Week 2 Video Transcript

0:00

okay so this is the second week um and we we are talking about

0:06

supervised learning more specifically about um classification and

0:13

regression and we will also um talk a little bit more about

0:20

uh the evaluation of models um and

0:25

how the learning process works so um just to uh just to recap what the

0:36

machine learning uh field is is this the sub field of artificial intelligence

0:42

where uh we are studying models or algorithms that can learn from data and

0:48

generalize to uh unseen data um without

0:53

explicit instructions obviously so we particularly when we're talking about

0:59

supervised learning and this is kind of a recap we're talking about

1:05

having a data set with uh variables

1:10

called features so the columns of a data set uh and dependent variable or the

1:17

target right so the output so usually you have um so usually you have let me

1:25

just uh change the color here usually you have a table where you have the

1:30

features and one of the features is the target or the or the class

1:36

um or the the value that you want to predict so we're talking about predicting this is the features or the

1:44

attributes uh each column of that so in the end of

1:49

the day through a what we call a training algorithm we will learn how to

1:56

predict the output giving the inputs um or the features right so that's what

2:03

we're learning we're learning a function that maps this uh columns this features

2:11

into uh into our target okay so more

2:18

specifically when we when we're talking about classification we're talking about um uh

2:26

predicting a category right so you could you could think about predicting a dog

2:31

or a cat from an image you uh you can um you can uh uh for example think about

2:40

an uh an application where you want to predict the eye colors uh of of people i

2:49

there's lots and lots of applications where the target is actually

2:54

um discrete categorical okay so you have a a limited a limited number of ca

3:02

categories uh when the target is numeric so it's a

3:07

continuous numeric value then we're talking about a prediction that is called a regression

3:14

okay so you're regret you're you're doing a regression you're finding this function it receives inputs and

3:20

then it's predicting a number okay so for example if we want to find out the

3:26

square uh the the price of a house based on the square footage of the house and

3:31

the neighborhood and um other uh like the the age of the house and other and

3:38

other features that is not a classification it's a prediction called regression okay it's not categorical

3:46

labels but actually a number that you're trying to classify okay so we're going

3:51

to start talking about the mother of all machine learning models uh which is so

3:57

regression again it's a continuous numeric target value and classification is just a discrete charact categorical

4:03

value label this all can be just a this not this doesn't need to be a video this

4:09

can be just a recap session for example okay because all of that that I said uh

4:15

up until now is something that they already had in the first week so let's talk about the mother of

4:22

all machine learning algorithms that it's called linear regression and linear regression is is really is is widely

4:28

studied in all um uh STEM courses and and mathematical more mathematical based

4:35

courses and yes it is a machine learning uh algorithm it's a simple one but it it

4:41

it really depicts what it's what what is a a machine learning right so

4:47

um so let's let's try to understand what a linear regression is in its simple uh

4:53

simplest form so in its simplest form is trying to determine a linear relationship between two variables right

5:00

so the input for example the experience you have uh the years of experience and

5:06

you're trying to predict the salary that you're going to earn okay so you're trying in the end of the day to fit an

5:13

equation to fit a line because it's a linear regression we have other types of regression but in the case of a linear

5:19

regression you're trying to fit a line to the data right and and you you

5:26

actually use a learning algorithm and the learning algorithm is this you have a data and during the training during

5:34

the learning you are actually trying to minimize the sum of squares of the

5:41

distance between the points and the regression line so let's say that you

5:47

start randomly with this line here right so you start randomly with this

5:55

line and the goal of this line so in

6:00

case of a a sing a line in one dimension like this is just um a it's beta x +

6:12

epsilon you you will have to find the slope the slope of the line and what we

6:18

call the bias term right um to

6:25

better to to better uh to to better fit

6:31

this data so you can see that the blue line is not fitting the data so you will tweak the beta and the

6:37

epsilon to kind of you know get the line get the line to the

6:45

best fit here okay so that's what we're doing in terms of learning the learning

6:50

algorithm will will get you from a initial random uh um line where you're

6:57

going to randomly assign initial values so this this this beta and epsilon and

7:05

and we can also recap the the the line equation as a uh um as a read section

7:14

like a you know point them to some some kind of a reading that uh that kinds of

7:22

recaps the equation of a line but you will randomly assign values for beta and

7:27

epsilon which is the slope of the of the of the of the line and the

7:34

intercept of the line the point that the line intercepts it's this and the slope

7:40

of the data is beta so how how uh how steep is the is the is the line but

7:48

anyways during the learning process you are you are actually updating the beta

7:53

and the epsilon so you're really kind of you're you're kind of turning the line and and and you're you're you're fitting

8:00

the line in order to make this difference so the difference between the point and the

8:08

line minimal actually it's not the difference between the point and the line but it's the squared difference

8:15

right so every training algorithm every

8:22

training algorithm is a is an optimization process okay so all machine

8:31

learning during learning we're we're doing an optimization optimiz oh my

8:36

godization process okay so you're optimizing beta and epsilon to get this

8:43

this little differences here the squares of these little differences uh minimized

8:49

okay why is that well because when again because we want to uh generalize to

8:57

unseen data when you want to know a new point of your data set so a you're

9:05

you're getting um you are uh hiring a new uh a new

9:12

person and you're actually you want to kind of estimate it's uh he or her

9:19

salary you will kind of put the experience here you will hear oh she uh

9:24

the person has 10 years of experience how much it will earn well I go and and

9:30

I I look up the result in my line oh my salary will be

9:37

$10,000 okay or $100,000

9:42

so that's and maybe you didn't have this this this as a point right so you don't

9:49

have when you have 10 years of experience you didn't have that specific data point in your data in your

9:56

historical data this is kind of a new information so you're just you're just really seeing the result in this

10:03

regression line if you have this blue distances this little blue distances

10:08

minimized you have a better fit okay and the the reason that it's squared is

10:14

because this difference here for example is positive and this one is

10:20

negative and I don't care about it so that's why I square because um I want

10:25

this differences even positive or negative either positive or negative to be minimal okay so this is so the

10:34

training algorithm is an optimization uh algorithm and it has a loss what we call a loss function meaning that during

10:42

training I want to minimize the loss function in this case for linear regression the loss function is actually

10:49

the uh what we call the mean squared error it's it's this function here okay

10:55

so during the training I am minimizing this loss function to do the better fit

11:02

so I'm I I'm fitting I'm choosing this

11:08

uh beta parameter and this epsilon parameter while minimizing this loss

11:14

function and this will give me the better fit the best fit so it is very important now for us to define what is a

11:22

parameter and what is not a parameter or some kind of nomenclature that appears a

11:28

lot that is a hyperparameter okay so a parameter it's

11:33

actually a model parameter is a weight or it's a value inside the model that

11:38

you have to learn so in this case it's beta and epsilon right and we are set

11:45

before um um we are set during this these values are set during the training

11:51

algorithm and they are updated and then finally when learning stops you will have a value for example for the slope

11:59

beta and for the intercept epsilon okay for example this is a linear

12:04

regression in just one dimension meaning that you have only one experience uh only one feature and uh it's really

12:13

simple you have only two parameters to learn so I I this is another uh

12:19

definition that I just said and I think it's it's worth uh it's worth to kind of go through it's a dimension dimension or

12:27

dimensionality it's the number of number of features in the input of the

12:34

model okay uh that you will use as the input so in this case we have only one

12:39

parameter so it's a one-dimensional model you're only kind of taking the

12:45

experience and outputting the salary okay if you were to see this as a data

12:50

set you would have the experience a column of ver lots of experience and the

12:56

salary as the target here and um you would have uh so this is a

13:03

one-dimensional uh data set so the other term that appears is the

13:11

hyperparameter right so h this is very common in in terms of hyperparameters

13:16

and we will see it a little bit more specifically of what it is but hyperparameters are configuration

13:24

settings to control the learning process okay so uh we will see that this is for

13:29

example um the the the rate of update how how do

13:35

you up how uh quick you update this parameters of the model what we call

13:41

learning rate um and we have several other types of hyperparameters and we

13:47

will discuss that right so you you actually choose the hyperparameters

13:52

before the learning starts and it's actually it will kind of um it will kind

13:59

of control the learning process and it requires experimentation you can learn

14:05

uh this the parameters of the model using a set of hyperparameters you can learn using another set of

14:10

hyperparameters this will change the um this will change the final values

14:18

that you will you will find uh and is an experimentation kind of um um it's it's

14:27

it's requires experimentation from the data scientist okay all right so again uh this this

14:36

this little example here is for one dimension but you can have more dimensions for example uh you could try

14:44

to um uh let's say that uh you could try to

14:52

um you could try to predict the the overall height of a

15:00

person using the person's arm length right so uh this also would be a

15:07

one-dimension and um a simple regression model would say okay uh you know uh when

15:15

a person is around um um when a per so like for example uh

15:23

let's so let's see you have the arm length and the

15:29

height you have several historical points

15:34

uh and you're kind of getting the overall trend and this line will

15:40

actually say oh at uh for every 20 cm of arm length well actually the uh the

15:49

height gets 30 cm of increase okay now

15:56

to get the model more accurate you can include more variables so you can get the dimensions up so you can say um so

16:05

you can say well the first variable could be the arm length arm length the

16:11

second variable could be the biological sex uh if it's a a boy or girl the if if

16:18

the another one an x3 here could be if it's a if if the person is athletic or

16:25

not and this extra inputs will help create this multi-dimensional model in

16:32

this case three um that better captures the variations in height and you will

16:38

always have the intercept where the line crosses and again in three dimensions we

16:44

cannot see the graph i cannot kind of show you a figure of a graph uh of of this line in three um

16:52

um actually in three dimensions I could but in in n dimensional problems I

16:57

cannot show you this this uh this line but it is a what we call a linear

17:03

combination of this features okay all right so this is the first um

17:11

it and it's very important to learn about linear regression because it really it really

17:17

is a very simple um model that we can kind of pose all this definitions hyperparameters

17:24

parameters dimensions what is a a training algorithm what is a loss function again this loss function just

17:31

to kind of uh uh finish here is it it it it will it is

17:37

um it is a um an optimization cost right

17:43

or loss so if you have uh if you want to minimize this um this differences here

17:51

for all points so that's why you're summing all of them um um the the the

17:59

the less the um the smaller this value the the

18:06

greater the fit right so that's what we're doing you can choose different loss functions okay um and during

18:14

training for linear regression the mean squared error is kind of the common the commonplace loss function but the loss

18:21

function can also change uh and you can tweak the loss function whatever you're

18:27

doing but that's uh that's essentially what it is okay all right so now let's

18:33

talk about logistic regression logistic regression is one of the most um it's a

18:39

variant of linear regression and probably the most basic classification algorithm so linear regression is uh

18:46

actually used so just go back a little bit linear regression is actually used uh for

18:53

regression all right so you're you're outputting uh the again the the height

18:59

which is a continuous value it's not categorical and um and uh and it's used

19:07

for reg predicting a number for logistic regression although the name is regression uh and you we will see why

