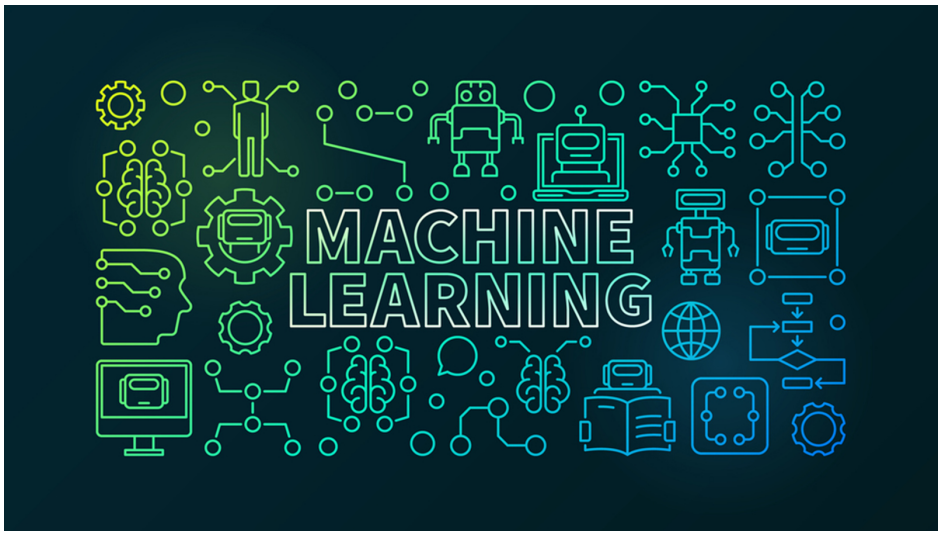
Statistical Methods for Machine Learning

UNIVERSITÀ
DEGLI STUDI
DI MILANO

Data Science and Economics
Università degli Studi di Milano

Andrea Ierardi

Lecture notes



Contents

List of Figures

Lecture 1 - 09-03-2020

1.1 Introduction

This is time for all good men to come to the aid of their party!

MACHINE LEARNING In this course we look at the principle behind design of Machine learning. Not just coding but have an idea of algorithm that can work with the data. We have to fix a mathematical framework: some statistic and mathematics. Work on ML on a higher level ML is data inference: make prediction about the future using data about the past Clustering —> grouping according to similarity Planning —> (robot to learn to interact in a certain environment) Classification —> (assign meaning to data) example: Spam filtering I want to predict the outcome of this individual or i want to predict whether a person click or not in a certain advertisement. Examples Classify data into categories: Medical diagnosis: data are medical records and • categories are diseases • Document analysis: data are texts and categories are topics • Image analysts: data are digital images and for categories name of objects in the image (but could be different). • Spam filtering: data are emails, categories are spam vs non spam. • Advertising prediction: data are features of web site visitors and categories could be click/non click on banners. Classification : Different from clustering since we do not have semantically classification (spam or not spam) —> like meaning of the image. I have a semantic label. Clustering: i want to group data with similarity function. Planning: Learning what to do next Clustering: Learn similarity function Classification: Learn semantic labels meaning of data Planning: Learn actions given state In classification is an easier than planning task since I'm able to make prediction telling what is the semantic label that goes with data points. If i can do classification i can clustering. If you do planning you probably classify (since you understanding meaning in your position) and then you can also do clustering probably. We will focus on classification because many tasks are about classification. Classify data in categories we can image a set of categories. For instance the tasks: 'predict income of a person' 'Predict tomorrow price for a stock' The label is a number and not an abstract thing. We can distinguish two cases: The label set —> set of possible categories for each data • point. For each of this could be finite set of abstract symbols (case of document classification, medical diagnosis). So the task is classification. • Real number (no bound on how many of them). My prediction will be a real number and is not a category. In this case we talk about a task of regression. Classification: task we want to give a label predefined point in abstract categories (like YES or NO) Regression: task we want to give label to data points but this label

are numbers. When we say prediction task: used both for classification and regression tasks. Supervised learning: Label attached to data (classification, regression) Unsupervised learning: No labels attached to data (clustering) In unsupervised the mathematical modelling and way algorithm are score and can learn from mistakes is a little bit harder. Problem of clustering is harder to model mathematically. You can cast planning as supervised learning: i can show the robot which is the right action to do in that state. But that depends on planning task is formalised. Planning is higher level of learning since include task of supervised and unsupervised learning. Why is this important ? Algorithm has to know how to given the label. In ML we want to teach the algorithm to perform prediction correctly. Initially algorithm will make mistakes in classifying data. We want to tell algorithm that classification was wrong and just want to perform a score. Like giving a grade to the algorithm to understand if it did bad or really bad. So we have mistakes! Algorithm predicts and something makes a mistake —> we can correct it. Then algorithm can be more precisely. We have to define this mistake. Mistakes in case of classification: If category is the wrong one (in the simple case). We • have a binary signal where we know that category is wrong. How to communicate it? We can use the loss function: we can tell the algorithm whether is wrong or not. Loss function: measure discrepancy between 'true' label and predicted label. So we may assume that every datapoint has a true label. If we have a set of topic this is the true topic that document is talking about. It is typical in supervised learning.

How good the algorithm did?

$$\ell(y, \hat{y}) \leq 0$$

were y is true label and $\hat{y}$ is predicted label

We want to build a spam filter were 0 is not spam and 1 is spam and that Classification task:

$$\ell(y, \hat{y} = \begin{cases} 0, & \text{if } \hat{y} = y \\ 1, & \text{if } \hat{y} \neq y \end{cases}$$

The loss function is the "interface" between algorithm and data. So algorithm know about the data through the loss function. If we give a useless loss function the algorithm will not perform good: is important to have a good loss function. Spam filtering We have two main mistakes: It is the same mistake? No if i have important email and you classify as spam that's bad and if you

show me a spam than it's ok. So we have to assign a different weight. Even in binary classification, mistakes are not equal. e Iotf.TFprIuos.uos True came razee Cussler aircN TASK spam ACG FIRM ftp.y GO IF F Y n is soon IF FEY 0 Nor spam ZERO CNE Cass n n Span No Seamy Binary Classification I 2 FALSE PEENE Mistake Y NON SPAM J Spam FN Mistake i f SPAM y NO spam 2 IF Fp Meter Airenita f Y F on positive y ye en MISTAKE 0 otherwise 0 otherwise

Lecture 2 - 07-04-2020

2.1 Argomento

Classification tasks
Semantic label space Y
Categorization Y finite and
small Regression Y appartiene ad |R
How to predict labels?
Using the lost function → ..
Binary classification
Label space is Y = -1, +1
Zero-one loss

$$\ell(y, \hat{y} = \begin{cases} 0, & \text{if } \hat{y} = y \\ 1, & \text{if } \hat{y} \neq y \end{cases}$$

$FP \quad \hat{y} = 1, \quad y = -1$
$FN \quad \hat{y} = -1, \quad y = 1$

Losses for regression?
y, and $\hat{y} \in \mathbb{R}$,
so they are numbers!
One example of loss is the absolute loss: absolute difference between numbers

2.2 Loss

2.2.1 Absolute Loss

$$\ell(y, \hat{y} = |y - \hat{y}| \Rightarrow absolute \quad loss$$

— DISEGNO —

Some inconvenient properties:

- ...
- Derivative only two values (not much informations)

2.2.2 Square Loss

$$\ell(y, \hat{y} = (y - \hat{y})^2 \Rightarrow \textit{square loss}$$

– DISEGNO —
Derivative :

- more informative
- and differentible

Real numbers as label → regression.
Whenever taking difference between two prediction make sense (value are numbers) then we are talking about regression problem.
Classification as categorization when we have small finite set.

2.2.3 Example of information of square loss

$\ell(y, \hat{y}) = (y - \hat{y})^2 = F(y)$
$F'(\hat{y}) = -2 \cdot (y - \hat{y})$

- I'm under sho or over and how much
- How much far away from the truth

$\ell(y, \hat{y}) = |y - \hat{y}| = F(y') \cdot F'(y) = Sign(y - \hat{y})$

Question about the future
Will it rain tomorrow?
We have a label and this is a binary classification problem.
My label space will be Y = "rain", "no rain"
We don't get a binary prediction, we need another space called prediction space (or decision space).
$Z = [0, 1]$
$\hat{y} \in Z$ $\quad$ *$\hat{y}$ is my prediction of rain tomorrow*
$\hat{y} = \mathbb{P}(y = "rain")$ $\quad \rightarrow$ *my guess is tomorrow will rain (not sure)*

$y \in Y$ $\quad$ $\hat{y} \in Z$
quadHow can we manage loss?
Put numbers in our space
$\{1, 0\}$ $\quad$ *where 1 is rain and 0 no rain*

I measure how much I'm far from reality.
So loss behave like this and the punishment is gonna go linearly??

26..

However is pretty annoying. Sometime I prefer to punish more so i going quadratically instead of linearly.
There are other way to punish this.
I called **logarithmic loss**
We are extending a lot the range of our loss function.

$$\ell(y, \hat{y}) = |y - \hat{y}| \in |0, 1| \qquad \ell(y, \hat{y}) = (y - \hat{y})^2 \in |0, 1|$$

If i want to expand the punishment i use logarithmic loss

$$\ell(y, \hat{y} = \begin{cases} ln\frac{1}{\hat{y}}, & \text{if } y = 1 \textit{(rain)} \\ ln\frac{1}{1-\hat{y}}, & \text{if } y = 0 \textit{(no rain} \end{cases}$$

$F(\hat{y}) \to$ *can be 0 if i predict with certainty*
If $\hat{y} = 0.5 \qquad \ell(y, \frac{1}{2}) = ln2$ *constant losses in each prediction*

$\lim_{\hat{y}\to 0^+} \ell(1, \hat{y}) = +\infty$
We give a vanishing probability not rain but tomorrow will rain.
So this is $+\infty$
$\lim_{\hat{y}\to 1^-} \ell(0, \hat{y}) = +\infty$

The algorithm will be punish high more the prediction is not real. Algorithm will not get 0 and 1 because for example is impossible to get a perfect prediction.
This loss is useful to give this information to the algorithm.

Now we talk about labels and losses

2.2.4 labels and losses

Data points: they have some semantic labels that denote some true about this data points and we want to predict this labels.
We need to define what data points are: number? Strings? File? Typically

they are stored in database records
They can have very precise structure or more homogeneously structured
A data point can be viewed as a vector in some d dimensional real space. So it's a vector of number

$$\mathbb{R}^d X = (x_1, x_2..., x_d) \in \mathbb{R}^c$$

Image can be viewed as a vector of pixel values (grey scale 0-255).
I can use geometry to learn because point are in my Euclidean space. Data can be represented as point in Euclidean space. Images are list of pixel that are pretty much the same range and structure (from 0 to 255). It's very natural to put them in a space.

