Predictive Analytics – Session 3

Machine Learning Tools: Supervised Learning
Introduction to Unsupervised Learning

Associate Professor Ole Maneesoonthorn

Associate Professor in Econometrics and Statistics

Melbourne Business School

O.Maneesoonthorn@mbs.edu

What predictive tools are available?

The Predictive Tools

Some common methods:

- Linear regressions
- Logistic regressions

MODEL-BASED PREDICTIONS

You have seen these in DA or BA

- Regression trees
- Neural networks
- Clustering: k-means, k-NN, etc…
- Ensembles: bagging, boosting, random forest

ALGORITHM-BASED PREDICTIONS/ MACHINE LEARNING

The Predictive Tools

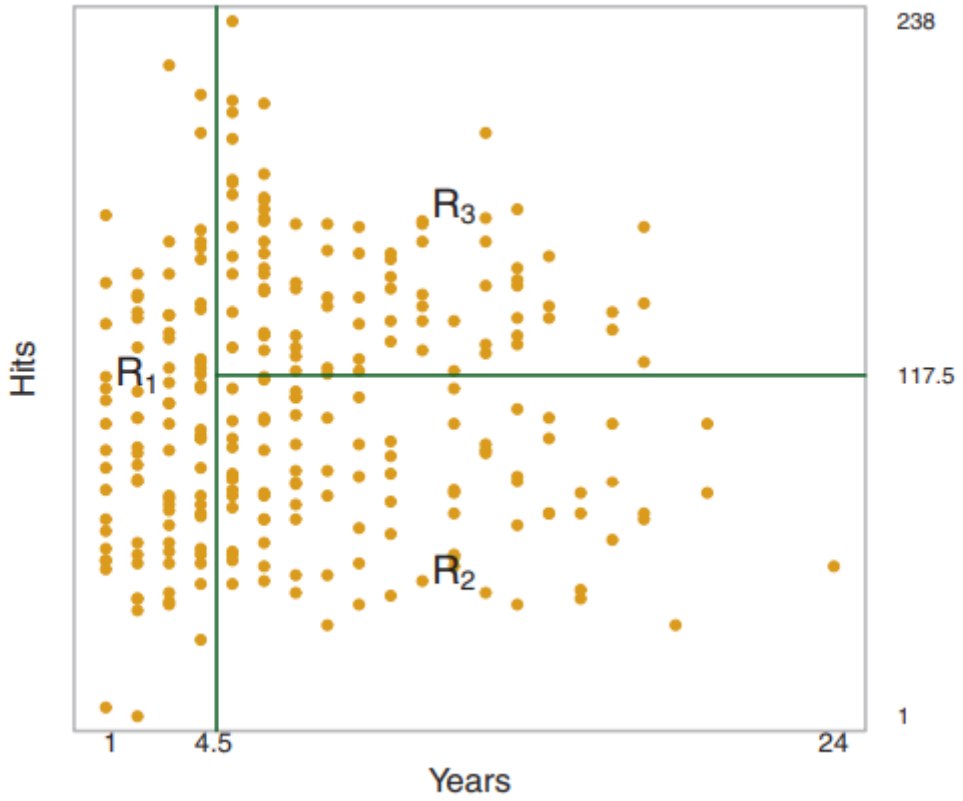
Regression Trees

- All about **segmenting** the space of the output variable Y
- By **cutting the space** into the regions of the input variables $X_1, X_2, \dots, X_k$
- The average of each sub-region is then taken to be the predicted value
- **Note**: it can be applied to both continuous data and classifications
  - Continuous data: "**Regression Trees**"
  - Classification: "**Decision Trees**"

