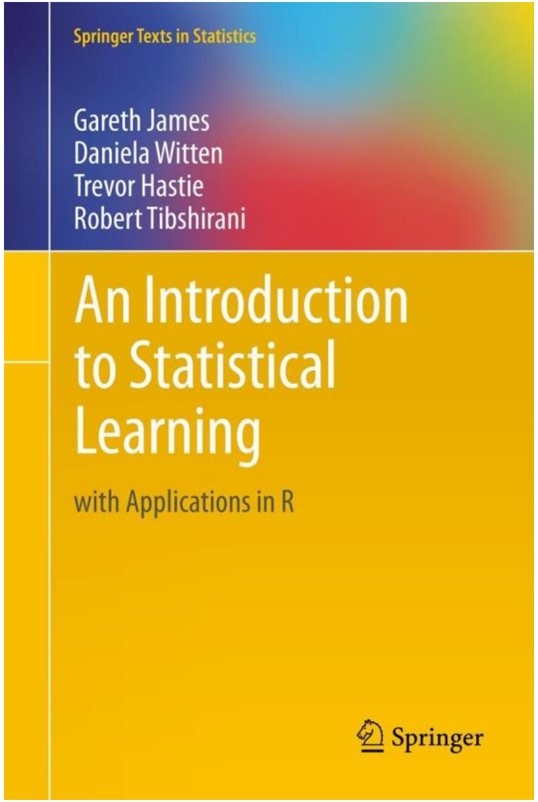
```
library(caret)

rladies_global %>%
  filter(city == ‘Leuven')
```

Classification



Springer Texts in Statistics

Gareth James
Daniela Witten
Trevor Hastie
Robert Tibshirani

An Introduction to Statistical Learning

with Applications in R

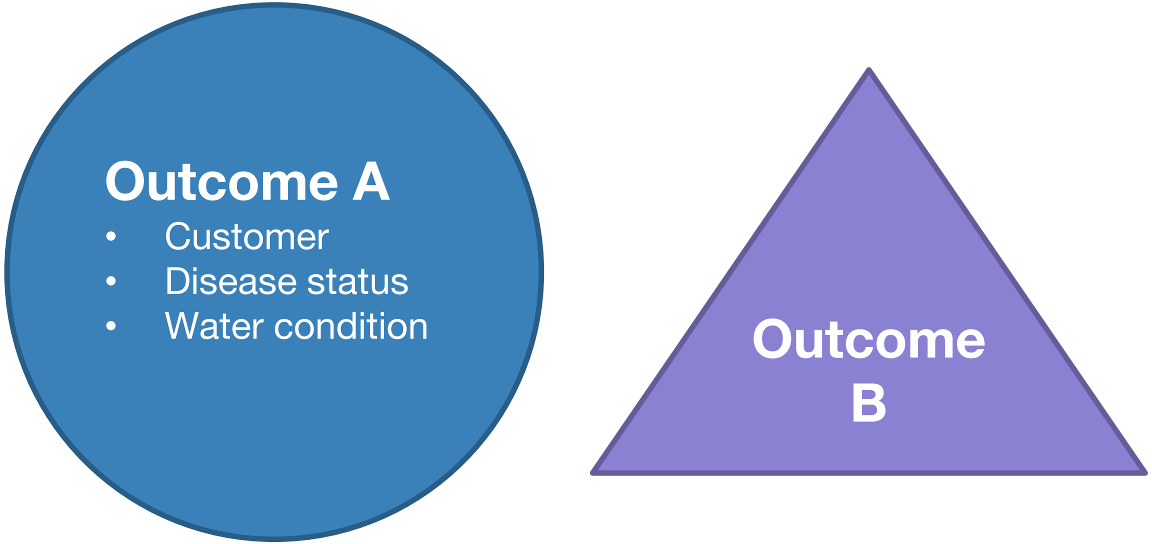
Springer

This book
is one of the best machine learning books out there.

It’s also free.

http://www-bcf.usc.edu/~gareth/ISL/ISLR%20First%20Printing.pdf

We have a labelled dataset.

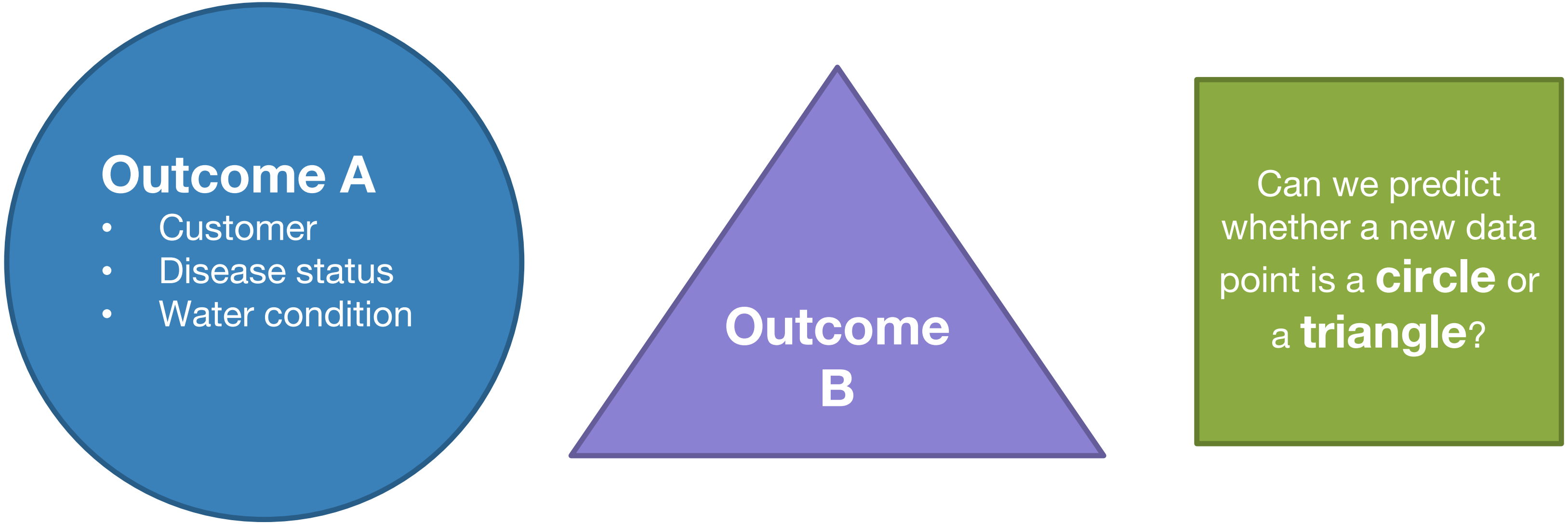


Outcome A

- Customer
- Disease status
- Water condition

Outcome B

We have a labelled dataset.



