Day 2

Classification Example

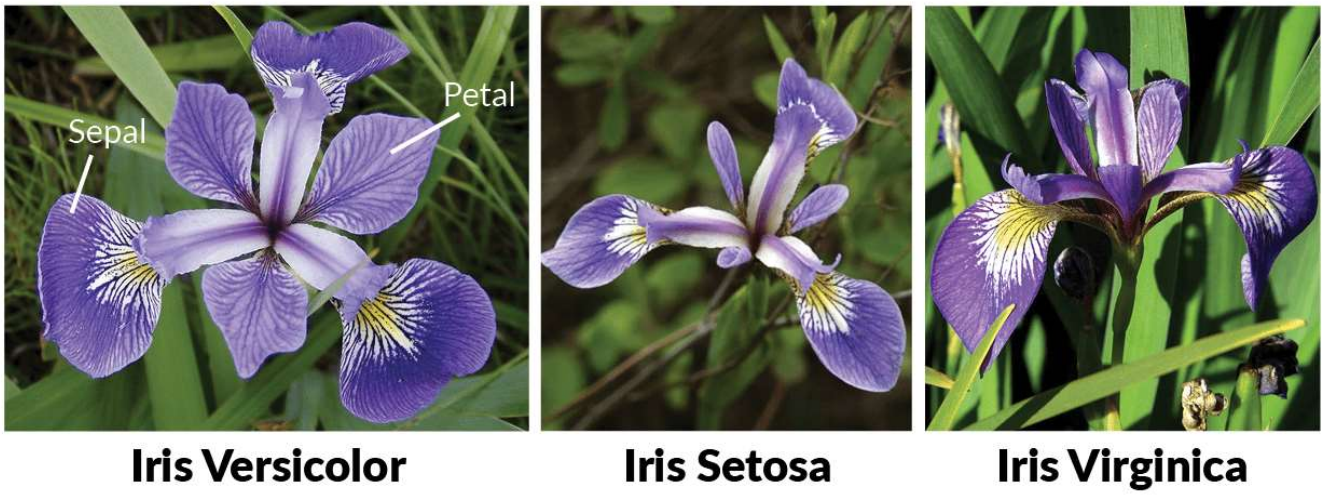
Our data: Iris data set

This is one of the earliest datasets used in the literature on classification methods and widely used in statistics and machine learning.

This famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris.

The species are *Iris setosa*, *versicolor*, and *virginica*.

Predicted attribute: class of iris plant.



From: Machine Learning in R for beginners

It is possible to download the data from the UCI Machine Learning Repository -- Iris Data Set.

The datasets library in R already contains it (data frame named **iris**).

▼ Code

```
data(iris)
head(iris)
```

```
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
```

Packages

Caret package

- The caret package was created by Max Kuhn and contains lots of helpful tools for each section of the modelling process, helping to build powerful and robust modelling pipelines (%>%).
- **Cla**ssification and **Re**gression **T**raining - Version 6.0-94
- Functions for training and plotting classification and regression models.

Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, 28(5), 1–26. https://doi.org/10.18637/jss.v028.i05

http://topepo.github.io/caret/index.html

The packages

▼ Code

```
library(tidyverse)
library(tidymodels)
library(skimr)
library(caret)
```

Summarize data set

▼ Code

```
glimpse(iris)
```

```
Rows: 150
Columns: 5
$ Sepal.Length <dbl> 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4, 4.…
$ Sepal.Width  <dbl> 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1, 3.7, 3.…
$ Petal.Length <dbl> 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.…
$ Petal.Width  <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.…
$ Species      <fct> setosa, setosa, setosa, setosa, setosa, setosa, setosa, s…
```

Descriptive statistics

▼ Code

```
iris %>% skim()
```

Data summary

Name	Piped data
Number of rows	150
Number of columns	5
_____	
Column type frequency:	
factor	1
numeric	4
_____	
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Species	0	1	FALSE	3	set: 50, ver: 50, vir: 50

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Sepal.Length	0	1	5.84	0.83	4.3	5.1	5.80	6.4	7.9	
Sepal.Width	0	1	3.06	0.44	2.0	2.8	3.00	3.3	4.4	
Petal.Length	0	1	3.76	1.77	1.0	1.6	4.35	5.1	6.9	
Petal.Width	0	1	1.20	0.76	0.1	0.3	1.30	1.8	2.5	

Optional: skim (iris)

Time to create some models

Classification

Predicted attribute: class of iris plant.

Model	Col2
LDA	Linear discriminant analysis
KNN	K-nearest neighbor
DT	Decision Tree
RF	Random Forest
SVM	Support vector machine

Metrics

- Accuracy
- Overall, sensitivity and specificity
- KAPPA
- Confusion matrix

Split data: train and test

Select 80% of the data for training and use the remaining 20% to test. The iris data frame was renamed to dataset.

