Exam 1 SG

Friday, February 23, 2018 8:54 PM

- If you are ever working in a data science role, you have to learn the CRISP-DM methodology, w process model. We will tie different aspects of the CRISP-DM cycle into the projects of the cou
  - The business understanding is where you are figuring out what you are trying to solve fo scientific progress.
  - The data understanding is when you try to find and comprehend the data you have to w
  - The data preparation is when you modify the data you have collected to fit the record st modeling process
  - The modeling process is where you determine the function that describes new knowled previous attributes
  - The evaluation process is when you evaluate the accuracy of your model, as well as the problem presenters
  - The deployment process is how you utilize your new model for exterior gain.

- Classifying takes in a set of data, and splits it into training and test data. It assigns a class attrib the attributes in the row. It takes training data that has preassigned classes, and builds a mod data. The test data is then used to determine the goodness of the model. The end goal is to be unseen records with a certain degree of confidence.
- Clustering finds data points that are similar in one another according to their attributes. These be things like Euclidian distance or other things. The data points that are similar are all assigne to find all the clusters for a data set, and be able to assign new attribute vectors to existing clu
- Association rules take a set or records with a number of items from a given collection, and pro that will predict if another item will appear in the collection based on previous items.
- Data preprocessing takes un/semi structured data and turns it into structured data. This mean databases and text documents turn into things like R dataframes and tables.
- There are types of attributes:
  - Nominal, like ID numbers or eye color. These are things that just represent identifying n to distinguish one attribute type from another.
  - Ordinal, which are rankings. These allow you to order attributes in a specific way, and co
  - Intervals, which are things where intervals between attribute values make sense, like da and distances between attribute values.
  - Ratios, which are things where both intervals and ratios make sense to look at. You wou

which is a data mining
urse.
or financial gain or

work with.
tructure used in the

ge from the data using

gain it delivers to your

bute based on a subset of
el based on the training
e able to classify previously

e similarity measures can
ed to one cluster. We want
usters.
oduce dependency rules

ns things like relational

ames for data. Allows you

ompare items.
ates. You can look at times

uldn't say that march 2nd is

twice as large as march 1st, but you would say that 2 minutes is twice as long as 1 minu
variables are intervals, but with the added condition that 0 of the measurement variabl
none of that variable.

- Attributes can be discrete or continuous. Discrete attributes are finite or countably infinite, ar
ordinal. Continuous variables have real numbers as attribute values, and are typically interval

- The dot product of two vectors is just the sum of the product of their corresponding entry val
is just a1b1 + a2b2.

Length

The *Euclidean norm* or *length* of a vector $\mathbf{a} \in \mathbb{R}^m$ is defined as

$$\|\mathbf{a}\| = \sqrt{\mathbf{a}^T\mathbf{a}} = \sqrt{a_1^2 + a_2^2 + \cdots + a_m^2} = \sqrt{\sum_{i=1}^{m} a_i^2}$$

Distance

From the Euclidean norm we can define the *Euclidean distance* between $\mathbf{a}$ and $\mathbf{b}$, as follows

-
$$\delta(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\| = \sqrt{(\mathbf{a} - \mathbf{b})^T(\mathbf{a} - \mathbf{b})} = \sqrt{\sum_{i=1}^{m}(a_i - b_i)^2} \quad (1.1)$$

Angle

The cosine of the smallest angle between vectors $\mathbf{a}$ and $\mathbf{b}$, also called the *cosine similarity*, is given as

$$\cos\theta = \frac{\mathbf{a}^T\mathbf{b}}{\|\mathbf{a}\|\,\|\mathbf{b}\|} = \left(\frac{\mathbf{a}}{\|\mathbf{a}\|}\right)^T\left(\frac{\mathbf{b}}{\|\mathbf{b}\|}\right) \quad (1.3)$$

Thus, the cosine of the angle between $\mathbf{a}$ and $\mathbf{b}$ is given as the dot product of the unit vectors $\frac{\mathbf{a}}{\|\mathbf{a}\|}$ and $\frac{\mathbf{b}}{\|\mathbf{b}\|}$.

- A percentile, given an ordinal or continuous attribute x and a number p between 0 and 100, th
xp of x such that p% or the observed values of x are less than xp.

- The **variance** is the most common measure of the spread of a set of points.

$$\text{variance}(x) = s_x^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \overline{x})^2$$

- The **standard deviation** is the square root of the variance.
- However, this is also sensitive to outliers, so that other measures are often used.

-
$$\text{AAD}(x) = \frac{1}{m}\sum_{i=1}^{m}|x_i - \overline{x}|$$

te. Jason says that ratio
e indicated that there is

nd are typically nominal or
or ratio typed.

ues. So [a1,a2] dot [b1,b2]

he pth percentile is a value

$$\text{MAD}(x) = median\left(\{|x_1 - \overline{x}|, \ldots, |x_m - \overline{x}|\}\right)$$

$$\text{interquartile range}(x) = x_{75\%} - x_{25\%}$$

- Covariance looks at every attribute value in a dataframe of two attributes. It is the sum of the corresponding attribute value divided by the number of rows -1. Think of it like scrolling throu multiplying the two attribute values together into a new attribute, summing that new attribut dividing by the number of rows in that vector -1.
- $$s_{j,k} = cov(x_j, x_k) = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)$$
- The covariance matrix is the matrix comprised of every possible covariance in your dataset. So of the matrix is your first attribute covarr'ed with all the other attributes, the second row is th second attribute, and so on and so forth. The correlation between two attributes is their entry (their covariance) divided by the square root of the products of their individual variances:
- $$r_{j,k} = \frac{s_{j,k}}{\sqrt{s_{j,j}s_{k,k}}}.$$
- Correlation measures the linear relationship between two continuous attributes.
- # Bayes Theorem

$$P(Y|X) = \frac{P(X|Y)}{P(X)}P(Y)$$

Posterior Probability = Support x Prior Probability

1. Start with some prior "belief" (P(Y)) and observe data X.
2. P(X | Y) is the likelihood of observing X given our belief Y. You want to normalize this by the likelihood of observing X (regardless of Y). The ratio of these two numbers is the support.
3. Update your belief in Y with this new information (P(Y | X), the posterior probability).

Key Idea: If the support is greater than 1, then the observed data X will increase your belief in Y

products of every
ugh the dataframe,
ce vector up, and then

o everything in the first row
he same thing but with the
y in the covariance matrix

data X will increase your belief in Y.

- P(X) = P(X|Y)P(Y) + P(X|!Y)P(!Y).
- So we adjust our prior/initial belief in Y by first assuming we did get Y, then testing X based on to get P(X), we look at the opposite, where we look at the probability of not getting Y and wha on not getting Y. Then we use the formula to get our updated belief in X given what we know

- **Proximity** refers to either similarity or dissimilarity between points/attribute values. Distance dissimilarity. To scale dissimilarities, use this formula:
  - (original dissim - minimum dissim value) / (max dissim value - min dissim value), where values from your original range of dissimilarities.
- The Euclidian distance is:
  - $$dist = \left(\sum_{k=1}^{n} | p_k - q_k |^r \right)^{\frac{1}{r}}$$
  - Where r = 2. You iterate through the attribute values, adding them together after subtra them. Then, you square root the result.
- The simple matching coefficient is used only for binary attributes. It is denoted as the numbe negatives divided by the number of all outcomes. Jaccard is an offshoot, and only calculates th matches, divided by the number of all attributes except for true negative matches. It is more a vectors with a lot of matching true negatives:
- $$SMC = \text{number of matches} / \text{number of attributes}$$
  $$= (M11 + M00) / (M01 + M10 + M11 + M00)$$
  $$J = \text{number of 11 matches} / \text{number of not-both-zero attrib}$$
  $$= (M11) / (M01 + M10 + M11)$$
- Cosine similarity treats the two attribute rows in the dataframe as vectors. It is the dot produc divided by the product of their norms:
- $$\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2||$$
- Remember that with proximity, different attributes have different scales so you should scale y computing distance.

