Administration

- Final
  - Will be done on the official university date for this class: 5/9, 1:30.
  - We will have a review session during the last scheduled lecture, 5/2.
  - The schedule has been updated accordingly.

- Projects:
  - Reports are due on 5/11.
  - In addition, instead of presentations, we will ask you to send a short video of your presentation **< 5 min**.
  - Project progress reports are due on 4/17.

Recap: Error Driven Learning

- Consider a distribution D over space $X \times Y$
- X - the instance space; Y - set of labels. (e.g. $+/-1$)
- Can think about the data generation process as governed by $D(x)$, and the labeling process as governed by $D(y|x)$, such that

$$D(x,y)=D(x)\ D(y|x)$$

- This can be used to model both the case where labels are generated by a function $y=f(x)$, as well as noisy cases and probabilistic generation of the label.
  - If the distribution D is known, there is no learning. We can simply predict

$$y = \text{argmax}_y\ D(y|x)$$

- If we are looking for a hypothesis, we can simply find the one that minimizes the probability of mislabeling:

$$h = \text{argmin}_h\ E_{(x,y)\sim D}\ [[h(x) \neq y]]$$

Recap: Error Driven Learning (2)

- Inductive learning comes into play when the distribution is not known.

- Then, there are two basic approaches to take.

  Discriminative (direct) learning

and

  Bayesian Learning (Generative)

- Running example: Text Correction:
- "I saw the girl it the park" → I saw the girl in the park

1: Direct Learning

- Model the problem of text correction as a problem of learning from examples.
- Goal: learn directly how to make predictions.

PARADIGM

- Look at many (positive/negative) examples.
- Discover some regularities in the data.
- Use these to construct a prediction policy.
- A policy (a function, a predictor) needs to be specific.

  [it/in] rule: if the occurs after the target ⇒in
- Assumptions comes in the form of a hypothesis class.

Bottom line: approximating h : X → Y is estimating P(Y|X).

Direct Learning (2)

- Consider a distribution D over space X×Y
- X - the instance space; Y - set of labels. (e.g. +/-1)
- Given a sample $\{(x,y)\}_1^m{}_{,}$, and a loss function L(x,y)
- Find $h \in H$ that minimizes

$$\Sigma_{i=1,m} D(x_i,y_i) L(h(x_i),y_i) + \text{Reg}$$

- L can be:

L(h(x),y)=1, h(x)≠y, o/w L(h(x),y) = 0 (0-1 loss)

L(h(x),y)=(h(x)-y)^2 , (L_2)

L(h(x),y)= max{0,1-y h(x)} (hinge loss)

L(h(x),y)= exp{- y h(x)} (exponential loss)

- Guarantees: If we find an algorithm that minimizes loss on the observed data. Then, learning theory guarantees good future behavior (as a function of H).

2: Generative Model

The model is called "generative" since it assumes how data X is generated given y

- Model the problem of text correction as that of generating correct sentences.
- Goal: learn a model of the language; use it to predict.

PARADIGM

- Learn a probability distribution over all sentences
  - In practice: make assumptions on the distribution's type
- Use it to estimate which sentence is more likely.
  - **Pr(I saw the girl it the park) <> Pr(I saw the girl in the park)**
  - In practice: a decision policy depends on the assumptions

Bottom line: the generating paradigm approximates
$P(X,Y) = P(X|Y) P(Y)$.

- Guarantees: We need to assume the "right" probability distribution

Probabilistic Learning

- There are actually two different notions.
- Learning probabilistic concepts
  - The learned concept is a function $c:X \rightarrow [0,1]$
  - c(x) may be interpreted as the probability that the label 1 is assigned to x
  - The learning theory that we have studied before is applicable (with some extensions).
- Bayesian Learning: Use of a probabilistic criterion in selecting a hypothesis
  - The hypothesis can be deterministic, a Boolean function.
- It's not the hypothesis – it's the process.

