Introduction to Machine Learning

All exercises

https://slds-lmu.github.io/i2ml/

Winterterm 2022/2023

Contents

Exercise 1: Car Price Prediction

Imagine you work at a second-hand car dealer and are tasked with finding for-sale vehicles your company can acquire at a reasonable price. You decide to address this challenge in a data-driven manner and develop a model that predicts adequate market prices (in EUR) from vehicles' properties.

a) Characterize the task at hand: supervised or unsupervised? Regression or classification? Learning to explain or learning to predict? Justify your answers.

b) How would you set up your data? Name potential features along with their respective data type and state the target variable.

c) Assume now that you have data on vehicles' age (days), mileage (km), and price (EUR). Explicitly define the feature space $\mathcal{X}$ and target space $\mathcal{Y}$.

d) You choose to use a linear model (LM) for this task. For this, you assume the targets to be conditionally independent given the features, i.e., $y^{(i)}|\mathbf{x}^{(i)} \in y^{(j)}|\mathbf{x}^{(j)}$ for all $i, j \in \{1, 2, \ldots, n\}, i \neq j$, with sample size n. The LM models the target as a linear function of the features with Gaussian error term: $\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \epsilon$, $\epsilon \sim N(\mathbf{0}, diag(\sigma^2)), \quad \sigma > 0$.

   State the hypothesis space for the corresponding model class. For this, assume the parameter vector $\boldsymbol{\theta}$ to include the intercept coefficient.

e) Which parameters need to be learned? Define the corresponding parameter space Θ.

f) State the loss function for the i-th observation using $L2$ loss.

g) In classical statistics, you would estimate the parameters via maximum likelihood estimation (MLE). The likelihood for the LM is given by:

$$\mathcal{L}(\boldsymbol{\theta}|\mathbf{x}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left(y^{(i)} - \boldsymbol{\theta}^T\mathbf{x}^{(i)}\right)^2\right)$$

   Describe how you can make use of the likelihood in empirical risk minimization (ERM) and write down the resulting empirical risk.

h) Now you need to optimize this risk to find the best parameters, and hence the best model, via empirical risk minimization. State the optimization problem formally and list the necessary steps to solve it.

Congratulations, you just designed your first machine learning project!

Exercise 1: HRO in `mlr3`

Throughout the lecture, we will frequently use the `R` package `mlr3` and its descendants, providing an integrated ecosystem for all common machine learning tasks. Let's recap the HRO principle and see how it is reflected in `mlr3`. An overview of the most important objects and their usage, illustrated with numerous examples, can be found at `https://mlr3book.mlr-org.com/basics.html`.

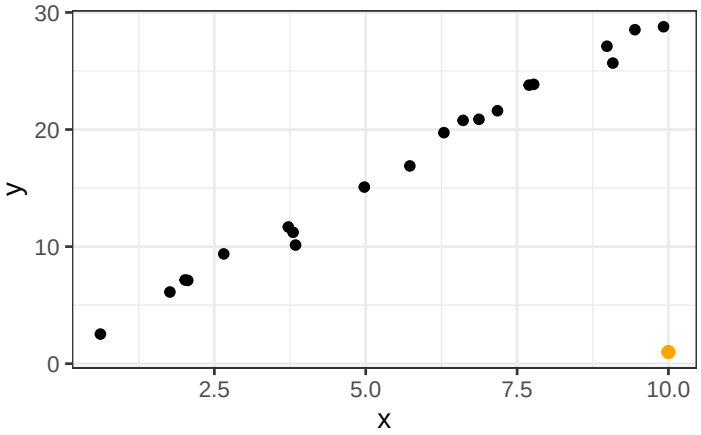
a) How are the key concepts (i.e., hypothesis space, risk and optimization) you learned about in the lecture videos implemented in `mlr3`?

b) Have a look at `mlr3::tsk("iris")`. What attributes does this `task` object store?

c) Pick an `mlr3` learner of your choice. What are the different settings for this learner?
(Hint: use `mlr3::mlr_learners$keys()` to see all available learners.)

Exercise 2: Loss Functions for Regression Tasks

In this exercise, we will examine loss functions for regression tasks somewhat more in depth.

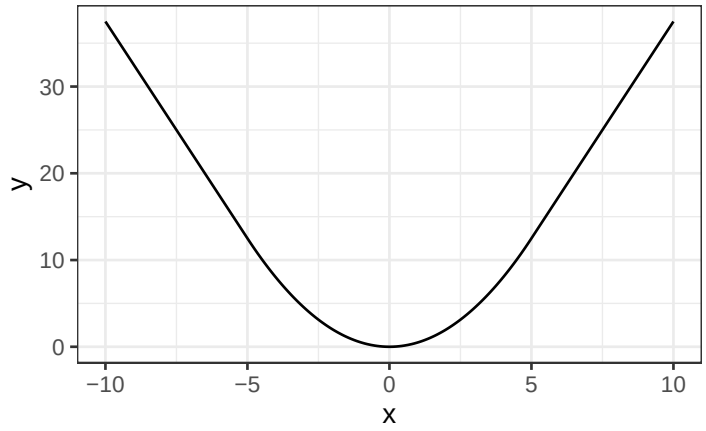


a) Consider the above linear regression task. How will the model parameters be affected by adding the new outlier point (orange) if you use

   i) $L1$ loss

   ii) $L2$ loss

   in the empirical risk? (You do not need to actually compute the parameter values.)



