UCLA Anderson
SCHOOL of MANAGEMENT

MGMTMFE 431:

Data Analytics and Machine Learning

Topic 9: Support Vector Machines, Neural Networks and Deep Learning

Spring 2019

Professor Lars A. Lochstoer

Agenda

a. Group presentations

b. Brief discussion of Support Vector Machines

c. Brief discussion of Neural Networks and Deep Learning

UCLA Anderson SCHOOL of MANAGEMENT

b. Support Vector Machines

Supervised learning technique

- Nonlinear classification
- We will cover two classes (e.g., default vs no default)
- Extension to multiple classes available

Developed in the Computer Science community

- Proved successful
- Applications spread to other fields
- Standard tool in the Machine Learning toolkit

UCLA Anderson
SCHOOL of MANAGEMENT

b. Hyperplane

Consider *P* features X_1, …, X_P.

- A hyperplane is the *P-1* dimensional subspace:

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P = 0$$

If two features (two dimensions, the hyperplane is a one-dimensional line)

Use as classifier:

- If $\beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P > 0$ classify as (say) 1
- If $\beta_0 + \beta_1 X_1 + \cdots + \beta_P X_P \leq 0$ classify as (say) 0

UCLA Anderson
SCHOOL of MANAGEMENT

b. Separating hyperplanes

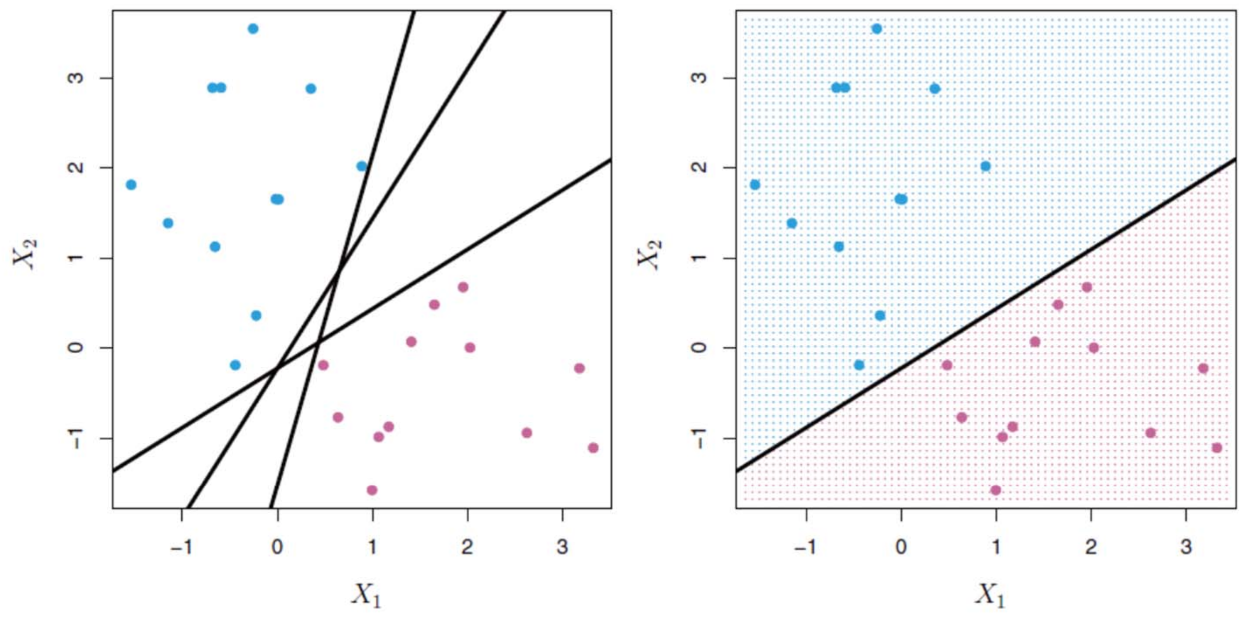


FIGURE 9.2. Left: *There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black.* Right: *A separating hyperplane is shown in black. The blue and purple grid indicates the decision rule made by a classifier based on this separating hyperplane: a test observation that falls in the blue portion of the grid will be assigned to the blue class, and a test observation that falls into the purple portion of the grid will be assigned to the purple class.*

b. Perfect linear separation may not exist

But, a separating hyperplane may not exist!

