MIS772

Predictive Analytics

Intro to Data Classification

Data Preparation

- Data Preparation

Intro to Data Classification

- Class vs classification
- Applications of classifiers
- Distance / similarity metrics
- Example Classifiers:
  - k-NN
  - Decision trees
- Model evaluation
  - Train, Test, Validate
  - Confusion matrix
- Performance vs interpretability

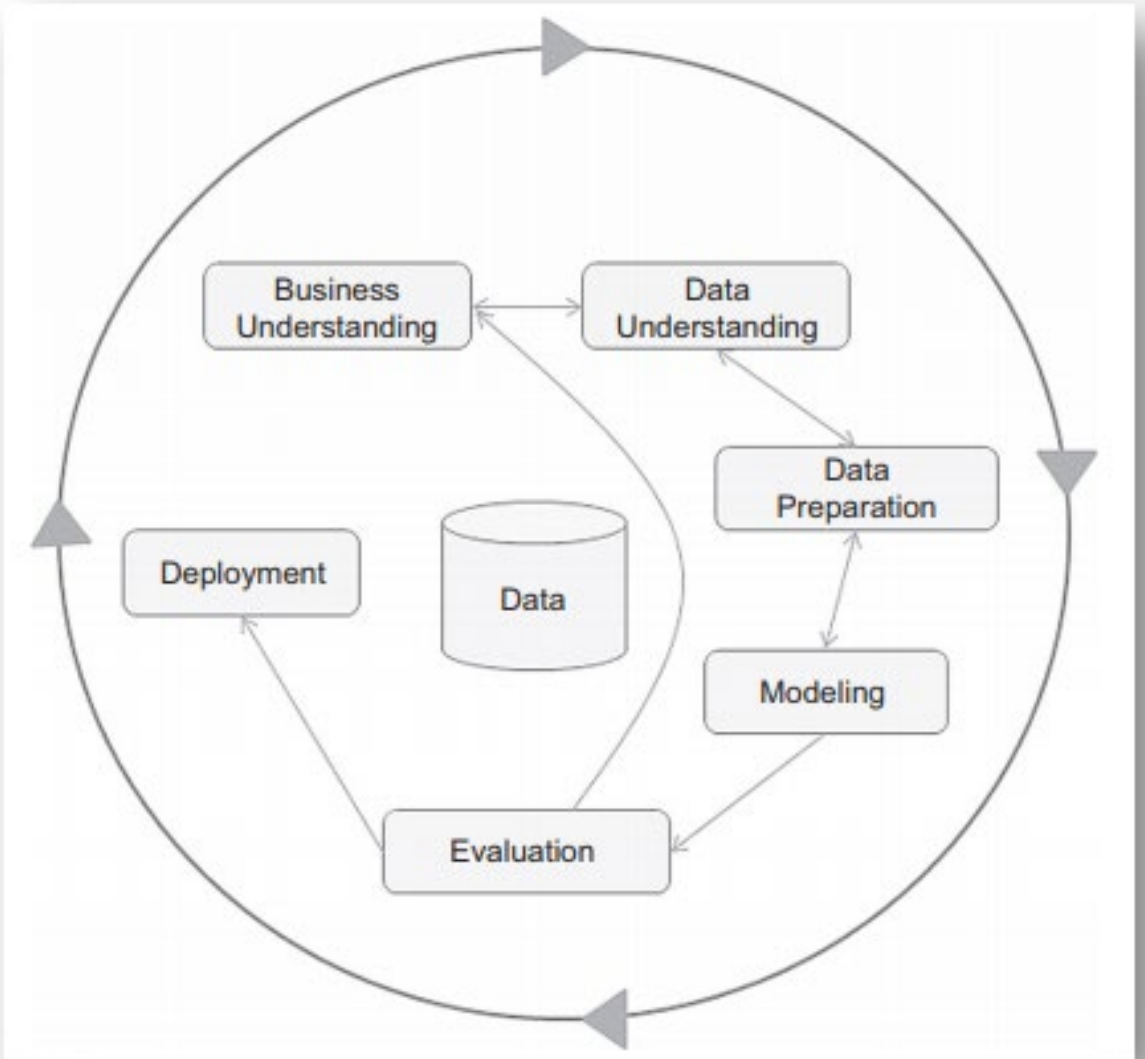
Refer to your textbook by Vijay Kotu and Bala Deshpande,
***Data Science: Concepts and Practice*, 2nd ed, Elsevier, 2018.**

DEAKIN BUSINESS SCHOOL

AACSB ACCREDITED

EFMD EQUIS ACCREDITED

The Predictive Analytics Process



DEAKIN BUSINESS SCHOOL

Data Quality

Rubbish in…rubbish out!

Very rarely data are available in the form suitable for predictive analysis

Data quality matters:

Accuracy: dealing with data errors or extreme cases that deviate from expectations.

Completeness: dealing with the lack of attribute values, lack of certain attributes, or presence of aggregated values only.

Consistency: dealing with discrepancies in the processes that generate the data.

Timeliness: dealing with the timeframes within which data are prepared.

…

(Alternatively: "The 3R's"):

Reliability
Relevance
Representativeness

Refer Reading: Seddon & Scheepers (2012). *European Journal Of Information Systems* 21(1), pp. 6–21

Mentimeter

DEAKIN BUSINESS SCHOOL

Data preparation

Preparation steps

Data cleaning
Data integration
Data transformation
Data reduction

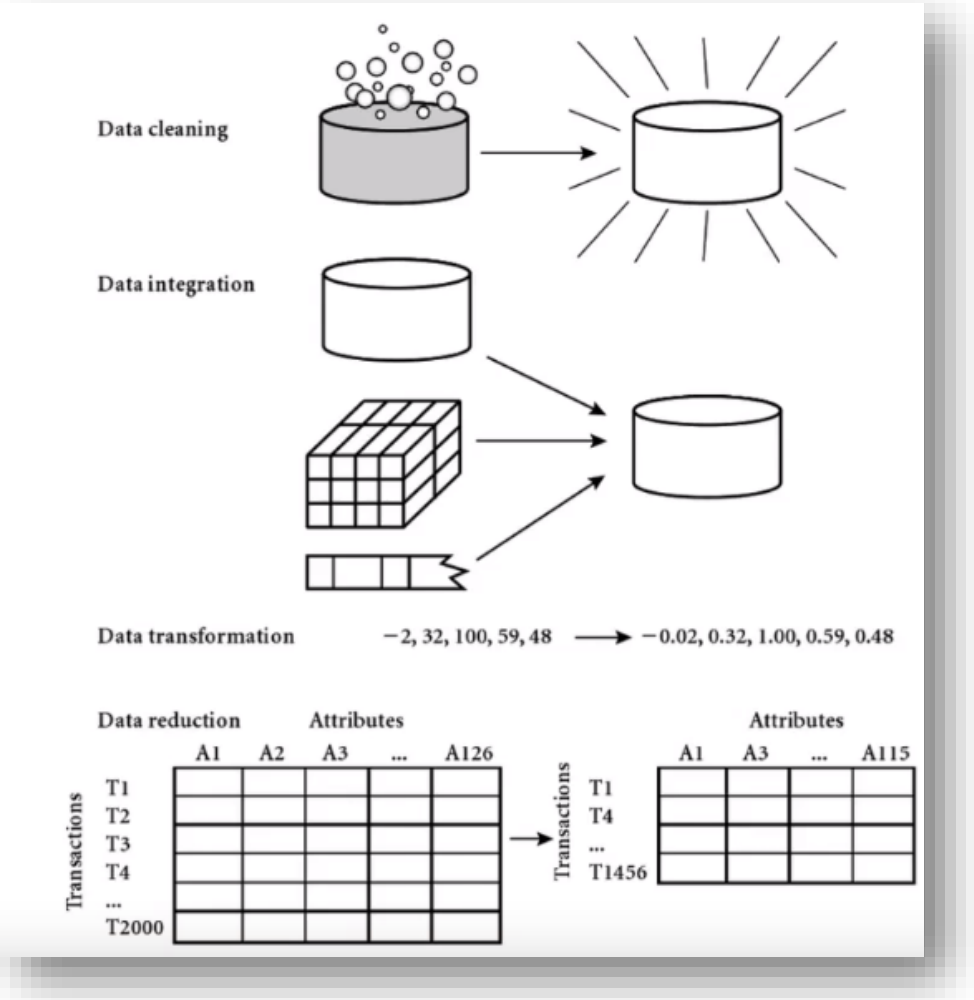


Image source: Page 87, Tan et al.
Introduction to data mining

Refer to your textbook by Vijay Kotu and Bala Deshpande,
***Data Science: Concepts and Practice*, 2nd ed,**
Elsevier, 2018., Chapter 2.

DEAKIN BUSINESS SCHOOL

Data Cleaning

Is the process of removing any inaccurate data (i.e., data error) or incomplete data (i.e., missing value) from a data set. This can be done via:
Replacing data errors or missing values
Modifying data errors
Deleting data errors or missing values

Table 2.1. Data Set

Borrower ID	Credit Score	Interest Rate
01	500	7.31%
02		6.70%
03	700	5.95%
04	700	6.40%
05	800	
06	8000	5.70%
07	750	5.90%
08	550	
09	650	6.50%
10	825	5.70%

missing values

data error, a value that deviates from expected values

Sample data set
Source of original table: Page 21, KD.Ch2
Manually changed for data errors and missing values

Missing value treatment

Approaches:

- Ignore or delete data points with missing values
- Ignore or delete data attributes with missing values
- Replace missing values with a constant value (e.g., 0 for numeric attributes)
- Replace missing values with the central tendency of the attribute
  - Mean for numeric attributes
  - Mode for nominal attributes
- Impute missing values with the most probable value (using learning methods such as k-NN)