Assume X can be a record with heterogeneous fields:
For example medical records, we have several values and each fields has his meaning by it's own. (Sex, weight, height, age, zip code)
Each one has a different range, in some cases is numerical but something have like age ..
Does have any sense to see a medical record as a point since coordinates have different meaning.
Fields are not comparable.
This is something that you do: when you want to solve some inference you have to decide which are the label and what is the label space and we have to encode the data points.

Data algorithm expect some homogenous interface. In this case algorithm has to build records with different values of fields.
This is something that we have to pay attention too.
You can always each range of values in number. So ages is number, sex you can give 0 and 1, weight number and zip code is number.
How ever geometry doesn't make sense since I cannot compare this coordinates.
Linear space i can sum up as vector: i can make linear combination of vectors.
Inner product to measure angles! (We will see in linear classifier).

I can scramble the number of my zip code.
So we get problems with sex and zip code

Why do we care about geometry? I can use geometry to learn.

However there is more to that, geometry will carry some semantically information that I'm going to preserve during prediction.
I want to encode my images as vectors in a space. Images with dog.....

PCA doesn't work because assume we encode in linear space.
We hope geometry will help us to predict label correctly and sometimes i hard to convert data into geometry point.
Example of comparable data: images, or documents.
Assume we have documents with corpus (set of documents).
Maybe in English and talk about different thing and different words.
X is a document and i want to encode X into a point fix in bidimensional space.
There is a way to encode a set of documents in point in a fixed dimensional space in such way it make sense this coordinate are comparable.
I can represent fields with [0,1] for Neural network for example. But they have no geometrical meaning

2.2.5 Example TF(idf) documents encoding

TF encoding of docs.

1. Extract where all the words from docs
2. Normalize words (nouns, adjectives, verbs ...)
3. Build a dictionary of normalized words

Doc $x = (x_1, .., x_d)$
I associate a coordinate for each word in a dictionary.
d = number of words in dictionary
I can decide that
$x_i = 1$ *If i-th word of dictionary occurs in doc.*
$x_i = 0$ *Else*

X_i *number of time i-th word occur in doc.*
Longer documents will have higher value of coordinates that are not zero.
Now i can do the TF encoding in which xi = frequency with which i-th word occur in dictionary.
You cannot sum dog and cat but we are considering them frequencies so we are summing frequency of words.
This encoding works well in real words.

I can choose different way of encoding my data and sometime i can encode a real vector

I want

1. A predictor $f : X \longrightarrow Y$ (in weather $X \longrightarrow Z$
2. X is our data space (where points live)
3. $X = \mathbb{R}^d$ images
4. $X = X_1 x ... x X_d$ Medical record
5. $\hat{y} = f(x)$ predictor for X

(x, y)

We want to predict a label that is much closer to our label. How?
Loss function: so this is my setting and is called and example.
Data point together with label is a "example"
We can get collection of example making measurements or asking people. So we can always recover the true label.
We want to replace this process with a predictor (so we don't have to bored a person).
y is the ground truth for x $\rightarrow$ mean reality!
If i want to predict stock for tomorrow, i will wait tomorrow to see the ground truth.

Lecture 3 - 07-04-2020

Data point x represented as sequences of measurement and we called this measurements features or attributes.

$x = (x_1, ..., x_d)$ x_1 *feature valuex* $\in X^d$ $X = \mathbb{R}^d$ $X = X_1 \cdot x \cdot ... \cdot X_d \cdot x$

Label space Y
Predictor $f : X \to Y$

Example (x, y) y is the label associated with x
($\to y$ is the correct label, the ground truth)

Learning with example $(x_1, y_1)...(x_m, y_m)$ *training set*

Training set is a set of examples with every algorithm can learn.....

Learning algorithm take training set as input and produces a predictor as output.

.....DISEGNO

With image recognition we use as measurement pixels.
How do we measure the power of a predictor?
A learning algorithm will look at training set, algorithm and generate the predictor. Now the problem is verify the score.
Now we can consider a test set collection of example

$$\textit{Test set} \qquad (x'_1, y'_1)...(x'_n, y'_n)$$

Typically we collect big dataset and then we split in training set and test set randomly.
Training and test are typically disjoint
How we measure the score of a predictor? We compute the average loss.
The error is the average loss in the element in the test set.

$$\textit{Test error} \qquad \frac{1}{n} \cdot \sum_{t=1}^{n} \ell(f(x'_t), y')$$

In order to simulate we collect the test set and take the average loss of the predictor of the test set. This will give us idea of how the..
Proportion of test and train depends in how big the dataset is in general.

Our **Goal**: A learning algorithm 'A' must output f with a small test error. A does not have access to the test set. (Test set is not part of input of A). Now we can think in general on how a learning algorithm should be design. We have a training set so algorithm can say:
'A' may choose f based on performance on training set.

$$\textit{Training error} \qquad \hat{\ell}(f) = \frac{1}{m} \cdot \sum_{t=1}^{m} \ell(f(x_t), y_t)$$

Given the training set $(x_1, ..., x_m)(y_1, ..., y_m)$
If $\hat{\ell}(f)$ for same f, then test of f is also small
Fix F set of predictors output $\hat{f}$

$$\hat{f} = arg\,min\,\hat{\ell}(f) f \in F$$

This algorithm is called Empirical Risk Minimiser (ERM)
When this strategy (ERM) fails?
ERM may fails if for the given training set there are:
Many $f \in F$ with small $\hat{\ell}(f)$, but not all of them have small test error

There could be many predictor with small error but some of them may have big test error. Predictor with the smallest training error doesn't mean we will have the smallest test error.
I would like to pick f^* such that:

$$f^* = arg\,min\frac{1}{n} \cdot \sum_{t=1}^{m} \ell(f(x'_t), y_t) \qquad f \in F$$

where $\ell(f(x'_t), y_t)$ is the test error
ERM works if $f^* such\ that \qquad f^* = arg\,min\,\hat{\ell}(f) \qquad f \in F$
So minimising training and test????? Check videolecture
We can think of f as finite since we are working on a finite computer.
We want to see why this can happen and we want to formalise a model in which we can avoid this to happen by design: We want when we run ERM choosing a good predictor with ..... PD

3.1 Overfitting

We called this as overfitting: specific situation in which 'A' (where A is the learning algorithm) overfits if f output by A tends to have a training error much smaller than the test error.
A is not doing his job (outputting large test error) this happen because test error is misleading.
Minimising training error doesn't mean minimising test error. Overfitting is bad.
Why this happens?
This happen because we have **noise in the data**

3.1.1 Noise in the data

Noise in the data: y_t is not deterministically associated with x_i.

Could be that datapoint appears more times in the same test set. Same datapoint is repeated actually I'm mislead since training and dataset not coincide. Minimising the training error can take me away from the point that minimise the test error.
Why this is the case?

- Some **human in the loop**: label assigned by people.(Like image contains certain object but human are not objective and people may have different opinion)
- **Lack of information**: in weather prediction i want to predict weather error. Weather is determined by a large complicated system. If i have humidity today is difficult to say for sure that tomorrow will rain.

When data are not noise i should be ok.
Labels are not noisy

Fix test set and trainign set.

$$\exists f^* \in F \qquad y'_t = f^*(x'_t) \qquad \forall (x'_t, y'_t) \quad \textit{in test set}$$

$$y_t = f^+(x_t) \qquad \forall (x_t, y_t) \quad \textit{in training set}$$

Think a problem in which we have 5 data points(vectors) :
$\vec{x_1}, ...\vec{x_5}$ *in some space* X
We have a binary classification problem $Y = \{0, 1\}$
$\{\vec{x_1}, ..., \vec{x_5}\} \in X \qquad Y = \{0, 1\}$

F contains all possible calssifier $2^5 = 32 \qquad f : \{x_1, ..., x_5\} \to \{0, 1\}$

Example					
	x_1	x_2	x_3	x_4	x_5
f	0	0	0	0	0
f'	0	0	0	0	1
$f"$	..	..	..	..	..

$$Training\ set \quad x_1, x_2, x_3 \quad f^+$$
$$Test\ set \quad x_4, x_5 \quad f^*$$

4 *classifier* $f \in F \qquad$ *will have* $\hat{\ell}(f) = 0$

$(x_1, 0) \quad (x_2, 1) \quad (x_3, 0)$
$(x_4, ?) \quad (x_5, ?)$
$f^*(x_4) \quad f^*(x_5)$
If not noise i will have deterministic data but in this example (worst case) we get problem.
I have 32 classifier to choose: i need a larger training set since i can't distinguish predictor with small and larger training(?) error. So overfitting noisy or can happen with no noisy but few point in the dataset to define which predictor is good.

3.2 Underfitting

'A' underfits when f output by A has training error close to test error but they are both large.
Close error test and training error is good but the are both large.

$A \equiv ERM$, *then A undefits if F is too small* $\to$ *not containing too much predictors*

In general, given a certain training set size:

- Overfitting when $|F|$ is too large (not enough points in training set)
- Underfitting when $|F|$ is too small

Proportion predictors and training set

$$|F|,\ i\ need\ ln|F| \quad bits\ of\ info\ to\ uniquely\ determine\ f^* \in F$$
$$m >> ln|F| \qquad when \quad |F| < \infty where\ m\ is\ the\ size\ of\ traning\ set$$

3.3 Nearest neighbour

This is completely different from ERM and is one of the first learning algorithm. This exploit the geometry of the data. Assume that our data space X is:
$X \equiv \mathbb{R}^d \qquad x = (x_1, ..., x_d) \qquad y - \{-1, 1\}$
S is the traning set $(x_1, y_1)...(x_m, y_m)$
$x_t \in \mathbb{R}^d \qquad y_t \in \{-1, 1\}$

$d = 2 \rightarrow$ *2-dimensional vector*

....– DISEGNO –...
where + and - are labels

Point of test set
If i want to predict this point?
Maybe if point is close to point with label i know then. Maybe they have the same label.
$\hat{y} = + \quad or \quad \hat{y} = -$

.....– DISEGNO – ...
I can came up with some sort of classifier.

Given S training set, i can define $h_{NN}\ X \rightarrow \{-1, 1\}$
$h_{NN}(x) =$ label y_t of the point x_t in S closest to X
(the breaking rule for ties)
For the closest we mean euclidian distance
$X = \mathbb{R}^d$

$$\|x - x_t\| = \sqrt{\sum_{e=1}^{d} (x_e - x_t, e)^2}$$

$$\hat{\ell}(h_{NN}) = 0$$
$$h_{NN}(x_t) = y_t$$

training error is 0!