19:14

it's a regression um but in the end of the day is used for classification okay so although the

19:20

name although the name comes and brings it regression this is used for classification

19:27

okay and um and

19:32

Um and uh instead of fitting a line to for example numerical value v variables

19:41

um that is presumably has a presumably linear relationship you can predict a a

19:48

categorical output right so here's an example we want to predict

19:56

the um um let's say that we want to predict

20:03

the imagine we want to predict whether a fruit is an apple or an orange based on two features its weight and its color

20:12

okay uh so for example it's a numerical scale let's say that when it's greener

20:18

the score is lower when it's orange or reddish it the score is higher so we would have a data set like the

20:26

color and the um the

20:32

weight and if it's it and it and the class if it's so you have several data

20:38

you're learning for his from historical data and as soon so this is orange this is apple this is orange this is apple as

20:45

soon as a new apple comes in or a new orange comes in you want your model to classify that right So using a a linear

20:53

regression wouldn't help us because we're not predicting a continuous val value okay we're we're we're we're we're we're

21:00

predicting a class so that's why we would do a logistic regression okay so

21:08

instead of fitting a linear line what we do in logistic regression is we fit a

21:16

sigmoid function okay so here so let's say that our output it's zero if it's

21:22

orange our output is one if it's apple and uh uh I'm here showing in this graph

21:29

only about the color like because I'm I'm just uh doing one one dimensions to kind of show you the graph okay two

21:36

dimensions three dimensions I can show you a graph it's a little bit more it's more difficult but after three

21:42

dimensions then I really can't show you a graph okay but anyways you have the you have

21:48

uh you have you want your model to when a color comes in a color and a weight

21:53

but here I'm just showing the color you want to output here's the model you're

21:58

you're fitting a function that when the color comes in you want to see if you want to u classify in apple or orange so

22:06

the straight line is actually you cannot fit a straight line for different classes you can see that some of the

22:14

oranges and some of the apples they have it's it's hard to

22:19

say with the same color let's say the color is here with the say same color

22:25

some instances of this data set are orange oranges some instances are apple

22:30

they're kind of more orang-ish reddish orang-ish kind of mixture okay but it it's not we're not

22:38

able to fit a particular line that will give me good results for this this this

22:44

classification but look at the logistic regression when we fit this function

22:50

we're going to fit this function that is curvy it has an S curve kind of

22:56

shape and it will predict whether Y lies within uh so it's

23:05

it's kind of it's it's kind of this curvy thing and it will it can predict and the sigmoid function is a bounded

23:12

function between zero and one between zero and one and it will give

23:19

you a um it will give you a value all

23:25

right um that is is convenient conveniently put

23:32

between zero and one and will now conveniently tell you the probability of

23:38

a data of a new data falling into a certain class right so let let's say that um this apple this this fruit here

23:48

this fruit here with this color if you go and you go to the curve that you

23:53

fitted it has for example an 80% because this value here is 08 it has a it has a

24:00

80% chance for example to be um um to be

24:06

a an apple right so this is an apple So the logistic regression is

24:12

actually using a probabilistic fitting curve to do a classification and you

24:18

will have a threshold right so uh you will say that things bigger than 0.5 are apple

24:26

things bigger than point uh things less than.5 are um are oranges okay and this

24:34

threshold actually can can actually change okay depending on the distribution of your po of your

24:40

historical data okay um if you if you were to imagine that you didn't have

24:47

this point okay so you can say that actually uh they are not overlapping

24:54

that much so you can with some confidence say that things bigger than 7 are apples so you

25:01

can change the threshold okay and this this changing this the fit of the curve

25:08

the parameters of this S-curve that I'm not going to show the equation because it's a little bit a little bit more uh

25:15

mathematically uh intricate the parameter of fitting this curve and even the threshold you will

25:22

learn you will learn doing the al the training algorithm by minimizing a loss function of the better fit right um in

25:31

this case for classification the the the we usually um the the loss function will

25:38

always also be a mean squared error between uh y1 and y0 so if when we're

25:45

passing this to a to a training algorithm what is orange is zero what is

25:52

we actually transform this class to a numerical value but it's discrete right

25:58

it's not continuous value like height it's discreet it's a category category 0

26:04

category 1 category 0 category 1 category 0 category 1 and we will we

26:10

will uh also minimize the loss function we will choose a loss function that in

26:16

this case uh is the the the mean squared error as well so for example uh let's

26:22

say that the result of this new uh thing should be should have been an apple but

26:28

it outputed 03 and we know that 03 is an orange so uh it should have output one

26:37

and it output 03 so you have a loss of 7 and you want to minimize that loss

26:44

okay all right so the other the uh

26:49

the other [Music] um algorithm that we have is what we

26:56

call uh k nearest um k nearest neighbors okay and although

27:04

this is this is um considered a machine learning

27:09

algorithm it's it actually doesn't have a learning

27:15

um it's it's it doesn't have an it it doesn't have a learning step it's it's

27:20

what we call a non-parametric algorithm okay so we don't have to fit anything

27:26

um we don't have to to this is a this is kind of a it's a machine learning

27:33

algorithm by uh because because it's very because it's used for uh

27:38

classification and um or even regression both of them

27:44

but it's actually very simple and it's it's it's considered to be a machine

27:52

learning algorithm just because it's it's used so much But it doesn't have a learning step it's so it's called

27:59

non-parametric and actually we we use it on the go so the idea is simple right so

28:05

giving a new data point of your for example a

28:11

classification and here's let me crop this this is copied from somewhere else

28:17

uh all right so um so this is

28:22

KN&N K nearest neighbors so let's say that you have male and you want to predict if a person you want to classify

28:30

a person of a male and female giving the height of that person well what one way

28:37

to do is to learn for example a logistic regression the other way to do is actually

28:44

um to say that the gender of the person the new person will be the same as the

28:50

majority of the five people closest to um uh the five people closest in height

28:59

so you're you're analyzing the height let's say that the height is this one here is this height here for this new

29:05

person let's say 179 cm and what you do

29:11

is you analyze the the height of the closest five closest person so what are

29:16

the five closest persons this one

29:21

two three four and five right so these are

29:27

the closest persons that has this heights and then you will say that it

29:33

the you will assign the the

29:39

the class of the majority so the majority are males you will say okay

29:44

this person is also male it's a very very very simple kind of

29:50

algorithm and um it's called K nearest neighbors and the K is actually the num

29:57

this number that you set that that you that you set right it's a it's a

30:03

parameter of the algorithm um because we don't have a learning uh you won't you

30:09

won't learn you will experiment with it okay um and you can also you so you can use

30:17

it for classification classification you can use it for regression uh as well and for

30:24

example as regression you want to determine the you predict the chest circumference giving the height of the

30:31

person let's say that a new person comes in and it has a height of of I don't

30:37

know like as a 50 50.5 right and then you will see okay oh

30:45

I I this is a three nearest neighbor so the the the

30:52

the algorithm will say you will say oh uh I want to to to predict that so the

31:01

the the height the chest circumference of that person will be the average of

31:08

the three closest people right so um if

31:13

the sorry this this is not height uh this is okay blah let me start again

31:19

this is the height uh let's say 180 cm and you can see that I want the chest

31:26

circumference that's what I'm predicting what I'm going to do I'm going to take the average of the three

31:32

people that has uh the closest heights so this pe this person has 181 this

31:38

person has 179 9 this one has 180.5 and they have 50 55 51 I will do the average

31:46

and I'll say that this person has a 52 uh average okay so this is called the k

31:52

nearest neighbors and although it's very simple it can be very very effective still okay to do classation and

32:01

regression okay so now let's see another another model another model is called

32:06

the support vector machines And this model is is a is a machine learning a

32:13

proper machine learning algorithm because it has a learning step right and

32:18

uh it was originally designed for classification tasks but is but it it

32:23

can also be used for regression task and the core concept of the algorithm is to

32:28

draw a decision boundary between uh data points that separates the tr the the

32:36

training set right uh so if you if you kind of get if you draw a decision

32:43

boundary that separates the data points that you're training on as a new unseen

32:50

data point comes in uh hopefully you will possibly classify it um with a good

32:57

uh with a good efficacy okay so let's say that we're trying to classify

33:03

animals by their weight and length right and uh so we have a two-dimensional

33:09

problem we have the length of the uh sorry the the length of the nose and the

33:15

weight of the animal and you want to kind of say if they're cats or elephants

33:21

okay well um you you the

33:27

the SVM will kind of create a decision boundary right and it could be it could

33:33

be the simplest SVM is actually um a fit of a line that's the simplest

33:40

the SVM algorithm the simplest decision boundary that it can trace is a line

33:47

okay so this line that it's tracing can separate perfectly cats from elephants

33:53

okay and the way it does it during the training is it uh it it it maximizes

34:00

what we call the margin and let me redraw because uh it's so here is a

34:07

line and what the oh god this is the

34:13

line and what we're doing is we are during training

34:21

[Music] during training we

34:27

are maximizing what we call uh what we call the margin margin or so this is

34:33

what we call the margin is the space between um it's the space between diff points of

34:41

the different the external points of different classes so this points here

34:47

they are the the most external ones of each class and you want to maximize the

34:53

space between this one and this one so you want to maximize what we call the margin okay

35:01

and the support vectors they they can uh so the loss

35:06

function in the the training of support vectors is the the size of the margin we

35:12

want to maximize the s the size of the margin this is an optimization problem as well and you want to maximize this

35:19

margin um but it it doesn't only um it doesn't only uh do lines it could do any

35:27

kind of uh decision boundary for example this one is circular it could even if

35:33

you want like if this is a simple example and it's very separable with a line but let's say that uh you could

35:40

also do this kinds of of decision boundary okay meaning that if a new

35:47

point comes in like let's say this you will say oh it's it's upward the

35:53

decision boundary so it's an elephant or oh it's it's um it's here so the point

36:00

no point came in it's it's downwards this decision boundary so it's a cat you

36:06

can do a circle like here in the right hand side right so uh for example you

36:12

can have um um

36:18

uh you can have I don't know like apples and oranges and uh the the the weight of

36:25

the oranges and the colors of the oranges and the apples and you can see that no line could separate could do

36:31

could draw a boundary between these classes the SVM can actually draw this

36:36

crazy boundaries and in this example it's drawing a circle boundary a circular boundary okay so SVMs are very

36:46

powerful and to do this kind of squiggles or circle

36:51

boundaries we use what we call kernel functions but this is out of the scope

36:56

and um but it's it's it uh it because it creates this curves you can also use it

37:03

to for regression okay um so in the end of the day you can uh

37:10

you can fit this this decision boundaries as lines as polomials as circles um as u several different

37:19

functions okay and it's it's a fairly simple classifier with no um um um all

37:27

the classifiers up until now they they they don't have lots and lots of

37:34

computational uh burden because they're are simple algorithms linear regression logistic