Data preparation

- Data cleaning
- Data integration
- Data transformation
- Data reduction

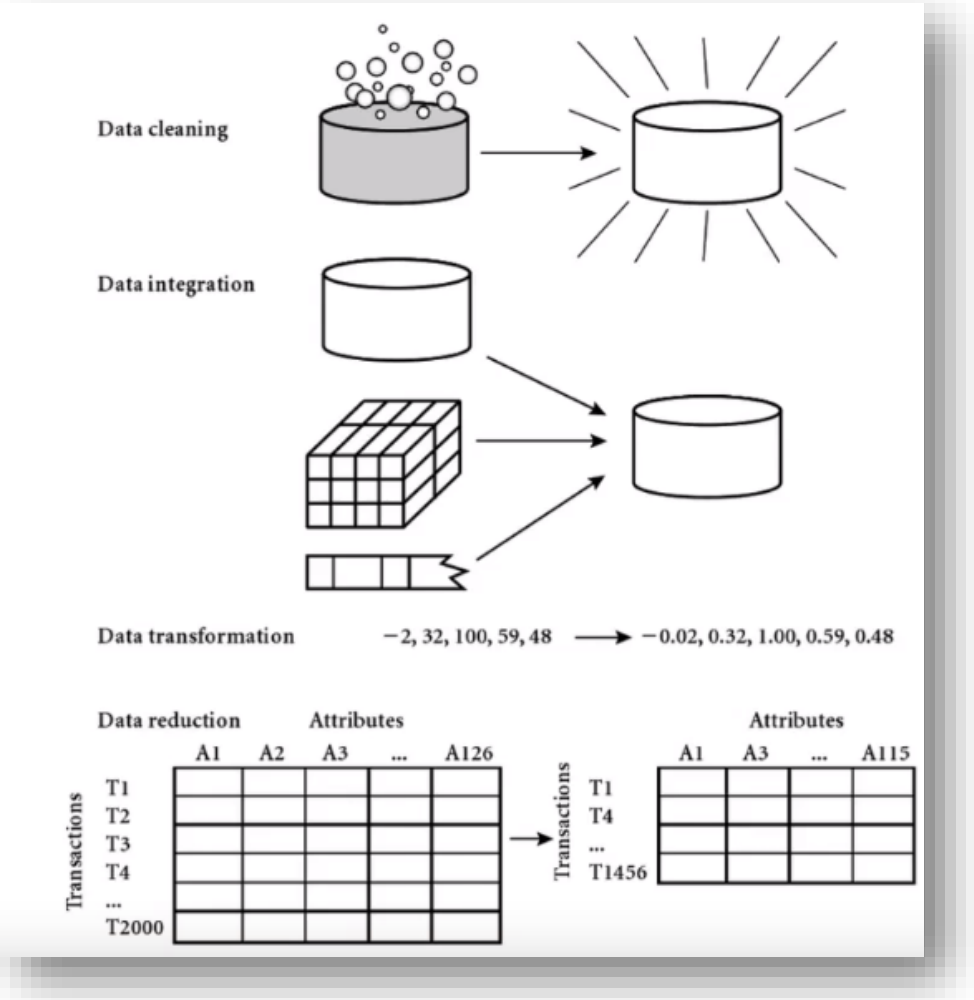


Image source: Page 87, Tan et al.
Introduction to data mining

Introduction to Data Classification

DEAKIN BUSINESS SCHOOL

Data classification

Class **of an example is a group defined by a unique nominal value of one of its attributes. For example vehicles can be grouped by:**

- **Colour, such as “red”, “yellow” or “green”**
- **Size, e.g. “small”, “medium” or “large”**
- **Numerical attributes, such as “x” and “y” map coordinates of the vehicle parking location, are not classes, however they can be useful in classification tasks**

Classification **is the process of classifying examples based on their attribute values, i.e. deciding what class value should be given to its label attribute, with a view to determine the membership of an example in a particular group:**

- **Vehicles type, such as “sedan” or “truck”, which can be predicted from other attributes, e.g. size**

Classifier **is a model predicting the class of examples, and thus capable of automating classification of future observations.**

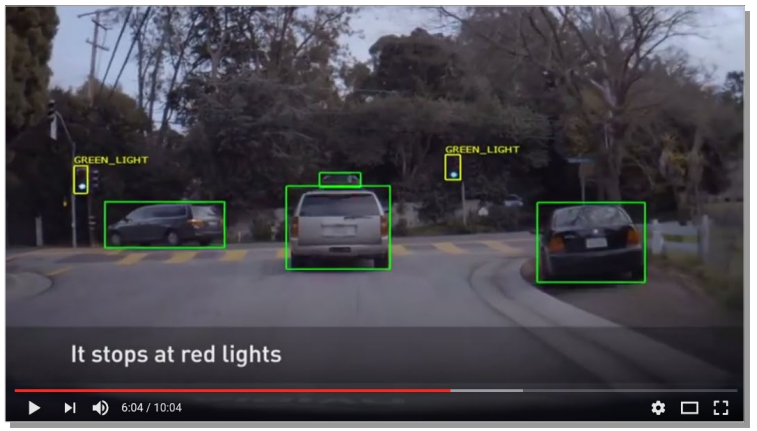
Group membership by colour

Group membership by vehicle size

Also see KD 4 on classification

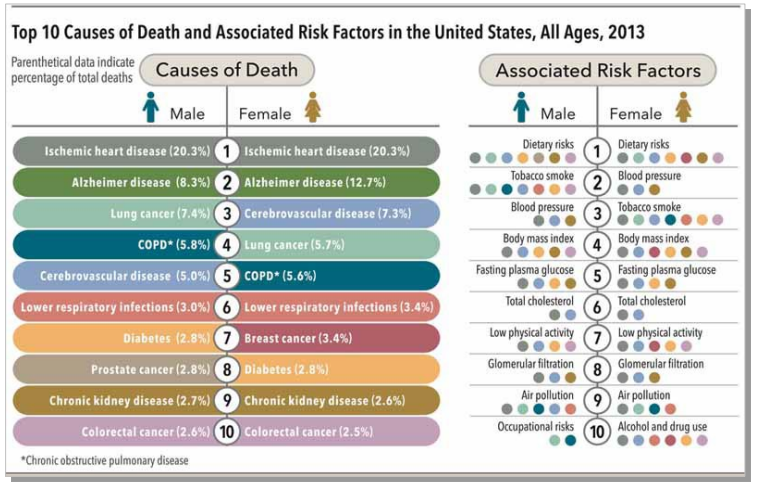
DEAKIN BUSINESS SCHOOL

Classifying objects



NVIDIA self-driving cars:
Should I drive or stop?
www.youtube.com/watch?v=MF9NwOTLLgE

Christopher Healey:
Is this tweet positive or negative towards the lecturer?
www.csc2.ncsu.edu/faculty/healey/tweet_viz/



USA Institute of Health Metrics and Evaluation:
What are my health risks?
www.healthdata.org/infographic/when-and-why-people-die...

IBM Watson – Morgan movie trailer:
Is this movie clip sufficiently scary to be included in a trailer?
www.youtube.com/watch?v=gJEzuYynaiw

Different kinds of classifications, i.e. to detect obstacles in front of self-driving cars, sense emotion of movie viewers, explain health risks and identify sentiment of Twitter messages. Once we classify past examples, we can then apply such classification to future individuals (predict their class / decision).

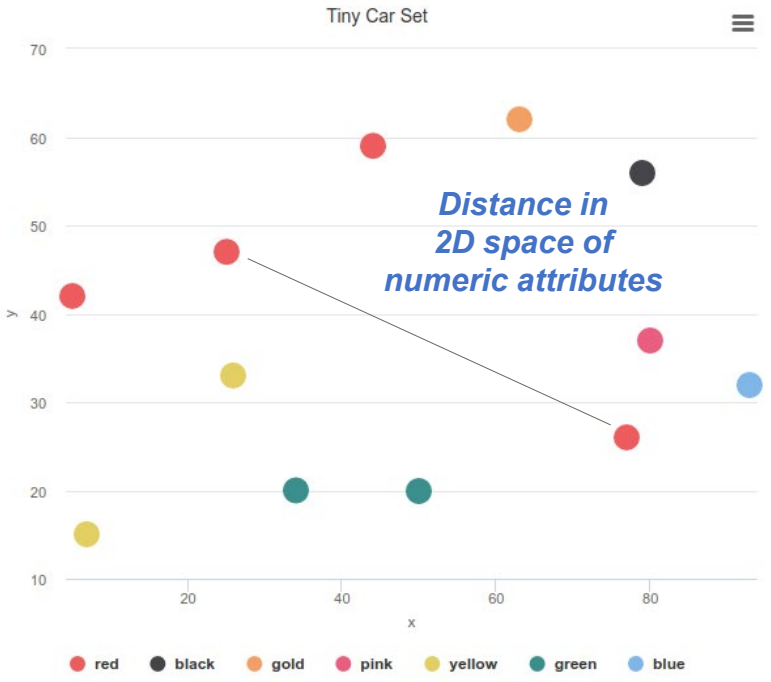
DEAKIN BUSINESS SCHOOL

Classification:
k Nearest Neighbour (k-NN)

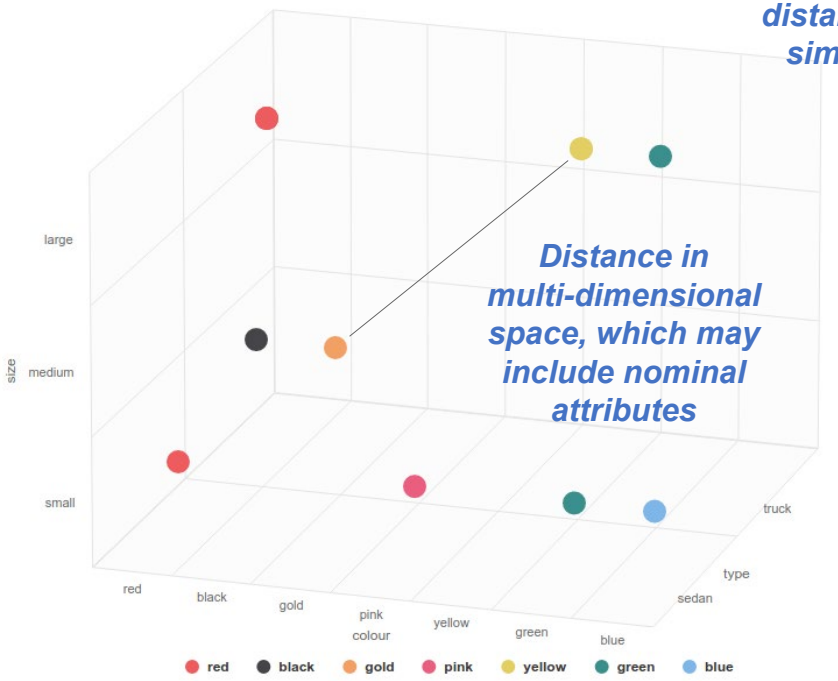
DEAKIN BUSINESS SCHOOL

Nearest Neighbour

- *Similarity models* compare new observations to the previously classified examples or their carefully selected subset called *prototypes*.
- Classification that looks at *k* most similar examples or prototypes is called a *k Nearest Neighbour* (or *k-NN*).
- In training k-NN classifiers, we could simply keep all previously classified examples.
- Such models are therefore called *lazy models* - they do very little!
- Applying the k-NN model: we first must identify k "nearest" examples, find their classification, and then determine the class to be assigned, e.g. the majority (of neighbours) wins.
- Measuring neighbour similarity or their distance is easy for numeric attributes, e.g. vehicle parking coordinates "x" and "y", but hard for nominal attributes such as "colour"."size" or "type".
- Nominal attributes can either be converted to numeric, e.g. using *dummy encoding*, or we can use non-numeric measures such as *nominal* or *mixed measures*.



