

# **DSCI 552, Machine Learning for Data Science**

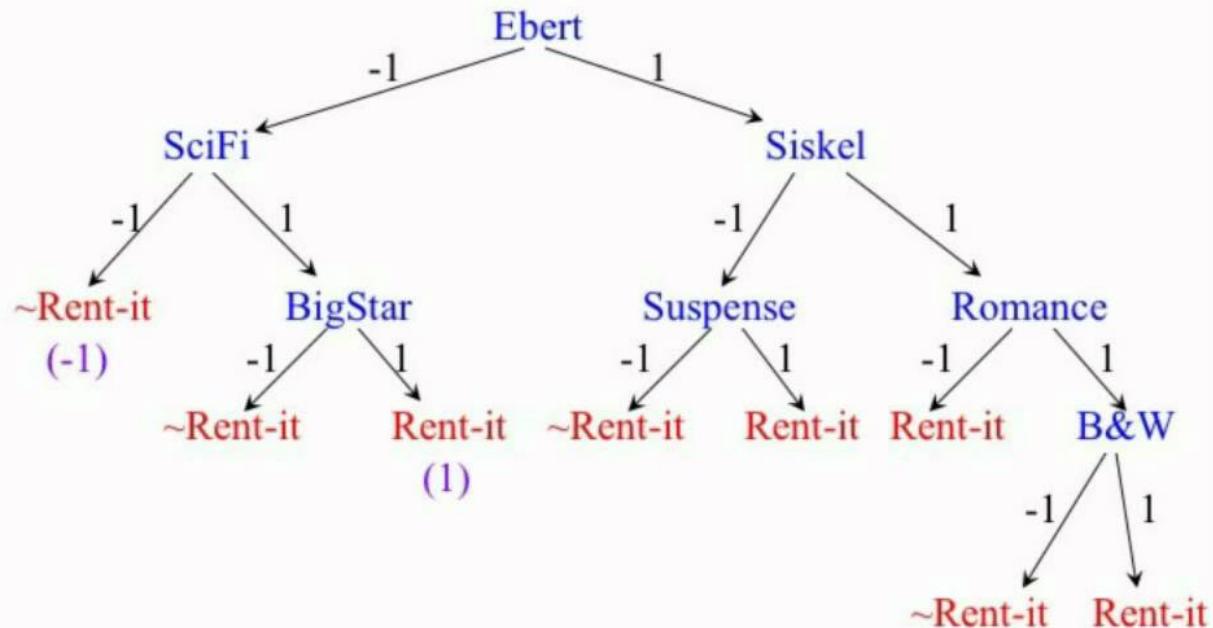
University of Southern California

M. R. Rajati, PhD

# Tree-Based Methods

## Decision tree classifiers

[ SciFi = -1, Suspense = 1, Romance = -1, Ebert = 1, Siskel = 1, ..., Rent-it?? ]



# Tree-based Methods

- Here we describe *tree-based* methods for regression and classification.
- These involve *stratifying* or *segmenting* the predictor space into a number of simple regions.
- Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as *decision-tree* methods.

# Pros and Cons

- Tree-based methods are simple and useful for interpretation.
- However they typically are not competitive with the best supervised learning approaches in terms of prediction accuracy.

# Pros and Cons

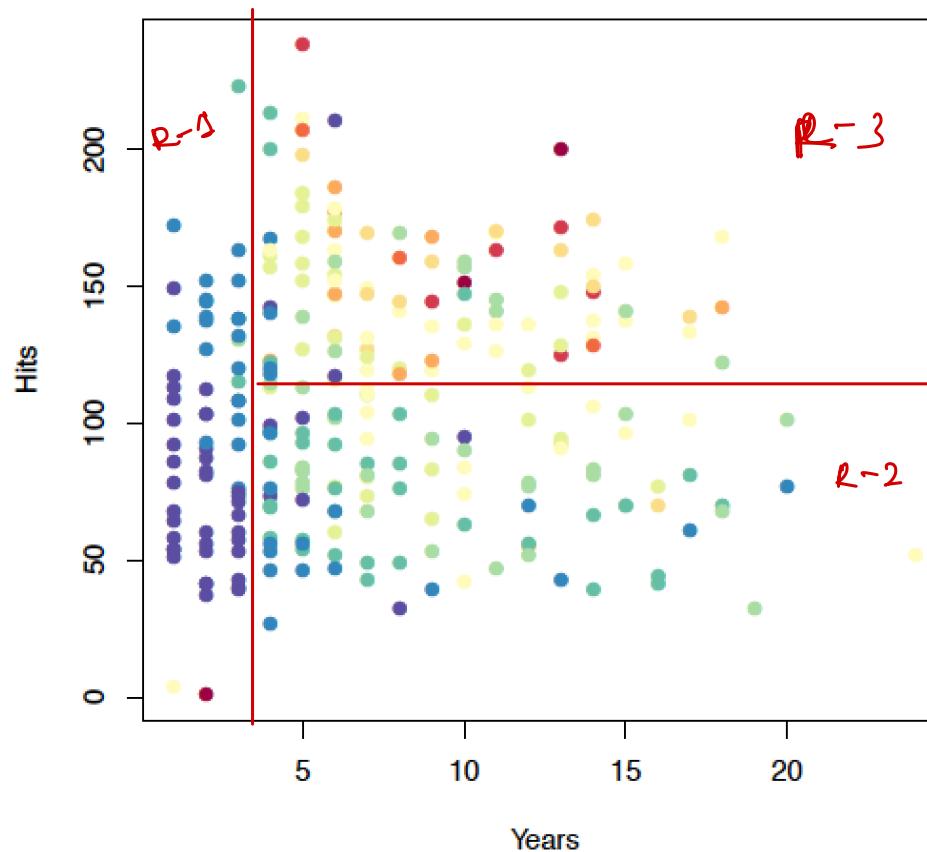
- Hence we also discuss *bagging*, *random forests*, and *boosting*. These methods grow multiple trees which are then combined to yield a single consensus prediction.
- Combining a large number of trees can often result in dramatic improvements in prediction accuracy, at the expense of some loss of interpretation.

# The Basics of Decision Trees

- Decision trees can be applied to both regression and classification problems.
- We first consider regression problems, and then move on to classification.

# Baseball salary data: how would you stratify it?

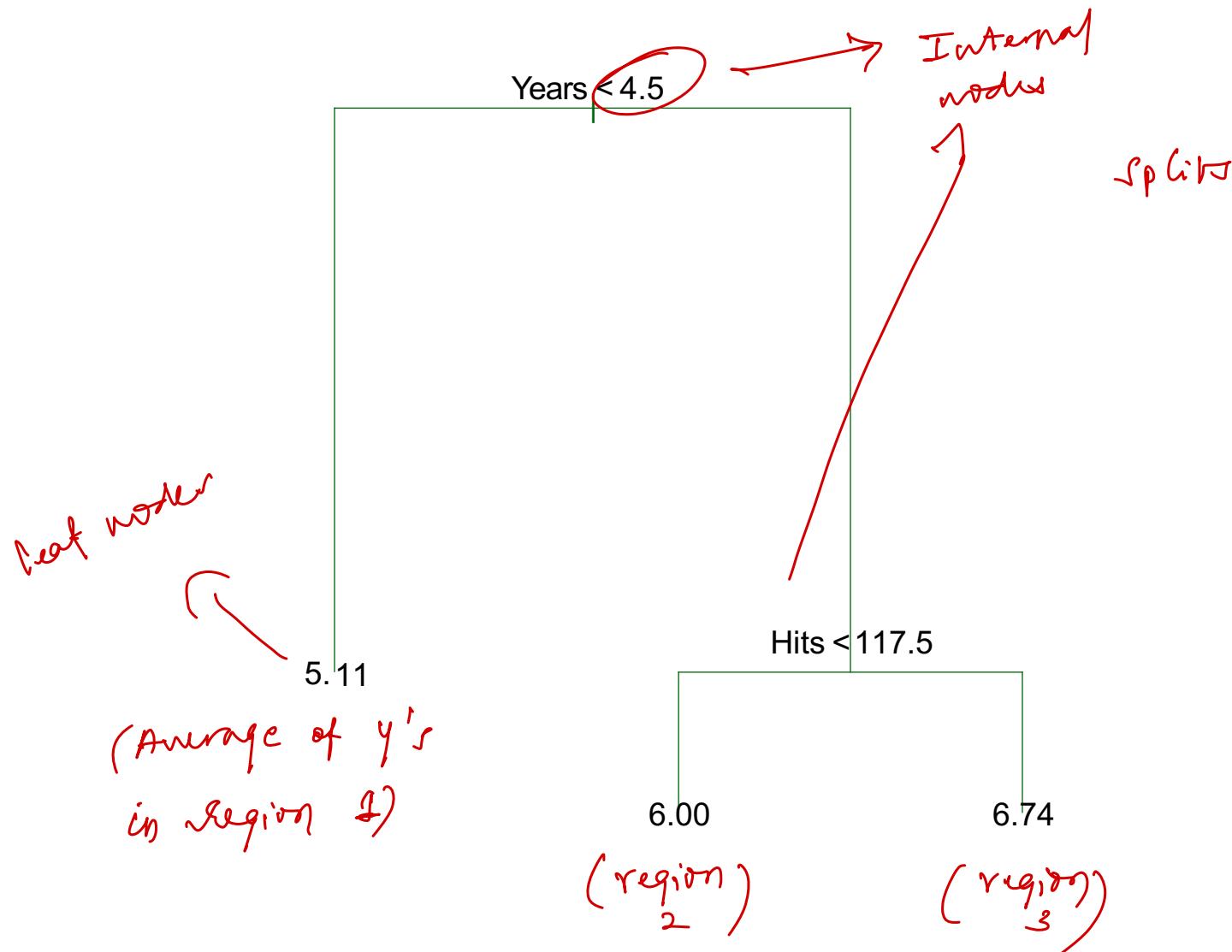
Salary is color-coded from low (blue, green) to high (yellow, red)



$\log_{10}$  Salary

Here **Year** is most important feature.

## Decision tree for these data



# Details of previous figure

- For the Hitters data, a regression tree for predicting the log salary of a baseball player, based on the number of years that he has played in the major leagues and the number of hits that he made in the previous year.

# Details of previous figure

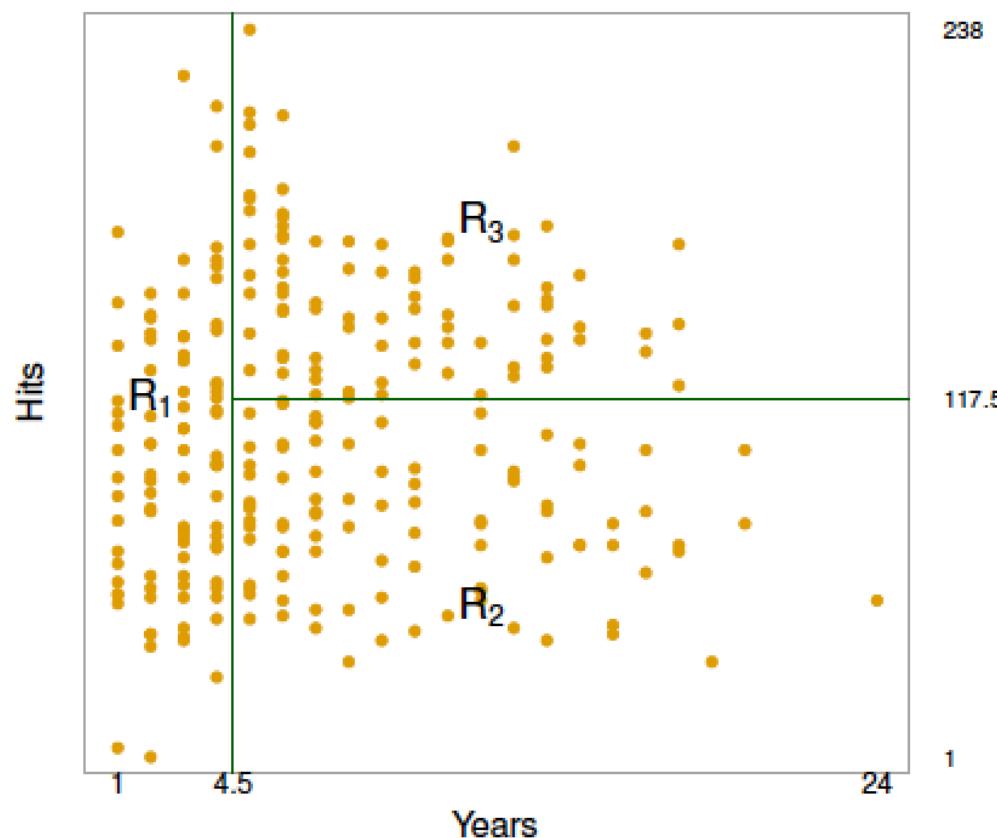
- At a given internal node, the label (of the form  $X_j < t_k$ ) indicates the left-hand branch emanating from that split, and the right-hand branch corresponds to  $X_j \geq t_k$ . For instance, the split at the top of the tree results in two large branches. The left-hand branch corresponds to **Years** < 4.5, and the right-hand branch corresponds to **Years**  $\geq$  4.5.

# Details of previous figure

- The tree has two internal nodes and three terminal nodes, or leaves. The number in each leaf is the mean of the response for the observations that fall there.

# Results

- Overall, the tree stratifies or segments the players into three regions of predictor space:  $R_1 = \{X \mid \text{Years} < 4.5\}$ ,  $R_2 = \{X \mid \text{Years} \geq 4.5, \text{Hits} < 117.5\}$ , and  $R_3 = \{X \mid \text{Years} \geq 4.5, \text{Hits} \geq 117.5\}$ .



# Terminology for Trees

- In keeping with the *tree* analogy, the regions  $R_1$ ,  $R_2$ , and  $R_3$  are known as *terminal nodes*
- Decision trees are typically drawn *upside down*, in the sense that the leaves are at the bottom of the tree.

# Terminology for Trees

- The points along the tree where the predictor space is split are referred to as *internal nodes*
- In the hitters tree, the two internal nodes are indicated by the text Years<4.5 and Hits<117.5.

# Interpretation of Results

- **Years** is the most important factor in determining **Salary**, and players with less experience earn lower salaries than more experienced players.
- Given that a player is less experienced, the number of **Hits** that he made in the previous year seems to play little role in his **Salary**.

# Interpretation of Results

- For players with an experience of five or more years, the number of **Hits** made in the previous year does affect **Salary**, and players who made more **Hits** last year tend to have higher salaries.

# Interpretation of Results

- Surely an **over-simplification**,  
but compared to a regression  
model, it is easy to display,  
interpret and explain

# Details of the tree-building process

*for now, let's fix J.*

1. We divide the predictor space — that is, the set of possible values for  $X_1, X_2, \dots, X_p$  — into  $J$  distinct and non-overlapping regions,  $R_1, R_2, \dots, R_J$ .
2. For every observation that falls into the region  $R_j$ , we make the same prediction, which is simply the mean of the response values for the training observations in  $R_j$ .

$$\hat{y}_{R_j}$$

# More details of the tree-building process

- In theory, the regions could have any shape. However, we choose to divide the predictor space into high dimensional rectangles, or *boxes*, for simplicity and for ease of interpretation of the resulting predictive model.

# More details of the tree-building process

- The goal is to find boxes  $R_1, \dots, R_J$  that minimize the RSS, given by

$$\text{RSS} = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

this is np hard problem.

- where  $\hat{y}_{R_j}$  is the mean response for the training observations within the  $j^{\text{th}}$  box.  
*↑, one only consider few splits.*

# More details of the tree-building process

- Unfortunately, it is computationally infeasible to consider every possible partition of the feature space into  $J$  boxes.
- For this reason, we take a *top-down, greedy* approach that is known as recursive binary splitting.

# More details of the tree-building process

- The approach is *top-down* because it begins at the top of the tree and then successively splits the predictor space; each split is indicated via two new branches further down on the tree.

# More details of the tree-building process

- It is *greedy* because at each step of the tree-building process, the *best* split is made at that particular step, rather than looking ahead and picking a split that will lead to a better tree in some future step.

# Details— Continued

- We first select the predictor  $X_j$  and the cutpoint  $s$  such that splitting the predictor space into the regions  $\{X|X_j < s\}$  and  $\{X|X_j \geq s\}$  leads to the greatest possible reduction in RSS.
- Next, we repeat the process, looking for the best predictor and best cutpoint in order to split the data further so as to minimize the RSS within each of the resulting regions.

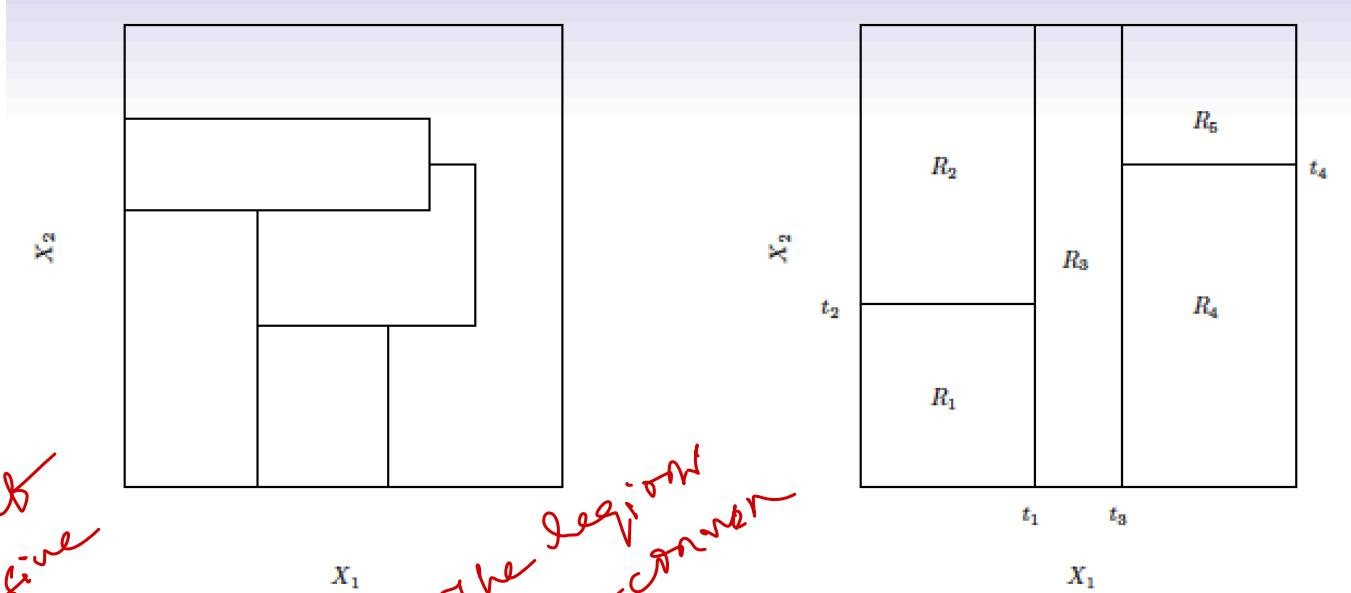
## Details— Continued

- However, this time, instead of splitting the entire predictor space, we split one of the two previously identified regions. We now have three regions.
- Again, we look to split one of these three regions further, so as to minimize the RSS. The process continues until a stopping criterion is reached; for instance, we may continue until no region contains more than five observations.

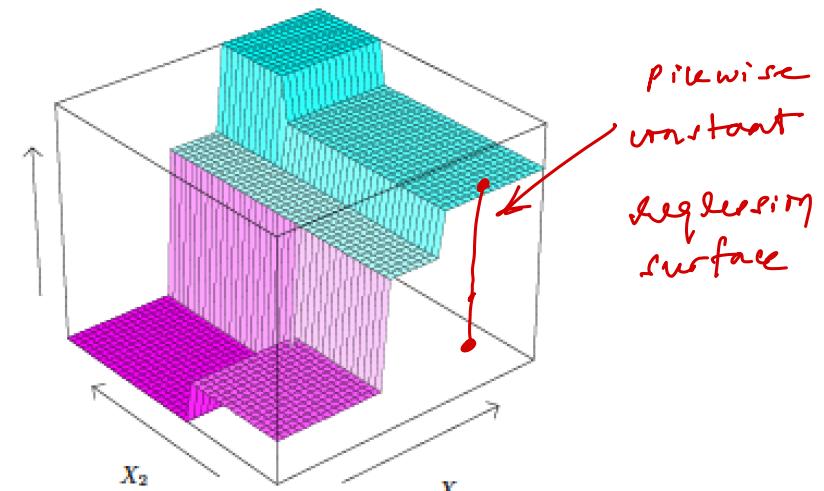
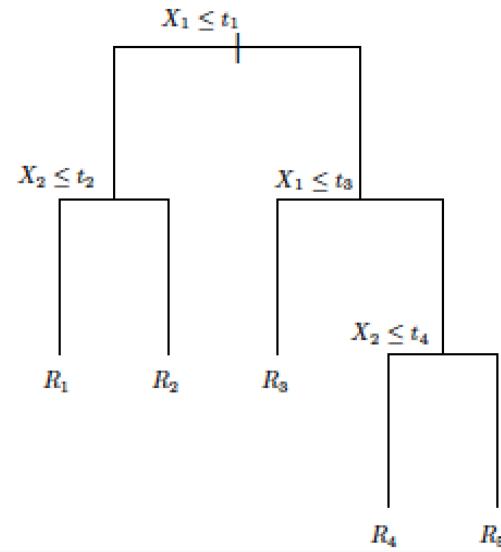
# Predictions

- We predict the response for a given test observation using the mean of the training observations in the region to which that test observation belongs.
- A five-region example of this approach is shown in the next slide.

Can't be  
the result  
of recursive  
binary splitting

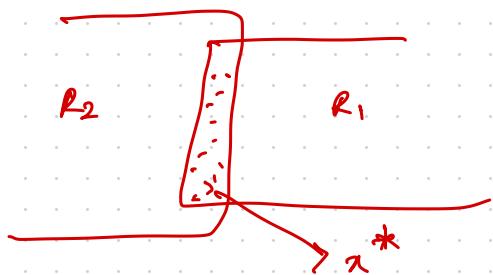


The region  
is non-convex



## Improvements:

- 1) In each region instead of the null model we can use a model such as linear regression or kNN. These models are called "model trees".
- 2) Regression surface isn't continuous.  
We can use regions that allow for degrees of membership & make the prediction of each data point a mixture of predictions of the regions in which it's a member.



$$\mu_{R_1}(x^*)$$

$$\mu_{R_2}(x^*)$$

$$\hat{y}^* = \mu_{R_1}(x^*) \hat{y}_{R_1} + \mu_{R_2}(x^*) \hat{y}_{R_2}$$

Classical sets do not allow for degrees of membership.

So, ~~for~~ to implement this concept we need fuzzy sets, which are classes with unsharp and blurry boundaries.

This will give rise to  
fuzzy decision trees or  
fuzzy systems.

# Details of previous figure

*Top Left:* A partition of two-dimensional feature space that could not result from recursive binary splitting.

*Top Right:* The output of recursive binary splitting on a two-dimensional example.

*Bottom Left:* A tree corresponding to the partition in the top right panel.

*Bottom Right:* A perspective plot of the prediction surface corresponding to that tree.

# Pruning a tree

- The process described above may produce good predictions on the training set, but is likely to *overfit* the data, leading to poor test set performance. *Why?* (*We can make it as a lookup table as well*).
- A smaller tree with fewer splits (that is, fewer regions  $R_1, \dots, R_J$ ) might lead to lower variance and better interpretation at the cost of a little bias.

# Pruning a tree

- One possible alternative to the process described above is to grow the tree only so long as the decrease in the RSS due to each split exceeds some (high) threshold.

# Pruning a tree

- This strategy will result in smaller trees, but is too *short-sighted*: a seemingly worthless split early on in the tree might be followed by a very good split — that is, a split that leads to a large reduction in RSS later on.

# Pruning a tree— continued

- A better strategy is to grow a very large tree  $T_0$ , and then *prune* it back in order to obtain a *subtree*
- *Cost complexity pruning* — also known as *weakest link pruning* — is used to do this

# Pruning a tree— continued

- We consider a sequence of trees indexed by a nonnegative tuning parameter  $\alpha$ . For each value of  $\alpha$  there corresponds a subtree  $T \subset T_0$  such that

A grid of  $\alpha$ 's

$$RSS = \sum_{m=1}^{|T|} \sum_{i: x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

is as small as possible. Here  $|T|$  indicates the number of terminal nodes of the tree  $T$ ,  $R_m$  is the rectangle (i.e. the subset of predictor space) corresponding to the  $m^{\text{th}}$  terminal node, and  $\hat{y}_{R_m}$  is the mean of the training observations in  $R_m$ .

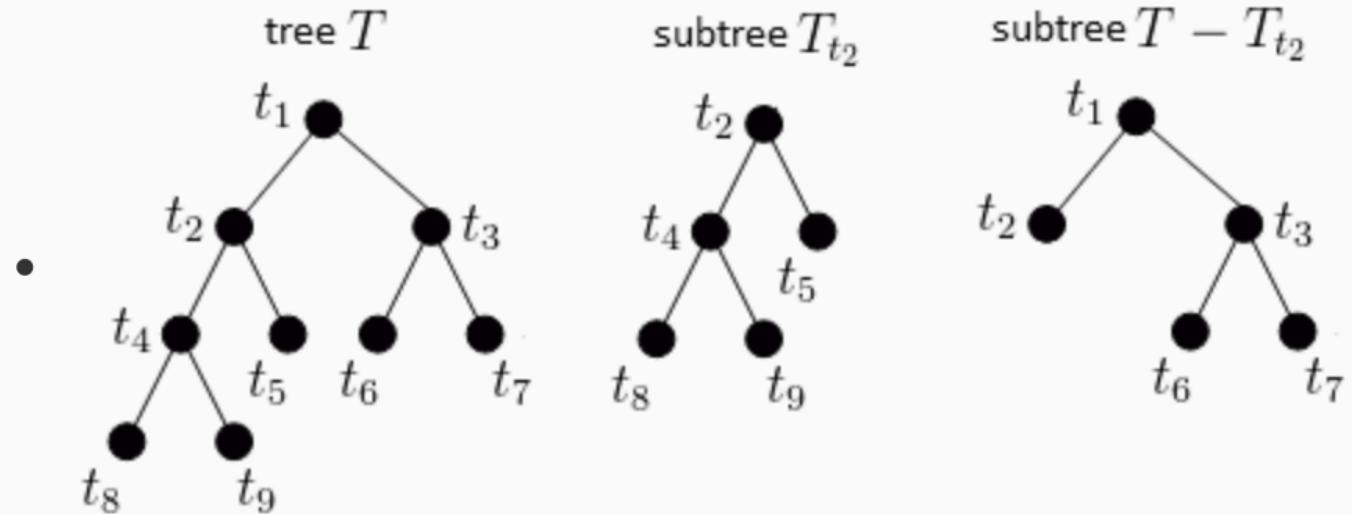
cost + penalty

# Pruning a tree— continued

$$\sum_{m=1}^{|T|} \sum_{i: x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T|$$

## Pruning Subtrees

Subtrees:



# Choosing the best subtree

- The tuning parameter  $\alpha$  controls a trade-off between the subtree's complexity and its fit to the training data.
    - We select an optimal value  $\hat{\alpha}$  using cross-validation.
    - We then return to the full data set and obtain the subtree corresponding to  $\hat{\alpha}$ .
- Variation* ↗ complexity
- ↗ bias

# Summary: tree algorithm

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.
2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of  $\alpha$ .

We don't know  $\alpha$  -

$$\alpha = 1, 5, 9, 100$$

larger the ' $\alpha$ ' size of the subtree will be smaller.

$$\alpha = 1$$

# Summary: tree algorithm

3. Use K-fold cross-validation to choose  $\alpha$ . For each

$$k-1 \rightarrow T_\alpha$$

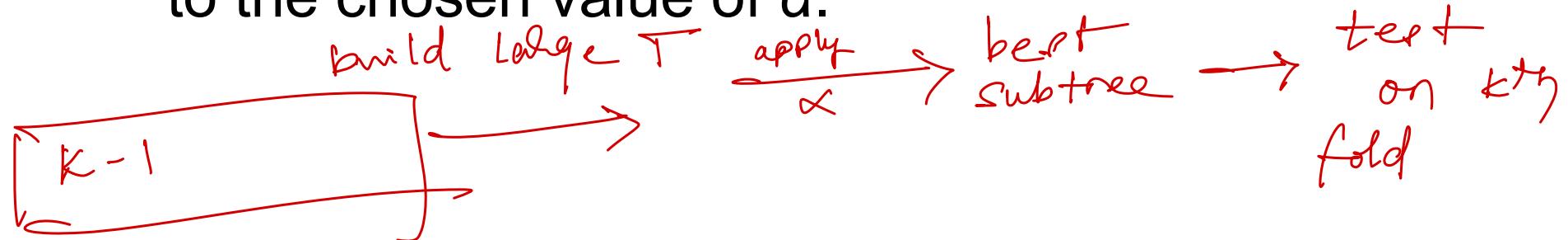
$k = 1, \dots, K$ :

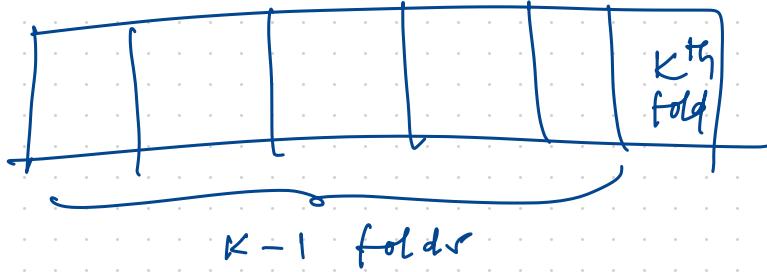
3.1 Repeat Steps 1 and 2 on the  $\frac{K-1}{K}$  th fraction of the training data, excluding the  $k$ th fold.

3.2 Evaluate the mean squared prediction error on the data in the left-out  $k$ th fold, as a function of  $\alpha$ .

Average the results, and pick  $\alpha$  to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of  $\alpha$ .





large tree

|| use  $\alpha$   $\xrightarrow{RS^S + \alpha |T|}$

Best subtree

|| test on  $k^{th}$  fold

Cross validation error  
 $CV(K)$

$$\frac{1}{K} \sum_{k=1}^K CV(k) = CV \text{ for } \alpha$$

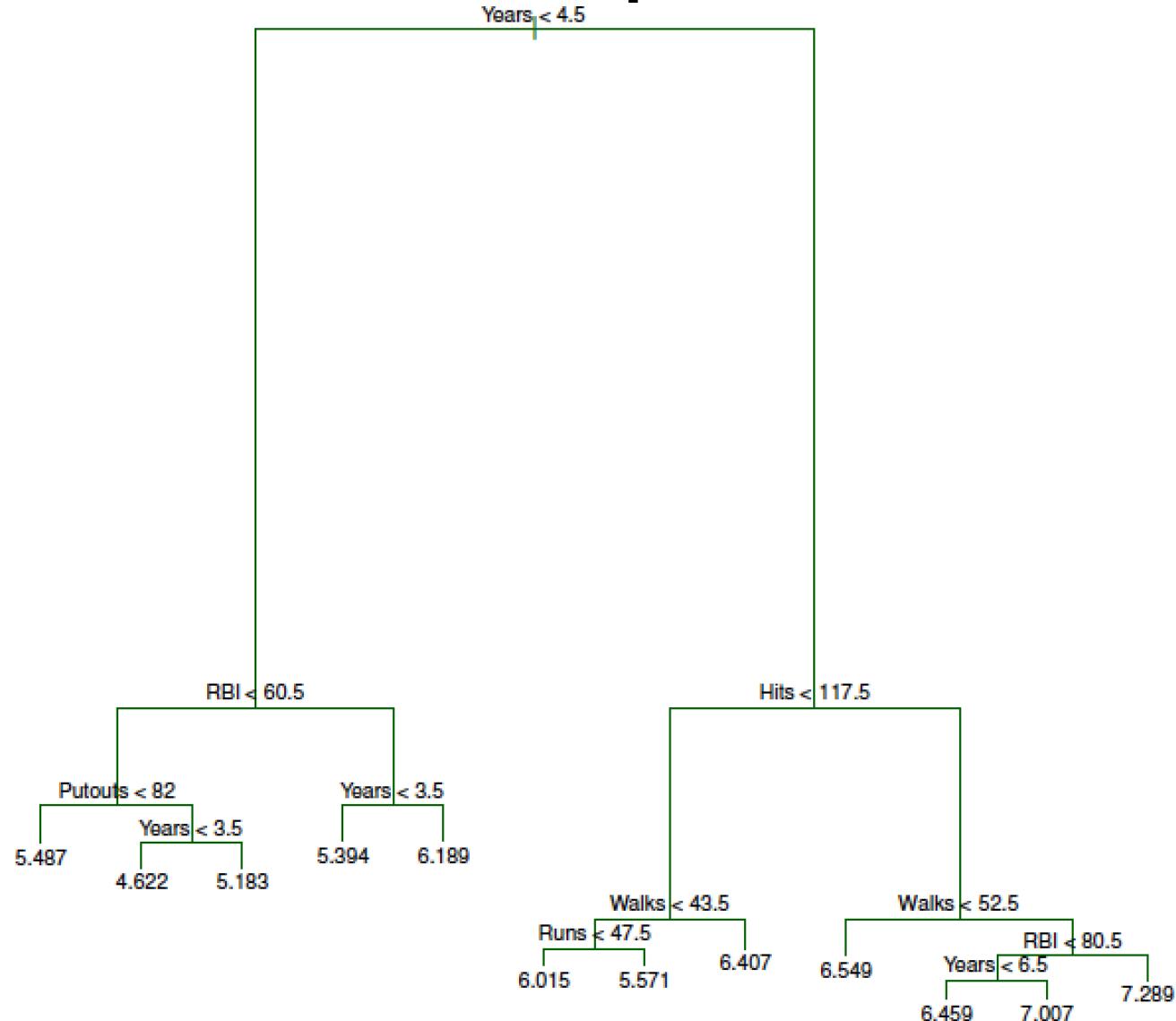
# Baseball example continued

- First, we randomly divided the data set in half, yielding 132 observations in the training set and 131 observations in the test set.

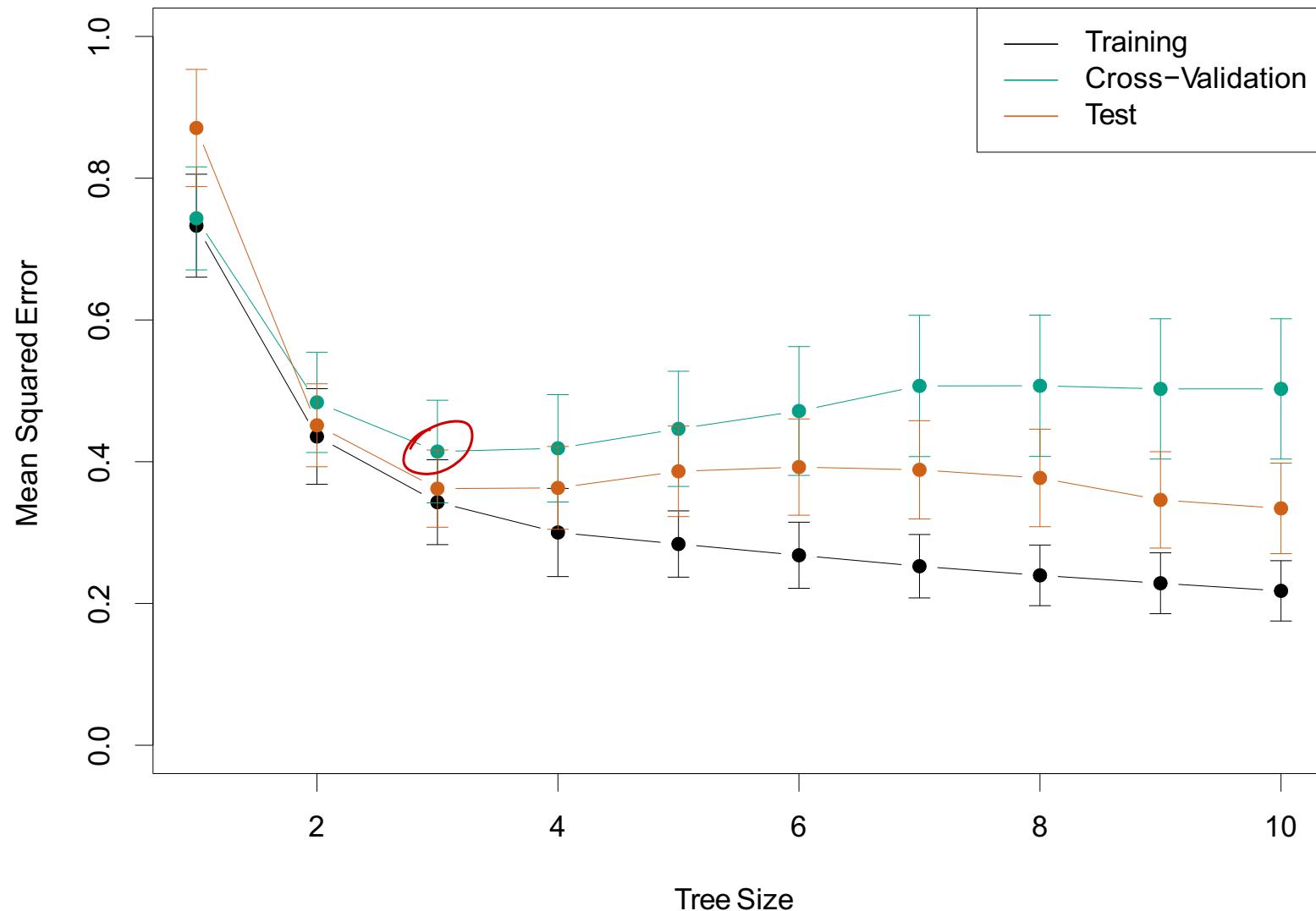
# Baseball example continued

- We then built a large regression tree on the training data and varied  $\alpha$  in order to create subtrees with different numbers of terminal nodes.
- Finally, we performed six-fold cross-validation in order to estimate the cross-validated MSE of the trees as a function of  $\alpha$ .

# Baseball example continued



# Baseball example continued



# Classification Trees

- Very similar to a regression tree, except that it is used to predict a qualitative response rather than a quantitative one. *So we can't, average  
take*
- For a classification tree, we predict that each observation belongs to the *most commonly occurring class* of training observations in the region to which it belongs.

# Details of classification trees

- Just as in the regression setting, we use recursive binary splitting to grow a classification tree.
- In the classification setting, RSS cannot be used as a criterion for making the binary splits

# Details of classification trees

- A natural alternative to RSS is the **classification error rate**, which is simply the fraction of the training observations in that region that do not belong to the most common class:

$$E = 1 - \max(\hat{p}_{mk}).$$

- Here  $\hat{p}_{mk}$  represents the proportion of training observations in the  $m^{\text{th}}$  region that are from the  $k^{\text{th}}$  class.

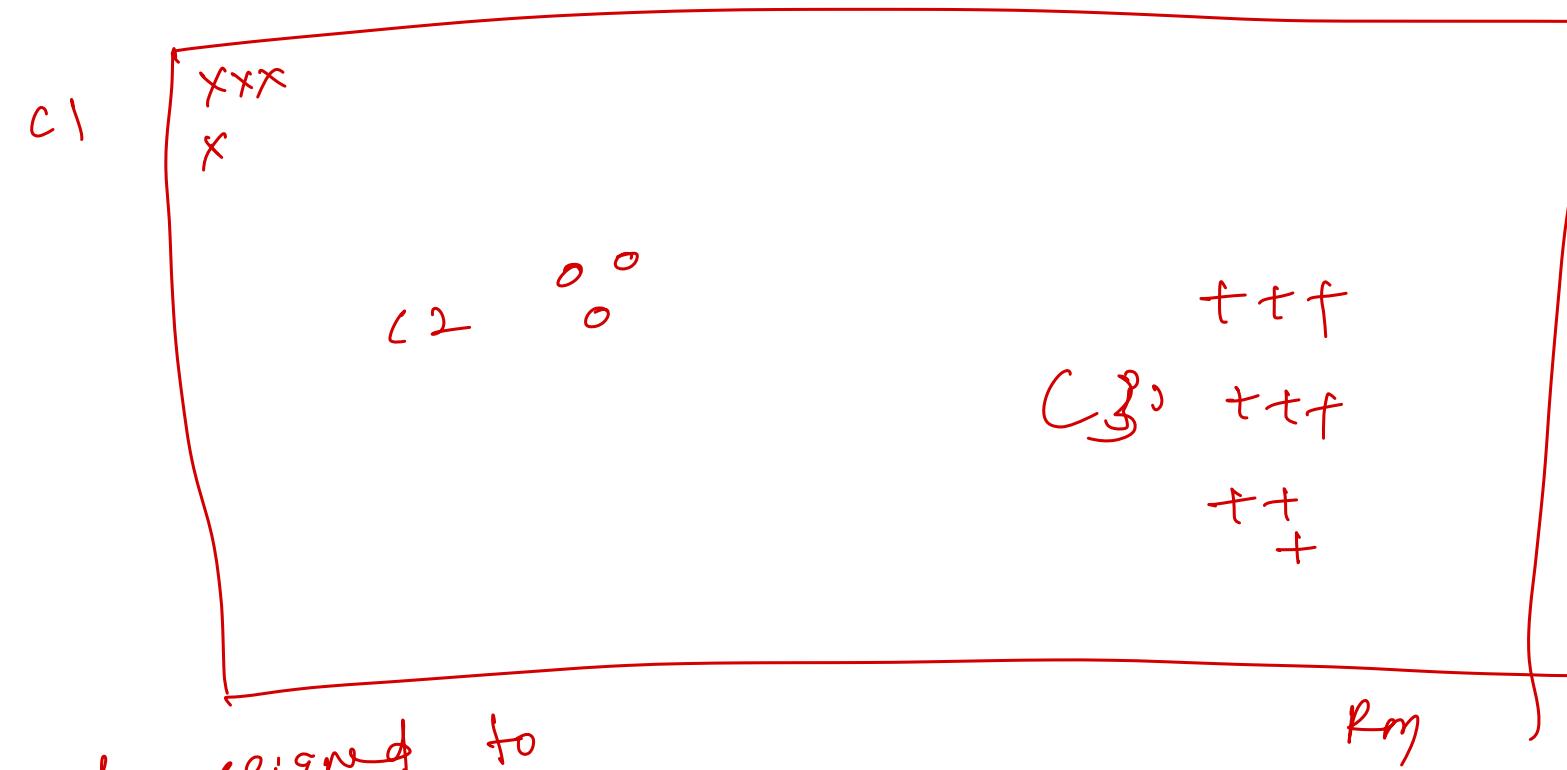
# Example

A simple example:

$$\hat{P}_{m_3} = \frac{9}{9+3+4} = \frac{9}{16}$$

$$\hat{P}_{m_2} = \frac{3}{16}$$

$$\hat{P}_{m_1} = \frac{4}{16}$$



Label assigned to  
this region is: C3

$$\text{Misclassification rate} \rightarrow \frac{4+3}{16} = \frac{7}{16}$$

$$= 1 - \hat{P}_{m_3} = 1 - \frac{9}{16}$$

# Details of classification trees

- However classification error is not sufficiently sensitive for tree-growing, and in practice two other measures are preferable.

$$I \leftarrow \max_k (\hat{P}_{mk})$$

If variance is more then the data will have lot of information.

## Gini index and Deviance

- The *Gini index* is defined by

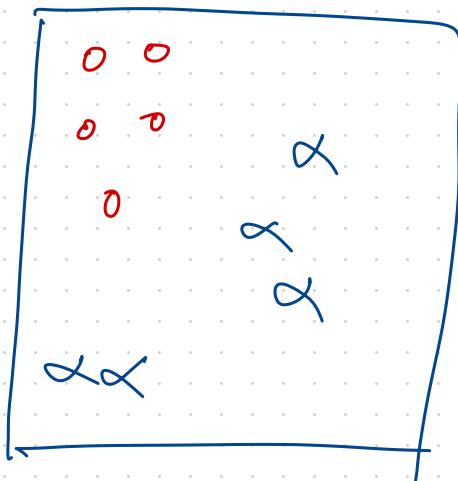
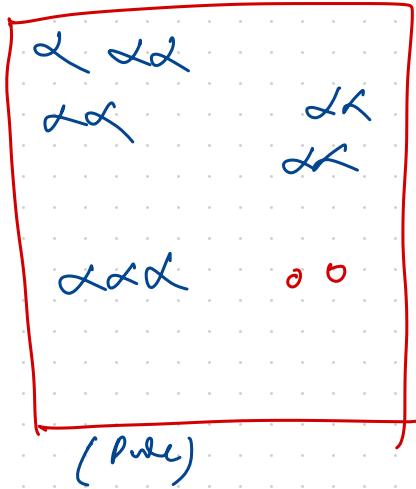
$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk}).$$

a measure of total variance across the  $K$  classes. The Gini index takes on a *small value* if all of the  $\hat{p}_{mk}$ 's are close to zero or one.

Bernoulli  $P$

$$\text{Var}_P = P(1-P)$$

$$\sum \hat{P}_{mk}(1 - \hat{P}_{mk})$$



left region is better

$$\hat{P}_{MK} \approx 1$$

$$\hat{P}_{MK} \approx 0$$

# Gini index and Deviance

$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk}).$$

- For this reason the Gini index is referred to as a measure of node *purity* — a small value indicates that a node contains predominantly observations from a single class.
- The Tree Algorithm in this case is called CART (Classification and Regression Trees)

# Gini index and Deviance

- An alternative to the Gini index is *cross-entropy*, given by

$$D = - \sum_{k=1}^K \hat{p}_{mk} \log \hat{p}_{mk}.$$

$$\Rightarrow - \hat{p}_{mk} \log \hat{p}_{mk} \approx 0$$

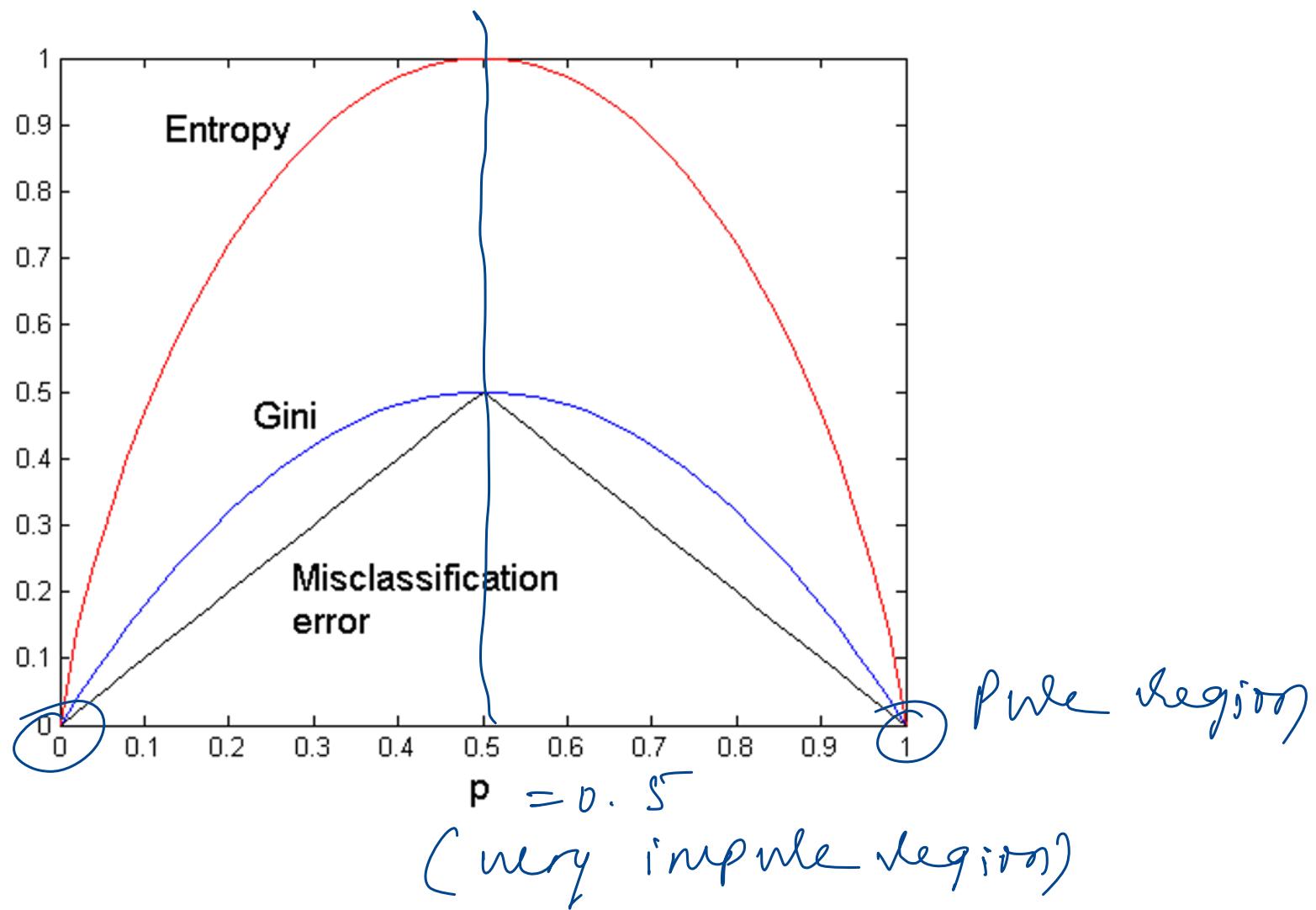
- The tree algorithms that use cross entropy are ID3, C4.5, and C5.0
- It turns out that the Gini index and the cross-entropy are very similar numerically.

$$\lim_{x \rightarrow 0} \frac{x}{0} \underbrace{\log x}_{-\infty} = 0$$

' $x$ ' is farter

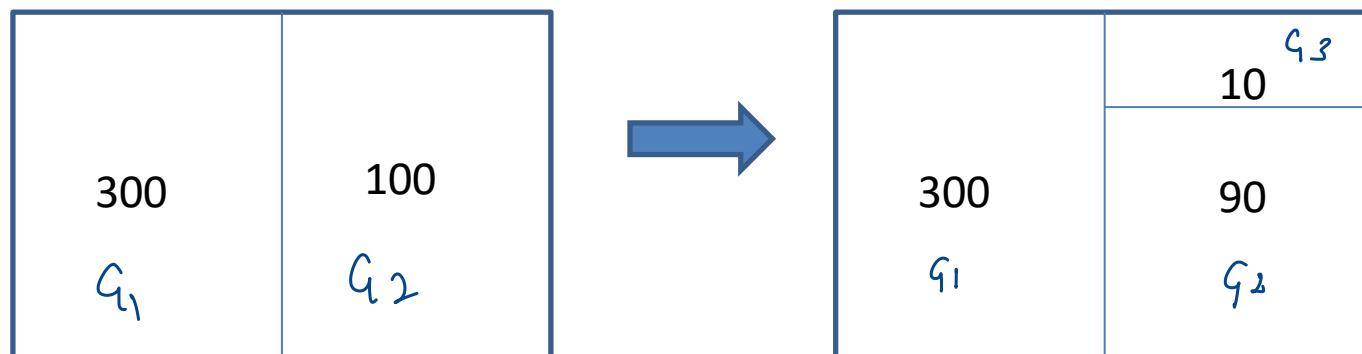
# Comparison of Impurity Measures

- For a two-class problem



# Weighted Impurity

- Most of the time, to be more fair, we add up the *weighted impurity* of the regions resulting from a split.
- We choose weights to be the fraction of data points in each region.



Simple sum :  $g_1 + g_2 \Rightarrow g_1 + g_2 + g_3$

Weight sum

$$\frac{300}{400} g_1 + \frac{100}{400} g_2 \Rightarrow \frac{300}{400} g_1 + \frac{90}{400} g_2 + \frac{10}{400} g_3$$

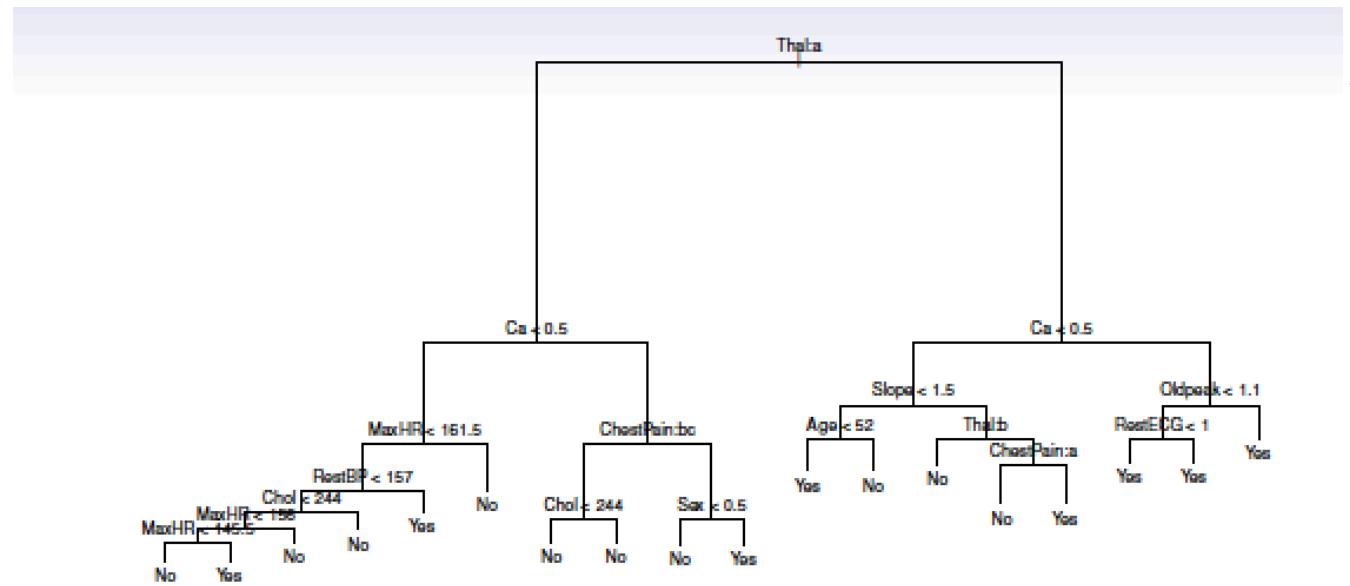
We can do the same for RSS, but  
weighted sum are more common for  
classification

# Example: heart data

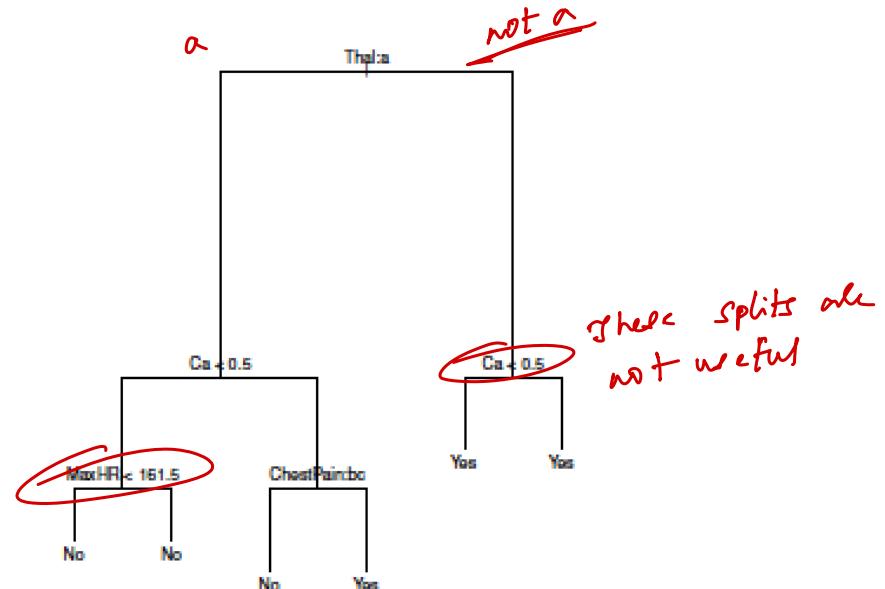
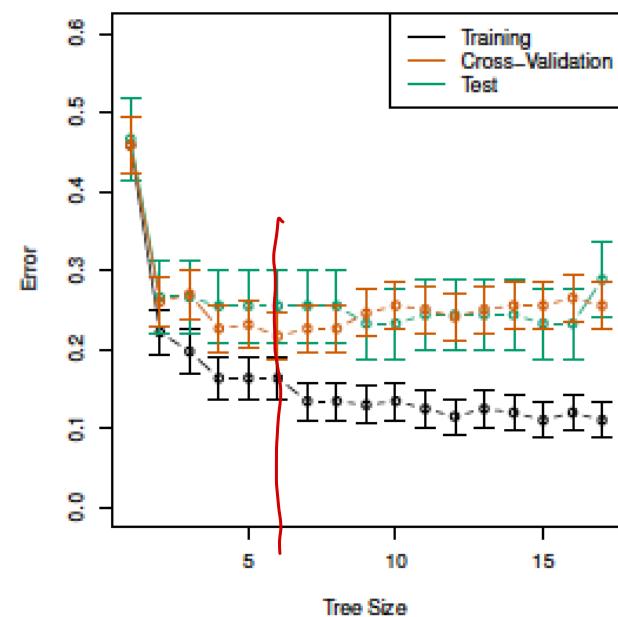
- These data contain a binary outcome **HD** for 303 patients who presented with chest pain.
- An outcome value of **Yes** indicates the presence of heart disease based on an angiographic test, while **No** means no heart disease.

## Example: heart data

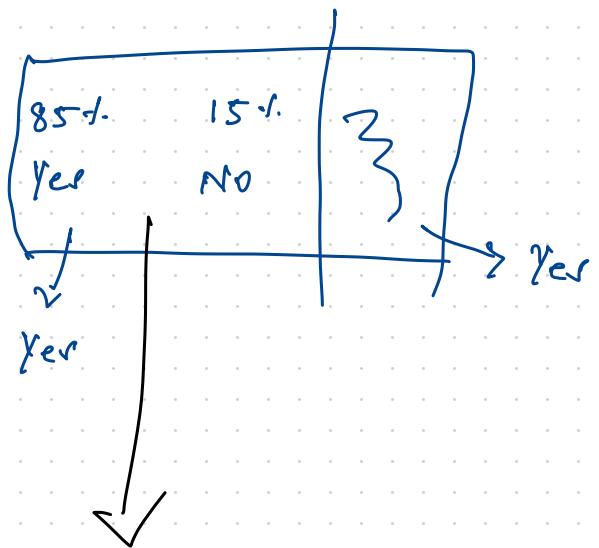
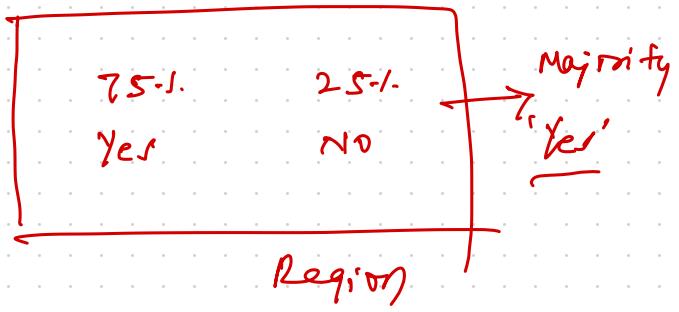
- There are 13 predictors including **Age**, **Sex**, **Chol** (a cholesterol measurement), and other heart and lung function measurements.
- Cross-validation yields a tree with six terminal nodes. See next figure.



T0

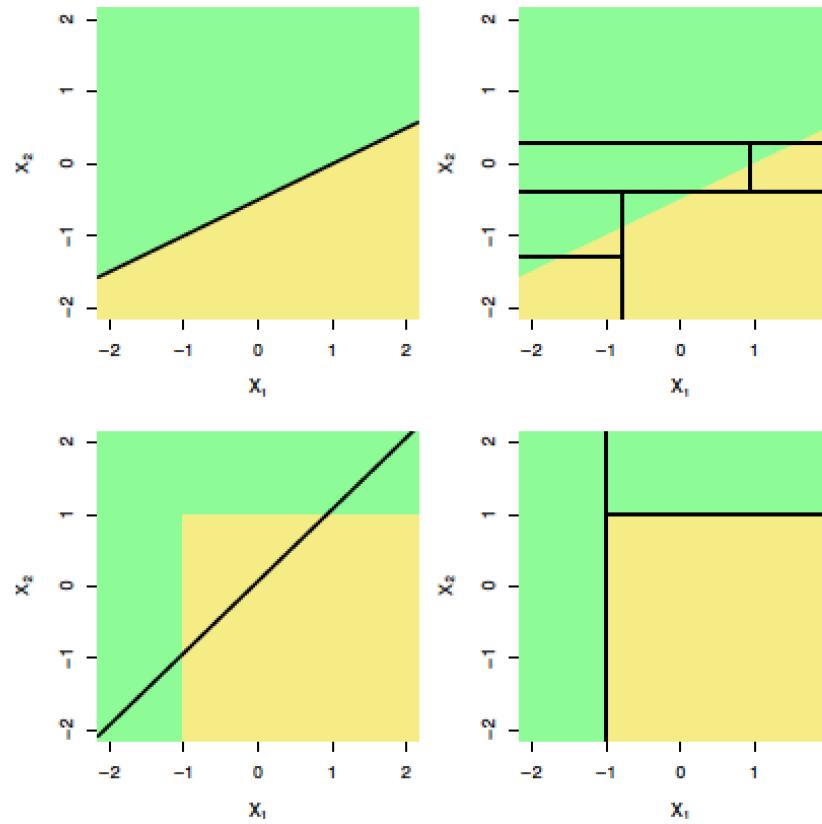


these splits are  
not useful



Although the decision is still yes, in this region, we are more certain that the patient has heart disease.  
(85%)

# Trees Versus Linear Models



Top Row: True linear boundary; Bottom row: true non-linear boundary.

Left column: linear model; Right column: tree-based model

# Advantages and Disadvantages of Trees

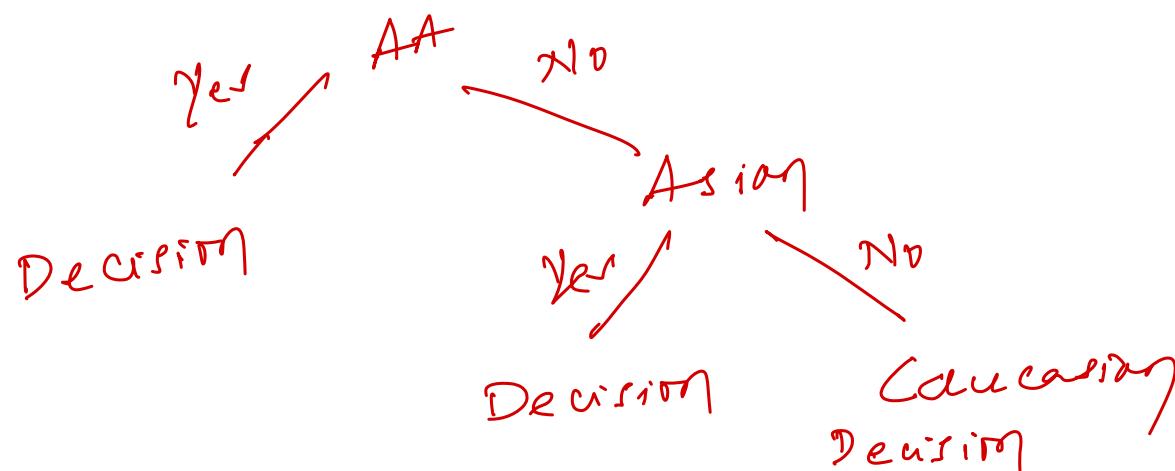
- Trees are very easy to explain to people. In fact, they are sometimes even easier to explain than linear regression!

# Advantages and Disadvantages of Trees

- Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches seen in previous chapters.

# Advantages and Disadvantages of Trees

- Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
- Trees can easily handle qualitative predictors without the need to create dummy variables.



# Advantages and Disadvantages of Trees

- Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches seen in this book.

However, by aggregating many decision trees, the predictive performance of trees can be substantially improved. We introduce these concepts next.

# Bagging

- *Bootstrap aggregation*, or *bagging*, is a general-purpose (wrapper) procedure for reducing the variance of a statistical learning method; we introduce it here because it is particularly useful and frequently used in the context of decision trees.
- It is an *ensemble method*.

# Bagging

- Recall that given a set of  $n$  independent observations  $Z_1, \dots, Z_n$ , each with variance  $\sigma^2$ , the variance of the mean  $\bar{Z}$  of the observations is given by  $\sigma^2/n$ .
- In other words, *averaging a set of observations reduces variance*. Of course, this is not practical because we generally do not have access to multiple training sets.

# Bagging— continued

- Instead, we can bootstrap, by taking repeated samples from the (single) training data set.
- In this approach we generate  $B$  different bootstrapped training data sets. We then train our method on the  $b^{\text{th}}$  bootstrapped training set in order to get  $\hat{f}^{*b}(x)$ , the prediction at a point  $x$ . We then average all the predictions to obtain

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{*b}(x).$$

This is called *bagging*.

# Bagging classification trees

- The above prescription can be applied to regression trees
- For classification trees: for each test observation, we record the class predicted by each of the  $B$  trees, and take a *majority vote*: the overall prediction is the most commonly occurring class among the  $B$  predictions.

# Out-of-Bag Error Estimation

- It turns out that there is a very straightforward way to estimate the test error of a bagged model.

# Out-of-Bag Error Estimation

- Recall that the key to bagging is that trees are repeatedly fit to bootstrapped subsets of the observations.
- One can show that on average, each bagged tree makes use of around two-thirds of the observations.

Bagging

# Out-of-Bag Error Estimation

- The remaining one-third of the observations not used to fit a given bagged tree are called the *out-of-bag* (OOB) observations.

original  
trainin^  
~ data set

(d<sub>1</sub>)

(d<sub>2</sub>)

⋮

(d<sub>n</sub>)

find trees  
that don't  
use it for  
training



Average  $\Rightarrow$  some kind of  
LOOCV

# Out-of-Bag Error Estimation

- We can predict the response for the  $i^{\text{th}}$  observation using each of the trees in which that observation was OOB. This will yield around  $B/3$  predictions for the  $i^{\text{th}}$  observation, which we average.
- This estimate is essentially the LOO cross-validation error for bagging, if  $B$  is large.

# Random Forests

- *Random forests* provide an improvement over bagged trees by way of a small tweak that *decorrelates* the trees. This reduces the variance when we average the trees.
- As in bagging, we build a number of decision trees on bootstrapped training samples.

# Random Forests

- But when building these decision trees, each time a split in a tree is considered, *a random selection of  $m$  predictors* is chosen as split candidates from the full set of  $p$  predictors. The split is allowed to use only one of those  $m$  predictors.

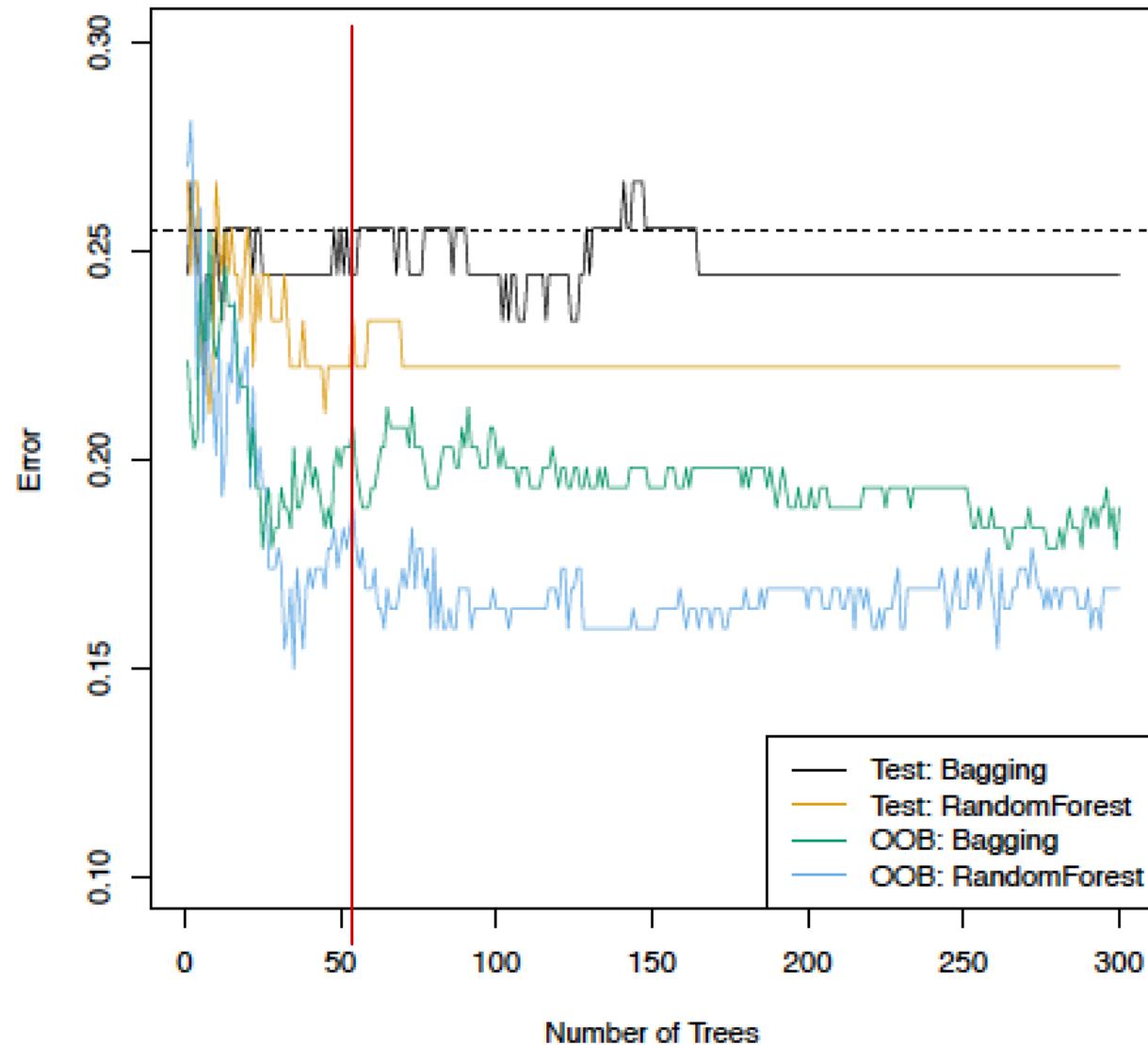
# Random Forests

- A fresh selection of  $m$  predictors is taken at each split, and typically we choose  $m \approx p^{1/2}$  — that is, the number of predictors considered at each split is approximately equal to the square root of the total number of predictors (4 out of the 13 for the Heart data).

# Random Forests: An Alternative Version

- Alternatively, one can build  $B$  decision trees, each of which on a random subset of  $m$  features.
- Instead of a bootstrap sample from the training set, one can even use a subsample with replacement from the training set, which is equivalent to a bootstrap sample with a size different from the training set.

# Bagging the heart data



After certain point  
the tree test error  
rate becomes flat  
as we are not  
adding complexity  
the trees are  
more or less  
learning same.

# Details of previous figure

Bagging and random forest results for the Heart data.

- The test error (black and orange) is shown as a function of  $B$ , the number of bootstrapped training sets used.
- Random forests were applied with  $m = p^{1/2}$ .
- The dashed line indicates the test error resulting from a single classification tree.
- The green and blue traces show the OOB error, which in this case is considerably lower

# Example: gene expression data

- We applied random forests to a high-dimensional biological data set consisting of expression measurements of 4,718 genes measured on tissue samples from 349 patients.

# Example: gene expression data

- There are around 20,000 genes in humans, and individual genes have different levels of activity, or expression, in particular cells, tissues, and biological conditions.

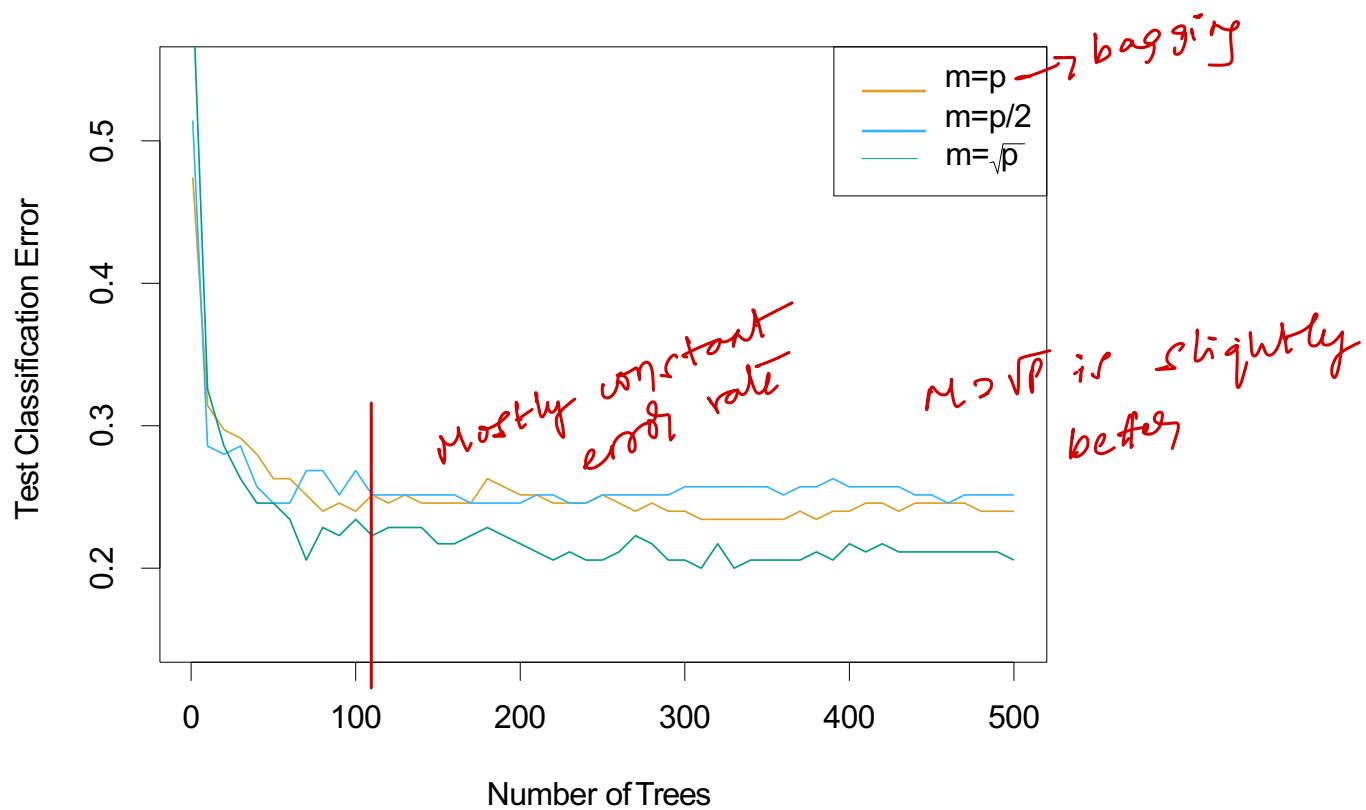
# Example: gene expression data

- Each of the patient samples has a qualitative label with 15 different levels: either normal or one of 14 different types of cancer.
- We use random forests to predict cancer type based on the 500 genes that have the largest variance in the training set.

# Example: gene expression data

- We randomly divided the observations into a training and a test set, and applied random forests to the training set for three different values of the number of splitting variables  $m$ .

# Results: gene expression data



# Details of previous figure

- Results from random forests for the fifteen-class gene expression data set with  $p = 500$  predictors.
- The test error is displayed as a function of the number of trees. Each colored line corresponds to a different value of  $m$ , the number of predictors available for splitting at each interior tree node.

## Details of previous figure

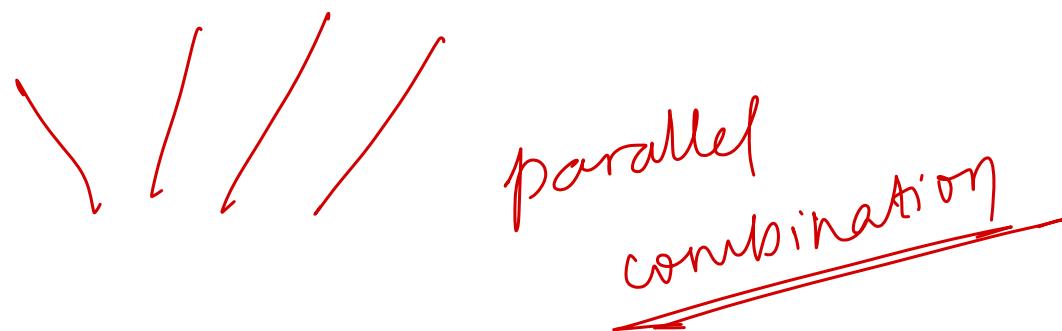
- Random forests ( $m < p$ ) lead to a slight improvement over bagging ( $m = p$ ). A single classification tree has an error rate of 45.7%.

# Boosting

- Like bagging, boosting is a general approach that can be applied to many statistical learning methods for regression or classification. We only discuss boosting for decision trees.

# Boosting

- Recall that bagging involves creating multiple copies of the original training data set using the bootstrap, fitting a separate decision tree to each copy, and then combining all of the trees in order to create a single predictive model.



# Boosting

- Notably, each tree is built on a bootstrap data set, independent of the other trees.
- Boosting works in a similar way, except that the trees are grown *sequentially*: each tree is grown using information from previously grown trees.

# Boosting algorithm for regression trees

1. Set  $\hat{f}(x) = 0$  and  $r_i = y_i$  for all  $i$  in the training set.

2. For  $b = 1, 2, \dots, B$ , repeat:

1. Fit a tree  $\hat{f}^b$  with  $d$  splits ( $d + 1$  terminal nodes) to the training data  $(X, r)$ .

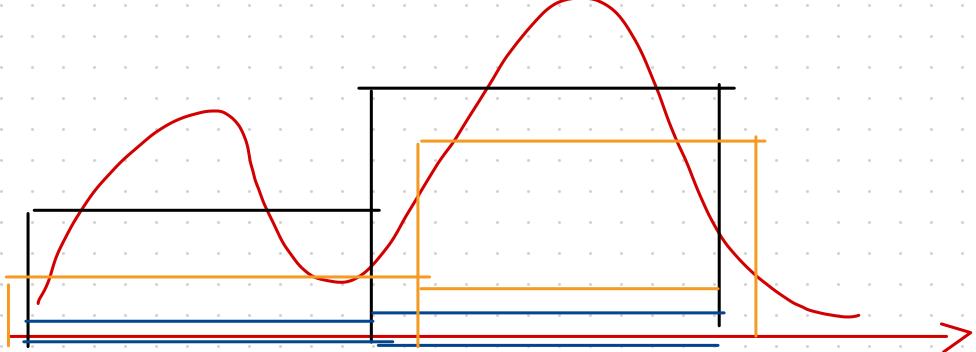
2. Update  $f$  by adding in a shrunken version of the new tree:

$$f(x) \leftarrow f(x) + \lambda \hat{f}^b(x).$$

3. Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$$

$$y_i \sim \lambda \hat{f}'(x_i)$$



# Boosting algorithm for regression trees

Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^B \lambda \hat{f}^b(x).$$

# What is the idea behind this procedure?

- Unlike fitting a single large decision tree to the data, which amounts to *fitting the data hard* and potentially overfitting, the boosting approach instead *learns slowly*.

# What is the idea behind this procedure?

- Given the current model, we fit a decision tree to the residuals from the model. We then add this new decision tree into the fitted function in order to update the residuals.

# What is the idea behind this procedure?

- Each of these trees can be rather small, with just a few terminal nodes, determined by the parameter  $d$  in the algorithm.

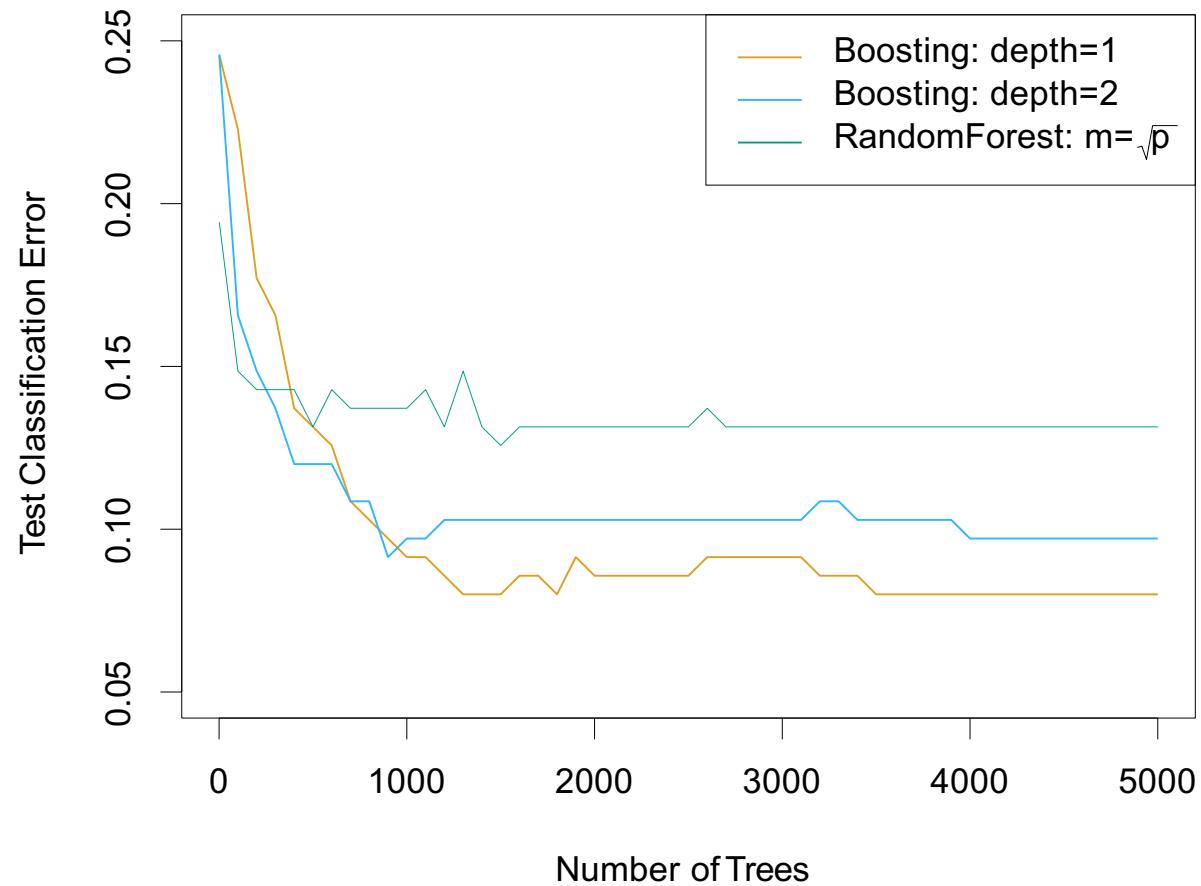
# What is the idea behind this procedure?

- By fitting small trees to the residuals, we slowly improve  $f$  in areas where it does not perform well. The shrinkage parameter  $\lambda$  slows the process down even further, allowing more and different shaped trees to attack the residuals.

# Boosting for classification

- Boosting for classification is similar in spirit to boosting for regression, but is a bit more complex. We will not go into detail here (see appendix), nor does the text book.
- More details in *Elements of Statistical Learning, chapter 10 or Appendix.*
- The R package `gbm` (gradient boosted models) handles a variety of regression and classification problems.

# Gene expression data continued



# Details of previous figure

- Results from performing boosting and random forests on the fifteen-class gene expression data set in order to predict *cancer* versus *normal*.

# Details of previous figure

- The test error is displayed as a function of the number of trees. For the two boosted models,  $\lambda = 0.01$ . Depth-1 trees slightly outperform depth-2 trees, and both outperform the random forest, although the standard errors are around 0.02, making none of these differences significant.
- The test error rate for a single tree is 24%.

# Tuning parameters for boosting

1. The *number of trees*  $B$ . Unlike bagging and random forests, boosting can overfit if  $B$  is too large, although this overfitting tends to occur slowly if at all. We use cross-validation to select  $B$ .

# Early stopping

(Based on validation set)



# Tuning parameters for boosting

2. The *shrinkage parameter*  $\lambda$ , a small positive number. This controls the rate at which boosting learns. Typical values are 0.01 or 0.001, and the right choice can depend on the problem. Very small  $\lambda$  can require using a very large value of  $B$  in order to achieve good performance.

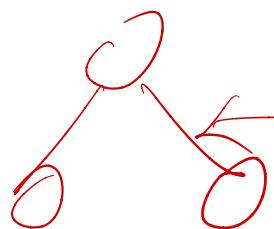
$\lambda \downarrow B \uparrow$

cross validate only on  $B$ .

# Tuning parameters for boosting

3. The *number of splits*  $d$  in each tree, which controls the complexity of the boosted ensemble. Often  $d = 1$  works well, in which case each tree is a *stump*, consisting of a single split and resulting in an additive model. More generally  $d$  is the *interaction depth*, and controls the interaction order of the boosted model, since  $d$  splits can involve at most  $d$  variables.

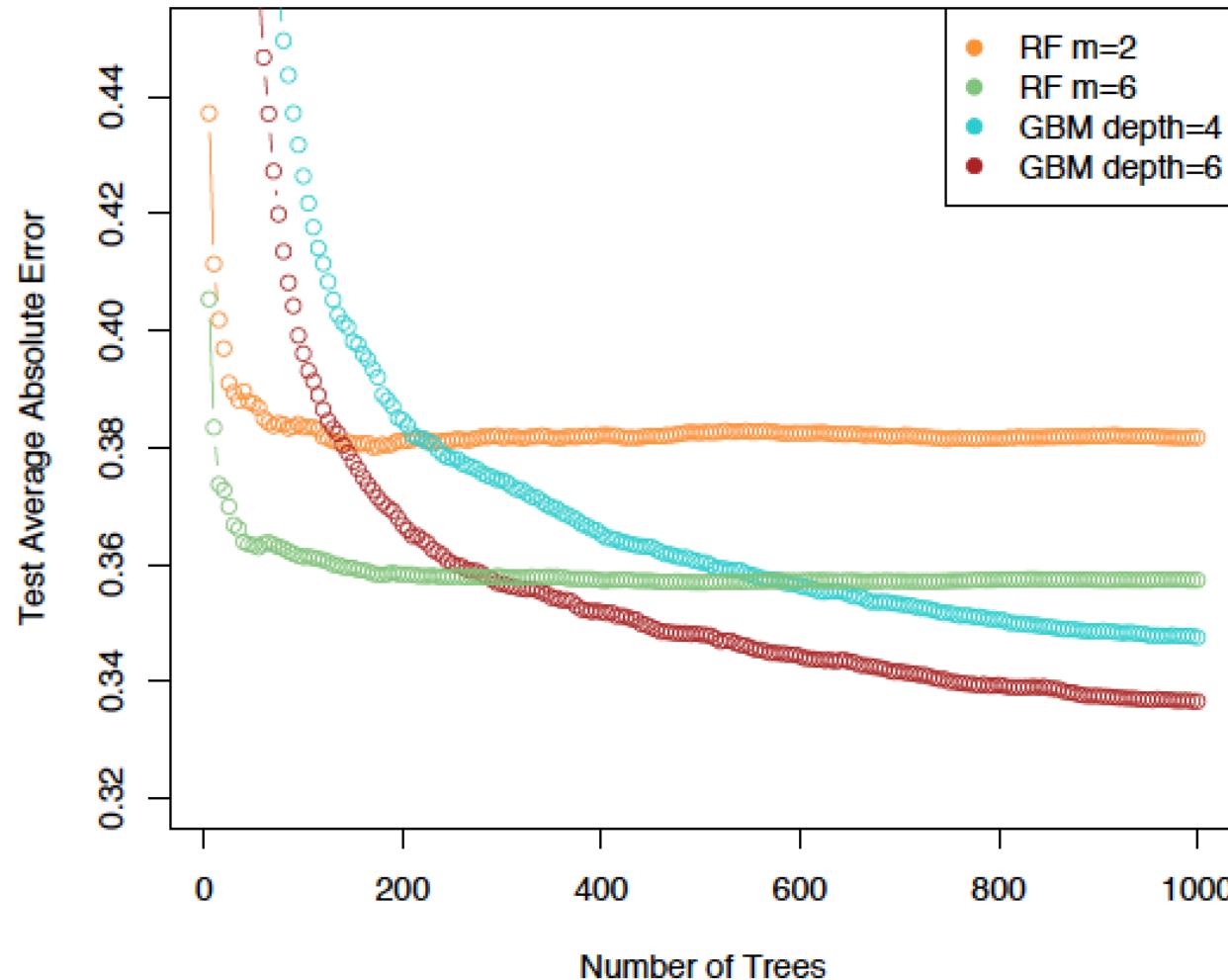
$D=1$   
 $d$  variable



Why?

# Another regression example

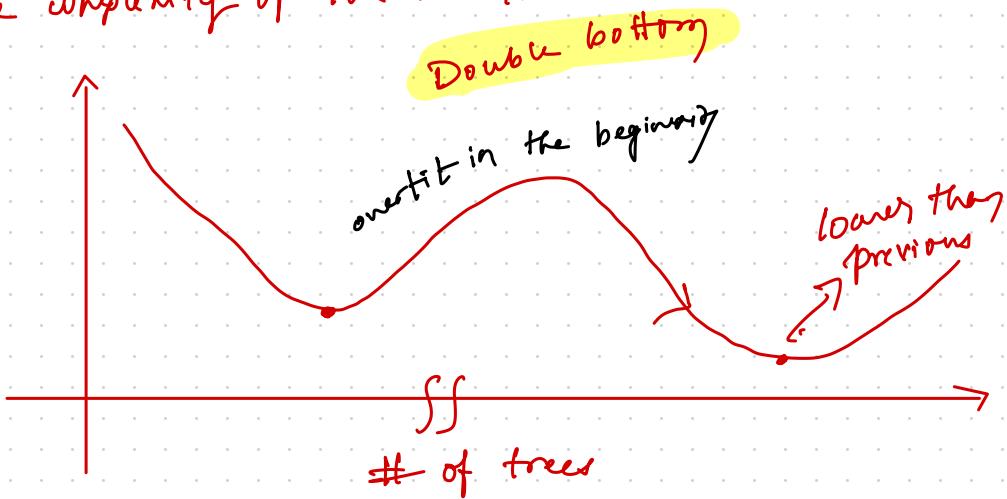
California Housing Data



Super-weird phenomenon is Boosting  
and neural networks.

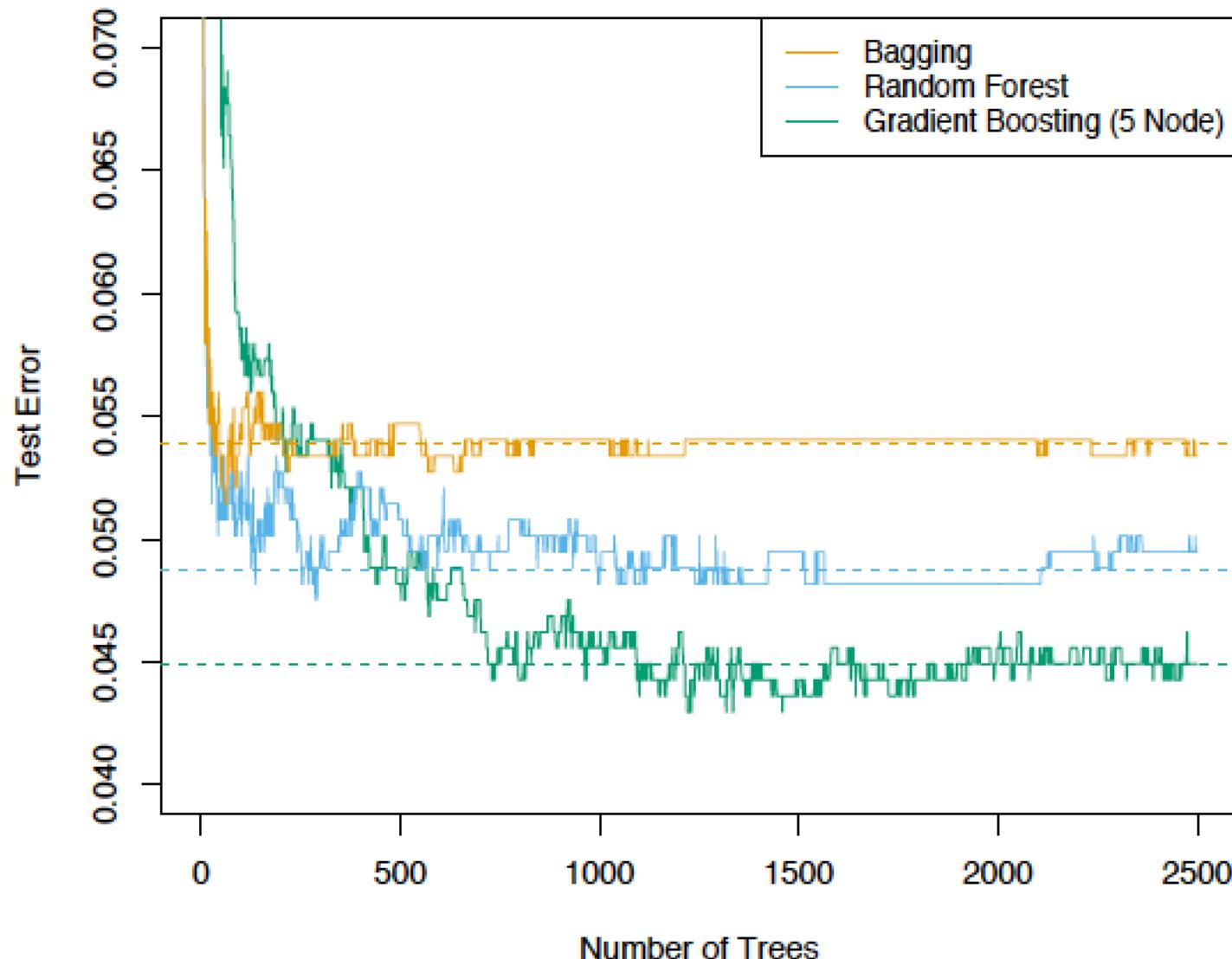


# of parameters/trees aren't always equivalent to the complexity of the model.



# Another classification example

Spam Data



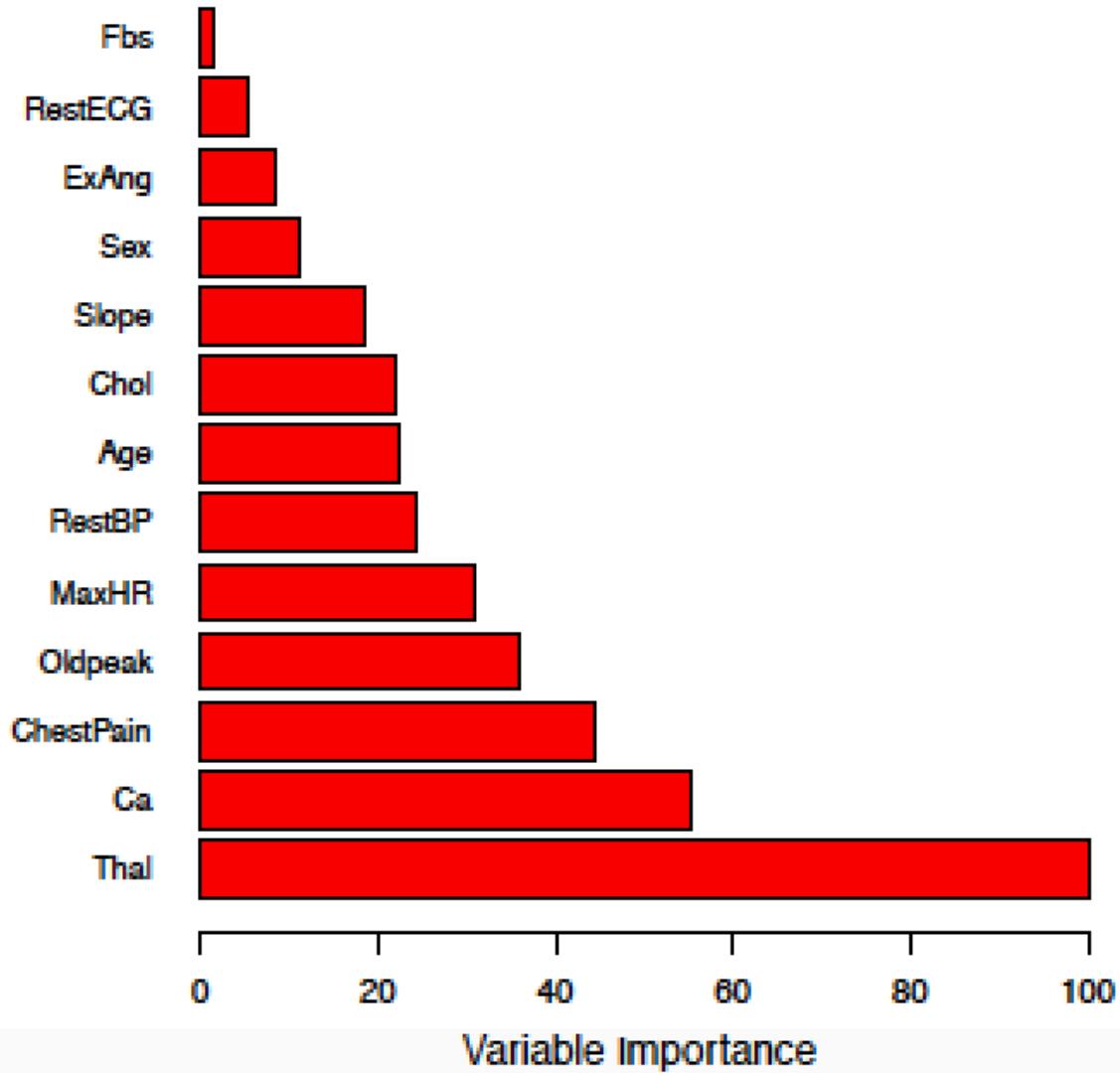
# Variable importance measure

- For bagged/RF regression trees, we record the total amount that the RSS is decreased due to splits over a given predictor, averaged over all  $B$  trees. A large value indicates an important predictor.

# Variable importance measure

- Similarly, for bagged/RF classification trees, we add up the total amount that the Gini index is decreased by splits over a given predictor, averaged over all  $B$  trees.

# Variable importance measure



Variable  
importance  
plot for the  
Heart data

$$\frac{\text{Total Drop}(x_i)}{\text{Max Drop}(x_i)}$$

# Summary

- Decision trees are simple and interpretable models for regression and classification
- However they are often not competitive with other methods in terms of prediction accuracy

# Summary

- Bagging, random forests and boosting are good methods for improving the prediction accuracy of trees. They work by growing many trees on the training data and then combining the predictions of the resulting ensemble of trees.
- The latter two methods—random forests and boosting—are among the state-of-the-art methods for supervised learning. However their results can be difficult to interpret.

# Appendix

## Extra Trees

*Extra tree*

## ~~Boosting (Appendix)~~

- Extremely Randomized Trees were proposed by Geurts et al. in 2006.
- The idea is to make the trees even weaker, but to compensate that with the large number of estimators in the ensemble.

# Boosting (Appendix)

More specifically, not only the features and samples shown to the tree are randomized, but also the growing of individual trees is random.

# Boosting (Appendix)

- In particular the split point i.e., the thresholds of comparisons is randomized. This makes training each tree faster.
- Implemented as `sklearn.ensemble.ExtraTreesClassifier`.

# Boosting (Appendix)

## Boosting for Classification

Boosting iteratively learns weak classifiers; a weak classifier is one whose error rate is only slightly better than random guessing.

# Boosting (Appendix)

## Boosting for Classification

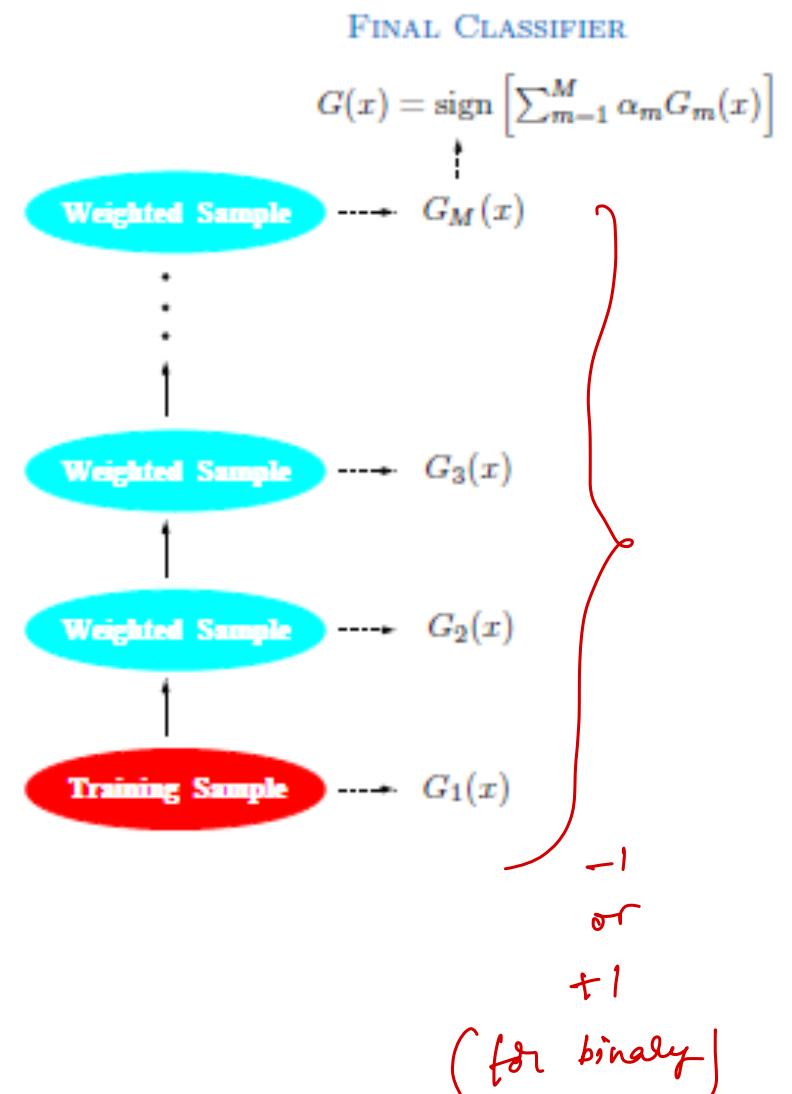
The purpose of boosting is to sequentially apply the weak classification algorithm to repeatedly modified versions of the data, thereby producing a sequence of weak classifiers.

The predictions from all of them are then combined through a weighted majority vote to produce the final prediction.

# Boosting (cont.)

## Boosting for Classification (cont'd)

Thus, the final result is the weighted sum of the results of weak classifiers.

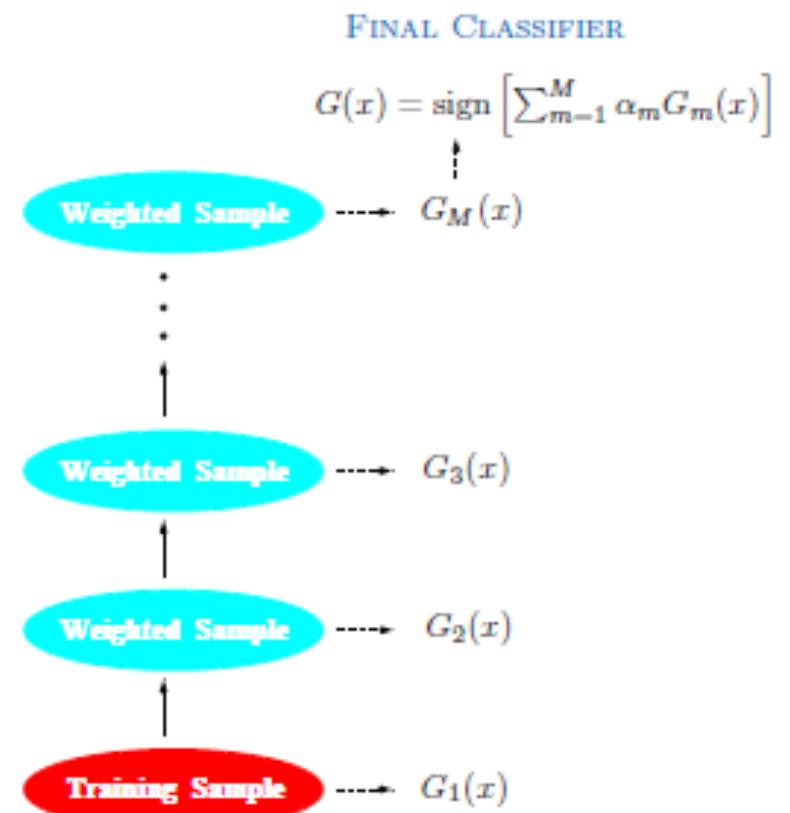


# Boosting (cont.)

## Boosting for Classification (cont'd)

The alphas in the final classifier are computed by the boosting algorithm, which weight the contribution to give higher influence to more accurate classifiers in the sequence.

Weights are modified at each boosting step!



# Boosting (cont.)

**Up-weight** data that are difficult; incorrectly classified in the previous round. **Down-weight** data that are easy; correctly classified in the previous round.

*(Not always)*

## Boosting (cont.)

There are many different boosting algorithms for classification:

AdaBoost, LPBoost, BrownBoost,  
LogitBoost, Gradient Boosting, etc.

# Boosting (cont.)

## Example of Boosting Algorithm (AdaBoost):

---

**Algorithm 10.1 AdaBoost.M1.**

---

1. Initialize the observation weights  $w_i = 1/N$ ,  $i = 1, 2, \dots, N$ .
  2. For  $m = 1$  to  $M$ :  
    (a) Fit a classifier  $G_m(x)$  to the training data using weights  $w_i$ .  
    (b) Compute  
$$\text{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}$$
  
$$\alpha_m = \log\left(\frac{1}{\text{err}_m} - 1\right)$$
  
    (c) Compute  $\alpha_m = \log((1 - \text{err}_m)/\text{err}_m)$ .  
    (d) Set  $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))]$ ,  $i = 1, 2, \dots, N$ .
  3. Output  $G(x) = \text{sign} \left[ \sum_{m=1}^M \alpha_m G_m(x) \right]$ .
- 

$\alpha_m$  = weight of the classifier

If the data point isn't misclassified  
 $w_i$  will not change

$$\therefore I(y_i \neq g_m(x_i)) = 0$$

But, if the data point is misclassified

$$w_i \leftarrow w_i e^{\alpha y}$$
$$= w_i \left( \frac{1}{\text{err}_m} - 1 \right)$$

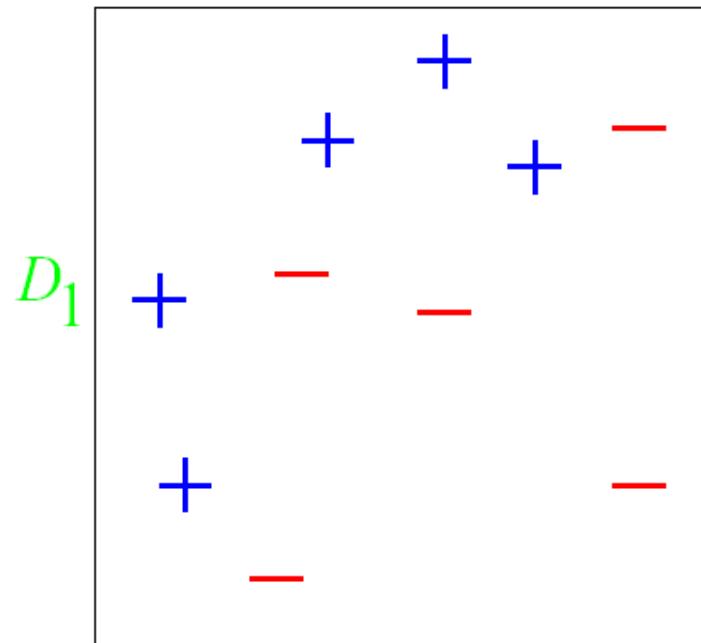
# Boosting Intuition

- We adaptively weigh each data case.
- Data cases which are wrongly classified get high weight (the algorithm will focus on them)
- Each boosting round learns a new (simple) classifier on the weighed dataset.

# Boosting Intuition

- These classifiers are weighed to combine them into a single powerful classifier.
- Classifiers that obtain low training error rate have high weight.
- We stop by using monitoring a hold out set (cross-validation). *Validation set*

# An Illustrative Example



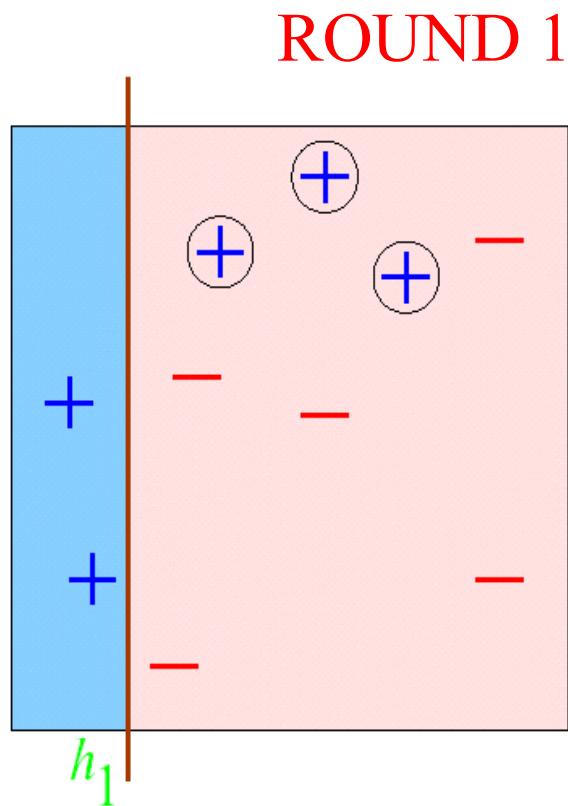
Original training set: equal weights to all training samples

Taken from “**A Tutorial on Boosting**” by Yoav Freund and Rob Schapire

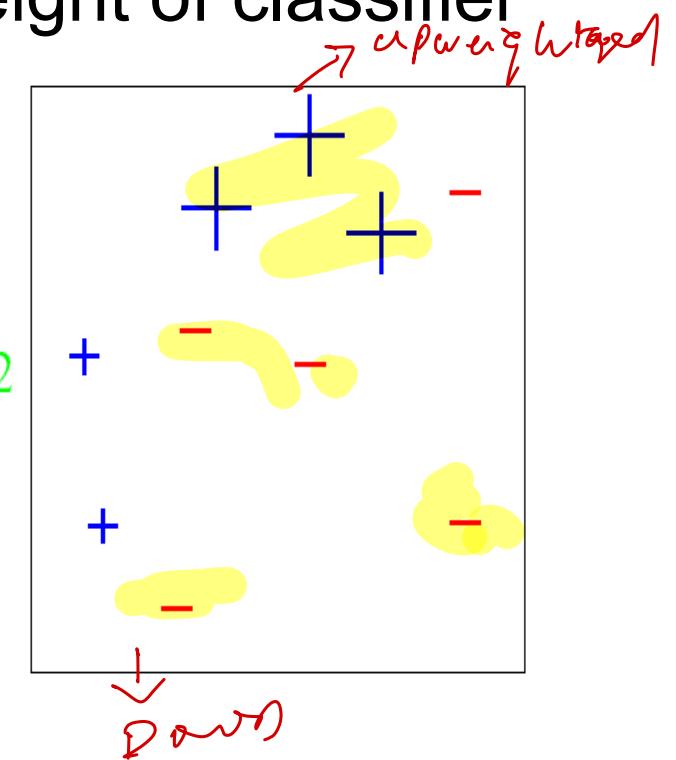
# AdaBoost example

$\varepsilon$  = error rate of classifier

$\alpha$  = weight of classifier

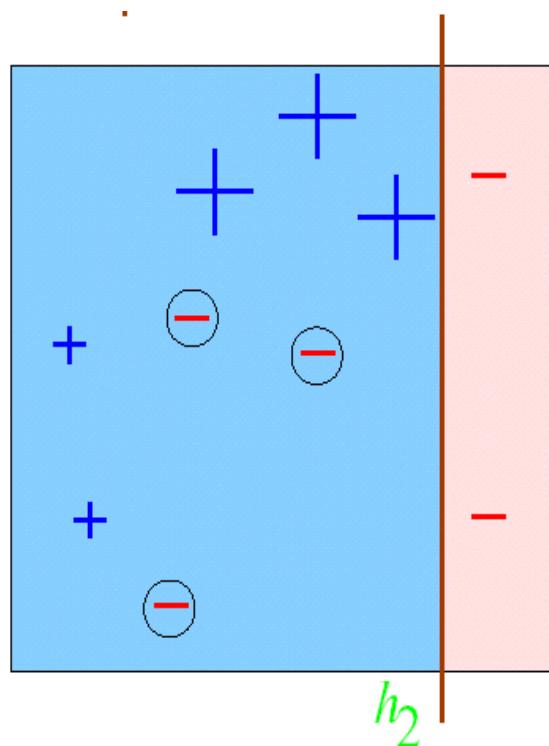


$$\begin{aligned}\varepsilon_1 &= 0.30 \\ \alpha_1 &= 0.42\end{aligned}$$

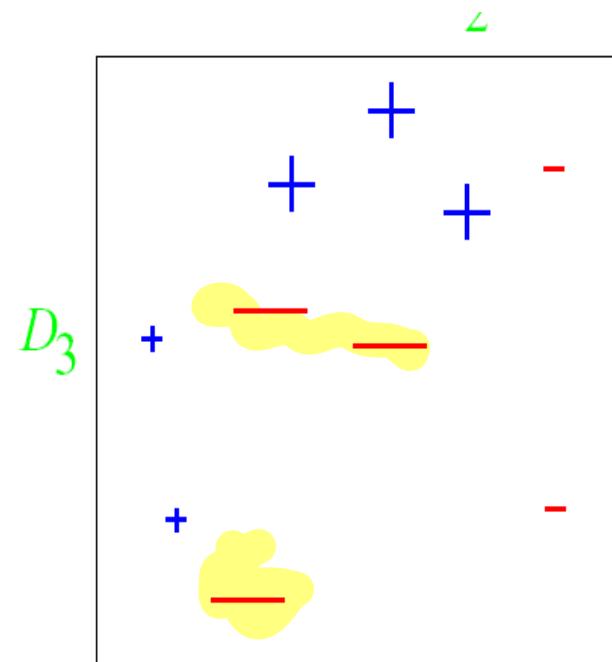


# AdaBoost example

ROUND 2

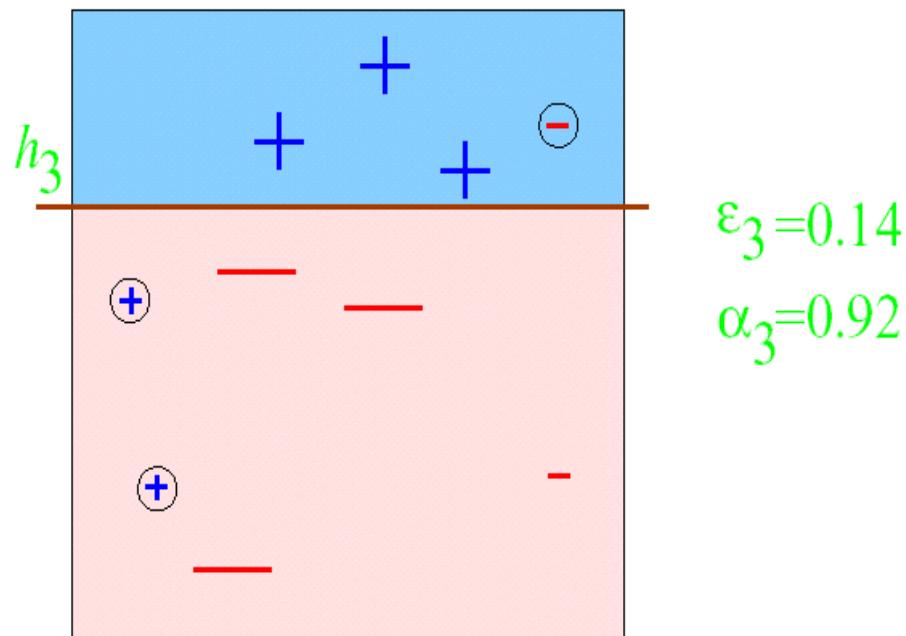


$$\begin{aligned}\varepsilon_2 &= 0.21 \\ \alpha_2 &= 0.65\end{aligned}$$

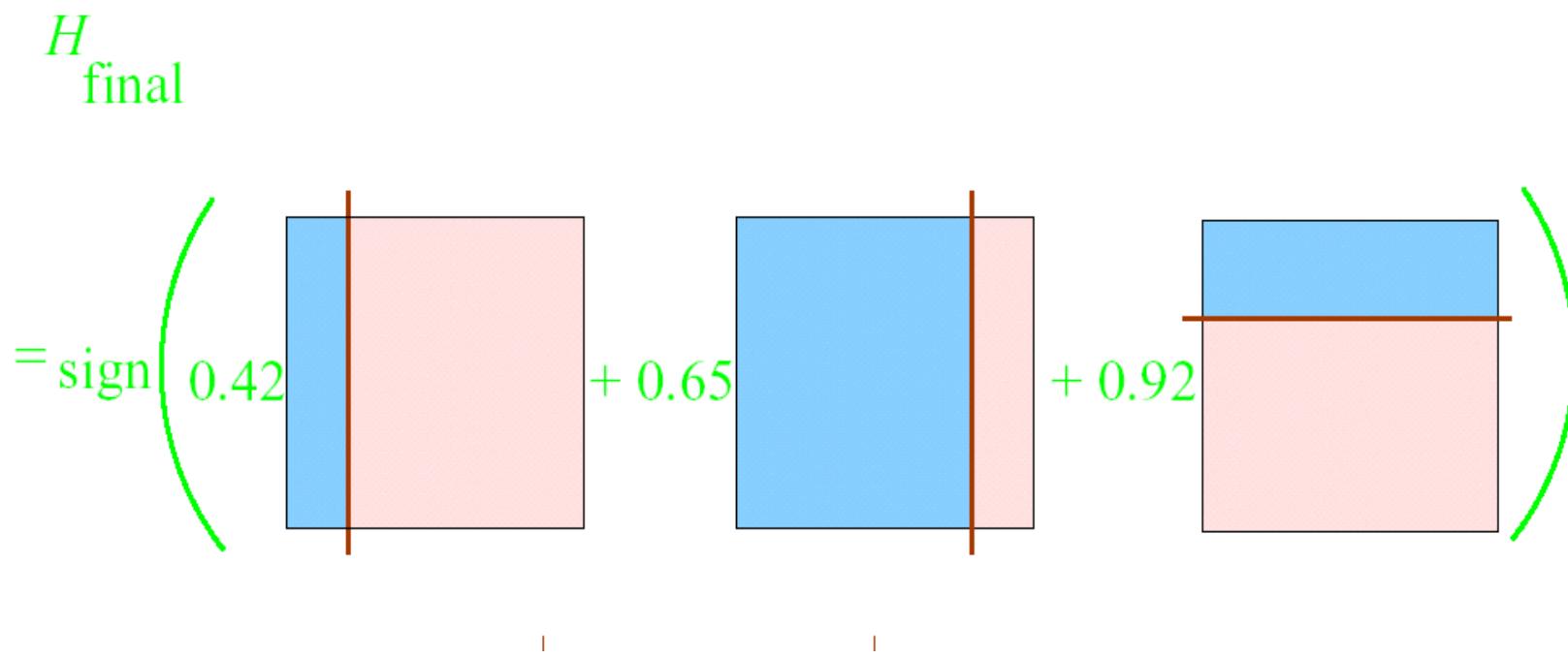


# AdaBoost example

ROUND 3



# AdaBoost example



## Boosting (cont.)

Boosting is remarkably resistant to overfitting, and it is fast and simple.

In fact, it can often continue to improve even when the training error has gone to zero.

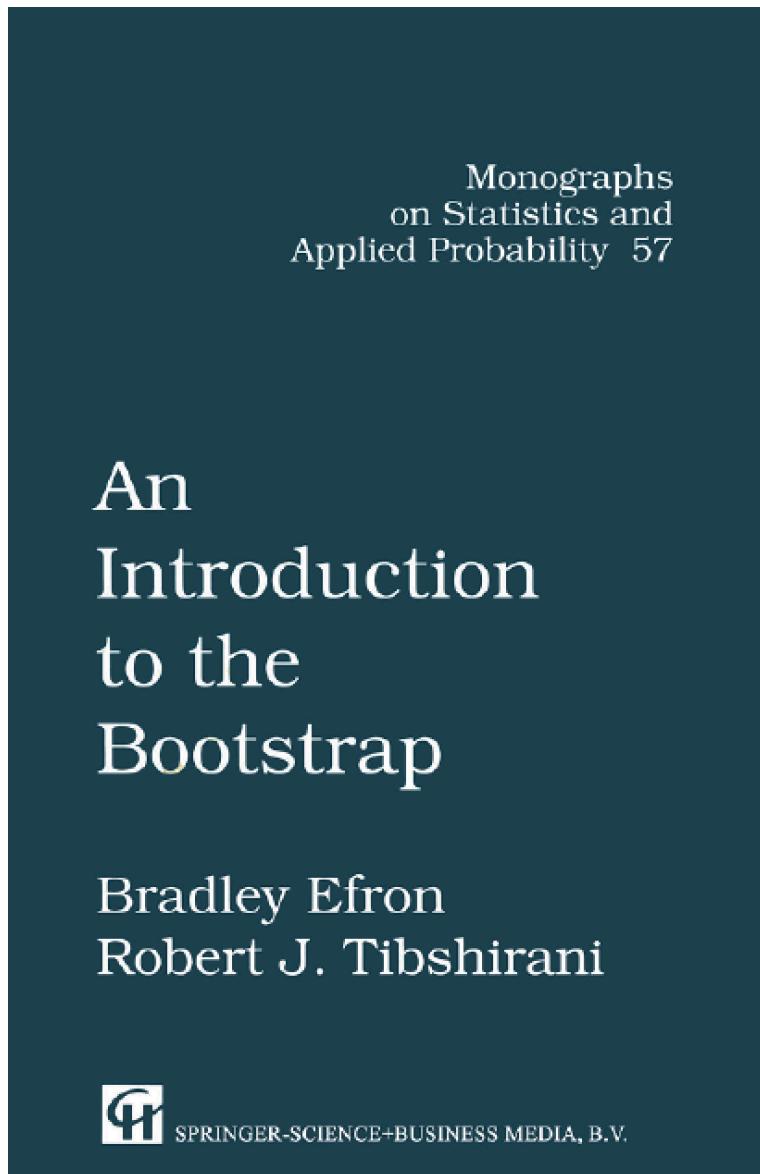
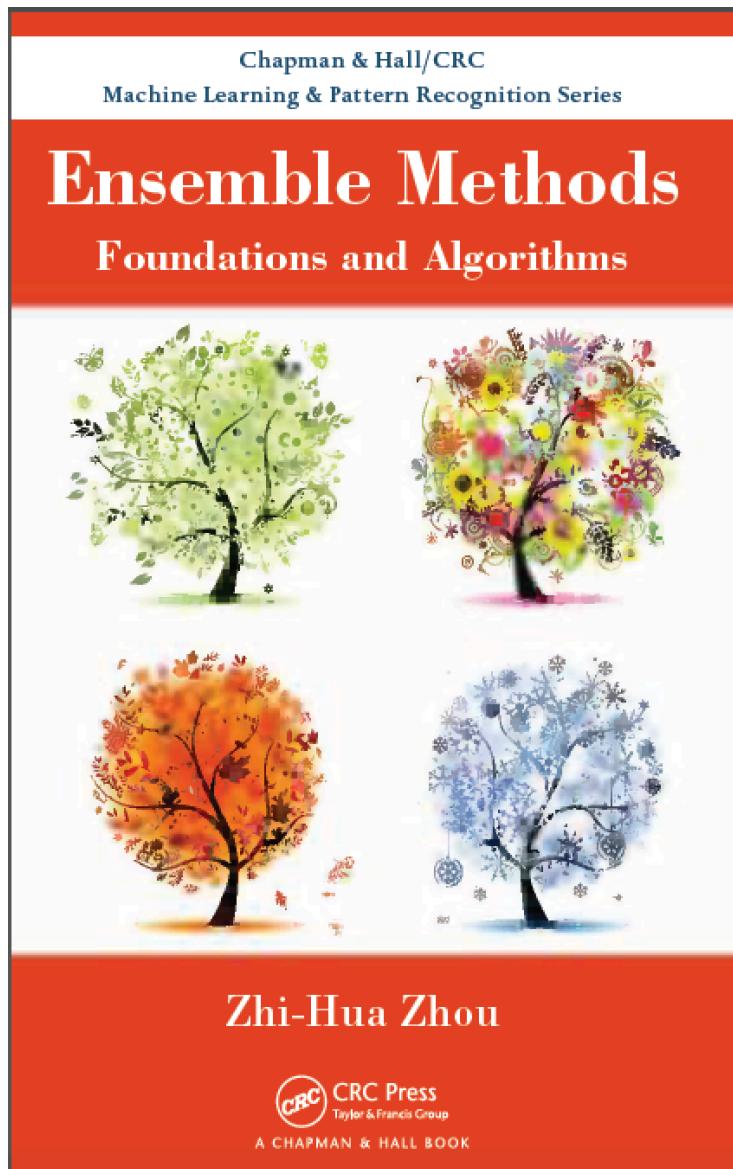
## Boosting (cont.)

It improves the performance of many kinds of machine learning algorithms.

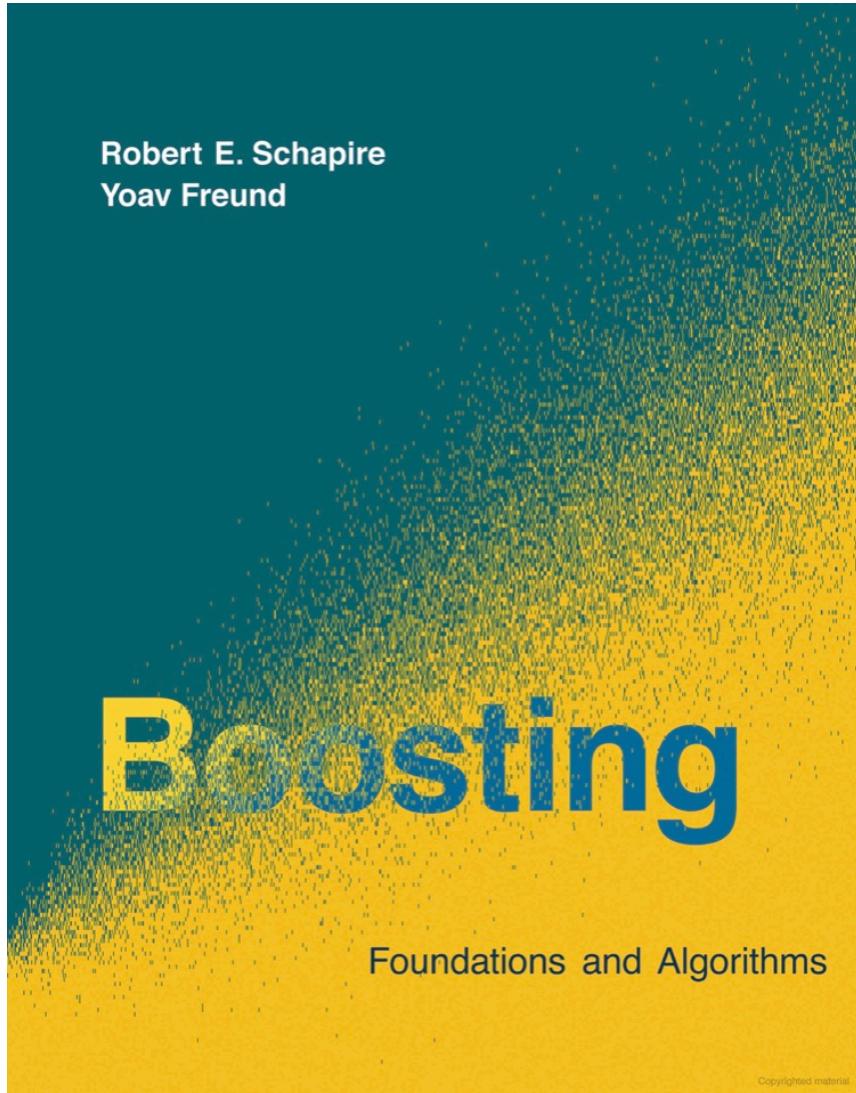
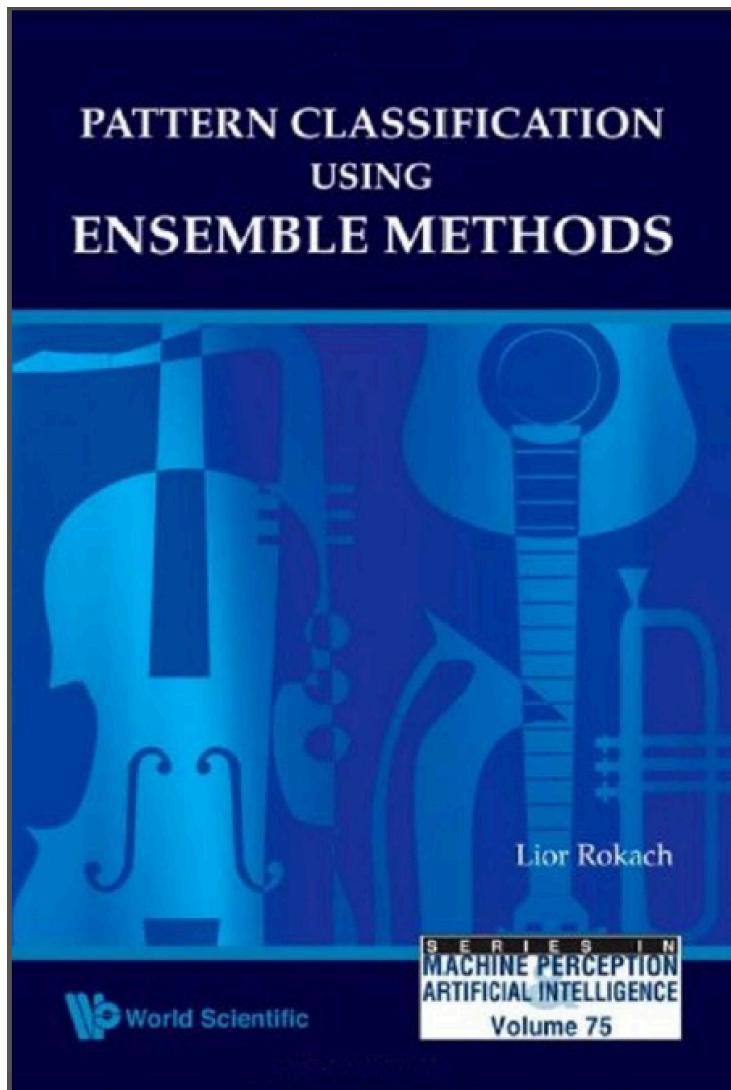
Boosting does not work when:

Not enough data, base learner is too weak or too strong, and/or susceptible to noisy data.