- Need a “soft” classifier (allow some observations to be off)

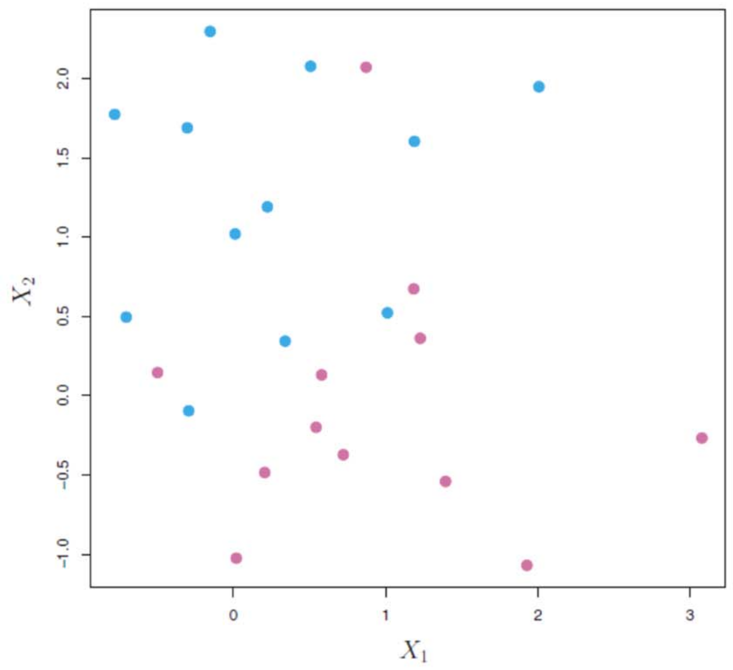


FIGURE 9.4. *There are two classes of observations, shown in blue and in purple. In this case, the two classes are not separable by a hyperplane, and so the maximal margin classifier cannot be used.*

b. Support Vector Classifier

Allow for "support vectors" or *margins* where errors are tolerated

- The width of these margins (support vectors) is a tuning parameter found through cross-validation
- Draw the optimal hyperplane where we allow for misclassification within the margins.
- Constraint in betas: L2 (like ridge regression)

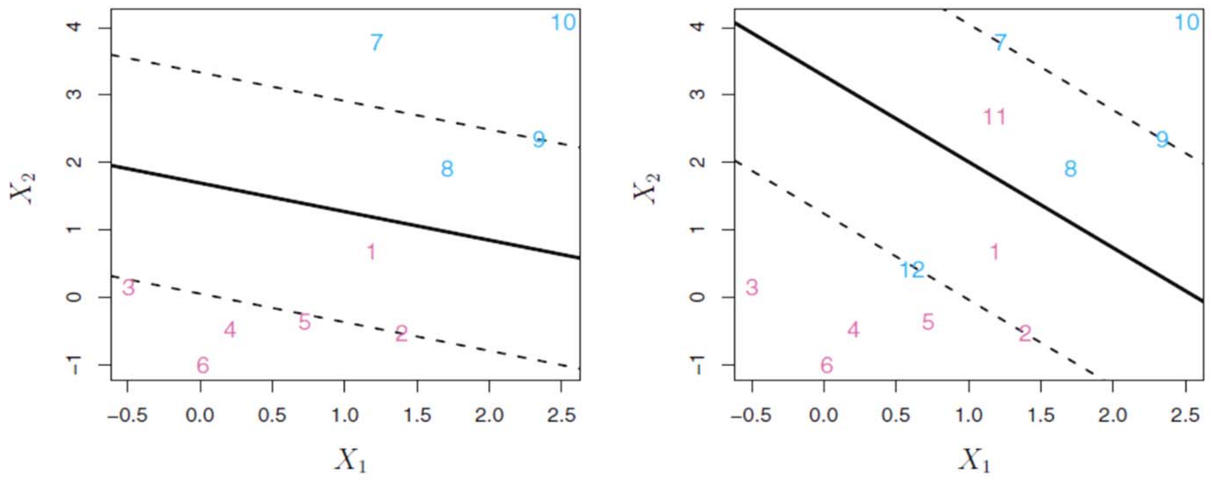


FIGURE 9.6. Left: *A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines.* Purple observations: *Observations* 3, 4, 5, *and* 6 *are on the correct side of the margin, observation* 2 *is on the margin, and observation 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.*