Lecture 4 - 07-04-2020

We spoke about Knn classifier with voronoi diagram

$$\hat{\ell}(h_{NN}) = 0 \qquad \forall \, Traning \; set$$

h_{NN} predictor needs to store entire dataset.

4.1 Computing h_{NN}

Computing $h_{NN}(x)$ requires computing distances between x and points in the traning set.

$$\Theta(d) \quad time \; for \; each \; distance$$

NN $\rightarrow$ 1-NN
We can generalise NN in K-NN with $k = 1, 3, 5, 7$ so odd K
$h_{k-NN}(x) =$ label corresponding to the majority of labels of the k closet point to x in the training set.

How big could K be if i have n point?
I look at the k closest point
When $k = m$?
The majority, will be a constant classifier h_{k-NN} is constant and corresponds to the majority of training labels
Training error is always 0 for h_{NN}, while for h_{k-NN} will be typically > 0, with $k > 1$
Image: one dimensional classifier and training set is repeated. Is the plot of 1-NN classifier.
Positive and negative. $K = 1$ error is 0.
In the second line we switch to $k = 3$. Second point doesn't switch and third will be classify to positive and we have training mistake.
Switches corresponds to border of voronoi partition.

$$K_{NN} \qquad For \; multiclass \; classification$$

$$(|Y| > 2) \qquad for \; regression \; Y \equiv \mathbb{R}$$

Average of labels of K neighbours $\rightarrow$ i will get a number with prediction.
I can weight average by distance

You can vary this algorithm as you want.

Let's go back to Binary classification.
The k parameter is the effect of making the structure of classifier more complex and less complex for small value of k.

–.. DISEGNO ..–
Fix training set and test set
Accury as oppose to the error

Show a plot. Training error is 0 at $k = 0$.
As i go further training error is higher and test error goes down. At some point after which training and set met and then after that training and test error goes up (accuracy goes down).
If i run algorithm is going to be overfitting: training error and test error is high and also underfitting since testing and training are close and both high.
Trade off point is the point in $x = 23$ (more or less).
There are some heuristic to run NN algorithm without value of k.

History

- K_{NN}: from 1960 $\rightarrow X \equiv \mathbb{R}^d$
- Tree predictor: from 1980

4.2 Tree Predictor

If a give you data not welled defined in a Euclidean space.
$X = X_1 \cdot x \cdot ... \cdot X_d \cdot x$ Medical Record
$X_1 = \{Male, Female\}$
$X_2 = \{Yes, No\}$
so we have different data

I want to avoid comparing x_i with x_j, $i \neq j$
so comparing different feature and we want to compare each feature with each self. I don't want to mix them up.
We can use a tree!
I have 3 features:

- outlook $= \{sunny, overcast, rain\}$

- humidity = $\{[0, 100]\}$
- windy = $\{yes, no\}$

... – DISEGNO – ...

Tree is a natural way of doing decision and abstraction of decision process of one person. It is a good way to deal with categorical variables.
What kind of tree we are talking about?
Tree has inner node and leaves. Leaves are associated with labels (Y) and inner nodes are associated with test.

- Inner node $\rightarrow$ test
- Leaves $\rightarrow$ label in Y

Test if a function f (NOT A PREDICTOR!)
Test $f_i\, X_i \rightarrow \{1, ..., k\}$
where k is the number of children (inner node) to which test is assigned
In a tree predictor we have:

- Root node
- Children are ordered(i know the order of each branch that come out from the node)

$X = \{Sunny, 50\%, No\} \quad \rightarrow \quad$ *are the parameters for* $\{outlook.humidity, windy\}$

$$f_i = \begin{cases} 1, & \text{if } x_2 \in [30\%, 60\%] \\ 2, & \text{if } otherwise \end{cases}$$

where the numbers 1 and 2 are the children
A test is partitioning the range of values of a certain attribute in a number of elements equal to number of children of of the node to which the test is assigned.
$h_T(x)$ is always the label of a leaf of T
This leaf is the leaf to which x is **routed**
Data space for this problem (outlook,..) is partitioned in the leaves of the tree. It won't be like voronoi graph. How do I build a tree given a training set? How do i learn a tree predictor given a training set?

- Decide tree structure (how • many node, leaves ecc..)
- Decide test on inner nodes

- Decide labels on leaves

We have to do this all together and process will be more dynamic. For simplicity binary classification and fix two children for each inner node.

$Y = \{-1, +1\}$
2 children for each inner node

What's the simplest way?
Initial tree and correspond to a costant classifier

– DISEGNO –

Majority of all example

– DISEGNO –

$(x_1, y_1)...(x_m, y_m)$
$x_t \in X \qquad y_t \in \{-1, +1\}$
Training set $S = \{(x, y) \in S$, x is routed to $\ell\}$
S_ℓ^+

– DISEGNO –

S_ℓ and S'_ℓ are given by the result of the test, not the labels and ℓ and ℓ'.

Lecture 5 - 07-04-2020

5.1 Tree Classifier

Supposed we groped a tree up to this point and we are wandering how to grow it.
S Training set $(x_1, y_1)...(x_m, y_m)$, $x_1 \in X$

– DISEGNO

$$S_\ell \equiv \{(x_1, y_1)\, x_t \quad is\ router\ to\ \ell\}$$

$y_1 \in \{-1, 1\}$

$$S_\ell^+ \equiv \{(x_1, y_1) \in S_\ell : \quad y_t = +1\}$$
$$S_\ell^- \equiv \{(x_1, y_1) \in S_\ell : \quad y_t = -1\} \qquad S_\ell^+ \cap S_\ell^- \equiv 0 \qquad S_\ell \equiv S_\ell^+ \cup S_\ell^-$$

$$N_\ell = |S_\ell| \qquad N_\ell^+ = |S_\ell^+| \qquad N_\ell^- = |S_\ell^-|$$
$$N_\ell = N_\ell^- + N_\ell^+$$

leaf ℓ classifies all traning example (S_ℓ)

$$Y_\ell = \begin{cases} +1, & \text{If } N_\ell^+ \geq N_\ell^- \\ -1, & \text{If } otherwise \end{cases}$$

ℓ makes a mistake on $min\{N_\ell^+, N_\ell^-\}$ example in S_ℓ

$$\hat{\ell}(h_T) = \frac{1}{m} \cdot \sum_\ell min\{\frac{N_\ell^+}{N_\ell}, \frac{N_\ell^-}{N_\ell}\} \cdot N_\ell =$$

$$= \frac{1}{m} \cdot \sum_\ell \psi \cdot (\frac{N_\ell +}{N_\ell}) \cdot N_\ell \quad \longrightarrow \qquad \frac{N_\ell^+}{N_\ell} = 1 - \frac{N_\ell}{N_\ell ??}$$

where $\psi(a) = min\{a, 1-a\} \qquad a \in [0, 1]$
I want to replace inner node with other leaves.
– DISEGNO –

How is traning error going to change? (when i replace inner nodes with other leaves)

I'm hoping my algorithm is not going to overfit (if training error goes to 0 also testing error goes to 0).

5.2 Jensen's inequality

If ψ is a concave function $\longrightarrow$ (like *log* or $\sqrt[2]{..}$)
Also ψ is a function that map 0 to 1, $\longrightarrow$ $\psi\ [0,1] \to \mathbb{R}$

$\psi(\alpha \cdot a + (1-\alpha)\cdot b) \geq \alpha \cdot \psi(a) + (1-\alpha)\cdot\psi(b)$ *Also 2° derivative is negative*

– DISEGNO –

$$\hat{\ell}(h_T) = \frac{1}{m} \cdot \sum_{\ell} \psi(\frac{N_\ell^+}{N_\ell}) \cdot N_\ell$$

Look a single contribution fo a leaf ℓ to training error

$$\psi(\frac{N_\ell^+}{N_\ell}) \cdot N_\ell = \psi(\frac{N_\ell'^+}{N_\ell'} \cdot \frac{N_\ell'}{N_\ell} + \frac{N_\ell"^+}{N_\ell"} \cdot \frac{N_\ell"}{N_\ell}) \cdot N_\ell$$

where $\frac{N_\ell'}{N_\ell} = \alpha$ and $\frac{N_\ell"}{N_\ell} = 1-\alpha$ so $\frac{N_\ell'}{N_\ell} + \frac{N_\ell"}{N_\ell} = 1$ $\longrightarrow \alpha + 1 - \alpha = 1$

$N_{\ell'}^+ + N_{\ell"}^+ = N_\ell$

I want to check function *min* concave between 0 and 1.

$$min(0,1) = 0 \qquad \psi(a) = min(\alpha, 1-\alpha)$$

– DISEGNO –

This is a concave function and now I can apply Jensen's inquality

$$\psi(\frac{N_\ell^+}{N_\ell}) \cdot N_\ell \geq (\frac{N_\ell'}{N_\ell} \cdot \psi(\frac{N_\ell'^+}{N_\ell'}) + \frac{N_\ell"}{N_\ell} \cdot \psi(\frac{N_\ell"^+}{N_\ell"})) \cdot N_\ell =$$

$$= \boxed{\psi(\tfrac{N_\ell'^+}{N_\ell'}) \cdot N_\ell' + \psi(\tfrac{N_\ell"^+}{N_\ell"}) \cdot N_\ell"}$$

This are the contribuion of ℓ' and $\ell"$ to the training error

Every time i split my tree my training error is never going to increase since we have a concave function.
Whenever I'm growing my tree training error is going to be smaller.

Every time a leaf is expanded the training error never goes up. (Hopelly will go down)
I'll should always grow the tree by expanding leave that decrease the training error as much as possible.
If i take the effort of growing the tree i should get benefits. I can imaging that if i grow the tree at random my training error is going to drop down error (but maybe will derive overfitting).
For now is just an intuition since we will introduced statistical learning model.

Could be complicated and tree big may have 100 leave and there could be many way of associating a test with that leaves.
I can spent a lot of time to select which leave is the best promising to split.

- Grow the tree by expanding leave that decrease the training error as much as possible
- In general we can assume:
  greedy algorithm at each step pick the pair leaf and test that cause (approximative) the largest decrease in training error

In practise we want optimise this all the way since it's time expensive. That's the approximately since we are not every time sure.

