Technische Universität Chemnitz
Faculty of Mathematics

Prof. Dr. O. Ernst
J. Blechschmidt

14th October 2018
Hand in homework until 28th October 2018

Introduction to Data Science

Sheet 2: Data import and linear regression

Exercise 1: Introduction – Data Import

Download the files `Advertising.csv` and `importAdvertising.py` from the homepage's exercise section and take a short look at the `csv`-file using a spreadsheet, e.g., `LibreOffice`. The file contains information about the sales of products in different markets, along with advertising budgets in the three media: TV, radio and newspaper.

Execute the tasks in the file `importAdvertising.py`, i.e., import the `csv`-file using `np.genfromtxt` and present the data graphically.

Exercise 2: Writing a function in Python

The mean squared error, short MSE, is one of the most important performance indicators for the quality of data fitting.

The goal of this problem is to implement a function `computeMSE` with the following input:

- the observations $y_i \in Y$, $i = 1, \ldots, N$ that belong to measurements $x_i \in X$, $i = 1, \ldots, N$
- the prediction of $f(x_i)$: $\hat{f}(x_i), i = 1, \ldots, N$

and corresponding output:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{f}(x_i))^2$$

Download the file `mse.py` from the homepage and execute the included tasks.

Exercise 3: "Nonlinear" linear regression

Download the file `higherOrderLinReg.py` from the lecture's homepage. The goal of this problem is to approximate given data points (x_i, y_i) for $i = 1, \ldots, n$ by polynomials of degree p. This can be done by solving the linear regression problem:

$$y_i \approx \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \ldots + \beta_p x_i^p$$

By splitting our data into a training and test data set, we want to illustrate graphically the problem of overfitting.

To do so, you should use the functions implemented and used in Exercise 2. In particular, the function **computeMSE** should be copied into the current file, or imported by the following command

```
from mse import computeMSE
```

Execute the tasks in the script `higherOrderLinReg.py`. After that, you should take some time to play with the parameters **eps**, **n** and **ntrain**. What do you observe?

Hint: If you decide to embed the function **computeMSE** by the following code **from mse import computeMSE**, you have to be aware of the following: If you execute your script `higherOrderLinReg.py`, all commands from the script `mse.py` will also be executed. In order to avoid this, you have to change the structure of the script `mse.py` slightly:

```
import numpy as np
import matplotlib.pyplot as plt

def computeMSE(...):
    # Put here the definition of the function
    return ...

if __name__ == "__main__":
    # Put here all the rest
```

Homework 1: Introduction – Pandas

In Exercise 1, you got to know a method to import `csv`-files using the function **np.genfromtxt**. At one point or another, we would have to deal with a problem that is inherent to **numpy**-arrays, namely that **numpy**-arrays can only handle one data type at a time. If we have different kinds of data like booleans, floats, integers or strings, we have to take a different route.

One possible solution lies in the usage of the package **csv**. Here, every single row is scanned seperately, and thus can be handled to catch special cases.

Another possibility is to use the package **pandas**, whose complexity is between the other two. It can be imported by

```
import pandas as pd
```

and `csv`-files can be imported by **pandas.read_csv**.

Create the script `importAuto.py`, which performs the following tasks.

(a) Work through the tutorial https://pandas.pydata.org/pandas-docs/stable/10min.html#min

(b) Download the file `Auto.csv` from the class webpage. Import the `csv`-file using the `pandas` function `read_csv` as a `DataFrame` named `Auto`. Beware of the missing values in the `csv`-file. You can use the optional parameter `na_values` from the function `read_csv`. In this problem, we want to remove those data sets that contain missing values. You should use the method `dropna(axis=0, inplace=True)` for this purpose. Finally, the `DataFrame` should contain 392 observations with 9 features.

(c) Create a short summary of the most important statistics of the data set using the method `describe`.

(d) Create a graphical overview of the distributions of the input variables of the data set using the method `hist`. Infer from

```
pd.scatter_matrix(Auto, marker='o', alpha=.7)
```

possible relationships between the predictors.

(e) Create two figures that relate the variable "horsepower" with "mpg" and "weight", resp. Use the possibilities that are provided by `pandas`.

(f) Investigate a possible linear and quadratic relationship between "horsepower" and "mpg" as well as between "horsepower" and "weight". Use the mean squared error to justify your findings. What do you observe?