37:40

regression support vector machines kn&n doesn't even have a learning process so

37:45

there are se they're very they're computationally very efficient and they are very good classifiers and regression

37:52

mechanisms the the la the one one other um

38:02

um one other um classifier

38:09

uh is what what uh what we want what we call the the naive base actually there's

38:16

a family of B what we call base classifiers that that are probabilistic

38:22

classifiers so they they they come up with the probability of something being a class or not just as logistic

38:28

regression um there I'm showing the naive base which is the simple ones but we you have a family of base classifiers

38:36

that has different um that has different assumptions okay so let's talk a little

38:42

bit about base classifier just just a just um something important to say here i'm

38:48

using an example with with some bad words here right um we have to to kind

38:55

of come up with with I think something that's better okay um it could it could

39:01

be like uh because I'm using example of a spam email giving that you have this

39:06

words in the email but we can say um you know like

39:11

um I don't know we can change the words just not to use it but but this is

39:17

actually although this has this words um it's a very common example uh in machine

39:23

learning to show okay although it has this words but anyways we uh we can change that we can come up with uh with

39:30

different words I don't know like um um uh your um

39:39

um you have a prize so you could be like prize and um and

39:47

um lottery something like that well we can we can come up with the example

39:54

okay so the the the base family of classifier classifiers they are based in

40:00

what we call the base theorem right so the base theorem is a theorem um that is

40:06

from the 1800s and uh it's it's a theorem from statistics

40:13

um that um that tells us what's the probability

40:19

of something of an hypothesis given an evidence okay so usually uh and I I

40:27

always like to show this example uh with a what we call a probability tree

40:33

sometimes you can be sick or or healthy and then you can go and so let's say

40:41

you're you're you're um um you think you're having COVID 19 so

40:49

you can be sick or healthy because it could not be COVID 19 and you do a you

40:54

do a test you can have a positive test or a negative test either way if you're

41:00

if you're sick you can have a positive test you're um if you're sick you can

41:07

have a negative test which is not good because you will be a what we call a false uh positive a false negative and

41:15

you can be healthy and you can do a positive or a negative test okay but usually this is like

41:22

the this is what uh um the the tree showed the sequence of

41:28

of kind of natural steps first you are sick or not sick and then you do the

41:34

test right but this is what we call like we cannot see this right we this is kind

41:40

of a hidden step we never know okay so the this is what this is a hypothesis

41:47

this is what naturally happens with our bodies but but the way we the way we kind of test this

41:55

hypothesis is having an evidence so the evidence is this is if

42:01

it you have a positive or negative so this the the base theorem

42:06

says this what's the probability of being sick for example given that I was positive or what's the probability of my

42:14

hypothesis so my hypothesis is sick given that my evidence is positive so I

42:22

I got a positive i could have get a positive from being healthy or for being

42:27

sick and the base theorem gives me this hypothesis okay so I'm not I'm not

42:34

looking at the class I'm not looking at the class of being sick or being healthy i'm looking at at an evidence and the

42:42

base theorem is a equation is a theorem that gives me this answer right so this

42:48

is the nomenclature is what's the probability of being sick given this

42:54

this this dash here this upper I don't know how you call this is is is what we

43:01

call the given given that the evidence is positive and you have an equation right I'm not going to go through the

43:07

equation I'm just leaving it here we could actually point the student to uh I

43:13

have several videos that uh that I use in my lectures um that are very good i

43:18

have there there's lots of other videos that you can point out to the students to kind of understand the B theorem

43:26

better okay but in general this is what the B theorem will

43:31

give you it it will given an evidence it will give you the the probability of the

43:36

hypothesis right um and be it will give you the for

43:42

example the probability of being sick given that you're positive and the probability of also being um healthy

43:48

given that you made a positive exam right and they are they have to sum up to one okay so if you have one you have

43:55

the other so you have the classification in the end of the day you have the classification of of your uh of things

44:02

that you you're not seeing and that's for example what we do with spam fil with spam spam uh email so for example

44:11

what's the probability of an email being spam giving that the evidence so that's

44:18

my class and I like this is the hypothesis it's it's naturally a email

44:24

is a spam or not a spam um but I cannot see that it's kind of hidden to me it's

44:30

it's the question that I'm I'm I'm hypo hypothesizing and the evidence is that it has penis in Viagra for example right

44:36

or the lottery and prize um and and several other words right so there's a

44:43

formula that given by the the base theorem that will give you this

44:49

[Music] um this this value for example here it gave me a

44:56

92% a 92.8% of being spam so and then you will so then you use this as a

45:02

classifier and this would uh this would actually be 7.2%

45:09

uh chance of being um a um oops not spam

45:15

okay a not spam so this is the the not them the naive

45:20

base and it's called naive because this particular uh this particular

45:26

classifier uh assumes that all the evidence are independent like uh the the

45:35

word Viagra is independent of penis right for example um and actually we

45:42

know that Viagra and penis uh in in this pen males they kind of tend to to

45:48

they're not independent if you have one it's probable that you have the other

45:54

but naive base has this assumption that they're independent right um

46:00

um and that's okay it's it's a naive it's it's a it's a it's a simple

46:06

classifier so we call it naive uh but it's still very good it's still very

46:11

good it's still very powerful um um it's still very powerful classifier

46:19

okay so but and it's very efficient it's very simple and it has a learning it has

46:25

a learning stage so how how can it has a learning stage so you have a data set of

46:32

lots of emails and you have uh the columns of I don't know p lottery or

46:39

prize or penis or vi viagra and you have the class of spam which can be zero and

46:46

not spam that can be one and you have several emails okay so you have you have

46:51

a historical data and you want to learn um um from

46:58

this data so what you will see is that if it has penis

47:04

uh and via not have Viagra it could be a span if it has uh not penis and not

47:10

Viagra probably it's not a span sometimes both is a spam if you're doing

47:15

a treatment uh for some kind of disease for some kind of problem uh you you

47:22

could see penis and Viagra and it not it's not a span it's a it's mail it's an

47:28

email from your doctor so I mean you will have historical data and what you're going to do is from this

47:35

historical data you will learn the the parts of the equation you will learn the

47:41

parts of the equation and the parts of the equation are other probabilities so you will you um you will calculate the

47:49

probabilities uh that that you found on your data so there are other probabilities

47:56

uh that that you found on the on the data and then you will multiply all of these probabilities and have the the

48:03

classification so there is a learning uh step where you are actually learning

48:12

some probabilities from the data set

48:18

okay all right then there's decision trees okay decision trees are very

48:25

powerful and one of the most used um algorithms for um uh uh for

48:35

classification and even regression but but here's the here's the thing this is the this is the basis of a number of

48:44

more complex algorithms and very powerful ones so let's talk about it just uh before um I just want to say

48:52

something before we start decision trees uh there is a possibility of kind of

48:57

explaining a little bit better how uh like the parts of the equation and uh

49:02

how the learning works um uh for example for uh but but we can

49:10

do this by um by video or by a lecture or

49:19

uh video lecture or a um reading session but for example you have the the

49:26

historical uh database you have the ones that are spent so you know so you can just

49:33

calculate the probability of uh this word given it's a span and um and the

49:39

other one given that's a spam uh and so that's what you're learning from the

49:47

um from the data okay um learning learning meaning just calculating this

49:54

right so there's not a a loss function um in the sense of minimizing anything

50:00

here uh in other base kinds of algorithms you you will you can associate with the loss

50:06

function okay decision trees so in decision trees uh what is a decision

50:12

tree right so again it is a it's it's very famous and uh the decision trees are um

50:22

a series of questions of yes or no questions that partitions your data

50:28

right so here is for example a tree that classifies if a person is low risk or

50:35

high risk to for example a heart attack okay um so if the so you start at a root

50:45

node as a question a yes no question at the root node so is the age less than 18

50:51

then uh you have to go and look at the weight if it's less than 60 it's a low

50:56

risk um if it's more than 60 meaning that is a it's a young person but it's

51:03

obese then it's a high risk right uh not obese but you know overweight

51:11

um it if it's between 18 and 30 it's low risk

51:16

anyway and so on so what what the what the model

51:22

uh when a new data point comes in so let's say that you have a a data set

51:29

with the attributes age so you have age smoker and weight right weight and here

51:38

if it's low risk or high risk okay so you have uh several instances in this

51:45

data set and you will learn to find this parameters of the this questions of the

51:56

um of the tree right and when a new person comes in let's say it's an 18

52:02

it's a it's a 30 um smoker so one for

52:07

for the smoker column and a weight of uh I don't know a weight of 60 kilos uh you

52:15

will then uh go through the through the model the model and you will have your

52:20

answer so it's more than it's uh more than 30 so let's say 31 31 it's more

52:29

than 30 and it's a smoker so you have a classification of a high-risk person okay so during the learning so what's

52:36

what's the thing to find these splits right to find these parameters of splitting the

52:42

data we the way the algorithm does it is by um is a is the loss function is

52:51

actually uh being to maximize information gain so I'm not going to

52:56

explain uh what is information gain is a is uh so let's say that you have a a

53:02

data set that has outlook windy temp humid humidity and temperature and the class or target uh saying that you can

53:08

play or not outside yes or no so outlook can be sunny overcast or rainy windy can

53:15

be false or true uh humidity high and low and temperature hot mild or cold so

53:22

the the learning algorithm will start ask will start recursively testing what

53:29

is the best root node where to split where where to ask the question if it

53:34

starts asking the question of how is the outlook and branching on the possible

53:40

answers of sunny overcast and rainy it will it measures an information gain of

53:47

0.247 bits and it's a it's it's it's a fractional number like it's not but but

53:54

it's okay it's it's just a metric okay um if it it tests on windy like like

54:00

should I ask first the first node of my tree should ask if it's windy or not if

54:06

it's humidity or if it's temperature so in the end of the day we will choose outlook because it has the higher

54:13

information gain meaning that if you take a look at the results right

54:19

um um um if if you branch the algorithm will branch and and will analyze how

54:25

many instances um you will have in each category right

54:32

of Outlook so for example if it branches on Outlook two instances of your data set

54:39

is yes you can play and three are no you

54:46

cannot um if when it it asks the outlook and it

54:52

uh and you say "Oh it's overcast." Actually you have 1 2 3 four instances

54:57

that are yes and when you ask outlook and you're

55:02

rainy you have two uh instances that are rainy that three instances that are

55:08

rainy that you can play and two not so you can see that the information gain

55:14

has has to do with the pure nodes so look at this in when I branch in Outlook

55:20

and it's overcast it means it all instances that

55:26

are overcast means that I can play so the better the

55:31

tree the the better the pure nodes come out of my uh branching the better the

55:39

tree okay so you can see that this has not resulted in a pure node this has not

55:44

resulted in a pure node no no no no no so obviously this one is the one that

55:51

gives me more information uh information okay so your your tree

55:56

will start uh branching in outlook and then it will continue asking questions

56:02

should after outlook should I branch in temperature right should I branch in

56:08

temperature should I branch in windy should I branch in humidity and we'll

56:14

calculate this information gain and again the gain here of humidity

56:21

is the uh is the best one because it resulted in a pure node you can see here

56:26

right and so you you this is how the algorithm works it's maximizing it's an

56:33

optimization algorithm that is maximizing this information gain or maximizing the trees that have pure

