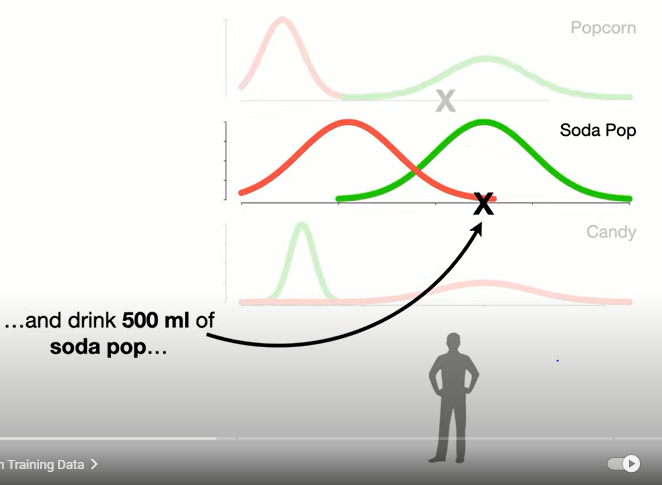
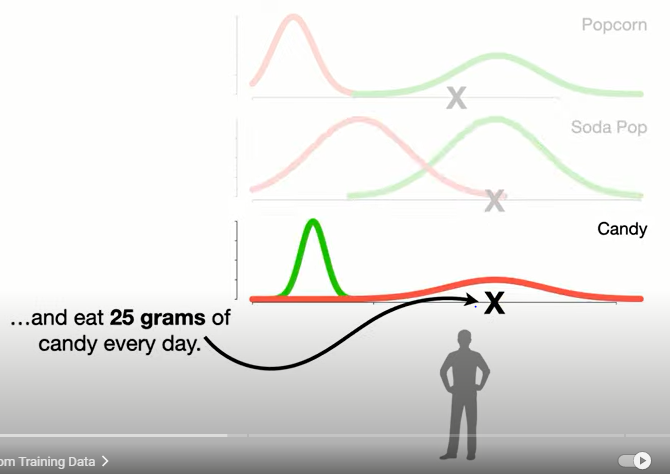
Now someone new

shows up

and says they eat 20 grams of popcorn

and drink 500 milliliters of soda pop

and eat 25 grams of candy every day

**let's use gaussian naive bays to decide**

**if they love troll 2**

**or not???**

The first thing we do is make an

initial guess that they love troll 2.