Basics of Bayesian Learning

- Goal: find the best hypothesis from some space H of hypotheses, given the observed data D.
- Define best to be: most probable hypothesis in H
- In order to do that, we need to assume a probability distribution over the class H.
- In addition, we need to know something about the relation between the data observed and the hypotheses (E.g., a coin problem.)
  - As we will see, we will be Bayesian about other things, e.g., the parameters of the model

Basics of Bayesian Learning

- P(h) - the prior probability of a hypothesis h
  Reflects background knowledge; before data is observed. If no information - uniform distribution.

- P(D) - The probability that this sample of the Data is observed. (No knowledge of the hypothesis)

- P(D|h): The probability of observing the sample D, given that hypothesis h is the target

- P(h|D): The posterior probability of h. The probability that h is the target, given that D has been observed.

Bayes Theorem

$$\mathbf{P(h \mid D) = P(D \mid h)\, {}^{P(h)}\!/_{P(D)}}$$

- P(h|D) increases with P(h) and with P(D|h)
- P(h|D) decreases with P(D)

Basic Probability

- **Product Rule:** $P(A,B) = P(A|B)P(B) = P(B|A)P(A)$
- **If A and B are independent:**
  - $P(A,B) = P(A)P(B)$; $P(A|B)= P(A)$, $P(A|B,C)=P(A|C)$
- **Sum Rule:** $P(A \vee B) = P(A)+P(B)-P(A,B)$
- **Bayes Rule:** $P(A|B) = P(B|A) P(A)/P(B)$
- **Total Probability:**
  - If events $A_1, A_2,\ldots A_n$ are mutually exclusive: $A_i \cap A_j = \phi$, $\sum_i P(A_i)= 1$
  - $P(B) = \sum P(B , A_i) = \sum_i P(B|A_i) P(A_i)$
- **Total Conditional Probability:**
  - If events $A_1, A_2,\ldots A_n$ are mutually exclusive: $A_i \cap A_j = \phi$, $\sum_i P(A_i)= 1$
  - $P(B|C) = \sum P(B , A_i|C) = \sum_i P(B|A_i,C) P(A_i|C)$

Learning Scenario

- $P(h|D) = P(D|h) P(h)/P(D)$

- The learner considers a set of candidate hypotheses H (models), and attempts to find the most probable one $h \in H$, given the observed data.

- Such maximally probable hypothesis is called maximum a posteriori hypothesis (MAP); Bayes theorem is used to compute it:

$$h_{MAP} = \text{argmax}_{h \in \mathcal{H}} P(h|D) = \text{argmax}_{h \in \mathcal{H}} P(D|h) P(h)/P(D)$$

$$= \text{argmax}_{h \in \mathcal{H}} P(D|h) P(h)$$

Learning Scenario (2)

$h_{MAP} = \text{argmax}_{h \in \mathcal{H}} P(h|D) = \text{argmax}_{h \in \mathcal{H}} P(D|h) P(h)$

- We may assume that a priori, hypotheses are equally probable: $P(h_i) = P(h_j) \; \forall \; h_i, h_j \in H$
- We get the Maximum Likelihood hypothesis:

$$\boxed{h_{ML} = \text{argmax}_{h \in \mathcal{H}} P(D|h)}$$

- Here we just look for the hypothesis that best explains the data

Examples

- $h_{MAP} = \text{argmax}_{h \in \mathcal{H}} P(h|D) = \text{argmax}_{h \in \mathcal{H}} P(D|h)\,P(h)$

- A given coin is either fair or has a 60% bias in favor of Head.
- Decide what is the bias of the coin [This is a learning problem!]

- Two hypotheses: h_1: P(H)=0.5; h_2: P(H)=0.6
  - Prior: P(h): $P(h_1)=0.75$ $P(h_2)=0.25$
  - Now we need Data. 1st Experiment: coin toss is H.
  - P(D|h):

    $P(D|h_1)=0.5$; $P(D|h_2)=0.6$
  - P(D):

    $$P(D)=P(D|h_1)P(h_1) + P(D|h_2)P(h_2)$$
    $$= 0.5 \bullet 0.75 + 0.6 \bullet 0.25 = 0.525$$
  - P(h|D):

    $P(h_1|D) = P(D|h_1)P(h_1)/P(D) = 0.5 \bullet 0.75/0.525 = 0.714$

    $P(h_2|D) = P(D|h_2)P(h_2)/P(D) = 0.6 \bullet 0.25/0.525 = 0.286$

Examples(2)

- $h_{MAP} = \text{argmax}_{h \in \mathcal{H}} P(h|D) = \text{argmax}_{h \in \mathcal{H}} P(D|h) P(h)$

- A given coin is either fair or has a 60% bias in favor of Head.
- Decide what is the bias of the coin [This is a learning problem!]