UCLA Anderson
SCHOOL of MANAGEMENT

b. Support Vector Machine

Sometimes linear classification is not appropriate given the data

- Support Vector Machines (SVMs) allow for nonlinear boundaries
- Popular functional forms are *polynomial* and *radial*
- These kernels require another tuning parameter (polynomial order of polynomial kernel or sensitivity in radial kernel)
- See section 9.3 in Introduction to Statistical Learning for details
- I will show graphically the outcomes of these two kernels and then present an example

UCLAAnderson
SCHOOL of MANAGEMENT

b. Support Vector Machine

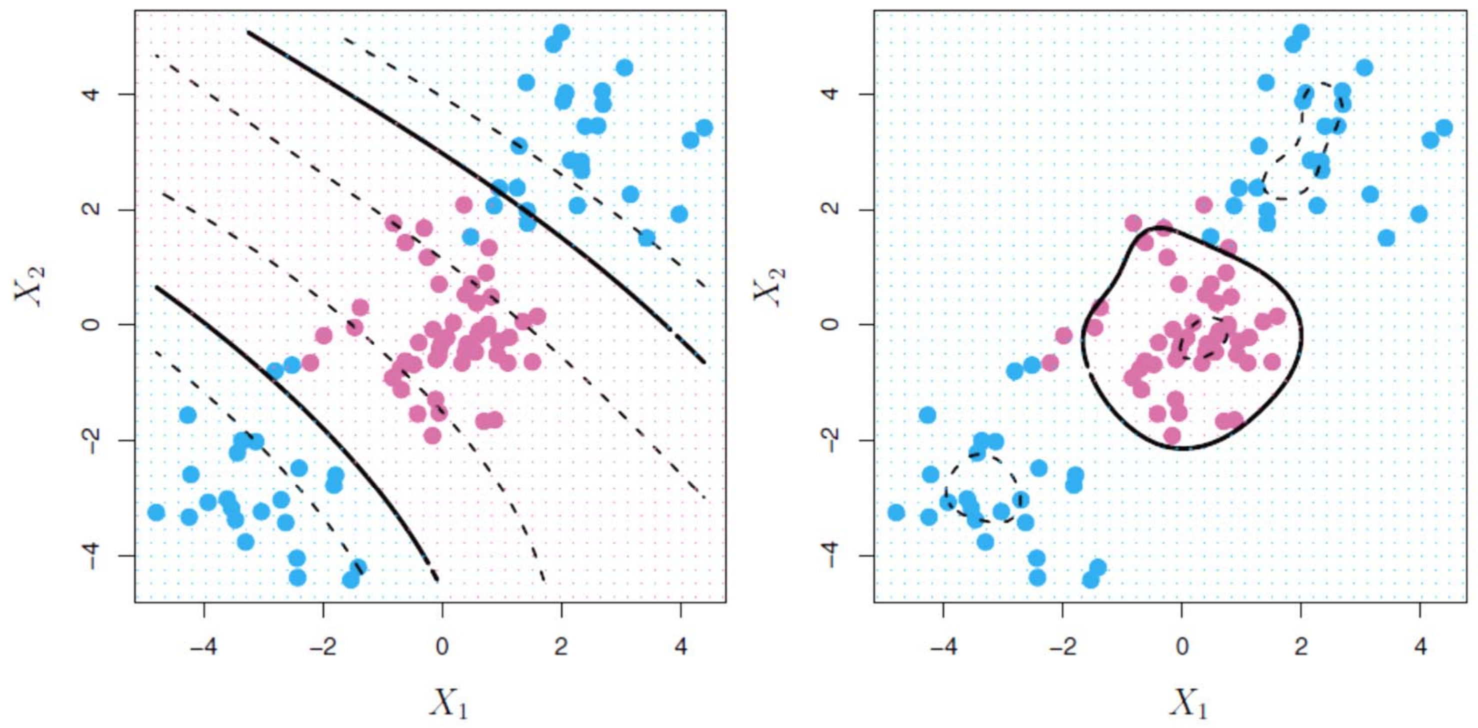


FIGURE 9.9. Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

UCLAAnderson
SCHOOL of MANAGEMENT

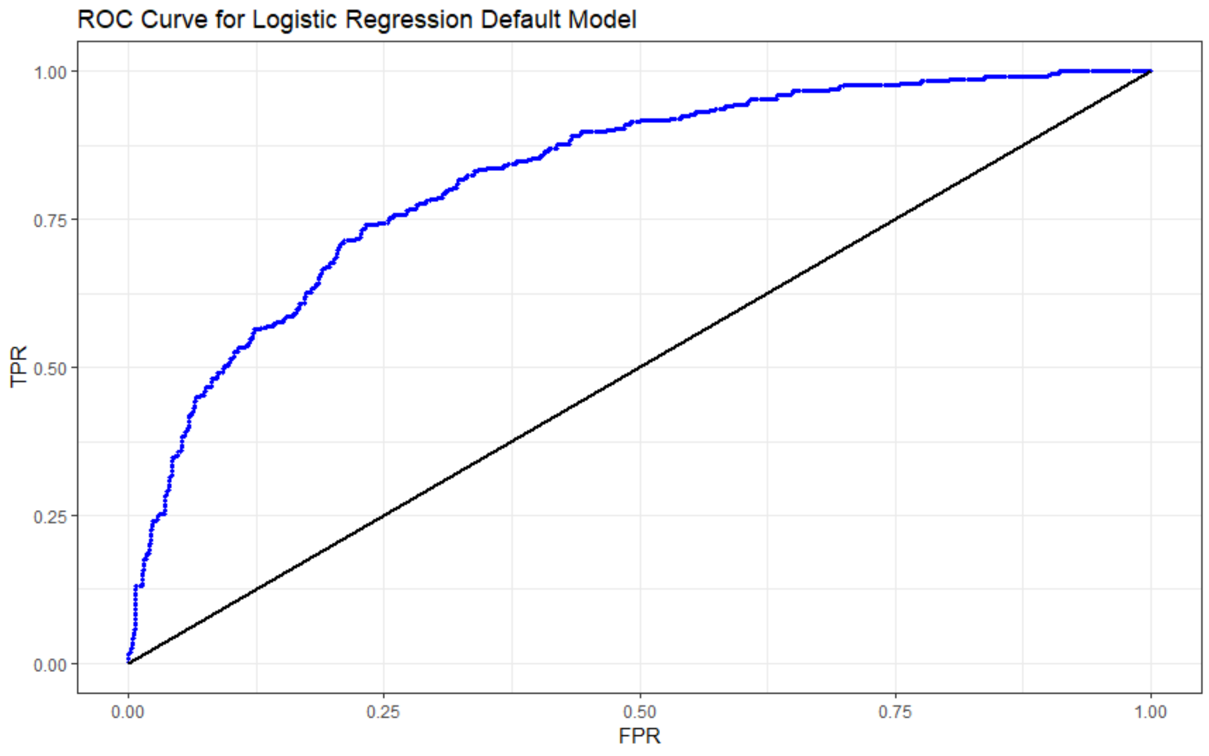
b. Support Vector Machine: Example

Use Loans data as in Topic 3

- Recall logistic regression model:

```
> x_data = loans[, c("default","StatChkA","CrdHist","Sav_Bnd","OthrDebt","OthrInstall“
            ,"Purpose","Duration","CrdAmt","InstallRate","Pstatus","Foreign")]
➢ outloans1=glm(default~.,data=x_data,family="binomial")

phat=predict(outloans1,type="response")
```



