Predictive Modelling Introduction

Dr Liew How Hui

June 2022

Definition

Predictive modelling uses “statistics” to build mathematical models which can be used for predicting. (https://en.wikipedia.org/wiki/Predictive_modelling)

Terminologies of similar meaning: **Statistical learning**, **Machine learning**, https://en.wikipedia.org/wiki/Predictive_analytics

Topics (To cover: ✓; Not cover: ✗)

- Supervised Learning Models:
  - Classifiers: kNN ✓, logistic regression ✓, Naive Bayes ✓, LDA ✓, classification trees ✓, neural network ✗, ...
  - Regressors: kNN ✓, linear regression ✓ and variations, regression tree ✓, support vector regressor (SVR ✗), ...
- Unsupervised Learning Models:
  - Dimensional Reduction: PCA ✓, ...
  - Clustering: k-Means ✓, HC ✓, ...
  - Anomaly detection ✗
  - Association ✗
  - Autoencoders ✗
- Self-supervised learning ✗
- Reinforcement learning ✗: Value-based, Policy-based, Model-based
- Generative adversarial network (GAN)

May 2020 Final Assessment Q1(b)

Write an essay with no more than 3 pages to **summarise** the various **unsupervised learning models** and **supervised learning models** you learned by using **appropriate mathematical formulation**. Based on what you learned from your assignment and the Internet, suggest **improvements** on this course and propose a good online teaching learning environment. Be warned that non-constructive remarks and insults will receive ZERO mark. (7 marks)

Purpose: You should summarise what you have learned.

Software and Data

Popular data analysis programming languages:

- R + misc libraries: `https://cran.r-project.org/`
- Python (+ Pandas + Sklearn + RPy2): Anaconda Python (`https://www.anaconda.com/products/individual`)
- Java (Weka)
- SQL, NoSQL, etc.

Popular data:

- https://archive.ics.uci.edu/ml/datasets.php
- Kaggle (need registration)

Online Learning Tools

- Microsoft Teams
- WBLE (based on Moodle)
- Lecturer Github Site: `https://liaohaohui.github.io/UECM3993`
- YouTube: E.g. Dr Kilian Weinberger's Machine Learning Lecture, StatQuest, etc.

Reference Books

- Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani, An Introduction to Statistical Learning: with Applications in R, 2nd ed., Springer 2021

  `https://statlearning.com/` (Second edition is available for download)
- Trevor Hastie, Robert Tibshirani, Jerome Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer-Verlag, 2008

  `https://hastie.su.domains/ElemStatLearn/`

Classes & Assessment

The ‘course outcomes’ of this subject are

- CO1: Describe the key concept of statistical learning;
- CO2: Compare statistical models for prediction and estimation through supervised learning;
- CO3: Identify relationship and structures from unlabelled data through unsupervised learning;
- CO4: Demonstrate supervised and unsupervised learning with statistical software;
- CO5: Interpret results from supervised and unsupervised learning.

Classes & Assessment (cont)

Week 1: Practical 1 starts (basic R). There will be 12 practicals. Tutorial 1 starts (11 tutorials).

- Lecture online @ ts5oqqt
- Practical classes are conducted in lab (Wednesday)
- Tutorials 1 & 2 are physical (11am–1pm); Tutorial 3 is online @ MS Teams (code: ts5oqqt) (4pm–5pm, feel free to record it if I forget to do so) with online lecture after it.
- Those who wish to join physical (online) tutorial classes can change to T1/T2 (T3) during Week 1.

Classes & Assessment (cont)

Week 3: Everyone in class should be part of an assignment group.

Week 4: Assignment starts.

Week 6 or 7: Quiz.

Week 11: Submission of assignment report and computer program. Oral presentation may start if the lectures are completed, otherwise, the oral presentation will be in Week 12.

Week 12: Wednesday is national day, no practical.
Week 14: Assignment markings should be out by Friday; if not, it would be Week 15.

Classes & Assessment (cont)

Coursework (50%)

- Physical Practical Quiz (CO4): 12%. **Week 6** during practical class
- Assignment (38%): Report 18% (CO1 + CO2 + CO3) + Programming Code 10% (CO4) + Oral Presentation 10% (CO5, Online, need to include the explanation of the best model's algorithm clearly!!!). Starts from Week 4, ends at Wed Week 11. After that: Oral Presentation.
- Week 1: Start to find assignment group members, 4–7 in a group.

Classes & Assessment (cont)

Final Exam (100 marks $\times 0.5 \Rightarrow$ 50%): 3 + 1 Questions.

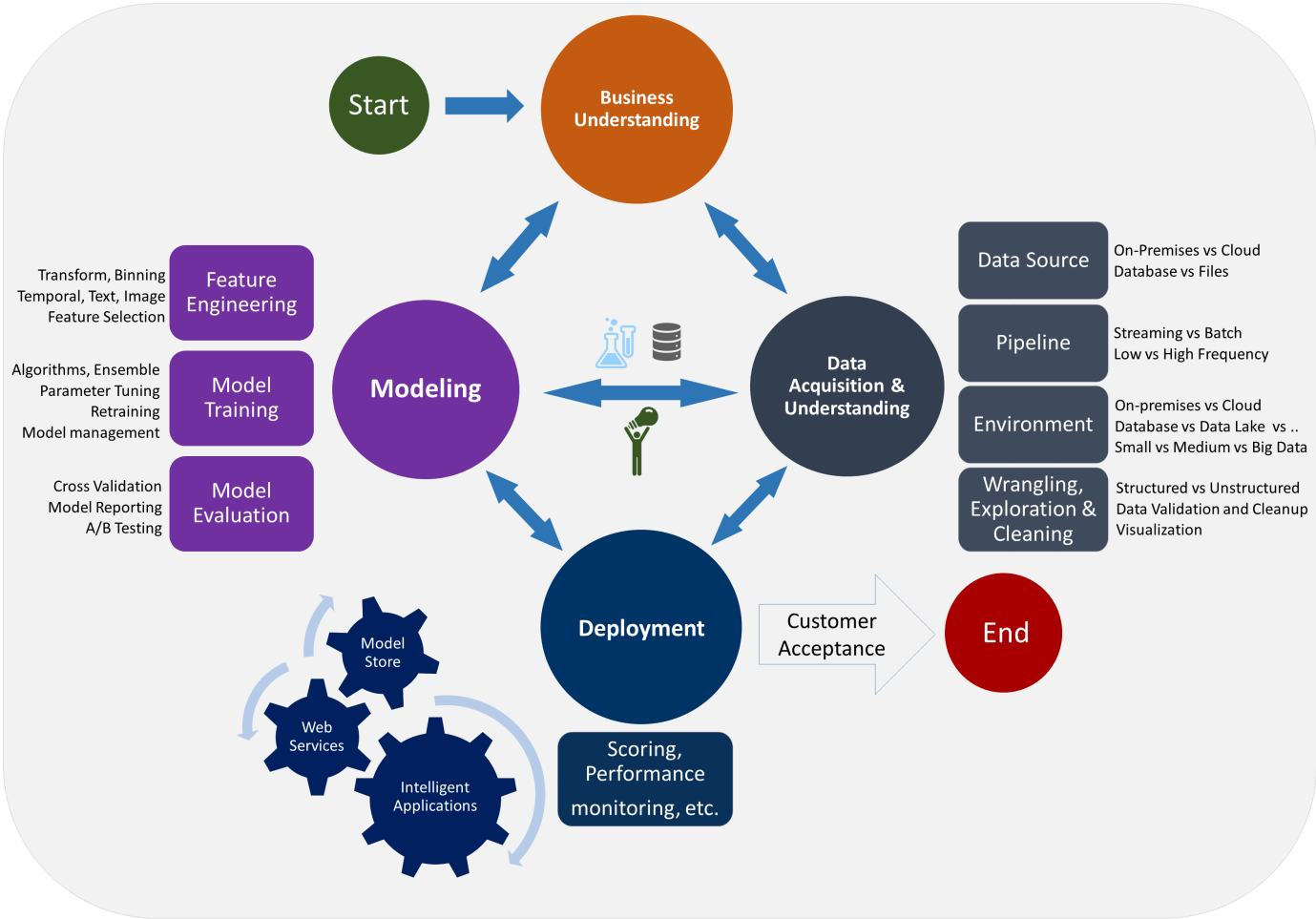
- Q1: CO1, Supervised + Unsupervised, 25 marks
- Q2: CO2, Supervised Learning Models, 25 marks
- Q3: CO3, UnSupervised Learning Models, 25 marks
- Q4/Q5: CO5, Supervised + Unsupervised, 25 marks

Data Science and CRISP-DM

Data Science / CRISP-DM (Cross Industry Standard Process for Data Mining)

- Business understanding
- Data understanding
- Data preparation / preprocessing
- Modelling
- Evaluation
- Deployment

Microsoft Data Science Lifecycle



https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle-business-understanding

Outline

Business Understanding

- Define the (business) goals that the data science techniques can target.
- Find the relevant data that helps you meet the goals / answer the questions
- Many of your seniors apply the predictive models on the data and choose the “best” model forgetting the “goal” (which is not the right direction)
- The right goal: “How do the factors influence the target?” OR “How does the best model help business?”

Business Understanding (cont)

Example: Suppose the **goal** is to understand the factors that affect the height of a person.

- Age
- Amount of carbohidrates
- Amount of protein
- Amount of fibre
- Quantity and quality of exercises
- Hours of sleep
- etc.

Business understanding: Which factors are the easiest to perform data collection and more likely to **influence** the height?