- Two hypotheses: h_1: P(H)=0.5; h_2: P(H)=0.6
  - Prior: P(h): $P(h_1)=0.75$ $P(h_2)=0.25$

- After 1st coin toss is H we still think that the coin is more likely to be fair

- If we were to use Maximum Likelihood approach (i.e., assume equal priors) we would think otherwise. The data supports the biased coin better.

- Try: 100 coin tosses; 70 heads.
- You will believe that the coin is biased.

Examples(2)

- $h_{MAP} = \text{argmax}_{h \in \mathcal{H}} P(h|D) = \text{argmax}_{h \in \mathcal{H}} P(D|h)\, P(h)$
- A given coin is either fair or has a 60% bias in favor of Head.
- Decide what is the bias of the coin [This is a learning problem!]

- Two hypotheses: h_1: P(H)=0.5; h_2: P(H)=0.6
  - Prior: P(h): $P(h_1)=0.75$ $P(h_2)=0.25$
- Case of 100 coin tosses; 70 heads.

$$P(D) = P(D|h_1)\, P(h_1) + P(D|h_2)\, P(h_2) =$$
$$= 0.5^{100} \cdot 0.75 + 0.6^{70} \cdot 0.4^{30} \cdot 0.25 =$$
$$= 7.9 \cdot 10^{-31} \cdot 0.75 + 3.4 \cdot 10^{-28} \cdot 0.25$$

$$0.0057 = P(h_1|D) = P(D|h_1)\, P(h_1)/P(D) << P(D|h_2)\, P(h_2)\,/P(D) = P(h_2|D) = 0.9943$$

Example: A Model of Language

- Model 1: There are 5 characters, A, B, C, D, E, and space
- At any point can generate any of them, according to:

  $P(A)= p_1$; $P(B) =p_2$; $P(C) =p_3$; $P(D)= p_4$; $P(E)= p_5$ $P(SP)= p_6$ $\sum_\iota p_i = 1$
- This is a family of distributions; learning is identifying a member of this family.

  E.g., $P(A)= 0.3$; $P(B) =0.1$; $P(C) =0.2$; $P(D)= 0.2$; $P(E)= 0.1$ $P(SP)=0.1$

- We assume a generative model of independent characters (fixed k):

$$P(U) = P(x_1, x_2,\ldots, x_k)= \Pi_{i=1,k} P(x_i \mid x_{i+1}, x_{i+2},\ldots, x_k)= \Pi_{i=1,k} P(x_i)$$

- The parameters of the model are the character generation probabilities (Unigram).
- Goal: to determine which of two strings U, V is more likely.
- The Bayesian way: compute the probability of each string, and decide which is more likely.

Consider Strings: AABBC & ABBBA

- Learning here is: learning the parameters of a known model family
- How?

You observe a string; use it to learn the language model.
E.g., S= AABBABC; Compute P(A)

1. The model we assumed is binomial. You could assume a different model! Next we will consider other models and see how to learn their parameters.

Maximum Likelihood Estimate

- Assume that you toss a (p,1-p) coin m times and get k Heads, m-k Tails. What is p?

