Lecture 13-14: Machine Learning

CS 486/686: Introduction to Artificial Intelligence

Plan for Today

- Introduction to Machine Learning
  - Components
  - Common Learning Tasks
  - Measuring Success
  - Bias
  - Learning as Search

Introduction

- Learning is the ability to improve behaviour based on experience.
  - Range of behavior is expanded
    - The agent can do more
  - The accuracy on tasks is improved
    - The agent can do things better
  - The speed is improved
    - The agent can do things faster

What is Machine Learning?

- Definition (T Mitchell):

A computer program is said to **learn** from **experience** E with respect to some class of **tasks** T and **performance measures** P, if its performance at tasks in T, as measured by P, improves with experience E.

- Task: the behaviour or task that is being improved
- Data: experiences that are used to improve performance
- Performance measure: how to evaluate improvement

Common Learning Tasks

- Supervised Learning: given a set of pre-defined training examples, learn the label of a new instance
- Unsupervised Learning: find natural classes of examples
- Reinforcement Learning: determine what to do based on rewards and punishments
- Transfer Learning: learning from an expert
- Active Learning: learner actively seeks what to learn

Learning from Feedback?

- Learning tasks can be categorized by feedback given to the learner
  - Supervised: What has to be learned is specified for each example
    - Example: x = “this movie is great” -> y=postive
  - Unsupervised: No classifications are given. Learner has to discover categories and patterns in the data
  - Reinforcement Learning: Feedback occurs after taking a sequence of actions

Measuring Success

- Goal: Generalization
  - The measure of success is not on the training examples, but on new (unseen) examples
- Example: Consider two agents solving a binary classification task
  - P claims the negative examples seen are the only negative examples. Every other instance is positive.
  - N claims the positive examples seen are the only positive examples. Every other instance is negative.
  - Both agents correctly classify every training example, but disagree on every other example.

Supervised Learning

- Given
  - Input features: X_1, …, X_n
  - Target features: Y_1,…, Y_k
  - Training examples where the values for the input features and the target features are given for each example (x, y=f(x))
  - Test examples where only the values for the input features are given
- Predict the values for the target features for the test examples
  - Classification: when Y_i are discrete
  - Regression: when Y_i are continuous

Supervised Learning

Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes



Goal: Return a function h that approximates f(x)

h is the **hypothesis**

Supervised Learning

- We want to learn a **target function** *f* but we only have examples (x, f(x))
- We start with a guess as to what form f might take (representation). Let H (hypothesis space) be the set of all possible hypotheses (functions) we will consider.

- Learning is basically **search** over H to find a good approximation of f (we will revisit this idea)

Inductive Learning Hypothesis

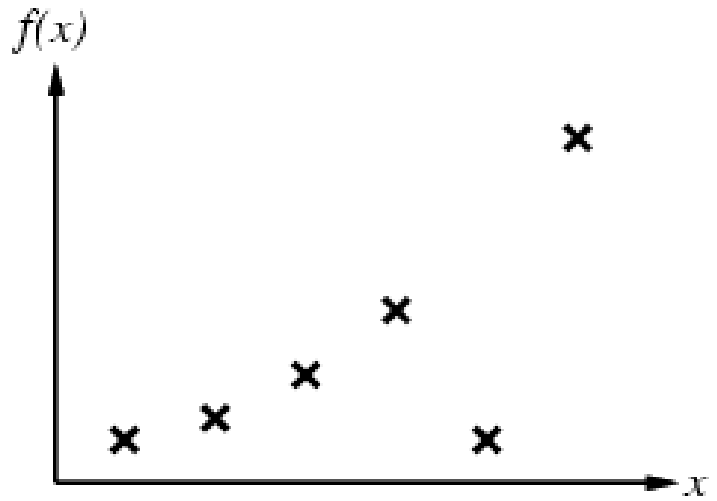
Any hypothesis found to approximate the target function over a sufficiently large set of training examples will also approximate the target function well over any unobserved examples.

Bias

- The preference for one hypothesis over another is called a bias
  - A bias is needed to make predictions on unseen data
  - Saying a hypothesis is better than N's or P's hypothesis isn't something that's obtained from the data
  - To have any inductive process make predictions will require a bias
  - What constitutes a good bias is an empirical question about which biases work best in practice.