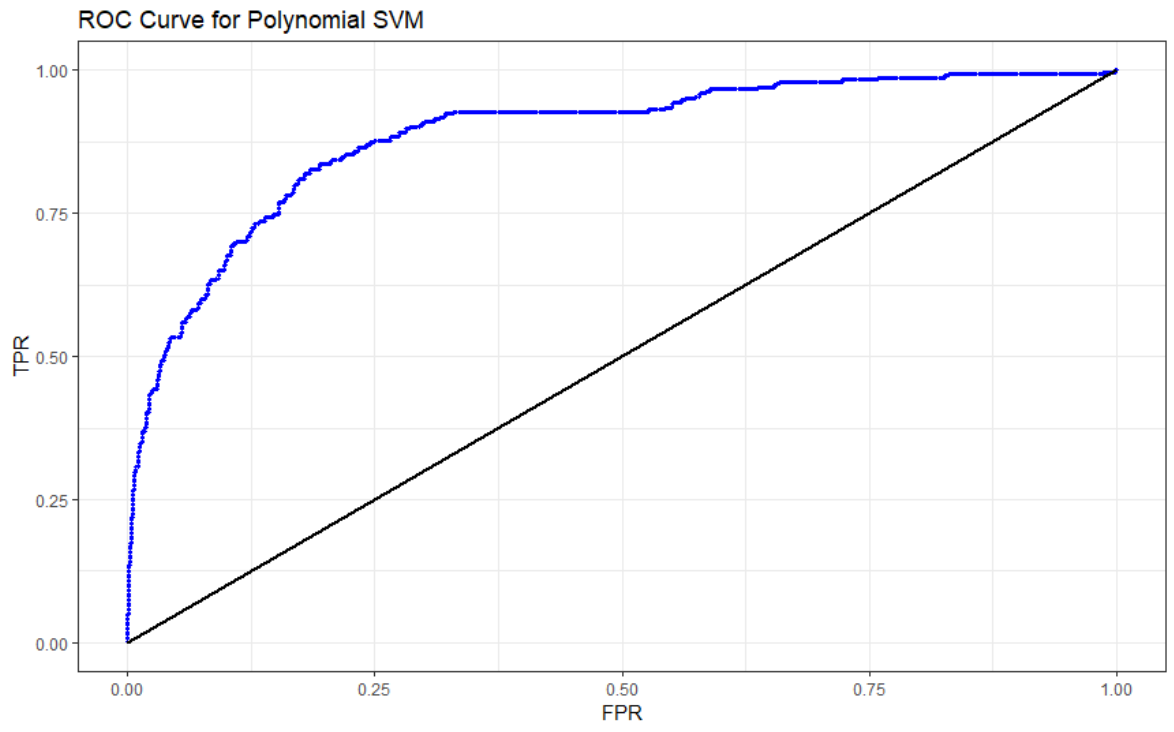
UCLAAnderson
SCHOOL of MANAGEMENT

b. Support Vector Machine: Example

Now fit two SVMs, polynomial and radial

- Use package e1071; First fit Polynomial
- Better than logistic regression

```
➢ require(e1071)
➢ outloans_svm_poly = svm(default~.,data=x_data,kernel = "polynomial",
            degree = 3, gamma = 1/11, probability = TRUE)
➢ phat_svm_poly=predict(outloans_svm_poly,x_data, probability = TRUE)
➢ svm_poly_roc <- simple_roc(x_data$default=="1", phat_svm_poly)
➢ q_poly <- qplot(FPR,TPR,xlab="FPR",ylab="TPR",col=I("blue"),main="ROC Curve for Polynomial SVM",size=I(0.75))
```