2. In practice, smoothing is advisable – deriving the right smoothing can be done by assuming a prior.

- If p is the probability of Head, the probability of the data observed is:

$$P(D|p) = p^k (1-p)^{m-k}$$

- The log Likelihood:

$$L(p) = \log P(D|p) = k \log(p) + (m-k)\log(1-p)$$

- To maximize, set the derivative w.r.t. p equal to 0:

$$dL(p)/dp = k/p – (m-k)/(1-p)$$

- Solving this for p, gives: $p=k/m$

Probability Distributions

- **Bernoulli Distribution:**
  - Random Variable X takes values {0, 1} s.t P(X=1) = p = 1 – P(X=0)
  - (Think of tossing a coin)
- **Binomial Distribution:**
  - Random Variable X takes values {1, 2,…, n} representing the number of successes (X=1) in n Bernoulli trials.
  - P(X=k) = f(n, p, k) = $C_n^k \, p^k \, (1-p)^{n-k}$
  - Note that if X ~ Binom(n, p) and Y ~ Bernulli (p), $X = \sum_{i=1,n} Y$
  - (Think of multiple coin tosses)

Probability Distributions(2)

- **Categorical Distribution:**
  - Random Variable X takes on values in {1,2,…k} s.t $P(X=i) = p_i$ and $\sum_1^k p_i = 1$
  - (Think of a dice)
- **Multinomial Distribution:**
  - Let the random variables X_i (i=1, 2,…, k) indicates the number of times outcome i was observed over the n trials.
  - The vector $X = (X_1, \ldots, X_k)$ follows a multinomial distribution (n,p) where $p = (p_1, \ldots, p_k)$ and $\sum_1^k p_i = 1$
  - $f(x_1, x_2, \ldots x_k, n, p) = P(X_1 = x_1, \ldots X_k = x_k) = \frac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}, \quad \text{when } \sum_{i=1}^k x_i = n$
  - (Think of n tosses of a k sided dice)

Our eventual goal will be: Given a document, predict whether it's "good" or "bad"

A Multinomial Bag of Words

- We are given a collection of documents written in a three word language {a, b, c}. All the documents have exactly n words (each word can be either a, b or c).
- We are given a labeled document collection $\{D_1, D_2 \ldots, D_m\}$. The label y_i of document D_i is 1 or 0, indicating whether D_i is "good" or "bad".
- This model uses the multinominal distribution. That is, a_i (b_i, c_i, resp.) is the number of times word a (b, c, resp.) appears in document D_i.
- Therefore: $a_i + b_i + c_i = |D_i| = n.$
- In this generative model, we have:

$$P(D_i|y = 1) = n!/(a_i!\, b_i!\, c_i!)\ \alpha_1^{a_i}\, \beta_1^{b_i}\, \gamma_1^{c_i}$$

where α_1 (β_1, γ_1 resp.) is the probability that a (b , c) appears in a "good" document.

- Similarly, $P(D_i|y = 0) = n!/(a_i!\, b_i!\, c_i!)\ \alpha_0^{a_i}\, \beta_0^{b_i}\, \gamma_0^{c_i}$
- Note that: $\alpha_0+\beta_0+\gamma_0=\alpha_1+\beta_1+\gamma_1 =1$

Unlike the discriminative case, the "game" here is different:

- ❑ We make an assumption on how the data is being generated.
  - ❑ (multinomial, with α_i, β_i, γ_i)
- ❑ Now, we observe documents, and estimate these parameters.
- ❑ Once we have the parameters, we can predict the corresponding label.

A Multinomial Bag of Words (2)

- We are given a collection of documents written in a three word language {a, b, c}. All the documents have exactly n words (each word can be either a, b or c).
- We are given a labeled document collection {D_1, D_2 ... , D_m}. The label y_i of document D_i is 1 or 0, indicating whether D_i is “good” or “bad”.

- The classification problem: given a document D, determine if it is good or bad; that is, determine P(y|D).

- This can be determined via Bayes rule: $P(y|D) = P(D|y) P(y)/P(D)$

- But, we need to know the parameters of the model to compute that.

