Machine Learning

Prof. Victor Adamchik

University of Southern California

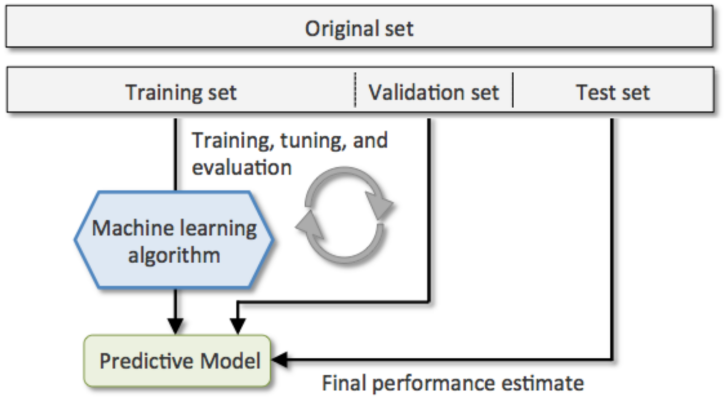
July 2, 2020

Outline

Outline

Data Sets

ML depends heavily on data, without data, it is impossible to learn. If your data set is not good enough, your entire project will fail. In every ML projects, classifying and labeling data sets takes most of our time.



Data Sets

- Training data set

  It is used to train an algorithm. It includes both input data and the expected output.

- Test data set

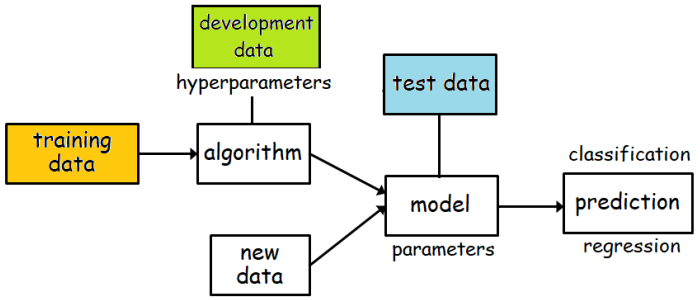
  It is used to evaluate how well your algorithm was trained with the training data set.

- Validation data set

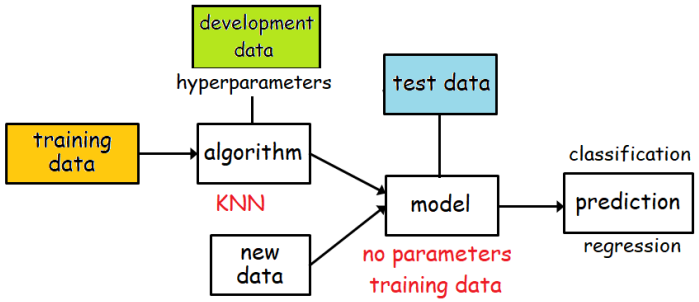
  It is used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The model never learns from this set. So, the validation set affects a model indirectly.

Supervised Learning

Supervised learning is a process of creating a model using data (labeled with desired outputs) to make a prediction on new data.

