
DATA7001

INTRODUCTION TO DATA SCIENCE

Module 4 Making the Data
Confess Part 4

Module Topics

- Hindsight (search and query),
 - What happened?
- **Insight (knowledge discovery)**
 - Why is it happening?
- **Foresight (prediction)**
 - What will happen?

→ What should happen (making it 'actionable')

Recap

- Setting the expectation and objectives
 - All models are wrong, some are useful.
 - Diagnostic analysis (insight), predictive analysis (foresight), prescriptive analysis
 - Focus on conceptual understanding, not the mathematics
- Overview of machine learning
 - Supervised, semi-supervised, and unsupervised learning
 - Learning as data collection -> choosing objective -> optimization
 - Two cultures: traditional statistics vs predictive
 - Statistical theory of learning and prediction
- Regression
 - Nearest neighbor regression
 - Simple and multiple linear regression
 - Nonlinear regression via basis expansion

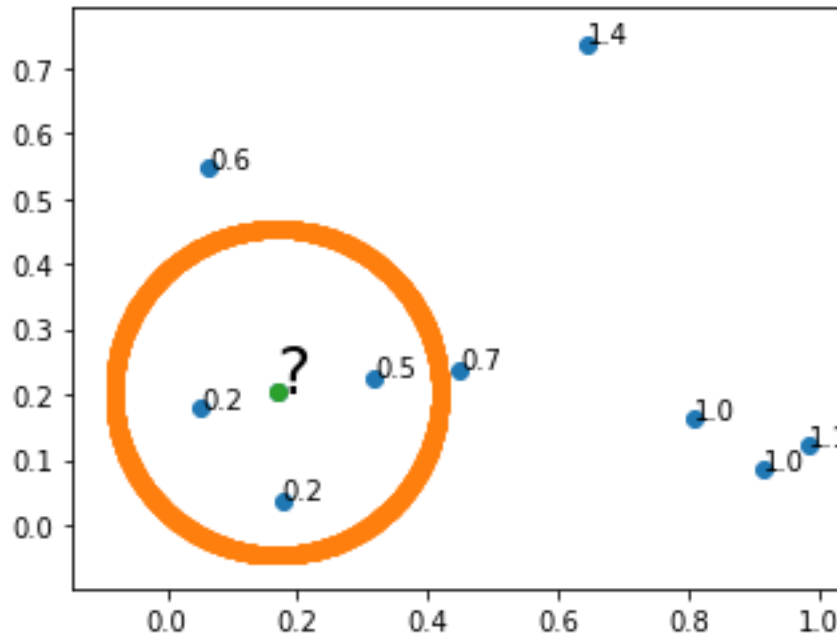
Recall: Nearest Neighbor Regression

- The idea in a Chinese proverb

近朱者赤，近墨者黑

(English: one takes on the color of one's company)

- Formally, given a set of training examples, to predict output for x
 - Find the k nearest neighbors of x (using some distance measure)
 - Predict the average output value for these k examples

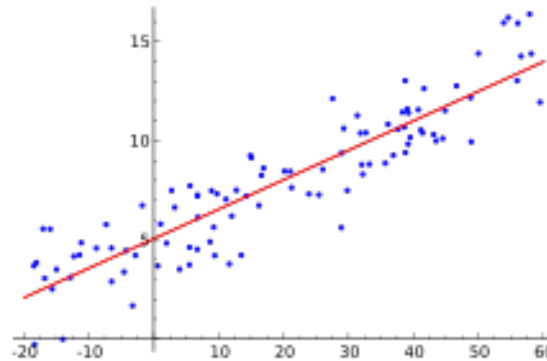


Recall: Linear Regression

- Output is assumed to relate to the predictors by

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon$$

where $\epsilon \sim N(0, \sigma^2)$



source: https://en.wikipedia.org/wiki/Linear_regression

- Questions
 - How to train the model?
 - How to compute goodness of fit?
 - How to validate model assumptions?
 - How to assess uncertainty in model parameters?
 - How to check whether a predictor (or all predictors) is significant?
 - How to make predictions

Recall: Non-linear Regression via Linear Regression

- We can add derived features (like interaction terms) and perform linear regression to obtain nonlinear models
- E.g. the following model can be trained using linear regression but it is nonlinear (with respect to the original predictors)

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2$$

What Will be Covered

- Overview of machine learning
- Regression
- Classification
 - Bayes optimal classifier
 - KNN
 - Logistic regression
 - Support vector machines
- Clustering
- Model selection

Classification

- In classification, we want to find out the relationship between the predictors and a categorical output.
 - E.g. determine the digit contained in a digit image

Digit recognition: image to $\{0, 1, \dots, 9\}$.



Spam filter: email to $\{\text{spam}, \text{not spam}\}$.



[Optional] Optimal Classifier

- If you know that my coin is biased with probability 0.9 of getting a head, you will always predict head to maximize your chances of being correct.
- In general, if we know the class distribution $P(y | x)$, then the classifier with minimum classification error is

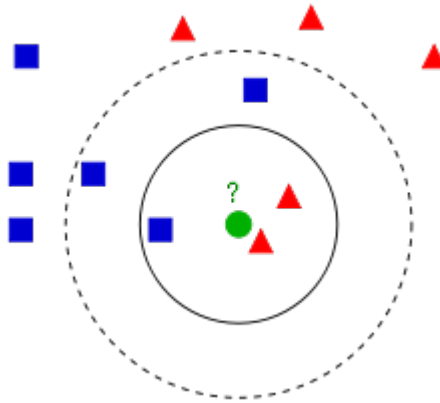
$$c(x) = \operatorname{argmax}_y P(y | x)$$

- The error of the optimal classifier is called the Bayes error rate.
- Question
 - If the probability that tomorrow is rainy, sunny, or cloudy is 0.5, 0.3, and 0.2 respectively. What is the weather condition that you should predict to minimize your prediction error? What is the Bayes error rate?

