HW1 – Linear Regression, Logistic Regression & Perceptron

Instructions – read before you start the exercise:

- Submission – in pairs, until 25/11/19 at 23:55.
  You are to submit a zip folder with the name HW1_ID1_ID2.zip where ID1 and ID2 are the IDs of the submitters. The zip should contain two files – a .py file for the code and a PDF with the (detailed) solution of the exercises.
- Only one member in the pair should submit the solution.
- It is your responsibility to make sure that the code will not crash – if it does, then the grade for the wet part will be 0.
- We will run the code in python version 3.6.8. Colab users - use python3.
- You may only import the following libraries:
  numpy, pandas, matplotlib.pyplot, sklearn, seaborn.
- The variables in your code should have informative name, e.g., don't name them a, b, c but use a name which describes the variable (for example, MSE_train, y_poly, etc.)
- Make sure your answers are organized, detailed, clear and readable (should you intend to write part of the solution by hand and scan it), as well as keeping your code organized and documented.
- When submitting the code, make sure that it doesn't save the plots (plt.savefig should be in remarks). You are to present the plots in the PDF file.
- No cheating – if you were to cheat and copy parts from the code and/or the answers from another group, the grade for both parties will be 0 and you would all risk being expelled from this course and from the Technion. You are welcome to work in groups, but you should write the answers in your own words.
- Any question about this exercise should be written in the forum.

Goodluck!

Question 1

In this question you will calculate a closed-form expression the least squares (LS) estimators in a one-dimensional linear regression, and you will prove an important claim about them.

Given observations $\{(x_i, y_i)\}_{i=1}^n$ with the relation $y_i = w_0 + w_1 x_i + \varepsilon_i$ where all $\varepsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$ are uncorrelated with x_i:

a. Calculate the LS estimators $\hat{w}_0, \hat{w}_1$, by minimizing the MSE, using the following steps:

   1. Define the MSE function $f(w_0, w_1) = \frac{1}{n}\sum_{i=1}^n \varepsilon_i^2$, when ε_i is expressed by y_i, w_0, w_1 and x_i.
   2. Calculate the partial derivatives $\frac{\partial}{\partial w_0} f(w_0, w_1), \frac{\partial}{\partial w_1} f(w_0, w_1)$.
   3. Find $\hat{w}_0, \hat{w}_1$ such that $\nabla f(\hat{w}_0, \hat{w}_1) = \left(\frac{\partial}{\partial w_0} f(\hat{w}_0, \hat{w}_1), \frac{\partial}{\partial w_1} f(\hat{w}_0, \hat{w}_1)\right) = \vec{0}$.

      Hints:

      - Extract from the equation $\frac{\partial}{\partial w_0} f(\hat{w}_0, \hat{w}_1) = 0$ the connection $\hat{w}_0 = \bar{y} - \hat{w}_1 \bar{x}$ when $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$ and $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$.
      - Solve the equation $\frac{\partial}{\partial w_1} f(\hat{w}_0, \hat{w}_1) = 0$ to find that:

$$\boldsymbol{\hat{w}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x}) \cdot y_i}{\sum_{i=1}^n (x_i - \bar{x})^2}}$$

        You may use the following claims without proof:

$$\bar{y} \cdot \sum_{i=1}^n x_i = \bar{x} \cdot \sum_{i=1}^n y_i\,, \qquad \sum_{i=1}^n x_i^2 - n \cdot \bar{x}^2 = \sum_{i=1}^n (x_i - \bar{x})^2$$

      - Conclude that:

$$\boldsymbol{\hat{w}_0 = \bar{y} - \frac{\sum_{i=1}^n (x_i - \bar{x}) \cdot y_i}{\sum_{i=1}^n (x_i - \bar{x})^2} \cdot \bar{x}}$$

b. Prove that the estimators you have found are unbiased estimators, meaning that $\mathbb{E}[\hat{w}_0] = w_0$ and $\mathbb{E}[\hat{w}_1] = w_1$.

   Which assumptions are needed for this claim to be true?

   What does this claim mean?

Hints:

- Prove that $\widehat{w}_1 = w_1 + \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \cdot \varepsilon_i}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$.
- After proving that $\mathbb{E}[\widehat{w}_1] = w_1$, use the connection $\widehat{w}_0 = \bar{y} - \widehat{w}_1 \cdot \bar{x}$ and prove that $\mathbb{E}[\widehat{w}_0] = w_0$.

Question 2

In this question, you will learn how to perform polynomial regression with linear regression tools using python.

Given train data $(x_1 = 0, y_1 = -1.25), (x_2 = 0.5, y_2 = -0.6), (x_3 = 2, y_3 = -4.85)$ and test data $(x_4 = -1, y_4 = -5.2), (x_5 = 1, y_5 = -0.9), (x_6 = 3, y_6 = -13)$:

a. Define a 3×1 two-dimensional matrix called X_train in which each line is an observation from the training data, and define a (one-dimensional) vector called y_train which contains the responses of these observations (in the same order). Repeat this process with the test data (X_test, y_test).
b. Calculate the regular LS estimators $\widehat{w}_0$, $\widehat{w}_1$ using only the training data with sklearn built-in functions. What are the predicted values for X_test?
c. What is the MSE of the regression on the train data? What is the MSE of the regression on the test data? What can you conclude from these values?
d. Write a function which receives a np-array of explanatory variables X and a np-array of responses Y and returns the least squares estimator using the closed-form expression we saw in class. There is no need to check that the input is valid. (don't forget to add ones!)
e. Plot the regression line in a dashed (--) black line. Scatter (with plt.scatter()) the points in the train data with marker='*' and scatter the points in the test data with marker='o'. You should use legend (for train and test data) and label the axes. The range in the x axis should be np.arange(-3, 5).
   Does it look like the regression fit the data?
f. We will now try to perform a 2nd degree polynomial regression, meaning we will assume now that $y_i \approx \gamma_0 + \gamma_1 \cdot x_i + \gamma_2 \cdot x_i^2$.
   1. If we would mark $z_i = (x_i, x_i^2)^T$, how can we write y_i in a linear form?
   2. Define a 3×2 matrix called Z_train in which the first column corresponds to x_i and the second column corresponds to x_i^2 for each x_i in the train data (in the same order as in section a.). Repeat this process with the test data (Z_test).