# More



# More



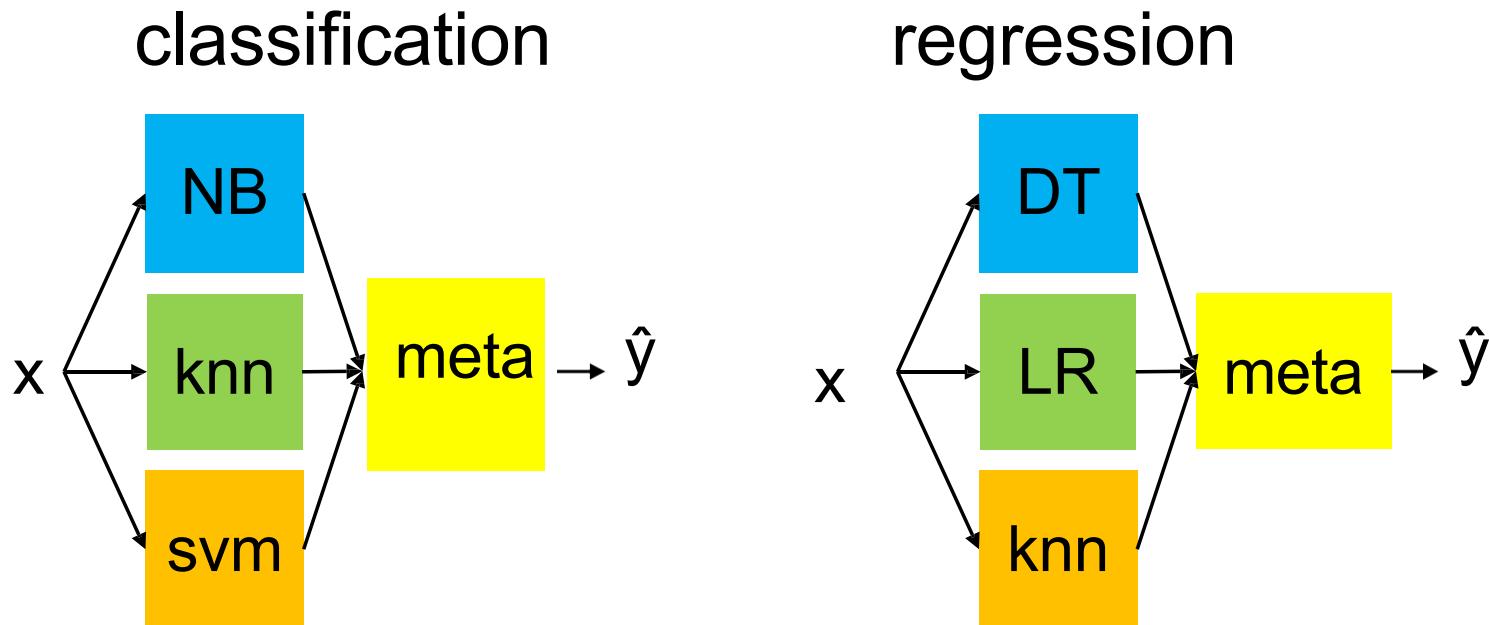
# Appendix

## More on Ensemble Methods

### Stacking

# Stacking

- Basic idea: use the output of multiple classifiers as input to a **meta-model (meta-learner)**.
- We ‘stack’ the meta-model on top of the base models



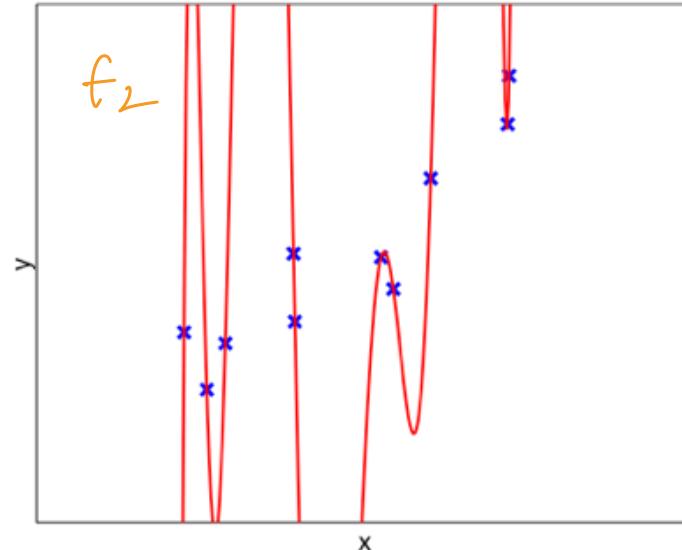
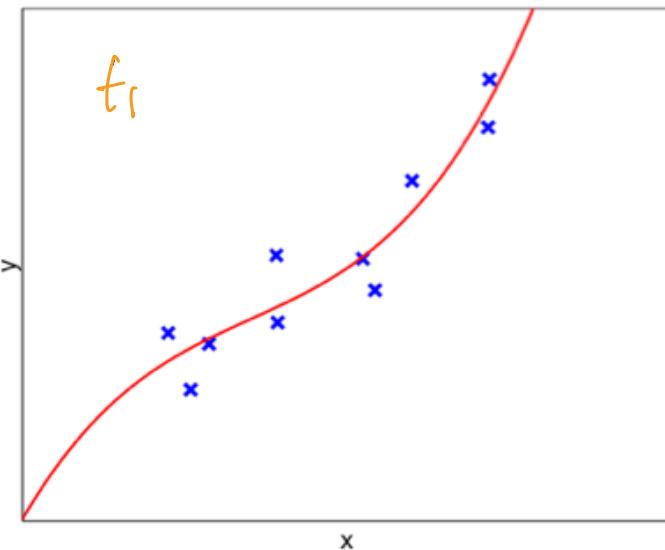
# Stacking – a naïve approach

- Let's consider the regression case
- Base model predictions are  $f_1(\mathbf{x}), \dots, f_L(\mathbf{x})$
- Meta learner could be a linear regression model
$$f_{\text{meta}}(\mathbf{x}) = \sum_{i=1}^L w_i f_i(\mathbf{x})$$
- If we could choose  $w_i$  to minimize true error, the stacked model would always be at least as good as any base model!
  - Worst case we set all  $w_i$  to 0 except that of the best base model
- But what if we minimize the train error instead?

# Stacking – a naïve approach

$$f = w_1 f_1 + w_2 f_2$$

- Consider the following base models



- What weights would minimize train set error?
- Does that yield good generalization error for the meta model?

Venkatesh Madi

# Stacking – a naïve approach

- Naïve implementation of stacking prefers over-fitted models
- Underlying problem: the outputs of the base models have been adapted to the labels.

# Stacking – a naïve approach

- Thus, inputs of the meta model are **not representative** of the inputs it will get at test-time.
- To avoid preference for overfitted models, inputs to the meta- model should not have seen the labels for the data points themselves

# Stacking – second attempt

- Now, we can train the meta-model on the data in the base model outputs paired with the target label
- Any base-model output is now a good indication of test-time behavior
- If the meta-model has free parameters itself, we can cross-validate using the same folds

# Stacking – second attempt

- Usually, the meta-model is relatively simple (e.g. linear regression or logistic regression)
- Empirical Recommendation: Do not have your base-learners as the meta-learner.  
For example, if you use logistic regression as the base-learner, use some other classifier (e.g. SVM) as the meta-learner.

# Testing the stacked model

- To test the stacked model, again we set aside a test set from the very beginning
- Have several versions of the base models from cross-validation!

# Testing the stacked model

- Two approaches:
  - Retrain the base models on the whole dataset
    - Possible disadvantage: slightly different input to meta-model
  - Use an average of the trained base models
    - Possible disadvantage: time cost
- Then feed the base model predictions into the trained meta- model

# Comparison to model selection

- If we force meta-learner to use just one base model (with weight 1) and set all other weights to 0, this is equivalent to selecting the best model with cross-validation
- More expressive meta-models (e.g. linear / logistic regression) can leverage the relative strength of multiple models

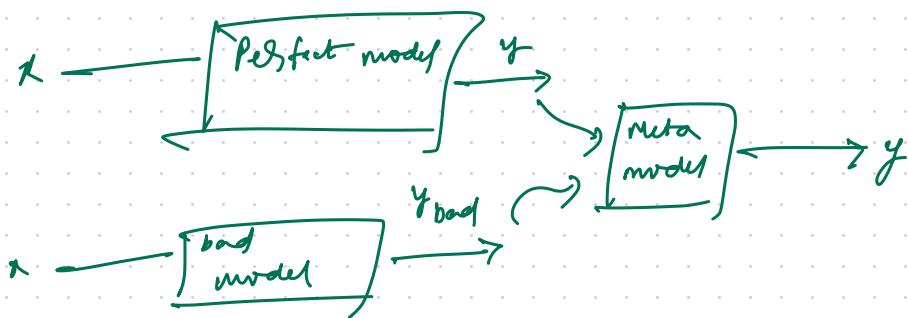
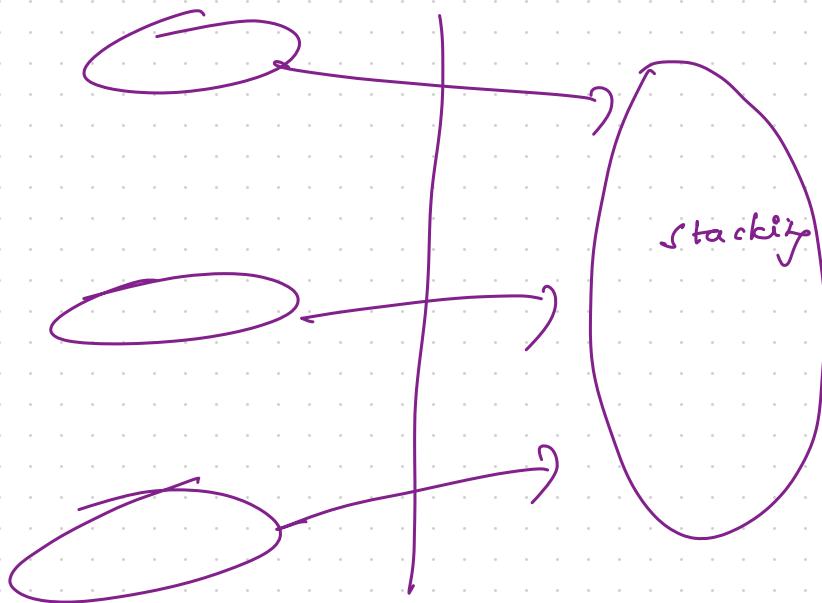
# Comparison to model selection

- A very complex meta-model (e.g. decision tree) could again easily overfit
- Could use cross-validation on the meta-level to ensure good generalization properties

# Effectiveness of stacking

- Stacking generally improves performance, but not by much
- Additional cost of training and evaluating multiple models

## Federated Learning (Distributed)



# Effectiveness of stacking

- If **interpretability or speed are important** consideration, stacking might not help you much.
- In competitions where **a small gain is important** and time cost is not so much of an issue, it is usually effective!
- Quite **useful in collaborative approaches** where everyone can integrate their own model in overall system

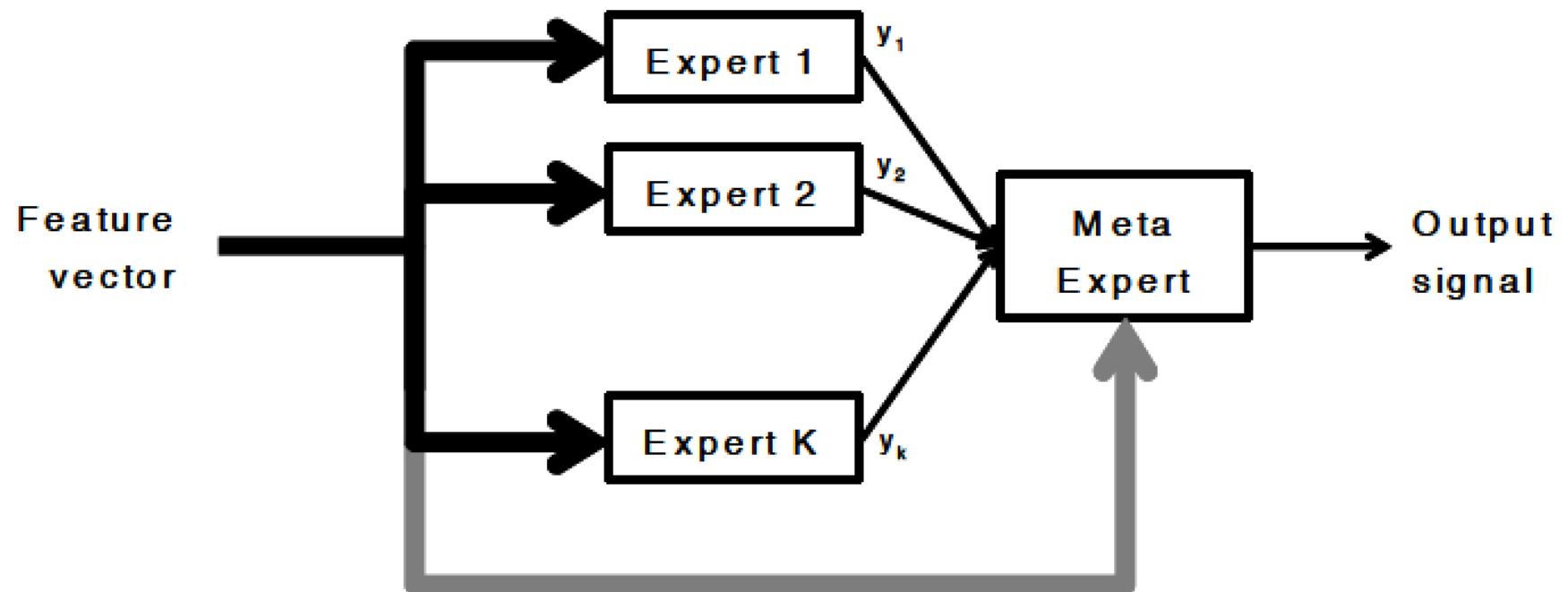
# Adaptive Meta-Learners

- The meta-learner is a function that depends on the input feature vector
- Thus, the ensemble implements a function that is local to each region in feature space
- This **divide-and-conquer approach** leads to **modular ensembles** where relatively simple classifiers **specialize** in different parts of I/O space

# Adaptive Meta-Learners

- In contrast with static-combiner ensembles, the individual experts here do not need to perform well for all inputs, only in their region of expertise
- Representative examples of this approach are **Mixture of Experts (ME)** and **Hierarchical ME** [Jacobs et al., 1991; Jordan and Jacobs, 1994]

# Adaptive Meta-Learners

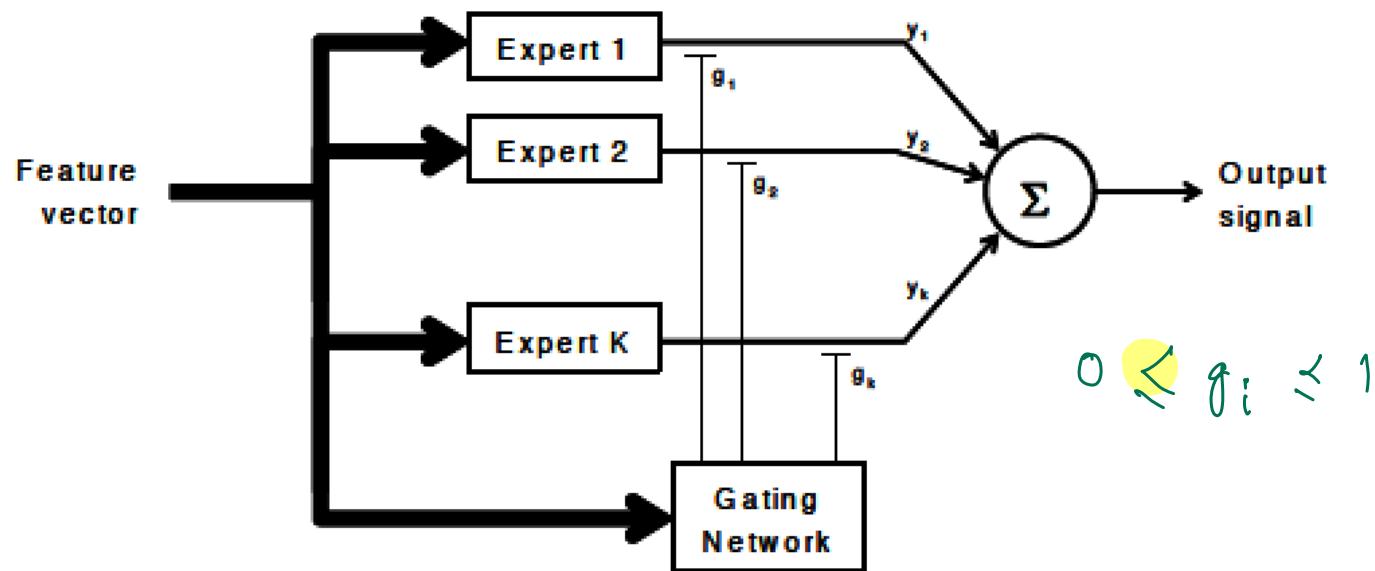


# Mixture of Experts

- ME is the classical adaptive ensemble method
- A **gating network** is used to partition feature space into different regions, with one expert in the ensemble being responsible for generating the correct output within that region [Jacobs et al., 1991]

# Mixture of Experts

- The experts in the ensemble and the gating network **are trained simultaneously**.
- ME can be extended to a multi-level hierarchical structure, where each component is itself a ME.



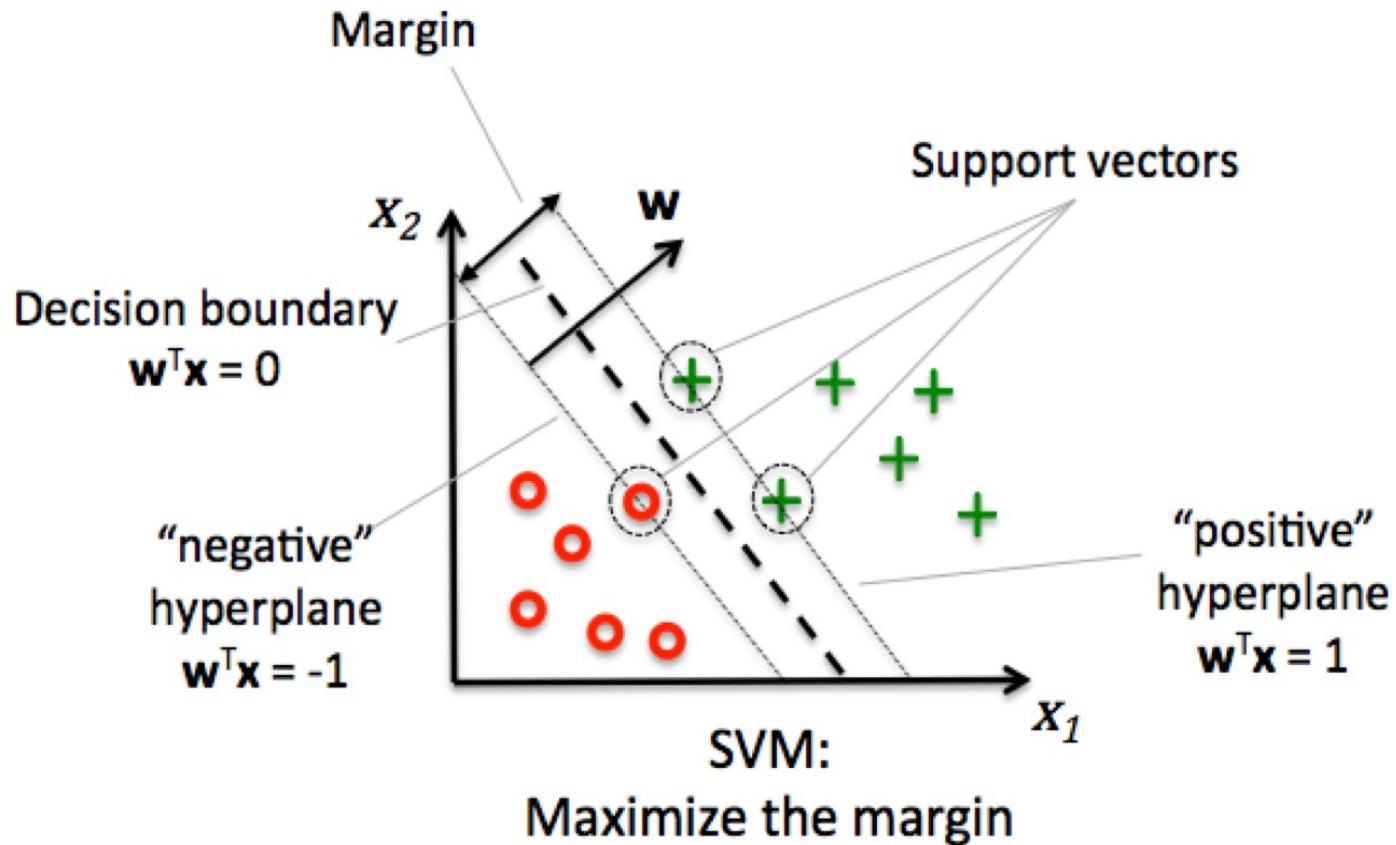
# **DSCI 552, Machine Learning for Data Science**

University of Southern California

M. R. Rajati, PhD

# Lesson 7

## Support Vector Machines



# Support Vector Machines

Here we approach the two-class classification problem in a direct way:

*We try and find a plane that separates the classes in feature space.*

If we cannot, we get creative in two ways:

- We soften what we mean by “separates”, and
- We enrich and enlarge the feature space so that separation is possible.

# What is a Hyperplane?

- A hyperplane in  $p$  dimensions is a flat affine subspace of dimension  $p - 1$ .  
*never passes the origin* *i.e. is 3D we have 2D plane*
- In general the equation for a hyperplane has the form

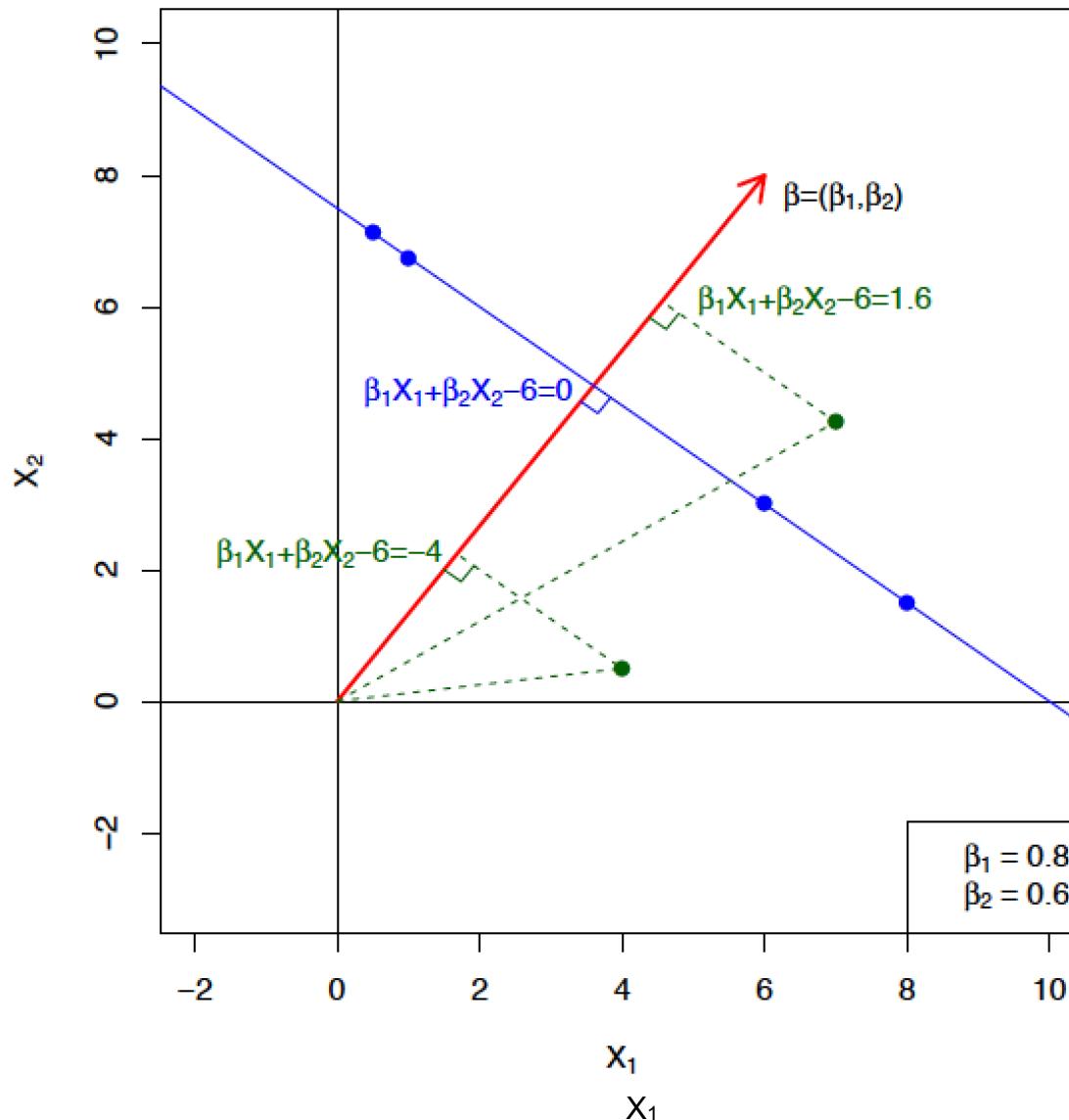
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = 0$$

- In  $p = 2$  dimensions a hyperplane is a line.

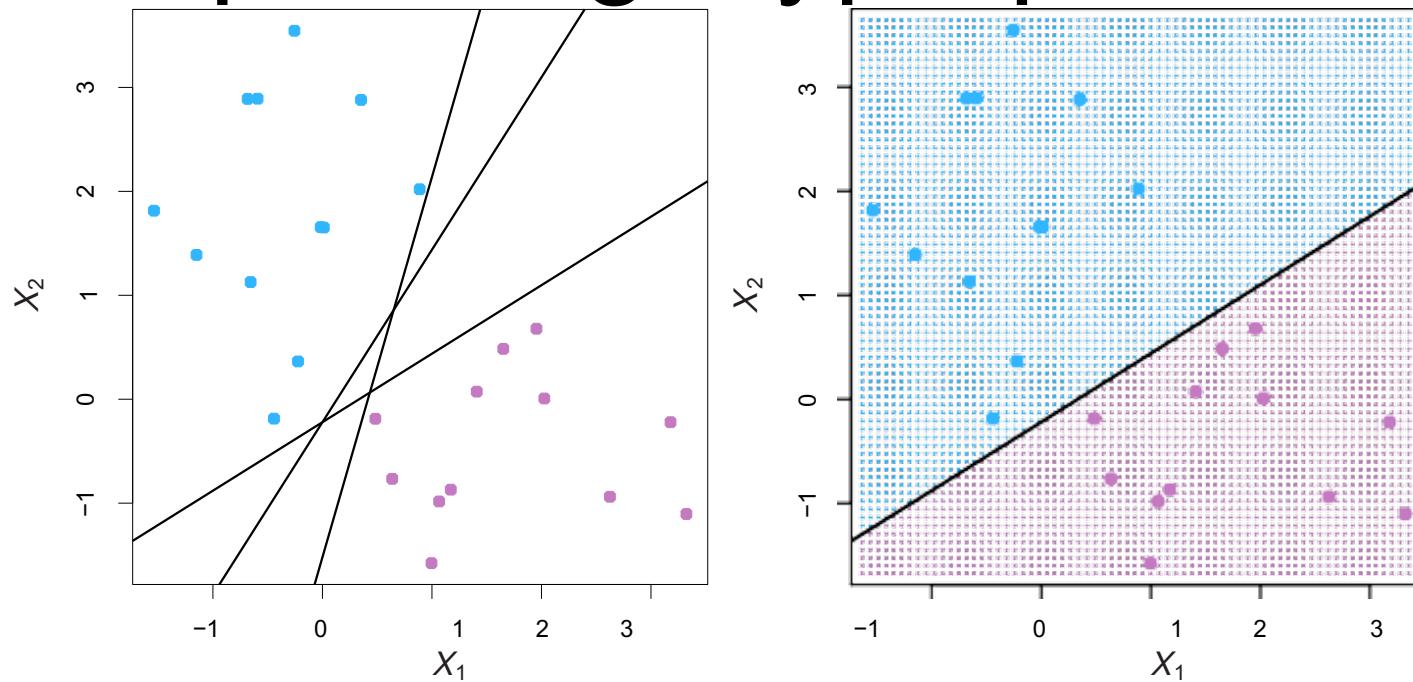
# What is a Hyperplane?

- If  $\beta_0 = 0$ , the hyperplane goes through the origin, otherwise not.
- The vector  $\beta = (\beta_1, \beta_2, \dots, \beta_p)$  is called the normal vector
  - it points in a direction orthogonal to the surface of a hyperplane.

# Hyperplane in 2 Dimensions



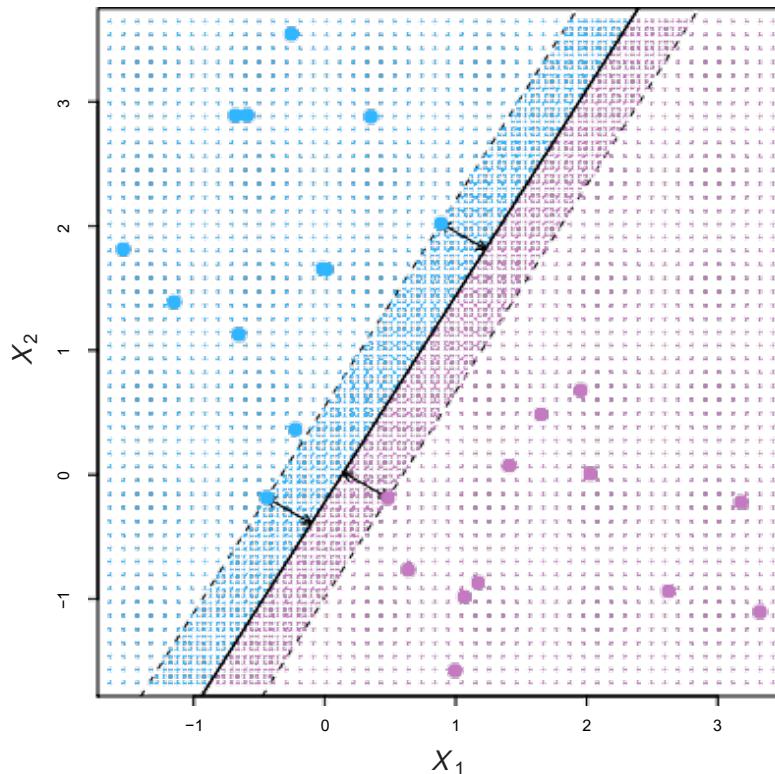
# Separating Hyperplanes



- If  $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ , then  $f(X) > 0$  for points on one side of the hyperplane, and  $f(X) < 0$  for points on the other.  $f(x) = 0$  for the points on the hyperplane.
- If we code the colored points as  $Y_i = +1$  for blue, say, and  $Y_i = -1$  for mauve, then if  $Y_i \cdot f(X_i) > 0$  for all  $i$ ,  $f(X) = 0$  defines a *separating hyperplane*.

# Maximal Margin Classifier

Among all separating hyperplanes, find the one that makes the biggest gap or margin between the two classes.



Constrained optimization problem

$$\underset{\beta_0, \beta_1, \dots, \beta_p}{\text{maximize}} M$$

$$\|\beta\|_2^2 = 1$$

$$\text{subject to } \sum_{j=1}^p \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M$$

$$\text{for all } i = 1, \dots, N.$$

This can be rephrased as a convex quadratic program, and solved efficiently.

# Maximal Margin Classifier

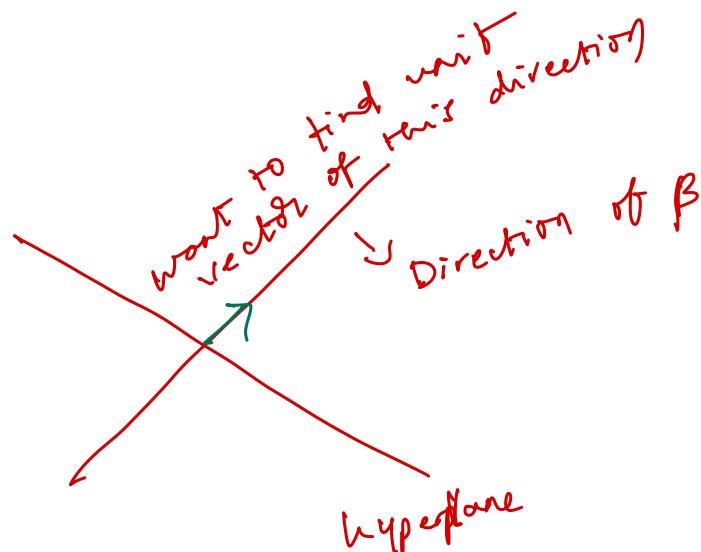
maximize  $M$   
 $\beta_0, \beta_1, \dots, \beta_p$

subject to  $\sum_{j=1}^p \beta_j^2 = 1,$

$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M$   
for all  $i = 1, \dots, N.$

$$\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p = 0$$

$$k(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p) = 0$$



The constraint  $\sum_{j=1}^p \beta_j^2 = 1$  makes sure that a unique solution for  $\beta_i$ 's exists.

Combined with

$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M$   
for all  $i = 1, \dots, N.$

it guarantees that the minimum margin is  $M$  if  $M$  is positive.

# Maximal Margin Classifier

For each observation to be on the correct side of the hyperplane we would simply need

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) > 0$$

so the constraint

$$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M$$

for all  $i = 1, \dots, N$ .

in fact requires that each observation be on the correct side of the hyperplane, *with some cushion*, provided that  $M$  is positive.

# Maximal Margin Classifier

Since if

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} = 0$$

Defines a hyperplane, then

$$k(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) = 0$$

also defines a hyperplane for any nonzero  $k$

$$\sum_{j=1}^p \beta_j^2 = 1$$

guarantees a unique set of  $\beta$ 's exists,

# Maximal Margin Classifier

Moreover, the constraint

$$\sum_{j=1}^p \beta_j^2 = 1$$

adds meaning to

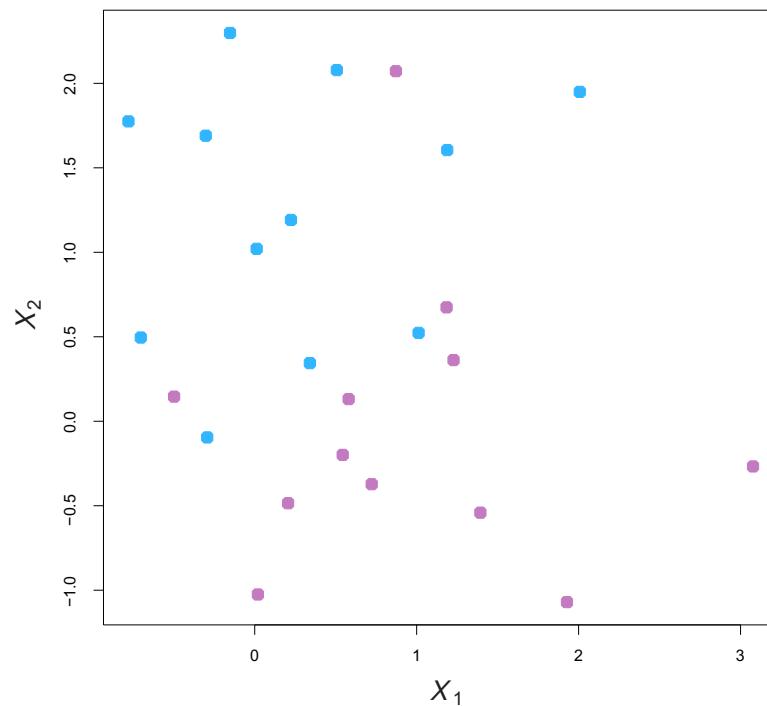
$$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M$$

for all  $i = 1, \dots, N.$

one can show that with this constraint the perpendicular distance from the  $i^{\text{th}}$  observation to the hyperplane is given by

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip})$$

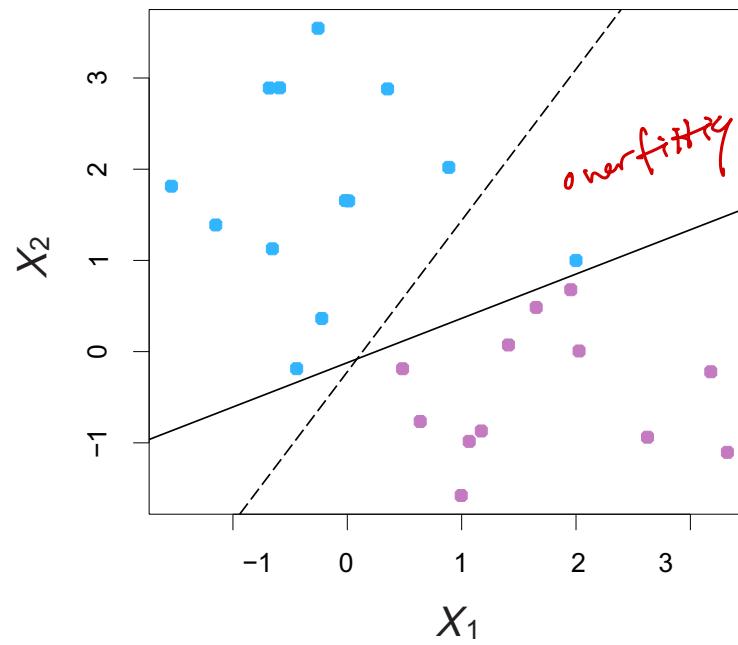
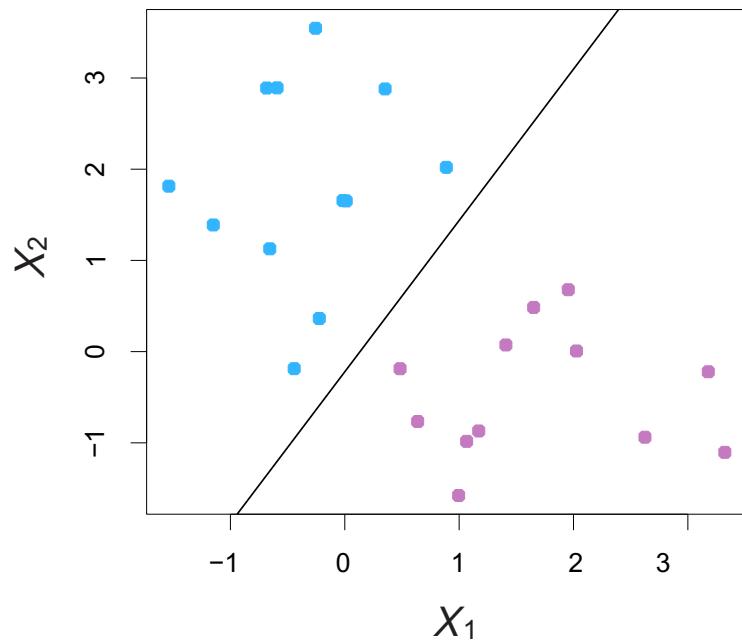
# Non-separable Data



The data on the left are not separable by a linear boundary.

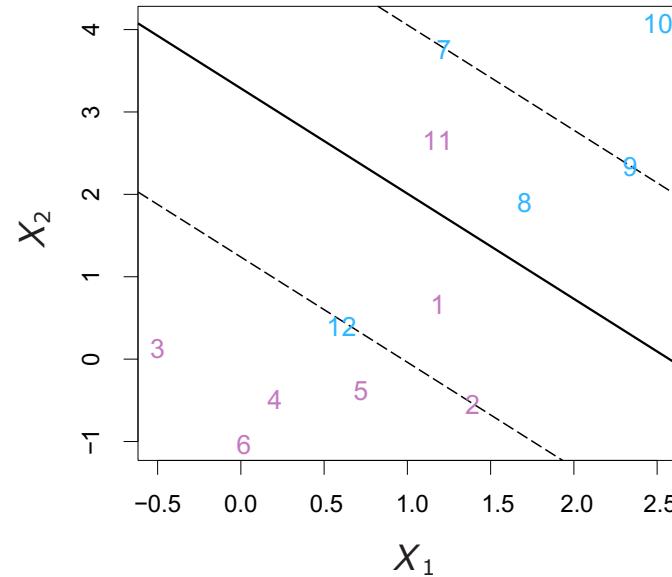
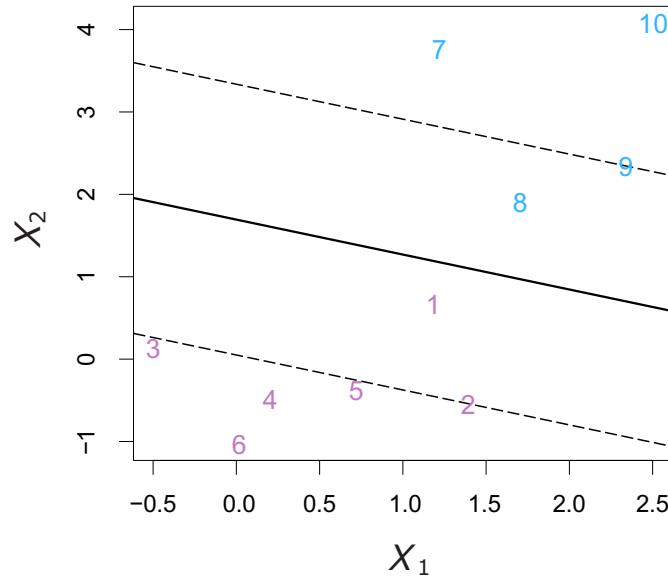
This is often the case, unless  $N < p$ .

# Noisy Data



Sometimes the data are separable, but noisy. This can lead to a poor solution for the maximal-margin classifier. The *support vector classifier* maximizes a *soft* margin.

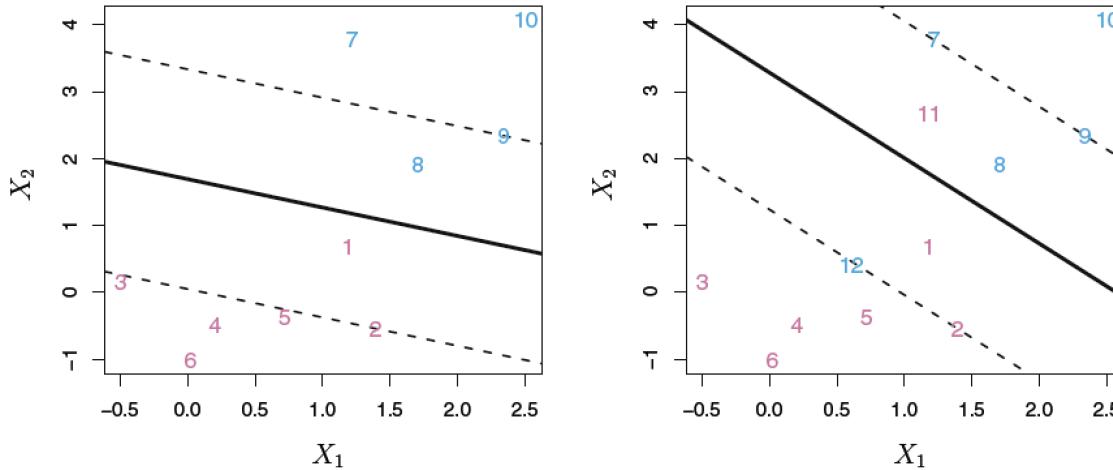
# Support Vector Classifier



$$\begin{aligned}
 & \underset{\beta_0, \beta_1, \dots, \beta_p, \epsilon_1, \dots, \epsilon_n}{\text{maximize}} \quad M \quad \text{subject to} \quad \sum_{j=1}^p \beta_j^2 = 1, \\
 & \quad \underbrace{\epsilon_1, \dots, \epsilon_n}_{\text{slack variables}} \\
 & y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip}) \geq M(1 - \epsilon_i), \\
 & \epsilon_i \geq 0, \quad \sum_{i=1}^n \epsilon_i \leq C,
 \end{aligned}$$

if  $0 < \epsilon_i \leq 1$ , margin will be less (1st image)  
 $\epsilon_i > 1$   
 (2nd case, allows misclassification)

# Support Vector Classifier



**FIGURE 9.6.** Left: A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines. Purple observations: Observations 3, 4, 5, and 6 are on the correct side of the margin, observation 2 is on the margin, and observation 1 is on the wrong side of the margin. Blue observations: Observations 7 and 10 are on the correct side of the margin, observation 9 is on the margin, and observation 8 is on the wrong side of the margin. No observations are on the wrong side of the hyperplane. Right: Same as left panel with two additional points, 11 and 12. These two observations are on the wrong side of the hyperplane and the wrong side of the margin.

# Details of Optimization Problem

The slack variable  $\varepsilon_i$  tells us where the  $i^{\text{th}}$  observation is located, relative to the hyperplane and relative to the margin.

- If  $\varepsilon_i = 0$  then the  $i^{\text{th}}$  observation is on the **correct side** of the margin

# Details of Optimization Problem

The slack variable  $\varepsilon_i$  tells us where the  $i^{\text{th}}$  observation is located, relative to the hyperplane and relative to the margin.

- If  $\varepsilon_i > 0$  then the  $i^{\text{th}}$  observation is on the **wrong side** of the margin, and we say that the  $i^{\text{th}}$  observation has *violated* the margin.

# Details of Optimization Problem

The slack variable  $\varepsilon_i$  tells us where the  $i^{\text{th}}$  observation is located, relative to the hyperplane and relative to the margin.

- If  $\varepsilon_i > 1$  then it is on the **wrong side of the hyperplane**.

# Details of Optimization Problem

- C bounds the sum of the  $\varepsilon_i$ 's, and so it determines the number and severity of the violations to the margin (and to the hyperplane) that we will tolerate.

# Details of Optimization Problem

- We can think of  $C$  as a budget for the amount that the margin can be violated by the  $N$  observations.
- If  $C = 0$  then there is no budget for violations to the margin, so all  $\varepsilon_i$ 's = 0, which leads to the maximal margin hyperplane optimization problem

# Details of Optimization Problem

For  $C > 0$  no more than  $C$  observations can be on the wrong side of the hyperplane, because if an observation is on the wrong side of the hyperplane then  $\varepsilon_i > 1$ , and the optimization problem requires that the sum of  $\varepsilon_i$ 's be less than  $C$ .

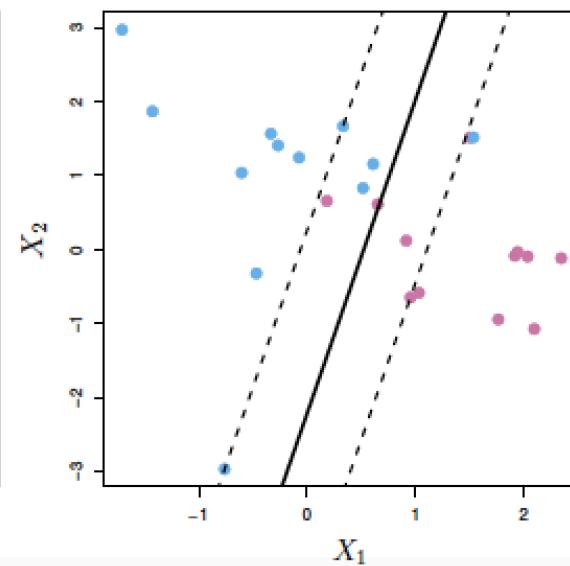
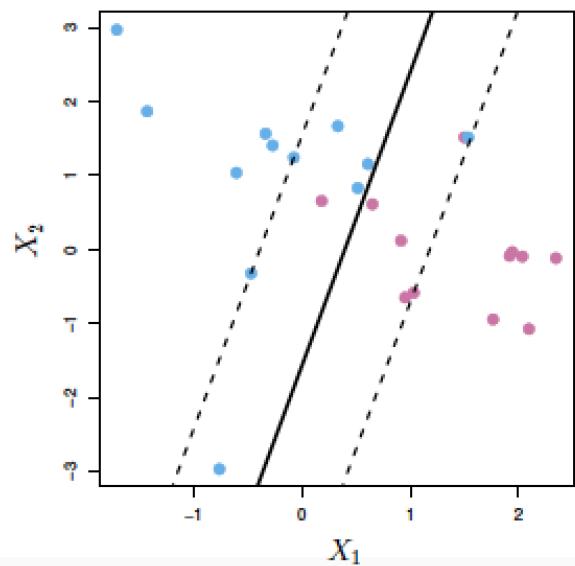
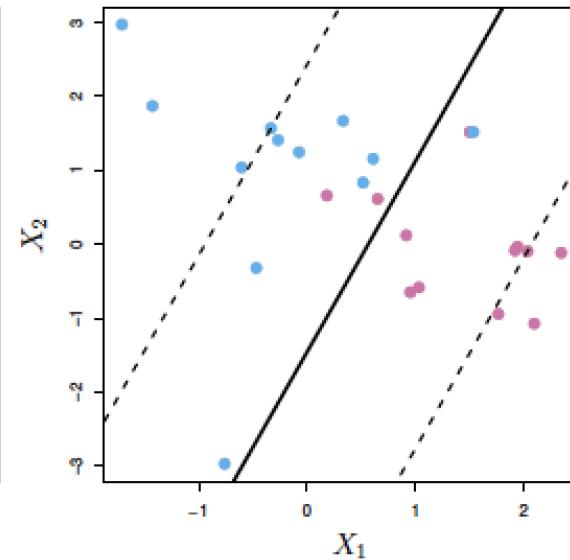
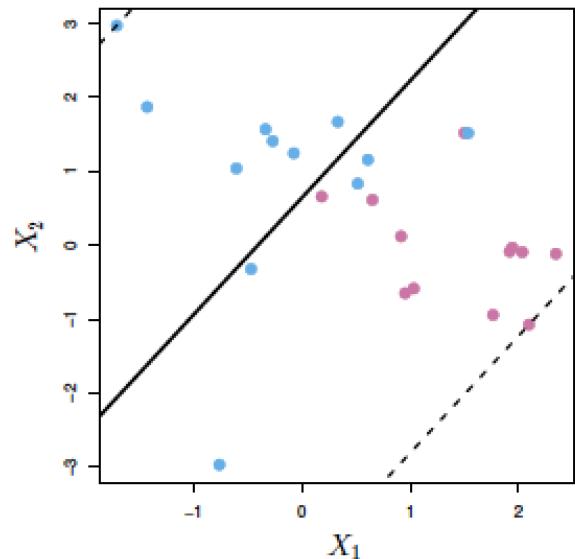
Determine ' $C$ ' by cross validation

# Details of Optimization Problem

- As the budget  $C$  increases, we become more tolerant of violations to the margin, and so the margin will widen.
- Conversely, as  $C$  decreases, we become less tolerant of violations to the margin and so the margin narrows.
  - An example is shown in the next slide.

# C is a regularization parameter

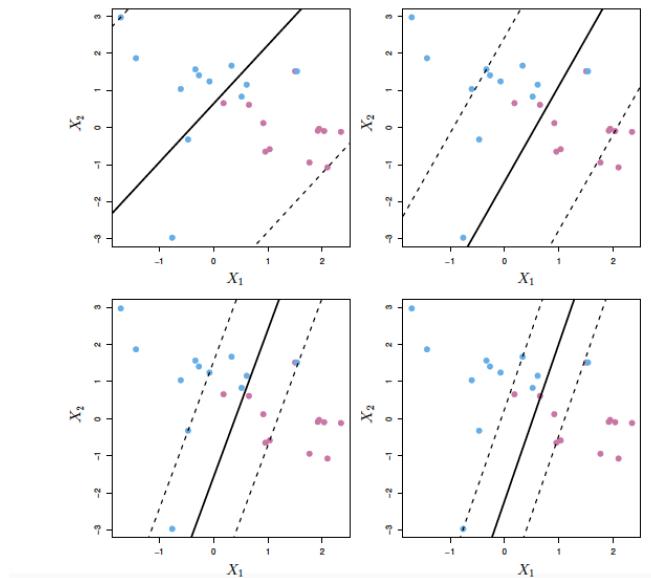
Large C



# C is a regularization parameter

A support vector classifier was fit using four different values of the tuning parameter C.

- The largest value of C was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels.



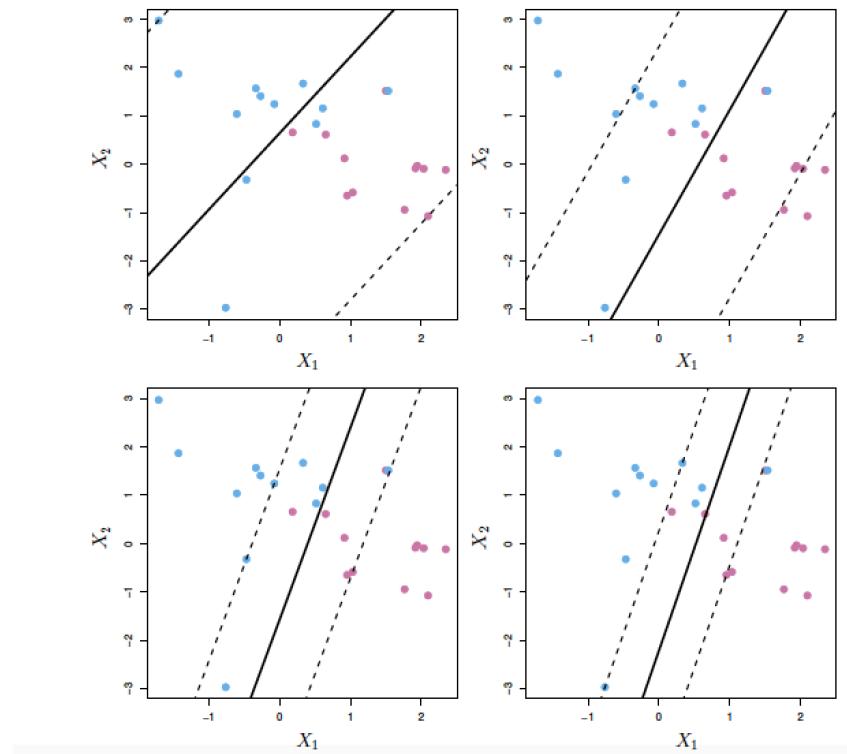
# C is a regularization parameter

A support vector classifier was fit using four different values of the tuning parameter C.

- When C is large, there is a high tolerance for observations being on the wrong side of the margin, and so the margin will be large.
- As C decreases, the tolerance for observations being on the wrong side of the margin decreases, and the margin narrows.

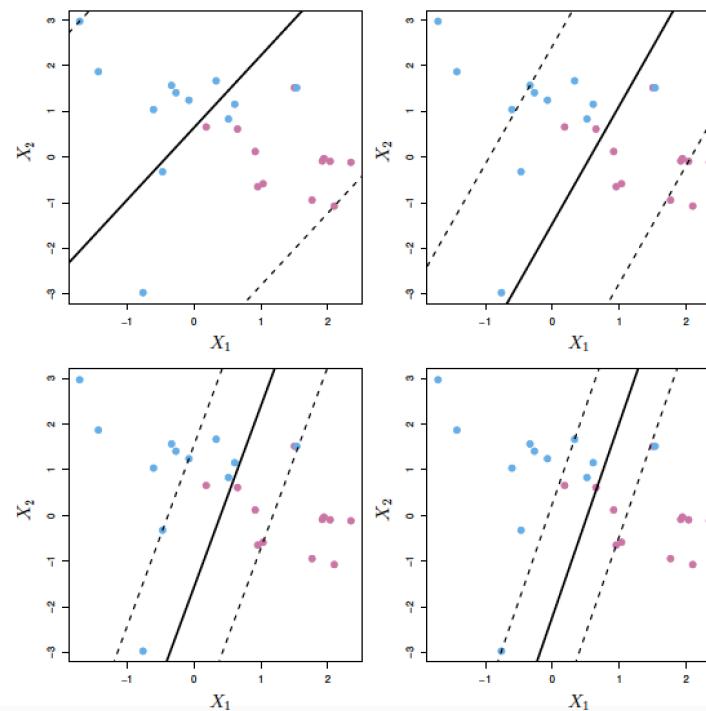
# C is a regularization parameter

- C is treated as a tuning parameter generally chosen via cross-validation.
- **C controls the bias-variance trade-off of the statistical learning technique.**



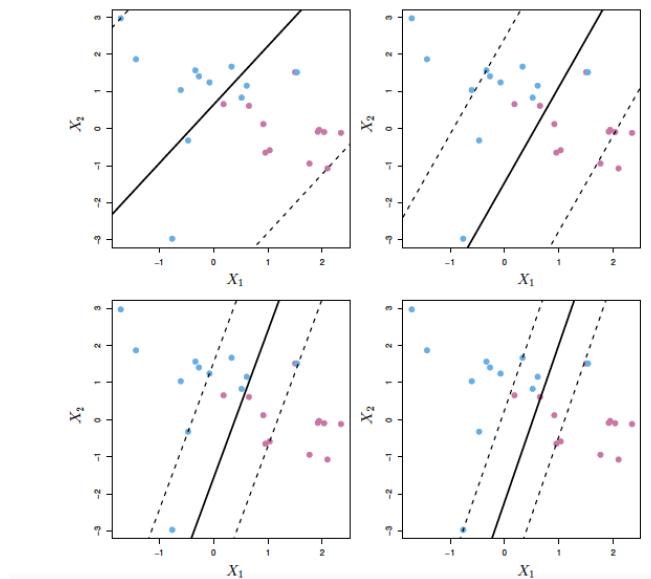
# C is a regularization parameter

- When C is small, we seek narrow margins that are rarely violated; this amounts to a classifier that is highly fit to the data, which may have low bias but high variance.



# C is a regularization parameter

- On the other hand, when C is larger, the margin is wider and we allow more violations to it; this amounts to fitting the data less hard and obtaining a classifier that is potentially more biased but may have lower variance.

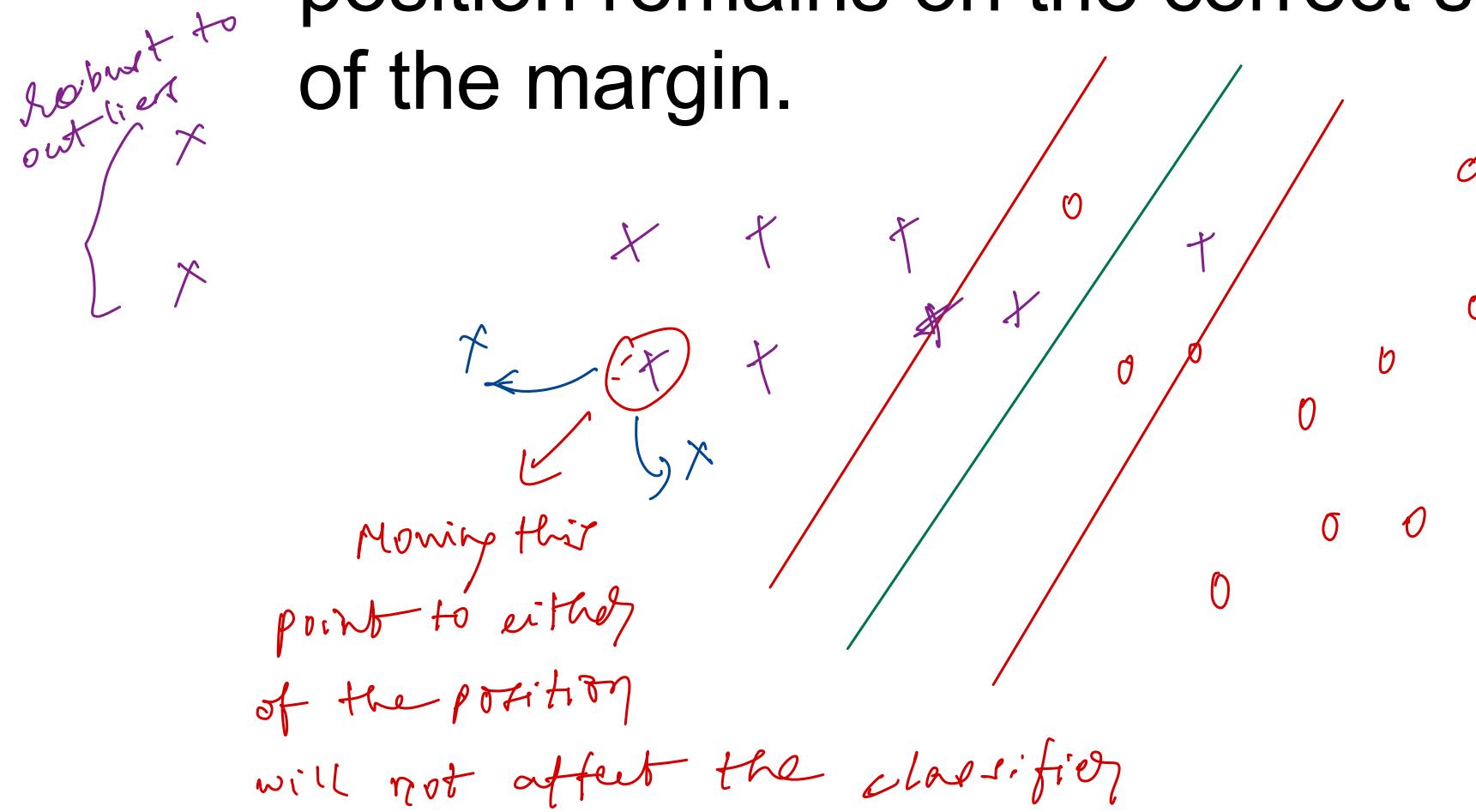


# Support Vectors

- Only observations that either lie on the margin or that violate the margin will affect the hyperplane, and hence the classifier obtained.
- In other words, an observation that lies strictly on the correct side of the margin does not affect the support vector classifier!

# Support Vectors

- Changing the position of that observation would not change the classifier at all, provided that its position remains on the correct side of the margin.



# Support Vectors

- Observations that lie directly on the margin, or on the wrong side of the margin for their class, are known as support vectors . These observations do affect the support vector classifier.

# Support Vectors

The fact that the support vector classifier's decision rule is based only on a potentially small subset of the training observations (the support vectors) means that it is quite robust to the behavior of observations that are far away from the hyperplane.

# Comparison with LDA

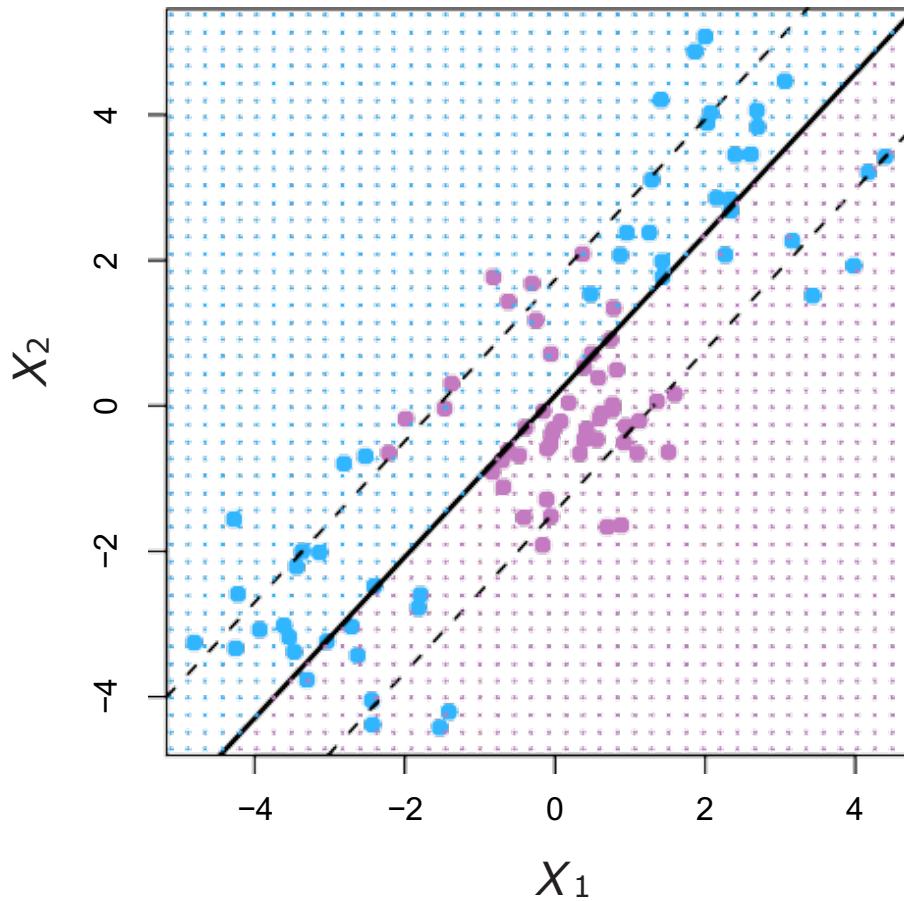
- This property is distinct from some of the other classification methods that we have seen, such as linear discriminant analysis.
- The LDA classification rule depends on the mean of all of the observations within each class, as well as the within-class covariance matrix computed using all of the observations.

LDA  
not robust  
to outliers

# Comparison with LR

In contrast, logistic regression, unlike LDA, has very low sensitivity to observations far from the decision boundary. In fact we will see that the support vector classifier and logistic regression are closely related.

# Linear boundary can fail



Sometimes a linear boundary simply won't work, no matter what value of  $C$ .

The example on the left is such a case.

What to do?

Feature transformation?

# Feature Expansion

Enlarge the space of features by including transformations; e.g.  $X_1^2, X_1^3, X_1X_2, X_1X_2^2, \dots$ . Hence go from a  $p$ -dimensional space to a  $M > p$  dimensional space.

- Fit a support-vector classifier in the enlarged space.
- This results in non-linear decision boundaries in the original space.

# Feature Expansion

Example: Suppose we use  $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$  instead of just  $(X_1, X_2)$ . Then the decision boundary would be of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0$$

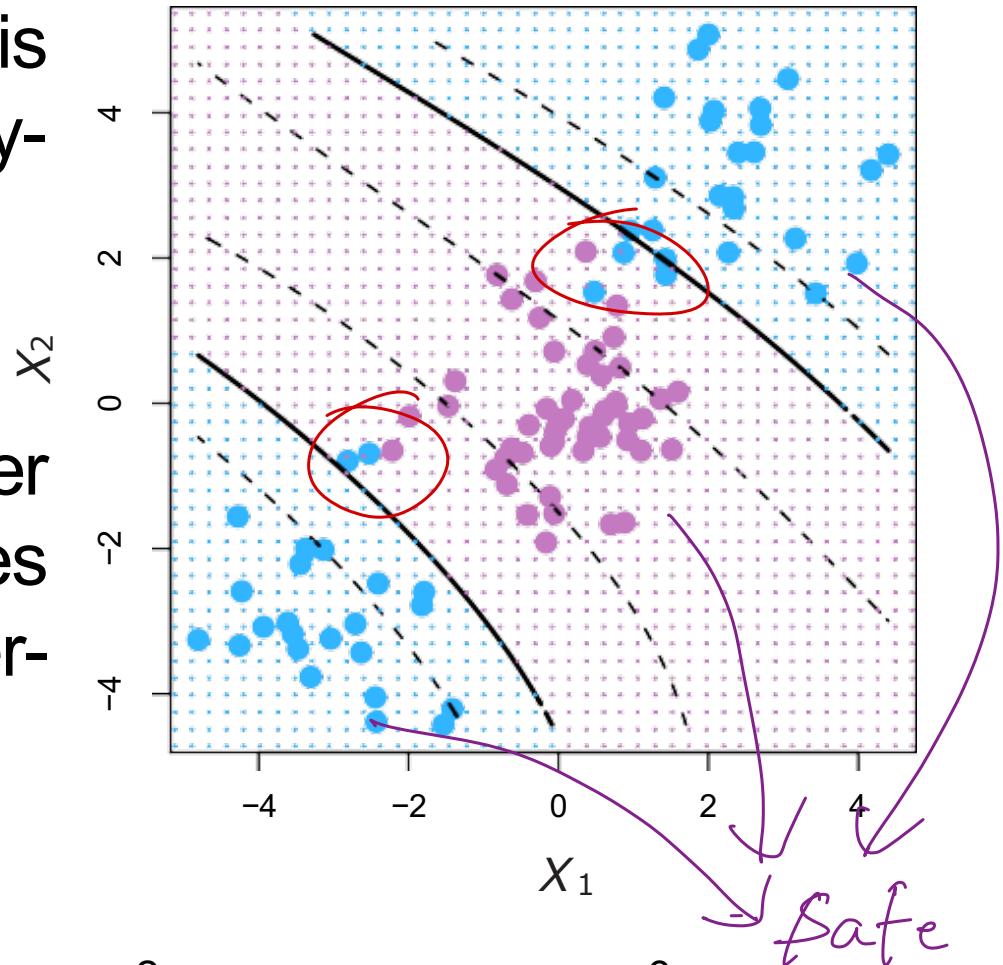
This leads to nonlinear decision boundaries in the original space (quadratic conic sections).

# Cubic Polynomials

Here we use a basis expansion of cubic polynomials

From 2 variables to 9

The support-vector classifier in the enlarged space solves the problem in the lower-dimensional space



$$\begin{aligned} & \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + \beta_6 X_1^3 \\ & + \beta_7 X_2^3 + \beta_8 X_1 X_2^2 + \beta_9 X_1^2 X_2 = 0 \end{aligned}$$

# Nonlinearities and Kernels

- Polynomials (especially high-dimensional ones) get wild rather fast.
- There is a more elegant and controlled way to introduce nonlinearities in support-vector classifiers — through the use of *kernels*.
- Before we discuss these, we must understand the role of *inner products* in support-vector classifiers.

$$\left\langle \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix} \right\rangle = \begin{array}{l} 1 \times 0 + 2 \times 1 + 3 \times 2 \\ = 8 \end{array}$$

# Inner products and support vectors

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^p x_{ij} x_{i'j} \text{ inner product between vectors}$$

- The linear support vector classifier can be represented as

test point

$$f(x) = \beta_0 + \sum_{i=1}^n \alpha_i \langle x, x_i \rangle$$

*with n parameters*

*Subject to*

$$\sum_{i=1}^n \alpha_i = 0$$

# Inner products and support vectors

- The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^n \alpha_i \langle x, x_i \rangle$$

*The intercept can be calculated as:*

$$\beta_0 = \frac{1}{y_j} - \sum_{i=1}^N \alpha_i \langle x_i, x_j \rangle$$

*where  $x_j$  is one of the support vectors and  $y_j$  is its label.*

# Inner products and support vectors

- The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^n \alpha_i \langle x, x_i \rangle \quad \sum_{i=1}^n \alpha_i = 0$$

*with  $n$  parameters*

in order to evaluate the function  $f(x)$ , we need to compute the inner product between the new point  $x$  and each of the training points  $x_i$ .

*If you happen to know  $f(x)$  [may be eyeballing!]*

$$f(x_j) = \beta_0 + \sum_{i=1}^n \alpha_i \langle x_j, x_i \rangle$$

↑ known      ↑ unknown       $\langle x_j, x_i \rangle$  known

$\binom{n}{2}$  equations like  
same

$$\sum_{i=1}^n \alpha_i = 0 \rightarrow \text{known}$$

we can use 'known' things to find  
the unknown.

# Inner products and support vectors

To estimate the parameters  $\alpha_1, \dots, \alpha_n$  and  $\beta_0$ , all we need are the  $\binom{n}{2}$  inner products  $\langle x_i, x_j \rangle$  between all pairs of training observations.

However, it turns out that  $\alpha_i$  is nonzero only for the support vectors in the solution—that is, if a training observation is not a support vector, then its  $\alpha_i$  equals zero.

# Inner products and support vectors

In other words

$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i \langle x, x_i \rangle$$

$\sum_{i \in S} \hat{\alpha}_i = 0$

*x<sub>i</sub>* is a support vector

$$\beta_0 = \frac{1}{y_j} - \sum_{i \in S} \hat{\alpha}_i \langle x_i, x_j \rangle$$

*x<sub>i</sub>* and *x<sub>j</sub>* are support vectors

where *S* is the support set of indices *i* for support vectors.

# Inner products and support vectors

To expand the feature space, one can use a transformed set of features  $u = \varphi(x)$ . Note that  $u$  does not need to be of the same dimension as  $x$ .

The classifier in the new feature space can be represented as

$$f_1(x) = f(\varphi(x)) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i \langle \varphi(x), \varphi(x_i) \rangle$$

*transformed feature of test point*  $\downarrow$  *transformed feature of a support vector point*  $\leftarrow$

$$x = \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \quad u = \varphi(x) = \begin{bmatrix} n_1 \\ n_2^2 \\ n_1 n_2 \\ n_1^2 + n_2^2 \end{bmatrix}$$

# Inner products and support vectors

It can be shown that a class of functions called *kernels* can be represented as inner products  $\langle \varphi(x), \varphi(x_i) \rangle = k(x, x_i)$

The interesting point is that we do not need to explicitly know the function  $\varphi(x)$  to use the kernel that correspond to it!

They are generalizations of the inner product.

(co-linear)



$\langle \underline{x}_1, \underline{x}_2 \rangle$  max the inner product or minimizes



$\langle \underline{x}_1, \underline{x}_2 \rangle = 0$   
(orthogonal)



$\langle \underline{x}_1, \underline{x}_2 \rangle$  max -ve

$$\langle \underline{u}, \underline{v} \rangle = \|\underline{u}\| \|\underline{v}\| \cos \phi$$

$$\cos \phi = \frac{\langle \underline{u}, \underline{v} \rangle}{\|\underline{u}\| \|\underline{v}\|}$$

Similarity  $\in [-1, 1]$

# Kernels and Support Vector Machines

Now suppose that every time the inner product appears in our equations, we replace it with

$$K(x_i, x_{i'})$$

where  $K$  is a *kernel*.

A kernel is a function that quantifies the similarity of two observations.

For SVC, the Kernel is the usual inner product, which is called a *linear kernel*, because  $\varphi(x) = x$ .

# Aside: Mercer's Condition

- A function  $K(x_i, x_j)$  is a Kernel function, i.e. it can be written in the following form

$$K(x_i, x_j) = [\varphi(x_i)]^T [\varphi(x_j)] = \langle \varphi(x_i), \varphi(x_j) \rangle$$

if and only if it satisfies Mercer's condition.

# Aside: Mercer's Condition

- Mercer's condition: for all finite sequences  $(x_1, x_2, \dots, x_n)$  and all choices from the sequence of real numbers  $(c_1, c_2, \dots, c_n)$ :

$$\sum_{i=1}^n \sum_{j=1}^n c_i c_j K(x_i, x_j) \geq 0$$

*positive  
semi-definiteness*

# Kernels and Support Vector Machines

For example

$$1 + \langle x_i, x_{i'} \rangle$$

$$K(x_i, x_{i'}) = \left( 1 + \sum_{j=1}^p x_{ij} x_{i'j} \right)^d$$

computes the inner-products needed for  $d$  dimensional polynomials-basis functions!

# Kernels and Support Vector Machines

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^p x_{ij}x_{i'j}\right)^d$$

*Try it for  $p = 2$  and  $d = 2$ .*

- The solution has the form

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \hat{\alpha}_i K(x, x_i).$$



$$u = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad v = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix}$$

test point  $\rightarrow$

$$K(u, v) = (1 + \langle u, v \rangle)^d \quad d \gg 2$$

(poly kernel)

$$\begin{aligned}
 &= (1 + x_1 x'_1 + x_2 x'_2)^2 \\
 &= (1 + x_1 x'_1)^2 + (x_2 x'_2)^2 + 2(1 + x_1 x'_1) x_2 x'_2 \\
 &= 1 + (x_1 x'_1)^2 + 2x_1 x'_1 + (x_2 x'_2)^2 + 2x_2 x'_2 + \\
 &\quad 2x_1 x'_1 x_2 x'_2
 \end{aligned}$$

Note that no matter how large the 'd' is  
 the max no. of parameters ( $d_i$ 's) we  
 need to learn from the data is  $n$   
 (in practice, a fraction of  $n$ )

$$\approx 5\text{-}10 \text{ or } 10^{-1} \text{ of } n$$


---

In contrast, direct enlargement of the feature space using polynomial's of degree ' $d$ ' would proliferate the no. of parameters to be learned!

' $d$ ' is a hyper parameter obtained by cross validation.

Radial Basis Function (RBF)  
Gaussian Kernel

# Radial Kernel

*we don't know*

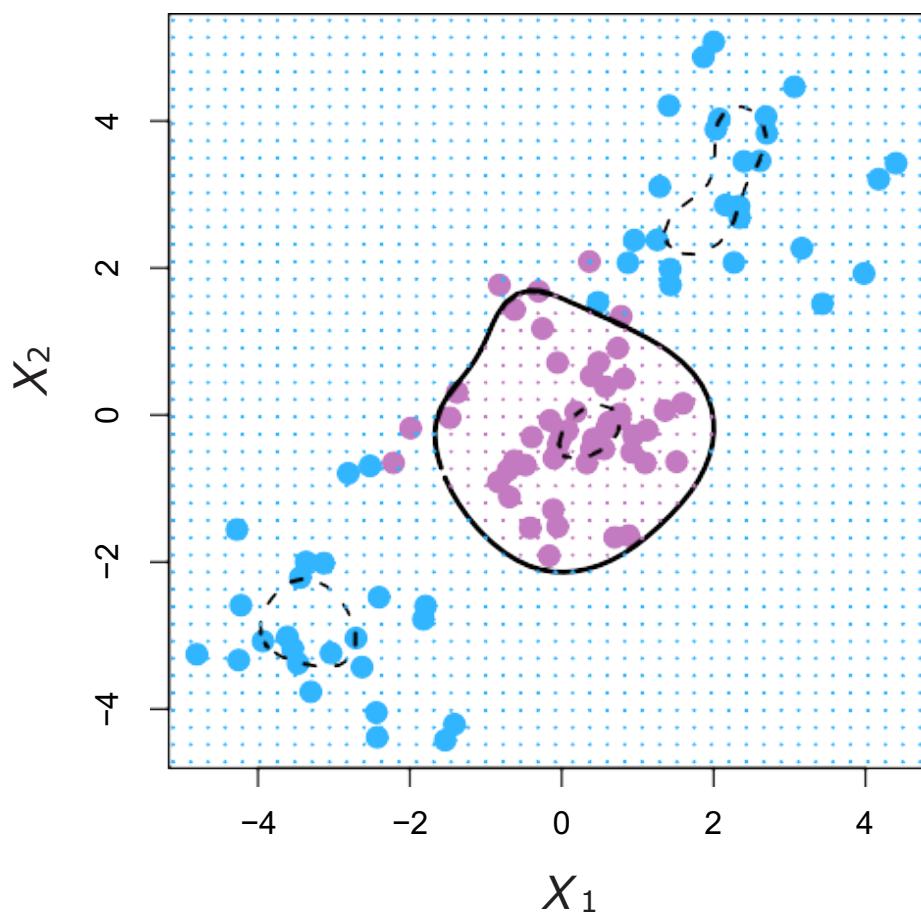
$\varphi(x_i)$

$\langle \varphi(x_i), \varphi(x_i') \rangle$

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^p (x_{ij} - x_{i'j})^2).$$

$$\|x_i - x_i'\|^2$$

$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i K(x, x_i)$$



Implicit feature space;  
very high dimensional.

Controls variance by  
squashing down most  
dimensions severely.

# RBF Kernel

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^p (x_{ij} - x_{i'j})^2).$$

- If a given test observation  $x^* = (x_1^* \dots x_p^*)^T$  is far from a training observation  $x_i$  in terms of Euclidean distance, then the exponent will be very negative, and so  $K(x^*, x_i)$  will be very tiny.

$$k(x^*, x_i) = \exp(-\gamma \|x^* - x_i\|_2^2)$$

$\approx 0$

# RBF Kernel

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^p (x_{ij} - x_{i'j})^2).$$

- Therefore,  $x_i$  will play virtually no role in  $f(x^*)$ .
- Recall that the predicted class label for the test observation  $x^*$  is based on the sign of  $f(x^*)$ .
- The radial kernel has very local behavior: only nearby training observations have an effect on the class label of a test observation. (Similar to?) KNN

If  $\gamma$  is large  $\Rightarrow$  more local behaviour

$\Rightarrow$  only a few <sup>training</sup> observations are needed to determine  $f(x^*)$   
 $\equiv$  small  $K$

$\gamma$  small  $\equiv$  large  $K$

$\gamma$  controls some bias-variance trade off  
just like  $K$ , so it must be determined using CV.

---

When using CV to determine parameters of a kernel, don't forget 'c' (the budget for margin violations (sum of slack variables  $\epsilon_i$ ))

Cross validate on

$(c, d)$  Kpoly

$(c, \gamma)$  KRF

# Computational Advantage

- What is the advantage of using a kernel rather than simply enlarging the feature space using functions of the original features?
- One advantage is computing without explicitly working in the enlarged feature space.

# Computational Advantage

- This is important because in many applications of SVMs, the enlarged feature space is so large that computations are intractable.
- For some kernels, such as the radial, the feature space is implicit and infinite-dimensional, so we could never do the computations there anyway!

We can use threshold instead

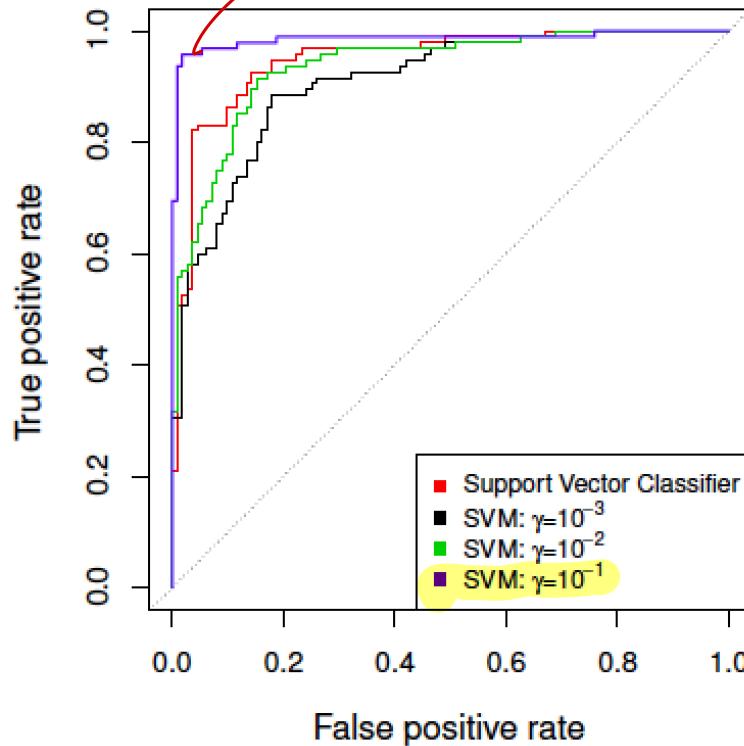
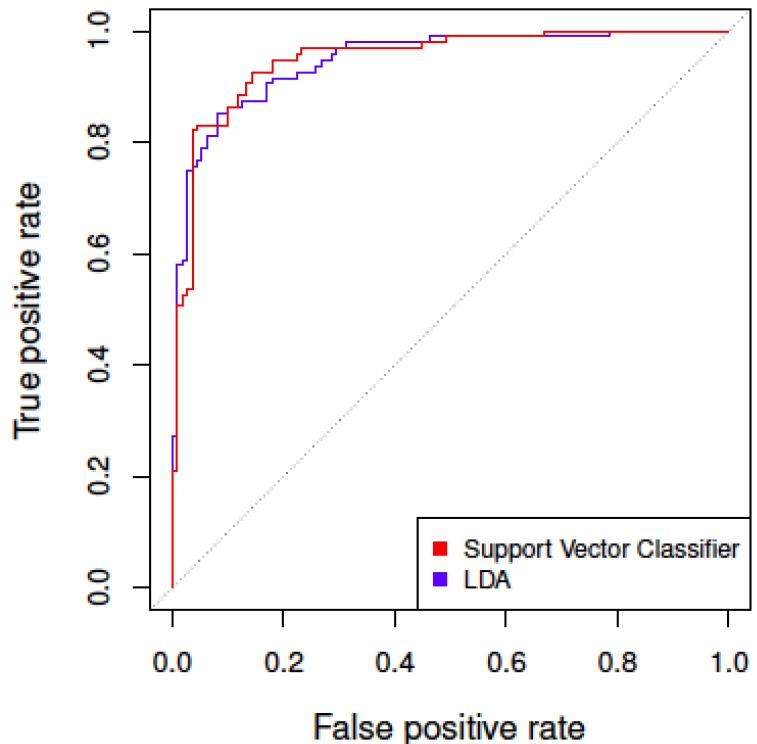
## Example: Heart Data *(train data)*

$$\hat{f}(x) > t$$

$$\hat{f}(x) < t$$

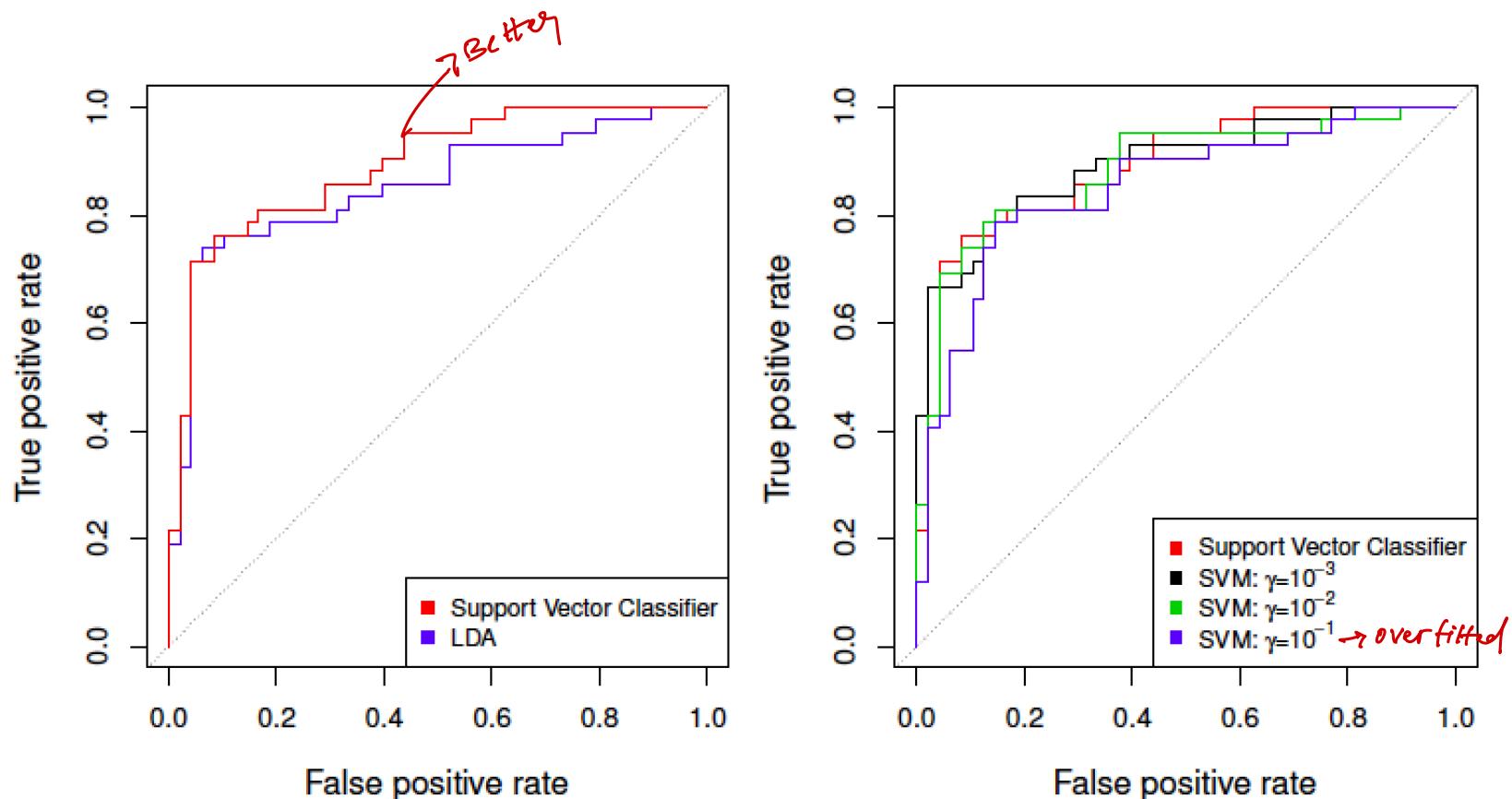
negative  
 $t$

=  
making it  
easier to  
classify  
a point  
in the  
positive  
class.



ROC curve is obtained by changing the threshold 0 to threshold  $t$  in  $\hat{f}(X) > t$ , and recording **false positive** and **true positive** rates as  $t$  varies. Here we see ROC curves on training data.

# Example continued: Heart Test Data



# Multi-Class and Multi-Label Classification Revisited

**Multiclass classification** means a classification task with more than two classes; e.g., classify a set of images of animals which may be horses, birds, or fish.

Multiclass classification makes the assumption that each sample is **assigned to one and only one label**: an animal can be either a horse or a bird but not both at the same time.

# SVMs: more than 2 classes?

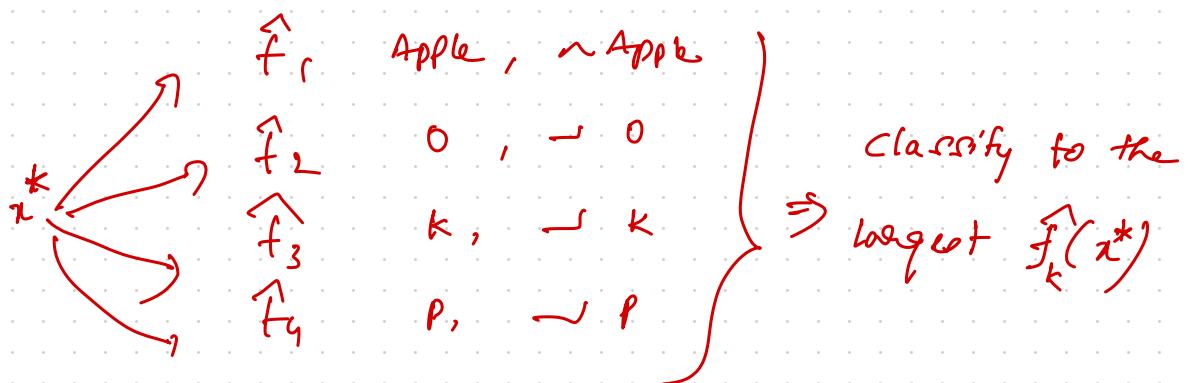
The SVM as defined works for  $K = 2$  classes. What do we do if we have  $K > 2$  classes?

*wrapper methods*

- **OVA** One versus All (The rest). Fit  $K$  different 2-class SVM classifiers  $\hat{f}_k(x)$ ,  $k = 1, \dots, K$ ; each class versus the rest. Classify  $x^*$  to the class for which  $\hat{f}_k(x^*)$  is largest.

We want to classify

Apples, Oranges, Kiwis, Pears



# SVMs: more than 2 classes?

The SVM as defined works for  $K = 2$  classes. What do we do if we have  $K > 2$  classes?

- **OVO** One versus One. Fit all  $\binom{K}{2}$  pairwise classifiers  $\hat{f}_{kl}(x)$ . Classify  $x^*$  to the class that wins the most pairwise competitions.

Which to choose? If  $K$  is not too large, use OVO.

Hello !

$A, O, K, P$        $\binom{4}{2} = 6$     classifying

$x^*$  

$A \Rightarrow O \Rightarrow A \}$     clarity to Apple.

$A \Rightarrow K \Rightarrow A \}$

$O, K \Rightarrow O$

$O, P \Rightarrow P$

$K, P \Rightarrow P$

$A \Rightarrow P \Rightarrow A \}$



# Multi-Class and Multi-Label Problems

**Multilabel classification** assigns to each sample a set of target labels. This can be thought as predicting properties of a data-point that are not mutually exclusive, such as topics that are relevant for a document.

A text might be about any of religion, politics, finance or education at the same time or none of these.

| doc       | $y_1$ | $y_2$ | $y_3$ | $y_4$ |
|-----------|-------|-------|-------|-------|
| religion  | 0     | 0     | 0     | 1     |
| politics  | 1     | 0     | 0     | 1     |
| finance   | 0     | 1     | 0     | 0     |
| education | 1     | 0     | 1     | 0     |
| None      | 0     | 0     | 0     | 0     |

# Multi-Class and Multi-Label Classification Revisited

- Three methods to solve a multi-label classification problem:
  - Problem Transformation
    - Transform multi-label problem into single-label problem(s)
  - Adapted Algorithms
    - adapting conventional algorithms to directly perform multi-label classification
  - Ensemble approaches
    - Combining multiple classifiers

# Multi-Class and Multi-Label Classification Revisited

- Problem Transformation
  - Binary Relevance
  - Classifier Chains
  - Label Powerset

# Multi-Class and Multi-Label Classification Revisited

- **Binary Relevance**
- The simplest technique, which treats each label as a separate binary or multi-class classification problem.
- **Example:** consider a case as shown below. X is the independent feature and Y's are the target variable.

| X                | Y <sub>1</sub> | Y <sub>2</sub> | Y <sub>3</sub> | Y <sub>4</sub> |
|------------------|----------------|----------------|----------------|----------------|
| x <sup>(1)</sup> | 0              | 1              | 1              | 0              |
| x <sup>(2)</sup> | 1              | 0              | 0              | 0              |
| x <sup>(3)</sup> | 0              | 1              | 0              | 0              |
| x <sup>(4)</sup> | 1              | 0              | 0              | 1              |
| x <sup>(5)</sup> | 0              | 0              | 0              | 1              |

# Multi-Class and Multi-Label Classification Revisited

- Binary Relevance
- Solution: The problem is broken into 4 different single class classification problems.  
*label of*

| X                | Y <sub>1</sub> | Y <sub>2</sub> | Y <sub>3</sub> | Y <sub>4</sub> |
|------------------|----------------|----------------|----------------|----------------|
| x <sup>(1)</sup> | 0              | 1              | 1              | 0              |
| x <sup>(2)</sup> | 1              | 0              | 0              | 0              |
| x <sup>(3)</sup> | 0              | 1              | 0              | 0              |
| x <sup>(4)</sup> | 1              | 0              | 0              | 1              |
| x <sup>(5)</sup> | 0              | 0              | 0              | 1              |



| X                | Y <sub>1</sub> |
|------------------|----------------|
| x <sup>(1)</sup> | 0              |
| x <sup>(2)</sup> | 1              |
| x <sup>(3)</sup> | 0              |
| x <sup>(4)</sup> | 1              |
| x <sup>(5)</sup> | 0              |

| X                | Y <sub>2</sub> |
|------------------|----------------|
| x <sup>(1)</sup> | 1              |
| x <sup>(2)</sup> | 0              |
| x <sup>(3)</sup> | 1              |
| x <sup>(4)</sup> | 0              |
| x <sup>(5)</sup> | 0              |

| X                | Y <sub>3</sub> |
|------------------|----------------|
| x <sup>(1)</sup> | 1              |
| x <sup>(2)</sup> | 0              |
| x <sup>(3)</sup> | 0              |
| x <sup>(4)</sup> | 0              |
| x <sup>(5)</sup> | 0              |

| X                | Y <sub>4</sub> |
|------------------|----------------|
| x <sup>(1)</sup> | 0              |
| x <sup>(2)</sup> | 0              |
| x <sup>(3)</sup> | 0              |
| x <sup>(4)</sup> | 1              |
| x <sup>(5)</sup> | 1              |

# Multi-Class and Multi-Label Classification Revisited

- **Binary Relevance**
- **Solution:** For a new data point  $\mathbf{x}^*$ , each of the labels  $Y_1, \dots, Y_4$  is predicted separately.
- **Note1:** Any classifier (e.g. Naïve Baye's, Logistic Regression, and SVM) can be used for predicting each label
- Note 2: Each label may give rise to a binary or multi-class classification problem.



| X                  | $Y_1$ | $Y_2$ | $Y_3$ | $Y_4$ |
|--------------------|-------|-------|-------|-------|
| $\mathbf{x}^{(1)}$ | 0     | 1     | 1     | 0     |
| $\mathbf{x}^{(2)}$ | 1     | 0     | 0     | 0     |
| $\mathbf{x}^{(3)}$ | 0     | 1     | 0     | 0     |
| $\mathbf{x}^{(4)}$ | 1     | 0     | 0     | 1     |
| $\mathbf{x}^{(5)}$ | 0     | 0     | 0     | 1     |

| X                  | $Y_1$ |
|--------------------|-------|
| $\mathbf{x}^{(1)}$ | 0     |
| $\mathbf{x}^{(2)}$ | 1     |
| $\mathbf{x}^{(3)}$ | 0     |
| $\mathbf{x}^{(4)}$ | 1     |
| $\mathbf{x}^{(5)}$ | 0     |

| X                  | $Y_2$ |
|--------------------|-------|
| $\mathbf{x}^{(1)}$ | 1     |
| $\mathbf{x}^{(2)}$ | 0     |
| $\mathbf{x}^{(3)}$ | 1     |
| $\mathbf{x}^{(4)}$ | 0     |
| $\mathbf{x}^{(5)}$ | 0     |

| X                  | $Y_3$ |
|--------------------|-------|
| $\mathbf{x}^{(1)}$ | 1     |
| $\mathbf{x}^{(2)}$ | 0     |
| $\mathbf{x}^{(3)}$ | 0     |
| $\mathbf{x}^{(4)}$ | 0     |
| $\mathbf{x}^{(5)}$ | 0     |

| X                  | $Y_4$ |
|--------------------|-------|
| $\mathbf{x}^{(1)}$ | 0     |
| $\mathbf{x}^{(2)}$ | 0     |
| $\mathbf{x}^{(3)}$ | 0     |
| $\mathbf{x}^{(4)}$ | 1     |
| $\mathbf{x}^{(5)}$ | 1     |

# Multi-Class and Multi-Label Classification Revisited

- **Classifier Chains**
- In this approach, the first classifier is trained just on the input data and then each next classifier is trained on the input space and all the previous classifiers in the chain.

# Multi-Class and Multi-Label Classification Revisited

- **Example:**  $X$  as the input space and  $Y$ 's as the labels.

| <b>X</b>  | <b>y1</b> | <b>y2</b> | <b>y3</b> | <b>y4</b> |
|-----------|-----------|-----------|-----------|-----------|
| <b>x1</b> | 0         | 1         | 1         | 0         |
| <b>x2</b> | 1         | 0         | 0         | 0         |
| <b>x3</b> | 0         | 1         | 0         | 0         |

# Multi-Class and Multi-Label Classification Revisited

- **Solution:** In classifier chains, this problem would be transformed into 4 different single label problems, just like shown below. Here yellow colored is the input space and the white part represent the target variable.

*Does order matter? Don't know.  
cross validate.)*

| X  | y1 |
|----|----|
| x1 | 0  |
| x2 | 1  |
| x3 | 0  |

Classifier 1

| X  | y1 | y2 |
|----|----|----|
| x1 | 0  | 1  |
| x2 | 1  | 0  |
| x3 | 0  | 1  |

Classifier 2

| X  | y1 | y2 | y3 |
|----|----|----|----|
| x1 | 0  | 1  | 1  |
| x2 | 1  | 0  | 0  |
| x3 | 0  | 1  | 0  |

Classifier 3

| X  | y1 | y2 | y3 | y4 |
|----|----|----|----|----|
| x1 | 0  | 1  | 1  | 0  |
| x2 | 1  | 0  | 0  | 0  |
| x3 | 0  | 1  | 0  | 0  |

Classifier 4

*While predicting also follow the chain*



# Multi-Class and Multi-Label Classification Revisited

- **Note:** This is quite similar to binary relevance, the only difference being it forms chains in order to preserve label correlation.

| X  | y1 |
|----|----|
| x1 | 0  |
| x2 | 1  |
| x3 | 0  |

Classifier 1

| X  | y1 | y2 |
|----|----|----|
| x1 | 0  | 1  |
| x2 | 1  | 0  |
| x3 | 0  | 1  |

Classifier 2

| X  | y1 | y2 | y3 |
|----|----|----|----|
| x1 | 0  | 1  | 1  |
| x2 | 1  | 0  | 0  |
| x3 | 0  | 1  | 0  |

Classifier 3

| X  | y1 | y2 | y3 | y4 |
|----|----|----|----|----|
| x1 | 0  | 1  | 1  | 0  |
| x2 | 1  | 0  | 0  | 0  |
| x3 | 0  | 1  | 0  | 0  |

Classifier 4

# Multi-Class and Multi-Label Classification Revisited

- [Label Powerset](#)
- This method transforms the problem into a multi-class problem with one multi-class classifier trained on all unique label combinations found in the training data.

# Multi-Class and Multi-Label Classification Revisited

- **Example:**  $X$  are features,  $Y_1, \dots, Y_4$  are labels

| <b>X</b>  | <b>y1</b> | <b>y2</b> | <b>y3</b> | <b>y4</b> | <i>y</i> |
|-----------|-----------|-----------|-----------|-----------|----------|
| <b>x1</b> | 0         | 1         | 1         | 0         | c1       |
| <b>x2</b> | 1         | 0         | 0         | 0         | c2       |
| <b>x3</b> | 0         | 1         | 0         | 0         | c3       |
| <b>x4</b> | 0         | 1         | 1         | 0         | c2       |
| <b>x5</b> | 1         | 1         | 1         | 1         | c4       |
| <b>x6</b> | 0         | 1         | 0         | 0         | c3       |

# Multi-Class and Multi-Label Classification Revisited

- **Solution:**  $x_1$  and  $x_4$  have the same labels, similarly,  $x_3$  and  $x_6$  have the same set of labels. So, label powerset transforms this problem into a single multi-class problem as shown below.

The diagram illustrates the transformation of a multi-label classification matrix into a single-class multi-label matrix. On the left, a 6x5 matrix represents multiple labels for each sample. The columns are labeled y1 through y4. The rows are labeled x1 through x6. The values in the matrix indicate which labels are present for each sample. An orange arrow points from this matrix to a smaller 6x2 matrix on the right, which represents the transformed single-class multi-label matrix. The columns are labeled x and y1. The rows are labeled x1 through x6. The values in the second column (y1) represent the class index for each sample based on its original labels.

| X  | y1 | y2 | y3 | y4 |
|----|----|----|----|----|
| x1 | 0  | 1  | 1  | 0  |
| x2 | 1  | 0  | 0  | 0  |
| x3 | 0  | 1  | 0  | 0  |
| x4 | 0  | 1  | 1  | 0  |
| x5 | 1  | 1  | 1  | 1  |
| x6 | 0  | 1  | 0  | 0  |

| X  | y1 |
|----|----|
| x1 | 1  |
| x2 | 2  |
| x3 | 3  |
| x4 | 1  |
| x5 | 4  |
| x6 | 3  |

What if the label isn't in the test set?

# Multi-Class and Multi-Label Classification Revisited

- **Solution:** The label powerset method has given a unique class to every possible label combination that is present in the training set.

The diagram illustrates a transformation from a multi-label classification dataset to a single-class dataset. On the left, a table shows six data points (x1 to x6) with binary labels for four categories (y1 to y4). An orange arrow points to the right, where another table shows the same six data points, but each is now assigned a unique integer value (1 through 6) based on its label combination. This mapping represents the label powerset method, where each unique combination of labels is assigned a distinct class index.

| X  | y1 | y2 | y3 | y4 |
|----|----|----|----|----|
| x1 | 0  | 1  | 1  | 0  |
| x2 | 1  | 0  | 0  | 0  |
| x3 | 0  | 1  | 0  | 0  |
| x4 | 0  | 1  | 1  | 0  |
| x5 | 1  | 1  | 1  | 1  |
| x6 | 0  | 1  | 0  | 0  |

| X  | y1 |
|----|----|
| x1 | 1  |
| x2 | 2  |
| x3 | 3  |
| x4 | 1  |
| x5 | 4  |
| x6 | 3  |

It is possible that there only a few or no data points with a specific label combination in the training set and we may encounter such label combinations in the test time.

This is especially a problem when we have too many labels giving rise to too many classes.

e.g.: 10 labels  $2^{10}$  combinations

# Multi-Class and Multi-Label Classification Revisited

- **Adapted Algorithms**
  - The most famous adapted algorithm is Multilabel kNN (MLkNN)

# Multi-Class and Multi-Label Classification Revisited

- ML-kNN uses the kNN algorithm independently for each label  $\ell$ .
- It finds the  $k$  nearest examples to the test instance and considers those that are labeled at least with  $\ell$  as positive and the rest as negative.

# Multi-Class and Multi-Label Classification Revisited

- What mainly differentiates this method from other binary relevance (BR) methods is the use of prior probabilities. ML-kNN can also rank labels.

# Multi-Class and Multi-Label Classification Revisited

- **Adapted Algorithms**
  - Decision Trees can be modified to perform multi label classification.
  - The main modification is in calculating purities for each region.

# Multi-Class and Multi-Label Classification Revisited

- Ensemble Algorithms
  - AdaBoost.MH and AdaBoost.MR

# Evaluation Metrics

- One cannot simply use our normal metrics to calculate the accuracy of the predictions.
- **Accuracy score/ Exact match metric:** This function calculates subset accuracy meaning the predicted set of labels should **exactly match** with the true set of labels.

# Evaluation Metrics

- **Hamming Loss:** The fraction of the wrong labels to the total number of labels, which is:

$$\frac{1}{NL} \sum_{i=1}^N \sum_{j=1}^L I(\hat{y}_{ij} \neq y_{ij})$$

mismatch

L

|            |            |            |            |            |            |            |
|------------|------------|------------|------------|------------|------------|------------|
| Blue       | Light Gray |
| Light Gray |            |            |            |            |            |            |
| Light Gray |            |            |            |            |            |            |
| Light Gray |            |            |            |            |            |            |

$\hat{y}_{ij} \neq y_{ij}$

$\hat{y}_{ij} = y_{ij}$

N

# Multi-Class and Multi-Label Classification Revisited

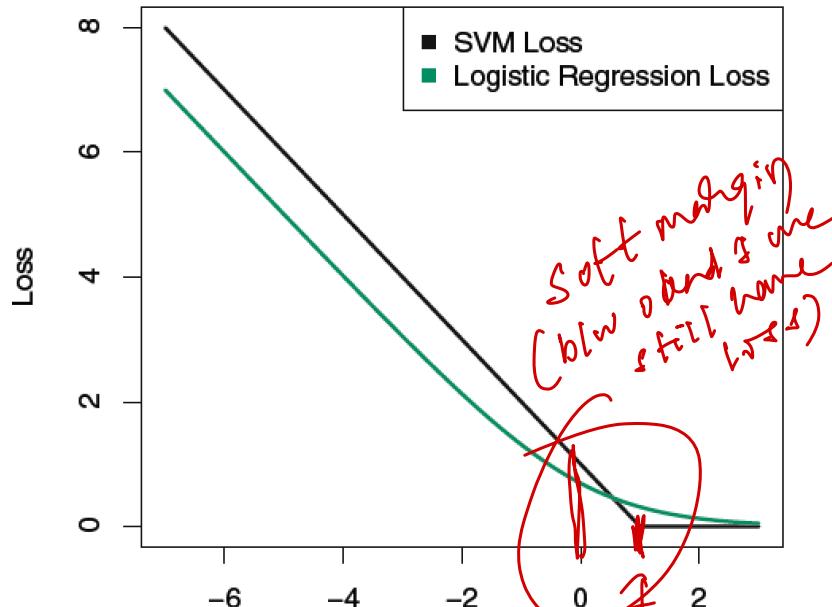
- For an interesting overview, see:
- <https://arxiv.org/pdf/1703.08991.pdf>

# SVC versus Logistic Regression?

With  $f(X) = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p$  one can rephrase support-vector classifier optimization as

$$\underset{\beta_0, \beta_1, \dots, \beta_p}{\text{minimize}} \left\{ \sum_{i=1}^n \max [0, 1 - y_i f(x_i)] + \lambda \sum_{j=1}^p \beta_j^2 \right\}$$

*L<sub>2</sub> regularizing (ridge regression)*



This has the form *loss plus penalty*.

The loss is known as the *hinge loss*.

Very similar to “loss” in logistic regression (negative log-likelihood).

$$\text{loss} = \max(0, 1 - y_i f(x_i))$$

$$P(y_i = -1 | x = n_i) + P(y_i = 1 | x = n_i) = 1$$

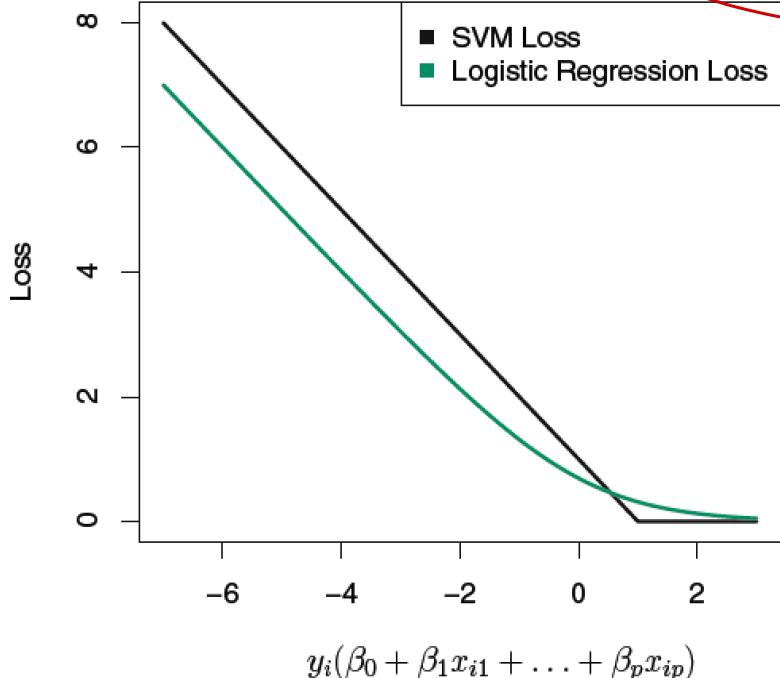
# SVC versus Logistic Regression?

Similarly, one can rewrite logistic regression when Y is -1 or 1 (instead of 0,1) as

$0 \leq 1$

$$P(Y = y_i | X = x_i) = \frac{1}{1 + \exp(-y_i f(x_i))}$$

$\leq 1$



The negative log likelihood loss function is:

$$-\log[P(Y = y_i | X = x_i)] \\ = \log[(1 + \exp(-y_i f(x_i)))]$$

$$NLL = -\sum_{i=1}^n \log P(Y = y_i | X = x_i)$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$$

# Loss Plus Penalty: L1 Regularization

$$\underset{\beta_0, \beta_1, \dots, \beta_p}{\text{minimize}} \left\{ \sum_{i=1}^n \max [0, 1 - y_i f(x_i)] + \lambda \sum_{j=1}^p \beta_j^2 \right\}$$

The above optimization problem has the form of *loss plus penalty*.

Note that the original penalty in SVC is a ridge penalty.

One can change the penalty with L1 penalty and perform variable selection along with Support Vector Classification.

*hinge loss = L1 loss*

*Squared hinge  
=  $\lambda^2$  loss*

# Loss Plus Penalty: L1 Regularization

$$\underset{\beta_0, \beta_1, \dots, \beta_p}{\text{minimize}} \left\{ \sum_{i=1}^n \max [0, 1 - y_i f(x_i)] + \lambda \sum_{j=1}^p \beta_j^2 \right\}$$

Because the hinge loss function is not differentiable, the solution paths for different values of  $\lambda$  may have jumps.

Therefore, some authors replace the hinge loss with a differentiable squared hinge loss.

Different combinations of hinge loss and squared hinge loss with L1 and L2 penalty result in various versions of SVM.

$$\max(0, 1 - y_i f(x_i))^2$$

Loss

Penalty

$\ell_1$  (hinge)

$\ell_1$

$\ell_2$  (squared hinge)

$\ell_2$

NLL (logistic reg)

elastic net

Different combo can be used

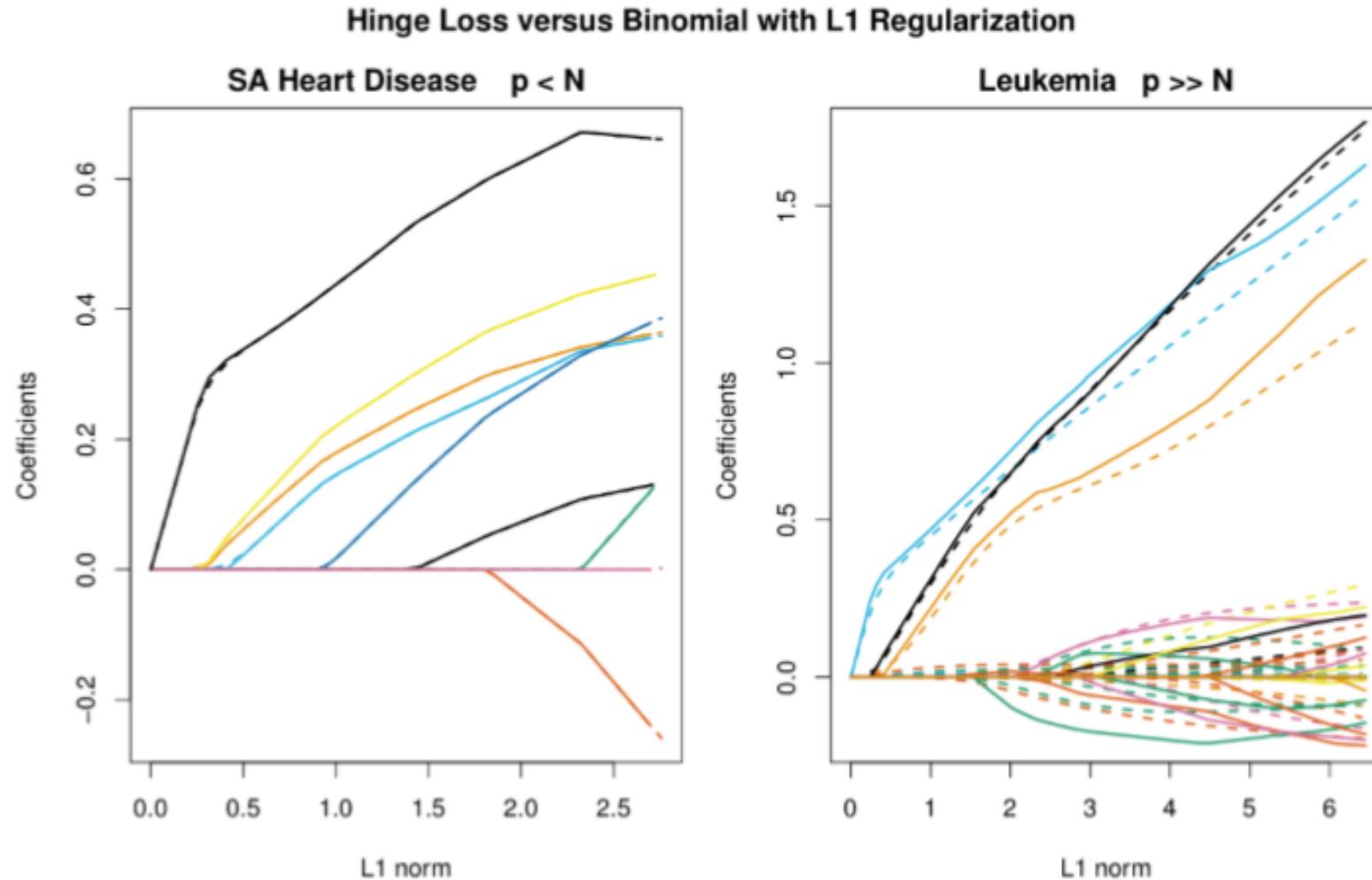
libSVC (< package)

---

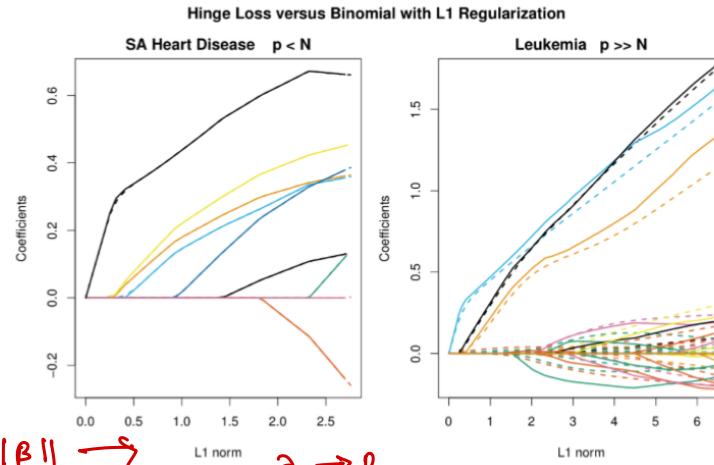
Kernel SVMs

libSVM (It also has linear kernel,  
but libSVC is highly recommended)

# Comparison of L1 Regularized SVM and Logistic Regression



# Details of Previous Figure



A comparison of the coefficient paths for the L1- SVM versus L1 logistic regression on two examples.

In the left we have the South African heart disease data ( $N = 462$  and  $p = 9$ ), and on the right the Leukemia data ( $N = 38$  and  $p = 6087$ ). The dashed lines are the SVM coefficients, the solid lines logistic regression. The similarity is striking in the left example, and strong in the right.

# Which to use: SVM or Logistic Regression

- When classes are (nearly) separable, SVM does better than LR. So does LDA.

# Which to use: SVM or Logistic Regression

- SVMs are popular in high-dimensional classification problems with  $p \gg n$ , since the computations are  $O(pn^2)$  for both linear and nonlinear kernels. (Some references say between  $O(n^2)$  and  $O(n^3)$ ).
- Additional efficiencies can be realized for linear SVMs (Shalev-Shwartz, Singer and Srebro 2007).

Circles include these.

# Which to use: SVM or Logistic Regression

- When classes are (nearly) separable, SVM does better than LR. So does LDA.
- When not, LR (**with ridge penalty**) and SVM very similar.

# Which one to use: SVM or Logistic Regression

- If you wish to estimate probabilities, LR is the choice.
- However, one can estimate probabilities from distances of data points from the SVC hyperplane:

$$P(y = y_i | x = x_i) = \frac{1}{1 + \exp(-y_i f(x_i))}$$

# Which one to use: SVM or Logistic Regression

- For nonlinear boundaries, kernel SVMs are popular, as we will see. Can use kernels with LR and Bayesian LDA as well, but computations are more expensive.
- Also, Kernels try to find a feature space in which decision boundaries are linear. That's not commensurate with characteristics of LR.

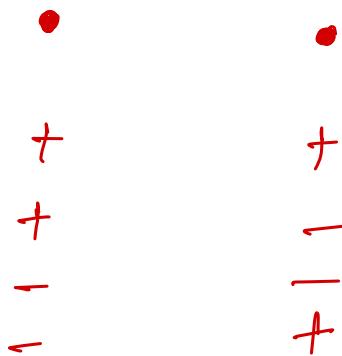
# The Vapnik-Chervonenkis Dimension

- The **VC dimension** is a measure of the **capacity** (complexity, expressive power, richness, or flexibility) of a space of functions that **can be learned** by a classification algorithm.

# The Vapnik-Chervonenkis Dimension

- A classification model  $f$  with some parameter vector  $\beta$  is said to *shatter* a set of data points  $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$  if for all assignments of labels to those points, there exists a  $\beta$  such that the model  $f$  makes no errors when evaluating that set of data points.

Binary  
Labels



These two points can be shattered i.e. can be classified in all the cases.

# The Vapnik-Chervonenkis Dimension

- The **VC dimension** of a model  $f$  is the maximum number of points that can be arranged so that  $f$  shatters them. (it's 3 in 2D)

1 data point (✓)

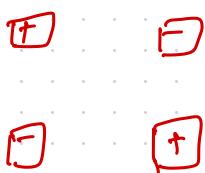
2 data points (✓)      If two points overlap • • .  
                                Re Arrange them ↗

3 data points



□ □ □  
what it aligned is a line  
(Arrange them differently)

4 data points



# The Vapnik-Chervonenkis Dimension

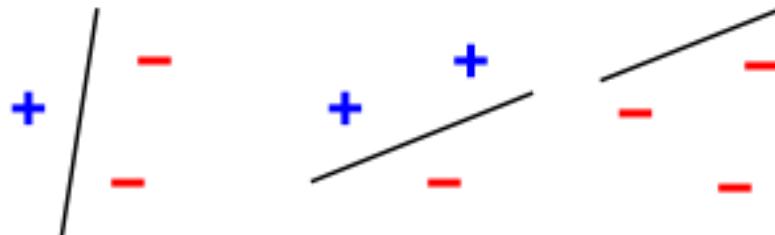
- Example:  $f$  is a straight line as a classification model on points in a two-dimensional plane (this is the model used by a perceptron).
- The line should separate positive data points from negative data points.

# The Vapnik-Chervonenkis Dimension

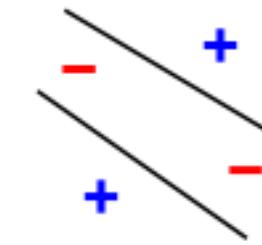
- There exist sets of 3 points that can indeed be shattered using this model (any 3 points that are not collinear can be shattered). However, no set of 4 points can be shattered. Thus, the VC dimension of this particular classifier is 3.

# The Vapnik-Chervonenkis Dimension

- Note, only  $3$  of the  $2^3 = 8$  possible label assignments are shown for the three points.



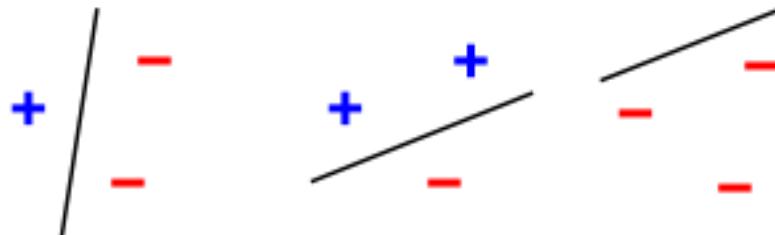
**3 points shattered**



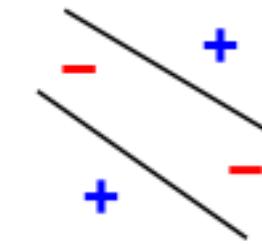
**4 points impossible**

# The Vapnik-Chervonenkis Dimension

- Note, only  $3$  of the  $2^3 = 8$  possible label assignments are shown for the three points.



**3 points shattered**



**4 points impossible**

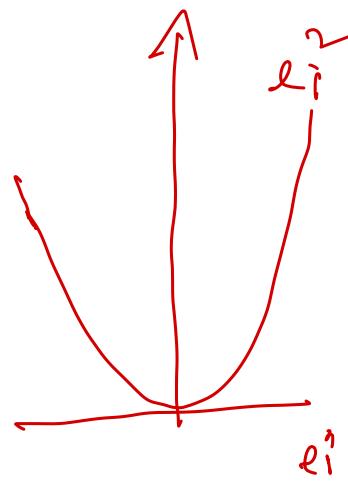
# The Vapnik-Chervonenkis Dimension

- The VC Dimension of affine classifiers of the form  $f(\mathbf{x}) = \boldsymbol{\beta}^T \mathbf{x} + \beta_0$ ,  $\boldsymbol{\beta} \in \mathbb{R}^p$  is  $p+1$ .
- This includes SVC, a linear SVM.
- The VC Dimension of an SVM equipped with an RBF kernel is infinite.

this is only for training data like kNN when  $k=1$

# Support Vector Regression

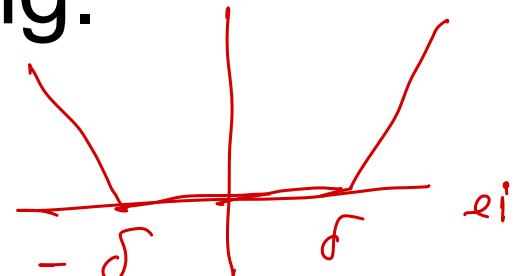
- There is an extension of the SVM for regression, called support vector regression (SVR) .
- We saw that least squares regression seeks coefficients  $\beta_0, \beta_1, \dots, \beta_p$  such that the sum of squared residuals is as small as possible.



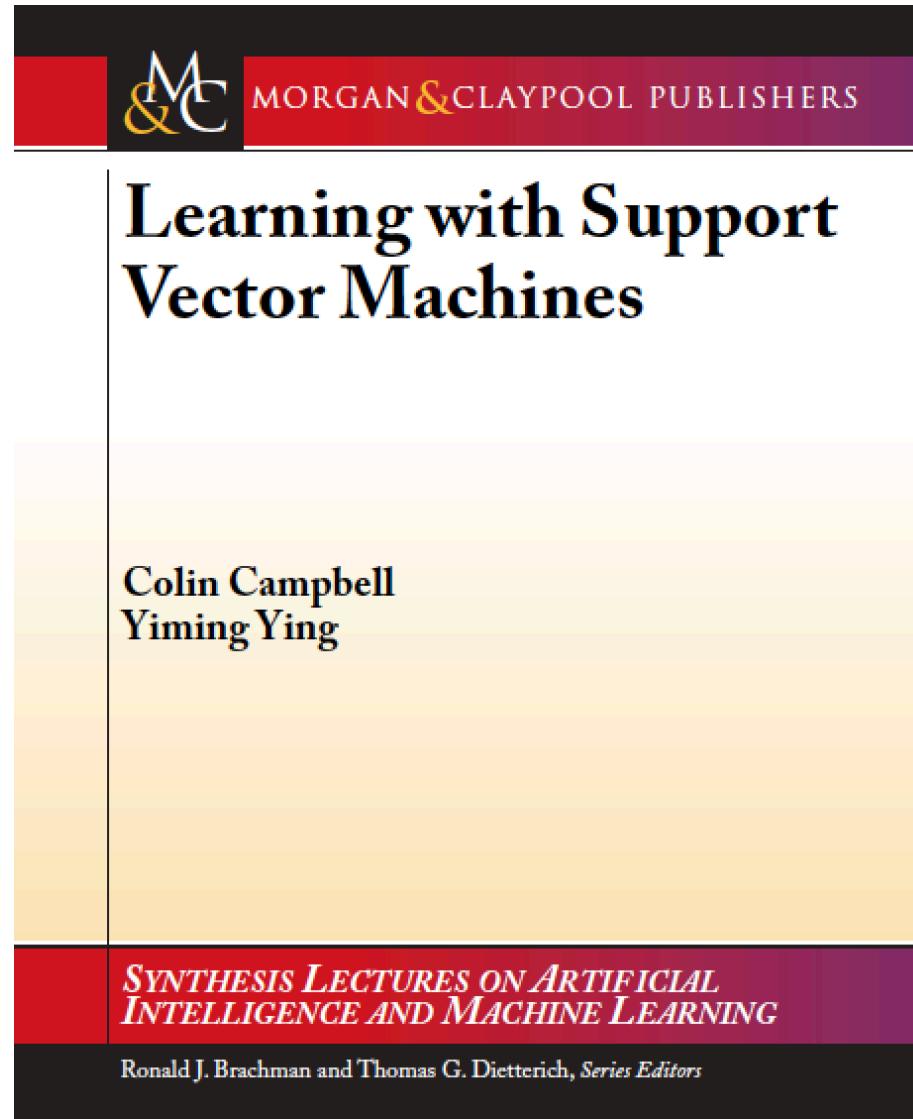
$$\sum_{i=1}^n e_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

# Support Vector Regression

- Support vector regression instead seeks coefficients that minimize a different type of loss, where only residuals larger in absolute value than some positive constant contribute to the loss function.
- This is an extension of the margin used in support vector classifiers to the regression setting.