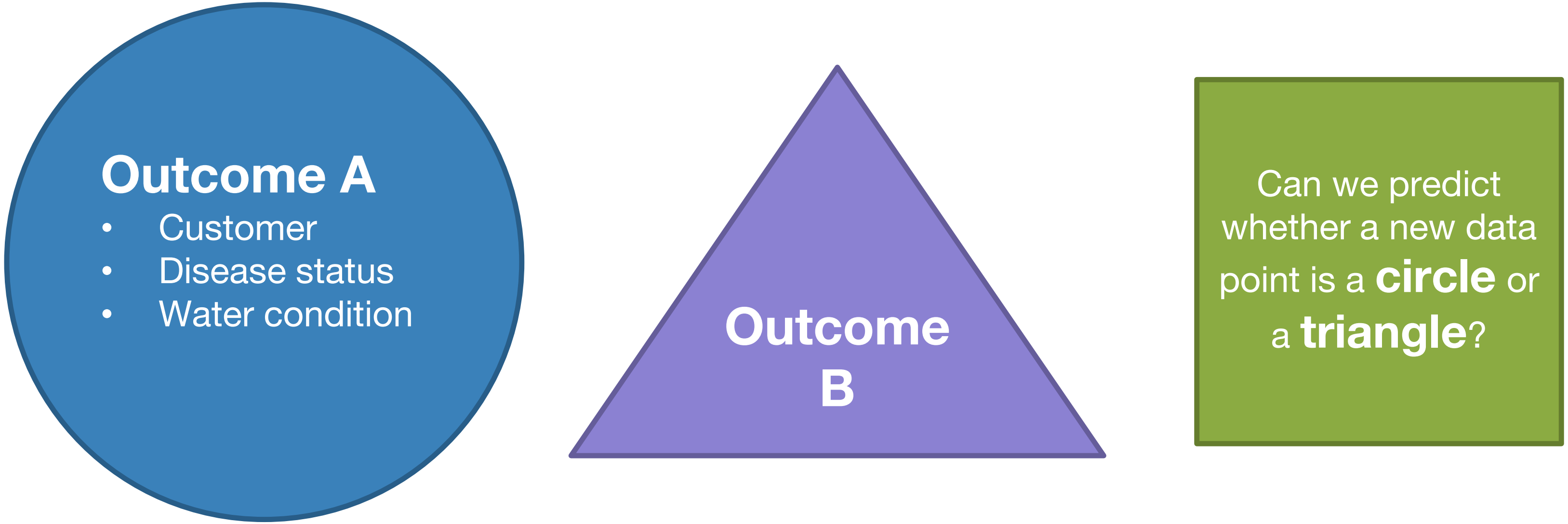
Outcome A

- Customer
- Disease status
- Water condition

Outcome B

Can we predict whether a new data point is a **circle** or a **triangle**?

We have a labelled dataset.



Outcome A

- Customer
- Disease status
- Water condition

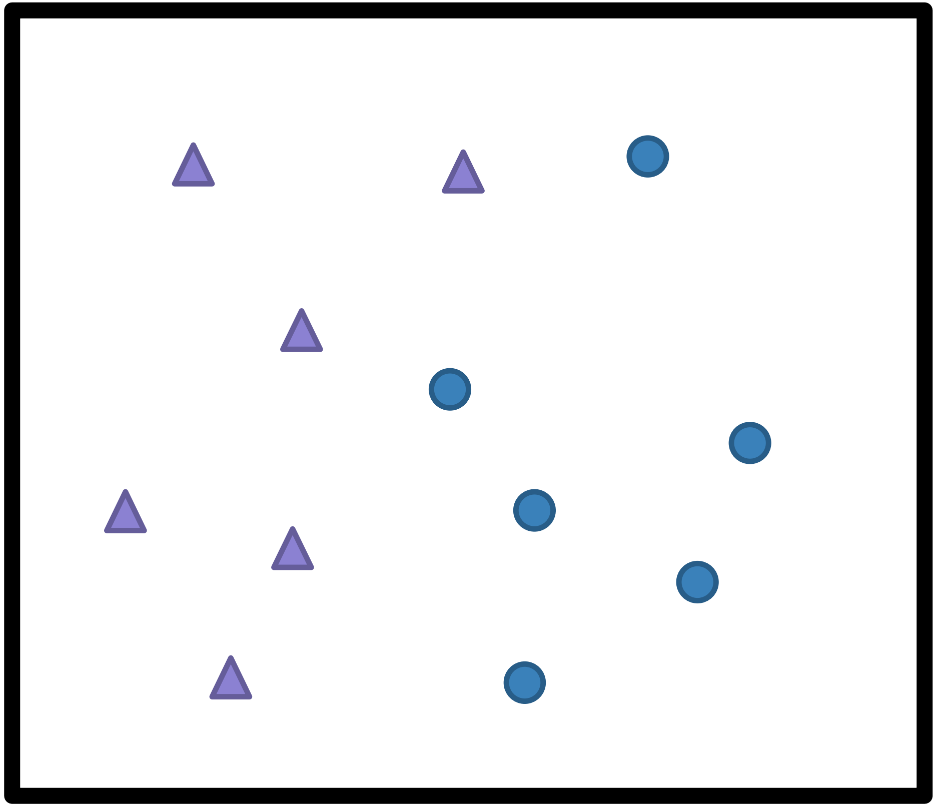
Outcome B

Can we predict whether a new data point is a **circle** or a **triangle**?