b) The second plot visualizes another loss function popular in regression tasks, the so-called *Huber loss* (depending on $\epsilon > 0$; here: $\epsilon = 5$). Describe how the Huber loss deals with residuals as compared to $L1$ and $L2$ loss. Can you guess its definition?

c) Derive the least-squares estimator, i.e., the solution to the linear model when using $L2$ loss, analytically via

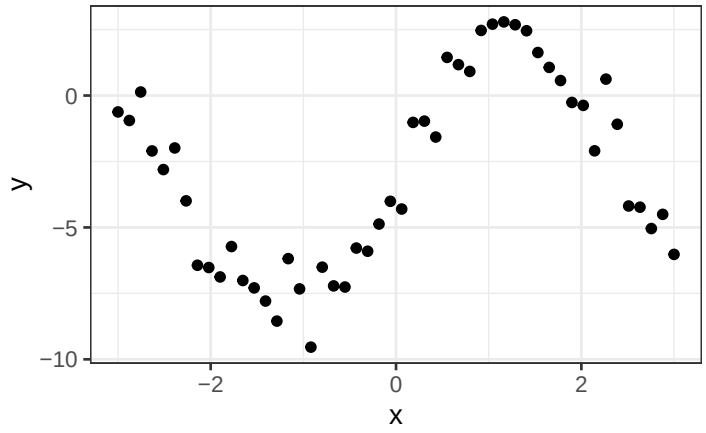
$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \Theta} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2.$$

Exercise 3: Polynomial Regression

Assume the following (noisy) data-generating process from which we have observed 50 realizations:

$$y = -3 + 5 \cdot \sin(0.4\pi x) + \epsilon$$

with $\epsilon \sim \mathcal{N}(0, 1)$.



a) We decide to model the data with a cubic polynomial (including intercept term). State the corresponding hypothesis space.

b) Demonstrate that this hypothesis space is simply a parameterized family of curves by plotting in `R` curves for 3 different models belonging to the considered model class.

c) State the empirical risk w.r.t. $\boldsymbol{\theta}$ for a member of the hypothesis space. Use $L2$ loss and be as explicit as possible.

d) We can minimize this risk using gradient descent. In order to make this somewhat easier, we will denote the transformed feature matrix, containing x to the power from 0 to 3, by $\tilde{\mathbf{X}}$, such that we can express our model by $\tilde{\mathbf{X}}\boldsymbol{\theta}$ (note that the model is still linear in its parameters, even if $\mathbf{X}$ has been transformed in a non-linear manner!). Derive the gradient of the empirical risk w.r.t $\boldsymbol{\theta}$.

e) Using the result from d), state the calculation to update the current parameter $\boldsymbol{\theta}^{[t]}$.

f) You will not be able to fit the data perfectly with a cubic polynomial. Describe the advantages and disadvantages that a more flexible model class would have. Would you opt for a more flexible learner?

Exercise 4: Predicting `abalone`

We want to predict the age of an abalone using its longest shell measurement and its weight.

See `https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/` for more details.

```
url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.data"
abalone <- read.table(url, sep = ",", row.names = NULL)
colnames(abalone) <- c(
  "sex", "longest_shell", "diameter", "height", "whole_weight",
  "shucked_weight", "visceral_weight", "shell_weight", "rings")
abalone <- abalone[, c("longest_shell", "whole_weight", "rings")]
```

a) Plot `LongestShell` and `WholeWeight` on the x- and y-axis, respectively, and color points according to `Rings`.

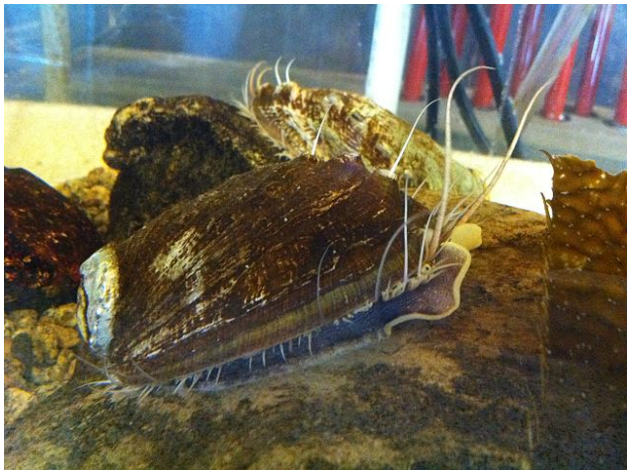
Using `mlr3`:

b) Create an `mlr3` task for the `abalone` data.

c) Define a linear regression learner (for this you will need to load the `mlr3learners` extension package first) and use it to train a linear model on the `abalone` data.

d) Compare the fitted and observed targets visually.
(Hint: use `autoplot()`.)

e) Assess the model's training loss in terms of MAE.
(Hint: losses are retrieved by calling `$score()`, which accepts different `mlr_measures`, on the prediction object.)



https://en.wikipedia.org/wiki/Abalone#/media/File:LivingAbalone.JPG

Exercise 1: Logistic Regression Basics

a) What is the relationship between softmax

$$\pi_k(\mathbf{x} \mid \boldsymbol{\theta}) = \frac{\exp(\boldsymbol{\theta}_k^\top \mathbf{x})}{\sum_{j=1}^{g} \exp(\boldsymbol{\theta}_j^\top \mathbf{x})}, \quad k \in \{1, \ldots, g\}$$

and the logistic function

$$\pi(\mathbf{x} \mid \boldsymbol{\theta}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})}$$

for $g = 2$ (binary classification)?