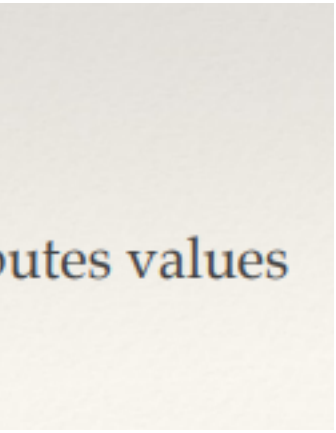
n that to get P(X|Y). Then,
at P(X|Y) *would* be based
about Y.

is a common term for

the max and mins are the

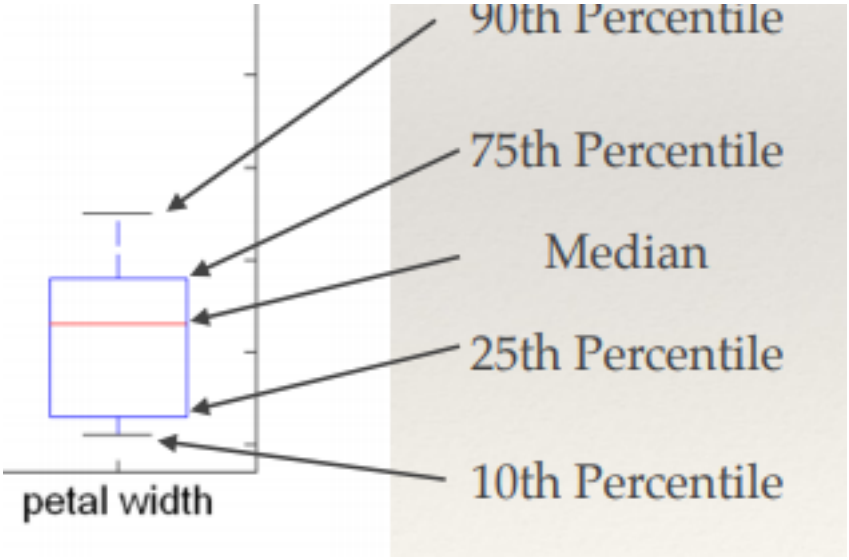
acting them, squaring

r of true positives and true
he number of true positive
appropriate for attribute



ct of the two vectors

your data before

- 

- **Noise** in data refers to modification of original values, it can occur to measurement error or h numerous other things. It creates flawed biased data.
- Outliers and missing values are also big problems in data mining.

Data Preprocessing

- This is a technique used to make data more suitable for data mining. You select data objects a or create/change attributes so that mining is easier. There are many types:
    - Aggregation combines two or more attributes into a single attribute, so as to reduce dir create more stable data with less variability. You can lose interesting data this way too.
    - Sampling is done by only looking at one or a couple of portions from an entire dataset. Y without replacement, so as to repeat draws or not from your data. If the sample is repr dataset, it will work almost as well as if you were using the entire thing.
- The curse of dimensionality is when you have higher dimensions of attributes in your datafram becomes increasingly sparse (more empty values). Distance and density become less meaning
- Dimensionality Reduction methods are done to reduce the number of attributes in one row of creating new attributes as combinations of your old attributes.

Principle Component Analysis

- An eigenvector is a vector is a vector associated with a transformation matrix that does not ch multiplied by the matrix, just magnitude. An N x N matrix has N eigenvectors. You can scale it the transformation matrix and it will remain the same, just larger in magnitude. Also, eigenvec with a magnitude of 1. So if you find a bigger one, you just scale down.
- Eigenvalues are associated with eigenvectors. The amount the eigenvector is scaled up by wh transformation matrix is the eigenvalue. So if the eigenvector was 1, 1 before and becomes 4,
- To find the eigenvectors of a matrix, create a candidate set of likely vectors, multiply them by see which entries all scale up by the same amount. That amount is your eigenvalue, and the v eigenvector!

uman error, among

nd attributes for analysis,

mensionality, change scale,

You can sample with or
esentative of your entire

me, the available data
ful.
f your dataframe by

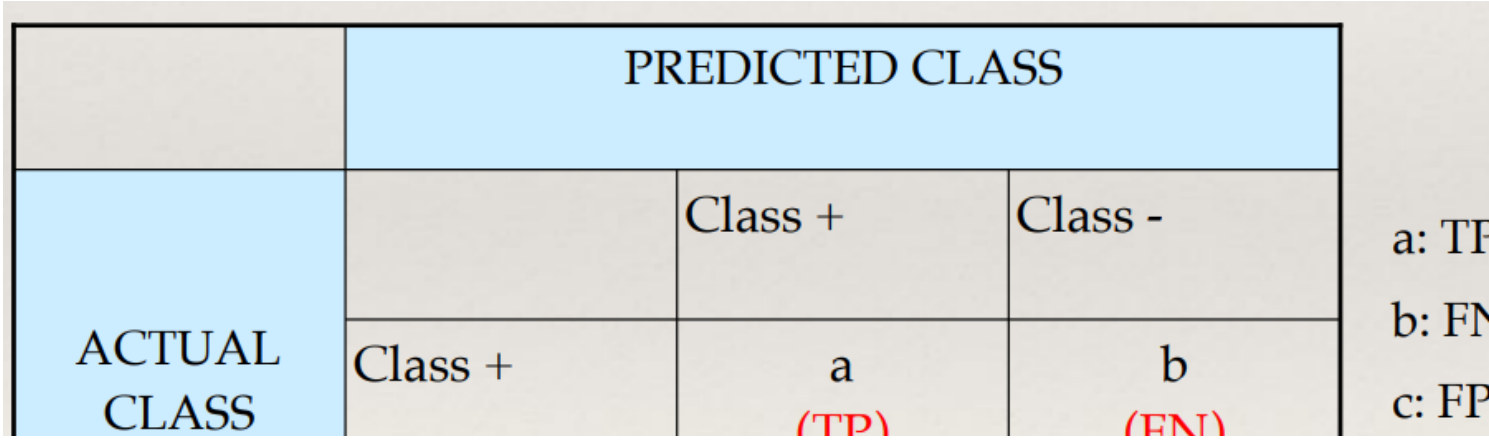
hange orientation when
before multiplying with
ctors are usually expressed

en multiplying by the
4, then the eigenvalue is 4.
your square matrix, and
ector pre scaling is an

- PCA is a strategy to compress data from a number of dimensions to a smaller amount of dime still contain a lot of the interesting variation in the data, but are easier to work with as less dir
    - Step 1 is to get data.
    - Step 2 is to subtract the average of each attribute column from every element in that co
    - Step 3 is to Calculate the covariance matrix of your dataset
    - Step 4 is to Find the eigenvalues and eigenvectors of the covariance matrix, and make su eigenvalues/vectors, IE, their magnitude is 1.
    - In step 5, the eigenvector with the highest eigenvalue is the principle component of the vector that most closely points down the middle of both attributes on a scatterplot of y order the eigenvectors by the value of their eigenvalues, and construct a matrix of the e vector) you want to keep. If an eigenvector has a low eigenvalue, you can eliminate it fr losing a lot of information about your original dataset.
    - Step 6 involves deriving the new dataset. You transpose your dataset and the feature ve everything together. This expresses your dataset in the terms of your feature vector, wi dimensions of your feature vector. So if you have a 1 eigenvector feature vector, you re dimension.