Outline

Data Understanding

Data sources (what lecturer learned from IT visit):

- SQL: Microsoft SQL (most popular), Oracle, PostgresQL, MySQL (used to be very popular in Web mangement), MariaDB, etc.
- SPARQL: https://en.wikipedia.org/wiki/SPARQL, http://spark.apache.org/sql/, https://pypi.org/project/sparkql/
- ...

Data Understanding (cont)

Structured Data (EDA can be used):

- Most companies usually stored data in structured data format in SQL database using SQL or Excel Table.

Unstructured Data (EDA cannot be used):

- Texts: Reports in Word/PDF; Twitters; etc.
- Images (of different sizes and resolutions)
- Biometric data
- Songs / Lyrics
- Time series: Stock price; Online game control sequence; Industrial robot control sequence; etc.

Data Understanding (cont)

To know the **structured data** — Exploratory Data Analysis (EDA):

- Univariate data summary:
  - Categorical (or nominal) data (e.g. Gender): mode. R's `factor`; Python's `astype("category")`
  - Ordinal data (e.g. Student grade): mode? R's `ordered`
  - Numerical data (e.g. Temperature): mean, median, quantiles, variance
- Measurement units? E.g. 1 metre or 100 cm?
- Data correlation analysis
- Appropriate Data Visualisation:
  - histogram
  - barplot / bar graph
  - box plot
  - heatmap, etc.

EDA and Data Understanding (cont)

EDA tools for the univariate data in R:

- R's `summary`, Python's `describe`
- For (continuous) numerical data: R's `hist` (histogram), `stem` (stem-and-leaf plot)
- For integral data and categorical data, R's `table`.

EDA tools for the bivariate data in R:

- categorical vs categorical: `barplot`, `table`
- categorical vs numerical: `boxplot`
- numerical vs numerical:
  - `cor(x, y, method="pearson")`,
  - (scatter) `plot`

Data Understanding (cont)

What about **unstructured data**? There is no single answer, we need to convert them to structured data in various ways!

- We need to convert images to 2D matrix or 3D array of a fixed shape
- We need to convert texts to 2D matrix with individual words as columns, rows as word counts. This is called the **bag of word** representation. Another more power representation is the TF-IDF (Term Frequency-Inverse Document Frequency) representation. See `https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/` for excellent example.
- Human speech signals? Musics? Videos? etc.

Data Understanding (cont)

Whether the real-world data comes from the 'databases' or from Excel files or Internet, there may be noise, i.e. missing values, wrong values, etc. To clean up the data:

- **Identify** missing/wrong values and **impute** if necessary
- Don't treat any column as outlier!
- Outliers are rows which are unique (e.g. the only row with a unique value. E.g. only one row with customer status VVVIP)

Predictive modelling deals with 'majority' data not a very very special / rare case.

Outline

Data Preparation / Preprocessing

Data preparation usually assumed the data is **structured** and it is **combined** with the modelling process. Different predictive models have different 'requirements' on the inputs and outputs of the data. E.g.

1. The output may be required to be binary or K-class or continuous
2. The input may be required to have no missing values, etc.
3. The input may be required to be numeric only
4. The input may be required to have small difference in variance and uncorrelated in the numeric inputs, etc.

Data Preparation (cont)

For item 1, we usually just pick the right predictive models (classifiers or regressors) for our job.

For item 2,

- if there are very few missing values (e.g. less than 1%) we may throw away the rows with few missing values;
- if there are too many missing values (e.g. $> 50\%$), we need to check where do they come from.
  - If one or two columns is the cause, we remove them
  - If the missing values are spreaded everywhere, we use missing value heatmap to try to identify some patterns and decide on how to impute the data if necessary. E.g.
    - for naive Bayes, we need to impute the data;
    - for CART decision tree, we may not need to impute the data

Data Preparation (cont)

For item 3, we may convert ordinal inputs to integers, nominal inputs to one-hot-encoding (this may not be a good idea if there are too many classes in the nominal input).

The base R provides the model.matrix function which allows the individual columns to be one-hot encoded. However, it is not easy to use.

```
model.matrix(~0+col1+col2, data=mydata)
```

The caret library provides the following one-hot encoding method which is easier to use.

```
library(caret)
oneh = dummyVars( ~ ., data=d.f)
final_df = data.frame(predict(oneh, newdata=d.f))
```

Data Preparation (cont)

For item 4, we may perform:

- Standardisation of datasets — column scaling: `scale(d.f)`
- Min-max scaling

For data with special properties (e.g. nonlinear), we may need to perform specific data transformations before modelling.

- Normalisation — row scaling
- Non-linear and custom transformation
- Discretization: `arules::discretize`
- Generating polynomial features: `poly()`

Outline

Modelling

Real-world data are usually stored in various places:

- Files and folders (usually computer users);
- Cloud storage (some smartphone users);
- Databases (nearly all companies and government agencies).

Some real-world data are time-related, e.g. financial time series, econometric data, etc. They are studied by using time series analysis.
Some real-world data are time-independent. For example, photos, marketing data (usually collects customer age, gender, income range, etc. purchase items etc.), genetic data, etc.

Modelling (cont)

If the time-independent data can be transformed to a tabular form (structured data), then we can apply either unsupervised or supervised learning models.

- If the data is of the shape $n \times p$ with **no label**, then unsupervised learning could be applied.
- If the data is of the shape $n \times (p + 1)$, i.e. the input X is $n \times p$ and the output (also known as **label**) Y is $n \times 1$, then unsupervised learning could be applied to 'identify' patterns X and supervised learning models may be applied on (X, Y).

Modelling: Unsupervised Learning

If the data is mostly numeric (or can be converted to numeric values) and **has no labels** such as

- new genetic information (e.g. new variation of COVID-19 viruses or other coronavirus);
- new customer/marketing data in which patterns need to be uncovered; etc.

Unsupervised Learning methods below could be tried:

- Descriptive Statistics / EDA
- Visualisation → Dashboard
- Dimensionality Reduction. E.g. PCA (in syllabus)
- Clustering. E.g. k-means, HC (in syllabus)
- `https://en.wikipedia.org/wiki/Association_rule_learning`
- `https://en.wikipedia.org/wiki/Anomaly_detection`

Modelling: Supervised Learning

If the data **has label** Y which is

- numerical / quantitative (corresponding supervised learning problem is called **regression problem**) : E.g. sales figure; or
- categorical / qualitative (corresponding supervised learning problem is called **classification problem**): E.g. success / fail

The "data type" of the output allows us to classify the model $Y = h(X)$ into

- **Regressor**: Use to solve regression problems;
- **Classifier**: Use to solve classification problems.

Modelling (cont)

Modelling means to use a mathematical model $Y = h(X)$ to fit the observed data:

$$(\boldsymbol{x}_1, y_1),\ (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_n, y_n).$$

such that the total errors $\sum_i diff(y_i, h(\boldsymbol{x}_i))$ is acceptably small. Note that $(\boldsymbol{x}_i, y_i)$ can come be marketing or scientific data obtained through surveys or observations and they are usually stored in a tabular form which have been cleaned.

Note that the $\boldsymbol{x}_i$ are usually called inputs / attributes / features / columns / independent variables / etc. and may have more than one components:

$$\boldsymbol{x}_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,p}).$$

The y_i are usually called the outputs / labels / targets / response / dependent variable / etc. and is usually a single component.

Modelling (cont)

The mathematical model $Y = h(X)$ which is used to fit the observed data is called a **predictive model** and can be expanded into the form:

$$\underbrace{Y}_{\text{output}} = \underbrace{h(X_1, \cdots, X_p)}_{input}$$

The model h may be used for

- Prediction: If we just want to know "for a given input $(x_1, \cdots, x_p)$, what is the value of y?"
- Inference: Is the model correct? How the output Y is changing w.r.t. the input X_i? E.g. What factors "improves" sales?

Modelling (cont)

Example (Simple Linear Regression): If the linear regression (a kind of supervised learning method) is used to model the relation between y and x as follows.

y	23.82	47.16	66.66	88.39	110.54
x	1	2	3	4	5
y	131.1	174.15	214.72	233.9	252.14
x	6	8	10	11	12

Predict the value at $x = 7$ using the linear regression model.

[Ans: $\hat{y} = 150.93$]

Modelling (cont)

An approach to classify the predictive models (classifiers in particular) based on the 'Bayesian statistics' point of view:

- **Discriminative models**:

$$h(X) = \text{argmax}_j \mathbb{P}(Y = j | X_1, \cdots, X_p)$$

  E.g. linear regression, kNN, logistic model, etc.

- **Generative models**

$$h(X) = \text{argmax}_j \mathbb{P}(X_1, \cdots, X_p | Y = j) \mathbb{P}(Y = j)$$

  E.g. Naive Bayes, LDA, etc.

In the discriminative modelling, one aims to learn a predictor given observations; In the generative modelling, one aims to learn a joint distribution over all variables.