b) The likelihood function for a multinomially distributed target variable with g target classes is given by[1]

$$\mathcal{L}_i(\boldsymbol{\theta}) = \mathbb{P}(y^{(i)} | \mathbf{x}^{(i)}, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_g) = \prod_{j=1}^{g} \pi_j \left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)^{\mathbb{I}(y^{(i)}=j)}$$

where the posterior class probabilities $\pi_1\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right), \pi_2\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right), \ldots, \pi_g\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)$ are modeled with softmax regression. Derive the likelihood function for n independent observations.

c) We have already addressed the connection that holds between maximum likelihood estimation and empirical risk minimization. Transform the joint likelihood function into an empirical risk function.

Hints:

- By following the maximum likelihood principle, we should look for parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_g$ that maximize the likelihood function.
- The expressions $\prod \mathcal{L}_i$ and $\log \prod \mathcal{L}_i$, if defined, are maximized by the same parameters.
- Minimizing a scalar function multiplied with -1 is equivalent to maximizing the original function.

State the associated risk function.

d) Write down the discriminant functions of multiclass logistic regression resulting from this minimization objective. How do we arrive at the final prediction?

e) State the parameter space Θ and corresponding hypothesis space $\mathcal{H}$ for the multiclass case.

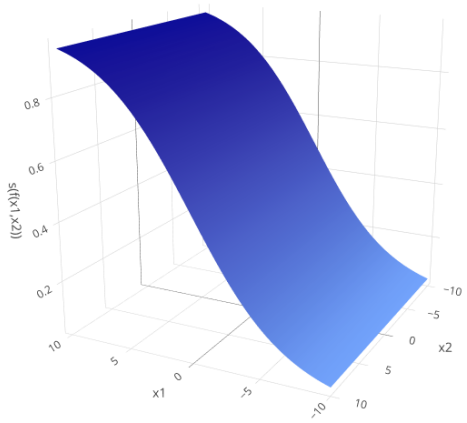
Exercise 2: Decision Boundaries & Thresholds in Logistic Regression

In logistic regression (binary case), we estimate the probability $\mathbb{P}(y = 1 \mid \mathbf{x}, \boldsymbol{\theta}) = \pi(\mathbf{x} \mid \boldsymbol{\theta})$. In order to decide about the class of an observation, we set $\hat{y} = 1$ iff $\hat{\pi}(\mathbf{x} \mid \hat{\boldsymbol{\theta}}) \geq \alpha$ for some $\alpha \in (0, 1)$.

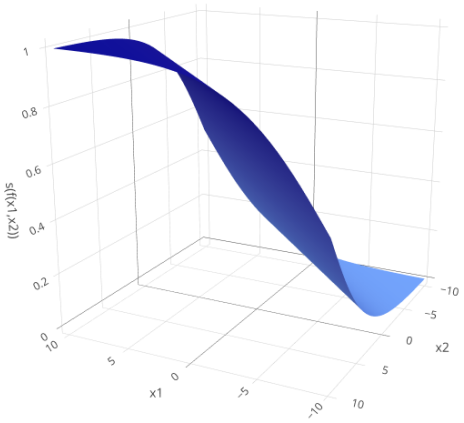
a) Show that the decision boundary of the logistic classifier is a (linear!) hyperplane.
Hint: derive the value of $\hat{\boldsymbol{\theta}}^T \mathbf{x}$ (depending on α) starting from which you predict $\hat{y} = 1$ rather than $\hat{y} = 0$.

[1] While this might look somewhat complicated, it is actually just a very concise way to express the multinomial likelihood: for each observation, all factors but the one corresponding to the true class j' will be 1 (due to the 0 exponent), so the result is simply $\pi_{j'}\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)$.

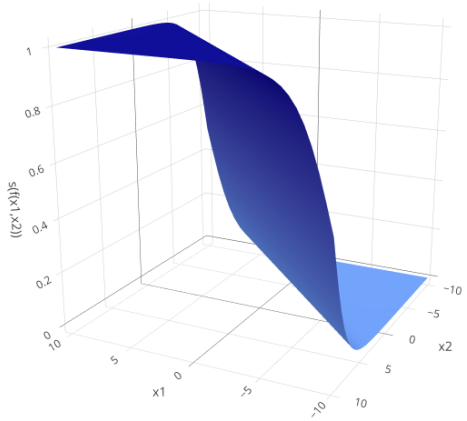
b) Below you see the logistic function for a binary classification problem with two input features for different values $\hat{\boldsymbol{\theta}} = (\hat{\theta}_1, \hat{\theta}_2)$ (plots 1-3) as well as α (plot 4). What can you deduce for the values of $\hat{\theta}_1$, $\hat{\theta}_2$ and α? What are the implications for classification in the different scenarios?