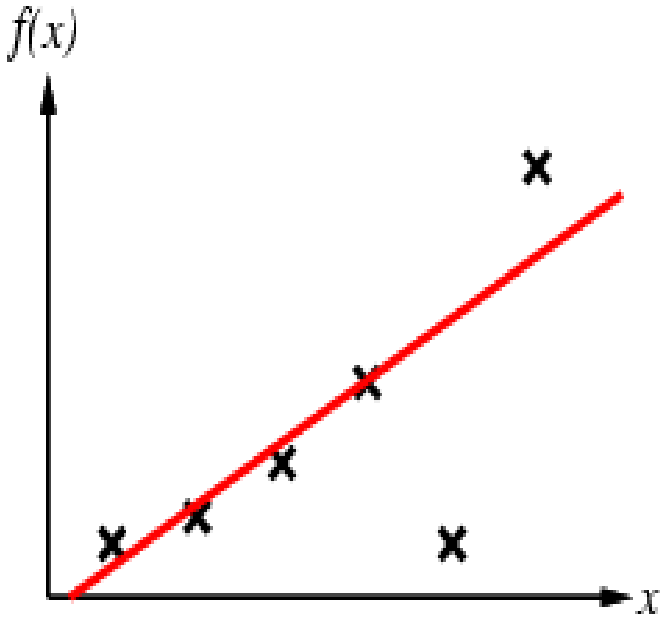
Inductive Learning

- Construct/adjust h to agree with f on the training set.
- h is consistent if it agrees with f on all examples
- Question: What is a good h?



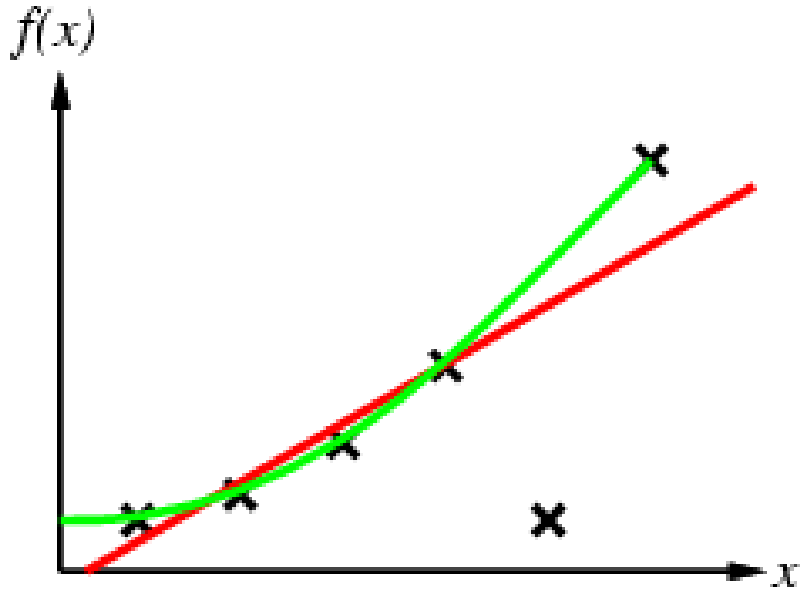
Inductive Learning

- Construct/adjust h to agree with f on the training set.
- h is consistent if it agrees with f on all examples
- Question: What is a good h?



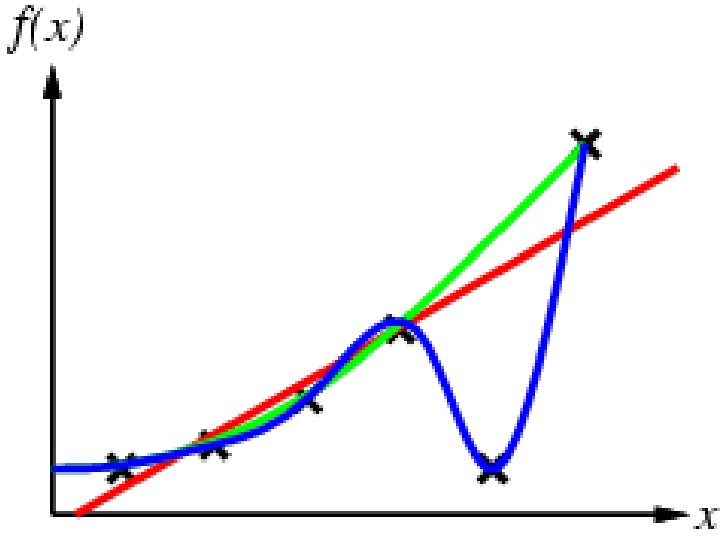
Inductive Learning

- Construct/adjust h to agree with f on the training set.
- h is consistent if it agrees with f on all examples
- Question: What is a good h?