— MANCA PARTE —
— IMMAGINE —

$p = 0.8 \qquad q = 1 \qquad r = 1 \qquad \alpha = 60\%$
Net Change in number of mistakes

$$\psi(p) - (\alpha \cdot \psi(q) + (1 - \alpha) \cdot \psi(r)) =$$
$$\ell \quad - \qquad \ell' \quad + \qquad \ell"$$

Fraction of example miss classified $\ell-$ error $\ell'+$ error $\ell"$

$$= 0.2 - (\frac{1}{2} \cdot 0.4 + \frac{1}{2} \cdot 0) = 0$$

— DISEGNO —

Idea is to replace minimum function with convex combination.

$$\psi(\alpha) = min\ \{\alpha, 1-\alpha\} \qquad \psi(a) \geq \psi(\alpha)$$

$$\begin{cases} \psi_1(\alpha) = 2 \cdot \alpha \cdot (1-\alpha) \longrightarrow GNI \\ \psi_2(\alpha) = -\frac{\alpha}{2} \cdot \ln \alpha - \frac{1-\alpha}{2} \cdot \ln(1-\alpha) \longrightarrow ENTROPY \\ \psi_3(\alpha) = \sqrt{\alpha \cdot (1-\alpha)} \end{cases}$$

All this functions has this shape (concave???)
– DISEGNO –
In practise Machine Learning algorithm use GNI or entropy to control the split

5.3 Tree Predictor

- Multi class classification $|Y| > 2 \longrightarrow$ take majority
- Regression $Y = \mathbb{R} \longrightarrow$ take average of labels in S_ℓ

I still take majority among different classes.
Take average of labels in S_ℓ
Unless $\frac{N_\ell^+}{N_\ell} \in 0, 1 \qquad \forall$ leaves ℓ, $\hat{\ell}(h_T) > 0$
Unless leaves are *"pured"*, the training error will be bigger than 0.

In general, i can always write $\hat{\ell}(h_t)$ to 0 by growing enough the tree unless there are x_1 in the Time Series such that $(x_t, y_t)(x_t, y_t')$ with $y_t \neq y_t'$ both occur.
— DISEGNO —-

$$if (x_1 = \alpha) \wedge (x_2 = \geq \alpha) \vee (x_1 = b) \vee (x_1 = c) \wedge (x_3 = y)$$
$$then\ predict\ 1$$
$$else$$
$$then\ predict\ -1$$

— Picture of tree classifier of iris dataset. —

I'm using due attribute at the time.
Each data point is a flower and i can measure how petal and sepal are long. I can use two attribute and i test this two. I can see the plot of the tree classifier (second one) making test splitting data space into region that has this sort of "blackish" shape (like boxes: blue box, red box, yellow box)
A good exercise in which I want to reconstruct the tree given this picture.

5.4 Statistical model for Machine Learning

To understand Tree classifier, nearest neighbour and other algorithm...
It's important to understand that the only way to have a guideline in which model to choose.

This mathematical model are developed to learning and choose learning algorithm.

Now let start with theoretical model.

- How example (x, y) are generated to create test set and training set?
  We get the dataset but we need to have a mathematical model for this process. (x, y) are drawn from a fixed but unknown probability distribution on the pairs X and Y (X data space, Y label set o label space)

- Why X should be random?
  In general we assumed that not all the x in X are equally likely to be observed. I have some distribution over my data point and this said that I'm most like to get a datapoint to another.

- How much label?
  Often labels are not determined uniquely by their datapoints because labels are given by human that have their subjective thoughts and also natural phenomena. Labels are stochastic phenomena given a datapoint: i will have a distribution.

We're going to write (in capital) (X, Y) since they are random variable drawn from D on $X \cdot Y$ A dataset $(X_1, Y_1)...(X_m, Y_m)$ they are drawn independently from D (distribution on examples)
When I get a training the abstraction of process collecting a training set

D is a joint probability distribution over $X \cdot Y$
where D_x is the marginal over $X \rightarrow D_y|x$ (conditional of Y given X).
I can divided my draw in two part. I draw sample and label from conditional.??
Any dataset (training or test) is a random sample (campione casuale) in the statistical sense $\longrightarrow$ so we can use all stastical tools to make inference.

Lecture 6 - 07-04-2020

(X, Y) We random variables drawn iid from D on $X \cdot Y \longrightarrow$ where D is fixed but unknown

Independence does not hold. We do not collect datapoints to an independent process.
Example: identify new article and i want to put categories. The feed is highly depend on what is happening in the world and there are some news highly correlated. Why do we make an assumption that follows reality? Is very convenient in mathematical term. If you assume Independence you can make a lot of process in mathematical term in making the algorithm.
If you have enough data they look independent enough. Statistical learning is not the only way of analyse algorithms —> we will see in linear ML algorithm and at the end you can use both statistical model s

6.1 Bayes Optimal Predictor

$$f^* : X \to Y$$
$$f^*(x) = argmin\, \mathbb{E}\,[\,\ell(y, \hat{y})|X = x\,] \qquad \hat{y} \in Y$$

In general Y given X has distribution $D_y|X = x$
Clearly $\forall\ h \quad X \to Y$

$$\mathbb{E}\,[\,\ell(y, f^*(x))|X = x\,] \leq \mathbb{E}\,[\,\ell(y, h(x)|X = x\,]$$
$$X, Y \qquad \mathbb{E}\,[\,Y|X = x\,] = F(x) \quad \longrightarrow Conditional Expectation$$
$$\mathbb{E}\,[\,\mathbb{E}\,[\,Y|X\,]\,] = \mathbb{E}(Y)$$

Now take Expectation for distribution

$$\mathbb{E}\,[\,\ell(y, f^*(x))\,] \leq [\,\mathbb{E}(\ell(y, h(x))\,]$$

where risk is smaller in f^*
I can look at the quantity before
l_d Bayes risk $\longrightarrow$ Smallest possible risk given a learning problm

$$l_d(f^*) > 0 \qquad because\ y\ are\ still\ stochastic\ given\ X$$

Learning problem can be complem → large risk

6.1.1 Square Loss

$$\ell(y, \hat{y} = (y - \hat{y})^2$$

I want to compute bayes optimal predictor
$\hat{y}, y \in \mathbb{R}$

$$f^*(x) = argmin\, \mathbb{E}\left[(y - \hat{y})^2 | X = x \right] = \qquad \hat{y} \in \mathbb{R}$$

we use $\mathbb{E}\left[X + Y \right] = \mathbb{E}[X] + \mathbb{E}[Y] = argmin\, \mathbb{E}\left[y^2 + \hat{y}^2 - 2 \cdot y \cdot \hat{y}^2 | X = x \right] =$

Dropping y^2 i remove something that is not important for $\hat{y}$

$$= argmin(\mathbb{E}\left[y^2 | X = x \right] + \hat{y}^2 - 2 \cdot \hat{y} \cdot \mathbb{E}\left[y | X = x \right]) =$$
$$= argmin(\hat{y}^2 - 2 \cdot \hat{y} \cdot \mathbb{E}\left[y | X = x \right]) =$$

Expectation is a number, so it's a constant
Assume $\boxdot = y^2$

$$argmin\ \left[\boxdot + \hat{y}^2 + 2 \cdot \hat{y} \cdot \mathbb{E}\left[Y | X = x \right] \right]$$

where red$G(\hat{y})$ is equal to the part between [...]

$$\frac{dG(\hat{y})}{d\hat{y}} = 2 \cdot \hat{y} - 2 \cdot \mathbb{E}\left[y | X = x \right] = 0 \quad \longrightarrow \quad \textit{So setting derivative to 0}$$

— DISEGNO OPT CURVE —

$G'(\hat{y}) = \hat{y}^2 - 2 \cdot b \cdot \hat{y}$

$$\hat{y} = \mathbb{E}\left[y | X = x \right] \qquad f^*(x) = \mathbb{E}\left[y | X = x \right]$$

Square loss is nice because expected prediction is ...
In order to predict the best possibile we have to estimate the value given

data point.

$$\mathbb{E}\left[\,(y - f^*(x))^2 | X = x\,\right] =$$
$$= \mathbb{E}\left[\,(y - \mathbb{E}\left[\,y|X = x\,\right])^2 | X = x\,\right] = Var\left[\,Y|X = x\,\right]$$

6.1.2 Zero-one loss for binary classification

$Y = \{-1, 1\}$

$$\ell(y, \hat{y}) = I\{\hat{y} \neq y\} \qquad I_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

If $\hat{y} \neq y$ true, indicator function will give us 1, otherwise it will give 0

$$D \quad on \qquad X \cdot Y \qquad D_x^* \quad D_{y|x} = D$$

$$D_x \qquad \eta : X \longrightarrow [\,0, 1\,] \qquad \eta = \mathbb{P}\,(y = 1 | X = x)$$

$$D \rightsquigarrow (D_x, \eta) \quad \longrightarrow \quad \textit{Distribution 0-1 loss}$$

$X \backsim D_x \quad \longrightarrow \quad$ *Where $\backsim$ mean "draw from" and D_x is marginal distribution*

$$Y = 1 \qquad \textit{with probability } \eta(x)$$

$$D_{y|x} = \{\eta(x), 1 - \eta(x)\}$$

Suppose we have a learning domain
— DISEGNO –
where η is a function of x, so i can plot it
η will te me $Prob(x) =$
η tells me a lot how hard is learning problem in the domain
$\eta(x)$ is not necessary continous
— DISEGNO —

$\eta(x) \in \{0, 1\}$ $\quad$ y is always determined by x

How to get f^* from the graph?

$$f^+ : X \to \{-1, 1\}$$
$$Y = \{-1, +1\}$$

— DISEGNO —
==========
MANCA ROBAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
==========

$$f^*(x) = argmin\, \mathbb{E}\,[\,\ell(y, \hat{y})|X = x\,] = \qquad \longrightarrow \hat{y} \in \{-1, +1\}$$

$$= argmin\, \mathbb{E}\,[\,I\{\hat{y} = 1\} \cdot I\{Y = -1\} + I\{\hat{y} = -1\} \cdot I\{y = 1\}\,|\,X = x\,] =$$

we are splitting wrong cases

$$= argmin\,(\,I\{\hat{y} = 1\} \cdot \mathbb{E}\,[\,I\{Y = -1\}|\,X = x\,] + I\{\hat{y} = -1\} \cdot \mathbb{E}\,[\,I\{y = 1\}\,|\,X = x\,]\,) = \quad \divideontimes$$

We know that:

$$\mathbb{E}\,[\,I\{y = -1\}\,|\,X = x\,] = 1 \cdot \mathbb{P}(\hat{y} = -1|X = x) + 0 \cdot \mathbb{P}(y = 1|X = x) =$$
$$\mathbb{P}(x = -1|X = x) = 1 - \eta(x)$$

$$\divideontimes = argmin\,(\,I\{\hat{y} = 1\} \cdot (1 - \eta(x)) + I\{\hat{y} = -1\} \cdot (\eta(x)\,)$$

where Blue colored $I\{...\} = 1°$ and Orange $I\{...\} = 2°$

I have to choose -1 or +1 so we will **remove one of the two (1° or 2°)**
It depend on $\eta(x)$:

- If $\eta(x) < \frac{1}{2}$ $\longrightarrow$ kill 1°
- Else $\eta(x) \geq \frac{1}{2}$ $\longrightarrow$ kill 2°

$$f^*(x) = \begin{cases} +1 & if\, \eta(x) \geq \frac{1}{2} \\ -1 & if\, \eta(x) < \frac{1}{2} \end{cases}$$

6.2 Bayes Risk

$$\mathbb{E}\left[\, I\{y \neq f^*(x)\} \,|\, X = x \,\right] = \mathbb{P}(y \neq f^*(x) | X = x)$$

$$\eta(x) \geq \frac{1}{2} \quad \Rightarrow \quad \hat{y} = 1 \quad \Rightarrow \quad \mathbb{P}(y \neq 1 | X = x) = 1 - \eta(x)$$

$$\eta(x) < \frac{1}{2} \quad \Rightarrow \quad \hat{y} = -1 \quad \Rightarrow \quad \mathbb{P}(y \neq 1 | X = x) = \eta(x)$$

Conditiona risk for 0-1 loss is:

$$\mathbb{E}\left[\, \ell(y, f^*(x)) \,|\, X = x \,\right] \quad = \quad I\{\eta(x) \geq \frac{1}{2}\} \cdot (1 - \eta(x)) + I\{\eta(x) < \frac{1}{2}\} \cdot \eta(x) =$$
$$= min\left\{\eta(x), 1 - \eta(x)\right\}$$

$$\mathbb{E}\left[\, \ell, f^*(x) \,\right] = \mathbb{E}\left[\, min\left\{\eta(x), 1 - \eta(x)\right\} \,\right]$$

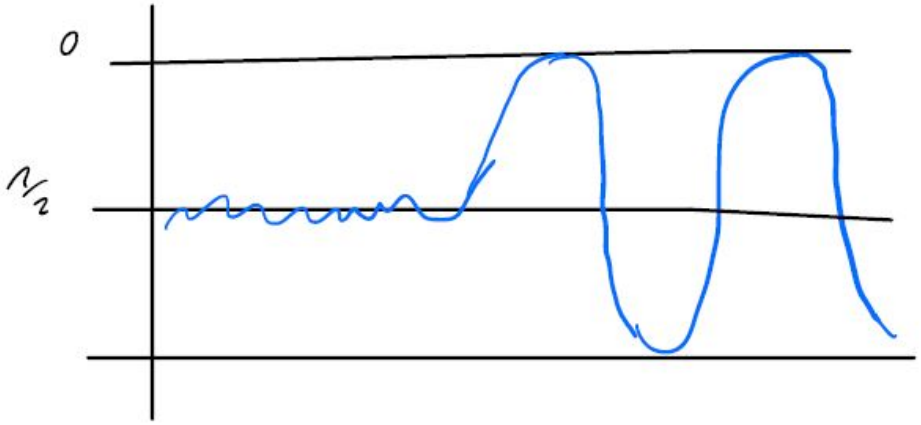


Figure 6.1: Example of Bayes Risk

Conditional risk will be high aroun the half so min between the two is around the half since the labels are random i will get an error near 50%.
My condition risk will be 0 in the region in the bottom since label are going to be deterministic.

Lecture 7 - 07-04-2020

Bounding statistical risk of a predictor
Design a learning algorithm that predict with small statistical risk

$$(D, \ell) \qquad \ell_d(h) = \mathbb{E}\left[\, \ell(y), h(x) \,\right]$$

were D is unknown

$$\ell(y, \hat{y}) \in [0, 1] \quad \forall y, \hat{y} \in Y$$

We cannot compute statistical risk of all predictor.
We assume statistical loss is bounded so between 0 and 1. Not true for all losses (like logarithmic).
Before design a learning algorithm with lowest risk, How can we estimate risk?
We can use test error → way to measure performances of a predictor h. We want to link test error and risk.
Test set $S' = \{(x'_1, y'_1)...(x'_n, y'_n)\}$ is a random sample from D
How can we use this assumption?
Go back to the definition of test error

Sample mean (IT: Media campionaria)

$$\hat{\ell}_s(h) = \frac{1}{n} \cdot \sum_{t=1}^{n} \ell(\hat{y}_t, h(x'_t))$$

i can look at this as a random variable $\ell(y'_t, h(x'_t))$

$$\mathbb{E}\left[\, \ell(y'_t, h(x'_t)) \right] = \ell_D(h) \longrightarrow risk$$

Using law of large number (LLN), i know that:

$$\hat{\ell} \longrightarrow \ell_D(h) \qquad as \quad n \to \infty$$

We cannot have a sample of $n = \infty$ so we will introduce another assumption: the Chernoff-Hoffding bound

7.1 Chernoff-Hoffding bound

$$Z_1, ..., Z_n \quad iid\ random\ variable \qquad \mathbb{E}\left[Z_t\right] = u$$

all drawn for the same distribution

$$t = 1, ..., n \qquad and \qquad 0 \leq Z_t \leq 1 \qquad t = 1, ..., n \quad then \quad \forall \varepsilon > 0$$

$$\mathbb{P}\left(\frac{1}{n}\cdot\sum_{t=1}^{n} z_t > u+\varepsilon\right) \leq e^{-2\,\varepsilon^2\,n} \qquad or \qquad \mathbb{P}\left(\frac{1}{n}\cdot\sum_{t=1}^{n} z_t < u+\varepsilon\right) \leq e^{-2\,\varepsilon^2\,n}$$

as sample size then $\downarrow$

$$Z_t = \ell(Y_t', h(X_t')) \in [0,1]$$

$(X_1', Y_1')...(X_n', Y_N')$ are *iid* therefore,
$\ell\,(Y_t', h\,(X_t'))\quad t = 1, ..., n\quad$ are also *iid*
We are using the bound of e to bound the deviation of this.

7.2 Union Bound

Union bound: a collection of event not necessary disjoint, then i know that probability of the union of this event is the at most the sum of the probabilities of individual events

$$A_1, ..., A_n \qquad \mathbb{P}\,(A_1 \cup ... \cup A_n) \leq \sum_{t=1}^{n} \mathbb{P}\,(A_t)$$

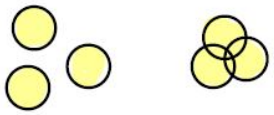


Figure 7.1: Example

that's why $\leq$

$$\mathbb{P}\left(|\,\hat{\ell}_{s'}\,(h) - \ell_D\,(h)\,| > \varepsilon\right)$$

This is the probability according to the random draw of the test set.

If test error differ from the risk by a number epsilon > 0. I want to bound the probability. This two thing will differ by more than epsilon. How can i use the Chernoff bound?

$$|\,\hat{\ell}_{s'}\,(h) - \ell_D\,(h)\,| > \varepsilon \quad \Rightarrow \quad \hat{\ell}_{s'}\,(h) - \ell_D\,(h) > \varepsilon \quad \vee \quad \hat{\ell}_D\,(h) - \ell_{s'}\,(h) > \varepsilon$$

Figure 7.2: Example

$$A, B \qquad A \Rightarrow B \qquad \mathbb{P}(A) < \mathbb{P}(B)$$

$$\mathbb{P}\left(\mid \hat{\ell}_{s'}(h) - \ell_D(h) \mid > \varepsilon\right) \leq \mathbb{P}\left(\left|\hat{\ell}_{s'}(h) - \ell_D(h)\right|\right) \quad \cup \quad \mathbb{P}\left(\left|\hat{\ell}_D(h) - \ell_{s'}(h)\right|\right) \leq$$

$$\leq \mathbb{P}\left(\hat{\ell}_{s'} > \ell_D(h) + \varepsilon\right) + \mathbb{P}\left(\hat{\ell}_{s'} < \ell_D(h) - \varepsilon\right) \quad \leq \quad 2 \cdot e^{-2\varepsilon^2 n} \quad \Rightarrow \textit{we call it } \delta$$

$$\varepsilon = \sqrt{\frac{1}{2 \cdot n} \ln \frac{2}{\delta}}$$

The two events are disjoint

This mean that probability of this deviation is at least delta!

$$\mid \hat{\ell}_{s'}(h) - \ell_D(h) \mid \leq \sqrt{\frac{1}{2 \cdot n} \ln \frac{2}{\delta}} \qquad \textit{with probability at least } 1 - \delta$$

Test error of true estimate is going to be good for this value (δ)
Confidence interval for risk at confidence level 1-delta.

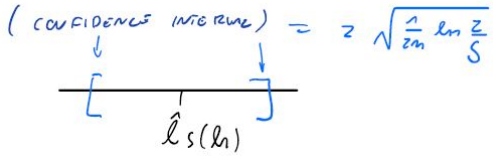


Figure 7.3: Example

I want to take $\delta = 0,05$ so that $1 - \delta$ is 95%. So test error is going to be an estimate of the true risk which is precise that depend on how big is the test set (n).
As n grows I can pin down the position of the true risk.
This is how we can use probability to make sense of what we do in practise.
If we take a predictor h we can compute the risk error estimate.

We can measure how accurate is our risk error estimate.
Test error is an estimate of risk for a given predictor (h).

$$\mathbb{E}\left[\ell\left(Y_t', h\left(X_t'\right)\right)\right] = \ell_D\left(h\right)$$

h is fixed with respect to S' $\longrightarrow$ h does not depend on the test set. So learning algorithm which produce h not have access to test set.
If we use test set we break down this equation.