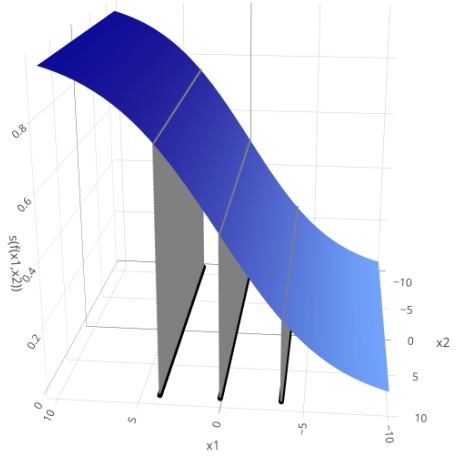
Plot (1)



Plot (2)



Plot (3)



Plot (4)

c) Derive the equation for the decision boundary hyperplane if we choose $\alpha = 0.5$.

d) Explain when it might be sensible to set α to 0.5.

Exercise 1: Naive Bayes

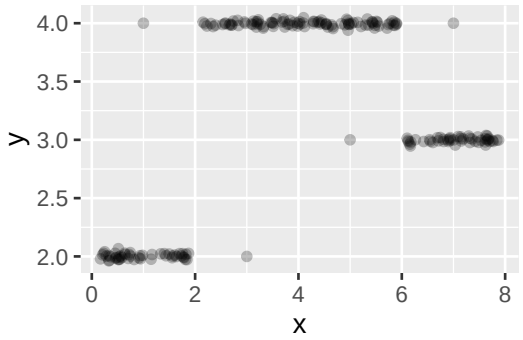
You are given the following table with the target variable `Banana`:

ID	Color	Form	Origin	Banana
1	yellow	oblong	imported	yes
2	yellow	round	domestic	no
3	yellow	oblong	imported	no
4	brown	oblong	imported	yes
5	brown	round	domestic	no
6	green	round	imported	yes
7	green	oblong	domestic	no
8	red	round	imported	no

a) We want to use a Naive Bayes classifier to predict whether a new fruit is a `Banana` or not. Estimate the posterior probability $\hat{\pi}(\mathbf{x}_*)$ for a new observation $\mathbf{x}_* = (\text{yellow}, \text{round}, \text{imported})$. How would you classify the object?

b) Assume you have an additional feature `Length` that measures the length in cm. Describe in 1-2 sentences how you would handle this numeric feature with Naive Bayes.

Exercise 2: Discriminant Analysis



The above plot shows $\mathcal{D} = \left(\left(\mathbf{x}^{(1)}, y^{(1)}\right), \ldots, \left(\mathbf{x}^{(n)}, y^{(n)}\right)\right)$, a data set with $n = 200$ observations of a continuous target variable y and a continuous, 1-dimensional feature variable $\mathbf{x}$. In the following, we aim at predicting y with a machine learning model that takes $\mathbf{x}$ as input.

a) To prepare the data for classification, we categorize the target variable y in 3 classes and call the transformed target variable z, as follows:

$$z^{(i)} = \begin{cases} 1, & y^{(i)} \in (-\infty, 2.5] \\ 2, & y^{(i)} \in (2.5, 3.5] \\ 3, & y^{(i)} \in (3.5, \infty) \end{cases}$$

Now we can apply quadratic discriminant analysis (QDA):

i) Estimate the class means $\mu_k = \mathbb{E}(\mathbf{x}|z = k)$ for each of the three classes $k \in \{1, 2, 3\}$ visually from the plot. Do not overcomplicate this, a rough estimate is sufficient here.

ii) Make a plot that visualizes the different estimated densities per class.

iii) How would your plot from ii) change if we used linear discriminant analysis (LDA) instead of QDA? Explain your answer.

iv) Why is QDA preferable over LDA for this data?

b) Given are two new observations $\mathbf{x}_{*1} = -10$ and $\mathbf{x}_{*2} = 7$. State the prediction for QDA and explain how you arrive there.

Exercise 3: Decision Boundaries for `mlr3` Learners

We will now visualize how well different learners classify the three-class `mlbench::mlbench.cassini` data set. Generate 1000 points from `cassini`, perturb the `x.2` dimension with Gaussian noise (mean 0, standard deviation 0.5), and consider the classifiers already introduced in the lecture:

- LDA,
- QDA, and
- Naive Bayes.

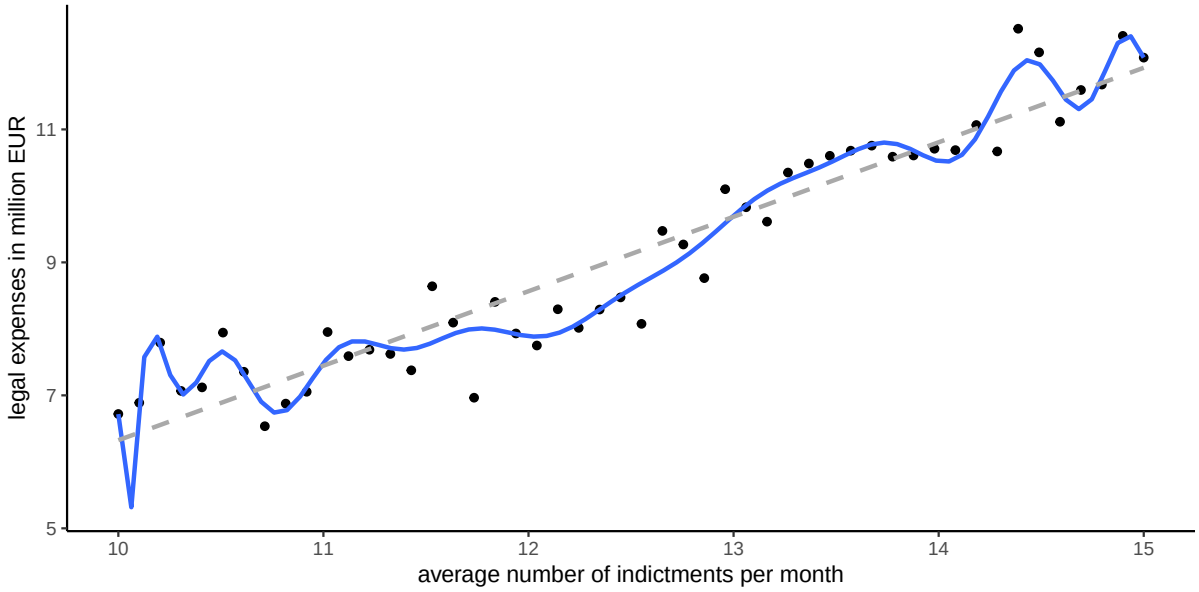
Plot the learners' decision boundaries. Can you spot differences in separation ability?