Modelling (cont)

We usually use a 'family' of models $h(X)$ with 'internal' parameters. The characteristic of the "number of parameters" allows us to classify models into

- **Parametric models**:
  - Models with **fixed** set of parameters
  - "Training" tries to find the most suitable parameter values to minimise "errors"
  - E.g. logistic regression
- **Nonparametric models**:
  - Models without fixed set of parameters. Internal "representation" **grows as data increases**.
  - "Training" tries to "fit" the data into the model!
  - E.g. kNN

Modelling (cont)

Example: Suppose the output y is governed by the input x following the following equation:

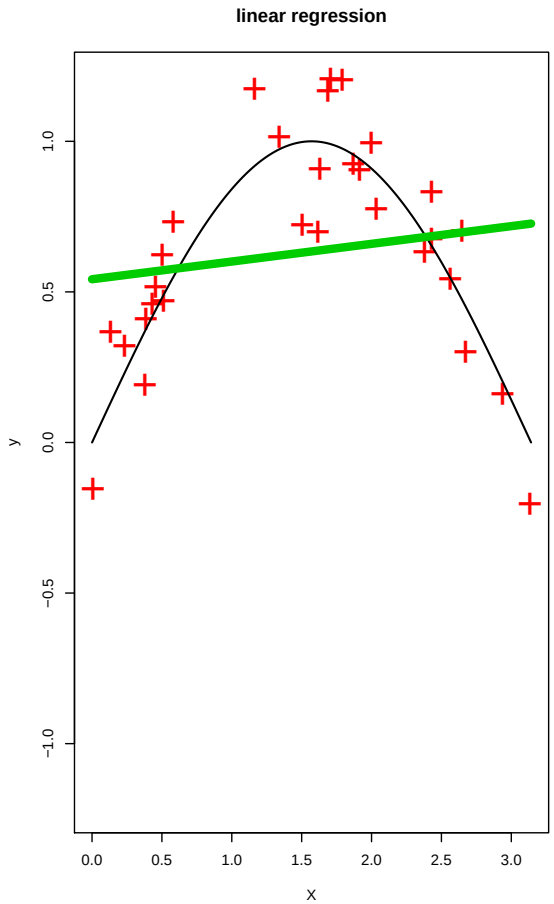
$$y = \sin(x) + R, \quad R \sim Normal(0, 0.2^2)$$

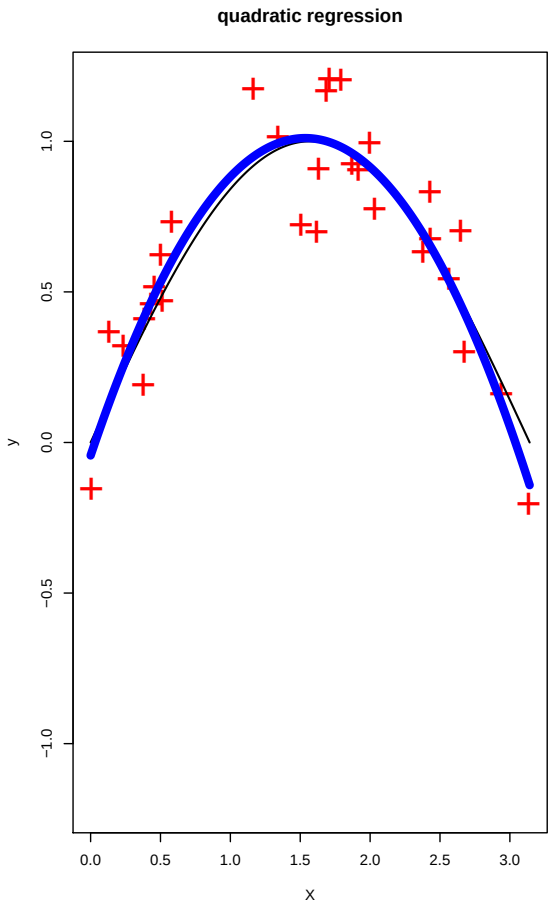
for the range $x \in [0, \pi]$.

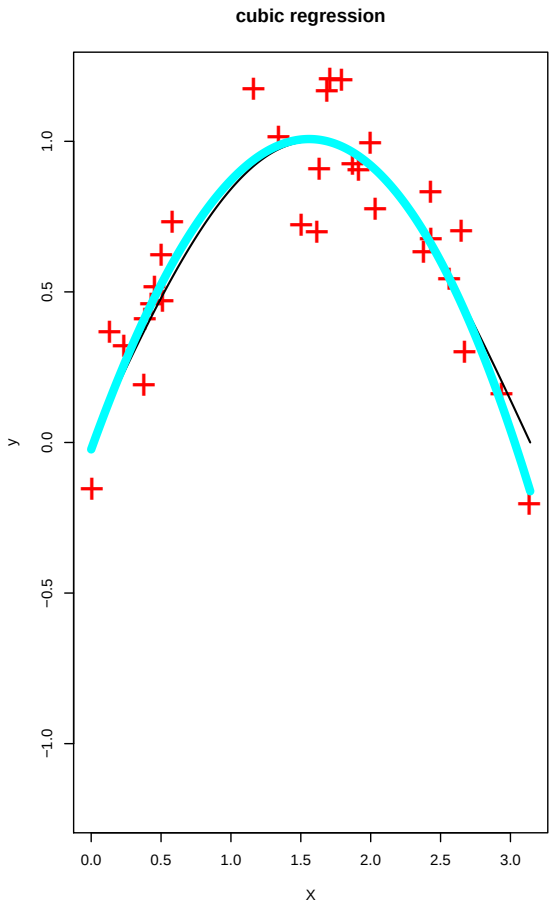
We try the following regression models:

- Linear regression: $y = ax + b + \epsilon$
- Quadratic regression: $y = a_2x^2 + a_1x + b + \epsilon$
- Cubic regression: $y = a_3x^3 + a_2x^2 + a_1x + b + \epsilon$
- kNN (Topic 2)
- Neural Network with 1 hidden layer 10 nodes (= $1 \times 10 + 10 + 10 \times 1 + 1 = 31$) parameters
- Regression tree

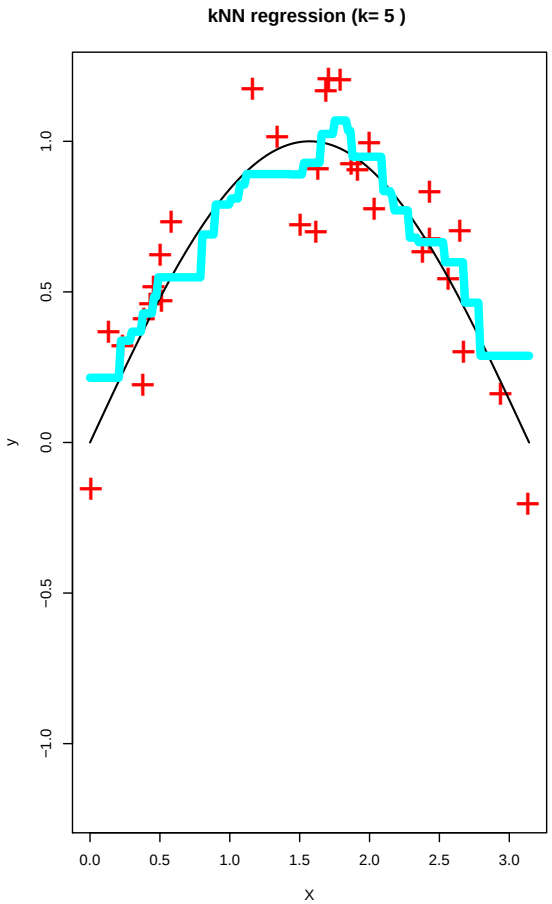
Modelling (cont)

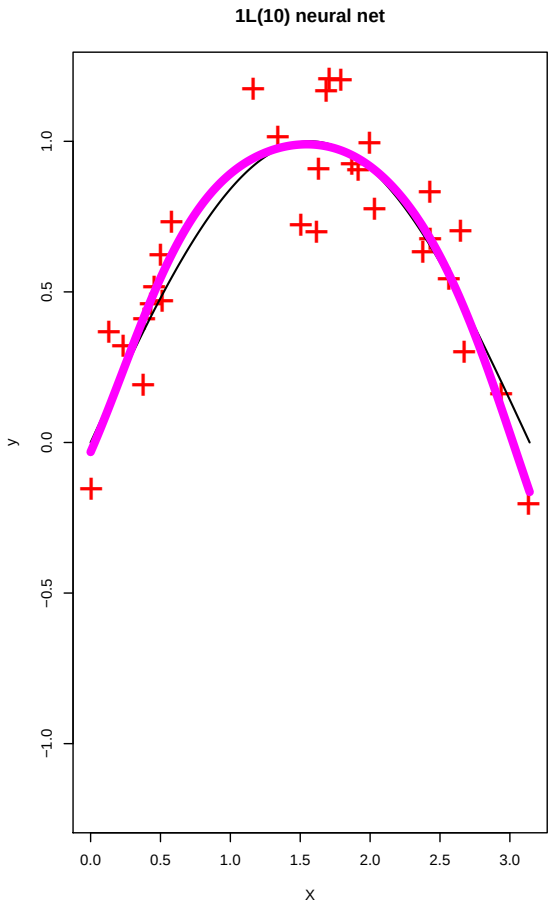


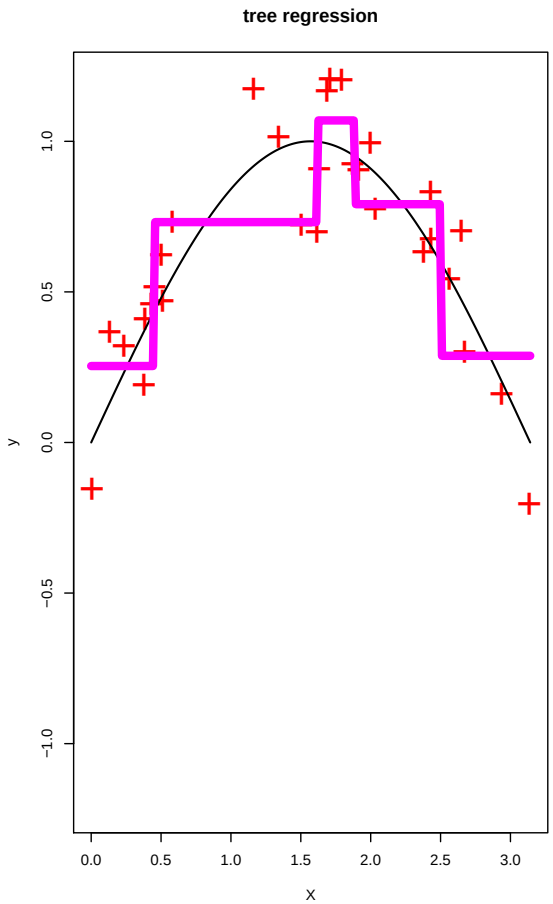




Modelling (cont)