56:40

nodes okay so if we go back to here um the the algorithm

56:47

uh during training learned that branching first at age gives you a

56:53

better tree better classification so it branches first on

56:58

age and it has discovered in the algorithm that less than 18 is where you

57:05

should um um this parameter and this parameter and this parameter was learned

57:12

and after that it's it's recursively tested to split on weight of smoke

57:18

smoker and it chose weight and so on okay so the splits is

57:24

how pure you you have a node so you can look at here you have the instances of a

57:31

for example low risk or high risk if you um if you choose a split here you

57:38

actually don't have pure nodes so you you this tree is less informative it's

57:46

it it does a poor poorer job than the one on the right where you have uh pure

57:52

nodes okay so that's that's how the training works

57:57

okay um and it's based on information gain and the so now I want to talk about

58:05

assemble ensemble algorithms um this very basic algorithm of decision trees

58:12

can uh turn out to be a very powerful uh one when you combine many decision trees

58:18

together um we call it begging right um where uh

58:25

for example let's talk about uh I'm going to talk about only about random forests I don't need so when we're doing

58:32

this image you can take out of this so an example of ensemble algorithm is what

58:37

call a random forest and uh random forests are actually several trees so

58:44

multiple models that um that actually

58:49

vote on the classification of your data and the by the majority um

58:55

um by the majority of votes then you can classify your new data okay and um it

59:03

they are very powerful and they can actually be used both for classification and

59:08

regression and um the way to do this is you so if you

59:16

um uh if you zoom in a little bit you will see that the this the trees will

59:23

vote on the different classes and and they are different trees and the way you

59:29

you you kind of come up with different trees and that's why it's called random

59:34

is that you exclude some nodes of this trees randomly right so one of them uh

59:43

will have uh different so uh different trees will have different nodes that

59:48

you've excluded randomly and this will give you different trees um with different

59:55

characteristics and you put them all to vote on the class and by the majority of

1:00:01

votes you can actually do the classifier okay um so the training algorithm it's

1:00:08

uh you train individual uh trees as we said about the information gain and um

1:00:16

but randomly kind of cutting or pruning some some of the nodes and and then uh

1:00:22

you make this voting uh kind of scheme after okay start neural networks um I want to

1:00:31

present you with the basic unit of a neural network um that I will call a

1:00:38

unit right now uh you can call it neuron it's it it is

1:00:44

um basically inspired in in the human uh brain meaning that in the human brain

1:00:50

you have the units that are connected and um and you can activ you you you can

1:00:57

have different neurons that are activated or not activated in the human

1:01:03

brain so this is how uh we kind of modeled um this networks of this this

1:01:10

units that we will call neurons in ma mathematically okay so I will start with

1:01:17

the most basic unit so it's not a network yet but it's it's it's an a one

1:01:24

neuron okay so uh the the neuron is this kind of circled uh thing you can call

1:01:30

call it a unit and this unit performs a mathematical a mathematical uh

1:01:37

calculation okay and um this unit has this arrows here that

1:01:44

you can see uh that they receive an input for example x1 and x2 and so let

1:01:52

me just change something here that's important x1 and x2 x1 and x2

1:02:01

okay so um and and um the representation this is a

1:02:08

graphical representation of this unit and this unit this unit or neuron will

1:02:14

hold the value right this is this is done inside a computer so in the end of the day this is what what is happening

1:02:21

you have each each unit of this of a network in this case we don't have a

1:02:26

network yet but it will hold the value okay um and we will call this value an

1:02:33

activation if it's active or inactive okay so let's say uh the the first uh

1:02:40

depiction of a network is this one here uh on the left of sorry of a neuron is

1:02:48

here so you have an input another input you can see that the calculation that

1:02:55

the neuron is doing is performing is actually it's getting the first input x1

1:03:02

multiplying by w1 we will call a weight getting x2 multiplying by w2 which is a

1:03:11

second weight and summing up what we call a uh a bias term or you can think

1:03:18

about this bias term w0 kind kind of multiplying one multiplying no no input

1:03:24

is being multiplied but it's it's actually um being used by this round unit named a

1:03:33

neuron so the neuron the calculation that is that is a is is a neuron so

1:03:39

inside the computer a neuron is a neuron is actually doing this it's it's this

1:03:45

equation it takes uh x1 multiplies by w1 takes x2 multiplies by w2 takes w0 which

1:03:54

is a uh bias term and applies a decision or what we call a function g to

1:04:02

this sum okay to this linear sum of inputs and uh and and it will this will

1:04:10

hold a number this will have a result and this result the base it will uh this

1:04:17

result is a number that this neuron will hold so let's see what kind of learning

1:04:23

um um so a neural network is a learning algorithm it will learn the

1:04:29

W's that you have here so this is the model parameter and the way it learns is

1:04:35

through data okay and um historical data so let's say uh start

1:04:42

with actually this this one here okay this the this data set so this data set

1:04:48

tells you if your mom authorized you to play and your dad authorized you to play and if you if you should play or not

1:04:56

okay so it's if you if you think about that this is called the or data set so

1:05:02

mom mom did not authorize you dad did not authorize you so you shouldn't play

1:05:08

because nobody did authorize you um but mommy did not authorize you but if daddy

1:05:15

authorized you then you can play right so zero for mom one for dad you can play

1:05:22

one for mom zero for dad means that mom has authorized you so you can play and

1:05:27

if both authorized you then that's awesome both of you gave you gave permission and you should play okay so

1:05:35

how let's say let's see how one simple neuron could actually learn to

1:05:43

um this function here learn this learn from this historical data okay um and we

1:05:51

will do it by hand so that we understand the

1:06:02

calculation all right so what we have here is

1:06:10

um so I'm going to set this uh I'm going to set some weights for you so that you

1:06:16

can see that this works and I'm I'm kind of doing the learning process by hand but you will understand so let's say

1:06:23

that you did not authorize and dad did not authorize mom did not authorize dad

1:06:28

did not authorize so this is 0 * 1 this is 0 this is x20 * w22 uh 2

1:06:40

0 -1 so the number that the the number here is minus one okay and we actually

1:06:51

are applying a function on top of that what we call a decision function or an

1:06:57

activation function so if you think about this activation

1:07:02

function is usually called the step function and if it's minus

1:07:09

one actually it's the the value is here on it's way to the left and the result

1:07:15

the number that your um neuron will hold is zero okay so I actually I learned how

1:07:22

to do this and you can test all other options so let's say that at a point if

1:07:29

any one of the two dad dad or mom authorize you will have what changes

1:07:36

will be that one of this terms will be one and 1 minus one will give you a g of

1:07:43

0 and you can see that a g of 0 is actually it it is here and then you can

1:07:49

go play right you can do to the other side as well like x11 and x2

1:07:57

And if both of them authorize you can see if both of them

1:08:04

authorize you can see that this will be 1 + 1 here minus one so the result is g

1:08:14

of 1 which is way to the right right and then you can also play so you actually

1:08:22

we learned how to do to kind of um to replicate this function with historical

1:08:29

data and we learned this weights i mean I put it this weight uh because I I

1:08:34

already know that this weights uh solve the problem but we didn't uh we didn't

1:08:40

uh need to do it manually we could have trained the this this mathematical unit

1:08:47

okay you can also think about doing another function

1:08:53

right you could achieve another function for example this function so it would be

1:08:59

like this uh the x1 would be mom authorized you play and it's not

1:09:06

raining right so this is mom authorized mom

1:09:12

so you you can only play if your mom authorized it and it is not raining okay

1:09:20

um if your mom authorized but it's but it's raining you cannot play right so

1:09:26

you can also uh and and this will take time and I'm gagging a lot but this is

1:09:31

how we will do like the explanation you can also uh learn uh to do the end function

1:09:39

and the way the the neuron or this this unit would learn would be and so we

1:09:47

would delete this okay and this would be a square i have to change this image

1:09:52

this is a square so this is just for me my reference but anyways so in the end

1:09:58

of the day you would be using one one and minus2 as the bias if you do that

1:10:04

you can test if it's zero and zero this will be zero and minus two you cannot

1:10:09

play because you're extremely to the left here so that's this case if one of

1:10:15

them is one if one of them is one it will be 0

1:10:20

* 0 + 1 so 0

1:10:26

+ 1 - 2 is still minus one so you're still don't play and the only the only

1:10:35

uh way you this output will be one is if

1:10:41

you is if you oh jeez is if you have both of them

1:10:47

both of the inputs okay both of the inputs one so 1 * 1 is 2 minus 2 is zero

1:10:54

and then you activate the neuron okay so the value the final value that this neuron will hold is one meaning that it

1:11:03

will be only activated when both inputs is one okay so this is the basic

1:11:08

perceptron and I'm using uh very easy functions but you could use functions

1:11:14

with three inputs but in the end of the day this mathematical unit is

1:11:20

doing W1 uh uh it's it's doing weight times the input plus weight times the

1:11:26

input plus weight times the input plus a bias and um and applying a decision

1:11:34

function right um meaning that so the the if it's if

1:11:42

the value it holds is zero we say that this neuron is not activated if it holds

1:11:47

one it's activated so uh trying to um kind of mimic a a activated neuron in

1:11:54

our brain okay now we know that this

1:12:00

activation function the step function is the s is like the simplest one but we

1:12:06

don't we don't tend to use it we tend to use the sigmoid function we tend to use

1:12:12

like the sigmoid function which is this one that gives you kind of more more of

1:12:18

a probabilistic um classification right so and it it's

1:12:25

we we tend to use the sigmoid function um because it is differentiable and has

1:12:31

some mathematical reasons okay we have and we can use other activation functions as well like as relu and other

1:12:39

activation functions okay but this is the basic unit it's a

1:12:45

mathematical unit of the neural network now this neural network is all connected right it's all connected and it can

1:12:52

learn uh you can connect one uh unit so here are the

1:12:58

inputs right and you can you can kind of um connect I will I'll make this figures

1:13:06

better but you can connect uh one unit of a neuron in one layer to other uh uh

1:13:15

units this is called a multi-layer perceptron multi-layer

1:13:20

perceptron right um and the multi-layer perceptron is uh is very powerful

1:13:27

because this every every neuron from the previous layer is connected to one

1:13:34

neuron of the next layer so this neuron here is connected to the all previous ones and they the the the number that