Now, how to **build a good algorithm?**
Training set $S = \{(x_1, y_1) \ldots (x_m, y_m)\}$ random sample
A $A\left(S\right) = h$ predictor output by A given S where A is learning algorithm as function of traning set S.
$\forall\, S$ $A\left(S\right) \in H$ $h^* \in H$

$\ell_D\left(h^*\right) = min\, \ell_D\left(h\right)$ $\hat{\ell}_s\left(h^*\right)$ *is closed to* $\ell_D\left(h^*\right)$ $\longrightarrow$ **it is going to have small error**
where $\ell_D\left(h^*\right)$ is the training error of h^*

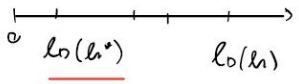


Figure 7.4: Example

This guy $\ell_D\left(h^*\right)$ is closest to 0 since optimum

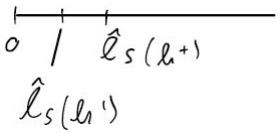


Figure 7.5: Example

In risk we get opt in h^* but in empirical one we could get another h' better than h^+

In order to fix on a concrete algorithm we are going to take the empirical Islam minimiser (ERM) algorithm.
A is ERM on H $(A) = \hat{h} = (\in)\, argmin\, \hat{\ell}_S\left(h\right)$
Once I piack $\hat{h}$ i can look at training error of ERM

$$\hat{\ell}_S\left(\hat{h}\right) of \hat{h} = A(S)$$

where $\hat{\ell}_S$ is the training error

Should $\hat{\ell}_S\left(\hat{h}\right)$ be close to $\ell_D\left(\hat{h}\right)$?
I'm interested in empirical error minimiser and do a trivial decomposition.

$$\begin{aligned}\ell_d\left(\hat{h}\right) = \quad & \ell_D\left(\hat{h}\right) - \ell_d\left(h^*\right) + && \longrightarrow \textbf{Variance error} \Rightarrow \textbf{Overfitting} \\ & + \ell_d\left(h^+\right) - \ell_d\left(f^*\right) + && \longrightarrow \textbf{Bias error} \Rightarrow \textbf{Underfitting} \\ & + \ell_D\left(f^*\right) && \longrightarrow \textbf{Bayes risk} \Rightarrow \textbf{Unavoidable}\end{aligned}$$

Even the best predictor is going to suffer that

$$f^* \textit{ is } \textbf{\textit{Bayes Optimal}} \textit{ for } (D, \ell)$$
$$\forall h \qquad \ell_D\left(h\right) \geq \ell_D\left(f^*\right)$$

If $f^* \notin H$ then $\ell_D\left(h^*\right) > \ell_D(f^*)$

If i pick h^* I will pick some error because we are not close enough to the risk.
We called this component **bias error**.
Bias error is responsible for underfitting (when training and test are close to each but they are both high :( )
Variance error over fitting

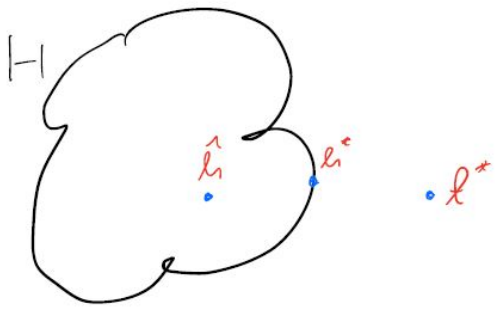


Figure 7.6: Draw of how $\hat{h}$, h^* and f^* are represented

Variance is a random quantity and we want to study this. We can always get risk from training error.

7.3 Studying overfitting of a ERM

We can bound it with probability.
I add and subtract trivial traning error $\hat{\ell}_S(h)$

$$\begin{aligned}\ell_D\left(\hat{h}\right) - \ell_d(h^*) &= \ell_D\left(\hat{h}\right) - \hat{\ell}_S(h) + \hat{\ell}_S\left(\hat{h}\right) - \ell_D(h^*) \leq \\ &\leq \ell_D\left(\hat{h}\right) - \hat{\ell}_S\left(\hat{h}\right) + \hat{\ell}_S(h^*) - \ell_D(h^*) \leq \\ &\leq |\,\ell_D\left(\hat{h}\right) - \hat{\ell}_S(h)\,| + |\,\hat{\ell}_S\left(h^+\right) - \ell_D(h^*)\,| \leq \\ &\leq 2 \cdot max\,|\hat{\ell}_S(h) - \ell_D(h)\,|\end{aligned}$$

(no probability here)
Any given $\hat{h}$ **minising** $\hat{\ell}_S(h)$

Now assume we have a large deviation

$$Assume \quad \ell_D\left(\hat{h}\right) - \ell_D(h^*) > \varepsilon \qquad \Rightarrow \qquad max\,|\,\hat{\ell}_S(h) - \ell_D(h)\,| > \frac{\varepsilon}{2}$$

We know $\ell_d\left(\hat{h}\right) - \ell_D(h^*) \leq 2 \cdot max\,|\,\hat{\ell}_S(h) - \ell_D(h)\,| \Rightarrow$

$$\Rightarrow \quad \exists h \in H \qquad |\,\hat{\ell}_S(h) - \ell_D(h)\,| > \frac{3}{2} \qquad \Rightarrow$$

with $|H| < \infty$

$$\Rightarrow U\left(|\,\hat{\ell}_S(h) - \ell_D(h)\,|\right) > \frac{3}{2}$$

$$\mathbb{P}\left(\ell_D\left(\hat{h}\right) - \ell_D(h^*) > \varepsilon\right) \leq \mathbb{P}\left(U\left(|\,\hat{\ell}_S(h) - \ell_D(h)\,|\right) > \frac{3}{2}\right) \leq$$

$$\leq \sum_{h \in H} \mathbb{P}\left(|\,\hat{\ell}_S(h) - \ell_D(h)\,| > \frac{3}{2}\right) \leq \sum_{h \in H} 2 \cdot e^{-2\left(\frac{\varepsilon}{2}\right)^2 m} \leq$$

Union Bound Chernoff. Hoffding bound ($\mathbb{P}(...)$)

$$\leq 2 \cdot |H| e^{-\frac{\varepsilon^2}{2} m}$$

Solve for $\varepsilon \qquad 2 \cdot |H| e^{-\frac{\varepsilon^2}{2} m} = \delta$

$$Solve\ for\ \varepsilon \longrightarrow \quad \varepsilon = \sqrt{\frac{2}{m} \cdot \ln \cdot \frac{2|H|}{\delta}}$$

$$\ell_D\left(\hat{h}\right) - \ell_D\left(h^*\right) \quad \leq \quad \sqrt{\frac{2}{m} \cdot \ln \cdot \frac{2|H|}{\delta}}$$

With probability at least $1 - \delta$ with respect to random draw of S.
We want $m >> ln|H|$ $\longrightarrow$ in order to avoid overfitting

Lecture 8 - 07-04-2020

$$|H| < \infty \qquad \hat{h} = argmin\, \hat{\ell}_S(h) \qquad h^* = argmin\, \ell_D(h)$$

minimise risk

Bias-Variance decomposition

$$\ell_d\left(\hat{h}_S\right) = \ell_D\left(\hat{h}_S\right) - \ell_d(h^*) + \longrightarrow \textbf{Variance error} \Rightarrow \textbf{Overfitting}$$

$$+\ell_d\left(h^+\right) - \ell_d(f^*) + \longrightarrow \textbf{Bias error} \Rightarrow \textbf{Underfitting}$$

$$+\ell_D(f^*) \longrightarrow \textbf{Bayes risk} \Rightarrow \textbf{Unavoidable}$$

We state this for all algorithm but we studied for ERM.

$$\ell_D\left(\hat{h}_S\right) \leq \ell_D(h^*) + \sqrt{\frac{2}{m} \ln \frac{2|H|}{\delta}} \qquad \textit{with probability at least } 1-\delta \textit{ over the draw of } S$$

we want **this** to be small when $m >> \ln|H|$

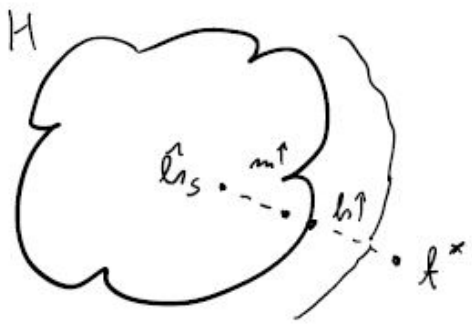


Figure 8.1: Representation of $\hat{h}$, h^* and f^*

Size of model fix, increase m (size of sample) when m is bigger $\longrightarrow$ variance goes down.
When $|H| \longrightarrow$ big model will be closer to optimal

- m grows $\Rightarrow$ variance error goes down (if not overfitting)
- $|H|$ grows $\Rightarrow$ bias error goes down (if not underfitting)

ERM with $|H| < \infty$
A H such that $\forall s\ A(S) \in H$

We controlled this event:

$$\forall h \in H \qquad \hat{\ell}_S(h) - \ell_D(h) \quad \leq \quad \sqrt{\frac{2}{m} \ln \frac{2|H|}{\delta}} \qquad \textit{with probability at least } 1-\delta$$

We assure that training error is a good proxy for the true risk.

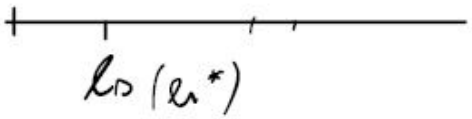


Figure 8.2: Example

If I do the empirical way(for a specific training set) i get something different.

8.1 The problem of estimating risk in practise

Test error is a good estimate for a risk, provided $\frac{1}{\sqrt{n}}$ is small

It's usually good to take a small test set.
80/20 big data, if low data better to look at good estimate...=??
Usually small test set is still ok

Typically we are not given the predictor, instead we start from a learning algorithm. It's true that A has parameter (how many nodes for a tree classifier for example).
In general, if I have parameters then i get a set of different classifier Imaging Algorithm that has parameter θ:

$$A \qquad \{A_\theta : \theta \in \Theta\} \qquad A_\theta(S) = \hat{h}_S$$

$\mathbb{E}[\ell_D(A_\theta(S))]$ **typically averaged over S of size m for fixed m**

In general you may also be interest looking at the best choice of parameter for your algorithm: Suppose now that i want to look at mini sing the risk

with the respect to the choice of the parameter

$\mathbb{E}\left[min\,\ell_D\left(A_\theta\left(S\right)\right)\right]$ i want to choose parameter that minimise risk running algorithm on training set.

I would like to choose best possible value for my parameter k in the k-NN and i want to estimate the risk.
θ can be a set of parameters (so more than 1 parameter), so theta are the **hyper parameters of A**.
In general this parameter are the choices that algorithm that can make: parameter that define a classifier (example nodes and test in the internal nodes and label in the leaves)

Hyper parameters are not determined by the training set $\rightarrow$ chosen before the training set.
I fix k for K_{NN} and I choose an upper bound of number of nodes after the training set is given.
So, some parameters are given after training set is given and some parameter fixed before.
Now the idea is that parameter are determined after training set is given, while hyper parameters are given before training set to get then a predictor.
I have a family of algorithms, so I choose a hyper parameters to get a one algorithm.
Most algorithm are given as family. We have first decide hyper parameters not determined on the training set.
One way to move is to take you dataset and the split that in 3 parts:

- Training set
- Development (or validation) test
- Test sets

There should not be a leak of information between test and train and development.
We get family of algorithm, train with train set and then use dev set to test the parameter and choice the parameters. Once i found the parameter I retrain the algorithm with a part of train and dev set to being then tested.
Development set is like a fake test set $\rightarrow$ it usefull to choose parameters.
Algorithm steps:

1. Train A_θ on training set for each $\theta \in \Theta$ (**grid search**)

2. Find θ such that $\hat{h} = A_{\hat{\theta}}(S)$ minimise developement error
3. Train $A_{\hat{\theta}}$ on training + development set
4. Test resulting predictor on test set

(There's theory about this but it's difficult and we are not going to do that)

It's heuristic and kinda simple to do. This technics work for every learning algorithm.
One parameter: grid on this
Two parameters ecc..
It's quadratic!
Good learning algorithm should have small number of hyperparameters.