Classification/Model Evaluation

- The input data for a classification task is a tuple (x,y), where x is a row of attributes, and y is th row.
- Classification is the task of learning a function f that maps each attribute set x to the class y.
- Class labels are pretty much only binary or nominal.
- The generalization capability of a classifier is its ability to correctly map previously unseen dat this, it creates a model based on a training set, which is a dataset with a set of attributes ident is a set of rows of attributes that must be classified.
- To test the performance of our model, we have to use a confusion matrix, which denotes whe was correctly classified or not:



		PREDICTED CLASS	
		Class +	Class -
ACTUAL CLASS	Class +	a (TP)	b (FN)

a: TP
b: FN
c: FP

nsions. Those dimensions
mensions = less work.

olumn.

ure they are the unit

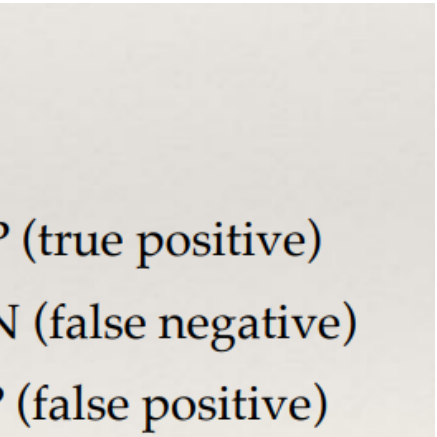
dataset. This represents a
our attribute data. You
eigenvectors (feature
om this matrix without

ector, and multiply
th the dimension being the
duce your dataset to one

ne class assigned to that

a to its class label. To do
tical to the test set, which

ether a class in the test set



		(TP)	(FN)
	Class -	c (FP)	d (TN)

d: TN

- True positive is when it correctly classifies something, as well as true negative. False negative errors.

- $$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+}{TP+TN+}$$

- Error Rate = (FN + FP) / (TP + FN + FP + TN) =
- True Positive Rate (TPR) or Sensitivity = TP / (T
- True Negative Rate (TNR) or Specificity = TN /
- False Positive Rate (FPR) = FP / (FP + TN) = 1 -
- False Negative Rate (FNR) = FN / (FN + TP) = 1

- $$\text{Precision (p)} = \frac{a}{a+c}$$
- $$\text{Recall (r)} = \frac{a}{a+b}$$
- $$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

N (true negative)

and false positive are

$$\frac{TN}{FP + FN}$$

= 1-Accuracy
TP + FN)
(TN + FP)
TNR
1 - TPR

Decision Trees

- Decision tree classifiers have three elements, a root node making an initial decision, internal n subsequent decisions, and leaf nodes representing ultimate classifications of rows. Root and i two or more edges pointing out representing different questions about a single attribute in th
- Hunts Algorithm is a way to recursively create a decision tree:
    - Step 1 is if all rows in a set of rows with class labels are of the same class label, create a label.
    - Step 2 is if all the rows in a set of rows with class labels are not of the same class label, t condition to split them up into separate sets of data with minimal difference in the class and apply the algorithm to all the new separate sets of data.
- So we need both a good way to know how to split test data into different subsets, and to know new nodes in the tree. First we will look at splitting on attributes.
    - We can split nominal and ordinal attributes into their own edges, or groups of them into
    - We can split continuous data into intervals, like <, >, or x < y < z.
- There are measures we can use to determine the best way to divide rows into different sets b These are called Entropy, Gini, and Classification Error. They measure leaf node impurity. The from these measurements are usually the best possible split based on an attribute.
- The Gini index is written as:
    - $$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$
- Where p(j|t) is simply the amount of a type of class in one of the resulting subsets after an att (class 0 / # rows in subset)^2 - … (class n / # rows in subset)^2. To figure out the quality of a split equation (you want to minimize GINI):
- $$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,

n = number of records at node p.

nodes which represent
nternal nodes can have
e row.

leaf node for this class

then find some optimal test
s labels. Then, go through

w when to stop creating

o their own edges.

ased on attribute edges.
lowest value of impurity

tribute split. So it is 1 -
using GINI, use this

- When splitting continuous attributes, line all of the attributes up into a row sorted ascending. that are in between the sorted actual values. Then, test for <= and > on all the continuous valu

Cheat	No		No		No		Yes		Yes		Yes		
	Taxable Income												
	60		70		75		85		90		95		
→	55		65		72		80		87		92		97
→	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3
No	0	7	1	6	2	5	3	4	3	4	3	4	3
Gini	0.420		0.400		0.375		0.343		0.417		0.400		0.30

- Another metric you can use to split up attributes and look at how pure they are is entropy. Ins score, it uses the information gain score to judge a split (you want the lowest value for pure e

- $$Entropy(t) = -\sum_{j} p(j \mid t) \log$$

- $$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy \right.$$

- Information gain is defined as the above, Where k is the number of attributes values the split number of records in the parent node you are splitting up, ni is the number of records in the p node, and Entropy(i) is the entropy of the partitioned child internal node. You want the larges
- Problems can arise when using info gain, as splitting into a large number of sets with very few lead to less reliable predictions, so we use the gain ratio to adjust information gain by penalizi partitions:

Then, find discrete values
ues:

No	No	No	No
100	120	125	220

	110		122		172		230	
>	<=	>	<=	>	<=	>	<=	>
0	3	0	3	0	3	0	3	0
4	4	3	5	2	6	1	7	0
00	0.343		0.375		0.400		0.420	

stead of the GINI split
ntropy):

$$\log_2 p(j \mid t)$$

$$(i)\Big)$$

can take on, n is the
partitioned child internal
st gain possible.
records in each. This can
ing large numbers of small

partitions.

- $GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO}$ $SplitINFO =$
- Error is another metric:
- $Error(t) = 1 - \max_i P(i$
- To measure the gain in error, just use information gain but with error stuff instead.
- If you have to find the information gain of an entire dataset of data, Entropy(p) is actually the classes of the rows.
- There are two ways to handle tree overfitting when inductively growing a tree. The first is to p some good stopping condition for when you do not need to split up an impure node any furth involve if all classes or attribute values are the same, or if the number of rows in the new nod specified constant. Another stopping measure is if the Gini split or info gain score is not impro created. This can be problematic.
- The other option is post pruning. In post pruning, we try to simplify the decision tree to avoid two ways, through subtree replacement, where we replace a subtree in the tree with its majo raising, where we replace the contents of an entire subtree with its most often used branch. A split is bad is to look at the error before and after splitting, and if the error is higher, you prun node using subtree replacement with the majority class.
- Decision trees can suffer from data fragmentation, as the leaf nodes could be too small to ma significant decision. Also, finding an optimal tree is NP Hard.

Estimating Generalization Error

- It is generally agreed that the complexity of a model has a factor in its tendency to overfit data to estimate the generalization error, which is the error that will occur when classifying unknow ways to do this, using Resubstitution Error (the error of the dataframe used to train the mode error, pessimistic error, and validation set error.
- Optimistic error just assumes that the error for the test set will be the same as the error for th your training data. This is typically not the best way to handle error. The easy way to find opti tree is to just count the number of rows in a leaf node that are not in the majority class, and d total number of rows. So go through every box, find the smaller number, and add it to every c divide by the total number of rows classified.
- Pessimistic error thinks generalization error is the sum of training error and a penalty term for means a more complex model will result in a higher error. This is done to comply with Occam'

$$-\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

$$i \mid t)$$

entropy of just all the

re prune the tree, or have
er. Typical conditions
e is less than a user
ved from the last node

overfitting. We do this in
rity class, and subtree
A good way to test when a
e things into just one leaf

ke any significantly

a. So there has to be a way
wn rows. There are three
l) known as optimistic

he decision tree created by
mistic error in a decision
livide that amount by the
other smaller number and

r model complexity. This
s Razor, which says that

simpler model is better. To do this, when counting the number of error cases in your leaf node total number of rows, add on this amount: (penalty multiplier * number of leaf nodes).