A Mult

Notice that this is an important trick to write down the joint probability without knowing what the outcome of the experiment is. The ith expression evaluates to $p(D_i , y_i)$

(Could be written as a sum with multiplicative y_i but less convenient)

- How do we estimate the paramet
- We derive the most likely value of th rs defined above, by maximizing the log likelihood of the observed data.
- PD = $\Pi_i P(y_i , D_i) = \Pi_i P(D_i |y_i) P(y_i) =$

Labeled data, assuming that the examples are independent

  - We denote by $P(y_i) = \eta$ the probability that a example is “good” ($y_i=1$; otherwise $y_i=0$). Then:
- $\Pi_i P(y, D_i) = \Pi_i [(\eta\, n!/(a_i!\, b_i!\, c_i!)\, \alpha_1^{a_i} \beta_1^{b_i} \gamma_1^{c_i})^{y_i} \cdot((1 - \eta)\ \ n!/(a_i!\, b_i!\, c_i!)\, \alpha_0^{a_i} \beta_0^{b_i} \gamma_0^{c_i})^{1-y_i}]$

- We want to maximize it with respect to each of the parameters. We first compute log (PD) and then differentiate:
- $\log(PD) = \Sigma_i\, y_i \quad [\log(\eta) + C + a_i \log(\alpha_1) + b_i \log(\beta_1) + c_i \log(\gamma_1) +$
  $(1- y_i)\, [\log(1-\eta) + C' + a_i \log(\alpha_0) + b_i \log(\beta_0) + c_i \log(\gamma_0)\,]$
- $d\log PD/d\, \eta = \Sigma_i\, [y_i /\eta - (1-y_i)/(1-\eta)] = 0$ ➔ $\Sigma_i (y_i - \eta) = 0$ ➔ $\eta = \Sigma_i\, y_i /m$

- The same can be done for the other 6 parameters. However, notice that they are not independent: $\alpha_0+\beta_0+\gamma_0= \alpha_1+\beta_1+\gamma_1 =1$ and also $a_i + b_i + c_i = |D_i| = n$.

Other Examples (HMMs)

- Consider data over 5 characters, x=a, b, c, d, e, and 2 states s=B, I

he Beginning of each phrase, I is Inside

❑ We can do the same exercise we did before.

❑ Data: $\{(x_1, x_2,\ldots x_m, s_1, s_2,\ldots s_m)\}_1^n$

❑ Find the most likely parameters of the model:
$P(x_i | s_i), P(s_{i+1} | s_i), p(s_1)$

❑ Given an unlabeled example
$x = (x_1, x_2,\ldots x_m)$

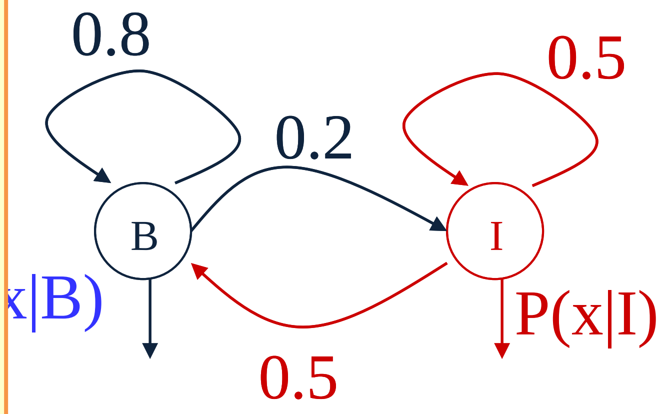
❑ use Bayes rule to predict the label $\ell=(s_1, s_2,\ldots s_m)$:

$$\ell^* = \text{argmax}_{\ell}\, P(\ell|x) = \text{argmax}_{l}\, P(x|\ell)\, P(\ell)/P(x)$$

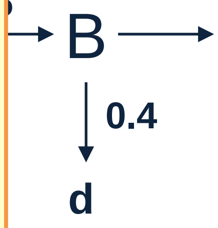
❑ The only issue is computational: there are 2^m possible values of ℓ

❑ This is an HMM model, but nothing was hidden;

  ❑ next week, $s_1, s_2,\ldots s_m$ will be hidden



e observed sequence.



Bayes Optimal Classifier

- How should we use the general formalism?
- What should H be?

- H can be a collection of functions. Given the training data, choose an optimal function. Then, given new data, evaluate the selected function on it.

- H can be a collection of possible predictions. Given the data, try to directly choose the optimal prediction.

- Could be different!