Let’s see how these algorithms work.

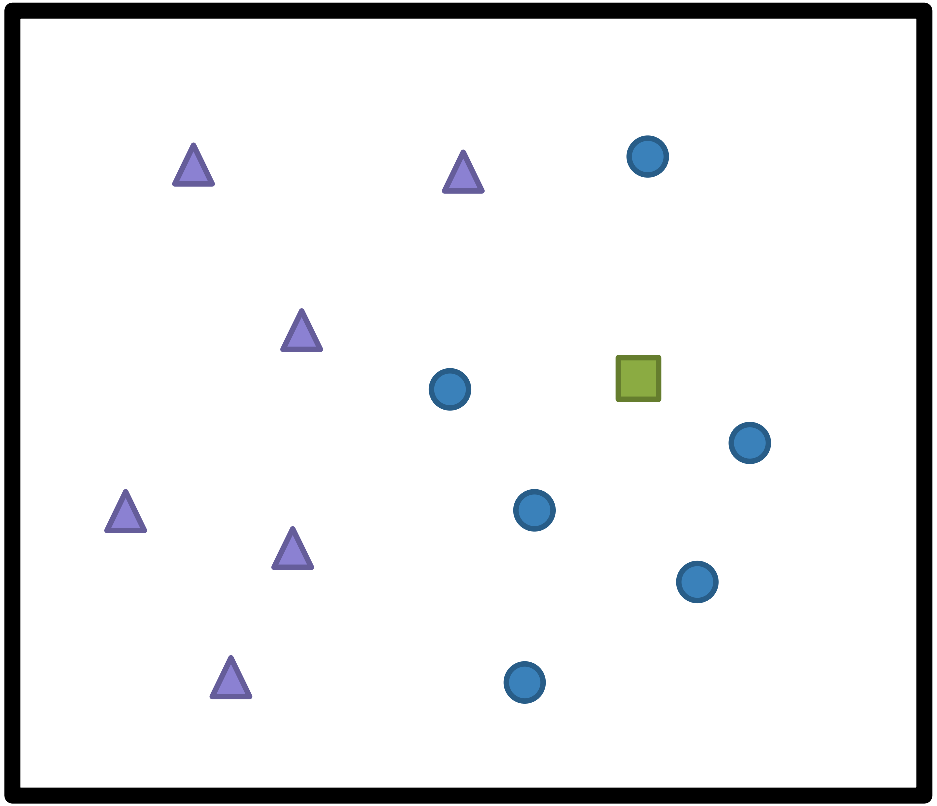
K Nearest Neighbours

Labels are assigned to known datapoints.



Labels are assigned to known datapoints.

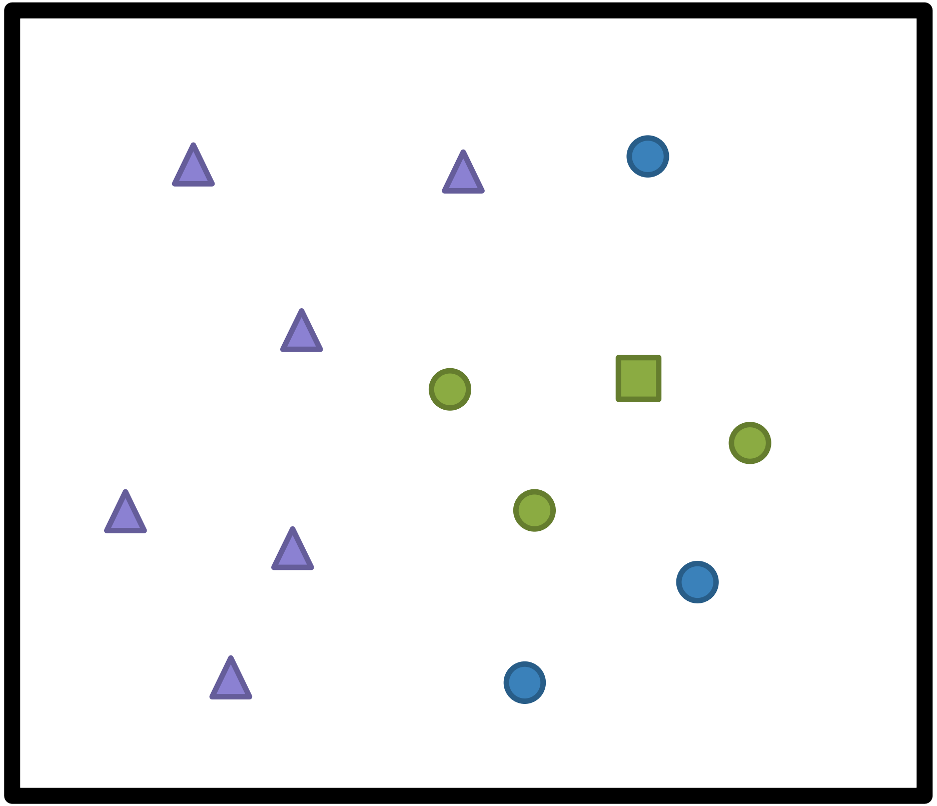
New point is placed within **known space**.



Labels are assigned to known datapoints.

New point is placed within **known space**.

Find **K neighbours** (here, K = 3).