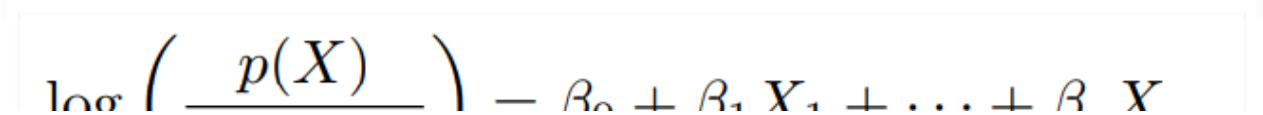
- Validation set error is one way to measure generalization error. It is also called reduced error separate validation set to measure the error. It just uses a separate set to compute error value validation and 2/3 for training the model. This is the best way to approach generalization error validation set in the optimistic mode. There are a couple of ways to make validation sets:
    - Single holdout just reserves 2/3 of a set of data for training and 1/3 for testing, or variat
    - Random subsampling just repeats single holdout k times with different selections of row
    - Cross validation partitions data into k disjoint subsets, trains on k-1 of those partitions a It then repeats, ensuring that all k subsets are used as training sets. It uses the averages sets to predict the eventual generalization error.
    - Bootstrapping samples with replacement from the original dataset, with each bootstrap same size as the original dataset.

ROC Curves

- The precision of a classifier refers to its ability to correctly identify actual positive classes as po false positives, you have low precision.
- The recall of a classifier refers to its ability to correctly classify positive examples as positive, n lot of false negatives, you have low recall.
- The F measure is the harmonic mean of recall and precision. It is a good way to build a strong
- A ROC curve plots the percentages of your True Positive Rate on the y axis and False Positive R want the ROC curve generated by your classifier to hang towards the upper left corner.
- To draw a ROC curve, the classifier needs to output the likelihood that a specific record will be
- The steps are as follows:
    - Sort the likelihoods that the classifier outputs from lowest to highest.
    - Start with the lowest record. Assign it and every record above it to the positive class, an positives, the false positives, the true negatives, and the false negatives. Next, obviously rate and the false positive rate as the first point on the curve.
    - Next, move up to the next column in the table. Classify the new column as positive, as w right of it as positive, and the columns to the left of it as negative. Recompute TP, FP, TN accordingly. Repeat this until you hit the highest likelihood of positivity record column.
- And that's how you do it.

Logistic Regression

- A logistic regression takes on the form of:



es, before dividing by the

pruning, which uses a
es, so like 1/3 of the set for
r. Also, you run the

tions of those proportions.
ws
nd tests on the remaining.
of the error on these test

validation set having the

positive. If you have a lot of

not negative. If you have a

classifier.
Rate on the x axis. You

e positive.

d then compute the true
y calculate the true positive

well as everything to the
N, FN and graph the rates

- $$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$$

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}{1+e^{\beta_0+\beta_1 X_1+\cdots+\beta_p X_p}}$$

- It is a function that is created based on a set of data. Bo is a variable that is estimated based o intercept. B1 and subsequent variables are also estimated, and these are labeled based on the

KNN

- KNN works as follows:
  - Take a single row of data
  - Compare its distance using a proximity measure to every other point in your model (wh rows)
  - Find the k nearest points, and the class of the new row is the class of the majority of tho
  - You can also weight the distances by taking the distance, squaring it, dividing 1 by it, an largest value instead of the smallest value when looking at the k neighbors.
  - KNN are lazy, they do not build models explicitly, they just use training data. They are su are based on local info instead of a built global model.
  - A good value for K is usually around 10 or so. Small K overfit, large k underfit.

Rule Based Classifiers

- A rule based classifiers is a set of rules r that are comprised of implications. The antecedent st attribute tests, and the consequent is always a class.
- We say a rule covers a row if the antecedent matches that row, or the attributes in the row m
- The coverage of a rule is equal to the fraction of records in the dataframe that trigger the rule fraction of records that trigger the rule AND are also correctly classified by the rule, with the c being the rows the rule covers.
- A good ruleset is mutually exclusive, and this occurs when no two rows trigger the same rule. meaning each row triggers at least one rule. Combining these, we ensure that every row is cov
- If we cannot accomplish this, we use the default rule, which is assigned to the rows that do no rules in the ruleset.
- If a ruleset is not mutually exclusive, we can deal with this in two ways:
  - Ordering the rules, meaning the first rule triggered assigns the class.
  - Using the rule classes as a vote, assigning the row the class with the most amount of rul
- The Sequential Covering method algorithm is as follows:
  - Start from an empty rule
  - Grow a rule using the Learn One Rule function

n the data called the
e attributes in the data.

ich is just another set of

se k points.
d then looking for the

usceptible to noise, as they

ates a conjunction of

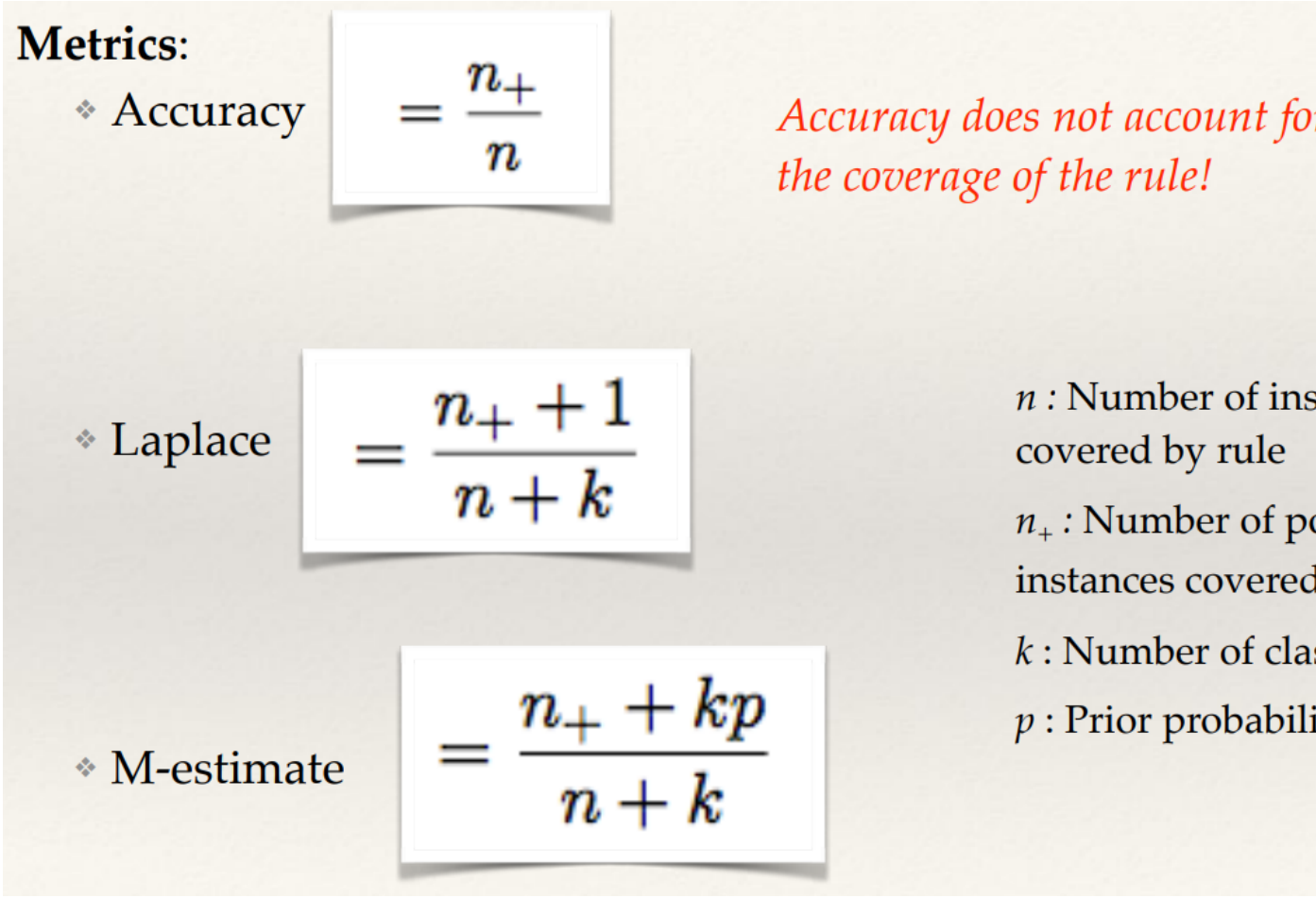
ake the antecedent true.
. The accuracy is the
denominator of the fraction

