1

Generalization error is basically the error that our hypothesis attains on the true distribution, and training error is the error that our hypothesis gets on the samples used during training.

The fixed distribution is important because it allows us to make claims about how our performance on the training set will generalize. If the distribution is allowed to change the can we really make any generalization claims after training?

See slide 20 PAC learning. Online learning:

1. No assumptions about the distribution of examples
2. learning as a equence of trials one example at a time updating with mistakes
3. The goal is to bound the total number of mistakes made over time

Batch Learning:

1. Examples are drawn from a fixed distribution
2. Learning uses a training set drawn from our fixed distribution
3. the goal is to find a hypothesis taht has low chance of making a mistake on a new example from our distribution.

2

A concept class C is PAC learnable by L using H if $\forall f \in C$, $\forall D$ over X, and fixed $0 < \epsilon$, $\delta < 1$ given M examples sampled independently according to D, the algorithm L produces with probability at least $(1 - \delta)$ a hypothesis $h \in H$ that has error at most ϵ where m is polynomial

C is efficiently PAC learnable if C is PAC learnable as in the above definition and

1. the number of examples that L takes is bounded by some polynomial in n as well
2. L runs in time asymptotically bounded by some polynomial in n as well.

3

Decision Trees: The hypothesis space is the set of all Boolean functions. There are 2^{2^n} of them. Since $\log |H|$ is exponential, decision trees are not PAC learnable.

Monotone conjunctions/disjunctions. The hypothesis space is the set of monotone conjunctions. The number of monotone conjunctions/disjunctions with n variables is 2^n. Since $\log |H| = n$ is polynomial, these are PAC learnable. The elimination algorithm (COLT silde 25) will learn monotone conjunctions in $O(n)$ time, so they are efficiently PAC learnable. We can write a similar algorithm for disjunctions too.

General conjunctions/disjunctions. The hypothesis space is the set of monotone conjunctions. The number of monotone conjunctions/disjunctions with n variables is 3^n. Since $\log|H| = n \log 3$ is polynomial, these are PAC learnable. They are efficiently PAC learnable. (Find an algorithm similar to elimination that will learn these classes of functions.)

k-CNF's: See COLT slide 61-62. The learning algorithm will look a lot like elimination algorithm (but with a feature transformation to the space of k-conjunctions).

4

How many boolean functions are there? 2^{2^n} so $\log(|H|)$ is exponential and thus not pac learnable.

5

Agnostic learning is when we do not know if the true function is in our hypothesis space. And no, just no...

6

See COLT lecture slide 52 and onward.

7

See COLT Lecture slide 77 and onward.

8

See COLT Lecture slide 103 and onward.

9

Size of hypothesis space and confidence in our solution see COLT slide 54

10

A concept class is weakly learnable if there is a learning algorithm that can produce a classifier that does slightly better than chance. See last COLT slide.

11

Lecture Boosting slide 34

12

$$\gamma = \min_{x_i, y_i} \frac{y_i(w^T x_i + b)}{\|w\|}$$

See (15)

13

$$\text{SVM} = \min_w \frac{1}{2} w^T w + C \sum_i \max(0, 1 - y_i w^T x_i)$$
$$\text{Logreg} = \frac{1}{2\sigma^2} w^T w + \sum_i \log(1 + \exp(-y w^T x))$$

14

SVM: See Lecture SVM slide 46 onward (57) has what you're looking for. Logistic regression: Homework 5 solutions

15

It prevents over fitting look at the Vapnik's theorem on generalization and margins. Let H be the set of linear classifiers that separate the training set by a margin of at least γ then:

$$VC(H) \leq \min\left(\frac{R^2}{\gamma^2}, d\right) + 1$$

where R is the radius of the smallest sphere containing the data. This implies that the larger the margin is the lower the VC dimension is as well. We also know that the lower the VC dimension is the better the generalization is. So Minimizing $\|m\|$ maximizes the margin because:

$$\gamma = \min_{x_i, y_i} \frac{y_i(w^T x_i + b)}{\|w\|}$$

so $\gamma \propto 1/\|w\|$

16

$$\text{Logistic loss}(y, x, w) = \log(1 + \exp(-yw^T x))$$

17

Gradient descent looks at all of the samples before making an update. Stochastic gradient descent looks at a random subset of the samples

18

SVM slide 74 think of the support vectors:

$$w = \sum_{i=1}^{m} \alpha_i y_i x_i$$

19

See Homework 4 solutions

20

See kernel slides 89-92. (1) Yes- is the linear kernel. (2) Yes- is a polynomial kernel of degree 2. (3) Yes- use kernel rules it is the sum of a polynomial kernel and constant multiplied with an exponentiated polynomial kernel.

