Introduction to Machine Learning

Working Group "Computational Statistics" – Bernd Bischl et al.

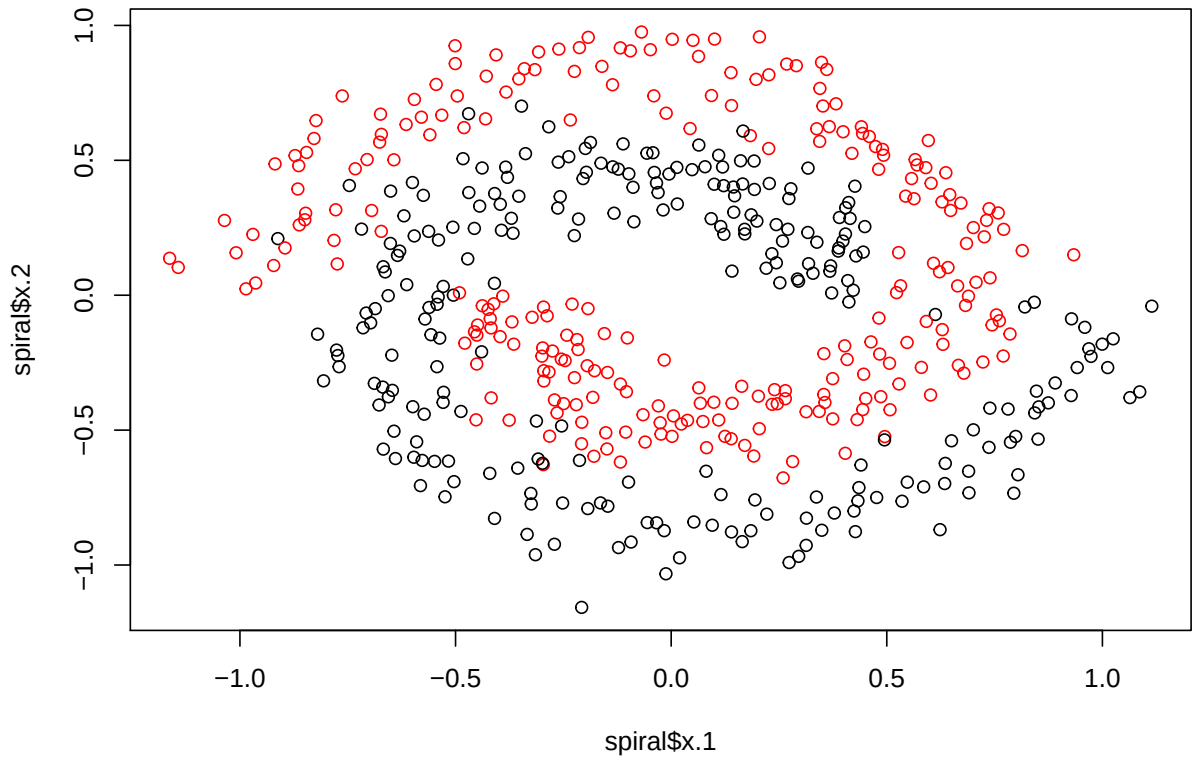
Code demo about overfitting

Overfitting kNN

Why do we have to split the data into **training sets** and **test sets**?

