Elements of Machine Learning, WS 2022/2023

Jilles Vreeken and Aleksandar Bojchevski

Exercise Sheet #1: *Linear Regression*

Deadline: Thursday, November 17, 2022, 16:00

Before you start solving the exercise, please consult the instructions in the course website. Importantly, note that the submission to each problem must be contained in an individual file.

- For each theoretical problem you must provide a single `pdf` file containing your answer to the respective problem. This file may be a scan of your (legible) handwriting.
- For each practical problem, you need to provide
  - the completed jupyter notebook (`.ipynb`) file, and
  - any necessary files required to reproduce your results. Additionally,
  - a `pdf` report generated from the jupyter notebook that shows all your results.

All the above files must be compressed into a `zip` archive and uploaded using the content management system of the course.

Please remember, that each team member has to submit the signed Code of Conduct for each assignment. The Code of Conduct can be found in the material section on the course website.

Problem 1 (T, 5 Points). **Principles of statistical learning**

Explain in short and concise words the concepts of these pairs of terms:

1. unsupervised and supervised learning
2. prediction and inference
3. classification and regression
4. training and test data
5. parametric and non-parametric methods

You are allowed to use a maximum of **300 words** for this exercise, every **ten** more words will lead to losing **one** point.

Problem 2 (T, 6 Points). **Bias-variance trade-off**

1. [$5pts$] Prove that the expected test mean square error (MSE), for a given value x_0, can always be decomposed into the sum of three fundamental quantities: the variance of $\hat{f}(x_0)$, the squared bias of $\hat{f}(x_0)$ and the variance of the error terms ϵ. Please note that you should prove both qualities (both equals relation), enlist all assumptions and define all parameters and variables.

$$\begin{aligned}\mathbb{E}\left[(y_0 - \hat{f}(x_0))^2\right] &= \mathbb{E}\left[(\hat{f}(x_0) - \mathbb{E}(\hat{f}(x_0)))^2\right] + \left[\mathbb{E}(\hat{f}(x_0) - f(x_0))\right]^2 + \mathrm{Var}(\epsilon)\\ &= \mathrm{Var}(\hat{f}(x_0)) + \left[\mathrm{Bias}(\hat{f}(x_0))\right]^2 + \mathrm{Var}(\epsilon).\end{aligned}$$

   Note that this is Equation (2.7) from ISLR (p. 34), which contains Equation (2.3) from ISLR (p. 19).

2. [$1pts$] Explain in your own words irreducible and reducible error as well as their difference.

Problem 3 (T, 6 Points). **Gauss-Markov theorem**

1. [2*pts*] Explain in your own words what the Gauss-Markov theorem is and why it is important.
2. [3*pts*] Explain in your own words and write down the formal math formulation of the three *Gauss-Markov error term assumptions.*
3. [1*pts*] Is it also the *best* (in terms of test error) linear unbiased estimate (argue with the bias-variance trade-off)?

Problem 4 (T, 8 Points). **Least Squares Error**

1. [2*pts*] Let Y be a random variable. Show that

$$\mathbb{E}(Y) = \text{argmin}_c \mathbb{E}\left[(Y-c)^2\right].$$

   Explain why this might be an important result for Linear Regression.

2. [6*pts*] The R^2 statistic is a common measure of model fit corresponding to the fraction of variance in the data that is explained by the model. In general, R^2 is given by the formula

$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}}.$$

   - Show that for univariate regression, $R^2 = Cor(X,Y)^2$ holds.
   - Show that in the univariate case, $R^2 = Cor(Y,\hat{Y})^2$ holds.

Problem 5 (P, 14 Points). Solving this problem will make it easier to solve the following programming one. This exercise uses the *Auto* data set which is provided in the *zip* file. For this problem you need to write your solutions in the Jupyter notebook provided with the assignment `Practical_Problem_1.ipynb`. Reminder:

- Please rename submission files to include your name and matriculation number in your pdf and .pynb file (example: `john_123456_practical_problem_1.ipynb`).
- Please answer each question in a few sentences as precise and short as possible and focus on the most outstanding facts. Explicitly describe and name the information you retain from plots and tables.
- Make sure you code runs by running each cell in the Jupyter notebook and save all results.