(Note that logistic regression cannot handle more than two classes and is therefore not listed here.)

Exercise 1: Evaluating regression learners

Imagine you work for a data science start-up and sell turn-key statistical models. Based on a set of training data, you develop a regression model to predict a customer's legal expenses from the average monthly number of indictments brought against their firm.

a) Due to the financial sensitivity of the situation, you opt for a very flexible learner that fits the customer's data ($n_{\text{train}} = 50$ observations) well, and end up with a degree-21 polynomial (blue, solid). Your colleague is skeptical and argues for a much simpler linear learner (gray, dashed). Which of the models will have a lower empirical risk if standard $L2$ loss is used?



b) Why might evaluation based on training error not be a good idea here?

c) Evaluate both learners on the following test data ($n_{\text{test}} = 10$), using

   i) mean squared error (MSE), and

   ii) mean absolute error (MAE).

   State your performance assessment and explain potential differences.

   (Hint: use `R` if you don't feel like computing a degree-21 polynomial regression by hand.)

```r
set.seed(123)
x_train <- seq(10, 15, length.out = 50)
y_train <- 10 + 3 * sin(0.15 * pi * x_train) + rnorm(length(x_train), sd = 0.5)
data_train <- data.frame(x = x_train, y = y_train)

set.seed(321)
x_test <- seq(10, 15, length.out = 10)
y_test <- 10 + 3 * sin(0.15 * pi * x_test) + rnorm(length(x_test), sd = 0.5)
data_test <- data.frame(x = x_test, y = y_test)
```

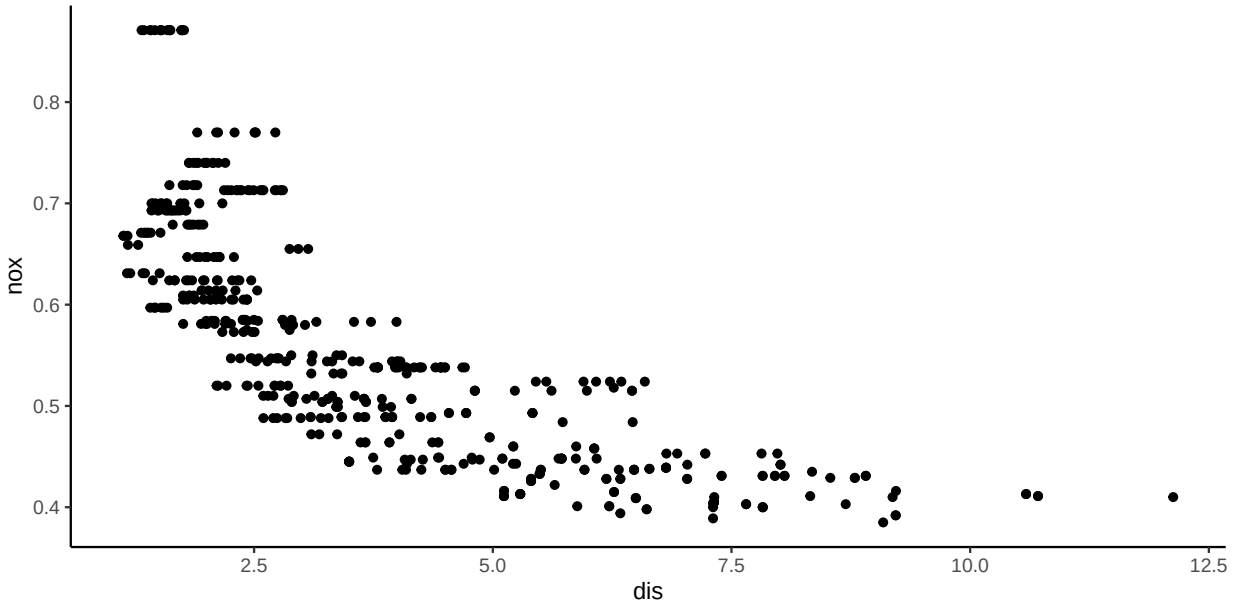
Exercise 2: Importance of train-test split