Inverse of distance is similarity



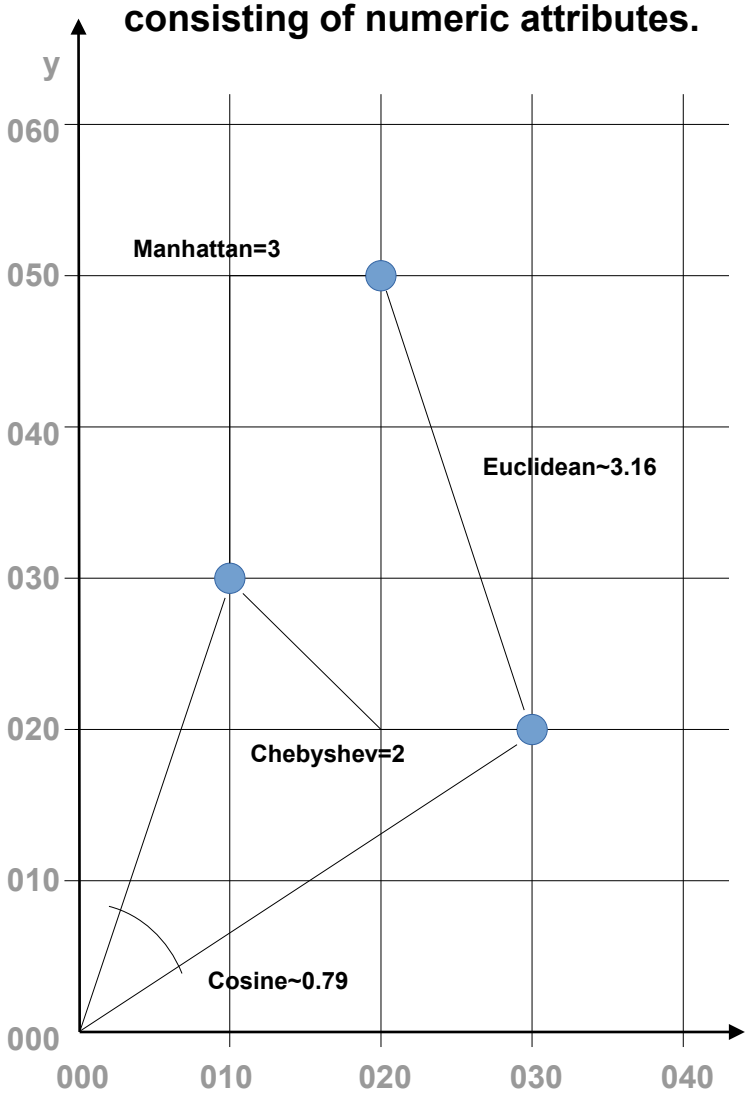
Also see KD 4.3 on k-Nearest Neighbour

DEAKIN BUSINESS SCHOOL

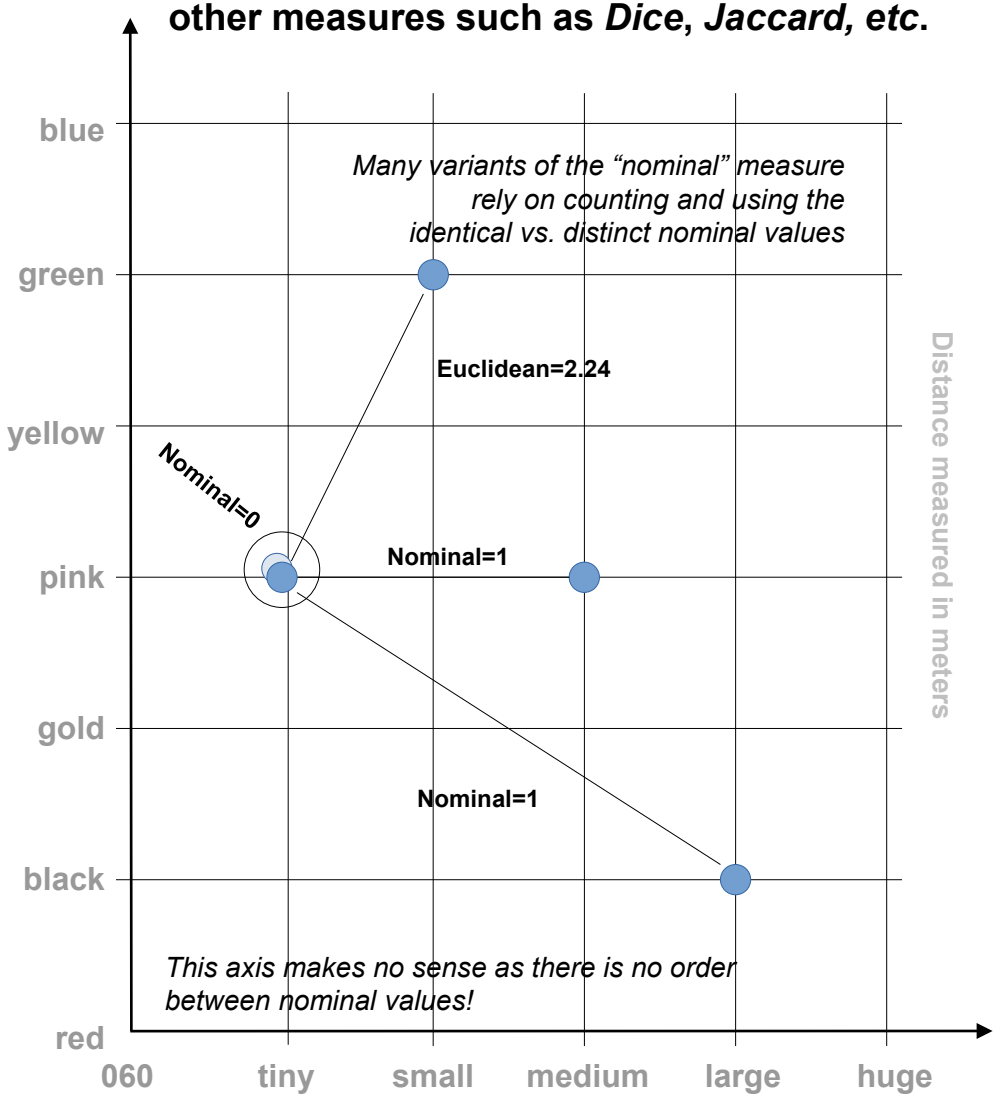
Consider Similarity, Distance also Neighbourhood and Proximity

How to measure similarity in data

There are many measures of distance / similarity between data points, e.g. *Euclidean*, *Manhattan*, *Chebyshev* or *Cosine* for data consisting of numeric attributes.



For data points made of ordinal attributes we can still use *Euclidean* measures, otherwise we can use a simple *Nominal* measure, or other measures such as *Dice*, *Jaccard, etc*.



RapidMiner supports many more metrics, which you can match against your analytic problem. For example, if the cars were to be parked in different street locations in New York then Manhattan measure would work great. If the car park was scattered across stars in Milky Way, then a cosine measure would work for managers stationed on Earth. If you wanted to determine if two cars were both on the same side of the car park, either left or right, then a nominal measure would be appropriate.

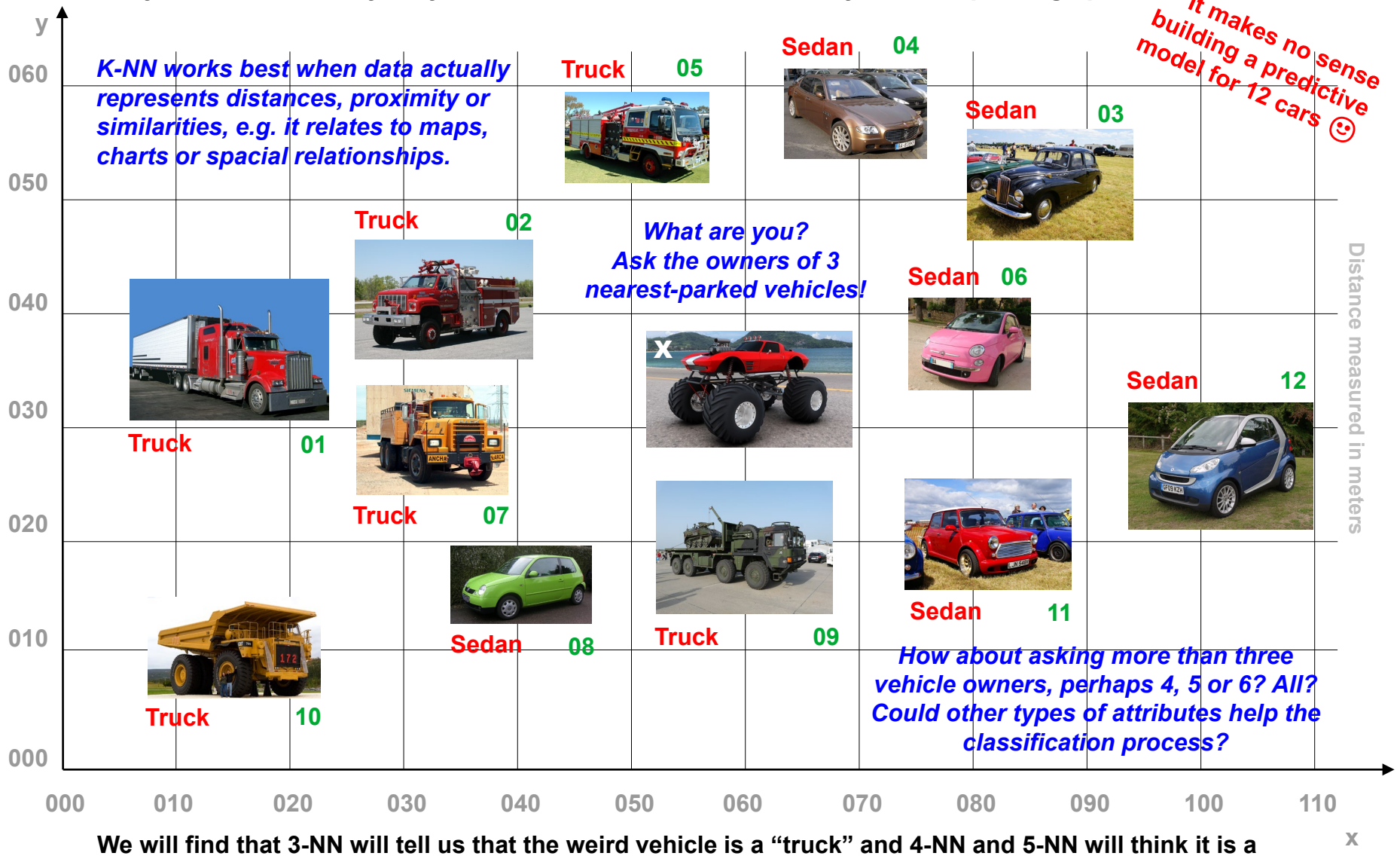
Also see KD 4.3 on measure of proximity

DEAKIN BUSINESS SCHOOL

Classifying Observations using k-NN Classifiers

Let us consider a tiny sample of 12 vehicles, some of which are trucks and some are sedans. In the far away O'Donalds car park, management want to find out which of the parked vehicles are truly trucks and which are truly sedans. Normally they ask the owners.