Modelling (cont)

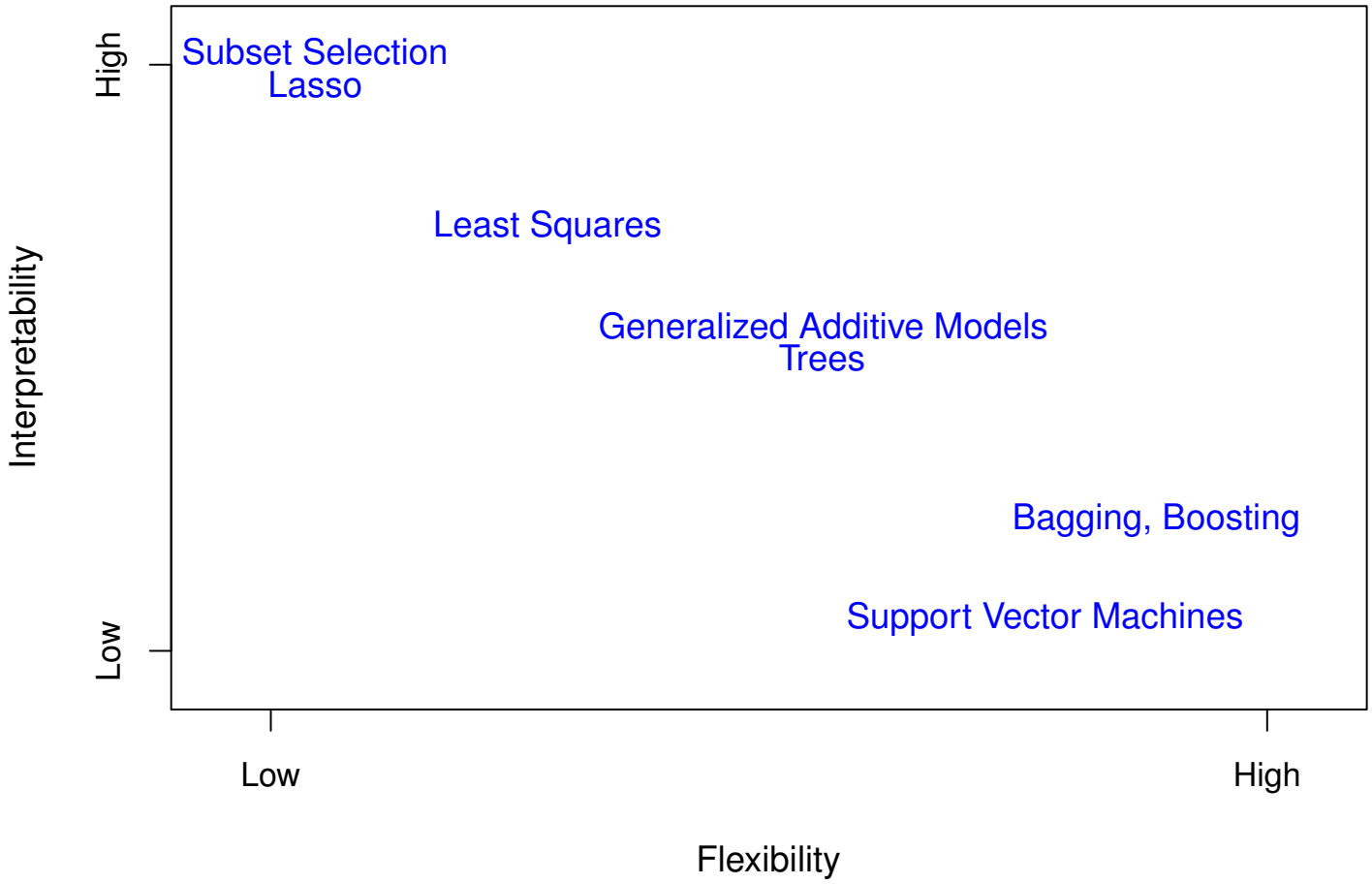
Which model is the best?

Things to consider: Flexibility (more parameters, model more complex) vs Interpretability (less parameters, model simpler)

- Inflexible $\Rightarrow$ Simpler math formula $\Rightarrow$ Poorer Predictibility, better inference(?)
- Flexible $\Rightarrow$ Complicated math formula $\Rightarrow$ Good Predictibility, poorer inference(?)

Modelling (cont)

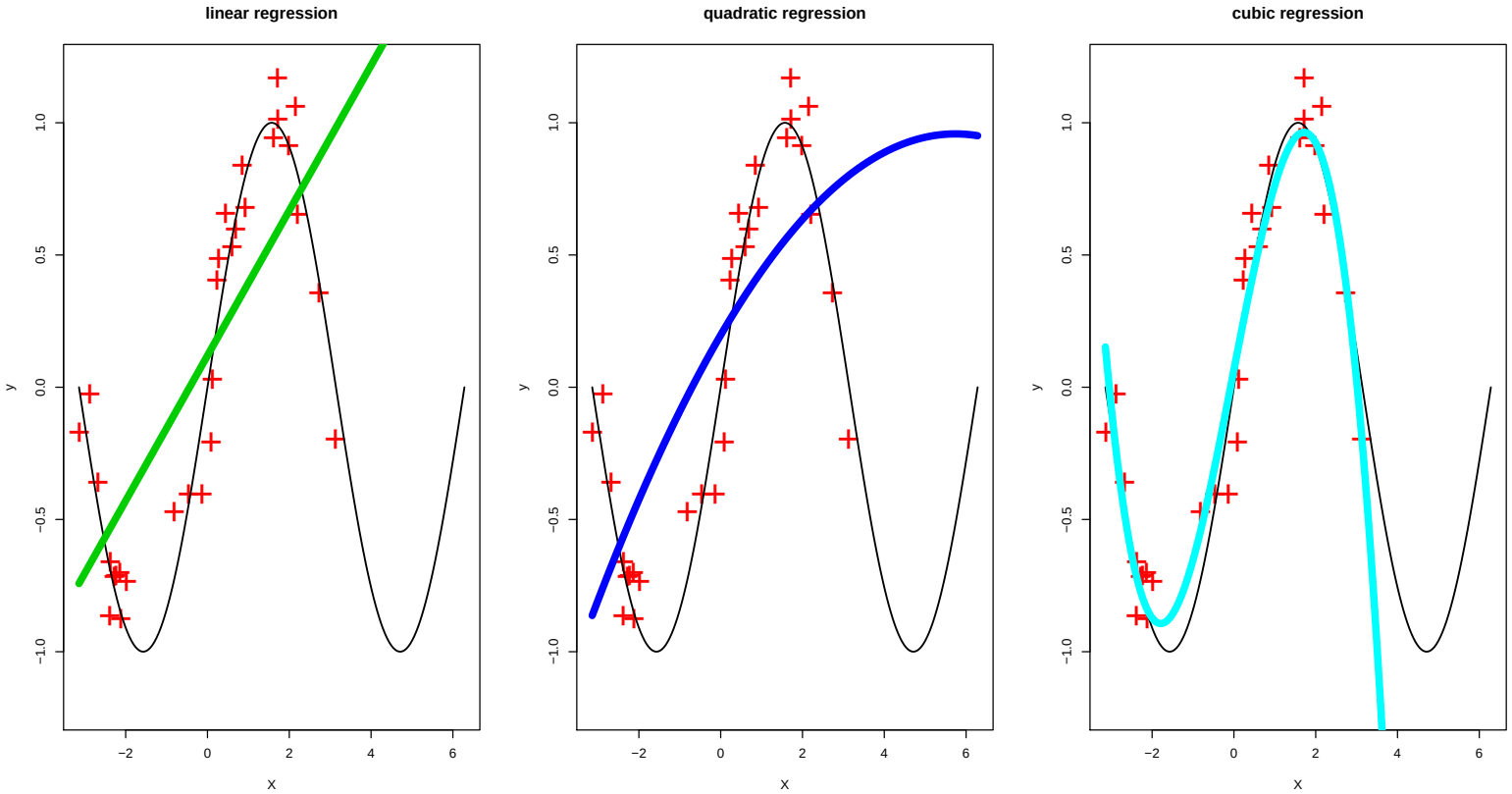
Usually, the interpretability and the flexibility are 'inverse' of each other like what the textbook shows:



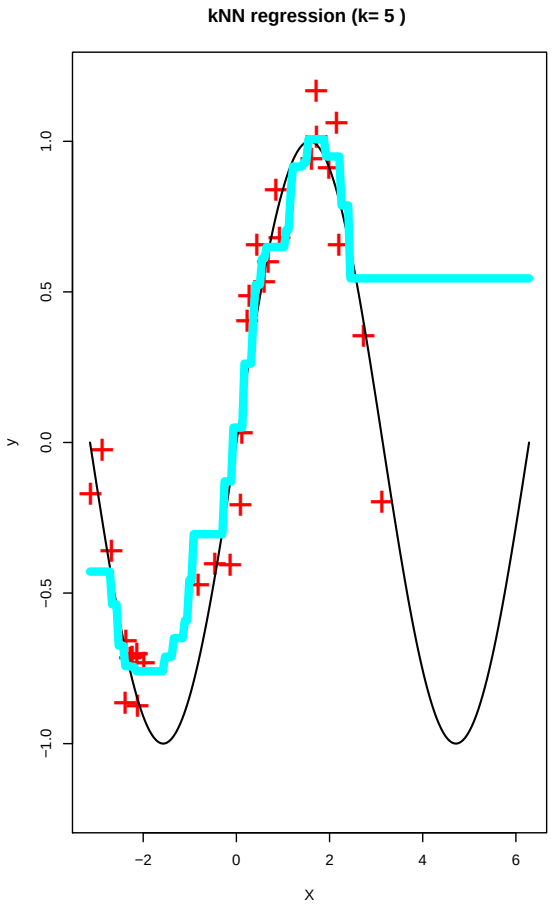
Modelling (cont)