We consider the `BostonHousing` data for which we would like to predict the nitric oxides concentration (`nox`) from the distance to a number of firms (`dis`).

```
library(mlbench)
data(BostonHousing)
data_pollution <- data.frame(dis = BostonHousing$dis, nox = BostonHousing$nox)
data_pollution <- data_pollution[order(data_pollution$dis), ]
head(data_pollution)

##        dis   nox
## 373 1.1296 0.668
## 375 1.1370 0.668
## 372 1.1691 0.631
## 374 1.1742 0.668
## 407 1.1781 0.659
## 371 1.2024 0.631

ggplot2::ggplot(data_pollution, ggplot2::aes(x = dis, y = nox)) +
  ggplot2::geom_point() +
  ggplot2::theme_classic()
```



a) Use the first ten observations as training data to compute a linear model with `mlr3` and evaluate the performance of your learner on the remaining data using MSE.

b) What might be disadvantageous about the train-test split in a)?

c) Now, sample your training observations from the data set at random. Use a share of 0.1 through 0.9, in 0.1 steps, of observations for training and repeat this procedure ten times. Afterwards, plot the resulting test errors (in terms of MSE) in a suitable manner.
(Hint: `rsmp` is a convenient function for splitting data – you will want to choose the "holdout" strategy. Afterwards, `resample` can be used to repeatedly fit the learner.)

d) Interpret the findings from c).

Exercise 1: Overfitting & underfitting

Assume a polynomial regression model with a continuous target variable y and a continuous, p-dimensional feature vector $\mathbf{x}$ and polynomials of degree d, i.e.,

$$f\left(\mathbf{x}^{(i)}\right) = \sum_{j=1}^{p} \sum_{k=0}^{d} \theta_{j,k} (\mathbf{x}_j^{(i)})^k,$$

and $y^{(i)} = f\left(\mathbf{x}^{(i)}\right) + \epsilon^{(i)}$ where the $\epsilon^{(i)}$ are iid with $\mathsf{Var}(\epsilon^{(i)}) = \sigma^2 \; \forall i \in \{1, \ldots, n\}$.

a) For each of the following situations, indicate whether we would generally expect the performance of a flexible polynomial learner (high d) to be better or worse than an inflexible one (low d). Justify your answer.

   (i) The sample size n is extremely large, and the number of features p is small.

   (ii) The number of features p is extremely large, and the number of observations n is small.

   (iii) The true relationship between the features and the response is highly non-linear.

   (iv) The variance of the error terms, σ^2, is extremely high.

b) Are overfitting and underfitting properties of a learner or of a fixed model? Explain your answer.

c) Should we aim to completely avoid both overfitting and underfitting?

Exercise 2: Resampling strategies

a) Why would we apply resampling rather than a single holdout split?

b) Using `mlr3`, classify the `german_credit` data into solvent and insolvent debtors using logistic regression. Compute the training error w.r.t. MCE.

c) In order to evaluate your learner, compare test MCE using

   i) three times ten-fold cross validation (3x10-CV)

   ii) 10x3-CV

   iii) 3x10-CV with stratification for the feature `foreign_worker` to ensure equal representation in all folds

   iv) a single holdout split with 90% training data

   (Hint: you will need `rsmp`, `resample` and `aggregate`.)

d) Discuss and compare your findings from c) and compare them to the training error from b).

e) Would you consider LOO-CV to be a good alternative?

Exercise 1: ROC metrics

Consider a binary classification algorithm that yielded the following results on 10 observations. The table shows true classes and predicted probabilities for class 1:

ID	True class	Prediction
1	0	0.33
2	0	0.27
3	0	0.11
4	1	0.38
5	1	0.17
6	1	0.63
7	1	0.62
8	1	0.33
9	0	0.15
10	0	0.57

a) Create a confusion matrix assuming a threshold of 0.5. Point out which values correspond to true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

b) Calculate: PPV, NPV, TPR, FPR, ACC, MCE and $F1$ measure.

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

d) Calculate the AUC.

e) How would the ROC curve change if you had chosen a different threshold in a)?

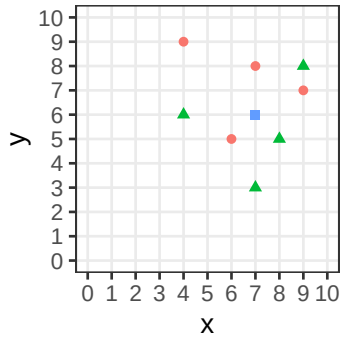
Exercise 2: k-NN

a) Let the two-dimensional feature vectors in the following figure be instances of two different classes (triangles and circles). Classify the point (7, 6) – represented by a square in the picture – with a k-NN classifier using $L1$ norm (Manhattan distance):

$$d_{\text{Manhattan}}(\mathbf{x}, \tilde{\mathbf{x}}) = \sum_{j=1}^{p} |x_j - \tilde{x}_j|.$$

As a decision rule, use the unweighted number of the individual classes in the k-neighborhood, i.e., assign the point to the class that represents most neighbors.

i) $k = 3$
ii) $k = 5$
iii) $k = 7$



b) Now consider the same constellation but assume a regression problem this time, where the circle-shaped points have a target value of 2 and the triangles have a value of 4.

Again, predict for the square point (7, 9), using both the *unweighted* and the *weighted* mean in the neighborhood (still with Manhattan distance).

i) $k = 3$

ii) $k = 5$

iii) $k = 7$

Exercise 1: Splitting criteria

Given are the data set

x	1.0	2.0	7.0	10.0	20.0
y	1.0	1.0	0.5	10.0	11.0

and the same with log-transformed feature x:

$\log x$	0.0	0.7	1.9	2.3	3.0
y	1.0	1.0	0.5	10.0	11.0

a) Compute the first split point the CART algorithm would find for each data set (with pen and paper or in `R`).

b) State the optimal constant predictor for a node $\mathcal{N}$ when minimizing the empirical risk under $L2$ loss and explain why this is equivalent to minimizing "variance impurity".