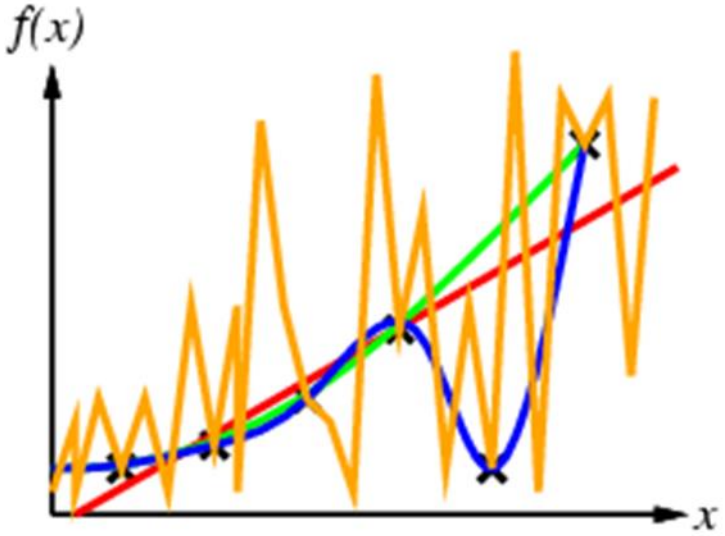
Inductive Learning

- Construct/adjust h to agree with f on the training set.
- h is consistent if it agrees with f on all examples
- Question: What is a good h?



Inductive Learning

- Construct/adjust h to agree with f on the training set.
- h is consistent if it agrees with f on all examples
- Question: What is a good h?



Inductive Bias (Ockham's Razor):
Prefer the simplest hypothesis consistent with the data
Underfitting: h is too simple to capture patterns in f
Overfitting: h is too expressive and learns noise in f

Learning as Search

- Given a representation and a bias, the problem of learning is reduced to one of search
  - Learning is search through the space of possible representations, looking for the representation(s) that best fit the data given the bias
  - These search spaces are typically prohibitively large for systematic search
  - A learning algorithm is made of a search space, an evaluation method, and a search method

Some Notes About Data

- Data is not perfect
  - The features given are often inadequate to predict classification
  - There are examples with missing features
  - Data is just incorrect (e.g. labelled incorrectly)
  - It is incomplete
  - …
- Overfitting
  - Finding patterns in the data where there is no actual pattern

Evaluating Predictions

- Suppose Y is a feature and e is an example
  - Y(e) is the value of feature Y for example e
  - h(e) is the predicted value of feature Y for feature e

- The error of the prediction is a measure of how close h(e) is to Y(e).
- There are many possible errors that could be measured.

Measures: Absolute Error

- Suppose Y is a feature and e is an example
  - Y(e) is the value of feature Y for example e
  - h(e) is the predicted value of feature Y for feature e

$$\sum_{e \in E} \sum_{Y_i} |Y_i(e) - h_i(e)|$$

Measures: Sum of Squares

- Suppose Y is a feature and e is an example
  - Y(e) is the value of feature Y for example e
  - h(e) is the predicted value of feature Y for feature e

$$\sum_{e \in E} \sum_{Y_i} (Y_i(e) - h_i(e))^2$$

Measures: Worst Case Error

- Suppose Y is a feature and e is an example
  - Y(e) is the value of feature Y for example e
  - h(e) is the predicted value of feature Y for feature e

$$\max_{e \in E} \max_{Y_i} |Y_i(e) - h_i(e)|$$

Not All Errors are Equal

- Not all errors are equal
  - Predict a patient has a disease when they do not
  - Predict a patient does not have a disease when they do

Predicted

Actual	T	F
T	True Positive (TP)	False Negative (FN)
F	False Positive (FP)	True Negative (TN)