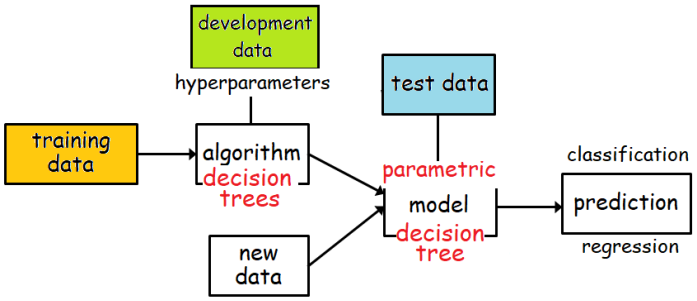


KNN Algorithm



Outline

Decision Trees



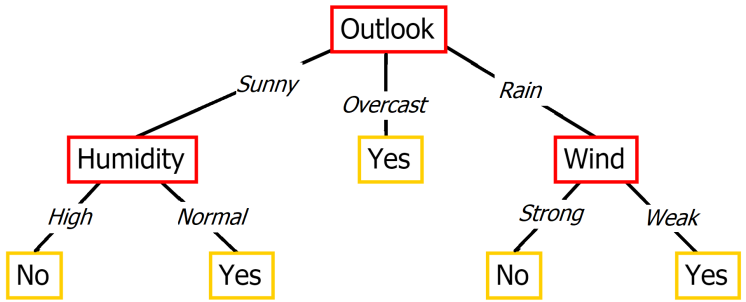
Reading: ESL chapters 9.2, 8.7, 15.

Decision trees

Decision tree is another ML model for classification and regression:

- the learned function is represented by a decision tree.
- also can be represented as sets of if-then rules.
- simple, popular, successfully applied to a broad range of tasks.

Example: playing tennis



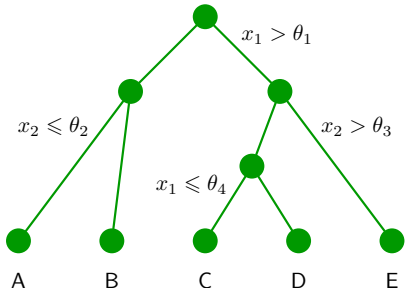
- Each node in the tree specifies a test of some attribute (feature)
- Each branch from a node corresponds to one of the possible values for this attribute

A more abstract example of decision trees with five classes

Input: $\boldsymbol{x} = (x_1, x_2)$, assume just two features

Output: $f(\boldsymbol{x})$ determined naturally by traversing the tree

- start from the root
- test at each node to decide which child to visit next
- finally the leaf gives the prediction $f(\boldsymbol{x})$

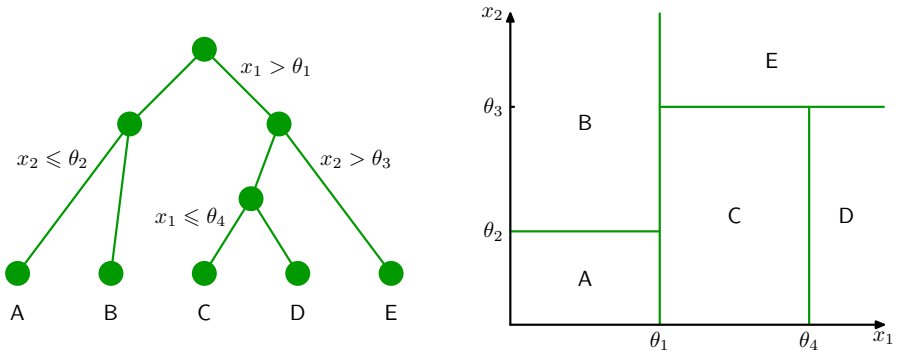


For example, $f((\theta_1 - 1, \theta_2 + 1)) = \text{B}$

Complex to formally write it down, but easy to represent pictorially or to code.

The decision boundary

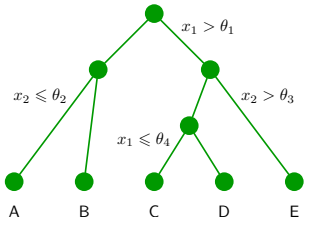
Corresponds to a classifier with boundaries:



Parameters

Parameters to learn for a decision tree:

- the structure of the tree, such as the depth, #branches, #nodes, etc
  - the structure of a tree is *not fixed in advance, but learned from data*
  - some of them are sometimes considered as hyperparameters
- the test at each internal node
  - which feature(s) to test on?
  - categorical vs. continuous attributes.
  - if the feature is continuous, what threshold $(\theta_1, \theta_2, \ldots)$?



- the value/prediction of the leaves (A, B, ...)

Learning the parameters

Given a set of examples (training set), the task is to construct an optimal decision tree.