1:13:42

that they are holding will activate or not this neuron so the the numbers that

1:13:47

are that the neurons are holding here they serve as inputs to the to the next

1:13:53

one okay um and if you think about

1:13:58

um the w the the the the bias term this w0 is responsible for

1:14:07

for kind of setting a strength strength of activation if you think about the or

1:14:14

when you have one one uh one one um it the the sum the number

1:14:20

sum is two so it's it's more activated okay um so it controls how how strong the

1:14:29

activation is um and the weights are kind of waiting

1:14:35

in the inputs okay so we're we're we're we're we're kind of

1:14:40

um doing this classification um and it also can be used for

1:14:46

regression okay um we we can um and then the multi-layer perceptrons can learn

1:14:53

way way way way more um difficult and more intricate

1:15:00

functions than just an end on an or function right um you

1:15:06

uh because the ender or or the or functions um with only with only one

1:15:12

perceptron is actually kind of you're you're kind of learning a a linear

1:15:18

decision um and uh you're just applying the

1:15:24

classification function to it right multi-layer perceptron is this important

1:15:31

like one way to understand neural networks is also working with images right so uh let's say that you have

1:15:39

handwritten digits and you have to classify them in the number that they represent for example 0 1 2 3 4 5 6 7 8

1:15:46

9 okay um you could use a logistic regression but it will be very difficult because

1:15:52

because the logistic regression takes the input and fits this kind of squiggle

1:16:00

uh to classify uh to classify obviously I'm showing a classification of only two

1:16:06

classes right A or B um but you can imagine that it's very it's very

1:16:12

difficult because you rely too much in the input um meaning that the input has to be

1:16:19

informative right so when you when you fit this uh this squiggle you um

1:16:27

uh as we fit with the with the arm length and the height of the person you

1:16:33

know that the the the arm length is kind of a informative feature right and and

1:16:39

then the fit if you have if you do a good fit your classifier will be a good classifier but look at uh the beauty of

1:16:47

neural network is that with their training step you can actually the

1:16:53

feature the importance of the feature and the feature selection manually selecting a fe feature is not that

1:17:00

necessary anymore um um that's the beauty of neural

1:17:05

networks um so think about uh think about number one say that this is a uh a

1:17:12

grid of pixels and somebody wrote number one here right so it has uh this pixel

1:17:19

this pixel this pixel this pixel and this pixel um uh like uh with values one

1:17:28

right and the other pixels are value zero uh if you were to consider the

1:17:35

inputs exactly this pixels this pixels in pink that I'm kind of highlighting um

1:17:43

and you could you could fit a squiggle and your classifier would be good right

1:17:48

but the thing is that different people write one in different ways and in

1:17:53

different uh spaces so for example I could have wrote one here right um And

1:18:01

and then if I if I'm selecting this features in pink I will actually say

1:18:07

that this is like I don't know what number is this because I I I'm not even

1:18:13

activating um the the neurons that learned the

1:18:19

neurons that needs to be activated to output number one so you have all the

1:18:26

pixels of your image these are the are the inputs they're connected to several

1:18:32

other uh neurons and in the end of the day you want a a a final neuron to spit

1:18:38

out number one right so there will be some u there will be patterns of of um

1:18:46

neurons that will be activated that will activate this class number one here okay

1:18:56

um oh sorry i think so uh a good example of

1:19:04

showing how deep learning is good with feature extraction and it helps elimin eliminate

1:19:10

the the need for feature extraction uh meaning that you don't rely on a specific set of

1:19:18

features um is for example in the in the past to do

1:19:25

X-ray diagnosis you inputed all this pixels from an X-ray uh to for example a

1:19:32

logistic regression or an SVM to diagnose pneumonia but the the actually

1:19:38

the doctors they uh uh not only the doctors but also feature extraction

1:19:45

algorithms they they uh they had to be kind of

1:19:50

you had to have this feature extraction step uh and uh they discovered and using

1:19:58

the domain experts as well uh like the doctors that the for for for example for

1:20:04

this region 4 a the angle between this this line and this line uh and the angle

1:20:12

of other lines and the gradient of the color gradient from other lines are important for diagnosing so given this

1:20:20

attributes right given this attributes the uh like the logistic regression the

1:20:27

SVM or other uh kind of uh machine learning classification algorithm will

1:20:32

come up with oh it it it has pneumonia the the person has pneumonia with the

1:20:37

advent of deep learning you you will actually you won't have to create this

1:20:43

features you won't have to do feature extraction and engineering and then use it for classification you will input the

1:20:51

pixels of this image the raw data the intensity the color intensities the

1:20:58

pixels of this image right um and the the deep uh neural network

1:21:05

will learn from these raw features right so this capacity for eliminating

1:21:14

um the uh the knowledge expert is is what makes neural networks and deep

1:21:21

learning so powerful um it it learns from this kind of all the features you

1:21:27

have and from raw features uh so there the

1:21:33

the there's this this image here

1:21:38

uh where I'm not sure where it is let me go back this video is super good but

1:21:47

um so so it uh before you had this feature extraction step uh with other

1:21:55

mainly uh with other um with other classification methods or

1:22:02

regression methods uh you can actually put all the all several features

1:22:10

um but but SVMs for example logistic regression and others they kind of get

1:22:16

confused if if you have so many columns um and doing feature selection feature

1:22:22

extraction um kind of selecting the most important ones then make their uh their e uh their

1:22:32

efficacy better but deep learning is this is this kind of very robust

1:22:39

algorithm that you don't care you just it will learn from the data uh and you

1:22:45

don't even need this step

1:22:50

okay as um let me go back a little bit we can have mult multi-layer perceptrons

1:22:57

can actually have multiclass classification so just not zero and one you can have several different classes

1:23:04

for uh classification for example here number zero number one number two number three um and usually the way it's done

1:23:12

is the um the neurons on the last layer right so if you go back here the neurons

1:23:20

on the last layer would be several like let's say that you want to classify

1:23:26

um number uh uh one two three right so

1:23:32

you would have three uh layers three output neurons and if this one gets

1:23:39

active it means it's number one if this one gets a active it means it's number two if this one gets a active it means

1:23:47

as number three so we can do um multi-layer classifications okay and

1:23:53

again I'm showing these examples at classifications but the you can also do

1:23:59

regression right you can think about the per the simple perceptron or the simple

1:24:05

neuron as doing is actually doing a a regression right so after you calculate

1:24:13

the sort of let's say that you have the arm length the biological sex and um how

1:24:21

how sporty you are in terms of u sport

1:24:26

interest and then you want to output the height well it's doing a linear

1:24:31

combination of this and you can actually apply um a a a function for example

1:24:41

um a linear function for example meaning that the the if this is actually you are

1:24:49

actually doing exactly a linear regression because whatever number you

1:24:54

have inside the parentheses if it's linear will will will be the output Right the linear function is whatever is

1:25:02

in it will be out okay so you can also use the perceptron to do regression and

1:25:12

multilayers as also doing regression okay um for regression because it's just

1:25:18

one number as the output you have just one uh neuron on the output in the

1:25:24

output but so but it will uh it will hold whatever value it is oh the height

1:25:30

is 180 cm okay but you can also do regression um so another way to uh to

1:25:39

kind of interpret deep learning um is and and here the term deep learning means exact exactly this multi-layer

1:25:47

perceptrons that has lots of layers in between the output and the input um so

1:25:53

that's the meaning of deep learning and the the beauty of deep learning is as I

1:25:58

said about the features extraction right so it's it comes in in automatically

1:26:05

um and you can learn without even thinking about selecting uh features

1:26:11

obviously you if you do that if you have the prior knowledge and do that that's best for your model but it it carries

1:26:18

out lots of information extraction uh on the go so I will just give you an

1:26:23

example uh this is a a figure that we have to to change but imagine that you had um a figure a 4x4 figure meaning you

1:26:32

have some pixels 16 pixels and you're actually inputting

1:26:38

um a uh a figure of a a um oh my god

1:26:43

this is of a uh of a bird right so in the end of the day you want a neuron to

1:26:51

be uh to be active that says "Oh this is a

1:26:58

bird." Okay uh this is a bird my god okay and

1:27:06

what what the neural network would be doing the learning process is doing is think about having a a input layer

1:27:14

with 16 pixels right for the for the 4x4 picture and the first layer of this

1:27:23

would be would be an edge detector right so um let's say that you have um a

1:27:30

neuron that uh uh that identifies little lines right so let's say that this

1:27:36

neuron is active when it identifies a little line uh let's say it identified

1:27:42

this line so um um the the the the in

1:27:48

the input um the combination of this inputs and

1:27:54

say like the the the figure is one so it's a black and white figure the background is is black and the

1:28:01

foreground is is white so the the the birdie has a intensity of one and the uh

1:28:09

the background has an intensity of zero so this is an edge detector uh it detected one one edge and

1:28:17

this one for example detected another edge here right the following layer you

1:28:24

another neuron can be activated when um if it found two edges and the angle is

1:28:33

uh it's a it's a it's a a low angle between them right you will activate

1:28:40

this this neuron which is a beak detector right and and uh and after the

1:28:47

beak you could have neurons that detect his that detects eyes that detects

1:28:52

um that detect uh feathers and wings and

1:28:58

in the end of the day if if if you have a if if you have a pattern of neurons

1:29:06

that means beak eyes feathers uh it will output a neuron named bird okay and and

1:29:15

the way you kind of uh set up this the the first edge detector and then the

1:29:20

beak detector and then the eye detector and so on is that for example

1:29:28

um u um I'll try to do this the if you if you think about the pixel here right

1:29:36

um you can think about having this pixel being at a neuron um this neuron here

1:29:42

which is the the edge detector vector it will in in it will uh it will

1:29:48

activate when it sees um when it sees one in this particular

1:29:56

uh pixel so it has a weight of one and one and in the neighboring pixels

1:30:04

uh when they are not activate they're not activated right so the ne when the

1:30:10

neighboring pixels are zero okay when the neighboring pixels are

1:30:16

zero and this particular pixel is one you will be

1:30:22

activating the beak detector for example in this position and you can have

1:30:27

um for each position of the image you could have this uh edge detector and now

1:30:35

you can have this for example being active when two edges are um are active

1:30:41

right so then this is a big detector and so on so what you're doing is you are

1:30:48

adjusting you are adjusting this weights uh to to detect your um your class or

1:30:58

your regression okay and um this this

1:31:03

could be billions of parameters and again deep learning means this multi-layer perceptrons that are that

