Map Business Problems to Machine Learning

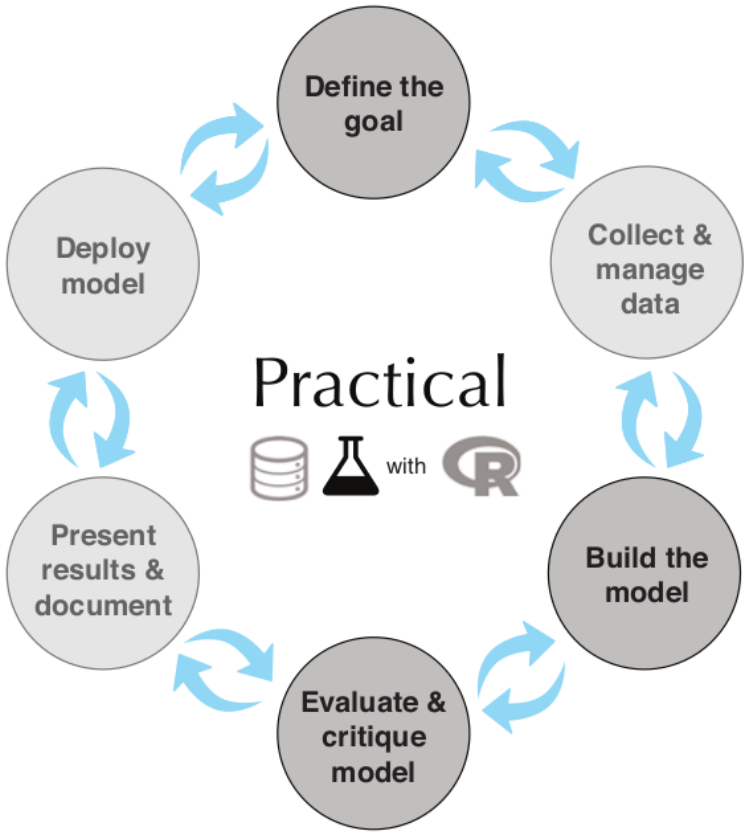
CITS4009 Computational Data Analysis

Dr. Mubashar Hassan

Department of Computer Science and Software Engineering
The University of Western Australia