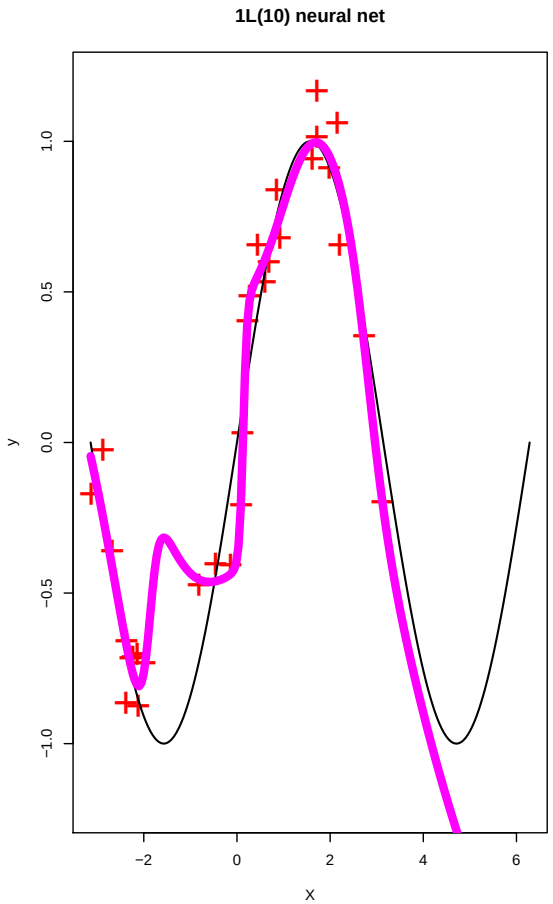
In the previous example, we have looked at the model for the range $x \in [0, \pi]$, now, let us look at $x \in [-\pi, 2\pi]$ with the same formula:

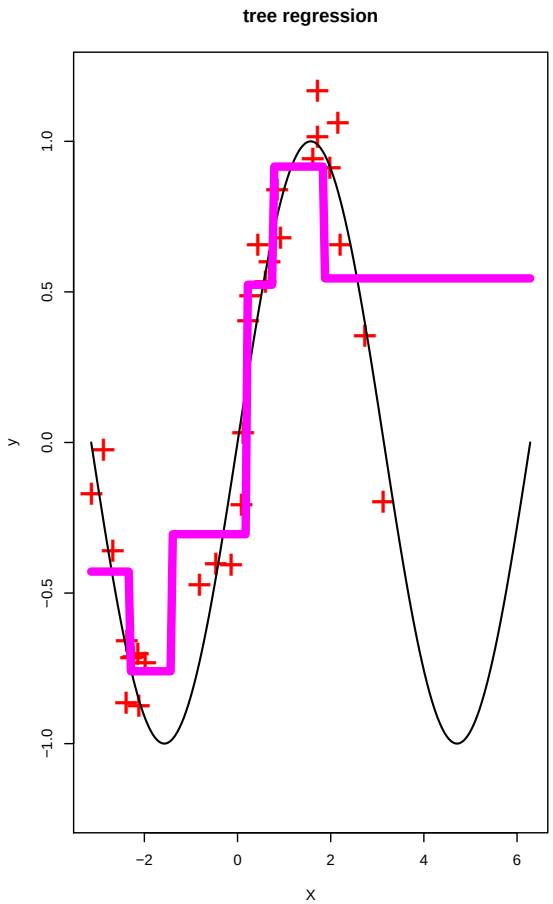
$$y = \sin(x) + R, \quad R \sim Normal(0, 0.2^2).$$



Modelling (cont)