In the O'Donalds car park, owners are asked to stay inside, trucks are to be parked near trucks and sedans are to be parked near sedans – drivers not always obey the rules. Let us use k-NN to identify the vehicle types, when owners are not there to ask, based only on their parking spot.



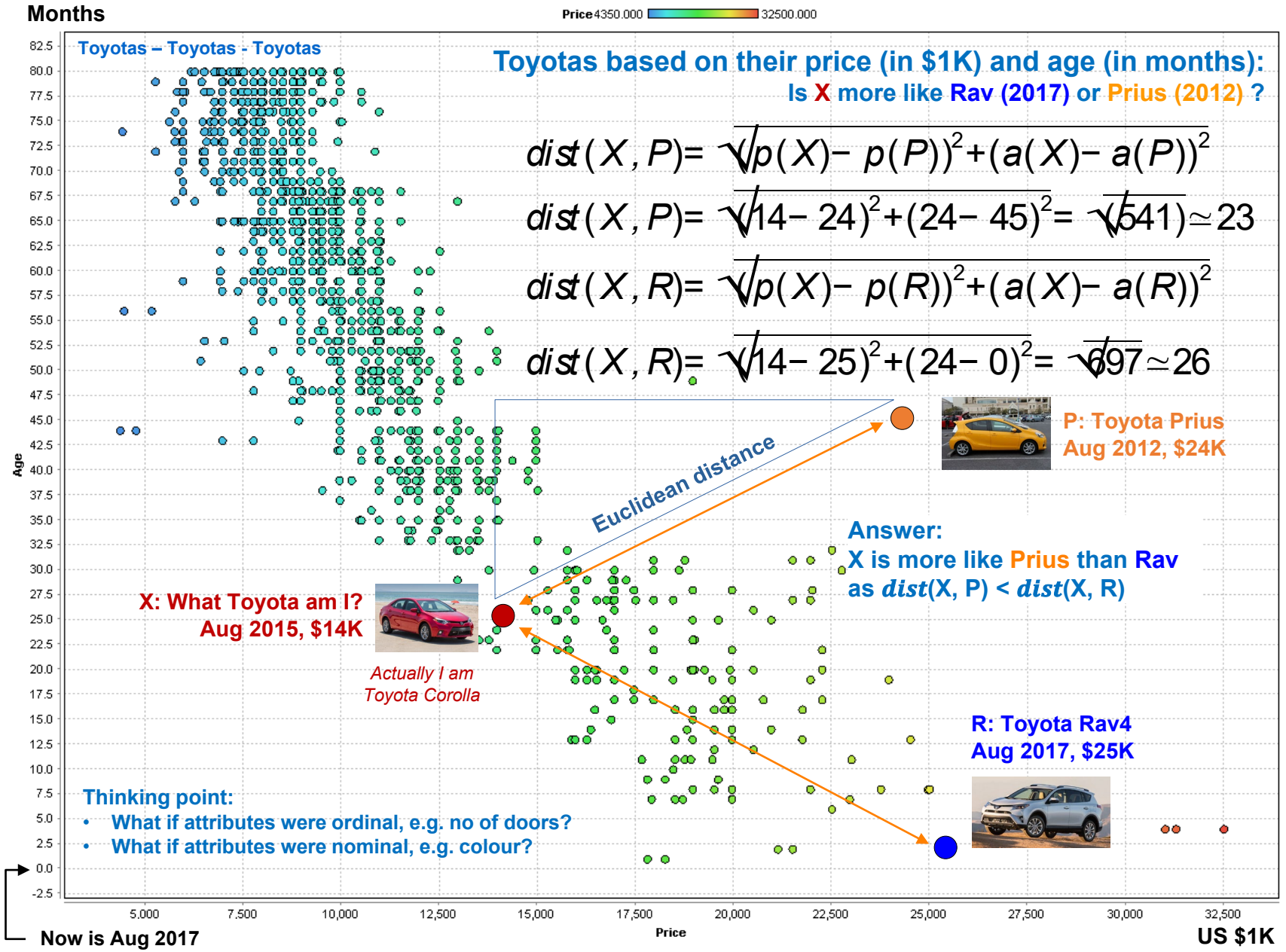
We will find that 3-NN will tell us that the weird vehicle is a "truck" and 4-NN and 5-NN will think it is a "sedan". 3-NN based on dummy encoding of nominal attributes will classify the vehicle as a "sedan".

DEAKIN BUSINESS SCHOOL

Lots more cars…

Euclidean distance

Let's use a different data set of 1437 Toyotas described by their age and price, parked in a secret underground tunnel, with no known geo-location and no owners to ask questions. Now k-NN may need to calculate car similarity in a different way, using *Euclidean distance*.



DEAKIN BUSINESS SCHOOL

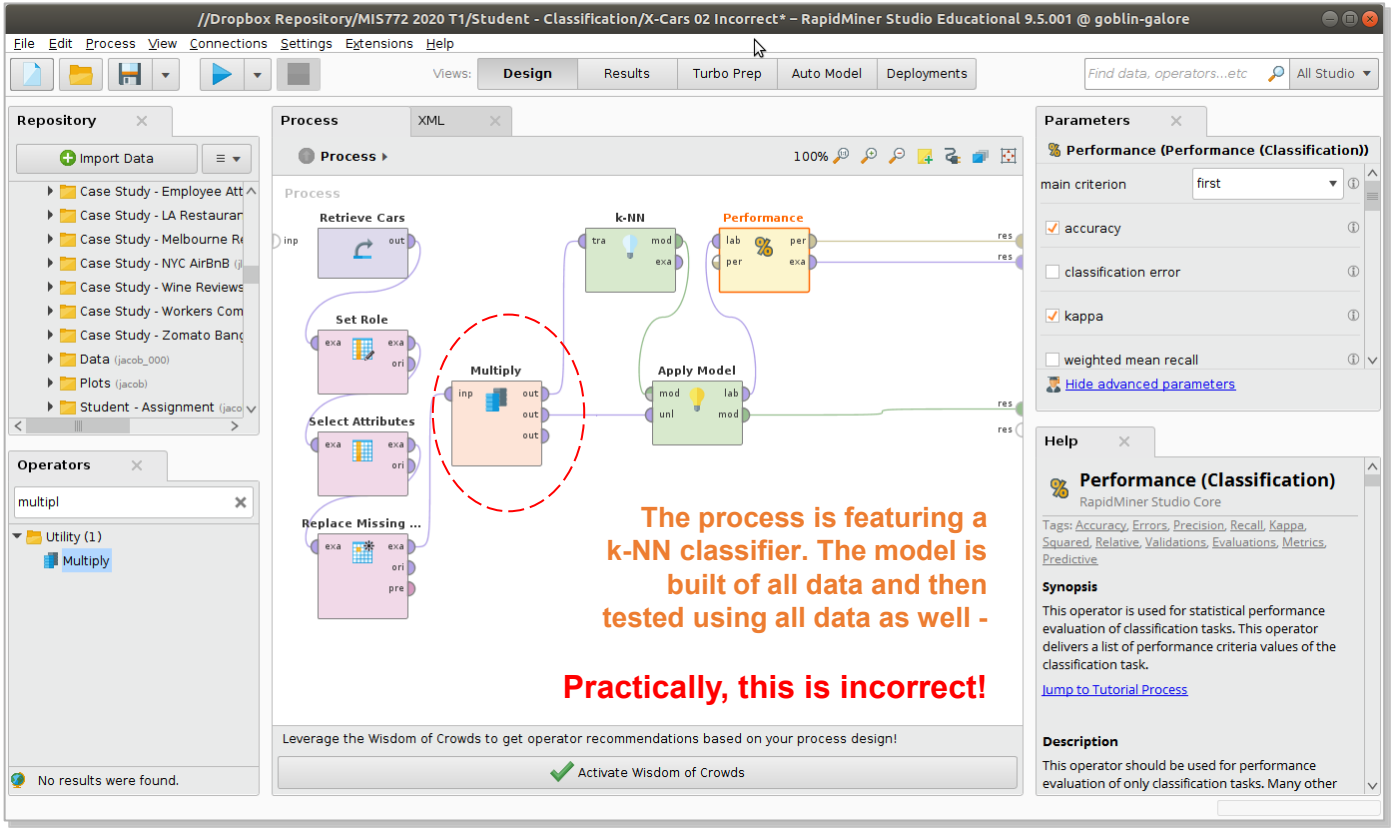
Measuring the Model Performance

How would we know how accurate the classifier would be when it is eventually deployed, i.e. it is applied to data to be collected in the future

Example: Data + Prep + 1st Approach to Processing

- Let's consider another car-related data set consisting of expert evaluations of cars before their purchase by the car yard.
- We would like to use this data to create a classifier to help out deciding whether or not a car is acceptable for purchase.
- We have 1728 of the past evaluations, unfortunately some have been badly entered and have missing attribute values.
- To assess the model future performance, we run an experiment by training and testing it using the available data.

Mentimeter

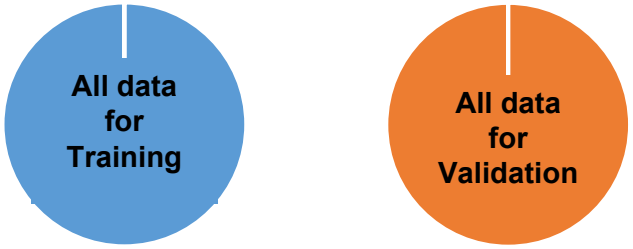


DEAKIN BUSINESS SCHOOL

Holdout Validation

About to be applied

Inexperienced data analysts use the same data for model training and then for model validation.
This tests only if the model was able to remember the examples it had seen before.



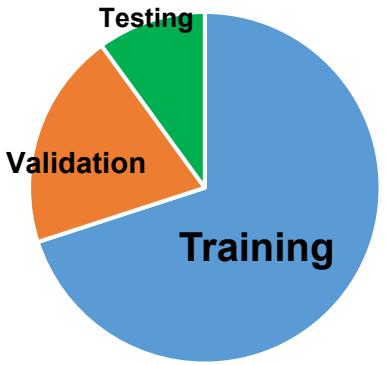
Such a test can be useful to check that the model is capable of learning (at all), but is not very useful to assess its performance on data yet to be collected.
Instead we aim for the model to learn *patterns* from a small sample available to us, so that later we could detect the same patterns in a new sample of data, never seen before.