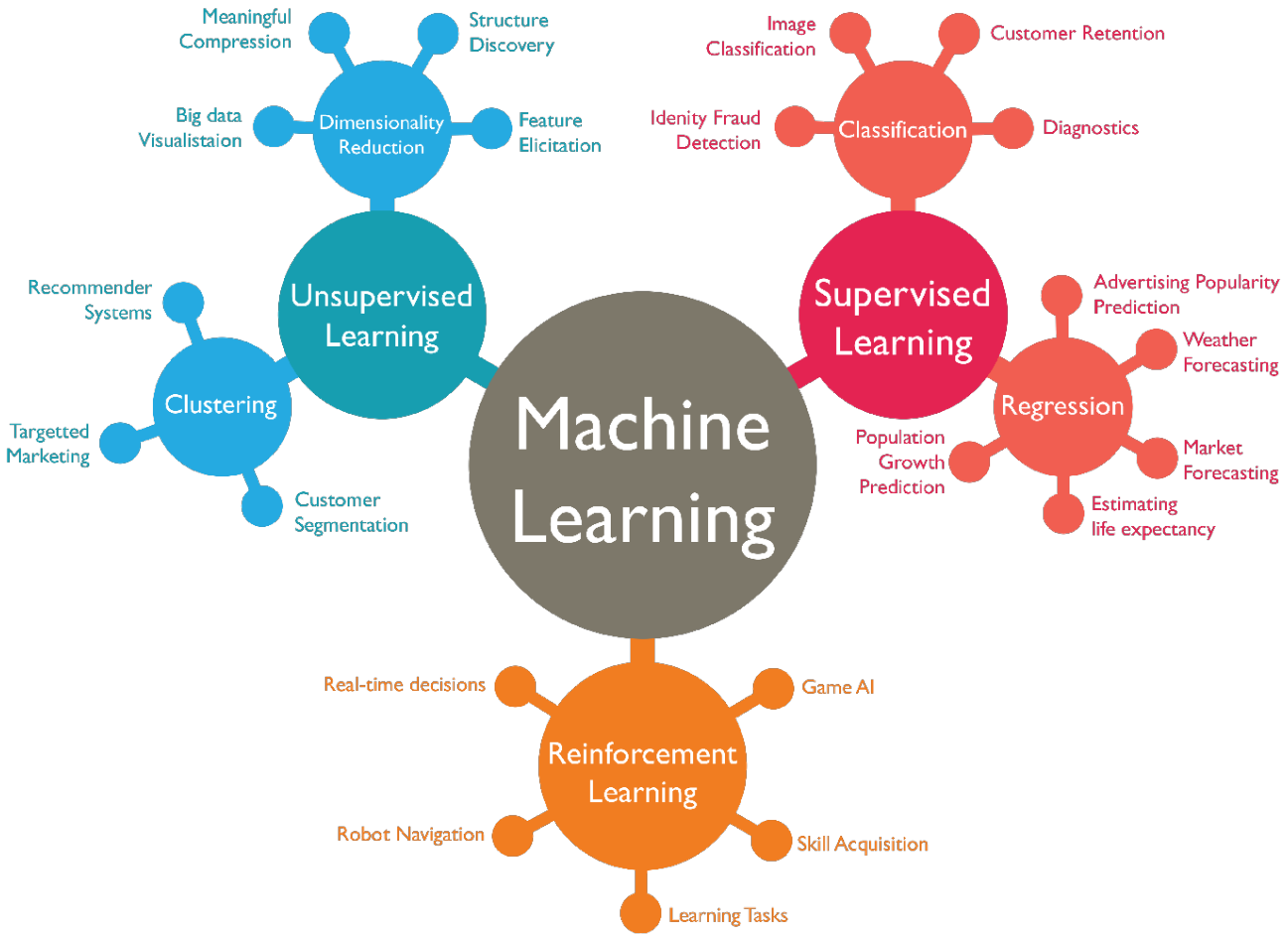
Semester 2, 2024

The modelling process in the lifecycle of a DS project



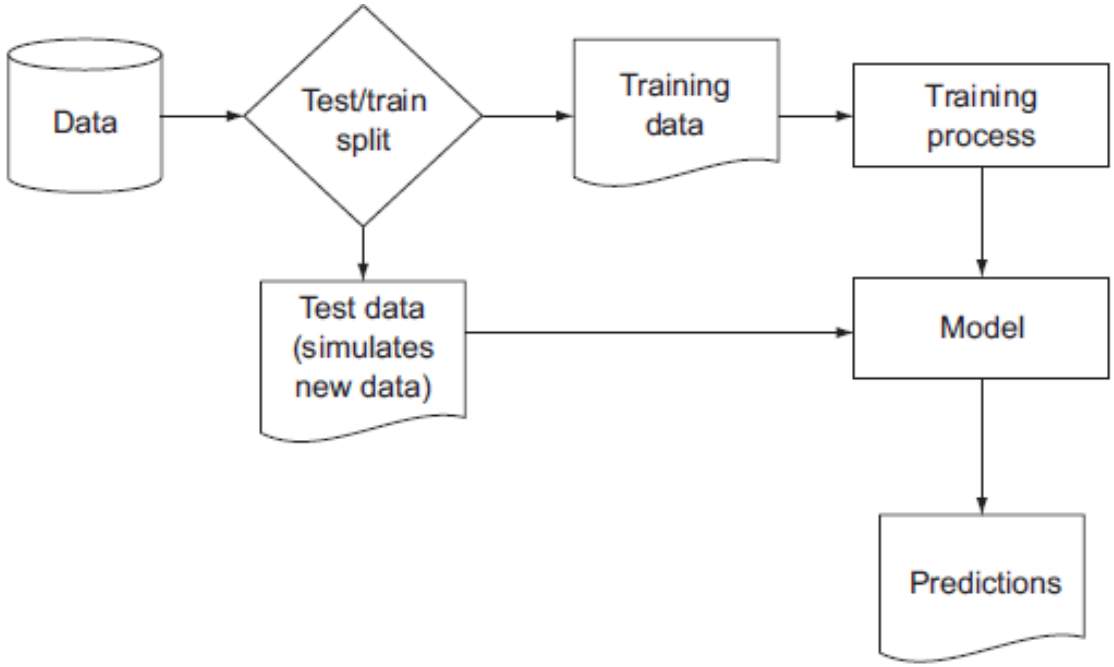
Practical Data Science with R, *Nina Zumel, John Mount*, Manning, 2nd Ed., 2020 (Chapter 6)

Machine learning - the core of data science



https://towardsdatascience.com/machine-learning-types-2-c1291d4f04b1?gi=102f83adbb40