21

High variance. You are overfitting because, the training set is well fit but the testing set is not very well fit.

22

The regularization term helps reduce variance. It makes the model more general. Think about trying to learn the best fit polynomial line through some dataset. If the feature set looks like this:

$$x = [x, x^2, x^3, x^4, ..., x^100]$$

then it is clear that if we let w be whatever we want this function could fit just about any data set super well. However a 100 degree polynomial can fit a ton of data points perfectly it is probably not going to do very well if there is any noise. Regularization ensures that the model remains more simple.

23

Probabilistic learning/naive Bayes lecture first half. In essence MAP learning is Maximum A posteriori learning. Trying to find the highest posterior probability of a hypothesis given the data, whereas MAP prediction tries to solve the slightly different problem of maximizing the posterior probability of the predictions given the data.

Both naive Bayes and logistic regression are trained using MAP principle and can be used for MAP prediction.

24

See probabilistic learning/Naive bayes slide 29

25

See probabilistic learning/Naive bayes slide 35

26

That all of the features are independent.

$$P(x|y) = \prod_i P(x_i|y)$$

27

See Naive Bayes slide 66 onward

28

This is very similar to the previous question. The only difference is that the features are real valued.

To make this simpler, for now let us assume that the standard deviations are constant σ that is known. The model states that the probability of the i^{th} feature taking value x_i when the label is 1 is defined by a normal distribution with mean $\mu_{1,i}$.

$$P(x_i|y=1) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x_i-\mu_{1,i})^2}{2\sigma^2}\right)$$

When the label is 0, this is defined by a normal distribution with mean $\mu_{0,i}$ as

$$P(x_i|y=0) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x_i-\mu_{0,i})^2}{2\sigma^2}\right)$$

Using indicator functions defined in slide 68, we get

$$P(x_i|y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-[y=1]\frac{(x_i-\mu_{1,i})^2}{2\sigma^2}\right) \exp\left(-[y=0]\frac{(x_i-\mu_{0,i})^2}{2\sigma^2}\right)$$

This can be simplified into

$$P(x_i|y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-[y=1]\frac{(x_i-\mu_{1,i})^2}{2\sigma^2} - [y=0]\frac{(x_i-\mu_{0,i})^2}{2\sigma^2}\right)$$

If $P(y=1)=p$, we can now use the naive Bayes assumption to write down the probability of a labeled example $(\mathbf{x}, y)$, where $\mathbf{x}$ is an n dimensional vector. We have

$$P(\mathbf{x}, y) = p^{[y=1]}(1-p)^{[y=0]} \prod_i \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-[y=1]\frac{(x_i-\mu_{1,i})^2}{2\sigma^2} - [y=0]\frac{(x_i-\mu_{0,i})^2}{2\sigma^2}\right)$$

Given a dataset $S = \{(\mathbf{x}_j, y_j)\}$, we can now write down the log-likelihood of the data as

$$\begin{aligned}\text{Log-likelihood}(S) = \sum_j &\left([y_j=1]\log p + [y_j=0]\log(1-p)\right) \\ &- \log(\sigma\sqrt{2\pi}) - [y=1]\frac{(x_{ji}-\mu_{1,i})^2}{2\sigma^2} - [y=0]\frac{(x_{ji}-\mu_{0,i})^2}{2\sigma^2} \qquad .\end{aligned}$$

Take the derivative of this and set it to zero to get the MLE for the means as

$$p = \frac{\text{number of examples with label 1}}{\text{number of examples}}$$

$$\mu_{1,i} = \frac{\text{mean of all the } i^{th} \text{ feature for all examples labeled by with 1}}{\text{number of examples labeled with 1}}$$

$$\mu_{0,i} = \frac{\text{mean of all the } i^{th} \text{ feature for all examples labeled by with 0}}{\text{number of examples labeled with 0}}$$

29

Generative classifiers learn a model of the joint probability $P(x, y)$ of the inputs x and label y, and mkae their predictions by using Bayes rules to calculate $P(y|x)$. A Discriminative classifier models the posterior $P(y|x)$ directly.

30

Logistic regression slides slide 29 onwards.

31

This is the EM slides example.