So, a different way of developing a model is to randomly split data into three partitions - one to be used for model training, and two are *held-out* for validation and testing. We cannot pick our "random" split to make the model "better" – it is cheating!

The *model is developed iteratively* - at each step we apply the model to both training and validation data partitions, and we aim to *improve data pre-processing* and *tune the model parameters* (e.g. change "k").

If data partitioning is completely random at each step we cannot be sure if the performance changes are due to our actions or because of a distinct data split. So we "freeze" the split by setting a *random seed*. However, if the partitions do not change, then the model may over-fit the validation data and be biased!

To address this issue, we estimate the model performance not from the final validation run but from a separate test run on data never used in the cycle of improvements. We call it *honest testing*.



Unfortunately, when the sample is very small, we often skip the testing step.

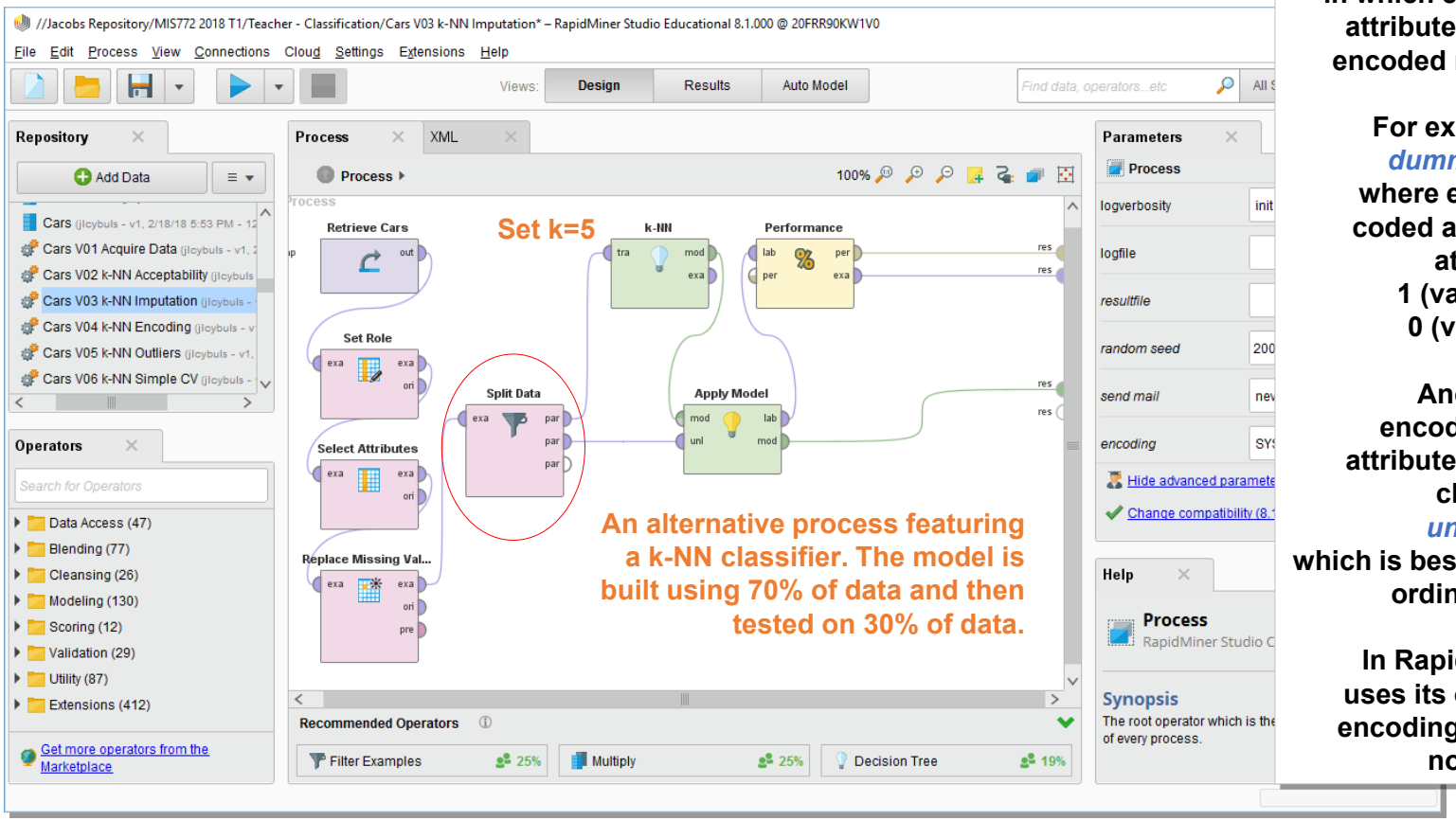
DEAKIN BUSINESS SCHOOL

Also see KD 2.3.1 on training and testing datasets

Example: Data + Prep + 2nd Approach to Processing

- The previous approach was incorrect!
- We will therefore change the process and instead of multiplying (duplicating) the available data for training and validation, we will split it for training (70%) and *holdout validation* (30%).
- If the volume of data allows, we would also create a test partition (e.g. 70-20-10%).
- To control the cycle of improvements, we can set the *random seed* (any number, use 2023) of the data split. This ensures that the split is random, but can be replicated each time.

- Many things may influence the model performance:
  - The model parameters (e.g. "k")
  - The choice of predictors
  - The size of data partitions used for training, validation and testing
  - The approach to dealing with missing values
  - The mix of class values while performing dummy encoding
  - All data pre-processing
  - Etc.



**** Encoding:**
Many models require only numeric attributes, in which case nominal attributes need to be encoded numerically!

For example using *dummy encoding* where each class is coded as a separate attribute with:
1 (value present)
0 (value absent)

Another way of encoding nominal attributes is for each class to use a *unique integer* which is best in cases of ordinal attributes

In RapidMiner k-NN uses its own internal encoding, which may not be optimal

DEAKIN BUSINESS SCHOOL

The Process

What we have: data from the past, e.g. car evaluations classified by car acceptability for purchase (Data with Label).

How are we going to do this: we will use training data to create a model capable of automatic car classification (Training).

ExampleSet (1210 examples, 1 special attribute, 6 regular attributes)

Row No.	Accepta...	Buying	Maint	Doors	Persons	Lug_Boot	Safety
1182	acc	low	low	4	4	small	med
1199	acc	low	low	5more	4	small	med
839	good	med	low	2			
840	good	med	low	2			
846	good	med	low	2	more	med	high

Training – Create a predictive model able to classify existing data

ExampleSet (518 examples, 6 special attributes, 6 regular attributes)

Row No.	Acceptability	prediction(Acceptability)	confidence(unacc)	confidence(acc)	confidence(vgood)	confidence(good)	Buyi
127	acc	acc	0.400	0.600	0	0	vhigl
128	acc			0.600			vhigl
129	unacc			0.100			vhigl
130	unacc			0.300	0	0	vhigl
131	acc	acc	0.400	0.500	0	0.100	vhigl

Validation – Classify existing cases and compare against known classification

Confidence factors actually define classification

How would we know if it worked: we will use the validation data not used in training to test accuracy of predictions (Validation).

Honest testing is advisable!

ExampleSet (1728 examples, 0 special attributes, 6 regular attributes)

Row No.	Buying	Maint	Doors	Persons	Lug_Boot	Safety
646	high	high	5more	more	big	low
64			5more	more	big	med
64			5more	more	big	high
64			2	2	small	low
650	high	med	2	2	small	med

New Cases – Collect new unclassified data, i.e. with missing information

How about future cases: we will collect information about new cars with unknown classification.

What we want:
we will apply the validated model to classify all new cars, i.e. we will predict their acceptability for purchase (Application/Deployment).

ExampleSet (172 examples, 5 special attributes, 6 regular attributes)

Row No.	prediction...	confi...	confi...	confi...	confi...	Buying	Maint	Doors	Persons	Lug_Boot	Safety
4	unacc	1.000	0	0	0	vhigh	vhigh	3	2	med	low
5	unacc	1.000	0	0	0	vhigh	vhigh	3	more	big	low
6	unacc					vhigh	vhigh	3	more	big	high
7	unacc					vhigh	vhigh	4	4	small	med
8	unacc					vhigh	vhigh	4	4	med	low

Application - Use the predictive model to classify all new cases

DEAKIN BUSINESS SCHOOL

Measuring Performance
Confusion Matrix

How can we measure the classifier performance?

- We apply the classifier to the past examples from the validation data set, where the label values are known.
- We then collect all results and note their *predicted* classifications.
- We compare them with their known *true* classifications.
- We create a *confusion matrix*, which shows rows and columns, of predictions and true classifications.
- We will then collect all classification pairs, i.e. the predicted vs true class value and enter them in their "cell" by adding 1 each time (a simple count).
- At the end, we will have a matrix summarising all possible outcomes of running the classifier on the validation data set.

- All correctly classified examples will be accounted for on a diagonal.
- All incorrectly classified examples will be accounted for elsewhere.
- The classifier *accuracy* is calculated as the number of the examples correctly classified (on a diagonal) over the total number of all examples (in the matrix).
- The *misclassification rate* is often the preferred indicator of classifier performance, which is calculated as:

  misc = 1 – accuracy.
- The totals of each column represent how well the classifier *recalled* the true classifications it was trained on (for each class).
- The totals of each row represent how *precise* are its predictions (for each class).