Objectives

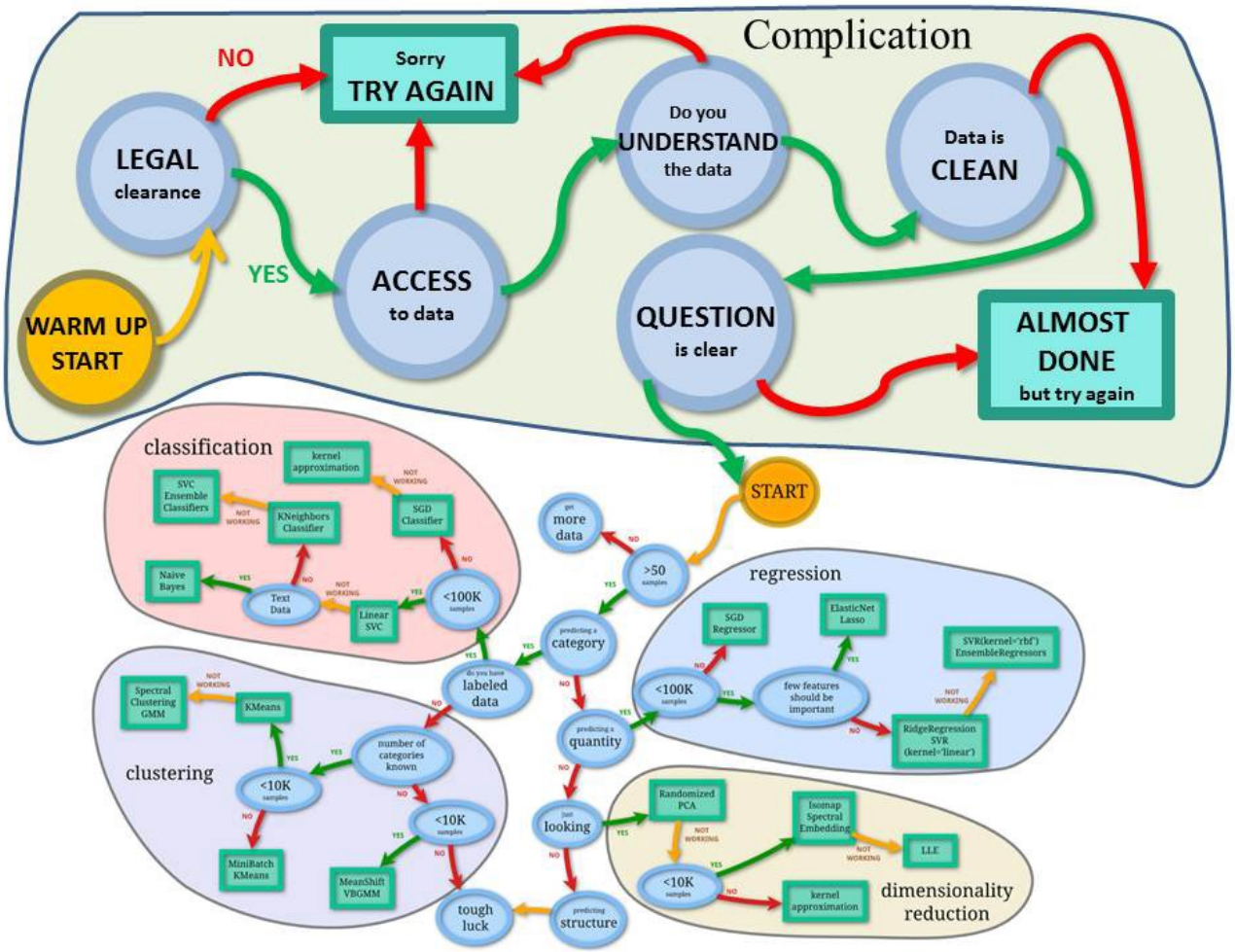


- Mapping business problems to machine learning tasks
- Evaluating model quality
- Validating model soundness

Example of problems in an online retail company:

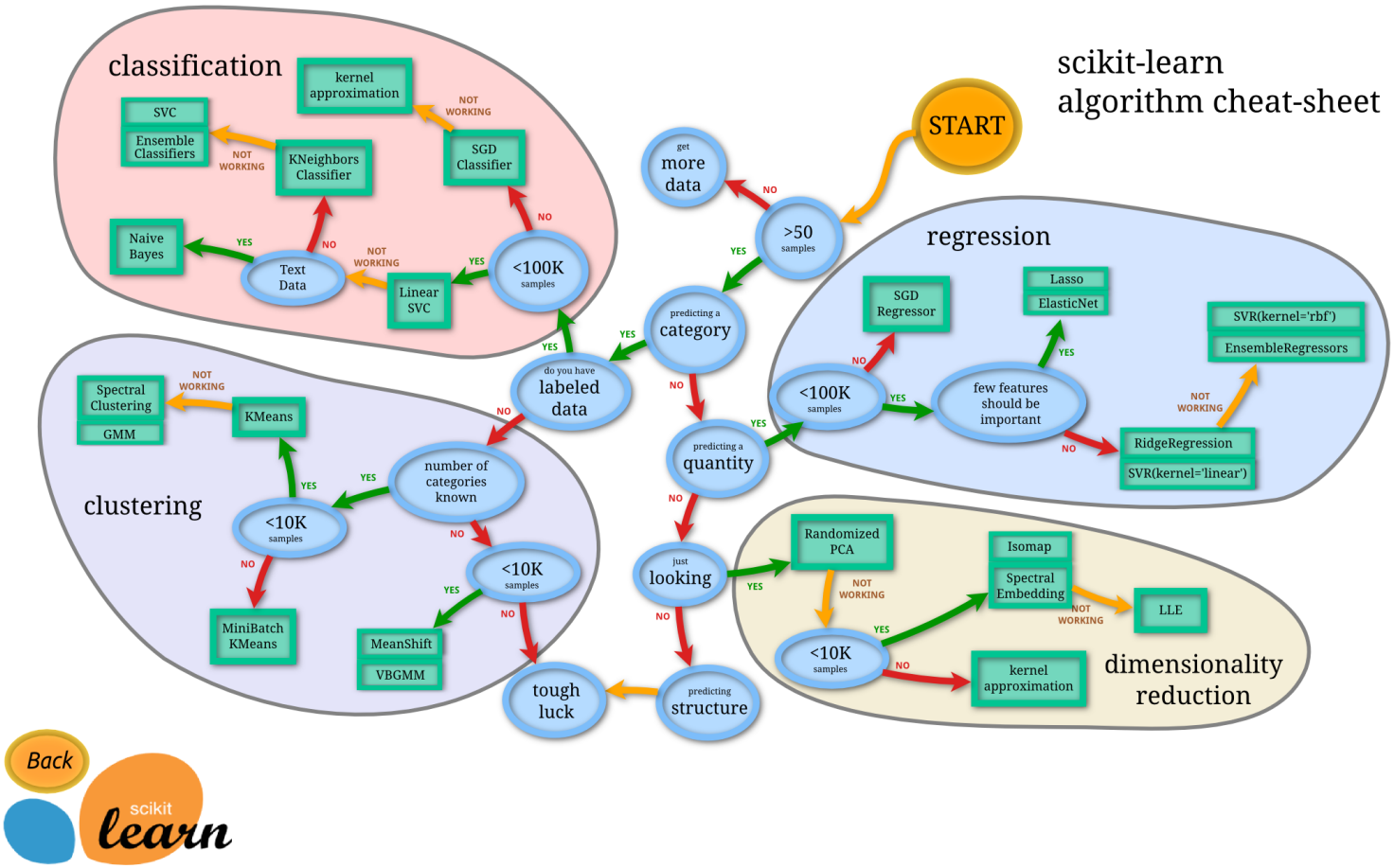
- Predicting what customers might buy, based on past transactions
- Identifying fraudulent transactions
- Determining price elasticity (the rate at which a price increase will decrease sales, and vice versa) of various products or product classes
- Determining the best way to present product listings when a customer searches for an item
- Customer segmentation: grouping customers with similar purchasing behavior
- AdWord valuation: how much the company should spend to buy certain AdWords on search engines
- Evaluating marketing campaigns

What methods should we use?



https://medium.com/@chris_bour/an-extended-version-of-the-scikit-learn-cheat-sheet-5f46efc6cbb

What methods should we use?



http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

References

- **Practical Data Science with R**, *Nina Zumel, John Mount*, Manning, 2nd Ed., 2020 (Chapter 6)
- 5 most useful differences between data science and machine learning: https://www.educba.com/data-science-vs-machine-learning/

An Overview on Classification

CITS4009 Computational Data Analysis

Dr. Mubashar Hassan

Department of Computer Science and Software Engineering
The University of Western Australia