Using the resulting decision tree, we want to classify new instances of examples.

The typical ML approach is to find the parameters that minimize some loss.

This is unfortunately *not feasible for trees*

- suppose there are Z nodes, there are roughly $\#\text{features}^Z$ different ways to decide
- enumerating all these configurations to find the one that minimizes some loss is too computationally expensive.

Instead, we turn to a **greedy approach**, which will create a sub-optimal tree.

A running example

[Russell & Norvig, AIMA]

- 12 examples
- predict whether a customer will wait for a table at a restaurant
- 10 features (all discrete)

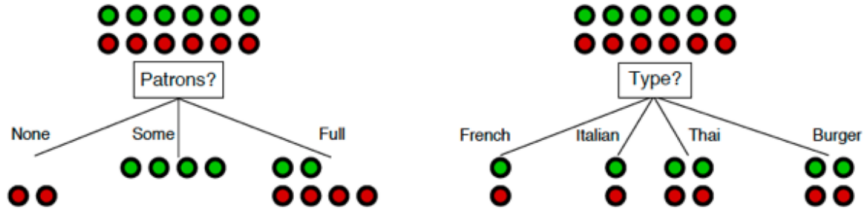
Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

List of attributes

- Alternate: whether there is a suitable alternative restaurant nearby.
- Bar. whether the restaurant has a comfortable bar area to wait in.
- Fri/Sat: true on Fridays and Saturdays.
- Hungry: whether we are hungry.
- Patrons: how many people are in the restaurant (values are None, Some, and Full).
- Price: the restaurant's price range ($, $$, $$$).
- Raining: whether it is raining outside.
- Reservation: whether we made a reservation.
- Type: the kind of restaurant (French, Italian, Thai, or Burger).
- WaitEstimate: the wait estimated by the host (0-10 minutes, and so).

First step: how to build the root?

I.e., which feature should we test at the root? Examples:



Which split is better?

- intuitively "patrons" is a better feature since it leads to "more pure" or "more certain" children
- How to quantitatively measure which one is better?

Choosing the Best Attribute

Use Shannon's information theory to choose the attribute that gives the smallest *entropy* that is defined by:

$$H(P) = -\sum_{k=1}^{\mathsf{C}} P(Y=k)\log P(Y=k)$$

- Entropy is always non-negative
- Entropy is a measure of impurity (disorder).
- maximized if P is uniform ($H(P) = \log \mathsf{C}$): most uncertain case

$$H(P) = -\sum_{k=1}^{\mathsf{C}} \frac{1}{\mathsf{C}}\log\frac{1}{\mathsf{C}} = \mathsf{C}\times\frac{1}{\mathsf{C}}\log\mathsf{C} = \log\mathsf{C}$$

- minimized if P focuses on one class ($H(P) = 0$): most certain case
  - $0\log 0$ is defined naturally as $\lim_{z\to 0+} z\log z = 0$

Examples of entropy

H = 2bits *H = 1.0bits* *H = 0.3bits* *H = 0.0bits*

$$H(P) = -\frac{1}{16}\log\frac{1}{16} - \frac{15}{16}\log\frac{15}{16} = 0.34$$

Our greed (in the greedy approach) is to minimize entropy.

Restaurant example

Entropy of each child if root tests on "patrons"

For "None" branch

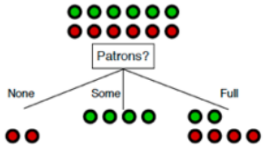
$$-\left(\frac{0}{0+2}\log\frac{0}{0+2}+\frac{2}{0+2}\log\frac{2}{0+2}\right)=0$$

For "Some" branch

$$-\left(\frac{4}{4+0}\log\frac{4}{4+0}+\frac{0}{4+0}\log\frac{0}{4+0}\right)=0$$

For "Full" branch

$$-\left(\frac{2}{2+4}\log\frac{2}{2+4}+\frac{4}{2+4}\log\frac{4}{2+4}\right)\approx 0.9$$



So how good is choosing "patrons" overall?