# SVM Learning



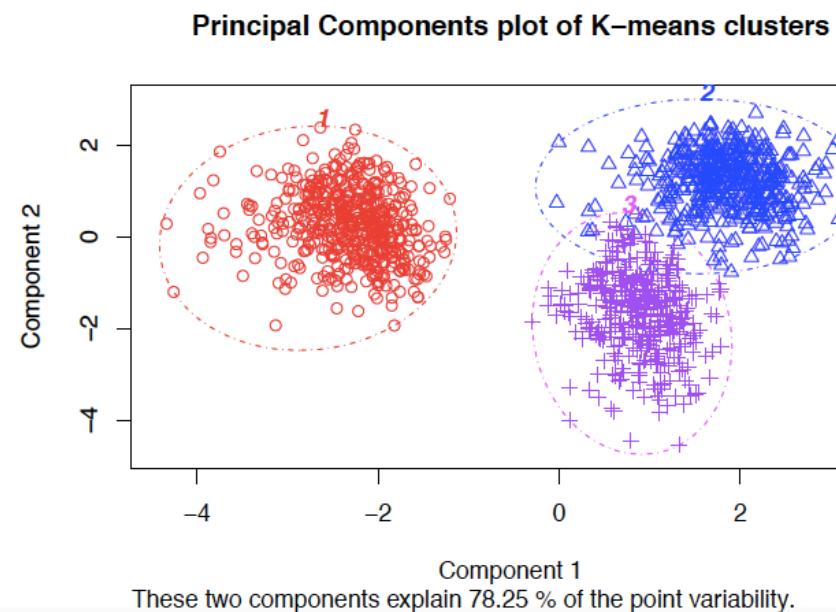
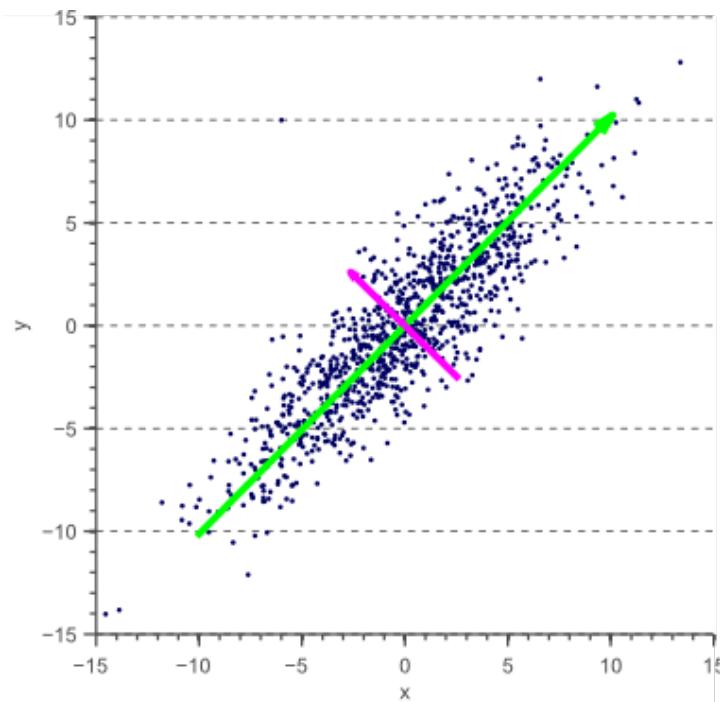
# **DSCI 552, Machine Learning for Data Science**

University of Southern California

M. R. Rajati, PhD

# Lesson 8

## Unsupervised Learning



# Unsupervised Learning

*Unsupervised vs Supervised Learning:*

- Most of this course focuses on *supervised learning* methods such as regression and classification.
- In that setting we observe both a set of features  $X_1, X_2, \dots, X_p$  for each object, as well as a response or outcome variable  $Y$ . The goal is then to predict  $Y$  using  $X_1, X_2, \dots, X_p$ .

# Unsupervised Learning

*Unsupervised vs Supervised Learning:*

- Here we instead focus on *unsupervised learning*, where we observe only the features  $X_1, X_2, \dots, X_p$ .
- We are not interested in prediction, because we do not have an associated response variable  $Y$ .

# The Goals of Unsupervised Learning

- The goal is to discover interesting things about the measurements: is there an informative way to visualize the data? Can we discover subgroups among the variables or among the observations?
- We discuss two methods:
  - *principal components analysis*, a tool used for data visualization or data pre-processing before supervised techniques are applied, and
  - *clustering*, a broad class of methods for discovering unknown **subgroups** in data.

# The Challenge of Unsupervised Learning

- Unsupervised learning is more subjective than supervised learning, as there is no simple goal for the analysis, such as prediction of a response.
- But techniques for unsupervised learning are of growing importance in a number of fields:
  - subgroups of breast cancer patients grouped by their gene expression measurements,
  - groups of shoppers characterized by their browsing and purchase histories,
  - movies grouped by the ratings assigned by movie viewers.

# Another advantage

- It is often easier to obtain *unlabeled data* — from a lab instrument or a computer — than *labeled data*, which can require human intervention.
- For example it is difficult to automatically assess the overall sentiment of a movie review: is it favorable or not?

# Clustering

- *Clustering* refers to a very broad set of techniques for finding *subgroups*, or *clusters*, in a data set.
- We seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other.

# Clustering

- We must define what it means for two or more observations to be *similar* or *different*.
- Indeed, this is often a domain-specific consideration that must be made based on knowledge of the data being studied.

# Clustering for Market Segmentation

- Suppose we have access to a large number of measurements (e.g. median household income, occupation, distance from nearest urban area, and so forth) for a large number of people.

# Clustering for Market Segmentation

- Our goal is to perform *market segmentation* by identifying subgroups of people who might be more receptive to a particular form of advertising, or more likely to purchase a particular product.

# Clustering for Market Segmentation

- The task of performing market segmentation amounts to clustering the people in the data set.

# Two clustering methods

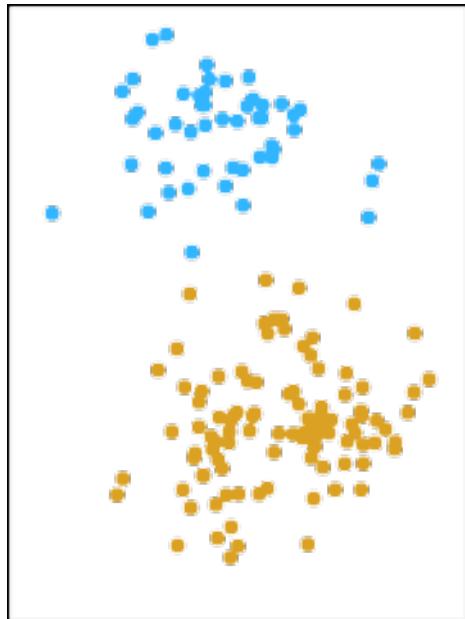
- In *K-means clustering*, we seek to partition the observations into a pre-specified number of clusters.

# Two clustering methods

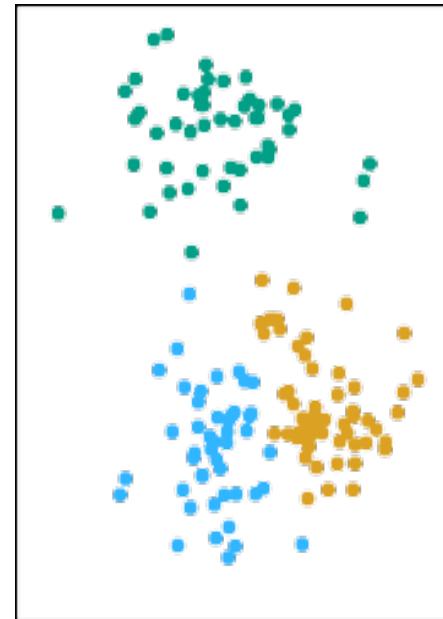
- In *hierarchical clustering*, we do not know in advance how many clusters we want; in fact, we end up with a tree-like visual representation of the observations, called a *dendrogram*, that allows us to view at once the clusterings obtained for each possible number of clusters, from 1 to  $n$ .

# $K$ -means clustering

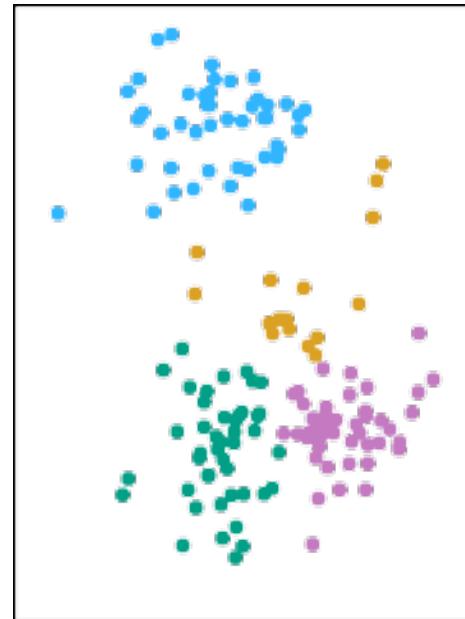
$K=2$



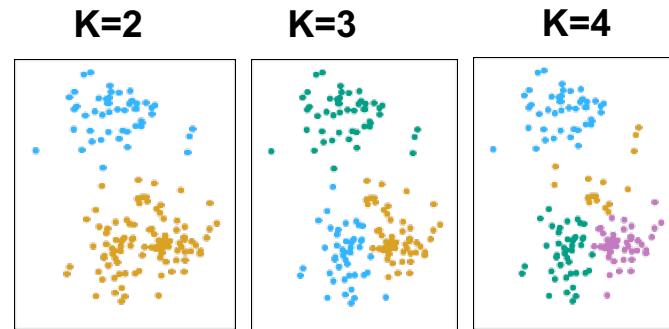
$K=3$



$K=4$



## $K$ -means clustering



A simulated data set with 150 observations in 2-dimensional space. Panels show the results of applying K-means clustering with different values of  $K$ , the number of clusters. The color of each observation indicates the cluster to which it was assigned using the K-means clustering algorithm. Note that there is no ordering of the clusters, so the cluster coloring is arbitrary. These cluster labels were not used in clustering; instead, they are the outputs of the clustering procedure.

# Details of $K$ -means clustering

Let  $C_1, \dots, C_K$  denote sets containing the indices of the observations in each cluster. These sets satisfy two properties:

1.  $C_1 \cup C_2 \cup \dots \cup C_K = \{1, \dots, n\}$ . In other words, each observation belongs to at least one of the  $K$  clusters. *Collectively exhaust the clusters.*
2.  $C_k \cap C_{k'} = \emptyset$  for all distinct  $k$  and  $k'$ . In other words, the clusters are non-overlapping: no observation belongs to more than one cluster.

For instance, if the  $i^{\text{th}}$  observation is in the  $k^{\text{th}}$  cluster, then  $i \in C_k$ .

# Details of $K$ -means clustering: continued

- The idea behind  $K$ -means clustering is that a *good* clustering is one for which the *within-cluster variation* is as small as possible.

# Details of $K$ -means clustering: continued

- The within-cluster variation for cluster  $C_k$  is a measure  $\text{WCV}(C_k)$  of the amount by which the observations within a cluster differ from each other.
- Hence we want to solve the problem

$$\underset{C_1, \dots, C_K}{\text{minimize}} \left\{ \sum_{k=1}^K \text{WCV}(C_k) \right\}$$

*Set ahead of time* 

# Details of $K$ -means clustering: continued

- In words, this formula says that we want to partition the observations into  $K$  clusters such that the total within-cluster variation, summed over all  $K$  clusters, is as small as possible.

# How to define within-cluster variation?

- Typically we use Euclidean distance

$$\text{WCV}(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^p (x_{ij} - \bar{x}_{i'})^2$$

$\|\underline{x}_i - \underline{x}_{i'}\|_2^2$

where  $|C_k|$  denotes the number of observations in the  $k^{\text{th}}$  cluster.

# How to define within-cluster variation?

- Combining the above equations gives the optimization problem that defines K-means clustering,

$$\|\underline{x}_i - \underline{z}_i\|_F^2$$

$$\underset{C_1, \dots, C_K}{\text{minimize}} \left\{ \sum_{k=1}^K \frac{1}{|C_k|} \sum_{i, i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2 \right\}$$

*wcv<sup>o</sup>(C<sub>k</sub>)*

Lloyd's algorithm

# K-Means Clustering Algorithm

1. Randomly assign a number, from 1 to  $K$ , to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
  1. For each of the  $K$  clusters, compute the cluster *centroid*.  
The  $k^{\text{th}}$  cluster centroid is the vector of the  $p$  feature means for the observations in the  $k^{\text{th}}$  cluster.
  2. Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

Reclutely

(Lloyd's alg)  
K-means is a version of the EM  
(We need 'n' data inputation)

Iterative solution of Logistic regression

- BP + SGD
- Active learning / Semi-supervised
- Q learning

Venkatesh Murthy  
Hello

# Properties of the Algorithm

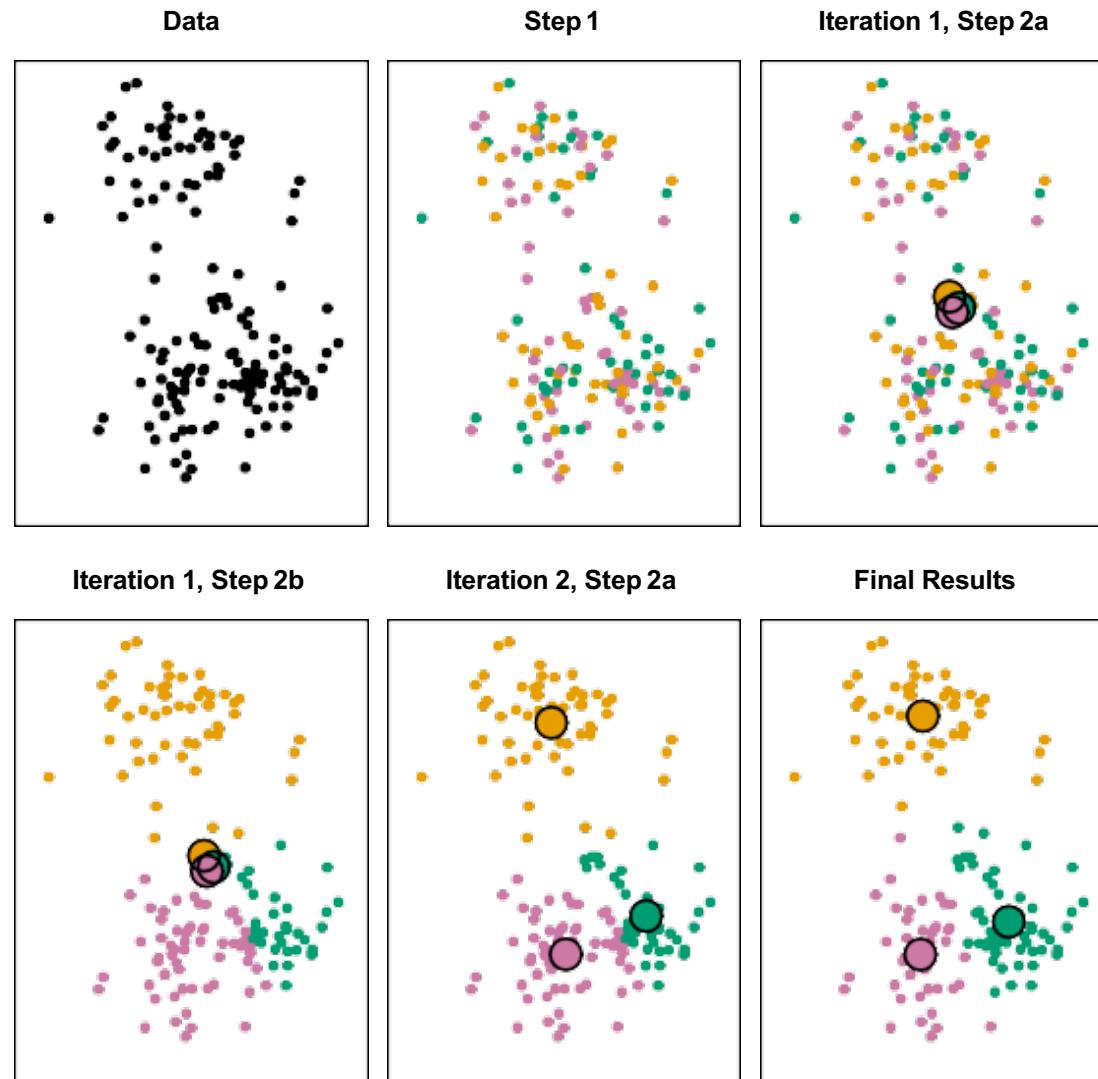
- This algorithm is guaranteed to decrease the value of the objective function at each step.  
*Why?* Note that

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^p (x_{ij} - \bar{x}_{kj})^2$$

where  $\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$  is the mean for feature  $j$  in cluster  $C_k$ .

- however it is not guaranteed to give the global minimum. *Why not?*

# Example



# Details of Previous Figure

The progress of the K-means algorithm with  $K=3$ .

- *Top left:* The observations are shown.
- *Top center:* In Step 1 of the algorithm, each observation is randomly assigned to a cluster.
- *Top right:* In Step 2(a), the cluster centroids are computed. These are shown as large colored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.

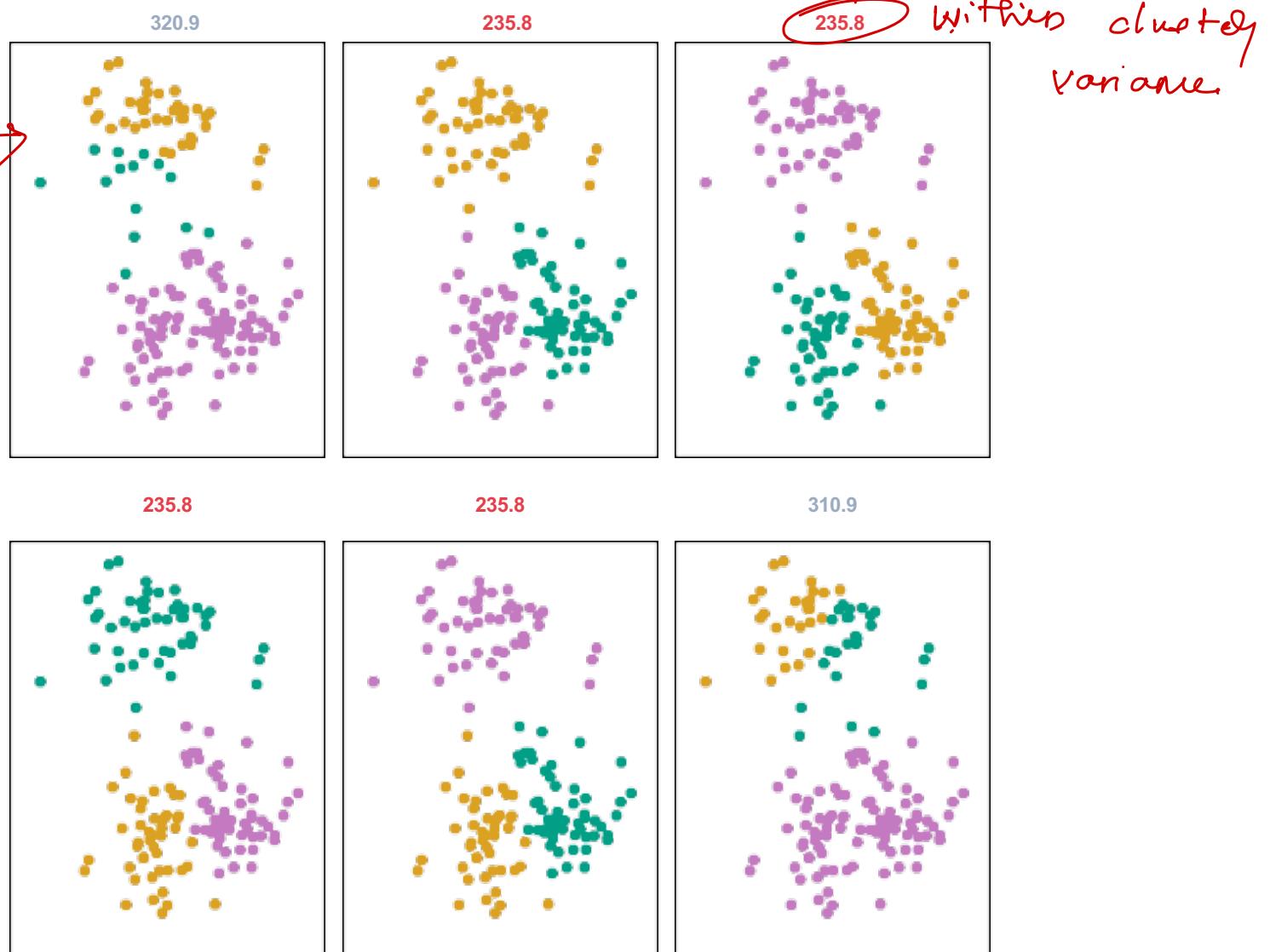
# Details of Previous Figure

The progress of the K-means algorithm with  $K=3$ .

- *Bottom left:* In Step 2(b), each observation is assigned to the nearest centroid.
- *Bottom center:* Step 2(a) is once again performed, leading to new cluster centroids.
- *Bottom right:* The results obtained after 10 iterations.

# Example: different starting values

Different  
clusters  
(optimal isn't  
guaranteed)



To deal with the problem of premature clusters, one can initialize the clusters randomly multiple times and run the k-means algo and pick the cluster that has lowest <sup>within</sup> cluster variation

that has lowest within cluster variation (nice handwriting)

# Details of Previous Figure

$K$ -means clustering performed six times on the data from previous figure with  $K = 3$ , each time with a different random assignment of the observations in Step 1 of the  $K$ -means algorithm.

# Details of Previous Figure

Above each plot is the value of the objective

$$\underset{C_1, \dots, C_K}{\text{minimize}} \left\{ \sum_{k=1}^K \frac{1}{|C_k|} \sum_{i, i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2 \right\}$$

Three different local optima were obtained, one of which resulted in a smaller value of the objective and provides better separation between the clusters.

Those labeled in red all achieved the same best solution, with an objective value of 235.8

# K-Medoids Clustering

- Similar to K-means, but in each step we do not compute the centroid of each cluster.
- We compute the **medoid**, which is a data point that has the smallest average dissimilarity with other data points in the cluster. *distance*
- This way you limit the center of your cluster to be one of your data points.

Sometimes, we need a good "representative" for each cluster to interpret it.

e.g.: Image clustering

Sometimes, calculating the centroid is impossible / meaningless

e.g.: When we have categorical features.

$$\begin{bmatrix} \text{blue} \\ 1 \\ 2 \end{bmatrix} \quad \begin{bmatrix} \text{black} \\ 0 \\ 3 \end{bmatrix} \quad \begin{bmatrix} \text{purple} \\ ; \\ ; \end{bmatrix}$$

Centroid?

# Hierarchical Clustering

- $K$ -means clustering requires us to pre-specify the number of clusters  $K$ . This can be a disadvantage (later we discuss strategies for choosing  $K$ )
- *Hierarchical clustering* is an alternative approach which does not require that we commit to a particular choice of  $K$ .

# Hierarchical Clustering

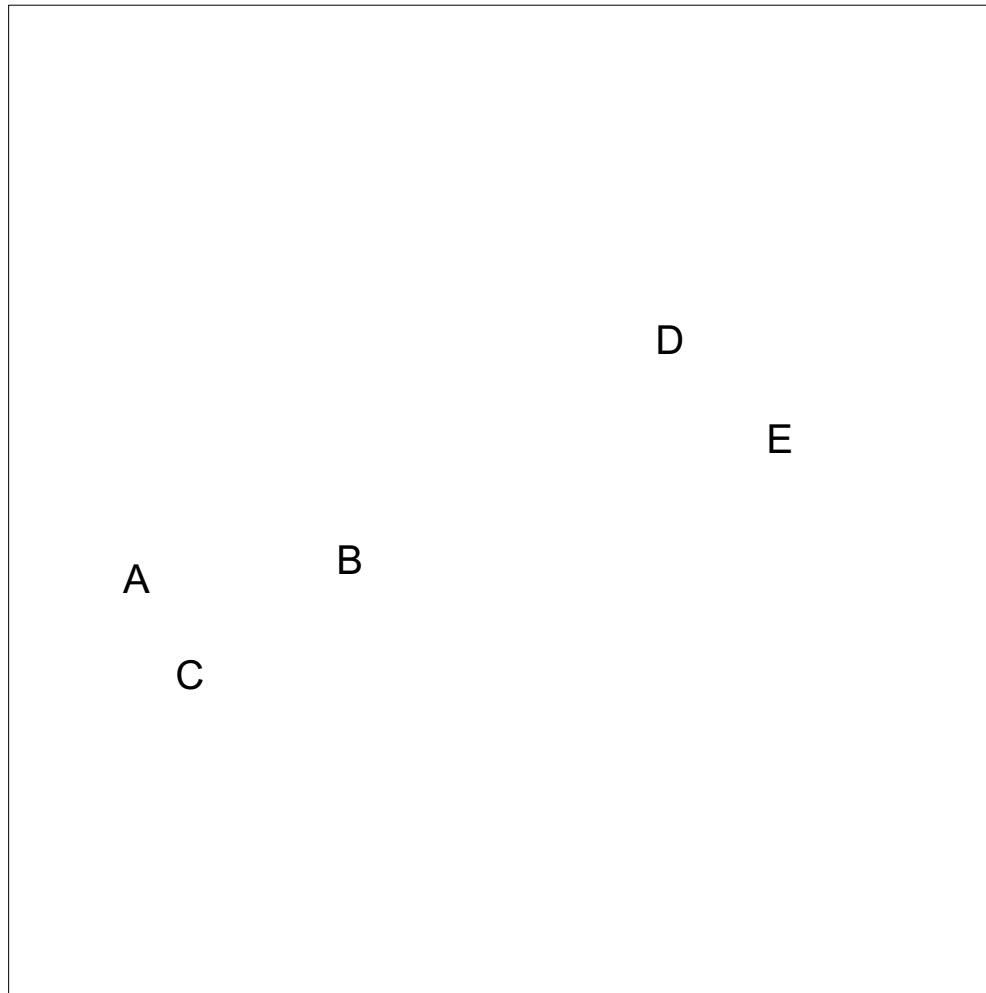
- In this section, we describe *bottom-up* or *agglomerative* clustering. This is the most common type of hierarchical clustering, and refers to the fact that a dendrogram is built starting from the leaves and combining clusters up to the trunk.

Top-down ?

(left to us)

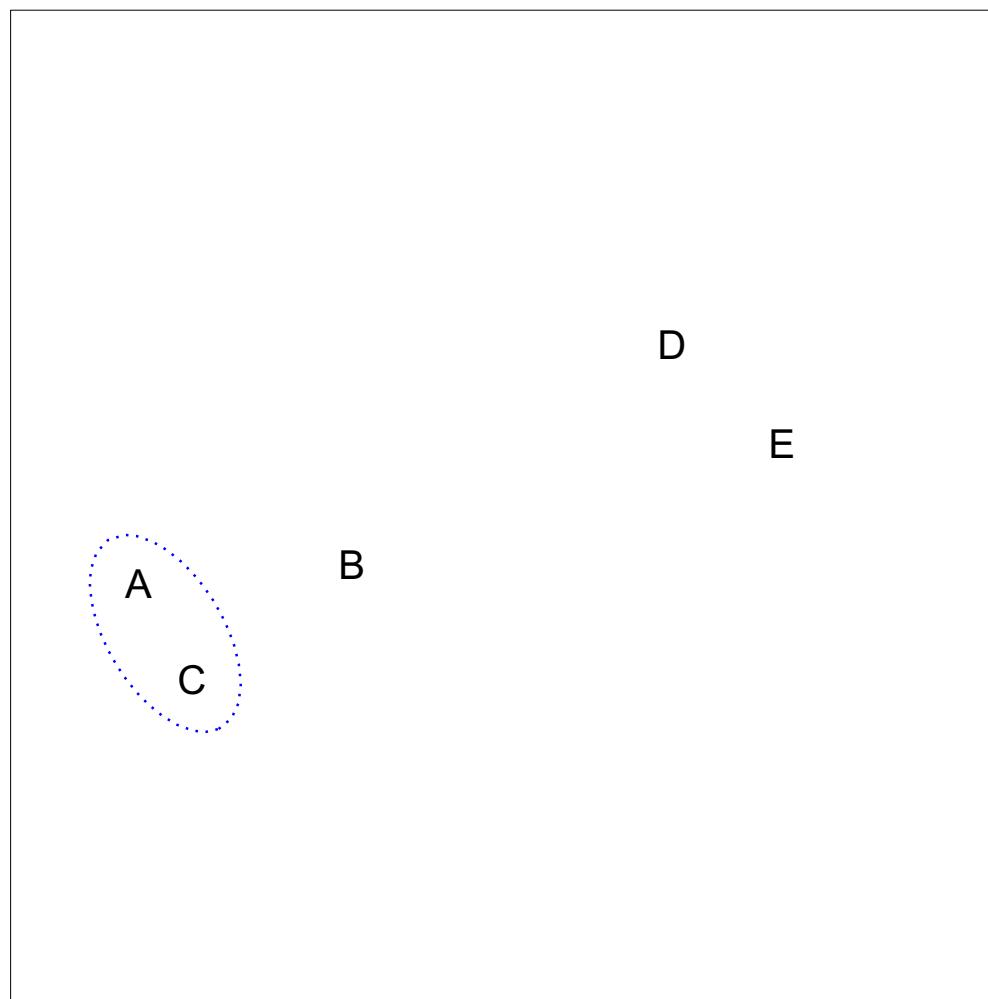
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



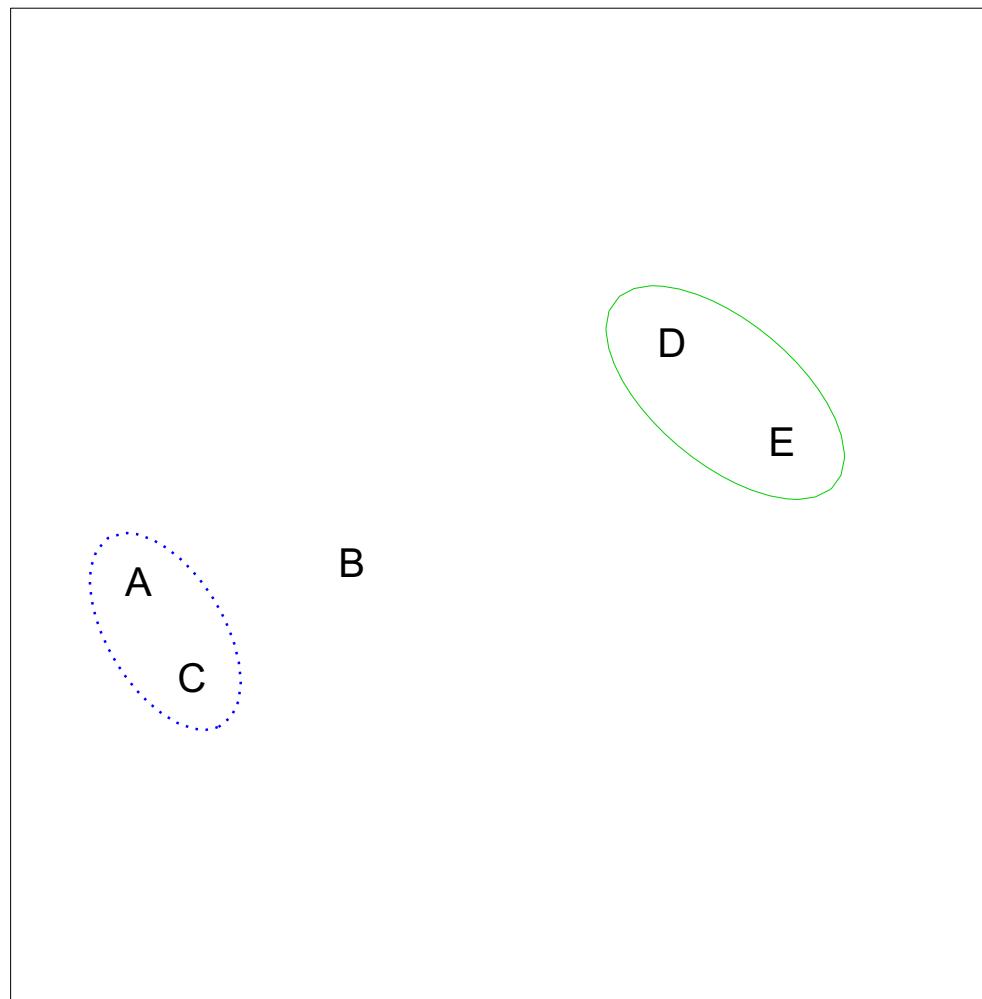
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



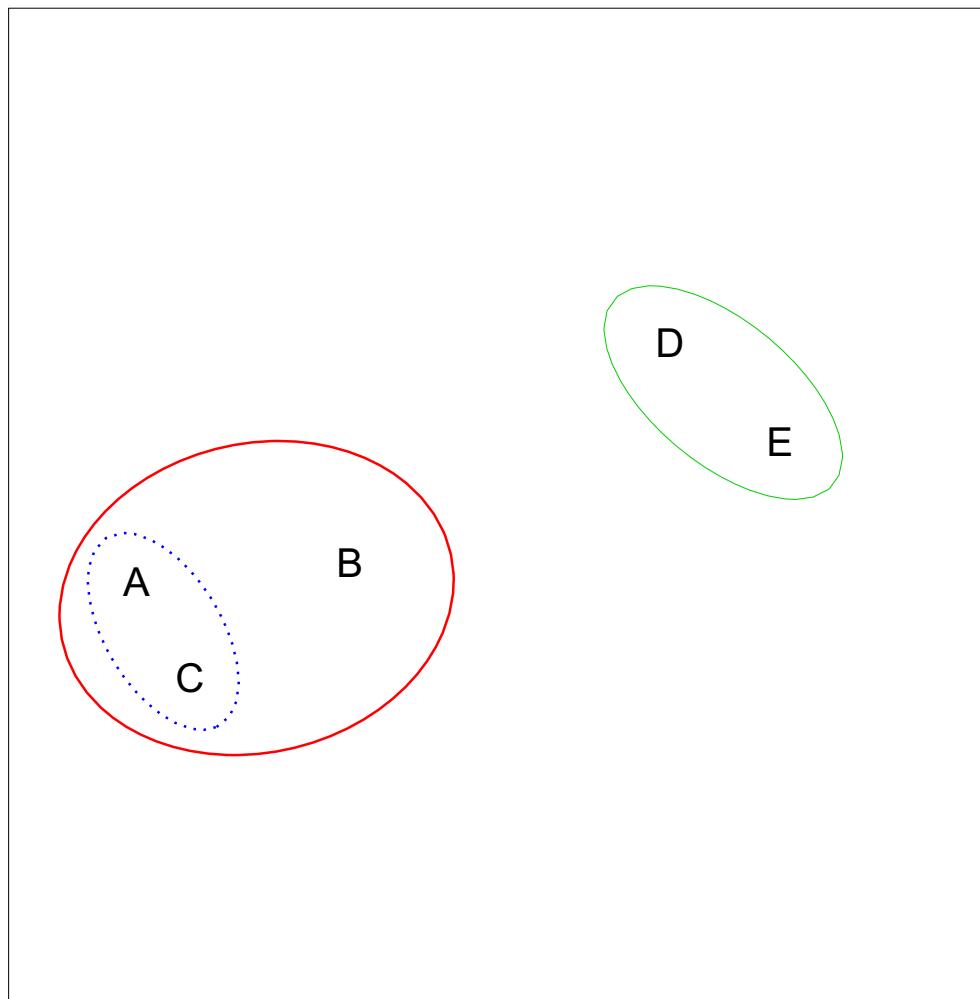
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



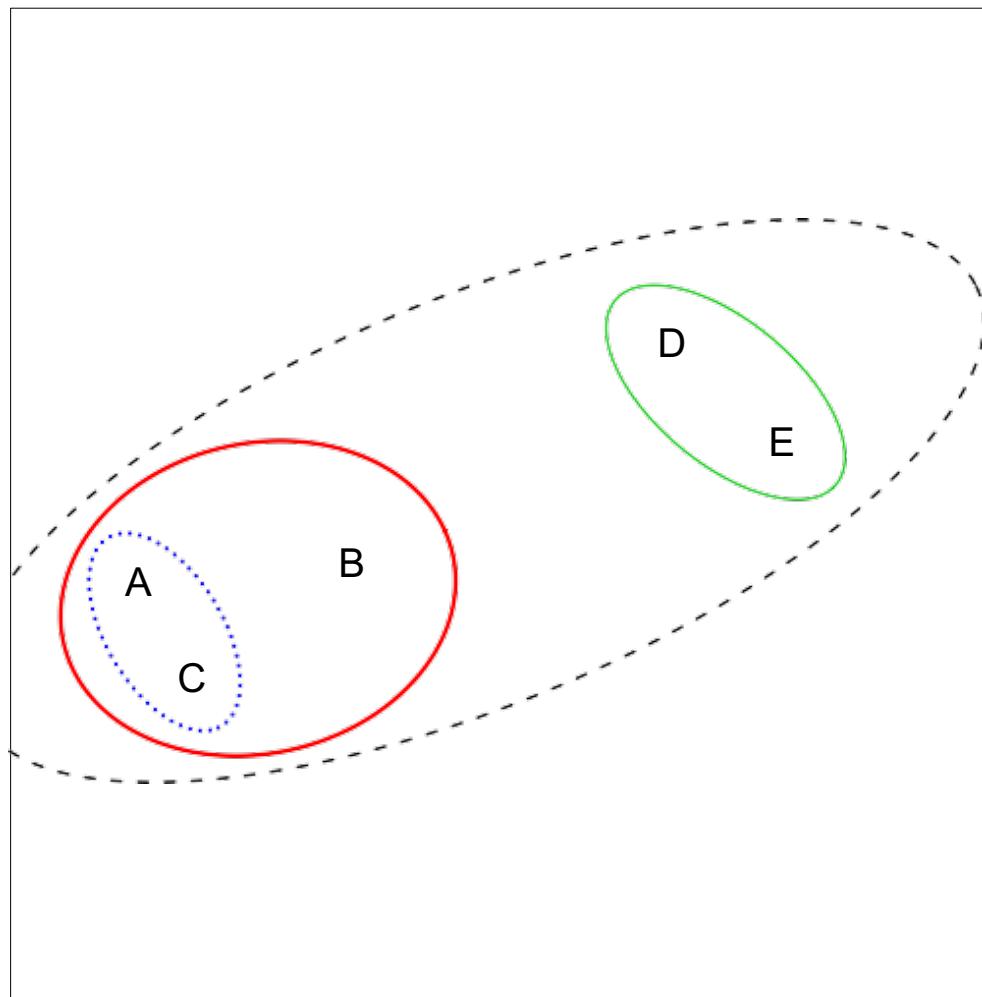
# Hierarchical Clustering: the idea

Builds a hierarchy in a “bottom-up” fashion...



# Hierarchical Clustering: the idea

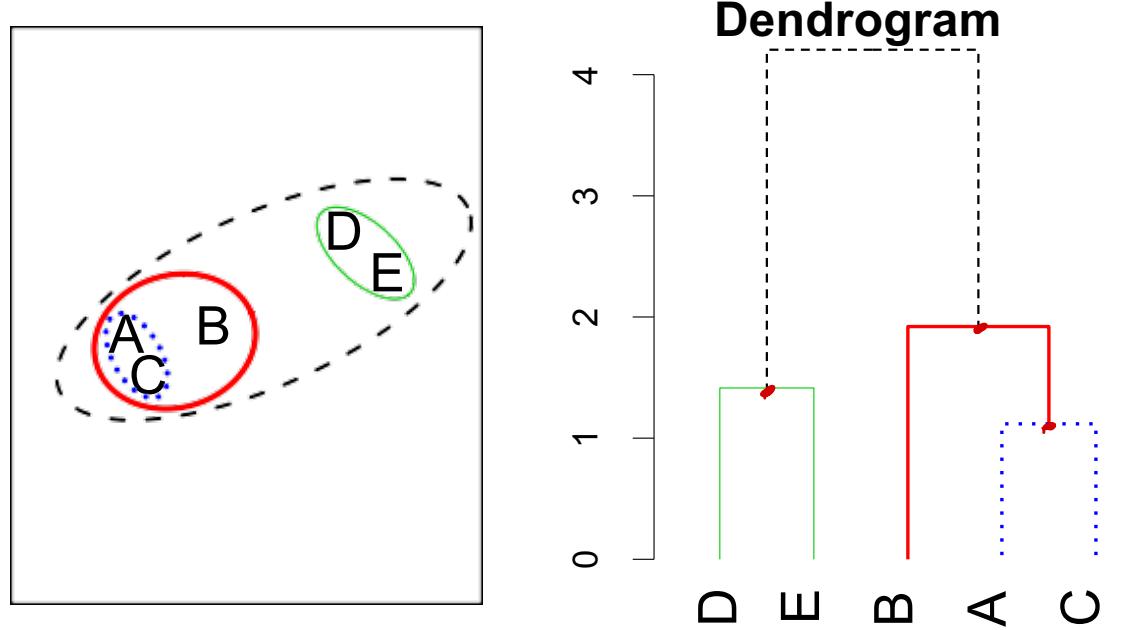
Builds a hierarchy in a “bottom-up” fashion...



# Hierarchical Clustering Algorithm

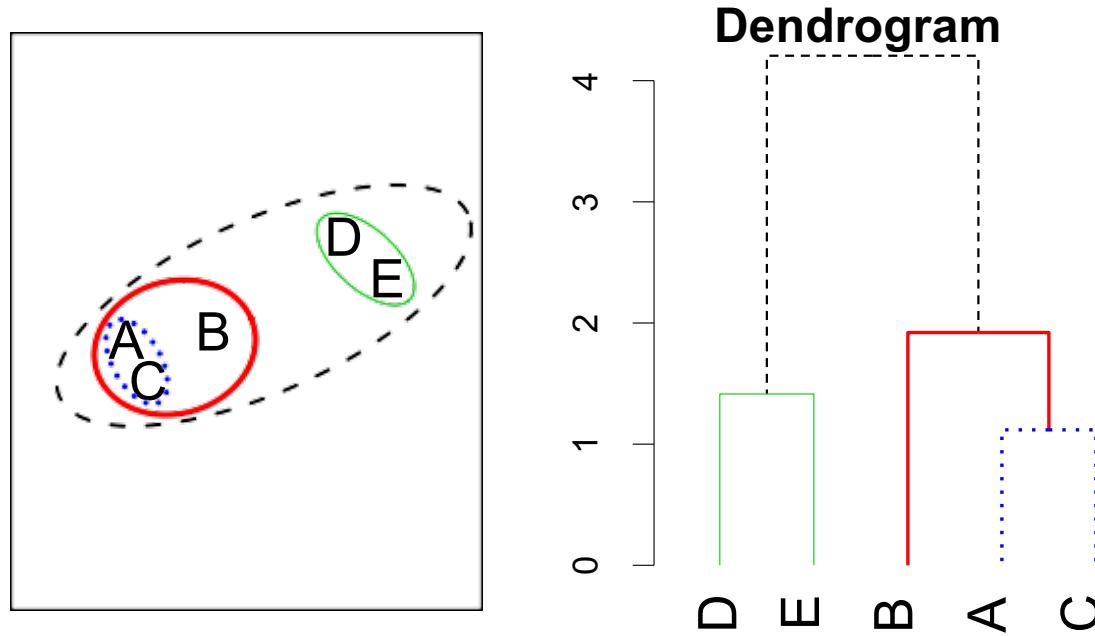
The approach in words:

- Start with each point in its own cluster.
- Identify the **closest** two clusters and merge them.
- Repeat.
- Ends when all points are in a single cluster.

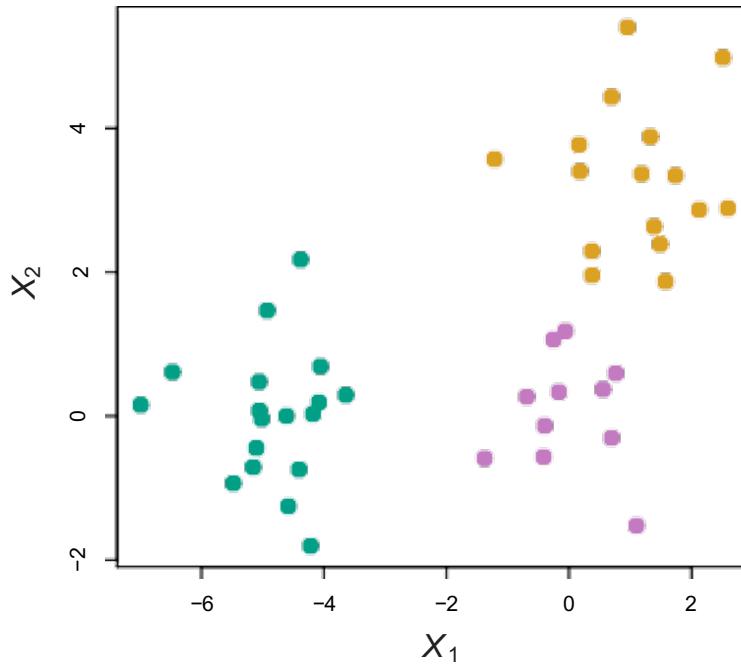


# Hierarchical Clustering Algorithm

The height of each node in the plot is proportional to the value of the intergroup dissimilarity between its two daughters



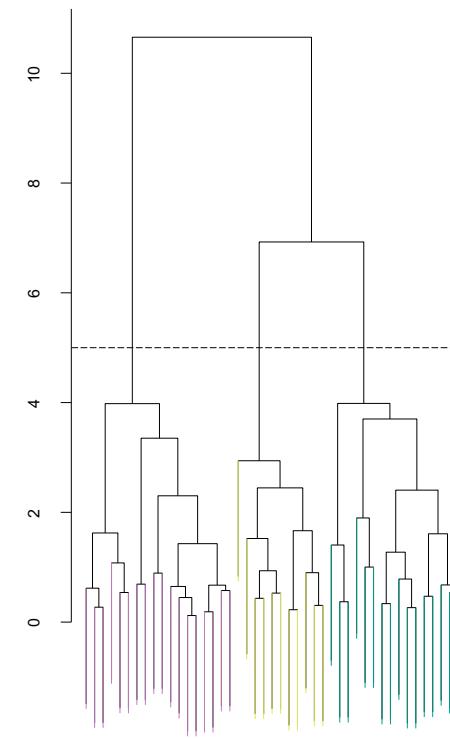
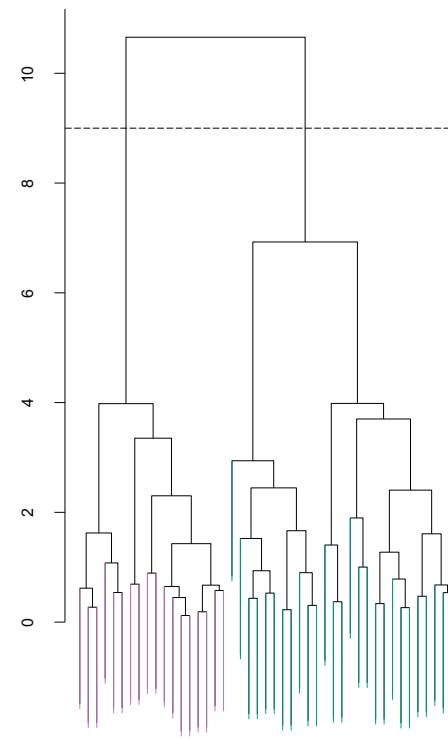
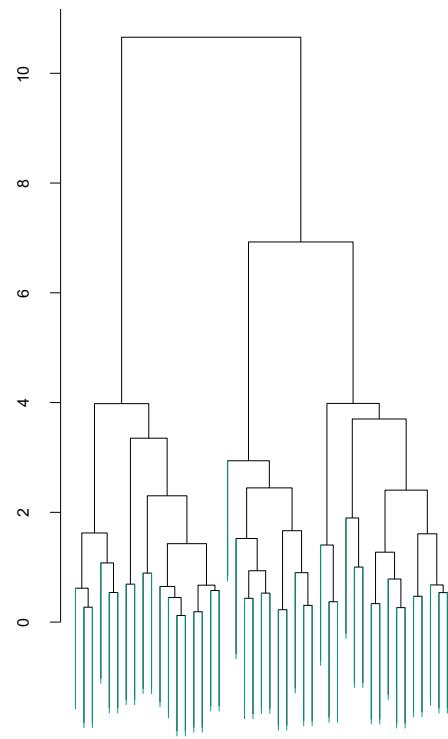
# An Example



45 observations generated in 2-dimensional space. In reality there are three distinct classes, shown in separate colors.

However, we will treat these class labels as unknown and will seek to cluster the observations in order to discover the classes from the data.

# Application of hierarchical clustering



# Details of previous figure

- *Left*: Dendrogram obtained from hierarchically clustering the data from previous slide, with complete linkage and Euclidean distance.
- *Center*: The dendrogram from the left-hand panel, cut at a height of 9 (indicated by the dashed line). This cut results in two distinct clusters, shown in different colors.

# Details of previous figure

- *Right:* The dendrogram from the left-hand panel, now cut at a height of 5. This cut results in three distinct clusters, shown in different colors.
- Note that the colors were not used in clustering, but are simply used for display purposes in this figure

# Algorithm

---

## Algorithm 10.2 *Hierarchical Clustering*

---

1. Begin with  $n$  observations and a measure (such as Euclidean distance) of all the  $\binom{n}{2} = n(n - 1)/2$  pairwise dissimilarities. Treat each observation as its own cluster.  
*dissimilarity*
  2. For  $i = n, n - 1, \dots, 2$ :
    - (a) Examine all pairwise inter-cluster dissimilarities among the  $i$  clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.
    - (b) Compute the new pairwise inter-cluster dissimilarities among the  $i - 1$  remaining clusters.
-

# Nested Clusters

- The term hierarchical refers to the fact that clusters obtained by cutting the dendrogram at a given height are necessarily **nested** within the clusters obtained by cutting the dendrogram at any greater height.
- However, on an arbitrary data set, this assumption of hierarchical structure might be unrealistic
- Hierarchical clustering can sometimes yield worse (i.e. less accurate) results than K - means clustering for a given number of clusters

# Nested Clusters: Example

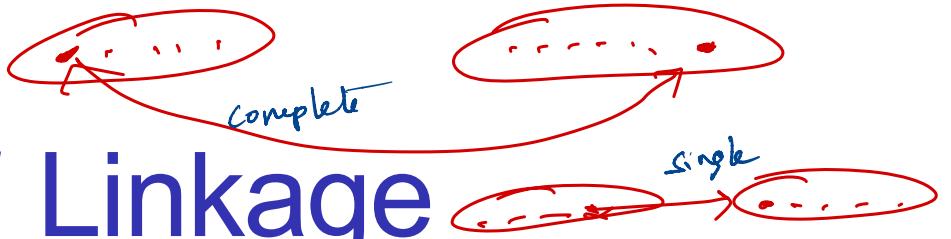
- Observations: a 50–50 split of males and females, evenly split among Americans, Japanese, and French.
- The best division into two groups might split these people by gender.
- The best division into three groups might split them by nationality.

# Nested Clusters: Example

- The true clusters are not nested
- The best division into three groups does not result from taking the best division into two groups and splitting up one of those groups.
- This situation could not be well-represented by hierarchical clustering.

# Dissimilarity between Groups

- The concept of **dissimilarity** between a pair of observations needs to be extended to a pair of groups of observations .
- This extension is achieved by developing the notion of **linkage**, which defines the dissimilarity between two groups of observations. The four most common types of linkage—**complete**, **average**, **single**, and **centroid**



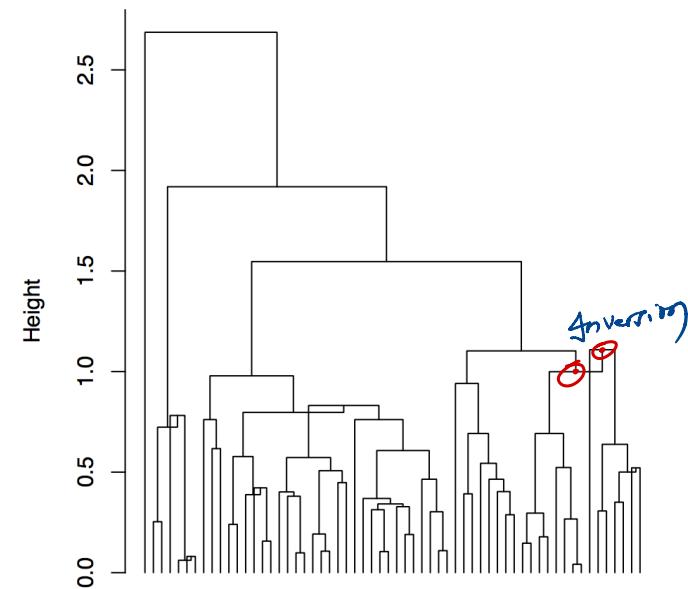
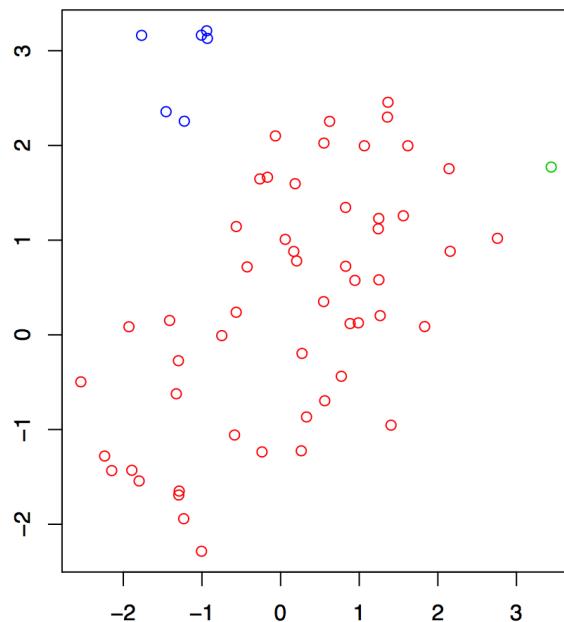
# Types of Linkage

| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.  |
| Single         | Minimal inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>smallest</i> of these dissimilarities. |
| Average        | Mean inter-cluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>average</i> of these dissimilarities.     |
| Centroid       | Dissimilarity between the centroid for cluster A (a mean vector of length $p$ ) and the centroid for cluster B. Centroid linkage can result in undesirable <i>inversions</i> .                              |



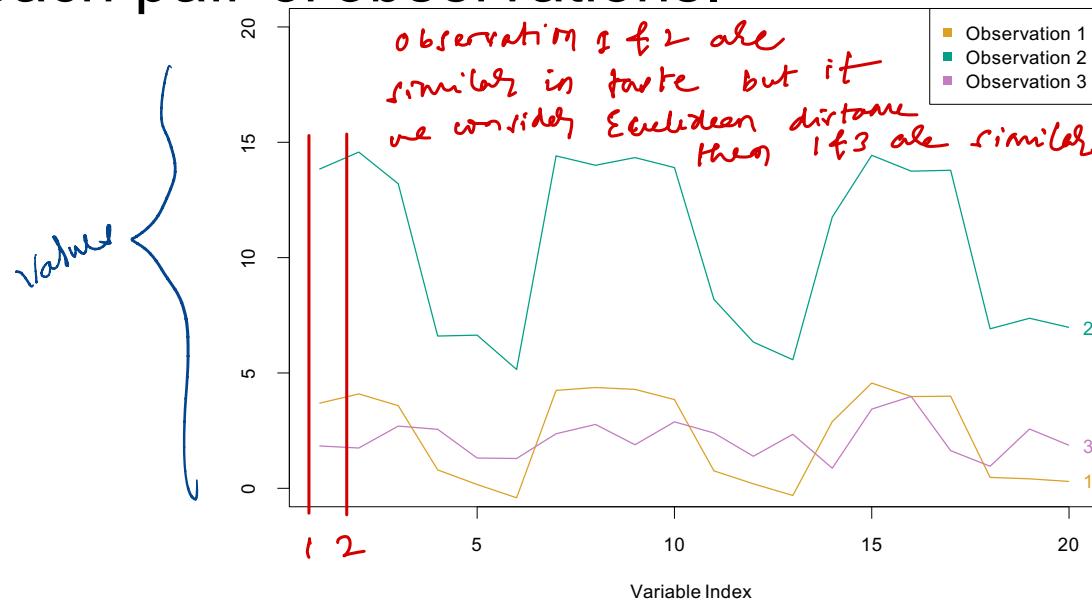
# Types of Linkage

- For centroid dissimilarity, **inversion** can occur, whereby two clusters are fused at a height below either of the individual clusters in the dendrogram. This can lead to difficulties in visualization as well as in interpretation of the dendrogram.



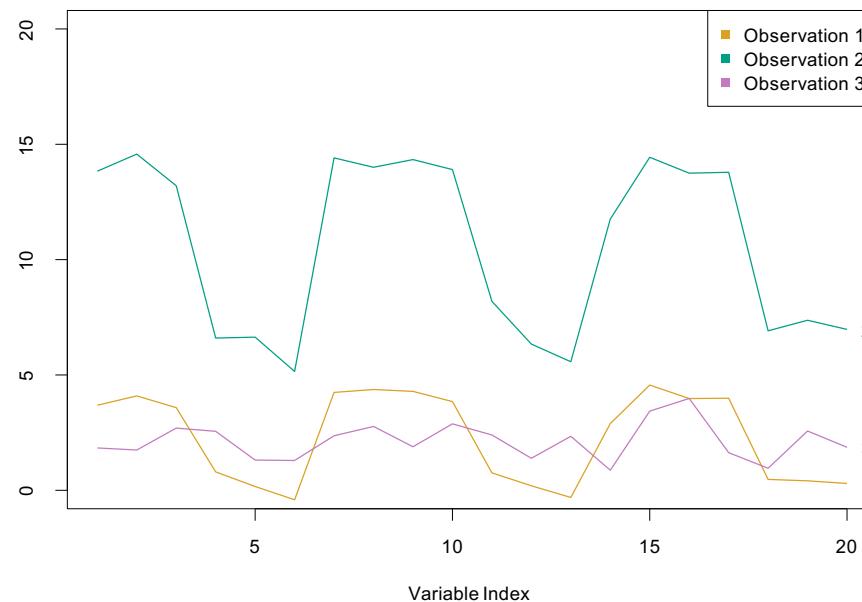
# Choice of Dissimilarity Measure

- So far have used Euclidean distance.
- An alternative is **correlation-based distance** which considers two observations to be similar if their features are highly correlated.
- This is an unusual use of correlation, which is normally computed between variables; here it is computed between the observation profiles for each pair of observations.



# Choice of Dissimilarity Measure

- Correlation-based distance focuses on the shapes of observation profiles rather than their magnitudes.
- Observations 1, 3 have small Euclidean distance but weak correlation (low correlation similarity).
- Observations 1, 2 have large Euclidean distance but strong correlation (high correlation similarity).



Venky

# Practical issues

- Should the observations or features first be standardized in some way? For instance, maybe the variables should be centered to have mean zero and scaled to have standard deviation one.
- In the case of hierarchical clustering,
  - What dissimilarity measure should be used?
  - What type of linkage should be used?
- How many clusters to choose? (in both  $K$ -means or hierarchical clustering). Difficult problem. No agreed-upon method. See Elements of Statistical Learning, chapter 13 for more details.

# How many clusters?

- Sometimes, we might have no problem specifying the number of clusters  $K$  ahead of time, e.g.,
  - Segmenting a client database into  $K$  clusters for  $K$  salesmen
- Other times,  $K$  is implicitly defined by cutting a hierarchical clustering tree at a given height
- In most exploratory applications,  $K$  is unknown.
- What is the “right” value of  $K$ ?

# This is a hard problem

Determining the number of clusters is a  
**hard problem!**

Why is it hard?

- Determining the number of clusters is a hard task for humans to **perform** (unless the data are low-dimensional). Not only that, it's just as hard to **explain** what it is we're looking for. Usually, statistical learning is successful when at least one of these is possible

# This is a hard problem

- Why is it important?
  - E.g., it might mean a big difference scientifically if we were convinced that there were  $K = 2$  subtypes of breast cancer vs.  $K = 3$  subtypes
  - One of the (larger) goals of data mining/statistical learning is automatic inference; choosing  $K$  is certainly part of this.

# Reminder: within-cluster variation

We focus on K-means, but most ideas apply to other settings

Recall: given the number of clusters  $K$ , the  $K$ -means algorithm approximately minimizes the within-cluster variation:

$$W = \sum_{k=1}^K \sum_{C(i)=k} \|X_i - \bar{X}_k\|_2^2$$

*when  $K=N$  the wcv is zero*

$\nwarrow$  centroid

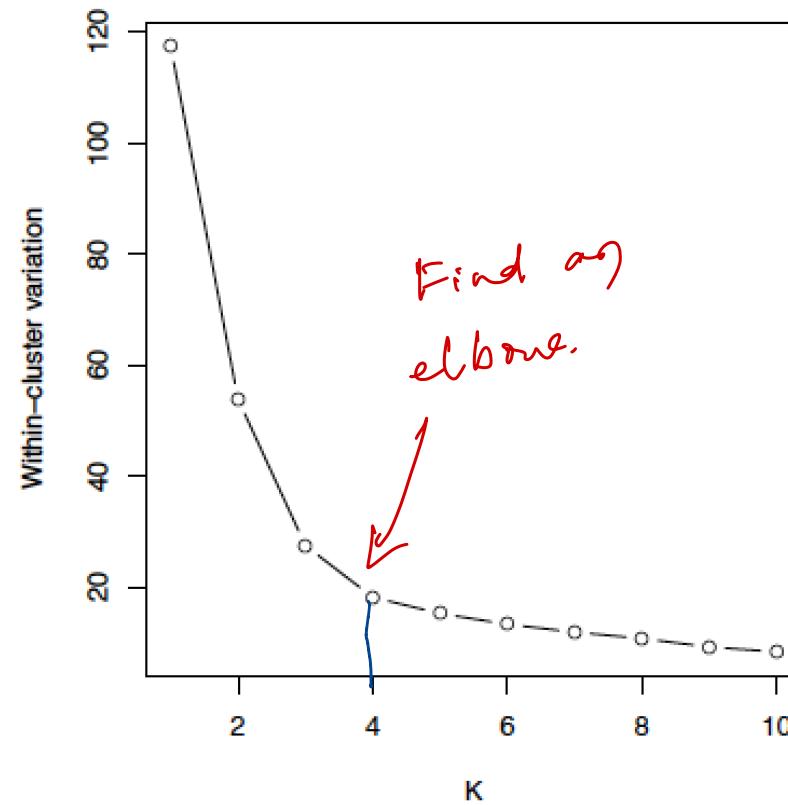
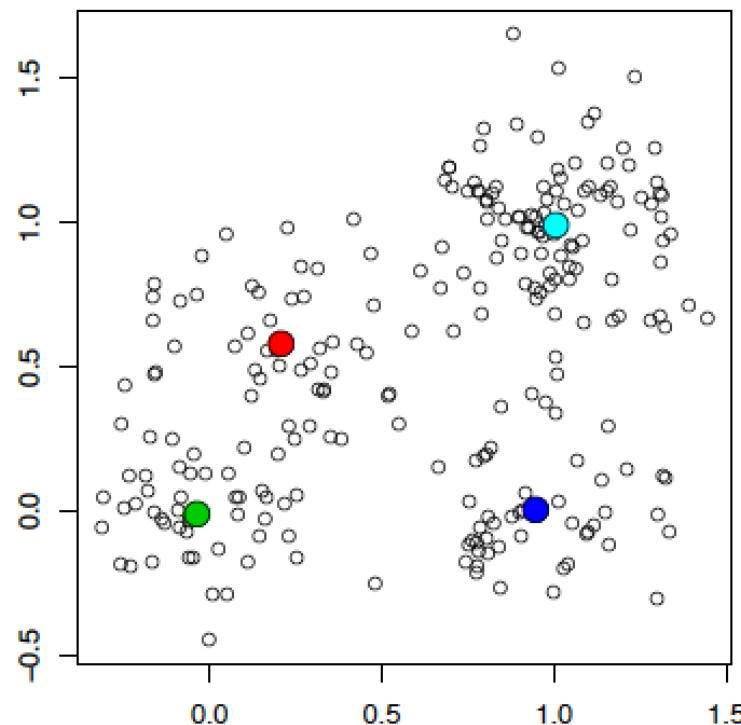
over clustering assignments  $C$ , where  $\bar{X}_k$  is the average of points in group  $k$

Clearly a lower value of  $W$  is better. So why not just run  $K$ -means for a bunch of different values of  $K$ , and choose the value of  $K$  that gives the smallest  $W(K)$ ?

# That's not going to work

Problem: within-cluster variation just keeps decreasing: **scree plot**

Example:  $n = 250, p = 2, K = 1, \dots, 10$



# Between-cluster variation

Within-cluster variation measures how **tightly grouped** the clusters are. As we increase the number of clusters  $K$ , this just keeps going down. What are we missing?

**Between-cluster variation** measures how spread apart the groups are from each other:

$$B = \sum_{k=1}^K n_k \|\bar{X}_k - \bar{X}\|_2^2$$

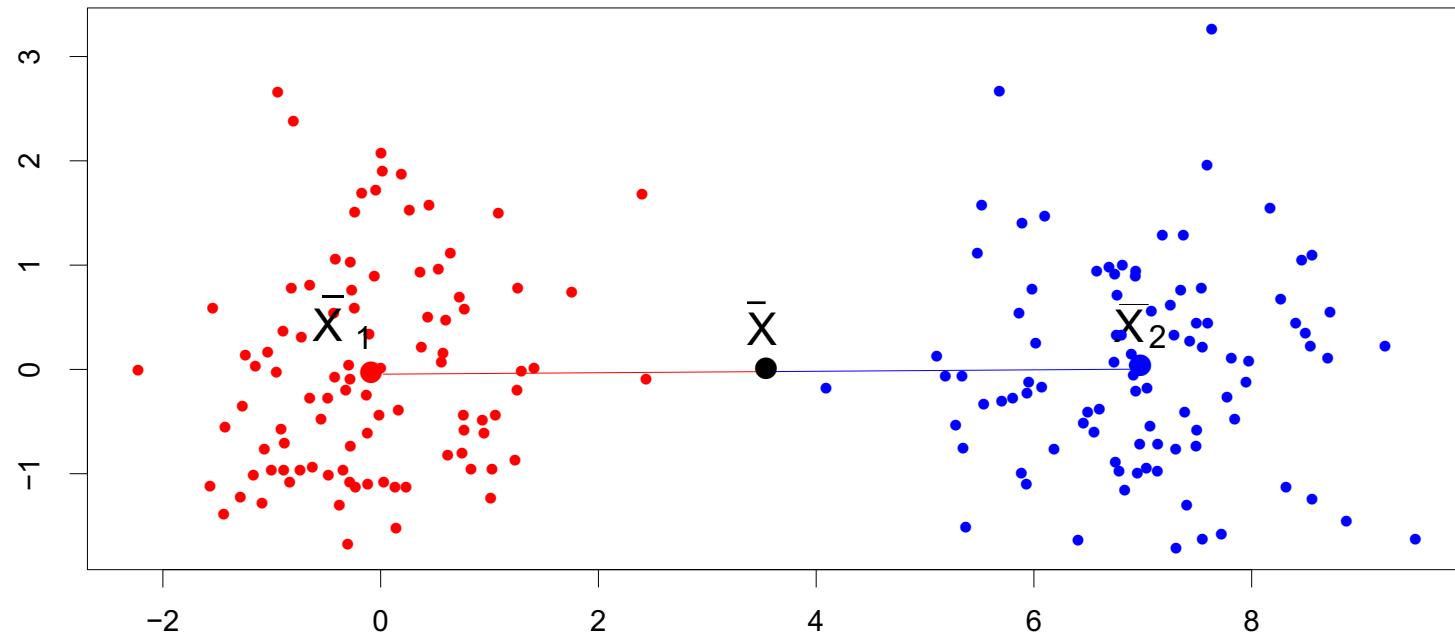
*centroïd of whole data  
of cluster  $k$ .*

where as before  $\bar{X}_k$  is the average of points in group  $k$ , and  $\bar{X}$  is the overall average, i.e.

$$\bar{X}_k = \frac{1}{n_k} \sum_{C(i)=k} X_i \quad \text{and} \quad \bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$$

# Example: between-cluster variation

Example:  $n = 100$ ,  $p = 2$ ,  $K = 2$



$$B = n_1 \|\bar{X}_1 - \bar{X}\|_2^2 + n_2 \|\bar{X}_2 - \bar{X}\|_2^2$$

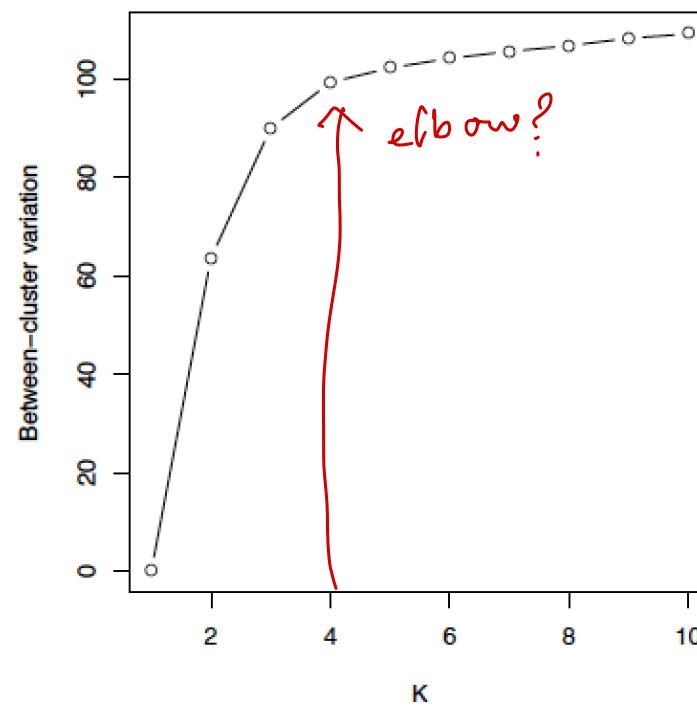
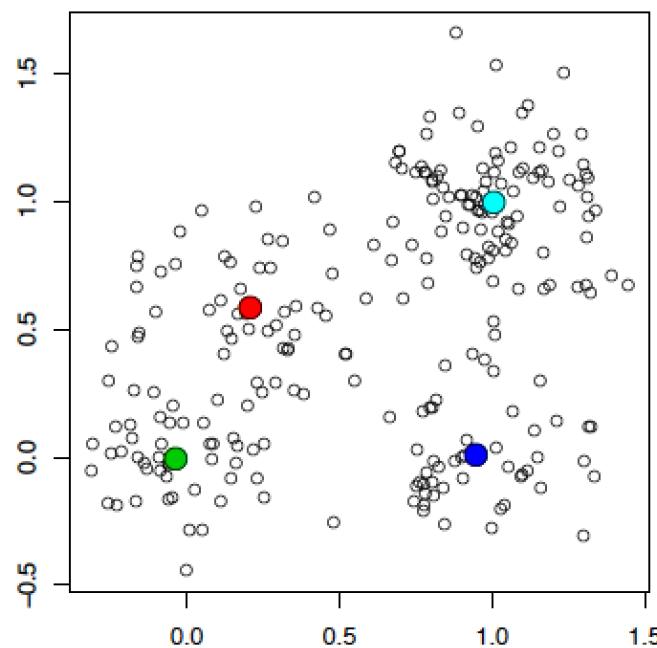
$$W = \sum_{C(i)=1} \|X_i - \bar{X}_1\|_2^2 + \sum_{C(i)=2} \|X_i - \bar{X}_2\|_2^2$$

# Still not going to work

Bigger B is better, can we use it to choose  $K$ ?

Problem: between- cluster variation just keeps increasing

Running example:  $n = 250$ ,  $p = 2$ ,  $K = 1, \dots, 10$



Scree  
plot

# CH index

- Ideally we'd like our clustering assignments  $C$  to **simultaneously** have a small  $W$  and a large  $B$
- This is the idea behind the **CH index** proposed by Calinski and Harabasz (1974), in “A dendrite method for cluster analysis”
- For clustering assignments coming from  $K$  clusters, we record CH score:

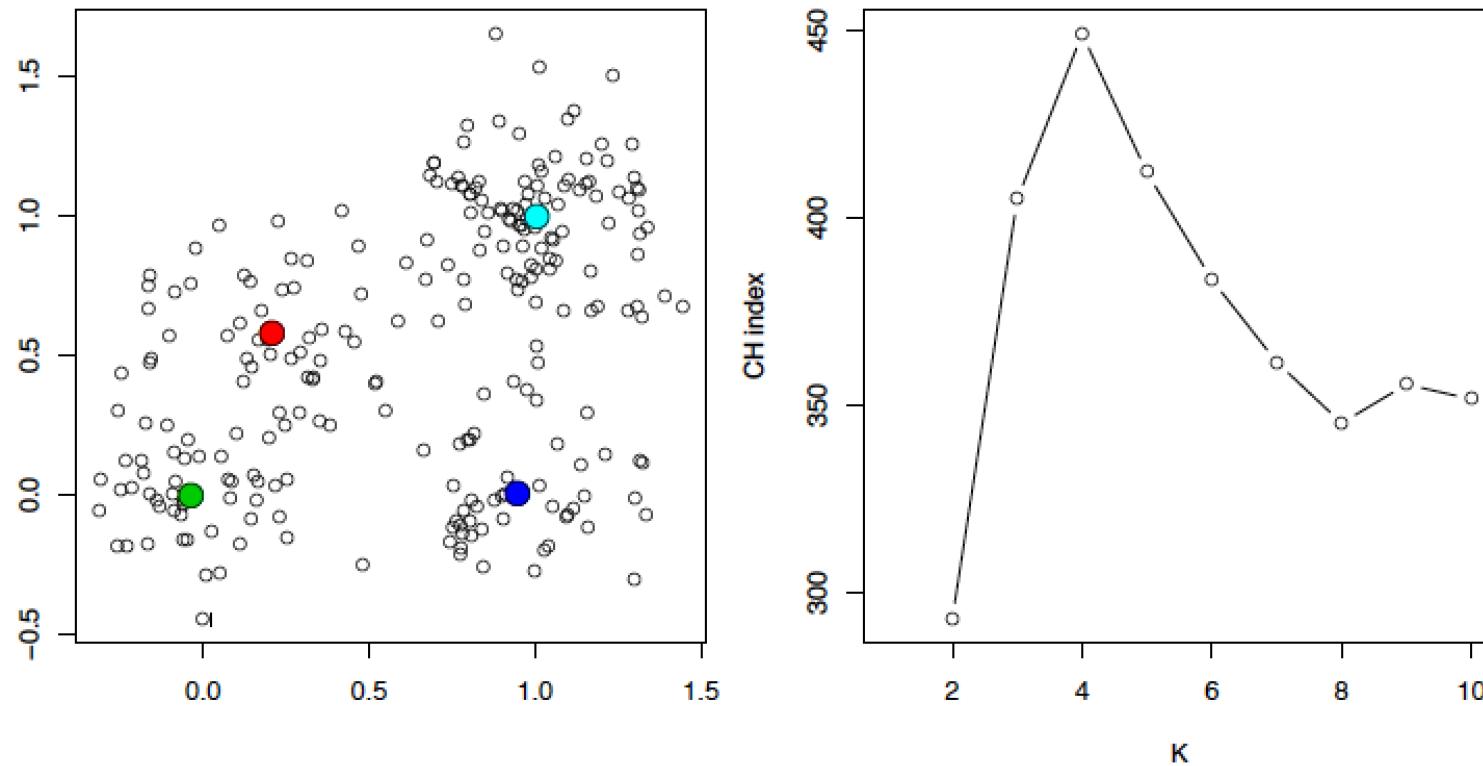
$$CH(K) = \frac{B(K)/(K - 1)}{W(K)/(n - K)}$$

*Similar to F  
score*

- To choose  $K$ , just pick some maximum number of clusters to be considered  $K_{\max}$  (e.g.,  $K = 20$ ), and choose the value of  $K$  with the largest score  $CH(K)$ .

# Example: CH index

Running example:  $n = 250$ ,  $p = 2$ ,  $K = 2, \dots, 10$ .



We would choose  $K = 4$  clusters, which seems reasonable

General problem: the CH index is not defined for  $K = 1$ . We could never choose just one cluster (the null model)!

Sometimes, we have a large data set  
and we can only examine

$$K \leq \text{man-value}$$

and it is possible that the actual number  
of clusters is larger than the man-value  
and we see an increasing CH-index over  
the range 0-K.

# Gap statistic

It's true that  $W(K)$  keeps dropping, but  
**how much it drops** at any one  $K$  should  
be informative

The gap statistic was introduced by  
Tibshirani et al. (2001), “Estimating the  
number of clusters in a data set via the  
gap statistic”

# Gap statistic

It is based on this idea. We compare the observed within-cluster variation  $W(K)$  to  $W_{\text{unif}}(K)$ , the within-cluster variation we'd see if we instead had points distributed uniformly (over an *encapsulating box*). The gap for  $K$  clusters is defined as

$$\text{Gap}(K) = \log W_{\text{unif}}(K) - \log W(K)$$

Annotations in red:

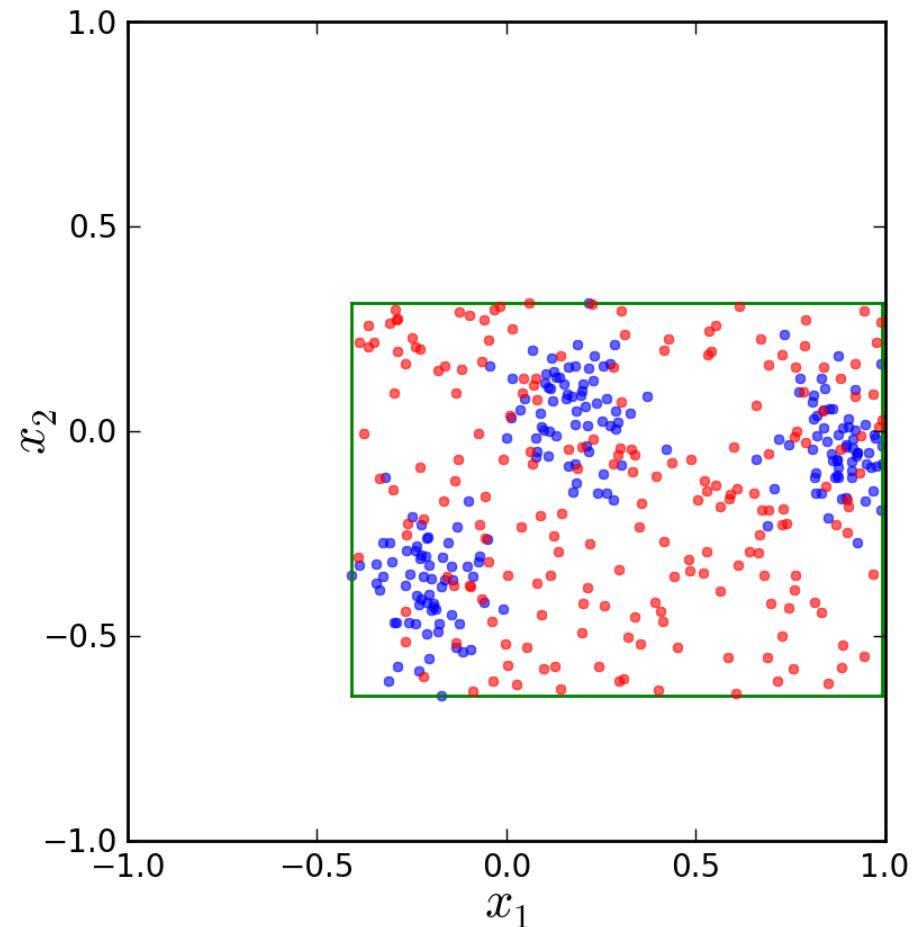
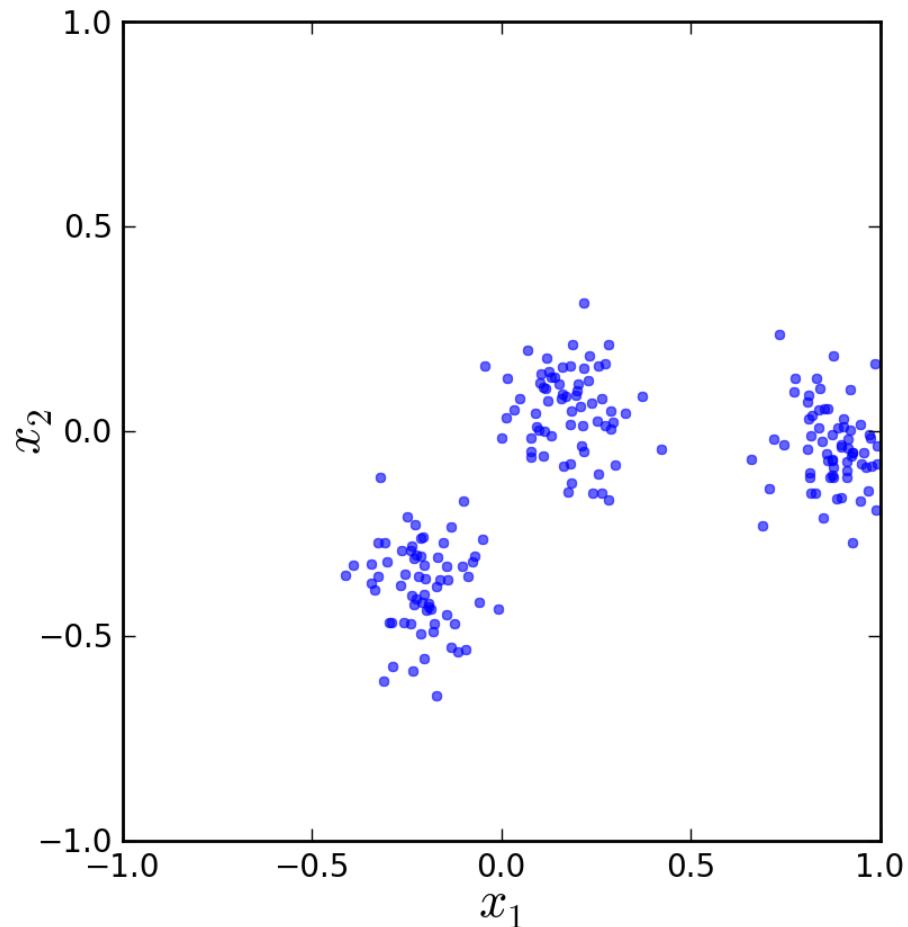
- A red arrow points from the word "Random" to the term  $W_{\text{unif}}(K)$ .
- A red arrow points from the word "Random variable" to the term  $W(K)$ .
- A red bracket underlines the expression  $\log W_{\text{unif}}(K)$  with the handwritten note "not random".

# Gap statistic

The reference datasets are in our case generated by sampling uniformly from the original dataset's bounding box (see green box in the upper right plot of the figures below). To obtain the estimate  $\log W_{\text{unif}}(K)$ , we compute the average of  $B$  copies of  $\log W(K)$ , each of which is generated from the uniform sample, and their standard deviation  $s(K)$ .

# Gap statistic

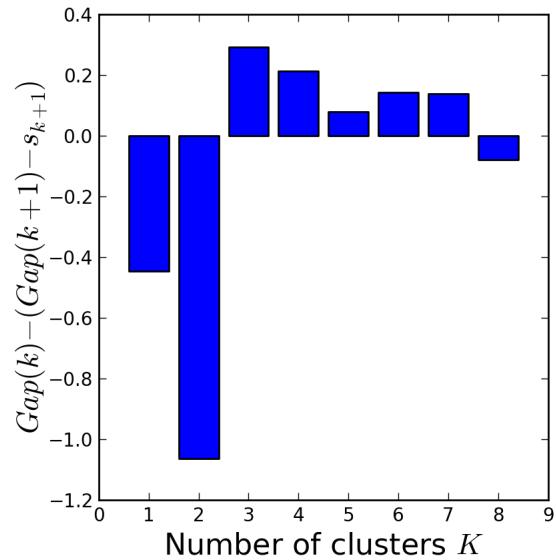
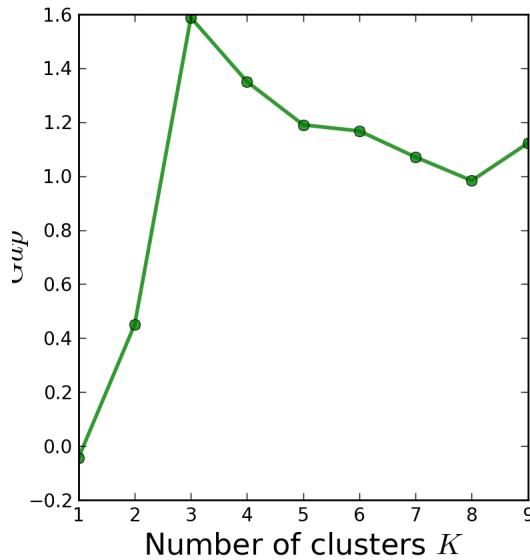
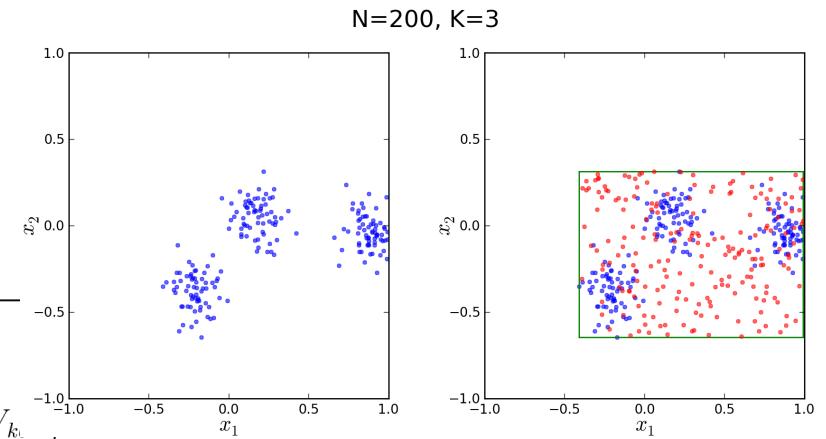
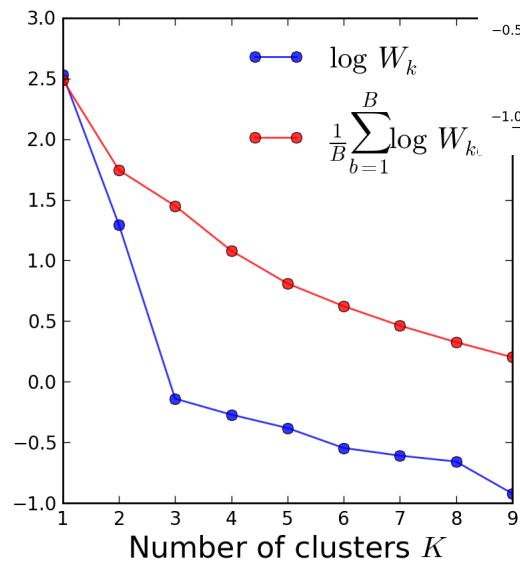
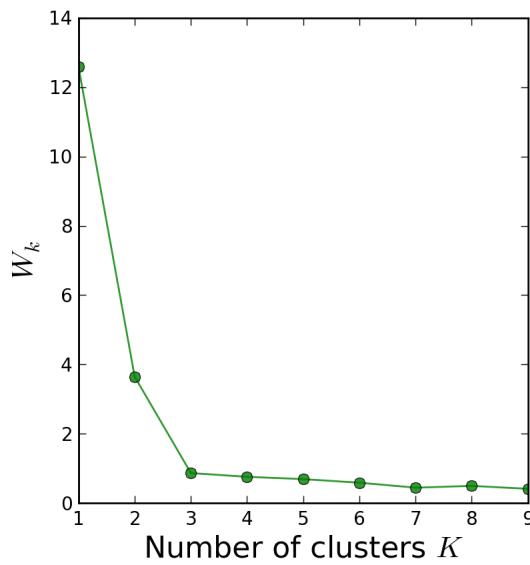
N=200, K=3



$B$  unit. sampler  $\Rightarrow$   $B$  group( $K$ )

Sample mean  
Sample std

# Gap statistic



# Gap statistic

Then we choose  $K$  by:

$$\hat{K} = \min_{K \in \{1, \dots, K_{\max}\}} : \text{Gap}(K) \geq \text{Gap}(K+1) - s(K+1),$$

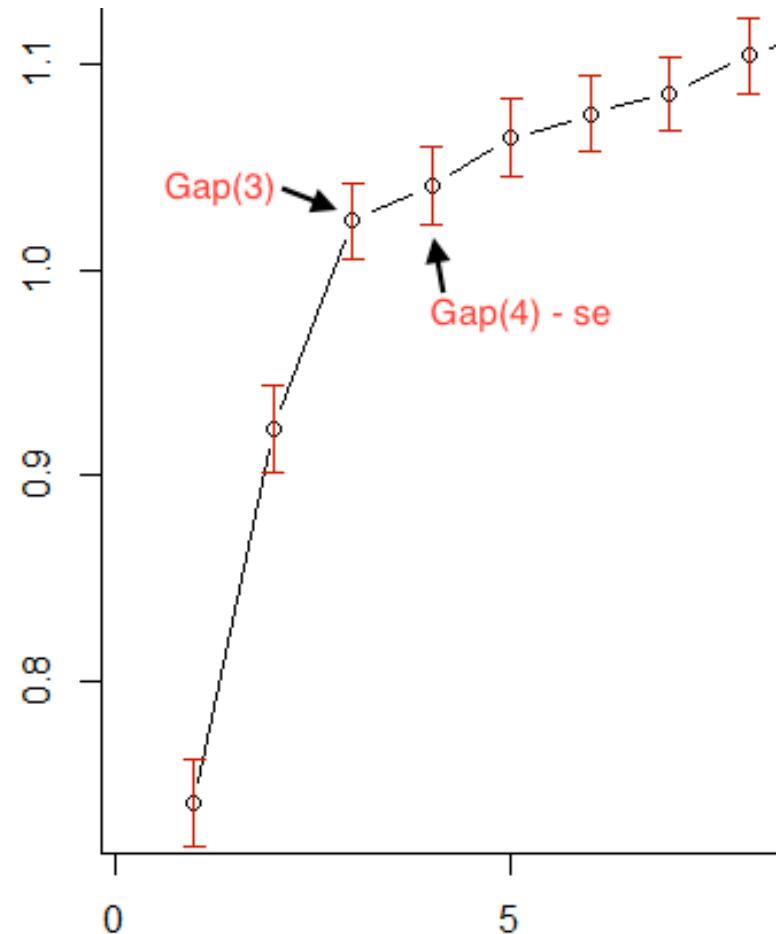
The first  $K$

# Gap statistic

This means:

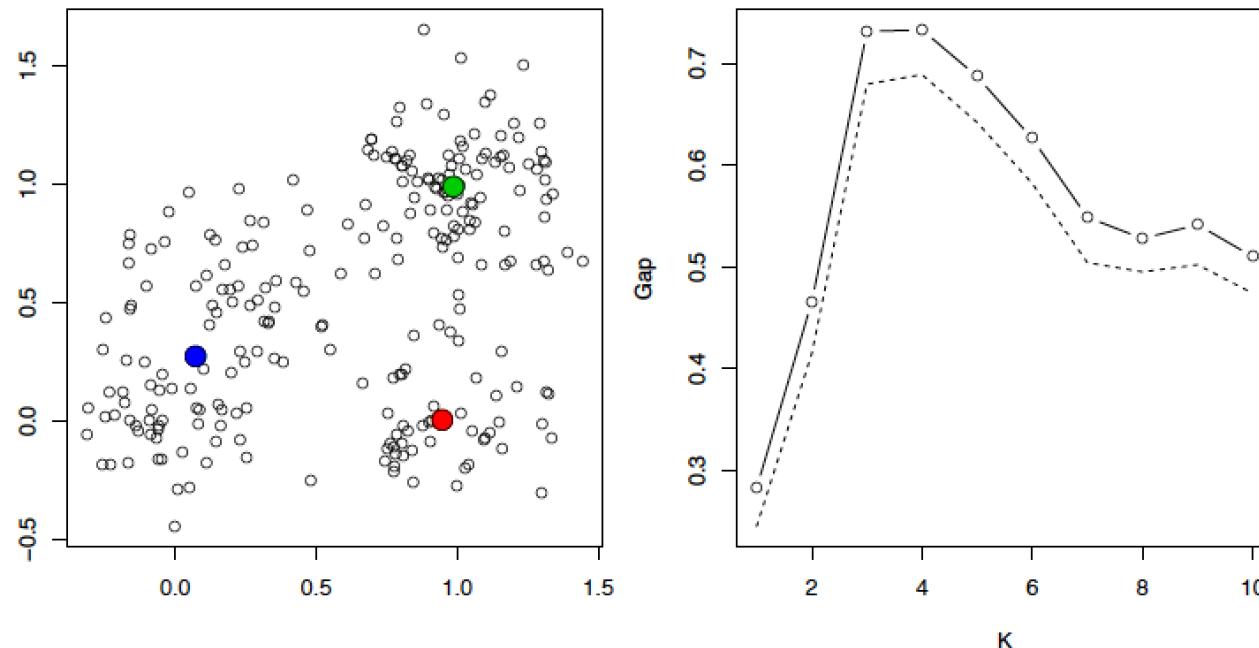
Choose the cluster size  $\hat{k}$  to be the smallest  $k$  such that

$$\text{Gap}(k) \geq \text{Gap}(k+1) - s(k+1)$$



# Example: gap statistic

Running example:  $n = 250$ ,  $p = 2$ ,  $K = 1, \dots, 10$



We would choose  $K = 3$  clusters, which is also reasonable

The gap statistic does especially well when the data **fall into one cluster**. (Why? Hint: think about the null distribution that it uses)

# CH index and gap statistic in R

The CH index can be computed using the kmeans function in the base distribution, which returns both the within-cluster variation and the between-cluster variation

E.g.,

$k = 5$

```
km = kmeans(x, k, alg="Lloyd")
names(km)
# Now use some of these return items to compute ch
```

The gap statistic is implemented by the function gap in the package lga, and by the function gap in the package SAGx. (Beware: these functions are poorly documented ... it's unclear what clustering method they're using)

# Silhouette Analysis

- A method of interpretation and validation of consistency within clusters of data
- Provides a succinct graphical representation of how well each object lies within its cluster

# Silhouette Analysis

- Assume the data have been clustered via any technique, such as k-means, into  $k$  clusters.
- For each sample  $\mathbf{x}_i$ , let  $a_i$  be the average distance between  $\mathbf{x}_i$  and all other data within the same cluster.
- Can interpret  $a_i$  as a measure of **how well  $\mathbf{x}_i$  is assigned to its cluster** (the smaller the value, the better the assignment).

# Silhouette Analysis

- We then define the average dissimilarity of point  $\mathbf{x}_i$  to a cluster  $c$  as the average of the distance from  $\mathbf{x}_i$  to all points in  $c$ .
- Let  $b_i$  be the lowest average distance of  $\mathbf{x}_i$  to all points in any other cluster, of which  $\mathbf{x}_i$  is not a member.

# Silhouette Analysis

- The cluster with this lowest average dissimilarity is said to be the "**neighboring cluster**" of  $\mathbf{x}_i$ , because it is the next best fit cluster for point  $\mathbf{x}_i$ . We now define a silhouette:

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

- or

$$s_i = \begin{cases} 1 - a_i / b_i & a_i < b_i \\ 0 & a_i = b_i \\ b_i / a_i - 1 & a_i > b_i \end{cases}$$

# Silhouette Analysis

- From

$$s_i = \begin{cases} 1 - a_i / b_i & a_i < b_i \\ 0 & a_i = b_i \\ b_i / a_i - 1 & a_i > b_i \end{cases}$$

$0 < s_i \leq 1$   
 $s_i = 0$   
 $-1 \leq s_i < 0$

it is obvious that  $-1 \leq s_i \leq 1$ .

If  $a_i < b_i$ , it belongs to the same cluster if not the neighbour.

$a_i = b_i$ , unclear

$a_i > b_i$ , not correctly assigned.

# Silhouette Analysis

- $s_i$  close to 1 requires  $a_i \ll b_i$ .
- $a_i$  is a measure of dissimilarity of  $\mathbf{x}_i$  to its own cluster
- Thus a small  $a_i$  means  $\mathbf{x}_i$  is well matched.
- Large  $b_i$  implies that  $\mathbf{x}_i$  is badly matched to its neighboring cluster.
- Thus an  $s_i$  close to one means that  $\mathbf{x}_i$  is appropriately clustered.

# Silhouette Analysis

- $s_i$  close to negative one, by the same logic means that  $\mathbf{x}_i$  would be more appropriate if it was clustered in its neighboring cluster.
- An  $s_i$  near zero means that  $\mathbf{x}_i$  is on the border of two natural clusters.

# Silhouette Analysis

- The average  $s_i$  over a cluster measures how tightly grouped all the data in the cluster are.
- Thus the average  $s_i$  over the entire dataset is a measure of how appropriately the data have been clustered.

# Silhouette Analysis

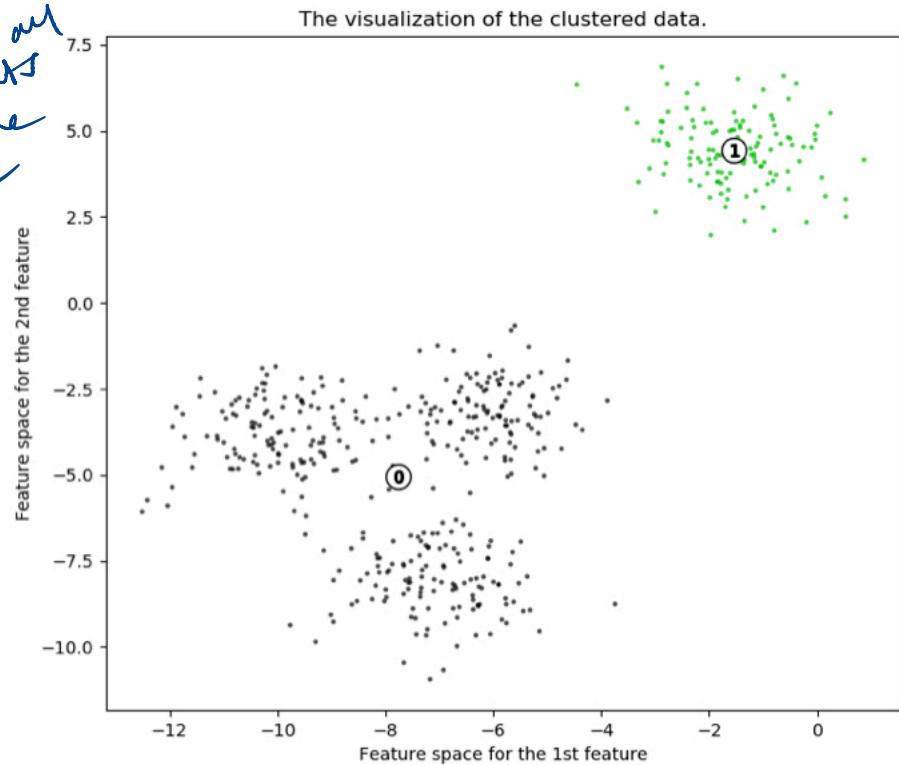
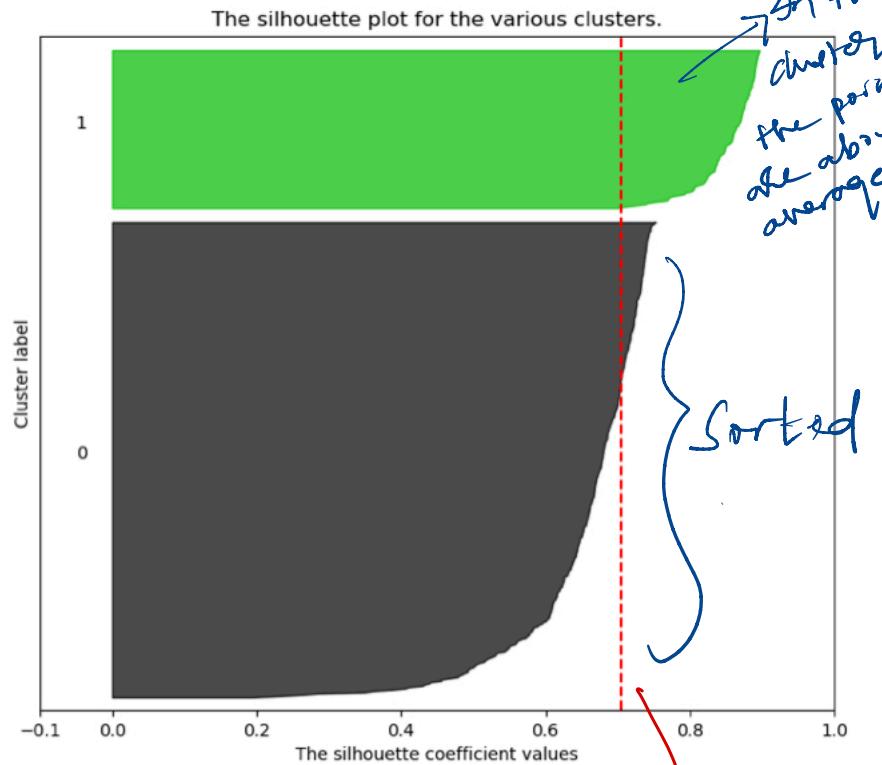
- If there are too many or too few clusters, some of the clusters will display narrower silhouettes than the rest.
- So silhouette plots and averages may be used to determine the natural number of clusters within a dataset.

# Silhouette Analysis

- One can also increase the likelihood of the silhouette being maximized at the correct number of clusters by re-scaling the data using feature weights that are cluster specific.

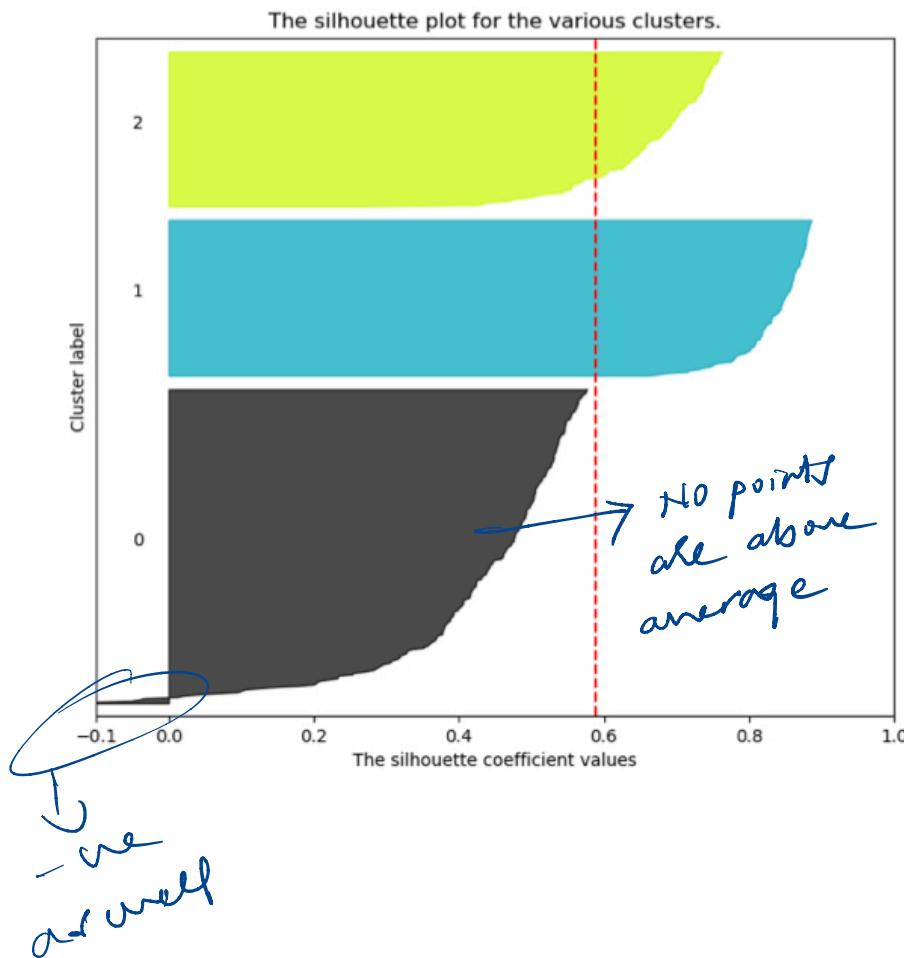
# Silhouette Plot

Silhouette analysis for KMeans clustering on sample data with n\_clusters = 2



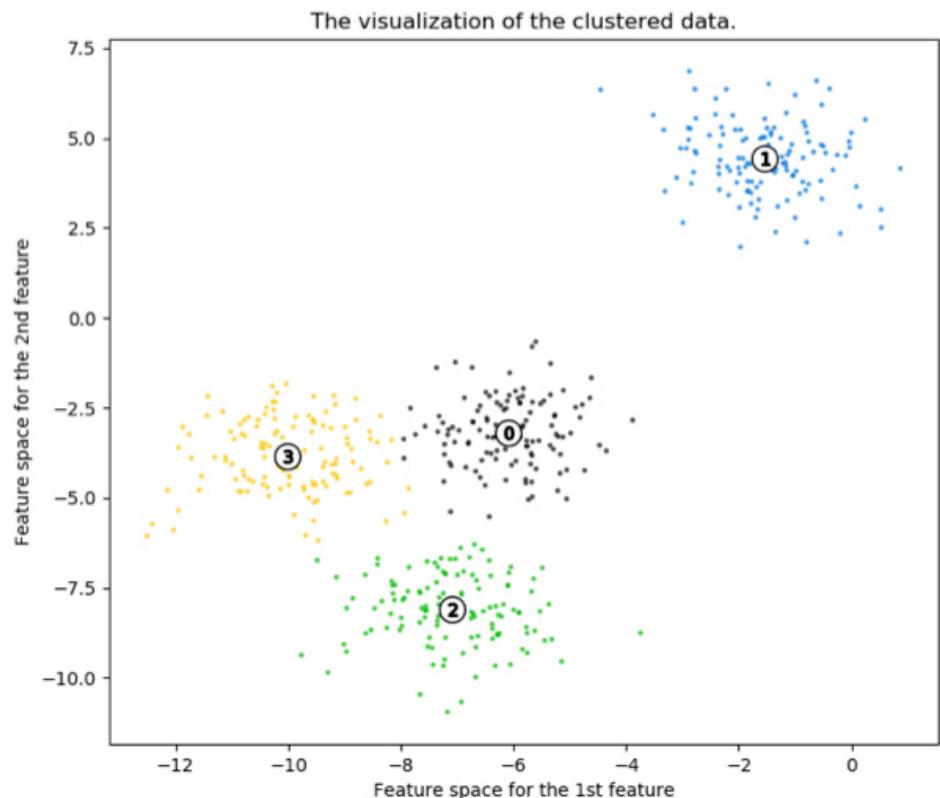
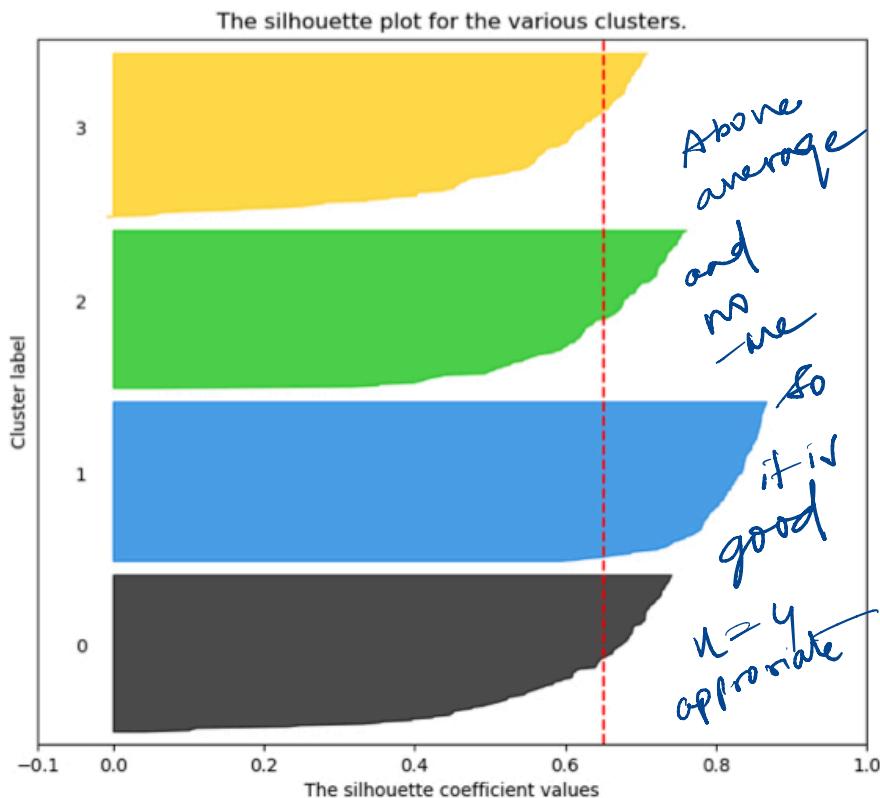
# Silhouette Plot

Silhouette analysis for KMeans clustering on sample data with  $n\_clusters = 3$



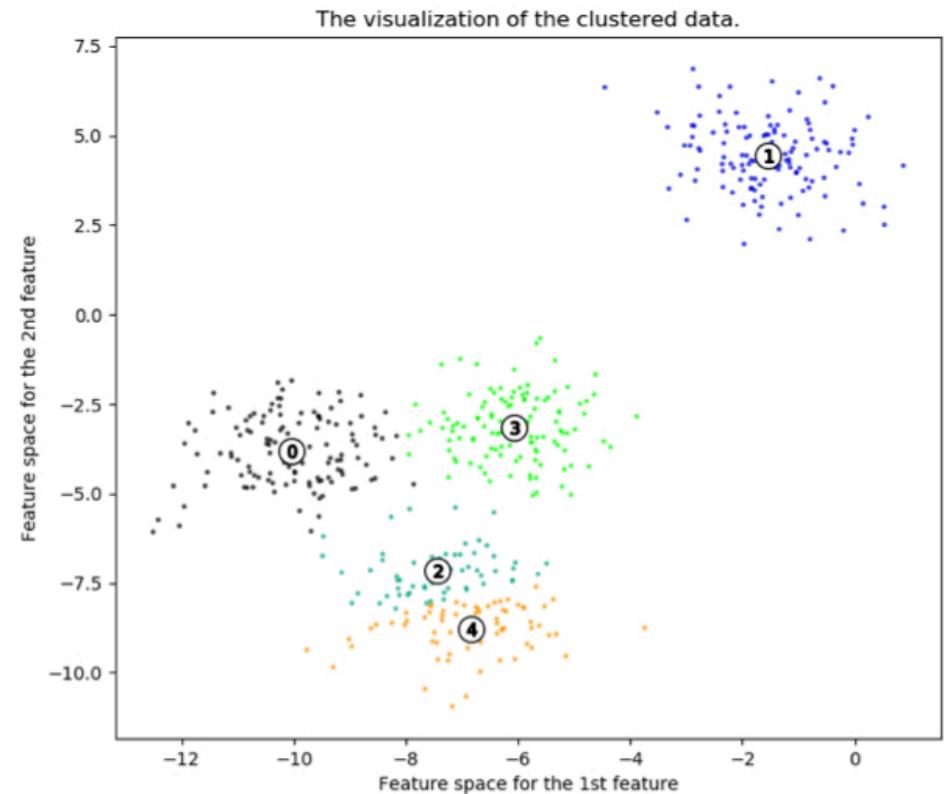
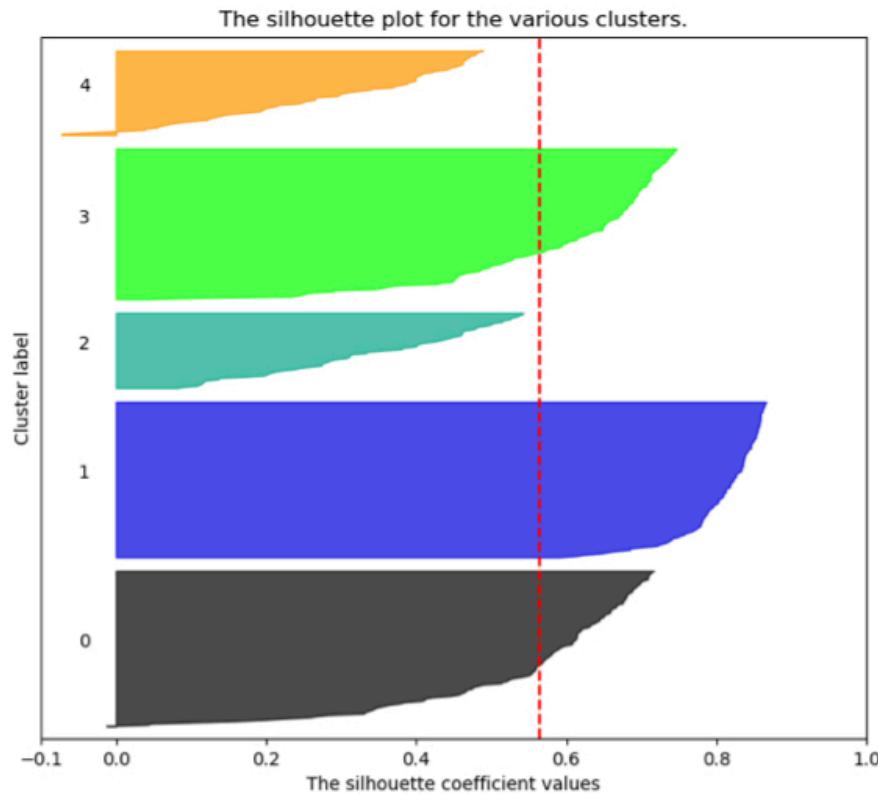
# Silhouette Plot

Silhouette analysis for KMeans clustering on sample data with  $n\_clusters = 4$



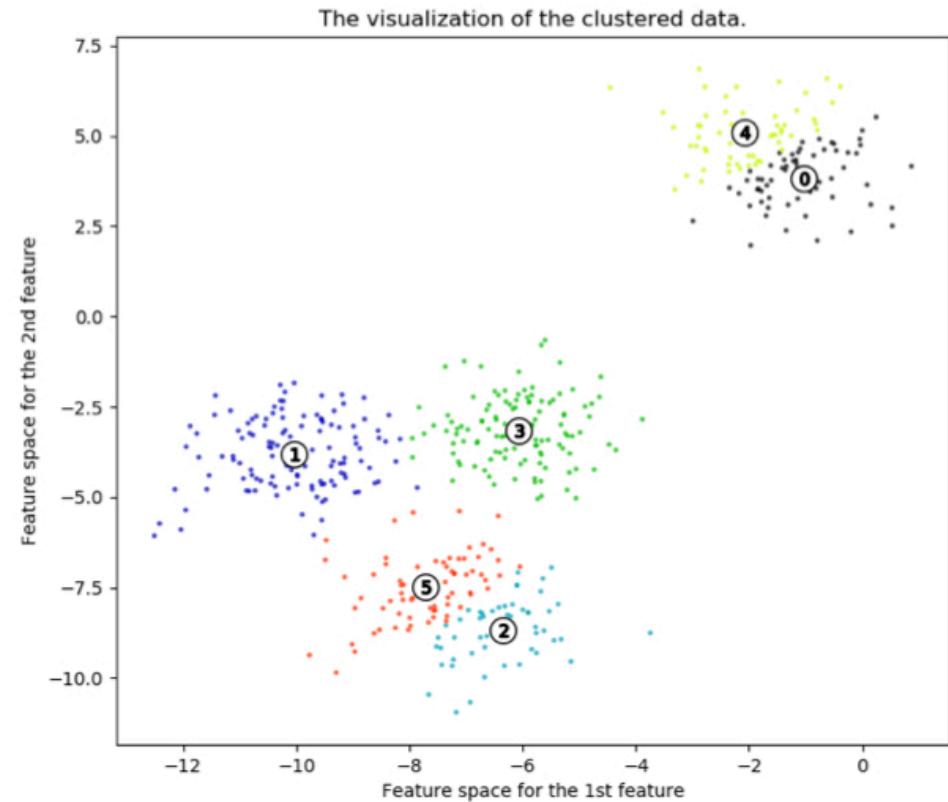
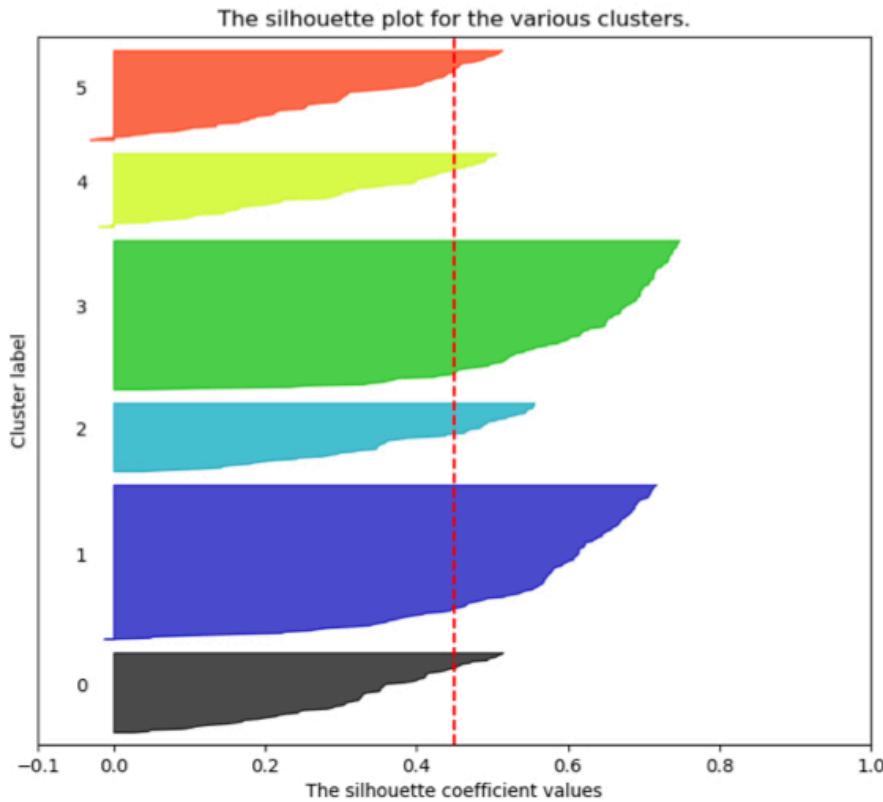
# Silhouette Plot

**Silhouette analysis for KMeans clustering on sample data with n\_clusters = 5**



# Silhouette Plot

**Silhouette analysis for KMeans clustering on sample data with n\_clusters = 6**



# Silhouette Plot

- The silhouette plot shows that k= 3, 5 and 6 are a bad pick for the given data due to the presence of clusters with below average silhouette scores and also due to wide fluctuations in the size of the silhouette plots.
- Silhouette analysis is more ambivalent in deciding between 2 and 4.

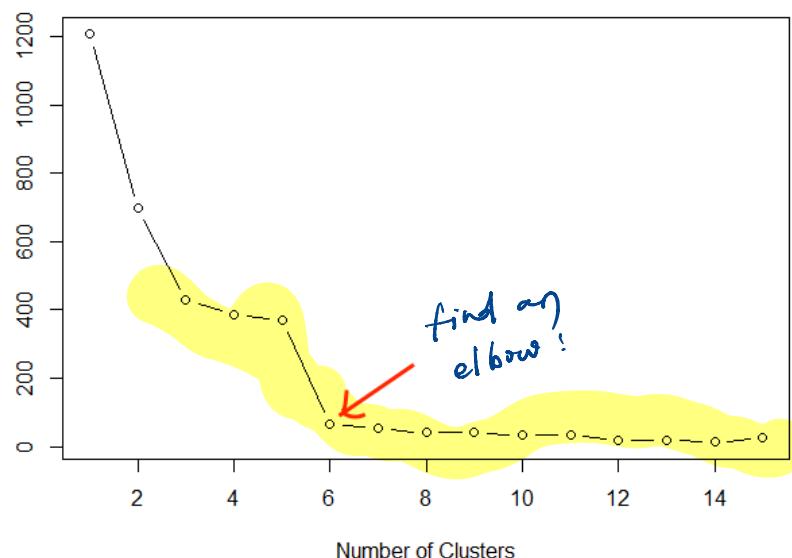
# Cross Validation

- The data is partitioned into  $v$  folds.
- Each of the folds is then held out at turn as a test set, a clustering model computed on the other  $v - 1$  training sets
- The value of an objective function is calculated for the test set.
- Example of objective function: the sum of the squared distances to the centroids for  $k$ -means

$$\sum_{k=1}^K \frac{1}{|C_k|} \|x_i - \underline{x}(k)\|_2^2$$

# Cross Validation

- These  $v$  values are calculated and averaged for each alternative number of clusters  $c$ , and the cluster number selected such that further increase in number of clusters leads to only a small reduction in the objective function (scree plots)



# More

- <https://stackoverflow.com/questions/15376075/cluster-analysis-in-r-determine-the-optimal-number-of-clusters/15376462#15376462>

# More

- <https://stats.stackexchange.com/questions/3685/where-to-cut-a-dendrogram>

# Once again, it really is a hard problem

## Background

### Just How Many Clusters are there in the Galaxy Data?

- ▶ Galaxy Data from Postman *et al.* (1986): measurements of velocities in  $10^3$  km/sec of 82 galaxies from a survey of the Corona Borealis region.
- ▶ Roeder (1990): at least 3, no more than 7 modes (Confidence set)
- ▶ Others are in consensus

|       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|
| 9172  | 9350  | 9483  | 9558  | 9775  | 10227 |
| 10406 | 16084 | 16170 | 18419 | 18552 | 18600 |
| 18927 | 19052 | 19070 | 19330 | 19343 | 19349 |
| 19440 | 19473 | 19529 | 19541 | 19547 | 19663 |
| 19846 | 19856 | 19863 | 19914 | 19918 | 19973 |
| 19989 | 20166 | 20175 | 20179 | 20196 | 20215 |
| 20221 | 20415 | 20629 | 20795 | 20821 | 20846 |
| 20875 | 20986 | 21137 | 21492 | 21701 | 21814 |
| 21921 | 21960 | 22185 | 22209 | 22242 | 22249 |
| 22314 | 22374 | 22495 | 22746 | 22747 | 22888 |
| 22914 | 23206 | 23241 | 23263 | 23484 | 23538 |
| 23542 | 23666 | 23706 | 23711 | 24129 | 24285 |
| 24289 | 24366 | 24717 | 24990 | 25633 | 26960 |
| 26995 | 32065 | 32789 | 34279 |       |       |

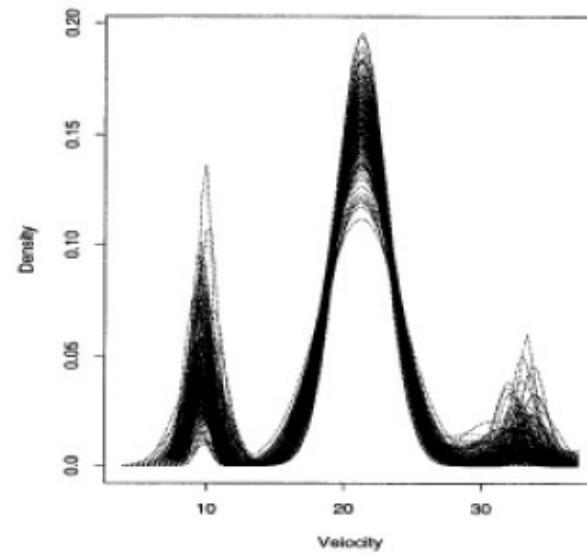
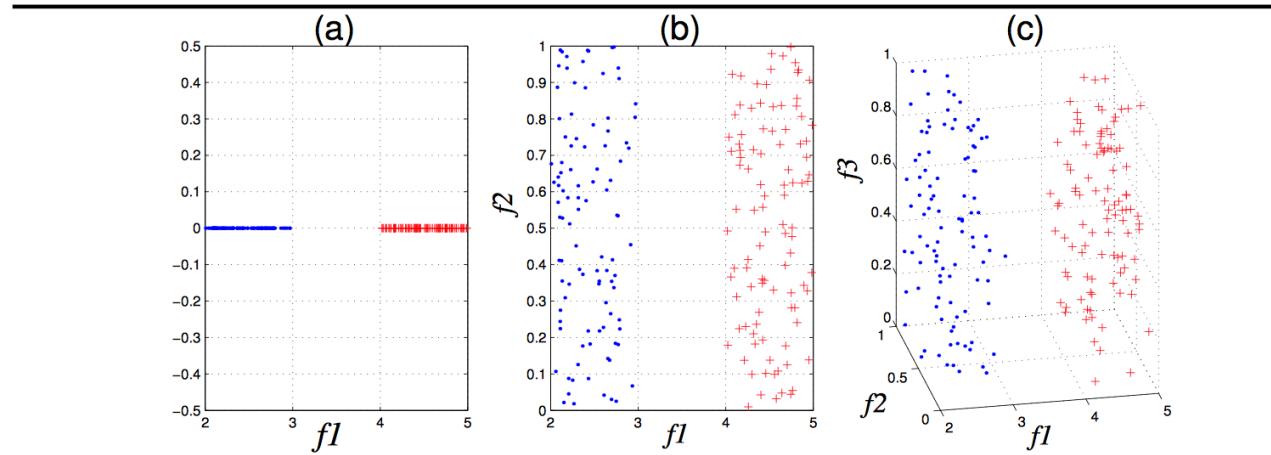


Figure 1. Densities Obtained From the Markov Chain Monte Carlo Sampler Using the Astronomy Data From Roeder (1992).

- ▶ Histogram from Roeder and Wasserman (1997)

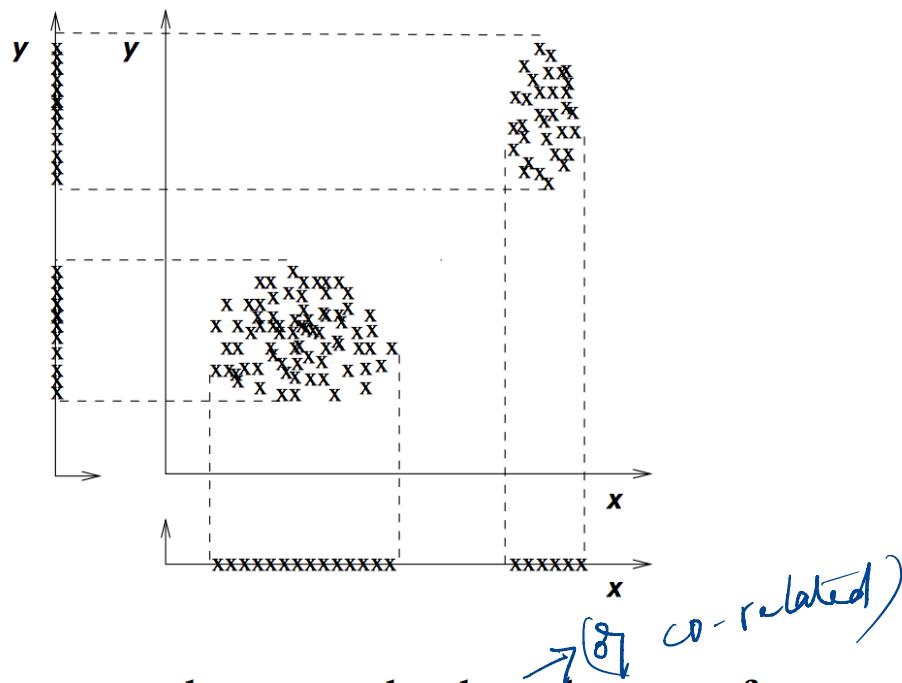
# Variable Selection for Clustering

Venkatesh  
Venkatesh  
Venkatesh  
Venkatesh



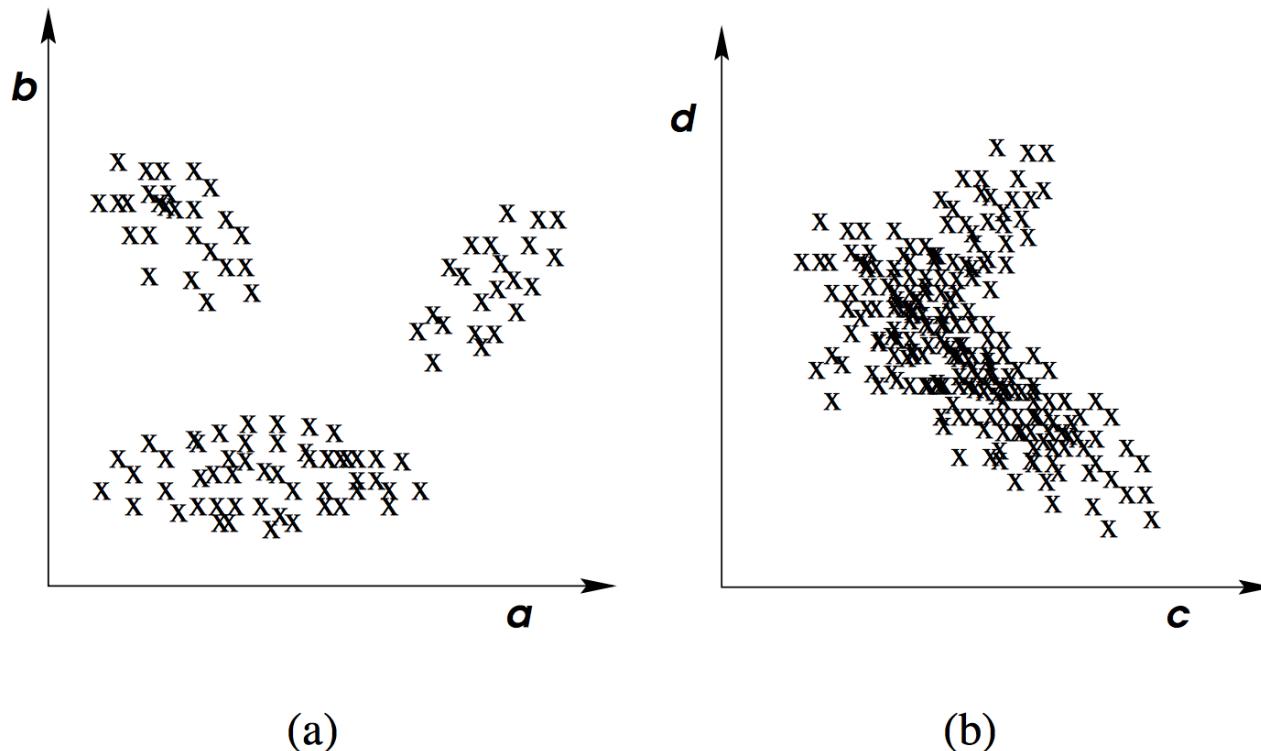
Feature  $f_1$  is relevant while  $f_2$  and  $f_3$  are irrelevant. We are able to distinguish the two clusters from  $f_1$  only. Thus, removing  $f_2$  and  $f_3$  will not effect the accuracy of clustering.