this guess can be any probability that

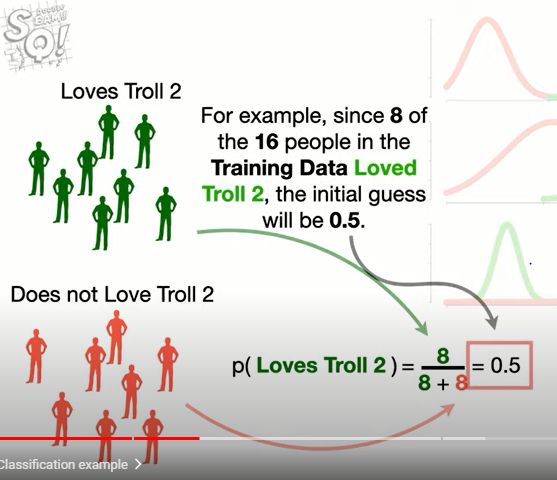
we want but a common guess is

estimated from the training data

for example since 8 of the 16 people in

the training data

loved troll 2 the initial guess will be 0.5



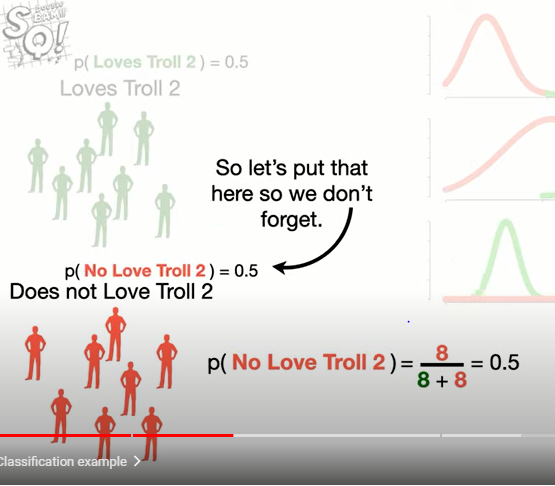
so we'll put that up here so we don't

forget

likewise the initial guess for does not

love

troll 2 is 0.5



The initial guesses are called **prior**

**probabilities**

now the score for love's troll 2

is the initial guess that the person

loves troll 2 times the likelihood that

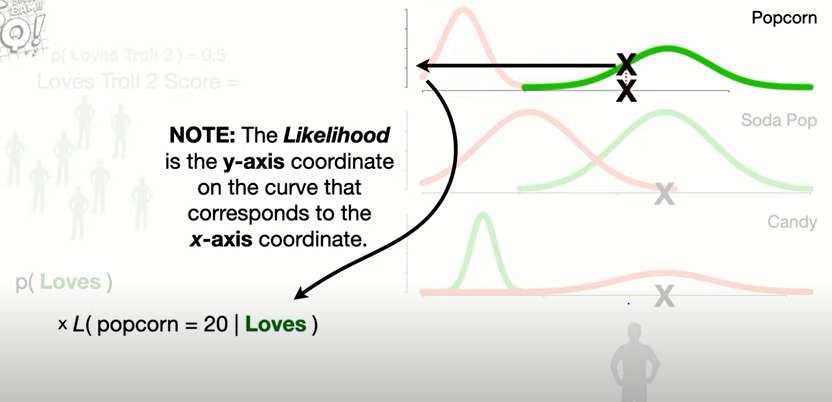
they eat 10 grams of popcorn

given that they love troll 2.

**Note** the likelihood is the y-axis

coordinate on the curve that corresponds

to the x-axis coordinate



and we multiply that by the likelihood

that they drink 500 milliliters of soda

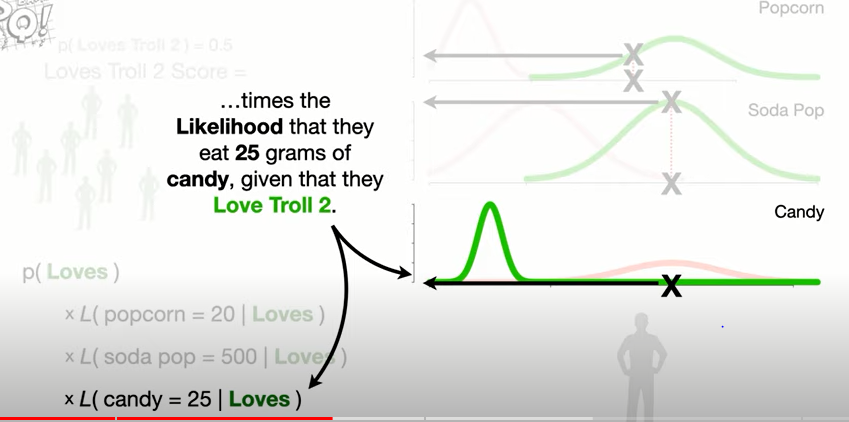
pop

given that they love troll 2

times the likelihood that they eat 25

grams of candy

given that they love troll 2.