Semester 2, 2024

Classification - automatically categorise products

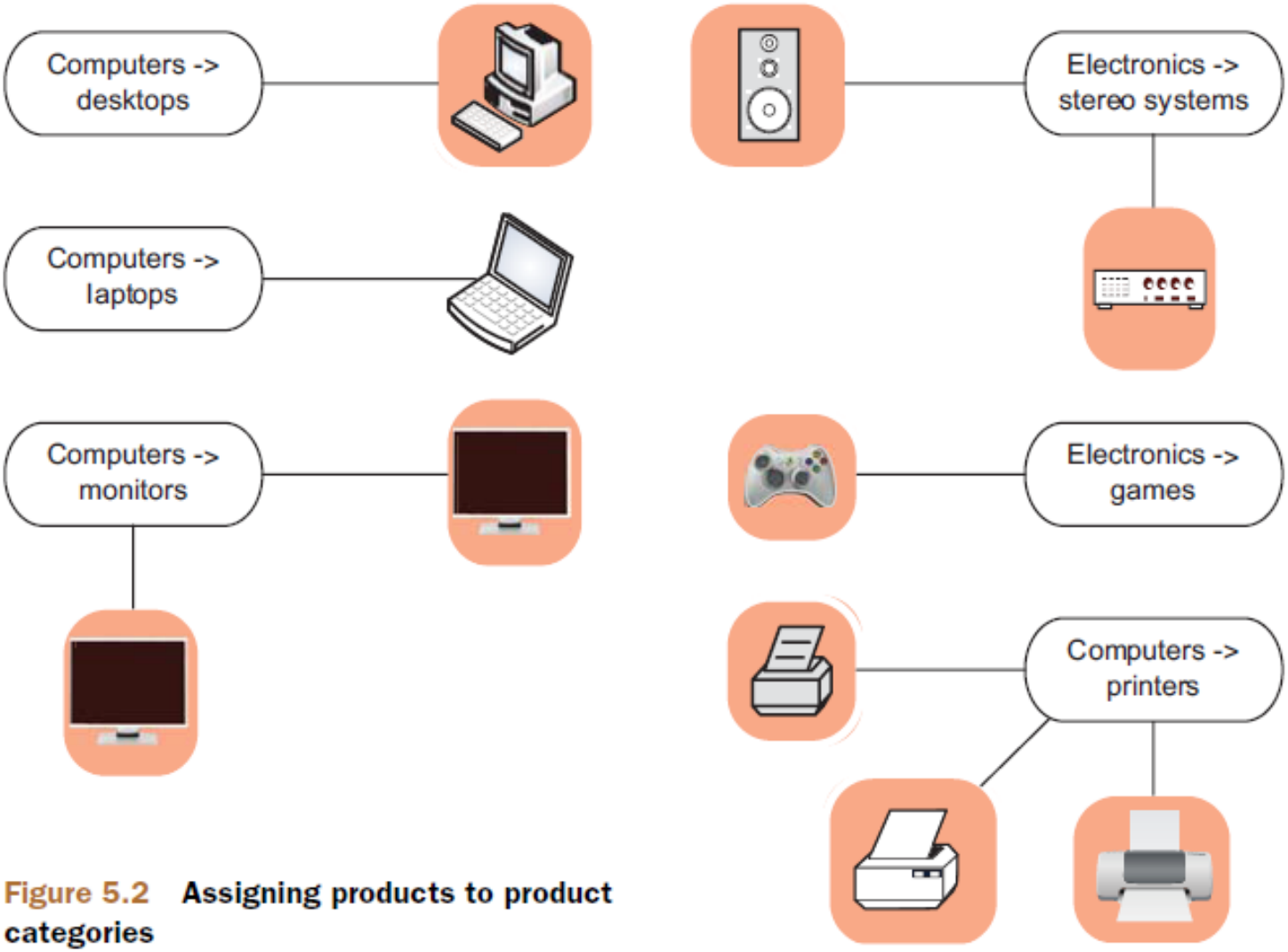


Figure 5.2 **Assigning products to product categories**

What is classification?

Learn from data to decide how to assign labels to an object. The response variable is categorical.

Also known as supervised learning, because we need labelled training data.

What information do we need for classification?

Feature collection

- For example, collect product attributes and/or text descriptions of known products

Build a training set

- Building training data is the major expense for most classification tasks, especially text-related ones.
- Multi-category vs two category classification
  - Most classification algorithms are specialized for two-category, binary, or binomial, classification
  - Use "one versus rest'' classifier for multi-class, better with dedicated multi-class classifiers.

Common classification methods: Naïve Bayes

- Especially useful for problems with many input variables, categorical input variables with a very large number of possible values, and text classification.
- **Naïve Bayes**: Bayes would be a good first attempt at solving the product categorization problem.

Common classification methods: Naïve Bayes (An example)

Let's say we have 300 birds and 300 dogs. From these animals, we obtain the frequencies for the features indicated by the column names below:

	can.swim	is.brown
Bird	74	57
Dog	250	62

(The numbers in the table are the frequencies of birds and dogs that can swim and that are brown, e.g., 74 out of the 300 birds can swim and the remaining 226 can't swim.)