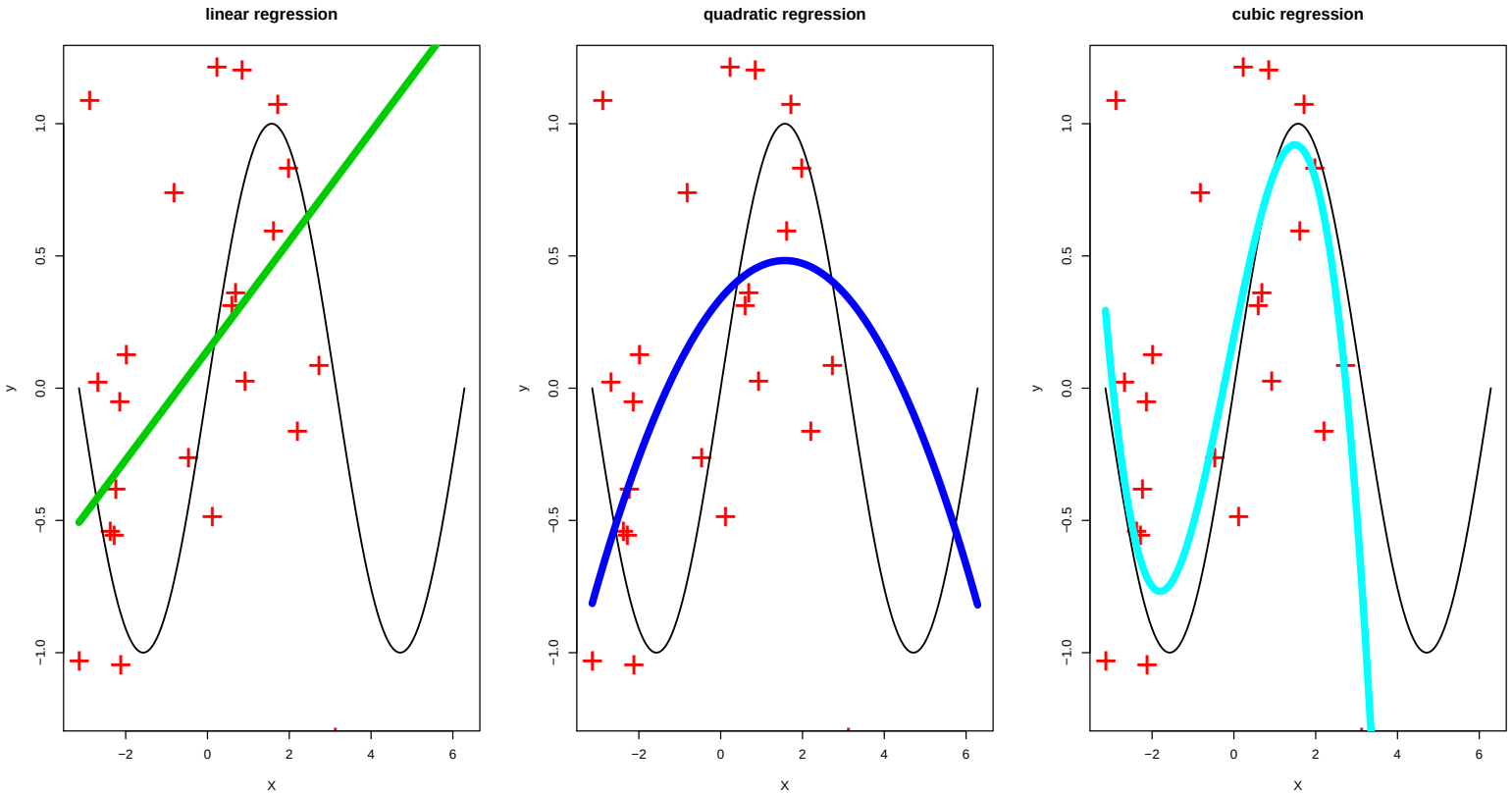




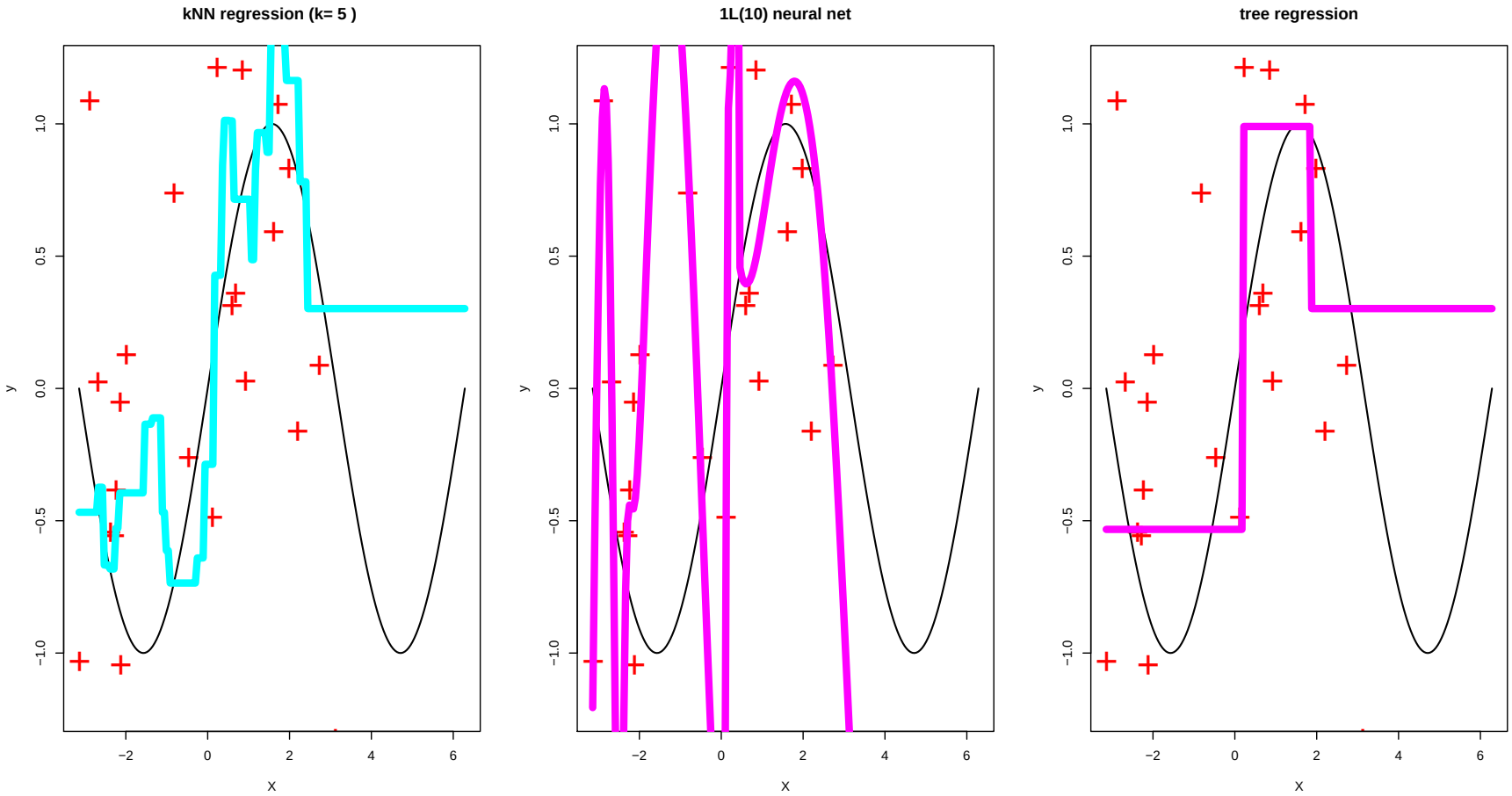
Modelling (cont)

Example: For a model with large 'noise' (i.e. σ increases from 0.2 to 1.2):

$$y = \sin(x) + R, \quad -\pi \leq x \leq 2\pi, \quad R \sim Normal(0, 1.2^2).$$



Modelling (cont)



Neural network is not performing well when the noise is large!

Modelling (cont)

Model training and deployment in R

```
#install.packages("SomePackageName")
#library(SomePackageName)
model = lm(y ~ ., data=Xy)
# Some models can be used for statistical inference
print(model)  # or print(summary(model))
# Deployment: prediction
predicted = predict(model, newdata=data.frame(x1=...,x2=...))
```

Model `.fit` and `.predict` in Anaconda Python

```
from sklearn.linear_model import LinearRegression
lrobj = LinearRegression()
model = lrobj.fit(Xy.iloc[:,3:4],Xy.iloc[:,4])
newdata = pd.DataFrame({'x1':..., 'x2':...})
predicted = model.predict(newdata)
```

Outline

Model Validation / Evaluation

For Unsupervised Learning:

- There is **NO** measures for evaluation;
- Only for **pattern identification**: regular patterns (e.g. special shapes), cluster patterns, random pattern (particular probability distribution).

For Supervised Learning:

- We have output Y associated with an input X, so we can put in the input of an observation $\boldsymbol{x}_i$ to the predictive model $h(\boldsymbol{x}_i)$ and **compare** it to the actual observed y_i;
- The 'theoretical' measurement of the **difference** between a true model Y and a predictive model $h(X)$ is called a **generalisation error**.

Model Validation (cont)

For regression problems, popular accuracy measures associated with the generalisation error are

- Mean Squared Error (MSE): $\frac{1}{n}\sum_{i=1}^{n}(y_i - h(\boldsymbol{x}_i))^2$
- R^2 error: $R^2 = 1 - \dfrac{\sum_{i=1}^{n}(y_i - h(\boldsymbol{x}_i))^2}{\sum_{i=1}^{n}(y_i - \frac{1}{n}\sum_{i=1}^{n} y_i)^2}$

For k-class classification problems, popular accuracy measures associated with the generalisation error are

- Accuracy: $\frac{1}{n}\sum_{i=1}^{n} I(y_i = \hat{y}_i)$
- Error rate: $\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$
- When $k = 2$, Cohen's Kappa, ROC/AUC, etc.

Accuracy of Regressors

We will be using the estimation of the **regression problem**'s theoretical MSE

$$\mathbb{E}[(f(X) + \epsilon - h_D(X))^2]$$

as an example for accessing model accuracy.

Note that MSE is a statistic for measuring the difference between the **true/theoretical model**

$$Y = f(X) + \epsilon$$

and the **predictive model**

$$\hat{Y} = h_D(X).$$

Accuracy of Regressors (cont)

The problem we are facing in assessing model accuracy is this: we **don't know the true model**

$$Y = f(X) + \epsilon$$

we only have the data (from the true model):

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i, \quad i = 1, 2, ..., n.$$

If we use all data $(\boldsymbol{x}_i, y_i)$, $i = 1, 2, ..., n$ for training a predictive model $h_D(X)$ and if we use the mean squared error to **pick** the models, we may pick the models that **overfit to the noise**.

Accuracy of Regressors (cont)

Example: (Calculating the **Empirical Regression Error**): Given the data with

X	Actual	Prediction
4.21	11.64	10.96
2.85	7.47	7.77
2.13	5.96	6.09
0.69	2.36	2.71
0.82	3.21	3.02
4.68	10.80	12.04
2.69	7.15	7.38
4.99	13.11	12.77
2.39	7.38	6.71
3.87	10.52	10.15

Accuracy of Regressors (cont)

Example: (cont) The **default** empirical regression error is usually the mean square error that we are familiar with:

$$\begin{aligned} MSE &= \frac{1}{10}((11.64 - 10.96)^2 + ... + (10.52 - 10.15)^2) \\ &= 0.30231 \end{aligned}$$

Occationally, empirical MAE will be used:

$$\begin{aligned} MAE &= \frac{1}{10}(|11.64 - 10.96| + ... + |10.52 - 10.15|) \\ &= 0.453 \end{aligned}$$

Accuracy of Regressors (cont)

Why is overfitting a problem?

- An overfitting model works extremely well on **historical data** but may perform terribly with new data when dealing with new data. For example, an email spam filter that works well with known spam/ham but when it sees new spam emails, it treats them as ham, then users may receive a lot of spam mixing with ham which is annoying.
- An overfitting model is said to be **not generalising**, i.e. the model will still **provide a reasonable prediction on an unseen data** rather than providing a wrong prediction based on the overfitted noise.

Accuracy of Regressors (cont)

For example, if we have a 4-dimensional data D:

All Historical Data, D				
Index	X1	X2	X3	X4
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
h_D(X1,…,X4)				

If we train a predictive model with the all the data D, some predictive models give us **empirical error** of 0. This implies that **noise** ϵ_i is also accepted into the model h_D and when we apply the trained model to new data, the zero-empirical-error model will produce bad predictions nearly 100% of the time!

Accuracy of Regressors (cont)

To prevent overfitting problem, the **holdout method** (or **validation set approach**, or train/test split method) is used: some data (10% to 50%, typically 30%) are hold out for testing.

Holdout for Training, D1

Index	X1	X2	X3	X4
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

h_D1(X1,…,X4)

Holdout for Testing, D2

Index	X1	X2	X3	X4
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

The predictive model $h_{D_1}(X)$ trained on data D_1 can be tested against the 'unseened data' D_2 to h_{D_1}. So if the error is small, we can be a bit more confident that the model h didn't fit the noise too much.

Model Validation (cont)

Holdout method, validation set approach in R:

```
library(datasets)
set.seed(0)
test.index = sample(1:nrow(iris), size=0.4*nrow(iris))
X_y.test  = iris[ test.index, ]
X_y.train = iris[-test.index, ]
library(e1071)
clf = svm(Species ~ ., data = X_y.train, kernel='linear')
predicted = predict(clf, newdata=X_y.test)
conftbl   = table(predicted, X_y.test$Species)
# Accuracy of prediction
sum(diag(conftbl))/sum(conftbl)
```

Model Validation (cont)

Holdout method, validation set approach in Python:

```
from sklearn.model_selection import train_test_split
from sklearn import datasets, svm
X, y = datasets.load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.4, random_state=0)
clf_obj = svm.SVC(kernel='linear', C=1)
clf = clf_obj.fit(X_train, y_train)
# Accuracy of prediction (see Confusion matrix)
clf.score(X_test, y_test)
```