- While the Bayes optimal classifier can minimize the expected classification error, we can't compute it because we don't know the true class distribution.
- This motivates the following approach
 - estimate the true class distribution first,
 - then predict the most likely class according to the estimated distribution
 - E.g. kNN, logistic regression

Nearest neighbor Classifier

- kNN predicts the class of a test point x as follows
 - Find the k nearest neighbors of x
 - Assign x to the most commonly-occurring class of these neighbors



[source: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm]

- Essentially, kNN approximates the class distribution $P(y | x)$ by the class distribution for the nearest neighbors.
- Various distance can be used to find nearest neighbors
 - Euclidean, Manhattan, edit, ...
- It is important that the predictors are on the same scale
 - Having a predictor in cm and another in meter can give misleading answers
 - Often we normalize the predictors first
 - Same remarks apply to kNN regression

- (Curse of dimension) For high-dimensional data
 - kNN often requires a lot of examples to perform well
 - It is computationally very demanding
- Questions
 - What is the accuracy of 1-NN on the training set?
 - Is 1-NN the best classifier?

Logistic Regression

- Logistic regression aims to mimic the optimal classifier by learning $P(y | x)$.

- For a *binary* $Y \in \{0, 1\}$, it assumes $P(Y = 1|x)$ is

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

- Equivalently, log-odds is linear:

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

- Interpretation of parameters

- β_0 is the log-odds when $x=0$.
- $\beta_1 > 0$: increasing x increases the probability of positive
- $\beta_1 < 0$: increasing x decreases the probability of positive
- One unit increase in x leads to an increase of β_1 in the log-odds

- Learning

- Parameters found by *Maximum Likelihood*

- Prediction

- Given a logistic regression model $p(X)$, classify X as follows:
 - If $p(X) > p^*$, then $Y = 1$; otherwise $Y = 0$.
 - Typically, $p^* = 1/2$.

- Extensions

- Similar to linear regression, we can extend the number of predictors and include interaction effects.
 - Logistic regression can be extended to more than two categories (not discussed here).

Credit Card Payment Default

- Consider predicting default given income and balance.
 - Orange: default = yes; blue: default = no
 - Income seems to have little effect, but balance do.

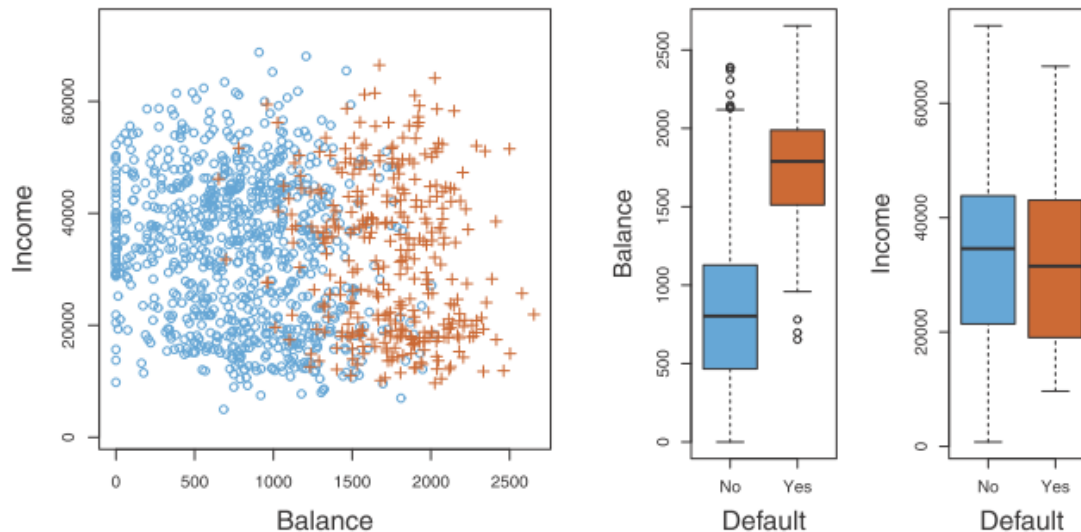


FIGURE 4.1. *The **Default** data set. Left: The annual incomes and monthly credit card balances of a number of individuals. The individuals who defaulted on their credit card payments are shown in orange, and those who did not are shown in blue. Center: Boxplots of **balance** as a function of **default** status. Right: Boxplots of **income** as a function of **default** status.*

- Linear vs. logistic regression
 - Linear regression doesn't produce probability values

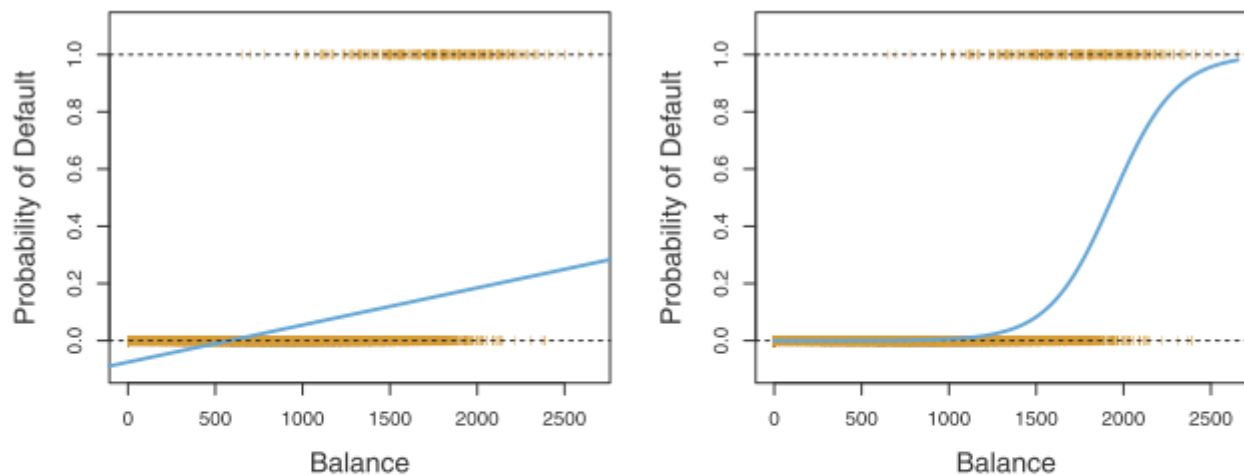


FIGURE 4.2. Classification using the **Default** data. Left: Estimated probability of **default** using linear regression. Some estimated probabilities are negative! The orange ticks indicate the 0/1 values coded for **default**(No or Yes). Right: Predicted probabilities of **default** using logistic regression. All probabilities lie between 0 and 1.