The Predictive Tools

Regression Trees

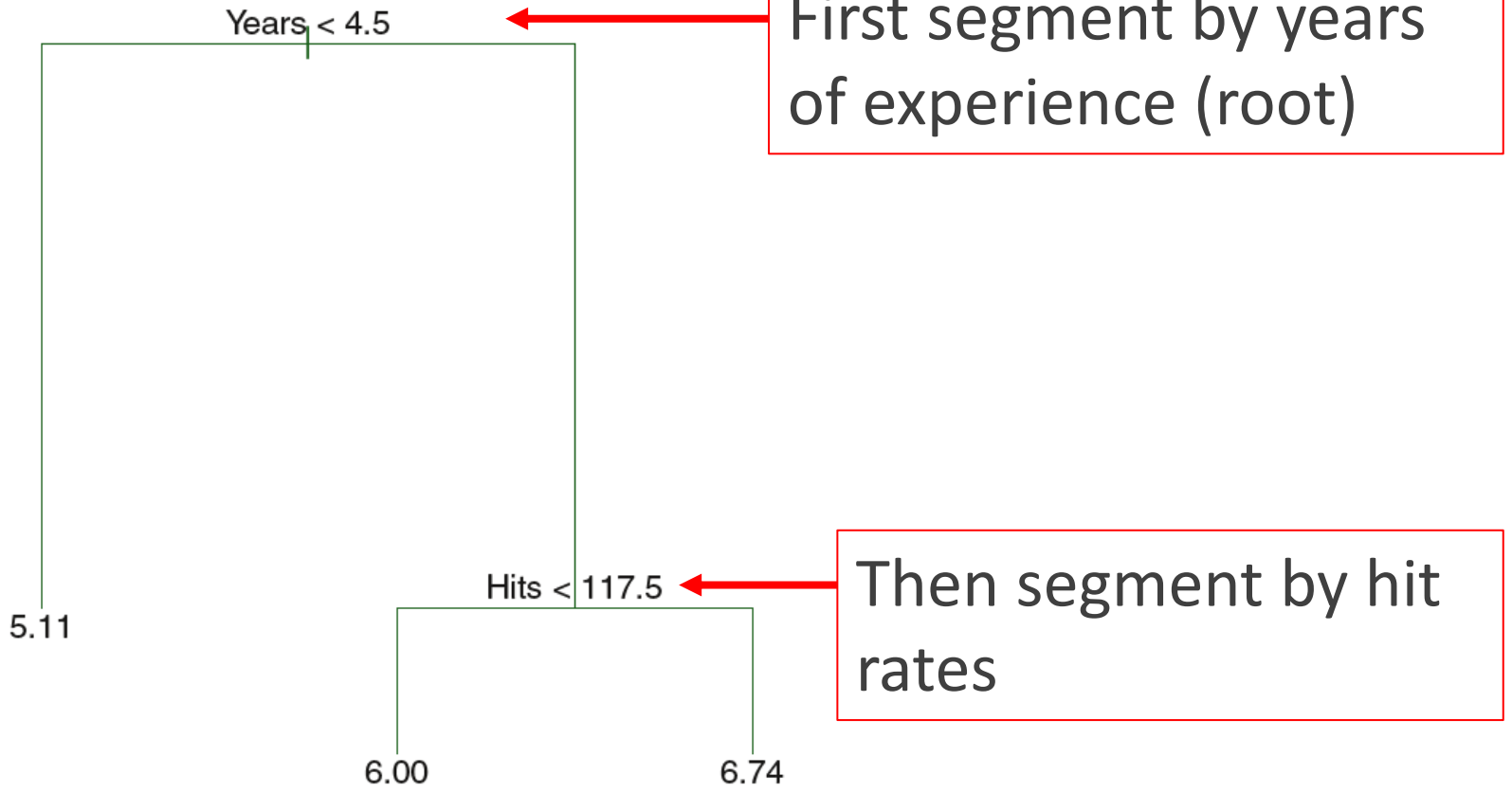
- A simple example: Baseball player salary



The Predictive Tools

Regression Trees

- The tree representation:



The Predictive Tools

Regression Trees

- Goal: Find the segmentation that **minimizes sum of squared errors**
- BUT, to look at every single combination is cumbersome
- Use **algorithm** for segmentation → a **tree construction**
- Also known as **CART** (Classification and Regression Tree) method

The Predictive Tools

Regression Trees

Considerations in tree constructions:

- Which variable is going to form the "**root**"?
- How do we **split** the variable at each node?
- How **big (deep/tall)** should the tree be? That is when do we decide that a node is a "leaf"?
- Once we arrive at the leaf node, what value(s) would be our prediction?