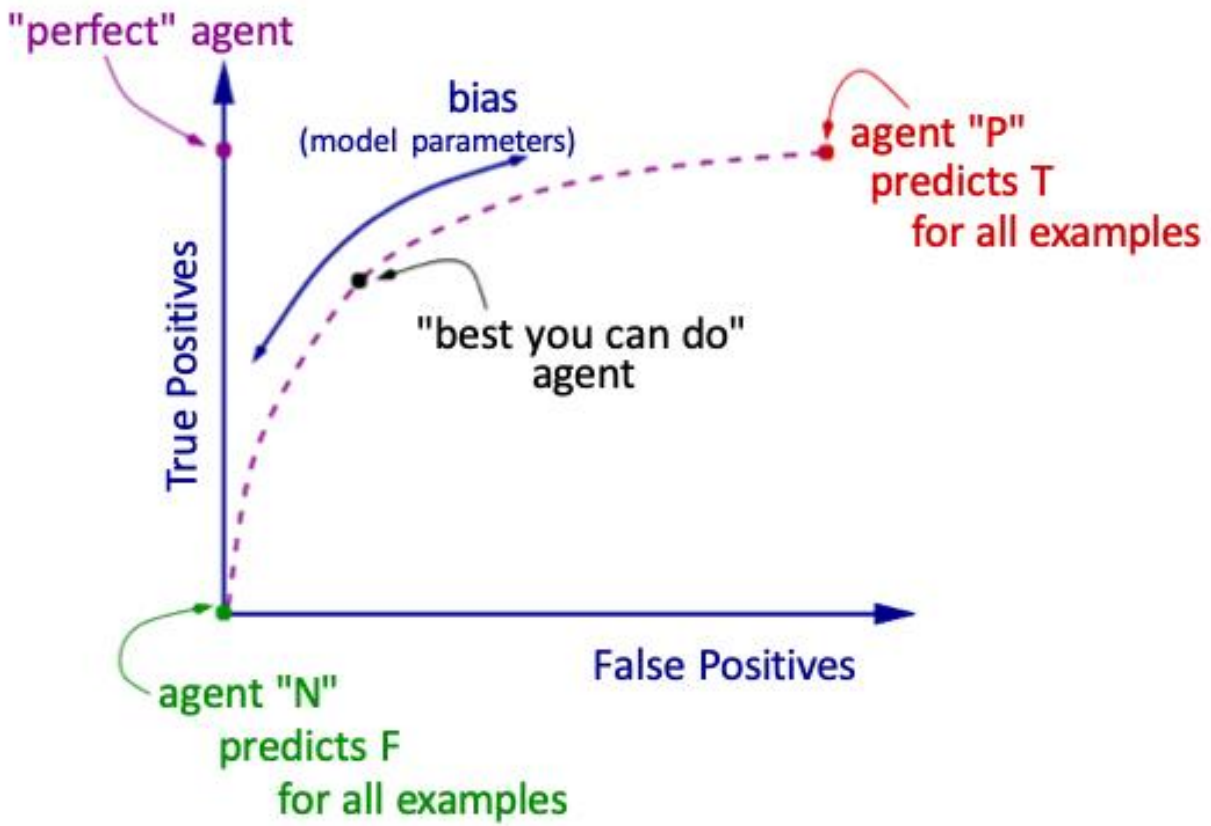
Not All Errors are Equal

- Need to map out both types of errors to find the best trade-off
  - Recall = Sensitivity = TP/(TP+FN)
  - Specificity = TN/(TN+FP)
  - Precision = TP/(TP+FP)
  - F-measure = 2*Precision*Recall/(Precision + Recall)

		Predicted	
Actual		T	F
	T	True Positive (TP)	False Negative (FN)
	F	False Positive (FP)	True Negative (TN)