1:31:11

are have several parameters meaning that all these weights of connectivity

1:31:16

are are found in a learning process okay so again you have all the weights you

1:31:24

have the weight between this neuron and the other neuron so between neuron one

1:31:31

and one from layer 1 to layer two this one is um the connectivity between the

1:31:37

the weight that this uh neuron one uh with neuron 2 of of of the layer so it's

1:31:46

very intricate the the name of the algorithm of the learning uh of the

1:31:52

learning process is named uh

1:31:57

um so um I changed here for a little movie a little video uh but the

1:32:09

the what what the gradient descent mean means is that it's the optimization

1:32:16

algorithm used to uh to train the deep

1:32:24

uh the deep learning models right so this optimization

1:32:30

um this optimization algorithm called gradient descent iteratively adjusts the

1:32:36

model's parameters to minimize uh to minimize the errors okay um

1:32:46

so let's say that for the birdie example you start by uh

1:32:53

um by setting up random weights to that billion parameters right

1:32:59

um and you

1:33:05

um um so obviously if you run this

1:33:10

neural network it will it will output whatever it is like if uh let's say that

1:33:17

this neural network would should uh output one for birdie and it outputed 03

1:33:23

meaning it's not classifying correctly right because you're you're you're

1:33:28

you're showing a birdie in the input so it's it's kind of classifying poorly and

1:33:34

you have this error okay you have this uh this error uh that

1:33:40

is 7 okay from one to.3 so what you're what what you're

1:33:48

doing is you iteratively tweak or you

1:33:53

kind of turn the knobs of each of that weights to minimize the error right

1:33:59

which is our cost loss or cost function that we already discussed in the case of

1:34:04

this uh kind of classification um we could use the mean

1:34:10

squared error right as well so it's it's it would it would be 07 squared that

1:34:16

will be the the error that you have now after kind of tweaking um tweaking the

1:34:24

weights uh a first time your error should in theory and and it will uh

1:34:32

decrease okay and then so your cost function is at each training step is

1:34:39

minimized and we're going to define the training step or epoch uh a little bit

1:34:45

further on okay so I I like to show this image here so if you think about that

1:34:54

um think about the the weights right uh

1:35:00

the weights here I'm showing just one weight right but imagine that uh W is uh

1:35:08

like the aggregation of all the weights of a matrix of all the parameters of a

1:35:14

neural network and at each time at each learning step uh your loss function gets

1:35:24

uh smaller and smaller up until a point where you will reach a minimum okay so I

1:35:30

always say that this is like a hiker um trying to find the lowest point right uh

1:35:38

or you can compare that um with

1:35:45

the a ball a ball like going downhill okay uh I'll show you another image

1:35:54

here so the the the algorithm is

1:36:03

called gradient descent right because the gradient is a mathematical entity that

1:36:11

shows um the the that that shows the direction

1:36:18

right uh of the steepest downhill right so so the the the

1:36:27

algorithm kinds of tweaks kinds of adjusts the model parameters in a

1:36:33

optimal way in in a way that it's making

1:36:39

the error decrease right in uh in the

1:36:44

fastest way okay so AC for each step for each

1:36:50

learning step um you present your data set your learning set and you adjust the

1:36:58

weights and then your error will decrease then you then you show again

1:37:05

the your data set and you adjust again right so at each step the the algorithm

1:37:12

the gradient descent algorithms calculates this gradient calculates this mathematical entity that shows the

1:37:20

adjustments needed for this parameters to minimize the error in this in the

1:37:29

fastest way and this process continues until the mo the model reaches this

1:37:34

minimum or it stops right so it's like a ball rolling uh down a

1:37:41

mountain and it will uh find the minimum of the loss function sometimes you you

1:37:48

will you some neur when you train a neural network sometimes you will get a very good uh fit for the neural network

1:37:56

sometimes you'll stop at what we call local minimum and you have to maybe start in another start the ball into

1:38:03

another point where the ball will kind of find uh like if you start the ball

1:38:08

here um like if you randomly start here with your weights your weights will be

1:38:15

updated and we'll find a global minimum so that's why during training of neural networks we train it several times okay

1:38:23

we train it several times starting with random weights in different like

1:38:29

starting with different random weights because if you started here you will

1:38:34

uh probably be uh find a local minima um because you found when you found a

1:38:41

minima when you found when you find a minimum you stop the algorithm but if you start here you will find the global

1:38:49

minimum okay and the one of The

1:38:59

um one of the implementations of this uh

1:39:04

of this algorithm is called back propagation and that's how we implement

1:39:11

this algorithm in neural networks meaning that um if you go back here to

1:39:21

let me find the neural network

1:39:26

um when you find so here you go when you so this is

1:39:34

a a network a neural network um when when you run the neural network the

1:39:40

first time with random weights all of this connection s you will you will have

1:39:46

an error then you you will calculate the

1:39:51

gradient and then you start by updating by tweaking this the weights of the last

1:39:58

layer and after that the weights of the previous layer and after that the

1:40:04

weights of the other layer and the other layer and the other layer so that's what

1:40:10

we call back propagating so we are minimizing the error in a optimal way

1:40:16

because we're using a a gradient descent where the gradient is telling us how to tweak this weights to optimally minimize

1:40:25

the error and you and you get you propagate uh this twix you you kind of

1:40:32

do different um adjustments in a back propagation way so you start by tweaking

1:40:38

this then that then that and then that then that then that then that and then you run the second

1:40:45

step you run again the network you will have an error this error is going to be

1:40:50

used for the the second learning step and then you back propagate again back prop back prop backrop back prop backrop

1:40:57

back prop this is the second adjustment of the network then you then you show

1:41:02

for example a cat or a bird sorry and the error is not.3 anymore it's not what

1:41:10

7 anymore it's kind of you you're you're closer to one

1:41:17

so you got you get again uh you you show the cat you will get now for example 6

1:41:24

so you have a 0 2 error uh so you still have an error you calculate the gradient

1:41:30

which will show you that to minimize this error you have to tweak this weights uh in the the fastest way and um

1:41:39

you back prop kind of adjusting all of these layers up until you find the

1:41:45

minimal error right um meaning that you can find

1:41:55

uh you find the minimum error at some sometimes the error will if you keep

1:42:01

tweaking the the uh the weights the arrow will um will go up right uh so you

1:42:09

want to stop when you achieve a minimum so you keep calculating the error if the

1:42:14

if you're here and the error kind of tends to go up you halt the algorithm

1:42:20

you stop the algorithm so you can stay in the minimum okay

1:42:26

um um so we talked about back propagation gradient descent uh but all

1:42:33

that I've talked here was about multi-layer perceptron and what we call the feed forward architecture mean in

1:42:41

the feed forward forward architecture we have neural networks

1:42:46

that are connected that the information goes from the input to the output let me show you

1:42:55

this um this is a great video that I'm pointing out for more resources but this

1:43:02

is the feed forward uh kind of connectivity the the input comes in and

1:43:08

the neurons are activated like the the neuron in the next layer is activated up

1:43:14

until the uh the output but there are so many

1:43:19

architectures different architectures right so uh I'm just going to to kind

1:43:25

of talk about one uh couple of them uh

1:43:30

and you can kind of go and and and search for more resources uh one of

1:43:36

these neural networks are called convolutional neural networks and it's used for uh image processing

1:43:42

classification with images this this is the architecture that is very powerful

1:43:48

for image processing okay so the that vanilla model that we talked about uh

1:43:55

the multi-layer perceptron finding a bird um that is like uh quote unquote

1:44:01

old technology they're much better architectures um the other

1:44:09

architecture is this one is the recurrent neural network so you can see

1:44:15

that the activation state of one of a neuron ahead

1:44:21

of let's say that I am this neuron the activation of this neuron ahead of me

1:44:27

will actually change my activation right so um when the when the second so so

1:44:36

this these kinds of networks are very good to um to be used with forecasting

1:44:43

time series right so forecasting um uh things that happens in time so the

1:44:50

next steps why is that well because for example let's say that the second

1:44:56

uh uh point in time so let's say that you have values of the stock market

1:45:02

okay and the first first value comes in

1:45:08

and you want to predict the second value you're learning right you're learning you have a data set and you're you you

1:45:15

want to predict um so let me draw this I think it's it's going to be better

1:45:22

um say that you have this data set of the stock market where you have a point

1:45:30

of the stock market value and you want to uh forecast the second value from the

1:45:37

second value you want to forecast the third value

1:45:42

um and so on from the third value this is a two you want to forecast a fourth

1:45:48

so this is my my target for my supervised learning

1:45:55

and this is the attribute right so this is my input

1:46:00

so what happens is that if we go back to that

1:46:06

video when when when you when let's say your first input comes in you want to

1:46:13

you're learning so you will have um you will learn how to predict X2

1:46:20

okay now when X2 comes in the the value of this

1:46:27

neuron before when X1 was the input will kind of change the activation of this

1:46:35

guy so this is what we call systems with memory the this this connection here is

1:46:42

kind of um it's a temporal connection meaning that the state of of this neuron

1:46:50

when the input was one will actually change the state of this neuron when the

1:46:56

second input comes in so uh you actually you can this weight here is memorizing

1:47:04

stuff so it's very good it's very good uh architecture to be used uh with

1:47:12

forecasting time series okay like stock markets and so on and that's it for

1:47:21

um for uh for neural networks several good

1:47:27

resources uh here it's very hard to kind of go in a shallow way Uh but I think

1:47:36

that with all the extra resources and obviously I will have to uh to

1:47:42

streamline this um script and everything

1:47:47

uh we will make to continue now you now we saw some models uh I want to talk a

1:47:53

little bit more about I want to talk a little bit more

1:47:58

about um model complexity okay so the model complexity refers how

1:48:04

sophisticated a machine learning model is in terms of its ability to capture

1:48:10

patterns in the data right to a more complex model has more parameters and

1:48:15

can learn more complicated relationship like a neural network with many layers

1:48:20

that has billion parameters and a simple model like a linear regression that has

1:48:27

uh that has uh uh less parameters so

1:48:32

finding the right level of complexity is crucial um and a two simple algorithm like this

1:48:40

is a it's a it's very simple and for example here it's it's it's an algorithm that it's um you you

1:48:48

want to regress for example this point so you did you do a a regression

1:48:54

um this could be a quadratic uh polomial so you're not fitting a

1:49:01

linear regression you're fitting a polomial regression you can see that it's better into into

1:49:08

capturing uh the the the model the the points the the training data and this

1:49:15

one is very very very very complex it could be a polomial uh like a high order

1:49:20

polomial and you you can see that it's fitting totally the data okay

1:49:26

now you could say oh this model is better because it's fitting the data perfectly and that is very import

1:49:34

concept very important because this is not right right um if if the if the

1:49:41

model is too simple it fails in capture the important pattern right so you can

1:49:47

see that this pattern is more quadratic and if it's too complex it will learn to fit all the training data

1:49:56

and usually noise um and this is and it will also poorly capture capture uh the

1:50:04

the pattern and this is called overfitting okay so um let's discuss a little bit

1:50:12

more about overfitting and what's noise so if you think about

1:50:17

a let's see if I have so if you think about a cinosoid let's say that this is

1:50:22

something that you captured in a lab uh about with a with a with a with a with

1:50:29

sensors so this is data and the you can

1:50:35

see that like it is a cinosoid but and this is like the the the trend of the data is a

1:50:42

cinosoid but because of errors in the sensors you know the data is

1:50:48

spread you have measurement errors you can have um um sensors errors and and

1:50:56

and what your model should do is to ignore this this spreadness of the data

1:51:03

we call this kind of there's some noise in the data if you were to have a very complex

