Assignment 2

Data Mining and Machine Learning (2IIG0)

Stiven Dias, Sibylle Hess and Ghada Sokar

This is your second assignment of the Data Mining and Machine learning course. The assignment is composed of six questions, five of them are auto-graded. The last question is open-ended. Multiple-choice questions have four possible choices; among these, only one answer is correct. The number of points for each question is written at the beginning of it. The maximum achievable score is 100 points. In all the questions, you have to implement the algorithm yourself required in the question using Python. You are only allowed to use scikit-learn for some steps where it is explicitly stated in the questions.

Submit the answers for the first five questions in the **HW2** quiz in Canvas. You have to submit this quiz individually. That is, every group member has to fill out their own quiz, even when you are encouraged to work on the exercises together. You can submit the quiz multiple times, the latest one will be graded. Please fill this quiz out carefully.

The open question can be submitted on a group level by means of the **Open Question HW2** assignment in Canvas. That is, one of your group members should upload:

- A PDF document containing the answer to the open question. Include a paragraph about the workload distribution as well. We assume that you can decide yourself how to organize the work in your group. But if someone didn't contribute at all, or only marginally, then you should note this in the PDF.
- A Python file/ notebook containing the implementations you used to answer the implementation-based questions. This notebook is your insurance, it shows what you did and which results you got. If you want to argue for a higher grade/get partial points then you can back it up with the results stated in the uploaded notebook. However, if we see that you couldn't provide a working implementation, but just guessed the correct answer in the MC format, then we will also subtract the corresponding points from your grade.

1. (10 points) **Regression with polynomial basis functions.** Load the California housing dataset from sklearn. This is a regression dataset which poses the task to predict house prices. The following command should download the data and give a description of features:

   ```
   california = sklearn.datasets.fetch_california_housing()
   print(california.DESCR)
   ```

   Compute the design matrix for this dataset when using a polynomial with degree 2. You can use the following function from sklearn to do so:

   ```
   from sklearn.preprocessing import PolynomialFeatures
   poly = PolynomialFeatures(degree,include_bias=True)
   poly.get_feature_names_out(california.feature_names)
   ```

   Compute the regression model minimizing the RSS for the polynomial design matrix. Which of the polynomial features gets the weight that is closest to zero in the regression model?

2. (20 points) **Lasso vs. Ridge Regression** In this exercise, we try to find out if and how we can predict the cancer survival time based on gene expressions, using Lasso and Ridge regression. We use here the *METABRIC_RNA_Mutation* dataset, that is available in canvas. You can gain more information about this dataset here: `www.kaggle.com/datasets/raghadalharbi/breast-cancer-gene-expression-profiles-metabric`. Run the following code to load the data and to obtain your regression dataset and target vector:

   ```
   import pandas as pd
   df = pd.read_csv("./Downloads/METABRIC_RNA_Mutation.csv")#Adapt the path
   df_D = pd.concat([df['age_at_diagnosis'], df.iloc[:, 31:520]],axis=1)
   D = df_D.to_numpy()
   y = df['overall_survival'].to_numpy()
   ```

   We use the feature `overall_survival` as the regression target variable. For the observations, we use the feature `age_at_diagnosis` and the gene expressions. We compute the regression models using a design matrix X for affine basis function.

   We compare the solutions to the following regression objectives (n is the number of observations in X):

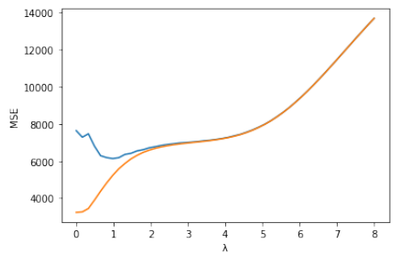
   $$\min_{\beta} \frac{1}{2n}\|y - X\beta\|^2 + \lambda\|\beta\|_2^2 \quad \text{(Ridge Regression)}$$
   $$\min_{\beta} \frac{1}{2n}\|y - X\beta\|^2 + \lambda\|\beta\|_1 \quad \text{(Lasso)}$$

   To this end:

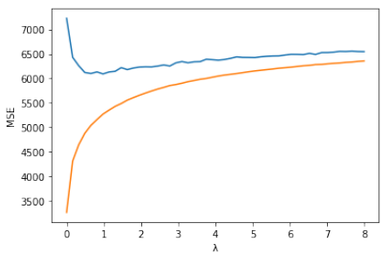
   - Implement a function that fits a ridge regression model to a given training dataset (you have to adapt the formula for β stated in the lecture according to the constants used here to make both objectives comparable).
   - Implement a function that returns the predictions of your ridge regression model for a training or test dataset.
   - Use for fitting (training) and testing of the Lasso model the implementation provided by sklearn.
   - For a range of $\lambda \in [10^{-10}, 8]$ plot the 5-fold cross-validated MSE of the training- and test-data and the average number of selected features (we say a feature is selected if the weight $|\beta_s| > 10^{-16}$) against the weight λ. You can use sklearn for the cross validation.