Assessing Performance: Receiver Operating Curve

- ROC gives full range of performance on an algorithm across different biases



Assessing Performance: Train/validation split

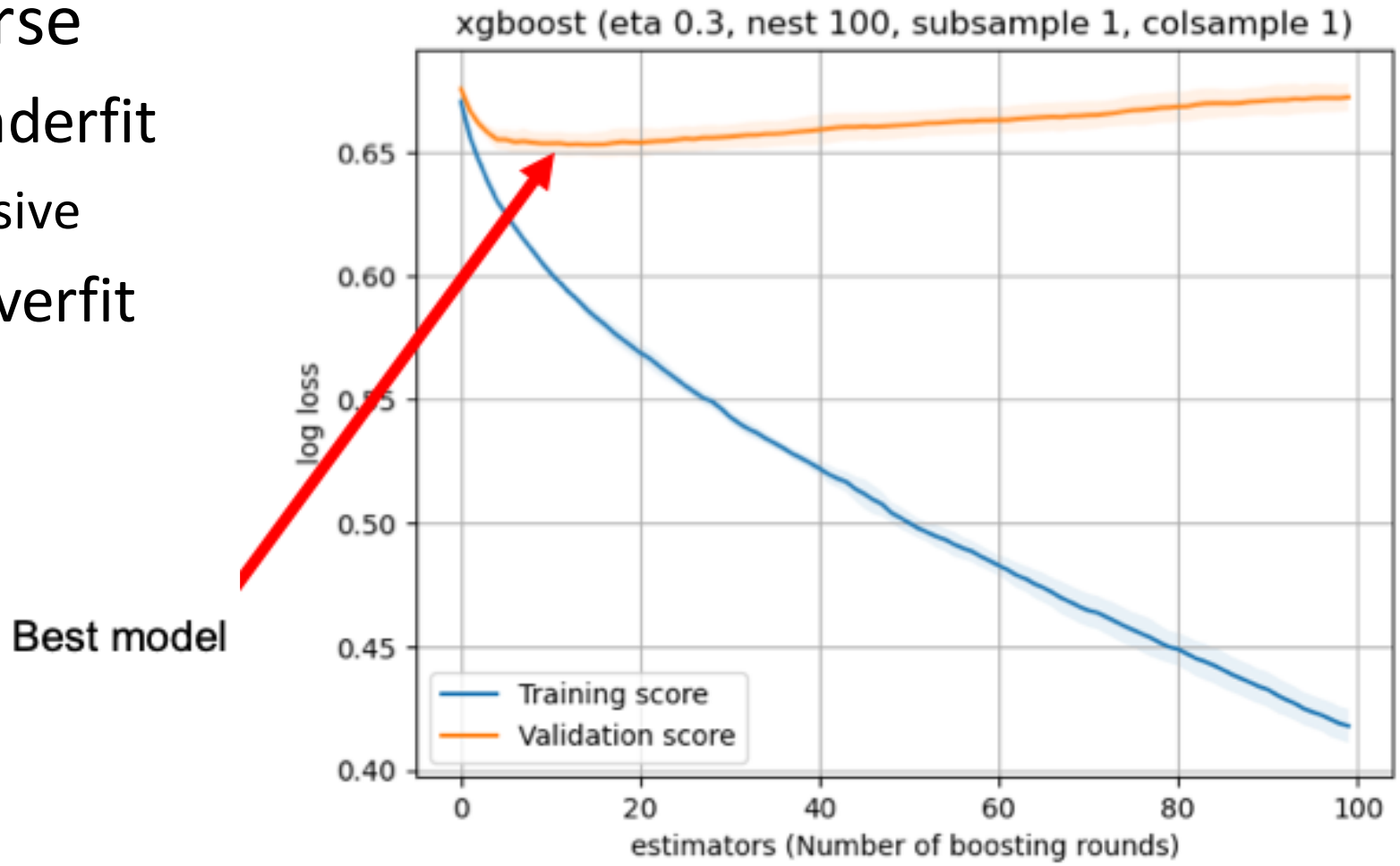
- Suppose you have a dataset
  - Subdivide into train/test splits
- Apply learning algorithm to find h on training set
- Measure performance of h on test set
- A learning algorithm should ***NEVER*** be allowed to see the test set data before the hypothesis is tested on it!!!!

Assessing Performance: train/validation split

- Every time you want to compare performance of a hypothesis on a test set you must use a new test set!
  - Alternatively: dataset->train/validation/test
  - Train h on training set, track performance on validation set
  - Once found best h*, evaluate one last time on the test set
  - Cross validation: average performance on many random splits

Assessing Performance: train/validation split

- A typical learning curse
  - Best “best” point: underfit
    - Make h more expressive
  - After ”best” point: overfit
    - Make h simpler
    - Regularization (later)



Example: Spam Filter

- Input: an email
- Output: spam/ham
- Setup:
  - Get a large collection of emails labelled as “spam” or “ham”
  - Note: someone has to hand label all these emails
  - Want to learn to predict label of new future emails
- Features
  - Words: FREE!!
  - Test pattern: $dd CAPS
  - Non-text: SenderInContacts
  - …



Dear Sir.

First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial and top secret. …



TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

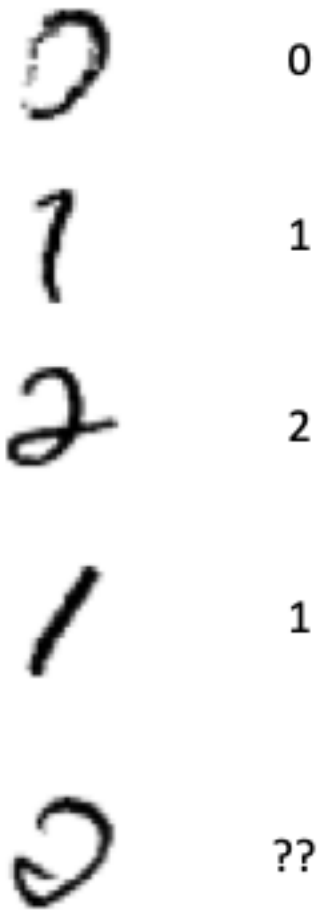
99 MILLION EMAIL ADDRESSES
FOR ONLY $99



Ok, Iknow this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Example: Digit Recognition

- Input: images/pixel grids
- Output: a digit 0-9
- Setup:
  - Get a collection of example images, each labelled with a digit
  - Note: someone has to label this data!
  - Want to learn to predict label of new future images
- Features:
  - Pixels: (6,8)=ON
  - Shape Patterns,…