Suppose that now we have an unknown animal having the following features: `can.swim=TRUE` and `is.brown=TRUE`. What is the probability that it is a bird? or a dog?

Naïve Bayes (An example) (cont.)

Naïve Bayes follows the Bayes rule, but it makes the "naïve" assumption that the input variables are independent:

$$P(\text{Bird} \mid \text{can.swim \& is.brown}) = \frac{P(\text{Bird \& can.swim \& is.brown})}{P(\text{can.swim \& is.brown})}$$

$$= \frac{P(\text{can.swim \& is.brown} \mid \text{Bird})\, P(\text{Bird})}{\sum_{\text{animal}=\text{Bird,Dog}} P(\text{can.swim \& is.brown} \mid \text{animal})\, P(\text{animal})}$$

$$= \frac{P(\text{can.swim} \mid \text{Bird})\, P(\text{is.brown} \mid \text{Bird})\, P(\text{Bird})}{\sum_{\text{animal}=\text{Bird,Dog}} P(\text{can.swim \& is.brown} \mid \text{animal})\, P(\text{animal})}$$

The independence assumption means
$P(\text{can.swim \& is.brown} \mid \text{bird}) = P(\text{can.swim} \mid \text{bird})\, P(\text{is.brown} \mid \text{bird})$

The denominator term is the same for dog. So we only need to focus on the numerator term.

Naïve Bayes (An example) (cont.)

$$P(\text{can.swim \& is.brown} \mid \text{Bird})\, P(\text{Bird}) = \frac{74}{300} \times \frac{57}{300} \times \frac{1}{2} = 0.02343333$$

Similarly,

$$P(\text{can.swim \& is.brown} \mid \text{Dog})\, P(\text{Dog}) = \frac{250}{300} \times \frac{62}{300} \times \frac{1}{2} = 0.08611111$$

The denominator is just the total of the two numbers: 0.1095444

Thus, given that the unknown animal can swim and is brown, the probability

- that it is a bird is $0.02343333 \;/\; 0.1095444 = 0.2139163$
- that it is a dog is $0.08611111 \;/\; 0.1095444 = 0.7860841$

Since dog has the highest probability, the predicted animal is dog.

Naïve Bayes classifier

Naïve Bayes works well on problems that have categorical variables.

The `naivebayes` library provides a `naive_bayes` function for training a Naïve Bayes classifier. For example, if our training set `df` (a data frame) is (only the first 6 rows are shown):

can.swim	is.brown	label
FALSE	FALSE	Bird
TRUE	FALSE	Dog
TRUE	FALSE	Dog
FALSE	TRUE	Bird
FALSE	FALSE	Bird
TRUE	FALSE	Bird

then training a Naïve Bayes classifier on `df` is simply:

```
library(naivebayes)
model <- naive_bayes(formula=label ~ can.swim + is.brown, data=df[1:6,])
```

Common classification methods: Decision Trees

- Useful when the input variables interact with the output in "if-then" ways
  - `IF age > 65, THEN has.health.insurance=T`
- They are also suitable when inputs have an AND relationship to each other when input variables are redundant or correlated.
  - `IF age < 25 AND student=T, THEN...`
- The decision rules are more interpretable for non-technical users than the decision processes of other classifiers.

Common classification methods: Decision Trees (Cont.)

An example: IF age > 65, THEN has.health.insurance=T

```
library(dplyr)
# create a small data frame df to deal with seniors (65+ years old)
df <- within(custdata, {is.senior <- age > 65}) %>%
      subset(select=c("is.senior","health.ins"))

# ggplot(df) + geom_bar(aes(x=is.senior, fill=health.ins),
#                       position="fill") +
#       labs(x="is senior", y="proportion")

# compute the proportion of seniors having health insurance
p <- sum( df$is.senior & df$health.ins ) / sum( df$is.senior )

cat("Proportion of seniors having health insurance is: ", p)
## Proportion of seniors having health insurance is:  0.995283
```

Common classification methods: Logistic Regression

- Useful for estimating the class probabilities in addition to class assignments.
- A good first choice method for binary classification problems.
- Probably the most important (and most used) member of a class of models called *generalized linear models*.
- Estimate the relative impact of different input variables on the output.
  - For example, a \$100 increase in transaction size increases the odds that the transaction is fraud by 2%, all else being equal.
- Strictly speaking, logistic regression is scoring. To turn a scoring algorithm into a classifier requires a threshold.