Very naturally, we take the weighted average of entropy:

$$\frac{2}{12}\times 0+\frac{4}{12}\times 0+\frac{6}{12}\times 0.9=0.45$$

Restaurant example

Entropy of each child if root tests on "type"

For "French" branch

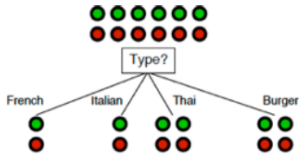
$$-\left(\frac{1}{1+1}\log\frac{1}{1+1}+\frac{1}{1+1}\log\frac{1}{1+1}\right)=1$$

For "Italian" branch

$$-\left(\frac{1}{1+1}\log\frac{1}{1+1}+\frac{1}{1+1}\log\frac{1}{1+1}\right)=1$$

For "Thai" and "Burger" branches

$$-\left(\frac{2}{2+2}\log\frac{2}{2+2}+\frac{2}{2+2}\log\frac{2}{2+2}\right)=1$$

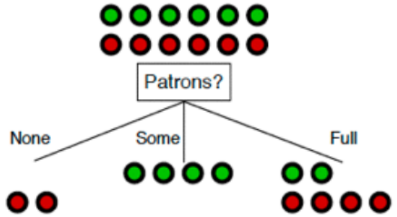


The weighted average of entropy is
$\frac{2}{12}\times 1+\frac{2}{12}\times 1+\frac{4}{12}\times 1+\frac{4}{12}\times 1=1>0.45$
Pick the feature that leads to the smallest entropy.
So splitting with "patrons" is better than splitting with "type".
We are now done with building the root.

Repeat recursively

Split each child in the same way.

- but no need to split children "none" and "some": they are pure already and become leaves
- for "full", repeat, focusing on those 6 examples:



	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	WillWait
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Putting it together: the ID3 algorithm

DecisionTreeLearning(Examples, Features)

- if Examples have the same class, return a leaf with this class
- else if Features is empty, return a leaf with the majority class
- else if Examples is empty, return a leaf with majority class of parent
- else

  find the best feature A to split (e.g. based on conditional entropy)

  Tree ← a root with test on A

  For each value a of A:

  Child ← **DecisionTreeLearning**(Examples with $A=a$, Features$-\{A\}$)
  add **Child** to **Tree** as a new branch
- return Tree

Variants

Popular decision tree algorithms (e.g. C4.5, CART, etc) are all based on this framework.

Variants:

- replace entropy by **Gini impurity**:

$$G(P) = \sum_{k=1}^{C} P(Y = k)(1 - P(Y = k))$$

is used to provide an indication of how "pure" the leaf nodes are.

Overfitting

Some reasons for overfitting:

- Large number of attributes
- Too little training data
- Many kinds of "noise"(same feature but different classes),values of attributes are incorrect , classes are incorrect)

How can we avoid overfitting?

- Stop growing when you reach some depth or number of nodes
- Stop growing when data split is not statistically significant
- Acquire more training data
- Remove irrelevant attributes
- Grow a full tree, then **prune** it

Reduced-Error Pruning

Pruning is done by replacing a whole subtree by a leaf node and assigning the most common class to that node.

Split data into training and validation sets.

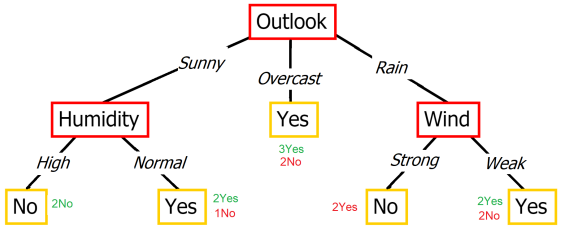
Grow a full tree based on training set.