The Predictive Tools

Regression Trees

The algorithm:

1. Choose the root variable to split and the split point

2. Do this again at the next level to break the node

3. Repeat Step 2. until the incremental gain reaches a tolerance level (that is, you have reached the leaf nodes)

The Predictive Tools

Regression Trees

Some important **constraints/controls**:

- Minimum number of observations per node in order for a split to be allowed (default is 20 in R)
- Minimum number of observations per leaf node
- Tolerance level for predictive gain (this is known as the complexity parameter in R)
  - Higher number → a certain split must achieve a lot to proceed → smaller tree
  - Lower number → a certain split only have achieve a small gain to proceed → larger tree

The Predictive Tools

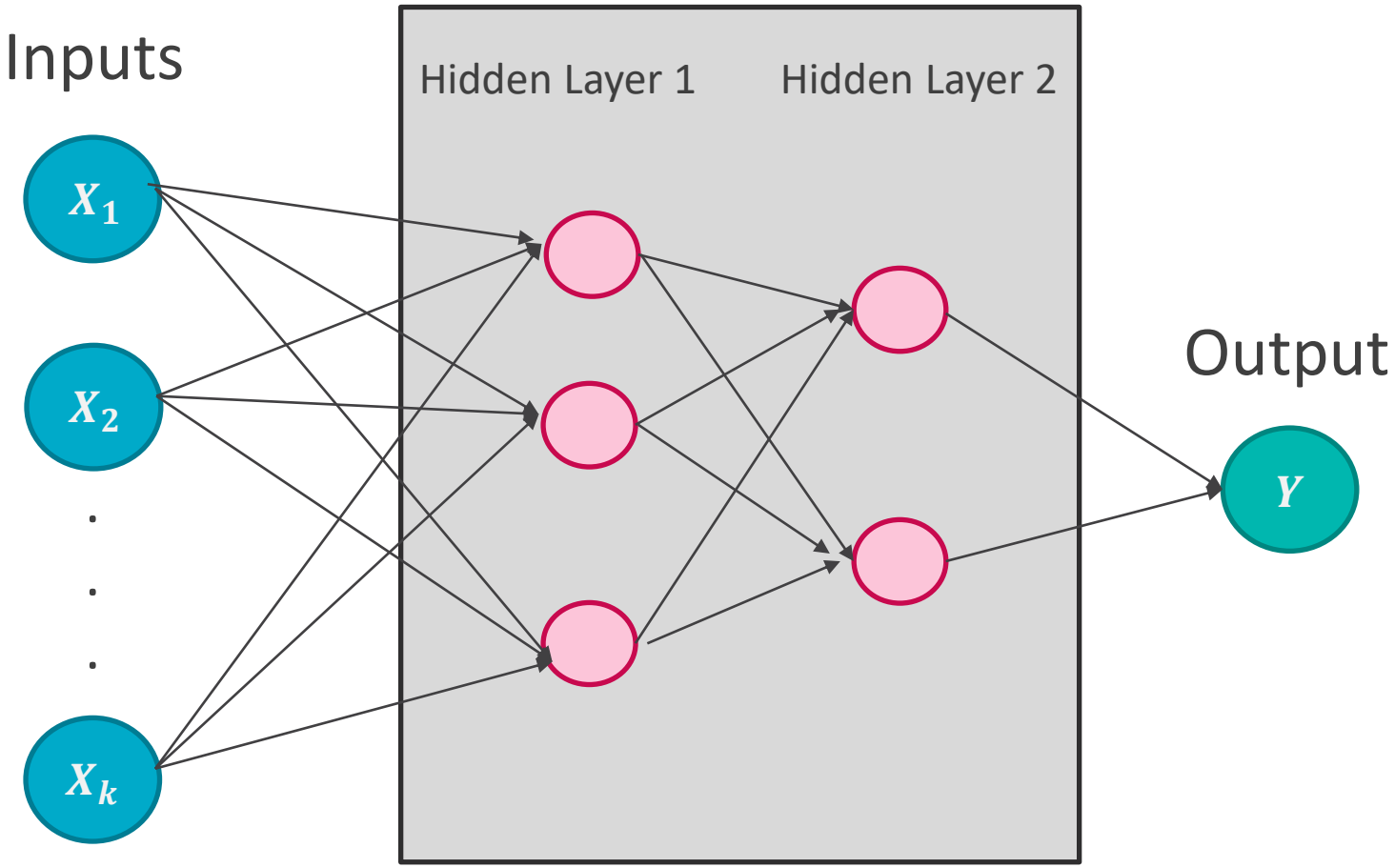
Neural Networks

- You have some input variables $X_1, X_2, \ldots, X_k$
- You know it is **related to** (and can help predict) the output Y
- But you do not want to assume a specific linear relationship
- Input and output variables are linked via **hidden layer neurons**

The Predictive Tools

Neural Networks

- … with two hidden layers



The Predictive Tools

Neural Networks

- **Special case:** Neural network with no hidden layer & linear activation function
  - → Linear regression

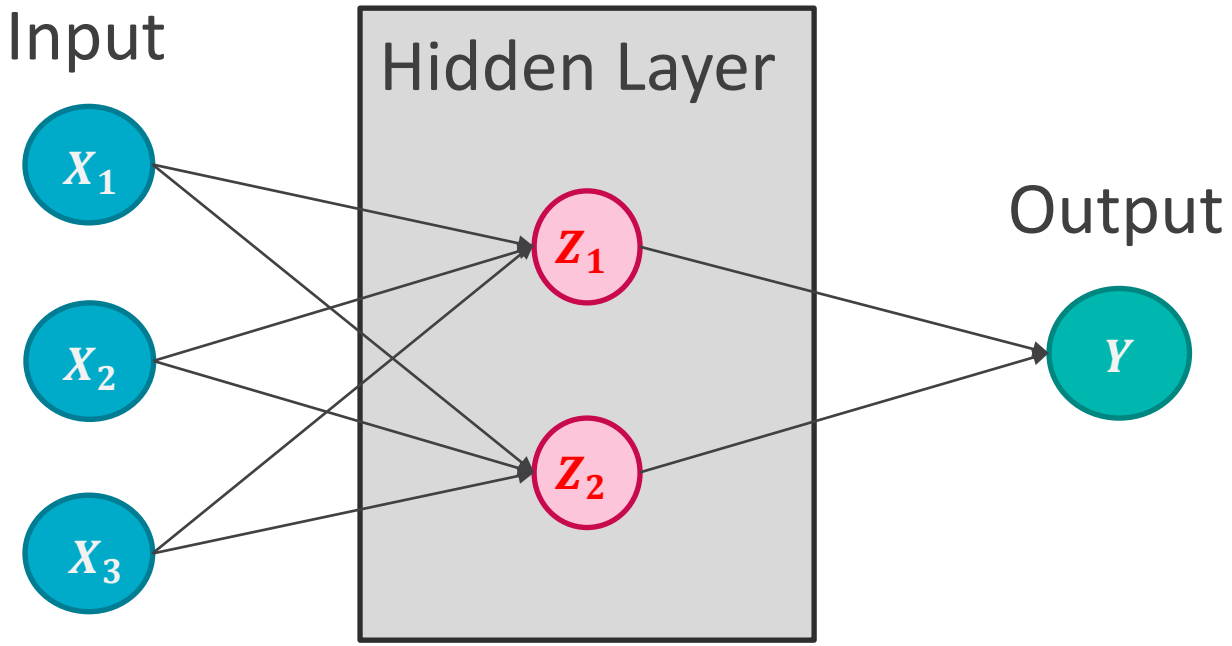
- **Choices** to be made:
  - How many hidden layers?
  - How many nodes at each layer?

- Choose the structure that predicts best!