   (a) (10 points) Which of the following test- and train- MSE plots resemble the most the ones you get for Lasso and Ridge Regression? The test-MSE is in blue and the train-MSE is in orange.

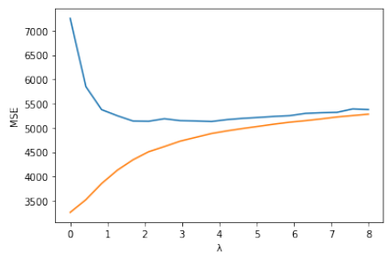
A.



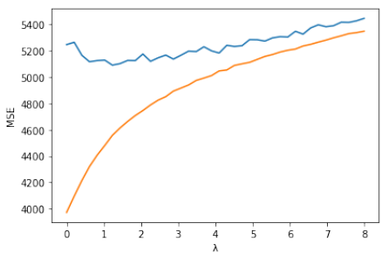
B.



C.

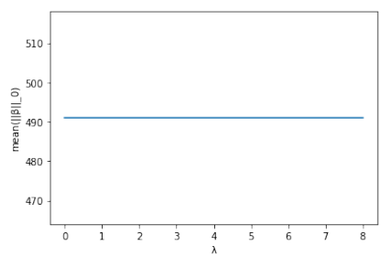


D.

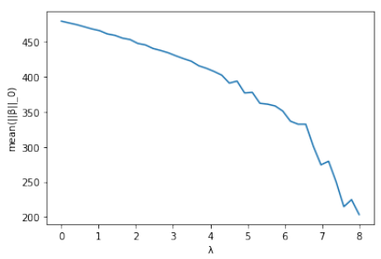


(b) (10 points) Which of the following plots of the average number of selected features resemble the most the ones you get for Lasso and Ridge Regression?

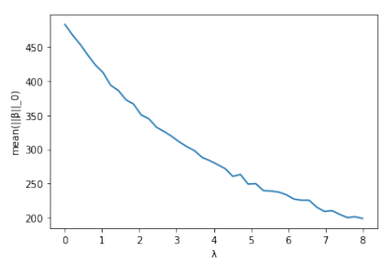
A.



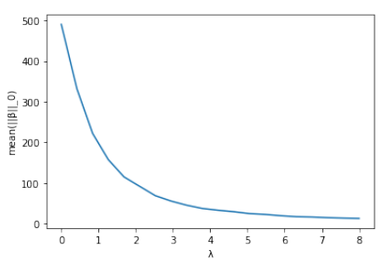
B.



C.



D.



3. (10 points) Consider the following true regression function:

$$f^*(x) = \tan(\pi x)$$

Imagine you fit three regression models on i.i.d. data samples $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ and obtain the following models:

$$f_{\mathcal{D}_1}(x) = x - 0.1$$
$$f_{\mathcal{D}_2}(x) = 3x + 0.1$$
$$f_{\mathcal{D}_3}(x) = 5x + 0.2$$

Compute for $x_0 = 0.1$ the sample

(a) **bias**^{2} and

(b) **variance**.

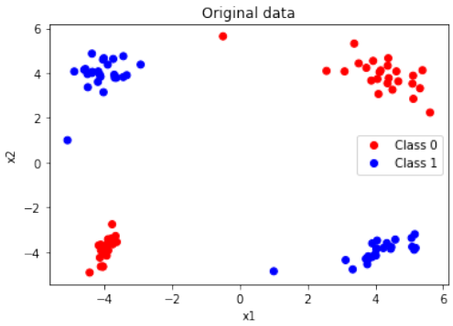
You can round your answer to 2 decimals after the point.

Hint: you may use the corresponding formulas given in the lecture and interpret $\mathbb{E}_{\mathcal{D}}[f_{\mathcal{D}}(x_0)]$ as the mean over the samples listed above at x_0.

4. (10 points) This exercise is about the theoretical understanding of Multi Layer Perceptron (MLP) network. Consider the two-classes dataset shown in the figure below. The data has two features (x_1, x_2). Suppose that we want to fit this data using a multi-layer feed-forward neural network.

Which of the following statements is **correct**?

This dataset can be modeled with zero *training* error using:



A. A neural network with a single hidden layer and linear activation.

B. A neural network with at least two hidden layers and linear activation.

C. A neural network with at least one hidden layer and non-linear activation. Note though that the single layer network will probably be more economic in terms of units than a network with multiple hidden layers to achieve zero training error.