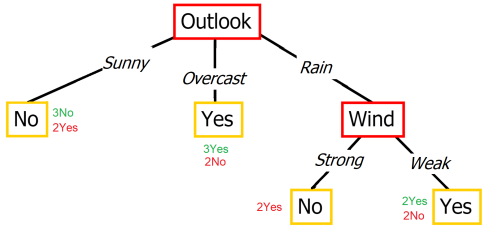
Do pruning until it is harmful:

- Evaluate impact on validation set of pruning each possible node.
- Greedily remove the node that most improves validation set accuracy.

Accuracy is the number of correct predictions made divided by the total number of predictions made.

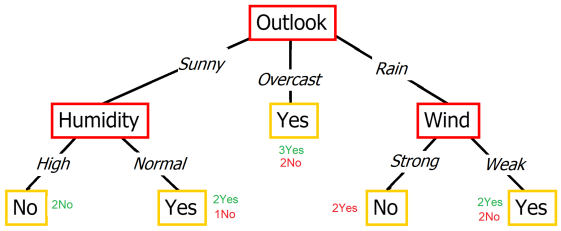
Reduced-Error Pruning: Example



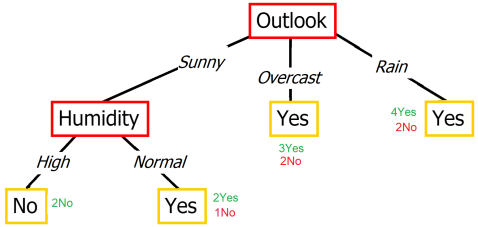


- Errors = $\frac{7}{16}$
- Let us remove Humidity.
- We will assign No, to this node, since there are three No examples.
- New errors = $\frac{8}{16}$
- Do not prune it!

Reduced-Error Pruning: Example



- Errors $= \frac{7}{16}$
- Let us remove Wind.
- We will assign Yes, to this node, since there are four Yes examples.
- New errors $= \frac{5}{16}$
- Prune it!



Outline

Problem 1

Compute Gini Index for both cases

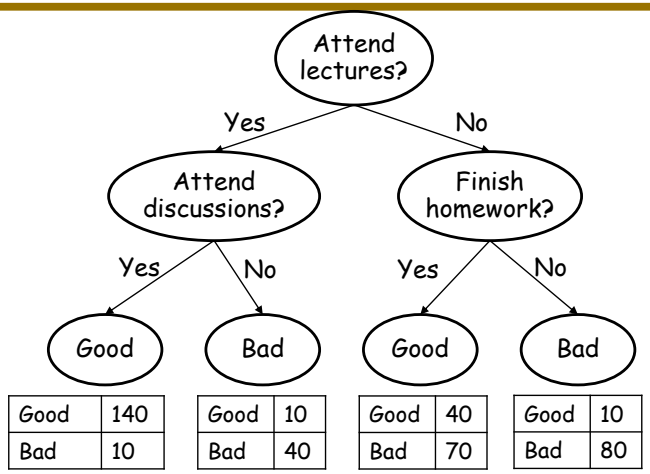
Problem 2

Name	Hair	Height	Weight	Lotion	Result
Sarah	blonde	average	light	no	sunburned
Dana	blonde	tall	average	yes	none
Alex	brown	short	average	yes	none
Annie	blonde	short	average	no	sunburned
Emily	red	average	heavy	no	sunburned
Pete	brown	tall	heavy	no	none
John	brown	average	heavy	no	none
Katie	blonde	short	light	yes	none

1) Which attribute will be best for the root of the tree? Assume any attribute other than name. Don't do any computation. Instead, try to make an estimate.

2) Using the best test at the root, finish the tree.

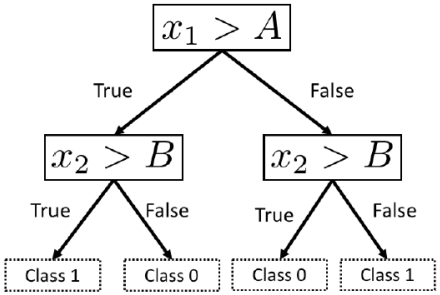
Problem 3



Using REP, which subtree will you prune and why?