Correct classification

Misclassification

accuracy: 88.13%

Accuracy Performance

	true unacc	true acc	class precision
pred. unacc	358	59	85.85%
pred. acc	2	95	97.94%
class recall	99.44%	61.69%	

DEAKIN BUSINESS SCHOOL

Also see KD 8.1 on confusion matrix and performance measurements

Classification:
Decision Tree (DT)

DEAKIN
BUSINESS
SCHOOL

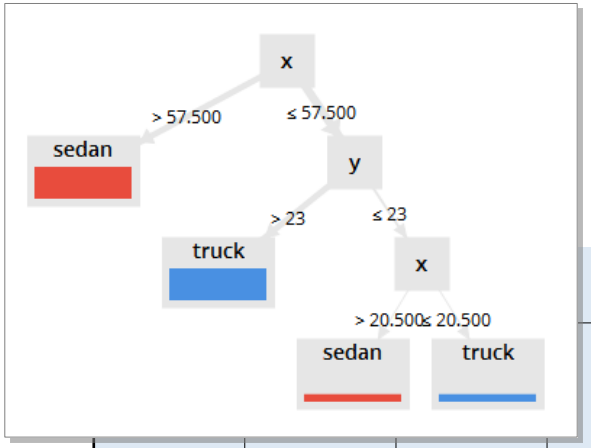
Decision Trees

Decision tree is another type of classifier

- Decision trees represent a hierarchy of attribute tests to arrive at the best classification
- They can be used for both classification and estimation
- The label can be nominal or numeric (regression trees)
- They can deal with missing values
- Decision tree structure can generate (business) *decision rules*
- Decision trees may not be optimal and require pruning, i.e. we cut out the lowest branches to improve validation
- Decision rules are hierarchical, as an upside-down tree-like structure with *nodes* connected by *links*
- Nodes represent decision rules and the links order the rules
- The top node is called a *root*, the bottom nodes with only one connection are called *leaves* and the middle nodes are called *interior nodes*

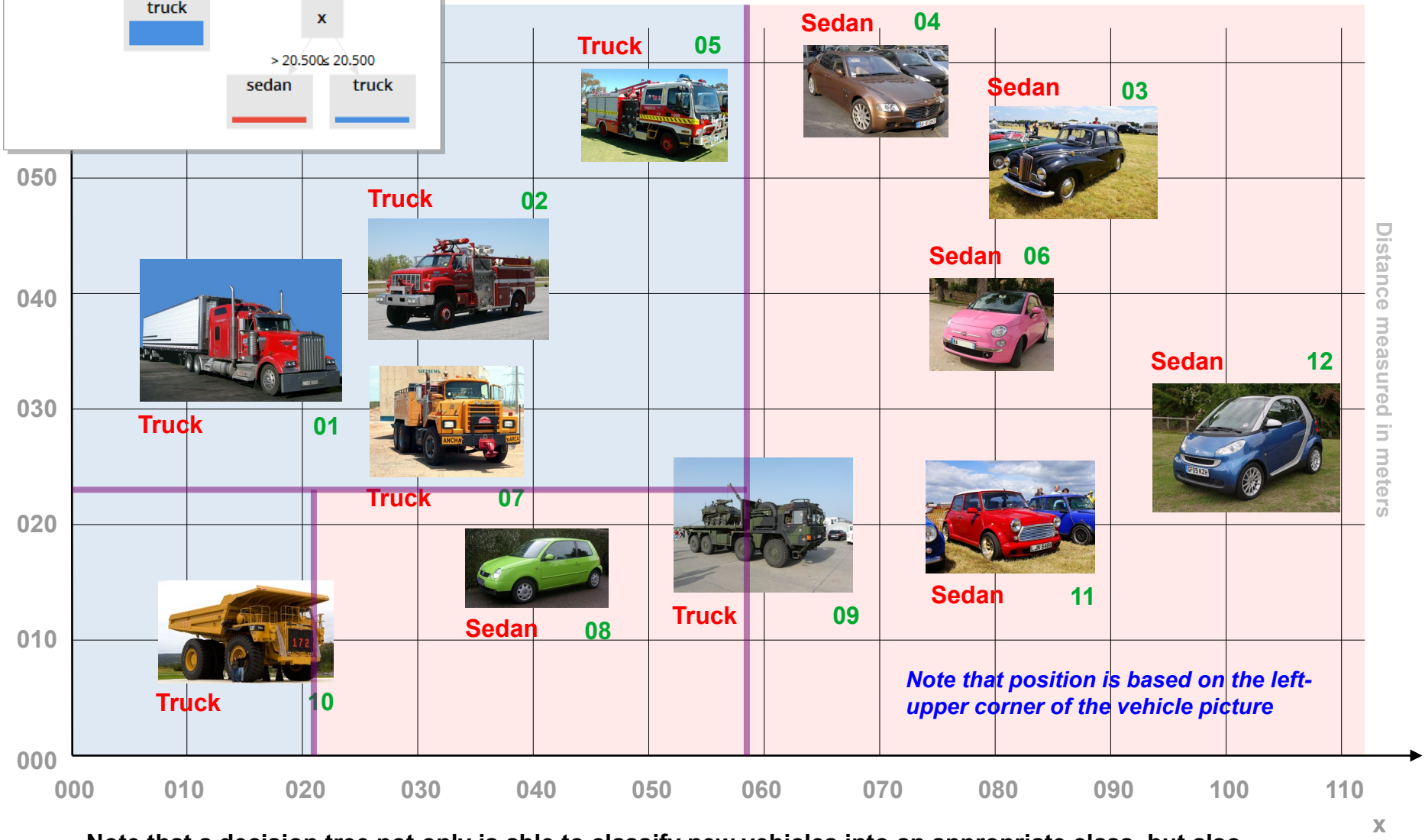
Also see KD 4.1 on Decision Trees

Classifying Observations using DT Classifiers



attribute	weight
x	0.844
y	0.156

Let us apply a decision tree to the classification of vehicles based on their x, y coordinates. Observe the weights of attributes and the way decisions partition data into sub-sets, based on the "purity" of outcome.



Note that a decision tree not only is able to classify new vehicles into an appropriate class, but also illustrates how the data space is recursively sub-divided into subsets of increasingly more precise and cohesive groups of data points.

DEAKIN BUSINESS SCHOOL

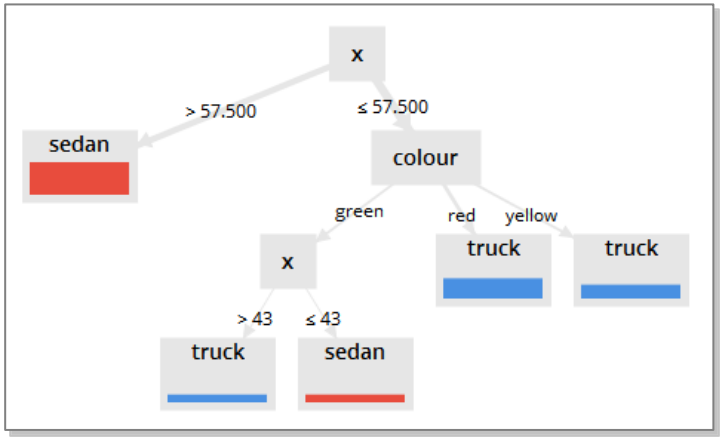
Classifying Observations using DT Classifiers (cont'd)

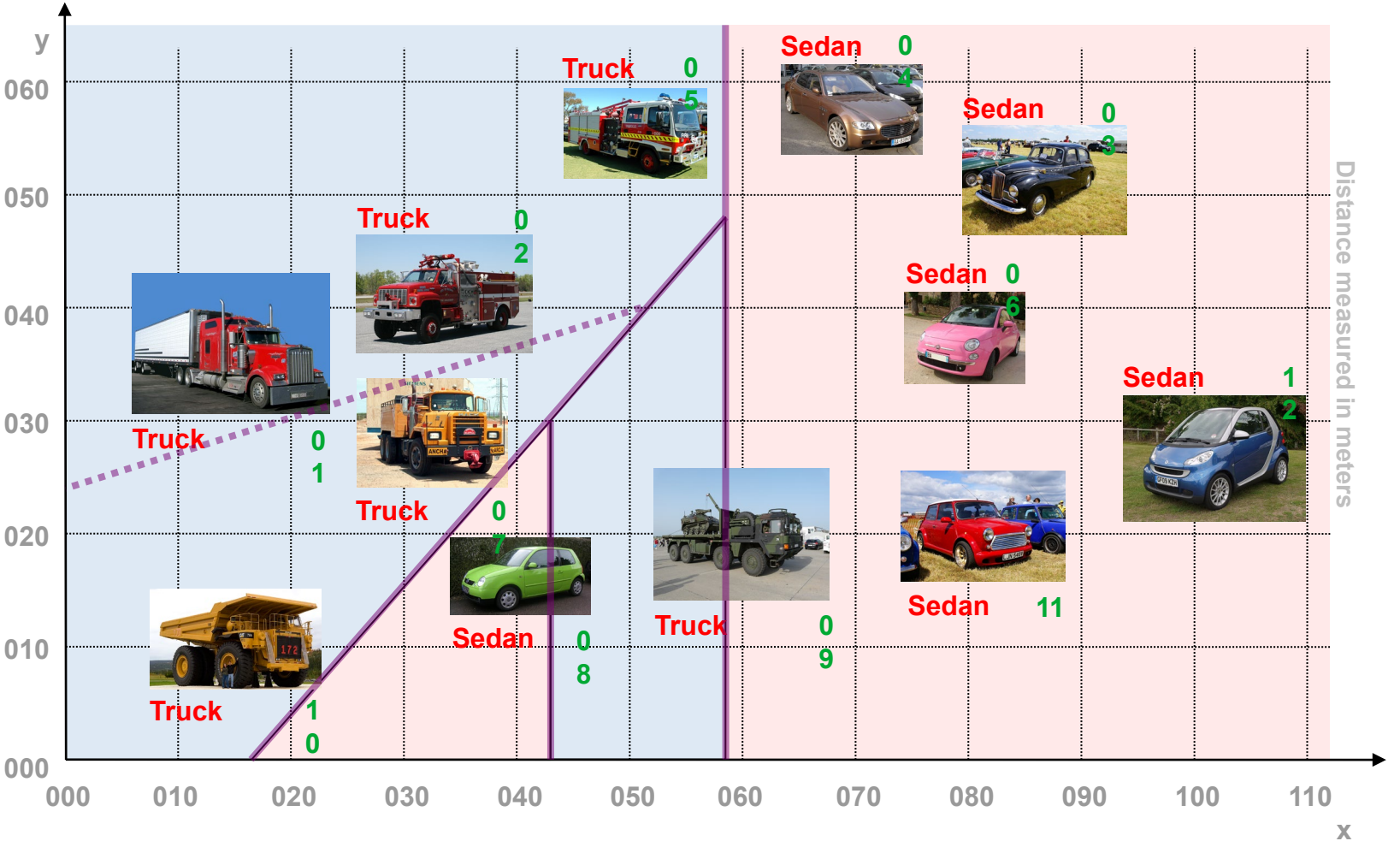
Decision space seems to be more complex when adding more dimensions (e.g., an additional nominal attribute **colour**)

The "**colour**" attribute was determined more important than **"y" attribute** and so it was used in the second level decision making.

The **"y" attribute** was completely eliminated from the decision chain.