# Variable Selection for Clustering



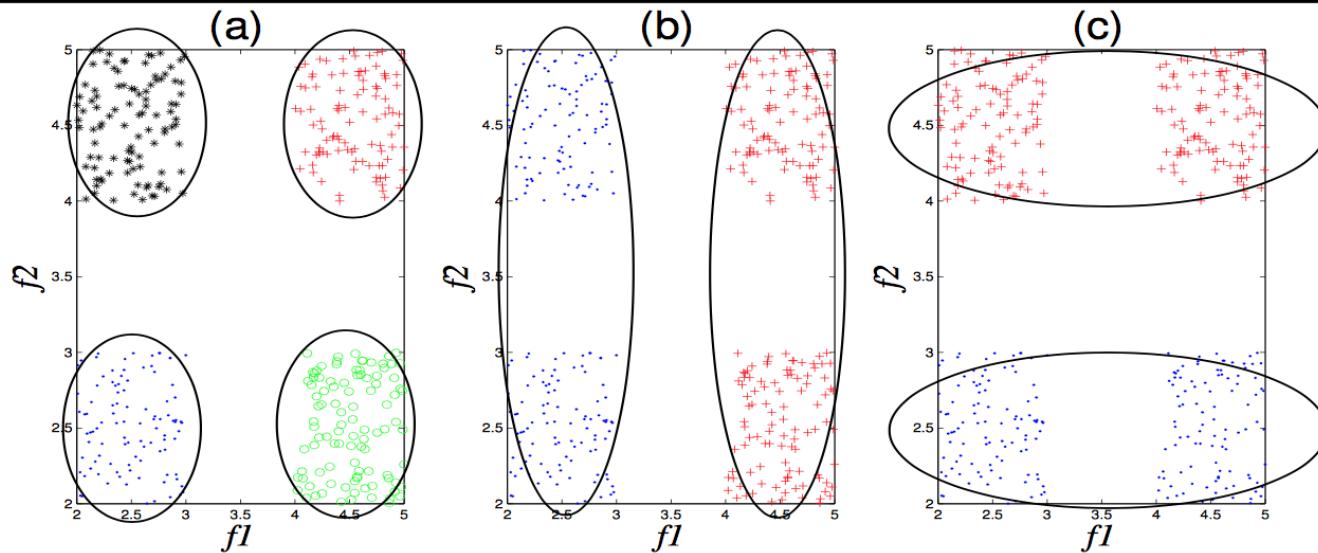
In this example, features  $x$  and  $y$  are redundant, because feature  $x$  provides the same information as feature  $y$  with regard to discriminating the two clusters.

# Variable Selection for Clustering



A more complex example. Figure a is the scatterplot of the data on features  $a$  and  $b$ . Figure b is the scatterplot of the data on features  $c$  and  $d$ .

# Variable Selection for Clustering



Different sets of features may produce different clustering.

# Variable Selection for Clustering

- The goal of feature selection for unsupervised learning is to find the **smallest feature subset that best uncovers** “interesting <sup>→ subjective</sup> natural” groupings (clusters) from data according to the chosen criterion.

# Subset Selection

- Two paradigms:
  - feature subset selection criteria should be the criteria used for clustering
  - The two criteria need not be the same

# Subset Selection

Searching for the best subset of features,  
we run into a new problem:

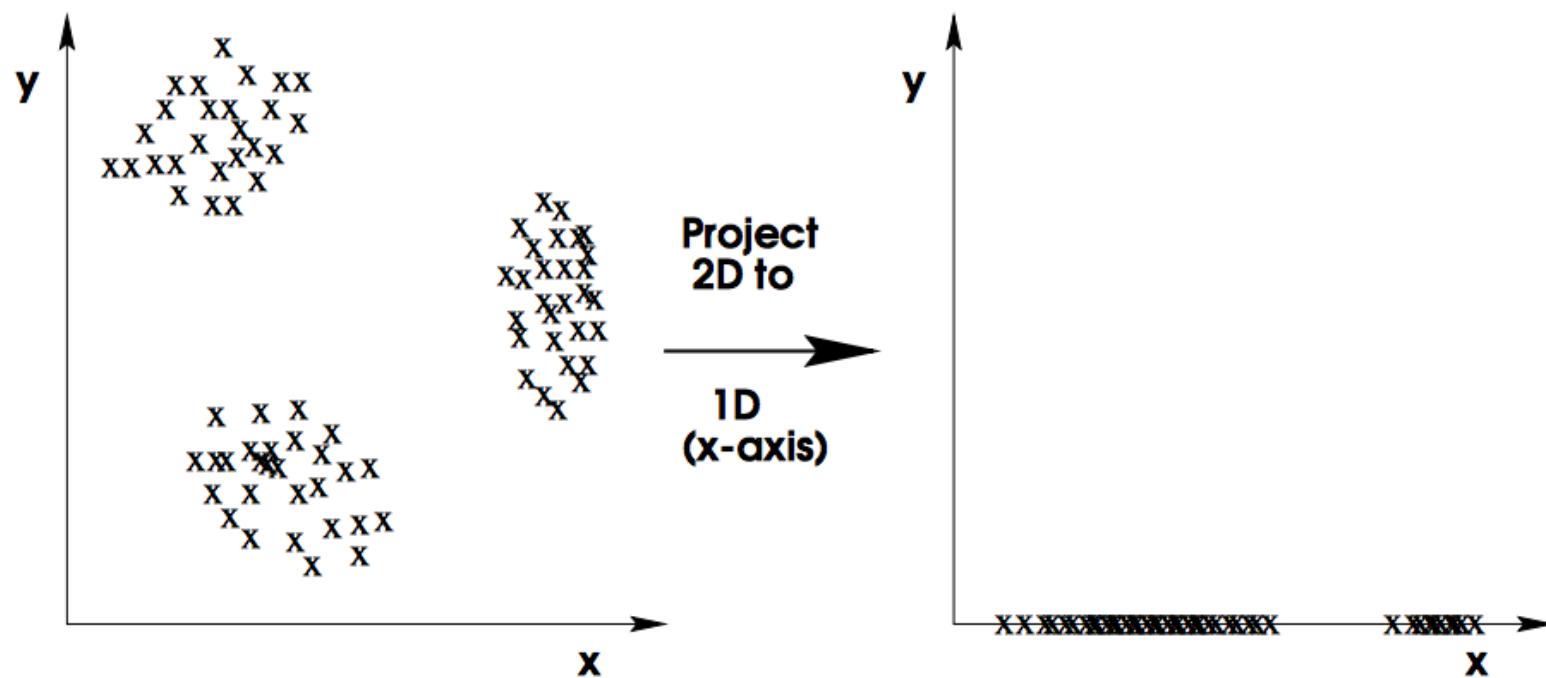
*k depends on the feature subset.*

*K isn't  
fixed  
we need  
to select*

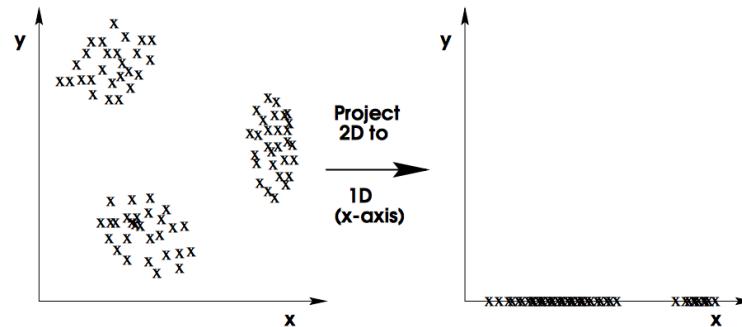
Therefore, in each step of subset selection, **the best k has to be found.**

Many ways are conceivable for best subset selection for clustering. The details are left to the students.

# Subset Selection



# Subset Selection



Two dimensions: three clusters

One-dimension: only two clusters.

A fixed number of clusters for all feature sets does not model the data in the respective subspace correctly.

# K-Means

Large number of k-means variations were proposed to handle feature selection

Most start cluster the data into  $k$  clusters. Then, assign weight to each feature.

The feature that minimizes the within-cluster distance / maximizes between-cluster distance is preferred, hence, gets higher weight.

# Sparse Methods Based on L1 Norm

Witten and Tibshirani formulate clustering using a parameter vector  $w$ , and use L1 penalty and optimization to obtain a sparse set of features.

See

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2930825/pdf/nihms201124.pdf>

(PCA) (PCA) (PCA) (PCA)

# Principal Components Analysis

- PCA produces a low-dimensional representation of a dataset. It finds a sequence of linear combinations of the variables that have maximal variance, and are mutually uncorrelated.
- Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.

# Principal Components Analysis: details

- The *first principal component* of a set of features  $X_1, X_2, \dots, X_p$  is the normalized linear combination of the features

$$Z_1 = \varphi_{11}X_1 + \varphi_{21}X_2 + \dots + \varphi_{p1}X_p$$

that has the largest variance. By normalized, we mean that

$$\|\varphi_1\|_2^2 = 1$$

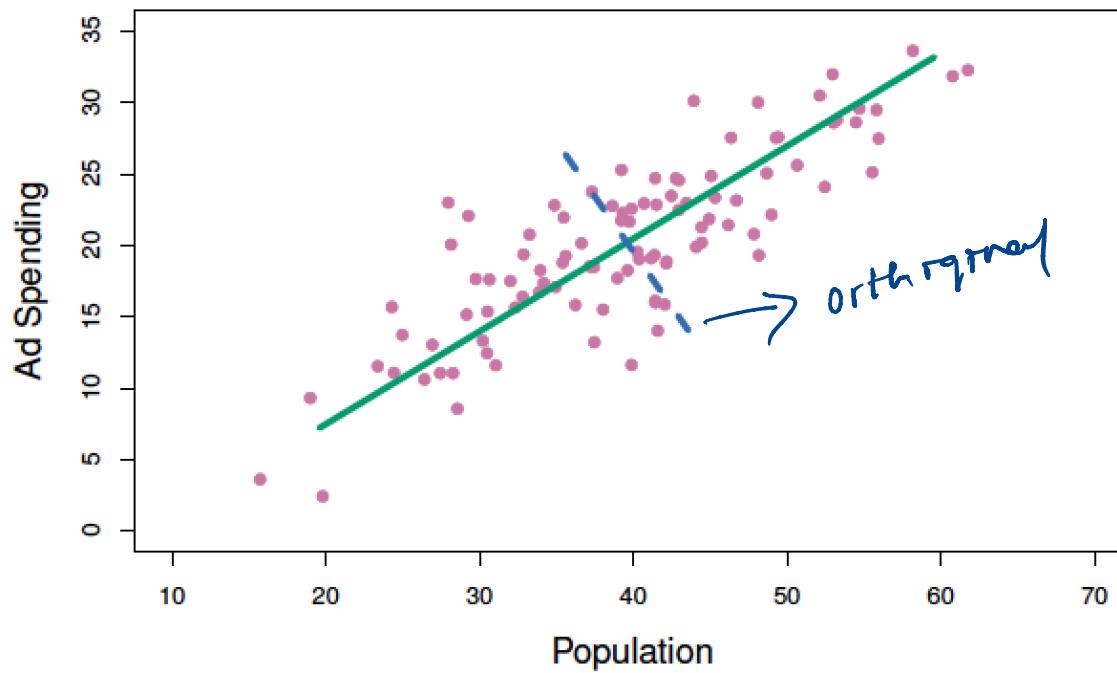
$$\sum_{j=1}^p \varphi_{j1}^2 = 1$$

$$\varphi_1 = (\varphi_{11}, \varphi_{21}, \dots, \varphi_{p1})$$

# Principal Components Analysis: details

- We refer to the elements  $\varphi_{11}, \dots, \varphi_{p1}$  as the loadings of the first principal component; together, the loadings make up the principal component loading vector,  
 $\varphi_1 = (\varphi_{11} \varphi_{21} \dots \varphi_{p1})^T$ .
- We constrain the loadings so that their sum of squares is equal to one, since otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance.

# PCA: example



The population size (**pop**) and ad spending (**ad**) for 100 different cities are shown as purple circles. The green solid line indicates the first principal component direction, and the blue dashed line indicates the second principal component direction.

# Computation of Principal Components

- Suppose we have a  $n \times p$  data set. Since we are only interested in variance, we assume  $\mathbf{X}$  that each of the variables in  $\mathbf{X}$  has been centered to have mean zero (that is, the column means of  $\mathbf{X}$  are zero).
- We then look for the linear combination of the sample feature values of the form  $z_{i1} = \varphi_{11}x_{i1} + \varphi_{21}x_{i2} + \dots + \varphi_{p1}x_{ip}$  for  $i = 1, \dots, n$  that has largest sample variance, subject to the constraint that

$$\sum_{j=1}^p \phi_{j1}^2 = 1$$

|          | $x_1$        | $x_2$        | ... | $x_p$        | $z_i$                   |
|----------|--------------|--------------|-----|--------------|-------------------------|
| data 1   |              |              |     |              | $z_{i1}$                |
| data 2   |              |              |     |              | $z_{i2}$                |
| ⋮        |              |              |     |              | ⋮                       |
| data $n$ |              |              |     |              | $z_{in}$                |
|          | zero<br>mean | zero<br>mean |     | zero<br>mean | $\Rightarrow$ zero mean |

$$Z_{i1} = \varphi_{11}x_{i1} + \varphi_{21}x_{i2} + \dots + \varphi_{p1}x_{ip}$$

Sample variance of  $Z_i = s_{Z_i}$

$$= \frac{1}{n} \sum_{j=1}^n z_{ij}^2$$

$$\bar{Z}_i = 0$$

(sample mean)

Assumption:  $x_j$ 's are zero mean

(Easy to do it: calculate the mean and subtract it from all the data points)

# Computation of Principal Components

- Since each of the  $x_{ij}$  has mean zero, then so does  $z_{i1}$  (for any values of  $\varphi_{j1}$ ). Hence the sample variance of the  $z_{i1}$  can be written as

$$\frac{1}{n} \sum_{i=1}^n z_{i1}^2.$$

# Computation: continued

- Therefore, the first principal component loading vector solves the optimization problem

$$\underset{\phi_{11}, \dots, \phi_{p1}}{\text{maximize}} \frac{1}{n} \sum_{i=1}^n \left( \underbrace{\sum_{j=1}^p \phi_{j1} x_{ij}}_{Z_{i1}} \right)^2 \text{ subject to } \sum_{j=1}^p \phi_{j1}^2 = 1$$

- This problem can be solved via a singular-value decomposition of the matrix  $\mathbf{X}$ , a standard technique in linear algebra.
- We refer to  $Z_1$  as the first principal component, with realized values  $Z_{11}, \dots, Z_{n1}$

*random variable = new feature.*

# Geometry of PCA

- The loading vector  $\varphi_1$  with elements  $\varphi_{11}, \varphi_{21}, \dots, \varphi_{p1}$  defines a direction in feature space along which the data vary the most.
- If we project the  $n$  data points  $x_1, \dots, x_n$  onto this direction, the projected values are the principal component scores  $z_{11}, \dots, z_{n1}$  themselves.

# Further principal components

- The second principal component is the linear combination of  $X_1, \dots, X_p$  that has maximal variance among all linear combinations that are *uncorrelated* with  $Z_1$ .
- The second principal component scores  $Z_{12}, Z_{22}, \dots, Z_{n2}$  take the form

$$Z_{i2} = \varphi_{12}X_{i1} + \varphi_{22}X_{i2} + \dots + \varphi_{p2}X_{ip},$$

where  $\varphi_2$  is the second principal component loading vector, with elements  $\varphi_{12}, \varphi_{22}, \dots, \varphi_{p2}$ .

# Further principal components: continued

- It turns out that constraining  $Z_2$  to be uncorrelated with  $Z_1$  is equivalent to constraining the direction  $\varphi_2$  <sup>→ this also normalized</sup> to be orthogonal (perpendicular) to the direction  $\varphi_1$ . And so on.

# Further principal components: continued

- The principal component directions  $\varphi_1, \varphi_2, \varphi_3, \dots$  are the ordered sequence of right singular vectors of the matrix  $\mathbf{X}$ , and the variances of the components are  $1/n$  times the squares of the singular values. There are at most  $\min(n - 1, p)$  principal components.

$$\text{Var}(z_i) = \frac{1}{n} \tau_i^2 \rightarrow \text{singular values}$$

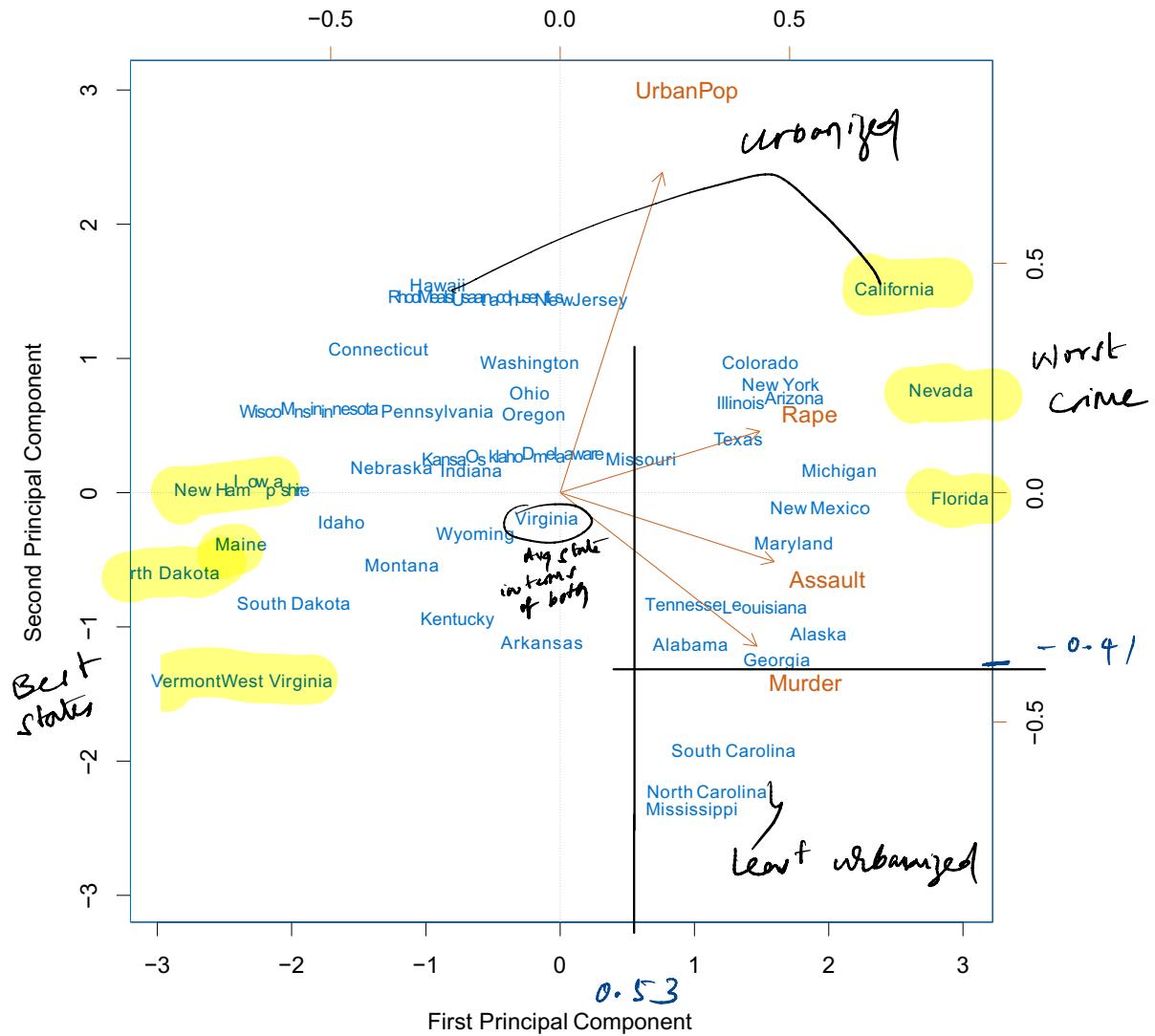
# Illustration

- **USAarrests** data: For each of the fifty states in the United States, the data set contains the number of arrests per 100, 000 residents for each of three crimes: **Assault**, **Murder**, and **Rape**. We also record **UrbanPop** (the percent of the population in each state living in urban areas).

# Illustration

- The principal component score vectors have length  $n = 50$ , and the principal component loading vectors have length  $p = 4$ .
- PCA was performed after standardizing each variable to have mean zero and standard deviation one.

# USAarrests data: PCA plot



# Figure details

The first two principal components for the USArrests data.

- The blue state names represent the scores for the first two principal components.

# PCA loadings

|          | PC1       | PC2        |
|----------|-----------|------------|
| Murder   | 0.5358995 | -0.4181809 |
| Assault  | 0.5831836 | -0.1879856 |
| UrbanPop | 0.2781909 | 0.8728062  |
| Rape     | 0.5434321 | 0.1673186  |

The contribution of each variable in each of the principal components PC1 and PC2 is shown as a vector. For example, Assault is shown as  $[0.58, -0.19]^T$

# Figure details

- The orange arrows indicate the first two principal component loading vectors (with axes on the top and right). For example, the loading for **Rape** on the first component is 0.54, and its loading on the second principal component 0.17 [the word **Rape** is centered at the point (0.54, 0.17)].

# Figure details

- This figure is known as a *biplot*, because it displays both the principal component scores and the principal component loadings.

# Figure details

- The first loading vector places approximately equal weight on **Assault**, **Murder**, and **Rape**, with much less weight on **UrbanPop**.
- Hence this component roughly corresponds to a measure of overall rates of serious crimes.

# Figure details

- The second loading vector places most of its weight on **UrbanPop** and much less weight on the other three features.
- Hence, this component roughly corresponds to the level of urbanization of the state.

# Figure details

- The crime-related variables (**Murder**, **Assault**, and **Rape**) are located close to each other
- The **UrbanPop** variable is far from the other three.

# Figure details

- The crime-related variables are correlated with each other: states with high murder rates tend to have high assault and rape rates
- The **UrbanPop** variable is less correlated with the other three.

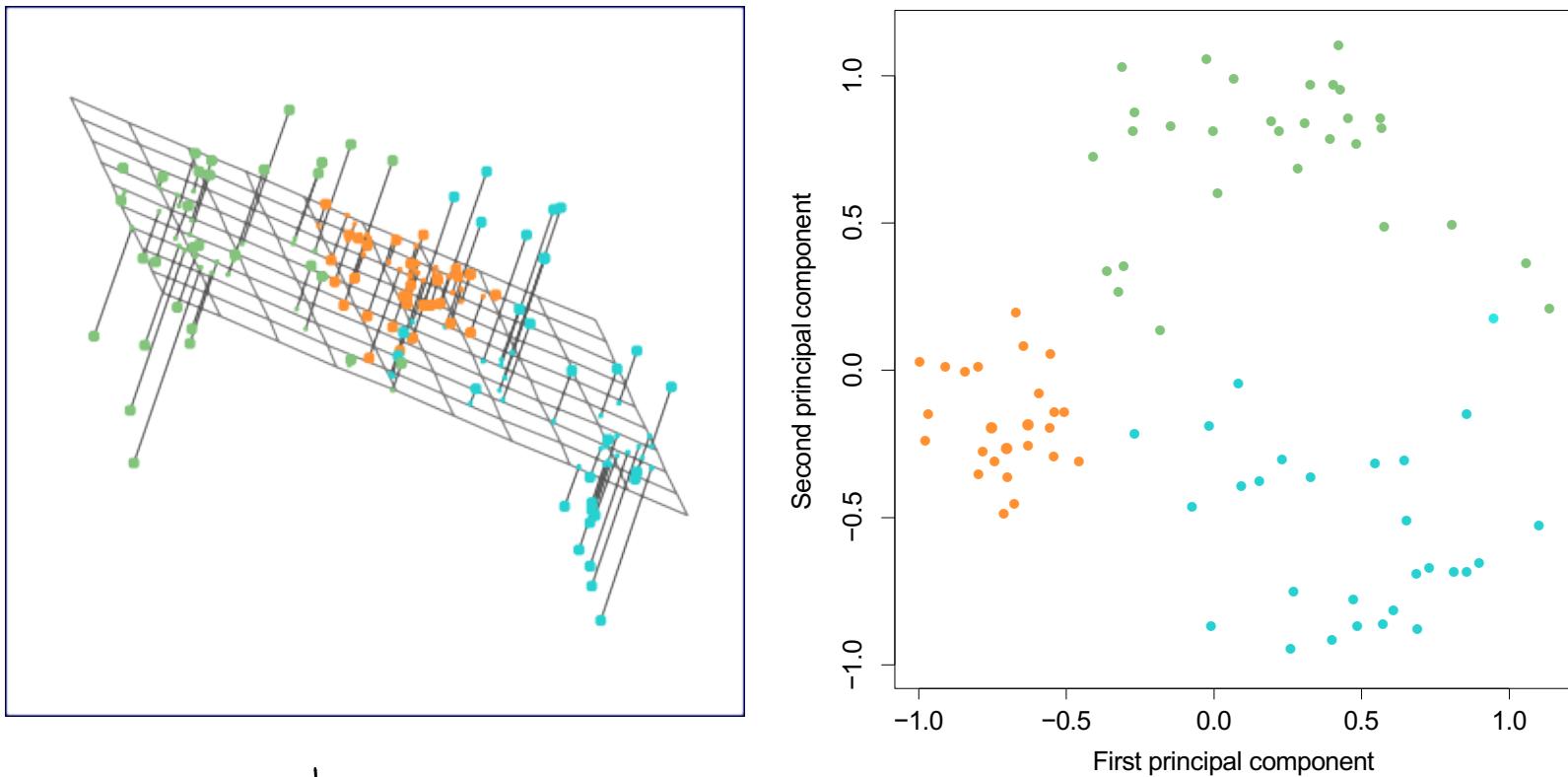
# Figure details

- States with large positive scores on the first component, such as **California**, **Nevada** and **Florida**, have high crime rates
- States like **North Dakota**, with negative scores on the first component, have low crime rates.

# Figure details

- California has a high score on the second component, hence a high level of urbanization
- The opposite is true for states like Mississippi.
- States close to zero on both components, such as Virginia, have approximately average levels of both crime and urbanization.

# Another Interpretation of Principal Components



The first ~~three~~<sup>two</sup> principal components of a data set span the ~~three~~<sup>two</sup>-dimensional hyperplane that is closest to the  $n$  observations,

# PCA find the hyperplane closest to the observations

- The first principal component loading vector has a very special property: it defines the line in  $p$ -dimensional space that is *closest* to the  $n$  observations (using average squared Euclidean distance as a measure of closeness)

# PCA find the hyperplane closest to the observations

- The notion of principal components as the dimensions that are closest to the  $n$  observations extends beyond just the first principal component.
- For instance, the first two principal components of a data set span the plane that is closest to the  $n$  observations, in terms of average squared Euclidean distance.

# Scaling of the variables matters

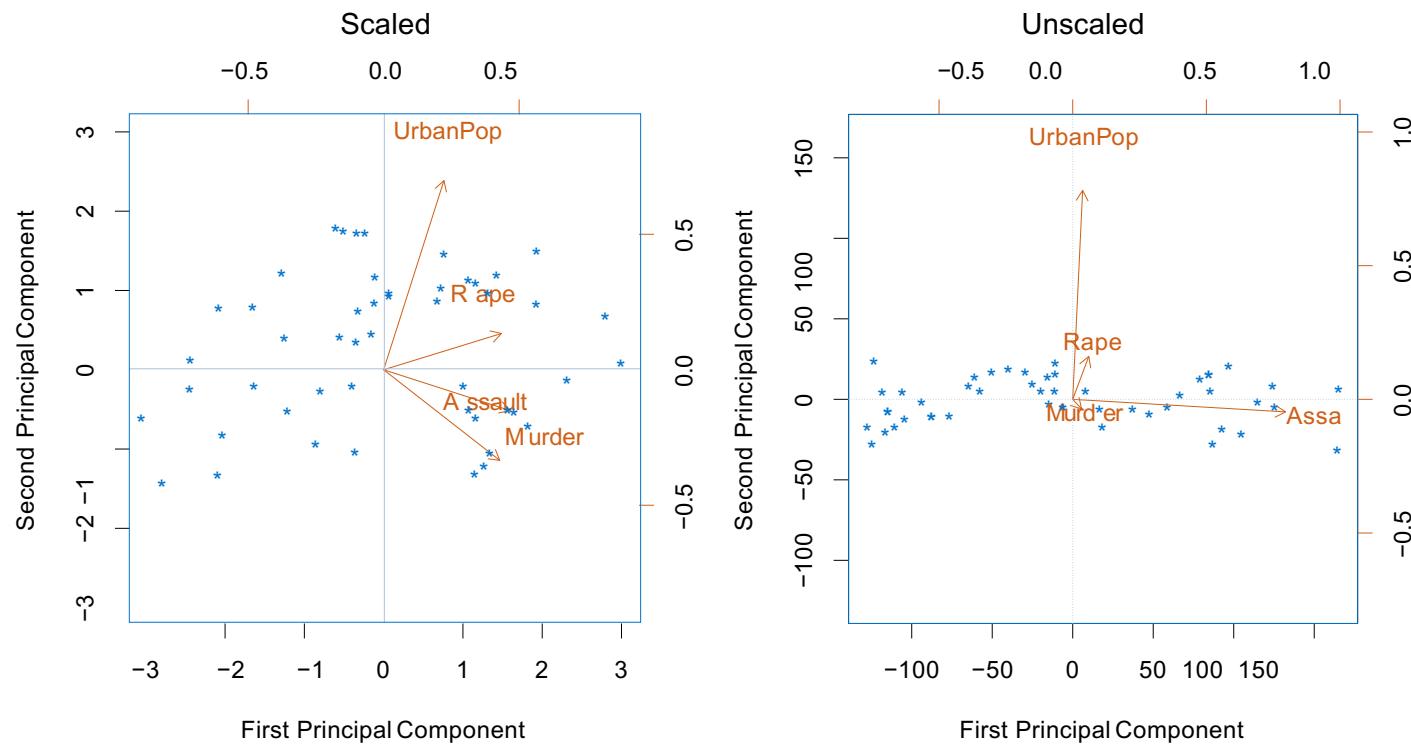
- Murder, Rape, Assault, and UrbanPop have variance 18. 97, 87. 73, 6945. 16, and 209. 5, respectively.
- In PCA on the unscaled variables, the first principal component loading vector will have a very large loading for Assault

# Scaling of the variables matters

- This is simply a consequence of the scales of the variables.
  - For instance, if **Assault** were measured in units of the number of occurrences per 100 people (rather than number of occurrences per 100,000 people), then this would amount to dividing all of the elements of that variable by 1,000. Then the variance of the variable would be tiny

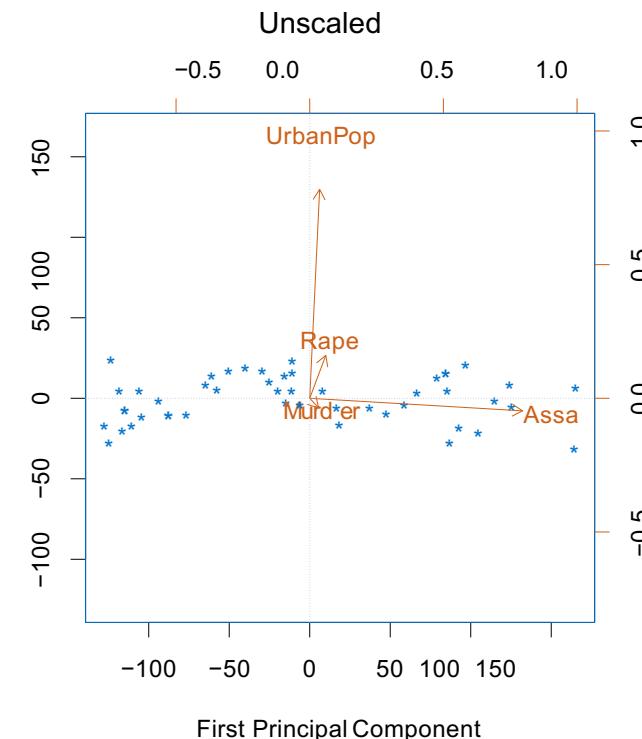
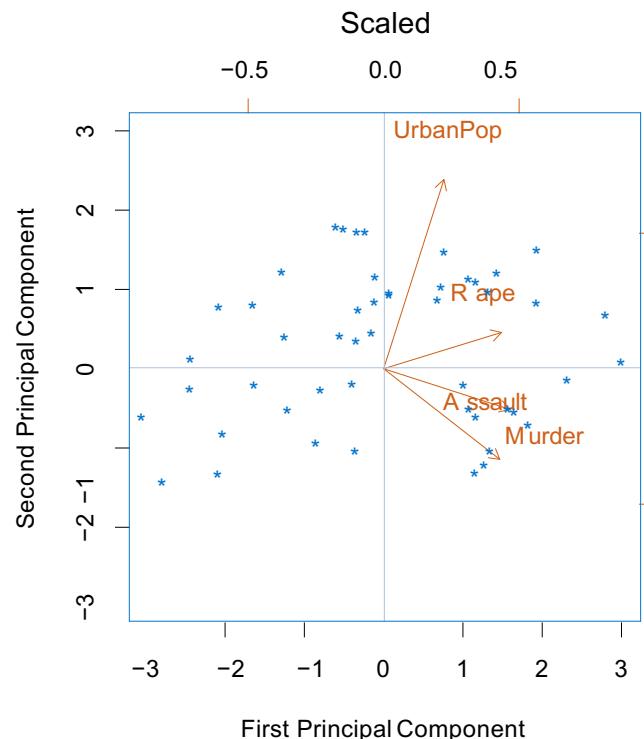
# Scaling of the variables matters

- It is undesirable for the principal components obtained to depend on an arbitrary choice of scaling
- We typically scale each variable to have standard deviation one before we perform PCA.



# Scaling of the variables matters

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended.
- If they are in the same units, you might or might not scale the variables.



$$\text{max \# of PCs} = \min(n-1, d)$$

# Proportion Variance Explained

- We can now ask a natural question:  
how much of the information in a given  
data set is lost by projecting the  
observations onto the first few  
principal components?
- That is, how much of the variance in  
the data is not contained in the first  
few principal components?

# Proportion Variance Explained

- To understand the strength of each component, we are interested in knowing the proportion of variance explained (PVE) by each one.
- The *total variance* present in a data set (assuming that the variables have been centered to have mean zero) is defined as

$$\sum_{j=1}^p \text{Var}(X_j) = \sum_{j=1}^p \frac{1}{n} \sum_{i=1}^n x_{ij}^2$$

# Proportion Variance Explained

The variance explained by the  $m$ th principal component is



$$\text{Var}(Z_m) = \frac{1}{n} \sum_{i=1}^n z_{im}^2$$

- It can be shown that

$$\sum_{j=1}^p \text{Var}(X_j) = \sum_{m=1}^M \text{Var}(Z_m)$$

with  $M = \min(n - 1, p)$ .

# Proportion Variance Explained: continued

- Therefore, the PVE of the  $m^{\text{th}}$  principal component is given by the positive quantity between 0 and 1

$$\frac{\text{Var}(z_m)}{\sum_{j=1}^p \text{Var}(x_j)} = \frac{n \sum_{i=1}^n z_{im}^2}{\sum_{j=1}^p n \sum_{i=1}^n x_{ij}^2}$$

Principal Component

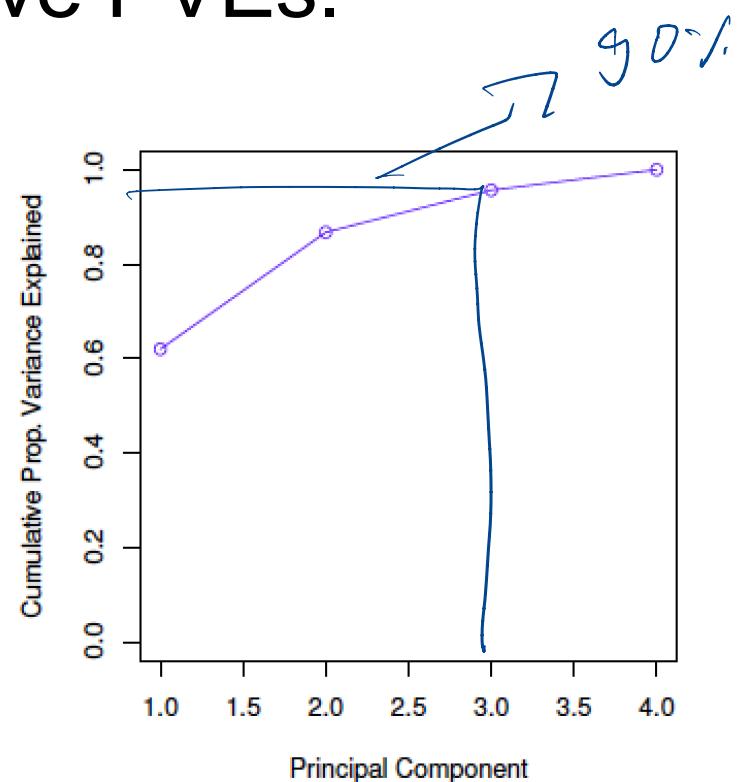
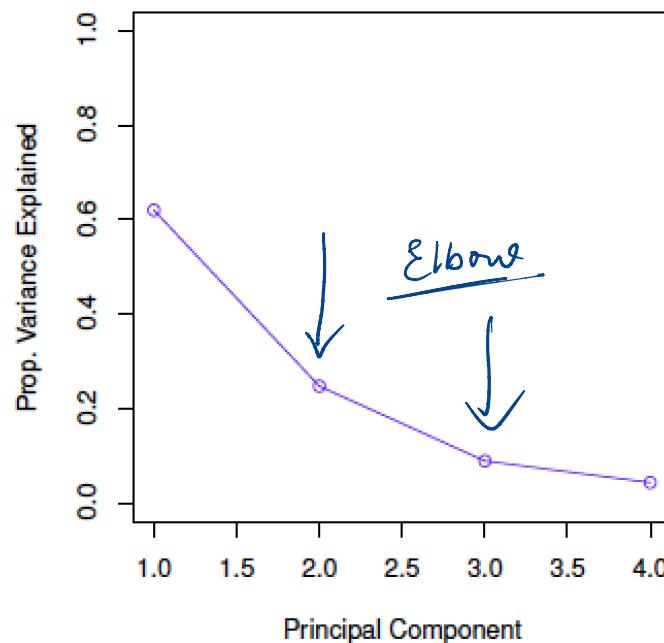
Principal Component

$$\sum_{m=1}^M \text{Var}_m = \frac{\sum_{m=1}^M \text{Var}(2m)}{\sum_{j=1}^P \text{Var}(x_j)} = 1$$

thus will be 1.

# Proportion Variance Explained: continued

- The PVEs sum to one. We sometimes display the cumulative PVEs.



Elbow = The # of PCs that yield 90% of the total variance.

# How many principal components should we use?

If we use principal components as a summary of our data, how many components are sufficient?

- No simple answer to this question, as cross-validation is not available for this purpose.
  - *Why not?*

# How many principal components should we use?

- When could we use cross-validation to select the number of components?  
*When we use PCA as pre-processing for supervised learning*
  - the “scree plot” on the previous slide can be used as a guide: we look for an “elbow”.

# How many principal components should we use?

- If we compute principal components in a supervised analysis, then we can treat the number of principal component score vectors to be used as a **tuning parameter** to be selected via **cross-validation**

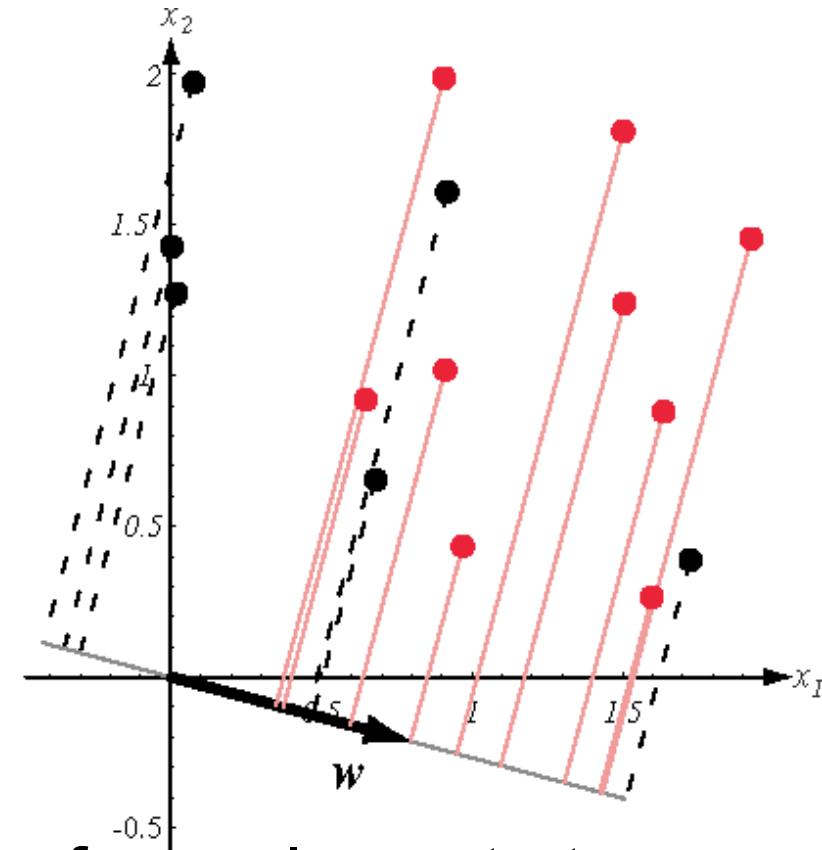
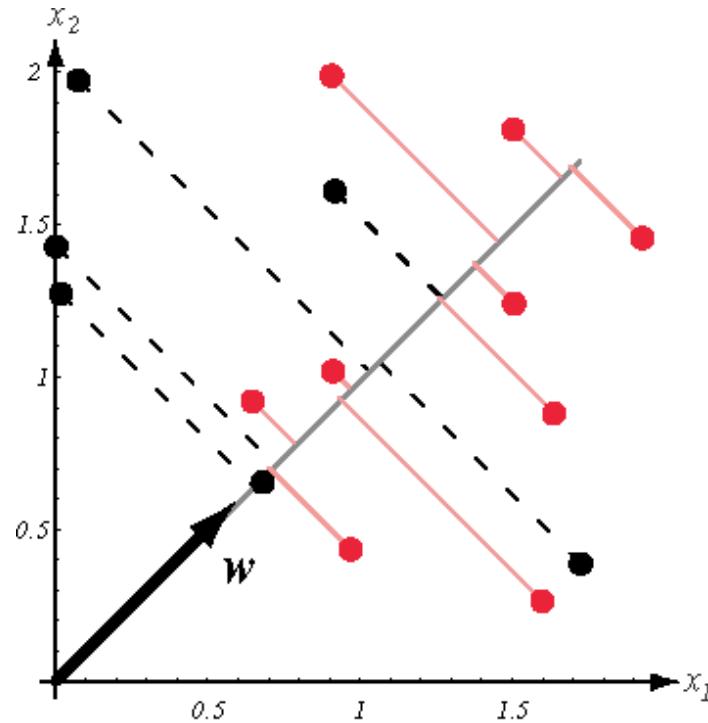
# PCA vs Clustering

- PCA looks for a low-dimensional representation of the observations that explains a good fraction of the variance.
- Clustering looks for homogeneous subgroups among the observations.

# Fisher's Linear Discriminant Analysis

- Whereas PCA seeks directions that are efficient for representation, discriminant analysis seeks directions that are efficient for discrimination.
- It is in some sense a supervised version of PCA.

# Fisher's Linear Discriminant Analysis



Projection of the same set of samples onto two different lines in the directions marked as  $w$ . The figure on the right shows greater separation between the red and black projected points.

## L DA

# Fisher's Linear Discriminant Analysis

(Not Bayesian LDA)

- Given  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^p$  divided into two subsets  $\mathcal{D}_1$  and  $\mathcal{D}_2$  corresponding to the classes 1 and 2, respectively, the goal is to find a projection onto a line defined as

features created

by Fisher's  
LDA

$$y = \beta^T \mathbf{x} = \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

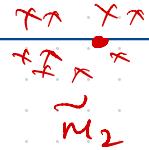
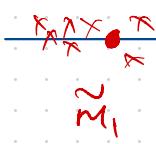
where the points corresponding to  $\mathcal{D}_1$  and  $\mathcal{D}_2$  are well separated.

# Fisher's Linear Discriminant Analysis

- This is called the *Fisher's linear discriminant* with the geometric interpretation that the best projection makes the difference between the means as large as possible relative to the variance.

$$\text{members of } D_1 \rightarrow \beta^T \underline{x} \quad \tilde{m}_1 \quad \tilde{s}_1^2$$

$$\text{members of } D_2 \rightarrow \beta^T \underline{x} \quad \tilde{m}_2 \quad \tilde{s}_2^2$$



$|\tilde{m}_1 - \tilde{m}_2|$  large

$\tilde{s}_1^2, \tilde{s}_2^2$  small

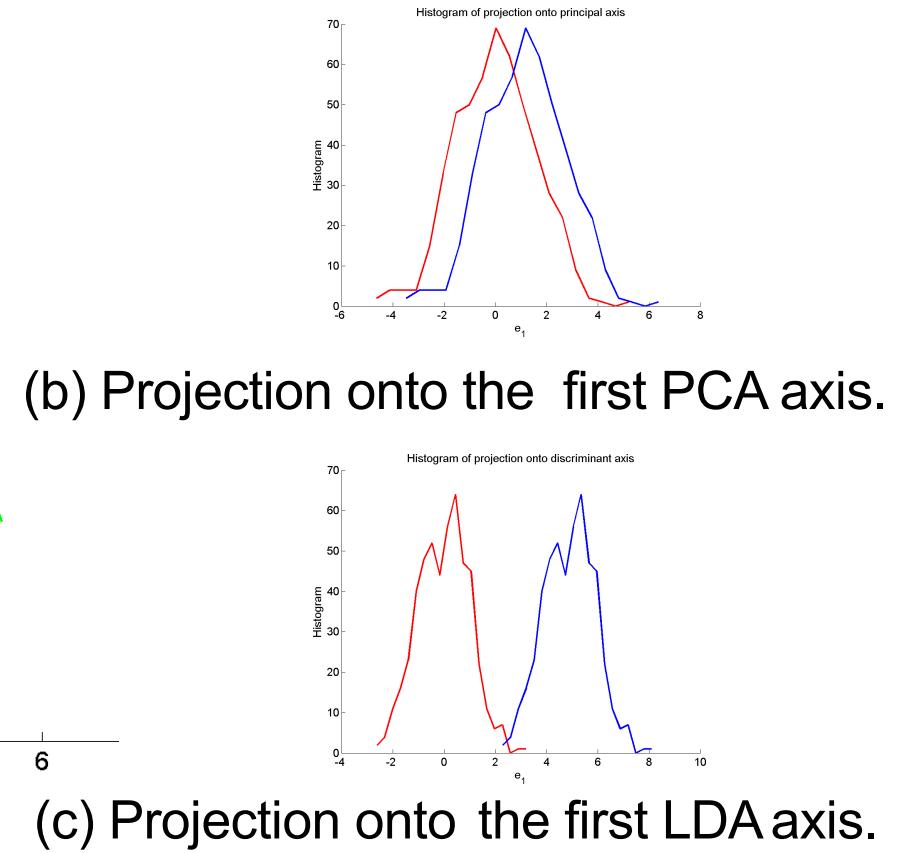
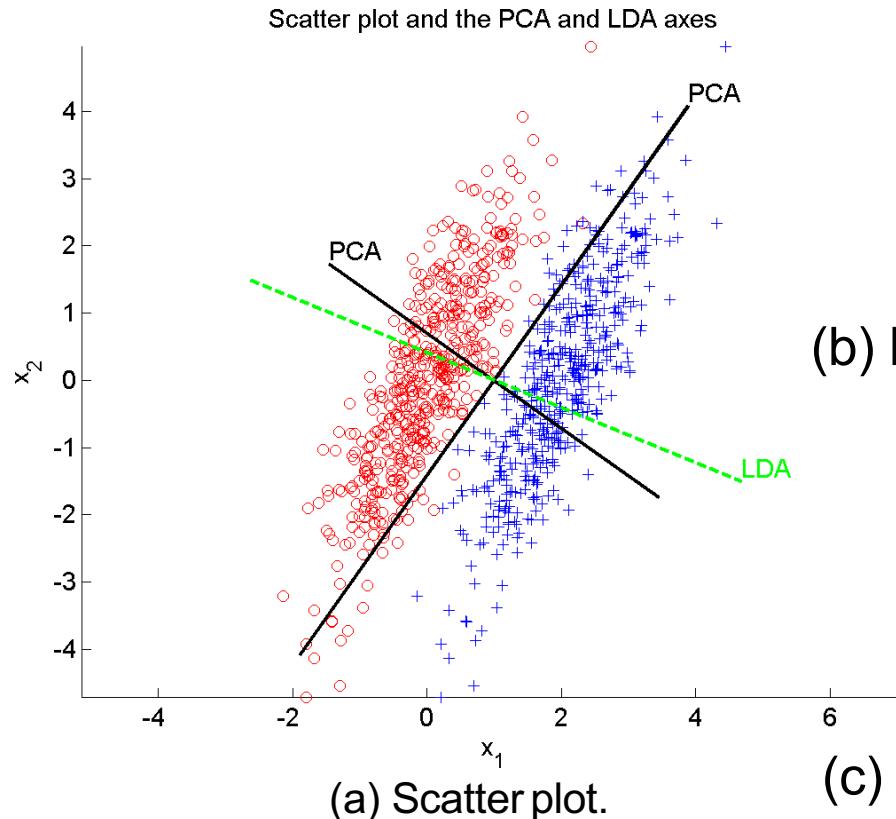
maximize

$$f(\beta) = \frac{|\tilde{m}_1 - \tilde{m}_2|}{\tilde{s}_1^2 + \tilde{s}_2^2} \rightarrow \begin{matrix} \text{maximize} \\ \text{minimize} \end{matrix}$$

# Fisher's Linear Discriminant Analysis

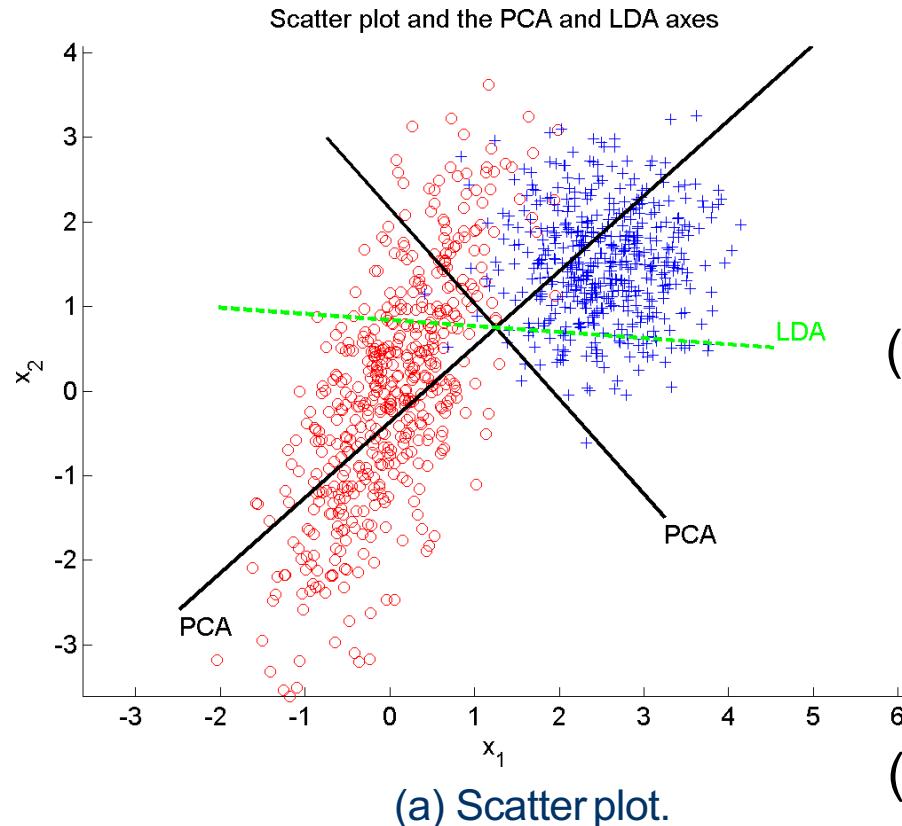
- Once the transformation from the  $p$ -dimensional original feature space to a lower dimensional subspace is done using PCA or Fisher's LDA, classification methods can be used to train pertinent classifiers.

# Fisher's Linear Discriminant Analysis

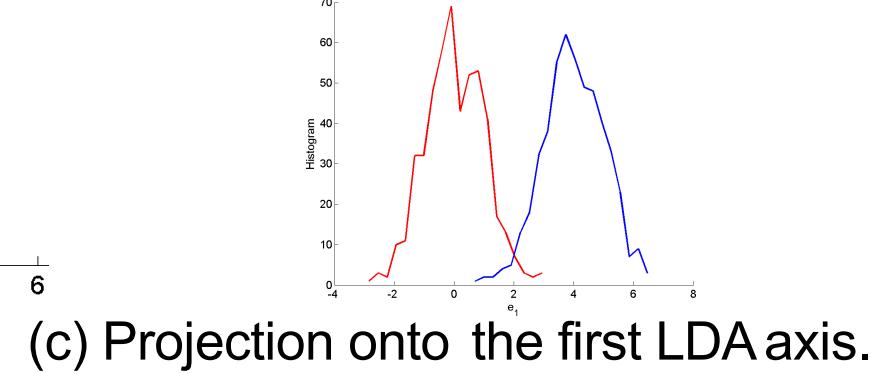
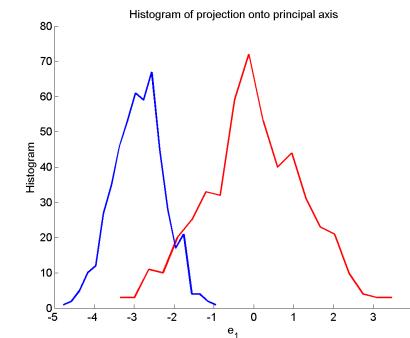


Scatter plot and the PCA and LDA axes for a bivariate sample with two classes. Histogram of the projection onto the first LDA axis shows better separation than the projection onto the first PCA axis.

# Fisher's Linear Discriminant Analysis



(b) Projection onto the first PCA axis.



(c) Projection onto the first LDA axis.

Scatter plot and the PCA and LDA axes for a bivariate sample with two classes. Histogram of the projection onto the first LDA axis shows better separation than the projection onto the first PCA axis.

# Conclusions

- *Unsupervised learning* is important for understanding the variation and grouping structure of a set of unlabeled data, and can be a useful pre-processor for supervised learning
- It is intrinsically more difficult than *supervised learning* because there is no gold standard (like an outcome variable) and no single objective (like test set accuracy)

# Conclusions

- It is an active field of research, with many recently developed tools such as *self-organizing maps, independent components analysis and spectral clustering.*

See *The Elements of Statistical Learning*, chapter 14.

Data visualization → t-SNE

# Appendix 1

## Novelty/Anomaly/Outlier Detection

Out of distribution

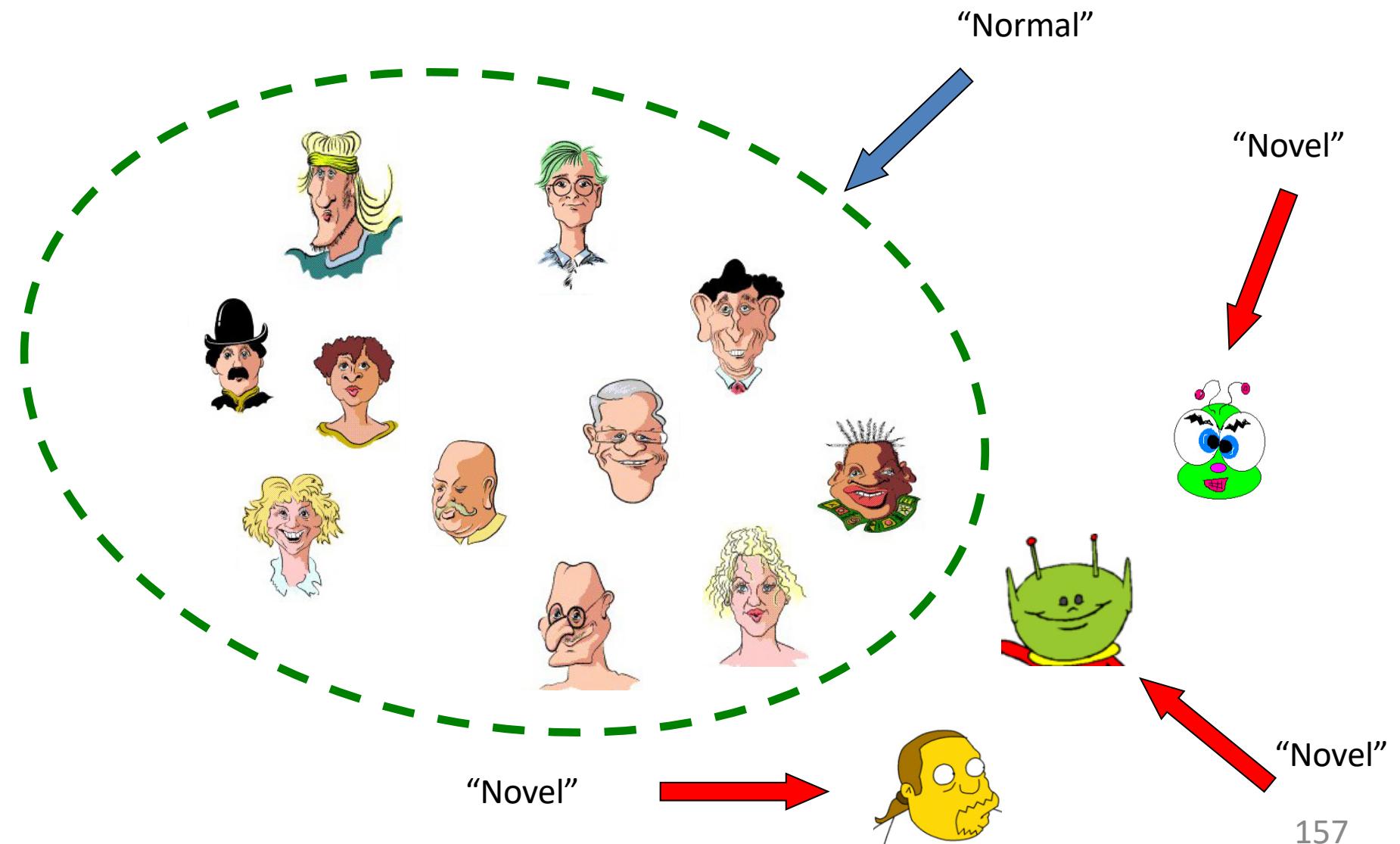
Detection

# Novelty Detection is

with deep learning  
we can have a  
supervised  
formulation

- An unsupervised learning problem (data unlabeled)
- About the identification of new or unknown data or signal that a machine learning system is not aware of during training

# Example 1



So what's seems to be the problem?

It's a 2-Class problem.

“Normal vs. “Novel”

**Wrong!**

# The Problem is

- That “All positive examples are alike but each negative example is negative in its own way”.

## Example 2

Suppose we want to build a classifier that recognizes web pages about “jigsaw puzzles”.

How can we collect a training data?

We can surf the web and pretty easily assemble a sample to be our collection of **positive examples**.



## Example 2

What about **negative examples** ?

The negative examples are... the rest of the web. That is  
~("pickup sticks web page")

So the **negative examples** come from an unknown #  
of negative classes.

Traditional ML algorithms cannot work with  
such a huge data set in a supervised  
manner. Deep learning can!!

# Applications

Many exist

Intrusion detection

Network security?

Fraud detection

Fault detection

Robotics

Medical diagnosis

E-Commerce

And more...

---

Life long learning machine

- which doesn't stop learning

# Possible Approaches

## Density Estimation:

Estimate a *density* based on training data

Threshold the estimated density for test

points → points far from the mean (center)

Like hypothesis testing

## Quantile Estimation:

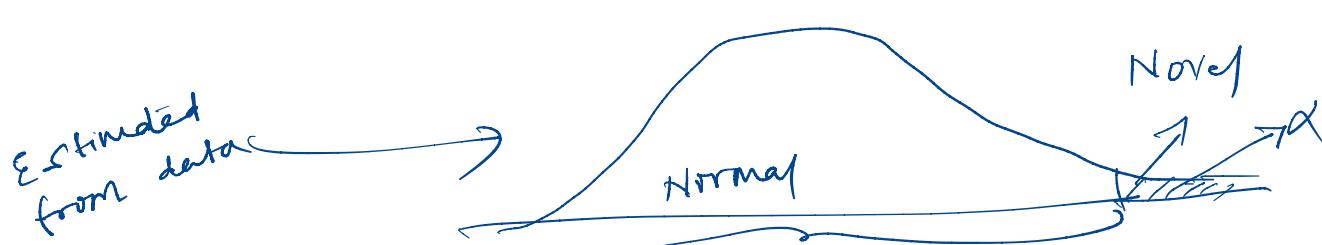
Estimate a *quantile* of the distribution

underlying the training data: for a fixed

constant  $\alpha \in (0,1]$ , attempt to find a

small set  $S$  such that  $\Pr(x \in S) = \alpha$

Check whether test points are inside or  
outside  $S$



# One Class Support Vector Machine (OCSVM) Algorithm

Maps input data into a **high dimensional feature space via kernels**

Iteratively finds the maximal margin in the hyperplane which best separates the training data from the origin

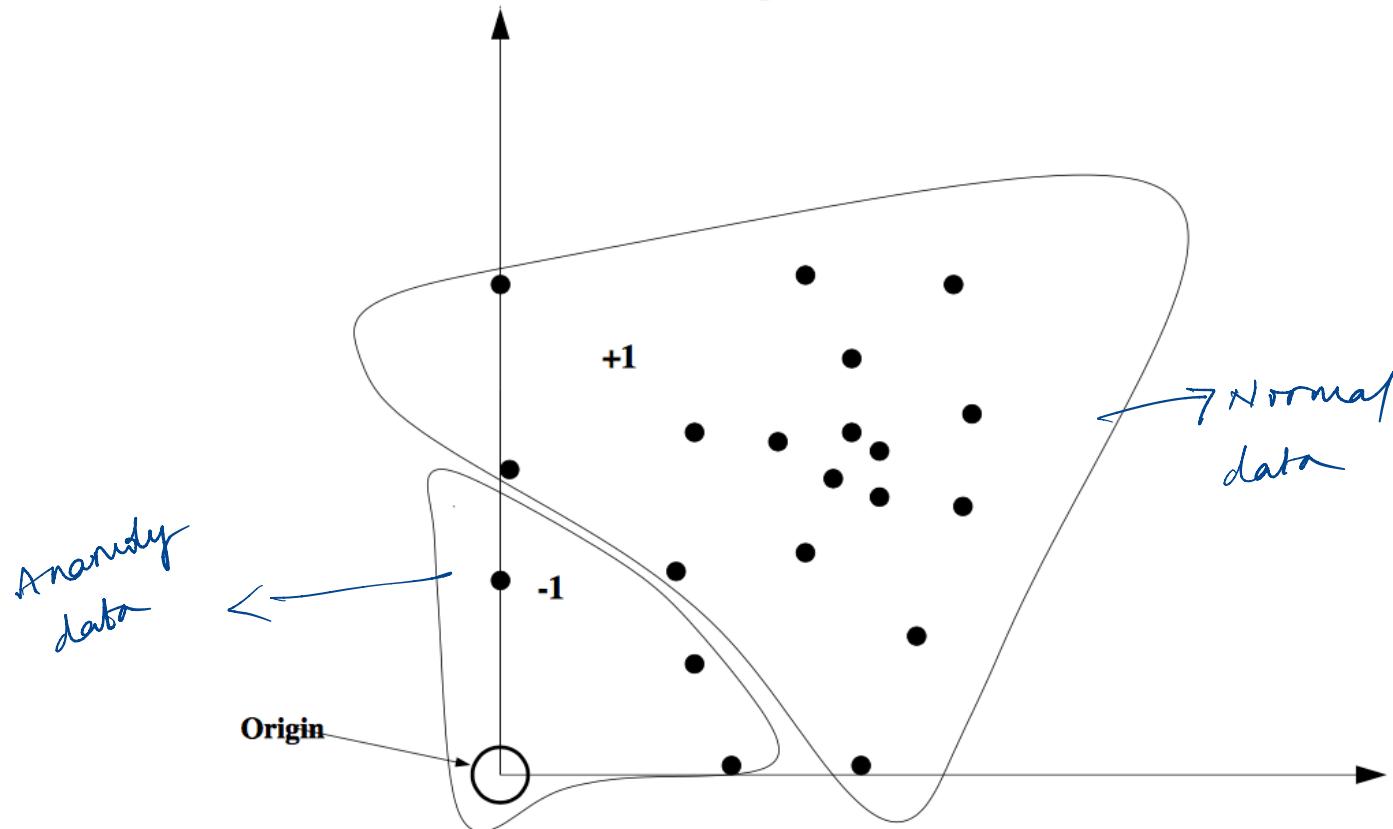
Solves optimization problem to find rule  $f$  with maximal margin

$$f(\mathbf{x}) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$

If  $f(\mathbf{x}) < 0$ , label  $\mathbf{x}$  as anomalous

# OCSVM

Figure 1: One-class SVM



One-Class SVM Classifier. The origin is the only original member of the second class.

# Efficiency

## Complexity of OCSVM

Time:  $O(pL^3)$

Space:  $O(p(L+T))$

$p$  – number of dimensions

$L$  – number of records in training dataset

$T$  – number of records in test dataset

# Lifelong Machine Learning

Classification (Traditional)

$$\text{classes} = \{1, 2, \dots, k\}$$

Build  $c(\bar{x}) \in \text{classes}$  that has the lowest test error.

First step to make a classifier a lifelong learner?

make it know when it doesn't know

= out of Distribution Detection

Novelty

Anomaly

Example: Trained a classifier to yield probability of each class  
(Apples, Oranges, Kiwi)

We present a pear for the classifier to

say "I don't know" it should yield a uniform prob distribution in the op

$$\text{e.g.: } \Pr(\text{Apple}) = \Pr(\text{Orange}) = \Pr(\text{Pear}) = 1/3$$

This is not easy to do with traditional ML methods such as Logistic regression.

---

Deep learning with huge image data can do this.

We can show a Huge set of out of distribution data to a deep learning and make it learn when it doesn't know.

Then we can modify our model to have (1 or more) extra class & learn it.

# Appendix 2

In this Appendix, we introduce SOMs

# Self-Organizing Maps

## Key idea (Kohonen, 1982)

- Idea: to learn a *map* of the data, in a low-dimensional embedding.
- Self-Organizing Maps can be thought of as a **constrained version of  $k$ -means** clustering, in which the prototypes are encouraged to lie in a **one- or two-dimensional manifold** in the feature space.

# Self-Organizing Maps

Key idea (Kohonen, 1982)

- This manifold is also referred to as *constrained topological map*, since the original high-dimensional observations can be mapped down onto the two-dimensional coordinate system.
- The original SOM algorithm was online, but batch versions have also been proposed.

# Self-Organizing Maps

- We consider a SOM with a two-dimensional rectangular grid of  $k$  prototypes *with  $p$  components*. The choice of a grid is arbitrary, other choices like hexagonal grids are also possible.
- Each of the  $k$  prototypes are parameterized with respect to an integer coordinate pair  $I_j \in Q_1 \times Q_2$ .

# Self-Organizing Maps

- Here  $Q_1 = \{1, 2, \dots, q_1\}$ ,  $Q_2 = \{1, 2, \dots, q_2\}$ , and  $k = q_1 q_2$ .
- One can think of the prototypes as "buttons" sewn on the principal component plane in a regular pattern.
- Intuitively, the SOM tries to bend the plane so that the buttons approximate the data points as well as possible.
- Once the model is fit, the observations can be mapped onto the two-dimensional grid.

# Self-Organizing Maps

- The  $m_j$  (prototypes) are initialized, for example, to lie in the two-dimensional principal component plane of the data
- The observations  $\mathbf{x}_i$  are processed one at a time.
- We find the closest prototype  $\mathbf{m}_j$  to  $\mathbf{x}_i$  in Euclidean distance in  $\mathbb{R}^p$ , and then for all neighbors  $\mathbf{m}_k$  of  $\mathbf{m}_j$ , we move  $\mathbf{m}_k$  toward  $\mathbf{x}_i$  via the update  $\mathbf{m}_k \leftarrow \mathbf{m}_k + \alpha(\mathbf{x}_i - \mathbf{m}_k)$

# Self-Organizing Maps

- The neighbors of  $m_j$  are defined to be all  $m_k$  such that the distance between  $l_j$  and  $l_k$  is small.
- $\alpha \in \mathbb{R}$  is the learning rate and determines the scale of the step. The simplest approach uses Euclidean distance, and “small” is determined by a threshold  $s$ . The neighborhood always includes the closest prototype itself.

# Self-Organizing Maps

- Notice that the distance is defined in the space  $Q_1 \times Q_2$  of integer topological coordinates of the prototypes, rather than in the feature space  $\mathbb{R}^d$ .
- The effect of the update is to move the prototypes closer to the data, but also to maintain a smooth two-dimensional spatial relationship between the prototypes.

# SOMs: Practical Considerations

- The performance is influenced by the learning rate  $\alpha$  and the distance threshold  $s$ . Typically  $\alpha$  is decreased from 1 to 0 in a few thousand iterations.
- Similarly  $s$  is decreased linearly from starting value  $R$  to 1 over a few thousand iterations. The description above refers to the simplest version of SOM.

# SOMs: Practical Considerations

- In more advanced approaches, the update step is changed with distance:
- $m_k \leftarrow m_k + \alpha h(\|l_j - l_k\|)(x_i - m_k)$ , where the neighborhood function  $h$  gives more weight to prototypes  $m_k$  with indices  $l_k$  closer to  $l_j$  than those further away.

# SOMs: Link to k-means

- If  $s$  is chosen so small that each neighborhood contains only one point, then the spatial connection between prototypes is lost. In that case, SOM is an online version of k-means, and converges to a local minimum of the k-means objective.

# Self-Organizing Maps: Summary

- Self-Organizing Maps are a variant of k-means for nonlinear dimensionality reduction and visualisation.
- As in k-means, points are iteratively assigned to cluster means.
- Unlike k-means, updates of cluster means are constrained and synchronized based on a user-defined lattice between cluster means.

# Self-Organizing Maps: Video

- <https://www.youtube.com/watch?v=H9H6s-x-0YE>

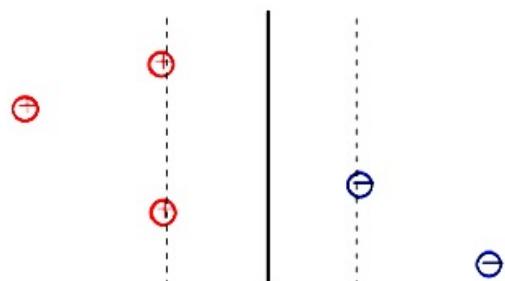
# **DSCI 552, Machine Learning for Data Science**

University of Southern California

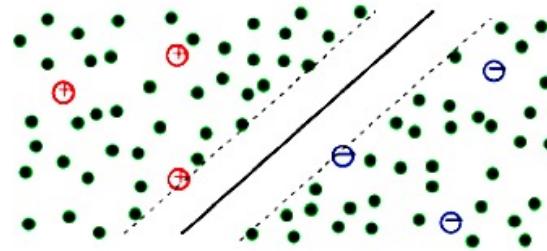
M. R. Rajati, PhD

# Lesson 9

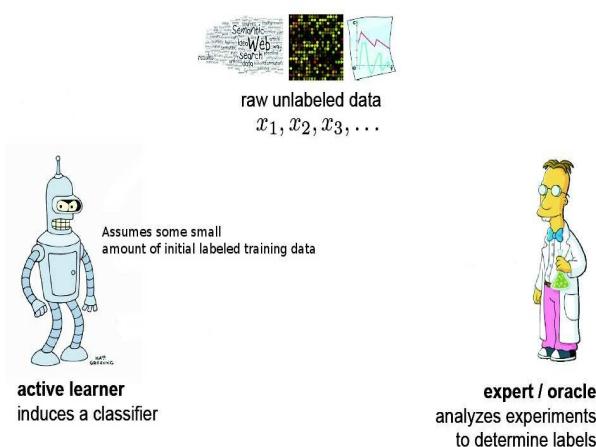
## Semi-Supervised and Active Learning



(a) SVM decision boundary



(b) S3VM decision boundary



# Introduction to Semi-Supervised Learning (SSL)

Classifier based methods

Co-Training, Yarowsky, and Their Combination

Data based methods (Not discussed here)

Manifold Regularization

Harmonic Mixtures

Information Regularization

# Learning Problems Revisited

## Supervised learning:

Given data consisting of feature-label pairs  $(x_i, y_i)$ , find the predictive relationship between features and labels.

## Unsupervised learning:

Given a sample consisting of only objects, look for interesting structures in the data, and group similar objects.

## What is Semi-supervised learning?

Supervised learning + Additional unlabeled data  
Unsupervised learning + Additional labeled data

# Motivation for SSL

Pragmatic:

Unlabeled data is cheap to collect.

**Example:** Classifying web pages,

There are some annotated web pages.

A huge amount of un-annotated pages is easily available by crawling the web.

Methodological:

The brain can exploit unlabeled data and learn from similarities.

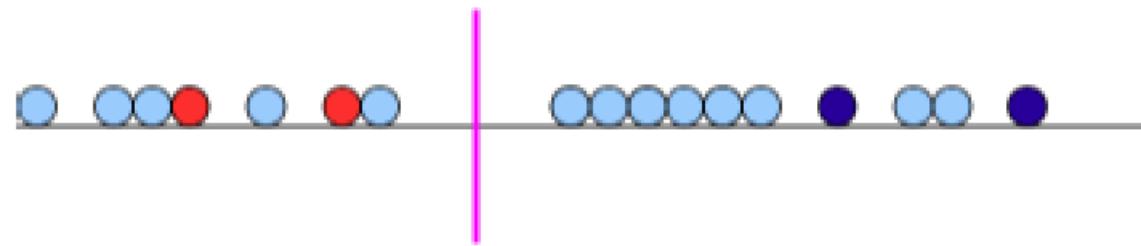
# How can unlabeled data help ?

Red: + 1, Dark Blue: -1



# How can unlabeled data help ?

Let's include some additional unlabeled data (Light Blue points)



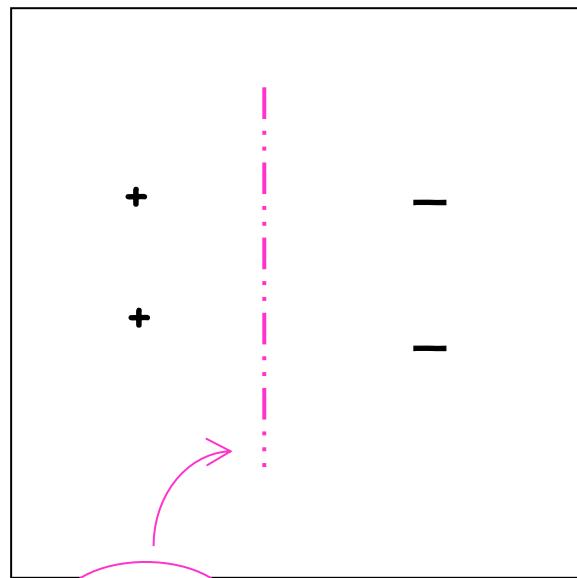
This might go wrong as well.

# How can unlabeled data help ?

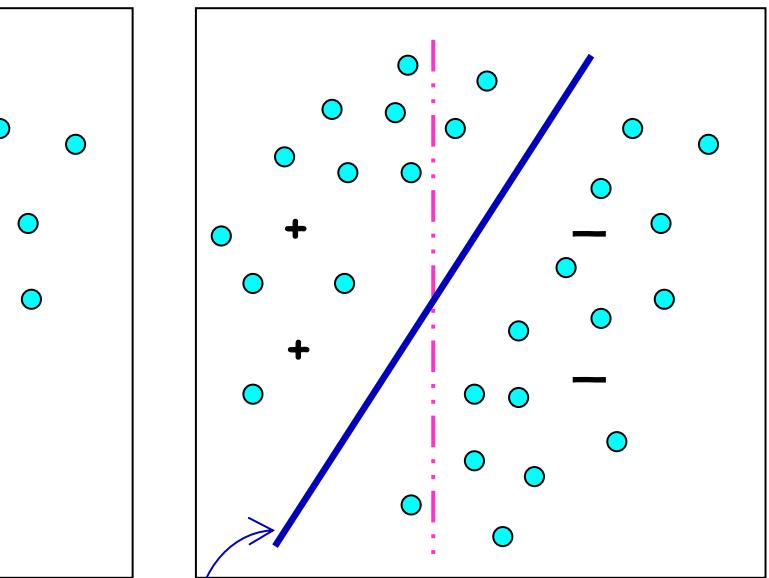
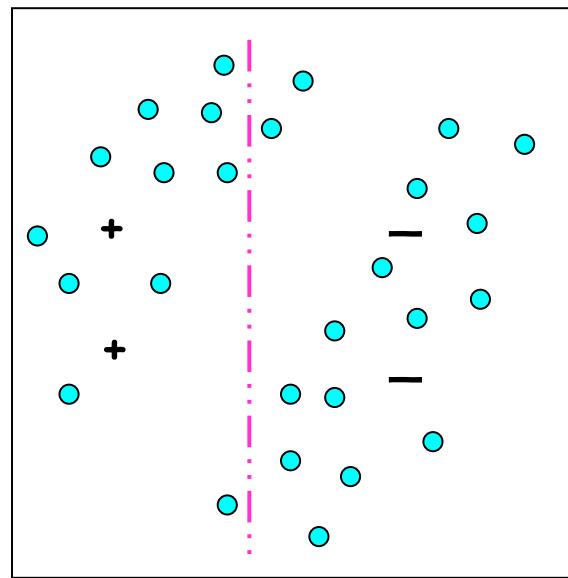
Assumption: Examples from the same class follow a coherent distribution

Unlabeled data can give a **better** sense of the class separation boundary

# Intuition



Labeled data only



Semi-Supervised SVM

# Inductive vs. Transductive

- **Transductive**: Produce label only for the available unlabeled data.
  - The output of the method is not a classifier.
- **Inductive**: Not only produce label for unlabeled data, but also produce a classifier.
- Let's first focus on inductive semi-supervised learning..

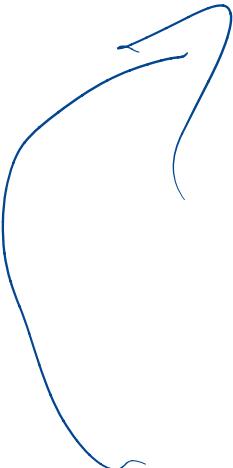
# Two Algorithmic Approaches

- Classifier based methods (Self-Training):
  - Start from initial classifier(s), and iteratively enhance it (them)
- Data based methods:
  - Discover an inherent geometry in the data, and exploit it in finding a good classifier.

Manifold learning

# Self-Training (Bootstrap)

## Self-Training/ The Yarowsky Algorithm

- 
- Train supervised model on labeled data  $L$
  - Test on unlabeled data  $U$
  - Add the most confidently classified members of  $U$  to  $L$
  - Repeat

Assume only supervised learning in probability based (e.g.: Logistic regression)

$$\Pr \approx 1$$

$$\Pr \approx 0$$

---

Discriminant based (svc?)

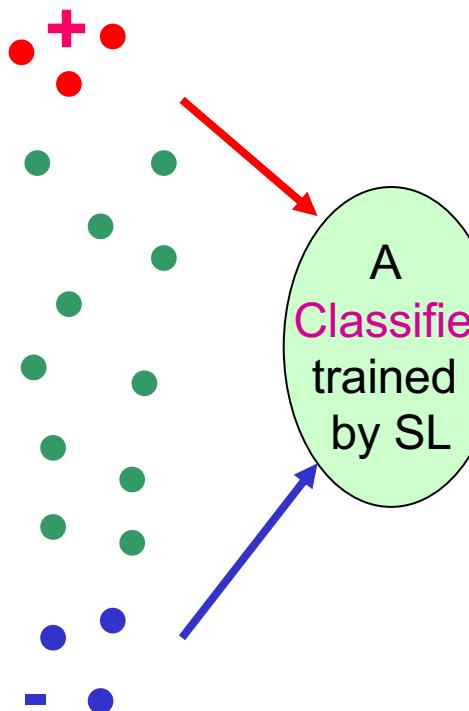
Large distance from the decision boundary

*Venkey  
for Smedhi*

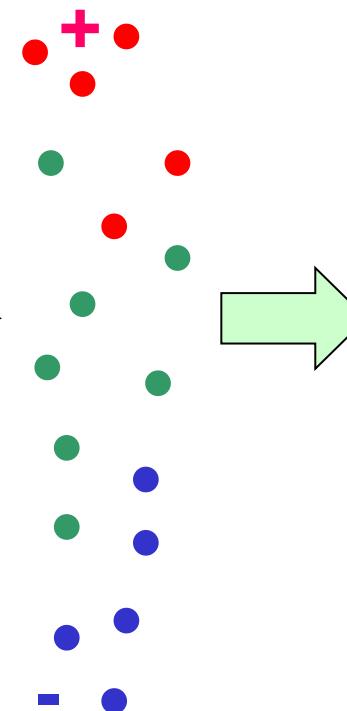
# Classifier Based Methods: The Yarowsky Algorithm

(Yarowsky 1995)

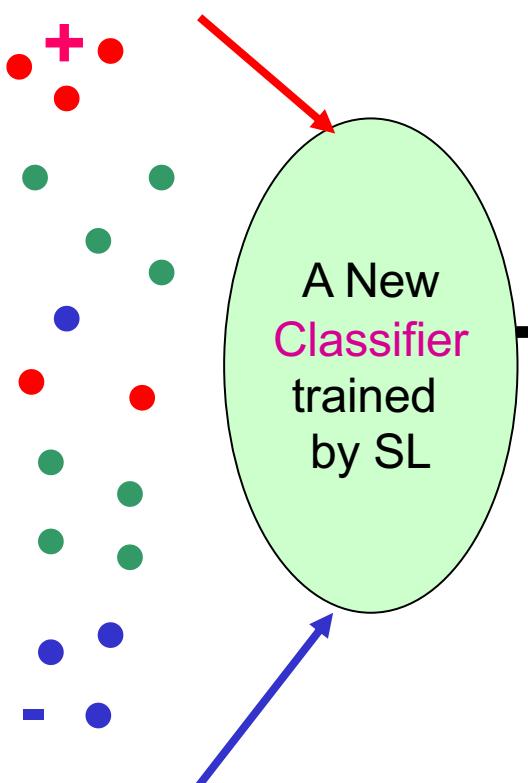
Iteration: 0



Iteration: 1



Iteration: 2



# Classifier-Based Methods: Refinement

- Refinement:
  - Reduce weight of unlabeled data to increase power of more accurate labeled data

# Advantages and Disadvantages of Self-Training

- **Advantages:**
  - The simplest semi-supervised learning method.
  - A wrapper method, applies to existing (complex) classifiers.
  - Often used in real tasks like natural language processing.
- **Disadvantages**
  - Early mistakes could reinforce themselves. Heuristic solutions, e.g. “un-label” an instance if its confidence falls below a threshold.

# Co-Training

- Instances contain two **sufficient sets of features**
  - i.e. an instance is  $x=(x_1, x_2)$
  - Each set of features is called a **View**



- Two views are **independent given the label**:

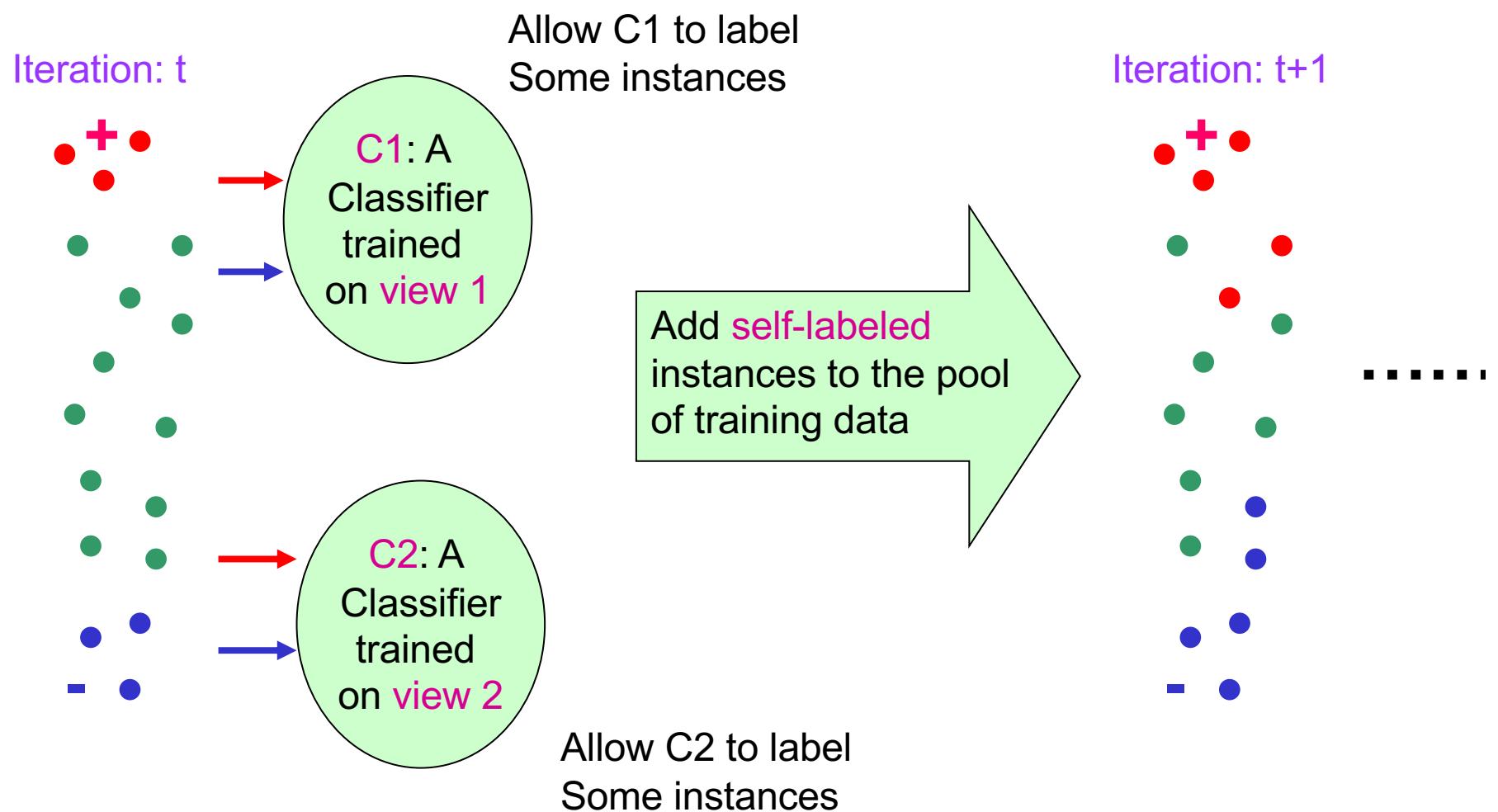
$$P(x_1|x_2, y) = P(x_1|y)$$

$$P(x_2|x_1, y) = P(x_2|y)$$

- Two views are **consistent**:

$$\exists c_1, c_2 : c^{opt}(x) = c_1(x_1) = c_2(x_2)$$

# Co-Training



# Co-Training

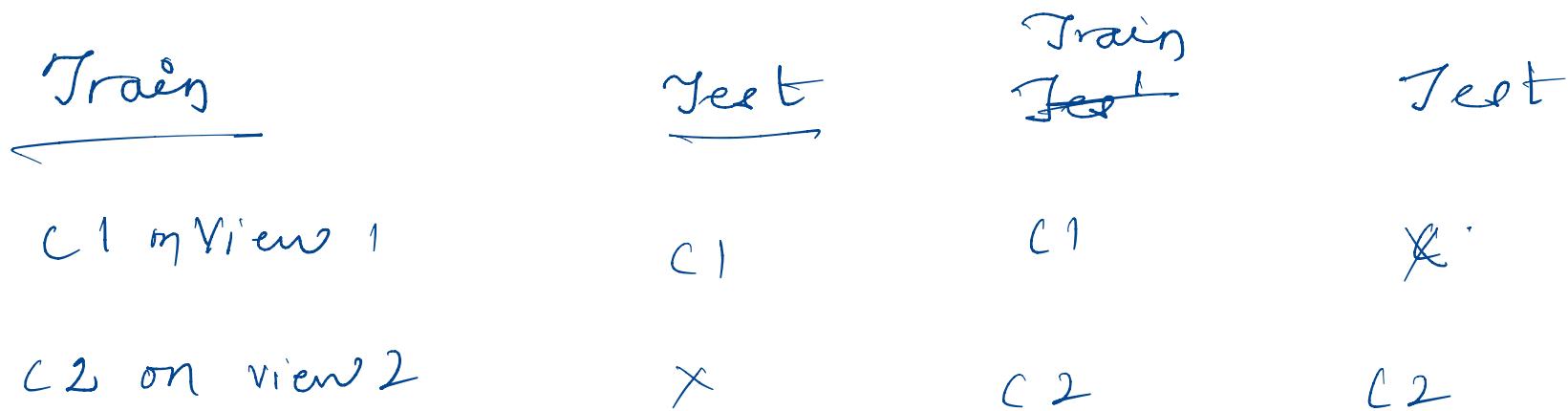
- . Example of learning from *multiple views* (multiple sets of attributes): classifying webpages
  - . First set of attributes describes content of web page
  - . Second set of attributes describes links from other pages
- . Independence Assumption:
  - . Reduces the probability of the models agreeing on incorrect labels

# Yarowsky+ Co-training

- . Like Yarowsky Algorithm for semi-supervised learning, but view is **switched** in each iteration
  - . Uses all the unlabeled data (probabilistically labeled) for training
  - . In each iteration, only one of the classifiers labels the data.

# Yarowsky+ Co-training

- Has been used successfully with neural networks and support vector machines.



# Example

The image shows two web pages side-by-side. On the left is the Stanford Computer Science faculty list, featuring a table of names, phone numbers, and office locations. On the right is Donald E. Knuth's homepage, which includes a photo, a brief bio, and a sidebar with various links. A red box highlights the sidebar on Knuth's page.

| Name                 | Office         |
|----------------------|----------------|
| Ron Fedkiw           | GATES 207      |
| Edward Feigenbaum    | GATES 237      |
| Richard Fikes        | Gates 505      |
| Hector Garcia-Molina | GATES 434      |
| Mike Genesereth      | GATES 220      |
| Leonidas Guibas      | CLARK S293     |
| Patrick Hanrahan     | GATES 370      |
| Jeff Heer            | Gates 375      |
| John Hennessy        | BLDG 10        |
| Mark Horowitz        | GATES 306      |
| Oussama Khatib       | GATES 144      |
| Scott Klemmer        | Gates 384      |
| <b>Don Knuth</b>     | GATES 477      |
| Daphne Koller        | GATES 142      |
| Vladlen Koltun       | Gates 374      |
| Christos Kozyrakis   | Gates Hall 304 |
| Monica Lam           | GATES 307      |

Donald E. Knuth (Donald E. Knuth), Professor Emeritus of [The Art of Computer Programming](#) at [Stanford University](#), welcomes you to his home page.

- ✉ Frequently Asked Questions
- ✉ Infrequently Asked Questions
- ✉ Recent News
- ✉ Computer Musings
- ✉ Known Errors in My Books
- ✉ Important Message to all Users of TeX
- ✉ Help Wanted
- ✉ Diamond Signs
- ✉ Preprints of Recent Papers
- ✉ Curriculum Vitae
- ✉ Pipe Organ

Classify web pages into category for students and category for professors

Two **views** of web page

**Content:** I am currently a professor of ...

**Hyperlinks:** a link to the faculty list of computer science department

# General Multiview Learning

*multiple views*

Train multiple diverse models on  $L$ .  
Those instances in  $U$  which **most**  
**models agree on** are placed in  $L$ .

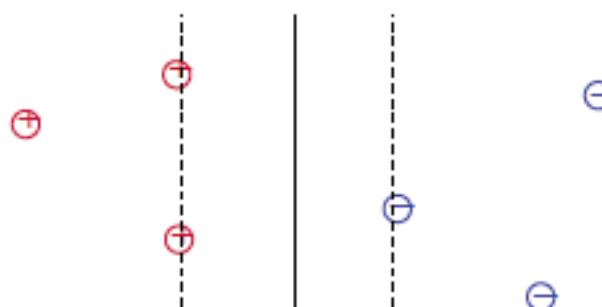
# Semi-Supervised SVM

## Semi-Supervised SVM (S3VM)

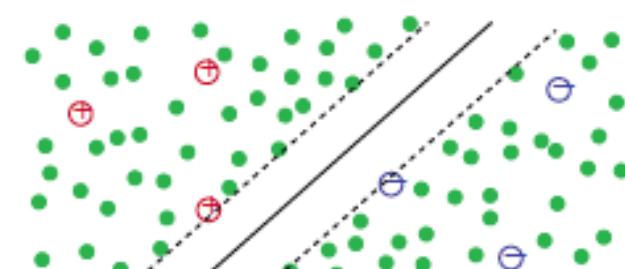
Maximize margin of both  $L$  and  $U$ . Decision surface placed in non-dense spaces

Assumes classes are "well-separated"

Can also try to simultaneously maintain class proportion on both sides similar to labeled proportion



(a) SVM decision boundary



(b) S3VM decision boundary

If  $\Pr(C_1) = 20\%$ .

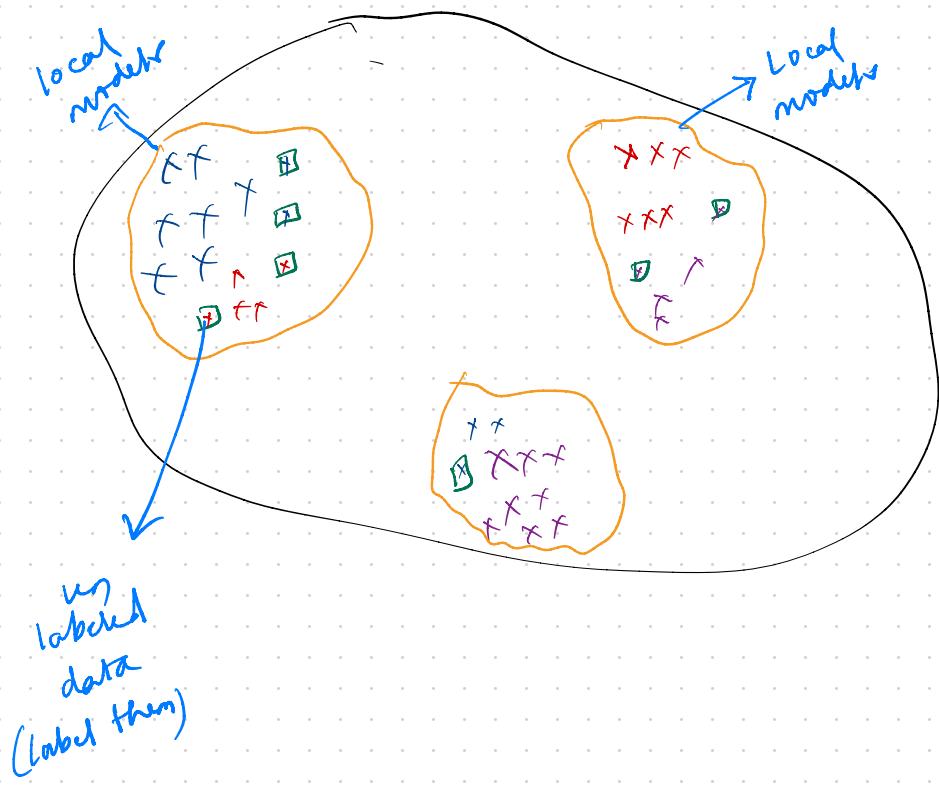
$\Pr(C_2) = 80\%$ .



20% of unlabeled data can be assigned  
to the side of the class  $C_1$ .

# Cluster-and-Label Approach

- Assumption: Clusters coincide with decision boundaries
- Poor results if this assumption is wrong
  - Cluster labeled and unlabeled data
  - For each cluster, train a classifier based on the labeled points within that cluster
  - Label all data in each cluster using the classifier designed for that cluster.
  - Train a model based on the whole data (that is now labeled)



Clustering as preprocessing for supervised learning.

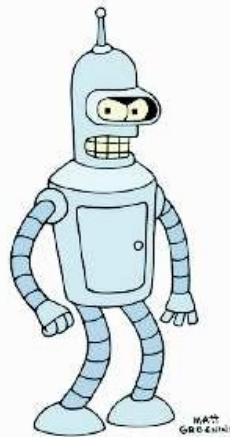
- 1) Cluster the whole dataset by disregarding the labels.
- 2) Build a local supervised learner for each cluster
- 3) For a test point  $x^*$ , first determine in which cluster it resides (cluster to the closest centroid), then predict its label  $y$  using the local model of that cluster.

# (Passive) Supervised Learning



raw unlabeled data

$x_1, x_2, x_3, \dots$



**supervised learner**  
induces a classifier



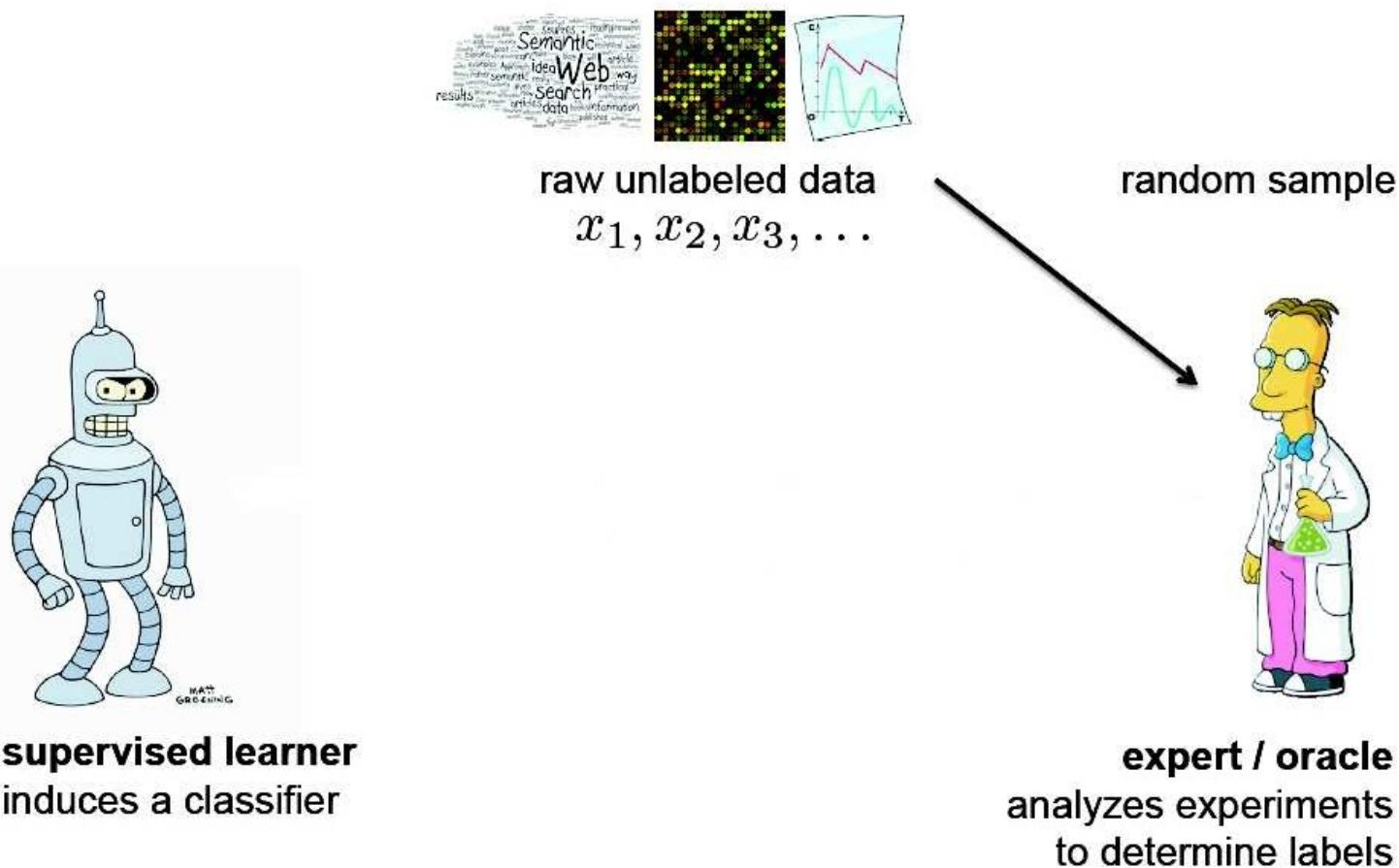
**expert / oracle**  
analyzes experiments  
to determine labels

1

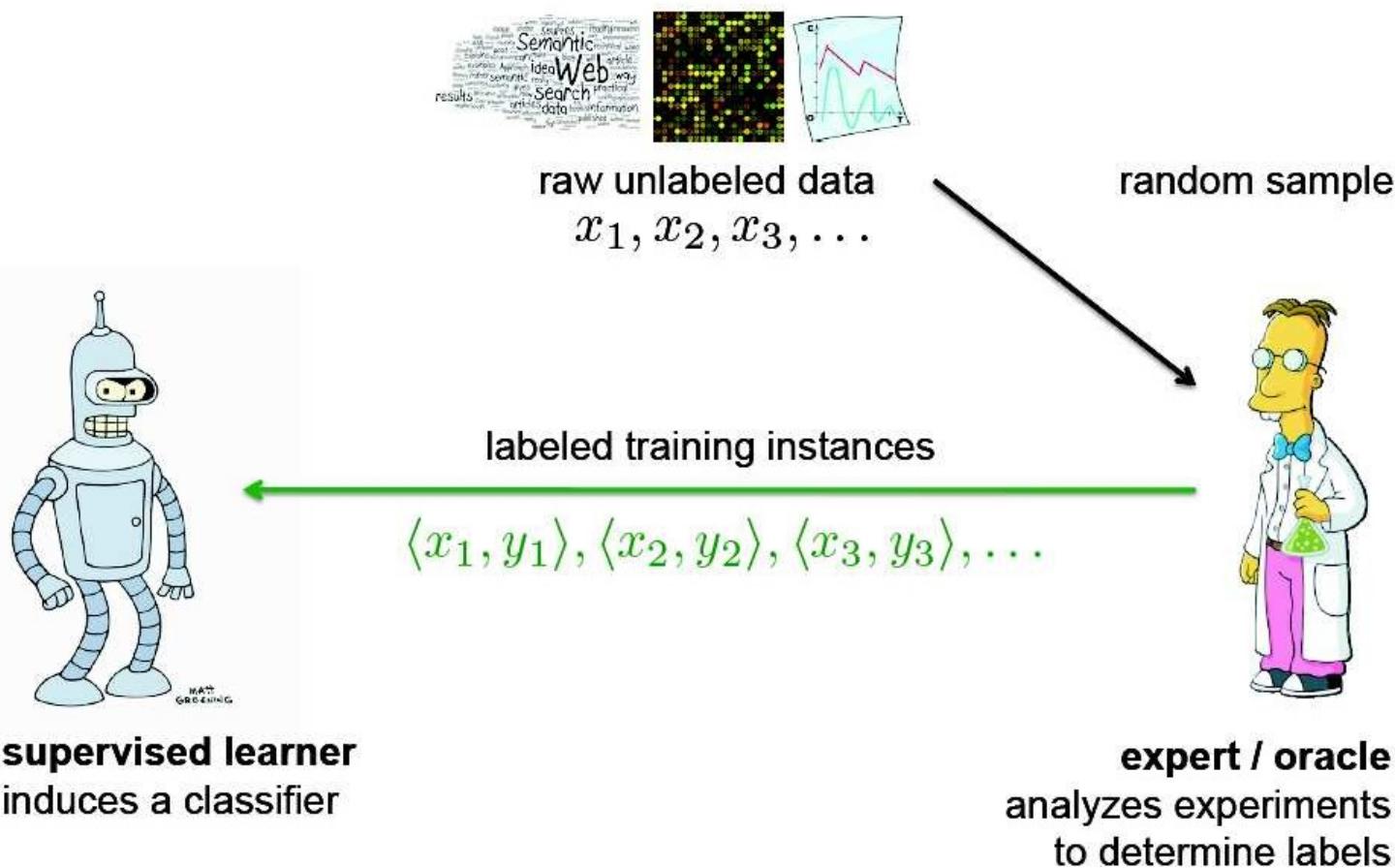
---

Some figures from Burr Settles

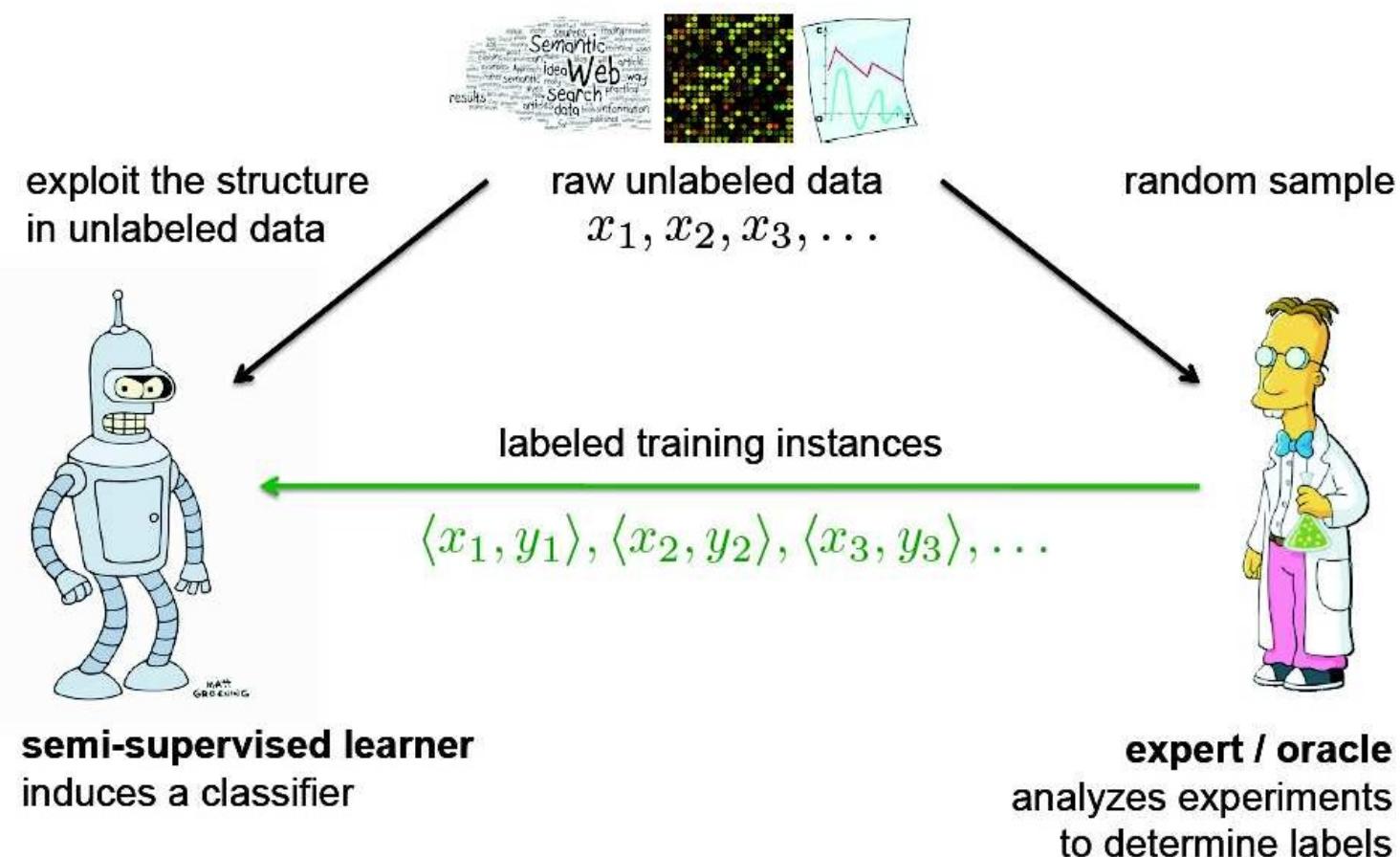
# (Passive) Supervised Learning



# (Passive) Supervised Learning



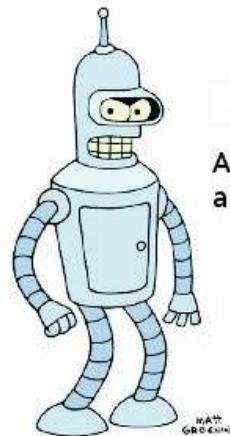
# Semi-supervised Learning



# Active Learning



raw unlabeled data  
 $x_1, x_2, x_3, \dots$



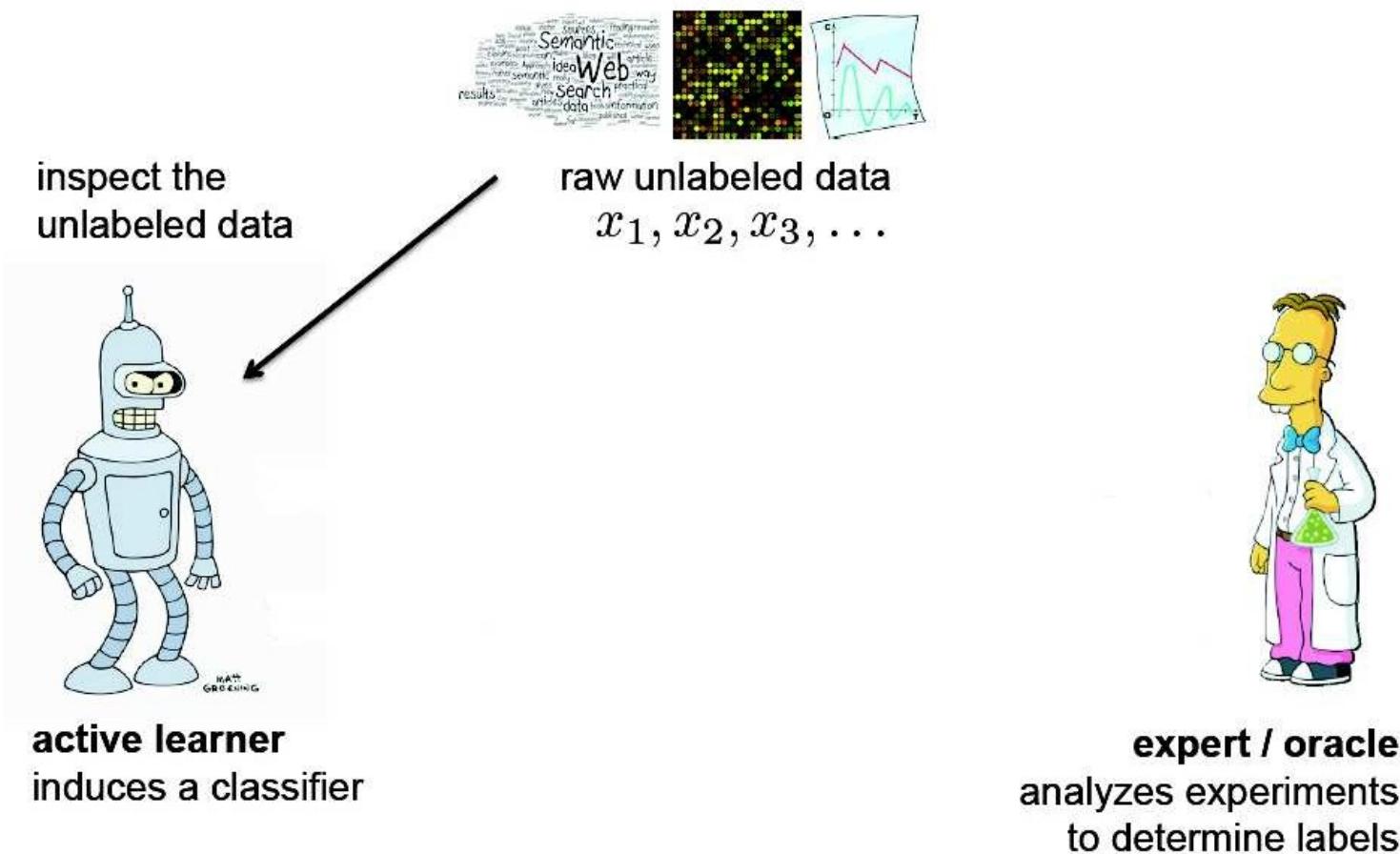
Assumes some small  
amount of initial labeled training data

**active learner**  
induces a classifier



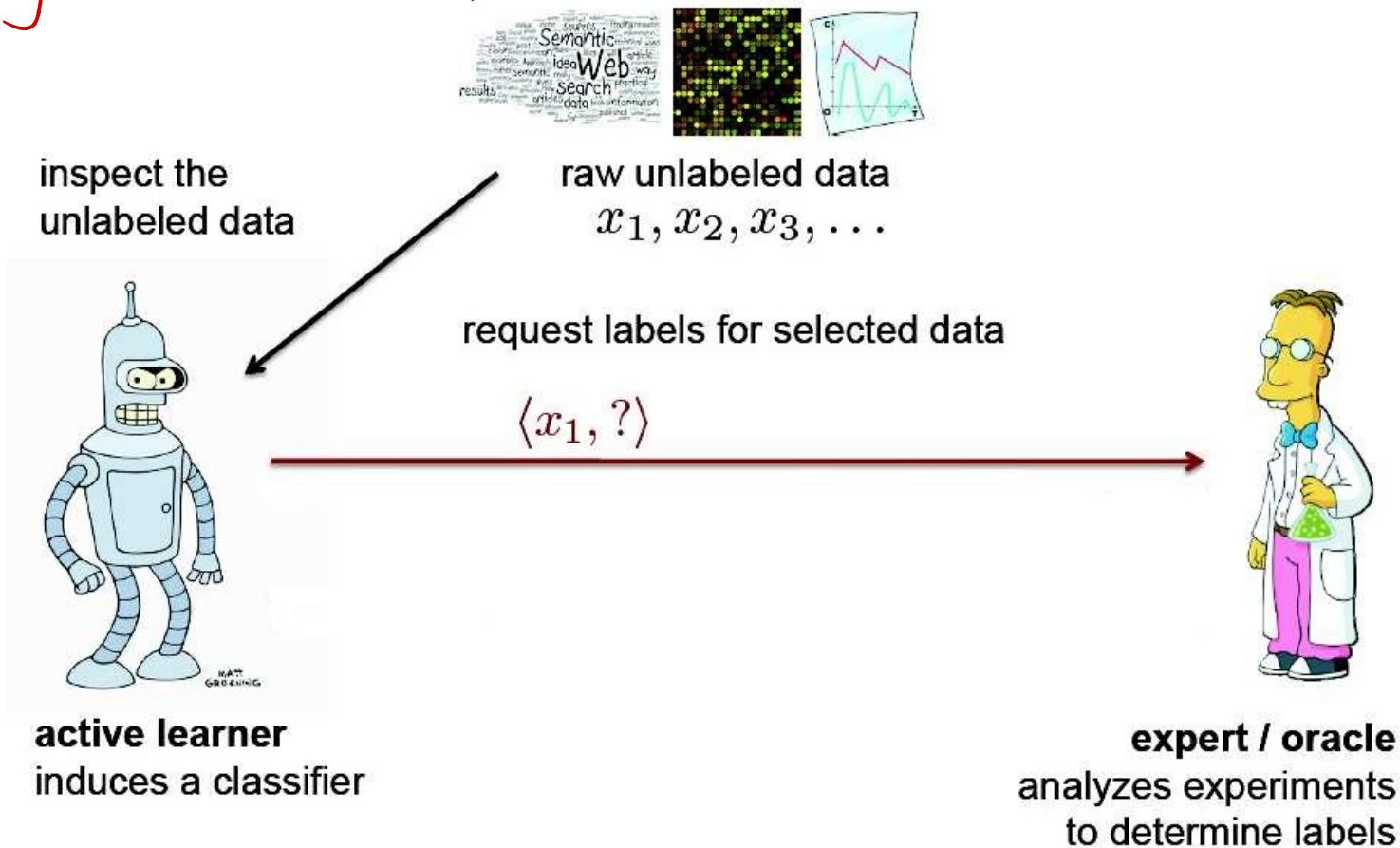
**expert / oracle**  
analyzes experiments  
to determine labels

# Active Learning

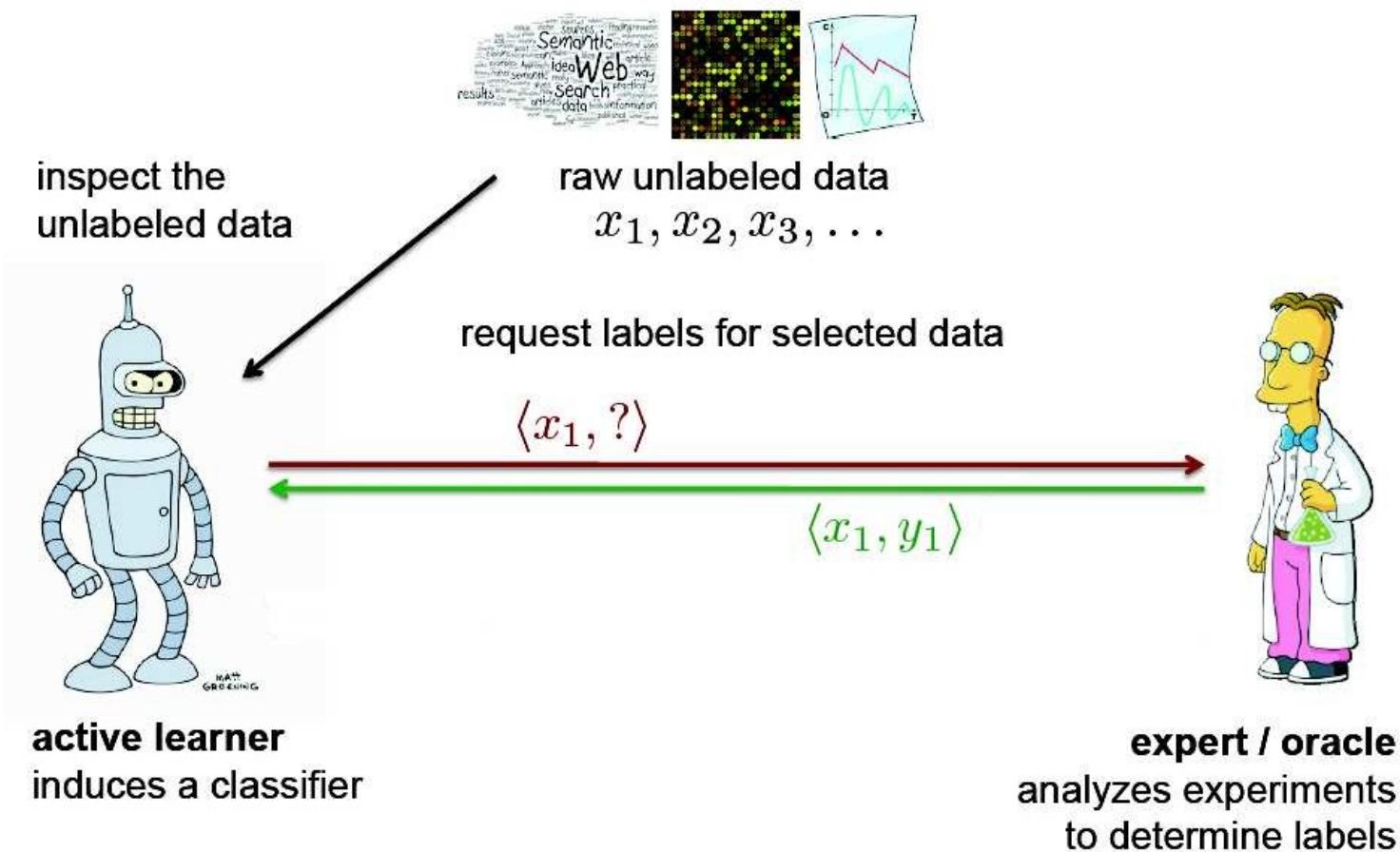


# Active Learning

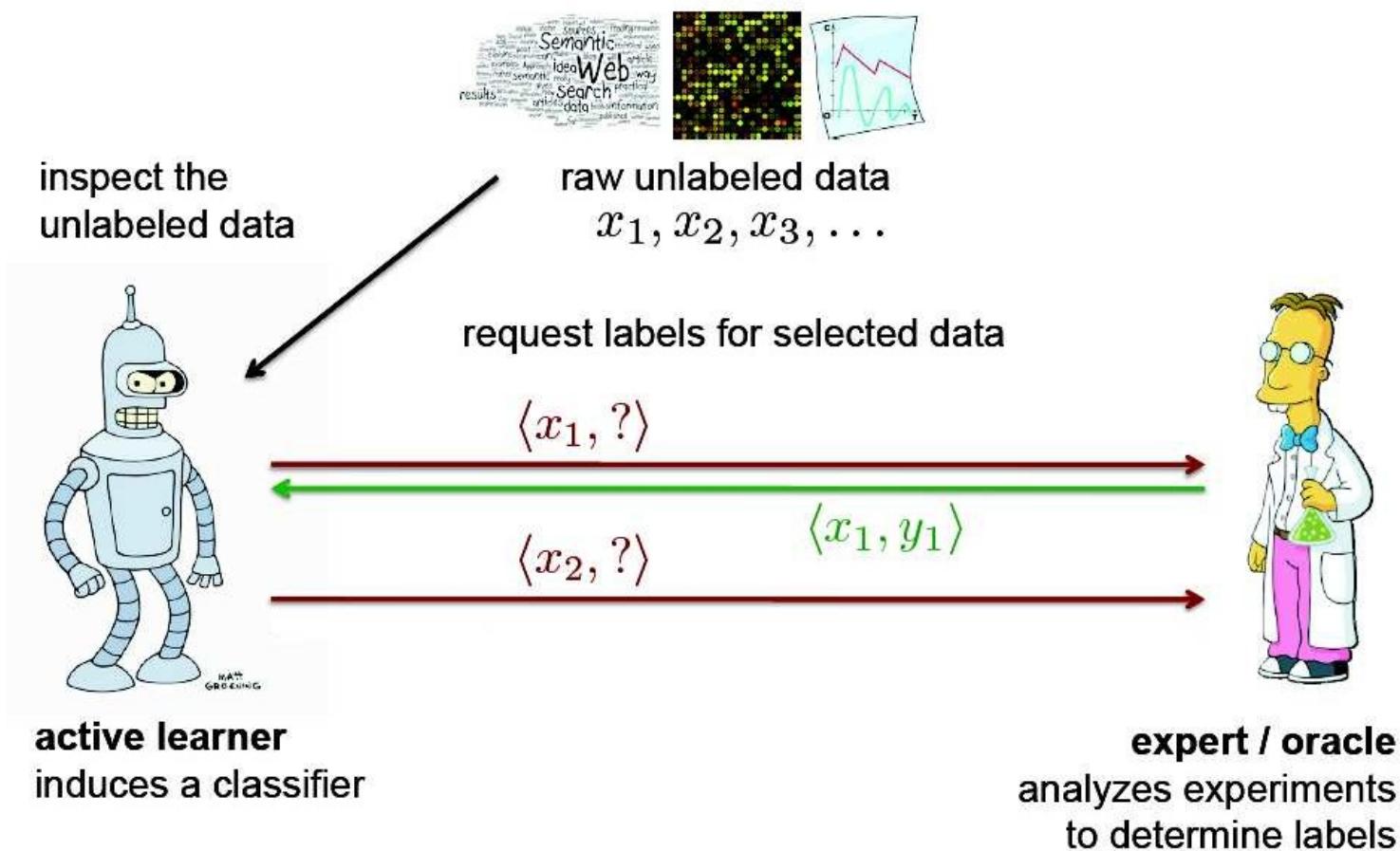
*Sampling isn't random*



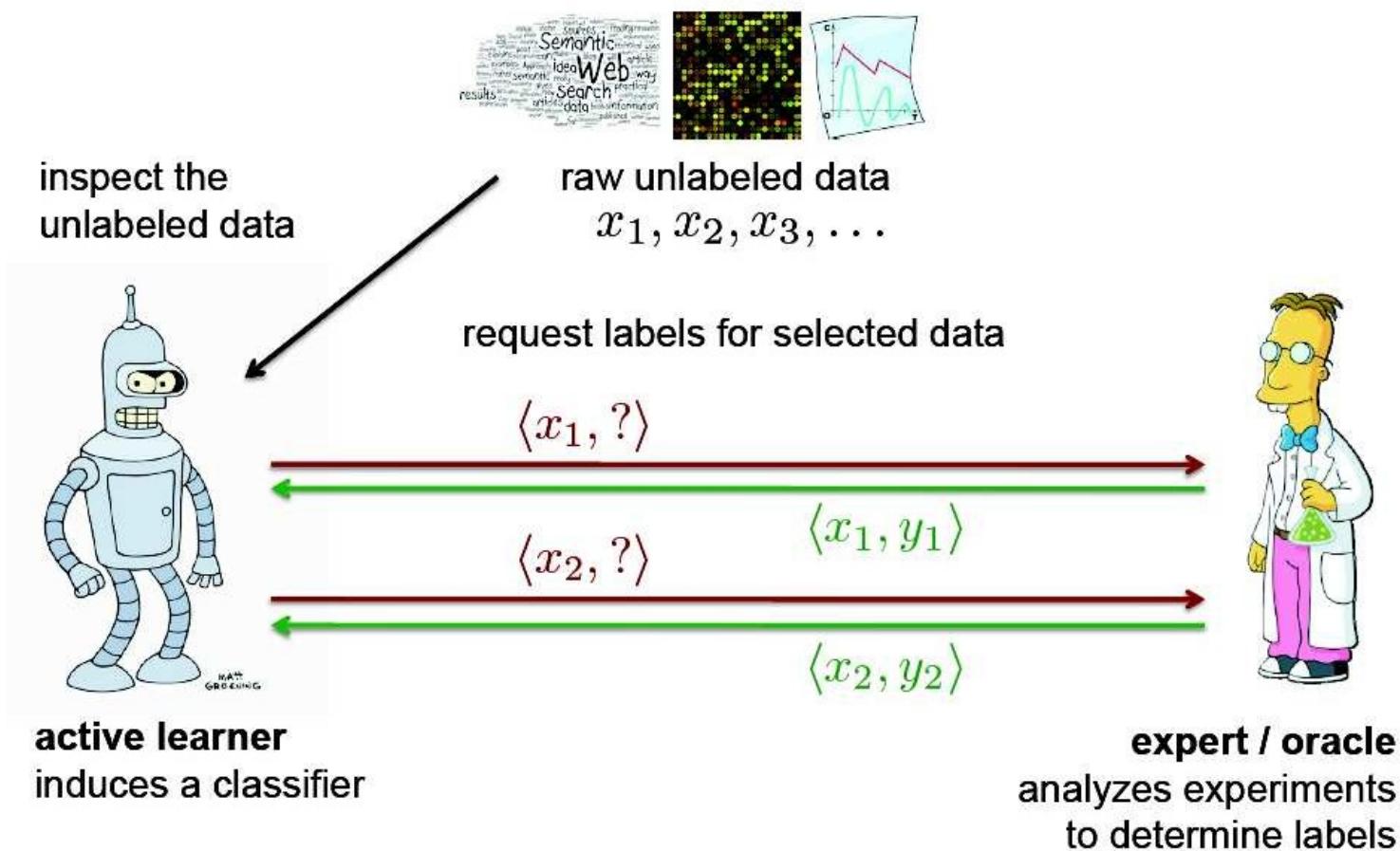
# Active Learning



# Active Learning



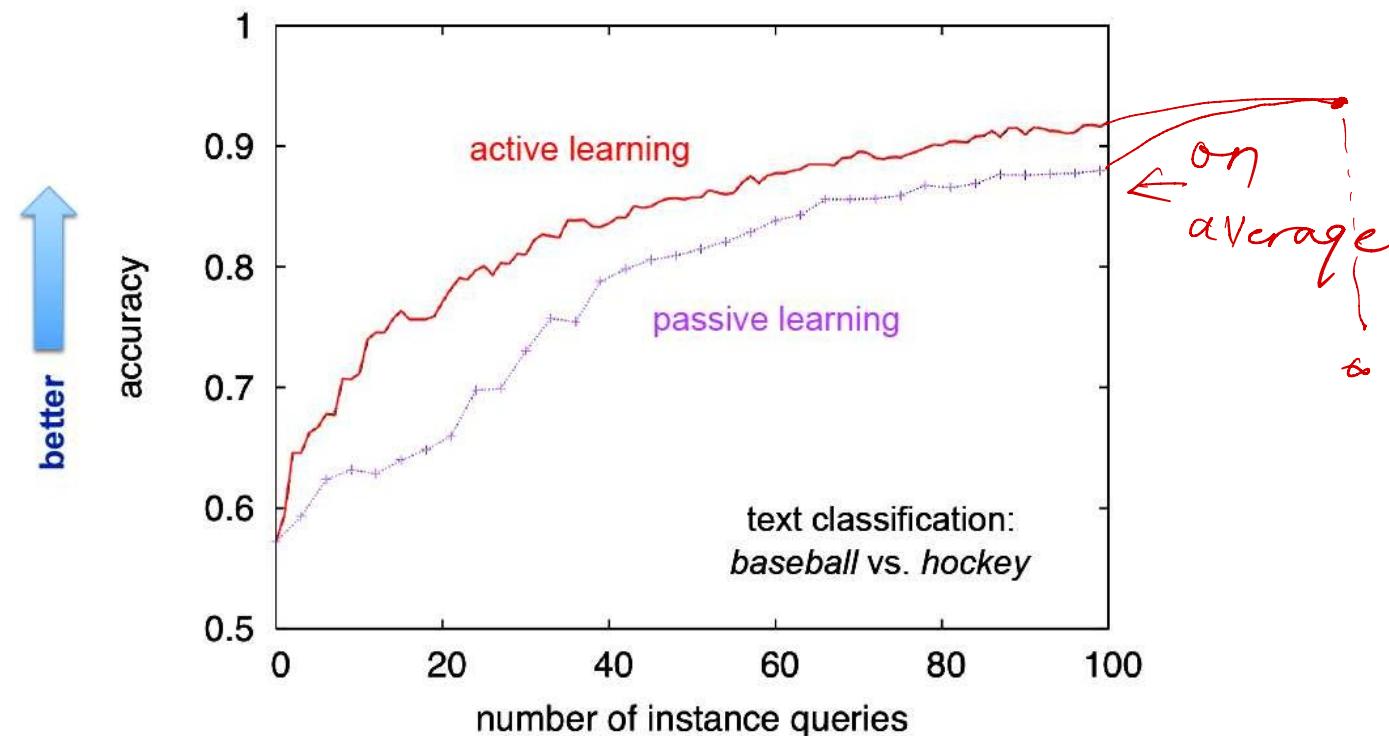
# Active Learning



# Active Learning vs Random Sampling

Passive Learning curve: Randomly selects examples to get labels for  
Active Learning curve: Active learning selects examples to get labels for

## Learning Curves

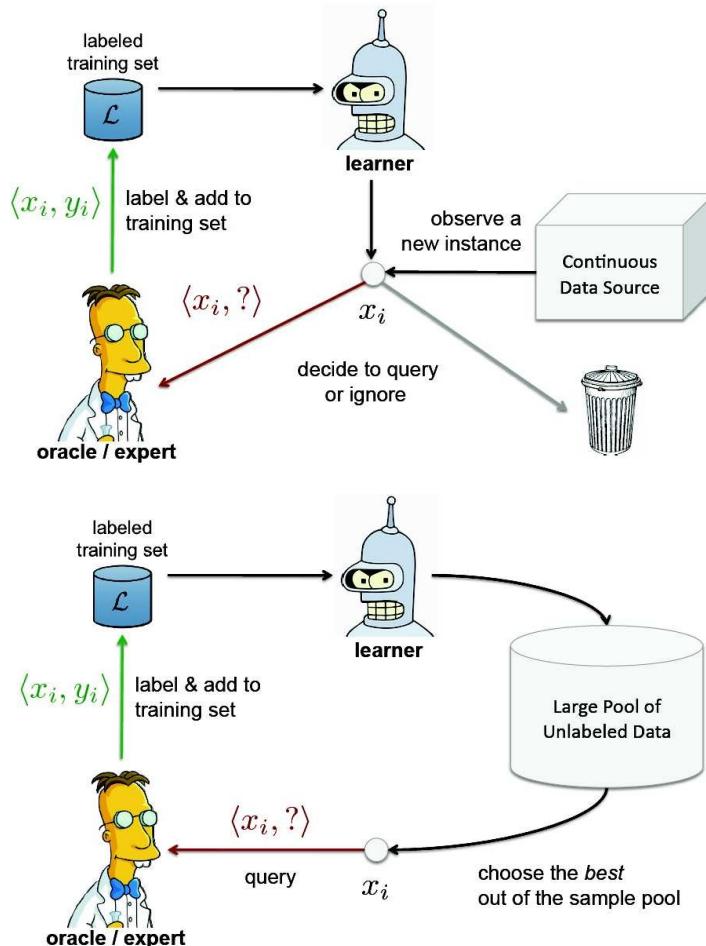


(Random experiment, samples might change)

# Types of Active Learning

Largely falls into one of these two types:

- Stream-Based Active Learning
  - Consider one unlabeled example at a time
  - Decide whether to **query its label** or **ignore it**
- Pool-Based Active Learning Given:
  - a large unlabeled pool of examples
  - **Rank examples in order of informativeness**
  - **Query the labels** for the most informative example(s)



# Query Selection Strategies

Any Active Learning algorithm requires a **query selection strategy**

Some examples:

- ✓ • Uncertainty Sampling
- ✓ • Query By Committee (QBC)
- ✗ • Expected Model Change
- ✗ • Expected Error Reduction
- ✗ { • Variance Reduction
- ✗ { • Density Weighted Methods

# How Active Learning Operates

- Active Learning **proceeds in rounds**
- Each round has a **current model** (learned using the labeled data seen so far)
- The **current model** is **used to assess informativeness** of unlabeled examples
  - ... using one of the query selection strategies
- The most informative example(s) is/ are selected

# How Active Learning Operates

- The **labels are obtained** (by the labeling oracle)
- The (now) labeled example(s)

is/are included in the training data

The **model is re-trained** using the

new training data

- The process repeats **until we have budget left** for getting labels

# Uncertainty Sampling

Select examples which the current model is the **most uncertain about**

Various ways to measure uncertainty. For example:

Based on the **distance from the hyperplane**

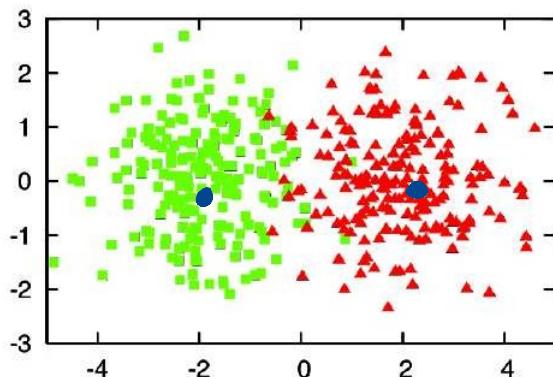
**Using the label probability**  $P(y|x)$  (for probabilistic models) *Probability 0.5*

$$P(y|x) \approx 0.5 \text{ (logistic reg)}$$

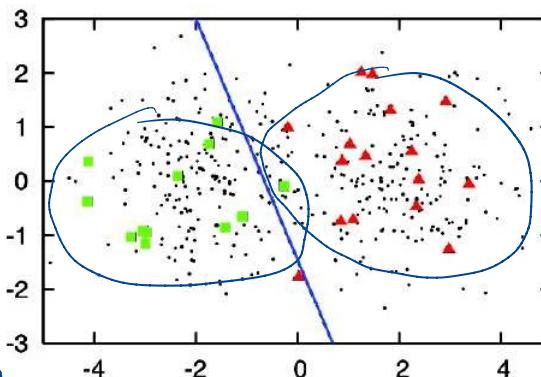
*SVL → point close to the hyperplane.*

# Uncertainty Sampling

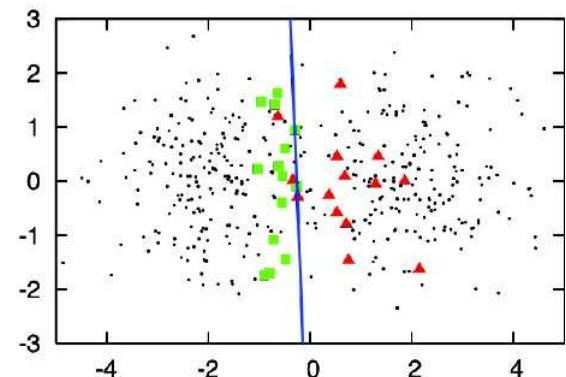
A simple illustration of uncertainty sampling based on the distance from the hyperplane (i.e., margin based)



400 instances sampled  
from 2 class Gaussians



random sampling  
30 labeled instances  
(accuracy=0.7)



uncertainty sampling  
30 labeled instances  
(accuracy=0.9)

*(Training set is biased  
in this case)*

# Query By Committee (QBC)

- QBC uses a committee of models
- All models trained using the currently available labeled data  $L$
- How is the committee constructed?
  - Some possible ways: Sampling different models from the model distribution
  - Using ensemble methods (bagging/boosting, etc.)