Problem 4

Assume a decision tree below



A and B are integers and each x_1 and x_2 are also integers excluding A and B. Can this decision tree be implemented as a 1-NN?
Either explain why it can be or provide a counter example.

Outline

Multiple trees

- A single decision tree does not perform well
- However, the decision tree method is super fast
- Can we create an ensemble of decision trees?
- How do we learn multiple trees?
- If we split the data randomly in different ways, decision trees give different results (high variance)

Bagging

For $t = 1, \ldots, T$

- Randomly sample with replacement N training examples
- Train a single decision tree (no pruning)

To make a prediction, we aggregate the predictions from T decision trees and either

- Take the majority vote (for classification)
- Take the average (for regression)

Random Forest

Bagged decision trees are not yet a Random Forest. We will add another parameter that controls the number of features for each tree.

Suppose we had a dataset with P features. Instead of trying all features for each decision node, we only try a subset — a random sample of M predictors for each tree, at each split.

Note that if $M = P$, then this is bagging.

We do this primarily to inject randomness that makes individual trees more unique and reduces correlation between trees.

Random Forest Tuning

In theory, each tree in the random forest is full, but in practice this can be computationally expensive

The inventors make the following recommendations:

- For classification, the default value for M is $\sqrt{P}$ and the minimum leaf node size is one.
- For regression, the default value for M is $P/3$ and the minimum leaf node size is five.

Outline

Bayes classifier

Today we consider a probabilistic model for classification called Naïve Bayes.

Suppose the data $(\boldsymbol{x}_n, y_n)$ is drawn from a joint distribution $P(x, y = c)$, the **Bayes classifier** is

$$f(\boldsymbol{x}) = \underset{c \in [\mathsf{C}]}{\operatorname{argmax}} P(y = c \mid \boldsymbol{x})$$

i.e. predict the class with the largest conditional probability.

Naïve Bayes $P(x, y = c)$ is a *generative model*.

Bayes rule

Recall the **Bayes rule** is

$$P(y = c \mid \boldsymbol{x}) = \frac{P(\boldsymbol{x} \mid y = c)P(c)}{P(\boldsymbol{x})}$$

How does it help us?

We can use a conditional independency for $P(\boldsymbol{x} \mid y = c)$.

A "naive" assumption

Naive Bayes assumption: **the x_d are conditionally independent given y**, which means

$$P(\boldsymbol{x} \mid y = c) = \prod_{d=1}^{\mathsf{D}} P(x_d \mid y = c)$$

This reduces complexity to linear.

Is this a reasonable assumption? Sometimes yes.

More often this assumption is *unrealistic and "naive"*, but still Naive Bayes can work very well even if the assumption is wrong.

A daily battle

Great news: I will be rich!

FROM THE DESK OF MR. AMINU SALEH
DIRECTOR, FOREIGN OPERATIONS DEPARTMENT
AFRI BANK PLC
Afribank Plaza,
14th Floormoney344.jpg
51/55 Broad Street,
P.M.B 12021 Lagos-Nigeria

Attention: Honorable Beneficiary,

IMMEDIATE PAYMENT NOTIFICATION

It is my modest obligation to write you th
financial institution (AFRI BANK PLC). I a
The British Government, in conjunction w
foreign payment matters, has empowered
release them to their appropriate benefici

owed payment through our most respected
tions Department, AFRI Bank Plc, NIGERIA.
NITED NATIONS ORGANIZATION on
tion, to handle all foreign payments and
leral Reserve Bank.

To facilitate the process of this transactio

tion below:

1) Your full Name and Address:
2) Phones, Fax and Mobile No. :
3) Profession, Age and Marital Status:
4) Copy of any valid form of your Identification:

How to tell spam from ham?