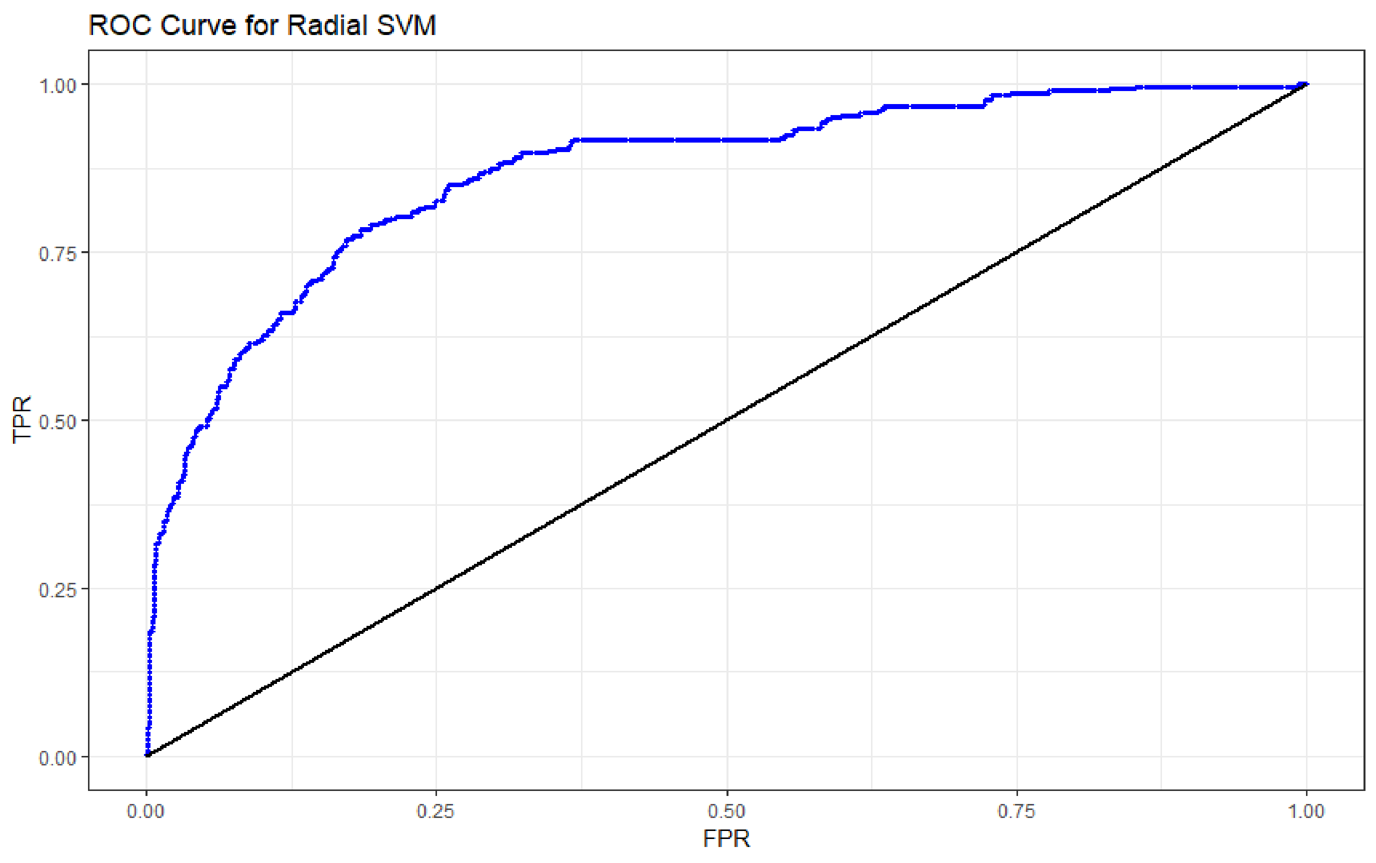
UCLAAnderson
SCHOOL of MANAGEMENT

b. Support Vector Machine: Example

Next, fit radial SVM

Better than logistic regression, but not quite as good as polynomial

```
> outloans_svm_radi = svm(default~.,data=x_data,kernel = "radial", gamma = 1/11, probability = TRUE)
➢ phat_svm_radi=predict(outloans_svm_radi,x_data, probability = TRUE)
➢ svm_radi_roc <- simple_roc(x_data$default=="1", phat_svm_radi)
➢ q_radi <- qplot(FPR,TPR,xlab="FPR",ylab="TPR",col=I("blue"),main="ROC Curve for Radial SVM",size=I(0.75))
```



UCLA Anderson
SCHOOL of MANAGEMENT

c. Neural Networks

"Deep learning" a current buzz-word in Machine Learning

- Based on multiple layers of neural networks

Multi-layer neural nets have shown good performance

- Used in language and image recognition (classification)
- Very 'black box'-y. Hard to understand exactly why something works
- Lots of training data could train very complicated, nonlinear models

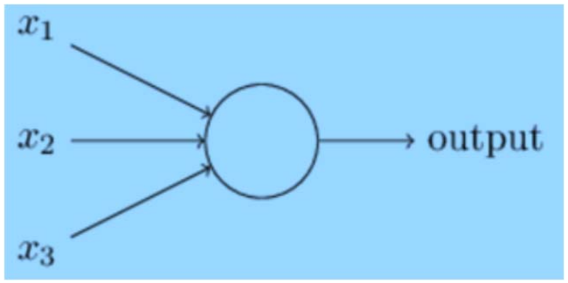
UCLA Anderson SCHOOL of MANAGEMENT

c. Perceptron

Input: binary data (x)

Output: binary 'decision'

Weights: *w*



$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

Typical output equation:

$$\text{output} = \begin{cases} 0 & \text{if } w \cdot x + b \leq 0 \\ 1 & \text{if } w \cdot x + b > 0 \end{cases}$$

c. Activation function: sigmoid

Want activation function (output function) that takes continuous inputs and has small change in prediction for output