1. [2*pts*] Using the **plot()** function in `matplotlib` and create scatterplots between all the variables. Is the relationship between those variables linear? Describe exactly four different connections between the variables. (Exclude the *name* variable, which is qualitative.)
2. [2*pts*] Detect the two variable pairs **in the scatterplots** that appear to be **the most highly** correlated and anti-correlated, respectively. Justify your choice using the **np.corrcoef()** function. Do the results from **np.corrcoef()** differ from what you see in the plot?
3. [4*pts*] Perform simple linear regression with *mpg* as the response using the variables *cylinders, displacement, horsepower* and *year*, respectively, as features using the **LinearRegression()** function provided in the `sklearn` package. Which predictors appear to have a statistically significant relationship to the outcome (use an appropriate measurement)? How good are the resulting models (provide all four R^2 values)?
4. [4*pts*] Use the **LinearRegression()** function to perform one multiple linear regression with *mpg* as the response and all other variables except *name* as the predictors. Use the **score()** and **get_params()** functions to print the results. Compare the full model to those generated in 5.3: (a) How is the model fit (using R^2)? (b) What can you observe in the different models concerning the significance of the relationship between response and individual predictors? What does the sign of the coefficient (i.e. of the estimate) tell you about the relationship between the predictor and the response? Provide an example from your fitted model.
5. [2*pts*] Use the **plot()** function to produce the residual versus fitted plot of the multiple linear regression fit. Identify the residual plot. Does the residual plot suggest any non-linearity in the data (provide an explanation)?

Problem 6 (Bonus). **Chebychev Polynomials** So far in this assignment we have used least squares to find linear coefficients of a set of provided features. Here, we explore the case where we can enhance the available features with new ones, that are functions of the given ones, for the task of fitting a smooth function on a set of noisy scalars. For this problem consult notebook `Bonus_Problem_1.ipynb`.
From a regression point of view, we are given pairs (x_i, y_i), with $x_i \in [-1, 1]$, $y_i \in \mathbb{R}$, $i = 1, \ldots, n$ and we seek to predict the scalar values y_i in `y_target.csv`. Here, we assume these points to lie on a polynomial curve on the domain $[-1, 1]$. Hence, we assume that the observation follows the functional form

$$y = w_0 + w_1 f_1(x) + ... + w_D f_D(x) + \epsilon, \qquad x \in [-1, 1]\,, \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)\,, \tag{6.1}$$

where we use as features $f_1(x_i), \ldots, f_D(x_i)$ the Chebychev polynomials, with $f_d(x_i)$ of degree $d = 1, \ldots, D$ computed on each input point x_i.

1. You first need to create a function that generates the feature vector for each input point. For this, you have to compute the Chebychev polynomials on the given input x_i for each degree from $1, \ldots, D$. Remember to also add the constant value $f_0(x_i) = 1$ that corresponds to the intercept w_0 in your feature vectors, too.

   Hint: You might find the commands np.linspace and np.polynomial.chebyshev useful, check the documentation for details.

2. Using the value of each Chebychev Polynomial as a feature, perform least squares estimation for the dependent variable y. For degree $D \geq 1$, this will be a Multiple Linear Regression problem. For this create

   - a function `fit_cheb`, that is given a vector of scalar inputs, their labels, and the maximum degree of the polynomials to use and computes the best linear coefficients w_d.
   - a function `predict_cheb` that is given the computed weights w_i, the input vector and the maximum degree of the polynomials to use, and computes the prediction $\hat{y}$ for each input point.

3. We now analyse the error of our predictions. First, create a function `evaluate_mode_on_dataset` that computes the mean squared error of your method, given a training and a testing set.

   Use the function `split_data` to split the dataset into a training and a test set. On the same split compute the prediction error for varying degree D and plot the results.

   What is the behaviour of the prediction error with respect to the degree? Based on your observation, which maximum polynomial degree would you use and why?

   Hint: Increasing the degree of the polynomial makes the model less rigid as discussed in the lecture.

4. Now compare the above splitting scheme with that of the code `split_data_around_point`. Then plot, once again, the prediction error for varying degree D. What is the behaviour of the prediction error with respect to the two different splitting schemes? Based on your observation, what is a good requirement for the statistics of the straining and the testing set?