FROM THE DESK OF MR. AMINU SALEH
DIRECTOR, FOREIGN OPERATIONS DEPARTMENT
AFRI BANK PLC
Afribank Plaza,
14th Floormoney344.jpg
51/55 Broad Street,
P.M.B 12021 Lagos-Nigeria

Attention: Honorable Beneficiary,

IMMEDIATE PAYMENT NOTIFICATION VALUED AT **US$10 MILLION**

Dear Dr.Sha,

I just would like to remind you of your scheduled presentation for CS597, Monday October 13, 12pm at OHE122.

If there is anything that you would need, please do not hesitate to contact me.

sincerely,

Christian Siagian

Intuition

How human solves the problem?

Spam emails

concentrated use of a lot of words like "money", "free", "bank account",

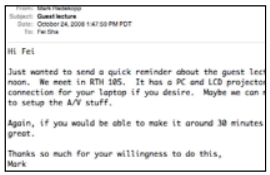
Ham emails

word usage pattern is more spread out

Simple strategy: count the words

Bag-of-word representation
of documents (and textual data)

$$\begin{pmatrix} \text{free} & 100 \\ \text{money} & 2 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$



$$\begin{pmatrix} \text{free} & 1 \\ \text{money} & 1 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

Weighted sum of those telltale words

different weights for spam and ham:
representing how compatible the
word usage pattern is to different
category

$$\begin{pmatrix} \text{free} & 100 \\ \text{money} & 2 \\ \vdots & \vdots \\ \text{account} & 2 \\ \vdots & \vdots \end{pmatrix}$$

$$\begin{pmatrix} 100 \times 0.2 \\ 2 \times 0.3 \\ \vdots \\ 2 \times 0.3 \\ \vdots \end{pmatrix}$$

= 3.2

$$\begin{pmatrix} 100 \times 0.01 \\ 2 \times 0.02 \\ \vdots \\ 2 \times 0.01 \\ \vdots \end{pmatrix}$$

= 1.03

Our intuitive model of classification

Assign weight to each word

Compute compatibility score to "spam"

of "free" x a_{free} + # of "account" x $a_{account}$ + # of "money" x a_{money}

Compute compatibility score to "ham":

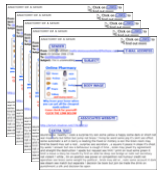
of "free" x b_{free} + # of "account" x $b_{account}$ + # of "money" x b_{money}

Make a decision:

if spam score > ham score then spam

else ham

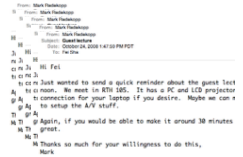
How we get the weights?



Learning from experience

get a lot of spams

get a lot of hams



But what to optimize?

Naive Bayes model for identifying spams

Class label: binary

y = {spam, ham}

Features: word counts in the document (Bag-of-word)

Ex: x = {('free', 100), ('lottery', 10), ('money', 10), , ('identification', 1)...}

Each pair is in the format of
(w_i, #w_i), namely, a unique word in the dictionary,
and the number of times it shows up

Naive Bayes classification rule

For any document x, we need to compute

$$p(\text{spam}|x) \quad \text{and} \quad p(\text{ham}|x)$$

Using Bayes rule, this gives rise to

$$p(\text{spam}|x) = \frac{p(x|\text{spam})p(\text{spam})}{p(x)}, \quad p(\text{ham}|x) = \frac{p(x|\text{ham})p(\text{ham})}{p(x)}$$

It is convenient to compute the logarithms, so we need only to compare

$$\log[p(x|\text{spam})p(\text{spam})] \quad \text{versus} \quad \log[p(x|\text{ham})p(\text{ham})]$$

as the denominators are the same

Classifier in the linear form of compatibility scores

$$\log[p(x|\text{spam})p(\text{spam})] = \log\left[\prod_i p(w_i|\text{spam})^{\#w_i} p(\text{spam})\right]$$
$$= \sum_i \#w_i \log p(w_i|\text{spam}) + \log p(\text{spam})$$

Similarly, we have

$$\log[p(x|\text{ham})p(\text{ham})] = \sum_i \#w_i \log p(w_i|\text{ham}) + \log p(\text{ham})$$

Namely, we are back to the idea of comparing weighted sum of # of word occurrences!
$\log p(\textit{spam})$ *and* $\log p(\textit{ham})$ *are called "priors" or "bias" (they are not in our intuition but they are crucially needed)*

How to predict?