- Use the logistic function

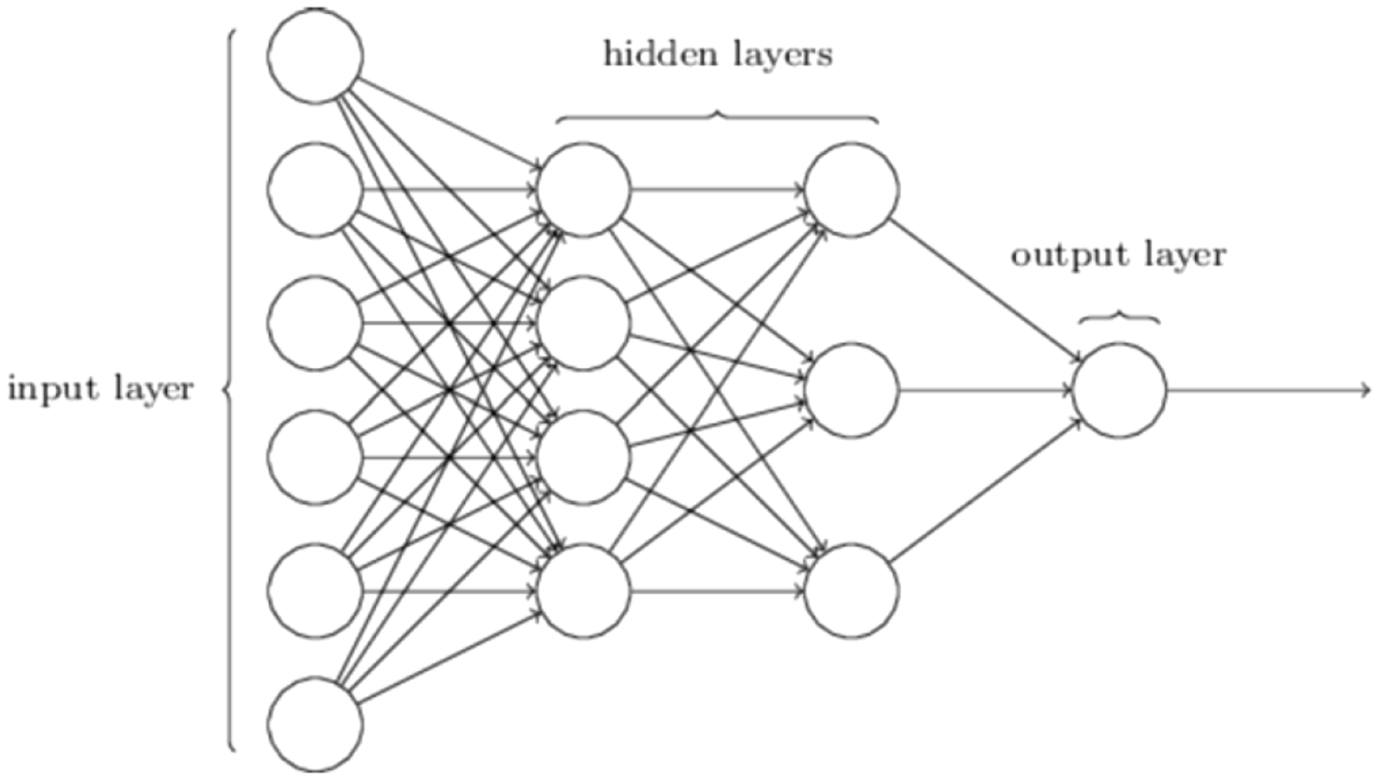
$$\frac{1}{1+\exp(-\sum_j w_j x_j - b)}.$$

UCLA Anderson SCHOOL of MANAGEMENT

c. Multi-layered network

This is the 'deep' part

- Even if used with sigmoids, often called multi-layered perceptrons (MLPs)



c. Deep learning packages

Neuralnet, nnet, kerasR, etc

- Need to specify number of layers and number of nodes in each layer
- Can be hard to calibrate
- I have posted a reference in Week 9
    - “Neural Networks and Deep Learning” which references neuralnetworksanddeeplearning.com
    - This site has more background info
    - I have also posted to package documentation for the NeuralNet package