▼ Code

```
dataset<-iris
```

▼ Code

```
validation_index <- createDataPartition(dataset$Species, p=0.80, list=FALSE)
test <- dataset[-validation_index,]
train <- dataset[validation_index,]
```

Train control

Control the computational *nuances* of the caret `train` function. The function `train` *train* the model.

Resampling methods, pre-processing,....

https://rdrr.io/rforge/caret/man/trainControl.html

▼ Code

```
control <- trainControl(method="cv", number=10)
```

We are using cross-validation and number of $folds = 10$.

Now, the models 😊!

Linear discriminant analysis

Training the model (named as *fit.lda*)

fit.lda <- train(**response variable ~. predictors**, **data**=train, **method**="machine learning method", **trControl**="nuances", **metric** = metric to select optimal model,...)

More information about the train function: tipe **?train** or **help(train)**

▼ Code

```r
set.seed(7)
fit.lda <- train(Species~., data=train, method="lda", trControl=control)
fit.lda
```

```
Linear Discriminant Analysis

120 samples
  4 predictor
  3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
Resampling results:

  Accuracy   Kappa
  0.9833333  0.975
```

KNN K-nearest neighbors

▼ Code

```r
set.seed(7)
fit.knn <- train(Species~., data=train, method="knn",  trControl=control)
fit.knn
```

```
k-Nearest Neighbors

120 samples
  4 predictor
  3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
Resampling results across tuning parameters:

  k  Accuracy   Kappa
  5  0.9833333  0.9750
  7  0.9750000  0.9625
  9  0.9750000  0.9625

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 5.
```

SVM

▼ Code

```r
set.seed(7)
fit.svm <- train(Species~., data=train, method="svmRadial",  trControl=control)
fit.svm
```

```
Support Vector Machines with Radial Basis Function Kernel

120 samples
  4 predictor
  3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
Resampling results across tuning parameters:

  C     Accuracy   Kappa
  0.25  0.9333333  0.9000
  0.50  0.9416667  0.9125
  1.00  0.9416667  0.9125
```

```
Tuning parameter 'sigma' was held constant at a value of 0.4862209
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were sigma = 0.4862209 and C = 0.5.
```

Decision tree

▼ Code

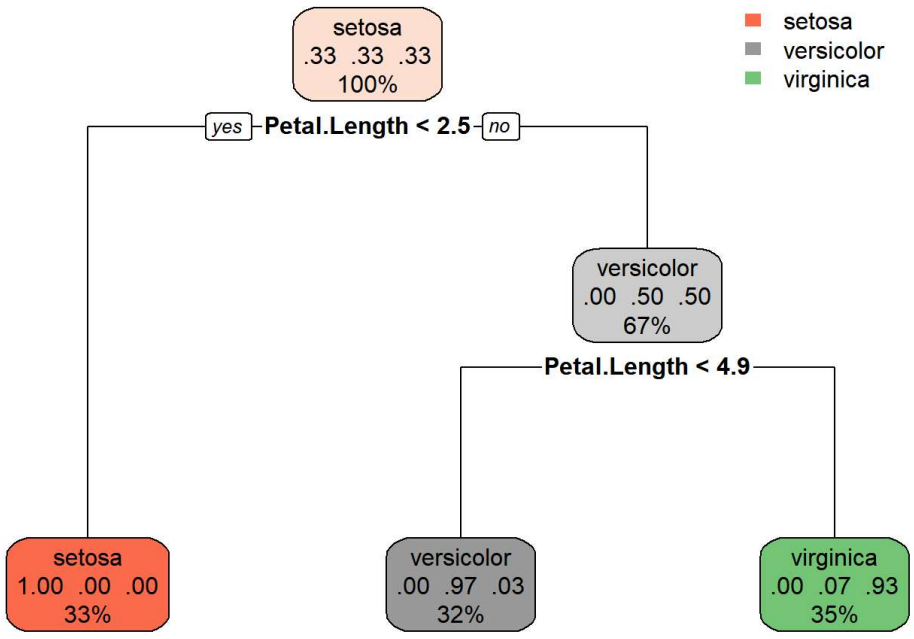
```r
set.seed(7)
fit.dt <- train(Species~., data=train, method="rpart", trControl=control)
fit.dt
```

```
CART

120 samples
  4 predictor
  3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
Resampling results across tuning parameters:

  cp    Accuracy   Kappa
  0.00  0.9250000  0.8875
  0.45  0.7750000  0.6625
  0.50  0.3333333  0.0000

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was cp = 0.
```

Plotting the regression tree

▼ Code

```r
library(rpart.plot)
rpart.plot(fit.dt$finalModel, extra=104)
```



Random Forest

▼ Code

```
set.seed(7)
fit.rf <- train(Species~., data=train, method="rf", trControl=control)
fit.rf
```

```
Random Forest

120 samples
  4 predictor
  3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
  2     0.9583333  0.9375
  3     0.9583333  0.9375
  4     0.9583333  0.9375

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.
```

Select the best model

- select best model - resampling and looking the results
- summarize accuracy of models : summary (results)