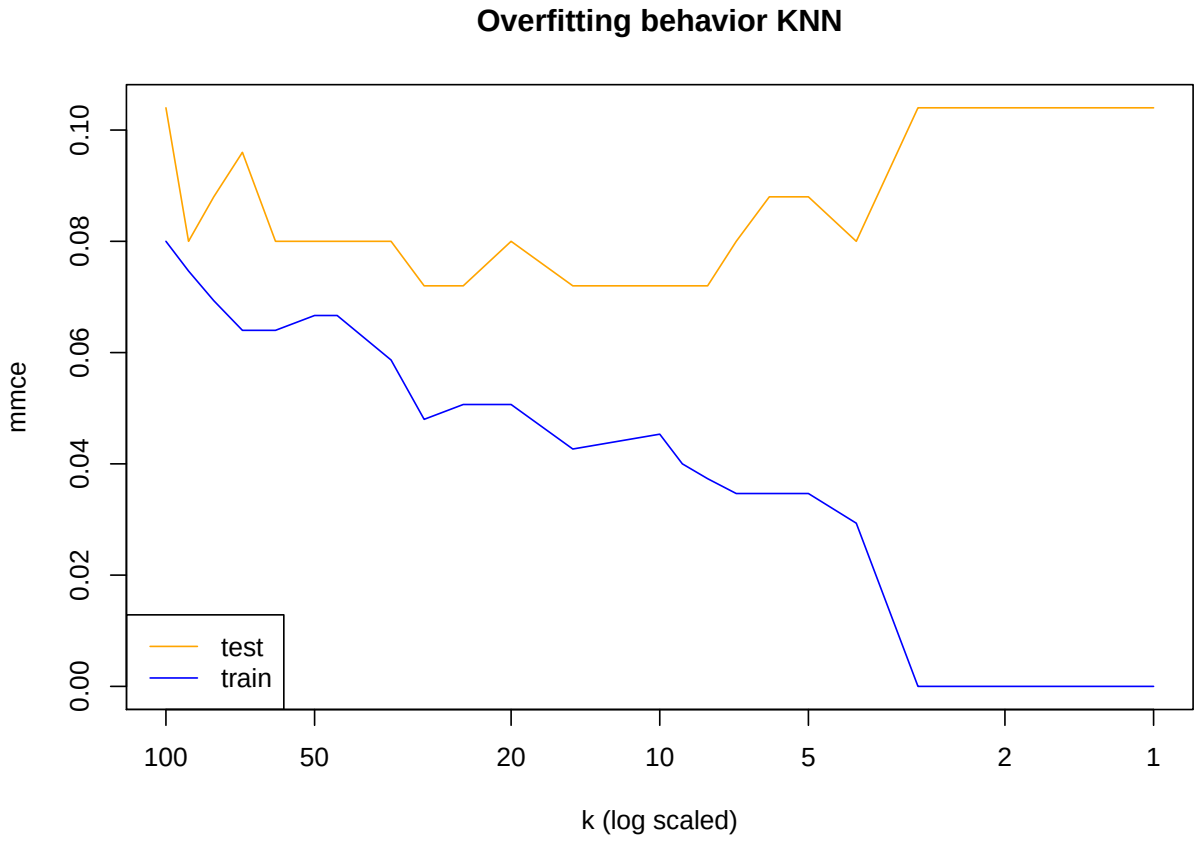
Check the performance of the kNN classifier on the test and train set depending on the hyperparameter k:

```
library(mlr3)
library(mlr3learners)
library(mlbench)

set.seed(13)
spiral <- as.data.frame(mlbench.spirals(n = 500, sd = 0.1))
plot(x = spiral$x.1, y = spiral$x.2, col = spiral$classes)
```




```
# split: 75 % for training, 25 % for test:
train_size <- 3 / 4
train_indices <- sample(
  x = seq(1, nrow(spiral), by = 1), size =
    ceiling(train_size * nrow(spiral)), replace = FALSE
)
spiral_train <- spiral[ train_indices, ]
spiral_test <- spiral[ -train_indices, ]



# run experiment
k_values <- rev(c(1:10, 15, 20, 25, 30, 35, 45, 50, 60, 70, 80, 90, 100))
```

```r
storage <- data.frame(matrix(NA, ncol = 3, nrow = length(k_values)))
colnames(storage) <- c("mmce_train", "mmce_test", "k")

spiral_task <- TaskClassif$new(
  id = "spirals", backend = spiral_train,
  target = "classes"
)
for (i in 1:length(k_values)) {
  spiral_learner <- lrn("classif.kknn", k = k_values[i])
  spiral_learner$train(task = spiral_task)

  # test data
  # choose additional adequate measures from: mlr3::mlr_measures
  spiral_pred <- spiral_learner$predict_newdata(newdata = spiral_test)
  storage[i, "mmce_test"] <- spiral_pred$score(msr("classif.ce"))

  # train data
  spiral_pred <- spiral_learner$predict_newdata(newdata = spiral_train)
  storage[i, "mmce_train"] <- spiral_pred$score(msr("classif.ce"))

  storage[i, "k"] <- k_values[i]
}

storage <- storage[rev(order(storage$k)), ]

plot(
  x = storage$k, y = storage$mmce_train, main = "Overfitting behavior KNN",
  xlab = "k (log scaled)", ylab = "mmce", col = "blue", type = "l",
  xlim = rev(range(storage$k)),
  ylim = c(
    min(storage$mmce_train, storage$mmce_test),
    max(storage$mmce_train, storage$mmce_test)
  ),
  log = "x"
)
lines(x = storage$k, y = storage$mmce_test, col = "orange")
legend("bottomleft", c("test", "train"), col = c("orange", "blue"), lty = 1)
```



⇒ The more complex our model (small k), the higher is the chance that the performance of the model on the training data (in blue) is much better than the performance on new *test data* (in orange).

How not to split: the good, the bad, the ugly

(How) Does the specific choice of splitting the data into train and test sets affect our estimation of the model performance?

The "good" split

We train on the first 30 data points and test on the next 20. Remember that `iris` is an ordered data set with the first 50 observations being "setosa", the next 50 "versicolor" and the last 50 "virginica".

```
library(mlr3)
library(mlr3learners)

task <- tsk("iris")
learner <- lrn("classif.kknn", k = 3)
learner$train(task = task, row_ids = 1:30)
pred <- learner$predict(task, row_ids = 31:50)

pred$score()
```

```
## classif.ce
##          0
```

```
pred$confusion
```

```
##             truth
## response     setosa versicolor virginica
##   setosa         20          0         0
##   versicolor      0          0         0
##   virginica       0          0         0
```

We get a MMCE of 0. Why should this not surprise us?

The "bad" split

We train on the first 100 data points and test on the next 50.

```
task <- tsk("iris")
learner <- lrn("classif.kknn", k = 3)
learner$train(task = task, row_ids = 1:100)
pred <- learner$predict(task, row_ids = 101:150)

pred$score()
```

```
## classif.ce
##          1
```

```
pred$confusion
```

```
##             truth
## response     setosa versicolor virginica
##   setosa          0          0         0
##   versicolor      0          0        50
##   virginica       0          0         0
```

Although we trained the learner, we get a test set MMCE of 1. The problem is that we never showed the learner training data with the "virginica" label, so the training data does not represent the actual population we want to classify. Try to formalize this in terms of the theory for supervised learners we have defined so far – which assumption about the (training and test) data is not fulfilled here?

In statistics, we would call this a particularly severe example of *selection bias*.

The ugly split

```
task <- tsk("iris")
learner <- lrn("classif.kknn", k = 3)
learner$train(task = task, row_ids = c(1:45, 51:95, 101:110))
pred <- learner$predict(task, row_ids = c(46:50, 96:100, 111:150))

pred$score()
```

```
## classif.ce
##       0.18
```

```
pred$confusion
```

```
##             truth
## response     setosa versicolor virginica
##   setosa          5          0         0
##   versicolor      0          5         9
##   virginica       0          0        31
```