It is also exhaustive,
vered by exactly one rule.
t trigger any of the other

e votes.

- Grow a rule using the Learn One Rule function
- Remove training records covered by the rule
- Repeat steps 2 and 3 until you meet stopping criteria

- The LOR function attempts to extract a rule that covers many positive examples and few nega
- You can either start with a really specific rule, or a really general rule as your empty rule.
- There a couple of ways you can evaluate a rule:

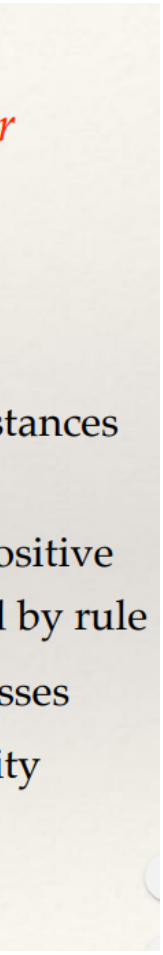


- Typically the prior is given in most questions.
- Two ways to know how to stop is to set a stopping criterion by measuring gain on accuracy/la gain is not significant you discard the rule. The other is rule pruning, where you remove one o and compare the error rate on a validation set before and after pruning. If error is better, pru

Naïve Bayes

- Bayseian Classifiers take a statistical approach to classification. It considers each attribute and variables. Given a record with attributes (x1, x2, … , xN), the goal is to predict class C. Specifica value of C that maximizes P(C | X1, X2, … XN).
- The approach is to use Bayes Theorem to predict the probability of the class given all the attri this, we need to find:
  - The probability of all the attributes together given a class P(X | C)
  - Times the probability of the class P (C)

tive examples.



place/m-estimate. If the
f the conjuncts in a rule,
ne.

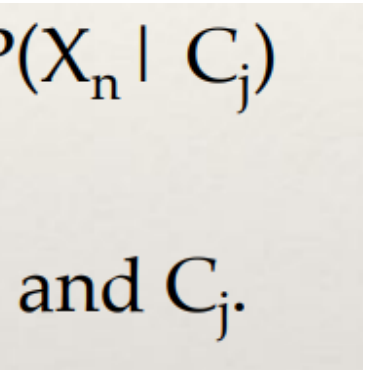
l class label as random
ally, we want to find the

bute values, and to find

    - With all of this divided by the probability of all the attributes. P(X)
    - P(X) gives kind of a scaling factor or normalizing factor, not very interesting. We are inte
- The primary question is how to estimate P(X | C). Bayes assumes the variables are independer that P(X|C) can be factored into a product of multiple, per attribute likelihoods. EX: P(X1, X2, . P(X2 | C) * … * P(XN | C).
- $P(X_1, X_2, \ldots, X_n \mid C) = P(X_1 \mid C_j)\, P(X_2 \mid C_j) \ldots P$

  ❖ Can separately estimate $P(X_i \mid C_j)$ for all X_i
-
- This algorithm is naïve since most attributes in an attribute vector are not independent of eac some correlation between attributes in an attribute vector. Bayes throws that out the windo
- For discrete attributes, P(Xi | Ci) = number of instances of rows having attribute Xi and class C times the class Ci appears in the dataset.
- If you have a continuous attribute, you can discretize the range into bins, like low, medium an estimate a probability distribution density estimation, by assuming the attributes are normall other distribution), and use the data to estimate the parameters for the normal distribution fo the distribution to estimate the probabilities of our continuous attributes. This is done using t sample variance of the data, which is plugged into this equation that outputs a probability:
- $$P(X_i \mid C) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{1}{2}\frac{(X_i-\mu)^2}{\sigma^2}}$$
- For example, if you have a row with three attributes, then P(X|C) = P(first attribute | C) * P(se P(third attribute | C).
- The slide 125 skips the step denoting the fact that P(X|Class = No) is equivalent to P(Class = N
- We run into an issue when one of the probabilities in the algorithm is zero. To fix this, we do L adds one to the number of attributes of type Xi that belong to class Ci, and adds the number o in the data set to the denominator, which if you recall is the number of classes in the dataset here:
- Original: $P(X_i \mid C) = \frac{N_{ic}}{N}$