- Fitted logistic regression model

	Coefficient	Std. error	Z-statistic	P-value
Intercept	−10.6513	0.3612	−29.5	<0.0001
balance	0.0055	0.0002	24.9	<0.0001

TABLE 4.1. For the **Default** data, estimated coefficients of the logistic regression model that predicts the probability of **default** using **balance**. A one-unit increase in **balance** is associated with an increase in the log odds of **default** by 0.0055 units.

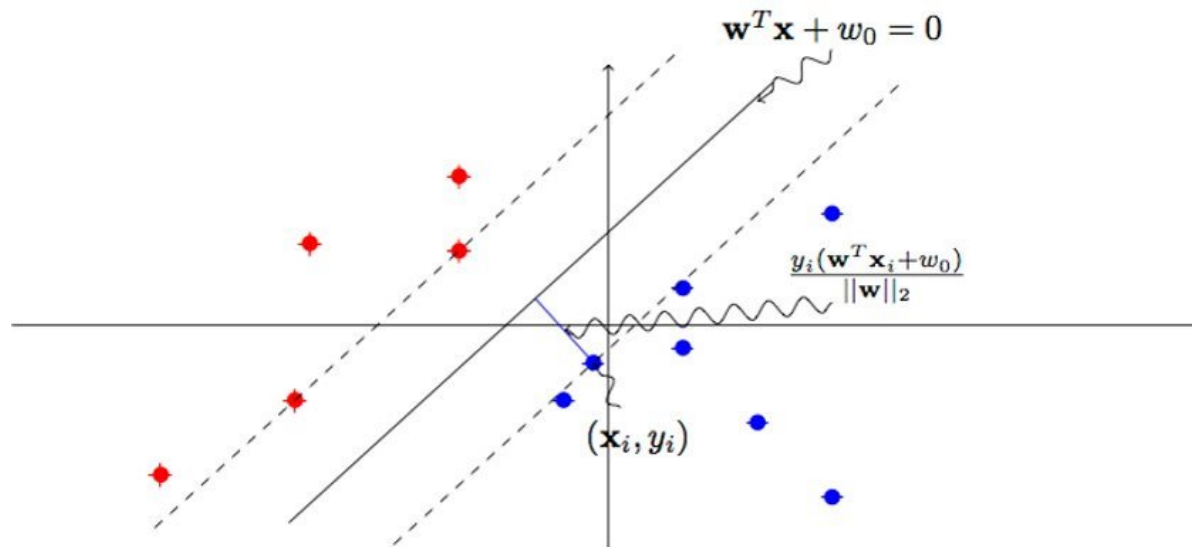
- A very negative intercept indicates very low default rate when balance is 0
 - In reality, you can't default with balance 0
 - So model agrees with reality reasonably well
- A positive intercept indicates that balance and default rate are positively correlated
 - Every unit increase in balance increases log odds by 0.0055

Support Vector Machines (SVMs)

- SVMs are advanced model capable of representing very complex functional relationships between the inputs and outputs
- They were the state-of-the-art models for numerous problems around the 2000s
- Today, they are still commonly used to solve many problems

The Idea in One Slide

- SVMs are based on the following geometric intuition
 - Consider a two-class dataset (red and blue)
 - When the dataset is linearly separable (that is, we can draw a hyperplane separating the two classes), we have many separating hyperplanes
 - In SVM, we prefer the hyperplane farthest away from the nearest example.



- We can write down the problem for finding the max-margin hyperplane as an optimization problem
- It turns out that there are efficient algorithms for us to solve it
 - Similarly to the case of logistic regression, these algorithms are quite sophisticated.

Nonlinearly Separable Data

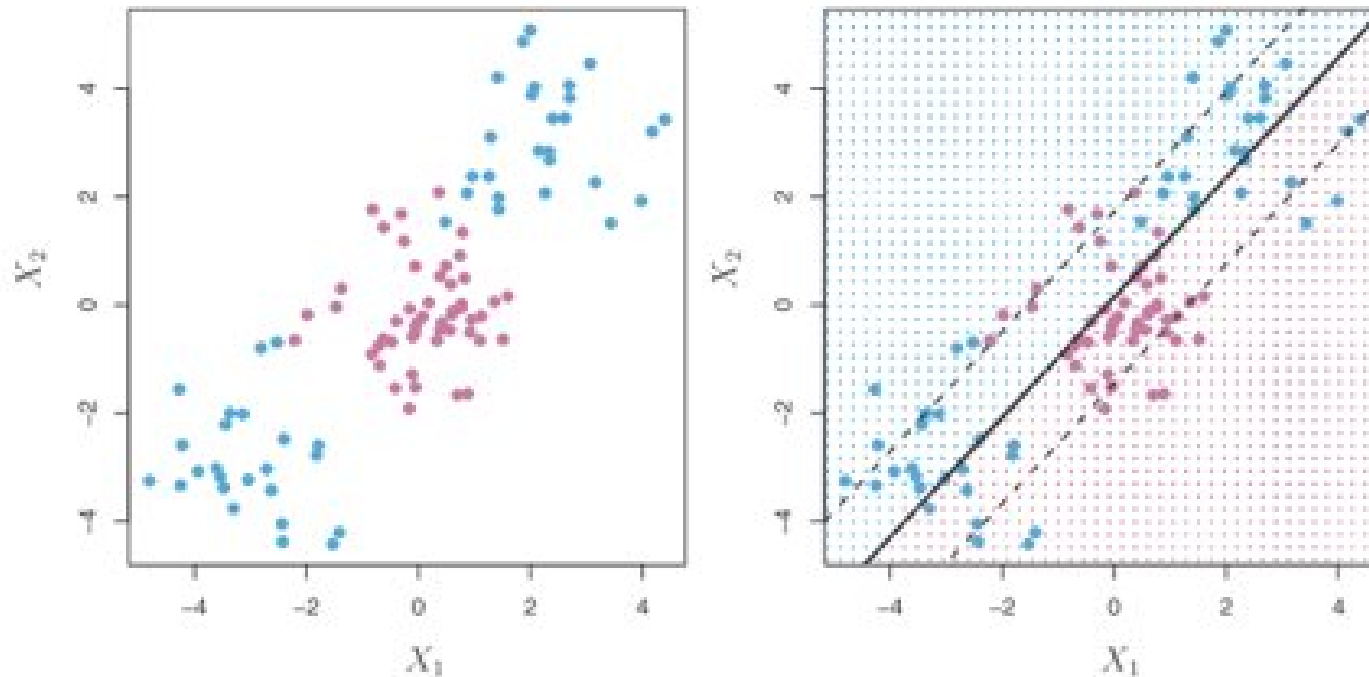
- We have only considered the case when the data is linearly separable, and the resulting SVM is often called hard-margin SVM.
- In practice, data may not be linearly separable, but we can extend hard-margin SVM to handle such data as well.
- The idea is to
 - Allow individual points to lie on the 'wrong' side
 - But penalize such anomalies
- The resulting model is called the soft-margin SVM.