8.2 Cross-validation

Solving an easier problem: suppose you have a dataset and what you do is that you can choose training set and test set.
Algorithm A with no hyper parameters and we would like to check how good is the predictor: how good can A be? A is good if the predictor that generate has low risk.
I split dataset in training and test set.

Figure 8.3: Splitting test and training set

$$h = A(A) \qquad \hat{\ell}_{S'}(h) \approx \ell_D(h)$$

I can use Cross-validation! (CV). It helps estimate the risk.

$$\textbf{CV:} \qquad \mathbb{E}\left[\ell_D(A(S))\right]$$

I would like to average ....
How do i do this estimate? Super easy, take my data assuming CV as parameter not of the algorithm A, but intrinsic to cross validation.

Parameter k-fold CV typically $k = 5$ or 10.
What do you do ? If i take a specific training set, i got a ruff estimate of A.
I shouldn't split one but several time.
There are different way but he give us another:
split data in k folds randomly!

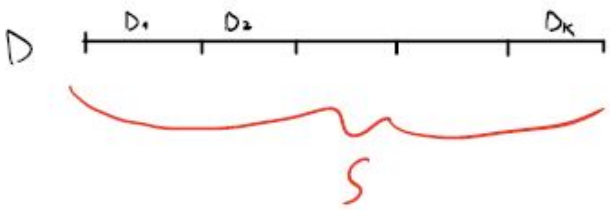


Figure 8.4: K-folds

For each k we define $S^{(k)}$ **take out k-st D**

$$S^{(k)} = S \backslash D_k$$

$$S^{(1)} = D_2 \cup D_3 \cup ... \cup D_k$$
$$S^{(2)} = D_1 \cup D_3 \cup ... \cup D_k$$

For each $k - 1, ...k$ **folds** $S^{(k)}$ training part and D_k test part

$$h_K = A\left(S^{(k)}\right) \qquad \hat{\ell}_{DK}\left(h_K\right) = \frac{k}{m} \sum_{(x,y) \in D_k} \ell\left(y, h_K\left(x\right)\right)$$

Repeat the procedure for $k = 1...k$ get $h_1, ..., h_k$

$$Compute \qquad \frac{1}{k} \cdot \sum_{k=1}^{K} \hat{\ell}_{DK}\left(h_k\right)$$

where CV is $\mathbb{E}\left[\ell_D\left(A\left(S\right)\right)\right]$ and this is called ..... —MANCA — Estimate

It's used a lot!
You get data from internet, then what to do? I want to try an algorithm, try K_{NN} so you do Cross validation and will give you the risk.
In some other cases you get a splitted dataset in training and testing set. You don't use CV since the dataset is already splitted in training and test.

8.3 Nested cross validation

The use of CV to solve the hyper parameters choice problem. If you are given test and training set you can solve splitting in training set and dev set. Suppose not given train and test, you can assign arbitrary .....
Now the idea: you give me a way to optimise splitting training set and test set and then avoiding using a ...
This is what cross validation is doing.

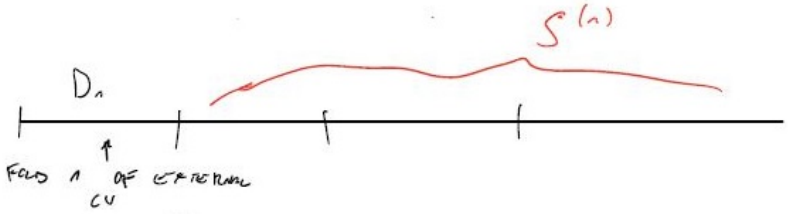


Figure 8.5: Nested Cross Validation

$\{A_\theta : \theta \in \Theta\}$ which i run in each fold?
The idea is **running interval CV for the folds**
I have fold D_1 and $S(1)$ and i perform external validation, then i run internal CV in $S(1)$.
On each training part of the external CV, run a internal CV for each A_θ.
When $\theta \in$ grid (Θ):

- I have a CV-estimate and for each A_θ pick the θ with best CV-estimate
- Run $A_{\hat{\theta}}$ on the entire training part of current external fold

Basically what I'm choosing the best hyper parameter on each fold of the external CV.
External CV is not testing $A_{\hat{\theta}}$ for a given $\hat{\theta} \in \Theta$
I am not measuring a goodness of predictor generate by algorithm for given value of hyper parameters but what I'm estimating is the average risk of the predictor output by learning algorithm when hyper parameters are optimise on the training set. This optimisation on training set is done into a internal CV. To avoid be depend i run and external CV.
Many platform like sklearn allow you to do that in two lines of code. So i can specify the grid, predictor, number of falls for internal and external. It took a bit but that's it in in two lines of code
Cross validation can be done in every order of D_1 or D_2 or D_k. So doesn't matter the order we start fold D2 or D1.

Lecture 9 - 07-04-2020

$\hat{h}$ is ERM predictor

$$\ell_D\left(\hat{h}\right) \leq min\ \ell_D\left(h\right) + \sqrt{\frac{2}{m}\ \ln\ \frac{2\,H}{\delta}} \qquad with\ prob.\ at\ least\ 1-\delta$$

Now we do it with tree predictors

9.1 Tree predictors

$$X = \{0,1\}^d \longrightarrow \textbf{Binary classification}$$
$$h : \{0,1\}^d \longrightarrow \textbf{Binary classification H}1$$

How big is this class?
Take the size of codomain power the domain $\longrightarrow |H| = 2^{2^d}$
Can we have a tree predictor that predict every H in this class?
For every $h : \{0,1\}^d \longleftrightarrow \{-1,1\} \quad \exists T$

We can **build a tree** such that $h_T = h$

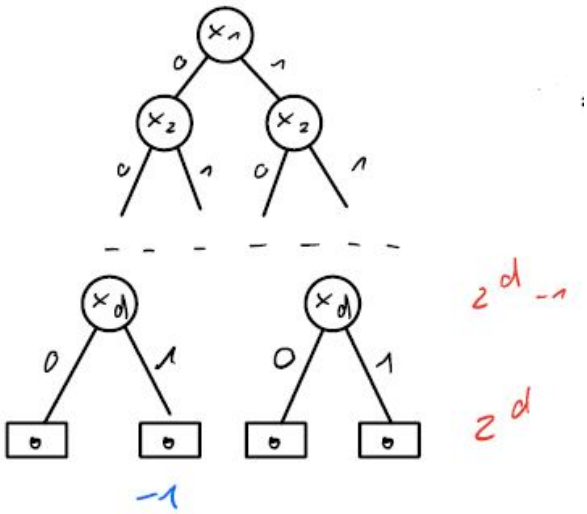


Figure 9.1: Tree building

$X = (0,0,1,...,1) \qquad h\left(x\right) = -1$

$x_1, x_2, x_3, ..., x_d$

I can apply my analisys to this predictors
If I run ERM on H

$$\ell_D\left(\hat{h}\right) \leq min\, \ell_D\left(\hat{h}\right) + \sqrt{\frac{2}{m} 2^d \ln 2 + \ln \frac{2}{\delta}} \qquad \longrightarrow \ln |H| + \ln \frac{2}{\delta}$$

No sense! What we find about training set that we need?
Worst case of overfitting $m >> 2^D = |X| \Rightarrow$ training sample larger

PROBLEM: cannot learn from a class to big (H is too big)
I can control H just limiting the number of nodes.

$H_N \longrightarrow$ tree T with at most N node, $N << 2^D$
$|H_N| = ?$

$|H_N| = (\#\ of\ trees\ with\ \leq N\ nodes) \times (\#\ of\ test\ on\ interval\ nodes\ ) \times (\ \#\ labels\ on\ leaves)$

$$|H_N| = \bigotimes \times d^M \times 2^{N-M}$$

N of which $N - M$ are leaves

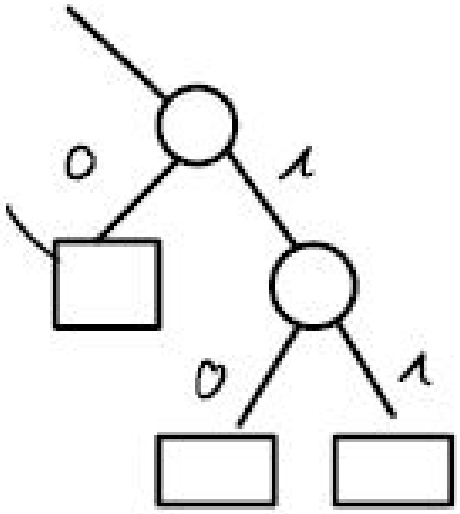


Figure 9.2: Tree with at most N node

$\bigotimes \#\ of\ binary\ trees\ with\ N\ nodes,\ called\ \textbf{Catalan Number}$

9.1.1 Catalan Number

*We are using a binomial *

$$\frac{1}{N}\binom{2N-2}{N-1} \quad \leq \quad \frac{1}{N}\left(e\,\frac{(2N-2)}{N-1}\right)^{N-1} = \frac{1}{N}\,(2e)^{N-1}$$

$$\binom{N}{K} \quad \leq \quad \left(\frac{e\,n}{k}\right)^{k} \qquad \textit{from Stirling approximation}$$

Counting the number of tree structure: a binary tree with exactly N nodes. Catalan counts this number. $\longrightarrow$ **but we need a quantity to interpret easily**. So we compute it in another way.
Now we can rearrange everything.

$$|H_N| \quad \leq \quad \frac{1}{N}\,(2e)^{N-1}\,H^M\,2^{N-M} \quad \leq \quad (2\,e\,d)^N$$

$$d \geq 2 \qquad \leq d^{N-M}$$

where **we ignore** $\frac{1}{N}$ **since we are going to use the** log

ERM on $H_N \quad \hat{h}$

$$\ell_D\left(\hat{h}\right) \leq \min_{\mathbf{h}\in\mathbf{H_N}} \ell_D\left(h\right) + \sqrt{\frac{2}{m}\left(N\cdot(1+\ln(2\cdot d)) + \ln\frac{2}{\delta}\right)}$$

were $N\cdot(1+\ln(2\cdot d)) \quad = \quad \ln(H_N)$

In order to not overfit $m >> N\cdot\ln d$
$N\cdot\ln d << 2^d$ for reasonable value of N
We grow the tree and a some point we stop.