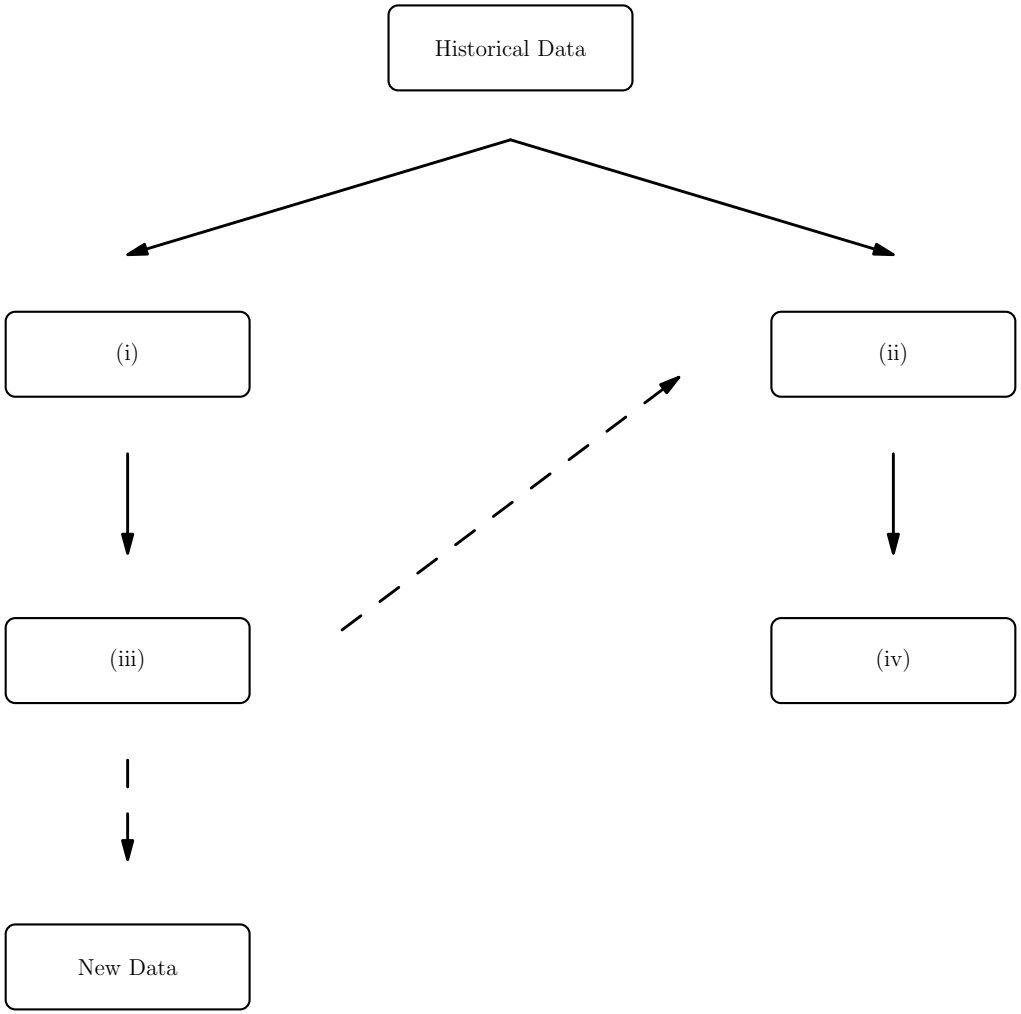
Model Validation (cont)

Example: (Final Exam Jan 2019, Q3)
(a) A predictive model can be built when historical data with known response are presented. The predictive model is then used to predict the response of a new data set with predictors given. Figure Q3(a) shows the process to form a predictive model.
Fill in the blanks (i) to (iv) in Figure Q3(a). State the differences between regression and classification for each step in the process of forming a predictive model.
(b) Give three examples on how statistical learning can help in risk/fraud analytics.

Model Validation (cont)

Example: (cont)

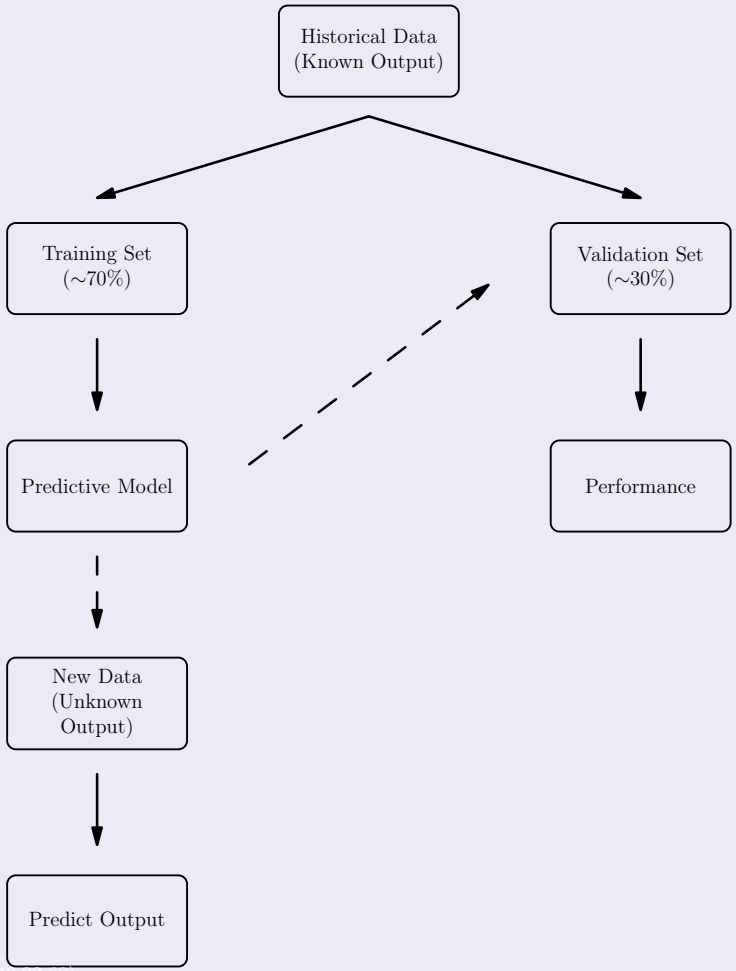
Figure Q3(a)



Model Validation (cont)

Example: (cont)

(a) Answer



Regression vs. classification

	Regression	Classification
Response	Numerical	Categorical
Split	Linear sampling	Stratified sampling
Performance	RSS, R^2	Confusion matrix
Scoring	$\hat{y}(\boldsymbol{x}) \pm$ s.d.	$\mathbb{P}(Y = j \mid \boldsymbol{X} = \boldsymbol{x})$

(b) Answer

- Banking industry uses credit scores to decide if an applicant can get a loan.
- Insurance industry predicts changes of an event to calculate premium.
- Financial institutions predicts frauds in transactions

May 2020 Final Assessment Q1(a)

Describe the differences between regression and classification for each step in the process of forming a predictive model with appropriate justifications.

(3 marks)

Answer

	Regression	Classification
Response	Numerical	Categorical
Split	Linear sampling	Stratified sampling
Performance	RSS, R^2	Confusion matrix
Scoring	$\hat{y}(\boldsymbol{x})\pm$ s.d.	$\mathbb{P}(Y = j\|\boldsymbol{X} = \boldsymbol{x})$

. . . . . [3 marks]

Accuracy of Multiclass Classifiers

Overfitting problem also occurs in classification problems. However, overfitting is usually less severe in classification problems compare to regression problems.

The fundamental accuracy measure of a classification problem is the **confusion matrix** or **contingency table** which is very easy to understand using the following picture.

To prevent the overfitting problem, the holdout method is used when a predictive model is tested. If the performance is good, then further statistical testing with more advanced resampling methods are required.

Accuracy of Classifiers (cont)

Example: Consider the Iris Flower Data in R, it has 3 classes, i.e. 3 species of iris flower — setosa, versicolor, virginica.

Consider training the kNN model with $k = 1$ (to be studied in Week 10) on all the 150 Iris data.

```
N = nrow(iris)
library(class)  # for knn
M = ncol(iris)  # 5 columns
X = iris[,-M]
Y = iris[, M]
Yhat = knn(train=X, test=X, cl=Y, k=1)
library(gmodels) # dependencies: gtools, gdata
CrossTable(Y, Yhat, prop.chisq=FALSE)
```

Confusion Matrix (cont)

```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  150

             | Yhat
           Y |     setosa | versicolor |  virginica |  Row Total |
-------------|------------|------------|------------|------------|
      setosa |         50 |          0 |          0 |         50 |
             |      1.000 |      0.000 |      0.000 |      0.333 |
             |      1.000 |      0.000 |      0.000 |            |
             |      0.333 |      0.000 |      0.000 |            |
-------------|------------|------------|------------|------------|
  versicolor |          0 |         50 |          0 |         50 |
             |      0.000 |      1.000 |      0.000 |      0.333 |
             |      0.000 |      1.000 |      0.000 |            |
             |      0.000 |      0.333 |      0.000 |            |
-------------|------------|------------|------------|------------|
   virginica |          0 |          0 |         50 |         50 |
             |      0.000 |      0.000 |      1.000 |      0.333 |
             |      0.000 |      0.000 |      1.000 |            |
             |      0.000 |      0.000 |      0.333 |            |
-------------|------------|------------|------------|------------|
Column Total |         50 |         50 |         50 |        150 |
             |      0.333 |      0.333 |      0.333 |            |
-------------|------------|------------|------------|------------|
```

Accuracy of Classifiers (cont)

Example: (cont)
The diagonal of the confusion matrix indicates a 100% data match! There are 50 species of setosa, 50 species of versicolor, 50 species of virginica and the table indicates a 100% match for each species. This means the empirical error is 0% and the accuracy is 100%

This is an overfitting case.

To avoid overfitting, we will use 70%-30% holdout method, i.e 70% is used for training and 30% is used for testing.

Accuracy of Classifiers (cont)

Example: (cont) The 70%-30% holdout method with linear sampling in R is shown below.

```
N = nrow(iris)
library(class)  # for knn
M = ncol(iris)  # 5 columns

# We are using linear sampling instead of stratified
# sampling since the data distribution is uniform.
set.seed(2022)
indices = sample(N, 0.7*N)  # Sample 70% w/out replacement
X.train = iris[ indices, -M]
Y.train = iris[ indices,  M]
X.test  = iris[-indices, -M]
Y.test  = iris[-indices,  M]
Yhat = knn(train=X.train, test=X.test, cl=Y.train, k=1)
table(Yhat,Y.test)
```

Accuracy of Classifiers (cont)

Example: (cont) The confusion matrix is

	Y.test		
Yhat	setosa	versicolor	virginica
setosa	15	0	0
versicolor	0	11	0
virginica	0	2	17

We can see that the accuracy = $\frac{15+11+17}{15+11+17+2} = 0.9556$

Not 100% but more than 90% which indicates that kNN ($k = 1$) may be acceptable. But to prevent overfitting. More resampling methods are needed to give a good confidence of the accuracy.