The **prediction** for a new example $\boldsymbol{x}$ is

$$\underset{c\in[\mathsf{C}]}{\text{argmax}}\ P(y=c \mid \boldsymbol{x}) = \underset{c\in[\mathsf{C}]}{\text{argmax}}\ \frac{P(\boldsymbol{x} \mid y=c)P(y=c)}{P(\boldsymbol{x})}$$
$$= \underset{c\in[\mathsf{C}]}{\text{argmax}}\ \left(P(y=c)\prod_{d=1}^{\mathsf{D}} P(x_d \mid y=c)\right)$$
$$= \underset{c\in[\mathsf{C}]}{\text{argmax}}\ \left(\ln P(y=c) + \sum_{d=1}^{\mathsf{D}} \ln P(x_d \mid y=c)\right)$$

These two probability terms can be estimated from training data.

For discrete features.

For a label $c \in [\mathsf{C}]$,

$$P(y=c) = \frac{|\{n : y_n = c\}|}{N} = \frac{\#\text{of data points labeled as c}}{N}$$

For each possible value k of a discrete feature d,

$$P(x_d = k \mid y=c) = \frac{|\{n : x_{nd} = k, y_n = c\}|}{|\{n : y_n = c\}|}$$

They can be estimated separately.

Translating back to our problem of detecting spam emails

- Collect a lot of ham and spam emails as training examples
- Estimate the "bias"

$$p(\text{ham}) = \frac{\#\text{of ham emails}}{\#\text{of emails}}, \quad p(\text{spam}) = \frac{\#\text{of spam emails}}{\#\text{of emails}}$$

- Estimate the weights (i.e., $p(\text{dollar}|\text{ham})$ etc)

$$p(\text{funny\_word}|\text{ham}) = \frac{\#\text{of funny\_word in ham emails}}{\#\text{of words in ham emails}}$$
$$p(\text{funny\_word}|\text{spam}) = \frac{\#\text{of funny\_word in spam emails}}{\#\text{of words in spam emails}}$$

- Compute argmax slide 43.

Continuous features

If the feature is continuous, we can do parametric estimation, via a Gaussian

$$P(x_d = x \mid y = c) = \frac{1}{\sqrt{2\pi}\sigma_{cd}} \exp\left(-\frac{(x - \mu_{cd})^2}{2\sigma_{cd}^2}\right)$$

where μ_{cd} and σ_{cd}^2 are the empirical mean and variance of feature d among all examples with label c.

We will see more on this model later in the course.

In particular, Naive Bayes model with missing labels.

Generative classifier

Naïve Bayes $P(X, Y = c)$ is a generative model.

We can generate a sample of the data

$$P(X) = \sum_c P(X \mid Y = c)P(Y = c)$$

We estimate parameters $P(X \mid Y = c)$ and $P(Y = c)$ directly from training data.

Later we will consider a probabilistic descriminative classifier - Logistic Regression.

Outline

Problem 5

Suppose you are given the following set of data with three Boolean input variables a, b, and c, and a single Boolean output variable K.

a	b	c	K
1	0	1	1
1	1	1	1
0	1	1	0
1	1	0	0
1	0	1	0
0	0	0	1
0	0	0	1
0	0	1	0

According to the naive Bayes classifier, what is a value of P(K = 1|a = 1,b = 1,c = 0)?

Problem 6

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

Given training examples above, classify a Red Domestic SUV.