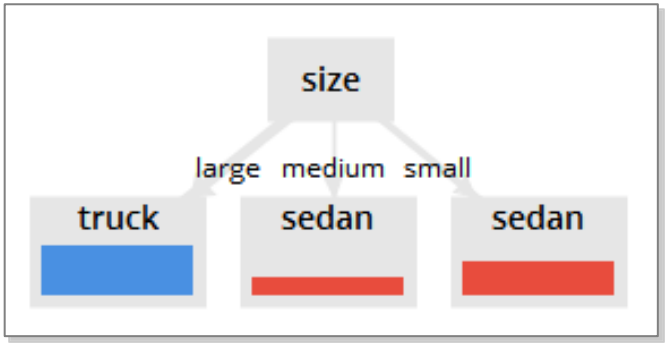
attribute	weight
colour	0.156
x	0.844





Classifying Observations using DT Classifiers (cont'd)

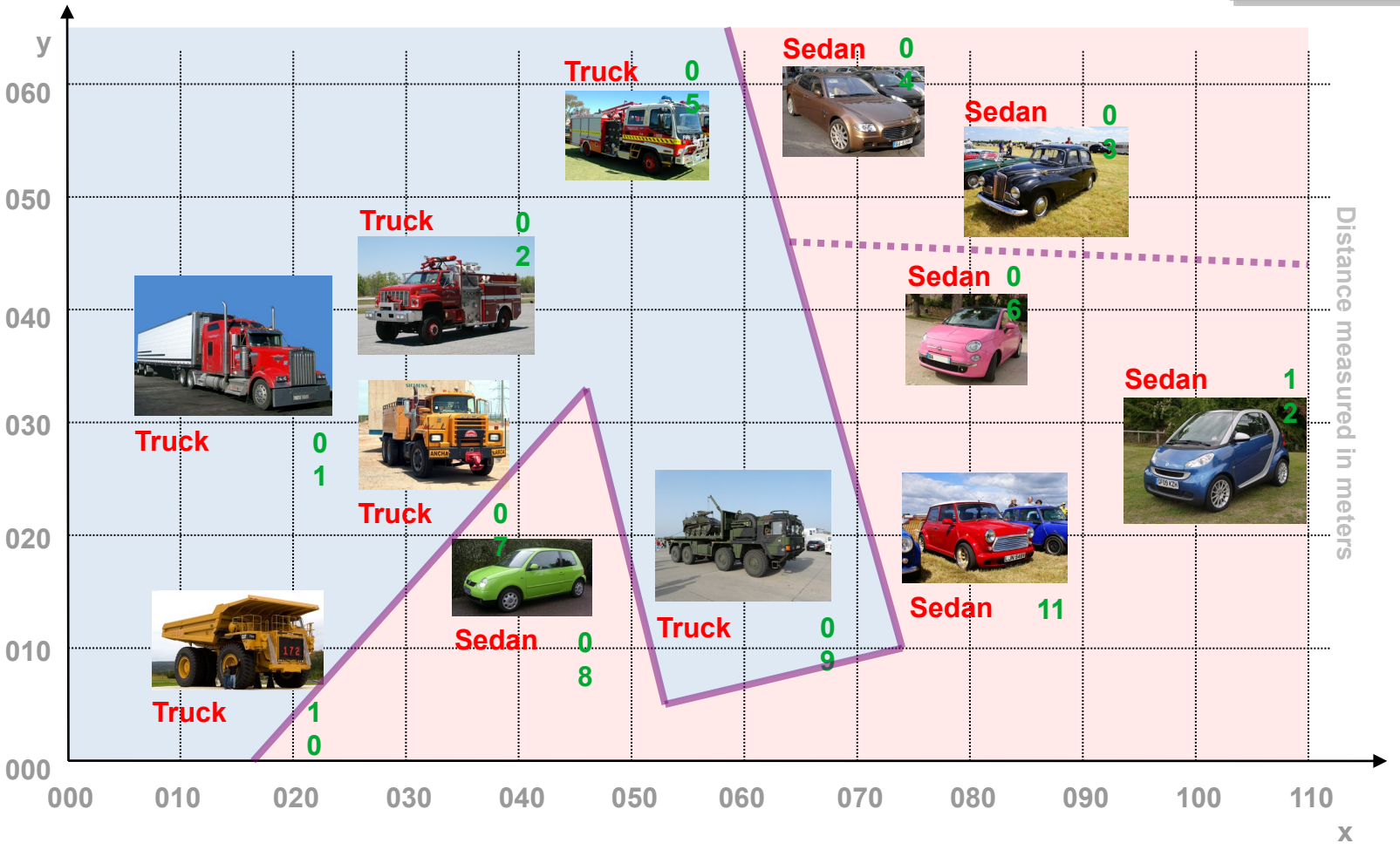
Let us apply a decision tree to the classification of vehicles based on **all available attributes including size**. Observe that only one attribute is now considered of critical importance to building a decision tree.



Adding more attributes does not necessarily enhance the model. It is the quality of the attributes and their importance to the decision making process that matters.

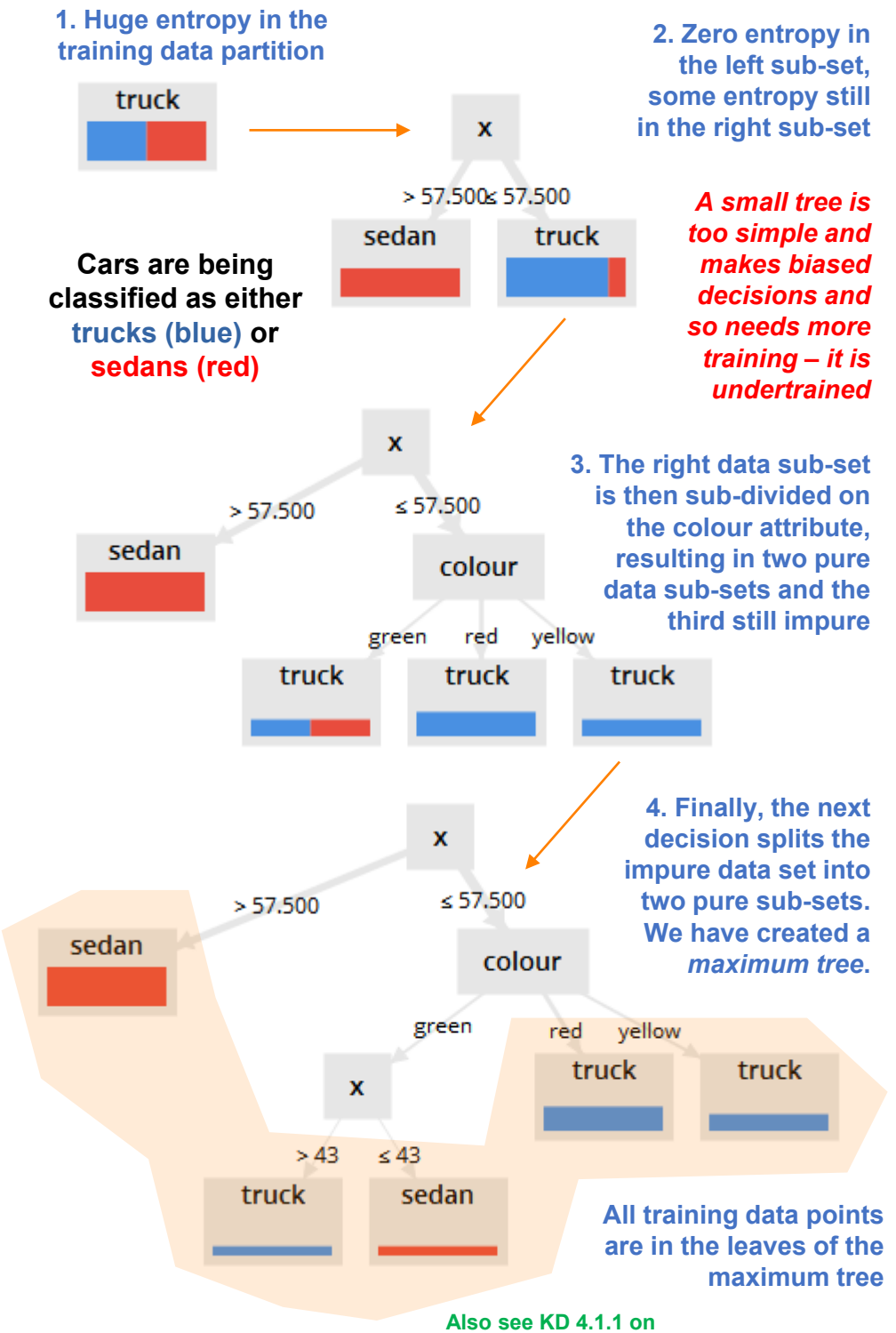
attribute	weight
size	1

Here a very simple decision tree of a **single attribute "size"** is capable of making quality decisions (and intuitively is right).



Building decision trees

- Trees can be developed using various algorithms.
- One of the most common relies on the repeated division of training data into sub-sets based on the attribute values, to ensure the greatest *purity* of the resulting sub-sets.
- *Impurity = misclassification!*
- The notion of purity is linked to the mathematical concept of *entropy* - the amount of noise in data, i.e. a set consisting of identical data points is pure and has entropy of zero, a set with equal mix of two kinds of data points is most impure, entropy of 1.
- At each step, the algorithm selects an attribute and its value (numeric attrs), resulting in a decision that splits the *parent* data set into *children* sub-sets (tree branches) that are more pure than the parent.
- This is done until all tree *leaves* end up with very small subsets (possibly of one example), which are most pure. We call this a *maximum tree*.