The Predictive Tools

Neural Networks

- How are the inputs linked to the output?
- Let's look at a simple example with one hidden layer



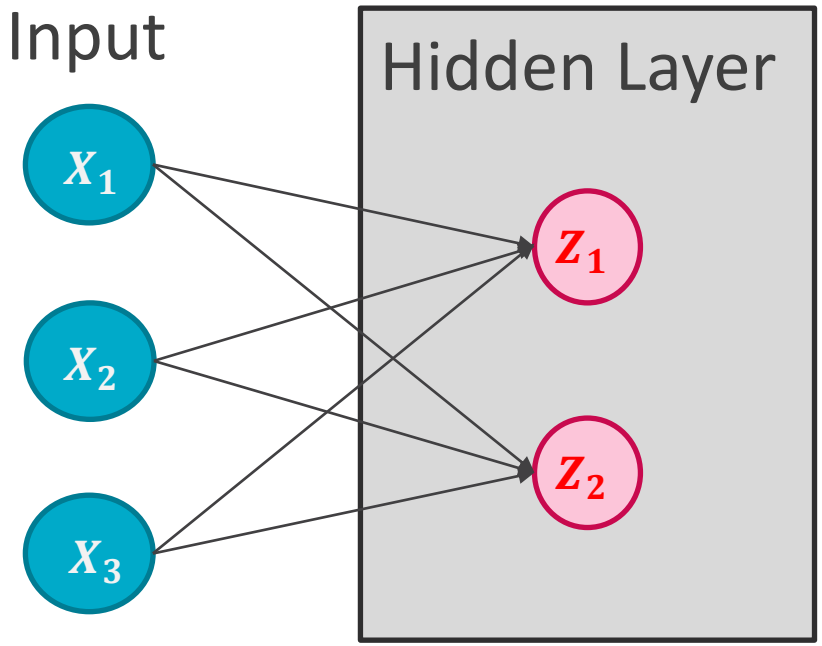
The Predictive Tools

Neural Networks

- Input to hidden layer nodes:

$$Z_1 = f_h(b_1 + w_{1,1}X_1 + w_{1,2}X_2 + w_{1,3}X_3)$$

$$Z_2 = f_h(b_2 + w_{2,1}X_1 + w_{2,2}X_2 + w_{2,3}X_3)$$



$f_h(.)$ is called the integration function

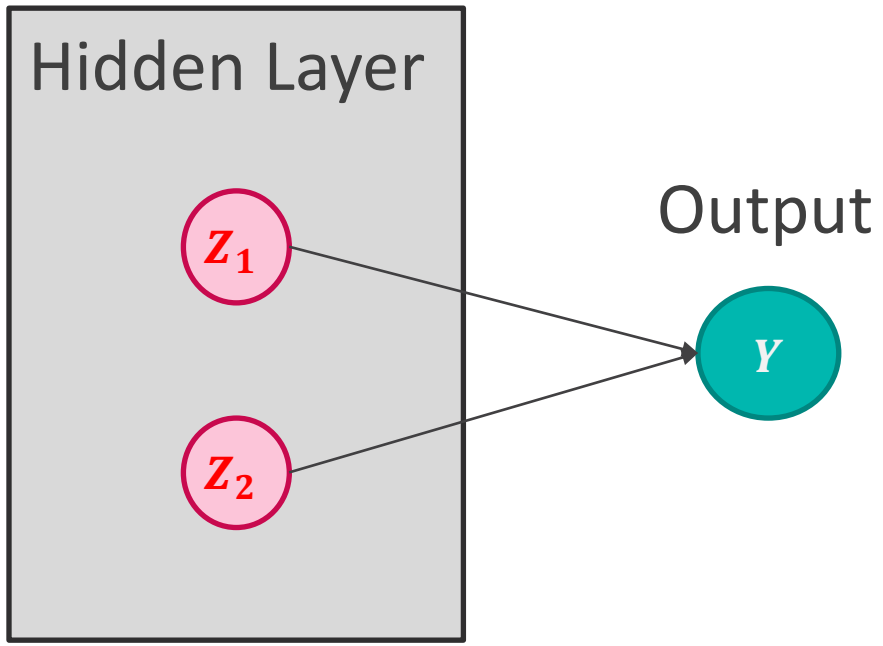
- The subscript "h" indicates it is for the hidden layer translation.
- Typical choices: linear or sigmoid
- The sigmoid function: $\frac{1}{1+e^{-x}}$