1:51:10

model that fits the data totally it would be a model that is learning this

1:51:16

kind this kind of pattern and this is not at all a sine

1:51:22

wave right this is not at all a sine wave so if you're reg if you're doing a

1:51:27

regression and then a new uh a new value of act of the input comes in and you

1:51:34

should output a value of a sign this this green curve will never do

1:51:39

it okay so this is probably this is overfitting overfitting

1:51:47

the data and if the data is actually a quadratic like this when a new point

1:51:53

comes in right when a new point comes in you will get it wrong you will get it

1:51:59

wrong although you've perfectly fit the training data you're not you're not having um

1:52:07

generaliz what we call a generalization power okay so

1:52:14

um we we we need a model that it's complex enough that actually doesn't

1:52:21

learn the noise right of the um of of the data and actually

1:52:29

captures everything uh and and that not captures everything it actually captures

1:52:35

the trend of the curve or the trend of the decision boundary the curve of whatever

1:52:42

you're doing okay so this is this is the discussion around

1:52:48

overfitting and having this balance is so very important okay um so um the the

1:52:57

optimal solution will always be um where your

1:53:04

um this this this figure is not informative uh I'll I'll do something

1:53:09

here so when when when you're training um when you're when you're training your

1:53:16

uh your model and you have your loss function and you want to minimize the

1:53:22

loss function for example um what what you need to do is this you

1:53:29

you you you keep training and your loss function for example your mean squared error in a linear regression or in a

1:53:36

regression um or a quadratic regression polomial regression whatever is

1:53:41

decreasing right so this is the the loss in the training data and you want it as

1:53:48

small as you can but if your model is too complex it will be so so small that

1:53:53

you're actually fitting the noise so Um we what we want is that your

1:54:02

uh your your if you if you uh think about

1:54:07

measuring the mean squared er error on on the

1:54:12

um on the the test data you want to this

1:54:18

is the test data so if new data comes in we we discussed about the test data and

1:54:24

the division of the the data set you hold out part of your data set to test your model but you want that to after

1:54:33

training you want the the the loss for the training data to be small

1:54:39

okay um what happens when your model overfits

1:54:44

is that actually you're learning the noise so the loss uh the the if you're

1:54:52

testing in a test data at some point if you keep learning you're you're learning

1:54:58

too much it's like a student that is memorizing not understanding the data

1:55:04

all right so the the loss in the test data starts to starts to uh actually increase

1:55:12

meaning that your generalization power is lost okay so one of the mechanism to

1:55:21

avoid overfitting is actually splitting the data the into three three parts

1:55:29

training validation and test so you have a hold out called the test data okay and

1:55:36

you you keep that uh to to test several models and to choose the best one right

1:55:42

uh giving a metric now what you do while you're training you keep assessing how

1:55:49

your your loss is u minimizing and you you keep a a uh you

1:55:56

you hold out a a a part of the data set called the validation data set so this

1:56:03

data set you will apply your model to the data set and you will see well you know

1:56:11

it's kind of a test set but because you use it to stop the training

1:56:17

um it's it's it's not an unseen data right so remember you always have to

1:56:23

test your model into this okay this test the unseen data this although it's not

1:56:32

used for training validation is not used for training it's used to it's used

1:56:38

uh to to kind of guide the training okay kind to guide the training

1:56:45

so it uh it it it will actually uh weight in the the the values that you're

1:56:53

choosing for your parameters okay so uh the in the validation set the the loss

1:57:03

functions as as you as you train so this is the train

1:57:10

time or we will we will discuss the epochs in in in the in terms of neural

1:57:17

networks uh I'll I'll I'll bring this concept just in a bit but the validation error

1:57:25

drops drops drops drops drops and as soon as starts to get up right because

1:57:31

you're starting to overfit you halt the training you stop the training here so

1:57:37

you stop the training right so that when you have the

1:57:44

test set when you have the let Let me choose another color when you have the

1:57:50

test set u in the test set because you stopped when it was starting to kind of

1:57:56

get um it's start it's starting to overfit it will still be uh it will still be a

1:58:05

good model so because you're stopping the training you're using this data set

1:58:10

kind of to halt the the values of the weights here of the parameters of your

1:58:16

model here it's uh this this validation although not using to training it is

1:58:22

kind of used to get your final model so you still have to use an completely

1:58:27

unseen uh model uh data for the test okay so let me introduce the concept of

1:58:35

epochs uh while we're training neural networks we present the data for neural

1:58:42

networks several times and then adjust the weights via back propagation um and

1:58:51

each time we show the data so each this each time we show the data this is epoch

1:58:57

one i have a loss then I adjust my weights of all the um of the

1:59:05

um of the layers then I show the data again and I adjust my weights again so

1:59:11

every little twick that I do in my weights during training is called an

1:59:18

epoch okay now validation is a very common

1:59:24

this split here during training is very common training validation and test and

1:59:30

actually a a um there is a algorithm

1:59:38

uh called uh cross validation that uh that um kind of gets this concept a

1:59:48

little bit better uh and it it um it does it uh to kind of um decrease

1:59:58

variability so let's say that you've divided your training set your validation set and your test set and you

2:00:05

had a result of a loss right so let's say that your loss um after training uh so

2:00:13

uh uh sorry that you have a metric evaluation metric of

2:00:19

98% of of goodness right your model is 98% good we will discuss measures of

2:00:26

goodness in a bit uh and then you if you change this uh randomly um

2:00:33

um the samples that are inside each of these groups groups you will have a 96%

2:00:39

and then you will have a 94% and if you keep doing you will have a 99% so in

2:00:45

average right with different holdouts with different um um kind of divisions

2:00:52

you will have a um a metric that you can use

2:00:57

right about something named cross validation uh and and the name

2:01:03

is it should it should be named cross testing um you can use the validation

2:01:10

set during cross validation but not necessarily so when we do training when

2:01:17

we do the division between uh training and testing uh and and actually

2:01:25

you want to do more training data the more the

2:01:31

better you have two problems the first problem is that um you're not using the

2:01:37

entire data set to train so you're holding out to this test and this will never use used it will never be used for

2:01:45

training so this is kind of a a bad thing and also you you have this

2:01:55

um [Music] um a tradeoff like if you if you uh want

2:02:04

to have more training uh data you will have the test set small okay so the way

2:02:11

people do that is uh by using what we

2:02:16

call cross validation again uh it should I don't know why but people use cross

2:02:21

validate it's called cross validation but it has nothing to do with the validation set per se it is like a

2:02:28

cross- testing okay um you can you can use the validation

2:02:34

set and we'll talk about that just in in some uh in some seconds um so the way it

2:02:40

works is uh you divide your data set

2:02:47

into pieces and you let me do uh a smaller

2:02:55

example one two three four five and you will you will use four

2:03:03

of this um of this data set smaller data

2:03:08

sets to train and you will test here so you will do one training

2:03:15

uh experiment here so you're going to train a neural network uh in this uh

2:03:21

training and test so this is one training and test then you will what you're going to

2:03:28

do is you're going to change the the the division so you're

2:03:36

going to test here and you're going to use other divisions the other four

2:03:41

divisions and you keep doing that okay you will you will keep doing that up

2:03:49

until the point where you use all the data for training so uh let's say that

2:03:57

now you will use this one this one is the test so you're going to use four

2:04:03

um four like four uh um splits of

2:04:09

training and then you're testing on this one and uh as as well as uh let's say

2:04:17

now you're going to train this you're going to test here right so in the end of the day what

2:04:24

you're going to have is going to test here uh you're going to

2:04:30

have all you can see that all all the data it uh is used for

2:04:38

training so you're testing on all the on all the data as

2:04:44

well so you're testing in all the data and you're doing several trainings

2:04:51

trainings and testing right so you're doing k what we call kffold cross

2:04:57

validation in this case 1 2 3 4 5 so you're doing a five-fold cross

2:05:03

validation meaning that you split your data into five buckets and in the end of

2:05:09

the day if you if you think about it this is t training used for training used for training used for training you

2:05:16

you used every every every every point in the data set for training and also

2:05:24

you you tested your algorithm um in all data in all data points right

2:05:31

so you tested here here here here and also in the last one

2:05:37

so in the end of the day you kind of you'll have five trainings and testing

2:05:43

steps and you will average you'll average that so this solves the problem

2:05:48

of having um uh of this of this challenges here and

2:05:56

um again you don't need to have the validation set but you can you can

2:06:01

actually uh have one of this buckets for validation

2:06:07

um in each training and test uh kind of experiment uh but again you don't need

2:06:14

to right you um um you don't need to have the validation test the validation

2:06:21

set is used for um kind of what we call the early stopping you stop the training

2:06:29

for not overfitting but you don't need to it's not necessary it's good but it's not necessary and cross validation can

2:06:36

be used nested with the with this with the validation set or not okay but this

2:06:43

solves the problem there's an awesome explanation here that we can use as well for the script and get a better figure

2:06:50

for this as well so let me just give you an example let's say that I have 200

2:06:55

data points uh data uh samples of my my data set and I will divide in five folds

2:07:02

so um uh you can see that each of these

2:07:09

data sets will have uh 40 uh 40

2:07:14

u u data points right um

2:07:21

and you will use so in the first testing you'll you'll use 40 data points for

2:07:27

test but you're actually using this one and and 40 data points for test here and

2:07:35

in the end of the day you're using the 200 data points for testing and also the 200 data points for for training and

2:07:43

then you're averaging this uh different um performances let's say that you test

2:07:50

and you have 98% 95% 99%

2:07:55

89% 71% 1 2 3 4 5 and you average so

2:08:01

this gives you a better uh you average this this will give you a better metric

2:08:08

of your classifier and you won't rely only on a certain split maybe you you

2:08:14

you're not lucky and you split your data in kind of a you know some kind of split

2:08:20

that your model is not that good um so that's why cross validation is

2:08:27

used okay so we're talking about goodness um goodness of of

2:08:33

um of a model and and metrics so let's talk a little bit more about that to

2:08:40

finish the the lecture for classification models we we use this

2:08:45

metrics here all of these metrics are um are important and we will talk about

2:08:51

that and for regression um there are other also other metrics so R R 2 mean

2:09:01

squared error we already talked about a little bit about that um so let's go and

2:09:06

and kind of um talk a little bit more about what the

2:09:12

what evaluation metrics come up in um in this

2:09:20

um in the when you test your models and you can test

2:09:27

different models and you can choose the best one in this test data set uh

2:09:34

regression we saw already the mean squared error i'm not going to uh kind

2:09:39

of talk about that again just back up the video a little bit and you will find

2:09:46

the definition of the mean square error the mean square error is uh it can

2:09:53

be used as the loss function during training and also as an evaluation metric after after training the model in

2:10:01

the test set all right so for regression um so you have your target numerical

2:10:09

target and here here's your attribute um you are fitting a line right the

2:10:17

attribute and the target you have your training points um during training you want to