Nonlinear Models

- In the SVM models that we described models a linear relationship.
- SVM can be extended to learn nonlinear relationship using the kernel trick.
- The idea is to transform x to new feature vector $\phi(x)$, and then learn a linear SVM.
- In practice, the new feature vector is implicitly defined via a kernel
 - The kernel $K(x, x')$ is the inner product of $\phi(x)$ and $\phi(x')$
- The solution has the following form $f(x) = b + \sum_i \alpha_i K(x, x_i)$.

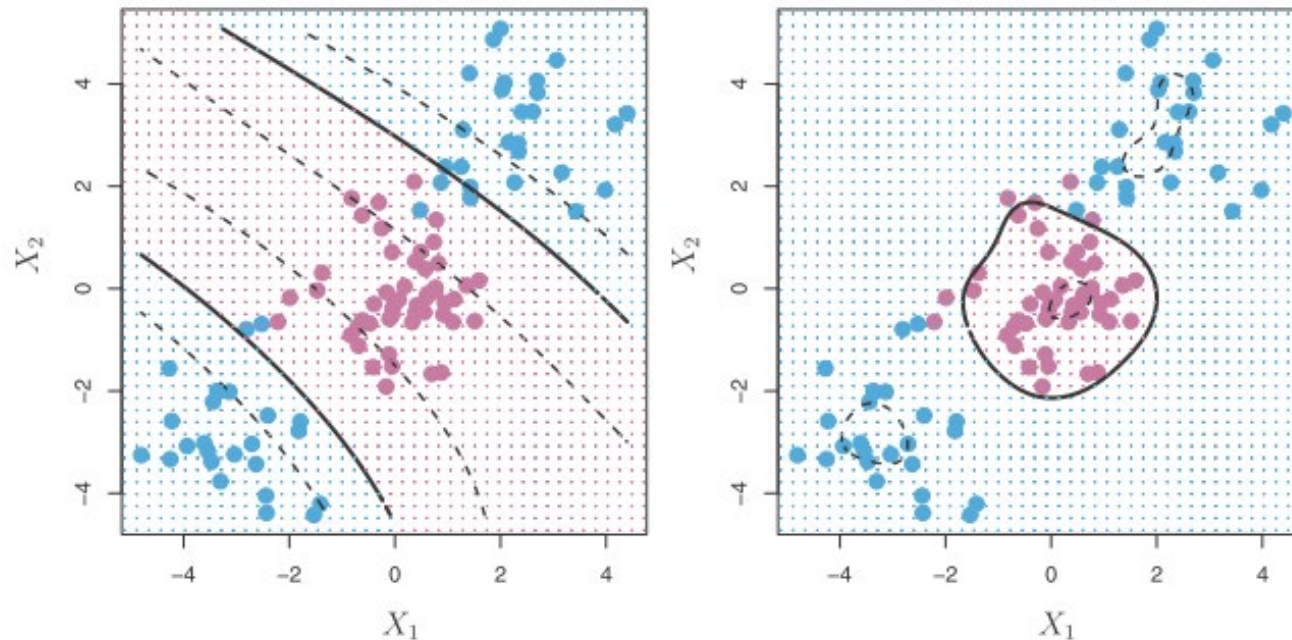
Illustrations

- Linear kernel
 - LHS is the data
 - RHS shows the decision boundary – a straight line (the solid line)



- Polynomial and radial kernels

- The decision boundaries using polynomial and radial kernels better separate the data than a linear kernel



Two Classes of Learning Methods

- Parametric
 - Assume functional form, e.g. linear
 - Model fitting / training through parameter fitting
- Nonparametric
 - No assumptions on functional form
 - Functional form is often data-dependent (e.g. k-NN)

- Flexibility/Interpretability tradeoff



Imbalanced Datasets

- For dataset where the classes are highly imbalanced, accuracy is a poor performance measure.
- For example, if 99% of the examples are negative, and 1% is positive, we can achieve 99% classification accuracy by simply always predict negative.
- However, in a lot of cases, we are interested in getting the rare positive class correct.
- We use more sophisticated performance measures for imbalanced datasets.

- Confusion matrix for binary classification

		True label	
		Positive	Negative
Predicted label	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Spam Email Classification Illustration

Measure	Typical business need	Follow-up question
Accuracy	"We need most of our decisions to be correct."	"Can we tolerate being wrong 5% of the time? And do users see mistakes like spam marked as non-spam or non-spam marked as spam as being equivalent?"
Precision	"Most of what we marked as spam had darn well better be spam."	"That would guarantee that most of what is in the spam folder is in fact spam, but it isn't the best way to measure what fraction of the user's legitimate email is lost. We could cheat on this goal by sending all our users a bunch of easy-to-identify spam that we correctly identify. Maybe we really want good specificity."
Recall	"We want to cut down on the amount of spam a user sees by a factor of 10 (eliminate 90% of the spam)."	"If 10% of the spam gets through, will the user see mostly non-spam mail or mostly spam? Will this result in a good user experience?"
Sensitivity	"We have to cut a lot of spam, otherwise the user won't see a benefit."	"If we cut spam down to 1% of what it is now, would that be a good user experience?"
Specificity	"We must be at least <i>three nines</i> on legitimate email; the user must see at least 99.9% of their non-spam email."	"Will the user tolerate missing 0.1% of their legitimate email, and should we keep a spam folder the user can look at?"