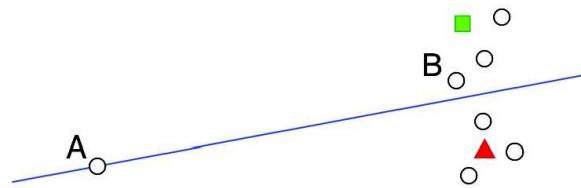
# Query By Committee (QBC)

- All models vote their predictions on the unlabeled pool
- The example(s) with maximum disagreement is/are chosen for labeling
- Each model in the committee is re-trained after including the new example(s)

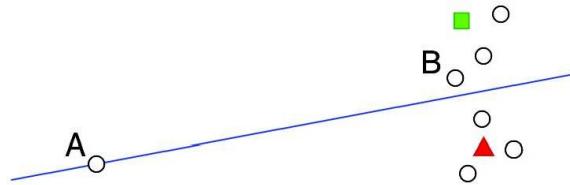
# Effect of Outlier Examples

Uncertainty Sampling or QBC may wrongly think an **outlier** to be an informative example

Such examples won't really help (and can even be **misleading**)



# Effect of Outlier Examples



Other robust query selection methods exist to deal with outliers

Idea: Instead of using the confidence of a model on an example, see how a **labeled example affects** the model itself (various ways to quantify this)

The example(s) that affects the model the most is probably the most informative

# Other Query Selection Methods

## Expected Model Change

Select the example whose inclusion brings about the maximum change in the model (e.g., the gradient of the loss function w.r.t. the parameters)

## Expected Error Reduction

Select example that reduces the expected generalization error the most

.. measured w.r.t. the remaining unlabeled examples  
(using the *expected* labels)

$$\text{Average } E[\Delta] = E[\Delta | \text{sample is +ve}] \Pr(\text{pos}) + E[\Delta | \text{sample is -ve}] \Pr(\text{neg})$$

*Just FYI*

# Other Query Selection Methods

## Variance Reduction

Select example(s) that **reduces the model variance by the most**

.. by **maximizing Fisher information** of model parameters (e.g., by *minimizing* the trace or determinant of the **inverse Fisher information matrix**)

Fisher information matrix: computed using the log-likelihood

## Density Weighting

**Weight the informativeness** of an example by its **average similarity to the entire unlabeled pool of examples**

An outlier will not get a substantial weight!

# Active Learning with Selective Sampling

Looking at one example at a time with a margin-based classifier

Input: Parameter  $b > 0$  (dictates how aggressively we want to query labels)

↳ budget burning parameter

For  $n = 1 : N$

Get  $x_n$ , compute  $p_n$

Predict  $\hat{y}_n = \text{sign}(p_n)$

Draw Bernoulli random variable  $Z \in \{0, 1\}$  with probability  $\frac{b}{b + |p|}$

If  $Z == 1$ , query the true label  $y_n$

Else if  $Z == 0$ , ignore the example  $x_n$  and don't update w

Flip a coin

of success  
Heads

$$\frac{b}{b + |p|}$$

Comments:

$|p_n|$  is the **absolute margin** of  $x_n$

Large  $|p_n| \Rightarrow$  Small label query probability

$$P(z \geq 1) = \frac{b}{b + |P|}$$

Assume  $b=1 \Rightarrow \frac{1}{1+|P|}$

$|P|$  small  $\Rightarrow$  Prob. of success  $\approx 1$

$|P|$  large  $\Rightarrow$  Prob. of success  $\approx 0$

---

$$b \rightarrow \infty \quad P(z \geq 1) = \frac{b}{b + |P|} \rightarrow \underline{\underline{1}}$$

$\Rightarrow$  Burn your budget fast

$b \rightarrow$  small

conservative, the distance ( $P$ ) has to be small for the data point to get labeled.

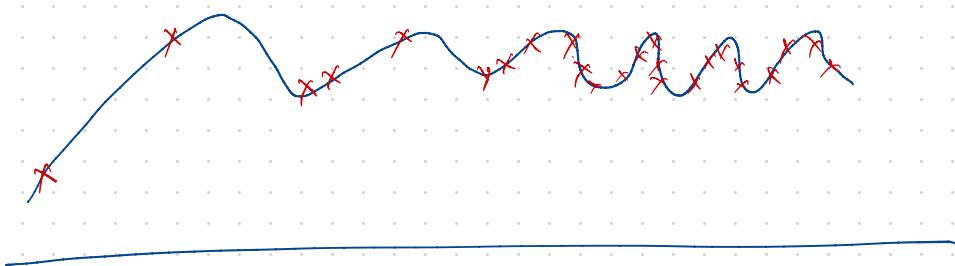
# Interesting Observations and Questions

- Active Learning: Label efficient learning strategy  
Based on judging the informativeness of examples
- Several variants possible. E.g.,
  - Different examples having different labeling costs
  - Access to multiple labeling oracles (possibly noisy)
  - Active Learning on features instead of labels (e.g., if features are expensive)
- Being “actively” used in industry (IBM, Microsoft, Siemens, Google, etc.)

# Interesting Observations and Questions

- Can an actively labeled dataset be reused to train a new different model?
- Sampling is biased. The actively labeled dataset doesn't reflect the true training/test data distribution. What could be the consequences? How could this be accounted for?
- Can we formulate active learning for regression?

## Active learning for regression:



Informativeness can be inferred  
from the gradient of the model  
learned.



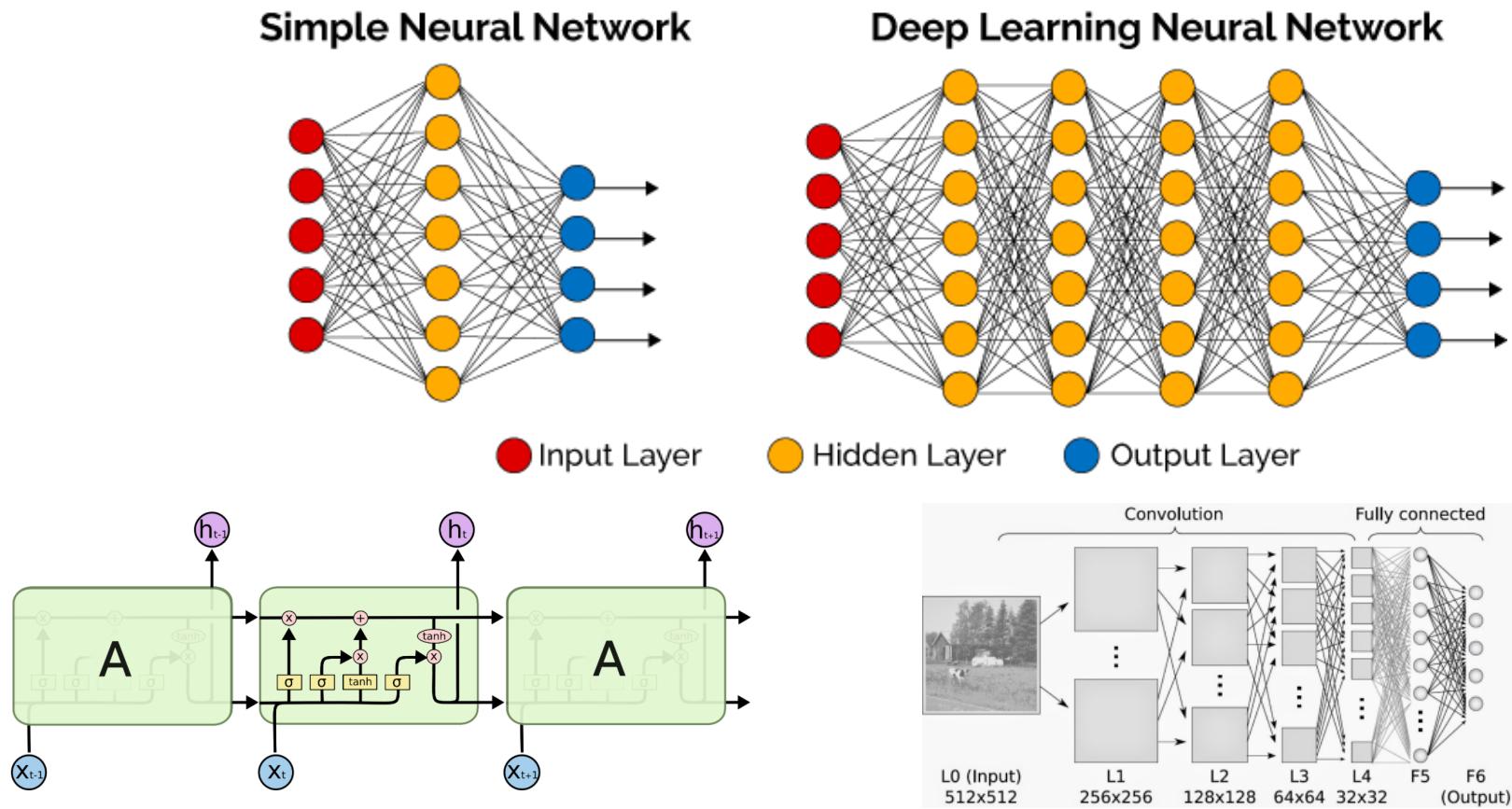
# **DSCI 552, Machine Learning for Data Science**

University of Southern California

M. R. Rajati, PhD

# Lesson 10

## Neural Networks and Deep Learning (Appendices)



# Appendix

Unsupervised Learning/ Feature  
Learning Using NNs

&

Pre-Training

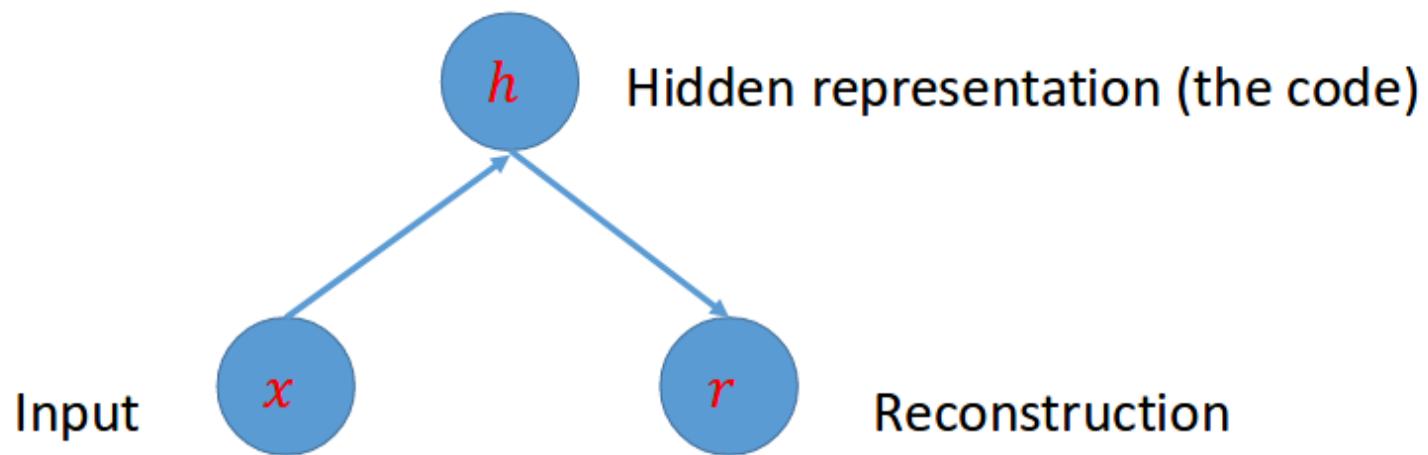
# Unsupervised Learning/ Feature Learning Using NNs

- One can use MLPs to learn low dimensional representations of high dimensional data
- This corresponds to dimensionality reduction or learning features from data
- Similar in nature to PCA

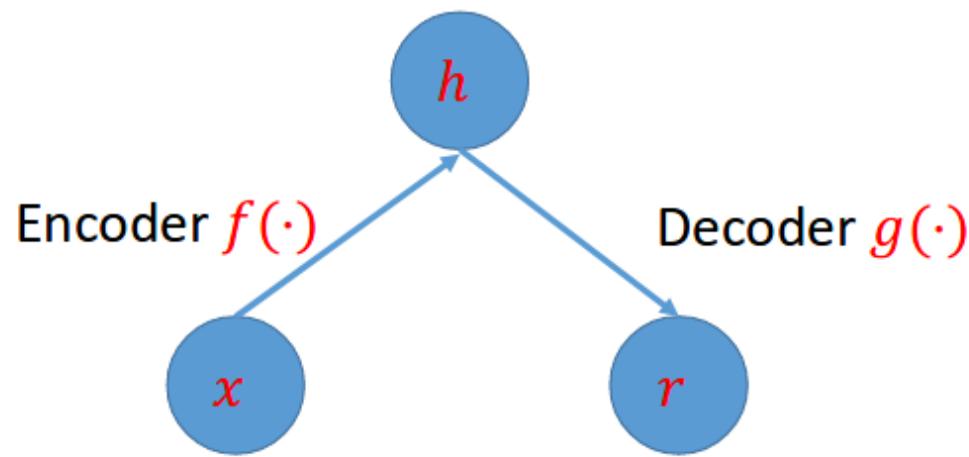
# Autoencoders

- Neural network trained to attempt to copy its input to its output
- Contains two parts:
- Encoder: map the input to a hidden representation
- Decoder: map the hidden representation to the output

# Autoencoders



# Autoencoders



$$h = f(x), r = g(h) = g(f(x))$$

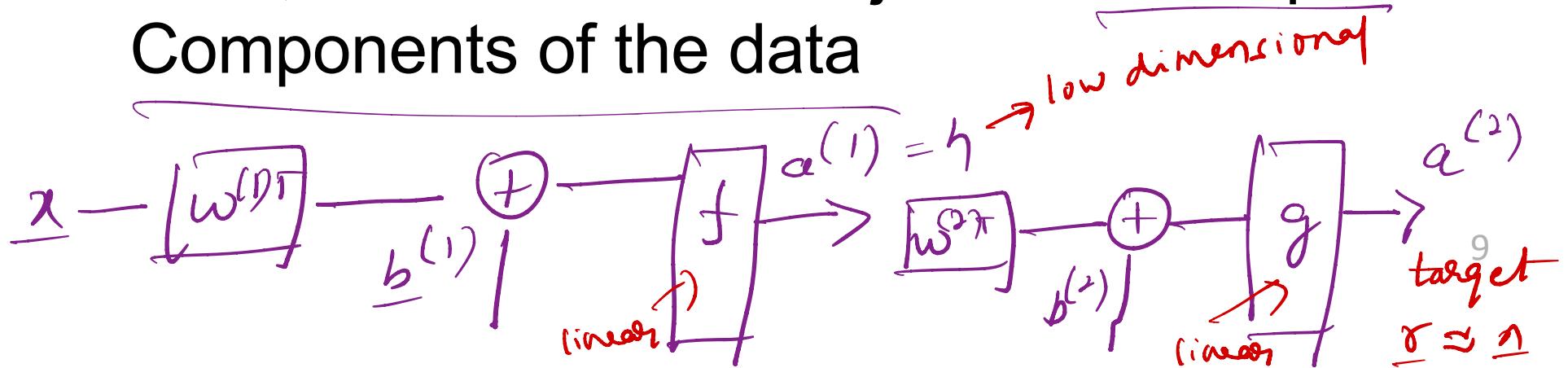
$$\gamma \approx n \quad g(f(x)) \approx \text{identity}$$

# Why Learn the Identity Function?

- We do not really care about copying
- NN will NOT be able to copy exactly but strives to do so
- Autoencoder:
  - forced to select which aspects to preserve and thus
  - hopefully can learn useful properties of the data

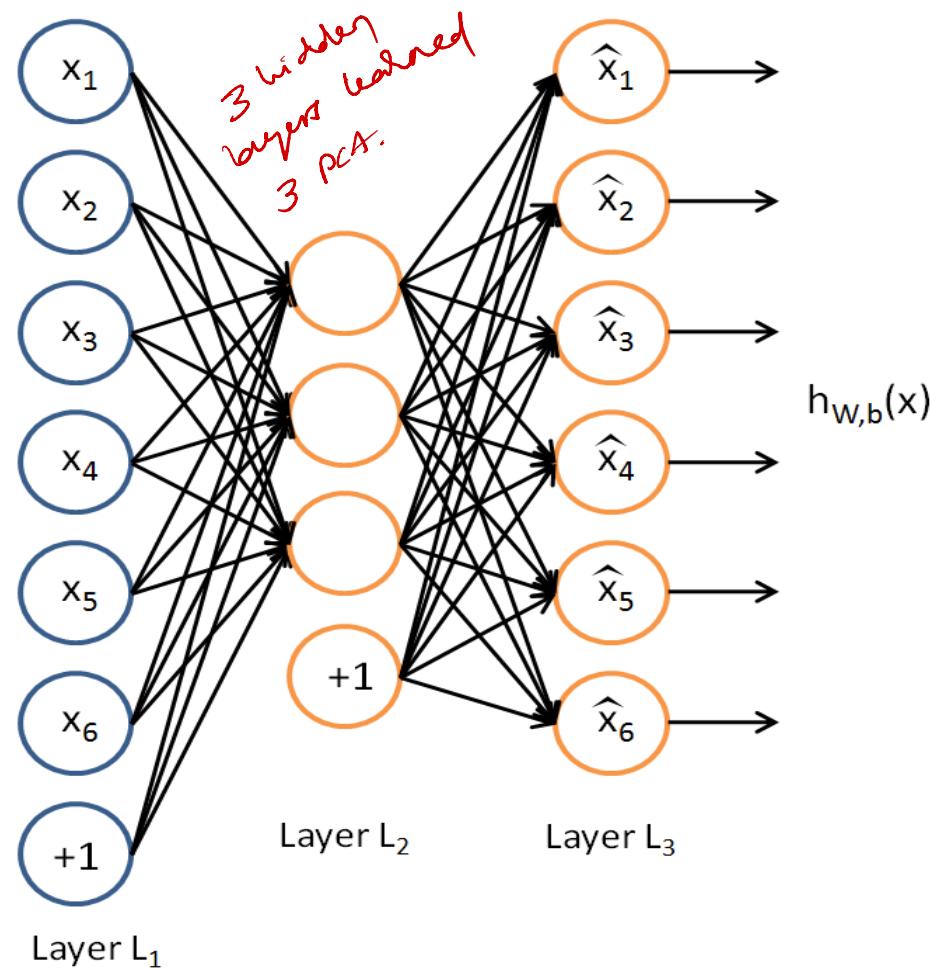
# Undercomplete Autoencoder

- Constrain the code to have smaller dimension than the input
- Training: minimize a loss function (e.g., MSE)
- $e = g(f(x)) - r$  *r reconstructed version of n*
- When both  $f$  and  $g$  are linear and MSE is used, the code contains just the Principal Components of the data



If  $f_1, f_2$  are non-linear (e.g., sigmoid)  
 $h$  is a non-linear version of  
PCA, because it resides on a  
non-linear surface that is closest  
to the data in Mean-squared error  
sense

# Undercomplete Autoencoder and PCA



# Undercomplete Autoencoder

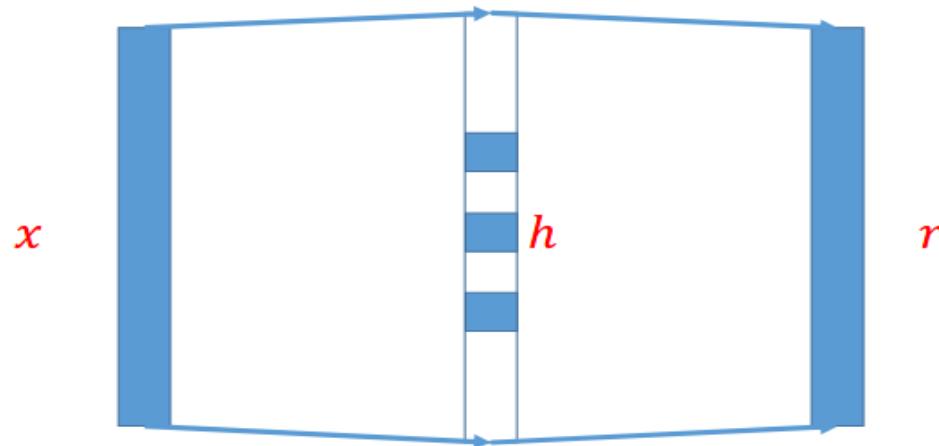
- Example of nonlinear encoder and decoder
- Capacity should not be too large
- Suppose given data
- $\mathbf{x}(1), \mathbf{x}(2), \dots, \mathbf{x}(N)$
- Encoder maps  $\mathbf{x}(i)$  to  $i$
- Decoder maps  $i$  to  $\mathbf{x}(i)$
- One dimensional  $h$  suffices for perfect reconstruction

# Regularization

- Often not performed.
- Regularized autoencoders: add regularization term that encourages the model to have other properties
  - Sparsity of the representation (sparse autoencoder)
  - Robustness to noise or to missing inputs (denoising autoencoder)
  - Smallness of the derivative of the representation (smoothness)

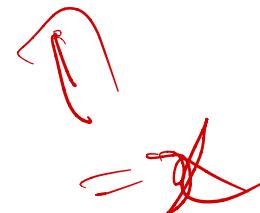
# Sparse Autoencoder

- Code is constrained to be sparse
- Code does not need to be “low dimensional”
- L1 or Probabilistic Regularization is used



# Denoising Autoencoder

- In order to force the hidden layer to discover more robust features and prevent it from simply learning the identity, we train the auto-encoder to reconstruct the input from a corrupted version of it.
- $e = g(f(x+noise)) - r$



# Denoising Auto-encoder

- The denoising auto-encoder is a *stochastic version of the auto-encoder*.
- It does two things: try to encode the input (preserve the information about the input), and try to undo the effect of a corruption process stochastically applied to the input of the auto-encoder.

# Visualizing Auto-encoder

- Having trained a (sparse) auto-encoder, we would now like to visualize the function learned by the algorithm, to try to understand what it has learned.
- Consider the case of training an auto-encoder on  $10 \times 10$  images, so that  $p=100$ .

# Visualizing Autoencoder

- Each neuron  $i$  computes a function of the input:

$$a_i^{(1)} = f \left( \sum_{j=1}^{100} w_{ij} x_j + b_i \right)$$

- We will visualize the function computed by neuron  $i$ —which depends on the parameters  $w_{ij}^{(1)}$  (ignoring the bias term for now)—using a 2D image.

# Visualizing Autoencoder

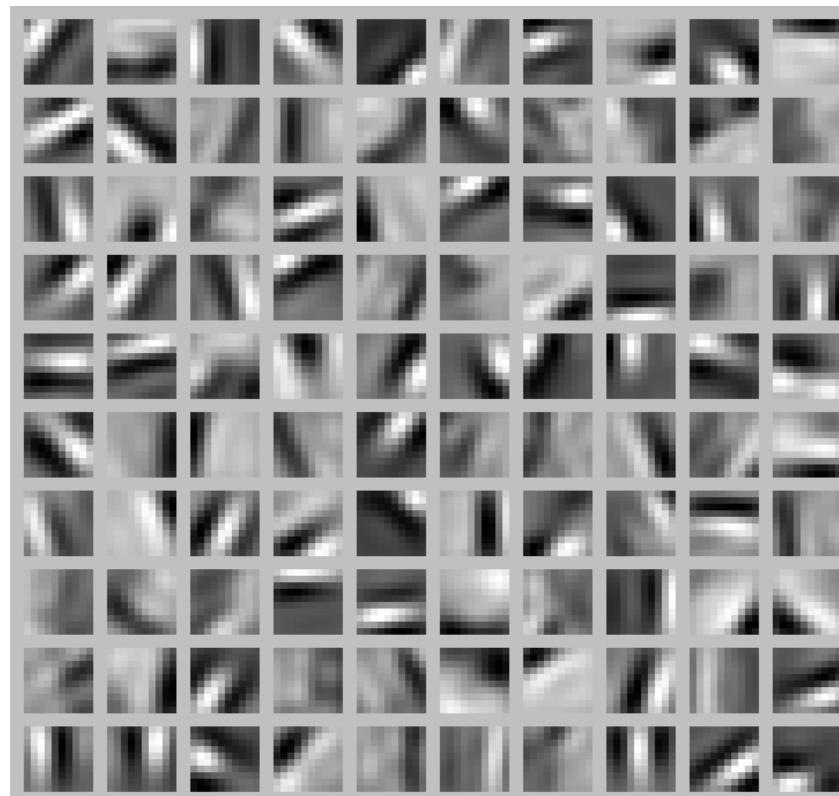
- Think of  $a_i^{(1)}$  as some non-linear feature of the input  $\mathbf{x}$ .
- What input image  $\mathbf{x}$  would cause  $a_i^{(1)}$  to be maximally activated?
- Less formally, what is the feature that neuron  $i$  is looking for?

# Visualizing Autoencoder

- If we have an auto-encoder with 100 neurons, then our visualization will have 100 such images—one per neuron.
- By examining these 100 images, we can try to understand what the ensemble of hidden units is learning.

# Visualizing Autoencoder

- When we do this for a sparse autoencoder (trained with 100 neurons on 10x10 pixel inputs) we get the following result



# Visualizing Autoencoder

- Each square in the figure shows the (normalized) input image  $\mathbf{x}$  that maximally activates one of 100 neurons.
- Different neurons have learned to detect edges at different positions and orientations in the image.

# Visualizing Autoencoder

- These features are useful for such tasks as object recognition and other vision tasks.
- When applied to other input domains (such as audio), this algorithm also learns useful representations/features for those domains too.

# Problems with Using Very Deep Networks?

- A famous problem with training very deep neural networks with sigmoid hidden units is *vanishing gradients*
- When the number of layers is large, net signals that go into each neuron can be very negative or very positive

# Problems with Using Very Deep Networks?

- Neurons become *saturated*, i.e. their output becomes close to 1 or -1
- Magnitudes of gradients in saturation areas are **very small**, so the weight updates will be very small and the network cannot learn from new data.

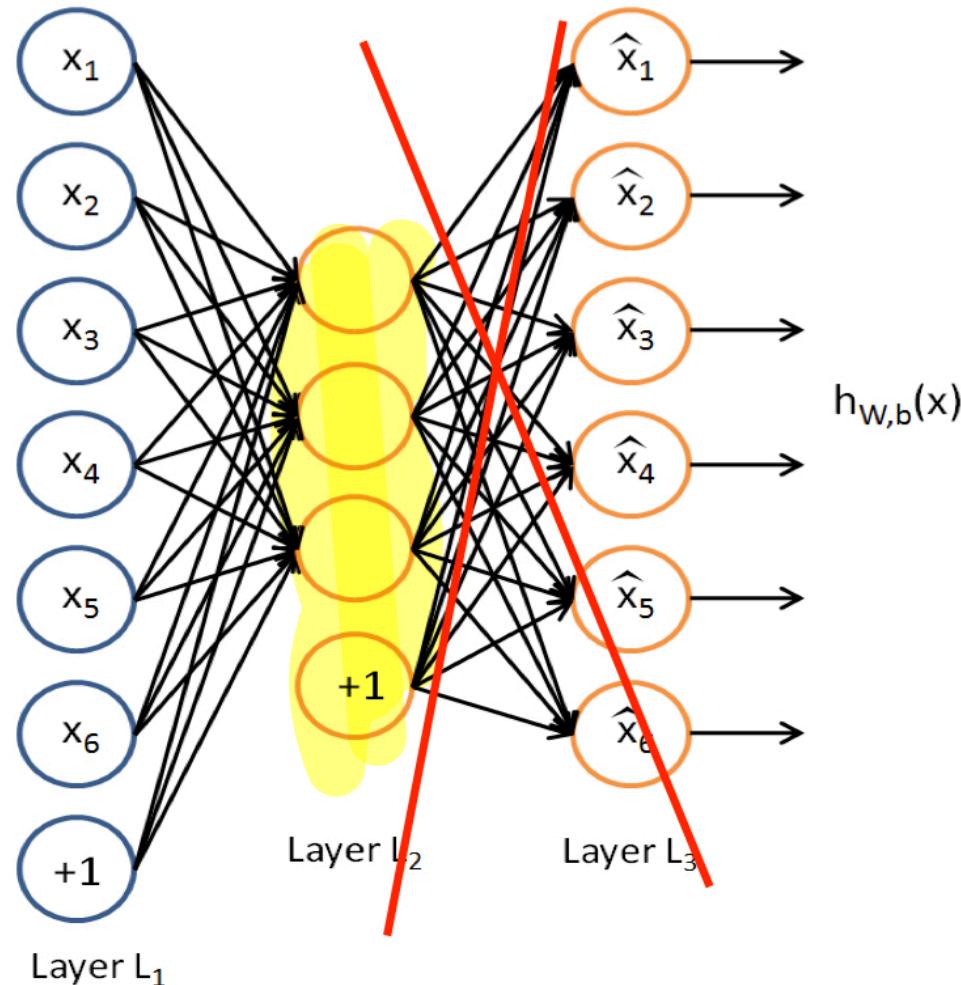
# Remedy: Pre-training

- In 2006, Hinton et al showed that **pre-training with auto-encoders** can remedy the problem of vanishing gradients.
- In basic application of backpropagation<sup>+SGD</sup> to update the weights, weights are initialized randomly
- That can easily cause **gradients to vanish**

# Remedy: Pre-training

- Instead, one can use layer-by-layer pre-training with auto-encoders
- In this method, an auto-encoder is used to learn a representation of the data.
- The weights of its output layer are discarded, but the weights from the input to hidden layer are kept as initial weights of the first layer.

# Remedy: Pre-training



# Remedy: Pre-training

- The outputs of the hidden layer can then be used to train a second auto-encoder.
- Again, the output weights of this auto-encoder are **discarded**.
- Only the weights from the input to hidden neurons of the second auto-encoder are used as initial weights of the second layer

# Remedy: Pre-training

- This process continues by **adding layers** and training an auto-encoder for the output of the last layer.
- Layer by layer pre-training made training deep networks possible
- The initial pre-trained weights act as good “guesses” that guide the algorithm to minimum error.

# Is Pre-training Outdated?

- Some think so!
- Reddit User:

“...but the last few years the pre-training approach has been largely obsoleted.

Nowadays, deep neural networks are a lot more similar to their 80's cousins. Instead of pre-training, the **difference** is now in the **activation functions** and **regularisation methods** used (and sometimes in the optimisation algorithm, although much more rarely).”

# Is Pre-training Outdated?

- Some think so!
- See this discussion by a Famous Reddit User and the responses:

([https://www.reddit.com/r/MachineLearning/comments/22u1yt/is\\_deep\\_learning\\_basically\\_just\\_neural\\_networks/cgqgy9w/](https://www.reddit.com/r/MachineLearning/comments/22u1yt/is_deep_learning_basically_just_neural_networks/cgqgy9w/))

“...but the last few years the pre-training approach has been largely obsoleted.

Nowadays, deep neural networks are a lot more similar to their 80's cousins. Instead of pre-training, the **difference** is now in the **activation functions** and **regularisation methods** used (and sometimes in the optimisation algorithm, although much more rarely).”

# Is Pre-training Outdated?

- Some think so!
- Famous Reddit User:  
“...the "pre-training era", which started around 2006, ended in the early '10s when people started using rectified linear units (ReLUs), and later dropout, and discovered that pre-training was no longer beneficial for this type of networks.”

# Is Pre-training Outdated?

- Some think so!
- Famous Reddit User:  
“ReLUs (and modern variants such as maxout) suffer significantly less from the vanishing gradient problem. Dropout is a strong regulariser that helps ensure the solution we get generalises well. These are precisely the two issues pre-training sought to solve, but they are now solved in different ways.”

# Is Pre-training Outdated?

- Some think so!
- Nonetheless, it is still used by the industry to some extent.
- It is very useful if **labeled data is not enough** to avoid overfitting by using dropout.
- It can be useful when unlabeled data is abundant: <https://arxiv.org/pdf/1412.6597.pdf>

# Appendix

# Adversarial Training

# Adversarial Training

- Machine learning techniques were originally designed for stationary environments in which the training and test data **are assumed to be generated from the same (although possibly unknown) distribution.**

# Adversarial Training

- In the presence of **intelligent and adaptive adversaries**, however, this working hypothesis is likely to be violated to at least some degree (depending on the adversary).

# Adversarial Training

- In fact, a **malicious adversary** can carefully manipulate the input data exploiting specific **vulnerabilities of learning algorithms** to compromise the whole system security.

# Do nets have Human-level understanding?

- In many cases, neural networks have begun to reach human level performance when evaluated on an i.i.d. test set
  - Have they reached human level understanding?
- To probe the level of understanding we can probe examples that model misclassifies
  - Even neural networks that perform at human level accuracy have a 100% error rate on examples intentionally constructed!

# Adversarial examples

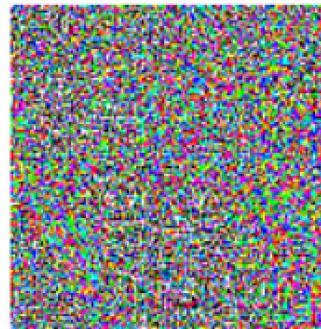
- An optimization procedure is used to search for an input  $\mathbf{x}'$  near data point  $\mathbf{x}$  such that the model output is very different at  $\mathbf{x}'$ 
  - In many cases,  $\mathbf{x}'$  can be so similar to  $\mathbf{x}$  that a human observer cannot tell the difference between the original example and the adversarial example
  - But the network makes a highly different prediction

# Adversarial Example Generation

We add to  $\mathbf{x}$  an imperceptibly small vector  
Its elements are equal to the sign of the elements of the  
gradient of the cost function with respect to the input. It  
changes Googlenet's classification of the image



$$+ .007 \times$$



=



$\mathbf{x}$

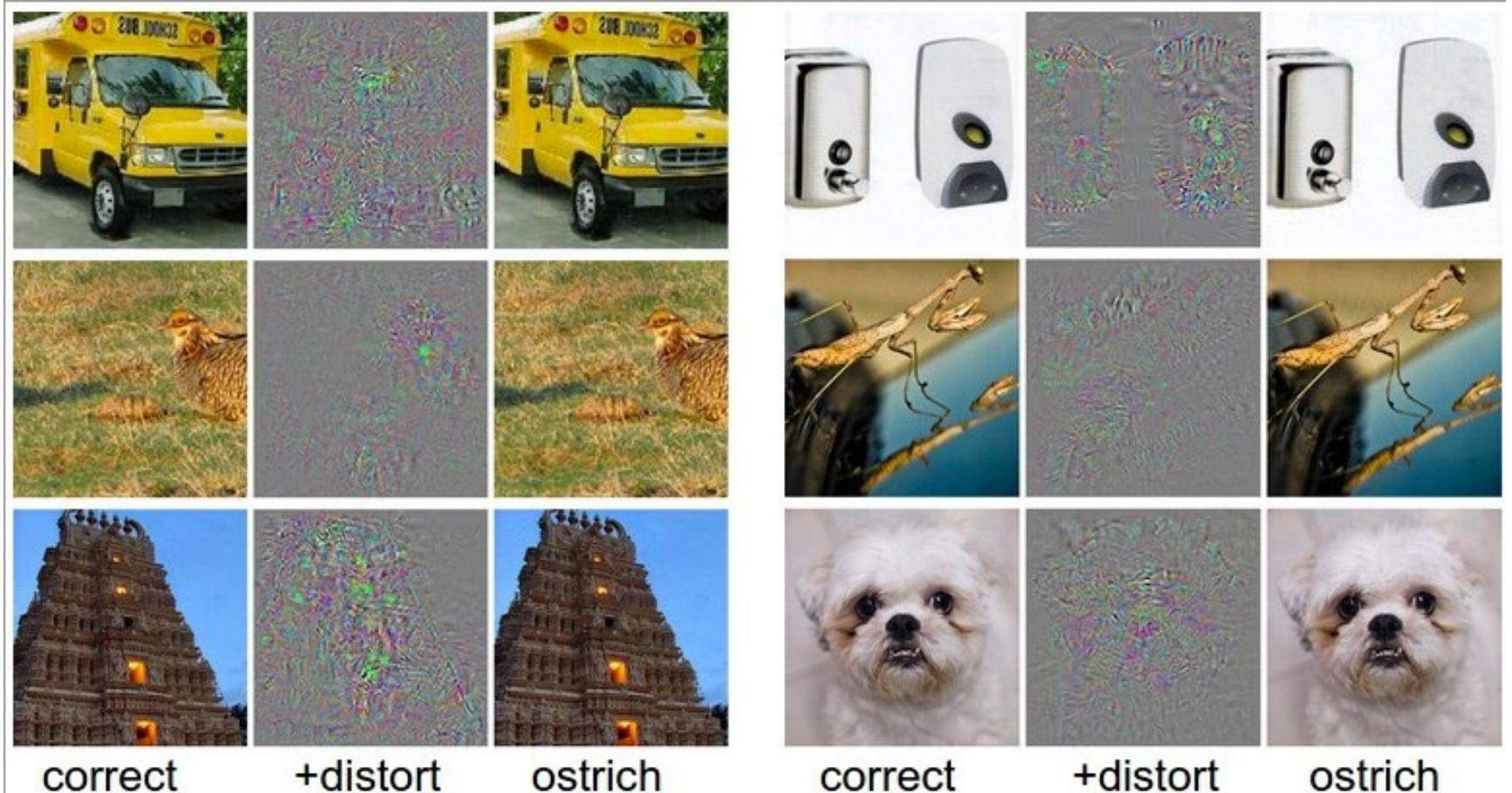
$y = \text{"panda"}$   
with 58% confidence

$\text{sign}(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y))$   
 $y = \text{"nematode"}$   
With 8.2% confidence

$\mathbf{x} +$   
 $\epsilon \text{sign}(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y))$   
 $y = \text{"gibbon"}$   
With 99% confidence

<https://arxiv.org/pdf/1412.6572.pdf>

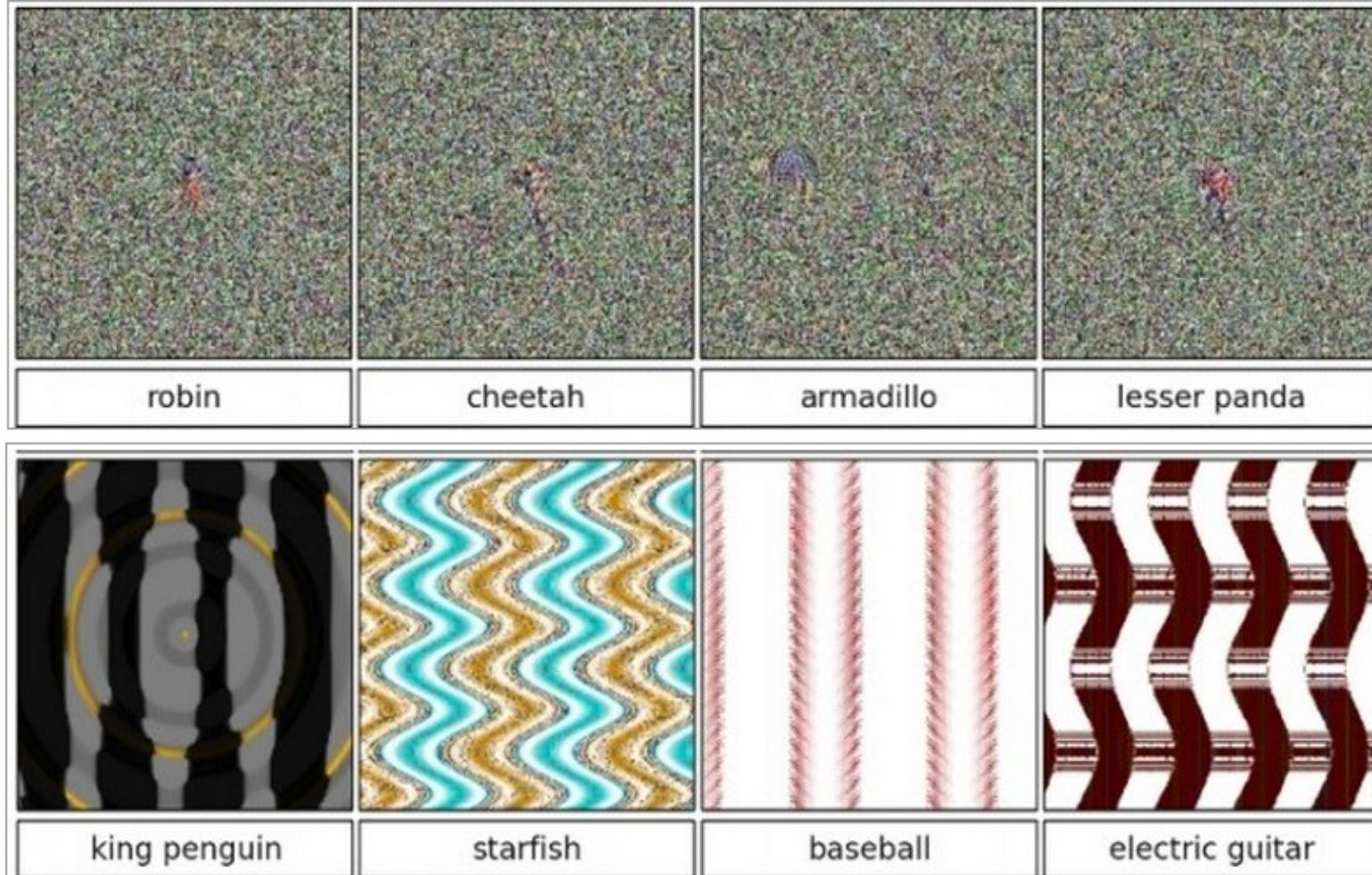
# More examples



Take a correctly classified image (left image in both columns), and add a tiny distortion (middle) to fool the ConvNet with the resulting image (right).

<http://karpathy.github.io/2015/03/30/breaking-convnets/>

# Some more examples



These images are classified with >99.6% confidence as the shown class by a Convolutional Network.

[http://karpathy.github.io/2015/03/30/breakin  
g-convnets/](http://karpathy.github.io/2015/03/30/breaking-convnets/)

# Uses of adversarial training

- Adversarial examples have many implications
  - E.g., they are useful in computer security
    - Adversarial examples are hard to defend against
  - They are interesting in the context of regularization
    - Using adversarially perturbed samples we can reduce error rate on test set

# Cause of adversarial examples

- Primary cause is **excessive** linearity
  - Neural networks are built primarily out of linear building blocks
    - The overall function often proves to be (close to) linear
  - Linear functions are easy to optimize
  - But the value of a linear function can change rapidly with numerous inputs
  - If we change input by  $\epsilon$  then a linear functions with weights  $w$  can change by  $\epsilon||w||$  which can be very large in high-dimensional spaces

# Adversarial Training

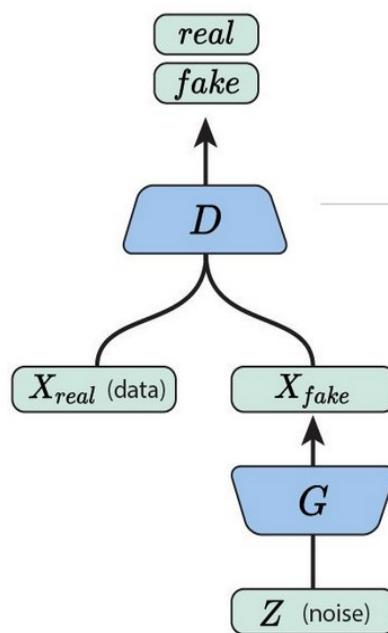
- Adversarial training discourages highly sensitive local behavior
- By encouraging network to be **locally constant** in the neighborhood of the training data
- This can be seen as a way of explicitly introducing a local constancy prior into supervised neural nets

# Adversarial Training

- Locally constant behavior is **robust** to adversarial examples in comparison to linear models such as logistic regression.
- Neural networks are able to represent functions that can range from nearly linear to nearly locally constant
  - Thus can capture linear trends as well as learning to resist local perturbation

# Generative Adversarial Network

- GANs are a way to make a generative model by having two neural networks compete with each other



The discriminator tries to distinguish genuine data from forgeries created by the generator

The generator turns random noise into imitations of the data, in an attempt to fool the discriminator

# Generative Adversarial Network

- Training the **discriminator** involves presenting it with samples from the dataset, until it reaches some level of accuracy.
- Typically the generator is **seeded with a randomized input**

# Generative Adversarial Network

- Thereafter, samples synthesized by the generator are evaluated by the discriminator.
- Backpropagation is applied in both networks so that the generator produces better images, while the discriminator becomes more skilled at flagging synthetic images.

# Appendix

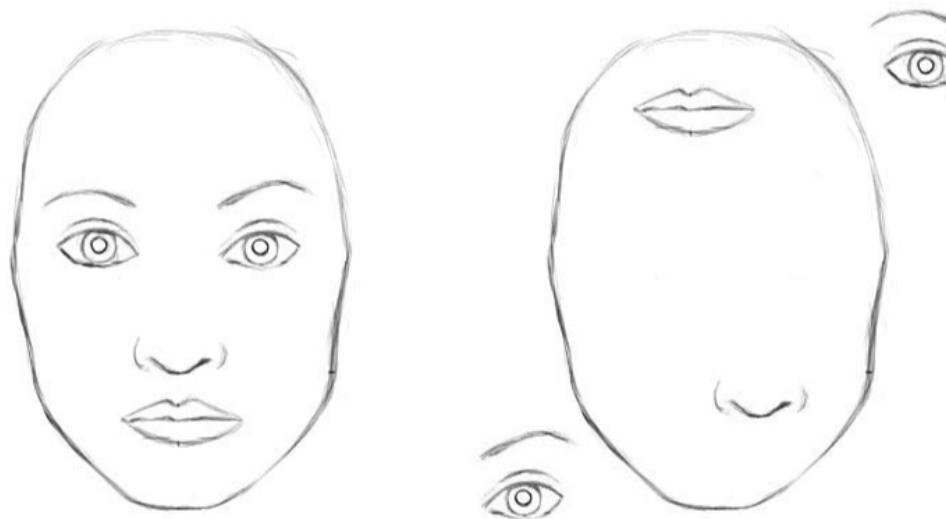
## Capsule Networks

# CNNs Have Important Drawbacks

- For a CNN, a mere presence of some objects can be a very strong indicator to consider that there is a pattern in the image.
- Orientational and relative spatial relationships between these components are not very important to a CNN.

# Orientation? Spatial Relationships?

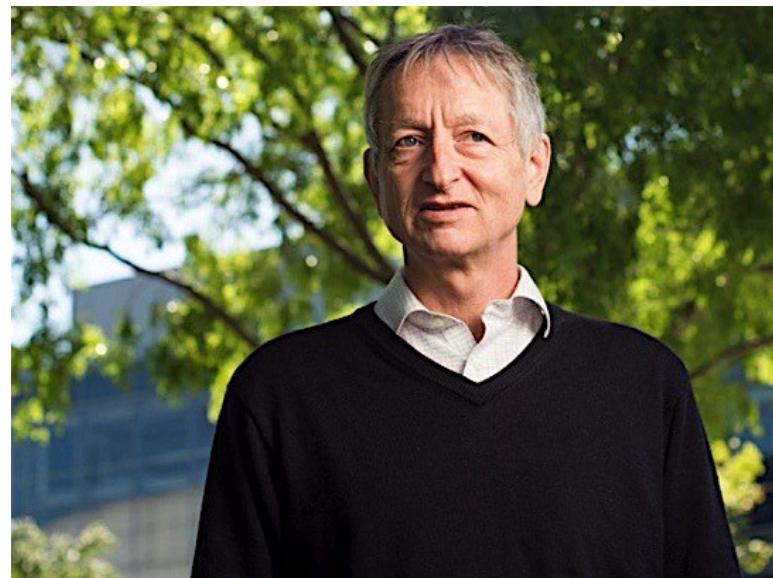
- Not considered important by design!



- To a CNN, both pictures are similar, since they both contain similar elements

# Max Pooling: to praise or to blame?

- *Hinton: “The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster.”*



# Max Pooling: to praise or to blame?

- Hinton:
  - Brains **deconstruct** a hierarchical representation of the world around us from **visual information received by eyes** and try to match it with already learned patterns and stored relationships.
  - **Representation of objects in the brain does not depend on view angle**

# Max Pooling: to praise or to blame?

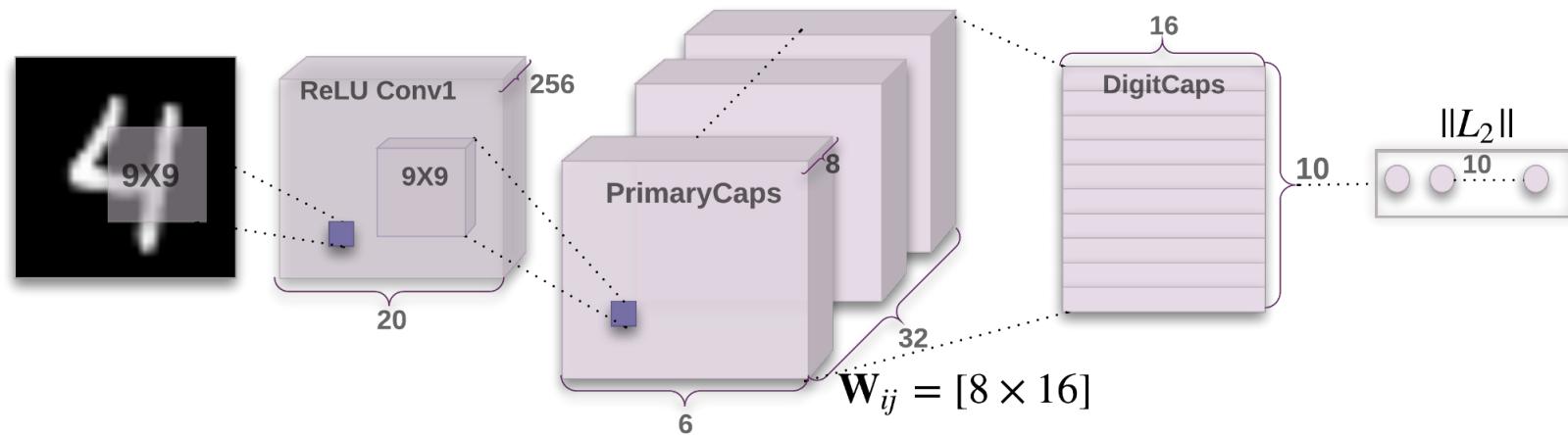
- Hinton:
  - Brains **deconstruct** a hierarchical representation of the world around us from **visual information received by eyes** and try to match it with already learned patterns and stored relationships.

# Max Pooling: to praise or to blame?

- **Representation of objects in the brain does not depend on view angle**
- A CNN is easily confused when viewing an image in a different orientation.
- One way to combat this is with **excessive training of all possible angles**, but this takes a lot of time and seems counter intuitive.
- CNN is susceptible to adversarial patterns

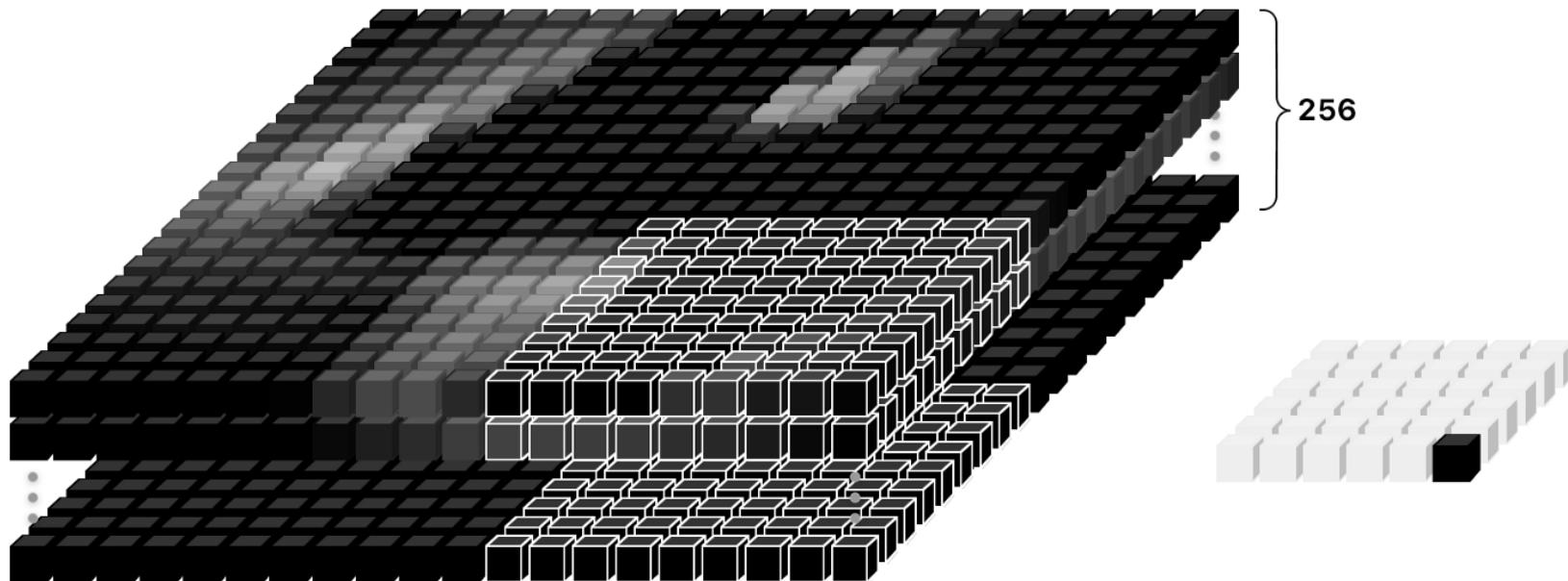
# Capsule Networks

- The first part of CapsNet is a traditional convolutional layer passed through ReLU. Depth = 256.



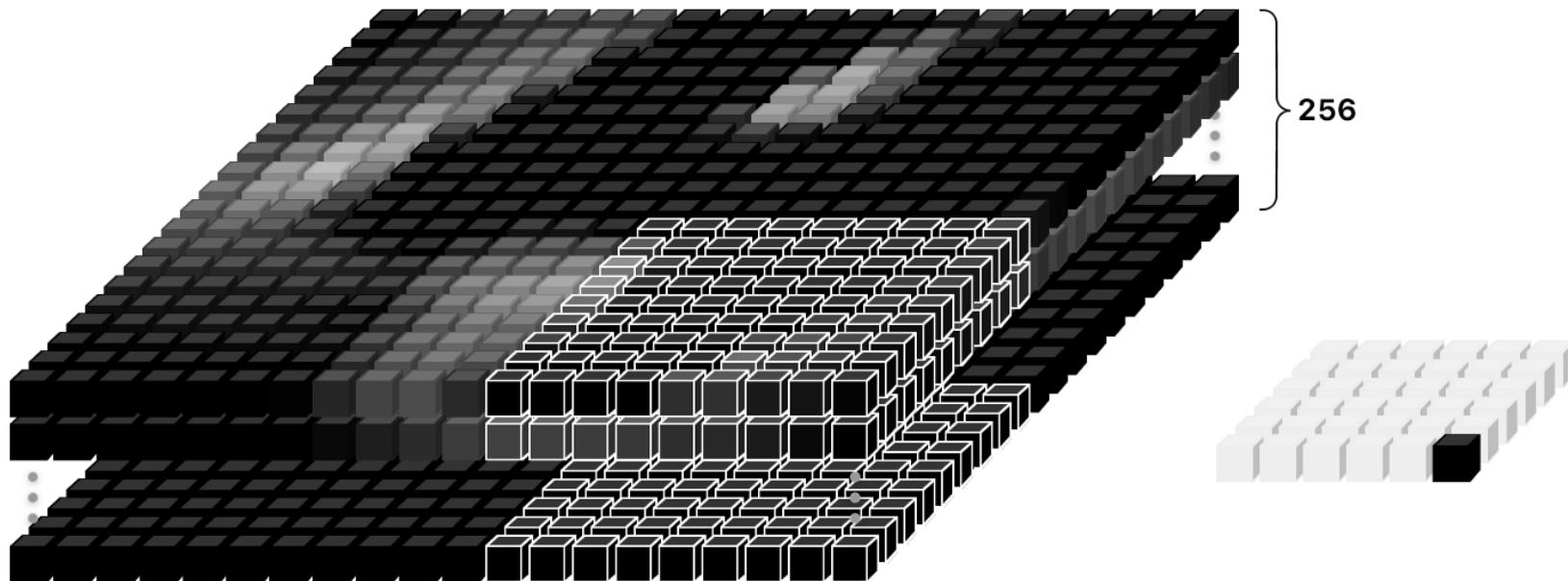
# Primary Caps

- The PrimaryCaps layer starts off as a normal convolution layer, but this time we are convolving over the stack of outputs from the previous convolutions.



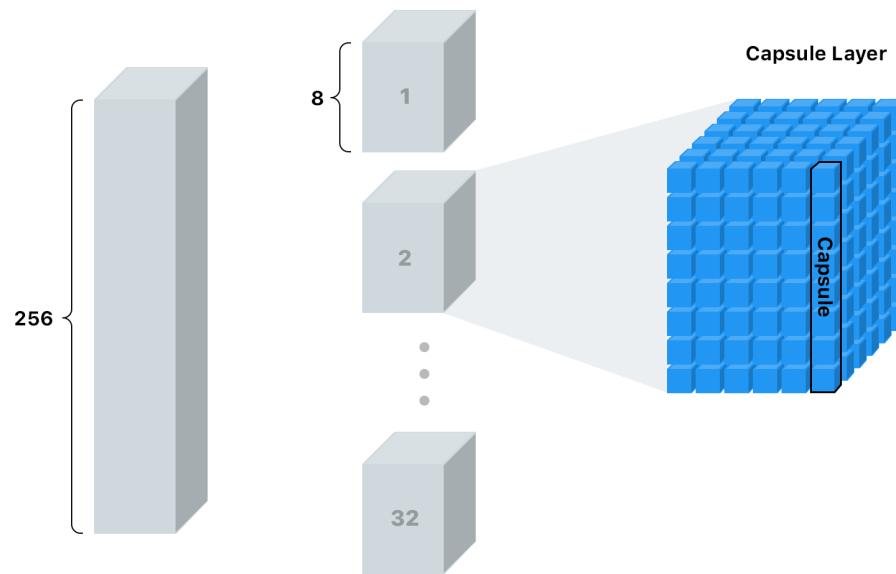
# Primary Caps

- 256 6x6 feature maps
- Strides are >1, e.g. 2, to **rapidly reduce dimensions**
- Looking for slightly more complex shapes from the features (e.g. edges) found earlier



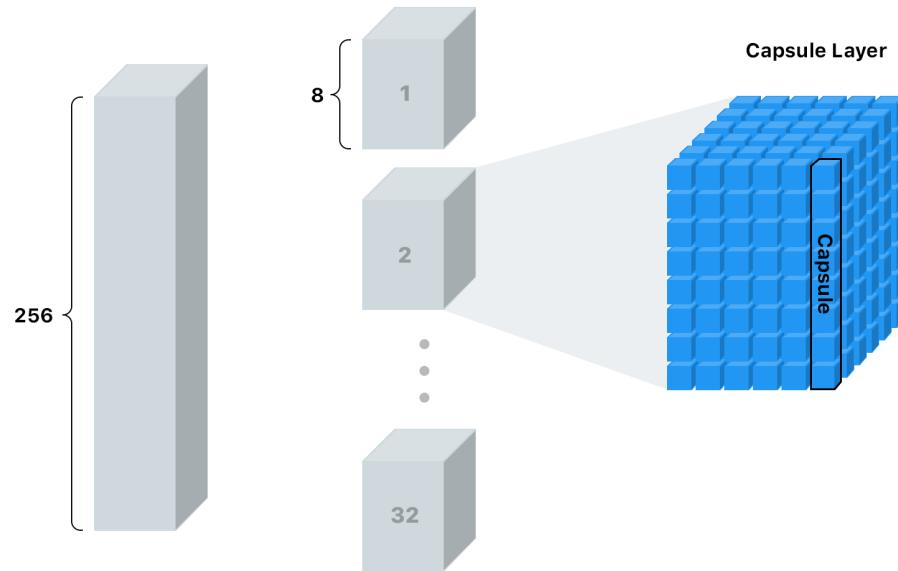
# Primary Caps

- The capsule operation reshapes 256 6x6 outputs to 32 8x6x6 outputs
- This is basically slicing the cube into 32 smaller cubes, called “capsules”



# Squashing (Instead of ReLU)

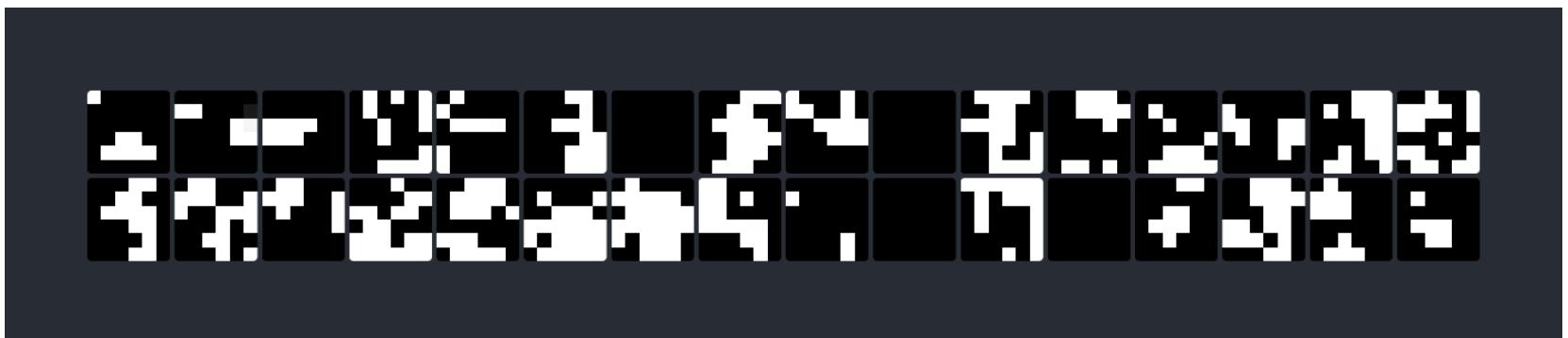
Each capsule is squashed (each vector of 8 elements is re-scaled so that its length is between 0 and 1 but its direction is not changed.)



$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1 + \|\mathbf{s}_j\|^2} \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$$

# Squashing (Instead of ReLU)

32 6X6 squares. Each pixel represents the length of a vector of 8 elements.



# Routing by Agreement

The capsules in the next layers also try to detect objects and their pose, but they work very differently, using an algorithm called routing by agreement. This is where most of the magic of CapsNets lies.

<https://www.youtube.com/watch?v=pPN8d0E3900>

<https://www.youtube.com/watch?v=YqazfBLLV4U>

# Appendix

- Representations of Natural Language Inputs and Outputs

# Representations of Natural Language Inputs and Outputs

- When words are output at each time step, generally the output consists of a **softmax** vector  $\mathbf{y}^{(t)}$  with *K elements* where  $K$  is the size of the vocabulary.
- Words are encoded with a binary (one hot) coding.

# Representations of Natural Language Inputs and Outputs

- A softmax layer is an element-wise logistic function that is normalized so that all of its components sum to one.
- Intuitively, these outputs correspond to the probabilities that each word is the correct output at that time step.

# Representations of Natural Language Inputs and Outputs

- Another simple architecture feeds the input one character at a time, and generates output one character at a time.
- Output is a softmax layer and characters are encoded with a binary (one-hot) encoding.

# Representations of Natural Language Inputs and Outputs

- One hot encoding is inefficient, requiring as many bits as the vocabulary is large.
- It offers no direct way to capture different aspects of similarity between words in the encoding itself.

# Distributed Representation of Natural Language

- It is common now to model words with a **distributed representation** using a **meaning vector** .
- The meanings are either learned given a large corpus of supervised data, or are based on word **co-occurrence statistics**.

# Word Vectors

- Freely available code to produce word vectors from these statistics include **GloVe** and **word2vec**

# Word2vec

- **Word2vec** is a group of related models used to produce word **embeddings**.
- These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words.

# Word2vec

- **Word2vec:**
- **Input:** a large corpus of text
- **Output:** a vector space of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space.
- Words **sharing common contexts** in the corpus are **located close to one another** in the space

# Evaluation of RNNs for Language Modeling: Obstacles

- A serious obstacle: outputs are variable length sequences of words.
- Captioning or translation: multiple correct translations.
- A labeled dataset may contain **multiple reference** translations for each example.
  - Comparing against such a gold standard is harder than applying performance measures to binary classification.

# N-Grams

- An ***n*-gram** is a sequence of *n* items from a given sample of text or speech.
  - phonemes, syllables, letters, words or base pairs according to the application.
- When the items are words, *n*-grams may also be called ***shingles***

# BLEU score

- It is the geometric mean of the n-gram precisions for all values of  $n$  between 1 and some upper limit  $N$ .
- In practice, 4 is a typical value for  $N$ .

# BLEU score: Brevity

Precision can be made high by offering excessively short translations, so the BLEU score includes a brevity penalty  $B$ :

$$B = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}.$$

$c$  is average the length of the candidate translations and  $r$  the average length of the reference translations.

# BLEU score

The BLEU Score is:

$$BLEU = B \cdot \exp \left( \frac{1}{N} \sum_{n=1}^N \log p_n \right)$$

# BLEU score

$$BLEU = B \cdot \exp \left( \frac{1}{N} \sum_{n=1}^N \log p_n \right)$$

$p_n$  : the modified n-gram precision, which is the number of n-grams in the candidate translation that occur in any of the reference translations, divided by the total number of n-grams in the candidate translation.

# Unigram Metric

|                    |       |     |     |     |     |     |     |     |
|--------------------|-------|-----|-----|-----|-----|-----|-----|-----|
| <b>Candidate</b>   | the   | the | the | the | the | the | the | the |
| <b>Reference 1</b> | the   | cat | is  | on  | the | mat |     |     |
| <b>Reference 2</b> | there | is  | a   | cat | on  | the | mat |     |

Of the seven words in the candidate translation, all of them appear in the reference translations. Thus the candidate text is given a unigram precision of  
**Words in the cand. found in ref/ total number of words in cand.**=7/7

# BLEU Modified Precision

|                    |       |     |     |     |     |     |     |     |
|--------------------|-------|-----|-----|-----|-----|-----|-----|-----|
| <b>Candidate</b>   | the   | the | the | the | the | the | the | the |
| <b>Reference 1</b> | the   | cat | is  | on  | the | mat |     |     |
| <b>Reference 2</b> | there | is  | a   | cat | on  | the | mat |     |

For each word in the candidate translation, the algorithm takes its maximum total count  $m_{\max}$  in any of the reference translations. In the example above, the word "the" appears twice in reference 1, and once in reference 2. Thus  $m_{\max} = 2$

# BLEU Modified Precision

|                    |       |     |     |     |     |     |     |     |
|--------------------|-------|-----|-----|-----|-----|-----|-----|-----|
| <b>Candidate</b>   | the   | the | the | the | the | the | the | the |
| <b>Reference 1</b> | the   | cat | is  | on  | the | mat |     |     |
| <b>Reference 2</b> | there | is  | a   | cat | on  | the | mat |     |

For the candidate translation, the count  $m_w$  of each word is clipped to a maximum of  $m_{\max}$  for that word

# BLEU Modified Precision

|                    |       |     |     |     |     |     |     |     |
|--------------------|-------|-----|-----|-----|-----|-----|-----|-----|
| <b>Candidate</b>   | the   | the | the | the | the | the | the | the |
| <b>Reference 1</b> | the   | cat | is  | on  | the | mat |     |     |
| <b>Reference 2</b> | there | is  | a   | cat | on  | the | mat |     |

These clipped counts are then summed over all distinct words in the candidate. This sum is then divided by the total number of words (n-grams) in the candidate translation. In the above example, the modified unigram precision score would be:

**2/7**

# BLEU Modified Precision

Comparing metrics for candidate "the the cat"

| Model   | Set of grams         | Score                           |
|---------|----------------------|---------------------------------|
| Unigram | "the", "the", "cat"  | $\frac{1 + 1 + 1}{3} = 1$       |
| Bigram  | "the the", "the cat" | $\frac{0 + 1}{2} = \frac{1}{2}$ |

|                    |       |     |    |     |     |     |     |
|--------------------|-------|-----|----|-----|-----|-----|-----|
| <b>Reference 1</b> | the   | cat | is | on  | the | mat |     |
| <b>Reference 2</b> | there | is  | a  | cat | on  | the | mat |

# Research Other Methods

What are other evaluation methods?

Research them!

Example: METEOR

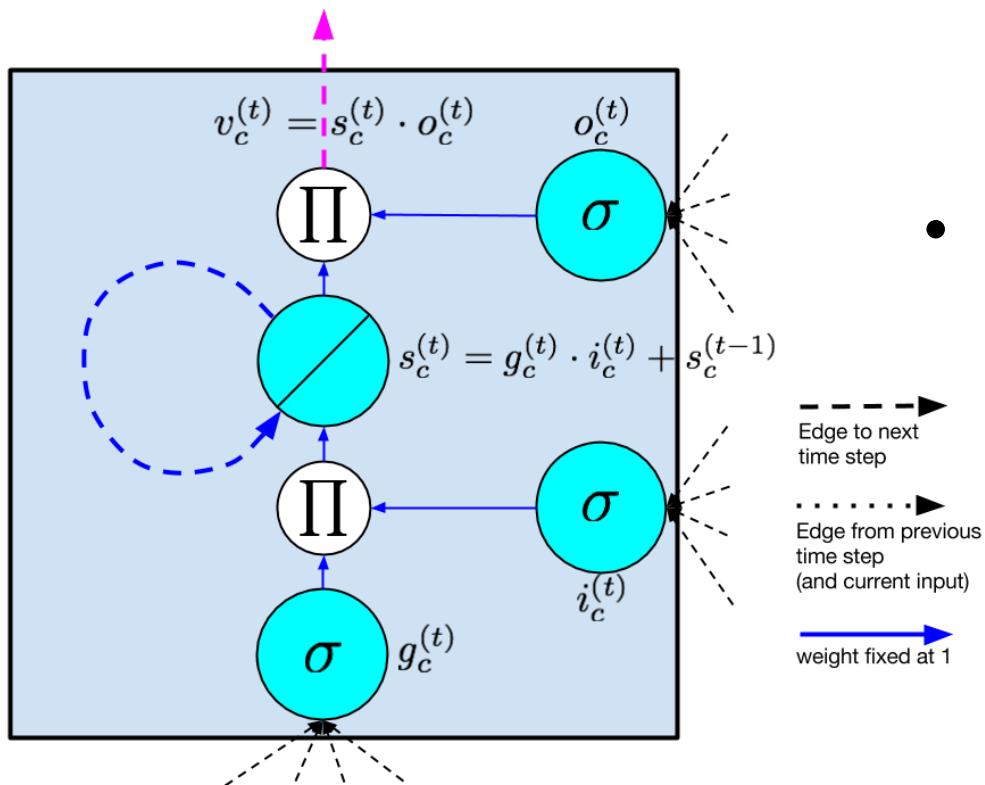
# Appendix

Another  
Representation of  
LSTM

# The LSTM

This model resembles a standard recurrent neural network with a hidden layer, but each ordinary Neuron in the hidden layer is replaced by a *memory cell*.

# The LSTM: Memory Cell



- One LSTM memory cell. The self-connected node is the internal state  $s$ .
- The diagonal line indicates that the identity link function is applied.
- The blue dashed line is the recurrent edge, which has fixed unit weight.
- Nodes marked  $\Pi$  output the product of their inputs. All edges into and from  $\Pi$  nodes also have fixed unit weight.

# The LSTM: Memory Cell

- Each memory cell contains a node with a **self-connected recurrent edge** of fixed weight one, ensuring that the **gradient can pass across many time steps** without vanishing or exploding.
- To distinguish references to a memory cell and not an ordinary neuron, we use the subscript  $c$  .

# The LSTM: Input Node

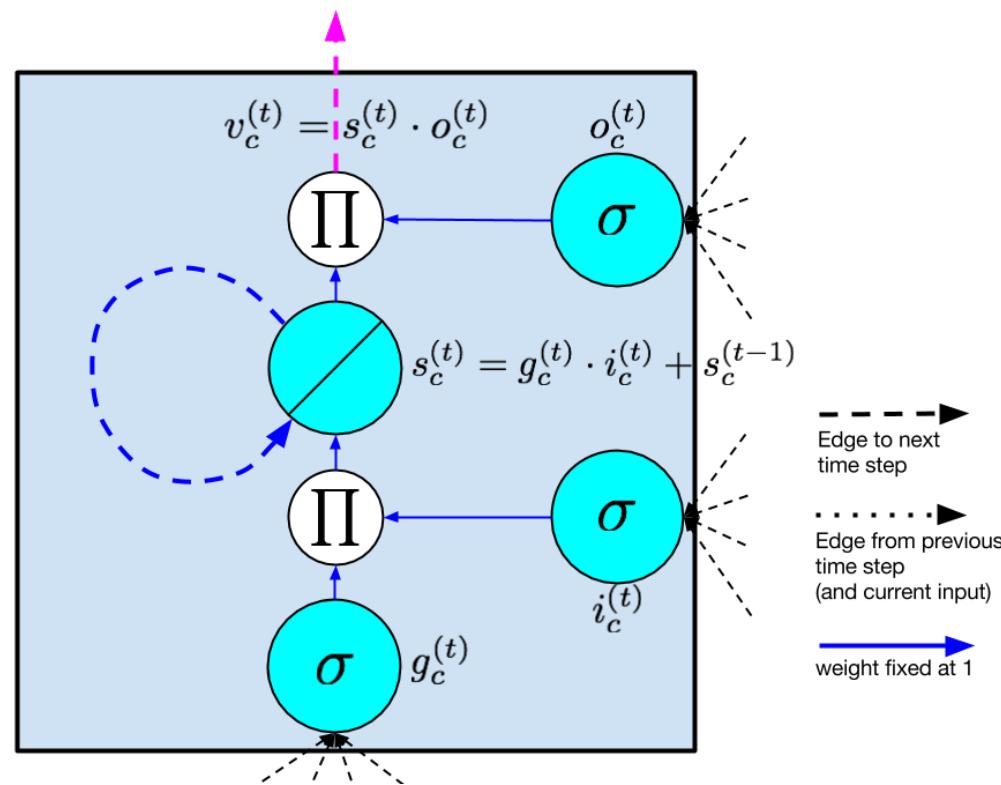
- Labeled  $g_c$ , it takes activation from the input layer  $\mathbf{x}^{(t)}$  at the current time step and (along recurrent edges) from the hidden layer at the previous time step  $\mathbf{h}^{(t-1)}$ .
- Typically, the summed weighted input is run through a *tanh* (or *sigmoid*) activation function.

# The LSTM: Input Gate

- Gates are a distinctive feature of the LSTM approach.
- A gate is a sigmoidal unit that, like the input node, takes activation from the current data point  $x^{(t)}$  as well as from the hidden layer at the previous time step.

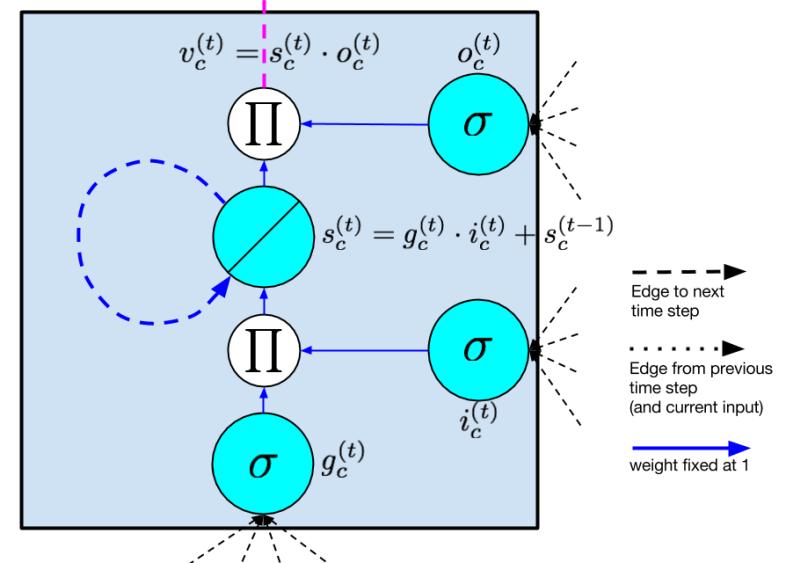
# The LSTM: Input Gate

- A gate is so-called because its value is used to multiply the value of another node.



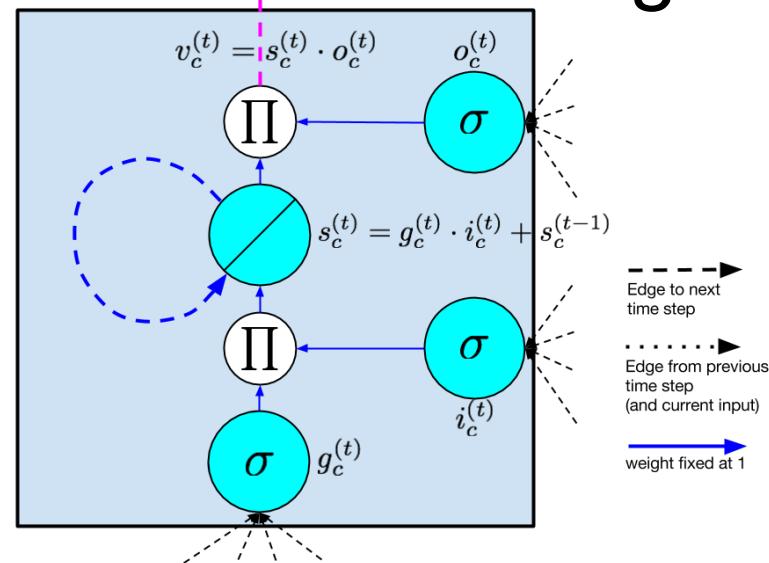
# The LSTM: Input Gate

- It is a gate because if its value is zero, then flow from the other node is cut off.
- If the value of the gate is one, all flow is passed through. The value of the input gate  $i_c$  multiplies the value of the input node.



# The LSTM: Internal State

- The heart of the memory cell is a node  $s_c$  with linear activation
- $s_c$  has a self-connected recurrent edge with fixed unit weight. Because this edge spans adjacent time steps with constant weight, error can flow across time steps without vanishing or exploding.

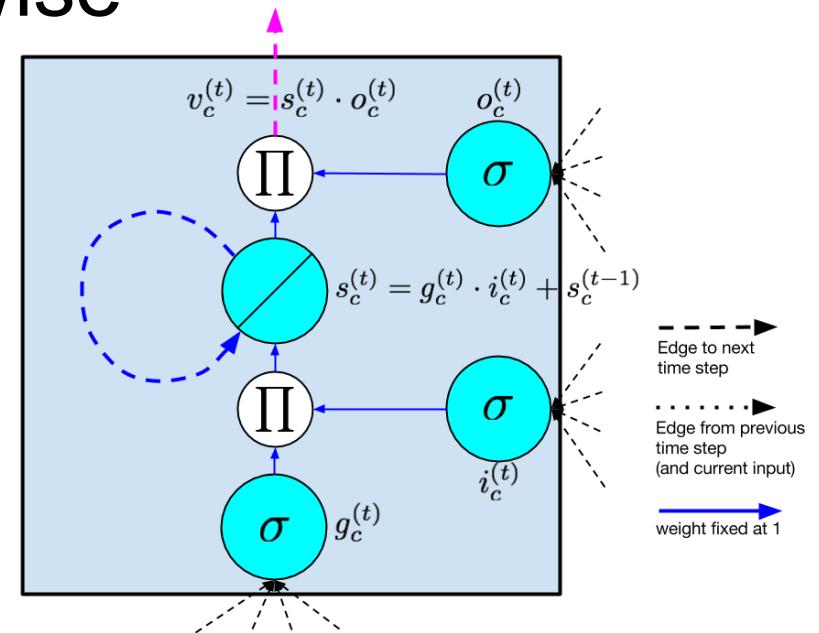


# The LSTM: Internal State

- This constant edge is often called the **constant error carousel**.
- In vector notation, the update for the internal state is:

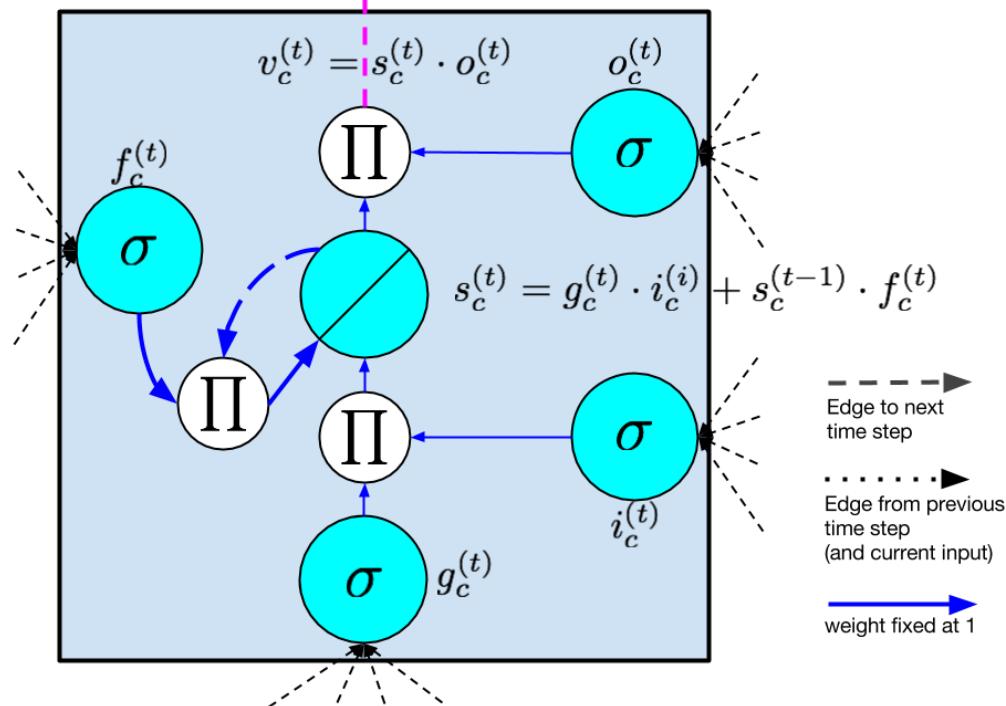
$$\mathbf{s}^{(t)} = \mathbf{g}^{(t)} \odot \mathbf{i}^{(t)} + \mathbf{s}^{(t-1)}$$

where  $\odot$  is elementwise multiplication.



# The LSTM: Forget Gate

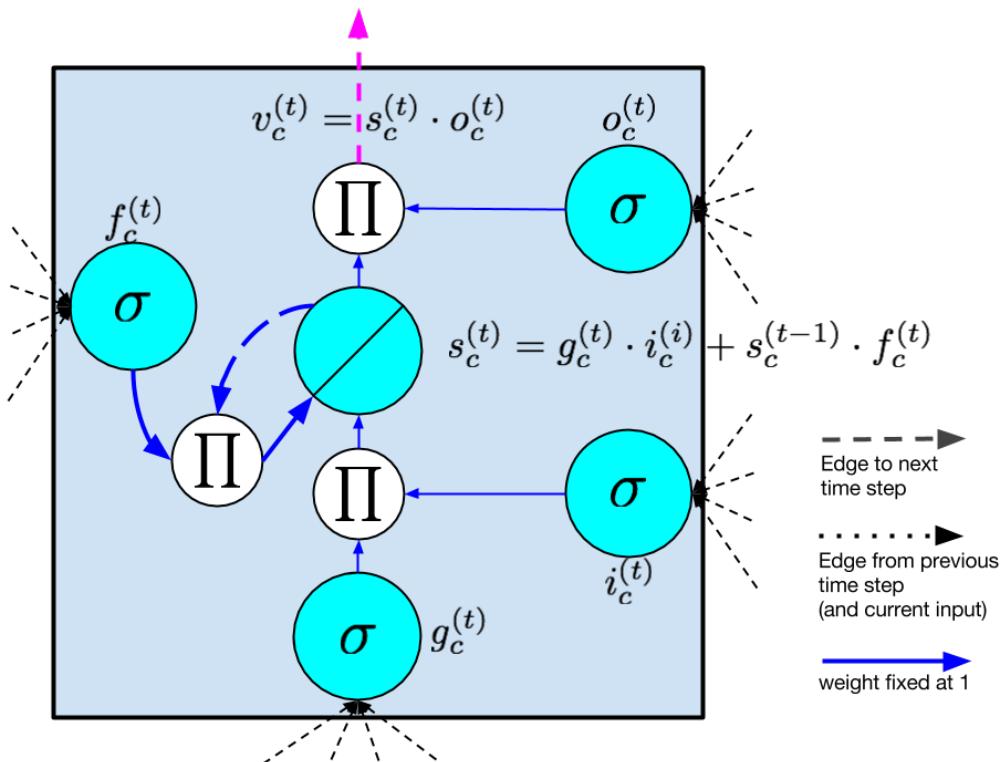
- Forget gates provide a method by which the network can learn to **flush the contents** of the internal state.
- This is especially useful in continuously running networks.



# The LSTM: Forget Gate

- With forget gates, the equation to calculate the internal state on the forward pass is

$$\mathbf{s}^{(t)} = \mathbf{g}^{(t)} \odot \mathbf{i}^{(t)} + \mathbf{f}^{(t)} \odot \mathbf{s}^{(t-1)}$$



# The LSTM: Output Gate

The value  $v_c$  ultimately produced by a memory cell is the value of the internal state  $s_c$  multiplied by the value of the output gate  $o_c$ .

$$v^{(t)} = s^{(t)} \odot o^{(t)}$$

It is customary that the internal state first be run through a tanh activation function, as this gives the output of each cell the same dynamic range as an ordinary tanh hidden unit.

# The LSTM: Output Gate

However, in other neural network research, rectified linear units, which have a greater dynamic range, are easier to train. Thus it seems plausible that the nonlinear function on the internal state might be omitted.

# The LSTM: Equations

$$\mathbf{g}^{(t)} = \phi(\mathbf{W}_{gx}^T \mathbf{x}^{(t)} + \mathbf{W}_{gh}^T \mathbf{h}^{(t-1)} + \mathbf{b}_g)$$

$$\mathbf{i}^{(t)} = \sigma(\mathbf{W}_{ix}^T \mathbf{x}^{(t)} + \mathbf{W}_{ih}^T \mathbf{h}^{(t-1)} + \mathbf{b}_i)$$

$$\mathbf{f}^{(t)} = \sigma(\mathbf{W}_{fx}^T \mathbf{x}(t) + \mathbf{W}_{fh}^T \mathbf{h}^{(t-1)} + \mathbf{b}_f)$$

$$\mathbf{o}^{(t)} = \sigma(\mathbf{W}_{ox}^T \mathbf{x}^{(t)} + \mathbf{W}_{oh}^T \mathbf{h}^{(t-1)} + \mathbf{b}_o)$$

$$\mathbf{s}^{(t)} = \mathbf{g}(t) \odot \mathbf{i}^{(t)} + \mathbf{s}^{(t-1)} \odot \mathbf{f}^{(t)}$$

$$\mathbf{h}^{(t)} = \phi(\mathbf{s}^{(t)}) \odot \mathbf{o}^{(t)}$$

$\phi$ :tanh

$\sigma$ : sigmoid

# The LSTM: How It Works

Intuitively, in terms of the forward pass, the LSTM can learn **when to let activation into the internal state**. As long as the input gate takes value zero, no activation can get in.

$$\mathbf{s}^{(t)} = \mathbf{g}(\mathbf{t}) \odot \mathbf{i}^{(t)} + \mathbf{s}^{(t-1)} \odot \mathbf{f}^{(t)}$$

# The LSTM: How It Works

Similarly, the output gate learns when to let the value out. When both gates are closed , the activation is trapped in the memory cell, neither growing nor shrinking, nor affecting the output at intermediate time steps.

# The LSTM: How It Works

- In terms of the backwards pass, the constant error carousel enables the gradient to propagate back across many time steps, neither exploding nor vanishing.

# The LSTM: How It Works

- The gates are learning when to let error in, and when to let it out.
- In practice, the LSTM has shown a superior ability to learn long range dependencies as compared to simple RNNs.

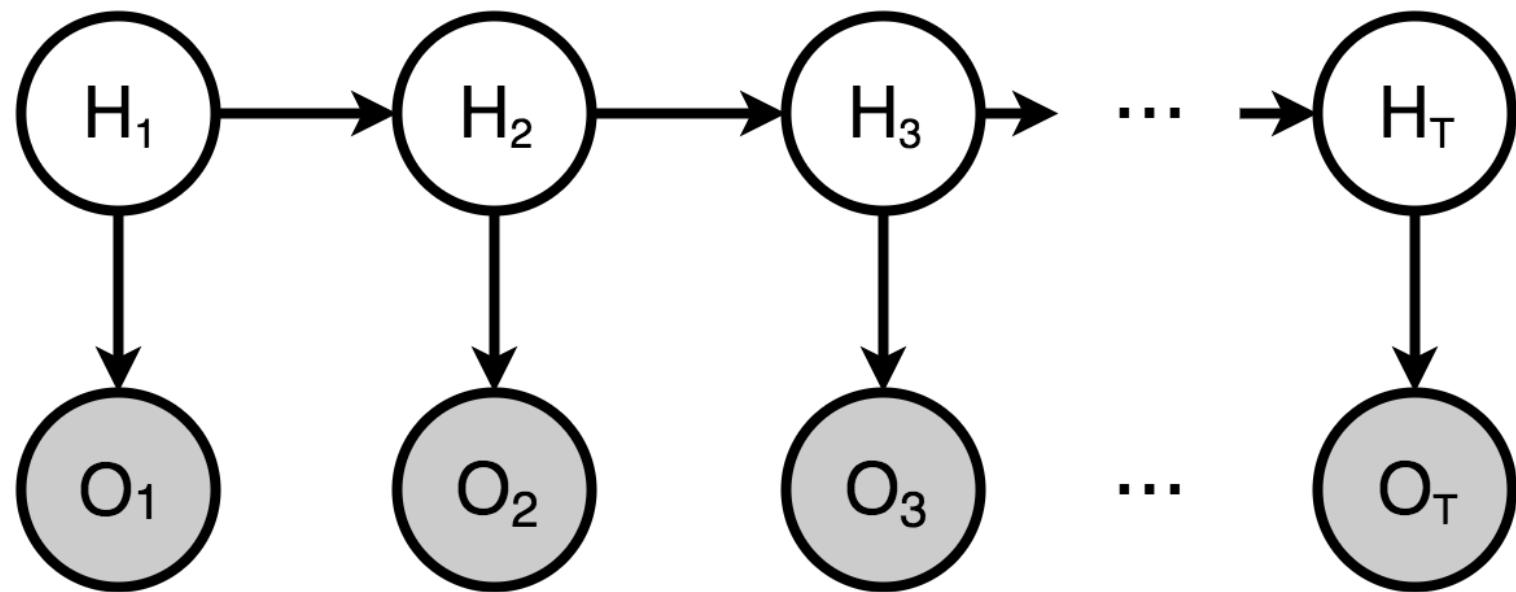
# **DSCI 552, Machine Learning for Data Science**

University of Southern California

M. R. Rajati, PhD

# Lesson 11

## Hidden Markov Models



# Hidden Markov Models

What is a hidden Markov model (HMM)?

A machine learning technique and...

...a discrete hill climb technique

Two for the price of one!

Where are HMMs used?

**Speech recognition**, information security,  
and too many other things to list

Q: Why are HMMs so useful?

A: Widely applicable and ***efficient algorithms***

# Markov Chain

Markov chain

“Memoryless random process”

Transitions depend only on current state (Markov chain of order 1)...

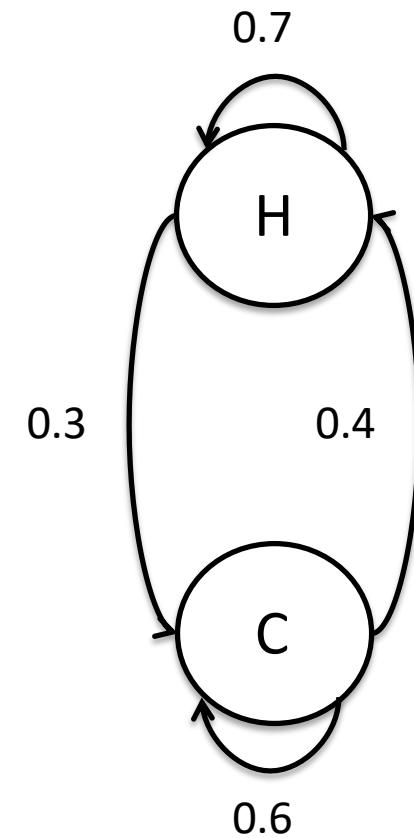
...and transition probability matrix

# Markov Chain

Suppose we're interested in average annual temperature

Only consider Hot and Cold  
From recorded history, obtain probabilities for...

Year-to-year transitions  
Based on thermometer readings for “recent” years



# Markov Chain

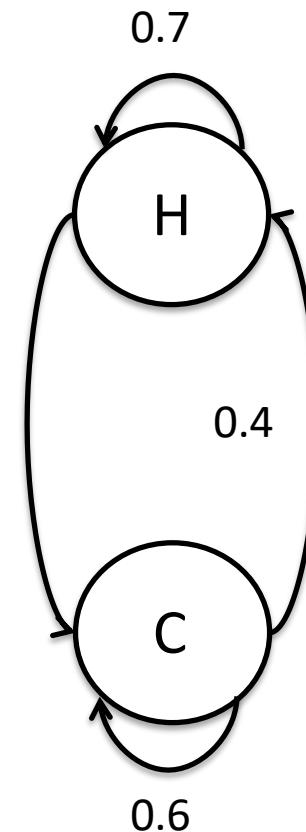
Transition probability matrix

Matrix is denoted as A

$$\begin{matrix} & H & C \\ H & \left[ \begin{array}{cc} 0.7 & 0.3 \\ 0.4 & 0.6 \end{array} \right] \\ C & \end{matrix}$$

Note, A is “row stochastic”

$$A = \left[ \begin{array}{cc} 0.7 & 0.3 \\ 0.4 & 0.6 \end{array} \right]$$



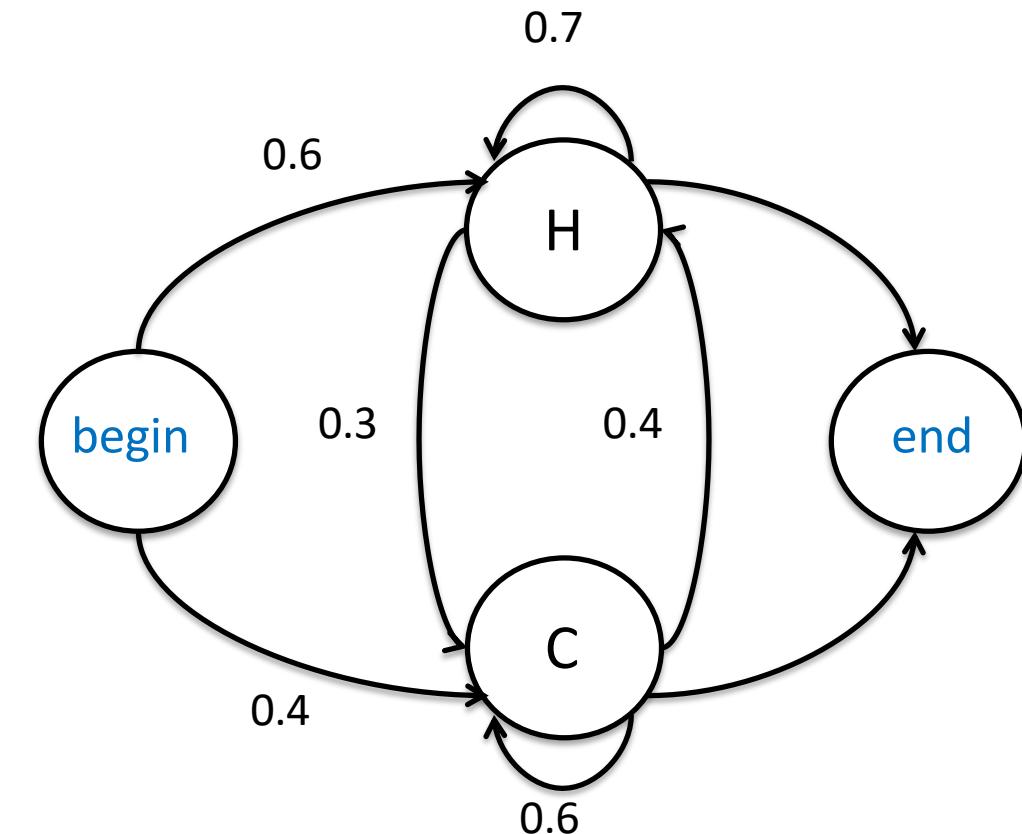
So, **each element of A is between 0 and 1** and each row satisfies the definition of a discrete probability distribution, thus the elements of any given row sum to 1.

# Markov Chain

Can also include  
**begin, end** states  
Matrix for begin  
state denoted  $\pi$   
In this example,

$$\pi = [ \begin{matrix} 0.6 & 0.4 \end{matrix} ]$$

Note that  $\pi$  is also  
row stochastic



# Hidden Markov Model

HMM includes a Markov chain/ process

- But the Markov process is “**hidden**”, i.e., we can’t directly observe the Markov process
- Instead, observe things that are **probabilistically** related to hidden states
- It’s as if there is a “**curtain**” between Markov chain and the observations

# HMM Example

Consider H/C annual temp example

Suppose we want to know H or C annual temperature in distant past

- Before thermometers were invented
- Note, we only distinguish between H and C

We assume transition between Hot and Cold years is same as today

Then the A matrix is **known**

# HMM Example

Temps in past follow a Markov process

But, we cannot observe temperature in past

We find evidence that **tree ring size** is related to temperature

Looking at historical data, we find this holds

We only consider 3 tree ring sizes

**Small, Medium, Large** (S, M, L, respectively)

Measure tree ring sizes and recorded temperatures to determine relationship

# HMM Example

We find that tree ring sizes and temperature related by

$$\begin{matrix} & S & M & L \\ H & \left[ \begin{matrix} 0.1 & 0.4 & 0.5 \end{matrix} \right] \\ C & \left[ \begin{matrix} 0.7 & 0.2 & 0.1 \end{matrix} \right] \end{matrix}$$

This is known as the B matrix

The matrix B is also row stochastic

$$B = \left[ \begin{matrix} 0.1 & 0.4 & 0.5 \\ \overbrace{0.7 & 0.2 & 0.1}^{\rightarrow} \end{matrix} \right]$$

# HMM Example

Can we now find H/C temps in past?

We cannot measure (observe) temps

But we can measure tree ring sizes...

...and tree ring sizes related to temps

By probabilities in the B matrix

Can we say something intelligent about temps over some interval in the past?

# HMM Notation

A lot of notation is required

Notation may be the most difficult part...

- $T$  = the length of the observation sequence
- $N$  = the number of states in the model
- $M$  = the number of observation symbols
- $Q$  =  $\{q_0, q_1, \dots, q_{N-1}\}$  = the states of the Markov process
- $V$  =  $\{0, 1, \dots, M - 1\}$  = set of possible observations
- $A$  = the state transition probabilities
- $B$  = the observation probability matrix
- $\pi$  = the initial state distribution
- $O$  =  $(O_0, O_1, \dots, O_{T-1})$  = observation sequence.

# HMM Notation

Note that for simplicity, observations taken from  $V = \{0, 1, \dots, M-1\}$

That is,  $O_i \in V$  for  $i = 0, 1, \dots, T-1$

The matrix  $A = \{a_{ij}\}$  is  $N \times N$ , where

$$a_{ij} = P(\text{state } q_j \text{ at } t+1 \mid \text{state } q_i \text{ at } t)$$

the element  
is in the  $i^{\text{th}}$   
row +  $j^{\text{th}}$   
column of  $A$ .

The matrix  $B = \{b_j(k)\}$  is  $N \times M$ , where

$$b_j(k) = P(\text{observation } k \text{ at } t \mid \text{state } q_j \text{ at } t).$$

# HMM Example

Consider our temperature example...

What are the possible observations?

$V = \{0, 1, 2\}$ , corresponding to S, M, L

What are states of Markov process?

$Q = \{H, C\}$

What are A, B,  $\pi$ , and T?

A, B,  $\pi$  on previous slides

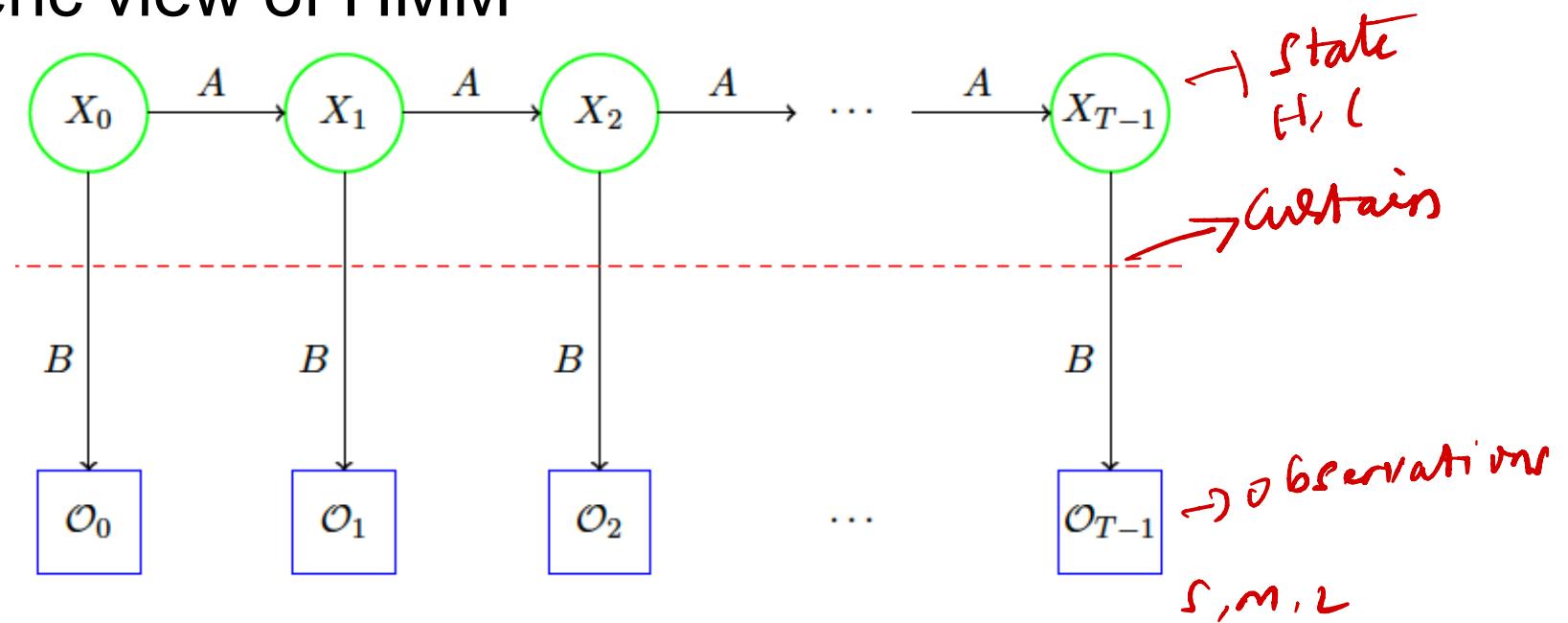
T is number of tree rings measured

What are N and M?

N = 2 and M = 3

# Generic HMM

## Generic view of HMM



HMM defined by  $A, B$ , and  $\pi$

We denote HMM “model” as  $\lambda = (A, B, \pi)$

# HMM Example

Suppose that we observe tree ring sizes

For a 4 year period of interest: S,M,S,L

Then  $\mathcal{O} = (\mathcal{O}_0, \mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3) = (0, 1, 0, 2)$

Most likely (hidden) state sequence?

That is, most likely  $X = (X_0, X_1, X_2, X_3)$

Let  $\pi_{x_0}$  be prob. of starting in state  $x_0$

Note  $b_{x_0}(\mathcal{O}_0)$  prob. of initial observation

And  $a_{x_0, x_1}$  is prob. of transition  $X_0$  to  $X_1$

And so on...

$$O = (S, M, L)$$

$$\pi = \begin{bmatrix} 0.6 & 0.4 \end{bmatrix}$$

# HMM Example

Bottom line?

$$\begin{array}{ccccc} & S & M & L & \\ H & \left[ \begin{array}{ccc} 0.1 & 0.4 & 0.5 \end{array} \right] & & H & \left[ \begin{array}{cc} 0.7 & 0.3 \end{array} \right] \\ C & \left[ \begin{array}{ccc} 0.7 & 0.2 & 0.1 \end{array} \right] & & C & \left[ \begin{array}{cc} 0.4 & 0.6 \end{array} \right] \end{array}$$

We can compute  $P(X, O)$  for any  $X$

For  $X = (X_0, X_1, X_2, X_3)$  we have  $P(X, O)$

$$= \pi_{x_0} b_{x_0}(\mathcal{O}_0) a_{x_0, x_1} b_{x_1}(\mathcal{O}_1) a_{x_1, x_2} b_{x_2}(\mathcal{O}_2) a_{x_2, x_3} b_{x_3}(\mathcal{O}_3)$$

*s m s l*

Suppose we observe (0,1,0,2), then what is probability of, say, HHCC? S,M,S,L

Plug into formula above to find  $P(HHCC, 0, 1, 0, 2)$

$$= \underline{0.6} (\underline{0.1}) (\underline{0.7}) (\underline{0.4}) (\underline{0.3}) (\underline{0.7}) (\underline{0.6}) (\underline{0.1}) = 0.000212.$$

# HMM Example

Do same for all 4-state sequences  
We find that the winner is...

CCCH

Not so fast!

$$\sum_{x} p(x, \theta) = p(\theta)$$

$x$

$$\frac{p(x, \theta)}{p(\theta)} = p(x | \theta)$$

16 States

$p(x, \theta)$

| state | probability | normalized probability |
|-------|-------------|------------------------|
| HHHH  | .000412     | .042787                |
| HHHC  | .000035     | .003635                |
| HHCH  | .000706     | .073320                |
| HHCC  | .000212     | .022017                |
| HCHH  | .000050     | .005193                |
| HCHC  | .000004     | .000415                |
| HCCH  | .000302     | .031364                |
| HCCC  | .000091     | .009451                |
| CHHH  | .001098     | .114031                |
| CHHC  | .000094     | .009762                |
| CHCH  | .001882     | .195451                |
| CHCC  | .000564     | .058573                |
| CCHH  | .000470     | .048811                |
| CCHC  | .000040     | .004154                |
| CCCH  | .002822     | .293073                |
| CCCC  | .000847     | .087963                |

Sum =  $P(O)$   $P(X|O)=P(X,O)/P(O)$

Maximum Likelihood  
Dynamic programming

# HMM Example

The **path** CCCH scores the highest

In dynamic programming (DP), we find  
highest scoring path

But, in HMM we maximize **expected  
number of correct states** Expectation Maximization

Sometimes called “EM algorithm”

For “Expectation Maximization”

How does HMM work in this example?

$$\sum_x p(x|o) = 1$$

$$\sum_x p(x|o) = \sum_x \frac{p(x,o)}{p(o)}$$

$$= \frac{\sum_x p(x,o)}{\sum_x p(x,o)}$$

$$= 1$$

# HMM Example

For first position...

Sum probabilities for all paths that have H in 1<sup>st</sup> position, compare to sum of probabilities for paths with C in 1<sup>st</sup> position:  
biggest wins

Repeat for each position and we find

|          | element  |          |          |          |
|----------|----------|----------|----------|----------|
|          | 0        | 1        | 2        | 3        |
| $P(H O)$ | 0.188182 | 0.519576 | 0.228788 | 0.804029 |
| $P(C O)$ | 0.811818 | 0.480424 | 0.771212 | 0.195971 |

sums up to 1

C H C H

# HMM Example

Note: We could use both  $P(X, O)$  and  $P(X|O)$  to decide the best. Especially in paper and pencil problems, it is easier to use the un-normalized  $P(X, O)$

|          | element  |          |          |          |
|----------|----------|----------|----------|----------|
|          | 0        | 1        | 2        | 3        |
| $P(H O)$ | 0.188182 | 0.519576 | 0.228788 | 0.804029 |
| $P(C O)$ | 0.811818 | 0.480424 | 0.771212 | 0.195971 |

# HMM Example

So, HMM solution gives us CHCH

While DP solution is CCCH

Which solution is better?

Neither solution is better!

Just using different definitions of “best”

|          | element  |          |          |          |
|----------|----------|----------|----------|----------|
|          | 0        | 1        | 2        | 3        |
| $P(H O)$ | 0.188182 | 0.519576 | 0.228788 | 0.804029 |
| $P(C O)$ | 0.811818 | 0.480424 | 0.771212 | 0.195971 |

# HMM Example

|          | element  |          |          |          |
|----------|----------|----------|----------|----------|
|          | 0        | 1        | 2        | 3        |
| $P(H O)$ | 0.188182 | 0.519576 | 0.228788 | 0.804029 |
| $P(C O)$ | 0.811818 | 0.480424 | 0.771212 | 0.195971 |

So, HMM solution gives us CHCH

While DP solution is CCCH

Which solution is better?

Neither solution is better!

Just using different definitions of “best”

Very cold very hot cold wet



## HMM Paradox?

HMM maximizes expected number of correct states

Whereas DP chooses “best” overall path  
Possible for HMM to choose a “path” that is impossible

Could be a transition probability of 0  
Cannot get impossible path with DP  
Is this a flaw with HMM?

No, it's a feature

# Probability of Observations

Table computed for

$$\mathcal{O} = (\mathcal{O}_0, \mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3) \\ = (0, 1, 0, 2)$$

For this sequence,

$$P(\mathcal{O}) = .000412 + .000035 \\ + .000706 + \dots +$$

.000847

= left to the reader

Similarly for other observations  $\mathcal{O}$

$$p(\mathcal{O}) = \sum_{\mathbf{x}} p(\mathbf{x}, \mathcal{O})$$

$$\sum_x \sum_{\mathcal{O}} p(x, \mathcal{O}) = 1$$

$(\text{possible states})^M$

| state  | probability | normalized probability |
|--------|-------------|------------------------|
| $HHHH$ | .000412     | .042787                |
| $HHHC$ | .000035     | .003635                |
| $HHCH$ | .000706     | .073320                |
| $HHCC$ | .000212     | .022017                |
| $HCHH$ | .000050     | .005193                |
| $HCHC$ | .000004     | .000415                |
| $HCCH$ | .000302     | .031364                |
| $HCCC$ | .000091     | .009451                |
| $CHHH$ | .001098     | .114031                |
| $CHHC$ | .000094     | .009762                |
| $CHCH$ | .001882     | .195451                |
| $CHCC$ | .000564     | .058573                |
| $CCHH$ | .000470     | .048811                |
| $CCHC$ | .000040     | .004154                |
| $CCCH$ | .002822     | .293073                |
| $CCCC$ | .000847     | .087963                |

$$\underline{3} \quad \underline{3} \quad \underline{3} \quad \underline{3} = 3^4 = 81$$

possible tables

# Probability of Observations

If this calculation is made for all possible 4-observation sequences then the sum of the resulting probabilities (not the normalized probabilities) will be 1.

| state       | probability | normalized probability |
|-------------|-------------|------------------------|
| <i>HHHH</i> | .000412     | .042787                |
| <i>HHHC</i> | .000035     | .003635                |
| <i>HHCH</i> | .000706     | .073320                |
| <i>HHCC</i> | .000212     | .022017                |
| <i>HCHH</i> | .000050     | .005193                |
| <i>HCHC</i> | .000004     | .000415                |
| <i>HCCH</i> | .000302     | .031364                |
| <i>HCCC</i> | .000091     | .009451                |
| <i>CHHH</i> | .001098     | .114031                |
| <i>CHHC</i> | .000094     | .009762                |
| <i>CHCH</i> | .001882     | .195451                |
| <i>CHCC</i> | .000564     | .058573                |
| <i>CCHH</i> | .000470     | .048811                |
| <i>CCHC</i> | .000040     | .004154                |
| <i>CCCH</i> | .002822     | .293073                |
| <i>CCCC</i> | .000847     | .087963                |

# HMM Model

$$A_{N \times N} \quad B_{N \times M}$$

An HMM is defined by the three matrices, A, B, and  $\pi$

Note that M and N are implied, since they are the dimensions of matrices  
So, we denote an HMM “model” as

$$\lambda = (A, B, \pi)$$

# The Three Problems

## HMMs used to solve 3 problems:

**Problem 1:** Given a model  $\lambda = (A, B, \pi)$  and observation sequence  $O$ , find  $P(O|\lambda)$

That is, we can **score** an observation sequence to see how well it fits a given model

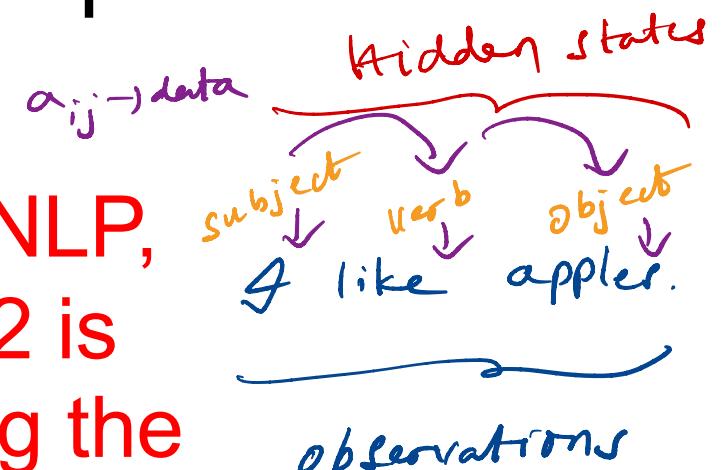
# The Three Problems

HMMs used to solve 3 problems

**Problem 2:** Given  $\lambda = (A, B, \pi)$  and  $O$ ,  
find an optimal state sequence (in  
HMM sense)

Uncover hidden part (like previous  
example)

In many applications in NLP,  
the solution to Problem 2 is  
crucial. Example: Finding the  
grammatical roles of words in  
a sentence.





# The Three Problems

HMMs used to solve 3 problems

**Problem 3:** Given  $O$ ,  $N$ , and  $M$ , find the model  $\lambda$  that maximizes probability of  $O$

That is, ***train*** a model to fit observations

# HMMs in Practice

Often, HMMs used as follows:

Given an observation sequence...

Assume that (hidden) Markov process exists

Train a model based on observations

That is, solve Problem 3

“**Best**”  $N$  can be found by trial and error

Then given a sequence of observations, score it  
versus the model we trained

This is Problem 1: high score implies similar to  
training data, low score says it’s not

# HMMs in Practice

In this sense, HMM is a “machine learning” technique

To train a model, we do not need to specify anything except the parameter N  
“Best” N often found by trial and error

So, we don’t need to “think” too much

Just train HMM and then use it

Fortunately, there are efficient algorithms for HMMs

# The Three Solutions

We give detailed solutions to 3 problems

Note: We must find **efficient** solutions

The three problems:

**Problem 1:** Score an observation sequence versus a given model

**Problem 2:** Given a model, “uncover” hidden part

**Problem 3:** Given an observation sequence, train a model *Welch - Baum*

Recall that we considered example for 2 and 1, but direct solutions are **very inefficient**

# The Three Solutions

## Appendix: Algorithmic Details

# Solution 1

Score observations versus a given model

Given model  $\lambda = (A, B, \pi)$  and observation sequence  $O = (O_0, O_1, \dots, O_{T-1})$ , find  $P(O|\lambda)$

Denote hidden states as

$$X = (X_0, X_1, \dots, X_{T-1})$$

Then from definition of  $B$ ,

$$P(O|X, \lambda) = b_{x_0}(O_0) b_{x_1}(O_1) \dots b_{x_{T-1}}(O_{T-1})$$

And from definition of  $A$  and  $\pi$ ,

$$P(X|\lambda) = \pi_{x_0} a_{x_0, x_1} a_{x_1, x_2} \dots a_{x_{T-2}, x_{T-1}}$$

# Solution 1

Elementary conditional probability fact:

$$P(O, X | \lambda) = P(O | X, \lambda) P(X | \lambda)$$

Sum over all possible state sequences  $X$ ,

$$\begin{aligned} P(O | \lambda) &= \sum P(O, X | \lambda) = \sum P(O | X, \lambda) P(X | \lambda) \\ &= \sum \pi_{x_0} b_{x_0}(O_0) a_{x_0, x_1} b_{x_1}(O_1) \dots a_{x_{T-2}, x_{T-1}} b_{x_{T-1}}(O_{T-1}) \end{aligned}$$

This “works” but way too costly

Requires about  $2TN^T$  multiplications

Why?

There should be a better way...

# Forward Algorithm

Instead, use *forward algorithm*

Or “alpha pass”

For  $t = 0, 1, \dots, T-1$  and  $i=0, 1, \dots, N-1$ , let

$$\alpha_t(i) = P(O_0, O_1, \dots, O_t, X_t=q_i | \lambda)$$

Probability of “partial observation” to  $t$ , and  
Markov process is in state  $q_i$  at step  $t$

Can be computed recursively, efficiently

# Forward Algorithm

Let  $\alpha_0(i) = \pi_i b_i(O_0)$  for  $i = 0, 1, \dots, N-1$

For  $t = 1, 2, \dots, T-1$  and  $i=0, 1, \dots, N-1$ , let

$$\alpha_t(i) = \left( \sum \alpha_{t-1}(j) a_{ji} \right) b_i(O_t)$$

Where the sum is from  $j = 0$  to  $N-1$

From definition of  $\alpha_t(i)$  we see

$$P(O|\lambda) = \sum \alpha_{T-1}(i)$$

Where the sum is from  $i = 0$  to  $N-1$

This requires only  $N^2T$  multiplications

# Forward Algorithm

**Note:** The score  $P(O|\lambda)$  is dependent on the length of the observation sequence. Consequently, to compare scores for sequences of different length, we can normalize to a per observation score, that is,  $\text{score} = \sum \alpha_{T-1}(i) / T$ .

# Forward Algorithm

In a nutshell,  $\alpha_T(i)$  is the probability that the sequence of observations up to time  $T$  correspond to the state being the  $i^{\text{th}}$  state at time  $T$ , given the model  $\lambda$ .

We compute  $\alpha_t(i)$  recursively using  $\alpha_{t-1}(j)$ 's.

# Solution 2

Given a model, find hidden states

Given  $\lambda = (A, B, \pi)$  and  $O$ , find an optimal state sequence

Recall that optimal means “maximize expected number of correct states”

In contrast, DP finds best scoring path

For temp/tree ring example, solved this

But hopelessly inefficient approach

A better way: ***backward algorithm***

Or “beta pass”

# Backward Algorithm

For  $t = 0, 1, \dots, T-1$  and  $i = 0, 1, \dots, N-1$ , let  $\beta_t(i)$   
 $= P(O_{t+1}, O_{t+2}, \dots, O_{T-1} | X_t = q_i, \lambda)$

Probability of partial observation from  $t$  to end and Markov process in state  $q_i$  at step  $t$

Analogous to the forward algorithm

As with forward algorithm, this can be computed recursively and efficiently

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \dots, O_{T-1} | X_t = q_i, \lambda)$$

# Backward Algorithm

Let  $\beta_{T-1}(i) = 1$  for  $i = 0, 1, \dots, N-1$

For  $t = T-2, T-3, \dots, 1$  and  $i = 0, 1, \dots, N-1$ ,  
let

$$\beta_t(i) = \sum a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)$$

Where the sum is from  $j = 0$  to  $N-1$

## Solution 2

For  $t = 1, 2, \dots, T-1$  and  $i=0, 1, \dots, N-1$  define

$$\gamma_t(i) = P(X_t=q_i|O,\lambda)$$

(The probability of being in state  $i$  at time  $t$ )

Most likely state at  $t$  is  $q_i$  that maximizes  $\gamma_t(i)$

$$\tilde{X}_t = \max_i \gamma_t(i).$$

Note that  $\gamma_t(i) = \alpha_t(i)\beta_t(i)/P(O|\lambda)$

And recall  $P(O|\lambda) = \sum \alpha_{T-1}(i)$

# Solution 2

The bottom line?

