Exercise 1: Linear models and GAMs

In this exercise we are looking at *generalized additive models* or *GAMs*, which are similar to linear models, but instead of terms only containing one feature itself, as in a simple linear model, the terms in GAMs can contain arbitrary smooth functions of one feature. This means that GAMs can model arbitrary non-linear relationships, but are still additive, meaning every summand only depends on one single feature. Therefore, they do not contain interactions.

Consider the following dataset with 11 observations and two features:

	1	2	3	4	5	6	7	8	9	10	11	$\sum_{i=1}^{n}$
y	-7.90	-6.08	-3.74	-1.18	-1.23	-0.55	0.05	0.88	4.74	2.93	2.55	-9.53
x_1	-1.00	-0.80	-0.60	-0.40	-0.20	0.00	0.20	0.40	0.60	0.80	1.00	0.00
x_2	0.95	0.65	0.40	0.07	0.06	0.02	0.02	0.14	0.34	0.60	0.98	4.23

where the last column corresponds to the sum of values of each row.

The following shows the output of an LM ($x_2 \sim x_1$, that means x_2 is modeled as a linear function of x_1) and a GAM ($x_2 \sim s(x_1)$, that means x_2 is modeled as some smooth function of x_1):

	LM			**GAM**		
Predictors	*Estimates*	*CI*	*p*	*Estimates*	*CI*	*p*
(Intercept)	0.38	0.12 – 0.65	**8.851e-03**	0.38	0.35 – 0.42	**3.196e-07**
x1	-0.01	-0.42 – 0.41	9.749e-01			
s(x1)						**2.542e-05**
Observations	11			11		
R^2 / R^2 adjusted	0.000 / -0.111			0.988		

The R^2-value for the GAM model is the adjusted one.

a) In case you do not know how the adjusted R^2 is defined and interpreted, look it up. Then interpret the R^2 and adjusted R^2 values for the two models.

b) What conclusions can you draw from the LM model output for the relationship between x_1 and x_2?

c) Considering the information provided by the GAM model: How can the previous statement about the relationship between x_1 and x_2 be extended?

Exercise 2:

Given some

- simple linear model,
- linear moel with interactions,
- logistic regression model,
- GAM ???,

compute (or mathematically derive?) the standard errors of the estimated coefficients and interpret them.

Exercise 3: Interpreting logistic regression for bike rentals

In this exercise, we want to analyze and interpret a logistic regression model which, for the bike rental data set, predicts whether a day shows high or low/medium bike-rental demand. You are given the bike rental data with the features `season`, `temp`, `hum`, `wind_speed`, and `days_since_2011`. For our task, a binary target variable y is created as follows:

- Class y=1: "high number of bike rentals", defined by $>$ 70% quantile (i.e., `cnt` > 5531),
- Class y=0: "low to medium number of bike rentals", meaning $\leq$ 70% quantile (i.e., `cnt` ≤ 5531).

For the first part of the exercise, we will have a look at the feature `season`. The following table shows the joint and marginal absolute frequencies of y and `season`.

	WINTER	SPRING	SUMMER	FALL	Σ
y=0	174.00	111.00	98.00	128.00	511.00
y=1	7.00	73.00	90.00	50.00	220.00
Σ	181.00	184.00	188.00	178.00	731.00

a) Calculate and interpret the odds of "high number of bike rentals" vs. "low to medium number of bike rentals" in winter ($\text{odds}_{\text{winter}}$).

b) Calculate and interpret the odds ratio of high vs. low number of bike rentals when `season` changes from winter to spring, from winter to summer, and from winter to fall.

c) We now fit a GLM on $y \sim$ `season`, i.e., with only this single feature. That means the GLM has the form

$$p = \sigma(\eta) = \frac{1}{1+e^{-\eta}} \qquad \eta \;=\; \beta_0 + \beta_{\texttt{SPRING}}\,\text{seasonSPRING} + \beta_{\texttt{SUMMER}}\,\text{seasonSUMMER} + \beta_{\texttt{FALL}}\,\text{seasonFALL}.$$

We could also write $\beta_{\texttt{WINTER}}$ for the intercept β_0. We receive the following output from the GLM:

	Estimate	Std. Error	Pr(>|z|)
(Intercept)	-3.2131	0.3854	0.0000
seasonSPRING	2.7941	0.4138	0.0000
seasonSUMMER	3.1280	0.4121	0.0000
seasonFALL	2.2731	0.4199	0.0000

Interpret the β-estimates for the intercept and the different seasons in terms of the odds and the odds ratio.

d) Next, we fit another (somewhat "orthogonal") GLM model which considers the other four features except season. For this model, we want to look at a different way of interpretation, which was already briefly discussed in the lecture.