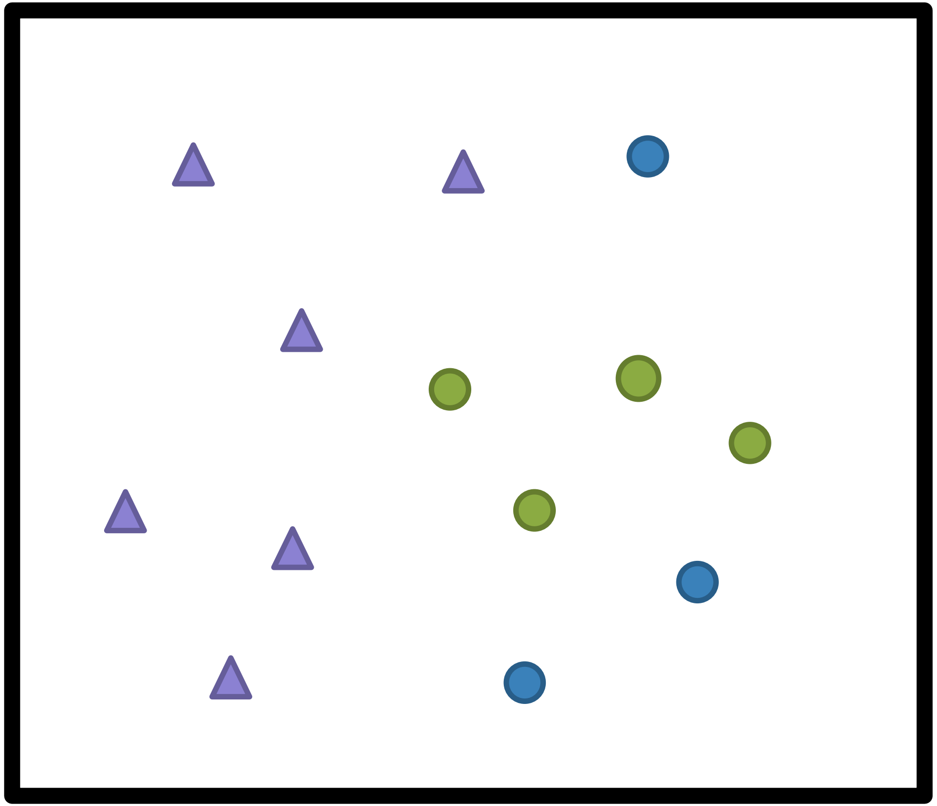


Labels are assigned to known datapoints.

New point is placed within **known space**.

Find **K neighbours** (here, K = 3).

All neighbours are circles -> **new point** is a circle.



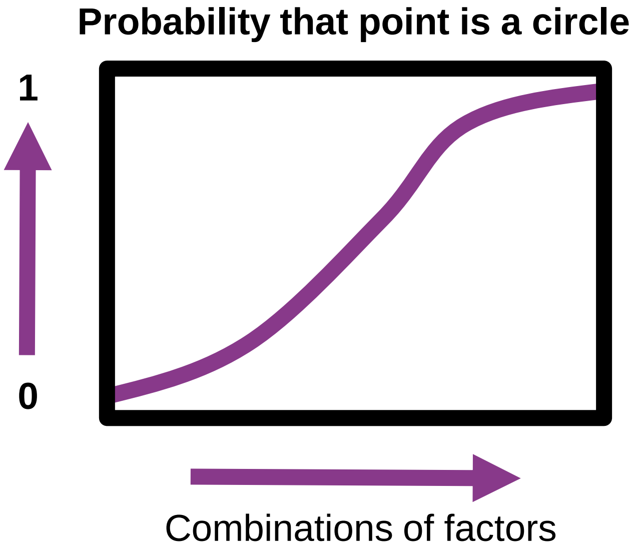
Logistic Regression

The **logistic function** returns probabilities between 1 and 0.

$$\sigma(t) = \frac{1}{1 + \ e^{-t}}$$

The **logistic function** returns probabilities between 1 and 0.

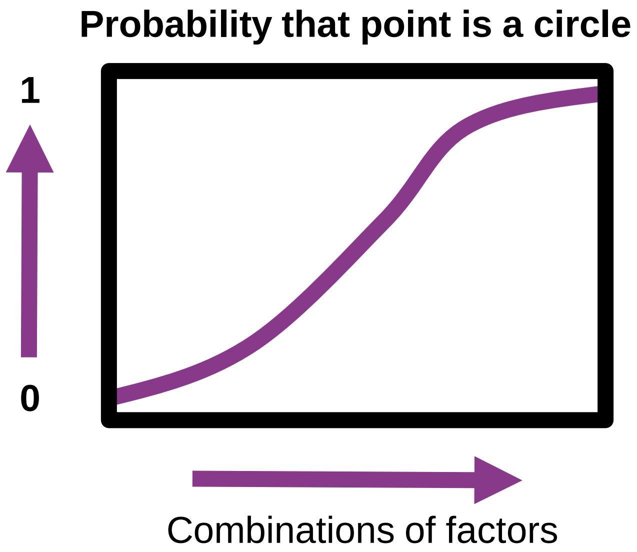
$$\sigma(t) = \frac{1}{1 + \ e^{-t}}$$



The **logistic function** returns probabilities between 1 and 0.

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

We cannot perform linear regression on this function, because it is non-linear.

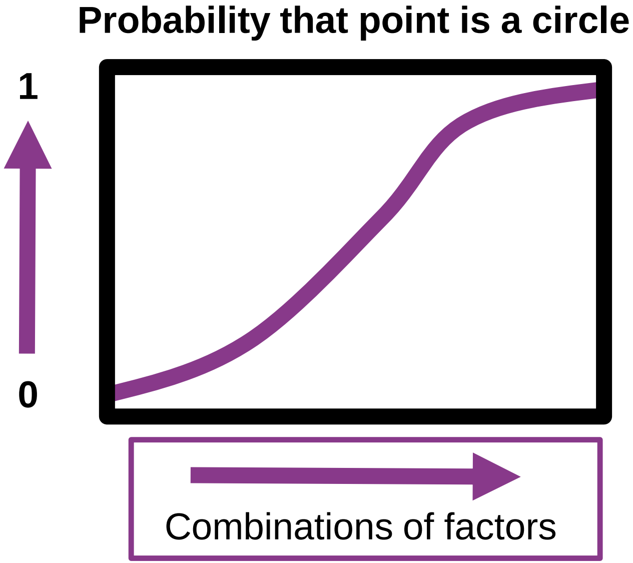


The **logistic function** returns probabilities between 1 and 0.