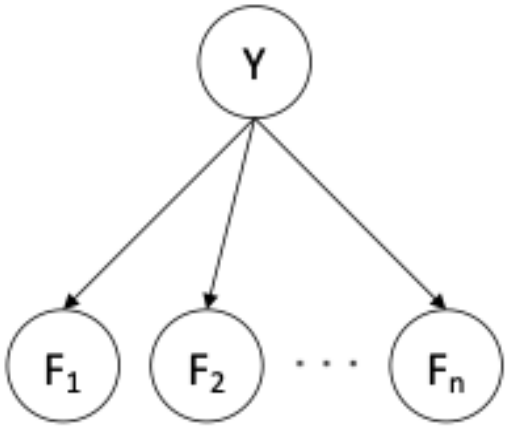


Model-Based Classification

- Model-based approach
  - Bulid a model (Bayes Net) where both the labels and the features are random variables
  - Instantiate any observed features
  - Query for the distribution of the label given the features
- Challenges
  - What structure should the Bayes Net have?
  - How should we learn the parameters?

Naïve Bayes for Digits

- Naïve Bayes: Assume all features are independent given the label
- Simple digit recognition:
  - One feature $F_{i,j}$ for each grid position <i,j>
  - Feature variables are on/off given intensity level
  - Each image maps to a feature vector
- Naïve Bayes Model
- What do we need to learn?



$$P(Y|F_{0,0}\dots F_{15,15}) \propto P(Y)\prod_{i,j} P(F_{i,j}|Y)$$

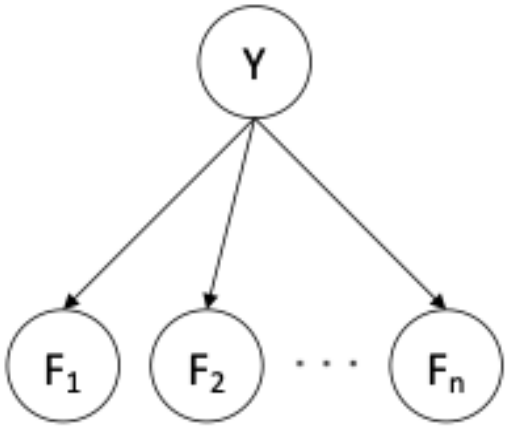
General Naïve Bayes

|Y| parameters

$$P(Y, F_1 \dots F_n) = P(Y) \prod_i P(F_i|Y)$$

|Y| x |F|^n values

n x |F| x |Y| parameters



We only need to specify how each feature depends on the class.
Total number of parameters is linear in n
Model is very simplistic but tends to work anyway!

Inference in Naïve Bayes

- Goal: compute posterior distribution over label variable Y
  - Step 1: get joint probability of label and evidence for each label

$$P(Y, f_1 \dots f_n) = \begin{bmatrix} P(y_1, f_1 \dots f_n) \\ P(y_2, f_1 \dots f_n) \\ \vdots \\ P(y_k, f_1 \dots f_n) \end{bmatrix} \Rightarrow \begin{bmatrix} P(y_1) \prod_i P(f_i|y_1) \\ P(y_2) \prod_i P(f_i|y_2) \\ \vdots \\ P(y_k) \prod_i P(f_i|y_k) \end{bmatrix}$$

$$\frac{\begin{bmatrix} P(y_1) \prod_i P(f_i|y_1) \\ P(y_2) \prod_i P(f_i|y_2) \\ \vdots \\ P(y_k) \prod_i P(f_i|y_k) \end{bmatrix}}{P(f_1 \dots f_n)} \quad (+)$$

  - Step 2: sum to get probability of evidence
  - Step 3: normalize by dividing Step 1 by Step 2

$$\Downarrow$$

$$P(Y|f_1 \dots f_n)$$

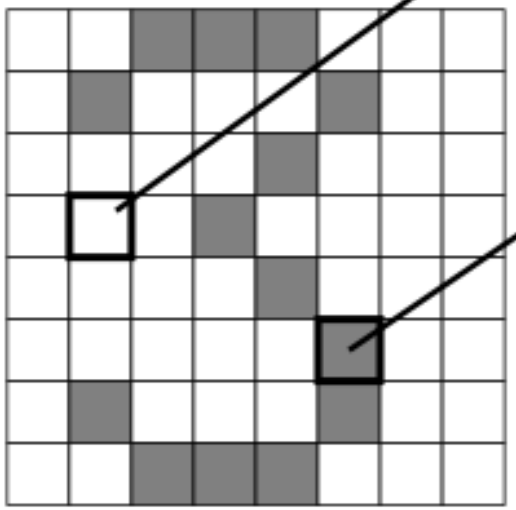
General Naïve Bayes

- What do we need to use Naïve Bayes?
  - Inference Method
    - Given P(Y) and P(Fi|Y) compute P(Y|F1,…,Fn)
  - Estimates of the CPTs
    - P(Y) prior over labels
    - P(Fi|Y) for each feature
    - These are called the parameters of the model, denoted by θ
    - We had assumed they were given, but actually learned from data (training data counts)