Forward algorithm solves Problem 1  
Forward/backward algorithms solve  
Problem 2 by computing  $\gamma_t(i)$  for each state  
at each time step (using both forward and  
backward paths) and choosing the state that  
maximizes it.

$$\tilde{X}_t = \max_i \gamma_t(i).$$

## Solution 2

Why is it necessary to normalize gamma by dividing by  $P(O | \lambda)$ ? Because these probabilities are computed assuming the observation sequence is known (i.e., given  $O$ ), as opposed to being computed relative to the larger probability space.

# Solution 3

Train a model: Given  $O$ ,  $N$ , and  $M$ , find  $\lambda$  that maximizes probability of  $O$

We'll iteratively adjust  $\lambda = (A, B, \pi)$  to better fit the given observations  $O$

The size of matrices are fixed ( $N$  and  $M$ )

But elements of matrices can change

It is nice that this works...

...and amazing that it's efficient!

# Solution 3

For  $t=0,1,\dots,T-2$  and  $i,j$  in  $\{0,1,\dots,N-1\}$ , define  
“di-gammas” as

$$\gamma_t(i,j) = P(X_t=q_i, X_{t+1}=q_j | O, \lambda)$$

Note  $\gamma_t(i,j)$  is prob of being in state  $q_i$  at time  $t$  and transitioning to state  $q_j$  at  $t+1$

$$\text{Then } \gamma_t(i,j) = \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) / P(O|\lambda)$$

$$\text{And } \gamma_t(i) = \sum \gamma_t(i,j)$$

Where sum is from  $j = 0$  to  $N - 1$

# Baum-Welch Model Re-estimation

Given di-gammas and gammas...

For  $i = 0, 1, \dots, N-1$  let  $\pi_i = \gamma_0(i)$

For  $i = 0, 1, \dots, N-1$  and  $j = 0, 1, \dots, N-1$

$$a_{ij} = \sum \gamma_t(i,j) / \sum \gamma_t(i)$$

Where both sums are from  $t = 0$  to  $T-2$

For  $j = 0, 1, \dots, N-1$  and  $k = 0, 1, \dots, M-1$

$$b_j(k) = \sum \gamma_t(j) / \sum \gamma_t(j)$$

Both sums from from  $t = 0$  to  $T-1$  but only  $t$  for which  $O_t = k$  are counted in numerator

# Baum-Welch Model Re-estimation

## Why does this work?

For the given observation sequence  $O$ , the sum  $\sum \gamma_t(i)$  gives us the current best estimate for the total probability of being in state  $i$ , while  $\sum \gamma_t(i,j)$  gives us the total probability of transitioning from state  $i$  to state  $j$ . Hence, the ratio  $\sum \gamma_t(i,j)/\sum \gamma_t(i)$  enables us to re-estimate  $a_{ij}$  based on the current model parameters and observation sequence.

# Solution 3

To summarize...

1. Initialize  $\lambda = (A, B, \pi)$
2. Compute  $\alpha_t(i)$ ,  $\beta_t(i)$ ,  $\gamma_t(i,j)$ ,  $\gamma_t(i)$
3. Re-estimate the model  $\lambda = (A, B, \pi)$
4. If  $P(O|\lambda)$  increases by more than  $\varepsilon$ ,  
goto 2 (where  $\varepsilon$  is small)

# Solution 3

Some fine points...

Model initialization

If we have a good guess for  $\lambda = (A, B, \pi)$  then we can use it for initialization

If not, let  $\pi_i \approx 1/N$ ,  $a_{i,j} \approx 1/N$ ,  $b_j(k) \approx 1/M$

Subject to row stochastic conditions

But, do **not** initialize to exactly uniform values

Stopping conditions

Stop after some number of iterations and/or...

Stop if increase in  $P(O|\lambda)$  is too small

# HMM as Discrete Hill Climb

Algorithm on previous slides shows that HMM is a “discrete hill climb”

HMM consists of discrete parameters

Specifically, the elements of the matrices And re-estimation process improves model by modifying parameters

So, process “climbs” toward improved model

This happens in a high-dimensional space

# Dynamic Programming: The Viterbi Algorithm

Brief detour...

For  $\lambda = (A, B, \pi)$  as above, it's easy to define a dynamic program (DP)

Executive summary:

DP is forward algorithm, with “sum” replaced by “max”

# Dynamic Programming: The Viterbi Algorithm

Let  $\delta_0(i) = \pi_i b_i(O_0)$  for  $i=0,1,\dots,N-1$

For  $t=1,2,\dots,T-1$  and  $i=0,1,\dots,N-1$  compute

$$\delta_t(i) = \max \left( \delta_{t-1}(j) a_{ji} \right) b_i(O_t)$$

Where the max is over  $j$  in  $\{0,1,\dots,N-1\}$

Note that **at each  $t$ , the DP computes best path for each state, up to that point**

So, probability of best path is  $\max \delta_{T-1}(j)$

This max gives the highest probability

Not the best path, for that, see next slide

# Dynamic Programming: The Viterbi Algorithm

To determine optimal path

While computing deltas, keep track of  
pointers to previous state

When finished, construct optimal path by  
tracing back points

# Dynamic Programming: The Viterbi Algorithm

For example, consider temp example: recall that we observe (0,1,0,2)

$$\pi = \begin{bmatrix} 0.6 & 0.4 \end{bmatrix}$$

$$\begin{array}{ccc} S & M & L \\ H & \left[ \begin{array}{ccc} 0.1 & 0.4 & 0.5 \end{array} \right] \\ C & \left[ \begin{array}{ccc} 0.7 & 0.2 & 0.1 \end{array} \right] \end{array}$$

$$\begin{array}{ccc} H & C \\ H & \left[ \begin{array}{cc} 0.7 & 0.3 \end{array} \right] \\ C & \left[ \begin{array}{cc} 0.4 & 0.6 \end{array} \right] \end{array}$$

Probabilities for path of length 1:

$$P(H) = \pi_0 b_0(0) = 0.6(0.1) = 0.06 \text{ and } P(C) = \pi_1 b_1(0) = 0.4(0.7) = 0.28.$$

These are the only “paths” of length 1

# Dynamic Programming: The Viterbi Algorithm

Probabilities for each path of length 2

$$P(HH) = 0.06(0.7)(0.4) = 0.0168$$

$$P(HC) = 0.06(0.3)(0.2) = 0.0036$$

$$P(CH) = 0.28(0.4)(0.4) = 0.0448$$

$$P(CC) = 0.28(0.6)(0.2) = 0.0336$$

Best path of length 2 ending with H is CH

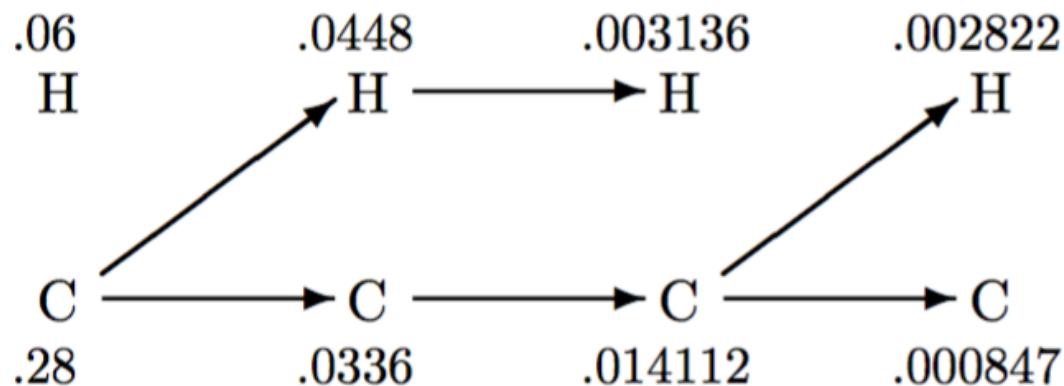
Best path of length 2 ending with C is CC

$$\delta_t(i) = \max \left( \delta_{t-1}(j) a_{ji} \right) b_i(O_t)$$

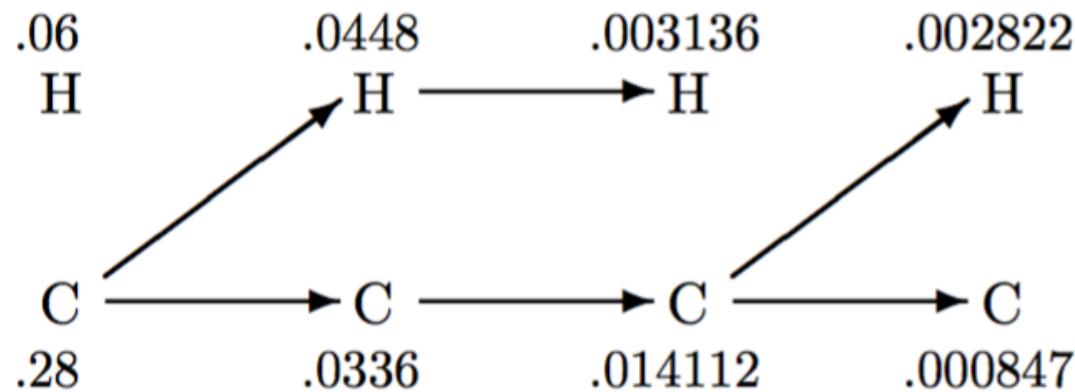
# Dynamic Program

Continuing, we compute best path ending at H and C at each step

And save pointers: why?



# Dynamic Program



Best final score is .002822

And thanks to pointers, best path is  
CCCH

But what about underflow?

A serious problem in bigger cases

# Dynamic Program

<https://www.youtube.com/watch?v=6JVqutwtzmo&t=4s>

# Underflow Resistant DP

Common trick to prevent underflow:

Instead of multiplying probabilities...

...add logarithms of probabilities

Why does this work?

Because  $\log(xy) = \log x + \log y$

Adding logs does not tend to 0

Note that these logs are negative...

...and we must avoid 0 probabilities

# Underflow Resistant DP

Underflow resistant DP algorithm:

Let  $\delta_0(i) = \log(\pi_i b_i(O_0))$

for  $i=0,1,\dots,N-1$

For  $t=1,2,\dots,T-1$  and  $i=0,1,\dots,N-1$  compute

$$\delta_t(i) = \max \left( \delta_{t-1}(j) + \log(a_{ji}) + \log(b_i(O_t)) \right)$$

Where the max is over  $j$  in  $\{0,1,\dots,N-1\}$

And **score** of best path is  $\max \delta_{T-1}(j)$

As before, must also keep track of paths

# HMM Scaling

Trickier to prevent underflow in HMM

We consider solution 3

Since it includes solutions 1 and 2

Recall for  $t = 1, 2, \dots, T-1$ ,  $i=0, 1, \dots, N-1$ ,

$$\alpha_t(i) = \left( \sum \alpha_{t-1}(j) a_{j,i} \right) b_i(O_t)$$

The idea is to normalize alphas so that they sum to 1

Algorithm on next slide

# HMM Scaling

Given  $\alpha_t(i) = \left( \sum \alpha_{t-1}(j) a_{j,i} \right) b_i(O_t)$

Let  $a_0(i) = \alpha_0(i)$  for  $i=0, 1, \dots, N-1$

Let  $c_0 = 1 / \sum a_0(j)$

For  $i = 0, 1, \dots, N-1$ , let  $\alpha_0(i) = c_0 a_0(i)$

This takes care of  $t = 0$  case

Algorithm continued on next slide...

# HMM Scaling

For  $t = 1, 2, \dots, T-1$  do the following:

For  $i = 0, 1, \dots, N-1$ ,

$$a_t(i) = \left( \sum \alpha_{t-1}(j)a_{j,i} \right) b_i(O_t)$$

Let  $c_t = 1/\sum a_t(j)$

For  $i = 0, 1, \dots, N-1$  let  $\alpha_t(i) = c_t a_t(i)$

# HMM Scaling

Easy to show  $\alpha_t(i) = c_0 c_1 \dots c_t \alpha_t(i)$  (‡)

Simple proof by induction

So,  $c_0 c_1 \dots c_t$  is scaling factor at step t

Also, easy to show that

$$\alpha_t(i) = \alpha_t(i) / \sum \alpha_t(j)$$

Which implies  $\sum \alpha_{T-1}(i) = 1$  (##)

# HMM Scaling

By combining (#) and (##), we have

$$\begin{aligned} 1 &= \sum \alpha_{T-1}(i) = c_0 c_1 \dots c_{T-1} \sum \alpha_{T-1}(i) \\ &= c_0 c_1 \dots c_{T-1} P(O|\lambda) \end{aligned}$$

Therefore,  $P(O|\lambda) = 1 / c_0 c_1 \dots c_{T-1}$

To avoid underflow, we compute

$$\log P(O|\lambda) = -\sum \log(c_j)$$

Where sum is from  $j = 0$  to  $T-1$

# HMM Scaling

Similarly, scale betas as  $c_t \beta_t(i)$

For re-estimation,

Compute  $\gamma_t(i,j)$  and  $\gamma_t(i)$  using original formulas, but with scaled alphas, betas

This gives us new values for  $\lambda = (A, B, \pi)$

“Easy exercise” to show re-estimate is exact when scaled alphas and betas used

Also,  $P(O|\lambda)$  cancels from formula

Use  $\log P(O|\lambda) = -\sum \log(c_j)$  to decide if iterate improves

# All Together Now

Complete pseudo code for Solution 3

Given:  $(O_0, O_1, \dots, O_{T-1})$  and  $N$  and  $M$

Initialize:  $\lambda = (A, B, \pi)$

$A$  is  $N \times N$ ,  $B$  is  $N \times M$  and  $\pi$  is  $1 \times N$

$\pi_i \approx 1/N$ ,  $a_{ij} \approx 1/N$ ,  $b_j(k) \approx 1/M$ , each matrix row stochastic, but not uniform

Initialize:

`maxIters` = max number of re-estimation steps

`iters` = 0

`oldLogProb` =  $-\infty$

# Forward Algorithm

## Forward algorithm With scaling

```
// compute  $\alpha_0(i)$ 
 $c_0 = 0$ 
for  $i = 0$  to  $N - 1$ 
   $\alpha_0(i) = \pi(i)b_i(\mathcal{O}_0)$ 
   $c_0 = c_0 + \alpha_0(i)$ 
next  $i$ 

// scale the  $\alpha_0(i)$ 
 $c_0 = 1/c_0$ 
for  $i = 0$  to  $N - 1$ 
   $\alpha_0(i) = c_0\alpha_0(i)$ 
next  $i$ 

// compute  $\alpha_t(i)$ 
for  $t = 1$  to  $T - 1$ 
   $c_t = 0$ 
  for  $i = 0$  to  $N - 1$ 
     $\alpha_t(i) = 0$ 
    for  $j = 0$  to  $N - 1$ 
       $\alpha_t(i) = \alpha_t(i) + \alpha_{t-1}(j)a_{ji}$ 
    next  $j$ 
     $\alpha_t(i) = \alpha_t(i)b_i(\mathcal{O}_t)$ 
     $c_t = c_t + \alpha_t(i)$ 
  next  $i$ 

  // scale  $\alpha_t(i)$ 
   $c_t = 1/c_t$ 
  for  $i = 0$  to  $N - 1$ 
     $\alpha_t(i) = c_t\alpha_t(i)$ 
  next  $i$ 
next  $t$ 
```

# Backward Algorithm

Backward algorithm  
or “beta pass”

With scaling

Note: same scaling  
factor as alphas

```
// Let  $\beta_{T-1}(i) = 1$  scaled by  $c_{T-1}$ 
for  $i = 0$  to  $N - 1$ 
     $\beta_{T-1}(i) = c_{T-1}$ 
next  $i$ 

//  $\beta$ -pass
for  $t = T - 2$  to  $0$  by  $-1$ 
    for  $i = 0$  to  $N - 1$ 
         $\beta_t(i) = 0$ 
        for  $j = 0$  to  $N - 1$ 
             $\beta_t(i) = \beta_t(i) + a_{ij} b_j(\mathcal{O}_{t+1}) \beta_{t+1}(j)$ 
        next  $j$ 
        // scale  $\beta_t(i)$  with same scale factor as  $\alpha_t(i)$ 
         $\beta_t(i) = c_t \beta_t(i)$ 
    next  $i$ 
next  $t$ 
```

# Gammas

## Using scaled alphas and betas

```
for  $t = 0$  to  $T - 2$ 
    denom = 0
    for  $i = 0$  to  $N - 1$ 
        for  $j = 0$  to  $N - 1$ 
            denom = denom +  $\alpha_t(i)a_{ij}b_j(\mathcal{O}_{t+1})\beta_{t+1}(j)$ 
        next  $j$ 
    next  $i$ 
    for  $i = 0$  to  $N - 1$ 
         $\gamma_t(i) = 0$ 
        for  $j = 0$  to  $N - 1$ 
             $\gamma_t(i, j) = (\alpha_t(i)a_{ij}b_j(\mathcal{O}_{t+1})\beta_{t+1}(j)) / \text{denom}$ 
             $\gamma_t(i) = \gamma_t(i) + \gamma_t(i, j)$ 
        next  $j$ 
    next  $i$ 
next  $t$ 
```

```
// Special case for  $\gamma_{T-1}(i)$ 
denom = 0
for  $i = 0$  to  $N - 1$ 
    denom = denom +  $\alpha_{T-1}(i)$ 
next  $i$ 
for  $i = 0$  to  $N - 1$ 
     $\gamma_{T-1}(i) = \alpha_{T-1}(i) / \text{denom}$ 
next  $i$ 
```

# Re-Estimation

Again, using  
scaled gammas  
So formulas  
unchanged

```
// re-estimate  $\pi$ 
for  $i = 0$  to  $N - 1$ 
     $\pi_i = \gamma_0(i)$ 
next  $i$ 

// re-estimate  $A$ 
for  $i = 0$  to  $N - 1$ 
    for  $j = 0$  to  $N - 1$ 
        numer = 0
        denom = 0
        for  $t = 0$  to  $T - 2$ 
            numer = numer +  $\gamma_t(i, j)$ 
            denom = denom +  $\gamma_t(i)$ 
        next  $t$ 
         $a_{ij} = \text{numer}/\text{denom}$ 
    next  $j$ 
next  $i$ 

// re-estimate  $B$ 
for  $i = 0$  to  $N - 1$ 
    for  $j = 0$  to  $M - 1$ 
        numer = 0
        denom = 0
        for  $t = 0$  to  $T - 2$ 
            if( $\mathcal{O}_t == j$ ) then
                numer = numer +  $\gamma_t(i)$ 
            end if
            denom = denom +  $\gamma_t(i)$ 
        next  $t$ 
         $b_i(j) = \text{numer}/\text{denom}$ 
    next  $j$ 
next  $i$ 
```

# Stopping Criteria

Check that  
probability increases

In practice, want  
 $\text{logProb} >$   
 $\text{oldLogProb} + \varepsilon$   
And don't exceed  
max iterations

```
// Compute  $\log[P(\mathcal{O} | \lambda)]$ 
logProb = 0
for  $i = 0$  to  $T - 1$ 
    logProb = logProb + log( $c_i$ )
next  $i$ 
logProb = -logProb

// To iterate or not to iterate, that is the question...
iters = iters + 1
if (iters < maxIters and logProb > oldLogProb) then
    oldLogProb = logProb
    goto  $\alpha$ -pass
else
    output  $\lambda = (\pi, A, B)$ 
end if
```

# References

M. Stamp, [A revealing introduction to hidden Markov models](#)

L.R. Rabiner, [A tutorial on hidden Markov models and selected applications in speech recognition](#)

R.L. Cave & L.P. Neuwirth, [Hidden Markov models for English](#)

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 30    |        |
| 3       | 25    |        |
| 4       | 25    |        |
| 5       | 15    |        |
| Total   | 120   |        |

1. A statistician is working on the amount of funding that companies obtain on a crowdsourcing website and has developed the following model. She used 26 companies to obtain the model

$$\hat{y} = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4 + b_5x_5$$

$$\hat{y} = 964.8 + 700.2x_1 + 317.5x_2 - 200.2x_3 + 15.3x_4 + 17.1x_5$$

The standard errors are:

$$\begin{aligned}s_{b_1} &= 12. \\ s_{b_2} &= 22.5 \\ s_{b_3} &= 101.8 \\ s_{b_4} &= 45.3 \\ s_{b_5} &= 2.3\end{aligned}$$

- $\hat{y}$ : the amount of funding obtained by a company in 1000 dollars
  - $x_1$ : the average annual salary of the founders
  - $x_2$ : the number of employees the startup hired
  - $x_3$ : a dummy variable that is 1 when the company's field is information technology and 0 otherwise
  - $x_4$ : the age of the company
  - $x_5$  is a dummy variable taking value 1 if the founders had previous failures and 0 otherwise
- (a) Interpret the estimated coefficients  $b_0 = 964.8$  and  $b_5 = 17.1$ .
- (b) Test, at the 5% level, the null hypothesis that the true coefficient on the dummy variable  $x_3$  is 0 against the alternative that it is not 0.
- (c) Find and interpret a 80% confidence interval for the parameter  $\beta_1$ .
- (d) If for the model,  $\text{SSR}=18147.5$  (Regression Sum of Squares) and  $\text{SSE} = 17136.5$  (Residual Sum of Squares), test the hypothesis that all the coefficients of the model are 0 (test overall significance of the model) using  $\alpha = 1\%$ .
- (e) Calculate  $R^2$  for the regression line.

2. Although in practice, we usually model conditional distributions of features in each class as Gaussians for Bayesian Discriminant Analysis, one must notice that Bayesian Discriminant Analysis is doable using any type of distribution. Assume that we wish to perform Discriminant Analysis with only one positive predictor  $x \geq 0$  (i.e.  $p = 1$ ), but instead of normal class pdfs, we have Lognormal class pdfs whose formula is: (40 pts)

$$f_k(x) = \frac{1}{x\sqrt{2\pi}\sigma_k} \exp\left(\frac{-(\ln x - \mu_k)^2}{2\sigma_k^2}\right), x \geq 0$$

We know that the posterior class probability for class  $k$  is:

$$\Pr(Y = k | X = x) = p_k(x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^K \pi_l f_l(x)}$$

where  $\pi_l$ 's are prior class probabilities.

- (a) Determine the simplest form for the discriminant score  $\delta_k(x)$  if we assume that  $\sigma_1 = \sigma_2 = \dots = \sigma_K = \sigma$ ; that is, there is a shared parameter  $\sigma_k = \sigma$  across all  $K$  classes. Are the discriminant scores linear functions of  $x$  in this case? Hint: follow the same procedure as the one we used for Gaussian pdfs.
- (b) For binary classification ( $K = 2$ ), when classes are balanced ( $\pi_1 = \pi_2$ ) and  $\sigma_1 = \sigma_2 = \sigma$ , show that the decision boundary between the two classes is the threshold

$$x^* = \sqrt{\exp(\mu_1)} \sqrt{\exp(\mu_2)}$$

$$f_k(n) = \frac{1}{\alpha \sqrt{2\pi} \sigma_k} e^{-\frac{(n - \mu_k)^2}{2\sigma_k^2}}$$

$$P_k(n) = \frac{\pi_k \cdot f_k(n)}{\sum_{l=1}^k \pi_l f_l(n)}$$

$$P_k(n) = \frac{\pi_k \cdot \frac{1}{\alpha \sqrt{2\pi} \sigma_k} \cdot e^{-\frac{(n - \mu_k)^2}{2\sigma_k^2}}}{\sum_{l=1}^k \pi_l \cdot f_l(n)}$$

$$\log(P_k(n)) = \log \pi_k - \log(\alpha \sqrt{2\pi} \sigma_k)$$

$$-\frac{(n - \mu_k)^2}{2\sigma_k^2}$$

$$= \log \pi_k - \log \alpha - \log \sqrt{2\pi} - \log \sigma_k$$

$$-\frac{(n - \mu_k)^2}{2\sigma_k^2} - \frac{(\mu_k)^2}{2\sigma_k^2} + \frac{2 \cdot n \mu_k}{2\sigma_k^2}$$

$$\delta_K(n) = \ln \bar{\mu}_K - \ln n - \frac{(n\bar{n})^2}{2\sigma^2} - \frac{(\bar{\mu}_K)^2}{2\sigma^2} + \frac{\ln(n\bar{n}\cdot\bar{\mu}_K)}{\sigma^2}$$

//

$$\delta_K(x) = \delta_2(x)$$

$$\ln \bar{\mu}_1 - \ln n - \frac{(n\bar{n})^2}{2\sigma^2} - \frac{(\bar{\mu}_1)^2}{2\sigma^2} + \frac{\ln(n\bar{n}\cdot\bar{\mu}_1)}{\sigma^2}$$

=

$$\ln \bar{\mu}_2 - \ln n - \frac{(n\bar{n})^2}{2\sigma^2} - \frac{(\bar{\mu}_2)^2}{2\sigma^2} + \frac{\ln(n\bar{n}\cdot\bar{\mu}_2)}{\sigma^2}$$

$$-\frac{(\bar{\mu}_1)^2}{2\sigma^2} + \frac{(\bar{\mu}_2)^2}{2\sigma^2} = \frac{\ln(n)\bar{\mu}_2}{\sigma^2} - \frac{\ln(n)\bar{\mu}_1}{\sigma^2}$$

$$\frac{\bar{\mu}_2^2 - \bar{\mu}_1^2}{\bar{\mu}_2 - \bar{\mu}_1} = 2\ln(x)$$

$$\bar{\mu}_1 + \bar{\mu}_2 - \ln(x)$$

$$\overrightarrow{2} \cdot \frac{M_1 + M_2}{2}$$

$$x = e^{\frac{M_1}{2}} e^{\frac{M_2}{2}}$$

$$= \sqrt{e^{M_1}} \sqrt{e^{M_2}}$$

3. Suppose we estimate the regression coefficients in a linear regression model by minimizing

$$\sum_{i=1}^n \left( y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right) \text{ subject to } \sum_{j=1}^p |\beta_j| \leq s$$

for a particular value of  $s$ . Indicate which of the following is true for items in 3a-3e, as we increase  $s$  from zero to infinity, and justify your answers:

- Increase initially, and then eventually start decreasing in an inverted U shape.
  - Decrease initially, and then eventually start increasing in a U shape.
  - Steadily increase.
  - Steadily decrease.
  - Remain constant.
- (a) Training RSS  
(b) Test RSS.  
(c) Variance.  
(d) (Squared) bias.  
(e) Irreducible error.

4. Consider the novel logistic regression method for binary classification ( $Y = 0$  or  $Y = 1$ ) with two features  $\mathbf{X} = (X_1, X_2)$ , formulated by

$$P(Y = 1|\mathbf{X}) = \frac{e^{\beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2}}{1 + e^{\beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2}}$$

Assume that using a dataset of 200 observations, we obtained the following estimates:

|           | Coefficient | Standard Error |
|-----------|-------------|----------------|
| $\beta_0$ | 1           | 0.2            |
| $\beta_1$ | 2           | 0.1            |
| $\beta_2$ | 1           | $s$            |
| $\beta_3$ | 1           | 0.5            |

- (a) Determine the equation for the decision boundary for this classifier.
- (b) Sketch the decision boundary for this classifier and clearly show the regions for the positive class and the negative class.
- (c) Find all values of  $s$  that makes the coefficient  $\beta_2$  statistically insignificant. You can consider the significance level to be  $\alpha = 0.05$ .

a) 
$$\frac{e^{\beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2}}{1 + e^{\beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2}} = 0.5$$

$$\Rightarrow e^{\beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2} = 1$$

$$1 + 2x_1 + x_1^2 + x_2 = 0$$

$$x_2 = -x_1^2 - 2x_1 - 1$$

b) for positive classes

$$P(Y=1|X=x) > 0.5$$

$$x_2 > -1 - 2x_1 - x_1^2$$

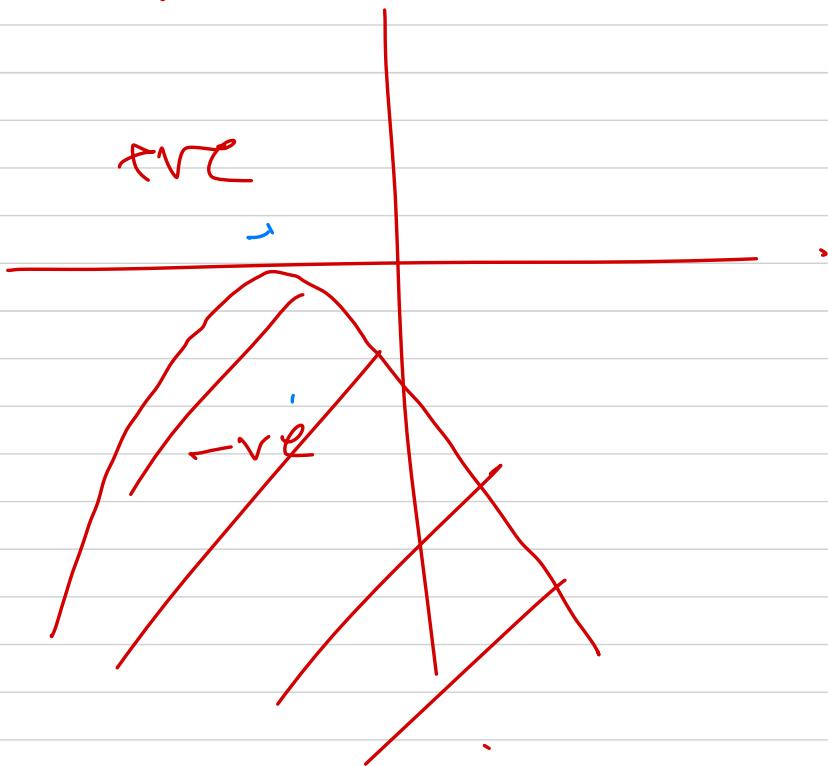
for -ve classes

$$P(Y=1 | X=x) \leq 0.5$$

$$x_2 \leq -2x_1 - x_1^2$$

Solving quadratic eqn.

$$\frac{c}{1+x^2} = 0.5$$



$$C_1 \quad \alpha = 0.05$$

$$t_{n-p-1, \alpha/2} = t_{199, 0.025} = 1.97$$

$$t = \frac{\beta_2 - 0}{SE(\beta_2)} = 1.97 = \frac{1}{SE(\beta_2)}$$

$$SE(\beta_2) = 0.50$$

$$0 < S < 0.50$$

5. The Federalist papers, authored by Alexander Hamilton, John Jay, and James Madison, consist a series of 85 papers published between October 1787 and April 1788 under the pseudonym PUBLIUS to convince the people of New York to ratify the US constitution. The authorship of some of the papers is in dispute. In particular, the authorship of 12 of the papers is disputed, while Hamilton and Madison later published their lists of authors of the rest, although even those lists have discrepancies. One can use Machine Learning algorithms to classify the disputed papers using papers with known authors.
- .
- (a) Formulate the above problem as a multi-class problem. How many classes does their outcome  $Y$  have? What are those classes?
  - (b) Some experts believe that some of the papers are collaborative efforts of two or sometimes all three of Hamilton, Jay, and Madison. Capturing multi-author papers can be done by formulating the problem as a multi-label problem. Explain what the labels are, if the labels are binary, and how the algorithm should label a paper that was solely written by Jay, a paper that was written by Hamilton and Madison, and a paper that was the collaborative work of Hamilton, Jay, and Madison.

Scratch paper

Name:

USC ID:

Scratch paper

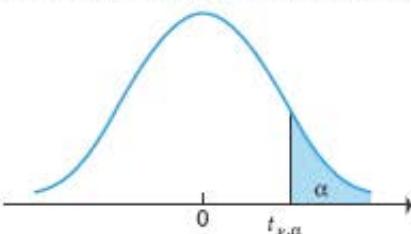
Name:

USC ID:

Cumulative Distribution Function,  $F(z)$ , of the Standard Normal Distribution Table

| $z$ | 0      | 0.01   | 0.02   | 0.03   | 0.04   | 0.05   | 0.06   | 0.07   | 0.08   | 0.09   |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 0.0 | 0.5000 | 0.5040 | 0.5080 | 0.5120 | 0.5160 | 0.5199 | 0.5239 | 0.5279 | 0.5319 | 0.5359 |
| 0.1 | 0.5398 | 0.5438 | 0.5478 | 0.5517 | 0.5557 | 0.5596 | 0.5636 | 0.5675 | 0.5714 | 0.5753 |
| 0.2 | 0.5793 | 0.5832 | 0.5871 | 0.5910 | 0.5948 | 0.5987 | 0.6026 | 0.6064 | 0.6103 | 0.6141 |
| 0.3 | 0.6179 | 0.6217 | 0.6255 | 0.6293 | 0.6331 | 0.6368 | 0.6406 | 0.6443 | 0.6480 | 0.6517 |
| 0.4 | 0.6554 | 0.6591 | 0.6628 | 0.6664 | 0.6700 | 0.6736 | 0.6772 | 0.6808 | 0.6844 | 0.6879 |
| 0.5 | 0.6915 | 0.6950 | 0.6985 | 0.7019 | 0.7054 | 0.7088 | 0.7123 | 0.7157 | 0.7190 | 0.7224 |
| 0.6 | 0.7257 | 0.7291 | 0.7324 | 0.7357 | 0.7389 | 0.7422 | 0.7454 | 0.7486 | 0.7517 | 0.7549 |
| 0.7 | 0.7580 | 0.7611 | 0.7642 | 0.7673 | 0.7704 | 0.7734 | 0.7764 | 0.7794 | 0.7823 | 0.7852 |
| 0.8 | 0.7881 | 0.7910 | 0.7939 | 0.7967 | 0.7995 | 0.8023 | 0.8051 | 0.8078 | 0.8106 | 0.8133 |
| 0.9 | 0.8159 | 0.8186 | 0.8212 | 0.8238 | 0.8264 | 0.8289 | 0.8315 | 0.8340 | 0.8365 | 0.8389 |
| 1.0 | 0.8413 | 0.8438 | 0.8461 | 0.8485 | 0.8508 | 0.8531 | 0.8554 | 0.8577 | 0.8599 | 0.8621 |
| 1.1 | 0.8643 | 0.8665 | 0.8686 | 0.8708 | 0.8729 | 0.8749 | 0.8770 | 0.8790 | 0.8810 | 0.8830 |
| 1.2 | 0.8849 | 0.8869 | 0.8888 | 0.8907 | 0.8925 | 0.8944 | 0.8962 | 0.8980 | 0.8997 | 0.9015 |
| 1.3 | 0.9032 | 0.9049 | 0.9066 | 0.9082 | 0.9099 | 0.9115 | 0.9131 | 0.9147 | 0.9162 | 0.9177 |
| 1.4 | 0.9192 | 0.9207 | 0.9222 | 0.9236 | 0.9251 | 0.9265 | 0.9279 | 0.9292 | 0.9306 | 0.9319 |
| 1.5 | 0.9332 | 0.9345 | 0.9357 | 0.9370 | 0.9382 | 0.9394 | 0.9406 | 0.9418 | 0.9429 | 0.9441 |
| 1.6 | 0.9452 | 0.9463 | 0.9474 | 0.9484 | 0.9495 | 0.9505 | 0.9515 | 0.9525 | 0.9535 | 0.9545 |
| 1.7 | 0.9554 | 0.9564 | 0.9573 | 0.9582 | 0.9591 | 0.9599 | 0.9608 | 0.9616 | 0.9625 | 0.9633 |
| 1.8 | 0.9641 | 0.9649 | 0.9656 | 0.9664 | 0.9671 | 0.9678 | 0.9686 | 0.9693 | 0.9699 | 0.9706 |
| 1.9 | 0.9713 | 0.9719 | 0.9726 | 0.9732 | 0.9738 | 0.9744 | 0.9750 | 0.9756 | 0.9761 | 0.9767 |
| 2.0 | 0.9772 | 0.9778 | 0.9783 | 0.9788 | 0.9793 | 0.9798 | 0.9803 | 0.9808 | 0.9812 | 0.9817 |
| 2.1 | 0.9821 | 0.9826 | 0.9830 | 0.9834 | 0.9838 | 0.9842 | 0.9846 | 0.9850 | 0.9854 | 0.9857 |
| 2.2 | 0.9861 | 0.9864 | 0.9868 | 0.9871 | 0.9875 | 0.9878 | 0.9881 | 0.9884 | 0.9887 | 0.9890 |
| 2.3 | 0.9893 | 0.9896 | 0.9898 | 0.9901 | 0.9904 | 0.9906 | 0.9909 | 0.9911 | 0.9913 | 0.9916 |
| 2.4 | 0.9918 | 0.9920 | 0.9922 | 0.9925 | 0.9927 | 0.9929 | 0.9931 | 0.9932 | 0.9934 | 0.9936 |
| 2.5 | 0.9938 | 0.9940 | 0.9941 | 0.9943 | 0.9945 | 0.9946 | 0.9948 | 0.9949 | 0.9951 | 0.9952 |
| 2.6 | 0.9953 | 0.9955 | 0.9956 | 0.9957 | 0.9959 | 0.9960 | 0.9961 | 0.9962 | 0.9963 | 0.9964 |
| 2.7 | 0.9965 | 0.9966 | 0.9967 | 0.9968 | 0.9969 | 0.9970 | 0.9971 | 0.9972 | 0.9973 | 0.9974 |
| 2.8 | 0.9974 | 0.9975 | 0.9976 | 0.9977 | 0.9977 | 0.9978 | 0.9979 | 0.9979 | 0.9980 | 0.9981 |
| 2.9 | 0.9981 | 0.9982 | 0.9982 | 0.9983 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9986 |
| 3.0 | 0.9987 | 0.9987 | 0.9987 | 0.9988 | 0.9988 | 0.9989 | 0.9989 | 0.9989 | 0.9990 | 0.9990 |
| 3.1 | 0.9990 | 0.9991 | 0.9991 | 0.9991 | 0.9992 | 0.9992 | 0.9992 | 0.9992 | 0.9993 | 0.9993 |
| 3.2 | 0.9993 | 0.9993 | 0.9994 | 0.9994 | 0.9994 | 0.9994 | 0.9994 | 0.9995 | 0.9995 | 0.9995 |
| 3.3 | 0.9995 | 0.9995 | 0.9995 | 0.9996 | 0.9996 | 0.9996 | 0.9996 | 0.9996 | 0.9996 | 0.9997 |

Cumulative Distribution Function,  $F(z)$ , of the Standard Normal Distribution Table

Upper Critical Values of Student's  $t$  Distribution with  $\nu$  Degrees of Freedom


For selected probabilities,  $\alpha$ , the table shows the values  $t_{\nu,\alpha}$  such that  $P(t_\nu > t_{\nu,\alpha}) = \alpha$ , where  $t_\nu$  is a Student's  $t$  random variable with  $\nu$  degrees of freedom. For example, the probability is .10 that a Student's  $t$  random variable with 10 degrees of freedom exceeds 1.372.

| $\nu$    | PROBABILITY OF EXCEEDING THE CRITICAL VALUE |       |        |        |        |         |
|----------|---|-------|--------|--------|--------|---------|
|          | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |
| 1        | 3.078                                       | 6.314 | 12.706 | 31.821 | 63.657 | 318.313 |
| 2        | 1.886                                       | 2.920 | 4.303  | 6.965  | 9.925  | 22.327  |
| 3        | 1.638                                       | 2.353 | 3.182  | 4.541  | 5.841  | 10.215  |
| 4        | 1.533                                       | 2.132 | 2.776  | 3.747  | 4.604  | 7.173   |
| 5        | 1.476                                       | 2.015 | 2.571  | 3.365  | 4.032  | 5.893   |
| 6        | 1.440                                       | 1.943 | 2.447  | 3.143  | 3.707  | 5.208   |
| 7        | 1.415                                       | 1.895 | 2.365  | 2.998  | 3.499  | 4.782   |
| 8        | 1.397                                       | 1.860 | 2.306  | 2.896  | 3.355  | 4.499   |
| 9        | 1.383                                       | 1.833 | 2.262  | 2.821  | 3.250  | 4.296   |
| 10       | 1.372                                       | 1.812 | 2.228  | 2.764  | 3.169  | 4.143   |
| 11       | 1.363                                       | 1.796 | 2.201  | 2.718  | 3.106  | 4.024   |
| 12       | 1.356                                       | 1.782 | 2.179  | 2.681  | 3.055  | 3.929   |
| 13       | 1.350                                       | 1.771 | 2.160  | 2.650  | 3.012  | 3.852   |
| 14       | 1.345                                       | 1.761 | 2.145  | 2.624  | 2.977  | 3.787   |
| 15       | 1.341                                       | 1.753 | 2.131  | 2.602  | 2.947  | 3.733   |
| 16       | 1.337                                       | 1.746 | 2.120  | 2.583  | 2.921  | 3.686   |
| 17       | 1.333                                       | 1.740 | 2.110  | 2.567  | 2.898  | 3.646   |
| 18       | 1.330                                       | 1.734 | 2.101  | 2.552  | 2.878  | 3.610   |
| 19       | 1.328                                       | 1.729 | 2.093  | 2.539  | 2.861  | 3.579   |
| 20       | 1.325                                       | 1.725 | 2.086  | 2.528  | 2.845  | 3.552   |
| 21       | 1.323                                       | 1.721 | 2.080  | 2.518  | 2.831  | 3.527   |
| 22       | 1.321                                       | 1.717 | 2.074  | 2.508  | 2.819  | 3.505   |
| 23       | 1.319                                       | 1.714 | 2.069  | 2.500  | 2.807  | 3.485   |
| 24       | 1.318                                       | 1.711 | 2.064  | 2.492  | 2.797  | 3.467   |
| 25       | 1.316                                       | 1.708 | 2.060  | 2.485  | 2.787  | 3.450   |
| 26       | 1.315                                       | 1.706 | 2.056  | 2.479  | 2.779  | 3.435   |
| 27       | 1.314                                       | 1.703 | 2.052  | 2.473  | 2.771  | 3.421   |
| 28       | 1.313                                       | 1.701 | 2.048  | 2.467  | 2.763  | 3.408   |
| 29       | 1.311                                       | 1.699 | 2.045  | 2.462  | 2.756  | 3.396   |
| 30       | 1.310                                       | 1.697 | 2.042  | 2.457  | 2.750  | 3.385   |
| 40       | 1.303                                       | 1.684 | 2.021  | 2.423  | 2.704  | 3.307   |
| 60       | 1.296                                       | 1.671 | 2.000  | 2.390  | 2.660  | 3.232   |
| 100      | 1.290                                       | 1.660 | 1.984  | 2.364  | 2.626  | 3.174   |
| $\infty$ | 1.282                                       | 1.645 | 1.960  | 2.326  | 2.576  | 3.090   |
| $\nu$    | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |

## F Table for $\alpha = 0.01$

| DF1 | $\alpha = 0.01$ |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |
|-----|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| DF2 | 1               | 2      | 3      | 4      | 5      | 6      | 7      | 8      | 9      | 10     | 12     | 15     | 20     | 24     | 30     | 40     | 60     | 120    | Inf    |
| 1   | 4052.2          | 4999.5 | 5403.4 | 5624.6 | 5763.7 | 5859   | 5928.4 | 5981.1 | 6022.5 | 6055.8 | 6106.3 | 6157.3 | 6208.7 | 6234.6 | 6260.6 | 6286.8 | 6313   | 6339.4 | 6365.9 |
| 2   | 98.503          | 99     | 99.166 | 99.249 | 99.299 | 99.333 | 99.356 | 99.374 | 99.388 | 99.399 | 99.416 | 99.433 | 99.449 | 99.458 | 99.466 | 99.474 | 99.482 | 99.491 | 99.499 |
| 3   | 34.116          | 30.817 | 29.457 | 28.71  | 28.237 | 27.911 | 27.672 | 27.489 | 27.345 | 27.229 | 27.052 | 26.872 | 26.69  | 26.598 | 26.505 | 26.411 | 26.316 | 26.221 | 26.125 |
| 4   | 21.198          | 18     | 16.694 | 15.977 | 15.522 | 15.207 | 14.976 | 14.799 | 14.659 | 14.546 | 14.374 | 14.198 | 14.02  | 13.929 | 13.838 | 13.745 | 13.652 | 13.558 | 13.463 |
| 5   | 16.258          | 13.274 | 12.06  | 11.392 | 10.967 | 10.672 | 10.456 | 10.289 | 10.158 | 10.051 | 9.888  | 9.722  | 9.553  | 9.466  | 9.379  | 9.291  | 9.202  | 9.112  | 9.02   |
| 6   | 13.745          | 10.925 | 9.78   | 9.148  | 8.746  | 8.466  | 8.26   | 8.102  | 7.976  | 7.874  | 7.718  | 7.559  | 7.396  | 7.313  | 7.229  | 7.143  | 7.057  | 6.969  | 6.88   |
| 7   | 12.246          | 9.547  | 8.451  | 7.847  | 7.46   | 7.191  | 6.993  | 6.84   | 6.719  | 6.62   | 6.469  | 6.314  | 6.155  | 6.074  | 5.992  | 5.908  | 5.824  | 5.737  | 5.65   |
| 8   | 11.259          | 8.649  | 7.591  | 7.006  | 6.632  | 6.371  | 6.178  | 6.029  | 5.911  | 5.814  | 5.667  | 5.515  | 5.359  | 5.279  | 5.198  | 5.116  | 5.032  | 4.946  | 4.859  |
| 9   | 10.561          | 8.022  | 6.992  | 6.422  | 6.057  | 5.802  | 5.613  | 5.467  | 5.351  | 5.257  | 5.111  | 4.962  | 4.808  | 4.729  | 4.649  | 4.567  | 4.483  | 4.398  | 4.31   |
| 10  | 10.044          | 7.559  | 6.552  | 5.994  | 5.636  | 5.386  | 5.2    | 5.057  | 4.942  | 4.849  | 4.706  | 4.558  | 4.405  | 4.327  | 4.247  | 4.165  | 4.082  | 3.996  | 3.909  |
| 11  | 9.646           | 7.206  | 6.217  | 5.668  | 5.316  | 5.069  | 4.886  | 4.744  | 4.632  | 4.539  | 4.397  | 4.251  | 4.099  | 4.021  | 3.941  | 3.86   | 3.776  | 3.69   | 3.602  |
| 12  | 9.33            | 6.927  | 5.953  | 5.412  | 5.064  | 4.821  | 4.64   | 4.499  | 4.388  | 4.296  | 4.155  | 4.01   | 3.858  | 3.78   | 3.701  | 3.619  | 3.535  | 3.449  | 3.362  |
| 13  | 9.074           | 6.701  | 5.739  | 5.205  | 4.862  | 4.62   | 4.441  | 4.302  | 4.191  | 4.1    | 3.96   | 3.815  | 3.665  | 3.587  | 3.507  | 3.425  | 3.341  | 3.255  | 3.165  |
| 14  | 8.862           | 6.515  | 5.564  | 5.035  | 4.695  | 4.456  | 4.278  | 4.14   | 4.03   | 3.939  | 3.8    | 3.656  | 3.505  | 3.427  | 3.348  | 3.266  | 3.181  | 3.094  | 3.004  |
| 15  | 8.683           | 6.359  | 5.417  | 4.893  | 4.556  | 4.318  | 4.142  | 4.004  | 3.895  | 3.805  | 3.666  | 3.522  | 3.372  | 3.294  | 3.214  | 3.132  | 3.047  | 2.959  | 2.868  |
| 16  | 8.531           | 6.226  | 5.292  | 4.773  | 4.437  | 4.202  | 4.026  | 3.89   | 3.78   | 3.691  | 3.553  | 3.409  | 3.259  | 3.181  | 3.101  | 3.018  | 2.933  | 2.845  | 2.753  |
| 17  | 8.4             | 6.112  | 5.185  | 4.669  | 4.336  | 4.102  | 3.927  | 3.791  | 3.682  | 3.593  | 3.455  | 3.312  | 3.162  | 3.084  | 3.003  | 2.92   | 2.835  | 2.746  | 2.653  |
| 18  | 8.285           | 6.013  | 5.092  | 4.579  | 4.248  | 4.015  | 3.841  | 3.705  | 3.597  | 3.508  | 3.371  | 3.227  | 3.077  | 2.999  | 2.919  | 2.835  | 2.749  | 2.66   | 2.566  |
| 19  | 8.185           | 5.926  | 5.01   | 4.5    | 4.171  | 3.939  | 3.765  | 3.631  | 3.523  | 3.434  | 3.297  | 3.153  | 3.003  | 2.925  | 2.844  | 2.761  | 2.674  | 2.584  | 2.489  |
| 20  | 8.096           | 5.849  | 4.938  | 4.431  | 4.103  | 3.871  | 3.699  | 3.564  | 3.457  | 3.368  | 3.231  | 3.088  | 2.938  | 2.859  | 2.778  | 2.695  | 2.608  | 2.517  | 2.422  |
| 21  | 8.017           | 5.78   | 4.874  | 4.369  | 4.042  | 3.812  | 3.64   | 3.506  | 3.398  | 3.31   | 3.173  | 3.03   | 2.88   | 2.801  | 2.72   | 2.636  | 2.548  | 2.457  | 2.36   |
| 22  | 7.945           | 5.719  | 4.817  | 4.313  | 3.988  | 3.758  | 3.587  | 3.453  | 3.346  | 3.258  | 3.121  | 2.978  | 2.827  | 2.749  | 2.667  | 2.583  | 2.495  | 2.403  | 2.305  |
| 23  | 7.881           | 5.664  | 4.765  | 4.264  | 3.939  | 3.71   | 3.539  | 3.406  | 3.299  | 3.211  | 3.074  | 2.931  | 2.781  | 2.702  | 2.62   | 2.535  | 2.447  | 2.354  | 2.256  |
| 24  | 7.823           | 5.614  | 4.718  | 4.218  | 3.895  | 3.667  | 3.496  | 3.363  | 3.256  | 3.168  | 3.032  | 2.889  | 2.738  | 2.659  | 2.577  | 2.492  | 2.403  | 2.31   | 2.21   |
| 25  | 7.77            | 5.568  | 4.675  | 4.177  | 3.855  | 3.627  | 3.457  | 3.324  | 3.217  | 3.129  | 2.993  | 2.85   | 2.699  | 2.62   | 2.538  | 2.453  | 2.364  | 2.27   | 2.165  |
| 26  | 7.721           | 5.526  | 4.637  | 4.14   | 3.818  | 3.591  | 3.421  | 3.288  | 3.182  | 3.094  | 2.958  | 2.815  | 2.664  | 2.585  | 2.503  | 2.417  | 2.327  | 2.233  | 2.13   |
| 27  | 7.677           | 5.488  | 4.601  | 4.106  | 3.785  | 3.558  | 3.388  | 3.256  | 3.149  | 3.062  | 2.926  | 2.783  | 2.632  | 2.552  | 2.47   | 2.384  | 2.294  | 2.198  | 2.09   |
| 28  | 7.636           | 5.453  | 4.568  | 4.074  | 3.754  | 3.528  | 3.358  | 3.226  | 3.12   | 3.032  | 2.896  | 2.753  | 2.602  | 2.522  | 2.44   | 2.354  | 2.263  | 2.167  | 2.064  |
| 29  | 7.598           | 5.42   | 4.538  | 4.045  | 3.725  | 3.499  | 3.33   | 3.198  | 3.092  | 3.005  | 2.868  | 2.726  | 2.574  | 2.495  | 2.412  | 2.325  | 2.234  | 2.138  | 2.034  |
| 30  | 7.562           | 5.39   | 4.51   | 4.018  | 3.699  | 3.473  | 3.304  | 3.173  | 3.067  | 2.979  | 2.843  | 2.7    | 2.549  | 2.469  | 2.386  | 2.299  | 2.208  | 2.111  | 2.006  |
| 40  | 7.314           | 5.179  | 4.313  | 3.828  | 3.514  | 3.291  | 3.124  | 2.993  | 2.888  | 2.801  | 2.665  | 2.522  | 2.369  | 2.288  | 2.203  | 2.114  | 2.019  | 1.917  | 1.805  |
| 60  | 7.077           | 4.977  | 4.126  | 3.649  | 3.339  | 3.119  | 2.953  | 2.823  | 2.718  | 2.632  | 2.496  | 2.352  | 2.198  | 2.115  | 2.028  | 1.936  | 1.836  | 1.726  | 1.602  |
| 120 | 6.851           | 4.787  | 3.949  | 3.48   | 3.174  | 2.956  | 2.792  | 2.663  | 2.559  | 2.472  | 2.336  | 2.192  | 2.035  | 1.95   | 1.86   | 1.763  | 1.656  | 1.533  | 1.382  |
| Inf | 6.635           | 4.605  | 3.782  | 3.319  | 3.017  | 2.802  | 2.639  | 2.511  | 2.407  | 2.321  | 2.185  | 2.039  | 1.878  | 1.791  | 1.696  | 1.592  | 1.473  | 1.325  | 1      |

**CSCI 567 Fall 2018 Midterm Exam**  
**DO NOT OPEN EXAM UNTIL INSTRUCTED TO DO SO**  
**PLEASE TURN OFF ALL CELL PHONES**

| Problem | 1  | 2  | 3  | 4  | 5  | 6  | Total |
|---------|----|----|----|----|----|----|-------|
| Max     | 16 | 10 | 16 | 42 | 24 | 12 | 120   |
| Points  |    |    |    |    |    |    |       |

Please read the following instructions carefully:

- The exam has a total of **22 pages** (double-sided, including this cover and two blank pages in the end) and **6 questions**. Each question have sub-questions. Once you are permitted to open your exam (and not before), you should check and make sure that you are not missing any pages.
- This is a **closed-book/notes** exam. Consulting any resources is NOT permitted.
- Any kind of cheating will lead to **score 0** for the entire exam and be reported to SJACS.
- Duration of the exam is 2 hours. Questions are not ordered by their difficulty. Budget your time on each question carefully.
- Answers should be **concise** and written down **legibly**. All questions can be done within 5-10 lines.
- You must answer each question on the page provided. You can use the last two blank pages as scratch paper. Raise your hand to ask a proctor for more if needed.
- Select **one and only one answer** for all multiple choice questions.
- You **may not** leave your seat **for any reason** unless you submit your exam at that point.

# 1 Machine Learning Concepts (16 points)

## 1.1 Multiple Choice (12 points)

- (a) Which of the following is a parametric model?
- (A) Nearest neighbor classifier
  - (B) Support vector machines with a Gaussian kernel
  - (C) Linear regression with a polynomial kernel
  - (D) fully connected feedforward neural nets

Ans: D.

(2 points)

- (b) Which of the following phenomenon is called overfitting?

- (A) low training error, low test error
- (B) low training error, high test error
- (C) high training error, low test error
- (D) high training error, high test error

Ans: B.

(2 points)

- (c) Suppose the conditional probability of seeing label  $k$  given  $\mathbf{x}$  is  $\mathbb{P}(y = k | \mathbf{x})$ . Which of the following is the Bayes optimal classifier?

- (A) A randomized classifier that predicts label  $k$  with probability  $\mathbb{P}(y = k | \mathbf{x})$  given  $\mathbf{x}$ .
- (B) A deterministic classifier that predicts  $\text{argmax}_k \mathbb{P}(y = k | \mathbf{x})$  given  $\mathbf{x}$ .
- (C) A deterministic classifier that predicts  $\text{argmin}_k \mathbb{P}(y = k | \mathbf{x})$  given  $\mathbf{x}$ .
- (D) Both (A) and (B).

Ans: B. See Lec 1.

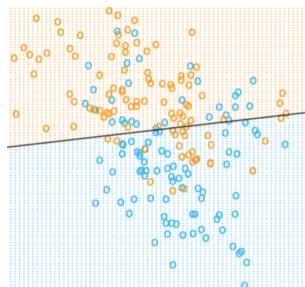
(2 points)

- (d) Which of the following cannot be used as regularization to control model complexity?

- (A)  $R(\mathbf{w}) = \sum_{d=1}^D |\mathbf{w}_d|$
- (B)  $R(\mathbf{w}) = \sum_{d=1}^D |\mathbf{w}_d|^3$
- (C)  $R(\mathbf{w}) = \sum_{d=1}^D \mathbf{w}_d^3$
- (D)  $R(\mathbf{w}) = \sum_{d=1}^D |\mathbf{w}_d|^4$

C.  $R$  is unbounded both above and below. As a result, minimizing or maximizing it will push the objective function to  $-\infty$  or  $\infty$ .  
(2 points)

- (e) Which of the these classifiers could have generated this decision boundary?



- (A) SVM
- (B) 1-nearest-neighbor with L2 distance
- (C) Logistic regression
- (D) A & C

Ans: D. Both are linear classifiers.

(2 points)

- (f) For a fixed multiclass problem, which of the following multiclass-to-binary reductions has the smallest testing time complexity?
- (A) One-versus-all
  - (B) One-versus-one
  - (C) Tree reduction
  - (D) Both (A) and (C)

Ans: C.

(2 points)

## 1.2 Multiclass reduction (4 points)

Show that one-versus-all can be seen as a special case of error-correcting-output-code (ECOC). Specifically, write down the code matrix  $M$  for ECOC for a problem with  $C$  labels so that executing ECOC is the same as doing one-versus-all. (Note: the entry of  $M$  should be either  $-1$  or  $+1$ .)

$M$  is a  $C \times C$  matrix with  $+1$  on the diagonal and  $-1$  on every other entries, or a  $C \times C$  matrix with  $-1$  on the diagonal and  $+1$  on every other entries. Either one will get full credit.

Only getting the dimension right will get 1 point.

## 2 Nearest Neighbor Classification (10 points)

### 2.1 First Problem (4 points)

We mentioned that the Euclidean/L2 distance is often used as the *default* distance for nearest neighbor classification. It is defined as

$$E(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2^2 = \sum_{d=1}^D (x_d - x'_d)^2 \quad (1)$$

In some applications such as information retrieval, the cosine distance is widely used too. It is defined as

$$C(\mathbf{x}, \mathbf{x}') = 1 - \frac{\mathbf{x}^T \mathbf{x}'}{\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2} = 1 - \frac{\sum_{d=1}^D (x_d \cdot x'_d)}{\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2}, \quad (2)$$

where the L2 norm of  $\mathbf{x}$  is defined as

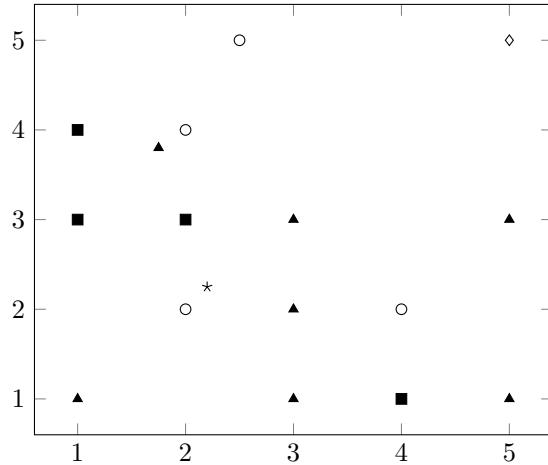
$$\|\mathbf{x}\|_2 = \sqrt{\sum_{d=1}^D x_d^2}. \quad (3)$$

Show that, if data is normalized with unit L2 norm, that is,  $\|\mathbf{x}\| = 1$  for all  $\mathbf{x}$  in the training and test sets, changing the distance function from the Euclidean distance to the cosine distance will NOT affect the nearest neighbor classification results.

When data is normalized, we have  $E(\mathbf{x}, \mathbf{x}') = \sum_{d=1}^D (x_d - x'_d)^2 = \sum_{d=1}^D (x_d^2 + x'^2_d - 2x_d x'_d) = 2(1 - \sum_{d=1}^D (x_d \cdot x'_d)) = 2C(\mathbf{x}, \mathbf{x}')$ . Scaling any distance measurement by a factor of two clearly will not affect the results of NNC. (4 points)

## 2.2 Second Problem (6 points)

For the data given below, squares, triangles, and open circles are three different classes of data in the training set and the diamond ( $\diamond$ ) and star (\*) are test points. We denote the total number of training points as  $N$  and consider K-nearest-neighbor (KNN) classifier with L2 distance.



- (a) When  $K = 1$ , how many *circles*(o) will be misclassified (as a validation point) when one performs leave-one-out validation (that is,  $N$ -fold cross validation)? (A single number as the answer is enough.)

3

(2 points)

- (b) What is the minimum value of  $K$  for which the *star*(\*) will be classified as a *triangle*? (A single number as the answer is enough.)

4

(2 points)

- (c) What is the *diamond* classified as for  $K = N$ ? Explain why.

Triangle

(1 point)

When  $K = N$  the prediction is always the majority label of the training set. Triangle is the majority in this example.

(1 point)

### 3 Linear Regression (16 points)

#### 3.1 First Problem (4 points)

For a training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) \in \mathbb{R}^D \times \mathbb{R}$ , let  $\mathbf{w}_*$  be the least square solution with no regularization (assume  $\mathbf{X}^\top \mathbf{X}$  is invertible where  $\mathbf{X} \in \mathbb{R}^{N \times D}$  is the data matrix with each row corresponding to the feature of an example, as used in the class). Find the least square solution (with no regularization again) of the following new training set:  $(\mathbf{x}_1, y_1 - \mathbf{w}_*^\top \mathbf{x}_1), \dots, (\mathbf{x}_N, y_N - \mathbf{w}_*^\top \mathbf{x}_N)$ .

Using the formula derived in the class, we know the new least square solution  $\mathbf{w}'_*$  is

$$\mathbf{w}'_* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top (\mathbf{y} - \mathbf{X} \mathbf{w}_*) = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} - (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{X} ((\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}) = \mathbf{0}.$$

Rubrics:

- Write down the correct formula for  $\mathbf{w}_*$ . (1 point)
- Use the correct formula to find  $\mathbf{w}'_*$  or re-derive in a correct way (e.g. correct gradient). (2 points)
- Get the correct answer finally  $\mathbf{w}'_* = \mathbf{0}$ . (1 point)

### 3.2 Second Problem (12 points)

Assume we have a training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) \in \mathbb{R}^D \times \mathbb{R}$ , where each outcome  $y_n$  is generated by a probabilistic model  $\mathbf{w}_*^\top \mathbf{x}_n + \epsilon_n$  with  $\epsilon_n$  being an independent Gaussian noise with zero-mean and variance  $\sigma^2$  for some  $\sigma > 0$ . In other words, the probability of seeing any outcome  $y \in \mathbb{R}$  given  $\mathbf{x}_n$  is

$$\mathbb{P}(y | \mathbf{x}_n; \mathbf{w}_*, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y - \mathbf{w}_*^\top \mathbf{x}_n)^2}{2\sigma^2}\right).$$

- (a) Assume  $\sigma$  is fixed and given, find the maximum likelihood estimation for  $\mathbf{w}_*$ . In other words, first write down the probability of seeing the outcomes  $y_1, \dots, y_N$  given  $\mathbf{x}_1, \dots, \mathbf{x}_N$  as a function of the value of  $\mathbf{w}_*$ ; then find the value of  $\mathbf{w}_*$  that maximizes this probability. You can assume  $\mathbf{X}^\top \mathbf{X}$  is invertible where  $\mathbf{X}$  is the data matrix as used in Problem 3.1.

The probability of seeing the outcomes  $y_1, \dots, y_N$  given  $\mathbf{x}_1, \dots, \mathbf{x}_N$  for a linear model  $\mathbf{w}$  is

$$\mathcal{P}(\mathbf{w}) = \prod_{i=1}^N \mathbb{P}(y_n | \mathbf{x}_n; \mathbf{w}, \sigma) = \prod_{i=1}^N \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(y_n - \mathbf{w}^\top \mathbf{x}_n)^2}{2\sigma^2}\right).$$

Taking the negative log, this becomes

$$F(\mathbf{w}) = N \ln \sqrt{2\pi} + N \ln \sigma + \frac{1}{2\sigma^2} \sum_{n=1}^N (y_n - \mathbf{w}^\top \mathbf{x}_n)^2 = N \ln \sqrt{2\pi} + N \ln \sigma + \frac{1}{2\sigma^2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$

Maximizing  $\mathcal{P}$  is the same as minimizing  $F$ , which is clearly the same as just minimizing  $\sum_{n=1}^N (y_n - \mathbf{w}^\top \mathbf{x}_n)^2$ , the same objective as for least square regression. Therefore the MLE for  $\mathbf{w}_*$  is exactly the same as the least square solution:

$$\mathbf{w}_* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Rubrics:

- Write down the correct likelihood function. (2 points)
- Any derivation that reveals this is the same as least square regression. (2 points)
- Arrive at the correct final answer. (2 points)

- (b) Now consider  $\sigma$  as a parameter of the probabilistic model too, that is, the model is specified by both  $\mathbf{w}_*$  and  $\sigma$ . Find the maximum likelihood estimation for  $\mathbf{w}_*$  and  $\sigma$ .

From previous calculation, the MLE for  $\mathbf{w}_*$  and  $\sigma$  is the minimizer of the function

$$F(\mathbf{w}, \sigma) = N \ln \sqrt{2\pi} + N \ln \sigma + \frac{1}{2\sigma^2} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2.$$

We first fix  $\sigma$  and minimize over  $\mathbf{w}$ , which leads to the same MLE for  $\mathbf{w}_*$ :

$$\mathbf{w}_* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Next we minimize  $F(\mathbf{w}_*, \sigma)$  as function of  $\sigma$  by setting the derivative w.r.t.  $\sigma$  to be 0:

$$\frac{\partial F(\mathbf{w}_*, \sigma)}{\partial \sigma} = \frac{N}{\sigma} - \frac{1}{\sigma^3} \|\mathbf{X}\mathbf{w}_* - \mathbf{y}\|_2^2 = 0.$$

Solving for  $\sigma$  gives the MLE estimate:

$$\sigma = \frac{1}{\sqrt{N}} \|\mathbf{X}\mathbf{w}_* - \mathbf{y}\|_2 = \frac{1}{\sqrt{N}} \|\mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} - \mathbf{y}\|_2.$$

Rubrics:

- Write down the correct likelihood as a function of both  $\mathbf{w}$  and  $\sigma$ . (1 point)
- Arrive at the correct solution for  $\mathbf{w}_*$ . (2 points)
- Correct derivative with respect to  $\sigma$ . (2 points)
- Arrive at the correct solution for  $\sigma$ . (1 point)

## 4 Linear Classifiers (42 points)

### 4.1 Multiple Choice (6 points)

(a) Which of the following is true?

- (A) When the data is linearly separable, logistic loss (without regularization) does not admit a minimizer.
- (B) When the data is linearly separable, perceptron converges to a max-margin classifier.
- (C) Normalizing the output  $\mathbf{w}$  of the perceptron algorithm so that  $\|\mathbf{w}\|_2 = 1$  changes its test error.
- (D) Normalizing the output  $\mathbf{w}$  of the perceptron algorithm so that  $\|\mathbf{w}\|_1 = 1$  changes its test error.

Ans: A. Take the binary case as an example, logistic loss is  $\sum_{n=1} \ln(1 + \exp(-y_n \mathbf{w}^T \mathbf{x}_n))$ . When data is separable, one can find  $\mathbf{w}$  such that  $y_n \mathbf{w}^T \mathbf{x}_n \geq 0$  for all  $n$ . Scaling this  $\mathbf{w}$  up will always lead to smaller loss and thus the function does not admit a minimizer. (2 points)

(b) The perceptron algorithm makes an update  $\mathbf{w}' \leftarrow \mathbf{w} + \eta y_n \mathbf{x}_n$  with  $\eta = 1$  when  $\mathbf{w}$  misclassifies  $\mathbf{x}_n$ . Using which of the following different values for  $\eta$  will make sure  $\mathbf{w}'$  classifies  $\mathbf{x}_n$  correctly?

- (A)  $\eta > \frac{y(\mathbf{w}^T \mathbf{x}_n)}{\|\mathbf{x}_n\|_2^2}$
- (B)  $\eta < \frac{-y(\mathbf{w}^T \mathbf{x}_n)}{\|\mathbf{x}_n\|_2^2 + 1}$
- (C)  $\eta < \frac{-y(\mathbf{w}^T \mathbf{x}_n)}{\|\mathbf{x}_n\|_2^2}$
- (D)  $\eta > \frac{-y(\mathbf{w}^T \mathbf{x}_n)}{\|\mathbf{x}_n\|_2^2}$

Ans: D. Solve  $y_n \mathbf{w}'^T \mathbf{x}_n = y_n (\mathbf{w} + \eta y_n \mathbf{x}_n)^T \mathbf{x}_n > 0$  for  $\eta$ . (2 points)

(c) Which of the following about SVM is true?

- (A) Support vectors are training points that are misclassified.
- (B) Support vectors are training points that are on the learned hyperplane.
- (C) It is possible that a support vector is on the learned hyperplane.
- (D) Only misclassified training points could be support vectors, but not all of them are.

Ans: C. (2 points)

### 4.2 Perceptron (4 points)

The following table shows a binary classification training set and the number of times each point is misclassified during a run of the perceptron algorithm. Write down the final output of the algorithm (that is, the weight vector that represents the hyperplane) assuming we append constant 1 as the first feature to all examples and start with the all-zero weight vector.

| $\mathbf{x}$ | y  | Times misclassified |
|--------------|----|---------------------|
| (-3, 2)      | +1 | 5                   |
| (-1, 1)      | -1 | 5                   |
| (5, 2)       | +1 | 3                   |
| (2, 2)       | -1 | 4                   |
| (1, -2)      | +1 | 3                   |

By the algorithm, the weight is a linear combination of data points weighted by the number of times they are misclassified and their label. That is (2 points)

$$\mathbf{w} = 5 \times (1, -3, 2) - 5 \times (1, -1, 1) + 3 \times (1, 5, 2) - 4 \times (1, 2, 2) + 3 \times (1, 1, -2) = (2, 0, -3). \quad (2 \text{ points})$$

### 4.3 Kernelized Perceptron (6 points)

---

**Algorithm 1:** Perceptron with nonlinear mapping  $\phi$

---

```

1 Input: A training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ 
2 Initialize:  $\mathbf{w} = \mathbf{0}$ 
3 while not converged do
4   randomly pick an example  $(\mathbf{x}_n, y_n)$ , make prediction  $\hat{y} = \text{sign}(\mathbf{w}^T \phi(\mathbf{x}_n))$ 
5   if  $\hat{y} \neq y_n$  then
6      $\mathbf{w} \leftarrow \mathbf{w} + y_n \phi(\mathbf{x}_n)$ 

```

---



---

**Algorithm 2:** Perceptron with kernel function  $k$

---

```

1 Input: A training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ 
2 Initialize:  $\alpha_n = 0$  for  $n = 1, \dots, N$  (2 points)
3 while not converged do
4   randomly pick an example  $(\mathbf{x}_n, y_n)$ , make prediction  $\hat{y} = \text{sign}\left(\sum_{m=1}^N \alpha_m k(\mathbf{x}_m, \mathbf{x}_n)\right)$  (2 points)
5   if  $\hat{y} \neq y_n$  then
6      $\alpha_n \leftarrow \alpha_n + y_n$  (2 points)

```

---

Algorithm 1 is the perceptron algorithm with a nonlinear mapping  $\phi$ . Now given the corresponding kernel function  $k$ , please kernelize the algorithm and fill out the missing details in Algorithm 2.

#### 4.4 Multiclass Perceptron (6 points)

Recall that a linear model for a multiclass classification problem with  $C$  classes is parameterized by  $C$  weight vectors  $\mathbf{w}_1, \dots, \mathbf{w}_C \in \mathbb{R}^D$ . In the class we derive the multiclass logistic regression by minimizing the multiclass logistic loss. In this problem you need to derive the multiclass perceptron algorithm in a similar way. Specifically, the multiclass perceptron loss on a training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) \in \mathbb{R}^D \times [C]$  is defined as

$$F(\mathbf{w}_1, \dots, \mathbf{w}_C) = \sum_{n=1}^N \max \left\{ 0, \max_{k \neq y_n} \mathbf{w}_k^\top \mathbf{x}_n - \mathbf{w}_{y_n}^\top \mathbf{x}_n \right\}.$$

Similarly to the binary case, multiclass perceptron is simply applying SGD with learning rate 1 to minimize the multiclass perceptron loss. Based on the above information, write down the multiclass perceptron algorithm below. (For simplicity, you do not need to worry about the non-differential points of  $F$ . In other words, the term  $\max_{k \neq y_n} \mathbf{w}_k^T \mathbf{x}_n - \mathbf{w}_{y_n}^T \mathbf{x}_n$  is never 0.)

## Solutions:

**Algorithm 3:** Multiclass Perceptron

- 1 Input:** A training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$
  - 2 Initialize:**  $\mathbf{w}_1 = \dots = \mathbf{w}_C = 0$  (or randomly)
  - 3 while**  $not converged$  **do**
  - 4** | randomly pick an example  $(\mathbf{x}_n, y_n)$ , make prediction  $\hat{y} = \operatorname{argmax}_{k \in [C]} \mathbf{w}_k^T \mathbf{x}_n$       (2 points)
  - 5** | **if**  $\hat{y} \neq y_n$  **then**
  - 6** | |  $\mathbf{w}_{\hat{y}} \leftarrow \mathbf{w}_{\hat{y}} - \mathbf{x}_n$       (2 points)
  - 7** | |  $\mathbf{w}_{y_n} \leftarrow \mathbf{w}_{y_n} + \mathbf{x}_n$       (2 points)

## 4.5 Perceptron for Non-separable Data (20 points)

*Warning: you might find this problem the hardest and want to finish other problems first.*

In Homework 1 you are asked to prove a bound on the number of mistakes that perceptron makes for a separable data set. In this problem you need to follow the steps below to prove a bound for the general case with possibly non-separable data. In particular we consider the following variant of perceptron where instead of randomly picking a data point in each iteration, we simply make one pass of the data set. We also assume that the data is normalized so that  $\|\mathbf{x}_n\|_2 = 1$  for all  $n \in [N]$ .

---

**Algorithm 4:** One-pass Perceptron

---

```

1 Input: A training set  $S = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ 
2 Initialize:  $\mathbf{w}_1 = \mathbf{0}$ 
3 for  $n = 1, \dots, N$  do
4   make prediction  $\hat{y} = \text{sign}(\mathbf{w}_n^T \mathbf{x}_n)$ 
5   if  $\hat{y} \neq y_n$  then
6      $\mathbf{w}_{n+1} = \mathbf{w}_n + y_n \mathbf{x}_n$ 

```

---

- (a) Let  $M$  be the number of mistakes Algorithm 4 makes. Prove  $\|\mathbf{w}_{N+1}\|_2 \leq \sqrt{M}$ .

This is exactly the same as the question in Homework 1. By the algorithm we have for any  $n$ ,

$$\begin{aligned}
\|\mathbf{w}_{n+1}\|_2^2 &= \|\mathbf{w}_n + \mathbb{I}[y_n \mathbf{w}_n^T \mathbf{x}_n \leq 0] y_n \mathbf{x}_n\|_2^2 \\
&= \|\mathbf{w}_n\|_2^2 + \mathbb{I}[y_n \mathbf{w}_n^T \mathbf{x}_n \leq 0] (2y_n \mathbf{w}_n^T \mathbf{x}_n + y_n^2 \|\mathbf{x}_n\|_2^2) \\
&\leq \|\mathbf{w}_n\|_2^2 + \mathbb{I}[y_n \mathbf{w}_n^T \mathbf{x}_n \leq 0]
\end{aligned} \tag{2 points}$$

Applying this inequality recursively we get  $\|\mathbf{w}_{N+1}\|_2^2 \leq \sum_{n=1}^N \mathbb{I}[y_n \mathbf{w}_n^T \mathbf{x}_n \leq 0] = M$  and thus  $\|\mathbf{w}_{N+1}\|_2 \leq \sqrt{M}$ . (1 point)

- (b) Next show that for any  $\mathbf{w}_*$  with  $\|\mathbf{w}_*\|_2 \leq 1$ , we have  $\mathbf{w}_{N+1}^\top \mathbf{w}_* \geq M - M_*$  where  $M_* = \sum_{n=1}^N \max\{0, 1 - y_n \mathbf{w}_*^\top \mathbf{x}_n\}$  is the total hinge loss of  $\mathbf{w}_*$ .

For any  $n$ , we have

$$\begin{aligned}\mathbf{w}_{n+1}^\top \mathbf{w}_* &= \mathbf{w}_n^\top \mathbf{w}_* + \mathbb{I}[y_n \mathbf{w}_n^\top \mathbf{x}_n \leq 0] y_n \mathbf{x}_n^\top \mathbf{w}_* \\ &= \mathbf{w}_n^\top \mathbf{w}_* + \mathbb{I}[y_n \mathbf{w}_n^\top \mathbf{x}_n \leq 0] (1 - (1 - y_n \mathbf{x}_n^\top \mathbf{w}_*)) \\ &\geq \mathbf{w}_n^\top \mathbf{w}_* + \mathbb{I}[y_n \mathbf{w}_n^\top \mathbf{x}_n \leq 0] (1 - \max\{0, 1 - y_n \mathbf{x}_n^\top \mathbf{w}_*\}) \\ &\geq \mathbf{w}_n^\top \mathbf{w}_* + \mathbb{I}[y_n \mathbf{w}_n^\top \mathbf{x}_n \leq 0] - \max\{0, 1 - y_n \mathbf{x}_n^\top \mathbf{w}_*\}\end{aligned}\quad \begin{matrix}(1 \text{ point}) \\ (1 \text{ point}) \\ (1 \text{ point})\end{matrix}$$

Applying this inequality recursively we get

$$\mathbf{w}_{N+1}^\top \mathbf{w}_* \geq \sum_{n=1}^N \mathbb{I}[y_n \mathbf{w}_n^\top \mathbf{x}_n \leq 0] - \sum_{n=1}^N \max\{0, 1 - y_n \mathbf{x}_n^\top \mathbf{w}_*\} = M - M_*.\quad (1 \text{ point})$$

- (c) Combine the above two results to prove a mistake bound  $M \leq 2M_* + 1$ . (Hint: as in Homework 1 you need to use the Cauchy-Schwarz inequality  $\mathbf{a}^\top \mathbf{b} \leq \|\mathbf{a}\|_2 \|\mathbf{b}\|_2$ )

Since

$$\mathbf{w}_{N+1}^\top \mathbf{w}_* \leq \|\mathbf{w}_{N+1}\|_2 \|\mathbf{w}_*\|_2 \leq \sqrt{M}\quad (1 \text{ point})$$

and

$$\sqrt{M} \leq \frac{1}{2}(M + 1),\quad (1 \text{ point})$$

we have  $M - M_* \leq \frac{1}{2}(M + 1)$ . Solving for  $M$  gives  $M \leq 2M_* + 1$ .  
(Directly solving  $M - M_* \leq \sqrt{M}$  for  $M$  also works too.)

- (d) Assume each data point in the training set  $S$  is an i.i.d. sample of a fixed distribution  $\mathcal{P}$ , which we denote as  $S \sim \mathcal{P}$ . Recall that the risk (in terms of 0-1 loss) of a classifier  $f$  is defined as  $R(f) = \mathbb{E}[\mathbb{I}[f(\mathbf{x}) \neq y]]$  where the expectation is with respect to the random draw of  $(\mathbf{x}, y) \sim \mathcal{P}$  and the potential internal randomness of the classifier  $f$ . We also define the risk in terms of the hinge loss of a linear classifier  $\mathbf{w}$  as  $R_{\text{hinge}}(\mathbf{w}) = \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{P}}[\max\{0, 1 - y\mathbf{w}^T \mathbf{x}\}]$ .

Now consider the following *randomized* classifier  $f$ : for any  $\mathbf{x}$ , the prediction is  $\text{sign}(\mathbf{w}_n^T \mathbf{x})$  where  $\mathbf{w}_n$  is drawn uniformly at random from  $\mathbf{w}_1, \dots, \mathbf{w}_N$  defined in Algorithm 4. Prove

$$\mathbb{E}_{S \sim \mathcal{P}}[R(f)] \leq 2 \min_{\mathbf{w}: \|\mathbf{w}\|_2 \leq 1} R_{\text{hinge}}(\mathbf{w}) + \frac{1}{N}.$$

The result from the last question can be written as

$$\frac{1}{N} \sum_{n=1}^N \mathbb{I}[\text{sign}(\mathbf{w}_n^T \mathbf{x}_n) \neq y_n] \leq \frac{2}{N} \sum_{n=1}^N \max\{0, 1 - y_n \mathbf{w}_n^T \mathbf{x}_n\} + \frac{1}{N} \quad (1 \text{ point})$$

for any  $\mathbf{w}$  such that  $\|\mathbf{w}\|_2 \leq 1$ . Taking the expectation on both sides with respect to  $S \sim \mathcal{P}$  we arrive at

$$\frac{1}{N} \sum_{n=1}^N \mathbb{E}_{(\mathbf{x}_1, y_1) \sim \mathcal{P}, \dots, (\mathbf{x}_{n-1}, y_{n-1}) \sim \mathcal{P}} \mathbb{E}_{(\mathbf{x}_n, y_n) \sim \mathcal{P}} [\mathbb{I}[\text{sign}(\mathbf{w}_n^T \mathbf{x}_n) \neq y_n]] \leq 2R_{\text{hinge}}(\mathbf{w}) + \frac{1}{N}. \quad (2 \text{ points})$$

Since  $\mathbf{w}_n$  is independent of the flash new example  $(\mathbf{x}_n, y_n)$ , we have

$$\mathbb{E}_{(\mathbf{x}_n, y_n) \sim \mathcal{P}} [\mathbb{I}[\text{sign}(\mathbf{w}_n^T \mathbf{x}_n) \neq y_n]] = R(\mathbf{w}_n), \quad (2 \text{ points})$$

the risk of the linear classifier  $\mathbf{w}_n$ . Therefore we have

$$\mathbb{E}_{S \sim \mathcal{P}} \left[ \frac{1}{N} \sum_{n=1}^N R(\mathbf{w}_n) \right] \leq 2R_{\text{hinge}}(\mathbf{w}) + \frac{1}{N}. \quad (1 \text{ point})$$

By the definition of the randomized classifier  $f$ , the left hand side is exactly  $\mathbb{E}_{S \sim \mathcal{P}}[R(f)]$ , proving the claimed statement since  $\mathbf{w}$  is any weight such that  $\|\mathbf{w}\|_2 \leq 1$ . (1 point)

- (e) Suppose we now make  $T$  passes (instead of one pass) of the data to obtain  $\mathbf{w}_1, \dots, \mathbf{w}_{TN}$  using the perceptron algorithm. Similar to  $f$  defined in the last question, we define a new randomized classifier  $f'$  which predicts by randomly picking one of the  $TN$  hyperplanes  $\mathbf{w}_1, \dots, \mathbf{w}_{TN}$  and then following the prediction of that hyperplane. Does the following hold

$$\mathbb{E}_{S \sim \mathcal{P}}[R(f')] \leq 2 \min_{\mathbf{w}: \|\mathbf{w}\|_2 \leq 1} R_{\text{hinge}}(\mathbf{w}) + \frac{1}{TN}?$$

Prove it if you think so, otherwise briefly explain why this is not true.

This is not true. (1 point)

Following the previous proof you will find that it breaks if one wants to claim for example

$$\mathbb{E}_{(\mathbf{x}_1, y_1) \sim \mathcal{P}}[\mathbb{I}[\text{sign}(\mathbf{w}_{N+1}^T \mathbf{x}_1) \neq y_1]] = R(\mathbf{w}_{N+1}).$$

This is because  $\mathbf{w}_{N+1}$  depends on  $(\mathbf{x}_1, y_1)$ .

Another way to see that this should not be true is: if it was true, then by making many passes of a training set of size one, the term  $\frac{1}{TN}$  will still go to zero, leading to a very strong risk bound, which is clearly impossible.

Similar explanations along this line will all get 2 points.

## 5 Neural Networks (24 points)

### 5.1 Multiple Choice (10 points)

- (a) Overfitting is a major problem for neural networks. Which of the following can help prevent overfitting?
- (A) Retraining on the same data many times.
  - (B) Using a larger learning rate for Backpropagation .
  - (C) Dropping random neurons in each iteration of Backpropagation.
  - (D) Training until you get the smallest training error.

Ans: C.

(2 points)

- (b) Which of the following is true about neural nets?

- (A) A fully connected feedforward neural net without nonlinear activation functions is the same as a linear model.
- (B) Since we usually apply Backpropagation (i.e. SGD) to learn neural networks, it is a convex problem.
- (C) A neural net with one hidden layer and a fixed number of neurons can represent any continuous function.
- (D) A max-pooling layer with a  $2 \times 2$  filter has 4 parameters to be learned.

Ans: A.

(2 points)

- (c) Suppose a convolution layer takes a  $4 \times 6$  image with 3 channels as input and outputs a  $3 \times 4 \times 6$  volume. Which of the following is a possible configuration of this layer?

- (A) One  $2 \times 3$  filter with depth 6, stride 1, no zero-padding.
- (B) Six  $2 \times 3$  filters with depth 3, stride 1, no zero-padding.
- (B) Six  $3 \times 4$  filters with depth 3, stride 2, no zero-padding.
- (B) Six  $3 \times 4$  filters with depth 3, stride 1, 1 pixel of zero-padding.

Ans: B.

(2 points)

- (d) How many parameters do we need to learn for the following network structure? An  $8 \times 8 \times 3$  image input, followed by a convolution layer with 2 filters of size  $2 \times 2$  (stride 1, no zero-padding), then another convolution layer with 4 filters of size  $3 \times 3$  (stride 2, no zero-padding), and finally a max-pooling layer with a  $2 \times 2$  filter (stride 1, no zero-padding). (Note: the depth of all filters are not explicitly spelled out, and we assume no bias/intercept terms are used.)

- (A) 96
- (B) 44
- (C) 100
- (D) 48

Ans: A.  $2 \times (2 \times 2 \times 3) + 4 \times (3 \times 3 \times 2) = 96$ .

(2 points)

- (e) What is the final output dimension of the last question?

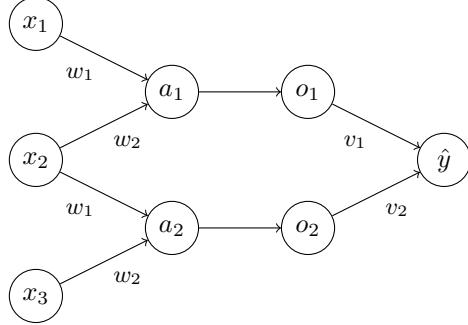
- (A)  $2 \times 2 \times 3$
- (B)  $5 \times 5 \times 4$
- (C)  $2 \times 2 \times 1$
- (D)  $2 \times 2 \times 4$

Ans: D.

(2 points)

## 5.2 Backpropagation for CNN (14 points)

Consider the following mini convolutional neural net, where  $(x_1, x_2, x_3)$  is the input, followed by a convolution layer with a filter  $(w_1, w_2)$ , a ReLU layer, and a fully connected layer with weight  $(v_1, v_2)$ .



More concretely, the computation is specified by

$$\begin{aligned} a_1 &= x_1 w_1 + x_2 w_2 \\ a_2 &= x_2 w_1 + x_3 w_2 \\ o_1 &= \max\{0, a_1\} \\ o_2 &= \max\{0, a_2\} \\ \hat{y} &= o_1 v_1 + o_2 v_2 \end{aligned}$$

For an example  $(\mathbf{x}, y) \in \mathbb{R}^3 \times \{-1, +1\}$ , we define the logistic loss of the CNN as

$$\ell = \ln(1 + \exp(-y\hat{y}))$$

which is a function of the parameters of the network:  $w_1, w_2, v_1, v_2$ .

- (a) Write down  $\frac{\partial \ell}{\partial v_1}$  and  $\frac{\partial \ell}{\partial v_2}$ . You can use the sigmoid function  $\sigma(z) = \frac{1}{1+e^{-z}}$  to simplify your notation.

$$\begin{aligned} \frac{\partial \ell}{\partial v_1} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial v_1} && (1 \text{ point}) \\ &= \frac{-ye^{-y\hat{y}}}{1+e^{-y\hat{y}}} o_1 = -\sigma(-y\hat{y})yo_1 = (\sigma(y\hat{y}) - 1)yo_1 && (1 \text{ point}) \end{aligned}$$

$$\begin{aligned} \frac{\partial \ell}{\partial v_2} &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial v_2} && (1 \text{ point}) \\ &= \frac{-ye^{-y\hat{y}}}{1+e^{-y\hat{y}}} o_2 = -\sigma(-y\hat{y})yo_2 = (\sigma(y\hat{y}) - 1)yo_2 && (1 \text{ point}) \end{aligned}$$

Either one of the last three expressions is acceptable.

- (b) Write down  $\frac{\partial \ell}{\partial w_1}$  and  $\frac{\partial \ell}{\partial w_2}$ . The derivative of the ReLU function is  $H(a) = \mathbb{I}[a > 0]$ , which you can use directly in your answer.

$$\begin{aligned}\frac{\partial \ell}{\partial w_1} &= \frac{\partial \ell}{\partial a_1} \frac{\partial a_1}{\partial w_1} + \frac{\partial \ell}{\partial a_2} \frac{\partial a_2}{\partial w_1} && (1 \text{ point}) \\ &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_1} \frac{\partial o_1}{\partial a_1} \frac{\partial a_1}{\partial w_1} + \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_2} \frac{\partial o_2}{\partial a_2} \frac{\partial a_2}{\partial w_1} = (\sigma(y\hat{y}) - 1)y(v_1 H(a_1)x_1 + v_2 H(a_2)x_2) && (1 \text{ point})\end{aligned}$$

Similarly

$$\begin{aligned}\frac{\partial \ell}{\partial w_2} &= \frac{\partial \ell}{\partial a_1} \frac{\partial a_1}{\partial w_2} + \frac{\partial \ell}{\partial a_2} \frac{\partial a_2}{\partial w_2} && (1 \text{ point}) \\ &= \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_1} \frac{\partial o_1}{\partial a_1} \frac{\partial a_1}{\partial w_2} + \frac{\partial \ell}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_2} \frac{\partial o_2}{\partial a_2} \frac{\partial a_2}{\partial w_2} = (\sigma(y\hat{y}) - 1)y(v_1 H(a_1)x_2 + v_2 H(a_2)x_3). && (1 \text{ point})\end{aligned}$$

Again, other equivalent expressions are acceptable.

- (c) Using the derivations above, fill out the missing details for the Backpropagation algorithm below that is used to train this mini CNN.

---

**Algorithm 5:** Backpropagation for the above mini CNN

---

```

1 Input: A training set  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$ , learning rate  $\eta$ 
2 Initialize:  $w_1 = w_2 = v_1 = v_2 = 0$  (or randomly) (1 point)
3 while not converged do
4   randomly pick an example  $(\mathbf{x}_n, y_n)$ 
5   Forward propagation: compute (2 points)
      
$$a_1 = x_1 w_1 + x_2 w_2, a_2 = x_2 w_1 + x_3 w_2$$

      
$$o_1 = \max\{0, a_1\}, o_2 = \max\{0, a_2\}, \hat{y} = o_1 v_1 + o_2 v_2$$

6   Backward propagation: update (3 points)
      
$$w_1 \leftarrow w_1 - \eta(\sigma(y\hat{y}) - 1)y(v_1 H(a_1)x_1 + v_2 H(a_2)x_2)$$

      
$$w_2 \leftarrow w_2 - \eta(\sigma(y\hat{y}) - 1)y(v_1 H(a_1)x_2 + v_2 H(a_2)x_3)$$

      
$$v_1 \leftarrow v_1 - \eta(\sigma(y\hat{y}) - 1)y o_1$$

      
$$v_2 \leftarrow v_2 - (\sigma(y\hat{y}) - 1)y o_2$$


```

---

Deduct 2 points if updating  $w_1/w_2$  with the updated value of  $v_1/v_2$ .

Do not deduct points for using the wrong gradients due to mistakes from previous two questions.

## 6 Kernel Methods (12 points)

### 6.1 Multiple Choice (6 points)

(a) Which of the following is true about kernel function?

- (A) If  $k_1$  and  $k_2$  are kernel, then  $c_1k_1 + c_2k_2$  is a kernel too for any  $c_1, c_2 \in \mathbb{R}$ .
- (B) Kernel function must be symmetric, that is,  $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$ .
- (C) If  $k$  is a kernel, then  $-k$  is a kernel too.
- (D) If  $k$  is a kernel, then  $\ln k$  is a kernel too.

Ans: B.

(2 points)

(b) Which of the following is not a kernel function?

- |   |  |
|---|--|
| (A) $k(x, x') = (\mathbf{x}^T \mathbf{x}')^2$     | (C) $k(x, x') = -\ \mathbf{x} - \mathbf{x}'\ _2^2$       |
| (B) $k(x, x') = (\mathbf{x}^T \mathbf{x}' + 1)^2$ | (D) $k(x, x') = \exp(-\ \mathbf{x} - \mathbf{x}'\ _2^2)$ |

Ans: C.

(2 points)

(c) Vovk's real polynomial kernel  $k : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$  is defined as:

$$k(\mathbf{x}, \mathbf{x}') = \frac{1 - (\mathbf{x}^T \mathbf{x}')^p}{1 - (\mathbf{x}^T \mathbf{x}')}}$$

where  $p$  is a non-negative integer. Which of the following is the corresponding non-linear mapping for this kernel when  $D = 2$  and  $p = 3$ ?

- |  |   |
|--|---|
| (A) $\phi(\mathbf{x}) = [x_1^2 x_2^2, 2x_1 x_2, x_1, x_2, 1]^T$        | (C) $\phi(\mathbf{x}) = [x_1^2, x_2^2, \sqrt{2}x_1 x_2, x_1, x_2, 1]^T$ |
| (B) $\phi(\mathbf{x}) = [x_1^2 x_2^2, \sqrt{2}x_1 x_2, x_1, x_2, 1]^T$ | (D) $\phi(\mathbf{x}) = [x_1^2, x_2^2, 2x_1 x_2, x_1, x_2, 1]^T$        |

Ans: C.

(2 points)

$$\begin{aligned} \frac{1 - (\mathbf{x}^T \mathbf{x}')^3}{1 - (\mathbf{x}^T \mathbf{x}')} &= \frac{(1 - (\mathbf{x}^T \mathbf{x}'))(1 + \mathbf{x}^T \mathbf{x}' + (\mathbf{x}^T \mathbf{x}')^2)}{1 - (\mathbf{x}^T \mathbf{x}')} \\ &= x_1 x'_1 + x_2 x'_2 + (x_1 x'_1 + x_2 x'_2)^2 + 1 \\ &= x_1^2 x'^2_1 + x_2^2 x'^2_2 + 2x_1 x'_1 x_2 x'_2 + x_1 x'_1 + x_2 x'_2 + 1 \end{aligned}$$

Thus,  $\phi(\mathbf{x}) = [x_1^2, x_2^2, \sqrt{2}x_1 x_2, x_1, x_2, 1]^T$ .

## 6.2 Kernel Composition (6 points)

Prove that if  $k_1, k_2 : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$  are both kernel functions, then  $k(\mathbf{x}, \mathbf{x}') = k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}')$  is a kernel function too. Specifically, suppose  $\phi_1$  and  $\phi_2$  are the corresponding mappings for  $k_1$  and  $k_2$  respectively. Construct the mapping  $\phi$  that certifies  $k$  being a kernel function.

Observe that

$$\begin{aligned} k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}') = \left( \sum_{i=1}^D \phi_1(\mathbf{x})_i \phi_1(\mathbf{x}')_i \right) \left( \sum_{j=1}^D \phi_2(\mathbf{x})_j \phi_2(\mathbf{x}')_j \right) \quad (2 \text{ points}) \\ &= \sum_{i=1}^D \sum_{j=1}^D \phi_1(\mathbf{x})_i \phi_1(\mathbf{x}')_i \phi_2(\mathbf{x})_j \phi_2(\mathbf{x}')_j = \sum_{i=1}^D \sum_{j=1}^D (\phi_1(\mathbf{x})_i \phi_2(\mathbf{x})_j) (\phi_1(\mathbf{x}')_i \phi_2(\mathbf{x}')_j) \quad (2 \text{ points}) \end{aligned}$$

Therefore,  $\phi(\mathbf{x}) = \phi_1(\mathbf{x})\phi_2(\mathbf{x})^\top$ . Writing it as a  $D^2$ -dimensional vector is acceptable too. (2 points)

This blank page can be used as scratch paper.

This blank page can be used as scratch paper.

13<sup>th</sup> October, 2020

DSCI - 552 Midterm

Submission by -

ABHINAV SINGH

3040418309

Ans-1) a)  $b_0 = 964.8$

This is the intercept of the linear model and it indicates the value of  $\hat{y}$  which is not interpreted by any of the dependant variables. Value of  $\hat{y}$  when all dependant variables are 0.

$$b_5 = 17.1$$

This indicates the amount of increase in funding if the founder has a previous failure. For example if the founder has a previous failure as compared to a founder who does not have a previous failure, considering all other feature values to be same. The founder with failure will have  $17.1 \times 1000$  dollars more funding.

b)  $H_0 : b_3 = 0$  (Null hypothesis)

$$H_a : b_3 \neq 0$$

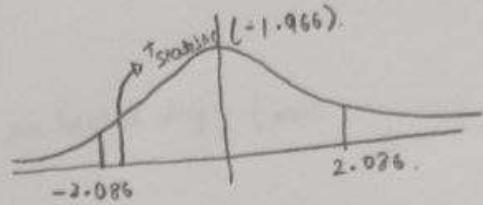
$$t_{\text{statistic}} = \frac{\hat{\beta}_3 - \beta}{SE(\hat{\beta}_3)} = \frac{b_3 - 0}{SE(b_3)} = \frac{-200.2}{101.8} = -1.966$$

$$t_{n-p-1, \alpha/2} = t_{26-5-1, 0.025} \quad \alpha = 5\% = 0.05$$

$$t_{20, 0.025} = 2.086 \quad [\text{From t-table}]$$

Now as we can see that

$$t_{\text{statistic}} = -1.966$$



which does not lie in the rejection region.

$\Rightarrow$  We cannot reject the null hypothesis.  $\beta_3$  is statistically insignificant

c) 80% confidence interval of  $\beta_1$

$$\text{CI} = \left[ \beta_1 \pm t_{n-p-1, \frac{\alpha}{2}} \text{SE}(\beta_1) \right]$$

$$n-p-1 = 20-5-1 = 20$$

$$\alpha = 0.2$$

$$\frac{\alpha}{2} = 0.1$$

$$t_{20, 0.1} = 1.325 \quad [\text{from t-table}]$$

$$\text{CI} = 700.2 \pm 1.325 \times 12$$

$$= 700.2 \pm 15.9$$

$$= [700.2 - 15.9, 700.2 + 15.9]$$

$$= [684.3, 716.1] \rightarrow 80\% \text{ confidence interval}$$

This value indicates that 80% of times the value of  $b_1$  comes within the range  $[684.3, 716.1]$

d)

$$\text{RegSS} = 18147.5$$

$$\text{RSS} = 17136.5$$

Overall significance of the model can be tested by first

$$\begin{aligned} f_{\text{statistic}} &= \frac{\text{RegSS}/P}{\text{RSS}/(n-p-1)} = \frac{18147.5/5}{17136.5/20} \\ &= \frac{18147.5}{8} \times \frac{20}{17136.5} \\ &= 4.235 \end{aligned}$$

$$f_{P, n-p-1, \alpha} = f_{5, 20, 0.01} \quad \alpha = \frac{1}{100}$$

$$f_{5, 20, 0.01} = 4.103$$

$$\text{As we see } f_{\text{statistic}} > f_{P, n-p-1, \alpha}$$

$$[4.23 > 4.103]$$

We can reject the null hypothesis.

$$\textcircled{v} \quad R^2 = 1 - \frac{\text{RSS}}{\text{RSS} + \text{RegSS}} = 1 - \frac{17136.5}{35284} = 1 - 0.485 = 0.515$$

Ans - 2) a)

$$p(x) = \frac{\prod_k f_k}{\sum_{k=1}^K \prod_k f_k}$$

For class k=1

$$p_1(x) = \frac{\pi_1 \frac{x}{\sigma_1^2} e^{-\frac{x^2}{2\sigma_1^2}}}{\pi_1 \frac{x}{\sigma_1^2} e^{-\frac{x^2}{2\sigma_1^2}} + \pi_2 \frac{1}{x\sqrt{2\pi}\sigma_2} e^{-\frac{(ln x - u_2)^2}{2\sigma_2^2}}}$$

Similarly for class k=2

$$p_2(x) = \frac{\pi_2 \frac{x}{\sigma_2^2} e^{-\frac{x^2}{2\sigma_2^2}}}{\pi_1 \frac{x}{\sigma_1^2} e^{-\frac{x^2}{2\sigma_1^2}} + \pi_2 \frac{1}{x\sqrt{2\pi}\sigma_2} e^{-\frac{(ln x - u_2)^2}{2\sigma_2^2}}}$$

To get  $\delta_1(x)$  &  $\delta_2(x)$ . Maximise just numerator as denominator is same.

for  $\delta_2(x)$  taking log

$$= \log \pi_2 - \log \left( \frac{x}{\sqrt{2\pi}\sigma_2} \right) - \frac{(\log x - u_2)^2}{2\sigma_2^2}$$

$\delta_2(x)$  can be linear in x if we consider means to be constant for both the classes.

$$\Rightarrow [u_1 = u_2] \rightarrow \text{linear in } x$$

b)  $p_1(x) = \frac{\pi_1 f_1(x)}{\pi_1 f_1(x) + \pi_2 f_2(x)}$        $p_2(x) = \frac{\pi_2 f_2(x)}{\pi_1 f_1(x) + \pi_2 f_2(x)}$

to classify  $x=10$ , we need to see the probability of both classes, and classify x to the class with greater probability. We can just compare the numerators of both to get the answer as denominator is same.

$$\pi_{1, f_1(10)} = 0.5 \times \frac{10}{1^2} e^{-\frac{10^2}{2}}$$

$$= 0.5 \times 10 e^{-50}$$

$$= 5 e^{-50}$$

$$\pi_{2, f_2(10)} = \frac{0.5 \times 1}{10 \sqrt{2\pi}} e^{\frac{-(\ln 10 - 10)^2}{2}}$$

$$= \frac{0.5}{10 \times 4.44} e^{-\frac{(2.3 - 10)^2}{2}}$$

$$= \frac{0.5}{10 \times 4.44} e^{-\frac{59.29}{2}}$$

$$= \frac{0.5}{44.4} e^{-29.64}$$

As we can see  $\pi_{2, f_2(10)}$  is much greater than  $\pi_{1, f_1(10)}$

we classify  $x=10$  to class 2.

- Ans - 3.
- a) The best subset approach will have the smallest training RSS for k predictors.

This is because, best subset model would have analysed all possible subsets of size k, which includes the solution generated by forward selection and backward selection and it chose the best model by evaluating the smallest RSS for k predictors.

Hence the best subset approach will give smallest training RSS.

- b) We cannot say which model will give smallest test RSS.  
We need to perform cross validation to test estimate the test error.  
It is possible that a model which performs amazingly well on train set performs very bad on test set as the train set was not able to model the ground truth.  
Hence we cannot say which model with k predictors will have smallest test RSS [No free lunch theorem]

Ans - 3) c) i) True

ii) True

iii) False

iv) False

v) False

Ans - 4)

a) Decision boundary of the classifier will come at

$$P(Y=1|X) = P(Y=0|X) = 0.5$$

$$\Rightarrow \frac{e^{\beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2}}{1 + e^{\beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2}} = \frac{1}{2}$$

Putting in the coefficient values we get

$$\Rightarrow 2 \times e^{1+2x_1+x_1^2+x_2} = 1 + e^{1+2x_1+x_1^2+x_2}$$

$$\Rightarrow e^{1+2x_1+x_1^2+x_2} = 1$$

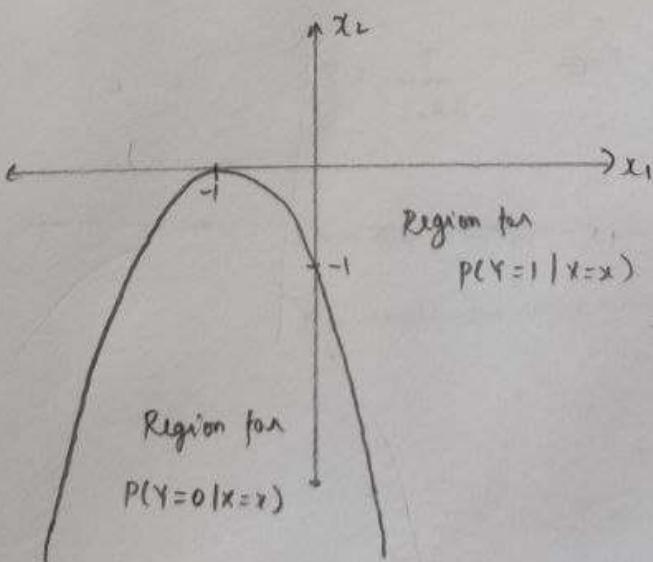
$$\Rightarrow x_1^2 + 2x_1 + 1 + x_2 = 0$$

$$\Rightarrow (x_1 + 1)^2 + x_2 = 0$$

$$\Rightarrow x_2 = -(x_1 + 1)^2 \rightarrow \text{Equation for decision boundary of classifier based on } x_1, x_2.$$

b) The decision boundary of this classifier is a parabola

$$x_1=0, x_2=0$$



$$\begin{aligned}
 P(Y=1|X=x) &= \frac{e^1}{1+e^1} \\
 &= \frac{2.71}{1+2.71} = \frac{2.71}{3.71} \\
 &= 0.73 \\
 P(Y=0|X=x) &= 0.27
 \end{aligned}$$

Everything inside the parabola = -ve class  
Everything outside the parabola = +ve class

c) To make coefficient of  $\beta_2$  statistically insignificant

$$H_0 \Rightarrow \beta_2 = 0 \quad (\text{Null hypothesis})$$

$$H_a \Rightarrow \beta_2 \neq 0 \quad (\text{Alternate hypothesis})$$

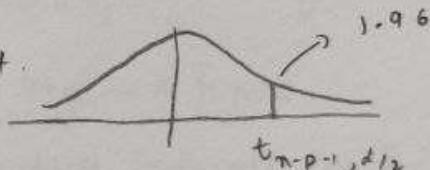
$$t_{\text{statistic}} = \frac{\hat{\beta}_2 - 0}{SE(\beta_2)} = \frac{1 - 0}{S} = \frac{1}{S}$$

$$t_{n-p-1, \alpha/2} = t_{200-3-1, 0.025} = t_{196, 0.025}$$

$$\alpha = 0.05$$

$$\Rightarrow t_{196, 0.025} = 1.96 \quad [\text{from t table}]$$

For  $\beta_2$  to be statistically insignificant  
We should not reject the null hypothesis



$$\Rightarrow t_{\text{statistic}} < t_{n-p-1, \alpha/2}$$

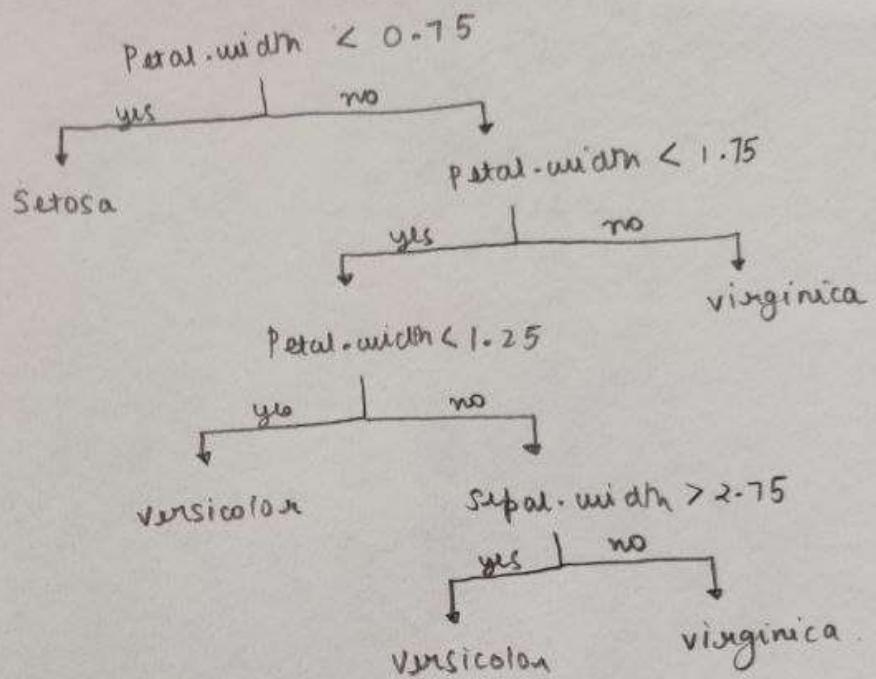
$$\Rightarrow \frac{1}{S} < 1.96$$

$$\Rightarrow S > \frac{1}{1.96} \Rightarrow S > 0.5102$$

For all values of  $S$  greater than 0.5102,  
 $\beta_2$  will be statistically significant.

Ans-5)

Decision tree →



This is the desired decision tree for the given figure.

We have 5 terminal / leaf nodes which indicate the 5 regions in the given figure.

We have 4 internal nodes in the tree which are the 4 levels of decisions we made in the tree.

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.
- The exam has 5 questions, 9 pages, and 20 points extra credit. However, your grade cannot exceed 100/100.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 30    |        |
| 3       | 25    |        |
| 4       | 20    |        |
| 5       | 20    |        |
| Total   | 120   |        |

1. For each Major League Baseball team we have the number of wins (`Wins`) and the total player salary in millions of dollars (`Salary`) for 2006. (You don't need to know anything about baseball for this question.) The total league payroll was \$2,326.707 million. For each team  $i$ , define

$$\text{SalaryShare}_i = \frac{\text{Salary}_i}{\sum_{j=1}^n \text{Salary}_j} = \frac{\text{Salary}_i}{2,326.707}$$

Now consider the following summary.

Call:

`lm(formula = Wins ~ SalaryShare)`  $\hat{=}$   $y = \text{SalaryShare} \gamma \text{Wins}$

Residuals:

|  | Min      | 1Q      | Median | 3Q     | Max     |
|--|----------|---------|--------|--------|---------|
|  | -17.7907 | -4.5503 | 0.3654 | 4.5352 | 17.4042 |

Coefficients:

|   | Estimate | Std. Error | t value | Pr(> t )    |
|---|----------|------------|---------|-------------|
| (Intercept)   | 67.982   | 4.178      | 16.271  | 8.4e-16 *** |
| SalaryShare   | 389.540  | 116.013    | 3.358   | 0.00228 **  |
| <hr/>   |          |            |         |             |
| Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 |          |            |         |             |

Residual standard error: 8.665 on 28 degrees of freedom

Multiple R-squared: 0.2871, Adjusted R-squared: 0.2616

F-statistic: 11.27 on 1 and 28 DF, p-value: 0.002277

- (a) But suppose that instead of regressing `Wins` on `SalaryShare` we used `Salary` itself as the input. Use the summary above to compute the estimates of the intercept  $\hat{\beta}_0$ , the slope  $\hat{\beta}_1$ , and the  $R^2$  value for this hypothetical regression.
- (b) Do we have reason to believe in a linear relationship between `Wins` and `Salary`, in the hypothetical regression in part 1a? State a formal hypothesis test, the value of the test statistic, and the conclusion. Use  $\alpha = 0.05$

Salary Share

$$\beta_0 = 67.982$$

$$\beta_1 = 389.540$$

$$y_i = \beta_0 + \frac{\beta_1 * \text{Salary}}{\sum \text{Salary}}$$

Salary

$$\hat{\beta}_0 =$$

$$\hat{\beta}_1 = \underline{\beta_1}$$

$\sum \text{Salary}$

=

∴ Since we are scaling salary,

$$\beta_0 \text{ is same, } \hat{\beta}_1 = \frac{\beta_1}{\sum \text{Sal}}, R^2 \text{ will be same}$$

Therefore, if we regress Wins on just Salary, we will get the same  $b_0$ . The new slope estimate will be  $= b_1 / 2,326.707 = 0.167$ . The residuals will also be the same, and hence the  $R^2$  will be the same. Intuitively, we are just changing the scale that we measure salary. It's exactly the same as changing prices from dollars to thousands of dollars: here instead of dividing by \$1,000, we are dividing by \$2,326.707 million, but it's the same idea. Changing the scale doesn't change how two variables are associated in the data.

b)

$$t = 3.358 \quad \alpha = 0.05$$

$$t_{n-p-1, \alpha/2}$$

$n-p-1$  = degree of freedom

$$t_{28, 0.025} = 2.048$$

$$\therefore t > t_{28, 0.025}$$

Under the null  $t = 2.358$  is  
greater than  $t_{28, 0.025} > 3.358$ .

$\therefore$  we can say that Wins & Salary  
have linear relation

2. In a classification problem with two classes and two features, the joint distribution of the features in each class is:

$$f_k(x_1, x_2) = \frac{1}{2\pi\sqrt{(1-k/4)}} \exp\left[-\frac{z}{2(1-k/4)}\right], \quad k = 1, 2$$

where

$$z = (x_1 - k)^2 - \sqrt{k}(x_1 - k)(x_2 - k^2) + (x_2 - k^2)^2$$

- (a) Assuming that the prior probability of class one is twice the prior probability of class two, in what class is the point  $(X_1, X_2) = (1, 5)$  is classified?
- (b) The marginal distributions of features in each class can be calculated from the joint distributions, and are:

$$f_k(x_1) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{(x_1 - k)^2}{2}\right] \quad f_k(x_2) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{(x_2 - k^2)^2}{2}\right]$$

The Naïve Bayes assumption clearly does not hold in this problem. However, classify  $(X_1, X_2) = (1, 5)$  pretending the Naïve Bayes assumption holds and compare the results with part 2a.

$$z = (x_1 - k)^2 - \sqrt{k} (x_1 - k) (x_2 - k^2) + (x_2 - k^2)^2$$

$$z_1(1, 5) = (1 - 1)^2 - \sqrt{1} (1 - 1) (5 - 1^2) + (5 - 1)^2$$

$$z_1 = 16$$

$$z_2(1, 5) = (1 - 2)^2 - \sqrt{2} (1 - 2) (5 - 4) + (5 - 4)^2$$

$$= 1 + \sqrt{2} + 1 = 2 + \sqrt{2}$$

$$= 3.414$$

$$f_1(1,5) = \frac{1}{2\pi \sqrt{1-\frac{1}{4}}} e^{-\frac{1}{6}} \approx 0.28 e(-6)$$

$$= \frac{1}{2\pi \sqrt{\frac{3}{4}}} e^{-\frac{3}{4}} = 0.28 e(-6)$$

$$\bar{\pi}_1 = 2 \bar{\pi}_2$$

$$\bar{\pi}_1 + \bar{\pi}_2 = 1$$

$$2\bar{\pi}_2 + \bar{\pi}_2 = 1$$

$$\bar{\pi}_2 = 0.33 \quad \bar{\pi}_1 = 0.66$$

$$\delta_1(x(1,5)) = 0.66 \times 0.28 \times e(-6)$$

$$\delta_2(x(1,5)) = 0.33 \times 0.04 e(-3)$$

$$f_1(x_1), f_2(x_2), f_1(x_2), f_2(x_2)$$

$$\delta_1(x_1, x_2) = \pi_1 \cdot f_1(x_1) \cdot f_1(x_2)$$

$$\delta_2(x_1, x_2) = \pi_2 \cdot f_2(x_1) \cdot f_2(x_2)$$

3. Consider a logistic regression problem in which there are no features, which means that:

$$\Pr(Y = 1) = \frac{e^{\beta_0}}{1 + e^{\beta_0}}$$

Assume that we have  $m$  data points with label  $Y = 1$  and  $n$  data points with label  $Y = 0$  (remember that features are irrelevant).

- (a) Write down the likelihood function  $l(\beta_0)$ .
- (b) Find the Maximum Likelihood estimate  $\hat{\beta}_0$  for this data set. [Hint: maximize  $\log_e l(\beta_0)$ ].
- (c) Determine conditions under which this simple classifier classifies data points into  $Y = 1$  or  $Y = 0$ .

a)

$$l(\beta_0) = \prod_{i=1}^{m+n} [P(x_i)]^{y_i} [1 - P(x_i)]^{1-y_i}$$

$$l(\beta_0) = \prod_{i=1}^{m+n} \left[ \frac{e^{\beta_0}}{1 + e^{\beta_0}} \right]^{y_i} \left[ \frac{1}{1 + e^{\beta_0}} \right]^{1-y_i}$$

$$b) \log_e l(\beta_0) \rightarrow \log \left( \prod_{i=1}^{m+n} \left[ \frac{e^{\beta_0}}{1 + e^{\beta_0}} \right]^{y_i} \left[ \frac{1}{1 + e^{\beta_0}} \right]^{1-y_i} \right)$$

$$= \sum_{i=1}^{m+n} y_i \left[ \log e^{\beta_0} - \log (1 + e^{\beta_0}) \right] \\ + (1 - y_i) \left[ \log (1) - \log (1 + e^{\beta_0}) \right]$$

$$= \sum_{i=1}^{m+n} y_i \cdot \beta_0 - y_i \log(1+e^{\beta_0})$$

$$= \log(1+e^{\beta_0}) + y_i \log(1+e^{\beta_0})$$

$$= \sum_{j=1}^{m+n} y_j \cdot \beta_0 - \log(1+e^{\beta_0})$$

c)



$$P_{\gamma}(Y=1) = \frac{c}{1+e^{\beta_0}}$$

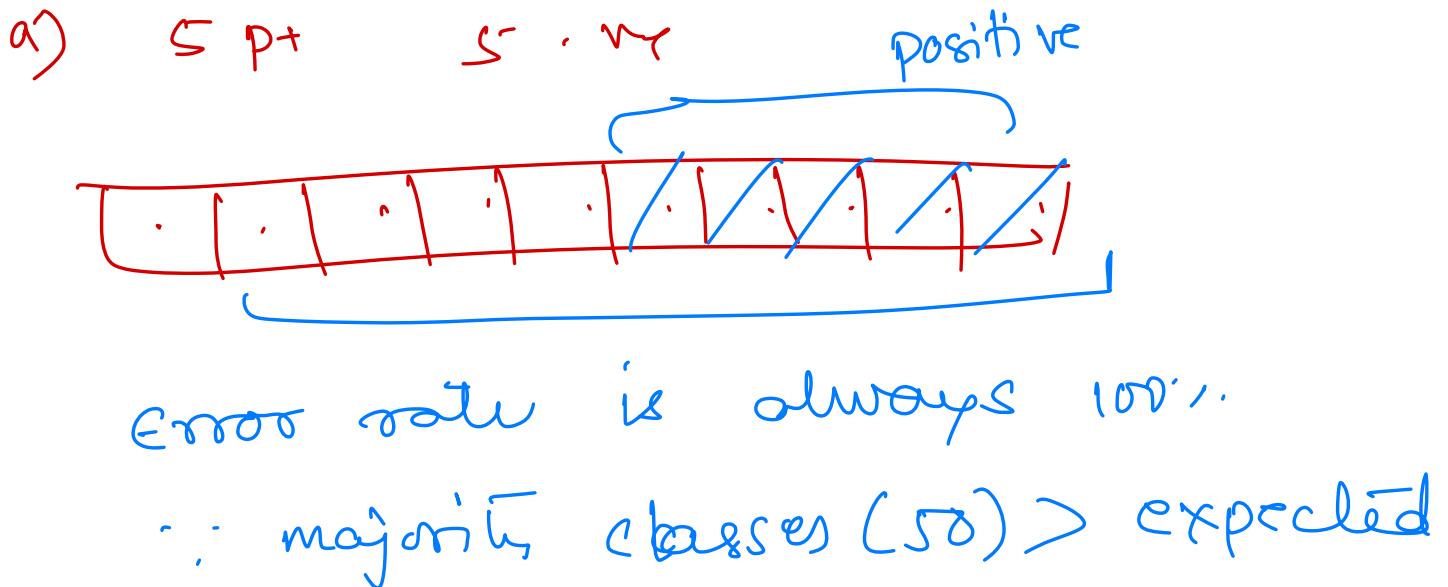
$$P_{\gamma}(Y=0) = 1 - \frac{e^{\beta_0}}{1+e^{\beta_0}}$$

if  $\frac{c}{1+e^{\beta_0}} > \frac{1}{1+e^{\beta_0}}$  then its class  $\stackrel{?}{=} \frac{1}{1+e^{\beta_0}}$

$$\Rightarrow c^{\beta_0} > 1 \Rightarrow \beta_0 > 0$$

4. Let us consider a data set containing 50 positive and 50 negative instances, where the attributes contain no information about the class labels. Hence, the generalization error rate of any classification model learned over this data is expected to be 0.5. Let us consider a classifier that assigns the majority class label of training instances (ties resolved by using the positive label as the default class) to any test instance, irrespective of its attribute values. We can call this approach *the majority inducer* classifier. Determine the error rate of this classifier using the following methods.

- (a) Leave-one-out cross validation.
- (b) 2-fold stratified cross-validation, where the proportion of class labels at every fold is kept same as that of the overall data.
- (c) From the results above, which method provides a more reliable evaluation of the classifier's generalization error rate?



**Answer:** Let us represent our data set as  $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{100}$ , where  $\mathbf{x}_i$  is the  $i^{\text{th}}$  data instance. We know  $\mathcal{D}$  has 50 positives and 50 negatives. In the leave-one-out method, the test error on every instance  $\mathbf{x}_i$  is computed by applying a classification model trained on all data instances in  $\mathcal{D}$  excluding  $\mathbf{x}_i$ , denoted as  $\mathcal{D}_{-i} = \mathcal{D} \setminus \{\mathbf{x}_i\}$ . If we consider the case where  $\mathbf{x}_i$  is positive, then  $\mathcal{D}_{-i}$  will end up with one less positive than  $\mathcal{D}$  (containing 49 positives and 50 negatives). The majority inducer classifier will thus assign  $\mathbf{x}_i$  to the majority class, which is negative, and thus incur an error. On the other hand, if we consider  $\mathbf{x}_i$  to be negative, then  $\mathcal{D}_{-i}$  will contain 49 negatives and 50 positives, and the majority inducer will incorrectly assign  $\mathbf{x}_i$  to be positive (the majority class). Hence, the majority inducer would make an error on every data instance using leave-one-out, and thus have an error rate of 1.

b) In 2 fold stratified cross validation we divide the data set into 2 halves where each halve will have 25 +ve & 25 -ve.

when we validate using validation data, +ve & -ve classes will be same

$$\therefore \text{error} = 0.5$$

c)

**Answer:** Cross-validation provides a more reliable estimate of the generalization error rate of the majority inducer classifier on this data set, which is expected to be 0.5. Leave-one-out is quite susceptible to changes in the number of positive and negative instances in the training set, even by a single count, leading to a high error rate of 1 for the majority inducer. As another example, if we consider the minority inducer classifier, which labels every test instance with the minority class in the training set, we would find that the leave-one-out method would result in an error rate of 0 for the minority inducer, which is quite misleading since the attributes contain no information about the classes and any classifier is expected to have an error rate of 0.5.

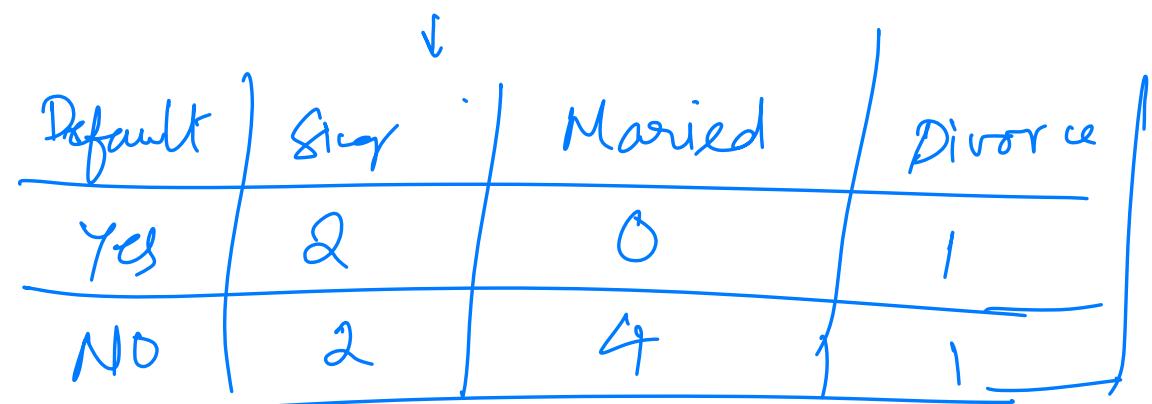
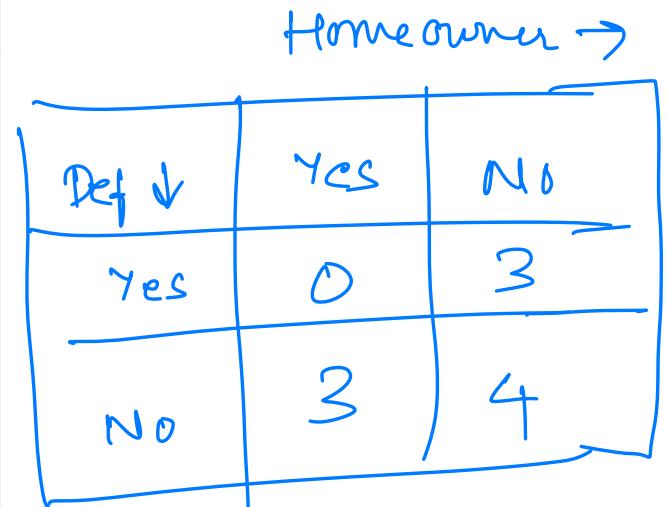
5. Consider the dataset presented in the following table for classification of loan defaults. Given a pair of categorical attribute values,  $V_1$  and  $V_2$ , the distance between them is defined as follows:

$$d_M(V_1, V_2) = \sum_{i=1}^k \left| \frac{n_{i1}}{n_1} - \frac{n_{i2}}{n_2} \right|$$

where  $n_{ij}$  is the number of examples from class  $i$  with attribute value  $V_j$  and  $n_j$  is the number of examples with attribute value  $V_j$ .