The model with four predictors then has the form:

$$\eta \;=\; \beta_0 + \beta_1\,\text{temp} + \beta_2\,\text{hum} + \beta_3\,\text{wind\_speed} + \beta_4\,\text{days\_since\_2011}, \qquad p = \sigma(\eta) = \frac{1}{1+e^{-\eta}}.$$

The fitted model is

Predictor	Symbol	Coefficient
Temperature (°C)	x_1	$\beta_1 = 0.29$
Humidity (%)	x_2	$\beta_2 = -0.0627$
Wind_speed (km h^{-1})	x_3	$\beta_3 = -0.0925$
Days since 01-Jan-2011	x_4	$\beta_4 = 0.0166$
Intercept	—	$\beta_0 = -8.52$

We want to calculate the effect of the first feature, `temperature`, on the probability of a day seeing a high number of bike rentals. This time, we will not look at the odds but at the probability directly, using the first derivative of the model.

(i) **Offset.** First, hold x_2, x_3, x_4 at their means and compute the offset for x_1, defined by $\delta = \beta_2\bar{x}_2 + \beta_3\bar{x}_3 + \beta_4\bar{x}_4$.
The sample means of the non-temperature features are:

$$\bar{x}_2 = 62.79, \qquad \bar{x}_3 = 12.76, \qquad \bar{x}_4 = 365.00.$$

(ii) **Probability table.** Using the reduced model $\eta(x_1) = \beta_0 + \delta + \beta_1 x_1$, which only takes into account the effect of the temperature, calculate the probabilities $p(x_1) = \sigma(\eta(x_1))$ for the remaining values of x_1 in the following table:

x_1 (°C)	$\eta(x_1)$	$p(x_1)$
10	-4.677	0.009
15	-3.227	0.038
20	-1.777	?
25	-0.327	?
30	?	?
35	?	?

(iii) **Derive the marginal effect.** Show step by step that $\dfrac{dp(x_1)}{dx_1} = p(1-p)\,\beta_1$.
This was already briefly discussed in the lecture, here we want to complete the proof. Also show why it does not matter whether we use the original model or the reduced model in this step.

(iv) **Evaluate dp/dx_1.** Compute dp/dx_1 at $x_1 = 15, 30, 35$ °C. Where is the temperature effect the largest?

(v) **Classification.** With a 0.5 threshold, which of the six temperatures in the table above are predicted "high-rental"?

e) Last, we compare the above results with the full model containing all features:

	Estimate	Std. Error	Pr(>|z|)
(Intercept)	-8.5176	1.2066	0.0000
seasonSPRING	1.7427	0.5977	0.0035
seasonSUMMER	-0.8566	0.7660	0.2635
seasonFALL	-0.6417	0.5543	0.2470
temp	0.2902	0.0391	0.0000
hum	-0.0627	0.0124	0.0000
wind_speed	-0.0925	0.0305	0.0024
days_since_2011	0.0166	0.0014	0.0000

Again, look at your interpretations of the β-coefficients and of the effects of single features from parts c) and d). What changes now in the full model?

Exercise 4: Improving CART decision trees

This exercise is about implementing an efficient algorithm for finding the splits in the CART algorithm. Recall that in CART, for every node, every possible split value for every possible feature is considered and the corresponding loss reduction calculated, in order to greedily find the optimal split in the current node. Alternatively, a quantile grid is used for every possible feature.

In this task, we consider regression problems and use the standard split criterion of impurity inside a node, in other words, the total variance within each node. Since we use a constant estimator inside each node, this is equivalent to minimizing the L_2-loss. Therefore, in order to calculate the loss reduction for some potential split, we have to evaluate the variance inside both potential child nodes. Since calculating the variance of some data set takes linear computation time, we end up with a total computation time of the order of $d \cdot n^2$ to compute one split. Here, d denotes the total number of features and n the number of data points inside the node to be split.

The idea to speed up this computation is to order the data points with respect to the feature of interest as well as the potential split values for this feature, and then to compute the potential loss reductions for the single split values in this order. We additionally keep track of the sum of target values and the sum of squares left and right to the current split value. In this way, we can simply update each of these sums by one summand when moving to the next potential split value. The loss reduction can then be calculated each time using these sums. This reduces the total computation time to an order of $d \cdot n$.

Your task in this exercise is to implement this more efficient algorithm.

We provide you with an implementation of the naive CART splitting algorithm for a single feature in the files "*CART.R*" and "*CART.py*" (depending on your programming language) on the Moodle website. The code also contains a function which uses some splitting algorithm to find the optimal split, i.e., the optimal feature and value. You should now complete the implementation of the function `search_split_fast`, by implementing the alternative algorithm described above.

On Moodle, you can also find the files "*CART_test.R*" and "*CART_test.py*", which you can use to test these different algorithms on some artificial data set. In the end, your implementation should produce the same results as the existing naive implementation or the reference implementations from the respective programming languages (from the rpart or the scikit-learn package).