erested in P(X|C)*P(C).
nt given class. This means
…, XN | C) = P(X1 | C) *

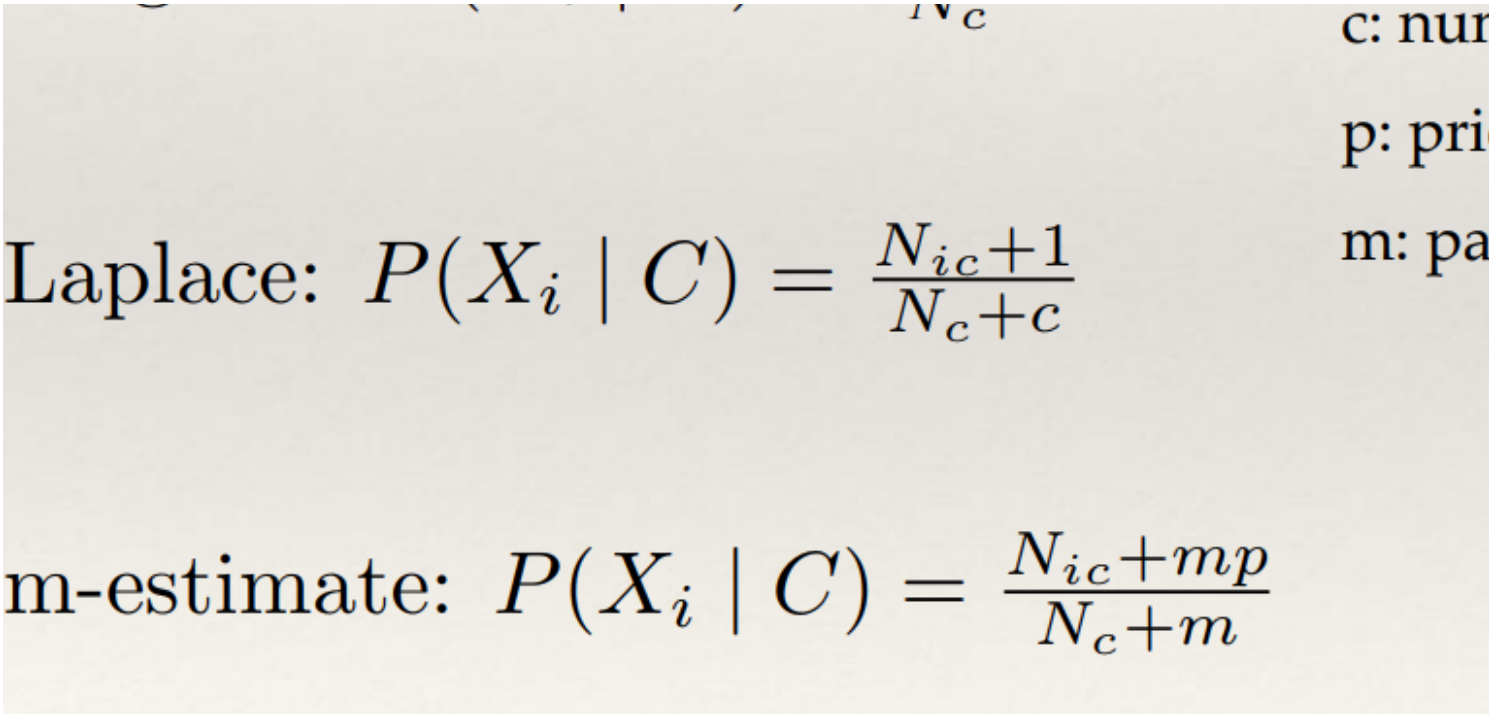


h other. There is generally
v.
i divided by the number of

d high. You can also
y distributed (or some
or each class, and then use
he sample mean and the

cond attribute | C) *

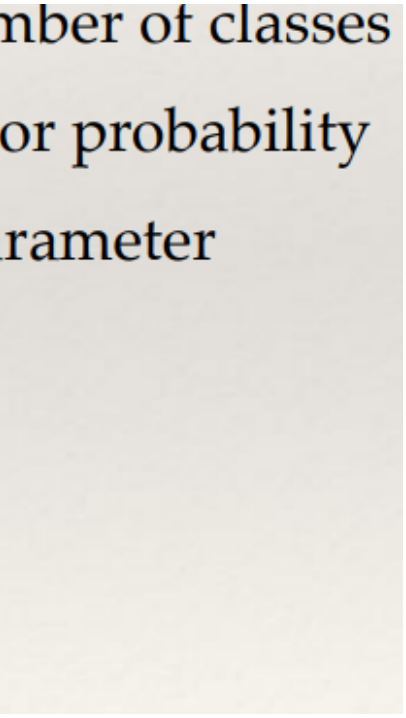
o | X1, X2…, XN).
Laplace Smoothing, which
of records with the class Ci
total. The equations are

- 

- Bayesian Classifiers have a goal to choose a value of C such that P(X1, X2, …, XN | C) * P(C) is exists to justify this maximization, that basically says that the maximum probability of class cre classifier.
- Bayes decision boundaries on simulated training data describe the lowest possible test error r maximum classifier probability/accuracy of classifier. Your model running on the test data can lower than the one that the BDB describes.
- In the real world, we do not have the ability to create the BDB as we cannot compute the actu of a collected dataset. This is why we need to come up with estimates for the true and unknow data. We hope that the algorithm that we use to estimate P(+ | x) from the data is close to the probability, so we can have a pretty solid test error rate.
- Don't worry too much about Bayesian Classifiers, he skipped the example slide.

Artificial Neural Networks

- An ANN is an interconnected assembly of nodes and directed edges that can be used to learn between a set of inputs and outputs. The arrangements of these nodes and edges is called the network.
- A ANN is made up of perceptrons linked together. The input node of the perceptron transmits the outgoing link without any transformation. The output node is a mathematical device that sum of its inputs, subtracts the bias term, and then produces an output that depends on the s can output different things as well based on the activation function it chooses. Here is the equ function of a sign AF:



maximized. A theorem
eates the most accurate

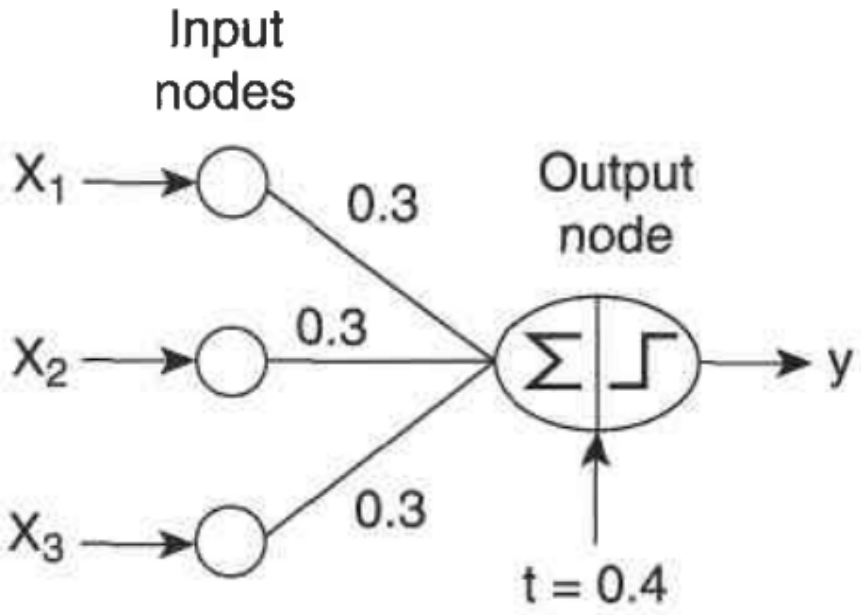
ate, which is based on the
not have an error rate

ual posterior probabilities
wn probabilities from the
e true posterior

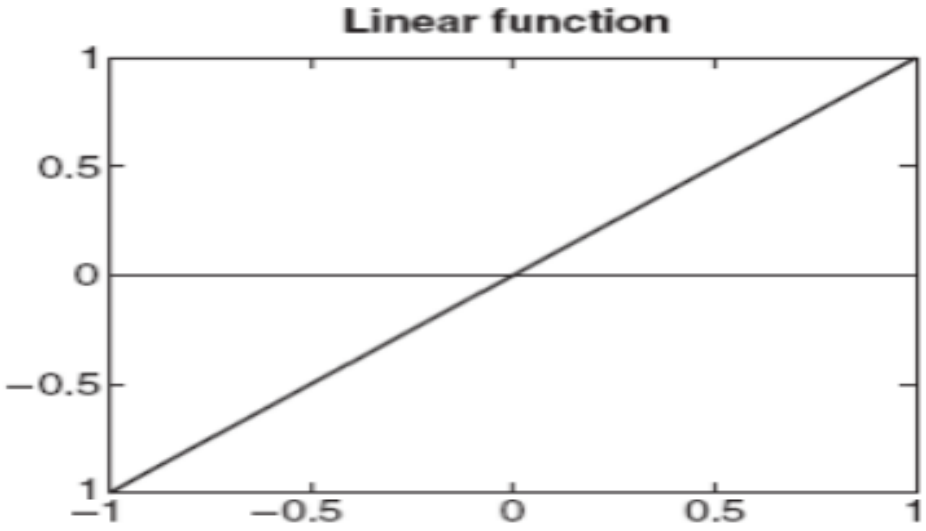
the functional relationship
e architecture of the