The Predictive Tools

Neural Networks

- Hidden layer to output:

$$\hat{Y} = f_o(a_0 + a_1 Z_1 + a_2 Z_2)$$



- $f_o(.)$ is the activation/linking function for output layer
- Typical choice: sigmoid, softmax (depends on the type of output variable)

The Predictive Tools

Neural Networks

- **Training the neural network**: search for "weights"

$$(b_1, w_{1,1}, w_{1,2}, w_{1,3}, b_2, w_{2,1}, w_{2,2}, w_{2,3}) \text{ and } (a_0, a_1, a_2)$$

- That minimizes the sum of squared error (SSE)

- **Algorithm** involves:
  - Set a starting value for weights – **simulation**
  - Use stochastic gradient descent to work out the next step – **simulation**
  - Repeat until the reduction in SSE is minimal

The Predictive Tools

Neural Networks

Issues to consider

- More layers (and more nodes) typically lead to reduction in SSE
  - Just like in multiple regression where R-squared increases with the number of inputs
  - Longer time to train – **optimization takes longer**
  - **Overfitting ≠ better prediction**
  - Too few layers/nodes means less flexibility to capture nonlinearity
- → Choice of number of hidden layers/nodes – **a balance** between flexibility and overfitting
- Results vary each time
  - Use of stochastic gradient descent & random starting values
  - Each package uses different algorithms

The Predictive Tools

Neural Networks

Issues to consider

- **Input scales**
  - Weights are **sensitive** to input data scale – can be unstable if data has really large/really small magnitudes
  - Solution – scale the input and output data first using **z-score**:

$$\text{Z score} = \frac{x - mean}{stdev}$$

  - **Train the network with scaled data**
  - Prediction will be in the scaled unit - convert back to raw data units by reversing the z-score calculation

The Predictive Tools

Neural Networks

- Predicting with neural network:

Apply the linking functions

- **Exact nature of relationship unknown!**

The Predictive Tools

Neural Networks

Relevant tools:

- Network plot
- Variable importance
  - Garson's relative importance
  - Olden's connection weights
- Sensitivity analysis

The Predictive Tools

Neural Networks

- **Garson's relative importance**
  - Measure the contribution in absolute relative magnitude
  - Ranges from 0 to 1 – higher indicate more important
  - Only applicable to network with one hidden layer and one output node
  - Direction of response cannot be determined

The Predictive Tools

Neural Networks

- **Olden's connection weights**
  - Sum of the product of input-hidden and hidden-output weights
  - Unit is sensitive to data scale and linking functions
  - Positive & negative weights will cancel
  - Can handle multiple hidden layers

The Predictive Tools

Neural Networks

- **Sensitivity analysis – Lek profile**
  - Set all explanatory variables fixed
  - Sequence the variable to interest from min to max
  - Predict the outcome
  - Plot to look into how the outcome changes as the variable of interest goes from min to max

The Predictive Tools

Neural Networks

Lek profile – how do we fix the 'other' variables?

1. Group by quantiles – fix at some percentiles
   - Default: min, 20th, 40th, 60th, 80th, max
   - You can change these

2. Group by clusters
   - If variables are related in a complex manner, grouping by quantiles may not make sense
   - E.g. when X1 is at the max, X2 may tend to be in its lower range
   - Group by cluster uses k-means clustering and fix the 'other' variables at the centre of each cluster
   - You just need to determine and number of clusters to use

The Predictive Tools

So far…

- Machine Learning methods:
  - Regression tree
  - Neural network
- These algorithms involve training the models using all input variables

  AND the outcome variable
- **→ "Supervised" learning**

The Predictive Tools

Unsupervised Learning

- Methods/models that are trained with **ONLY input variables**
- Outcome variable not part of the model training
- Typically have less structures (or no structure)
- Based on “similarities” of input variables
    - Definition of similarities vary with the method