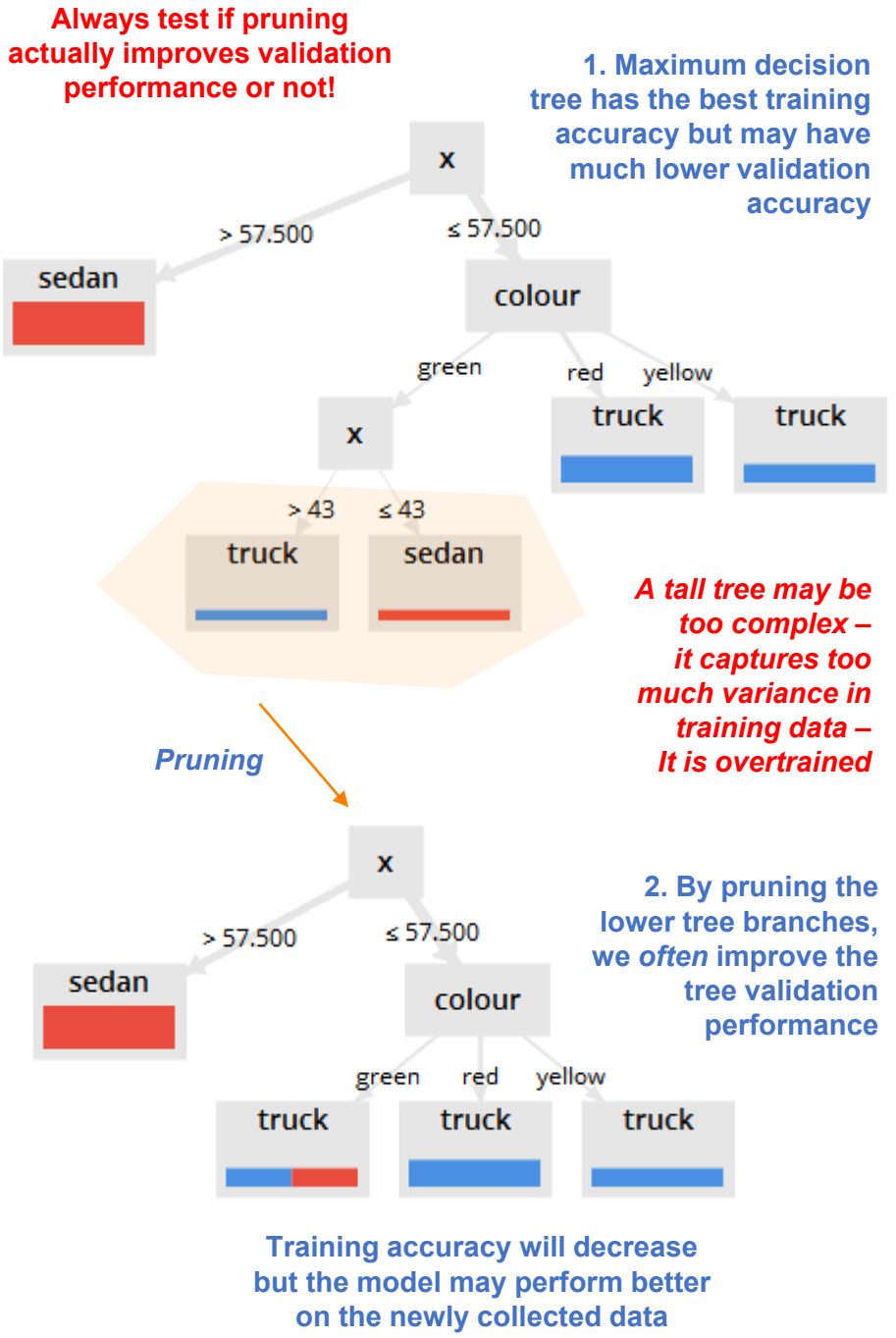


Also see KD 4.1.1 on DT's how it works

DEAKIN BUSINESS SCHOOL

Tree pruning

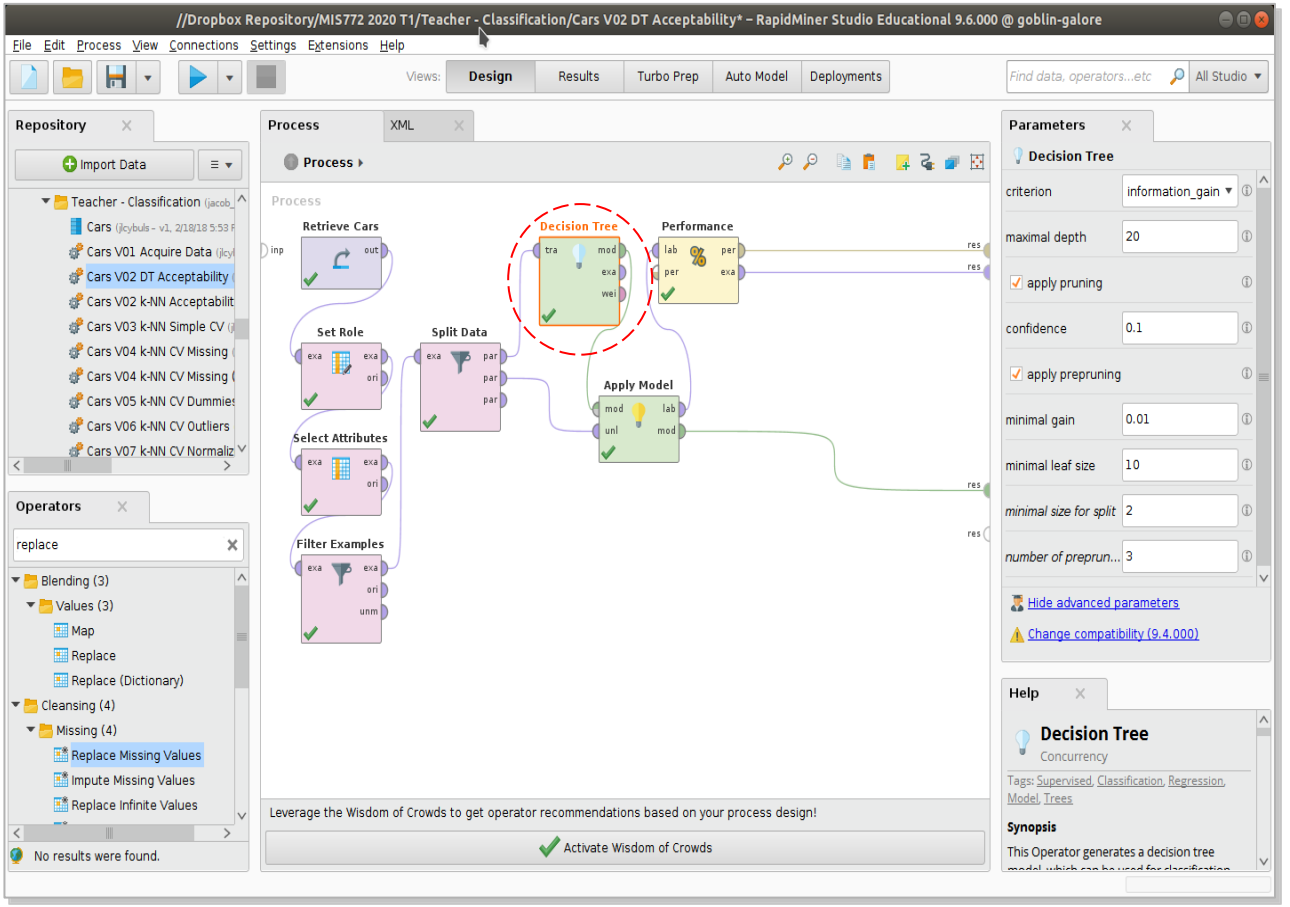
- The maximum tree will give the highest classification accuracy.
- We cannot assure 100% accuracy as some examples may not be fully distinguishable by the selected attributes.
- Furthermore, this accuracy was achieved on a training data partition. The maximum tree is often *overtrained*, i.e. its highest nodes represent large numbers of examples, a good sample of the entire population. However, the lowest branches (and leaves) represent very few examples, and so may be too specific for the training data sample.
- Commonly when we apply the decision tree to the validation data partition, the data points will distribute differently in the tree leaves than training data points, ending up with impure data sets, and lower validation accuracy.
- By *pruning*, or cutting out, some of the lowest tree branches we usually improve the validation accuracy. We do so layer by layer until validation accuracy no longer increases.



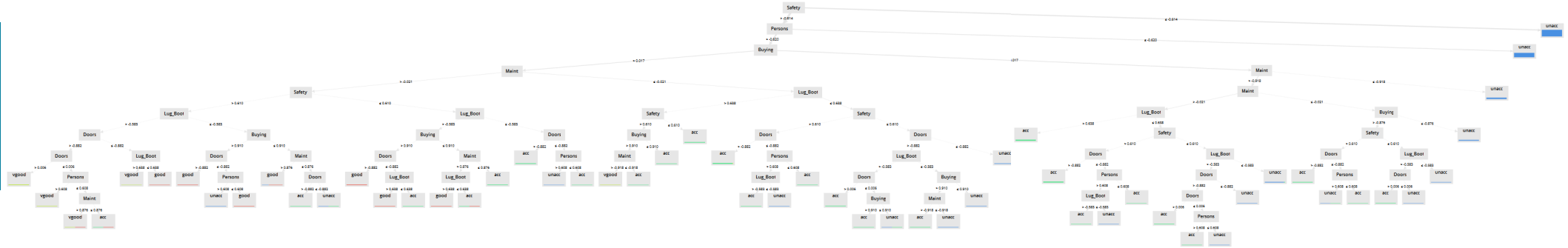
DEAKIN BUSINESS SCHOOL

Example: Decision Tree

- Once you developed a **classification process** it is reusable and adaptable – you can easily replace one model (such as k-NN) with another (such as Decision Tree).
- However, you need to be aware what are each of the model requirements and adjust data pre-processing.
- Run and inspect the results! Experiment with the model parameters – watch the performance! Compare against previously tested models.
- DTs characteristics:
  - DTs allow both numeric and nominal attributes
  - DTs do not worry about missing values in predictors
  - DTs hate missing values in labels



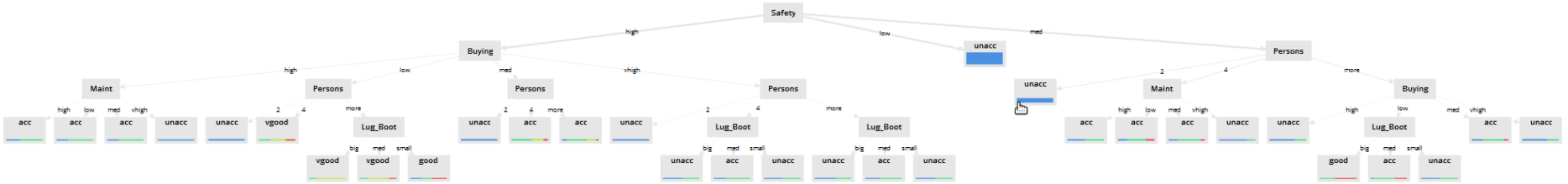
DEAKIN BUSINESS SCHOOL



Performance vs Interpretability

- Unfortunately the generated tree was huge and could not be easily interpreted to justify decisions!
- The model above gives 96.53% accuracy, however, its decisions can be explained with great difficulty.
- The model below is easier to interpret and justify its decisions, however its accuracy is only 79.5%.
- Which would you select?
- In this case we face a common dilemma of performance vs interpretability.

- In some applications, performance determines the model choices – usually performance translates into dollars and cents!
- In other applications, ability to interpret the model to justify its decisions is more important, even if these decisions are sub-optimal.
- The latter choices are often dictated by legal requirements in banks and insurance companies where decisions may have to be defended in court.



DEAKIN BUSINESS SCHOOL

Summary / Review

- What is a class?
  What is classification?
  What is a classifier?
- What are typical applications of classification?
- How do similarity models work?
- Describe the k-NN algorithm.
- Why is k-NN called “lazy”?
- Name 3 distance metrics often used with numeric / nominal attributes?
- Is it possible to measure distance between data points of mixed attributes? How?
- Explain the Euclidean distance.
- What should be done to reduce impact of attribute units on distance measurement?
- What data pre-processing is often recommended for k-NN training?
- Why using the same data for model training and validation is a bad practice? Explain.
- What is holdout validation?
- How can you measure classifier performance?
- Explain how to interpret confusion matrix. What is accuracy, recall and precision?
- Explain what is the decision tree root, parents, children and leaves?
- What is data purity/entropy?
  Why is it important?
- Why trees may need to be pruned?
- What happens when a model is undertrained / overtrained?
- Explain information leakage in model training, validation and testing.
- Explore other classifiers such as Logistic Regression, Naïve Bayes, Support Vector Machines (SVM).