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

We cannot perform linear regression on this function, because it is non-linear.

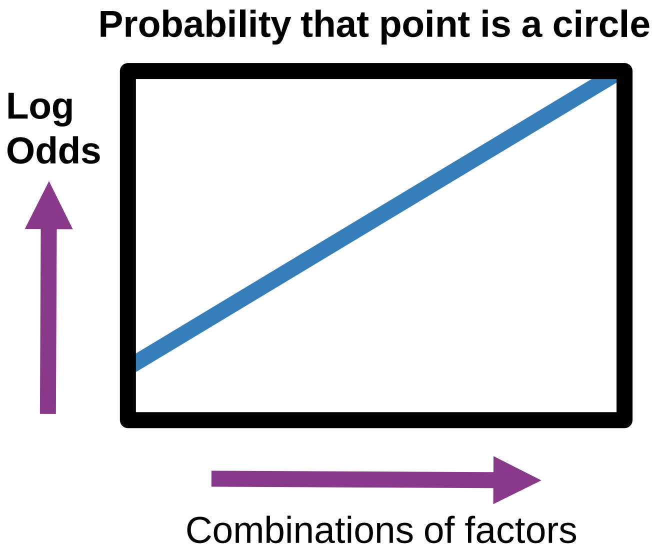
With a known t, we can define the **logit function** by taking the inverse of the **logistic function**.



The **logistic function** returns probabilities between 1 and 0.

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

We cannot perform linear regression on this function, because it is non-linear.



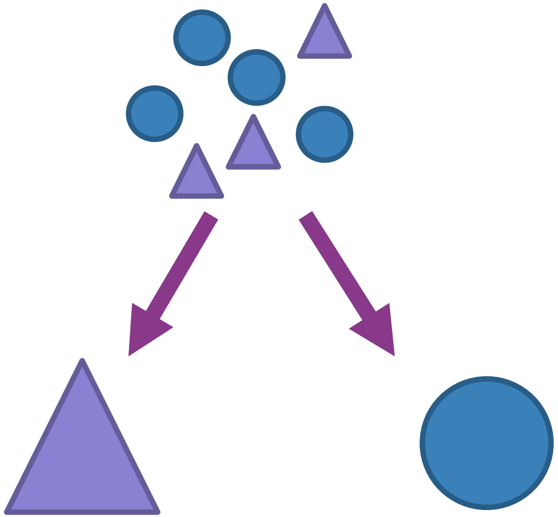
With a known t, we can define the **logit function** by taking the inverse of the **logistic function**.

The **logit function** is linear. Therefore, we can use linear regression to fit a model.

Random Forest

Random forests are like simple **decision trees**.

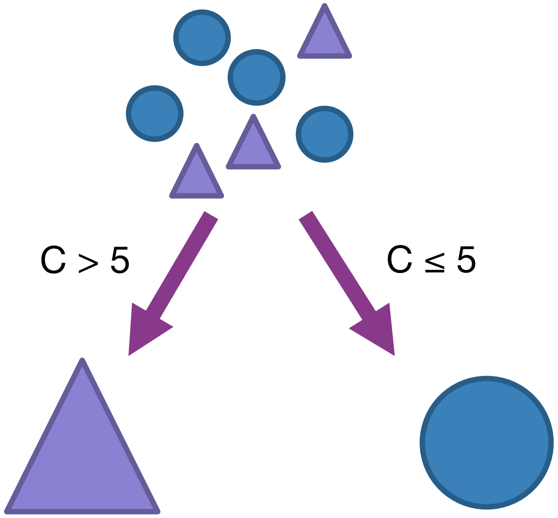
A **decision tree** finds the best way to split a dataset.



Random forests are like simple **decision trees**.

A **decision tree** finds the best way to split a dataset.

It looks through **all predictors** to find the one that has the smallest prediction error.

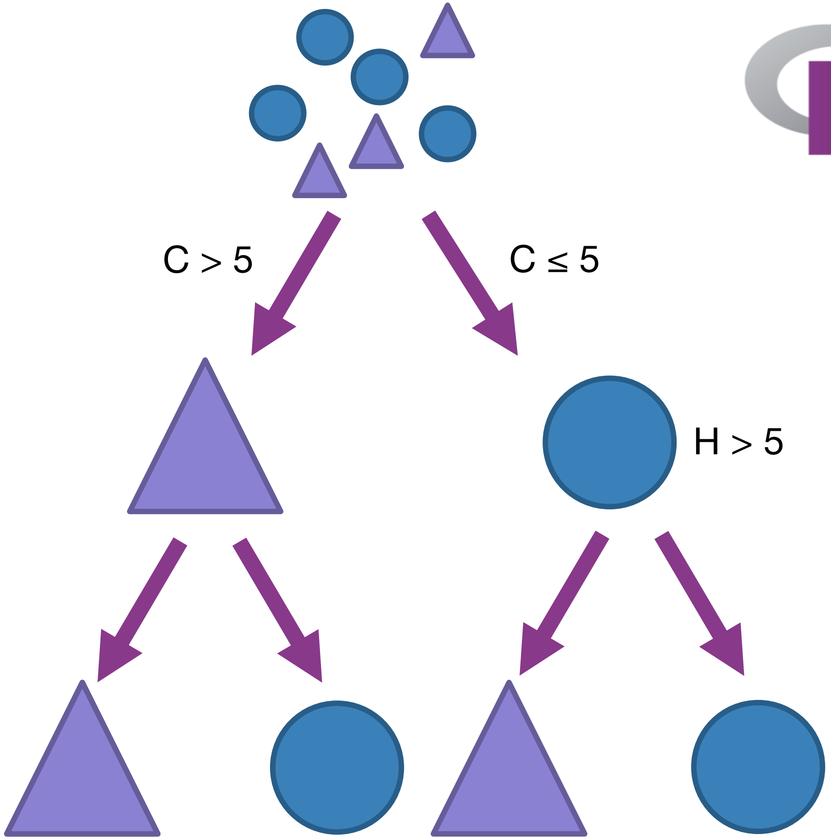


Random forests are like simple **decision trees**.