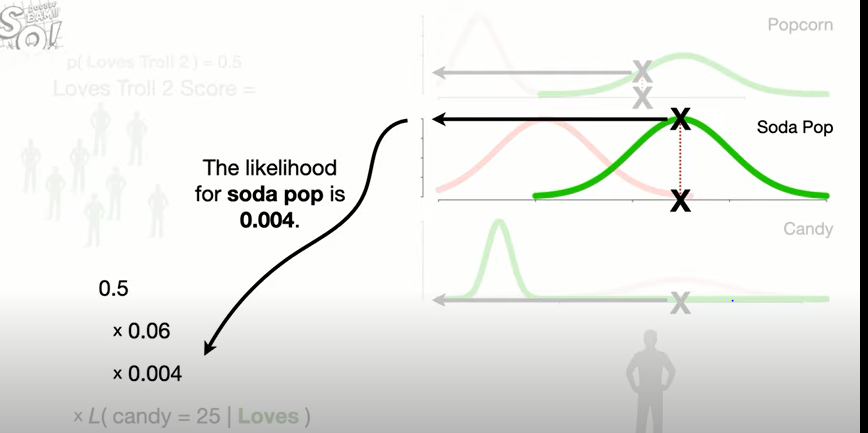


the initial guess that someone loves

troll 2 is 0.5

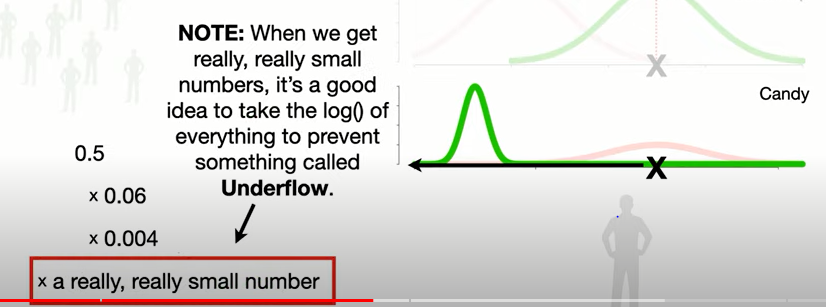
the likelihood for popcorn is 0.06

the likelihood for soda pop is 0.004



and the likelihood for candy is

a really really small number….



Note when we get really really small

numbers

it's a good idea to take the log of

everything to prevent something called

**Underflow**. The general idea of underflow

is every computer has a limit to how

close a number can get to zero

before it can no longer accurately keep

track of that number

when a number gets smaller than that

limit we run into

underflow problems and errors occur

so we use the log function to avoid

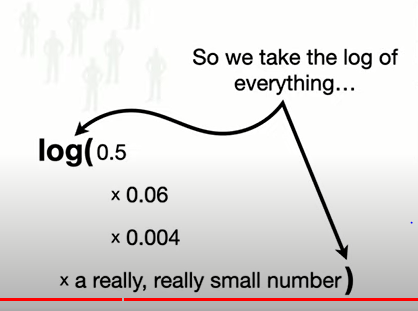
Underflow.

Note any log will do but the natural log

or log base e is the most commonly used log

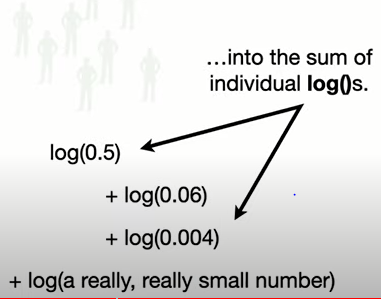
in statistics and machine learning

so we take the log of everything



and the log turns the multiplication

into the sum of the individual logs



the log base e of 0.5

is negative 0.69

the log of 0.06 is negative 2.8

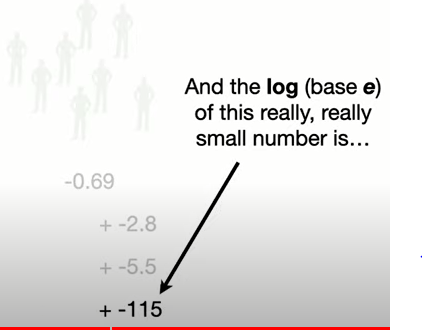
the log of 0.004 is

negative 5.5

and the log of this really really small

number is

negative hundred fifteen.