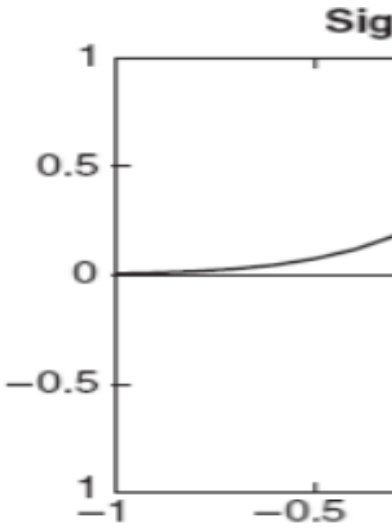
s the value it recieves to
computes the weighted
ign of the resulting sum. It
uation for the activation

- $y = sign(w_d x_d + w_{d-1} x_{d-1} + \ldots + w_2 x_2 + w$
- Where w is the value of the weights that deliver the x attribute to the perceptron, and t is the
-

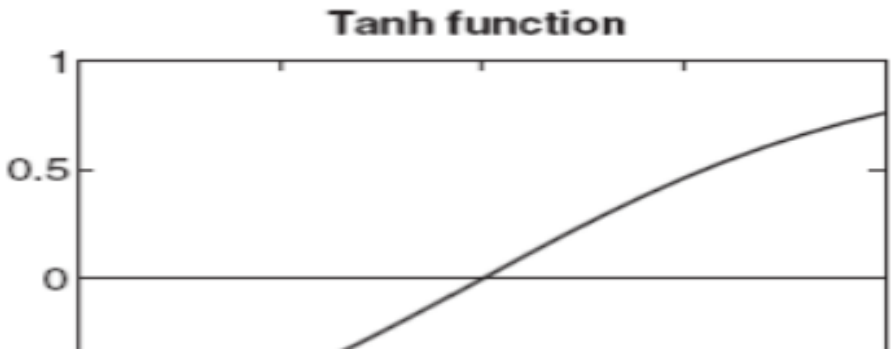


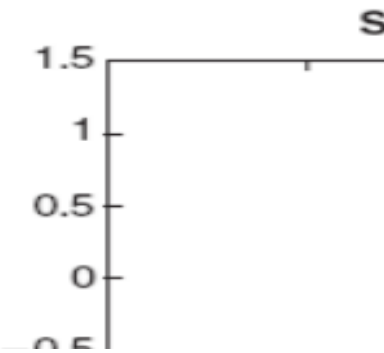
- Activation functions:





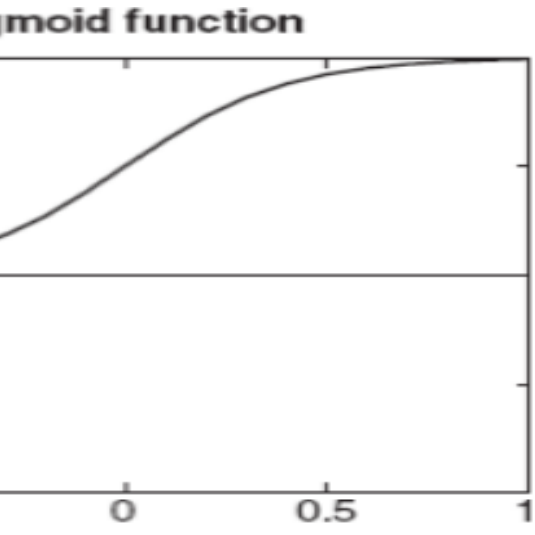
-

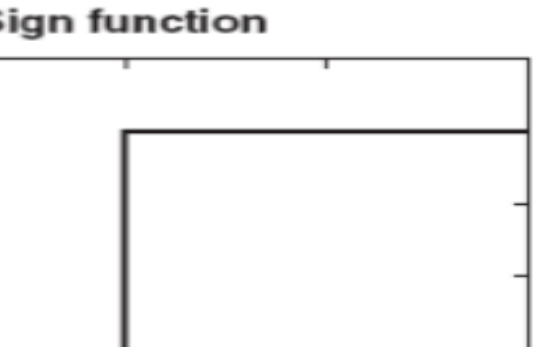


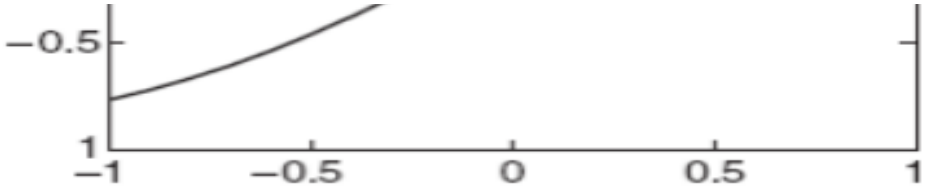


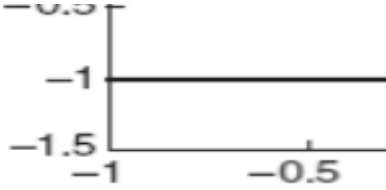
$v_1 x_1 - t)$

bias term:





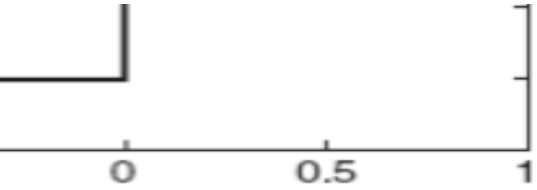




- The learning algorithm for an ANN is as follows:
  - Give each input weight a random value
  - For every training example, compute the predicted output.
  - For every weight, update the weight using the weight update formula:
  - $$w_j^{(k+1)} = w_j^{(k)} + \lambda\left(y_i - \hat{y}_i^{(k)}\right)x_i$$
- Where the new weight, wjk+1, is the old weight plus the user specified learning rate times the
  times the actual class of the row minus the predicted output of the perceptron.
- A single perceptron only separates linearly separable data. It will not converge with data that
- To get around this, an ANN is constructed with many perceptrons, with an initial layer being t
  layer being a bunch of perceptrons in the middle, and the output layer outputting the final y v
  this network, you adjust the weights in a way that minimizes the objective function so the AN
  class labels of your training examples:
  - *Objective function*:
  - $$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2.$$
- Backpropagation has two phases. In forwards phases, the weights obtained from the previous
  compute the output value of each neuron in the network, starting at the input level and work
  at a time. During the backward phase, the weight update formula is applied in the reverse dir
  output node and moving backwards to the input layer. All of this is done to find the weights th
  the objective function.

Support Vector Machines

- SVM is a function that finds a dividing line between data in a dataset. The line that maximizes
  two closest data points of two classes with a dividing line in the middle is the best line and the



j,

e input for that weight

is not linearly separable.
he input layer, the hidden
value it predicts. To train
N is consistent with the

s iteration are used to
ing to the output, one level
ection, starting at the
hat minimize the output of

the margin between the
e one the model attempts

to find.
- B1 is better than B2.
- Rows less than the margin are squares, rows greater than the margin are circles, and if anythi between b11 and b12, you can use slack variables to tolerate small errors in the training data.
- If your data's decision boundary is not linear, then you will have to transform the data into a h where the data is linearly separable.
- The kernel trick is what can do this without the curse of dimensionality. This trick basically rep attributes via their N x N pairwise similarity values. The function that computes the similarity, so that it represents a dot product in some high dimensional feature space, but is easily comp

Ensemble Classifiers

- Ensemble Classifiers construct a set of classifiers from the training data, and predict class labe records by aggregating the predictions made by multiple classifiers.
- If we have 25 base classifiers that run on a row, and we pick the majority vote class, AKA the c most classifiers. Another way is consensus, where the classifiers all have to be the same or the We assume independence of the classifiers.
- If we have an actual positive case, and 13 or greater of the classifiers predict a negative, this is
- The probability that ensemble classifiers make a wrong prediction:
- $$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^{i} (1-\varepsilon)^{25-i} = 0.06$$
- The error rate is the epsilon, and this uses an epsilon of 0.35. If the epsilon is greater than 0.5, classifier will perform worse. Also, if you use the same classifier 25 times, it won't do anything 0.35.
- Ensembles work as different classifiers can make up for the negative aspects of other classifie correlation between base classifiers improves the performance of this method.

