Exercise 1:

Shortly answer the following questions:

(a) What is the difference between inner and outer loss?

(b) Which model is more likely to overfit the training data:

- k-NN with 1 or with 10 neighbours?
- Logistic regression with 10 or 20 features?
- LDA or QDA?

(c) Which of the following methods yield an unbiased generalization error estimate?
Performance estimation ...

- on training data
- on test data
- on training and test data combined
- using cross validation
- using subsampling

(d) Which problem does resampling of training and test data solve?

(e) Which problem does nested resampling solve?

Exercise 2:

The Satellite dataset consists of pixels in 3x3 neighbourhoods in a satellite image, where each pixel is described by 4 spectral values, and the classification label of the central pixel. (for further information see `?Sattelite`) We fit a k-NN model to predict the class of the middle pixel. The performance is evaluated with the mmce.
Look at the following R code and output: The performance is estimated in different ways: using training data, test data and then with cross validation. How do the estimates differ and why? Which one should be used?

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

data(Satellite)
satellite_task <-
  TaskClassif$new(id = "satellite_task",
                  backend = Satellite,
                  target = "classes")
knn_learner <- lrn("classif.kknn", k = 3)

# Train and test subsets:
set.seed(42)
train_indices <-
  sample.int(nrow(Satellite), size = 0.8 * nrow(Satellite))
test_indices <- setdiff(1:nrow(Satellite), train_indices)
```

```r
# Training data performance estimate
knn_learner$train(task = satellite_task, row_ids = train_indices)
pred <-
  knn_learner$predict(task = satellite_task, row_ids = train_indices)
pred$score()
```

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

```r
# Test data performance estimate
pred <-
  knn_learner$predict(task = satellite_task, row_ids = test_indices)
pred$score()
```

```
## classif.ce
## 0.09246309
```

```r
# CV performance estimate
rdesc <- rsmp("cv", folds = 10)
res <- resample(satellite_task, knn_learner, rdesc)
```

```
## INFO  [08:24:16.612] Applying learner 'classif.kknn' on task 'satellite_task' (iter 1/10)
## INFO  [08:24:16.770] Applying learner 'classif.kknn' on task 'satellite_task' (iter 2/10)
## INFO  [08:24:16.920] Applying learner 'classif.kknn' on task 'satellite_task' (iter 3/10)
## INFO  [08:24:17.225] Applying learner 'classif.kknn' on task 'satellite_task' (iter 4/10)
## INFO  [08:24:17.379] Applying learner 'classif.kknn' on task 'satellite_task' (iter 5/10)
## INFO  [08:24:17.514] Applying learner 'classif.kknn' on task 'satellite_task' (iter 6/10)
## INFO  [08:24:17.806] Applying learner 'classif.kknn' on task 'satellite_task' (iter 7/10)
## INFO  [08:24:17.941] Applying learner 'classif.kknn' on task 'satellite_task' (iter 8/10)
## INFO  [08:24:18.085] Applying learner 'classif.kknn' on task 'satellite_task' (iter 9/10)
## INFO  [08:24:18.230] Applying learner 'classif.kknn' on task 'satellite_task' (iter 10/10)
```

```r
res$score()
```

```
##              task        task_id              learner   learner_id
##  1: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  2: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  3: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  4: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  5: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  6: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  7: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  8: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##  9: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
## 10: <TaskClassif> satellite_task <LearnerClassifKKNN> classif.kknn
##         resampling resampling_id iteration prediction classif.ce
##  1: <ResamplingCV>            cv         1     <list> 0.09627329
##  2: <ResamplingCV>            cv         2     <list> 0.07919255
##  3: <ResamplingCV>            cv         3     <list> 0.08540373
##  4: <ResamplingCV>            cv         4     <list> 0.09472050
##  5: <ResamplingCV>            cv         5     <list> 0.10559006
##  6: <ResamplingCV>            cv         6     <list> 0.08553655
##  7: <ResamplingCV>            cv         7     <list> 0.08864697
##  8: <ResamplingCV>            cv         8     <list> 0.10108865
##  9: <ResamplingCV>            cv         9     <list> 0.08864697
## 10: <ResamplingCV>            cv        10     <list> 0.10419907
```

```
res$aggregate()

## classif.ce
## 0.09292983
```

Exercise 3:

In preparing this course you already learned about `mlr3`. If you need to refresh your knowledge you can find help at `https://mlr3book.mlr-org.com/` under 'Basics'.

a) How many performance measures do you already know? Try to explain some of them. How can you see which of them are available in `mlr3`?

b) Use the `boston_housing` regression task from `mlr3` and split the data into 50 % training data and 50 % test data while training and predicting (i. e., use the `row_ids` argument of the `train` and `predict` function). Fit a prediction model (e. g. k-NN) to the training set and make predictions for the test set.

c) Compare the performance on training and test data. Use the `score` function.

d) Now use different observations (but still 50 % of them) for the training set. How does this affect the predictions and the error estimates of the test data?

e) Use 10 fold cross-validation to estimate the performance. Hint: Use the mlr functions `rsmp` and `resample`.

Exercise 4:

Given are the results of a scoring algorithm and the associated *true* classes of 10 observations:

ID	Actual Class	Score
1	0	0.33
2	0	0.27
3	1	0.11
4	1	0.38
5	1	0.17
6	0	0.63
7	1	0.62
8	1	0.33
9	0	0.15
10	0	0.57

a) Create a confusion matrix assuming the decision boundary at 0.5.

b) Calculate: precision, sensitivity, negative predictive value, specificity, accuracy, error rate and F-measure.

c) Draw the ROC curve and interpret it. Feel free to use R for the drawing.

d) Calculate the AUC.