Accuracy of Binary Classifiers

For a classification problem with binary outcomes (only 2 classes), positive (+) and negative (-), the confusion matrix / contingency table can be presented as follows.

		Actual observations		
		Positive (+)	Negative (-)	**Precision**
Predicted	Positive (+)	True Positive Count (TP)	False Positive Count (FP)	Positive Predictive Value (PPV)
	Negative (-)	False Negative Count (FN)	True Negative Count (TN)	Negative Predictive Value (NPV)
	Recall	True Positive Rate (TPR) (Sensitivity)	True Negative Rate (TNR) (Specificity)	Accuracy (ACR)

Accuracy of Binary Classifiers (cont)

The performance measures are incorporated to the usual confusion matrix in the previous slide with the following formula:

- $\text{TPR} = \frac{\text{TP}}{\text{TP+FN}}$
- $\text{TNR} = \frac{\text{TN}}{\text{FP+TN}}$
- $\text{PPV} = \frac{\text{TP}}{\text{TP+FP}}$
- $\text{NPV} = \frac{\text{TN}}{\text{FN+TN}}$
- $\text{ACR} = \frac{\text{TP+TN}}{\text{TP+FP+FN+TN}}$

Accuracy of Binary Classifiers (cont)

Example 1.7.2: A predictive model has been built by using the training set. After predicting the outcome (fraud, not fraud) by implementing the model into validation set, the results are recorded as follow:

- Numbers of customer predicted to be fraud and the prediction is correct = 70
- Numbers of customer predicted to be fraud and the prediction is incorrect = 30
- Numbers of customer predicted not to be fraud and the prediction is correct = 80
- Numbers of customer predicted not to be fraud and the prediction is incorrect = 20

Accuracy of Binary Classifiers (cont)

Example: (cont)

		True Class	
		Fraud (+)	Not Fraud (-)
Predicted Class	Fraud (+)	70 (TP)	30 (FP)
	Not Fraud (-)	20 (FN)	80 (TN)

Calculate the accuracy measures sensitivity, specificity, PPV, NPV, ACR, FPR, FNR.

Accuracy of Binary Classifiers (cont)

The R implementation of the performance measurements for the example is listed below.

```
lvs   <- c("not fraud", "fraud")  # -> class 1, 2
lvs.r <- c("fraud", "not fraud")  # Show fraud first
truth <- factor(rep(lvs, times=c(110, 90)),
                levels=lvs.r)
pred  <- factor(c(rep(lvs, times=c(80, 30)),
                  rep(lvs, times=c(20, 70))),
                levels=lvs.r)
xtab  <- table(pred, truth)
TPR = xtab[1,1]/sum(xtab[,1])  # Sensitivity
TNR = xtab[2,2]/sum(xtab[,2])  # Specificity
PPV = xtab[1,1]/sum(xtab[1,])
NPV = xtab[2,2]/sum(xtab[2,])
FPR = 1 - TNR
FNR = 1 - TPR
```

Accuracy of Binary Classifiers (cont)

Another common metric for binary classification problem is the F1 score:

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

which is the harmonic mean of precision and recall.

Example (cont): The F1 score is

$$F1 = \frac{70}{70 + 0.5(30 + 20)} = 0.7368421$$

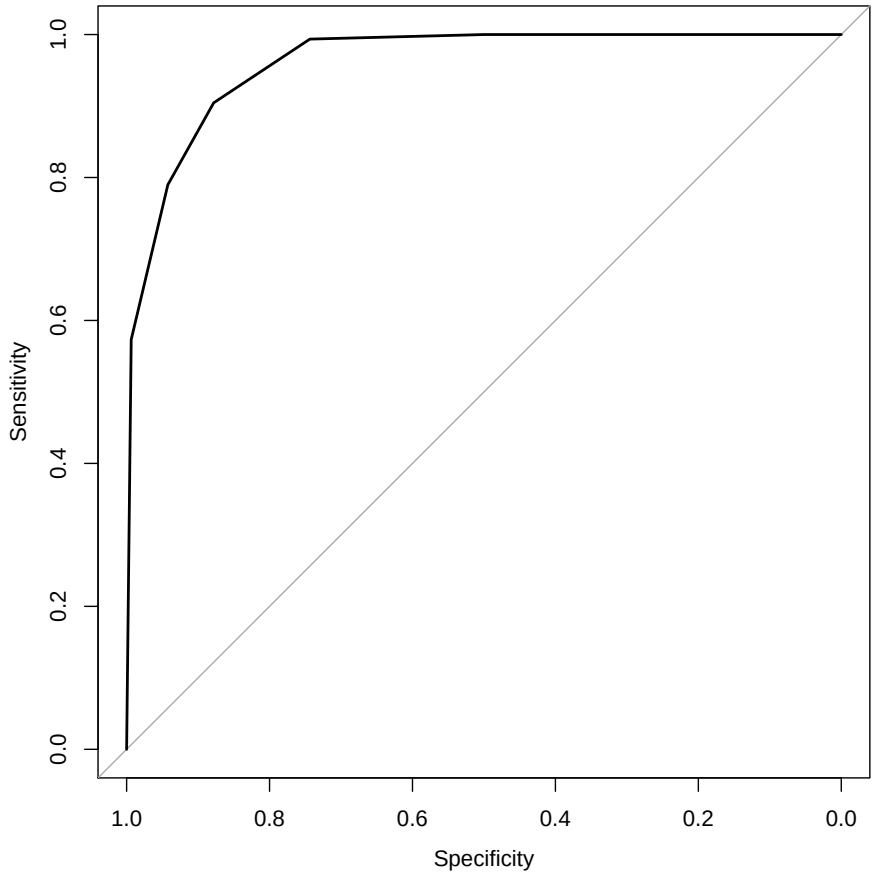
Accuracy of Binary Classifiers (cont)

Another accuracy measure of a binary classifier based on probability cut-off is the **Receiver Operating Characteristic (ROC)** curve, which is a plot of "sensitivity" vs "1−specificity" ("TPR vs FPR").

```
library(ISLR)              # Work with Example from Main Reference
Smarket = Smarket[,-1]   # Remove Year
N = nrow(Smarket)
set.seed(59)
train_idx = sample(N, size=0.75*N)
Smarket_train = Smarket[ train_idx,]
Smarket_test  = Smarket[-train_idx,]
library(class)  # for knn
M = ncol(Smarket)
Smarket_predict = knn(train=Smarket_train[,-M], test=Smarket_test[,-M],
                  cl=Smarket_train[,M], k=5, prob=TRUE)
library(pROC)
prob = attr(Smarket_predict,"prob")
prob = ifelse(Smarket_predict=="Up", prob, 1-prob)
proc.obj = roc(Smarket_test[,M], prob, plot=TRUE)
cat("AUC =", auc(Smarket_test[,M], prob), "\n")
```

Accuracy of Binary Classifiers (cont)

The ROC Curve generated by the R script is



The area under the ROC curve is 0.961661 (the higher the better).

Decision Making Example

You have just landed a great analytic job with ACME Inc., one of the largest telecommunication firms in United States. They are having a major problem with customer retention in their wireless business. In the Mid-Atlantic region, 20% of cell phone customers leave when their contracts expire, and it is getting increasingly difficult to acquire new customers.
Since the cell phone market is now saturated, the huge growth in the wireless market has tapered off. Communications companies are now engaged in battles to attract each other's customers while retaining their own. Customers switching from one company to another is called churn.

Decision Making Example (cont)

You and your team have been called in to help understand the problem and to devise a solution. The data set given consists of 2,000,000 observations with 10 predictors. Your team decided to build several predictive models using different methods. The models are then tested with the validation data set. The results of testing are shown as below:

Model	TP	FP	TN	FN
3-Nearest Neighbour	281,609	31,291	263,077	24,023
500-Nearest Neighbour	181,301	51,014	243,354	124,331
LR (all predictors)	243,344	55,194	239,174	62,288
LR (significant predictors)	249,487	61,493	232,875	56,145

Based on the information given, make a decision on the model that is suitable for churn prediction to be proposed to the company. Discuss on your decision. . . . . **Online Class Discussion?**

Outline

Deployment

Real-world deployment:

- Spam filtering for emails & SMS
- Intrusion detection
- ???

Designing dashboards:

- Tableau:
- Qlikview:
- Power BI:
- D3.js
- ...

Use Cases

Classification:

- Email → spam / non-spam;
- Tumour → benign / malignant;
- Writing → characters / words;
- Image → label;
- Activity → fraud / non-fraud

Use Cases (cont)

Regression:

- Predicting insurance premiums
- Motor insurance pricing using GLM
- Econometrics
- Physics / Engineering problems involving sensor measurements

Use Cases (cont)

Unsupervised learning:

- clustering COVID-19 viruses and other coronavirus
- market segmentation
- visualisation
- dimensionality reduction

Real-World Concern

Pros & Cons of AI.
E.g. AI in changing background

- Pro: Video production
- Con: Fake scene: A cooking background can be turned to a murder background.

E.g. AI in removing someone from the scene or changing face

- Pro: Video production, e.g. removing “tourists” from a documentary scene.
- Con: “Fake” CCTV, Scam, etc.