Exercise 2: Impurity reduction

The fractions of the classes $k = 1, \ldots, g$ in node $\mathcal{N}$ of a decision tree are $\pi_1^{(\mathcal{N})}, \ldots, \pi_g^{(\mathcal{N})}$. Assume we replace the classification rule in node $\mathcal{N}$

$$\hat{k} \mid \mathcal{N} = \arg\max_k \pi_k^{(\mathcal{N})}$$

with a randomizing rule

$$\hat{k} \sim \text{Cat}\left(\pi_1^{(\mathcal{N})}, \ldots, \pi_g^{(\mathcal{N})}\right),$$

in which we draw the classes in one node from the categorical distribution of their estimated probabilities (i.e., class k is predicted with probability $\pi_k^{(\mathcal{N})}$).

Compute the expected MCE in node $\mathcal{N}$ for data distributed i.i.d. like the training data. What do you notice?

(*Hint*: The observations and the predictions using the randomizing rule follow the same distribution.)

Exercise 1: Bagging

In this exercise, we briefly revisit why bagging is a useful technique to stabilize predictions.

For a fixed observation $(\mathbf{x}, y)$, show that the quadratic loss of the ensemble prediction $f^{[M]}(\mathbf{x})$ is less than or equal to the average quadratic loss over individual base learner predictions $b^{[m]}(\mathbf{x})$. You can assume an infinite theoretical ensemble and use $\mathbb{E}_{\mathcal{M}}$ to denote the expectation over its members.

Exercise 2: Classifying `spam`

a) Take a look at the `spam` dataset (`?mlr3::mlr_tasks_spam`). Shortly describe what kind of classification problem this is and access the corresponding task predefined in `mlr3`.

b) Use a decision tree to predict `spam`. Re-fit the tree using two random subsets of the data (each comprising 60% of observations). How stable are the trees?

(Hint: Use `rpart.plot()` from the package `rpart.plot` to visualize the trees.)

c) Forests come with a built-in estimate of their generalization ability via the out-of-bag (OOB) error.

i) Show that the probability for an observation to be OOB in an arbitrary bootstrap sample converges to $\frac{1}{e}$.

ii) Use the random forest learner `classif.ranger` to fit the model and state the out-of-bag (OOB) error.

d) You are interested in which variables have the greatest influence on the prediction quality. Explain how to determine this in a permutation-based approach and compute the importance scorses for the `spam` data.

(Hint: use an adequate variable importance filter as described in `https://mlr3filters.mlr-org.com/#variable-importance-filters`.)

Exercise 3: Proximities

You solve the `wine` task, predicting the `type` of a wine – with 3 classes – from a number of covariates. After training, you wish to determine how similar your observations are in terms of proximities.

For the following subset of the training data and the random forest model given below,

a) find the terminal node of each tree the observations are placed in,

b) compute the observations' pairwise proximities, and

c) construct a similarity matrix from these proximities.

Hint: The model information was created with `ranger::treeInfo()`, which assigns observations with values larger than `splitval` to the right child node in each split.

observation	alcalinity	alcohol	flavanoids	hue	malic	phenols
1	11.4	14.75	3.69	1.25	1.73	3.10
2	25.0	13.40	0.96	0.67	4.60	1.98
3	17.4	13.94	3.54	1.12	1.73	2.88

Tree 1:

nodeID	leftChild	rightChild	splitvarID	splitvarName	splitval	terminal	prediction
0	1	2	5	phenols	1.94	FALSE	NA
1	3	4	1	alcohol	12.43	FALSE	NA
2	5	6	1	alcohol	13.04	FALSE	NA
3	NA	NA	NA	NA	NA	TRUE	2
4	NA	NA	NA	NA	NA	TRUE	3
5	NA	NA	NA	NA	NA	TRUE	2
6	NA	NA	NA	NA	NA	TRUE	1

Tree 2:

nodeID	leftChild	rightChild	splitvarID	splitvarName	splitval	terminal	prediction
0	1	2	1	alcohol	12.78	FALSE	NA
1	3	4	3	hue	0.68	FALSE	NA
2	5	6	2	flavanoids	2.18	FALSE	NA
3	NA	NA	NA	NA	NA	TRUE	3
4	NA	NA	NA	NA	NA	TRUE	2
5	NA	NA	NA	NA	NA	TRUE	3
6	NA	NA	NA	NA	NA	TRUE	1

Tree 3:

nodeID	leftChild	rightChild	splitvarID	splitvarName	splitval	terminal	prediction
0	1	2	1	alcohol	12.79	FALSE	NA
1	3	4	5	phenols	2.01	FALSE	NA
2	5	6	5	phenols	2.28	FALSE	NA
3	NA	NA	NA	NA	NA	TRUE	2
4	NA	NA	NA	NA	NA	TRUE	2
5	NA	NA	NA	NA	NA	TRUE	3
6	NA	NA	NA	NA	NA	TRUE	1