The Predictive Tools

Unsupervised Learning

- Clustering methods
  - K-means
  - K-NN
  - Hierarchical clustering
- Support Vector Machines (SVM)

The Predictive Tools

Unsupervised Learning – K-means

- Partitioning data into ‘K’ clusters
- Clusters are **non-overlapping**
  - Each observation must belong to ONLY one of the K clusters
- Goal: find cluster membership that **minimize the within-cluster sum of square (WCSS) deviation from the centroid**
- **Centroid** = collection of mean values for each cluster
  - K-means matches similarities in the mean values

The Predictive Tools

Unsupervised Learning – K-means

The algorithm:

1. Start by assigning an integer (1 to K) at random to each observation. This is the cluster membership.
2. Compute the “centroid” of each of the K clusters – a collection of all means
3. Reassign the membership of each observation to the cluster with “closest” centroid (use WCSS)
4. Repeat 2. and 3. until membership assignment no longer changes

The Predictive Tools

Unsupervised Learning – K-means

Issues to consider: Raw data vs scaled data

- "Closest" distance is measured by within-cluster sum of square (WCSS)
- In each X dimension, we compute the squared distance between the observation and the cluster mean (centroid)
- WCSS is the sum of these distances over all X dimensions
- Raw data – each X dimension has different units! Results dominated by data with "larger" (numeric) measurement units

→ Use scaled data!

The Predictive Tools

Unsupervised Learning – K-means

Issues to consider: How many clusters should we use?

- Too few – no enough granularity in detected patterns
- Too many – overfitting & may end up with too few observations in each cluster (inaccuracy in any subsequent analysis)
- Three tools to help with selection ‘K’
    - The Elbow method
    - The Gap method
    - The Silhoutte method

The Predictive Tools

Unsupervised Learning – K-means

Issues to consider: How do we predict when output variable is not involved?

- Clustering typically used for exploratory analysis, but predictions can be derived
- Training data used to “train” the cluster – you will have “centroids” for each of the K clusters as a result
- Prediction – which cluster should the observation belong to given all of the X variables and the trained centroids?
    - Again, based on WCSS
    - This will give you “predicted” cluster assignment

The Predictive Tools

Unsupervised Learning – K-means

Issues to consider: How do we predict when output variable is not involved?

Option 1:

1. Compute the cluster membership using the training set (clustering based on X variables)
2. Once cluster membership is obtained, compute the mean of output variable Y for each cluster
3. Predict the cluster membership of test set/new observation
4. Use the relevant mean of Y as a prediction for the outcome

The Predictive Tools

Unsupervised Learning – K-means

Issues to consider: How do we predict when output variable is not involved?

Option 2:

1. Compute the cluster membership using the training set (clustering based on X variables)
2. Once cluster membership is obtained, use other supervised learning methods to construct E(Y|X)
3. You will end up with one model for each cluster
4. Predict the cluster membership of test set/new observation
5. Apply the predictive model for each cluster to predict the outcome variable Y

**Need to ensure there are enough observations in each cluster to construct a model

The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

- A clustering method
- Similarities now defined by distance between observations
  - Rather than distance from the centroids
- That is, for each observation, find its ‘K’ nearest neighbour observations
- There is no clusters per say, but there are neighbourhoods around an observation
- Neighbourhood can overlap

The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

- Similarities now defined by **distance between observations**
  - Rather than distance from the centroids
- That is, for each observation, find its ‘K’ nearest neighbour observations
- There is no clusters per say, but there are neighbourhoods around an observation
- **Neighbourhood can overlap**

The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

- Note that ‘K’ in this context is the number of observations in a neighbourhood
- Prediction for a particular observation is then obtained
  - Taking an average of the neighbourhood values for numerical variables
  - Calculating group probabilities for classification data
- K-NN regression in this session
- K-NN classification in Session 5

The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

- **K-NN regression**
  - For a particular point x_0 on the input space (this can be multiple dimension)
  - Find K neighbours around this value
  - Ask: for this neighbourhood, what is the average of the outcome variable?
  - This gives the predicted outcome $\hat{f}(x_0)$
  - When done on a range of input values, we can join the estimates to get a "regression" that is nonparametric