A **decision tree** finds the best way to split a dataset.

It looks through **all predictors** to find the one that has the smallest prediction error.

Then, a new branch is generated.



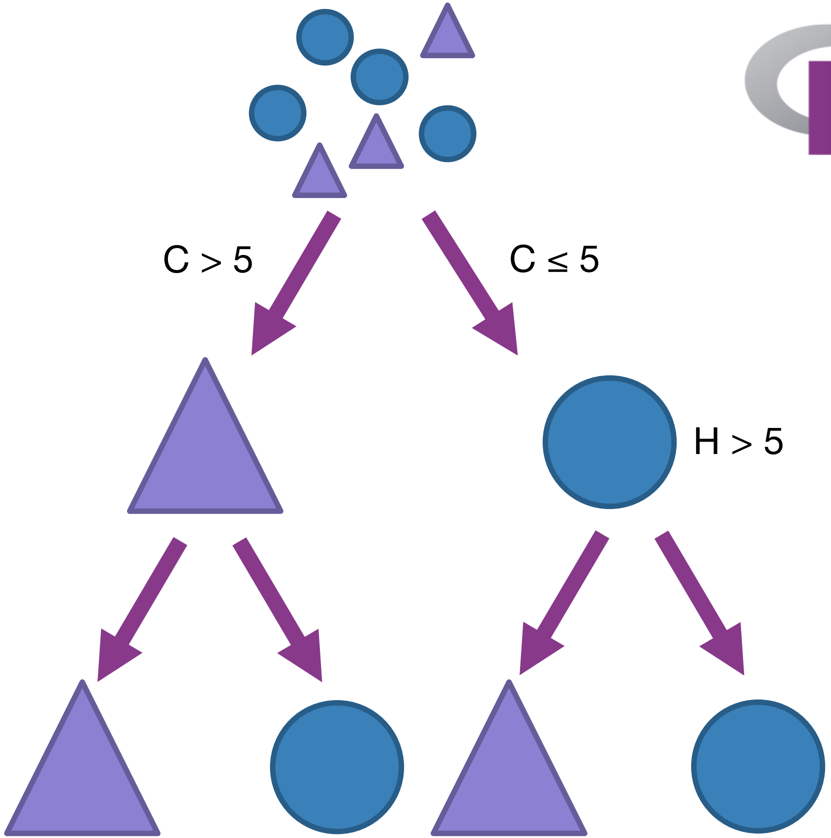
Random forests are like simple **decision trees**.

A **decision tree** finds the best way to split a dataset.

It looks through **all predictors** to find the one that has the smallest prediction error.

Then, a new branch is generated.

This results in high sensitivity to noise and to overfitting.



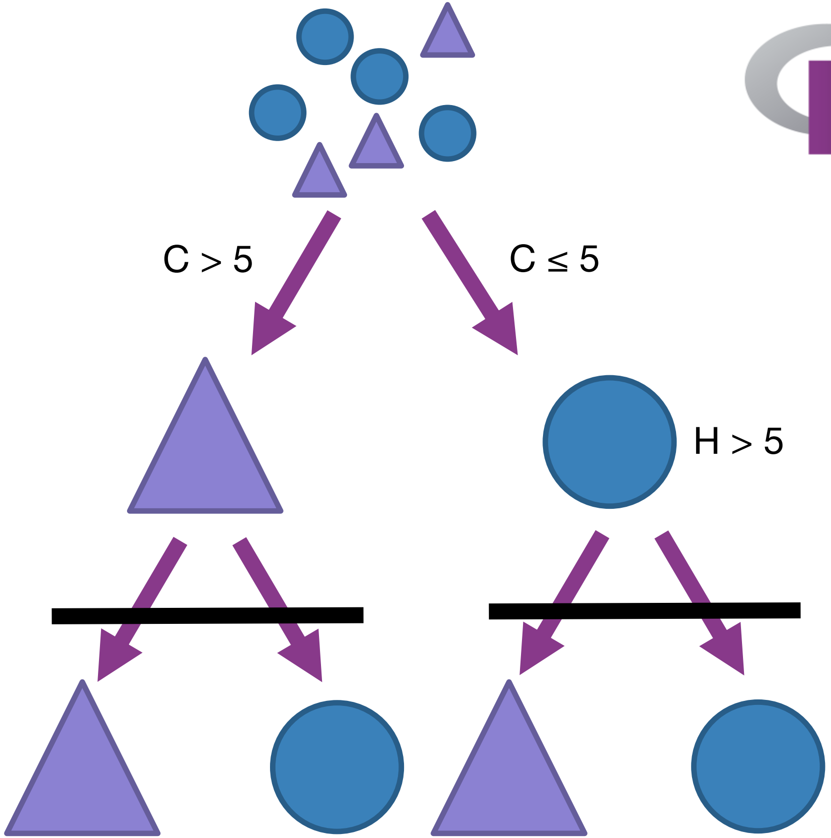
Random forests are like simple **decision trees**.

A **decision tree** finds the best way to split a dataset.

It looks through **all predictors** to find the one that has the smallest prediction error.