Measure	Formula
Accuracy	$(TP+TN) / (TP+FP+TN+FN)$
Precision	$TP / (TP+FP)$
Recall	$TP / (TP+FN)$
Sensitivity **	$TP / (TP+FN)$
Specificity ***	$TN / (TN+FP)$

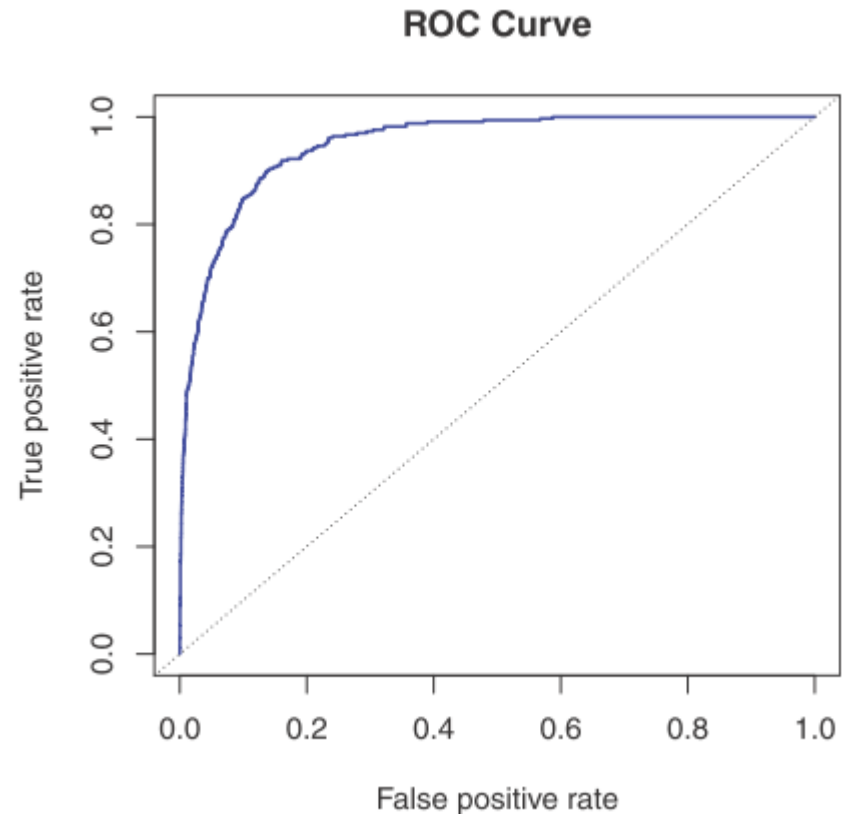
**Error or error rate* is $(FP + FN) / (TP + FP + TN + FN)$; it equals $1 - \text{Accuracy}$.

**Also called *true positive rate*, *power*, or $1 - \text{Type II error}$

***Also called *true negative rate*, $1 - \text{Type I error}$, $1 - \text{False Positive rate}$

ROC and AUC

- For a score function that indicates an instance's tendency to be positive, we can produce different classifiers using different thresholds.
- The *receiver operating characteristics* (ROC) plots the TP and FP rates as the threshold changes.
- A common quantitative measure of classifier performance is the *area under the (ROC) curve* (AUC)
- Higher AUC indicates that the score function is better at ranking an example's tendency to be positive.



POLL QUESTIONS – MAKING THE DATA CONFESS - CLASSIFICATION

What Will be Covered

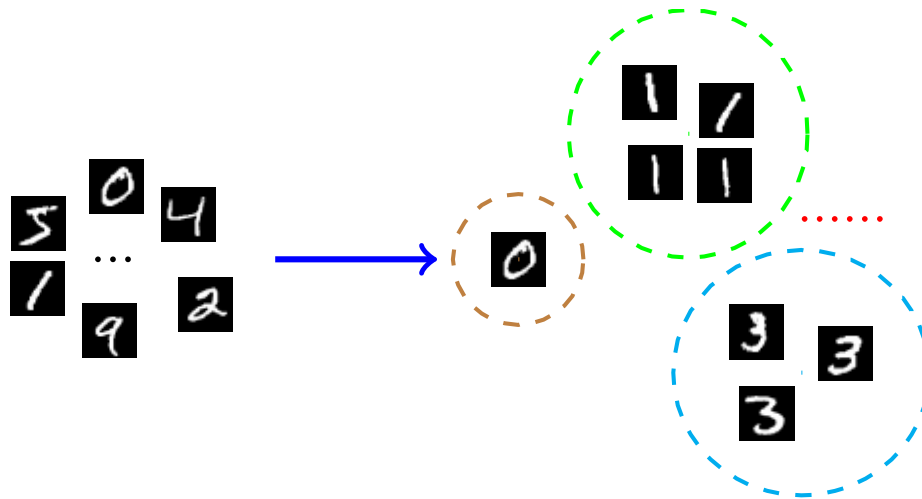
- Overview of machine learning
- Regression
- Classification
- Clustering
 - K-means
- Model selection

Unsupervised Learning

- No response Y
 - Regression/classification not possible.
- Frequently carried out hand in hand with EDA
- Clustering
 - Uncover distinct groups in data
 - K-means, Support Vector Machines, Hierarchical Clustering,...
- Principal Component Analysis (PCA)
 - Determine orthogonal directions in feature space in which data vary most
 - Not discussed further here;

K-Means Clustering

- Assign points to a given number K of clusters to minimise total “within cluster variation”, that is, we want each cluster to be as homogeneous as possible.



- Specifically, consider n feature vectors x_1, \dots, x_n , with x_{ij} being j -th feature of x_i
- K-means searches for a partition C_1, \dots, C_k of the indices $\{1, \dots, n\}$, so as to minimize

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

where $W(C_k)$ is

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

This is actually the sum of squared distances between points in the cluster to the cluster mean – thus the name of the algorithm.

- Naïve approach
 - Enumerate all possible clusterings, and find the one minimizing the objective function.
 - This does not work – there are too many possible clusterings.
- It turns out that there is a very simple iterative algorithm that finds a good clustering.