The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

K-NN regression – issues to consider

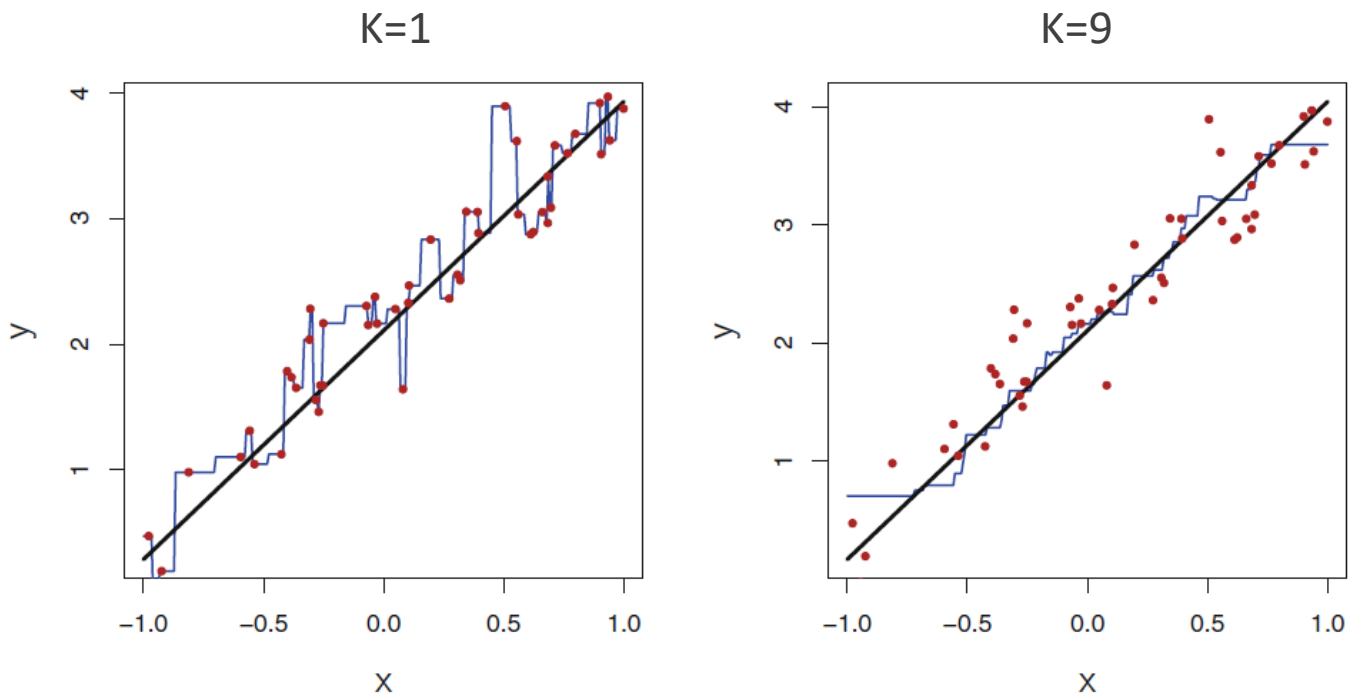
- What value of "K" should we use?
  - K=1 → Perfect fit to training data (no neighbour, just self)
  - Prediction function is a step function
  - Increasing K → smoother prediction function
  - **A general rule of thumb:** $\boldsymbol{K = \sqrt{N}}$

The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

K-NN regression – issues to consider

- What value of “K” should we use?



The Predictive Tools

Unsupervised Learning – k-Nearest Neighbour (k-NN)

K-NN regression – issues to consider

- How many input variables to include?
  - K-NN is known to suffer the **curse of dimensionality**
  - Suffers poor performance when there are many "noisy" predictors → poorly identified neighbourhoods
  - Works well when the input variables contain strong signals
  - Need a balance between number of observations and number of input variables

Reminder…

“All models are wrong…

but some are useful…”

George Box (1987)