The distance between a test point  $X^* = (HO^*, MS^*, An^*)$  and training point  $X = (HO, MS, An)$  is defined as  $d(X^*, X) = d_M(HO^*, HO) + d_M(MS^*, MS) + |An^* - An|$ , where  $HO, MS, An$  respectively stand for Home Owner, Marital Status, and Annual Income in \$1K. How is the test point (*Yes, Single, 110K*) classified using 3-, 5-, 7-nearest neighbors and  $d(X^*, X)$ ?

| Tid | Home Owner | Marital Status | Annual Income | Defaulted Borrower |
|-----|------------|----------------|---------------|--------------------|
| 1   | Yes        | Single         | 125K          | No                 |
| 2   | No         | Married        | 100K          | No                 |
| 3   | No         | Single         | 70K           | No                 |
| 4   | Yes        | Married        | 120K          | No                 |
| 5   | No         | Divorced       | 95K           | Yes                |
| 6   | No         | Married        | 60K           | No                 |
| 7   | Yes        | Divorced       | 220K          | No                 |
| 8   | No         | Single         | 85K           | Yes                |
| 9   | No         | Married        | 75K           | No                 |
| 10  | No         | Single         | 90K           | Yes                |



Scratch paper

Name:

USC ID:

$$d_m(v_1, v_2) = \sum_{i=1}^k \left| \frac{n_{i1}}{n_1} - \frac{n_{i2}}{n_2} \right|$$

$$d(\text{sing, man}) = \left[ \frac{2}{4} - \frac{0}{4} \right] + \left[ \frac{2}{4} - \frac{4}{4} \right] = \frac{2}{4} + \frac{2}{4} = 1$$

$$d(\text{sing, Div}) = \left[ \frac{2}{4} - \frac{1}{2} \right] + \left[ \frac{2}{4} - \frac{1}{2} \right] = 0$$

$$d(\text{man, Div}) = \left[ \frac{0}{4} - \frac{1}{2} \right] + \left[ \frac{4}{4} - \frac{1}{2} \right]$$

$$= +\frac{1}{2} + \frac{1}{2} = 1$$

Midterm Exam

DSCI 552, Instructor: Mohammad Reza Rajati

Oct 12, 2021

Scratch paper

Name:

USC ID:

$$\begin{aligned} d(4, N) &= \left| \frac{0}{3} - \frac{3}{7} \right| + \left| \frac{3}{3} - \frac{4}{7} \right| \\ &= \frac{3}{7} + \frac{3}{7} = \frac{6}{7} \end{aligned}$$

(Yes, single, 1101c)

$$d_1 = 0+0+15K = 15K.$$

$$d_2 = 0+1+10K = 10002$$

$$d_3 = 1+0+40K = 40001$$

$$d_4 = 0+1+10K = 10001$$

$$d_5 = 1+0+15K = 15001$$

$$d_6 = 1+1+50K = 50002$$

$$\left. \begin{array}{l} d_7 = 0+0+110K = 110000 \\ d_8 = 1+0+25K = 25001 \\ d_9 = 1+1+35K = 35002 \\ d_{10} = 1+0+20K = 20001 \end{array} \right\}$$

3 Neighbours = d<sub>4</sub>, d<sub>2</sub>, d<sub>1</sub>

= No, No, No

= No

3 Neighbours = d<sub>4</sub>, d<sub>2</sub>, d<sub>1</sub>, d<sub>5</sub>, d<sub>10</sub>

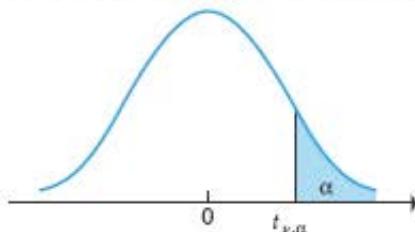
= No, No, No, Yes, Yes

= No

7 Neighbours = d<sub>4</sub>, d<sub>2</sub>, d<sub>1</sub>, d<sub>5</sub>, d<sub>10</sub>, d<sub>8</sub>, d<sub>9</sub>

= N, N, N, Y, Y, Y, N

= No

Upper Critical Values of Student's  $t$  Distribution with  $\nu$  Degrees of Freedom


For selected probabilities,  $\alpha$ , the table shows the values  $t_{\nu,\alpha}$  such that  $P(t_\nu > t_{\nu,\alpha}) = \alpha$ , where  $t_\nu$  is a Student's  $t$  random variable with  $\nu$  degrees of freedom. For example, the probability is .10 that a Student's  $t$  random variable with 10 degrees of freedom exceeds 1.372.

| $\nu$    | PROBABILITY OF EXCEEDING THE CRITICAL VALUE |       |        |        |        |         |
|----------|---|-------|--------|--------|--------|---------|
|          | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |
| 1        | 3.078                                       | 6.314 | 12.706 | 31.821 | 63.657 | 318.313 |
| 2        | 1.886                                       | 2.920 | 4.303  | 6.965  | 9.925  | 22.327  |
| 3        | 1.638                                       | 2.353 | 3.182  | 4.541  | 5.841  | 10.215  |
| 4        | 1.533                                       | 2.132 | 2.776  | 3.747  | 4.604  | 7.173   |
| 5        | 1.476                                       | 2.015 | 2.571  | 3.365  | 4.032  | 5.893   |
| 6        | 1.440                                       | 1.943 | 2.447  | 3.143  | 3.707  | 5.208   |
| 7        | 1.415                                       | 1.895 | 2.365  | 2.998  | 3.499  | 4.782   |
| 8        | 1.397                                       | 1.860 | 2.306  | 2.896  | 3.355  | 4.499   |
| 9        | 1.383                                       | 1.833 | 2.262  | 2.821  | 3.250  | 4.296   |
| 10       | 1.372                                       | 1.812 | 2.228  | 2.764  | 3.169  | 4.143   |
| 11       | 1.363                                       | 1.796 | 2.201  | 2.718  | 3.106  | 4.024   |
| 12       | 1.356                                       | 1.782 | 2.179  | 2.681  | 3.055  | 3.929   |
| 13       | 1.350                                       | 1.771 | 2.160  | 2.650  | 3.012  | 3.852   |
| 14       | 1.345                                       | 1.761 | 2.145  | 2.624  | 2.977  | 3.787   |
| 15       | 1.341                                       | 1.753 | 2.131  | 2.602  | 2.947  | 3.733   |
| 16       | 1.337                                       | 1.746 | 2.120  | 2.583  | 2.921  | 3.686   |
| 17       | 1.333                                       | 1.740 | 2.110  | 2.567  | 2.898  | 3.646   |
| 18       | 1.330                                       | 1.734 | 2.101  | 2.552  | 2.878  | 3.610   |
| 19       | 1.328                                       | 1.729 | 2.093  | 2.539  | 2.861  | 3.579   |
| 20       | 1.325                                       | 1.725 | 2.086  | 2.528  | 2.845  | 3.552   |
| 21       | 1.323                                       | 1.721 | 2.080  | 2.518  | 2.831  | 3.527   |
| 22       | 1.321                                       | 1.717 | 2.074  | 2.508  | 2.819  | 3.505   |
| 23       | 1.319                                       | 1.714 | 2.069  | 2.500  | 2.807  | 3.485   |
| 24       | 1.318                                       | 1.711 | 2.064  | 2.492  | 2.797  | 3.467   |
| 25       | 1.316                                       | 1.708 | 2.060  | 2.485  | 2.787  | 3.450   |
| 26       | 1.315                                       | 1.706 | 2.056  | 2.479  | 2.779  | 3.435   |
| 27       | 1.314                                       | 1.703 | 2.052  | 2.473  | 2.771  | 3.421   |
| 28       | 1.313                                       | 1.701 | 2.048  | 2.467  | 2.763  | 3.408   |
| 29       | 1.311                                       | 1.699 | 2.045  | 2.462  | 2.756  | 3.396   |
| 30       | 1.310                                       | 1.697 | 2.042  | 2.457  | 2.750  | 3.385   |
| 40       | 1.303                                       | 1.684 | 2.021  | 2.423  | 2.704  | 3.307   |
| 60       | 1.296                                       | 1.671 | 2.000  | 2.390  | 2.660  | 3.232   |
| 100      | 1.290                                       | 1.660 | 1.984  | 2.364  | 2.626  | 3.174   |
| $\infty$ | 1.282                                       | 1.645 | 1.960  | 2.326  | 2.576  | 3.090   |
| $\nu$    | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.
- The exam has 5 questions, 9 pages, and 20 points extra credit. However, your grade cannot exceed 100/100.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 30    |        |
| 3       | 25    |        |
| 4       | 20    |        |
| 5       | 20    |        |
| Total   | 120   |        |

1. For the simple regression model

$$Y = \beta_0 + \beta_1 X + \epsilon$$

the  $R^2$  was calculated to be 25%, from a sample size of  $n = 15$ . Assuming  $\epsilon$  is a zero-mean Gaussian, determine whether or not  $\beta_1$  is statistically significant, when  $\alpha = 0.01$ . (Important note: you may feel that the instructor has not provided enough information to solve this problem, but that is not correct. Everything you need to solve this problem is on this exam.)

$$R^2 = 25\% = 0.25$$

$$n = 15$$

$$p = 1$$

$$\alpha = 0.01$$

$$f_{p, n-p-1} = f_{1, 13} = 9.074$$

$$f = \frac{(TSS - RSS) / p}{(RSS) / (n-p-1)} = \frac{\cancel{TSS} - \cancel{RSS}}{\cancel{TSS}}$$

$$= \frac{\left( \frac{TSS - RSS}{TSS} \right) / p}{\left( \frac{RSS}{TSS} \right) / (n-p-1)}$$

$$= \frac{R^2 / p}{(1 - R^2) / n - p - 1}$$

$$= \frac{k_4}{(1 - k_4) \times V_{13}} = \frac{V_A \times B}{B/k_4}$$

$$f = 4.333$$

$f < f_{p, n-p-1} \therefore \text{Insignificant}$

2. In a classification problem with three classes and three features, the distribution of each feature in each class is Gaussian. The mean of feature  $j$  in class  $k$  is  $\mu_{jk} = jk$  and the standard deviation of feature  $j$  in class  $k$  is  $\sigma_{jk} = k$ , when  $j, k \in \{1, 2, 3\}$ . If the Naïve Bayes assumption holds and the classes have equal prior probabilities, in what class the feature vector  $(X_1, X_2, X_3) = (1, 5, 9)$  is classified?

calc  $F = \frac{R^2}{LR^2} \cdot \frac{n-p-1}{P}$

$F_{1,13,0.01} = 9.074$

$R^2 / n > 9.074$   
since  $\frac{13}{3} < 9.074$

$B_1$  is statistically significant

maximum

$\delta_1(1,5,9)$

$$\begin{aligned} &= -\frac{1}{2} \sum_{j=1}^3 \frac{(x_j - \mu_{jk})^2}{\sigma_{jk}^2} + \log \pi_k \\ &= -\frac{1}{2} \left[ \frac{(1-1)^2}{1^2} + \frac{(5-2)^2}{2^2} + \frac{(9-3)^2}{3^2} \right] = -\frac{1}{2} [9+36] \\ &= -\frac{45}{2} \end{aligned}$$

$\delta_2(1,5,9)$

$$\begin{aligned} &= -\frac{1}{2} \left[ \frac{(x_1 - \mu_{12})^2}{\sigma_{12}^2} + \frac{(x_2 - \mu_{22})^2}{\sigma_{22}^2} + \frac{(x_3 - \mu_{32})^2}{\sigma_{32}^2} \right] \\ &= -\frac{1}{2} \left[ \frac{(1-2)^2}{2^2} + \frac{(5-4)^2}{2^2} + \frac{(9-6)^2}{2^2} \right] \\ &= -\frac{1}{2} \left[ \frac{1}{4} + \frac{1}{4} + \frac{9}{4} \right] = -\frac{11}{8} \end{aligned}$$

$\delta_3(1,5,9)$

$$\begin{aligned} &= -\frac{1}{2} \left[ \frac{(x_1 - \mu_{13})^2}{\sigma_{13}^2} + \frac{(x_2 - \mu_{23})^2}{\sigma_{23}^2} + \frac{(x_3 - \mu_{33})^2}{\sigma_{33}^2} \right] \rightarrow \text{Assign to class 3} \\ &= -\frac{1}{2} \left[ \frac{(1-3)^2}{3^2} + \frac{(5-6)^2}{3^2} + \frac{(9-9)^2}{3^2} \right] = -\frac{1}{2} \left[ \frac{4}{9} + \frac{1}{9} \right] = -\frac{5}{18} \end{aligned}$$

3. In an unusual logistic regression problem,

$$\Pr(Y = 1|X_1, X_2) = \frac{X_1^2 X_2^2}{1 + X_1^2 X_2^2}$$

- (a) Write down the model for the odds of  $Y = 1$  given  $X_1$  and  $X_2$ .
- (b) Determine the equation for the decision boundary between classes  $Y = 1$  and  $Y = 0$  and sketch it in the  $X_1, X_2$  plane. Show the regions for each class.

a)

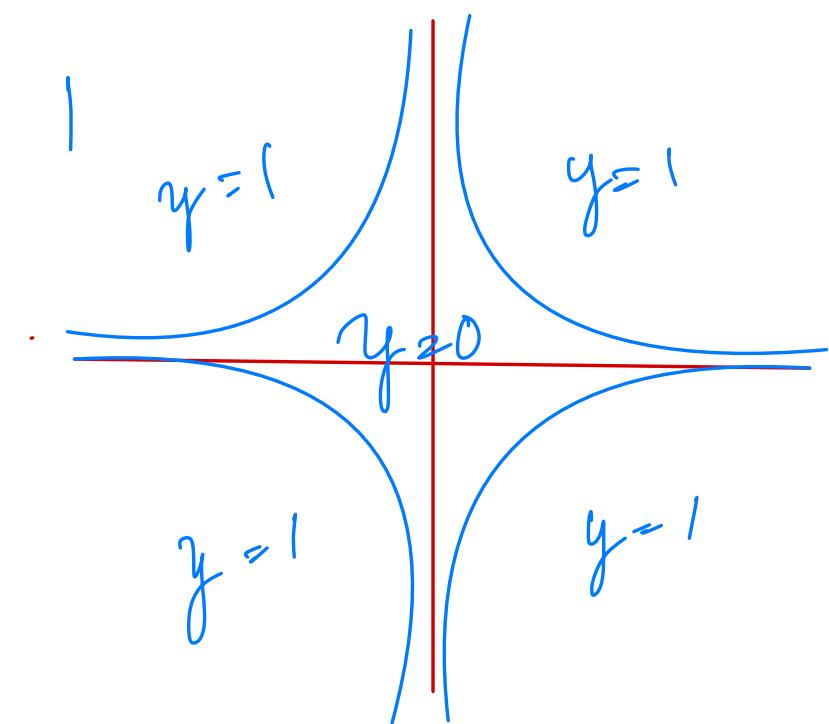
$$\frac{X_1^2 X_2^2}{1 + X_1^2 X_2^2} > \underline{\hspace{10cm}} = X_1^2 X_2^2$$

$$1 - \frac{X_1^2 X_2^2}{1 + X_1^2 X_2^2} = \underline{\hspace{1cm}} = 4.4$$

b)

$$X_1^2 X_2^2 = 1$$

$$X_2^2 = \frac{1}{X_1^2}$$



4. Consider the training set  $\{1, 2, 3\}$ . There are 10 *distinct* bootstrap samples with the same size that can be drawn from this training set. (For example,  $\{1, 2, 2\}$  is not distinct from  $\{2, 2, 1\}$ ). Using those bootstrap samples, build an 80% bootstrap confidence interval for the mean.

$$\begin{array}{rcl}
 1 & 1 & 1 \\
 & = & 3/3 \\
 \\ 
 1 & 1 & 2 \\
 & = & 4/3 \\
 \\ 
 1 & 1 & 3 \\
 & = & 5/3 \\
 \\ 
 \cancel{+} & 2 + & \\
 \\ 
 1 & 2 & 2 \\
 & = & 5/3 \\
 \\ 
 1 & 2 & 3 \\
 & = & 6/3 \\
 \\ 
 2 & 2 & 3 \\
 & = & 7/3 \\
 \\ 
 1 & 3 & 3 \\
 & = & 7/3 \\
 \\ 
 2 & 3 & 3 \\
 & = & 8/3 \\
 \\ 
 2 & 2 & 2 \\
 & = & 6/3 \\
 \\ 
 3 & 3 & 3 \\
 & = & 9/3
 \end{array}
 \quad
 \begin{array}{l}
 111 \\
 112 \\
 113 \\
 122 \\
 123 \\
 222 \\
 223 \\
 232 \\
 233 \\
 333
 \end{array}
 \quad
 \left( \begin{array}{c} 4/3 \\ 7/3 \\ 8/3 \end{array} \right)$$

5. Consider the one-dimensional data set shown below.

- (a) Classify the test data points  $x_1 = 5.0, y_1 = +$  and  $x_2 = 4.0, y_2 = -$  according to their 1-, 3-, 5-, and 9-nearest neighbors (using majority vote). Each point is excluded from its nearest neighbors.
- (b) Which  $k$  or  $k$ 's yield the best test results?

|   |     |     |     |     |     |     |     |     |     |     |
|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| x | 0.5 | 3.0 | 4.5 | 4.6 | 4.9 | 5.2 | 5.3 | 5.5 | 7.0 | 9.5 |
| y | -   | -   | +   | +   | +   | -   | -   | +   | -   | -   |

5.0    4.9    5.2    5.3    4.6    4.5    5.5    3.0    7.0    9.5  
 +       +       -       -       +       +       +       -       -       -

$k=1$     =    +  
 pred

$k=3$     -

$k=5$     +

$k=9$     -

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

## F Table for $\alpha = 0.01$

| DF1 | $\alpha = 0.01$ |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |
|-----|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| DF2 | 1               | 2      | 3      | 4      | 5      | 6      | 7      | 8      | 9      | 10     | 12     | 15     | 20     | 24     | 30     | 40     | 60     | 120    | Inf    |
| 1   | 4052.2          | 4999.5 | 5403.4 | 5624.6 | 5763.7 | 5859   | 5928.4 | 5981.1 | 6022.5 | 6055.8 | 6106.3 | 6157.3 | 6208.7 | 6234.6 | 6260.6 | 6286.8 | 6313   | 6339.4 | 6365.9 |
| 2   | 98.503          | 99     | 99.166 | 99.249 | 99.299 | 99.333 | 99.356 | 99.374 | 99.388 | 99.399 | 99.416 | 99.433 | 99.449 | 99.458 | 99.466 | 99.474 | 99.482 | 99.491 | 99.499 |
| 3   | 34.116          | 30.817 | 29.457 | 28.71  | 28.237 | 27.911 | 27.672 | 27.489 | 27.345 | 27.229 | 27.052 | 26.872 | 26.69  | 26.598 | 26.505 | 26.411 | 26.316 | 26.221 | 26.125 |
| 4   | 21.198          | 18     | 16.694 | 15.977 | 15.522 | 15.207 | 14.976 | 14.799 | 14.659 | 14.546 | 14.374 | 14.198 | 14.02  | 13.929 | 13.838 | 13.745 | 13.652 | 13.558 | 13.463 |
| 5   | 16.258          | 13.274 | 12.06  | 11.392 | 10.967 | 10.672 | 10.456 | 10.289 | 10.158 | 10.051 | 9.888  | 9.722  | 9.553  | 9.466  | 9.379  | 9.291  | 9.202  | 9.112  | 9.02   |
| 6   | 13.745          | 10.925 | 9.78   | 9.148  | 8.746  | 8.466  | 8.26   | 8.102  | 7.976  | 7.874  | 7.718  | 7.559  | 7.396  | 7.313  | 7.229  | 7.143  | 7.057  | 6.969  | 6.88   |
| 7   | 12.246          | 9.547  | 8.451  | 7.847  | 7.46   | 7.191  | 6.993  | 6.84   | 6.719  | 6.62   | 6.469  | 6.314  | 6.155  | 6.074  | 5.992  | 5.908  | 5.824  | 5.737  | 5.65   |
| 8   | 11.259          | 8.649  | 7.591  | 7.006  | 6.632  | 6.371  | 6.178  | 6.029  | 5.911  | 5.814  | 5.667  | 5.515  | 5.359  | 5.279  | 5.198  | 5.116  | 5.032  | 4.946  | 4.859  |
| 9   | 10.561          | 8.022  | 6.992  | 6.422  | 6.057  | 5.802  | 5.613  | 5.467  | 5.351  | 5.257  | 5.111  | 4.962  | 4.808  | 4.729  | 4.649  | 4.567  | 4.483  | 4.398  | 4.31   |
| 10  | 10.044          | 7.559  | 6.552  | 5.994  | 5.636  | 5.386  | 5.2    | 5.057  | 4.942  | 4.849  | 4.706  | 4.558  | 4.405  | 4.327  | 4.247  | 4.165  | 4.082  | 3.996  | 3.909  |
| 11  | 9.646           | 7.206  | 6.217  | 5.668  | 5.316  | 5.069  | 4.886  | 4.744  | 4.632  | 4.539  | 4.397  | 4.251  | 4.099  | 4.021  | 3.941  | 3.86   | 3.776  | 3.69   | 3.602  |
| 12  | 9.33            | 6.927  | 5.953  | 5.412  | 5.064  | 4.821  | 4.64   | 4.499  | 4.388  | 4.296  | 4.155  | 4.01   | 3.858  | 3.78   | 3.701  | 3.619  | 3.535  | 3.449  | 3.362  |
| 13  | 9.074           | 6.701  | 5.739  | 5.205  | 4.862  | 4.62   | 4.441  | 4.302  | 4.191  | 4.1    | 3.96   | 3.815  | 3.665  | 3.587  | 3.507  | 3.425  | 3.341  | 3.255  | 3.165  |
| 14  | 8.862           | 6.515  | 5.564  | 5.035  | 4.695  | 4.456  | 4.278  | 4.14   | 4.03   | 3.939  | 3.8    | 3.656  | 3.505  | 3.427  | 3.348  | 3.266  | 3.181  | 3.094  | 3.004  |
| 15  | 8.683           | 6.359  | 5.417  | 4.893  | 4.556  | 4.318  | 4.142  | 4.004  | 3.895  | 3.805  | 3.666  | 3.522  | 3.372  | 3.294  | 3.214  | 3.132  | 3.047  | 2.959  | 2.868  |
| 16  | 8.531           | 6.226  | 5.292  | 4.773  | 4.437  | 4.202  | 4.026  | 3.89   | 3.78   | 3.691  | 3.553  | 3.409  | 3.259  | 3.181  | 3.101  | 3.018  | 2.933  | 2.845  | 2.753  |
| 17  | 8.4             | 6.112  | 5.185  | 4.669  | 4.336  | 4.102  | 3.927  | 3.791  | 3.682  | 3.593  | 3.455  | 3.312  | 3.162  | 3.084  | 3.003  | 2.92   | 2.835  | 2.746  | 2.653  |
| 18  | 8.285           | 6.013  | 5.092  | 4.579  | 4.248  | 4.015  | 3.841  | 3.705  | 3.597  | 3.508  | 3.371  | 3.227  | 3.077  | 2.999  | 2.919  | 2.835  | 2.749  | 2.66   | 2.566  |
| 19  | 8.185           | 5.926  | 5.01   | 4.5    | 4.171  | 3.939  | 3.765  | 3.631  | 3.523  | 3.434  | 3.297  | 3.153  | 3.003  | 2.925  | 2.844  | 2.761  | 2.674  | 2.584  | 2.489  |
| 20  | 8.096           | 5.849  | 4.938  | 4.431  | 4.103  | 3.871  | 3.699  | 3.564  | 3.457  | 3.368  | 3.231  | 3.088  | 2.938  | 2.859  | 2.778  | 2.695  | 2.608  | 2.517  | 2.422  |
| 21  | 8.017           | 5.78   | 4.874  | 4.369  | 4.042  | 3.812  | 3.64   | 3.506  | 3.398  | 3.31   | 3.173  | 3.03   | 2.88   | 2.801  | 2.72   | 2.636  | 2.548  | 2.457  | 2.36   |
| 22  | 7.945           | 5.719  | 4.817  | 4.313  | 3.988  | 3.758  | 3.587  | 3.453  | 3.346  | 3.258  | 3.121  | 2.978  | 2.827  | 2.749  | 2.667  | 2.583  | 2.495  | 2.403  | 2.305  |
| 23  | 7.881           | 5.664  | 4.765  | 4.264  | 3.939  | 3.71   | 3.539  | 3.406  | 3.299  | 3.211  | 3.074  | 2.931  | 2.781  | 2.702  | 2.62   | 2.535  | 2.447  | 2.354  | 2.256  |
| 24  | 7.823           | 5.614  | 4.718  | 4.218  | 3.895  | 3.667  | 3.496  | 3.363  | 3.256  | 3.168  | 3.032  | 2.889  | 2.738  | 2.659  | 2.577  | 2.492  | 2.403  | 2.31   | 2.21   |
| 25  | 7.77            | 5.568  | 4.675  | 4.177  | 3.855  | 3.627  | 3.457  | 3.324  | 3.217  | 3.129  | 2.993  | 2.85   | 2.699  | 2.62   | 2.538  | 2.453  | 2.364  | 2.27   | 2.165  |
| 26  | 7.721           | 5.526  | 4.637  | 4.14   | 3.818  | 3.591  | 3.421  | 3.288  | 3.182  | 3.094  | 2.958  | 2.815  | 2.664  | 2.585  | 2.503  | 2.417  | 2.327  | 2.233  | 2.13   |
| 27  | 7.677           | 5.488  | 4.601  | 4.106  | 3.785  | 3.558  | 3.388  | 3.256  | 3.149  | 3.062  | 2.926  | 2.783  | 2.632  | 2.552  | 2.47   | 2.384  | 2.294  | 2.198  | 2.09   |
| 28  | 7.636           | 5.453  | 4.568  | 4.074  | 3.754  | 3.528  | 3.358  | 3.226  | 3.12   | 3.032  | 2.896  | 2.753  | 2.602  | 2.522  | 2.44   | 2.354  | 2.263  | 2.167  | 2.064  |
| 29  | 7.598           | 5.42   | 4.538  | 4.045  | 3.725  | 3.499  | 3.33   | 3.198  | 3.092  | 3.005  | 2.868  | 2.726  | 2.574  | 2.495  | 2.412  | 2.325  | 2.234  | 2.138  | 2.034  |
| 30  | 7.562           | 5.39   | 4.51   | 4.018  | 3.699  | 3.473  | 3.304  | 3.173  | 3.067  | 2.979  | 2.843  | 2.7    | 2.549  | 2.469  | 2.386  | 2.299  | 2.208  | 2.111  | 2.006  |
| 40  | 7.314           | 5.179  | 4.313  | 3.828  | 3.514  | 3.291  | 3.124  | 2.993  | 2.888  | 2.801  | 2.665  | 2.522  | 2.369  | 2.288  | 2.203  | 2.114  | 2.019  | 1.917  | 1.805  |
| 60  | 7.077           | 4.977  | 4.126  | 3.649  | 3.339  | 3.119  | 2.953  | 2.823  | 2.718  | 2.632  | 2.496  | 2.352  | 2.198  | 2.115  | 2.028  | 1.936  | 1.836  | 1.726  | 1.602  |
| 120 | 6.851           | 4.787  | 3.949  | 3.48   | 3.174  | 2.956  | 2.792  | 2.663  | 2.559  | 2.472  | 2.336  | 2.192  | 2.035  | 1.95   | 1.86   | 1.763  | 1.656  | 1.533  | 1.385  |
| Inf | 6.635           | 4.605  | 3.782  | 3.319  | 3.017  | 2.802  | 2.639  | 2.511  | 2.407  | 2.321  | 2.185  | 2.039  | 1.878  | 1.791  | 1.696  | 1.592  | 1.473  | 1.325  | 1      |

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 30    |        |
| 3       | 25    |        |
| 4       | 25    |        |
| 5       | 15    |        |
| Total   | 120   |        |

1. A statistician is working on the amount of funding that companies obtain on a crowdsourcing website and has developed the following model. She used 26 companies to obtain the model

$$\hat{y} = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4 + b_5x_5$$

$$\hat{y} = 964.8 + 700.2x_1 + 317.5x_2 - 200.2x_3 + 15.3x_4 + 17.1x_5$$

The standard errors are:

$$\begin{aligned}s_{b_1} &= 12. \\ s_{b_2} &= 22.5 \\ s_{b_3} &= 101.8 \\ s_{b_4} &= 45.3 \\ s_{b_5} &= 2.3\end{aligned}$$

- $\hat{y}$ : the amount of funding obtained by a company in 1000 dollars
  - $x_1$ : the average annual salary of the founders
  - $x_2$ : the number of employees the startup hired
  - $x_3$ : a dummy variable that is 1 when the company's field is information technology and 0 otherwise
  - $x_4$ : the age of the company
  - $x_5$  is a dummy variable taking value 1 if the founders had previous failures and 0 otherwise
- (a) Interpret the estimated coefficients  $b_0 = 964.8$  and  $b_5 = 17.1$ .
- (b) Test, at the 5% level, the null hypothesis that the true coefficient on the dummy variable  $x_3$  is 0 against the alternative that it is not 0.
- (c) Find and interpret a 80% confidence interval for the parameter  $\beta_1$ .
- (d) If for the model,  $\text{SSR}=18147.5$  (Regression Sum of Squares) and  $\text{SSE} = 17136.5$  (Residual Sum of Squares), test the hypothesis that all the coefficients of the model are 0 (test overall significance of the model) using  $\alpha = 1\%$ .
- (e) Calculate  $R^2$  for the regression line.

2. Assume that in a binary classification problem with one feature  $X$ , the distribution of  $X$  in class  $k = 1$  is

$$f_1(x) = \frac{x}{\sigma_1^2} \exp\left(\frac{-x^2}{2\sigma_1^2}\right), x \geq 0$$

and the distribution of  $X$  in class  $k = 2$  is

$$f_2(x) = \frac{1}{x\sqrt{2\pi}\sigma_2} \exp\left(\frac{-(\ln x - \mu_2)^2}{2\sigma_2^2}\right), x \geq 0$$

- (a) Are there any conditions under which the discriminant function is a linear function of  $x$ ?
- (b) If  $\sigma_1 = \sigma_2 = 1$ ,  $\mu_1 = 10$ , and  $\pi_1 = \pi_2 = 0.5$ , in what class will  $x = 10$  be classified?

3. We perform best subset, forward stepwise, and backward stepwise selection on a single data set. For each approach, we obtain  $p+1$  models, containing  $0, 1, 2, \dots, p$  predictors. Explain your answers:
- (a) Which of the three models with  $k$  predictors has the smallest training RSS?
  - (b) Which of the three models with  $k$  predictors has the smallest test RSS?
  - (c) True or False:
    - i. The predictors in the  $k$ -variable model identified by forward stepwise are a subset of the predictors in the  $(k + 1)$ -variable model identified by forward stepwise selection.
    - ii. The predictors in the  $k$ -variable model identified by backward stepwise are a subset of the predictors in the  $(k + 1)$ -variable model identified by backward stepwise selection.
    - iii. The predictors in the  $k$ -variable model identified by backward stepwise are a subset of the predictors in the  $(k + 1)$ -variable model identified by forward stepwise selection.
    - iv. The predictors in the  $k$ -variable model identified by forward stepwise are a subset of the predictors in the  $(k + 1)$ -variable model identified by backward stepwise selection.
    - v. The predictors in the  $k$ -variable model identified by best subset are a subset of the predictors in the  $(k + 1)$ -variable model identified by best subset selection.

4. Consider the novel logistic regression method for binary classification ( $Y = 0$  or  $Y = 1$ ) with two features  $\mathbf{X} = (X_1, X_2)$ , formulated by

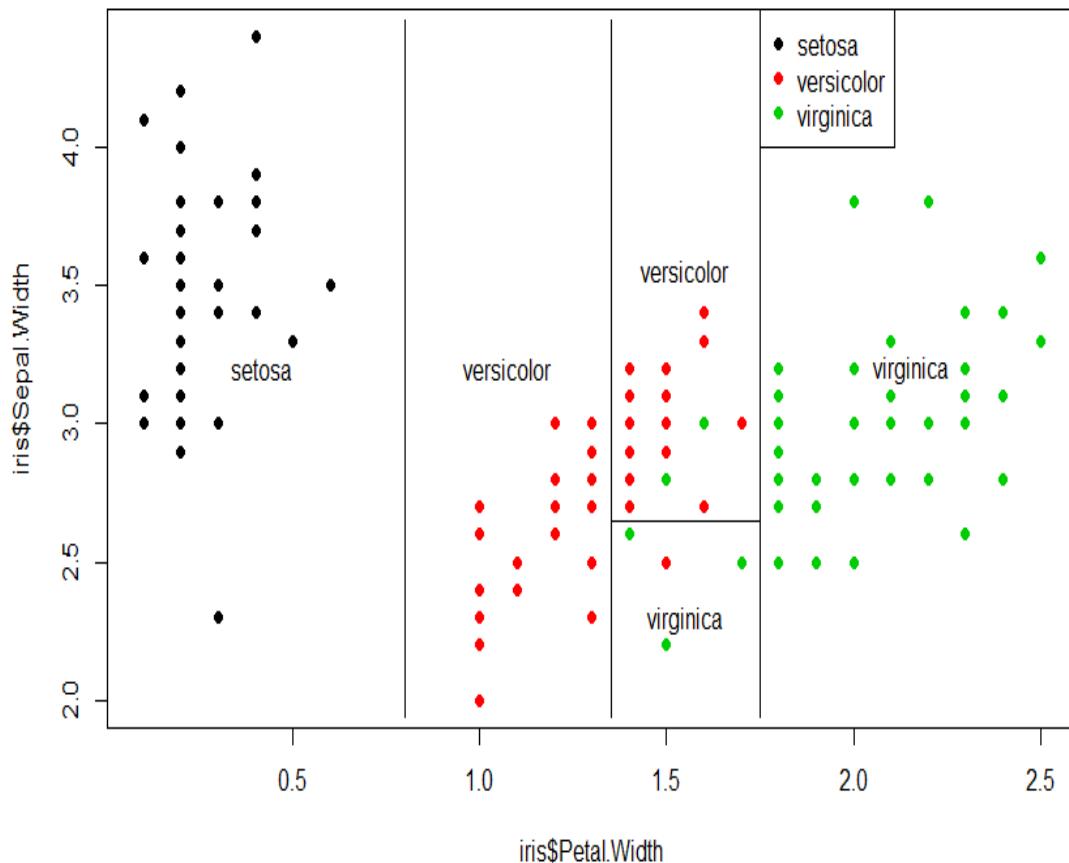
$$P(Y = 1|\mathbf{X}) = \frac{e^{\beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2}}{1 + e^{\beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2}}$$

Assume that using a dataset of 200 observations, we obtained the following estimates:

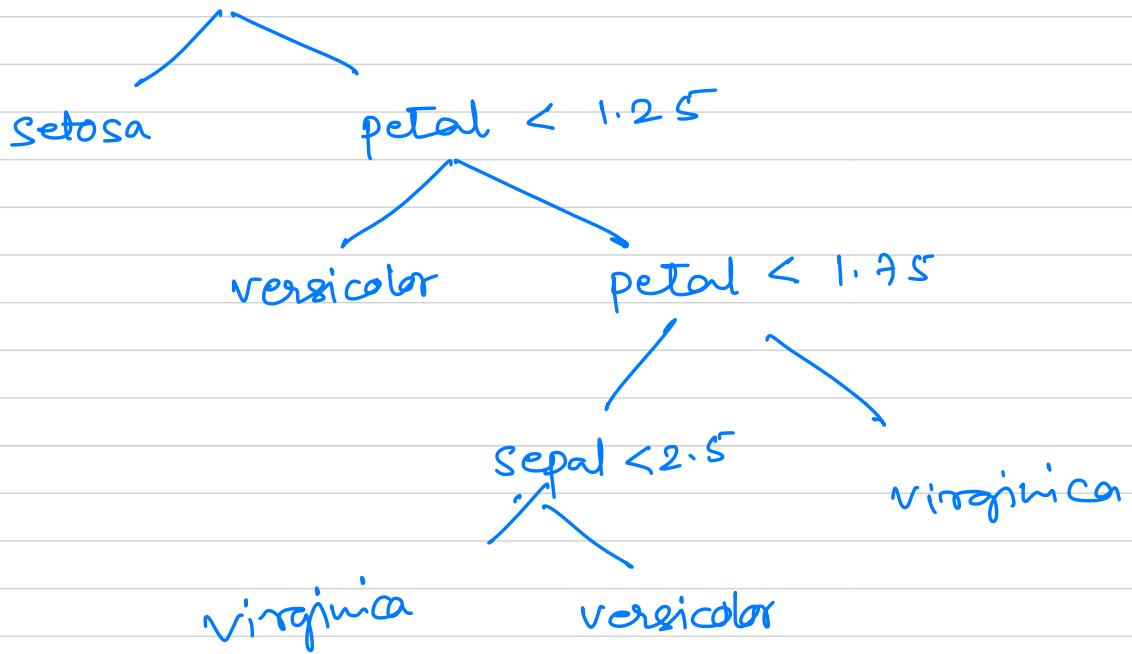
|           | Coefficient | Standard Error |
|-----------|-------------|----------------|
| $\beta_0$ | 1           | 0.2            |
| $\beta_1$ | 2           | 0.1            |
| $\beta_2$ | 1           | $s$            |
| $\beta_3$ | 1           | 0.5            |

- (a) Determine the equation for the decision boundary for this classifier.
- (b) Sketch the decision boundary for this classifier and clearly show the regions for the positive class and the negative class.
- (c) Find all values of  $s$  that makes the coefficient  $\beta_2$  statistically insignificant. You can consider the significance level to be  $\alpha = 0.05$ .

5. Consider the partitions that are shown in the following figure for predicting classes of the Iris dataset. Show the decision tree that corresponds to this partitioning. Assume uniform cutpoints for Petal width (0, 0.25, 0.5, ..., 2.25, 2.5) and also for sepal width (2.0, 2.25, 2.5,...,4).



petal width < 0.75



Scratch paper

Name:

USC ID:

Scratch paper

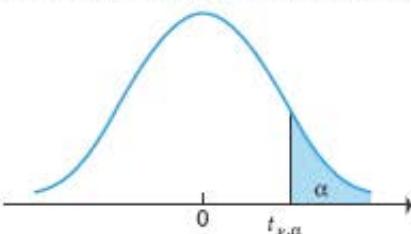
Name:

USC ID:

Cumulative Distribution Function,  $F(z)$ , of the Standard Normal Distribution Table

| $z$ | 0      | 0.01   | 0.02   | 0.03   | 0.04   | 0.05   | 0.06   | 0.07   | 0.08   | 0.09   |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 0.0 | 0.5000 | 0.5040 | 0.5080 | 0.5120 | 0.5160 | 0.5199 | 0.5239 | 0.5279 | 0.5319 | 0.5359 |
| 0.1 | 0.5398 | 0.5438 | 0.5478 | 0.5517 | 0.5557 | 0.5596 | 0.5636 | 0.5675 | 0.5714 | 0.5753 |
| 0.2 | 0.5793 | 0.5832 | 0.5871 | 0.5910 | 0.5948 | 0.5987 | 0.6026 | 0.6064 | 0.6103 | 0.6141 |
| 0.3 | 0.6179 | 0.6217 | 0.6255 | 0.6293 | 0.6331 | 0.6368 | 0.6406 | 0.6443 | 0.6480 | 0.6517 |
| 0.4 | 0.6554 | 0.6591 | 0.6628 | 0.6664 | 0.6700 | 0.6736 | 0.6772 | 0.6808 | 0.6844 | 0.6879 |
| 0.5 | 0.6915 | 0.6950 | 0.6985 | 0.7019 | 0.7054 | 0.7088 | 0.7123 | 0.7157 | 0.7190 | 0.7224 |
| 0.6 | 0.7257 | 0.7291 | 0.7324 | 0.7357 | 0.7389 | 0.7422 | 0.7454 | 0.7486 | 0.7517 | 0.7549 |
| 0.7 | 0.7580 | 0.7611 | 0.7642 | 0.7673 | 0.7704 | 0.7734 | 0.7764 | 0.7794 | 0.7823 | 0.7852 |
| 0.8 | 0.7881 | 0.7910 | 0.7939 | 0.7967 | 0.7995 | 0.8023 | 0.8051 | 0.8078 | 0.8106 | 0.8133 |
| 0.9 | 0.8159 | 0.8186 | 0.8212 | 0.8238 | 0.8264 | 0.8289 | 0.8315 | 0.8340 | 0.8365 | 0.8389 |
| 1.0 | 0.8413 | 0.8438 | 0.8461 | 0.8485 | 0.8508 | 0.8531 | 0.8554 | 0.8577 | 0.8599 | 0.8621 |
| 1.1 | 0.8643 | 0.8665 | 0.8686 | 0.8708 | 0.8729 | 0.8749 | 0.8770 | 0.8790 | 0.8810 | 0.8830 |
| 1.2 | 0.8849 | 0.8869 | 0.8888 | 0.8907 | 0.8925 | 0.8944 | 0.8962 | 0.8980 | 0.8997 | 0.9015 |
| 1.3 | 0.9032 | 0.9049 | 0.9066 | 0.9082 | 0.9099 | 0.9115 | 0.9131 | 0.9147 | 0.9162 | 0.9177 |
| 1.4 | 0.9192 | 0.9207 | 0.9222 | 0.9236 | 0.9251 | 0.9265 | 0.9279 | 0.9292 | 0.9306 | 0.9319 |
| 1.5 | 0.9332 | 0.9345 | 0.9357 | 0.9370 | 0.9382 | 0.9394 | 0.9406 | 0.9418 | 0.9429 | 0.9441 |
| 1.6 | 0.9452 | 0.9463 | 0.9474 | 0.9484 | 0.9495 | 0.9505 | 0.9515 | 0.9525 | 0.9535 | 0.9545 |
| 1.7 | 0.9554 | 0.9564 | 0.9573 | 0.9582 | 0.9591 | 0.9599 | 0.9608 | 0.9616 | 0.9625 | 0.9633 |
| 1.8 | 0.9641 | 0.9649 | 0.9656 | 0.9664 | 0.9671 | 0.9678 | 0.9686 | 0.9693 | 0.9699 | 0.9706 |
| 1.9 | 0.9713 | 0.9719 | 0.9726 | 0.9732 | 0.9738 | 0.9744 | 0.9750 | 0.9756 | 0.9761 | 0.9767 |
| 2.0 | 0.9772 | 0.9778 | 0.9783 | 0.9788 | 0.9793 | 0.9798 | 0.9803 | 0.9808 | 0.9812 | 0.9817 |
| 2.1 | 0.9821 | 0.9826 | 0.9830 | 0.9834 | 0.9838 | 0.9842 | 0.9846 | 0.9850 | 0.9854 | 0.9857 |
| 2.2 | 0.9861 | 0.9864 | 0.9868 | 0.9871 | 0.9875 | 0.9878 | 0.9881 | 0.9884 | 0.9887 | 0.9890 |
| 2.3 | 0.9893 | 0.9896 | 0.9898 | 0.9901 | 0.9904 | 0.9906 | 0.9909 | 0.9911 | 0.9913 | 0.9916 |
| 2.4 | 0.9918 | 0.9920 | 0.9922 | 0.9925 | 0.9927 | 0.9929 | 0.9931 | 0.9932 | 0.9934 | 0.9936 |
| 2.5 | 0.9938 | 0.9940 | 0.9941 | 0.9943 | 0.9945 | 0.9946 | 0.9948 | 0.9949 | 0.9951 | 0.9952 |
| 2.6 | 0.9953 | 0.9955 | 0.9956 | 0.9957 | 0.9959 | 0.9960 | 0.9961 | 0.9962 | 0.9963 | 0.9964 |
| 2.7 | 0.9965 | 0.9966 | 0.9967 | 0.9968 | 0.9969 | 0.9970 | 0.9971 | 0.9972 | 0.9973 | 0.9974 |
| 2.8 | 0.9974 | 0.9975 | 0.9976 | 0.9977 | 0.9977 | 0.9978 | 0.9979 | 0.9979 | 0.9980 | 0.9981 |
| 2.9 | 0.9981 | 0.9982 | 0.9982 | 0.9983 | 0.9984 | 0.9984 | 0.9985 | 0.9985 | 0.9986 | 0.9986 |
| 3.0 | 0.9987 | 0.9987 | 0.9987 | 0.9988 | 0.9988 | 0.9989 | 0.9989 | 0.9989 | 0.9990 | 0.9990 |
| 3.1 | 0.9990 | 0.9991 | 0.9991 | 0.9991 | 0.9992 | 0.9992 | 0.9992 | 0.9992 | 0.9993 | 0.9993 |
| 3.2 | 0.9993 | 0.9993 | 0.9994 | 0.9994 | 0.9994 | 0.9994 | 0.9994 | 0.9995 | 0.9995 | 0.9995 |
| 3.3 | 0.9995 | 0.9995 | 0.9995 | 0.9996 | 0.9996 | 0.9996 | 0.9996 | 0.9996 | 0.9996 | 0.9997 |

Cumulative Distribution Function,  $F(z)$ , of the Standard Normal Distribution Table

Upper Critical Values of Student's  $t$  Distribution with  $\nu$  Degrees of Freedom


For selected probabilities,  $\alpha$ , the table shows the values  $t_{\nu,\alpha}$  such that  $P(t_\nu > t_{\nu,\alpha}) = \alpha$ , where  $t_\nu$  is a Student's  $t$  random variable with  $\nu$  degrees of freedom. For example, the probability is .10 that a Student's  $t$  random variable with 10 degrees of freedom exceeds 1.372.

| $\nu$    | PROBABILITY OF EXCEEDING THE CRITICAL VALUE |       |        |        |        |         |
|----------|---|-------|--------|--------|--------|---------|
|          | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |
| 1        | 3.078                                       | 6.314 | 12.706 | 31.821 | 63.657 | 318.313 |
| 2        | 1.886                                       | 2.920 | 4.303  | 6.965  | 9.925  | 22.327  |
| 3        | 1.638                                       | 2.353 | 3.182  | 4.541  | 5.841  | 10.215  |
| 4        | 1.533                                       | 2.132 | 2.776  | 3.747  | 4.604  | 7.173   |
| 5        | 1.476                                       | 2.015 | 2.571  | 3.365  | 4.032  | 5.893   |
| 6        | 1.440                                       | 1.943 | 2.447  | 3.143  | 3.707  | 5.208   |
| 7        | 1.415                                       | 1.895 | 2.365  | 2.998  | 3.499  | 4.782   |
| 8        | 1.397                                       | 1.860 | 2.306  | 2.896  | 3.355  | 4.499   |
| 9        | 1.383                                       | 1.833 | 2.262  | 2.821  | 3.250  | 4.296   |
| 10       | 1.372                                       | 1.812 | 2.228  | 2.764  | 3.169  | 4.143   |
| 11       | 1.363                                       | 1.796 | 2.201  | 2.718  | 3.106  | 4.024   |
| 12       | 1.356                                       | 1.782 | 2.179  | 2.681  | 3.055  | 3.929   |
| 13       | 1.350                                       | 1.771 | 2.160  | 2.650  | 3.012  | 3.852   |
| 14       | 1.345                                       | 1.761 | 2.145  | 2.624  | 2.977  | 3.787   |
| 15       | 1.341                                       | 1.753 | 2.131  | 2.602  | 2.947  | 3.733   |
| 16       | 1.337                                       | 1.746 | 2.120  | 2.583  | 2.921  | 3.686   |
| 17       | 1.333                                       | 1.740 | 2.110  | 2.567  | 2.898  | 3.646   |
| 18       | 1.330                                       | 1.734 | 2.101  | 2.552  | 2.878  | 3.610   |
| 19       | 1.328                                       | 1.729 | 2.093  | 2.539  | 2.861  | 3.579   |
| 20       | 1.325                                       | 1.725 | 2.086  | 2.528  | 2.845  | 3.552   |
| 21       | 1.323                                       | 1.721 | 2.080  | 2.518  | 2.831  | 3.527   |
| 22       | 1.321                                       | 1.717 | 2.074  | 2.508  | 2.819  | 3.505   |
| 23       | 1.319                                       | 1.714 | 2.069  | 2.500  | 2.807  | 3.485   |
| 24       | 1.318                                       | 1.711 | 2.064  | 2.492  | 2.797  | 3.467   |
| 25       | 1.316                                       | 1.708 | 2.060  | 2.485  | 2.787  | 3.450   |
| 26       | 1.315                                       | 1.706 | 2.056  | 2.479  | 2.779  | 3.435   |
| 27       | 1.314                                       | 1.703 | 2.052  | 2.473  | 2.771  | 3.421   |
| 28       | 1.313                                       | 1.701 | 2.048  | 2.467  | 2.763  | 3.408   |
| 29       | 1.311                                       | 1.699 | 2.045  | 2.462  | 2.756  | 3.396   |
| 30       | 1.310                                       | 1.697 | 2.042  | 2.457  | 2.750  | 3.385   |
| 40       | 1.303                                       | 1.684 | 2.021  | 2.423  | 2.704  | 3.307   |
| 60       | 1.296                                       | 1.671 | 2.000  | 2.390  | 2.660  | 3.232   |
| 100      | 1.290                                       | 1.660 | 1.984  | 2.364  | 2.626  | 3.174   |
| $\infty$ | 1.282                                       | 1.645 | 1.960  | 2.326  | 2.576  | 3.090   |
| $\nu$    | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |

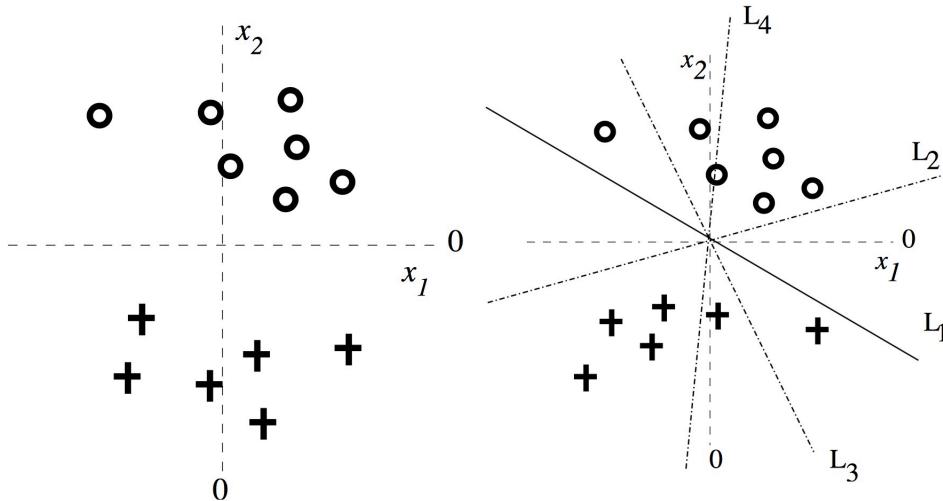
## F Table for $\alpha = 0.01$

| DF1 | $\alpha = 0.01$ |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |        |
|-----|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| DF2 | 1               | 2      | 3      | 4      | 5      | 6      | 7      | 8      | 9      | 10     | 12     | 15     | 20     | 24     | 30     | 40     | 60     | 120    | Inf    |
| 1   | 4052.2          | 4999.5 | 5403.4 | 5624.6 | 5763.7 | 5859   | 5928.4 | 5981.1 | 6022.5 | 6055.8 | 6106.3 | 6157.3 | 6208.7 | 6234.6 | 6260.6 | 6286.8 | 6313   | 6339.4 | 6365.9 |
| 2   | 98.503          | 99     | 99.166 | 99.249 | 99.299 | 99.333 | 99.356 | 99.374 | 99.388 | 99.399 | 99.416 | 99.433 | 99.449 | 99.458 | 99.466 | 99.474 | 99.482 | 99.491 | 99.499 |
| 3   | 34.116          | 30.817 | 29.457 | 28.71  | 28.237 | 27.911 | 27.672 | 27.489 | 27.345 | 27.229 | 27.052 | 26.872 | 26.69  | 26.598 | 26.505 | 26.411 | 26.316 | 26.221 | 26.125 |
| 4   | 21.198          | 18     | 16.694 | 15.977 | 15.522 | 15.207 | 14.976 | 14.799 | 14.659 | 14.546 | 14.374 | 14.198 | 14.02  | 13.929 | 13.838 | 13.745 | 13.652 | 13.558 | 13.463 |
| 5   | 16.258          | 13.274 | 12.06  | 11.392 | 10.967 | 10.672 | 10.456 | 10.289 | 10.158 | 10.051 | 9.888  | 9.722  | 9.553  | 9.466  | 9.379  | 9.291  | 9.202  | 9.112  | 9.02   |
| 6   | 13.745          | 10.925 | 9.78   | 9.148  | 8.746  | 8.466  | 8.26   | 8.102  | 7.976  | 7.874  | 7.718  | 7.559  | 7.396  | 7.313  | 7.229  | 7.143  | 7.057  | 6.969  | 6.88   |
| 7   | 12.246          | 9.547  | 8.451  | 7.847  | 7.46   | 7.191  | 6.993  | 6.84   | 6.719  | 6.62   | 6.469  | 6.314  | 6.155  | 6.074  | 5.992  | 5.908  | 5.824  | 5.737  | 5.65   |
| 8   | 11.259          | 8.649  | 7.591  | 7.006  | 6.632  | 6.371  | 6.178  | 6.029  | 5.911  | 5.814  | 5.667  | 5.515  | 5.359  | 5.279  | 5.198  | 5.116  | 5.032  | 4.946  | 4.859  |
| 9   | 10.561          | 8.022  | 6.992  | 6.422  | 6.057  | 5.802  | 5.613  | 5.467  | 5.351  | 5.257  | 5.111  | 4.962  | 4.808  | 4.729  | 4.649  | 4.567  | 4.483  | 4.398  | 4.31   |
| 10  | 10.044          | 7.559  | 6.552  | 5.994  | 5.636  | 5.386  | 5.2    | 5.057  | 4.942  | 4.849  | 4.706  | 4.558  | 4.405  | 4.327  | 4.247  | 4.165  | 4.082  | 3.996  | 3.909  |
| 11  | 9.646           | 7.206  | 6.217  | 5.668  | 5.316  | 5.069  | 4.886  | 4.744  | 4.632  | 4.539  | 4.397  | 4.251  | 4.099  | 4.021  | 3.941  | 3.86   | 3.776  | 3.69   | 3.602  |
| 12  | 9.33            | 6.927  | 5.953  | 5.412  | 5.064  | 4.821  | 4.64   | 4.499  | 4.388  | 4.296  | 4.155  | 4.01   | 3.858  | 3.78   | 3.701  | 3.619  | 3.535  | 3.449  | 3.362  |
| 13  | 9.074           | 6.701  | 5.739  | 5.205  | 4.862  | 4.62   | 4.441  | 4.302  | 4.191  | 4.1    | 3.96   | 3.815  | 3.665  | 3.587  | 3.507  | 3.425  | 3.341  | 3.255  | 3.165  |
| 14  | 8.862           | 6.515  | 5.564  | 5.035  | 4.695  | 4.456  | 4.278  | 4.14   | 4.03   | 3.939  | 3.8    | 3.656  | 3.505  | 3.427  | 3.348  | 3.266  | 3.181  | 3.094  | 3.004  |
| 15  | 8.683           | 6.359  | 5.417  | 4.893  | 4.556  | 4.318  | 4.142  | 4.004  | 3.895  | 3.805  | 3.666  | 3.522  | 3.372  | 3.294  | 3.214  | 3.132  | 3.047  | 2.959  | 2.868  |
| 16  | 8.531           | 6.226  | 5.292  | 4.773  | 4.437  | 4.202  | 4.026  | 3.89   | 3.78   | 3.691  | 3.553  | 3.409  | 3.259  | 3.181  | 3.101  | 3.018  | 2.933  | 2.845  | 2.753  |
| 17  | 8.4             | 6.112  | 5.185  | 4.669  | 4.336  | 4.102  | 3.927  | 3.791  | 3.682  | 3.593  | 3.455  | 3.312  | 3.162  | 3.084  | 3.003  | 2.92   | 2.835  | 2.746  | 2.653  |
| 18  | 8.285           | 6.013  | 5.092  | 4.579  | 4.248  | 4.015  | 3.841  | 3.705  | 3.597  | 3.508  | 3.371  | 3.227  | 3.077  | 2.999  | 2.919  | 2.835  | 2.749  | 2.66   | 2.566  |
| 19  | 8.185           | 5.926  | 5.01   | 4.5    | 4.171  | 3.939  | 3.765  | 3.631  | 3.523  | 3.434  | 3.297  | 3.153  | 3.003  | 2.925  | 2.844  | 2.761  | 2.674  | 2.584  | 2.489  |
| 20  | 8.096           | 5.849  | 4.938  | 4.431  | 4.103  | 3.871  | 3.699  | 3.564  | 3.457  | 3.368  | 3.231  | 3.088  | 2.938  | 2.859  | 2.778  | 2.695  | 2.608  | 2.517  | 2.422  |
| 21  | 8.017           | 5.78   | 4.874  | 4.369  | 4.042  | 3.812  | 3.64   | 3.506  | 3.398  | 3.31   | 3.173  | 3.03   | 2.88   | 2.801  | 2.72   | 2.636  | 2.548  | 2.457  | 2.36   |
| 22  | 7.945           | 5.719  | 4.817  | 4.313  | 3.988  | 3.758  | 3.587  | 3.453  | 3.346  | 3.258  | 3.121  | 2.978  | 2.827  | 2.749  | 2.667  | 2.583  | 2.495  | 2.403  | 2.305  |
| 23  | 7.881           | 5.664  | 4.765  | 4.264  | 3.939  | 3.71   | 3.539  | 3.406  | 3.299  | 3.211  | 3.074  | 2.931  | 2.781  | 2.702  | 2.62   | 2.535  | 2.447  | 2.354  | 2.256  |
| 24  | 7.823           | 5.614  | 4.718  | 4.218  | 3.895  | 3.667  | 3.496  | 3.363  | 3.256  | 3.168  | 3.032  | 2.889  | 2.738  | 2.659  | 2.577  | 2.492  | 2.403  | 2.31   | 2.21   |
| 25  | 7.77            | 5.568  | 4.675  | 4.177  | 3.855  | 3.627  | 3.457  | 3.324  | 3.217  | 3.129  | 2.993  | 2.85   | 2.699  | 2.62   | 2.538  | 2.453  | 2.364  | 2.27   | 2.165  |
| 26  | 7.721           | 5.526  | 4.637  | 4.14   | 3.818  | 3.591  | 3.421  | 3.288  | 3.182  | 3.094  | 2.958  | 2.815  | 2.664  | 2.585  | 2.503  | 2.417  | 2.327  | 2.233  | 2.13   |
| 27  | 7.677           | 5.488  | 4.601  | 4.106  | 3.785  | 3.558  | 3.388  | 3.256  | 3.149  | 3.062  | 2.926  | 2.783  | 2.632  | 2.552  | 2.47   | 2.384  | 2.294  | 2.198  | 2.09   |
| 28  | 7.636           | 5.453  | 4.568  | 4.074  | 3.754  | 3.528  | 3.358  | 3.226  | 3.12   | 3.032  | 2.896  | 2.753  | 2.602  | 2.522  | 2.44   | 2.354  | 2.263  | 2.167  | 2.064  |
| 29  | 7.598           | 5.42   | 4.538  | 4.045  | 3.725  | 3.499  | 3.33   | 3.198  | 3.092  | 3.005  | 2.868  | 2.726  | 2.574  | 2.495  | 2.412  | 2.325  | 2.234  | 2.138  | 2.034  |
| 30  | 7.562           | 5.39   | 4.51   | 4.018  | 3.699  | 3.473  | 3.304  | 3.173  | 3.067  | 2.979  | 2.843  | 2.7    | 2.549  | 2.469  | 2.386  | 2.299  | 2.208  | 2.111  | 2.006  |
| 40  | 7.314           | 5.179  | 4.313  | 3.828  | 3.514  | 3.291  | 3.124  | 2.993  | 2.888  | 2.801  | 2.665  | 2.522  | 2.369  | 2.288  | 2.203  | 2.114  | 2.019  | 1.917  | 1.805  |
| 60  | 7.077           | 4.977  | 4.126  | 3.649  | 3.339  | 3.119  | 2.953  | 2.823  | 2.718  | 2.632  | 2.496  | 2.352  | 2.198  | 2.115  | 2.028  | 1.936  | 1.836  | 1.726  | 1.602  |
| 120 | 6.851           | 4.787  | 3.949  | 3.48   | 3.174  | 2.956  | 2.792  | 2.663  | 2.559  | 2.472  | 2.336  | 2.192  | 2.035  | 1.95   | 1.86   | 1.763  | 1.656  | 1.533  | 1.385  |
| Inf | 6.635           | 4.605  | 3.782  | 3.319  | 3.017  | 2.802  | 2.639  | 2.511  | 2.407  | 2.321  | 2.185  | 2.039  | 1.878  | 1.791  | 1.696  | 1.592  | 1.473  | 1.325  | 1      |

1. Assume the binary classification task depicted in Figure ??, which we attempt to solve with the simple linear logistic regression model

$$\widehat{\Pr}(Y = 1 | X = x) = \hat{p}(x) = g(\beta_1 x_1 + \beta_2 x_2) = \frac{1}{1 + \exp(\beta_1 x_1 + \beta_2 x_2)}$$

for simplicity we do not use the parameter  $\beta_0$ . The training data is linearly separable, and the line  $L_1$  is the result of logistic regression, with zero training error..



Assume that we would like to find the classifier by *maximizing* the following regularized objective function, in which *only*  $\beta_2$  is regularized.

$$\prod_{i=1}^n [p(x_i)]^{y_i} [1 - p(x_i)]^{1-y_i} - \lambda \beta_2^2 = L(\beta_1, \beta_2) - \lambda \beta_2^2$$

- (a) Assume that  $\lambda$  is large. Which of the four lines  $L_2$ ,  $L_3$  or  $L_4$  determine whether it can result from regularizing  $\beta_2$ . Explain very briefly your reasons.
- (b) If we change the form of regularization to one-norm (absolute value) and also regularize  $\beta_2$  we get the following penalized log-likelihood

$$\prod_{i=1}^n [p(x_i)]^{y_i} [1 - p(x_i)]^{1-y_i} - \lambda(|\beta_1| + |\beta_2|) = L(\beta_1, \beta_2) - \lambda(|\beta_1| + |\beta_2|)$$

As we increase the regularization parameter  $\lambda$  which of the following scenarios is expected to be observed? Explain why.

- i. First  $\beta_1$  will become 0, then  $\beta_2$ .
- ii.  $\beta_1$  and  $\beta_2$  will become zero simultaneously.
- iii. First  $\beta_2$  will become 0, then  $\beta_1$ .
- iv. None of the weights will become exactly zero, only smaller as  $\lambda$  increases

**Solution:**<sup>1</sup>

- (a) When we regularize  $\beta_2$ , the resulting boundary can rely less on the value of  $x_2$  and therefore becomes more vertical.
- L2 here seems to be more horizontal than the unregularized solution so it cannot come as a result of penalizing  $\beta_2$ .
  - When  $\beta_2$  is small relative to  $\beta_1^2$  (as evidenced by high slope), and even though it would assign a rather low log-probability to the observed labels, it could be forced by a large regularization parameter  $\lambda$ . So L3 can arise as a result.
  - For very large  $\lambda$ , we obtain a separator that is very close to vertical (with negative slope) and in the limit, entirely vertical (line  $x_1 = 0$  or the  $x_2$  axis). L4 here is reflected across the  $x_2$  axis and has a positive slope, and therefore represents a poorer solution than its counterpart on the other side. For moderate regularization we have to get the best solution that we can construct while keeping  $\beta_2$  small. L4 is not the best and thus cannot come as a result of regularizing  $\beta_2$ .
- (b) The data can be classified with zero training error and therefore also with high log-probability by looking at the value of  $x_2$  alone, i.e. making  $\beta_1 = 0$ . Initially we might prefer to have a non-zero value for  $\beta_1$  but it will go to zero rather quickly as we increase regularization. Note that we pay a regularization penalty for a non-zero value of  $\beta_1$  and if it does not help classification why should the penalty be paid? The  $\mathcal{L}_1$  regularization ensures that  $\beta_1$  will indeed go to exactly zero. As  $\lambda$  increases further, even  $\beta_2$  will eventually become zero. We pay higher and higher cost for setting  $\beta_2$  to a non-zero value. Eventually this cost overwhelms the gain from the log-probability of labels that we can achieve with a non-zero  $\beta_2$ . Note that when  $\beta_1 = \beta_2 = 0$ , the log-probability of labels is a finite value  $n\Delta \log(0.5)$ .

---

<sup>1</sup>Important Note: Posting the course material to online forums or sharing it with other students is strictly prohibited. Instances will be reported to USC officials as academic dishonesty for disciplinary action.

2. A statistician is working on the amount of funding that companies obtain on a crwod-sourcing website and has developed the following model. She used 26 companies to obtain the model

$$\hat{y} = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + b_4x_4 + b_5x_5$$

$$\hat{y} = 964.8 + 700.2x_1 + 317.5x_2 - 200.2x_3 + 15.3x_4 + 17.1x_5$$

The standard errors are:

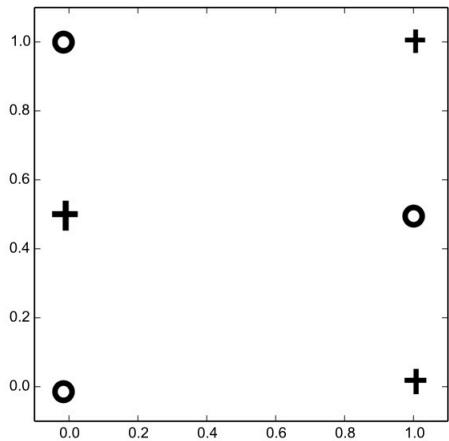
$$\begin{aligned}s_{b_1} &= 12. \\ s_{b_2} &= 22.5 \\ s_{b_3} &= 101.8 \\ s_{b_4} &= 45.3 \\ s_{b_5} &= 2.3\end{aligned}$$

- $\hat{y}$ : the amount of funding obtained by a company in 1000 dollars
  - $x_1$ : the average annual salary of the founders
  - $x_2$ : the number of employees the startup hired
  - $x_3$ : a dummy variable that is 1 when the company's field is information technology and 0 otherwise
  - $x_4$ : the age of the company
  - $x_5$  is a dummy variable taking value 1 if the founders had previous failures and 0 otherwise
- (a) Interpret the estimated coefficients  $b_0 = 964.8$  and  $b_3 = -200.2$  (10 pts)
- (b) Test, at the 2% level, the null hypothesis that the true coefficient on the dummy variable  $x_5$  is 0 against the alternative that it is not 0. (10 pts)
- (c) Find and interpret a 99.8% confidence interval for the parameter  $\beta_4$ . (10 pts)
- (d) If for the model, SSR=18147.5 (Regression Sum of Squares) and SSE = 17136.5 (Residual Sum of Squares), test the hypothesis that all the coefficients of the model are 0 (test overal significance of the model) using  $\alpha = 5\%$ . (10 pts).

**Solutions:**

- (a)  $b_0$  is the amount of the dependent variable that was not explained by the independent variables.  $b_3$  means that if the field of the company is information technology, on average, the funding that it will receive from the website will decrease by 200.2 units (1000 dollars)
- (b)  $\mathbf{H}_0 : \beta_5 = 0, \mathbf{H}_1 : \beta_5 \neq 0, t_{b_5} = \frac{b_5 - 0}{s_{b_5}} = \frac{17.1}{2.3} = 7.44$   
The rejection region is  $t > t_{n-K-1,\alpha/2} = t_{26-5-1,.01}$  or  $t < -t_{n-K-1,\alpha/2} = -t_{26-5-1,.01}$ .  
But from the table,  $t_{26-5-1,.01} = 2.528$ , therefore, we reject the null hypothesis that  $\beta_5 = 0$ .
- (c)  $t_{n-K-1,\alpha/2} = t_{26-5-1,0.001} =$ , so the Confidence interval for  $\beta_4$  is:  
 $[b_4 - t_{n-K-1,\alpha/2}s_{b_4}, b_4 + t_{n-K-1,\alpha/2}s_{b_4}] = [15.3 - (3.552)(45.3), 15.3 + (3.552)(45.3)] = [-145.61, 176.21]$ .
- (d)  $F = \frac{SSR/K}{SSE/(n-K-1)} = \frac{18147.5/5}{17136.5/(26-5-1)} = 4.2$ . The rejection region is  $F > F_{K,n-K-1,\alpha} = F_{5,20,0.05} = 2.7109$ , which means that we reject the null hypothesis that all coefficients are 0.

3. For the two dimensional training data shown below, determine whether or not each of the classification methods below, when trained appropriately, will have zero errors on the training set. In each case, briefly justify your answer. Moreover, provide a reasonable confusion matrix for each case.



- (a) Logistic Regression
- (b) SVM with Linear Kernel
- (c) SVM with RBF Kernel
- (d) Decision Tree
- (e) 3-Nearest-Neighbor Classifier (with Euclidean Distance).

**Solution:**

- (a) Logistic Regression and Linear SVM: linear decision boundaries,hence no.
- (b) SVM with RBF kernel: yes.
- (c) 3-NN: the 3 nearest neighbors of any point in our training set are 1 of the same class and 2 of the opposite class, hence 3-NN will be systematically wrong.
- (d) DT: yes, one can partition the space with lines orthogonal to the axes so that every sample ends up in a different region.

For methods with zero training error, the Confusion Matrix would be:

|         | Class o | Class + |
|---------|---------|---------|
| Class o | 3       | 0       |
| Class + | 0       | 3       |

For KNN:

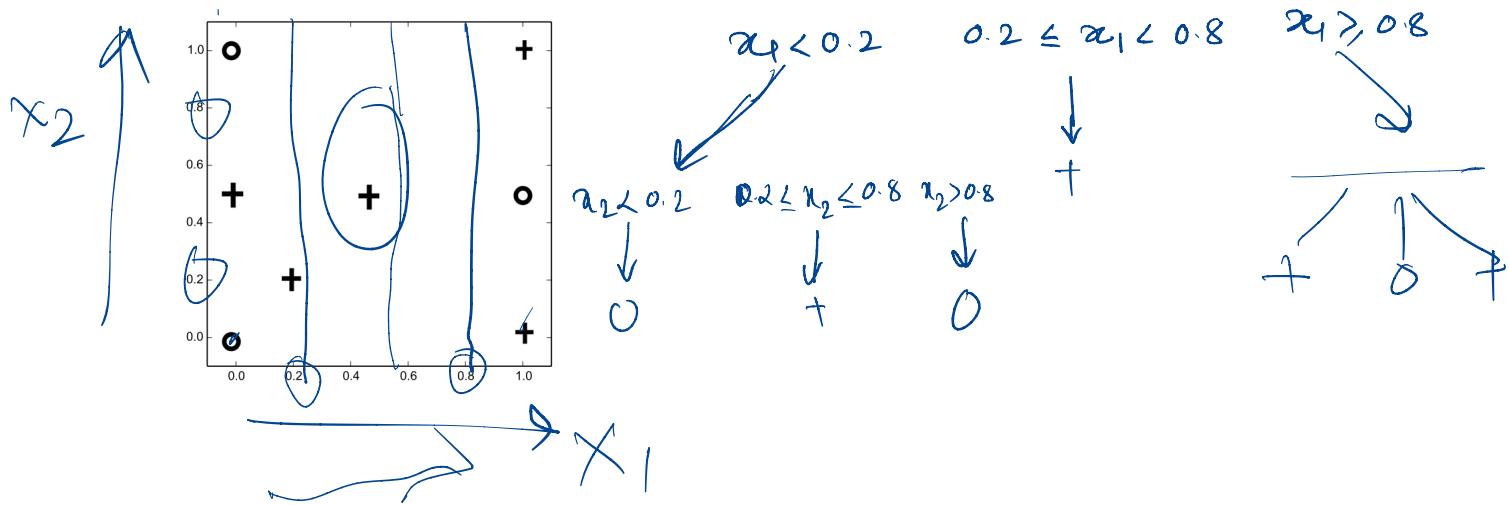
|         | Class o | Class + |
|---------|---------|---------|
| Class o | 0       | 3       |
| Class + | 3       | 0       |

For SVM, LR:

|         | Class o | Class + |
|---------|---------|---------|
| Class o | 2       | 1       |
| Class + | 1       | 2       |

4. For a numeric input, instead of a binary split in a decision tree, one can use a ternary split with two thresholds and three branches as  $X_j < s_1, s_1 \leq X_j < s_2, X_j \geq s_2$ .

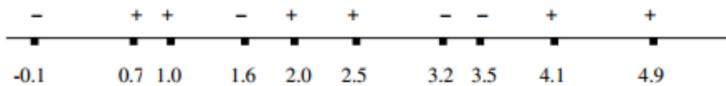
- (a) Propose a modification of the tree learning method to adjust the two thresholds,  $s_1$  and  $s_2$ .
- (b) What are the advantages and the disadvantages of such a node over a binary node?
- (c) How would you choose between a binary and a ternary decision tree for a given data set?
- (d) Perform two iterations of the ternary tree algorithm on the tree shown in below and draw the corresponding tree



**Solution:**

- (a) For the numeric attributes, instead of one split threshold, we need to try all possible pairs of split thresholds and choose the best. When there are two splits, there are three children, and in calculating the entropy/ Gini index after the splits, we need to sum up over the three sets corresponding to the instances taking the three branches.
- (b) The computational complexity of finding the best pair is higher and each node stores two thresholds instead of one and has three branches instead of two. The advantage is that one ternary node splits an input into three, whereas this requires two successive binary nodes.
- (c) Which one is better depends on the data at hand; if we have a ground truth that requires bounded intervals (e.g., rectangles), a ternary node may be advantageous. One has to choose between binary and ternary nodes using *Cross Validation*.
- (d) Left to the students.

5. Consider the following dataset with one real-valued input and one binary output (+ or -). The following questions assume that we are using  $k$ - nearest-neighbor learning with Euclidean distance to predict  $Y$  for an input  $X$ . What is the leave-one-out cross-validation error of 1-NN and 3-NN on this dataset, and decide which  $k$  is better.



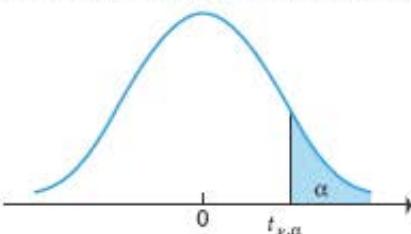
**Solution:** For each  $X_i$ , consider the majority vote of 1 nearest neighbors.

$$E_{LOOCV} = 0.4$$

For each  $X_i$ , consider the majority vote of 3 nearest neighbors.

$$E_{LOOCV} = 0.8$$

The 1-NN method is better

Upper Critical Values of Student's  $t$  Distribution with  $\nu$  Degrees of Freedom


For selected probabilities,  $\alpha$ , the table shows the values  $t_{\nu,\alpha}$  such that  $P(t_\nu > t_{\nu,\alpha}) = \alpha$ , where  $t_\nu$  is a Student's  $t$  random variable with  $\nu$  degrees of freedom. For example, the probability is .10 that a Student's  $t$  random variable with 10 degrees of freedom exceeds 1.372.

| $\nu$    | PROBABILITY OF EXCEEDING THE CRITICAL VALUE |       |        |        |        |         |
|----------|---|-------|--------|--------|--------|---------|
|          | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |
| 1        | 3.078                                       | 6.314 | 12.706 | 31.821 | 63.657 | 318.313 |
| 2        | 1.886                                       | 2.920 | 4.303  | 6.965  | 9.925  | 22.327  |
| 3        | 1.638                                       | 2.353 | 3.182  | 4.541  | 5.841  | 10.215  |
| 4        | 1.533                                       | 2.132 | 2.776  | 3.747  | 4.604  | 7.173   |
| 5        | 1.476                                       | 2.015 | 2.571  | 3.365  | 4.032  | 5.893   |
| 6        | 1.440                                       | 1.943 | 2.447  | 3.143  | 3.707  | 5.208   |
| 7        | 1.415                                       | 1.895 | 2.365  | 2.998  | 3.499  | 4.782   |
| 8        | 1.397                                       | 1.860 | 2.306  | 2.896  | 3.355  | 4.499   |
| 9        | 1.383                                       | 1.833 | 2.262  | 2.821  | 3.250  | 4.296   |
| 10       | 1.372                                       | 1.812 | 2.228  | 2.764  | 3.169  | 4.143   |
| 11       | 1.363                                       | 1.796 | 2.201  | 2.718  | 3.106  | 4.024   |
| 12       | 1.356                                       | 1.782 | 2.179  | 2.681  | 3.055  | 3.929   |
| 13       | 1.350                                       | 1.771 | 2.160  | 2.650  | 3.012  | 3.852   |
| 14       | 1.345                                       | 1.761 | 2.145  | 2.624  | 2.977  | 3.787   |
| 15       | 1.341                                       | 1.753 | 2.131  | 2.602  | 2.947  | 3.733   |
| 16       | 1.337                                       | 1.746 | 2.120  | 2.583  | 2.921  | 3.686   |
| 17       | 1.333                                       | 1.740 | 2.110  | 2.567  | 2.898  | 3.646   |
| 18       | 1.330                                       | 1.734 | 2.101  | 2.552  | 2.878  | 3.610   |
| 19       | 1.328                                       | 1.729 | 2.093  | 2.539  | 2.861  | 3.579   |
| 20       | 1.325                                       | 1.725 | 2.086  | 2.528  | 2.845  | 3.552   |
| 21       | 1.323                                       | 1.721 | 2.080  | 2.518  | 2.831  | 3.527   |
| 22       | 1.321                                       | 1.717 | 2.074  | 2.508  | 2.819  | 3.505   |
| 23       | 1.319                                       | 1.714 | 2.069  | 2.500  | 2.807  | 3.485   |
| 24       | 1.318                                       | 1.711 | 2.064  | 2.492  | 2.797  | 3.467   |
| 25       | 1.316                                       | 1.708 | 2.060  | 2.485  | 2.787  | 3.450   |
| 26       | 1.315                                       | 1.706 | 2.056  | 2.479  | 2.779  | 3.435   |
| 27       | 1.314                                       | 1.703 | 2.052  | 2.473  | 2.771  | 3.421   |
| 28       | 1.313                                       | 1.701 | 2.048  | 2.467  | 2.763  | 3.408   |
| 29       | 1.311                                       | 1.699 | 2.045  | 2.462  | 2.756  | 3.396   |
| 30       | 1.310                                       | 1.697 | 2.042  | 2.457  | 2.750  | 3.385   |
| 40       | 1.303                                       | 1.684 | 2.021  | 2.423  | 2.704  | 3.307   |
| 60       | 1.296                                       | 1.671 | 2.000  | 2.390  | 2.660  | 3.232   |
| 100      | 1.290                                       | 1.660 | 1.984  | 2.364  | 2.626  | 3.174   |
| $\infty$ | 1.282                                       | 1.645 | 1.960  | 2.326  | 2.576  | 3.090   |
| $\nu$    | 0.10  | 0.05  | 0.025  | 0.01   | 0.005  | 0.001   |

F - Distribution ( $\alpha = 0.05$  in the Right Tail)

|                        |        | Numerator Degrees of Freedom |        |        |        |        |        |        |        |        |
|------------------------|--------|------------------------------|--------|--------|--------|--------|--------|--------|--------|--------|
|                        |        | 1                            | 2      | 3      | 4      | 5      | 6      | 7      | 8      | 9      |
| $df_2 \backslash df_1$ | 1      | 199.50                       | 215.71 | 224.58 | 230.16 | 233.99 | 236.77 | 238.88 | 240.54 | 240.54 |
|                        | 2      | 18.513                       | 19.000 | 19.164 | 19.247 | 19.296 | 19.330 | 19.353 | 19.371 | 19.385 |
| 3                      | 10.128 | 9.5521                       | 9.2766 | 9.1172 | 9.0135 | 8.9406 | 8.8867 | 8.8452 | 8.8123 | 8.8123 |
| 4                      | 7.7086 | 9.9443                       | 6.5914 | 6.3882 | 6.2561 | 6.1631 | 6.0942 | 6.0410 | 6.9988 | 6.9988 |
| 5                      | 6.6079 | 5.7861                       | 5.4095 | 5.1922 | 5.0503 | 4.9503 | 4.8759 | 4.8183 | 4.7725 | 4.7725 |
| 6                      | 5.9874 | 5.1433                       | 4.7571 | 4.5337 | 4.3874 | 4.2839 | 4.2067 | 4.1468 | 4.0990 | 4.0990 |
| 7                      | 5.5914 | 4.7374                       | 4.3468 | 4.1203 | 3.9715 | 3.8660 | 3.7870 | 3.7257 | 3.6767 | 3.6767 |
| 8                      | 5.3177 | 4.4590                       | 4.0662 | 3.8379 | 3.6875 | 3.5806 | 3.5005 | 3.4381 | 3.3881 | 3.3881 |
| 9                      | 5.1174 | 4.2565                       | 3.8625 | 3.6331 | 3.4817 | 3.3738 | 3.2927 | 3.2296 | 3.1789 | 3.1789 |
| 10                     | 4.9646 | 4.1028                       | 3.7083 | 3.4780 | 3.3258 | 3.2172 | 3.1355 | 3.0717 | 3.0204 | 3.0204 |
| 11                     | 4.8443 | 3.9823                       | 3.5874 | 3.3567 | 3.2039 | 3.0946 | 3.0123 | 2.9480 | 2.8962 | 2.8962 |
| 12                     | 4.7472 | 3.8853                       | 3.4903 | 3.2592 | 3.1059 | 2.9961 | 2.9134 | 2.8486 | 2.7964 | 2.7964 |
| 13                     | 4.6672 | 3.8056                       | 3.4105 | 3.1791 | 3.0254 | 2.9153 | 2.8321 | 2.7669 | 2.7144 | 2.7144 |
| 14                     | 4.6001 | 3.7389                       | 3.3439 | 3.1122 | 2.9582 | 2.8477 | 2.7642 | 2.6987 | 2.6458 | 2.6458 |
| 15                     | 4.5431 | 3.6823                       | 3.2874 | 3.0556 | 2.9013 | 2.7905 | 2.7066 | 2.6408 | 2.5876 | 2.5876 |
| 16                     | 4.4940 | 3.6337                       | 3.2389 | 3.0069 | 2.8524 | 2.7413 | 2.6572 | 2.5911 | 2.5377 | 2.5377 |
| 17                     | 4.4513 | 3.5915                       | 3.1968 | 2.9647 | 2.8100 | 2.6987 | 2.6143 | 2.5480 | 2.4943 | 2.4943 |
| 18                     | 4.4139 | 3.5546                       | 3.1599 | 2.9277 | 2.7729 | 2.6613 | 2.5767 | 2.5102 | 2.4563 | 2.4563 |
| 19                     | 4.3807 | 3.5219                       | 3.1274 | 2.8951 | 2.7401 | 2.6283 | 2.5435 | 2.4768 | 2.4227 | 2.4227 |
| 20                     | 4.3512 | 3.4928                       | 3.0984 | 2.8661 | 2.7109 | 2.5990 | 2.5140 | 2.4471 | 2.3928 | 2.3928 |
| 21                     | 4.3248 | 3.4668                       | 3.0725 | 2.8401 | 2.6848 | 2.5727 | 2.4876 | 2.4205 | 2.3660 | 2.3660 |
| 22                     | 4.3009 | 3.4434                       | 3.0491 | 2.8167 | 2.6613 | 2.5491 | 2.4638 | 2.3965 | 2.3419 | 2.3419 |
| 23                     | 4.2793 | 3.4221                       | 3.0280 | 2.7955 | 2.6400 | 2.5277 | 2.4422 | 2.3748 | 2.3201 | 2.3201 |
| 24                     | 4.2597 | 3.4028                       | 3.0088 | 2.7763 | 2.6207 | 2.5082 | 2.4226 | 2.3551 | 2.3002 | 2.3002 |
| 25                     | 4.2417 | 3.3852                       | 2.9912 | 2.7587 | 2.6030 | 2.4904 | 2.4047 | 2.3371 | 2.2821 | 2.2821 |
| 26                     | 4.2252 | 3.3690                       | 2.9752 | 2.7426 | 2.5868 | 2.4741 | 2.3883 | 2.3205 | 2.2655 | 2.2655 |
| 27                     | 4.2100 | 3.3541                       | 2.9604 | 2.7278 | 2.5719 | 2.4591 | 2.3732 | 2.3053 | 2.2501 | 2.2501 |
| 28                     | 4.1960 | 3.3404                       | 2.9467 | 2.7141 | 2.5581 | 2.4453 | 2.3593 | 2.2913 | 2.2360 | 2.2360 |
| 29                     | 4.1830 | 3.3277                       | 2.9340 | 2.7014 | 2.5454 | 2.4324 | 2.3463 | 2.2783 | 2.2229 | 2.2229 |
| 30                     | 4.1709 | 3.3158                       | 2.9223 | 2.6896 | 2.5336 | 2.4205 | 2.3343 | 2.2662 | 2.2107 | 2.2107 |
| 40                     | 4.0847 | 3.2317                       | 2.8387 | 2.6060 | 2.4495 | 2.3359 | 2.2490 | 2.1802 | 2.1240 | 2.1240 |
| 60                     | 4.0012 | 3.1504                       | 2.7581 | 2.5252 | 2.3683 | 2.2541 | 2.1665 | 2.0970 | 2.0401 | 2.0401 |
| 120                    | 3.9201 | 3.0718                       | 2.6802 | 2.4472 | 2.2899 | 2.1750 | 2.0868 | 2.0164 | 1.9588 | 1.9588 |
| $\infty$               | 3.8415 | 2.9957                       | 2.6049 | 2.3719 | 2.2141 | 2.0986 | 2.0096 | 1.9384 | 1.8799 | 1.8799 |

Denominator Degrees of Freedom

## CSCI-567 Spring 2019 Midterm Exam [\[Rubric\]](#)

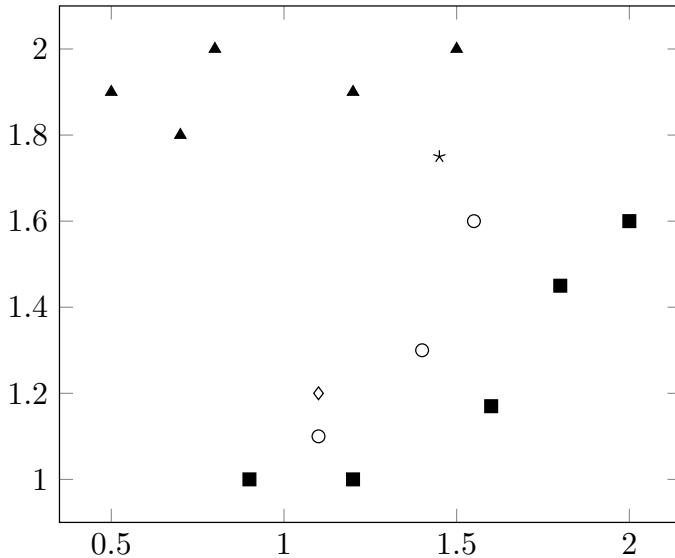
| Problem | 1  | 2  | 3  | 4  | 5  | Total |
|---------|----|----|----|----|----|-------|
| Points  | 28 | 20 | 20 | 15 | 17 | 100   |

Please read the following instructions carefully:

- The exam has a total of **15 pages** (including this cover and two blank pages in the end). Each problem have several questions. Once you are permitted to open your exam (and not before), you should check and make sure that you are not missing any pages.
- Duration of the exam is **3 hours**. Questions are not ordered by their difficulty. Budget your time on each question carefully.
- Select **one and only one answer** for all multiple choice questions.
- Answers should be **concise** and written down **legibly**. All questions can be done within 5-12 lines.
- You must answer each question on the page provided. You can use the last two blank pages as scratch paper. Raise your hand to ask a proctor for more if needed.
- This is a **closed-book/notes** exam. Consulting any resources is NOT permitted.
- Any kind of cheating will lead to **score 0** for the entire exam and be reported to SJACS.
- You **may not** leave your seat **for any reason** unless you submit your exam at that point.

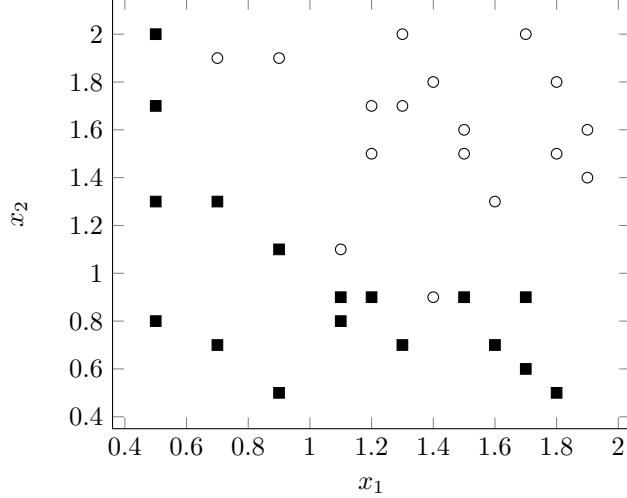
# 1 Multiple Choice (28 points)

Questions 1 through 3 deal with the following data, where squares, triangles, and open circles are three different classes of data in the training set and the diamond ( $\diamond$ ) and the star (\*) are test points. The distance metric is  $L_2$  distance.

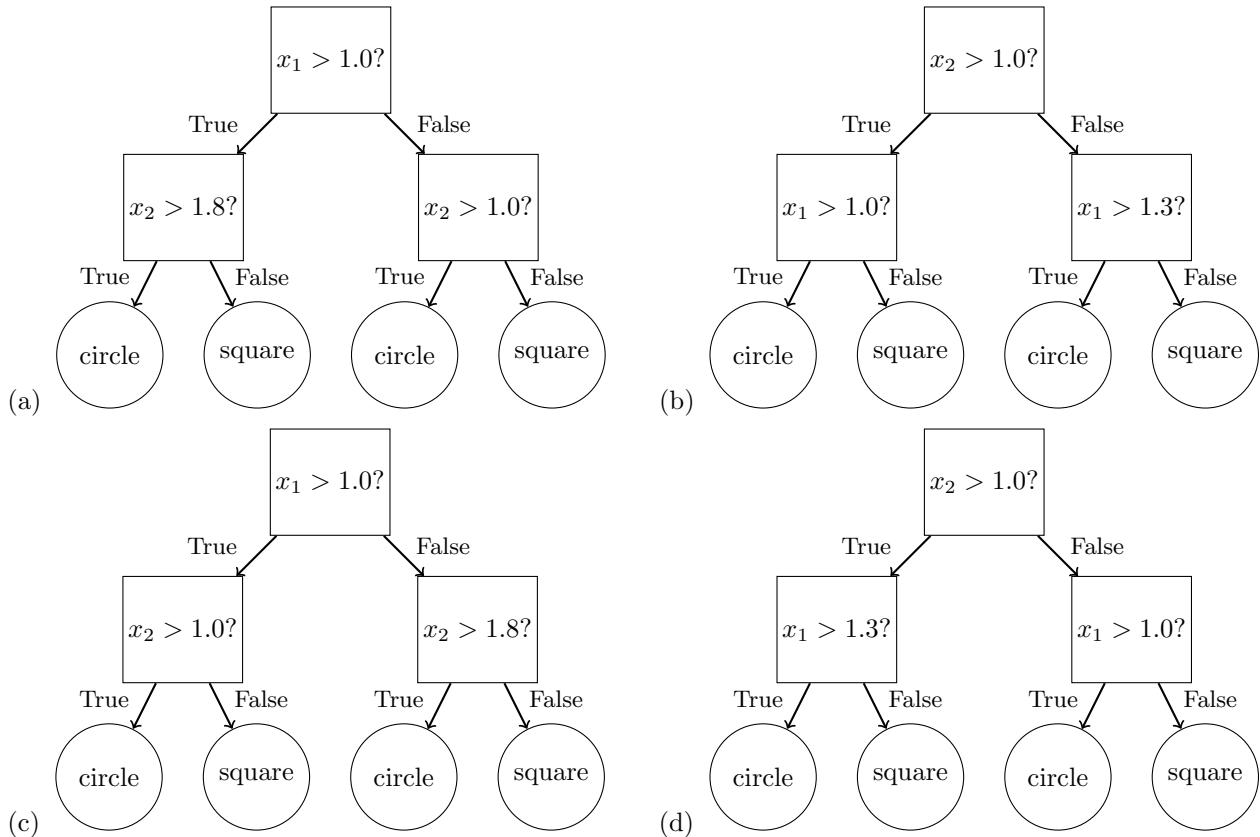


1. What label will we predict for diamond with  $k = 1$ ? Answer: B (2 points)
  - (a) Triangle
  - (b) Circle
  - (c) Square
  - (d) Cannot be determined from data available
  
2. What label will we predict for diamond with  $k = 5$ ? Answer: C (2 points)
  - (a) Triangle
  - (b) Circle
  - (c) Square
  - (d) Cannot be determined from data available
  
3. Which of the following values of  $k$  can be used to classify the star as triangle? Answer: B (2 points)
  - (a) 1
  - (b) 3
  - (c) 5
  - (d) None of above

Question 4 deals with the following data, where squares and circles are two different classes of data.



4. Suppose we use the data as training data, which of the following decision trees has the smallest training error? Answer: C (2 points)



5. K-fold cross-validation runtime complexity is [Answer: A](#) **(2 points)**
- (a) linear in K
  - (b) quadratic in K
  - (c) cubic in K
  - (d) exponential in K
6. The multiclass perceptron algorithm is essentially minimizing the multiclass perceptron loss via SGD with learning rate 1. Based on this information, which of the following is the multiclass perceptron loss? [Answer: C](#) **(2 points)**
- (a)  $\sum_{n=1}^N \max_{k \neq y_n} \mathbf{w}_k^\top \mathbf{x}_n - \mathbf{w}_{y_n}^\top \mathbf{x}_n$
  - (b)  $\sum_{n=1}^N \max \{0, \max_{k \neq y_n} \mathbf{w}_k^\top \mathbf{x}_n\}$
  - (c)  $\sum_{n=1}^N \max \{0, \max_{k \neq y_n} \mathbf{w}_k^\top \mathbf{x}_n - \mathbf{w}_{y_n}^\top \mathbf{x}_n\}$
  - (d)  $\sum_{n=1}^N \max \{0, \max_{k \in [C]} \mathbf{w}_k^\top \mathbf{x}_n\}$
7. Which is *not* an advantage of using kernels? [Answer: A](#) **(2 points)**
- (a) They can scale well to large number of data instances.
  - (b) They can convert linear models into nonlinear models.
  - (c) They can efficiently represent high dimensional inputs.
  - (d) They can be analyzed with statistical learning theory.
8. Which is *not* a valid kernel function, for samples  $x$  and  $y$  and kernels  $k_1(\cdot, \cdot)$  and  $k_2(\cdot, \cdot)$ ? [Answer: B](#) **(2 points)**
- (a)  $k(x, y) = 5$
  - (b)  $k(x, y) = x + y$
  - (c)  $k(x, y) = e^{x+y}$
  - (d)  $k(x, y) = \langle x, y \rangle^3 + (\langle x, y \rangle + 1)^2$
9. Suppose we apply the kernel trick with a kernel function  $k$  to the nearest neighbor algorithm (with L2 distance in the new feature space). What is the nearest neighbor of a new data point  $\mathbf{x}$  from a training set  $S = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ ? [Answer: D](#) **(2 points)**
- (a)  $\arg \min_{\mathbf{x}_n \in S} k(\mathbf{x}_n, \mathbf{x}_n) + k(\mathbf{x}, \mathbf{x}) + 2k(\mathbf{x}_n, \mathbf{x})$
  - (b)  $\arg \min_{\mathbf{x}_n \in S} k(\mathbf{x}_n, \mathbf{x})$
  - (c)  $\arg \min_{\mathbf{x}_n \in S} (k(\mathbf{x}_n, \mathbf{x}) - k(\mathbf{x}, \mathbf{x}_n))^2$
  - (d)  $\arg \min_{\mathbf{x}_n \in S} k(\mathbf{x}_n, \mathbf{x}_n) - 2k(\mathbf{x}_n, \mathbf{x})$

10. Which of the following is wrong about neural nets? [Answer: C](#). **(2 points)**
- (a) A fully connected feedforward neural net without nonlinear activation functions is the same as a linear model.
  - (b) Dropout technique prevents overfitting.
  - (c) A neural net with one hidden layer and a fixed number of neurons can represent any continuous function.
  - (d) A max-pooling layer has no parameters to be learned.
11. Suppose a convolution layer takes a  $4 \times 6 \times 3$  image as input and outputs a  $3 \times 4 \times 6$  tensor. Which of the following is a possible configuration of this layer? [Answer: D](#) **(2 points)**
- (a) Two  $2 \times 4 \times 3$  filters, stride 1, no zero-padding.
  - (b) Two  $2 \times 2 \times 3$  filters, stride 2, 1 zero-padding.
  - (c) Six  $2 \times 4 \times 3$  filters, stride 1, no zero-padding.
  - (d) Six  $2 \times 2 \times 3$  filters, stride 2, 1 zero-padding.
12. Recall that the Gini impurity of a distribution  $p$  over a set of  $K$  items is defined as  $\sum_{k=1}^K p(k)(1-p(k))$ . Which of the following distributions has the largest Gini impurity? [Answer: A](#). **(2 points)**
- (a) (0.25, 0.25, 0.25, 0.25)
  - (b) (0.2, 0.3, 0.5, 0)
  - (c) (1, 0, 0, 0)
  - (d) (0.5, 0.5, 0, 0)
13. Which of the following cannot be used as regularization to control model complexity? [Answer: C](#) **(2 points)**
- (a)  $R(\mathbf{w}) = \sum_{d=1}^D |w_d|$
  - (b)  $R(\mathbf{w}) = \sum_{d=1}^D |w_d|^3$
  - (c)  $R(\mathbf{w}) = \sum_{d=1}^D w_d^3$
  - (d)  $R(\mathbf{w}) = \sum_{d=1}^D |w_d|^4$
14. In K-Fold cross-validation, a large value of K could result in which of the following? [Answer: A](#) **(2 points)**
- (a) low bias, high variance
  - (b) high bias, low variance
  - (c) it could result in (a) or (b)
  - (d) none of the above

## 2 Naïve Bayes (20 points)

Assume we have a data set with three binary input attributes,  $A, B, C$ , and one binary outcome attribute  $Y$ . The three input attributes,  $A, B, C$  take values in the set  $\{0, 1\}$  while the  $Y$  attribute takes values in the set  $\{\text{True}, \text{False}\}$ .

| A | B | C | Y     |
|---|---|---|-------|
| 0 | 1 | 1 | True  |
| 1 | 1 | 0 | True  |
| 1 | 0 | 1 | False |
| 1 | 1 | 1 | False |
| 0 | 1 | 1 | True  |
| 0 | 0 | 0 | True  |
| 0 | 1 | 1 | False |
| 1 | 0 | 1 | False |
| 0 | 1 | 0 | True  |
| 1 | 1 | 1 | True  |

In this problem, a set  $S$  of input values which consists of  $A, B, C$  with  $P(S) > 0$  will have an unambiguous predicted classification of  $Y = \text{True} \leftrightarrow P(Y = \text{True}|S) > P(Y = \text{False}|S)$

15. How would you classify the record  $S = (A = 1, B = 1, C = 0)$  based on the data given in the table above? Write down both  $P(Y = \text{True}|A = 1, B = 1, C = 0)$  and  $P(Y = \text{False}|A = 1, B = 1, C = 0)$ , and explain if  $Y$  should be True or False. (10 points)

$$P(Y = \text{True}|A = 1, B = 1, C = 0) = \frac{P(A = 1, B = 1, C = 0|Y = \text{True})P(Y = \text{True})}{P(A = 1, B = 1, C = 0)}$$

$$P(Y = \text{False}|A = 1, B = 1, C = 0) = \frac{P(A = 1, B = 1, C = 0|Y = \text{False})P(Y = \text{False})}{P(A = 1, B = 1, C = 0)}$$

(2 points)

$$P(A = 1|Y = \text{True})P(B = 1|Y = \text{True})P(C = 0|Y = \text{True})P(Y = \text{True})$$

$$= \frac{2}{6} * \frac{5}{6} * \frac{3}{6} * \frac{6}{10}$$

$$= \frac{1}{12}$$

(3 points)

$$P(A = 1|Y = \text{False})P(B = 1|Y = \text{False})P(C = 0|Y = \text{False})P(Y = \text{False})$$

$$= \frac{3}{4} * \frac{2}{4} * \frac{0}{4} * \frac{4}{10}$$

$$= 0$$

(3 points)

$$\because P(A = 1|Y = \text{True})P(B = 1|Y = \text{True})P(C = 0|Y = \text{True})P(Y = \text{True}) > P(A = 1|Y = \text{False})P(B = 1|Y = \text{False})P(C = 0|Y = \text{False})P(Y = \text{False}) \quad (1 \text{ points})$$

$$\therefore Y = \text{True.} \quad (1 \text{ points})$$

16. How would you classify the record  $S = (A = 0, B = 0, C = 1)$  based on the data given in the table above? Write down both  $P(Y = True|A = 0, B = 0, C = 1)$  and  $P(Y = False|A = 0, B = 0, C = 1)$ , and explain if Y should be True or False. **(10 points)**

$$P(Y = True|A = 0, B = 0, C = 1) = \frac{P(A = 0, B = 0, C = 1|Y = True)P(Y = True)}{P(A = 0, B = 0, C = 1)}$$

$$P(Y = False|A = 0, B = 0, C = 1) = \frac{P(A = 0, B = 0, C = 1|Y = False)P(Y = False)}{P(A = 0, B = 0, C = 1)}$$

**(2 points)**

$$P(A = 0|Y = True)P(B = 0|Y = True)P(C = 1|Y = True)P(Y = True)$$

$$= \frac{4}{6} * \frac{1}{6} * \frac{3}{6} * \frac{6}{10}$$

$$= \frac{1}{30}$$

**(3 points)**

$$P(A = 0|Y = False)P(B = 0|Y = False)P(C = 1|Y = False)P(Y = False)$$

$$= \frac{1}{4} * \frac{2}{4} * \frac{4}{4} * \frac{4}{10}$$

$$= \frac{1}{20}$$

**(3 points)**

$$\because P(A = 0|Y = True)P(B = 0|Y = True)P(C = 1|Y = True)P(Y = True) < P(A = 0|Y = False)P(B = 0|Y = False)P(C = 1|Y = False)P(Y = False) \quad \text{(1 points)}$$

$$\therefore Y = False. \quad \text{(1 points)}$$

### 3 Logistic Regression and Kernels (20 points)

**Review** Recall that the logistic regression model is defined as:

$$p(y = 1|\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + b) \quad (1)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (2)$$

Given a training set  $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$ , where  $\mathbf{x}_n \in \mathbb{R}^{K \times 1}$  and  $y_n \in \{0, 1\}$ , we will minimize the cross-entropy error function to solve  $\mathbf{w}$ .

$$\min_{\mathbf{w}, b} L(\mathbf{w}, b) = \min_{\mathbf{w}, b} - \sum_n \{y_n \log[p(y_n = 1|\mathbf{x}_n)] + (1 - y_n) \log[p(y_n = 0|\mathbf{x}_n)]\} \quad (3)$$

$$= \min_{\mathbf{w}, b} - \sum_n \{y_n \log[\sigma(\mathbf{w}^T \mathbf{x}_n + b)] + (1 - y_n) \log[1 - \sigma(\mathbf{w}^T \mathbf{x}_n + b)]\} \quad (4)$$

17. Consider the dataset consisting of points  $(x, y)$ , where  $x \in \mathbb{R}$  and  $y \in \{0, 1\}$ . Suppose we have three training points  $(x_1, y_1) = (0, 0)$ ,  $(x_2, y_2) = (1, 1)$  and  $(x_3, y_3) = (-1, 1)$ .

Suppose our logistic regression model is  $p(y = 1|x) = \sigma(wx + b)$  with  $b = 1$ . What would be the optimal logistic regression classifier using the training data provided and what is the training accuracy?

$$\min_w L(w) = \min_w - \sum_n \{y_n \log[\sigma(wx_n + b)] + (1 - y_n) \log[1 - \sigma(wx_n + b)]\} \quad (\text{2 points})$$

$$\frac{\partial L(w)}{\partial w} = \sum_n \{[y_n - \sigma(wx_n + b)]x_n\}$$

Set gradient to 0,

$$\Rightarrow 0 = \sum_n \{[y_n - \sigma(wx_n + b)]x_n\}$$

Substitute the data points,

$$0 = 0 + [1 - \sigma(w + 1)] + [1 - \sigma(-w + 1)] * (-1) \quad (\text{2 points})$$

Therefore  $w^* = 0$  and the optimal logistic regression classifier is  $\hat{y} = 1$  (2 points)

Predictions on training data:

$$\hat{y}_1 = \mathbb{I}[\sigma(w^*x_1 + b) > 0.5] = 1 \neq y_1$$

$$\hat{y}_2 = \mathbb{I}[\sigma(w^*x_2 + b) > 0.5] = 1 = y_2$$

$$\hat{y}_3 = \mathbb{I}[\sigma(w^*x_3 + b) > 0.5] = 1 = y_3$$

The training accuracy is  $\frac{2}{3}$ . (4 points)

18. Consider the following function:

$$k(x, x') = \begin{cases} 1 & \text{if } x = x' \\ 0 & \text{otherwise} \end{cases}$$

Prove this is a valid kernel.

(10 points)

Kernel is valid because because gram matrix positive semi-definite, since it is the identity matrix with eigenvalue 1.

- i) Explanation of why Gram matrix is positive semi-definite (PSD), either from the definition of PSD or the fact that eigenvalues are 1 → +4 points
- ii) Gram matrix is PSD → +2 points
- iii) PSD → +2 points
- iv) Any logic → +2 points
- v) Any errors with Explanation in i) → see note in exam for deduction

## 4 K-means clustering (15 points)

Recall the  $K$ -means clustering algorithm. Given a training set  $\{\mathbf{x}_n \in \mathbb{R}^D\}_{n=1}^N$  of  $N$  samples, the  $K$ -means algorithm aims to minimize the distortion measure  $J$ ,

$$J(\{\boldsymbol{\mu}_k\}, \{\gamma_{nk}\}) = \sum_n \sum_k \gamma_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|_2^2 = \sum_n \sum_k \gamma_{nk} \sum_d (x_{nd} - \mu_{kd})^2 \quad (5)$$

w.r.t.  $\{\boldsymbol{\mu}_k \in \mathbb{R}^D\}_{k=1}^K$  and  $\{\gamma_{nk} \in \{0, 1\}\}_{n=1, k=1}^{N, K}$  iteratively, where  $\sum_{k=1}^K \gamma_{nk} = 1, \forall n$ . Alg. 1 outlines the algorithm, where  $\{\boldsymbol{\mu}_k^{(t)}\}_{k=1}^K$  and  $\{\gamma_{nk}^{(t)}\}_{n=1, k=1}^{N, K}$  are the learned parameters after iteration  $t$ .

---

**Algorithm 1:** The  $K$ -means algorithm (with  $K$  clusters)

---

- 1 **Initialization**  $t = 0, \{\boldsymbol{\mu}_k^{(0)} \in \mathbb{R}^D\}_{k=1}^K$
  - 2   **while**  $J$  is strictly decreasing **do**
  - 3      $t \leftarrow t + 1$
  - 4     (a)  $\gamma_{nk}^{(t)} = \begin{cases} 1, & \text{if } k = \arg \min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j^{(t-1)}\|_2^2, \\ 0, & \text{otherwise.} \end{cases}$
  - 4     (b)  $\boldsymbol{\mu}_k^{(t)} = \frac{\sum_{n=1}^N \gamma_{nk}^{(t)} \mathbf{x}_n}{\sum_{n=1}^N \gamma_{nk}^{(t)}} = \frac{\sum_{n: \gamma_{nk}^{(t)}=1} \mathbf{x}_n}{\sum_{n: \gamma_{nk}^{(t)}=1} 1}, \quad \forall k \in \{1, \dots, K\}.$
  - 5 **Output**  $\{\boldsymbol{\mu}_k^{(t)}\}_{k=1}^K$  and  $\{\gamma_{nk}^{(t)}\}_{n=1, k=1}^{N, K}$
- 

19.  **$K$ -means with  $L_1$  norm:** Now we use the  $L_1$  norm instead of  $L_2$  norm in the distortion measure  $J$ , that is,

$$J_{\ell_1}(\{\boldsymbol{\mu}_k\}, \{\gamma_{nk}\}) = \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \|x_n - \mu_k\|_1 = \sum_n \sum_k \gamma_{nk} \sum_d |x_{nd} - \mu_{kd}|,$$

where we sum up the absolute difference across all coordinates. Please derive the corresponding update rules of  $\gamma_{nk}$ , and  $\boldsymbol{\mu}_k$  (Line 3 and 4 in Alg. 1) for the distortion  $J_{\ell_1}$ . Answer for step (b) might not be unique, please write out one feasible answer. **(10 points)**

$$(a) \gamma_{nk}^{(t)} = \begin{cases} 1, & \text{if } k = \arg \min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j^{(t-1)}\|_1, \\ 0, & \text{otherwise.} \end{cases} \quad (5 \text{ points})$$

- i)  $\gamma_{nk} = 1$  correct  $\rightarrow 3$
- ii)  $\gamma_{nk} = 0$  correct  $\rightarrow 2$
- iii) unclear distance  $L_1$  or  $L_2 \rightarrow -1$

$$(b) \boldsymbol{\mu}_k^{(t)} = \text{median}_{n: \gamma_{nk}^{(t)}=1} \{\mathbf{x}_n\}, \quad \forall k \in \{1, \dots, K\} \quad (\text{coordinate-wise median}) \quad (5 \text{ points})$$

- i) mention the idea of median and provide final form  $\rightarrow 5$
- ii) mention the idea of median  $\rightarrow 2$

- iii) not mention median of data “with  $\gamma_{nk} = 1$ ” → -1
- iv) not provide final form → -2
- v) wrong update form → -3

20. ***K*-means with outliers:** If your data contains outliers, which version of K-means would you use: the  $L_1$  norm one or the original one with  $L_2$  norm? Explain your answer. **(5 points)**

*$L_1$  norm is better for data with outliers, because the median is robust to outliers. The median only cares about picking the middle point in the ordering. And the furthest points, possibly outliers, would not effect the cluster centroid.*

- i)  $L_1$  is better → 3.
- ii) The *median* is robust to outliers → 2. (To get full credit, the answer must explicitly mention median.)

## 5 Neural Network (17 points)

21. Consider the following convolutional neural network. A  $32 \times 32 \times 3$  image input, followed by a convolution layer with 2 filters of size  $5 \times 5$  (stride 1, no zero-padding), then another convolution layer with 3 filters of size  $5 \times 5$  (stride 1, no zero-padding), and finally an average-pooling layer with a  $2 \times 2$  filter (stride 2, no zero-padding).

- i. How many parameters do we need to learn for this network? (3 points)

$$(5 \times 5 \times 3 + 1) \times 2 + (5 \times 5 \times 2 + 1) \times 3 = 152 + 153 = 305$$

- i) 305 or 300 (without bias)  $\rightarrow$  3 points  
 ii) any errors  $\rightarrow$  -1 point for each

- ii. What is the picture final dimension? (3 points)

$$12 \times 12 \times 3$$

- i)  $12 \times 12 \times 3 \rightarrow$  3 points  
 ii) any errors  $\rightarrow$  -1 point for each

22. Suppose we have a neural network defined below: (11 points)

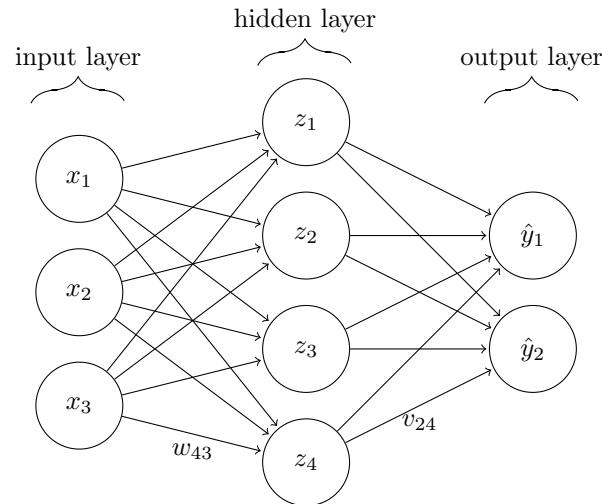


Figure 1: A neural network with one hidden layer.

$$\text{input layer} \quad x_i, \quad (6)$$

$$\text{hidden layer} \quad z_k = \arctan \left( \sum_{i=1}^3 w_{ki} x_i \right), \quad (7)$$

$$\text{output layer} \quad \hat{y}_j = \sum_{k=1}^4 v_{jk} z_k, \quad (8)$$

$$\text{loss function} \quad L(y, \hat{y}) = \sqrt{\frac{1}{2} ((\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2)}. \quad \hat{y}_j \text{ is prediction, } y_j \text{ is ground truth} \quad (9)$$

Write down  $\frac{\partial L}{\partial v_{jk}}$  and  $\frac{\partial L}{\partial w_{ki}}$  in terms of only  $x_i$ ,  $z_k$ ,  $y_j$ ,  $\hat{y}_j$ ,  $w_{ki}$ , and/or  $v_{jk}$ .

Hint:  $\frac{\partial \arctan(\alpha)}{\partial \alpha} = \frac{1}{\alpha^2 + 1}$ .

$$\begin{aligned}
\frac{\partial L}{\partial v_{jk}} &= \frac{\partial L}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial v_{jk}} \\
\beta &= \sqrt{\frac{1}{2} ((\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2)} \\
\frac{\partial L}{\partial \hat{y}_1} &= \frac{\partial}{\partial \hat{y}_1} \beta(\hat{y}_1) \\
&= \frac{1}{2\beta} (\hat{y}_1 - y_1) \\
\frac{\partial \hat{y}_j}{\partial v_{jk}} &= \sum_{k=1}^4 v_{jk} z_k = z_k \\
\rightarrow \frac{\partial L}{\partial v_{jk}} &= \frac{1}{2\beta} (\hat{y}_j - y_j) z_k
\end{aligned}$$

$$\begin{aligned}
\frac{\partial L}{\partial w_{ki}} &= \frac{\partial L}{\partial z_k} \frac{\partial z_k}{\partial w_{ki}} \\
\frac{\partial L}{\partial z_k} &= \frac{\partial L}{\partial \hat{y}_1} \frac{\partial \hat{y}_1}{\partial z_k} + \frac{\partial L}{\partial \hat{y}_2} \frac{\partial \hat{y}_2}{\partial z_k} \\
&= \sum_{j=1}^2 \frac{1}{2\beta} (\hat{y}_j - y_j) v_{jk} \\
\frac{\partial z_k}{\partial w_{ki}} &= \frac{\partial}{\partial w_{ki}} \arctan \left( \sum_{i=1}^3 w_{ki} x_i \right) \\
&= \frac{x_i}{\tan^2(z_k) + 1} \\
\rightarrow \frac{\partial L}{\partial w_{ki}} &= \left( \sum_{j=1}^2 \frac{1}{2\beta} (\hat{y}_j - y_j) v_{jk} \right) \frac{x_i}{\tan^2(z_k) + 1}
\end{aligned}$$

The rationale behind your grade for this question is reported on your exam.

You may use this page as scratch paper

You may use this page as scratch paper

## CSCI-567 Summer 2019 Midterm Exam

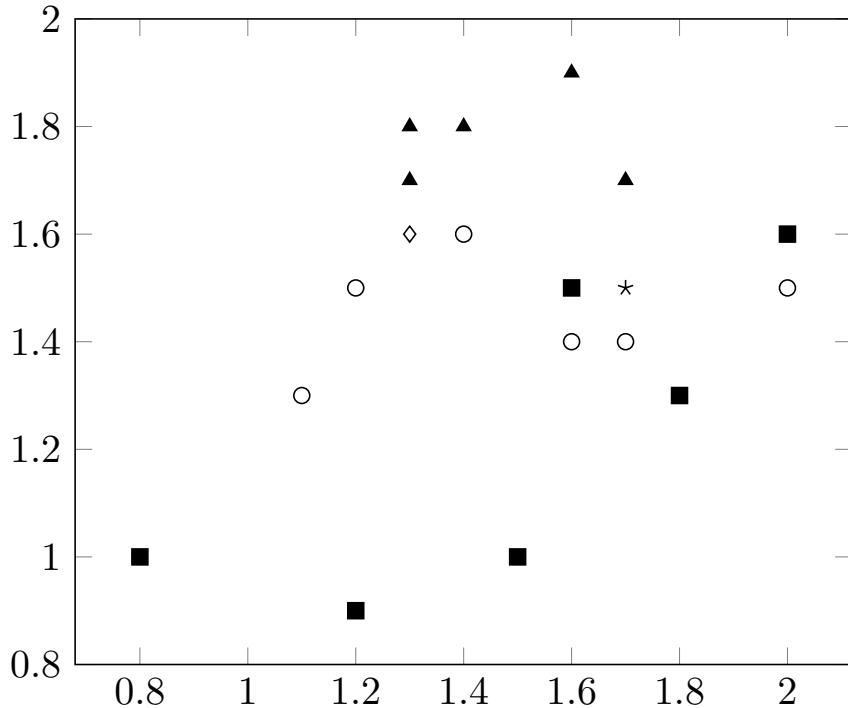
| Problem | 1 | 2 | 3 | 4 | 5 | Total |
|---------|---|---|---|---|---|-------|
| Points  |   |   |   |   |   | 100   |

Please read the following instructions carefully:

- The exam has a total of **15 pages** (including this cover and two blank pages in the end). Each problem have several questions. Once you are permitted to open your exam (and not before), you should check and make sure that you are not missing any pages.
- Duration of the exam is **2 hrs and 50 mins**. Questions are not ordered by their difficulty. Budget your time on each question carefully.
- Select **one and only one answer** for all multiple choice questions.
- Answers should be **concise** and written down **legibly**. All questions can be done within 5-12 lines.
- You must answer each question on the page provided. You can use the last two blank pages as scratch paper. Raise your hand to ask a proctor for more if needed.
- This is a **closed-book/notes** exam. Consulting any resources is NOT permitted.
- Any kind of cheating will lead to **score 0** for the entire exam and be reported to SJACS.
- You **may not** leave your seat **for any reason** unless you submit your exam at that point.

## 1 Multiple Choice Questions (20 points)

Questions 1 through 4 deal with the following data, where squares, triangles, and open circles are three different classes of data in the training set and the diamond ( $\diamond$ ) and the star (\*) are test points.



1. For  $k = 2$ , what will be the label for diamond test point?

- (a) Triangle
- (b) Square
- (c) Circle
- (d) Can't be determined

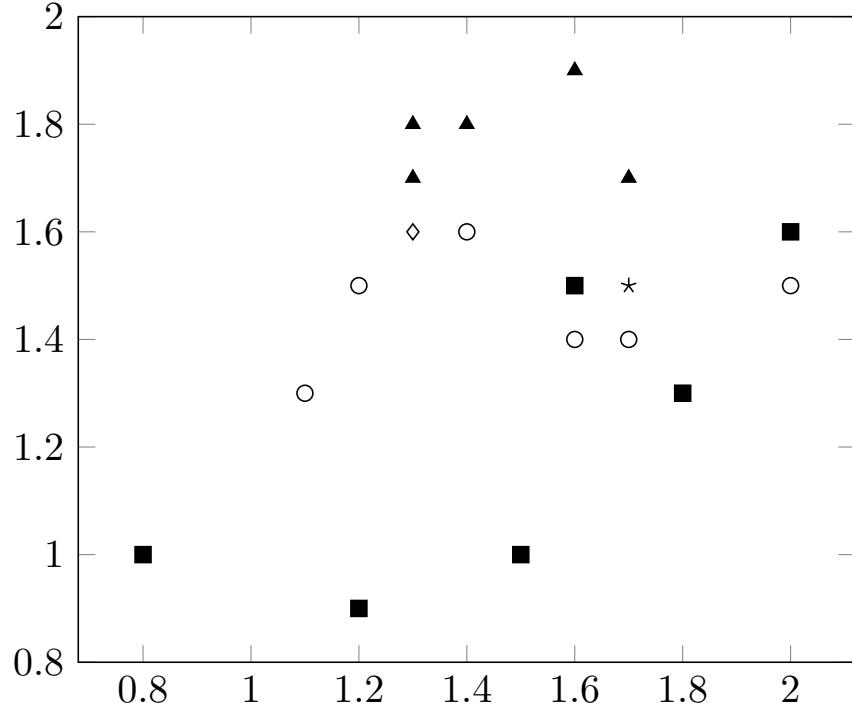
**Solution:** Can't be determined, as there is a tie

2. What is the minimum  $k$  for which diamond test point will get classified as Triangle?

- (a) 7
- (b) 3
- (c) 5
- (d) Not possible for any  $k$

**Solution:** 5

Following is same figure from previous page.



3. For  $k = 4$ , what is the label for star test point?

- (a) Circle
- (b) Square
- (c) Triangle
- (d) Can't be determined

**Solution:** Circle

4. What is the minimum  $k$  for which star test point will get classified as Triangle?

- (a) 7
- (b) 3
- (c) 5
- (d) Not possible for any  $k$

**Solution:** Not possible

For question 5 to 7 consider the following data and decision tree.

| A | B | C | Class |
|---|---|---|-------|
| 0 | 0 | 0 | +     |
| 0 | 0 | 1 | +     |
| 0 | 1 | 0 | +     |
| 0 | 1 | 1 | -     |
| 1 | 0 | 0 | +     |
| 1 | 0 | 0 | +     |
| 1 | 1 | 0 | -     |
| 1 | 0 | 1 | +     |
| 1 | 1 | 0 | -     |
| 1 | 1 | 0 | -     |

| A | B | C | Class |
|---|---|---|-------|
| 0 | 0 | 0 | +     |
| 0 | 1 | 1 | +     |
| 1 | 1 | 0 | +     |
| 1 | 0 | 1 | -     |
| 1 | 0 | 0 | +     |

Figure 1: Training Data    Figure 2: Validation Data

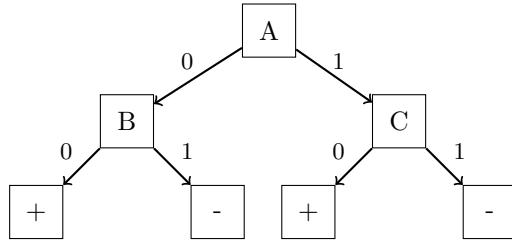


Figure 3: Decision Tree

5. Using the given decision tree, what is the training error?

- (a)  $\frac{1}{10}$
- (b)  $\frac{2}{10}$
- (c)  $\frac{3}{10}$
- (d)  $\frac{4}{10}$
- (e)  $\frac{5}{10}$
- (f)  $\frac{6}{10}$

wrong - 3,  
1111|

**Solution:** 5/10

6. Using the given decision tree, what is the validation error?

- (a)  $\frac{1}{5}$
- (b)  $\frac{2}{5}$
- (c)  $\frac{3}{5}$
- (d)  $\frac{4}{5}$

|

**Solution:** 1/5, only (0,1,1) is wrongly classified

7. To improve the validation performance, we decided to prune the decision tree. Pruning which node will improve validation set performance

- (a) B
- (b) C

(c) B and C

(d) None

**Solution:** Node B, as note that removing node B will make the decision tree make no error on validation set.

Questions 8 through 11 deal with the following data. Assume  $\mathbf{x} = [x_1, x_2]$  is the input and  $y$  is the output and we are trying to learn a linear regression model using this data with weights  $\mathbf{w}$  and bias  $b$ .

| $x_1$ | $x_2$ | $y$  |
|-------|-------|------|
| 0     | .5    | 1.7  |
| 0     | 1.5   | 2.7  |
| 2     | 1.3   | 6.5  |
| 0     | 3     | 4.2  |
| 3.5   | 2     | 10.2 |
| 0     | 1     | 2.2  |

Figure 4: Training Data

| $x_1$ | $x_2$ | $y$ |
|-------|-------|-----|
| 0     | 0     | 1.1 |
| 1     | 1.5   | 4.7 |
| 2     | .5    | 6.0 |
| 0.5   | 1.5   | 3.4 |

Figure 5: Validation Data

8. What is the value of  $\mathbf{w}, b$  when training the model with least squares objective without any regularization?

- (a)  $[1, 2], 1.2$
- (b)  $[2, 1], 1.2$
- (c)  $[1, 2], 1.3$
- (d)  $[2, 1], 1.0$

**Solution:**  $[2, 1], 1.2$

9. What is the mean absolute error on validation data?

- (a) 0.0375
- (b) 0.0475
- (c) 0.7
- (d) 0.175

**Solution:**  $0.175 = \frac{1}{4}(0.1 + 0 + 0.3 + 0.3)$

10. Let us define error between target  $y$  and prediction  $\hat{y}$  as

$$\text{Error}(\hat{y}, y) = \begin{cases} 0 & \text{if } |\hat{y} - y| \leq 0.15 \\ |\hat{y} - y| - 0.15 & \text{otherwise} \end{cases}$$

What is the mean error?

- (a) 0.7
- (b) 0.15
- (c) 0.075
- (d) 0.3

**Solution:**  $0.075 = \frac{1}{4}(0 + 0 + .15 + .15)$

Following is same data from previous page.

| $x_1$ | $x_2$ | $y$  |
|-------|-------|------|
| 0     | .5    | 1.7  |
| 0     | 1.5   | 2.7  |
| 2     | 1.3   | 6.5  |
| 0     | 3     | 4.2  |
| 3.5   | 2     | 10.2 |
| 0     | 1     | 2.2  |

Figure 6: Training Data

| $x_1$ | $x_2$ | $y$ |
|-------|-------|-----|
| 0     | 0     | 1.1 |
| 1     | 1.5   | 4.7 |
| 2     | .5    | 6.0 |
| 0.5   | 1.5   | 3.4 |

Figure 7: Validation Data

11. What would be the Gram Matrix for the validation data if we use the following transform function  $\phi(\mathbf{x}) = [1, x_1, x_2, x_1x_2 + 1]$

$$(a) \begin{bmatrix} 2 & 3.5 & 3 & 2.75 \\ 3.5 & 10.5 & 8.75 & 8.125 \\ 3 & 8.75 & 9.25 & 6.25 \\ 2.75 & 8.125 & 6.25 & 6.5625 \end{bmatrix}$$

$$(b) \begin{bmatrix} 2 & 3.25 & 3 & 2.75 \\ 3.25 & 10.75 & 8.5 & 8.125 \\ 3 & 8.5 & 9.25 & 6.25 \\ 2.75 & 8.125 & 6.25 & 6.5625 \end{bmatrix}$$

$$(c) \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 6.5 & 5.25 & 4.875 \\ 1 & 5.25 & 6.25 & 3.5 \\ 1 & 4.875 & 3.5 & 4.0625 \end{bmatrix}$$

$$(d) \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 6.25 & 5.5 & 4.875 \\ 1 & 5.5 & 6.25 & 3.5 \\ 1 & 4.875 & 3.5 & 4.0625 \end{bmatrix}$$

**Solution:**  $\begin{bmatrix} 2 & 3.5 & 3 & 2.75 \\ 3.5 & 10.5 & 8.75 & 8.125 \\ 3 & 8.75 & 9.25 & 6.25 \\ 2.75 & 8.125 & 6.25 & 6.5625 \end{bmatrix}$

$$k(\mathbf{x}_1, \mathbf{x}_2) = 2 + \mathbf{x}_1^T \mathbf{x}_2 + x_{11}x_{12}x_{21}x_{22} + x_{11}x_{12} + x_{21}x_{22}$$

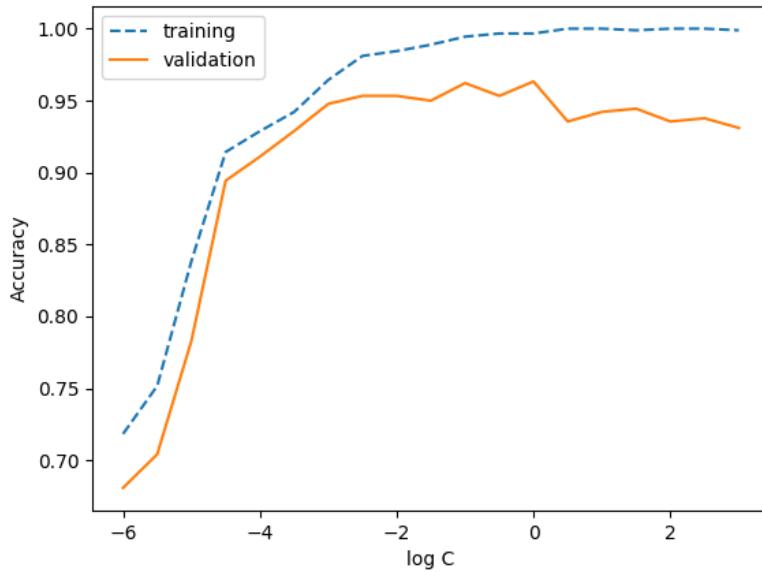


Figure 8: Training-validation curve for problem 12

12. In fig. 8 is shown training and validation curve for tuning some hyperparameter  $C$  of the model. Based on the figure, which of the following would be a good choice for  $\log C$ ?

- (a) -4
- (b) -3
- (c) 0.5
- (d) 2.0

**Solution:** value -3

13. In fig. 8 as  $C$  increases model is .....

- (a) overfitting
- (b) underfitting
- (c) can't be determined

**Solution:** overfitting, as gap between training and validation error is increasing

14. Let  $k(\mathbf{x}, \mathbf{y}), \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$  be a valid kernel, for what values of  $c$  is  $ck(\mathbf{x}, \mathbf{y})$  a valid kernel function.

- (a)  $c \geq 0$
- (b)  $c > 0$
- (c)  $c < 0$
- (d)  $c \in \mathbb{R}$

**Solution:**  $c \geq 0$ , All zero is valid kernel as it is symmetric and Gram Matrix is PSD. It might not be a useful kernel though

15. which of the following penalty cannot be a good idea to regularize model complexity?

- (a)  $R(\mathbf{w}) = \exp\{\sum_i |w_i|\}$
- (b)  $R(\mathbf{w}) = \exp\{-\sum_i |w_i|\}$
- (c)  $R(\mathbf{w}) = -\sum_i \log((|w_i| + 1)^{-1})$
- (d)  $R(\mathbf{w}) = \sum_i \exp\{|w_i|\}$

**Solution:**  $R(\mathbf{w}) = \exp\{-\sum_i |w_i|\}$

16. Suppose a convolution layer takes a  $4 \times 6 \times 3$  image as input and outputs a  $3 \times 4 \times 6$  tensor. Which of the following is a possible configuration of this layer?

- (a) Two  $2 \times 4 \times 3$  filters, stride 1, no zero-padding.
- (b) Two  $1 \times 1 \times 3$  filters, stride 2, 1 zero-padding.
- (c) Six  $2 \times 4 \times 3$  filters, stride 1, no zero-padding.
- (d) Six  $1 \times 1 \times 3$  filters, stride 2, 1 zero-padding.

**Solution:** D

17. Given a k-nearest neighbour model is trained on  $N$  training points using euclidean distance as distance criteria and each sample is  $d$  dimensional. What is time complexity of testing the model? (There are  $M$  samples in test data).

- (a)  $\mathcal{O}(MNd^2k)$
- (b)  $\mathcal{O}(MN^2dk)$
- (c)  $\mathcal{O}(MNd़k)$
- (d)  $\mathcal{O}(M^2Ndk)$

**Solution:**  $\mathcal{O}(MNd़k)$

18. Suppose we are training a neural network with batch-SGD, batch size 100, and there are 50000 training samples. How many updates would be while training during 1 epoch?

- (a) 500
- (b) 1
- (c) Can't be determined
- (d) 100

**Solution:**  $50000/100 = 500$

19. Recall, a one hidden layer neural network with ReLU non-linearity can perfectly fit a XOR function. What is the minimum number of hidden neurons required to fit the XOR function?

| $x_1$ | $x_2$ | $y$ |
|-------|-------|-----|
| 0     | 0     | 0   |
| 0     | 1     | 1   |
| 1     | 0     | 1   |
| 1     | 1     | 0   |

Table 1: Truth table for XOR function

- (a) 4
- (b) 2
- (c) 1
- (d) 8

**Solution:** 2 units are required.

First layer weights  $\mathbf{W} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ ,  $b = [0, 0]^T$  and then relu activation will transform inputs

$$\mathbf{X} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

to.

$$\mathbf{Z} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Now you can multiply this by positive weights to get the result.

20. Recall that the Gini impurity of a distribution  $p$  over a set of  $K$  items is defined as  $\sum_{k=1}^K p(k)(1-p(k))$ . Which of the following distributions has the largest Gini impurity?

- (a) (0.25, 0.25, 0.25, 0.25)

*Because it is like a forest  
a coin is this region*

(b)  $(0.2, 0.3, 0.5, 0)$

(c)  $(1, 0, 0, 0)$

(d)  $(.5, .5, 0, 0)$

**Solution:**  $(0.25, 0.25, 0.25, 0.25)$

## 2 Naive Bayes (20 points)

Suppose we want to use the Naive Bayes model to analyze the restaurant reviews which are crawled from online customer review websites. Now we preprocessed the text reviews as tabular features. The training data is shown in the following table. Each sample has three attributes (Food Taste, Price, Environment), where Food Taste  $\in \{Unappetizing(0), Delicious(1)\}$ , Price  $\in \{Expensive(0), Normal(1), Cost-Efficient(2)\}$ , Environment  $\in \{Bad(0), Good(1)\}$ . We hope this model can automatically classify users' future reviews into satisfaction rank value (A or B).

| Food Taste ( $X_1$ ) | Price( $X_2$ ) | Environment( $X_3$ ) | Satisfaction(Y) |
|----------------------|----------------|----------------------|-----------------|
| 1                    | 2              | 1                    | A               |
| 1                    | 1              | 1                    | A               |
| 0                    | 2              | 0                    | A               |
| 0                    | 2              | 1                    | B               |
| 1                    | 0              | 1                    | A               |
| 1                    | 1              | 0                    | B               |

21. What is the probability of  $P(\text{Price} = \text{Cost-Efficient} | \text{Satisfaction} = A)$ ? (4 points) **Solution:**  
 $P(X_2 = 2 | Y = A) = \frac{1}{2}$

22. if the a user's review is  $\{Delicious(1), Cost-Efficient(2), Bad(0)\}$  How would a naive Bayes classifier predict the satisfaction rank value ? (8 points)

**Solution:**

$$\begin{aligned} \underset{c \in [A, B]}{\operatorname{argmax}} P(y=c | \mathbf{x}) &= \underset{c \in [A, B]}{\operatorname{argmax}} P(X_1=1, X_2=2, X_3=0 | Y=c) P(Y=c) \\ &= \underset{c \in [A, B]}{\operatorname{argmax}} P(X_1=1 | Y=c) P(X_2=2 | Y=c) P(X_3=0 | Y=c) P(Y=c) \end{aligned}$$

When  $c = A$ , the possibility is  $\frac{1}{16}$ ; When  $c = B$ , the possibility is  $\frac{1}{24}$ . Thus, the prediction should be A.