Algorithm 10.1 *K*-Means Clustering

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:
 - (a) For each of the K clusters, compute the cluster *centroid*. The k th cluster centroid is the vector of the p feature means for the observations in the k th cluster.
 - (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).
-

- Illustration on a small dataset (one iteration)

x	Initial cluster
0	1
1	1
2	2
7	2
8	1
9	2

- Illustration on a small dataset (one iteration)

x	Initial cluster
0	1
1	1
2	2
7	2
8	1
9	2




cluster	centroid
1	$(0 + 1 + 8)/3 = 3$
2	$(2 + 7 + 9)/3 = 6$

- Illustration on a small dataset (one iteration)

x	Initial cluster
0	1
1	1
2	2
7	2
8	1
9	2

New cluster
1
1
1
2
2
2



cluster	centroid
1	$(0 + 1 + 8)/3 = 3$
2	$(2 + 7 + 9)/3 = 6$

- Illustration on a large dataset

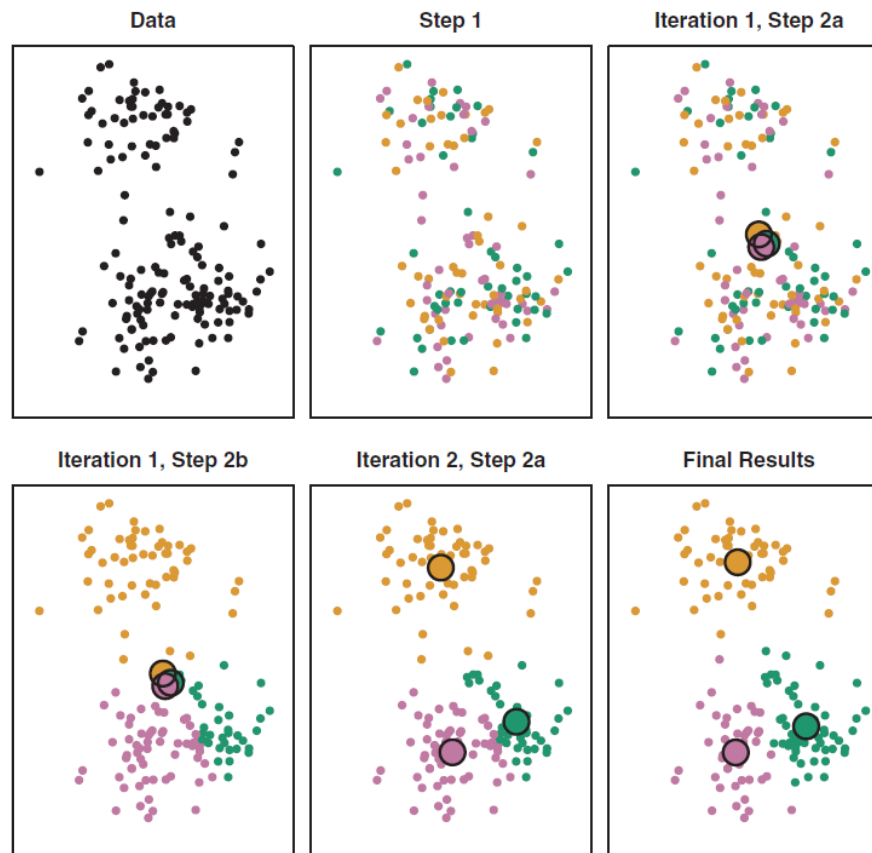


Figure reproduced from: G. James, D. Witten, T. Hastie, R. Tibshirani. An Introduction to Statistical Learning: with Applications in R. Corrected 6th printing, 2015. Springer, New York.

- The K-Means algorithm is guaranteed to find a *local* optimum.
- However, solution dependent on initial condition.
- Best practice is to repeat the process many times with different random initial conditions.