Common classification methods: Support Vector Machines

- Useful when there are very many input variables or when input variables interact with the outcome or with each other in complicated (nonlinear) ways.
- Compared to many other methods, SVMs make fewer assumptions about variable distribution. SVMs are particularly useful when the training data isn't completely representative of the data distribution in production.

References

- **Practical Data Science with R**, *Nina Zumel, John Mount*, Manning, 2nd Ed., 2020 (Chapter 6)

Classification Model Evaluation

CITS4009 Computational Data Analysis

Dr. Mubashar Hassan

Department of Computer Science and Software Engineering
The University of Western Australia

Semester 2, 2024

Section 1

Model Evaluation

Model types from evaluation point of view

- **Classification**
- Scoring
- Probability estimation
- Ranking
- Clustering

Models to calibrate against

1. **Null models**
   - Such a model tells us what **low performance** looks like. e.g., a typical null model is one that returns the same answer regardless of what the input is.
   - Note that it can be hard to do as good as the best null model. We always assume the null model we're comparing to is the best of all possible null models
2. **The best single-variable model** tells us what a simple model can achieve.
   - How to build single-variable models will be introduced later.

Null Models

Null models help to define the lower-bound of performance — **If you're not out-performing the null model, you're not delivering value.**

The two most common types of null models are:

1. a model that returns a single constant answer for all situations; or
2. a model that works independently of the data, i.e., it doesn't record any important relation or interaction between inputs and outputs.

For example,

- In a categorical problem, a model that always returns the most popular category (as this is the easy guess that is least often wrong) is a null model of type 1 above.
- For a score model, the null model is often the average of all the outcomes (as this has the least square deviation from all of the outcomes); and so on.

The best single variable model

We also suggest comparing any complicated model against the best single variable model you have available.

A complicated model can't be justified if it doesn't outperform the best single-variable model available from your training data.

Business analysts have many tools for building effective single-variable models (such as pivot tables), so if your client is an analyst, they're likely looking for performance above this level.

Section 2

Evaluating classification models

Spam email or not

```
path <- "../../data_v2/Spambase/spamD.tsv"
spamD <- read.table(path,header=T,sep='\t')
spamTrain <- subset(spamD,spamD$rgroup>=10)
spamTest <- subset(spamD,spamD$rgroup<10)
spamVars <- setdiff(colnames(spamD),list('rgroup','spam'))
spamFormula <- as.formula(paste('spam=="spam"',
                                paste(spamVars,collapse=' + '),
                                sep=' ~ '))
spamModel <- glm(spamFormula,family=binomial(link='logit'),
                 data=spamTrain)
spamTrain$pred <- predict(spamModel,newdata=spamTrain,
                          type='response')
spamTest$pred <- predict(spamModel,newdata=spamTest,
                         type='response')
```

Sample classifications

```
sample <- spamTest[c(7,35,224,327),c('spam','pred')]
kable(sample)
```

	spam	pred
115	spam	0.9903246
361	spam	0.4800498
2300	non-spam	0.0006847
3428	non-spam	0.0001434

Confusion matrix

A table counting how often each combination of known outcomes (the truth) occurred in combination with each prediction type.

```
cM <- table(truth=spamTest$spam,prediction=spamTest$pred>0.5)
kable(cM)
```

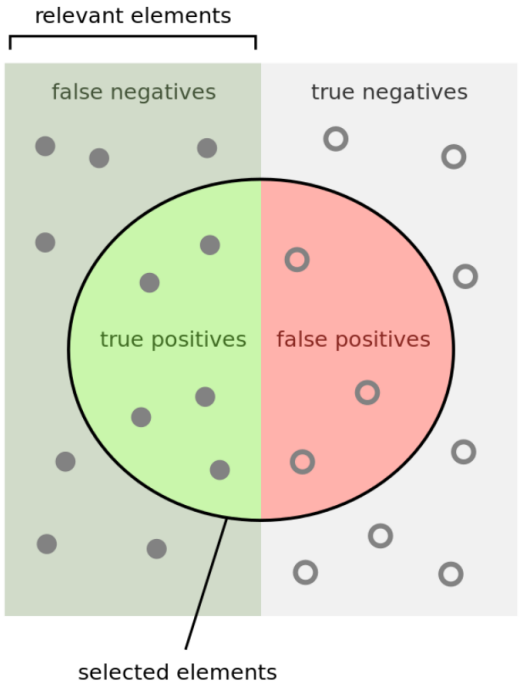
	FALSE	TRUE
non-spam	264	14
spam	22	158

For a two-by-two confusion matrix, each cell has a special name.