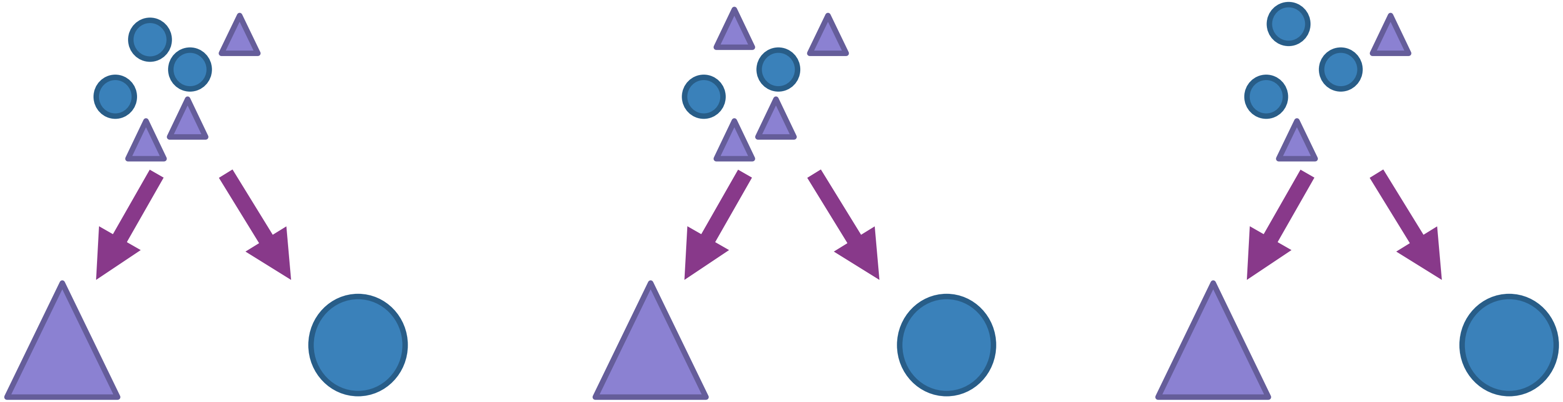
Then, a new branch is generated.

This results in high sensitivity to noise and to overfitting.

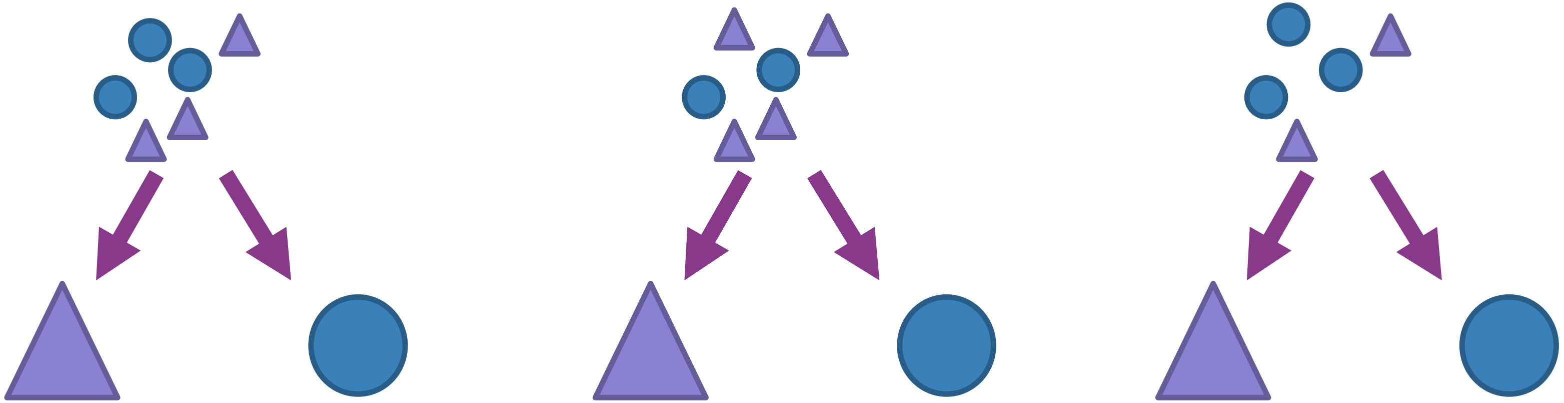


Why use one tree when you can have a forest?

Random forests are made from multiple decision “stumps”.

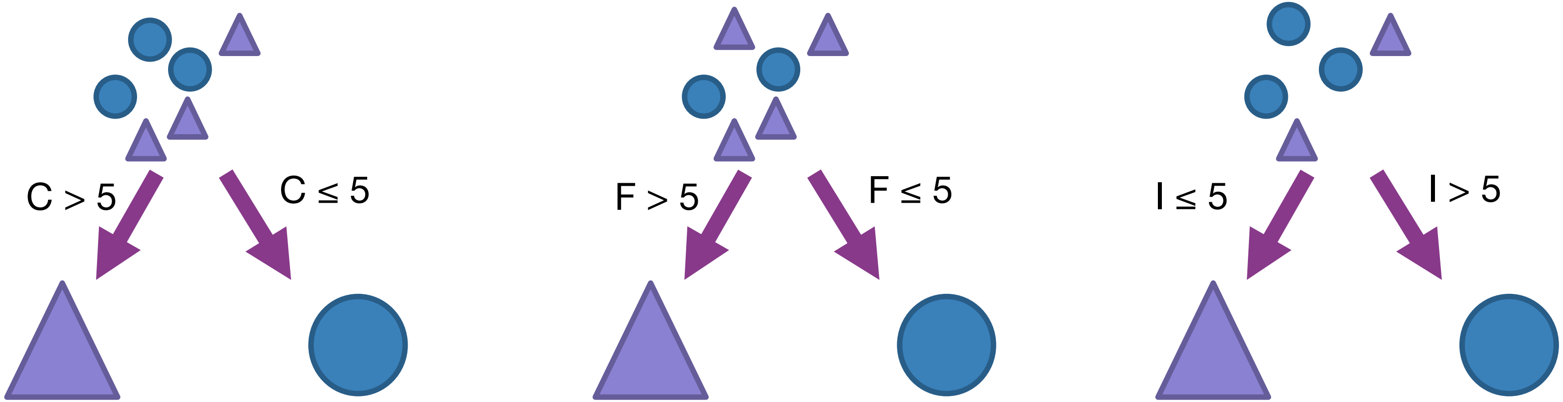


Random forests are made from multiple decision "stumps".