Example

$P(Y)$

1	0.1
2	0.1
3	0.1
4	0.1
5	0.1
6	0.1
7	0.1
8	0.1
9	0.1
0	0.1



$P(F_{3,1} = on|Y)$

1	0.01
2	0.05
3	0.05
4	0.30
5	0.80
6	0.90
7	0.05
8	0.60
9	0.50
0	0.80

$P(F_{5,5} = on|Y)$

1	0.05
2	0.01
3	0.90
4	0.80
5	0.90
6	0.90
7	0.25
8	0.85
9	0.60
0	0.80

Naïve Bayes for Text

- Bag-of-Words Model
  - Words are independent given the document type
  - Ie insensitive of word order

$$P(y|\text{document}) \propto \Pi_{i=1}^{|\text{Vocab}|} P(w_i|y)$$

Naïve Bayes for Spam

- Model: $P(Y, W_1 \dots W_n) = P(Y) \prod_i P(W_i|Y)$
- What are the parameters?

$P(Y)$

ham :	0.66
spam:	0.33

$P(W|\text{spam})$

the :	0.0156
to :	0.0153
and :	0.0115
of :	0.0095
you :	0.0093
a :	0.0086
with:	0.0080
from:	0.0075
...	

$P(W|\text{ham})$

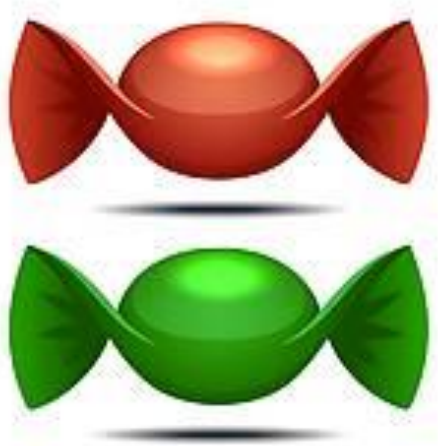
the :	0.0210
to :	0.0133
of :	0.0119
2002:	0.0110
with:	0.0108
from:	0.0107
and :	0.0105
a :	0.0100
...	

- Where do these tables come from?

Maximum Likelihood

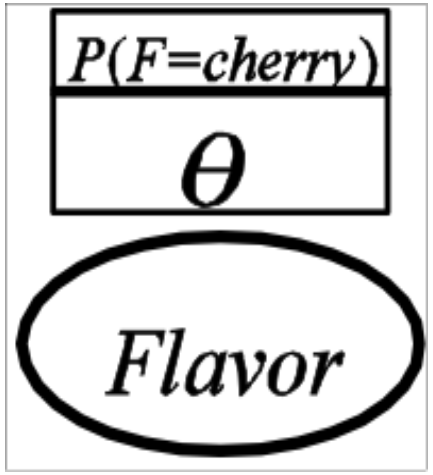
- Favourite candy sold in two flavours
  - Lime and Cherry
- Same wrapper for both flavours
- Sold in bags with different ratios
  - 100% cherry
  - 75% cherry, 25% lime
  - 50% cherry, 50% lime
  - 25% cherry, 75% lime
  - 100% lime

- You bought a bag of candy but do not know its flavour ratio
- After eating k candies
  - What is the flavour ratio of the bag?
  - What will be the flavour of the next candy?



Maximum Likelihood

- Hypothesis h_θ
  - P(cherry)=θ and P(lime)=1-θ
  - θ is our parameter
- Data d:
  - N candies (c cherry and l=N-c lime)

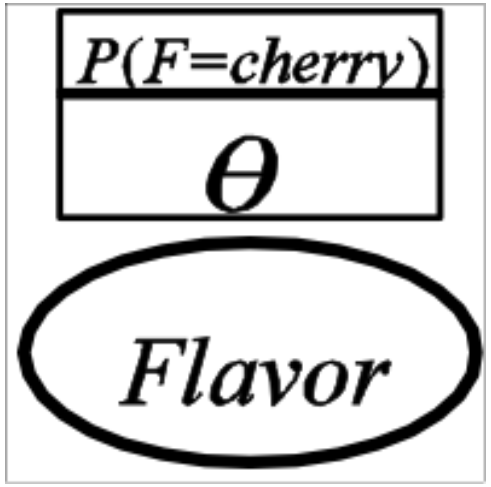


- What should θ be?