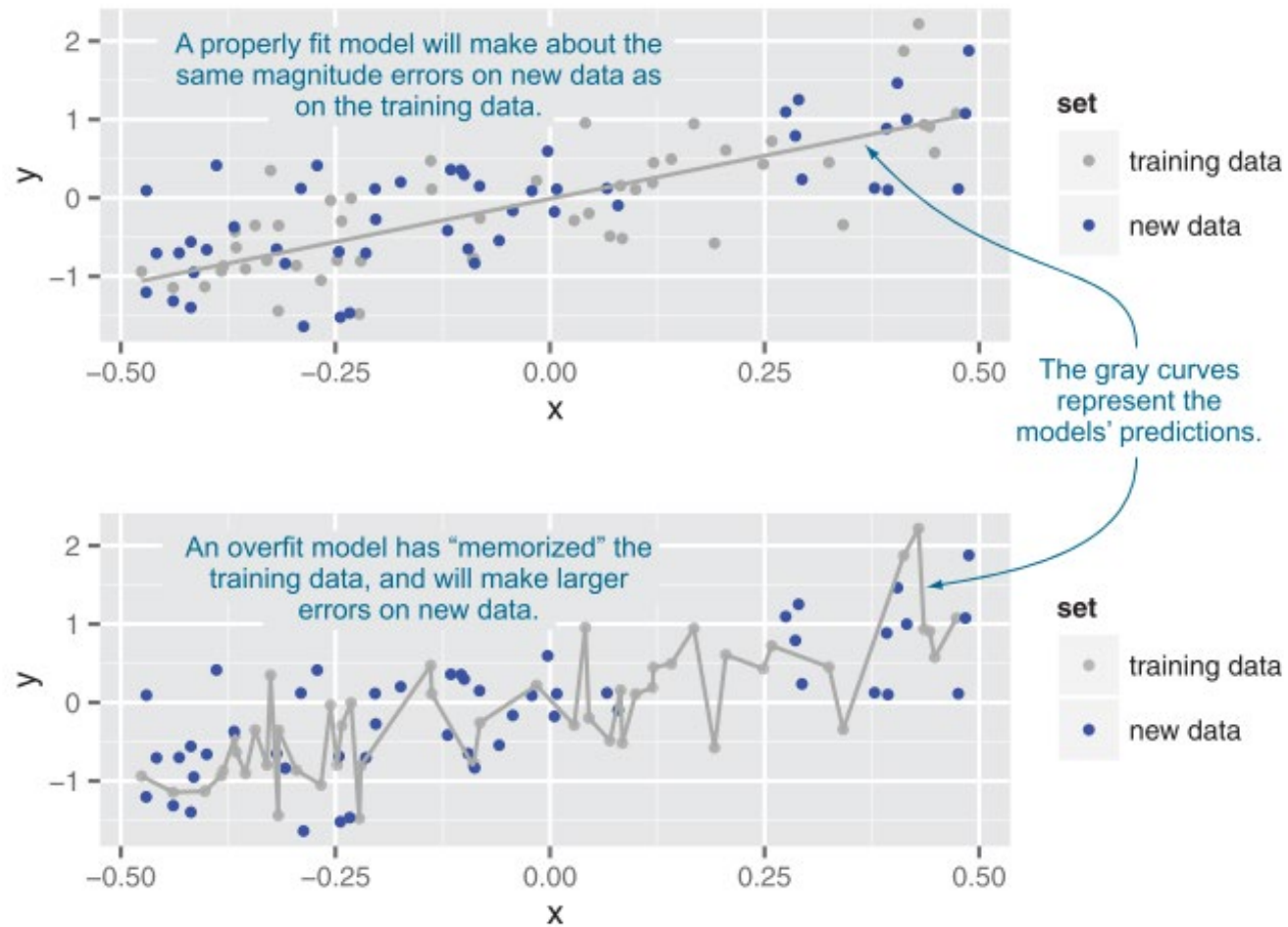
POLL QUESTIONS – MAKING THE DATA CONFESS - CLUSTERING

What Will be Covered

- Overview of machine learning
- Regression
- Classification
- Clustering
- Model selection
 - Overfitting
 - Validation set
 - Cross-validation

Model Selection

- We have already seen a few models for regression and classification.
- Given a set of data, we are interested in choosing a "best" model.
- Naïve approach
 - Choose the model that best fits the training data.
 - This does not work!
 - Training set performance is often a poor estimator of test set performance (i.e. how the model performs on new examples)



- Other naïve approaches
 - Choose the simplest model
 - Choose the most expressive model
- Both do not work well.
 - The simplest model may not contain the true model
 - The most expressive model may require too much data to learn
 - You need to pick a model of the right complexity
- [Optional] The bias-variance decomposition provides a mathematical description for such tradeoff
 - For a fixed X , assume the model predicts \hat{Y} given a random training set
 - The expected squared error is

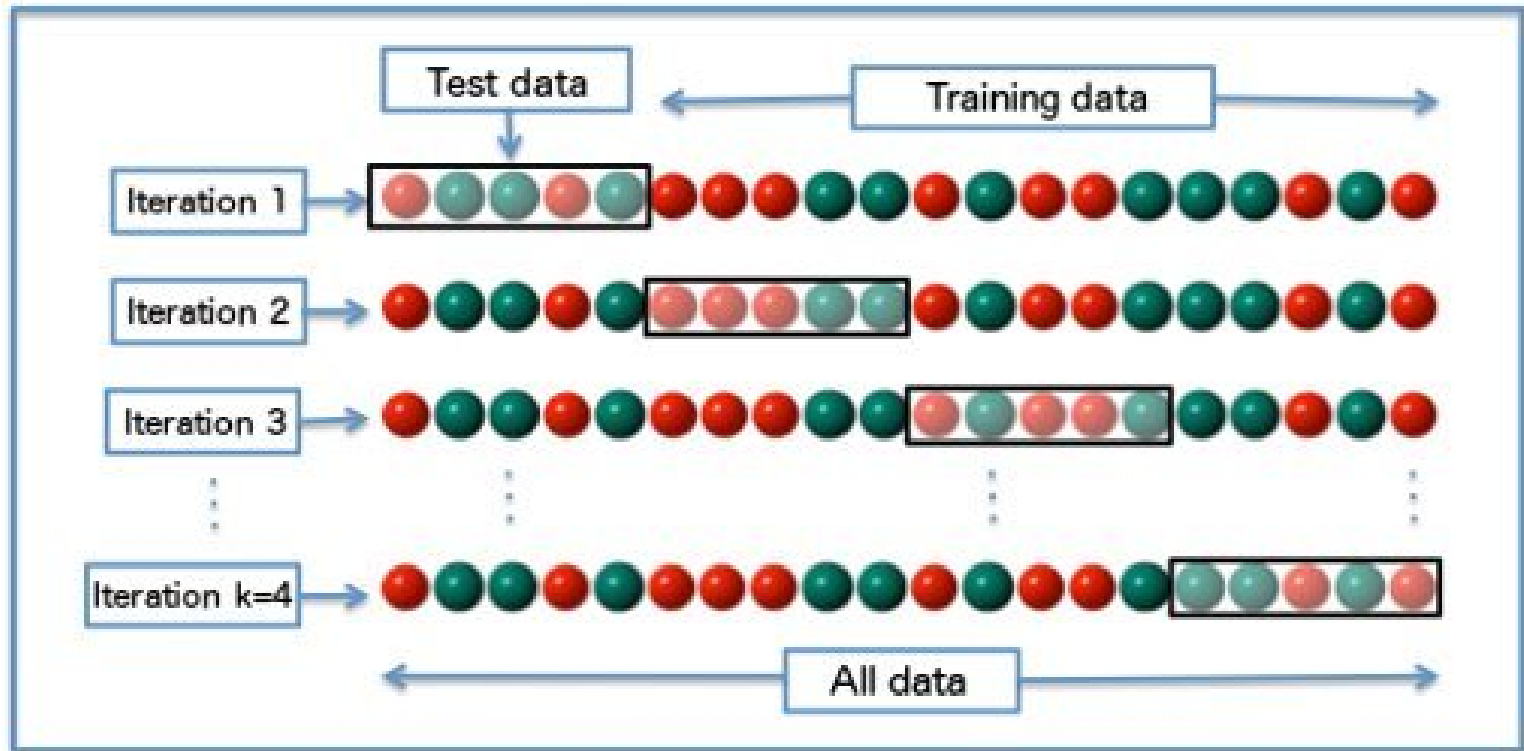
$$E(Y - \hat{Y})^2 = \underbrace{\text{Var}(\hat{Y})}_{\text{variance}} + \underbrace{(E(Y) - E(\hat{Y}))^2}_{\text{bias}} + \underbrace{\text{Var}(\varepsilon)}_{\text{irreducible}}$$
 - A simple model has high bias and low variance.
 - A complex model has low bias and high variance.

Train-Validation Split

- If you have a lot of training data, you can split it into a training set and a validation.
- Train candidate models on the training set.
- Choose the one with best predictive performance on the validation set.
- The validation set performance provides a much better estimate for a model's predictive performance on the test set.

Cross Validation (CV)

- When we don't have a lot of data, CV provides an alternative way to estimate a model's predictive performance using the training data.
- CV estimate a model's predictive performance as follows
 - Create multiple training and test splits for the training set
 - For each split
 - Fit a model on its training set
 - Compute the model's prediction error on the test set
 - Compute the average error using the above estimates.