▼ Code

```
results <- resamples(list(lda=fit.lda, knn=fit.knn, dt= fit.dt,svm=fit.svm,rf=fit.rf))
summary(results)
```

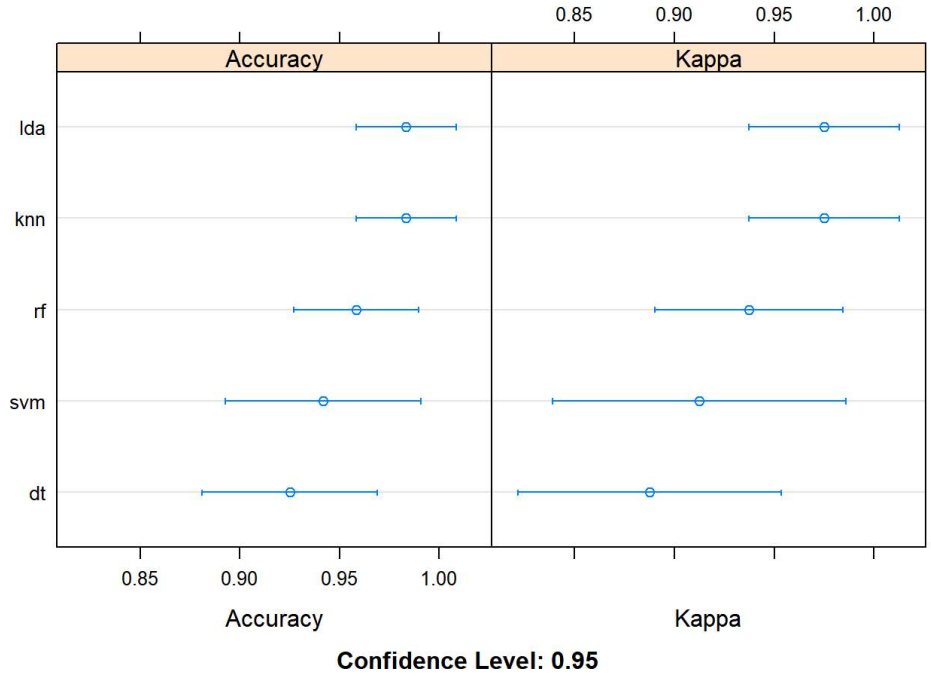
```
Call:
summary.resamples(object = results)

Models: lda, knn, dt, svm, rf
Number of resamples: 10

Accuracy
         Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
lda 0.9166667 1.0000000 1.0000000 0.9833333 1.0000000    1    0
knn 0.9166667 1.0000000 1.0000000 0.9833333 1.0000000    1    0
dt  0.8333333 0.9166667 0.9166667 0.9250000 0.9791667    1    0
svm 0.8333333 0.9166667 0.9583333 0.9416667 1.0000000    1    0
rf  0.9166667 0.9166667 0.9583333 0.9583333 1.0000000    1    0

Kappa
     Min. 1st Qu. Median   Mean 3rd Qu. Max. NA's
lda 0.875   1.000 1.0000 0.9750 1.00000    1    0
knn 0.875   1.000 1.0000 0.9750 1.00000    1    0
dt  0.750   0.875 0.8750 0.8875 0.96875    1    0
svm 0.750   0.875 0.9375 0.9125 1.00000    1    0
rf  0.875   0.875 0.9375 0.9375 1.00000    1    0
```

Compare accuracy of models

dotplot(results)

▼ Code

```
dotplot(results)
```



Best Model

LDA model 🌞

▼ Code

```
print(fit.lda)
```

```
Linear Discriminant Analysis

120 samples
  4 predictor
  3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
Resampling results:

  Accuracy   Kappa
  0.9833333  0.975
```

Make some predictions

Predictions with test data

▼ Code

```
predictions <- predict(fit.lda, test)
```

Accuracy

Our results

▼ Code

```
confusionMatrix(predictions, test$Species)
```

```
Confusion Matrix and Statistics
```

```
            Reference
Prediction   setosa versicolor virginica
  setosa         10          0         0
  versicolor      0          9         0
  virginica       0          1        10

Overall Statistics

               Accuracy : 0.9667
                 95% CI : (0.8278, 0.9992)
    No Information Rate : 0.3333
    P-Value [Acc > NIR] : 2.963e-13

                  Kappa : 0.95

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: setosa Class: versicolor Class: virginica
Sensitivity                 1.0000            0.9000           1.0000
Specificity                 1.0000            1.0000           0.9500
Pos Pred Value              1.0000            1.0000           0.9091
Neg Pred Value              1.0000            0.9524           1.0000
Prevalence                  0.3333            0.3333           0.3333
Detection Rate              0.3333            0.3000           0.3333
Detection Prevalence        0.3333            0.3000           0.3667
Balanced Accuracy           1.0000            0.9500           0.9750
```

▼ Code

```
test$predictions <- predictions
ggplot(test)+geom_point(aes(x=Species,y=predictions,color=Species))
```