Maximum Likelihood

- Likelihood of this particular dataset

$$P(d|h_\theta) = \theta^c(1-\theta)^l$$



- Log-likelihood

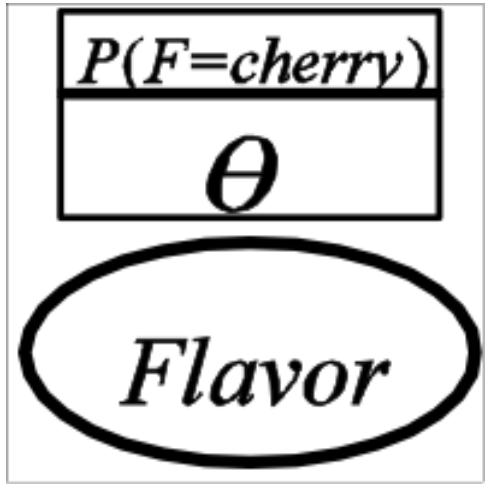
$$\begin{aligned} L(d|h_\theta) &= \log P(d|h_\theta) \\ &= c\log\theta + l\log(1-\theta) \end{aligned}$$

Maximum Likelihood

- Find the θ that maximizes the log likelihood

$$\frac{\partial L(d|h_\theta)}{\partial \theta} = \frac{c}{\theta} - \frac{l}{1-\theta} = 0$$

$$\theta = \frac{c}{c+l} = \frac{c}{N}$$



Elegant! This is the observed counts!

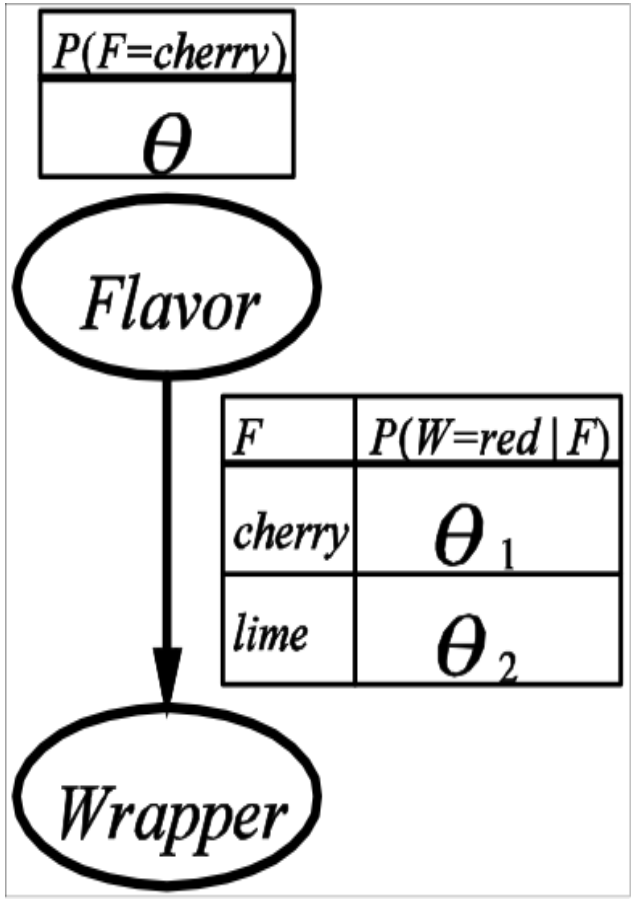
Maximum Likelihood

Hypothesis: $h_{\theta, \theta_1, \theta_2}$

Data:

c Cherries:
G_c green wrappers
R_c red wrappers

l Limes:
G_l green wrappers
R_l red wrappers



Maximum Likelihood

$$P(d|h_{\theta,\theta_1,\theta_2}) = \theta^c(1-\theta)^l\theta_1^{R_c}(1-\theta_1)^{G_c}\theta_2^{R_l}(1-\theta_2)^{G_l}$$

$$\begin{aligned} L(d|h_{\theta,\theta_1,\theta_2}) =& [c\log\theta + l\log(1-\theta)] \\ &+ [R_c\log\theta_1 + G_c\log(1-\theta_1)] \\ &+ [R_l\log\theta_2 + G_l\log(1-\theta_2)] \end{aligned}$$

$$\theta = \frac{c}{c+l}$$

$$\theta_1 = \frac{R_c}{R_c+G_c}$$

$$\theta_2 = \frac{R_l}{R_l+G_l}$$

Elegant! This is the observed counts!