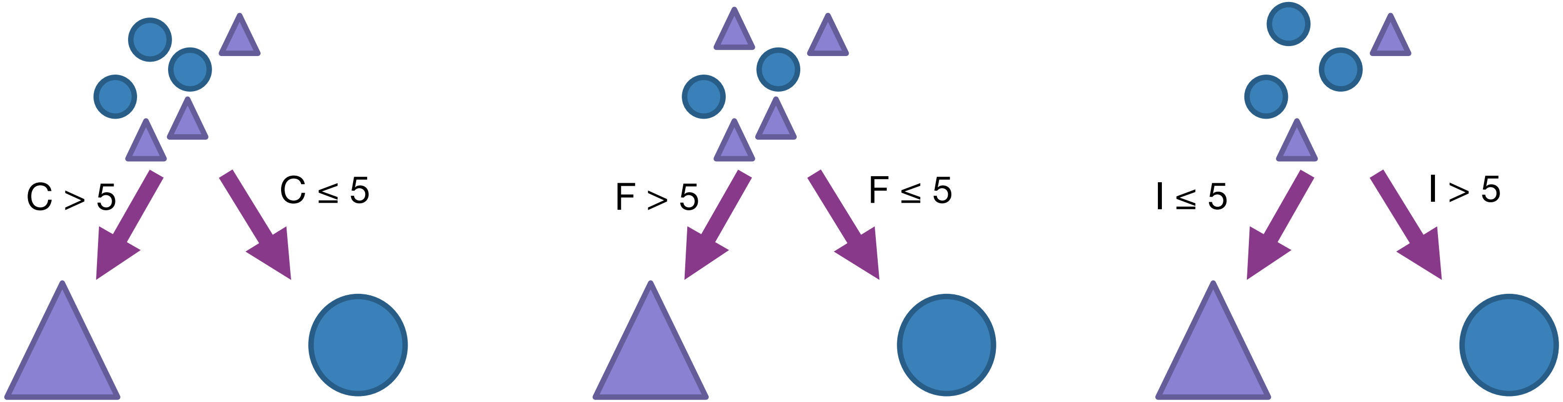
Each **stump** is given a training set **randomly sampled with replacement** from the original set.

Random forests are made from multiple decision “stumps”.



Each **stump** is given a training set **randomly sampled with replacement** from the original set. They use one predictor.

Random forests are made from multiple decision “stumps”.



Each **stump** is given a training set **randomly sampled with replacement** from the original set. They use one predictor.

The **random forest** averages the predictions (regression) or takes the majority vote (classification).

Classification
with caret

Data preparation

Model training

Model evaluation

Find a dataset

```
https://archive.ics.uci.edu/ml/datasets/Cervical+cancer
+%28Risk+Factors%29
```

Cervical cancer (Risk Factors) Data Set
Fernandes, K., Cardoso, J. S., & Fernandes, J. (2017, June). Transfer learning with partial observability applied to cervical cancer screening. In *Iberian conference on pattern recognition and image analysis* (pp. 243-250). Springer, Cham.

```
library(datasets)
data(iris)
```

Model training

Replace ? with NA

`read.csv`

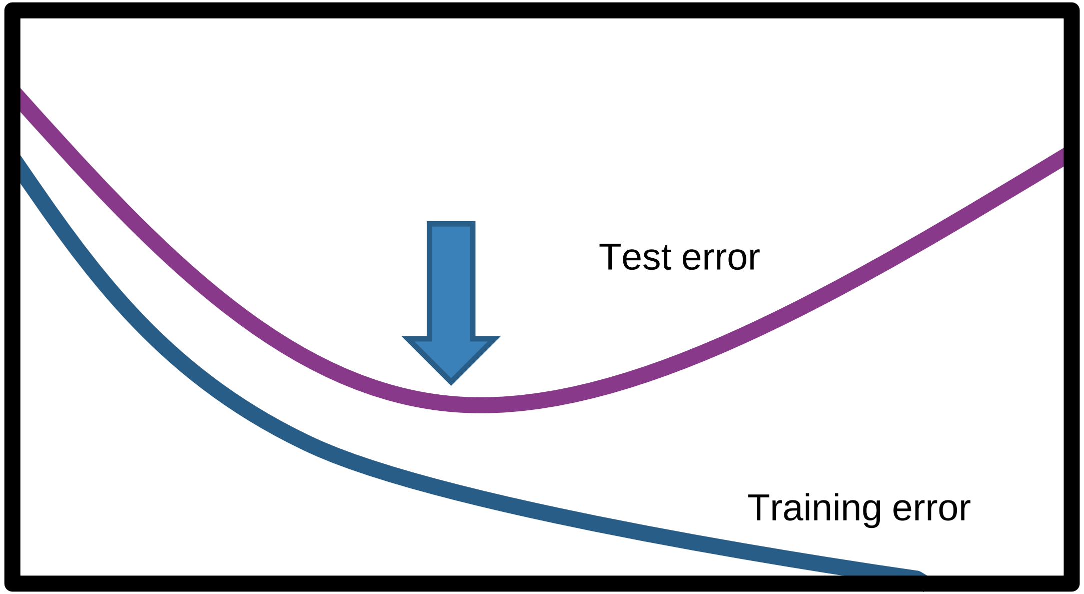
Split up your data in a training and a test set

`createDataPartition`

Train your model

`train`

Prevent overfitting!



Model testing

Test model

`predict`

Evaluate predictions

`confusionMatrix`

`LOOCV`

`Boot`

`cv`

Improve model

`kappa`

`trControl`

Unlabeled data

Upclass

RSSL