- Bias-variance decomposition is a formal method for analyzing the prediction error of a predict Variance squared + noise.
    - Bias is the difference between the average prediction and the actual value. Models with do not capture the regularities in the data.
    - Variance measures the spread of the predictions around the average prediction given sr training data. High variance methods overfit training data.
    - Bias and variance are inversely related to each other in general. We want low bias and l

ng exists in the margin

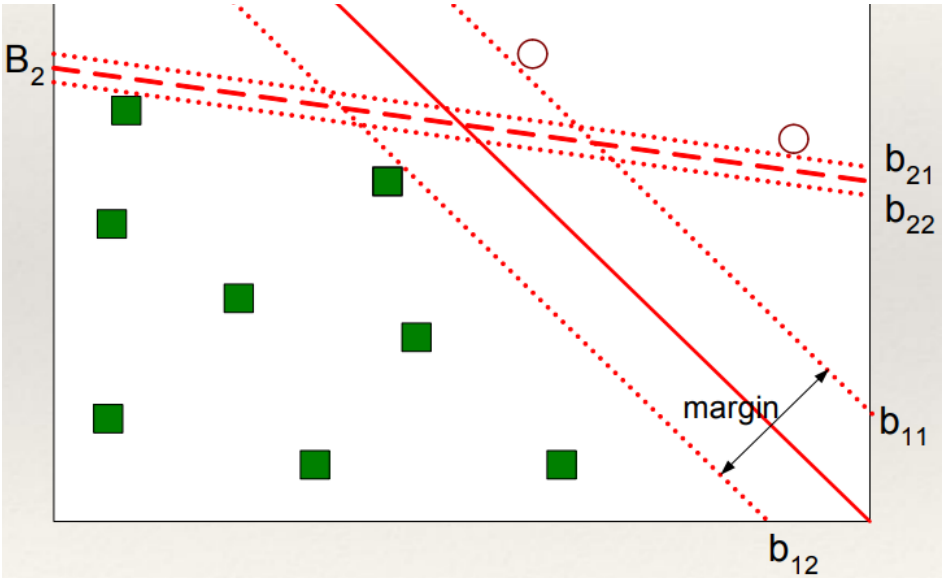
higher dimensional space

presents the input
called the kernel, is chsoen
uted.

ls of previously unseen

class that is returned by the
e class remains unknown.

s an error.



, then the ensemble
, the error rate will remain

rs in the ensemble. Low

tive model. Error = Bias +

a high bias underfit, and

mall perturbations in the

ow variance, with error

being just our noise, things we cannot control.

- Jason uses the term base classifier, which are the classifier algorithms used in the ensemble cl sample from the original training data and use an unstable learner, with the most common be want to pick classifiers with high variance, as combining these will reduce overall variance. Th have low bias.
- Unstable just means we have a high variance learner. This means if we change the training da model will be changed dramatically.
- We can construct an ensemble of classifiers in a couple of different ways:
    - Manipulating the training dataset (bagging)
    - Manipulating the input features (random forest)
    - Manipulating the learning algorithms for the classifiers
    - Manipulating the class labels (Usually done on large class number rows)
- In bagging, you randomly select row attributes with replacement multiple times and build clas follow the standard ensemble instructions
- On the 195 training set, the best binary decision stump can perform at a 30% error rate.
- Boosting is an iterative procedure to adaptively change distribution of training data by focusin misclassified records.
    - Intially, all N records are assigned equal weights, but these may change at the end of a b
    - Records with wrong classifieds increase weights, right classifieds decrease weights
    - Weights are used either to change the sampling distribution in each iteration or to bias
- Adaboost takes a base set of classifiers, and weights the importance of a classifier according t actually is.
- The simple algorithms typically have low variance, with high bias. These are logistic regressio naïve Bayes, and rule based. Complex typically are more flexible, high variance, and low bias w overfit.

Review Lecture

- Appendix material can be on the exam.
- Remember data mining is the non trivial extraction of implicit, previously unknown and poten from data. It is a blend of data analysis methods with sophisticated algorithms for processing
- We divide tasks into prediction and description. Before the midterm, we talked about predicti regression, and deviation detection. After, we'll cover descriptive methods like clustering and
- Proximity measures are really important!
- His highlights from chapters 2 and 3:
    - Attribute type: nominal, ordinal, interval, ratio
    - Know multidimensional density and distributions with the covariance matrix, and the co attributes. Have a good understanding of how to understand what the correlation is in a
    - Know Bayes Theorem. Maybe just write it down.

lassifier. Most ensembles
eing decision trees. So we
e unstable learners should

ta even just a little bit, the

ssifiers on them. Then you

ng more on previously

boosting round.

the underlying model
o how high its error rate

, KNN, decision trees,
which makes them tend to

tially useful information
large volumes of data.
ion methods, classification,
association.

orrelation value between
a graph.

- Proximity measures, again. Know the 1D views, and the measures for multiple dimensio and Minkowski. SMC, Jaccard, Cosine. Jaccard is better for sparse datasets, as is does nc can skew your proximity measure. Cosine is useful for nonbinary data, where attributes
- One thing that could be on the exam is to get a box or scatterplot and interpret someth comfortable with all the visualization tools.
- Data quality: noise/outliers, missing values
- Principle Component Analysis is fair game. Interpretation of data, or do part of the calcu you do PCA, what does it mean, what it is it trying to accomplish.
- Dimensional Reduction creates a new, reduced set of features from the original. Like go more descriptive attributes. Feature subset selection just takes a chunk of the attributes little to no transformations performed on them.

- His highlights from chapters 4 and 5:
  - The purpose of classification is to learn a model from your training data that generalizes data. Overfitting occurs when your model works well on your training set but only on yc generalize. Underfitting occurs when your model doesn't even fit your training set corre give any worthwhile description of your data.
  - You want a good balance point between the overfit and underfit of your model. You wa error.
  - Producing a class label can be broken down into two steps. For a given record, you outp or "score" for each possible class label. So a model takes in a record, and outputs these to predict the class with a varying threshold t between 0 and 1. If P(+|X) >= t, then x = + = 0.5 which is the default decision threshold, you output a confusion matrix. By altering build a ROC curve, and output a confusion matrix for each value of t.
  - By increasing the threshold, you typically decrease TP and FP, and increase FN and TN. U precision goes up when increasing t, you have a good model. This means that your mod assignments, as most TPs that are strongly believed to be positive will lie above high t va
  - Have a good idea of how decision trees, naïve Bayes, rule based and KNN work. Do not w about neural networks, SVM, and ensembles.

Exam Answer Notes

- An equal interval width is when descretizing a continuous variable where you create each inte equal rate, like 10, 20, 30, 40. An equal frequency distribution is when the same amount of co binned into each interval.

ons. He mentions Euclidian
ot count 00 matches which
are not T/F or 0/1.
ing from that. Be

ulation. Understand how

oing from 20 attributes to 4
s and only uses them with

s well to previously unseen
our training set, it doesn't
ectly, it is too general to

nt to minimize your test

out a posterior probability
scores. The second step is
. Otherwise, x = -. When t is
t incrementally, you can

Upon increasing t, if your
el is confident in its class
alues.
worry about deep details

erval of discrete type at an
ontinuous variables get