4-fold cross validation

- k-fold CV

- This is CV with k train/test splits created as follows
- Randomly partition the data into k subsets of roughly the same sizes
- Each train/test split consists one subset as the test set, and the other k-1 subsets as the training set

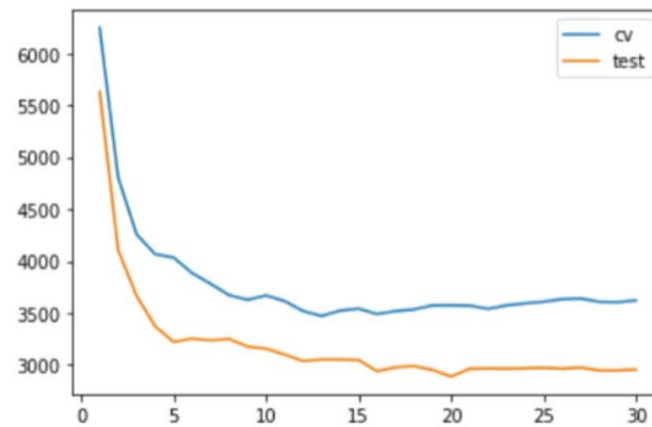
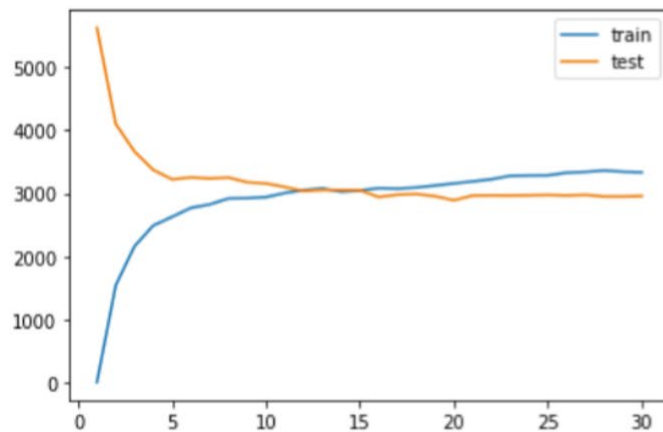
- Leave one out CV (LOOCV)

- This is n-fold CV ((i.e., each fold is a single example)

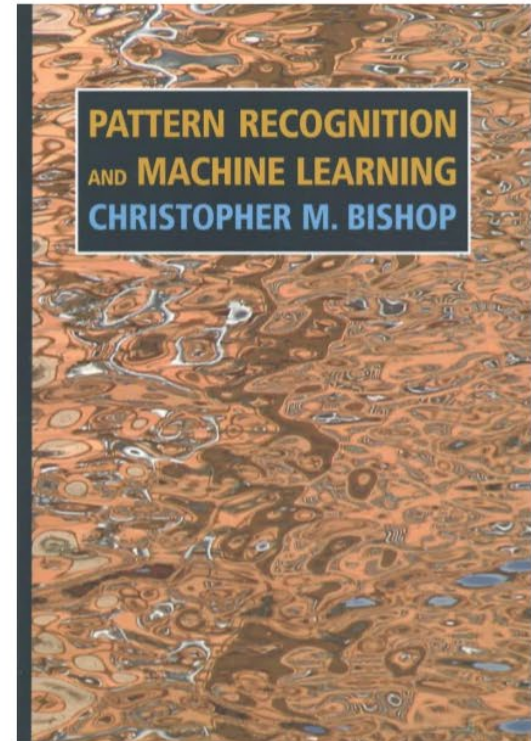
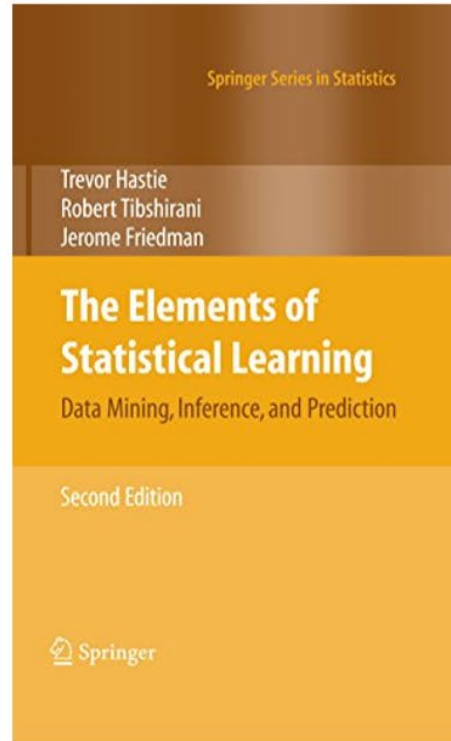
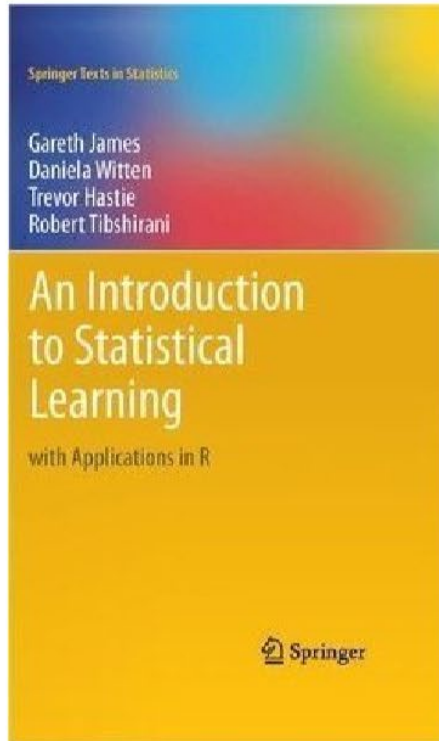
- Choosing between k-fold CV and LOOCV
 - For large (>hundreds) datasets, 10-fold CV is often recommended.
 - It is also generally not possible to use LOOCV because it's computationally expensive too.
 - For small datasets, LOOCV may be a better option.

- Illustration on kNN

- Typical plots for how training, test and CV errors vary as we increase k



Some Learning Resources



POLL QUESTIONS – MAKING THE DATA CONFESS – MODEL SELECTION