2:10:25

minimize these distances um squared of this

2:10:30

distances but you can also like this is the training data set but you can also

2:10:36

use this as an evaluation metric when a new point comes in this new point here

2:10:42

this is a unseen sample and uh uh in the test set so you

2:10:51

know the t you know the target what the target would be um and you get you get

2:10:59

the value the prediction from the line but it should be here right and um you

2:11:06

can also use this squared distance for the um the metric of evaluation so the

2:11:14

bad the the smallest the mean squared error of your model in your test set

2:11:23

uh and obviously your test set has several points um the better okay now I'm going to talk

2:11:30

about R squar R 2 and only those two for for uh for u regression

2:11:38

this is a very good resource um this images is from this movie uh let's say

2:11:45

that you're trying to predict the mouse weight from the mouse height so you're trying to fit this line

2:11:52

say you fit this line and you will have um a

2:11:58

calculation uh for those who who is uh for those who are uh more familiar to

2:12:05

statistics uh can understand this calculation by um

2:12:11

calculating the variance of the mean and the variance of the the fitted line but

2:12:16

there's a formula that will give your the a value a metric right so for

2:12:21

example in this this line we we were able to fit

2:12:30

um the the R squar metric is 81% what the R squar means is that there's

2:12:39

81% okay that um uh 81% of the variation

2:12:46

around the line right so this this this variation around the line is

2:12:53

explained by the mouse size so 81% of this variation

2:12:59

um so um a a bigger R squared is better so for

2:13:08

example look look at trying to uh determine the mouse weight from the time

2:13:14

spent sniffing a rock right if you think about that this you you will fit a line

2:13:22

here um and this this model here has a R

2:13:29

squared of 6% so you can say that this niffing

2:13:38

uh weight relationship just accounts for 6% of of

2:13:43

of the trend of the data so you see that actually the time spent sniffing a rock

2:13:49

doesn't tell you anything about the the the mouse weight right so the largest time sniffing a rock

2:13:57

um it's it's not it doesn't have a clear relationship as this one right so that

2:14:02

the the largest the mouse the more heavier it is

2:14:09

right so this is R 2 the bigger the R

2:14:14

squar in our model the better we can go through

2:14:19

this formulas in this video they they go through what is the variation of the

2:14:24

mean the variation of the line and how to calculate we can leave this maybe as a resource

2:14:31

okay okay and uh now we're going to talk about accuracy oops sorry uh

2:14:37

accuracy um and F1 score and a uh ROC AU

2:14:45

um both of these are uh important uh and uh I will leave some resources

2:14:53

um but the I will cover this once okay so

2:14:59

when you're doing classification we're talking about classification now you can Think about

2:15:09

uh let's say that you have a disease so you have sick

2:15:14

people and you have healthy people so the sick people I'm calling the positive

2:15:20

class right so I'm classifying it as positive or one and healthy people as

2:15:28

zero okay and here's the prediction if you corrected if if you uh

2:15:36

um in your prediction the samples that you classified in the test set as positive

2:15:44

and they are actually positive are called true positive if

2:15:51

you predicted a positive or being sick but actually they're healthy is called a

2:15:59

false positive right so let's say that you predicted that you have COVID 19 but

2:16:04

you're actually healthy so you're a false false positive if you predicted a negative

2:16:13

uh a zero meaning that you predicted that the person is healthy

2:16:20

um and that person is indeed healthy this is a true negative right so uh you

2:16:28

predicted that you don't have COVID 19 and you are healthy so you're a true negative and finally if you predicted

2:16:36

negative and you're actually sick then you have a false negative

2:16:42

right so um we can def this is what we call um

2:16:50

the outcomes of your classification okay so in statistics this is known as type

2:16:58

one error or type two error so the false positives are that false alarm or we

2:17:04

call it type one you're doing a type one error or when you're false negative we

2:17:10

call it a false two error okay so this is just nomenclature that comes from

2:17:15

statistics okay so let's define the first metric which is accuracy accuracy

2:17:23

is this formula here but in the end of the day it means the the correct all the

2:17:29

number of correct predictions you made predictions over all the

2:17:36

guesses okay so let's say here what is the corrected predictions well the

2:17:43

truths what the truths right so the true positive meaning that you corrected classified uh it is positive and you

2:17:50

correct it's classified as positive and the true negative this is are are the

2:17:55

the correct predictions that I have and divided by all the guesses right because

2:18:02

the you're summing up all the possible results so this is

2:18:08

accuracy so having a high accuracy is good but it has some problems for

2:18:15

example um let's say that you have a very rare disease and only one people in 100,000

2:18:23

have the disease you could have a great classifier

2:18:29

um with high accuracy if that classifier just outputs everyone as

2:18:36

healthy right so you're you're you're actually having a very very high um

2:18:43

classifica high accuracy um but you're missing the sick cases

2:18:51

right so this is one problem okay we

2:19:00

um [Music] we this I won't talk about the rates

2:19:06

here but anyways um the other score that

2:19:12

kinds of prevents it's a score that takes uh takes into

2:19:19

consideration this um um this balance right of having a

2:19:26

high accuracy but kind of missing missing up important

2:19:32

classifications it's the F1 score and it has a formula with there's some

2:19:38

definitions here of precision and recall um I leave I will leave it as a resource

2:19:44

so that you can understand what is precision and recall but in the end of the day you can calculate the formula

2:19:50

just using the uh what we call the confusion matrix i forgot to say that this is a confusion matrix

2:19:58

and um F1 is is the the it's a complete it's a more complete way to ask

2:20:07

uh the question of whether your like your your classifier has a good

2:20:14

accuracy and also kind of um

2:20:19

um trades off the the the false the

2:20:24

false negatives for example right and the false positives so in a very rare

2:20:31

disease what you're doing if you're classifying everyone as healthy is

2:20:39

that you're you're actually uh you have false negatives and and you

2:20:46

can't afford to have right you need to identify this so you want to kind of put

2:20:52

a weight in the false negatives um there are other cases where

2:20:59

the false positives are kind of uh diminished and this this um ma this metric gives

2:21:08

you a good insight of that it's very good to have a visualization

2:21:14

so we know that um uh let's say that you have a classifier

2:21:19

that the positive class is one the negative class is zero and you have a threshold in the middle meaning that the

2:21:26

output of your classifi classifier if it's bigger than.5 you're classifying as one if it's

2:21:34

smaller than uh 0.5 you're classifying as zero so that's the threshold it's the

2:21:40

decision threshold right um so

2:21:47

uh we talked about threshold before so you can go back into the sessions where we talk about threshold but if you if

2:21:54

you put this in the middle you will have a percentage that that are positive that are blue but they're kind of getting

2:22:01

into negative right so this is this is this are false negatives and this are

2:22:08

false positives because they are yellow they are negative but they're being classified as positive

2:22:15

so you can change the threshold of your classifier and this

2:22:23

classifier for example if the threshold is completely here right so uh or not here but if it's

2:22:32

completely here here for

2:22:38

example it means that you are classifying everyone as negative

2:22:45

Right so this is the case of um you're

2:22:51

missing the the false negatives right so the

2:22:56

threshold can go uh up and down the I mean I'm explaining this in a very loose

2:23:02

way but the this resource will be very important right because it it has a vid

2:23:08

an animation of the threshold kind of going up and down

2:23:15

finally we have what we we um it's it's

2:23:21

another very important um metric which is the AU the area under

2:23:28

the rock curve the the ROC or the rock curve the um the rock is the receiver

2:23:35

operating curve so the last concept is this uh I brought back the true positive

2:23:42

rate here and false positive rate the true positive rate means

2:23:48

um the the the true positives that you got in your prediction over all

2:23:56

positives right um because you have true positives false negatives are also

2:24:02

positives right like if you if you're a false um

2:24:09

um um this is the the the true positive rate because the the you you classified

2:24:17

the true positives as positives and you classified the um the the the negative

2:24:27

the false negatives as negatives but they are they're they are positives in the end so so this is the true positives

2:24:36

prediction the prediction of true positives that you had uh over all the

2:24:41

the the positives um that exist in your data

2:24:47

set so we want this rate to be high right um and we

2:24:54

also want the false positive rates to be low right so what is this this is the

2:25:04

false positive predictions over

2:25:13

um the um all the negatives all the

2:25:21

negatives okay and so we want a classifier that has this one high this

2:25:28

one low um again I I think I can get this explanation better by having a good

2:25:35

image i couldn't find it and I kind of running out of time so I have to make this better but anyways the the rock

2:25:42

curve is a curve that that shows this this this rate of false positive and

2:25:49

true positive right what we would like is a false positive rate of zero meaning

2:25:57

that you don't have any false positives and a

2:26:03

um true positive rate of one meaning that you you you

2:26:10

predicted your positives your true positives are actually all the positives

2:26:16

right because then um you're you're getting everything right so if you think about that

2:26:22

um we want a classifier um that at some threshold right using

2:26:30

some threshold of the decision making um and again go back to thresholding uh

2:26:38

this is actually this is achieved at some point of the thresholding okay so

2:26:45

the rock curve will will trace the will

2:26:50

will trace different thresholds for the classifier and and uh and we'll plot

2:26:59

right so for example with the with this particular with a particular threshold here you have a false positive rate and

2:27:05

a true positive rate and what you want is to be close to this pink curve right

2:27:13

um and the way you you do that is you can see that the area under this curve

2:27:20

here is the maximum the pink one so the the the greater the area under this curve

2:27:28

the best okay um so if you have two

2:27:35

classifiers if you have two classifiers and one has this curve and the other has

2:27:41

the other curve this area here is less than this area here so the best

2:27:47

classifier is the blue one and you can kind of pick it

2:27:52

okay so area under the curve is also a very good metric that balances the the

2:27:58

the false positive the false negative kind of um challenge of certain classifiers okay

2:28:10

um and finally I just want to say that during

2:28:16

training engineers AI engineers machine learning engineers can also penalize

2:28:22

um during the learning can penalize um having uh like like false negative

2:28:32

false negatives right for example in a in a rare disease classification

2:28:37

so you can you can use the metrics the evaluation metrics to pick best uh

2:28:44

models and you um so you're comparing between models but there are tweaks

2:28:51

during training that can also make um

2:28:56

prevent this kinds of classification challenges but that's out of scope uh so

2:29:04

this is it for week two uh for the prototyping uh what I think is that we

2:29:10

had a 15-minute video of introduction of Wacka wacka is a is a um a software that

2:29:17

they will use to do the prototype or anime uh they can choose right they will

2:29:22

do the same prototype or using this or that and a five minute video of showing

2:29:28

a uh a framework or a software uh named thinkable machine this is just for

2:29:35

um this is just kind of showing a classification on the go and these two

2:29:41

they will use to do the prototypes and then give the three cases

2:29:48

for them like a little little cases ask them to solve and to and to kind of use

2:29:54

different classifiers for example different uh regression models i'm not

2:30:01

sure how to assess this prototypes but we can discuss that later