Exercise 1: Tuning Principles

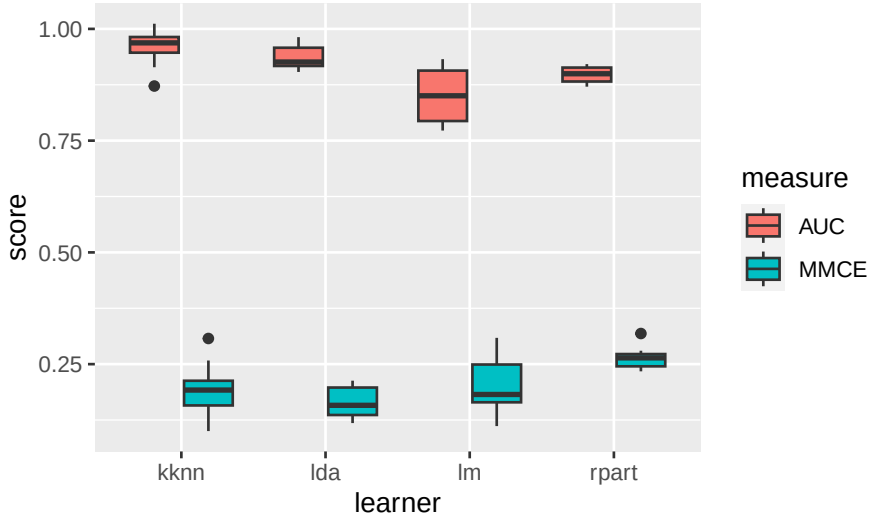
a) Suppose that we want to compare four different learners:

Learner	Tuning required
Logistic regression (`lm`)	no
CART (`rpart`)	yes
k-NN (`kknn`)	yes
LDA (`lda`)	no

For performance evaluation and subsequent comparison, we use 10-CV as outer resampling strategy. Within the inner tuning loop, applicable to CART and k-NN, we use 5-CV in combination with random search, drawing 200 hyperparameter configurations for each model. Our measure of interest is the AUC.

i) How many models need to be fitted in total to conduct the final benchmark?

ii) Giving the following benchmark result, which learner performs best? Explain your decision.



b) Recap briefly what is meant by the *bias-variance trade-off* in resampling.

c) Are the following statements true or not? Explain your answer in one sentence.

i) The bias of the generalization error estimate for 3-CV is higher than for 10-CV.

ii) Every outer loss can also be used as inner loss, assuming standard gradient-based optimization.

Exercise 2: AutoML with `mlr3`

In this exercise, we build a simple automated machine learning (AutoML) system that will make data-driven choices on which learner to use and also conduct the necessary tuning.

`mlr3pipelines` make this endeavor easy, modular and guarded against many common modeling errors.

We work on the `pima` data to classify patients as diabetic and design a system that is able to choose between k-NN and a random forest, both with tuned hyperparameters.

To this end, we will use a graph learner, a "single unit of data operation" that can be trained, resampled, evaluated, ... as a whole – in other words, treated as any other learner.

a) Create a task object in `mlr3` (the problem is pre-specified under the ID "pima").

b) Specify the above learners, where you need to give each learner a name as input to the `id` argument. Convert each learner to a pipe operator by wrapping them in the sugar function `po()`, and store them in a `list`.

c) Before starting the actual learning pipeline, take care of pre-processing. While this step is highly customizable, you can use an existing sequence to impute missing values, encode categorical features, and remove variables with constant value across all observations. For this, specify a pipeline (`ppl()`) of type `"robustify"` (setting `factors_to_numeric` to `TRUE`).

d) Create another `ppl`, of type `"branch"` this time, to enable selection between your learners.

e) Chain both pipelines using the double pipe and plot the resulting graph. Next, convert it into a graph learner with `as_learner()`.

f) Now you have a learner object just like any other. Take a look at its tunable hyperparameters. You will optimize the learner selection, the number of neighbors in k-NN (between 3 and 10), and the number of split candidates to try in the random forest (between 1 and 5). Define the search range for each like so:

```
<learner>$param_set$values$<hyperparameter> <- to_tune(p_int(lower, upper))
```

`p_int` marks an integer hyperparameter with lower and upper bounds as defined; similar objects exist for other data types. With `to_tune()`, you signal that the hyperparameter shall be optimized in the given range.

Hint: You need to define dependencies, since the tuning process is defined by which learner is selected in the first place (no need to tune k in a random forest).

g) Conveniently, there is a sugar function, `tune_nested()`, that takes care of nested resampling in one step. Use it to evaluate your tuned graph learner with

- mean classification error as inner loss,
- random search as tuning algorithm (allowing for 3 evaluations), and
- 3-CV in both inner and outer loop.

h) Lastly, extract performance estimates per outer fold (`score()`) and overall (`aggregate()`). If you want to risk a look under the hood, try `extract_inner_tuning_archives()`.

Congrats, you just designed a turn-key AutoML system that does (nearly) all the work with a few lines of code!

Exercise 3: Kaggle Challenge

Make yourself familiar with the Titanic Kaggle challenge (`https://www.kaggle.com/c/titanic`).

Based on everything you have learned in this course, do your best to achieve a good performance in the survival challenge.

- Try out different classifiers you have encountered during the course (or maybe even something new?)
- Improve the prediction by creating new features (feature engineering).
- Tune your parameters (see: `https://mlr3book.mlr-org.com/tuning.html`).
- How do you fare compared to the public leaderboard?

Hint: Use the `titanic` package to directly access the data. Use `titanic::titanic_train` for training and `titanic::titanic_test` for your final prediction.