	Prediction=NEGATIVE	Prediction=POSITIVE
Truth mark=NOT IN CATEGORY	True negatives (TN) `cM[1,1]=264`	False positives (FP) `cM[1,2]=14`
Truth mark=IN CATEGORY	False negatives (FN) `cM[2,1]=22`	True positives (TP) `cM[2,2]=158`

Measures

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)



Accuracy

Accuracy is by far the most widely known measure of classifier performance.

- For a classifier, *accuracy* is defined as the number of items categorized correctly divided by the total number of items.
- It is simply what fraction of the time the classifier is correct. At the very least, you want a classifier to be accurate.
  - $\frac{TP+TN}{TP+FP+TN+FN}$, which is `(cM[1,1]+cM[2,2])/sum(cM)`
    (recall that `cM` is the confusion matrix computed on a previous slide)
- ACCURACY IS AN INAPPROPRIATE MEASURE FOR UNBALANCED CLASSES
  - Suppose we have a situation where we have a rare event (say, severe complications during childbirth). If the event we're trying to predict is rare (say, around 1% of the population), the null model -– the rare event never happens – is very accurate.

Precision and Recall

These terms come from the field of information retrieval.

- *Precision* is what fraction of the items the classifier flags as being in the class actually are in the class.
  - $\frac{\text{TP}}{\text{TP+FP}}$, which is `cM[2,2]/(cM[2,2]+cM[1,2])`
  - Precision is how often a positive indication turns out to be correct.
- *Recall* is what fraction of the things that are in the class are detected by the classifier.
  - $\frac{\text{TP}}{\text{TP+FN}}$ which is `cM[2,2]/(cM[2,2]+cM[2,1])`

F1 Score

The **F1 score** is a useful combination of precision and recall. If either precision or recall is very small, then F1 is also very small.

- F1 is defined as the harmonic mean of precision and recall:

$$\frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Sensitivity and Specificity

Scientists and doctors tend to use *sensitivity* and *specificity*.

- **Sensitivity** is also called the true positive rate and is exactly equal to recall:

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **Specificity** is also called the true negative rate:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

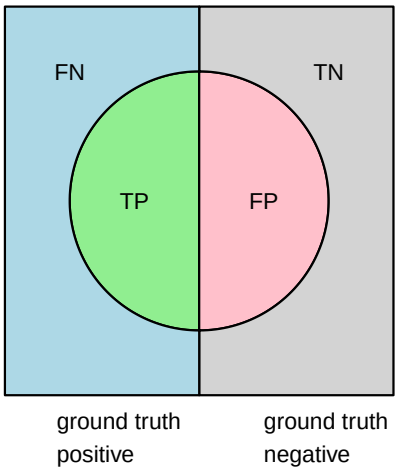
Sensitivity and Specificity (cont.)

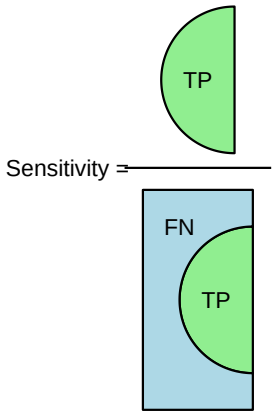
Both are measures of effect: what fraction of class members are identified as positive and what fraction of non-class members are identified as negative.

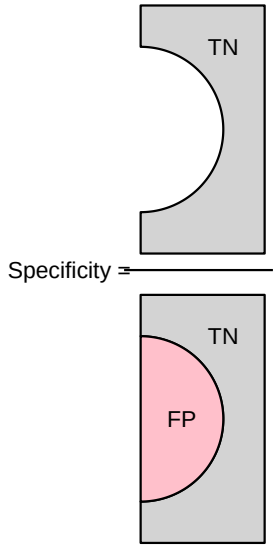
- If you flip your labels (switch from spam to non-spam being the class you're trying to identify), you just switch sensitivity and specificity.
- Any of the so-called null classifiers (classifiers that always say positive or always say negative) always return a zero score on either sensitivity or specificity. So useless classifiers always score poorly on at least one of these measures.
- Unlike *precision* and *accuracy*, *sensitivity* and *specificity* each only involves entries from a single row of the confusion matrix.

Sensitivity and Specificity (cont.)

	FALSE	TRUE
non-spam	TN	FP
spam	FN	TP







Identify the right measures that meet business needs

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 *three nines* 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?"

References

- Practical Data Science with R. By Nina Zumel and John Mount, Manning, 2014. (Chapter 5)
- Interpreting ROC: https://web.uvic.ca/~maryam/DMSpring94/Slides/9_roc.pdf