Bayes Optimal Classifier

- The first formalism suggests to learn a good hypothesis and use it.
- (Language modeling, grammar learning, etc. are here)

$$\boxed{\mathbf{h_{MAP} = argmax_{h \in H} P(h \mid D) = argmax_{h \in H} P(D \mid h) P(h)}}$$

- The second one suggests to directly choose a decision.[it/in]:
- This is the issue of “thresholding” vs. entertaining all options until the last minute. (Computational Issues)

Bayes Optimal Classifier: Example

- Assume a space of 3 hypotheses:
  - $P(h_1|D) = 0.4$; $P(h_2|D) = 0.3$; $P(h_3|D) = 0.3$ → $h_{MAP} = h_1$
- Given a new instance x, assume that
  - $h_1(x) = 1$ $h_2(x) = 0$ $h_3(x) = 0$
- In this case,
  - $P(f(x) = 1) = 0.4$; $P(f(x) = 0) = 0.6$ but $h_{MAP}(x) = 1$
- We want to determine the most probable classification by combining the prediction of all hypotheses, weighted by their posterior probabilities

Bayes Optimal Classifier: Example(2)

- Let V be a set of possible classifications

$$P(v_j\,|D)=\sum_{h_i\in H}\underline{P(v_j\,|h_i,D)}P(h_i\,|D)=\sum_{h_i\in H}\underline{P(v_j\,|h_i)}P(h_i\,|D)$$

- Bayes Optimal Classification:

$$v=\text{argmax}_{v_j\in V}P(v_j\,|D)=\text{argmax}_{v_j\in V}\sum_{h_i\in H}P(v_j\,|h_i)P(h_i\,|D)$$

- In the example:

$$P(1|D)=\sum_{h_i\in H}P(1|h_i)P(h_i\,|D)=1\bullet 0.4+0\bullet 0.3+0\bullet 0.3=0.4$$

$$P(0\,|D)=\sum_{h_i\in H}P(0\,|h_i)P(h_i\,|D)=0\bullet 0.4+1\bullet 0.3+1\bullet 0.3=0.6$$

- and the optimal prediction is indeed 0.
- The key example of using a “Bayes optimal Classifier” is that of the naïve Bayes algorithm.

Click here to move to the next lecture

Justification: Bayesian Approach

- The Bayes optimal function is

$$f_B(x) = \mathrm{argmax}_y D(x; y)$$

- That is, given input x, return the most likely label
- It can be shown that f_B has the lowest possible value for Err(f)

- Caveat: we can never construct this function: it is a function of D, which is unknown.
- But, it is a useful theoretical construct, and drives attempts to make assumptions on D

Maximum-Likelihood Estimates

- We attempt to model the underlying distribution

  $D(x, y)$ or $D(y \mid x)$

- To do that, we assume a model

  $P(x, y \mid \theta)$ or $P(y \mid x , \theta)$,

  where θ is the set of parameters of the model

- Example: Probabilistic Language Model (Markov Model):
  - We assume a model of language generation. Therefore, $P(x, y \mid \theta)$ was written as a function of symbol & state probabilities (the parameters).
- We typically look at the log-likelihood
- Given training samples $(x_i; y_i)$, maximize the log-likelihood
- $L(\theta) = \Sigma_i \log P (x_i; y_i \mid \theta)$ or $L(\theta) = \Sigma_i \log P (y_i \mid x_i , \theta))$

Justification: Bayesian Approach

- Assumption: Our selection of the model is good; there is some parameter setting θ^* such that the true distribution is really represented by our model

$$D(x, y) = P(x, y \mid \theta^*)$$

- Define the maximum-likelihood estimates:

$$\theta_{ML} = \text{argmax}_{\theta} L(\theta)$$

- As the training sample size goes to ∞, then

$P(x, y \mid \theta_{ML})$ converges to $D(x, y)$

Are we done?
We provided also Learning Theory explanations for why these algorithms work.

Given the assumption above, and the availability of enough data

$$\text{argmax}_y P(x, y \mid \theta_{ML})$$

converges to the Bayes-optimal function

$$f_B(x) = \text{argmax}_y D(x; y)$$