3. Use the function you wrote in section d. to calculate the LS estimators $\hat{\gamma} = (\hat{\gamma}_0, \hat{\gamma}_1, \hat{\gamma}_2)^T$ using only the training data (Z_train and y_train). What is the MSE of the regression on the train data? What is the MSE on the test data?
4. Plot the corresponding 2nd degree polynomial function in a red dashed line, alongside the original regression line in a black dashed line. Scatter the points in the training data (with marker='*'), as well as the points in the testing data (with marker='o'). You should use legend for both data type (train/test) and regression type (linear/polynomial) and label the axes. The range in the x axis should be np.arange(-3, 5). Name the plot 'Polynomial Regression vs. Linear Regression'.

   Which regression seems to perform better?

g. Which assumption did not hold, thus making the linear regression to fail?

Question 3

The course website contains a file called "parkinsons_updrs_data.csv" in which there are 5875 records. This file contains the attributes of speech recordings of 42 individuals with Parkinson's disease. In this question, we will try to predict the medical status of people with Parkinson's disease using their speech data. A file called "parkinsons_updrs.names.txt" with more details on the data is also in the website.

a. Load the data using the Pandas library.
b. How many males are in the data? How many females? Show a bar plot.
c. Visualize the distribution of the ages with a histogram.
d. How does the mean of the variable "motor_UPDRS" varies between different sexes? Visualize it using a bar plot (use groupby).
e. Choose 6 explanatory variables and display a scatter plot of them with the response variable motor_UPDRS (use the function scatter_matrix()).
f. Use sklearn's built-in functions to calculate the LS estimator, using only the 6 explanatory variables you chose in the previous section.
g. Use the function from question 2 to calculate the least squares estimator, using only the 6 explanatory variables you chose in the section e.
   Did you get the same answer as in the previous section?
   (hint: the answer should be yes. Otherwise, you have a problem)

Question 4

In this question you will implement the Perceptron algorithm:

1. $w^{(1)} = (0, \dots, 0)^T$.
2. *For* $t = 1$ *to T:*
   - *2.1 If there exists* i *such that* $y_i \cdot w^T x_i \leq 0$*:*
     - *2.1.1 Set* $w^{(t+1)} = w^{(t)} + y_i x_i$.
     - *2.1.2 Continue.*
   - *2.2 Else return* $w^{(t)}$.

Write a function called Perceptron which receives the following parameters:
X – a two-dimensional matrix (np-array) in $\mathbb{R}^{n \times d}$ which contains the observations.

y – a vector (np-array) in $\mathbb{R}^n$ which contains the responses.

The function should return w after the convergence of the Perceptron algorithm.

Notes:

1. This function should be generic (meaning it can receive data in any dimension).
2. You may assume that the data is linearly separable, and you are not required to check that the input is valid.
3. Do not forget to add a constant 1 to all observations – the function should produce a (d+1)-dimensional vector in which w_0 is the intercept and w_i is the coefficient of x_i for each $i \geq 1$.
4. You will use this function in the next exercises. For validating the quality of your function, you can use synthetic data from here: http://scikit-learn.org/stable/datasets/index.html.

Question 5

In this question you will use the sklearn's Iris dataset from which contains 150 observations of 3 Iris species. You will use Logistic Regression classifier in order to generate a multi-class classifier of type one-versus-rest.

a. Load the data using the following code:

```
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data
y = iris.target
```

X is a np-array with 150 rows and 4 columns: Sepal Length, Sepal Width, Petal Length and Petal Width.
y is a np-array with 150 rows and 1 column which contains 0, 1, or 2 for each Iris species: Setosa, Versicolour and Virginicacv (respectively).

b. Split the data to train and test data. Use the train_test_split function with random_state=1000.
c. A Logistic Regression classifier is a binary classifier, which also provides us a confidence level – the probability of belonging to each class of the two.
In the following sections, you will build a one-versus-rest classifier using the following scheme:
    - Create a new vector from the response vector (y_train) which contains 1 for the Setosa and -1 for the other species. Do the same for y_test.
    - Build a Logistic Regression classifier for the species Setosa (using the training data). Use the new response vector you have created.
d. Repeat section c for the other two species.
e. Create a function which receives all the above classifiers and a collection of observations as np-array and returns a one-versus-rest classification vector, in which each observation is classified to the class for which it has the maximal probability to belong to.

f. Use the function you wrote to perform a one-versus-rest classification for the test data. Use the output of this function to create and plot a confusion matrix. In order to calculate it, use the following code:

```
from sklearn.metrics import confusion_matrix
my_confusion_matrix = confusion_matrix(y_test, y_pred)
```

You are to display two different plots – one for the unnormalized (original) confusion matrix and one for the normalized (as seen in tutorial 3) matrix. In order to plot it, use the following code:

```
import seaborn as sn
sn.heatmap(my_confusion_matrix, annot=True, cmap="tab20b")
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.title('…') # change the title according to the type of the confusion matrix
plt.show()
```

g. Choose one observation which is misclassified and explain why you think the one-versus-rest classifier did not classify it right.