$$\ell_D\left(h\right) \leq \hat{\ell}_S\left(h\right) + \varepsilon \qquad \forall h \in H_N \qquad \textit{with probability at least } 1-\delta$$

remove N **in** H_N **and include** h **on** ε
we remove the N index in H_N adding h on ε

$$\ell_D\left(h\right) \leq \hat{\ell}_S\left(h\right) + \varepsilon_h \qquad \forall h \in H_{\not N}$$

$$W : H \longrightarrow [0,1] \qquad \sum_{h\in H} w\left(h\right) \leq 1$$

How to use this to control over risk?

$$\mathbb{P}\left(\exists h \in H \;:\; |\,\hat{\ell}_S\left(h\right) - \ell_D\left(h\right)\,| \;> \varepsilon_h\right) \quad \leq$$

where $\hat{\ell}_S$ is the prob my training set cases is true

$$\leq \sum_{h \in H} \mathbb{P}\left(\mid \hat{\ell}_S(h) - \ell_D(h) \mid > \varepsilon_h \right) \leq \sum_{h \in H} 2\, e^{-2\, m\, \varepsilon\, h^2} \leq$$

$$\leq \delta \qquad \longrightarrow \textit{since } w(h) \textit{ sum to } 1 \left(\sum_{h \in H} \right)$$

I want to choose $2\, e^{-2\, m\, \varepsilon\, h^2} = \delta\, w(h)$

$$2\, e^{-2\, m\, \varepsilon\, h^2} = \delta\, w(h) \qquad \Leftrightarrow \qquad \textit{— MANCA PARTEEEE —}$$

therefore:

$$\ell_D(h) \leq \hat{\ell}_S(h) + \sqrt{\frac{1}{2\, m} \cdot \left(\ln \frac{1}{w(h)} + ln \frac{2}{\delta} \right)} \quad \textit{w. p. at least } 1 - \delta \quad \forall h \in H$$

Now, instead of using ERM we use

$$\hat{h} = arg \min_{h \in H} \left(\hat{\ell}_S(h) + \sqrt{\frac{1}{2\, m} \cdot \left(\ln \frac{1}{w(h)} + ln \frac{2}{\delta} \right)} \right)$$

where $\sqrt{...}$ term is the penalisation term

Since our class is very large we add this part in order to avoid overfitting. Instead of minimising training error alone i minimise training error + penalisation error.

In order to pick w(h) we are going to use **coding theory**
The idea is I have my trees and i want to encode all tree predictors in H using strings of bits.

$\sigma : H \longrightarrow \{0,1\}^*$ **coding function for trees**
$\forall\, h, h' \in H$ $\sigma(h)$ not a prefix of $\sigma(h')$
$h \neq h'$ where $\sigma(h)$ and $\sigma(h')$ are **string of bits**

σ is called **istantaneous coding function**
Istantaneous coding function has a property called **kraft inequality**

$$\sum_{h \in H} 2^{-|\,\sigma(h)\,|} \leq 1 \qquad w(h) = 2^{-|\,\sigma(h)\,|}$$

I can design $\sigma : H \longrightarrow \{0,1\}^*$ *istantaneous* $|\,\sigma(h)\,|$

$$\ln |H_N| = O\left(N \cdot \ln d\right)$$

number of bits i need = number of node in h

Even if i insist in istantaneous i do not lose ... – MANCA PARTE –

$$|\,\sigma(h)\,| = O\left(N \cdot \ln d\right)$$

Using this σ and $w(h) = 2^{-|\,\sigma(h)\,|}$

$$\ell_D\left(h\right) \leq \hat{\ell}_S\left(h\right) + \sqrt{\frac{1}{2\,m} \cdot \left(c \cdot N \cdot \ln d + \ln \frac{2}{\delta}\right)} \qquad w.\ p.\ at\ least\ 1-\delta$$

where c is a constant

$$\hat{h} = arg \min_{h \in H} \left(\hat{\ell}_S\left(h\right) + \sqrt{\frac{1}{2\,m} \cdot \left(c \cdot N \cdot \ln d + \ln \frac{2}{\delta}\right)} \right)$$

where $m >> N \cdot h \cdot \ln d$
If training set size is very small then you should not run this algorithm.

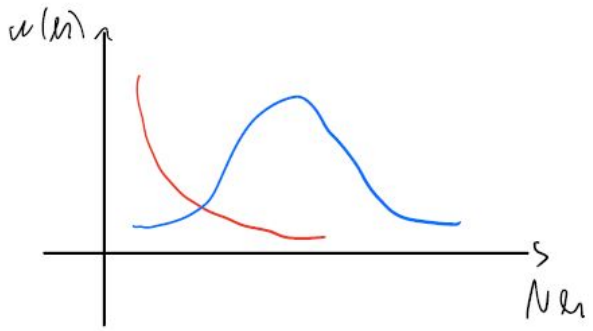


Figure 9.3: Algorithm for tree predictors

This blue curve is an alternative example. We can use Information criterion.

As I increase the number of nodes, N_h decrease so fast. You should take a smaller tree because it gives you a better bound. It's a principle known as Occam Razor (if I have two tree with the same error, if one is smaller than the other than i should pick this one).

Having N^*

Lecture 10 - 07-04-2020

10.1 TO BE DEFINE

$|E[z] = |E[|E[z|x]]$

$|E[X] = \sum_{t=1}^{m} |E[x\Pi(At)]$

$x \in \mathbb{R}^d$
$\mathbb{P}(Y_{\Pi(s,x)} = 1) =$

$\mathbb{E}[\Pi Y_{\Pi(s,x)} = 1] =$

$= \sum_{t=1}^{m} \mathbb{E}[\Pi\{Y_t = 1\} \cdot \Pi Pi(s,x) = t] =$

$= \sum_{t=1}^{m} \mathbb{E}[\mathbb{E}[\Pi\{Y_t = 1\} \cdot \Pi\{\Pi(s,x) = t\}|X_t]] =$

$given the fact that Y_t \sim \eta(X_t) \Rightarrow give me probability$
$Y_t = 1 and \Pi(s,x) = t are independent given X_Y (e.g. \mathbb{E}[Zx] = \mathbb{E}[x] * \cdot \mathbb{E}[z]$

$= \sum_{t=1}^{m} \mathbb{E}[\mathbb{E}[\Pi\{Y_t = 1\}|X_t] \cdot \mathbb{E}[\Pi(s,x) = t|Xt]] =$

$= \sum_{t=1}^{m} \mathbb{E}[\eta(X_t) \cdot \Pi \cdot \{\Pi(s,x) = t\}] =$

$= \mathbb{E}[\eta(X_{\Pi(s,x)}]$

$$\mathbb{P}(Y_{\Pi(s,x)}|X = x = \mathbb{E}[\eta(X_{\Pi}(s,x))]$$

$\mathbb{P}(Y_{\Pi(s,x)} = 1, y = -1) =$

$= \mathbb{E}[\Pi\{Y_{\Pi(s,x)} = 1\}\dot{\Pi}\{Y = -1|X\}]] =$

$= \mathbb{E}[\Pi\{Y_{\Pi(s,x)} = 1\} \cdot \Pi\{y = -1\}] =$

$= \mathbb{E}[\mathbb{E}[\Pi\{Y_{\Pi(s,x)} = 1\} \cdot \Pi\{y = -1|X\}]] =$

$$Y_{\Pi(s,x)} = 1 \qquad y = -1(1 - \eta(x)) \quad when \quad X = x$$

$$= \mathbb{E}[\mathbb{E}[\Pi\{Y_\Pi(s,x)\} = 1|X] \cdot \mathbb{E}[\Pi\{y = -1\}|X]] =$$

$$= \mathbb{E}[\eta_{\Pi(s,x)} \cdot (1 - \eta(x))] =$$

$$similarly: \quad \mathbb{P}(Y_{\Pi(s,x)} = -1, y = 1) =$$
$$\mathbb{E}[(1 - \eta_{\Pi(s,x)}) \cdot \eta(x)]$$

$$\mathbb{E}[\ell_D(\hat{h}_s)] = \mathbb{P}(Y_{\Pi(s,x)} \neq y) =$$

$$= \mathbb{P}(Y_{\Pi(s,x)} = 1, y = -1) + \mathbb{P}(Y_{Pi(s,x)} = -1, y = 1) =$$

$$= \mathbb{E}[\eta_{\Pi(s,x)} \cdot (1 - eta(x))] + \mathbb{E}[(1 - \eta_{\Pi(s,x)}) \cdot \eta(x)]$$

Make assumptions on $D_x \quad and \quad \eta$:

MANCAAAAAAA ROBAAA

$$\eta(x') <= \eta(x) + c||X - x'|| - - > euclideandistance$$

$$1 - \eta(x') <= 1 - \eta(x) + c||X - x'||$$

$$X' = X_{Pi(s,x)}$$

$$\eta(X) \cdot (1 - \eta(x')) + (1 - \eta(x)) \cdot \eta(x') <=$$

$$<= \eta(x) \cdot ((1 - \eta(x)) + \eta(x) \cdot c||X - x'|| + (1 - \eta(x)) \cdot c||X - x'|| =$$

$$= 2 \cdot \eta(x) \cdot (1 - \eta(x)) + c||X - x'||$$

$$\mathbb{E}[\ell_d \cdot (\hat{h}_s)] <= 2 \cdot \mathbb{E}[\eta(x) - (1 - \eta(x))] + c \cdot (E)[||X - x_{\Pi(s,x)}||]$$
where $<=$ mean at most

Compare risk for zero-one loss
$\mathbb{E}[min\{\eta(x), 1 - \eta(x)\}] = \ell_D(f*)$

$\eta(x) \cdot (1 - \eta(X)) <= min\{\eta(x), 1 - eta(x)\} \quad \forall x$

$\mathbb{E}[\eta(x) \cdot (1 - \eta(x)] <= \ell_D(f*)$

$\mathbb{E}[\ell_d(\hat{l}_s)] <= 2 \cdot \ell_D(f*) + c \cdot \mathbb{E}[||X - X_{\Pi(s,x)}||]$

$\eta(x) \in \{0, 1\}$

Depends on dimension: curse of dimensionality

–DISEGNO–

$\ell_d(f*) = 0 \iff min\{\eta(x), 1 - \eta(x)\} = 0$ with probability = 1
to be true $\eta(x) \in \{0, 1\}$

Bibliography