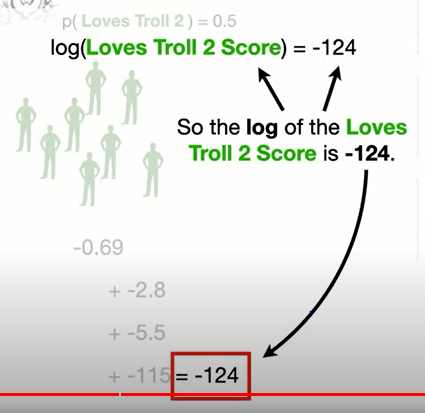


Now we just add this up and we get

negative one hundred twenty four (= -124) .

So the log of the love's troll 2 score

is negative one hundred twenty four .



Now let's calculate the score for not

loving troll 2.

we start with the initial guess that

someone does not love troll 2

times the likelihood that they eat 20

grams of popcorn

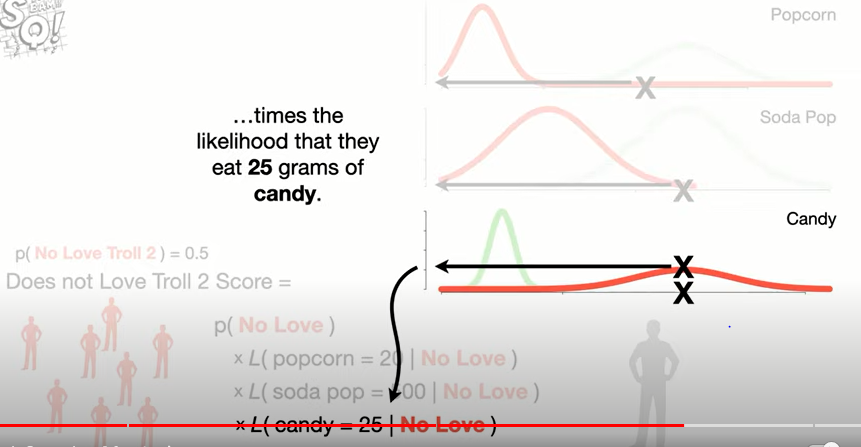
given that they do not love troll 2

times the likelihood that they drink 500

milliliters of soda pop

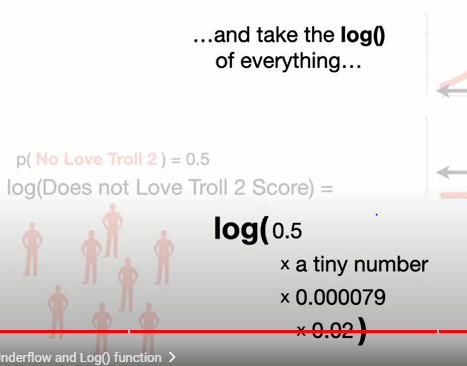
times the likelihood that they eat 25

grams of candy.



So let's plug in the numbers

and take the log of everything



and that

turns the multiplication

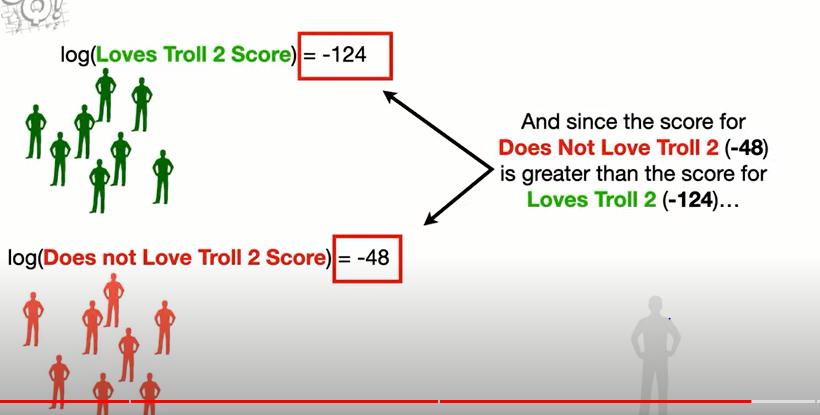
into the sum of logs now we just do the

math

and we get = -48 .

And since the score for does not love

troll 2 is greater than the score for love's troll 2



we will classify this person as someone

**who does not love troll 2 (Answer).**