23. Suppose we have a new sample  $\{Delicious(1), Normal(1), Bad(0)\}$  without label (the satisfaction rank value). What is the probability of that its label is A. (8 points)

**Solution:**

$$\begin{aligned}
 P(Y = A | X_1 = 1, X_2 = 1, X_3 = 0) &= \frac{P(X_1 = 1 | Y = A)P(X_2 = 1 | Y = A)P(X_3 = 0 | Y = A)P(Y = A)}{P(X_1 = 1, X_2 = 1, X_3 = 0)} \\
 &= \frac{P(X_1 = 1 | Y = A)P(X_2 = 1 | Y = A)P(X_3 = 0 | Y = A)P(Y = A)}{\sum_{c \in [A, B]} P(Y = c)P(X_1 = 1, X_2 = 1, X_3 = 0 | Y = c)} \\
 &= \frac{P(X_1 = 1 | Y = A)P(X_2 = 1 | Y = A)P(X_3 = 0 | Y = A)P(Y = A)}{\sum_{c \in [A, B]} P(Y = c)P(X_1 = 1 | Y = c)P(X_2 = 1 | Y = c)P(X_3 = 0 | Y = c)} = \frac{3}{7}
 \end{aligned}$$

### 3 Kernels (20 points)

Assume  $\phi_i : \mathbb{R}^d \rightarrow \mathbb{R}_i^k$ ,  $i = \{1, 2\}$  is some fixed transform,  $k_i(\mathbf{x}, \mathbf{z}) = \phi_i(\mathbf{x})^T \phi_i(\mathbf{z})$ ,  $i = \{1, 2\}$  are valid kernel functions and  $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$ .

24. Prove or disprove that following kernels valid.

(a)  $k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) + k_2(\mathbf{x}, \mathbf{z})$  (5 points)

(b)  $k(\mathbf{x}, \mathbf{z}) = f(x)k_1(\mathbf{x}, \mathbf{z})f(\mathbf{z})$ , where  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  is some function (5 points)

(c)  $k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{z}$ , where  $\mathbf{A} \in \mathbb{R}^{k \times d}$  (5 points)

(d)  $k(\mathbf{x}, \mathbf{z}) = 1 + \sqrt{2}\mathbf{x}^T \mathbf{z} + \sqrt{2}\exp\{-\gamma\|\mathbf{x} - \mathbf{z}\|^2\} + 5\mathbf{x}^T \mathbf{z}\exp\{-\gamma(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2)\}$ , where  $\gamma \in \mathbb{R}$  is some constant (5 points)

**Solution:**

- a.  $\phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x})]$
- b.  $\phi(\mathbf{x}) = f(\mathbf{x})\phi_1(\mathbf{x})$
- c.  $\phi(\mathbf{x}) = \mathbf{Ax}$
- d. Sum of 4 kernel functions.

For first two terms :  $\phi(x) = [1, 2^{25}x]$

For the last term, see that,  $5\mathbf{x}^T \mathbf{z}\exp\{-\gamma\|\mathbf{x}\|^2 - \gamma\|\mathbf{z}\|^2\} = 5\exp\{-\gamma\|\mathbf{x}\|^2\}\mathbf{x}^T \mathbf{z}\exp\{-\gamma\|\mathbf{z}\|^2\}$  and use result from b

For third term,  $\sqrt{2}\exp\{-\gamma\|\mathbf{x} - \mathbf{z}\|^2\} = 2^{25}\exp\{-\gamma\|\mathbf{x}\|^2\}2^{25}\exp\{2\gamma\mathbf{x}^T \mathbf{z}\}\exp\{-\gamma\|\mathbf{z}\|^2\}$

If  $\exp\{2\gamma\mathbf{x}^T \mathbf{z}\}$  is a kernel then it is valid kernel by (b).

$$\exp\{2\gamma\mathbf{x}^T \mathbf{z}\} = \sum_{i=0}^{\infty} \frac{(2\gamma\mathbf{x}^T \mathbf{z})^i}{i!}$$

This is kernel since,  $(\mathbf{x}^T \mathbf{z})^d$  is a kernel fn for  $d \geq 0$  and to have non-negative coeff  $\gamma \geq 0$

**Rubric:**

1. 1 point if you only mention identities but do not use them correctly.
2. Partial points for partially correct approaches.
  - (a) Realizing how to prove correctly gets 2 points
  - (b) 2 pts only if you prove for 2 dimension
  - (c) 2 points if you prove kernel fn psd and not Gram matrix

## 4 Neural Network (20 points)

Consider the following convolutional neural network, LeNet-5 (C1, S2, C3, S4, C5, F6, O1). LeNet takes color image of size (32, 32, 3) as input and outputs a prediction vector of probabilities for 10 classes.

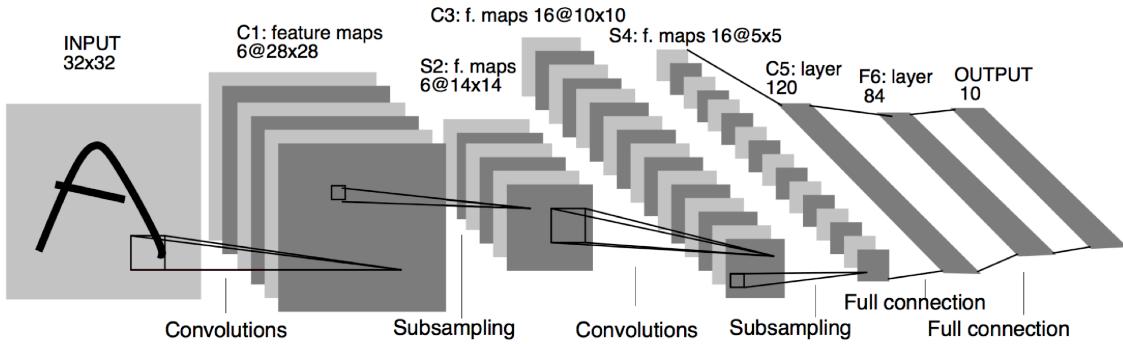


Figure 9: Architecture of LeNet-5, a Convolutional Neural Network. (LeCun et al. 1998)

25. For the first convolutional layer, we observe that the shape of input data is (32, 32, 3) and the shape of output data is (28, 28, 6) where 6@28 × 28 means the feature maps have 6 channels. Assuming there is no spatial padding for this layer and the stride is 1, what is the size of its convolutional kernel (also called filter). Please answer it in format (number of input channels, height, width, number of kernels).

**Solution:**

$$(3, 5, 5, 6)$$

Rubric: 2 points

26. Assuming there is no spatial padding for all convolutional layers and their stride is 1, and C5, F6 are fully connected layers without biases. F6 and the output are also fully connected. How many parameters do we need to learn for this network? (Your answer should tell your calculation process step by step).

**Solution:**

$$\text{Convolution 1: } 3 \times 5 \times 5 \times 6 = 450$$

$$\text{Convolution 2: } 6 \times 5 \times 5 \times 16 = 2400 \text{ text (Rubric: 2 points)}$$

$$\text{Fully Connected C5: } 16 \times 5 \times 5 \times 120 = 48000 \text{ text (Rubric: 2 points)}$$

$$\text{Fully Connected F6: } 120 \times 84 = 10080$$

$$\text{Fully Connected Output: } 84 \times 10 = 840$$

The total parameter number is: 61770 text (Rubric: 2 points)

Suppose we have a Multi-Class Neural Networks defined below. An illustration is provided in Fig. 10. Please answer the following questions.

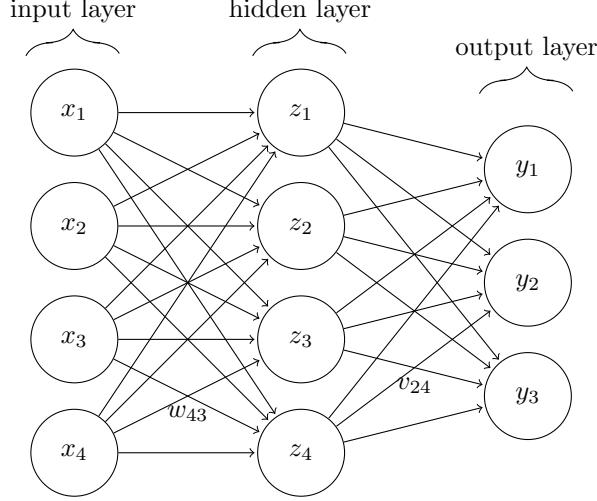


Figure 10: A neural network with one hidden layer.

**Forward Propagation.** For multi-class classification, we use softmax layer with cross-entropy loss as output. In the hidden layer, we use *tanh* activation function. The forward propagation can be expressed as:

$$\text{input layer} \quad x_i, \quad (1)$$

$$\text{hidden layer} \quad z_k = \tanh \left( \sum_{i=1}^4 w_{ki} x_i \right), \quad \tanh(\alpha) = \frac{e^\alpha - e^{-\alpha}}{e^\alpha + e^{-\alpha}} \quad (2)$$

$$\text{output layer} \quad \hat{y}_j = \text{softmax}(o_j) = \frac{\exp(o_j)}{\sum_{i=1}^3 \exp(o_i)}, \quad \text{where } o_j = \sum_{k=1}^4 v_{jk} z_k \quad (3)$$

$$\text{loss function} \quad L(y, \hat{y}) = - \sum_{j=1}^3 y_j \log \hat{y}_j, \quad \text{where } \hat{y}_j \text{ is prediction, } y_j \text{ is ground truth} \quad (4)$$

27. **Backpropagation** Please write down  $\frac{\partial L}{\partial v_{jk}}$  and  $\frac{\partial L}{\partial w_{ki}}$  in terms of only  $x_i$ ,  $z_k$ ,  $o_j$ ,  $y_j$ , and/or  $\hat{y}_j$  using backpropagation.

**Solution:**

$$\frac{\partial L}{\partial v_{jk}} = \frac{\partial L}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial o_j} \frac{\partial o_j}{\partial v_{jk}} \quad 1 \text{ point}$$

$$\begin{aligned} \frac{\partial L}{\partial \hat{y}_j} &= \frac{\partial}{\partial \hat{y}_1} \left[ -\sum_{j=1}^3 y_j \log \hat{y}_j \right] \\ &= -\frac{y_j}{\hat{y}_j} \end{aligned} \quad 2 \text{ points}$$

$$\begin{aligned} \frac{\partial \hat{y}_j}{\partial o_j} &= \frac{\partial}{\partial o_j} \left[ \frac{\exp(o_j)}{\sum_{i=1}^3 \exp(o_i)} \right] \\ &= \frac{g'(x)h(x) - h'(x)g(x)}{[h(x)]^2}, \text{ where } g(x) = \exp(o_j), h(x) = \sum_{i=1}^3 \exp(o_i) \end{aligned} \quad 2 \text{ points}$$

$$\frac{\partial o_j}{\partial v_{jk}} = \frac{\partial \sum_{k=1}^4 v_{jk} z_k}{\partial v_{jk}} = z_k \quad 2 \text{ points}$$

$$\rightarrow \frac{\partial L}{\partial v_{jk}} = \left( y_j \frac{\exp(o_j)}{\sum_k \exp(o_k)} - y_j \right) z_k = (y_j \hat{y}_j - y_j) z_k$$

$$\frac{\partial L}{\partial w_{ki}} = \frac{\partial L}{\partial z_k} \frac{\partial z_k}{\partial w_{ki}} \quad 1 \text{ point}$$

$$\begin{aligned} \frac{\partial L}{\partial z_k} &= \frac{\partial L}{\partial o_1} \frac{\partial o_1}{\partial z_k} + \frac{\partial L}{\partial o_2} \frac{\partial o_2}{\partial z_k} + \frac{\partial L}{\partial o_3} \frac{\partial o_3}{\partial z_k} \\ &= \sum_{j=1}^3 (y_j \hat{y}_j - y_j) v_{jk} \end{aligned} \quad 2 \text{ points}$$

$$\begin{aligned} \frac{\partial z_k}{\partial w_{ki}} &= \frac{\partial}{\partial w_{ki}} \tanh \left( \sum_{i=1}^3 w_{ki} x_i \right) \\ &= (1 - z_k^2) x_i \end{aligned} \quad 2 \text{ points}$$

$$\rightarrow \frac{\partial L}{\partial w_{ki}} = \left( \sum_{j=1}^3 (y_j \hat{y}_j - y_j) v_{jk} \right) (1 - z_k^2) x_i$$



Name:

USC ID:

**Notes:**

- Write your name and ID number in the solution you submit.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.
- The exam has 5 questions, 9 pages, and 13 points extra credit.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 22    |        |
| 2       | 25    |        |
| 3       | 22    |        |
| 4       | 22    |        |
| 5       | 22    |        |
| Total   | 113   |        |

Name:

USC ID:

**Notes:**

- Write your name and ID number in the solution you submit.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.
- The exam has 5 questions, 9 pages, and 13 points extra credit.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 22    |        |
| 2       | 25    |        |
| 3       | 22    |        |
| 4       | 22    |        |
| 5       | 22    |        |
| Total   | 113   |        |

1. Assume that we built a linear regression model with  $n = 22$  observations and  $p = 5$  predictors. Determine the minimum value of  $R^2$  for which at least one of the predictors is statistically significant when  $\alpha = 0.01$ .

$$n = 22, p = 5, \alpha = 0.01$$

$$R^2 = 1 - \frac{RSS}{TSS}$$

$$f_{p, n-p-1} = \frac{(TSS - RSS) / p}{\frac{RSS}{TSS} (n - p - 1)}.$$

$$f_{p, n-p-1} = 4.437$$

from table

$$= \frac{\frac{TSS - RSS}{TSS} \times (n - p - 1)}{\frac{RSS}{TSS} \times p}$$

$$4.437 = \frac{R^2}{1 - R^2} \cdot \frac{(16)}{5}$$

$$1 - R^2 = 0.721 R^2$$

$$1 = 1.7212 R^2 \Rightarrow R^2 = 0.58099$$

2. Choose either T (True) or F (False) (no need to explain why):

- (a) When the assumption of conditional independence of features holds, the Naïve Bayes' classifier provides the best accuracy among all possible classifiers.  T  F
- (b) The F1 score is not an appropriate measure for evaluating binary classifiers when data are not imbalanced.  T  F
- (c) Leave-One-Out Cross Validation has less bias in estimating the error of a classifier for a large data set than 5 fold cross validation.  T  F
- (d) When classifying imbalanced data into two classes, we can decrease the threshold on class conditional probability  $\Pr(Y = k | X_1 = x_1, \dots, X_p = x_p)$  to increase the true positive rate at the expense of increasing the false negative rate.  T  F
- (e) Logistic regression assumes that the conditional odds of the outcome  $Y$  given the features,  $\mathbb{O}[Y = k | X_1 = x_1, \dots, X_p = x_p]$ , is a logistic function of the features.  T  F

3. Assume that in a binary classification problem with one feature  $X$ , the distribution of  $X$  in class  $k = 1$  is

$$f_1(x) = \frac{x}{\sigma_1^2} \exp\left(\frac{-x^2}{2\sigma_1^2}\right), x \geq 0$$

and the distribution of  $X$  in class  $k = 2$  is

$$f_2(x) = \frac{1}{x\sqrt{2\pi\sigma_2}} \exp\left(\frac{-(\ln x - \mu_2)^2}{2\sigma_2^2}\right), x \geq 0$$

- (a) Are there any conditions under which the discriminant function is a linear function of  $x$ ?
- (b) If  $\sigma_1 = \sigma_2 = 1$ ,  $\mu_2 = 10$ , and  $\pi_1 = \pi_2 = 0.5$ , in what class will  $x = 10$  be classified?

$$P_k(x) = \frac{\pi_k \cdot f_k(x)}{\sum_{l=1}^k \pi_l \cdot f_l(x)}$$

$$\left( \frac{x^2}{2\sigma_1^2} \right)$$

$$P_1(x) = \frac{\pi_1 \cdot \frac{x}{\sigma_1^2} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}}{\sum_{l=1}^k \pi_l \cdot f_l(x)}.$$

$$-\frac{(x - \mu_2)^2}{2\sigma_2^2}$$

$$P_2(x) = \frac{\pi_2 \cdot \frac{1}{x\sqrt{2\pi\sigma_2}} e^{-\frac{-(\ln x - \mu_2)^2}{2\sigma_2^2}}}{\sum_{l=1}^k \pi_l \cdot f_l(x)}$$

$$\delta_1(a) = \log \pi_1 + \log\left(\frac{a}{\sigma_1}\right) - \frac{a^2}{2\sigma_1^2}$$

cannot be a linear function in  $n$

$$\begin{aligned}\delta_2(a) &= \log \pi_2 - \log n - \log(\sqrt{2\pi}) - \log \sigma_2 \\ &\quad - \frac{(ln n - \mu_2)^2}{2\sigma_2^2}\end{aligned}$$

$\delta_2(n)$  is a linear function of  $n$

if we consider

$$b) \quad \sigma_1 = \sigma_2 = 1, \mu_1 = 10, \mu_2 = 5 \quad \& \quad \pi_1 = \pi_2 = 0.5$$

$$x = 10?$$

$$\left( -\frac{x^2}{2\sigma_1^2} \right)$$

$$P_1(x) = \pi_1 \cdot \frac{x}{\sigma_1^2} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}$$

$$\sum_{l=1}^k \pi_l \cdot f_l(x)$$

$$P_1(x) = 0.5 \cdot \frac{10}{1} \cdot e^{-\frac{(10-10)^2}{2}} = 5 \cdot e^{-50}$$

constant

$$-\frac{(x - \mu_2)^2}{2\sigma_2^2}$$

$$P_2(x) = \pi_2 \cdot \frac{1}{x \sqrt{2\pi} \sigma_2} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}$$

$$\sum_{l=1}^k \pi_l \cdot f_l(x)$$

$x = 10$

$$-\frac{(10 - 5)^2}{2}$$

$$P_2(x) = 0.5 \cdot \frac{1}{10 \sqrt{2\pi}} e^{-\frac{(10-5)^2}{2}}$$

$$= \frac{0.5}{10 \sqrt{2\pi}} \times e^{-\frac{(2.5-10)^2}{2}}$$

$$= \frac{0.5}{44.4} e^{-29.64}$$

$P_2(x)$  is greater

$\therefore x = 10 \in K = 2$

4. Consider multinomial regression for multiclass classification with three features  $\mathbf{X} = (X_1, X_2, X_3)$ , formulated by

$$p_k(\mathbf{X}) = \frac{e^{\beta_{0k} + \beta_{1k}X_1 + \beta_{2k}X_2 + \beta_{3k}X_3}}{1 + e^{\beta_{01} + \beta_{11}X_1 + \beta_{21}X_2 + \beta_{31}X_3} + e^{\beta_{02} + \beta_{12}X_1 + \beta_{22}X_2 + \beta_{32}X_3}}, \quad k \in \{1, 2\}$$

where the classes are determined by  $k \in \{1, 2, 3\}$

Assume that using a data set of 210 observations from three classes, we obtained the following results:

| Coefficient  | Value |
|--------------|-------|
| $\beta_{01}$ | 1     |
| $\beta_{11}$ | -2    |
| $\beta_{21}$ | -1    |
| $\beta_{31}$ | 1     |
| $\beta_{02}$ | 0     |
| $\beta_{12}$ | 0     |
| $\beta_{22}$ | 1     |
| $\beta_{32}$ | 1     |

Assume that the coefficients are all statistically significant.

- (a) In what class will the classifier classify  $\mathbf{X}^* = (1, 0, -1)$ ?
- (b) Explain why despite having three classes, we formulated multinomial regression using ONLY TWO sets of parameters,  $(\beta_{01}, \beta_{11}, \beta_{21}, \beta_{31})$  and  $(\beta_{02}, \beta_{12}, \beta_{22}, \beta_{32})$ , specific to classes 1 and 2, respectively?

$$p_k(x) = \frac{e^{\beta_{0k} + \beta_{1k}x_1 + \beta_{2k}x_2 + \beta_{3k}x_3}}{1 + e^{\beta_{01} + \beta_{11}x_1 + \beta_{21}x_2 + \beta_{31}x_3} + e^{\beta_{02} + \beta_{12}x_1 + \beta_{22}x_2 + \beta_{32}x_3}}$$

$$\text{Denominator} = 1 + e^{1+(-2)(1)+(-1)(0)+(1)(-1)} + e^{0+0+0-1}$$

$$= 1 + e^{-2} + e^{-1}$$

$$= 1 + 0.1353 + 0.3678 = 1.5032$$

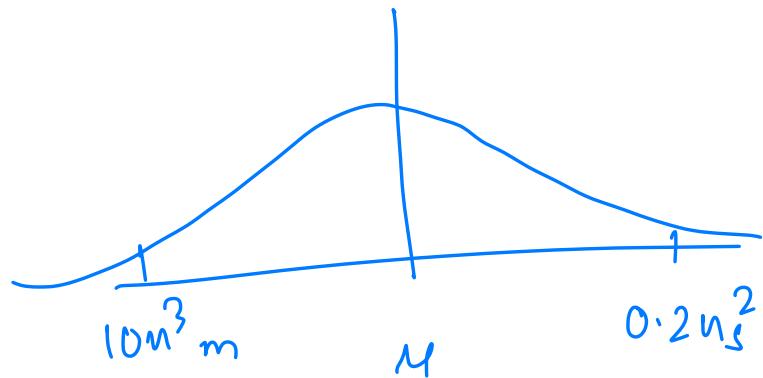
$$P_1(x) = \frac{0.1353}{1.5032} = 0.09$$

$$P_2(x) = \frac{0.3678}{1.5032} = 0.2446$$

$$P_3(x) = 1 - 0.09 - 0.2446 = 0.6653$$

$x^* = (1, 6, -1)$  belongs to class 3. i.e  $k=3$

5. In a weird simulated world, the net worth  $n_i$  of everyone who has a spouse is *uniformly* distributed between  $10n_m^3$  and  $0.2n_s^2$ , where  $n_m$  is the net worth of their mother and  $n_s$  is the net worth of their spouse.
- What is the estimate with minimum mean squared error of the net worth of Fej Zebos, whose mother's net worth is 10 Dollars and whose spouse's net worth is 500 Dollars?
  - What type of supervised learning problem are you solving in this question? Explain.



$$n_m = 10$$

$$n_s = 500$$

$$[10 \times 1000, 0.2 \times 500 \times 500]$$

$$[10000, 50000]$$

$$\mu = 400$$

$$\hat{\beta}_1 - 2SE(\hat{\beta}_1) = 10000$$

$$\hat{\beta}_1 + 2SE(\hat{\beta}_1) = 50000$$

$$\begin{aligned} 4SE(\hat{\beta}_1) &= 40000 - 10000 \\ SE(\hat{\beta}_1) &= 10000 \end{aligned}$$

$$\mu - 4\sigma = 10000$$

$$\mu + 4\sigma = 50000$$

$$8\sigma = 40000$$

$$\sigma = 5000$$

$$\mu = 30000$$

$B = 200\pi$

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

F - Distribution ( $\alpha = 0.01$  in the Right Tail)

| Denominator Degrees of Freedom<br>$df_2$ | $df_1$ | Numerator Degrees of Freedom |        |        |        |        |        |        |        |   |
|--|--------|------------------------------|--------|--------|--------|--------|--------|--------|--------|---|
|  |        | 1                            | 2      | 3      | 4      | 5      | 6      | 7      | 8      | 9 |
| 1  | 4052.2 | 4999.5                       | 5403.4 | 5624.6 | 5763.6 | 5859.0 | 5928.4 | 5981.1 | 6022.5 |   |
| 2  | 98.503 | 99.000                       | 99.166 | 99.249 | 99.299 | 99.333 | 99.356 | 99.374 | 99.388 |   |
| 3  | 34.116 | 30.817                       | 29.457 | 28.710 | 28.237 | 27.911 | 27.672 | 27.489 | 27.345 |   |
| 4  | 21.198 | 18.000                       | 16.694 | 15.977 | 15.522 | 15.207 | 14.976 | 14.799 | 14.659 |   |
| 5  | 16.258 | 13.274                       | 12.060 | 11.392 | 10.967 | 10.672 | 10.456 | 10.289 | 10.158 |   |
| 6  | 13.745 | 10.925                       | 9.7795 | 9.1483 | 8.7459 | 8.4661 | 8.2600 | 8.1017 | 7.9761 |   |
| 7  | 12.246 | 9.5466                       | 8.4513 | 7.8466 | 7.4604 | 7.1914 | 6.9928 | 6.8400 | 6.7188 |   |
| 8  | 11.259 | 8.6491                       | 7.5910 | 7.0061 | 6.6318 | 6.3707 | 6.1776 | 6.0289 | 5.9106 |   |
| 9  | 10.561 | 8.0215                       | 6.9919 | 6.4221 | 6.0569 | 5.8018 | 5.6129 | 5.4671 | 5.3511 |   |
| 10                                       | 10.044 | 7.5594                       | 6.5523 | 5.9943 | 5.6363 | 5.3858 | 5.2001 | 5.0567 | 4.9424 |   |
| 11                                       | 9.6460 | 7.2057                       | 6.2167 | 5.6683 | 5.3160 | 5.0692 | 4.8861 | 4.7445 | 4.6315 |   |
| 12                                       | 9.3302 | 6.9266                       | 5.9525 | 5.4120 | 5.0643 | 4.8206 | 4.6395 | 4.4994 | 4.3875 |   |
| 13                                       | 9.0738 | 6.7010                       | 5.7394 | 5.2053 | 4.8616 | 4.6204 | 4.4410 | 4.3021 | 4.1911 |   |
| 14                                       | 8.8616 | 6.5149                       | 5.5639 | 5.0354 | 4.6950 | 4.4558 | 4.2779 | 4.1399 | 4.0297 |   |
| 15                                       | 8.6831 | 6.3589                       | 5.4170 | 4.8932 | 4.5556 | 4.3183 | 4.1415 | 4.0045 | 3.8948 |   |
| 16                                       | 8.5310 | 6.2262                       | 5.2922 | 4.7726 | 4.4374 | 4.2016 | 4.0259 | 3.8896 | 3.7804 |   |
| 17                                       | 8.3997 | 6.1121                       | 5.1850 | 4.6690 | 4.3359 | 4.1015 | 3.9267 | 3.7910 | 3.6822 |   |
| 18                                       | 8.2854 | 6.0129                       | 5.0919 | 4.5790 | 4.2479 | 4.0146 | 3.8406 | 3.7054 | 3.5971 |   |
| 19                                       | 8.1849 | 5.9259                       | 5.0103 | 4.5003 | 4.1708 | 3.9386 | 3.7653 | 3.6305 | 3.5225 |   |
| 20                                       | 8.0960 | 5.8489                       | 4.9382 | 4.4307 | 4.1027 | 3.8714 | 3.6987 | 3.5644 | 3.4567 |   |
| 21                                       | 8.0166 | 5.7804                       | 4.8740 | 4.3688 | 4.0421 | 3.8117 | 3.6396 | 3.5056 | 3.3981 |   |
| 22                                       | 7.9454 | 5.7190                       | 4.8166 | 4.3134 | 3.9880 | 3.7583 | 3.5867 | 3.4530 | 3.3458 |   |
| 23                                       | 7.8811 | 5.6637                       | 4.7649 | 4.2636 | 3.9392 | 3.7102 | 3.5390 | 3.4057 | 3.2986 |   |
| 24                                       | 7.8229 | 5.6136                       | 4.7181 | 4.2184 | 3.8951 | 3.6667 | 3.4959 | 3.3629 | 3.2560 |   |
| 25                                       | 7.7698 | 5.5680                       | 4.6755 | 4.1774 | 3.8550 | 3.6272 | 3.4568 | 3.3239 | 3.2172 |   |
| 26                                       | 7.7213 | 5.5263                       | 4.6366 | 4.1400 | 3.8183 | 3.5911 | 3.4210 | 3.2884 | 3.1818 |   |
| 27                                       | 7.6767 | 5.4881                       | 4.6009 | 4.1056 | 3.7848 | 3.5580 | 3.3882 | 3.2558 | 3.1494 |   |
| 28                                       | 7.6356 | 5.4529                       | 4.5681 | 4.0740 | 3.7539 | 3.5276 | 3.3581 | 3.2259 | 3.1195 |   |
| 29                                       | 7.5977 | 5.4204                       | 4.5378 | 4.0449 | 3.7254 | 3.4995 | 3.3303 | 3.1982 | 3.0920 |   |
| 30                                       | 7.5625 | 5.3903                       | 4.5097 | 4.0179 | 3.6990 | 3.4735 | 3.3045 | 3.1726 | 3.0665 |   |
| 40                                       | 7.3141 | 5.1785                       | 4.3126 | 3.8283 | 3.5138 | 3.2910 | 3.1238 | 2.9930 | 2.8876 |   |
| 60                                       | 7.0771 | 4.9774                       | 4.1259 | 3.6490 | 3.3389 | 3.1187 | 2.9530 | 2.8233 | 2.7185 |   |
| 120                                      | 6.8509 | 4.7865                       | 3.9491 | 3.4795 | 3.1735 | 2.9559 | 2.7918 | 2.6629 | 2.5586 |   |
| $\infty$                                 | 6.6349 | 4.6052                       | 3.7816 | 3.3192 | 3.0173 | 2.8020 | 2.6393 | 2.5113 | 2.4073 |   |

---



a)  $t_{197, 0.025} = 1.97$

$$t = \frac{\beta_2 - 0}{SE(\beta_2)}$$

$$1.97 = \frac{1}{SE(\beta_2)}$$

$$SE(\beta_2) = \frac{1}{1.97} \approx 0.50$$

$$0 < s < 0.50$$

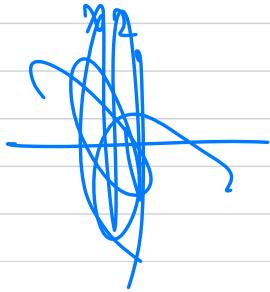
b)

$$\frac{e^{1+2x_1+x_2}}{1 + e^{1+2x_1+x_2}} = \frac{1}{2}$$

$$e^{1+2x_1+x_2} = 1$$

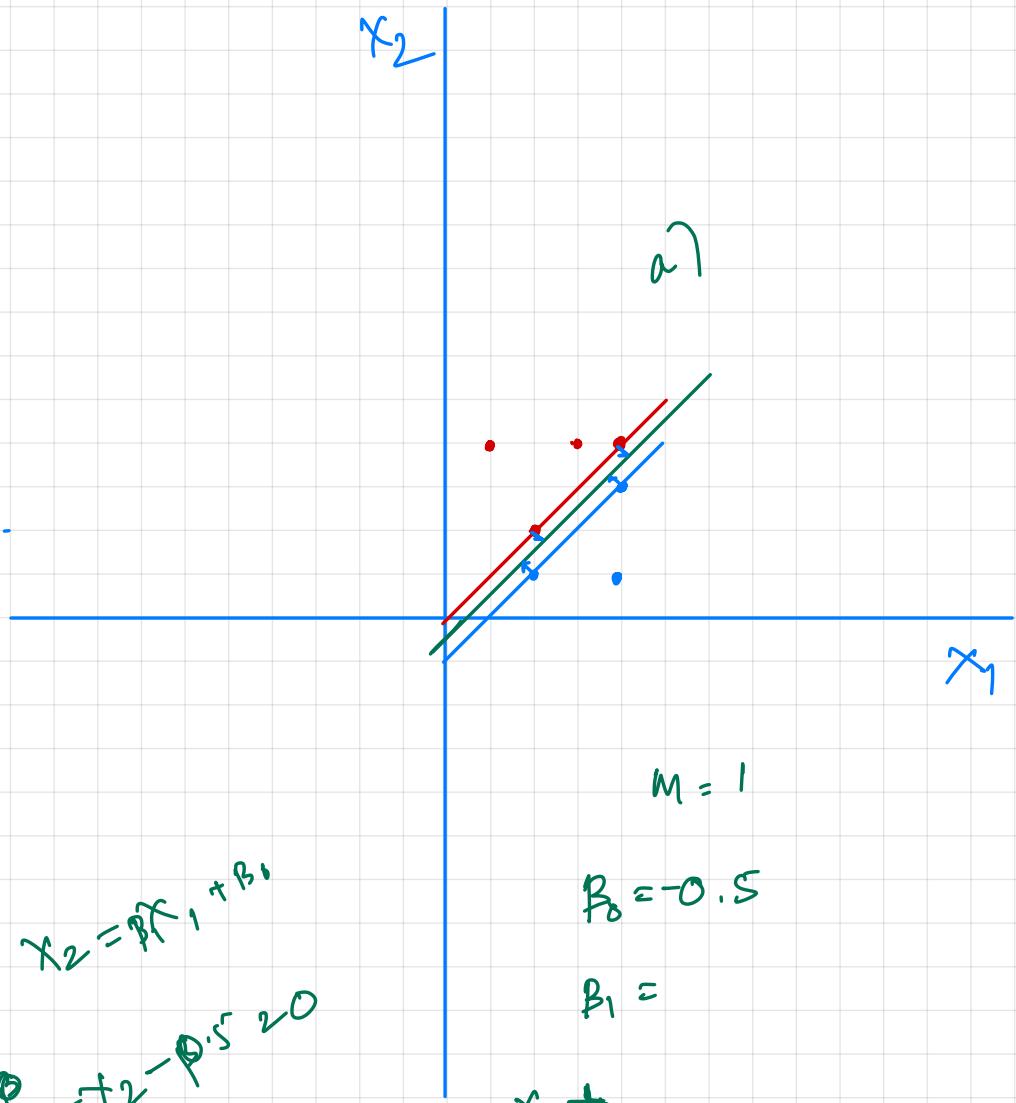
$$1+2x_1+x_2 = 0$$

$$x_2 = -(1+2x_1)$$



$$3) P(Y=1 | X = (-1, 1)) = \frac{e^{1+2(-1)+1}}{1+e^{1+2(-1)+1}}$$
$$= \frac{e^0}{1+e^0} = \frac{1}{2}$$

It lies on decision boundary



$$x_2 = B_0 + B_1 x_1$$

$$B_0 + x_2 - B_1 x_1 \geq 0$$

b)  $x_1 - x_2 - 0.5 \leq 0$

c)  $x_1 - x_2 - 0.5 \geq 0$

Red else Blue

2)

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 20    |        |
| 3       | 20    |        |
| 4       | 20    |        |
| 5       | 25    |        |
| Total   | 110   |        |

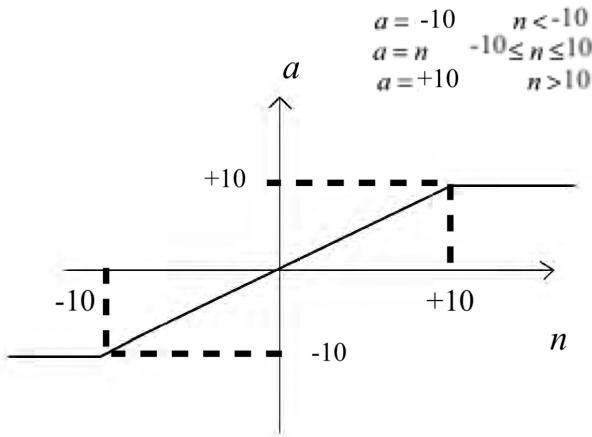
1. The purpose of this question is to design a Convolutional Neural Network to classify the word "REZA" encoded as class  $C_1 = [1 \ 0]^T$  using one-hot encoding from the word "JACK" encoded as  $C_2 = [0 \ 1]^T$ . Here, the convolution operator acts on "letters" instead of pixels. Letter are encoded using the following table:

**Conversion Table**

|   |   |    |   |   |    |   |   |    |
|---|---|----|---|---|----|---|---|----|
| A | = | 1  | K | = | 11 | U | = | 21 |
| B | = | 2  | L | = | 12 | V | = | 22 |
| C | = | 3  | M | = | 13 | W | = | 23 |
| D | = | 4  | N | = | 14 | X | = | 24 |
| E | = | 5  | O | = | 15 | Y | = | 25 |
| F | = | 6  | P | = | 16 | Z | = | 26 |
| G | = | 7  | Q | = | 17 |   |   |    |
| H | = | 8  | R | = | 18 |   |   |    |
| I | = | 9  | S | = | 19 |   |   |    |
| J | = | 10 | T | = | 20 |   |   |    |

Each word is represented as a *row vector*.

Only one feature map is created using the Kernel  $[-1 \ 2 \ -1]$  with stride 1. The resulting feature map is then passed through the saturating linear activation function described in the following figure:



Note that the mathematical formula for the saturated linear function  $\mathbf{f}^{(1)}$  is given in the figure. A maxpooling operator is then applied to the *whole* feature map that is output of the saturating linear function. For example, if the output of the saturating linear function is  $\mathbf{a}^{(1)} = [1 \ 10 \ -10 \ 5]^T$ , then  $a^{(2)} = \text{maxpooling}(\mathbf{a}^{(1)}) = 10$ . The output of the maxpooling is treated as the input to a Feedforward Neural Network with one layer whose weight matrix is  $\mathbf{W}^{(3)}$ . For simplicity, we assume that the network does not have bias. This one layer neural network has its own activation function  $\mathbf{f}^{(3)}$ .

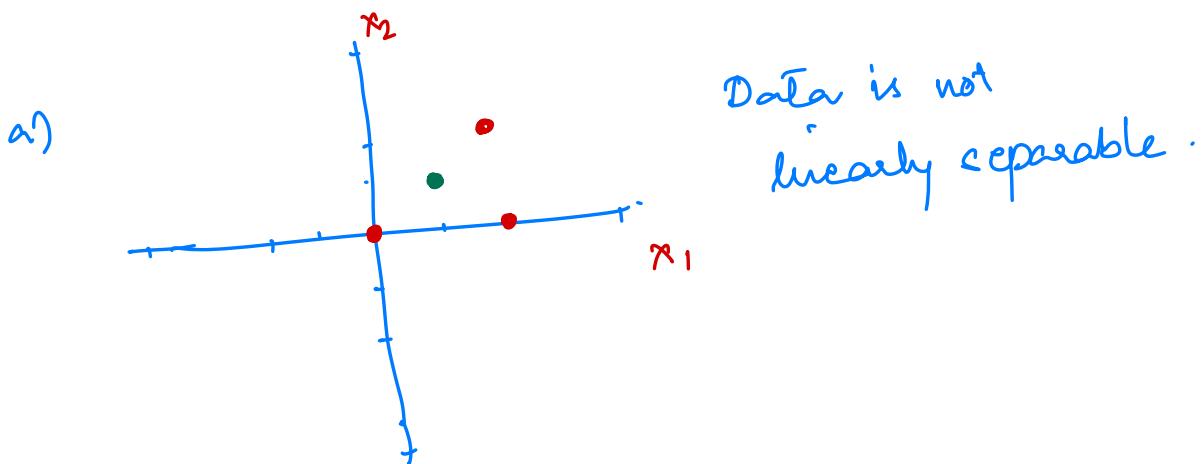
- (a) Draw a block diagram of this network.
- (b) Determine  $\mathbf{W}^{(3)}$  and  $\mathbf{f}^{(3)}$  such that REZA is classified as  $[1 \ 0]^T$  and JACK is classified as  $[0 \ 1]^T$  and show all the calculations that are needed to determine the output of the network for each of these two words.

2. A company with headquarters in the Bay Area has two offices in Los Angeles and San Diego. An employee in San Diego office is sent to the Los Angeles office the next day with probability 0.35 and stays in San Diego office with probability 0.65. An employee in Los Angeles office is sent to the San Diego office with probability 0.8 and stays in Los Angeles office with probability 0.2. A new employee is assigned to Los Angeles office with probability 0.4 and to San Diego office with probability 0.6. An employee in San Diego office works between six and eight hours per day with probability 0.7, works more than eight hours with probability 0.2, and works less than six hours per day with probability 0.1. An employee in Los Angeles office works between six and eight hours per day with probability 0.15, works more than eight hours with probability 0.25, and works less than six hours per day with probability 0.6. A manager in the headquarters can only observe the number of hours each employee worked each day.
- (a) Construct a Hidden Markov Model that models the observations of the manager in their headquarters. Clearly show the parameters with matrices and vectors and draw a state transition graph for the model.
  - (b) If the manager observes the number of hours a new employee worked in the first three consecutive days of work to be 6.5, 10, 7, what is the most likely sequence of places at which the employee worked in those three days?
  - (c) What sequence of three places has the maximum expected number of correct places?

**Solution:**

3. Consider the following data set: In class 1, we have  $[0 \ 0]^T, [0 \ 1]^T, [1 \ 1]^T$ . In class 2, we have  $[0.5 \ 0.5]^T$ .

- (a) Sketch the data set and determine whether or not it is linearly separable.
- (b) Regardless of the answer to 3a, find a quadratic feature  $X_3 = f(X_1, X_2) = aX_1^2 + bX_2^2 + cX_1X_2 + d$ , that makes the data linearly separable; that is,  $X_3 \geq 0$  for members of class 1, and  $X_3 < 0$  for members of class 2. Find the maximum margin classifier only based on  $X_3$ . Hint: The equation of the maximum margin classifier based on only one feature is  $X_3 = \beta_0$  and you should determine  $\beta_0$ .
- (c) By solving  $X_3 = f(X_1, X_2) = \beta_0$  for  $X_2$ , find the equation of the decision boundary in the original feature space and sketch it. Show the regions in the feature space that are classified as class 1 and class 2. You do not need to be very precise.



b)  $f(X_1, X_2) =$

$$\begin{pmatrix} X_1^2 \\ X_2^2 \\ X_1X_2 \end{pmatrix}$$

Class 1 =  $\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

Class 2 =  $\begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \end{pmatrix}$

4. Suppose that for a particular data set, we perform hierarchical clustering using single linkage (minimal intercluster dissimilarity) and using complete linkage (maximal intercluster dissimilarity). We obtain two dendograms.

- (a) At a certain point on the single linkage dendrogram, the clusters  $\{1, 2, 3\}$  and  $\{4, 5\}$  fuse. On the complete linkage dendrogram, the clusters  $\{1, 2, 3\}$  and  $\{4, 5\}$  also fuse at a certain point. Which fusion will occur higher on the tree, or will they fuse at the same height, or is there not enough information to tell?
- (b) At a certain point on the single linkage dendrogram, the clusters  $\{5\}$  and  $\{6\}$  fuse. On the complete linkage dendrogram, the clusters  $\{5\}$  and  $\{6\}$  also fuse at a certain point. Which fusion will occur higher on the tree, or will they fuse at the same height, or is there not enough information to tell?

b) They will fuse at same height.  
 $\therefore$  linkage doesn't affect leaf to leaf fusion

5. Choose either T (True) or F (False):

-  (a) One can design a classifier for the XOR problem using a MLP with linear activation functions in the hidden layer and sigmoids in the output layer. T F
- (b)  $\mathcal{L}_2$  regularization in the back-propagation+Stochastic Gradient Descent training of MLPs is equivalent to adding a forgetting factor to the weight update equation. T F
- (c) When any linear binary classifier results in classification close to random guessing, an RBF Kernel is the best kernel of choice to expand the feature space for a Support Vector Machine. T F
- (d) In Co-training, we use a labler or "Oracle" along with a classifier in a collaborative manner to train the classifier. T F
-  (e) The function of hidden layers of MLPs is equivalent to the function of Kernels in SVMs. T F
- (f) Convolutional Neural Networks act as feature extractors from images. T F
- (g) The responses of support vector classifiers and unregularized logistic regression trained on the same data set are very similar because of having very similar loss functions. T F
- (h) We cannot encode each binary label of a multi-label problem into an output of a MLP, because that architecture is reserved for multi-class classification. T F
-  (i) The Naïve Bayes' classifier cannot yield decision boundaries that are the same as those given by a support vector classifier. T F
- (j) To find  $K$  in the K-means algorithm, we can penalize the objective function WCV (Within Cluster Variation) using AIC or BIC, in the same way we use AIC or BIC in model selection for linear regression. T F

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

Name:

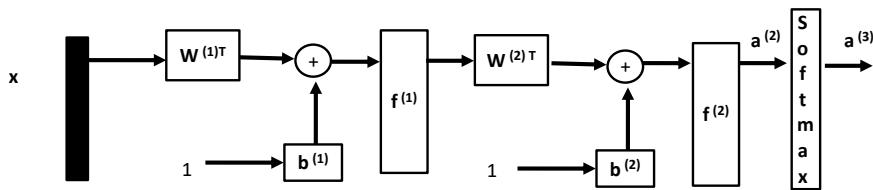
USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No cell phone, no books or other notes are permitted. Only two letter size cheat sheets (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.
- Please make sure to look at all pages in your exam.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 20    |        |
| 3       | 20    |        |
| 4       | 20    |        |
| 5       | 20    |        |
| Total   | 105   |        |

1. Consider the following MLP



where

$$\mathbf{W}^{(1)} = \begin{bmatrix} -1 & -2 \\ -1 & 2 \end{bmatrix}$$

$$\mathbf{W}^{(2)} = \begin{bmatrix} 1 & 2 & 1 \\ -1 & -2 & 0 \end{bmatrix}$$

$$\mathbf{b}^{(1)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{b}^{(2)} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix},$$

Also assume that the components of  $\mathbf{f}^{(1)}$  are ReLus, i.e. the equation of each component is  $f_1(x) = \max(0, x)$ , the components of  $\mathbf{f}^{(2)}$  are hyperbolic tangents and the equation of each component is  $f_2(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ . The second layer is followed by a softmax layer, and the output of each neuron in the softmax layer is calculated as:

$$a_j^{(3)} = \frac{e^{a_j^{(2)}}}{e^{a_1^{(2)}} + e^{a_2^{(2)}} + e^{a_3^{(2)}}}$$

where  $a_j^{(3)}$  is the output of  $j^{th}$  neuron in the third layer and  $a_j^{(2)}$  is the output of the  $j^{th}$  neuron in the second layer.

Answer the following questions:

- (a) How many neurons are there in the first, second, and third layers, respectively?
- (b) How many elements are there in the output  $\mathbf{a}$  (i.e. what is the dimension of the vector  $\mathbf{a}^{(3)}$ )?
- (c) What is the output of the network when  $\mathbf{x} = [2 \ 1]^T$ ? In what class will it be classified by the network?

a) First 2, Second 3, Third 3

16)  $3 \times 1$

**Solution:**

$$c) \quad \alpha = [2 \quad 1]^T$$

 $2 \times 2 \times 2 \times 1$ 

$$n_1 = w^{(1)T} \alpha + b^{(1)} = \begin{bmatrix} -1 & -1 \\ -2 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -2 - 1 \\ -4 + 2 \end{bmatrix} = \begin{bmatrix} -3 \\ -2 \end{bmatrix}$$

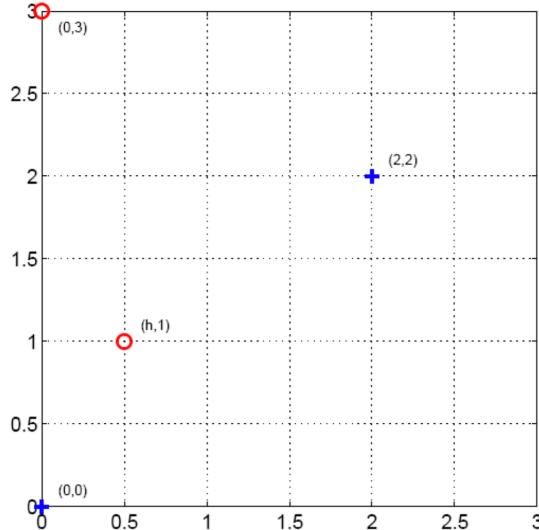
$$\alpha_1 = f_1(n_1) = \max(0, n_1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$n_2 = w^{(2)T} \cdot \alpha_1 + b^{(2)} = \begin{bmatrix} 1 & -1 \\ 2 & -2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

$$\alpha_2 = f_2(n_2) = \tanh(n_2) = \begin{bmatrix} 0 \\ -0.76 \\ 0 \end{bmatrix} \quad \frac{e^{-1} - e^1}{e^{-1} + e^1} = \frac{1 - e^2}{1 + e^2}$$

$$\alpha_3 = \text{softmax}(\alpha_2) = \begin{bmatrix} 0.405 \\ 0.19 \\ 0.405 \end{bmatrix} \quad \frac{1}{e^0 + e^{-0.19} + e^0}$$

2. Suppose we only have four training examples in two dimensions:



positive examples are  $\mathbf{x}_1 = [0 \ 0]^T$ ,  $\mathbf{x}_2 = [2 \ 2]^T$  and negative examples are  $\mathbf{x}_3 = [h \ 1]^T$ ,  $\mathbf{x}_4 = [0 \ 3]^T$ .  $h$  is a parameter.

- (a) What is the largest value of  $h$  for which the training data are still linearly separable?
- (b) Determine the support vectors when  $h = 0.5$ .
- (c) When the training points are separable, does the slope of the maximum margin classifier change? Why?
- (d) Assume that  $h = .5$  and we have unlabeled data  $\mathbf{x}_5 = [3 \ 3]^T$ ,  $\mathbf{x}_6 = [2 \ 0.5]^T$ ,  $\mathbf{x}_7 = [1 \ 1.5]^T$ ,  $\mathbf{x}_8 = [2.5 \ 1.5]^T$ . Which one will be labeled first, if we are performing self-training? Which one will be labeled first, if we are performing active learning?

a)  $h < 1$

b)  $(0.5, 1)$        $(0, 0)$   $(2, 2)$

c) No,  $\therefore \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$  will be support vector

d)

3. Consider the unlabeled dataset with one feature:  $\{0, 4, 5, 20, 25\}$ . Assume that we want to obtain two top-level clusters in this dataset, using bottom-up hierarchical clustering. What will single linkage (minimum distance between members of clusters), complete linkage (maximum distance between members of clusters), and average linkage (average distance between members of clusters) output as the two clusters?

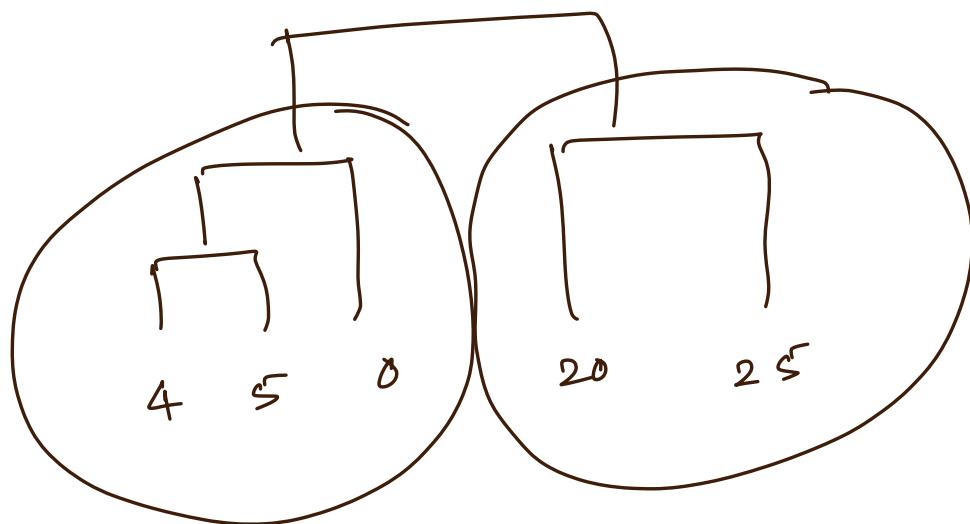
*Complete*

|    |    |    |    |    |    |  |
|----|----|----|----|----|----|--|
| 0  | 0  |    |    |    |    |  |
| 4  | 4  | 0  |    |    |    |  |
| 5  | 5  | 11 | 0  |    |    |  |
| 20 | 20 | 16 | 15 | 0  |    |  |
| 25 | 25 | 21 | 20 | 5  | 0  |  |
|    | 0  | 4  | 5  | 20 | 25 |  |

|    |    |    |    |   |  |
|----|----|----|----|---|--|
| 45 | 0  |    |    |   |  |
| 0  | 5  | 0  |    |   |  |
| 20 | 16 | 20 | 0  |   |  |
| 25 | 21 | 25 | 5  | 0 |  |
| 45 | 0  | 20 | 25 |   |  |

|     |    |    |   |
|-----|----|----|---|
| 450 | 0  |    |   |
| 20  | 20 | 0  |   |
| 25  | 25 | 5  | 0 |
| 450 | 20 | 25 |   |

|        |        |   |  |
|--------|--------|---|--|
| 450    | 0      |   |  |
| 20, 25 | 25     | 0 |  |
| 450    | 20, 25 |   |  |
| 450    | 20, 25 |   |  |



Single:

45 0

0 4 0

20 15 20 0

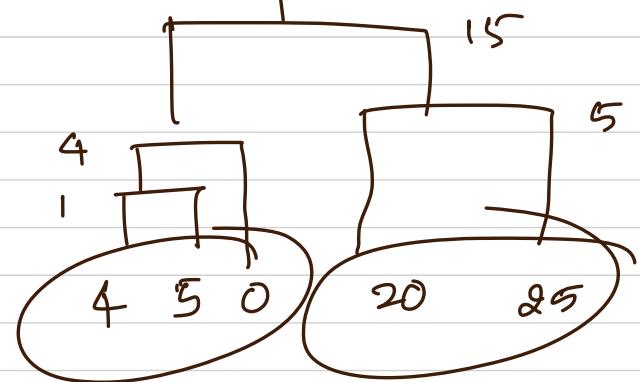
25 20 25 5 0  
45 0 20 25

450 0

20 15 6

25 20 5 0  
450 20 25

450 0  
20, 25 15 0  
450 20, 25



Average:

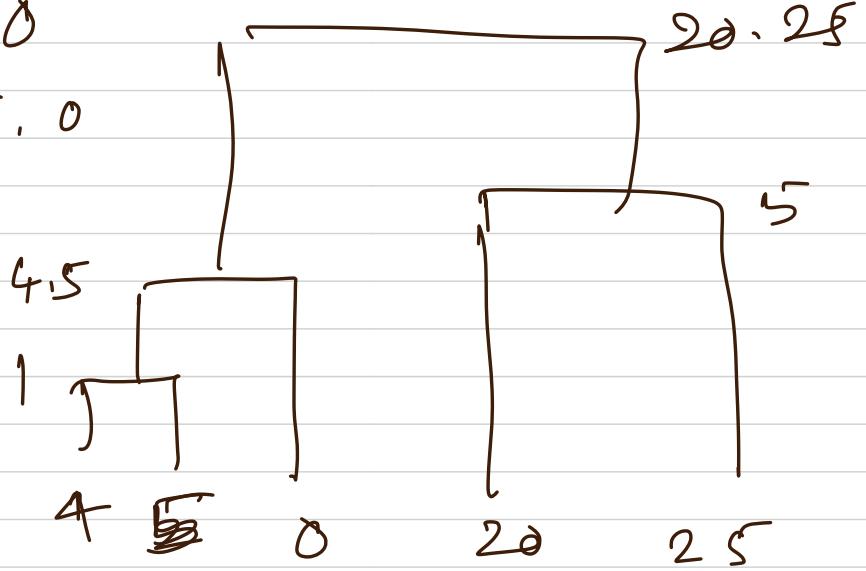
|    |    |    |    |    |    |  |
|----|----|----|----|----|----|--|
| 0  | 0  |    |    |    |    |  |
| 4  | 4  | 0  |    |    |    |  |
| 5  | 5  | 1  | 0  |    |    |  |
| 20 | 20 | 16 | 15 | 0  |    |  |
| 25 | 25 | 21 | 20 | 5  | 0  |  |
| 0  | 4  | 5  | 15 | 20 | 25 |  |

|     |      |    |    |   |  |
|-----|------|----|----|---|--|
| 4.5 | 0    |    |    |   |  |
| 0   | 4.5  | 0  |    |   |  |
| 20  | 15.5 | 20 | 0  |   |  |
| 25  | 20.5 | 25 | 5  | 0 |  |
| 4.5 | 0    | 20 | 25 |   |  |

|        |       |    |   |  |  |
|--------|-------|----|---|--|--|
| 4.5, 0 | 0     |    |   |  |  |
| 20     | 17.75 | 0  |   |  |  |
| 25     | 22.75 | 5  | 0 |  |  |
| 450    | 20    | 25 |   |  |  |

20, 25  
0

|        |                          |
|--------|--------------------------|
| 4.5, 0 | 20, 25 <sup>r</sup><br>0 |
| 20, 25 | 4.5, 0                   |



4. A company with headquarters in the Bay Area has two offices in Los Angeles and San Diego. An employee in San Diego office is sent to the Los Angeles office the next day with probability 0.25 and stays in San Diego office with probability 0.75. An employee in Los Angeles office is sent to the San Diego office with probability 0.3 and stays in Los Angeles office with probability 0.7. A new employee is assigned to Los Angeles office with probability 0.2 and to San Diego office with probability 0.8. An employee in San Diego office works between six and eight hours per day with probability 0.4, works more than eight hours with probability 0.4, and works less than six hours per day with probability 0.2. An employee in Los Angeles office works between six and eight hours per day with probability 0.1, works more than eight hours with probability 0.7, and works less than six hours per day with probability 0.2. A manager in the headquarters can only observe the number of hours each employee worked each day.
- (a) Construct a Hidden Markov Model that models the observations of the manager in their headquarters. Clearly show the parameters with matrices and vectors and draw a state transition graph for the model.
  - (b) If the manager observes the number of hours a new employee worked in the first three consecutive days of work to be 4, 7, 10, what is the most likely sequence of places at which the employee worked in those three days?
  - (c) What sequence of three places has the maximum expected number of correct places?

**Solution:**

5. Choose either T (True) or F (False):

- (a) When the assumption of conditional independence of features holds, the Naïve Bayes' classifier provides the best accuracy among all possible classifiers. T F
- (b) The F1 score is not an appropriate measure for evaluating binrary classifiers when data are not imbalanced. T F
- (c) Leave-One-Out Cross Validation has less bias in estimating the error of a classifier for a large data set than 5 fold cross validation. T F
- (d) When classifying imbalanced data into two classes, we can decrease the threshold on class conditional probability  $\Pr(Y = k | X_1 = x_1, \dots, X_p = x_p)$  to increase the true positive rate at the expense of increasing the false negative rate. T F
- (e) Logistic regression assumes that the conditional odds of the outcome  $Y$  given the features,  $\mathbb{O}[Y = k | X_1 = x_1, \dots, X_p = x_p]$ , is a logistic function of the features. T F

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only two letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.
- Make sure you submit ALL pages of your answers. Answers submitted after the exam is adjourned WILL NOT BE ACCEPTED.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 25    |        |
| 2       | 20    |        |
| 3       | 20    |        |
| 4       | 25    |        |
| 5       | 25    |        |
| Total   | 115   |        |

1. Consider a MLP with one input, two layers, one neuron in each layer, and one output. The activation function of the first layer is  $f^{(1)}(n) = \frac{1}{1+e^{-n}}$  and the activation function of the second layer is  $f^{(2)}(n) = \frac{e^n - e^{-n}}{e^n + e^{-n}}$ . The initial weights of the first and the second layers are respectively  $w^{(1)} = -2$  and  $w^{(2)} = 1$ . There are no bias terms, so  $b^{(1)} = b^{(2)} = 0$  and they are kept zero during training. Assume that we present the data point with  $x = 3$  and  $y = -2$  to the network. Perform one step of the Stochastic Gradient Descent algorithm by using the backpropagation algorithm, assuming the learning rate  $\alpha = 0.2$ . Use the objective function  $J = (y - a^{(2)})^2$ . This means that you should calculate the updated weights.

2. We are trying to estimate the temperature of consecutive years based on *observations* on tree ring sizes. Possible ring sizes are Very Small = VS, Small = S, Medium = M, Large = L, and Very Large = VL. Years can be Cold = C or Hot = H. Assume that we observed VS, VL tree ring sizes in two consecutive years. Also, Assume that  $\pi = [0.2 \ 0.8]$  shows the initial distribution of C and H, respectively. Which of the following HMMs is more likely to have given rise to the observation  $O = \{VS, VL\}$  and why? First rows of  $A_1, B_1, A_2, B_2$  represent C and second rows represent H.

$$(a) \quad \begin{matrix} C & H \end{matrix} \quad \begin{matrix} VS & S & M & L & VL \end{matrix}$$

$$A_1 = \begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix} \quad B_1 = \begin{bmatrix} 0.1 & 0.4 & 0.2 & 0.2 & 0.1 \\ 0.3 & 0.2 & 0.1 & 0.1 & 0.3 \end{bmatrix}$$

$$(b) \quad \begin{matrix} C & H \end{matrix} \quad \begin{matrix} VS & S & M & L & VL \end{matrix}$$

$$A_2 = \begin{bmatrix} 0.6 & 0.4 \\ 0.5 & 0.5 \end{bmatrix} \quad B_2 = \begin{bmatrix} 0.1 & 0.2 & 0.2 & 0.1 & 0.4 \\ 0.6 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}$$

**Solution:**

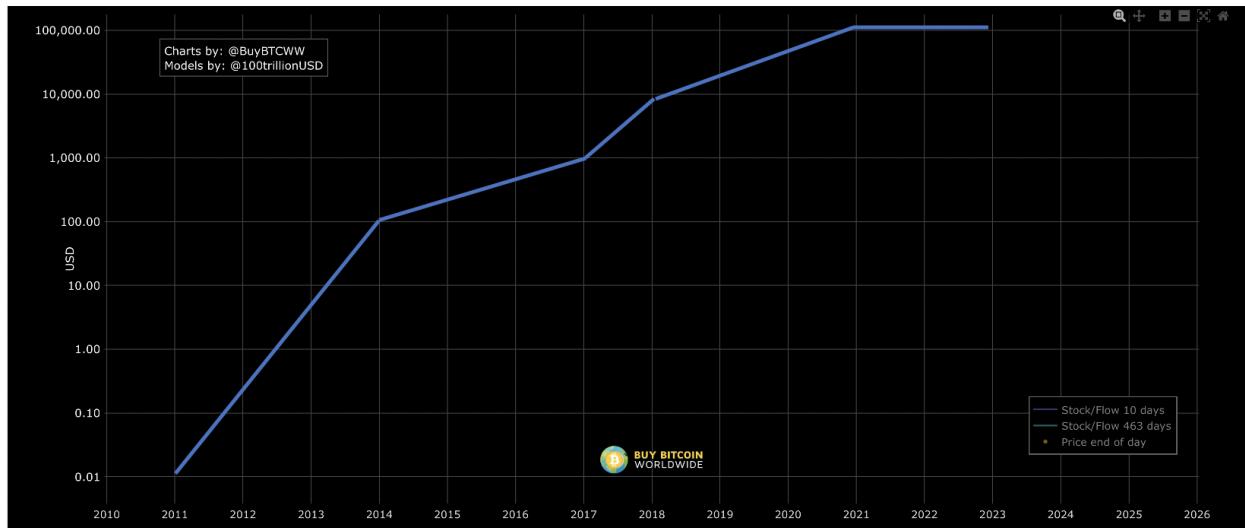
3. Assume the following co-training (multiview learning) self-training scenario: the positive class contains  $\mathbf{x}_1 = [1 \ 0]^T$  and the negative class contains  $\mathbf{x}_2 = [0 \ 1]^T$ . Assume that we first train a maximum margin classifier only based on the first feature of training vectors, then label the unlabeled vector  $\mathbf{x}_3 = [2/3 \ 1/3]^T$ , and then train a maximum margin classifier based on the second feature of  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ . Explain why the class associated with the point  $\mathbf{x}_4 = [2 \ 2]^T$  is *indeterminate* if it is classified using a majority poll between the maximum margin classifier that uses the first feature and the maximum margin classifier that used the second feature for classification.

4. Suppose that for a particular data set, we perform hierarchical clustering using single linkage (minimal intercluster dissimilarity) and using complete linkage (maximal intercluster dissimilarity). We obtain two dendograms.

- (a) At a certain point on the single linkage dendrogram, the clusters  $\{1, 3, 5\}$  and  $\{8, 9\}$  fuse. On the complete linkage dendrogram, the clusters  $\{1, 3, 5\}$  and  $\{8, 9\}$  also fuse at a certain point. Which fusion will occur higher on the tree, or will they fuse at the same height, or is there not enough information to tell?
- (b) At a certain point on the single linkage dendrogram, the clusters  $\{8\}$  and  $\{9\}$  fuse. On the complete linkage dendrogram, the clusters  $\{8\}$  and  $\{9\}$  also fuse at a certain point. Which fusion will occur higher on the tree, or will they fuse at the same height, or is there not enough information to tell?

b) They will fuse at same height.  
 $\therefore$  linkage doesn't affect leaf to leaf fusion

5. The bitcoin Stock to Flow (S2F) model was created by the famous twitter user PlanB. S2F models the price of bitcoin based on its rarity. A slightly modified version of S2F is shown below. Note that the dependent variable  $y$  is  $\log_{10} \text{price}$  and the independent variable is time in years since 2010  $t$ ; therefore  $t \in [1, 13]$ . Show this model using a decision tree and clearly determine the internal nodes and terminal nodes. Remember that this is a *model* tree, so the terminal nodes may contain *regression models*.



let year 2011 to 2014 be  $R_1$

$$\log_{10}(0.01) = -2$$

$$\text{for } R_1 \log_{10} \hat{y}_1 = \beta_0 + \beta_1 x$$

$$= -2 + \frac{4}{3}(\text{year} - 2011)$$

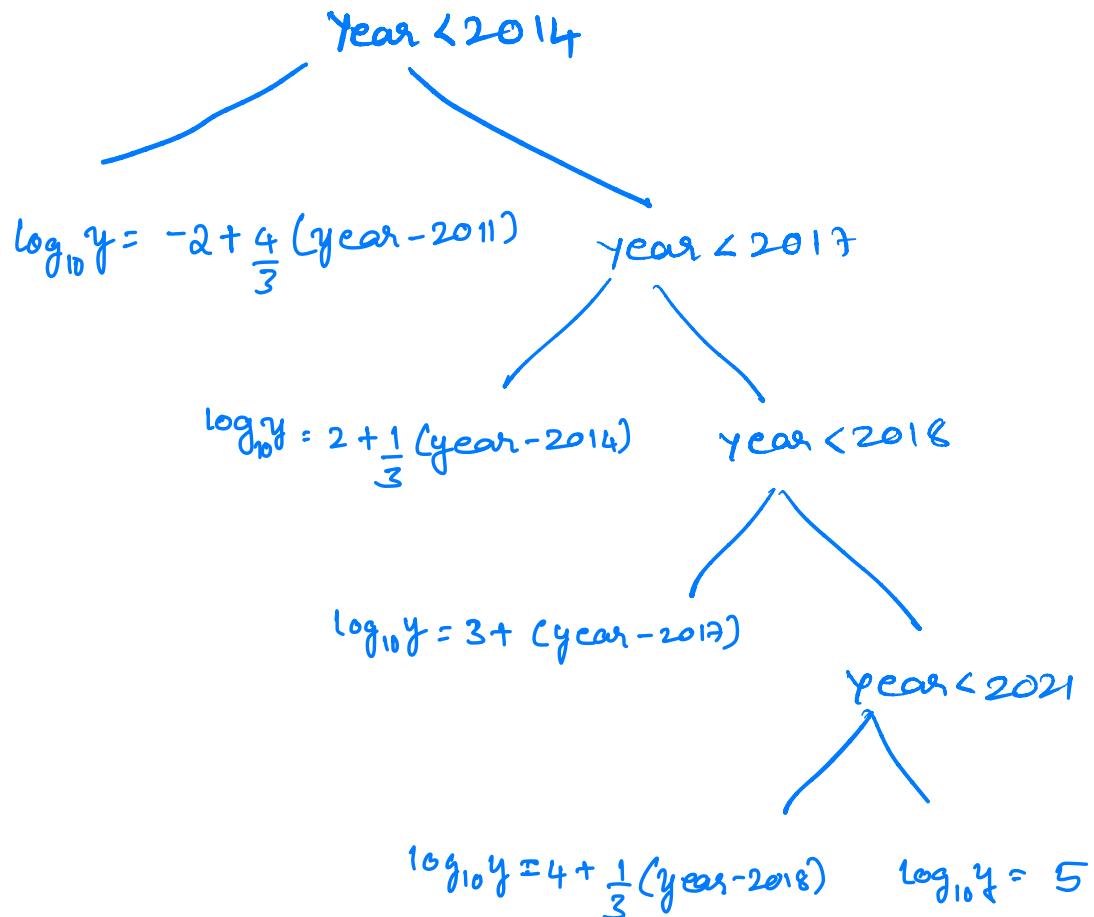
$2014 - 2017 \Rightarrow R_2$

$$\hat{y}_2 = \beta_{20} + \beta_{21} x$$

$$= 100 + \frac{1}{3}(\text{year} - 2014)$$

$$\hat{y}_3 = 1000 +$$

Solution:



Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

Name:

USC ID:

**Notes:**

- Write your name and ID number in the solution you submit.
- No books, cell phones or other notes are permitted. Only two letter size cheat sheets (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 20    |        |
| 2       | 20    |        |
| 3       | 20    |        |
| 4       | 20    |        |
| 5       | 20    |        |
| 6       | 20    |        |
| Total   | 120   |        |

1. Assume that we have a Ridge regression problem with only one predictor, and the true model is linear *without an intercept*, i.e.  $Y = \beta_1 X + \epsilon$ . Assume that we have  $n$  samples,  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$  and we want to find the  $\mathcal{L}_2$  regularized least squares estimate  $\hat{\beta}_1$  from the data.
  - (a) Formulate the objective function in terms of a candidate  $\hat{\beta}_1$  and  $x_i$ 's and  $y_i$ 's, which are known. Assume that the regularization parameter is  $\lambda$
  - (b) Find  $\hat{\beta}_1$  in terms of  $\lambda$  and the data.

2. The data the specialists have collected about malignant and benign tumors in a specific organ in the body is presented below. There are three attributes of each tumor that the cancer specialists can determine without an operation: 1) is the tumor large? 2) is the tumor hard? 3) is the tumor symmetrical? These attributes are abbreviated L, H and S respectively and their truth value is given in the table. For the collected data, the specialists also know whether the tumor turned out to be malignant or not. This fact is in the last column labeled with an M.

| L | H | S | M |
|---|---|---|---|
| T | F | F | T |
| F | F | F | T |
| F | T | T | T |
| F | F | T | F |
| T | F | T | F |

- (a) Train a decision tree on the data using cross entropy. The stopping criterion is zero entropy, i.e. when all data points in a region are in the same class.  
 (b) Using the decision tree you trained, predict the labels in the following test data:

| L | H | S | M |
|---|---|---|---|
| F | F | F | ? |
| T | T | T | ? |
| F | F | T | ? |

$$\text{Cross Entropy} = -\sum_{k=1}^K \hat{P}_{m_k} \log \hat{P}_{m_k}$$

| Malignant |   |
|-----------|---|
| T         | F |
| 3         | 2 |

$$E(M) = E(3, 2)$$

$$= -\frac{3}{5} \log \frac{3}{5} - \frac{2}{5} \log \frac{2}{5} = 0.2922$$

|       |   | Malignant |   |   |
|-------|---|-----------|---|---|
|       |   | T         | F |   |
| Large | T | 1         | 1 | 2 |
|       | F | 2         | 1 | 3 |
|       |   | T         | 0 | 1 |
| Hard  | F | 2         | 2 | 4 |
|       | T | 1         | 2 | 3 |
|       |   | F         | 0 | 2 |
| Sym   |   | T         | 3 | 3 |
| F     |   | 2         | 0 | 2 |

$$E(M, L) = \frac{2}{5} E(1, 1) + \frac{3}{5} E(2, 1)$$

$$= \frac{2}{5} \left( -\log \frac{1}{2} \right) + \frac{3}{5} \left( -\frac{2}{3} \log \frac{2}{3} - \frac{1}{3} \log \frac{1}{3} \right)$$

$$= 0.120 + 0.130$$

$$= 0.25$$

$$E(M, H) = \frac{1}{5} E(1, 0) + \frac{4}{5} E(2, 2)$$

$$= \frac{1}{5} (-\log 1) + \frac{4}{5} \left( -\frac{2}{4} \log \frac{2}{4} - \right.$$

$$+ 0.24$$

$$= 0.24$$

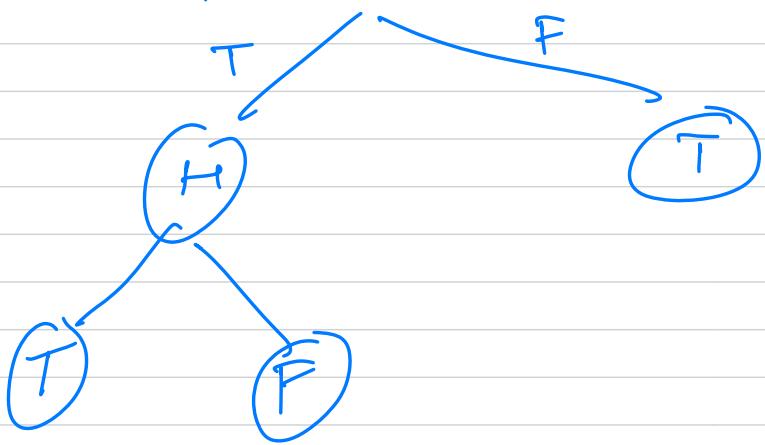
$$E(M, S) = \frac{3}{5} E(1, 2) + \frac{2}{5} E(2, 0)$$

$$= \frac{3}{5} \left( -\frac{1}{3} \log \frac{1}{3} + \frac{2}{3} \log \frac{2}{3} + \frac{2}{5} \right)$$

$$= 0.13 +$$

Smallest is Symmetry

Symmetris

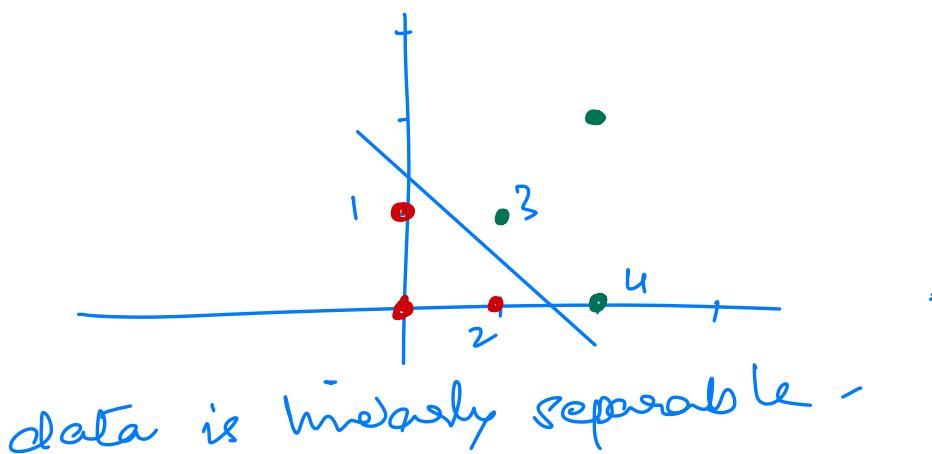


3. Consider the following training data:

| Index | $X_1$ | $X_2$ | $Y$ |
|-------|-------|-------|-----|
| 1     | 1     | 1     | 1   |
| 2     | 2     | 2     | 1   |
| 3     | 2     | 0     | 1   |
| 4     | 0     | 0     | -1  |
| 5     | 1     | 0     | -1  |
| 6     | 0     | 1     | -1  |

- (a) Carefully sketch these six training points. Are the classes linearly separable?
- (b) Construct the weight vector of the maximum margin hyperplane by inspection and identify the support vectors..
- (c) For each of the support vectors answer the following question: if you remove the support vector, does the size of the optimal margin remain the same, increase, or decrease?

a)



b)

$$\beta_0 = 1.5 \quad \beta_1 = -1$$

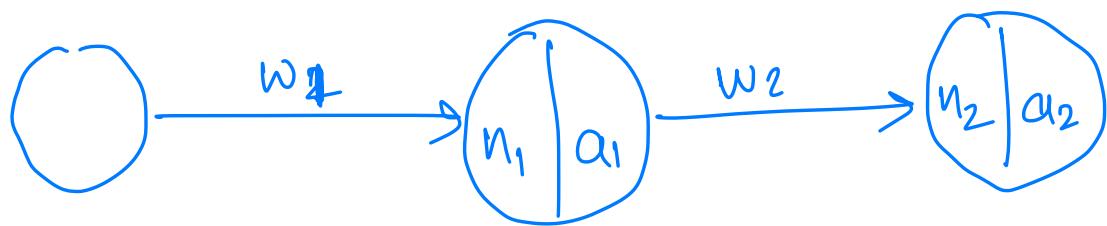
c) 1 → remains same

2 → Increase

3 → Increase

4 → remains same<sup>4</sup>

4. Consider a MLP with one input, two layers, one neuron in each layer, and one output. The activation function of the first layer is  $f^{(1)}(n) = 2n$  and the activation function of the second layer is  $f^{(2)}(n) = 3n$ . The initial weights of the first and the second layers are respectively  $w^{(1)} = 0$  and  $w^{(2)} = 0$ . Assume that we present the data point with  $x = 1$  and  $y = 2$  to the network. Perform one step of the Stochastic Gradient Descent algorithm by using the backpropagation algorithm, assuming the learning rate  $\alpha = 0.5$ . Use the MSE objective function, i.e.  $J = (y - a^{(2)})^2$ . This means that you should calculate the updated weights



$$n_1 = w_1 * x_1 = 0 * 1 = 0$$

$$a_1 = 2 \cdot n_1 = 2 \cdot 0 = 0$$

$$n_2 = w_2 * a_1 = 0 * 0 = 0$$

$$a_2 = 3 \cdot n_2 = 3 \cdot 0 = 0$$

$$J = (y - a^{(2)})^2 = (2 - 0)^2 = 4$$

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial a_2} * \frac{\partial a_2}{\partial n_2} * \frac{\partial n_2}{\partial w_2}$$

$$J = (y - a_2)^2 \quad \frac{\partial J}{\partial a_2} = 2 \cdot (y - a_2) = 2(2 - 0) = 4$$

5

$$= -4$$

$$a_2 = 3 \cdot n_2$$

$$\frac{\partial a_2}{\partial n_2} = 3$$

$$n_2 = a_1 \cdot w_2$$

$$\frac{\partial n_2}{\partial w_2} = a_1 = 0$$

$$\frac{\partial J}{\partial w_2} = 0$$

$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial a_1} * \frac{\partial a_1}{\partial n_1} * \frac{\partial n_1}{\partial w_1}$$

$$\frac{\partial J}{\partial a_1} = \frac{\partial J}{\partial n_2} * \frac{\partial n_2}{\partial a_1}$$

$$\frac{\partial J}{\partial n_2} = \frac{\partial J}{\partial a_2} * \frac{\partial a_2}{\partial n_2}$$

$$= -4 * 3 = -12$$

~~$\frac{\partial n_2}{\partial a_1}$~~ ,  $n_2 = w_2 * a_1$

$$\frac{\partial n_2}{\partial a_1} = w_2 = 0$$

$$\therefore \frac{\partial J}{\partial a_1} = 0 - 12 * 0 = 0$$

$$\therefore \frac{\partial J}{\partial w_1} = 0$$

$$w_2^{(1)} = w_1^{(1)} + \alpha \cdot \frac{\partial J}{\partial w_1} = 0 + \alpha \cdot 0 = 0$$

$$w_2^{(2)} = w_1^{(2)} + \alpha \cdot \frac{\partial J}{\partial w_2} = 0 + \alpha \cdot 0 = 0$$

5. Assume that you have a labeled dataset. Explain how you can use only K-means clustering to build a classification model for this dataset. What can go wrong?

6. Choose either T (True) or F (False):

- (a) Complexity of random forests significantly changes when the number of trees  $B$  increases, and they become more prone to overfitting. T F
- (b) Gaussian Kernels are always the kernels of choice for implicit expansion of feature space in SVMs. T F
- (c) Instead of majority polling, one can use SVMs in each segment of the feature space in a decision tree to predict labels. T F
- (d) Semi-supervised learning cannot be used for regression problems. T F
- (e) A Multilayer Perceptron with one hidden layer of sigmoids and an output layer with linear activation functions can learn a linear regression function with any precision and this does NOT defy the no free lunch theorem. T F
- (f) The  $\mathcal{L}_1$  regularizer can easily exclude correlated features when combined with logistic regression. T F
- (g) To convert a multiclass classification problem with 10,000 classes to multiple binary classification problems, it makes more sense to choose the One-Versus-All (OVA) method over the One-Versus-One (OVO) method. T F
- (h) For a multi-label classification problem with 100 binary labels, the label power set method is the best method of converting the problem into a multi-class problem. T F
- (i) In a binary classification problem, if we are interested in class conditional probabilities, we can not use Support Vector Classifiers. T F
- (j) The results of K-means and Hierarchical Clustering can be different for the same data set, because K-Means does not assume that clusters are nested. T F

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

Name:

USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No cell phone, no books or other notes are permitted. Only two letter size cheat sheets (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.

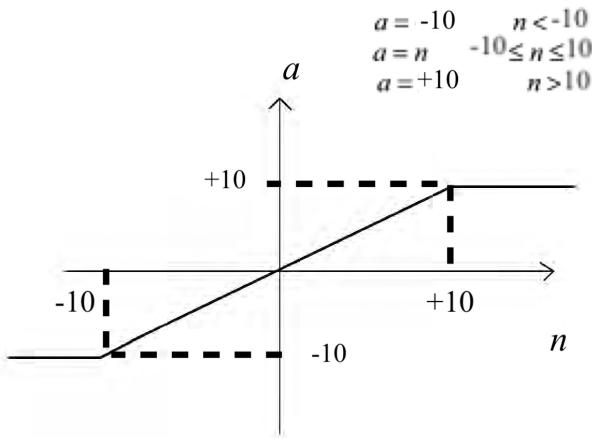
| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 20    |        |
| 2       | 20    |        |
| 3       | 20    |        |
| 4       | 20    |        |
| 5       | 20    |        |
| Total   | 120   |        |

1. The purpose of this question is to design a Convolutional Neural Network to classify the word "REZA" encoded as class  $C_1 = [1 \ 0]^T$  using one-hot encoding from the word "JACK" encoded as  $C_2 = [0 \ 1]^T$ . Here, the convolution operator acts on "letters" instead of pixels. Letters are encoded using the following table: Each word is represented as a *row vector*.

**Conversion Table**

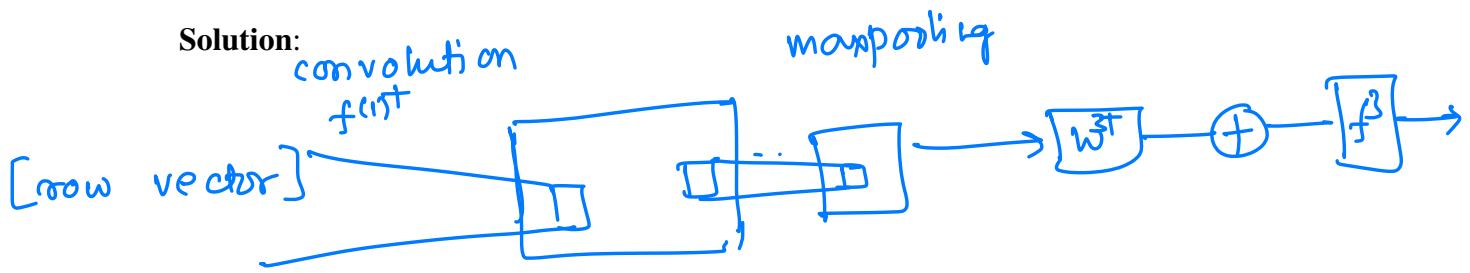
|        |        |        |
|--------|--------|--------|
| A = 1  | K = 11 | U = 21 |
| B = 2  | L = 12 | V = 22 |
| C = 3  | M = 13 | W = 23 |
| D = 4  | N = 14 | X = 24 |
| E = 5  | O = 15 | Y = 25 |
| F = 6  | P = 16 | Z = 26 |
| G = 7  | Q = 17 |        |
| H = 8  | R = 18 |        |
| I = 9  | S = 19 |        |
| J = 10 | T = 20 |        |

Only one feature map is created using the Kernel  $[-1 \ 2 \ -1]$  with stride 1. The resulting feature map is then passed through the saturating linear activation function described in the following figure:



Note that the mathematical formula for the saturated linear function  $\mathbf{f}^{(1)}$  is given in the figure. A maxpooling operator is then applied to the *whole* feature map that is output of the saturating linear function. For example, if the output of the saturating linear function is  $\mathbf{a}^{(1)} = [1 \ 10 \ -10 \ 5]^T$ , then  $\mathbf{a}^{(2)} = \text{maxpooling}(\mathbf{a}^{(1)}) = 10$ . The output of the maxpooling is treated as the input to a Feedforward Neural Network with one layer whose weight matrix is  $\mathbf{W}^{(3)}$ . For simplicity, we assume that the network does not have bias. This one layer neural network has its own activation function  $\mathbf{f}^{(3)}$ .

- (a) Draw a block diagram of this network.
- (b) Determine  $\mathbf{W}^{(3)}$  and  $\mathbf{f}^{(3)}$  such that REZA is classified as  $[1 \ 0]^T$  and JACK is classified as  $[0 \ 1]^T$  and show all the calculations that are needed to determine the output of the network for each of these two words.



$$\begin{aligned}
 [\text{RE2A}] &\rightarrow [18 \ 5 \ 26 \ 1] \xrightarrow{\times} [-18 + 10 - 26 \quad -5 + 52 - 1] \\
 &\qquad\qquad\qquad\downarrow f(0) \\
 &[-1 \ 2 \ -1] \qquad\qquad\qquad [-34 \ 46] \\
 &\qquad\qquad\qquad\downarrow \text{max pool} \\
 &\qquad\qquad\qquad [10]
 \end{aligned}$$

$$\begin{aligned}
 [J \ A \ C \ R] &\rightarrow [10 \ 1 \ 3 \ 1] = [-10 * 2 - 3 \quad -1 + 6 - 1] \\
 &\qquad\qquad\qquad\cdot [-11 \ 1 \ -6] \\
 &[-1 \ 2 \ -1] \qquad\qquad\qquad [-10 \ \downarrow \ -10] \\
 &\qquad\qquad\qquad\downarrow \text{max positive} \\
 &\qquad\qquad\qquad [-10]
 \end{aligned}$$

$$\text{let } w(1) = [1 \ 1] \qquad f(n) = \begin{cases} 0 & n < 0 \\ 1 & n \geq 0 \end{cases}$$

$$n_1 = w^T \cdot x = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 10 \\ 10 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \end{bmatrix}$$

$$a_1 = f(n) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

**Solution:**

$$\text{let } c = y - a = \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

$\alpha = 0.5$

$$w(1) = w_1 + 0.5 \cdot \alpha \cdot c(1)$$

$$\begin{aligned} &= \begin{bmatrix} 1 & 1 \end{bmatrix} + 0.5 \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -5 \end{bmatrix} \\ &= \begin{bmatrix} 1 & -4 \end{bmatrix} \end{aligned}$$

Run 2:

$$n = w^T \cdot x = \begin{bmatrix} 1 & -4 \end{bmatrix}^T \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix}^T$$

$$a = f(n) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$c(2) = y - a = \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0$$

Validation.For Jack  $\Rightarrow [-10]$

**Solution:**

$$n = w^T \cdot n = [1 \ -4]^T [-10] = [-10 \ 40]^T$$

$$a = f(n) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\text{error(jack)} = 0$$

=

$$\therefore w = [1 \ -4]$$

$$\alpha = 0.5$$

$$\text{activation function} = f(a) = \begin{cases} 1 & a \geq 0 \\ 0 & a < 0 \end{cases}$$

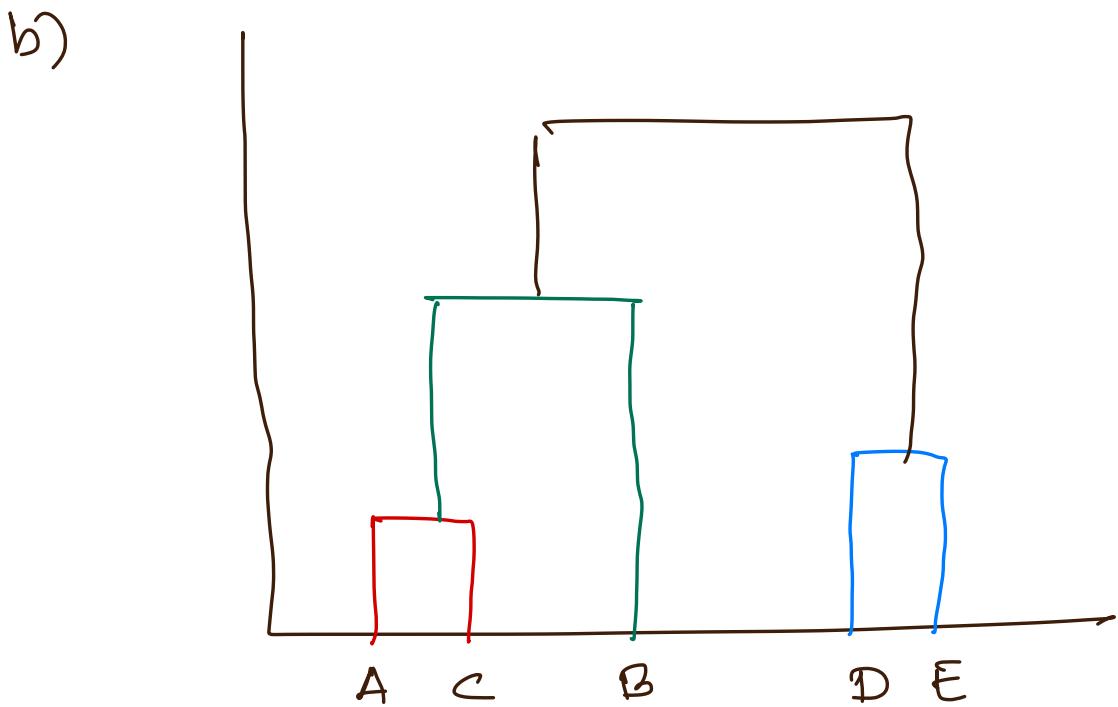
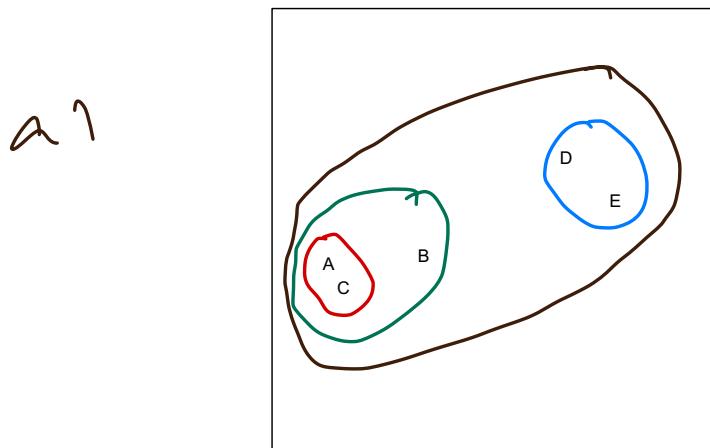
2. Consider the following data set: In class 1, we have  $[0\ 0]^T$ ,  $[0\ 1]^T$ ,  $[1\ 1]^T$ . In class 2, we have  $[0.5\ 0.5]^T$ .
- Sketch the data set and determine whether or not it is linearly separable.
  - Regardless of the answer to 2a, find a quadratic feature  $X_3 = f(X_1, X_2)$ , that makes the data linearly separable. Find the maximum margin classifier only based on  $X_3$ . Hint: The equation of the maximum margin classifier based on only one feature is  $X_3 = \beta_0$  and you should determine  $\beta_0$ .
  - By solving  $X_3 = f(X_1, X_2) = \beta_0$  for  $X_2$ , find the equation of the decision boundary in the original feature space and sketch it. Show the regions in the feature space that are classified as class 1 and class 2. You do not need to be very precise.

3. Choose either T (True) or F (False):

- (a) One can design a classifier for the XOR problem using a MLP with linear activation functions in the hidden layer and sigmoids in the output layer. T F
- (b)  $\mathcal{L}_2$  regularization in the back-propagation+Stochastic Gradient Descent training of MLPs is equivalent to adding a forgetting factor to the weight update equation. T F
- (c) When any linear binary classifier results in classification close to random guessing, an RBF Kernel is the best kernel of choice to expand the feature space for a Support Vector Machine. T F
- (d) In Co-training, we use a labler or "Oracle" along with a classifier in a collaborative manner to train the classifier. T F
- (e) The function of hidden layers of MLPs is equivalent to the function of Kernels in SVMs. T F
- (f) Convolutional Neural Networks act as feature extractors from images. T F
- (g) The responses of support vector classifiers and unregularized logistic regression trained on the same data set are very similar because of having very similar loss functions. T F
- (h) We cannot encode each binary label of a multi-label problem into an output of a MLP, because that architecture is reserved for multi-class classification. T F
- (i) The Naïve Bayes' classifier cannot yield decision boundaries that are the same as those given by a support vector classifier. T F
- (j) To find  $K$  in the K-means algorithm, we can penalize the objective function WCV (Within Cluster Variation) using AIC or BIC, in the same way we use AIC or BIC in model selection for linear regression. T F

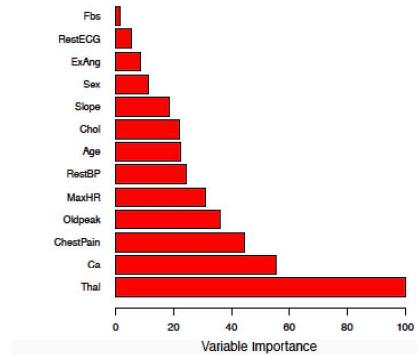
4. Consider the unlabeled two-dimensional data represented in the following figure.

- Draw hierarchical clusters for the data.
- Draw a dendrogram for hierarchical clusters.

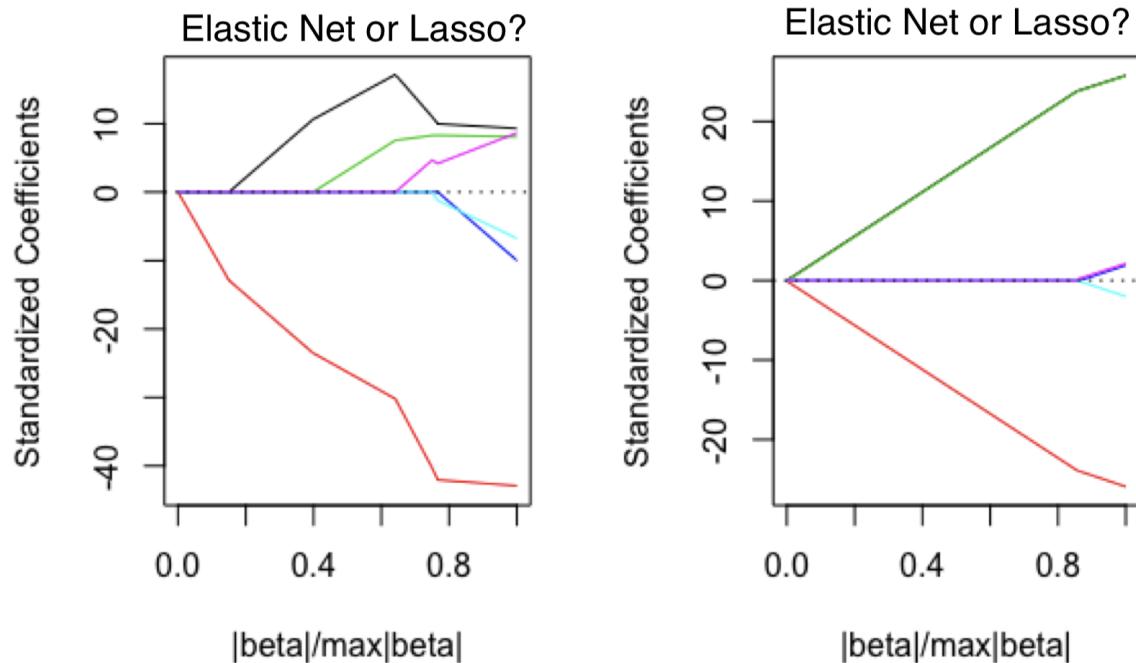


5. Assume that in a binary classification task, our data set has three highly correlated but relevant features  $X_1, X_2, X_3$  and three irrelevant features  $X_4, X_5, X_6$ .  $X_1$  and  $X_2$  have very large positive correlation.  $X_1$  and  $X_3$  have very large negative correlation.  $X_4$  and  $X_5$  have very high positive correlation, and  $X_5$  and  $X_6$  have very high negative correlation.

- (a) Assume that we use random forests for classification and it has low training error. Sketch a reasonable variable importance plot for this data set. As a reminder, a sample variable importance plot for random forests is shown below:



- (b) Assume that we perform classification using logistic regression with Elastic Net and with Lasso penalties. In the figure below, determine which penalty was used by circling either Elastic Net or Lasso on top of each subfigure. Also, on each figure, determine which coefficient path is associated with each of the variables  $X_1, X_2, \dots, X_6$ .



a) Imp:  $x_1 x_2 x_3$

Irrelevant:  $x_4 x_5 x_6$

Corr

|       | $x_1$ | $x_2$ | $x_3$ |
|-------|-------|-------|-------|
| $x_1$ | 0     | +1    | -1    |
| $x_2$ | +1    | 0     | ?     |
| $x_3$ | -1    | ?     | 0     |

Corr

|       | $x_4$ | $x_5$ | $x_6$ |
|-------|-------|-------|-------|
| $x_4$ | 0     | +1    | ?     |
| $x_5$ | +1    | 0     | -1    |
| $x_6$ | ?     | -1    | 0     |



6. Assume that we have a Ridge regression problem with only one predictor, and the true model is linear *without an intercept*, i.e.  $Y = \beta_1 X + \epsilon$ . Assume that we have  $n$  samples,  $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$  and we want to find the  $\mathcal{L}_2$  regularized least squares estimate  $\hat{\beta}_1$  from the data.
- Formulate the objective function in terms of a candidate  $\hat{\beta}_1$  and  $x_i$ 's and  $y_i$ 's, which are known. Assume that the regularization parameter is  $\lambda$
  - Find  $\hat{\beta}_1$  in terms of  $\lambda$  and the data.

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

Name:

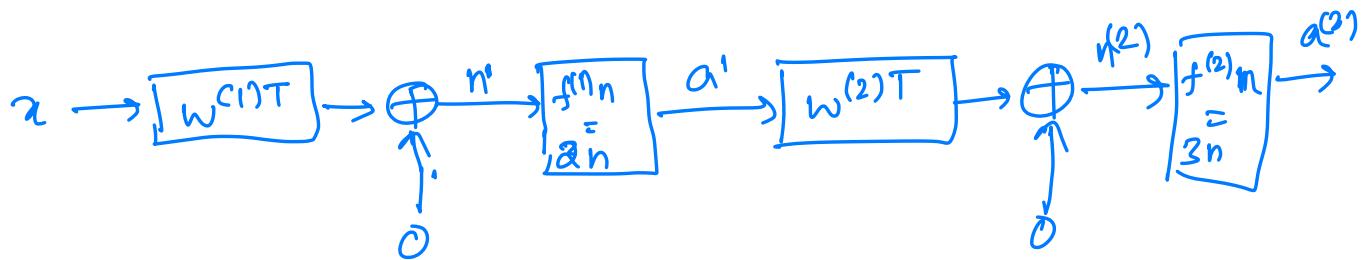
USC ID:

**Notes:**

- Write your name and ID number in the spaces above.
- No books, cell phones or other notes are permitted. Only one letter size cheat sheet (back and front) and a calculator are allowed.
- Problems are not sorted in terms of difficulty. Please avoid guess work and long and irrelevant answers.
- Show all your work and your final answer. Simplify your answer as much as you can.
- Open your exam only when you are instructed to do so.

| Problem | Score | Earned |
|---------|-------|--------|
| 1       | 20    |        |
| 2       | 25    |        |
| 3       | 20    |        |
| 4       | 20    |        |
| 5       | 25    |        |
| Total   | 110   |        |

1. Consider a MLP with one input, two layers, one neuron in each layer, and one output. The activation function of the first layer is  $f^{(1)}(n) = 2n$  and the activation function of the second layer is  $f^{(2)}(n) = 3n$ . The initial weights of the first and the second layers are respectively  $w^{(1)} = 1$  and  $w^{(2)} = -1$ . There are no bias terms, so  $b^{(1)} = b^{(2)} = 0$  and they are kept zero during training. Assume that we present the data point with  $x = 1$  and  $y = 2$  to the network. Perform one step of the Stochastic Gradient Descent algorithm by using the backpropagation algorithm, assuming the learning rate  $\alpha = 0.5$ . Use the objective function  $J = (y - a^{(2)})^2$ . This means that you should calculate the updated weights.



when  $n=1$  &  $y=2$

$$n^{(1)} = w^{(1)T} \cdot x = 1 \cdot 1 = 1$$

$$a^{(1)} = 2 \cdot 1 = 2$$

$$n^{(2)} = w^{(2)T} \cdot a^{(1)} = -1 \cdot 2 = -2$$

$$a^{(2)} = 3 \cdot (-2) = -6$$

$$J = (y - a^{(2)})^2 = (2 + 6)^2 = 64$$

To update errors we need to calculate  $\frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}$

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial a^{(2)}} \cdot \frac{\partial a^{(2)}}{\partial n^{(2)}} \cdot \frac{\partial n^{(2)}}{\partial w_2} \rightarrow ①$$

$$J = (y - a^{(2)})^2$$

$$\frac{\partial J}{\partial a_2} = 2 \cdot (y - a^{(2)}) \cdot (-1) = 2 (a^{(2)} - y)$$

$$= 2 (-6 - 2) = 2 (-8) = -16 \quad \xrightarrow{②}$$

$$a^{(2)} = 3 \cdot n^{(2)}$$

$$\frac{\partial a^{(2)}}{\partial n^{(2)}} = 3 \quad \rightarrow \textcircled{3}$$

$$n^{(2)} = a^{(1)} \cdot w^{(2)}$$

$$\frac{\partial n^{(2)}}{\partial w^{(2)}} = a^{(1)} = 2 \quad \rightarrow \textcircled{4}$$

Substituting ②, ③ ④ in ①

$$\frac{\partial J}{\partial w^{(2)}} = -16 \times 3 \times 2 = \underline{\underline{-96}}$$

$$\frac{\partial J}{\partial w^{(1)}} = \frac{\partial J}{\partial a^{(1)}} \cdot \frac{\partial a^{(1)}}{\partial n^{(1)}} \cdot \frac{\partial n^{(1)}}{\partial w^{(1)}} \quad \rightarrow \textcircled{5}$$

$$\frac{\partial J}{\partial a^{(1)}} = \frac{\partial J}{\partial a^{(2)}} \cdot \frac{\partial a^{(2)}}{\partial n^{(2)}} \cdot \frac{\partial n^{(2)}}{\partial a^{(1)}} \quad \rightarrow \textcircled{6}$$

$$n^{(2)} = w_i^{(2)T} \cdot a^{(1)}$$

$$\frac{\partial n^{(2)}}{\partial a^{(1)}} = w^{(2)} = -1$$

Substituting values in 8

$$\frac{\partial J}{\partial a^{(1)}} = -16 * 3 * -1 = 48$$

$$a^{(1)} = 2 \cdot n^{(1)}$$

$$\frac{\partial a^{(1)}}{\partial n^{(1)}} = 2 \cdot$$

$$n^{(1)} = w^{(1)} \cdot x$$

$$\frac{\partial n^{(1)}}{\partial w^{(1)}} = x = 1$$

$$\therefore \frac{\partial J}{\partial w^{(1)}} = 48 * 2 * 1 = 96$$

$$\therefore w^{(1)}_2 = w^{(1)} + \alpha \cdot \frac{\partial J}{\partial w^{(1)}} = 1 + 0.5 * 96 = 49$$

$$w^{(2)}_1 = w^{(2)} + \alpha \cdot \frac{\partial J}{\partial w^{(2)}} = -1 + 0.5 * (-96) = -1 - 48 = -49$$

2. We are trying to estimate the temperature of consecutive years based on *observations* on tree ring sizes. Possible ring sizes are Very Small = VS, Small = S, Medium = M, Large = L, and Very Large = VL. Years can be Cold = C or Hot = H. Assume that we observed VS, VL tree ring sizes in two consecutive years. Also, Assume that  $\pi = [0.8 \ 0.2]$  shows the initial distribution of C and H, respectively. Which of the following HMMs is more likely to have given rise to the observation  $O = \{VS, VL\}$  and why? First rows of  $A_1, B_1, A_2, B_2$  represent C and second rows represent H.

$$(a) \quad \begin{matrix} C & H \end{matrix} \quad \begin{matrix} VS & S & M & L & VL \end{matrix}$$

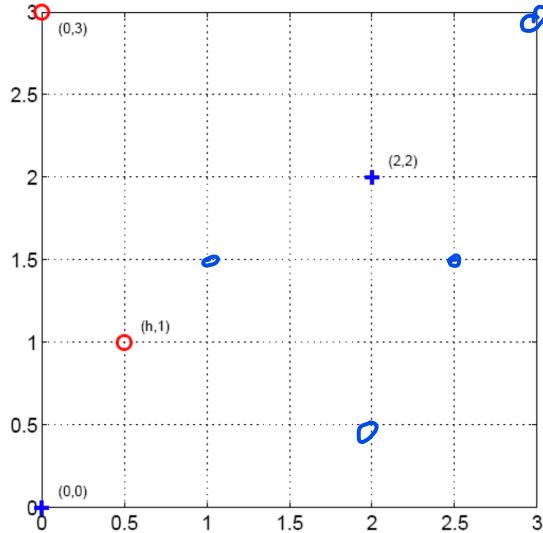
$$A_1 = \begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix} \quad B_1 = \begin{bmatrix} 0.2 & 0.3 & 0.3 & 0.1 & 0.1 \\ 0.3 & 0.2 & 0.1 & 0.1 & 0.3 \end{bmatrix}$$

$$(b) \quad \begin{matrix} C & H \end{matrix} \quad \begin{matrix} VS & S & M & L & VL \end{matrix}$$

$$A_2 = \begin{bmatrix} 0.5 & 0.5 \\ 0.6 & 0.4 \end{bmatrix} \quad B_2 = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.6 \\ 0.5 & 0.1 & 0.1 & 0.1 & 0.2 \end{bmatrix}$$

**Solution:**

3. Suppose we only have four training observations in two dimensions:

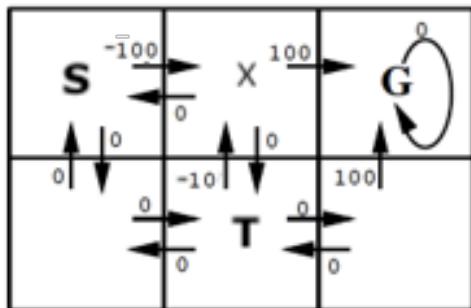


positive examples are  $\mathbf{x}_1 = [0 \ 0]^T$ ,  $\mathbf{x}_2 = [2 \ 2]^T$  and negative examples are  $\mathbf{x}_3 = [h \ 1]^T$ ,  $\mathbf{x}_4 = [0 \ 3]^T$ .  $h$  is a parameter.

- (a) What is the largest value of  $h$  for which the training data are still linearly separable?
- (b) Determine the support vectors when  $h = 0.5$ .
- (c) When the training points are separable, does the slope of the maximum margin classifier change? Why?
- (d) Assume that  $h = .5$  and we have unlabeled data  $\mathbf{x}_5 = [3 \ 3]^T$ ,  $\mathbf{x}_6 = [2 \ 0.5]^T$ ,  $\mathbf{x}_7 = [1 \ 1.5]^T$ ,  $\mathbf{x}_8 = [2.5 \ 1.5]^T$ . Which one will be labeled first, if we are performing self-training? Which one will be labeled first, if we are performing active learning?

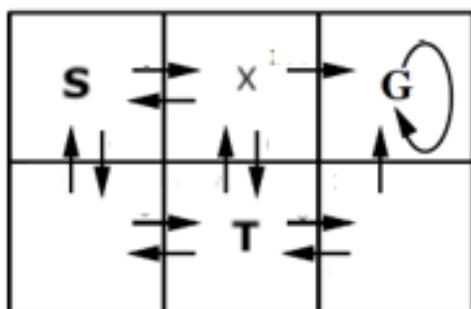
4. Consider the unlabeled dataset with one feature:  $\{1, 5, 6, 21, 27\}$ .
- (a) What will single linkage (minimum distance between members of clusters), complete linkage (maximum distance between members of clusters), and average linkage (average distance between members of clusters) output as the two clusters?
  - (b) Assume that we want to perform k-medoids clustering and initialized the algorithm with two clusters randomly and we got  $\{1, 5, 6, 27\}$  and  $\{21\}$  as the initial clusters. What are the final two clusters?

5. Consider the following grid world, in which an agent can explore the environment until it finds the Goal (G). In this problem, you will update the estimates of the  $Q$  function based on experiences of the agent. In this environment, all actions in all squares result in a zero reward, *except the actions that result in entering the goal square and the actions that result entering the danger square X that result in a punishment, i.e. negative reward*. The rewards  $r(s, a)$  of each action  $a$  in state  $s$  was shown in the below figure. Assume that the initial estimate  $\hat{Q}(s, a)$  is zero for all state and action pairs.



$r(s, a)$  (immediate reward) values

The agent starts at S and performs the following actions:  $\rightarrow, \downarrow, \uparrow, \rightarrow$ . Then it starts from T and performs  $\uparrow, \rightarrow$ . Show the updated estimates  $\hat{Q}(s, a)$  for all states after the agent completes both paths. Use discount factor  $\gamma = 0.9$  if needed. You can use the following table in each update to show  $\hat{Q}(s, a)$  for each state and action pair.



$\hat{Q}(s, a)$  values

**Solution:**

Scratch paper

Name:

USC ID:

Scratch paper

Name:

USC ID:

1. For the following classification problem, design a single-layer perceptron that has zero error on training set.

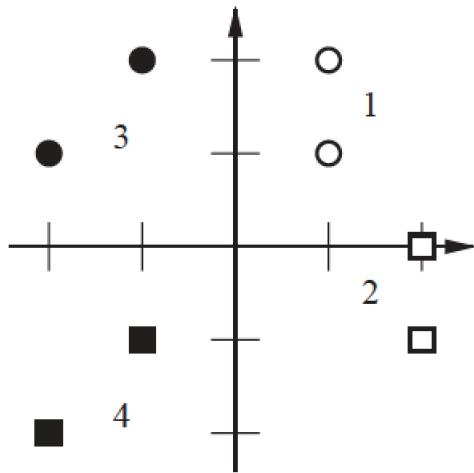
$$C_1 = \left\{ \mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right\}$$

$$C_2 = \left\{ \mathbf{x}_3 = \begin{bmatrix} 2 \\ -1 \end{bmatrix}, \mathbf{x}_4 = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \right\}$$

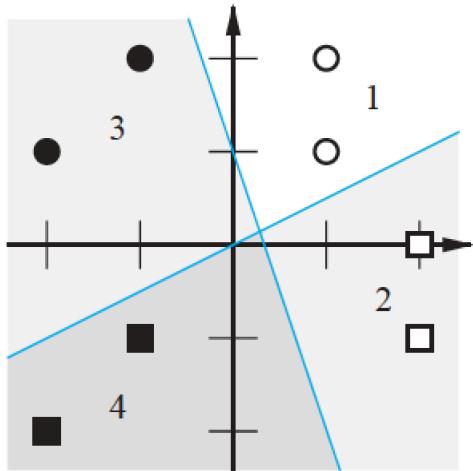
$$C_3 = \left\{ \mathbf{x}_5 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}, \mathbf{x}_6 = \begin{bmatrix} -2 \\ 1 \end{bmatrix} \right\}$$

$$C_4 = \left\{ \mathbf{x}_3 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \mathbf{x}_4 = \begin{bmatrix} -2 \\ -2 \end{bmatrix} \right\}$$

**Solution:** The patterns are shown in the following figure:



And tentative decision boundaries can be those in the following figure:



It is left to the student to verify that a single layer perceptron with two neurons can classify the vectors given that the four classes are modeled using vectors

$$C_1 \rightarrow t_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$$C_2 \rightarrow t_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$C_3 \rightarrow t_3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$C_4 \rightarrow t_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

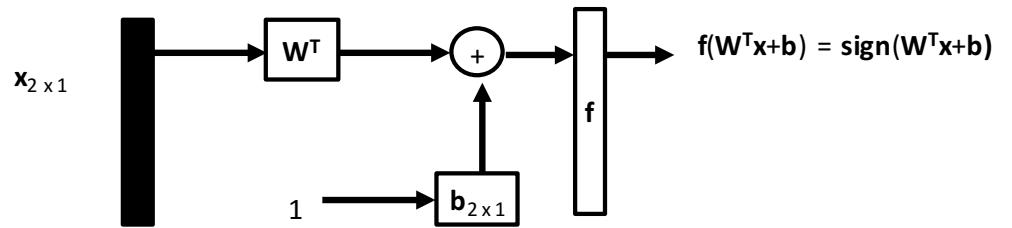
and the weight vectors are columns of the following matrix

$$\mathbf{W} = \begin{bmatrix} -3 & 1 \\ -1 & -2 \end{bmatrix}$$

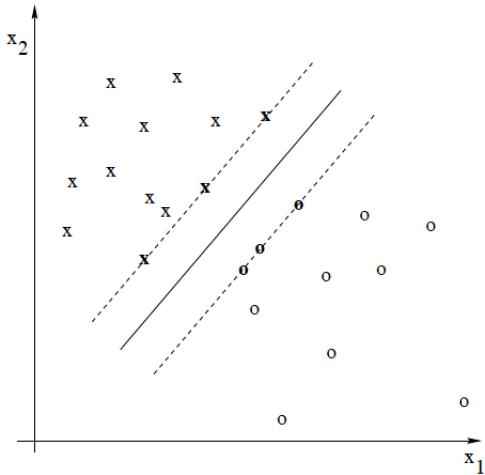
and the biases in the vector:

$$\mathbf{b} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The architecture of the Perceptron is:



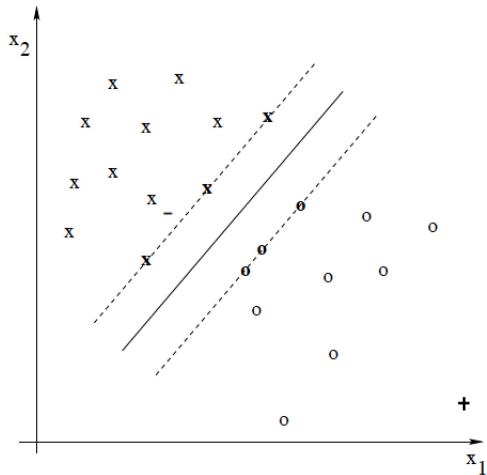
2. Answer the following questions about the figure:



- (a) What is the leave-one-out cross-validation error estimate for maximum margin separation in the figure? (The answer is a number)
- (b) True or False? We would expect the support vectors to remain the same in general as we move from a linear kernel to higher order polynomial kernels.
- (c) If each of the classes is modeled with a normal distribution and the variances are assumed equal, how many parameters are needed to describe the decision boundary for classification?
- (d) Assume that all data points *outside* the margin are unlabeled. Mark on the figure with a - the first data point that has to be labeled, when implementing active learning. Also, mark on the figure with a + the first data point that has to be labeled, when implementing self-training.

**Solution:**

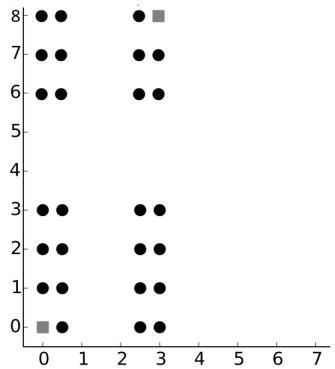
- (a) Based on the figure we can see that removing any single point would not chance the resulting maximum margin separator. Since all the points are initially classified correctly, the leave-one-out error is zero.
- (b) There are no guarantees that the support vectors remain the same. The feature vectors corresponding to polynomial kernels are non-linear functions of the original input vectors and thus the support points for maximum margin separation in the feature space can be quite different.
- (c) Modeling classes with normal densities with equal variances creates linear decision boundaries. A linear decision boundary with two variables  $x_1$  and  $x_2$  is described by three parameters  $(w_0, w_1, w_2)$ , i.e.:  $g(x_1, x_2) = w_0 + w_1x_1 + w_2x_2$ .



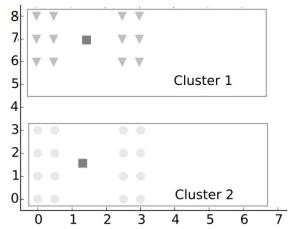
(d)

3. Consider the unlabeled two-dimensional data represented on the following figure.

- (a) Using the two points marked as squares as initial centroids, draw (on that same figure) the clusters obtained after one iteration of the k-means algorithm ( $k = 2$ ).
- (b) Does your solution change after another iteration of the k-means algorithm?



**Solution:**



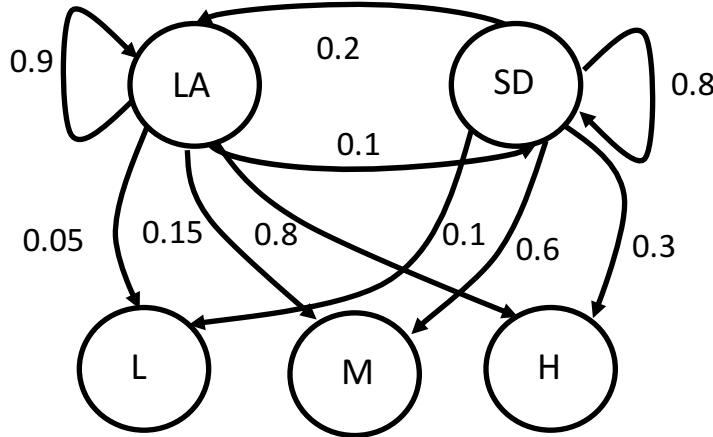
(a)

(b) No.

4. A company with headquarters in the Bay Area has two offices in Los Angeles and San Diego. An employee in San Diego office is sent to the Los Angeles office the next day with probability 0.2 and stays in San Diego office with probability 0.8. An employee in Los Angeles office is sent to the San Diego office with probability 0.1 and stays in Los Angeles office with probability 0.9. A new employee is assigned to Los Angeles office with probability 0.7 and to San Diego office with probability 0.3. An employee in San Diego office works between six and eight hours per day with probability 0.6, works more than eight hours with probability 0.3, and works less than six hours per day with probability 0.1. An employee in Los Angeles office works between six and eight hours per day with probability 0.15, works more than eight hours with probability 0.8, and works less than six hours per day with probability 0.05. A manager in the headquarters can only observe the number of hours each employee worked each day.
- (a) Construct a Hidden Markov Model that models the observations of the manager in their headquarters. Clearly show the parameters with matrices and vectors and draw a state transition graph for the model.
  - (b) If the manager observes the number of hours a new employee worked in the first three consecutive days of work to be 5, 9, 7, what is the most likely sequence of places at which the employee worked in those three days?
  - (c) What sequence of three places has the maximum expected number of correct places?

**Solution:**

- (a) The figure below shows the HMM model of the problem. We used  $L, M, H$  to show working less than six hours, between six and eight hours, and more than eight hours per day, respectively.



The parameters will be:

$$\begin{aligned}\pi_{x_0} &= \begin{bmatrix} 0.7 & 0.3 \end{bmatrix} \\ A &= \begin{bmatrix} 0.9 & 0.1 \\ 0.2 & 0.8 \end{bmatrix} \\ B &= \begin{bmatrix} 0.05 & 0.15 & 0.8 \\ 0.1 & 0.6 & 0.3 \end{bmatrix}\end{aligned}$$

where the rows show LA and SD.

- (b) The sequence of observations is  $L, H, M$ . We do not need to use the Viterbi algorithm here. An exhaustive search is easy:

| State             | Probability  |
|-------------------|--|
| LA, LA, LA        | $0.7 \times 0.9 \times 0.9 \times 0.05 \times 0.15 = 0.003402$                   |
| LA, LA, SD        | $0.7 \times 0.9 \times 0.1 \times 0.05 \times 0.8 \times 0.6 = 0.001512$         |
| LA, SD, SD        | $0.7 \times 0.1 \times 0.8 \times 0.05 \times 0.3 \times 0.6 = 0.000504$         |
| LA, SD, LA        | $0.7 \times 0.1 \times 0.2 \times 0.05 \times 0.3 \times 0.15 = 0.0000315$       |
| <b>SD, SD, SD</b> | $0.3 \times 0.8 \times 0.8 \times 0.1 \times 0.3 \times 0.6 = \mathbf{0.003456}$ |
| SD, LA, SD        | $0.3 \times 0.2 \times 0.1 \times 0.1 \times 0.8 \times 0.6 = 0.000288$          |
| SD, SD, LA        | $0.3 \times 0.8 \times 0.2 \times 0.1 \times 0.3 \times 0.15 = 0.000216$         |
| SD, LA, LA        | $0.3 \times 0.2 \times 0.9 \times 0.1 \times 0.8 \times 0.15 = 0.000648$         |

Therefore, the most likely sequence of places is SD, SD, SD.

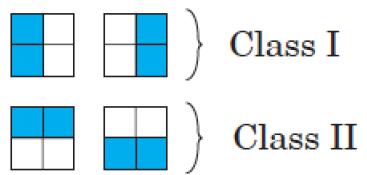
- (c) For the maximum number of correct states (Expectation Maximization) we calculate the probability of each of states at each step from the table above:<sup>1</sup>

<sup>1</sup>Note that in principle, the probabilities have to be normalized and then summed up, because we are essentially calculating the conditional probability of each state occurring at each time step, *given the sequence of observations*. Nevertheless, normalization will not change the solution, so for simplicity, we do not do it.

| State       | 0                | 1                | 2                |
|-------------|------------------|------------------|------------------|
| $P(LA LHM)$ | <b>0.5418345</b> | <b>0.5816555</b> | 0.4272931        |
| $P(SD LHM)$ | 0.458166         | 0.4183445        | <b>0.5727069</b> |

Hence, the answer using this method is LA, LA, SD.

5. Consider the two classes of patterns that are shown in the figure below. Design a multilayer network to distinguish these categories.

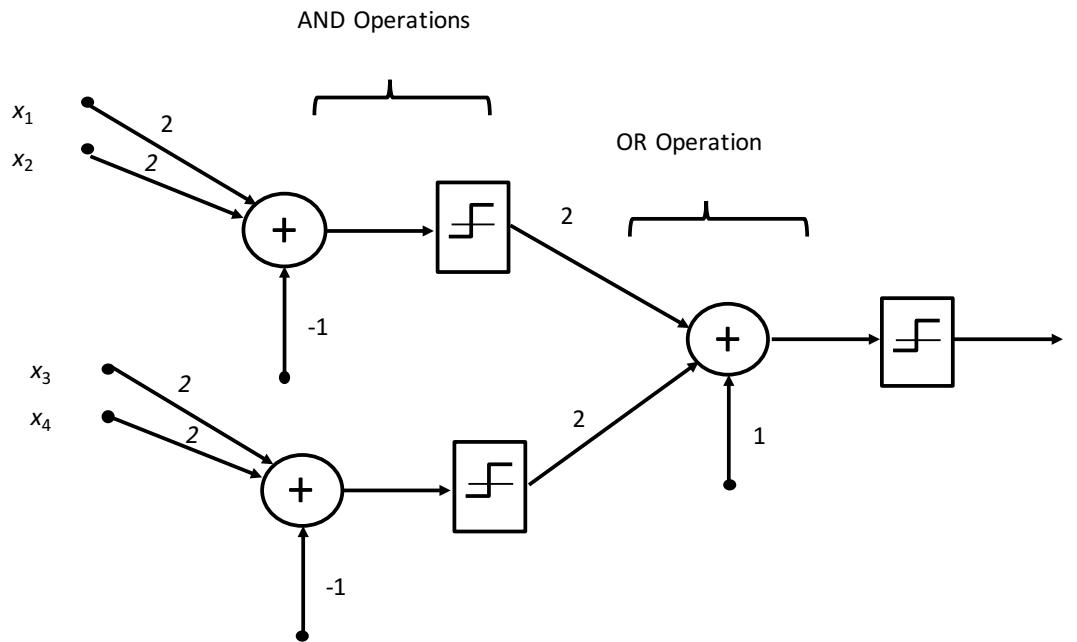


**Solution:**

There are many different multilayer networks that could solve this problem. The vectors in each class can be represented as:

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{x}_3 = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{x}_4 = \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

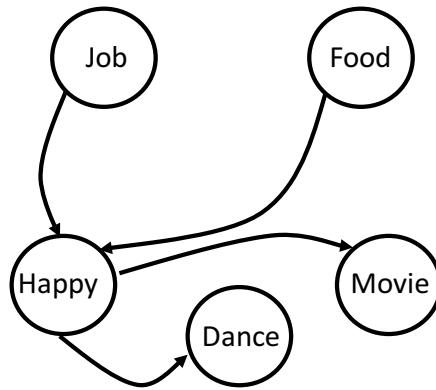
We will design a network by first noting that for the Class I vectors either the first two elements or the last two elements will be “1”. The Class II vectors have alternating “1” and “-1” patterns. This leads to the network shown in the figure below. Note that  $x_1, x_2, x_3, x_4$  denote the four elements in each of the vectors  $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ .



The first neuron in the first layer tests the first two elements of the input vector. If they are both “1” it outputs a “1”, otherwise it outputs a “-1”. The second neuron in the first layer tests the last two elements of the input vector in the same way. Both of the neurons in the first layer perform AND operations. The second layer of the network tests whether either of the outputs of the first layer are “1”. It performs an OR operation. In this way, the network will output a “1” if either the first two elements or the last two elements of the input vector are both “1”.

6. The following diagram shows a person's state of mind and the actions they may perform due to happiness. Assume that

$$\begin{aligned}
 P(J) &= 0.1 \\
 P(F) &= .8 \\
 P(H|J, F) &= 0.9 \\
 P(H|J, \sim F) &= .8 \\
 P(H|\sim J, F) &= 0.7 \\
 P(H|\sim J, \sim F) &= 0.1 \\
 P(M|H) &= 0.7 \\
 P(M|\sim H) &= 0.2 \\
 P(D|H) &= 0.3 \\
 P(D|\sim H) &= 0.05
 \end{aligned}$$



- (a) Calculate the probability that the person watches a movie.
- (b) Show that if the person is happy, having had food explains away finding a job.
- (c) Calculate the probability that the person goes to a movie and dances given that the person is happy.

**Solution:**

(a) By the law of total probability we have:

$$P(M) = P(M|H)P(H) + P(M|\sim H)P(\sim H)$$

Therefore, we need  $P(H)$ :

$$\begin{aligned} P(H) &= P(H|J, F)P(J)P(F) + P(H|\sim J, F)P(\sim J)P(F) \\ &\quad + P(H|J, \sim F)P(J)P(\sim F) + P(H|\sim J, \sim F)P(\sim J)P(\sim F) \\ &= 0.9(0.1)(0.8) + (0.7)(0.9)(0.8) + (0.8)(0.1)(0.2) + (0.1)(0.9)(0.2) = 0.61 \end{aligned}$$

Hence

$$P(M) = (0.7)(0.61) + (0.2)(0.39) = 0.505$$

(b) We need to show  $P(F|H) > P(F|H, J)$ . By Bayes' Rule:

$$P(F|H) = \frac{P(H|F)P(F)}{P(H)}$$

But by the law of total probability:

$$P(H|F) = P(H|J, F)P(J) + P(H|\sim J, F)P(\sim J) = (0.9)(0.1) + (0.7)(0.9) = 0.72$$

Therefore:

$$P(F|H) = \frac{(0.72)(0.8)}{0.61} = 0.9442623$$

Next

$$P(F|H, J) = \frac{P(H|J, F)P(J, F)}{P(H, J)} = \frac{P(H|J, F)P(F|J)P(J)}{P(H|J)P(J)} = \frac{P(H|J, F)P(F|J)}{P(H|J)}$$

On the other hand  $P(F|J) = P(F)$ , because  $F$  and  $J$  are independent. Also, by the law of total probability,  $P(H|J) = P(H|J, F)P(F) + P(H|J, \sim F)P(\sim F) = (0.9)(0.8) + (0.8)(0.2) = 0.88$ .

Therefore,

$$P(F|H, J) = \frac{P(H|J, F)P(F|J)}{P(H|J)} = \frac{P(H|J, F)P(F)}{P(H|J)} = \frac{(0.9)(0.8)}{0.88} = 0.81818182$$

Obviously,  $P(F|H) > P(F|H, J)$ .

(c) Given  $H$ ,  $M$  and  $D$  are conditionally independent

$$\begin{aligned} P(M, D|H) &= \frac{P(M, D, H)}{P(H)} = \frac{P(H)P(M|H)P(D|H)}{P(H)} = P(D|H)P(M|H) \\ &= (0.3)(0.7) = 0.21 \end{aligned}$$