D. A neural network with an arbitrary number of hidden layers ($\neq 0$) and non-linear activation, albeit the total number of units required to get zero training error may reduce as we increase the number of hidden-layers.

E. The data could not be modeled with zero training error using a MLP network since the data set is not linearly separable.

5. (10 points) This exercise is about the theoretical understanding of convolutional neural networks. Consider the figure below which represents a convolutional neural network that converts a 64×64 RGB image into 2 output values for a binary classification task. The network consists of the following layers from input to output:

Input C=3@W=64xH=64 RGB image;

L0 C0@W0xH0 padded input;

L1 C1 feature maps with size W1xH1 after convolutional layer;

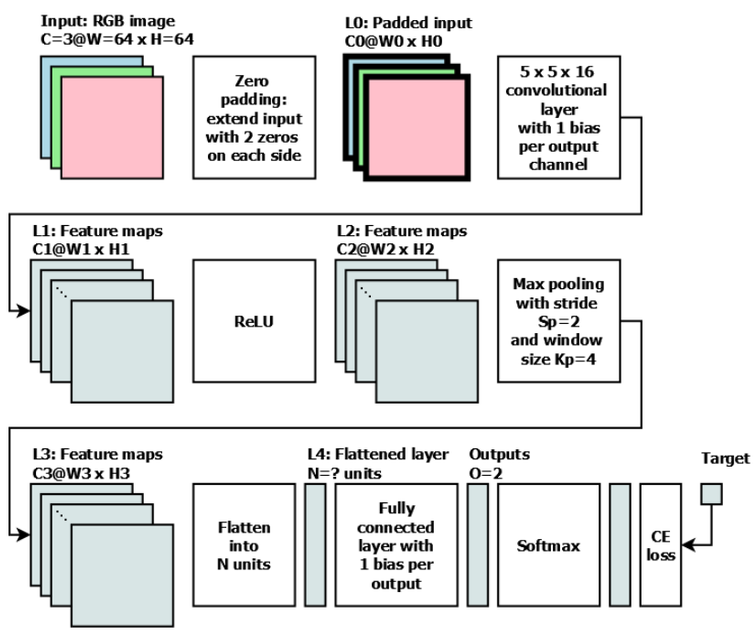
L2 C2 feature maps with size W2xH2 after ReLU activation;

L3 C3 feature maps with size W3xH3 after max pooling;

L4 Flattened layer with $N =?$ units;

Outputs Fully-connected layer with 2 units followed by soft-max.

The notation C@WxH means C channels of size WxH.



Which of the following statements is **correct**?

A. The number of weight parameters of the kernel in the convolutional layer is 400.

B. The number of feature maps C3 in layer L3 is 32.

C. The number of units N in the flattened layer L4 is 16384.

D. The total number of network parameters (weights, bias) is 31970.

6. (40 points) **Open Question** In this exercise, you are asked to implement a Multi Layer Perceptron (MLP) network **from scratch**. In the assignment .zip file, you can find Python notebook (MLP.ipynb) that contains a skeleton for parts of the code. You may add as many input parameters as you need to any of the existing function prototypes. You can also find a small two-classes dataset divided into two *.csv* files: *data/train_data.csv* and *data/validate_data.csv*. Each row represents a data point. The first two columns (x_1, x_2) represent the two input features. The third column (y) represents the class label (0 or 1).

   We would like to fit this data using the MLP network. The network consists of 2 hidden layers. Each layer consists of 10 neurons. The network should be trained using Backpropagation (see also the JupyterBook for further details about the Backpropagation algorithm) and mini-batches Stochastic Gradient Descent (SGD) as explained in the lecture. To this end:

   (a) Import the data in Python and explore it a bit.

   (b) (5 points) Perform data normalization. Choose one of the data normalization methods discussed in the lecture (e.g. Zero-mean-unit-variance, Min-Max normalization, etc.) and apply it on the data. Report the formula that you used in the report and discuss your choice. Include in your report the first 4 samples of the training data after transformation.

   (c) (1 points) Determine the size of the input and output layers. Report your answer by drawing the MLP model's structure.

   (d) (2 points) Choose a suitable loss function and activation function for the first and second hidden layers ϕ_1, ϕ_2. Report and justify your choice.

   (e) (2 points) Choose an initialization of the parameter values. Report and discuss your choice.

   (f) (10 points) Implement the MLP network from scratch with the specification stated above and the Backpropagation algorithm to train the network. We will run and check the uploaded Python code. To obtain the points for this subproblem: i) the Python code has to run with **no errors** and ii) the MLP model and the Backpropagation algorithm have to be implemented completely from scratch. **Note** that you are not allowed to use any library which implements MLP models, but you are allowed to use auxiliary libraries (e.g. Numpy, Matplotlib, Pandas).

   (g) (10 points) Tune the hyperparameters: learning rate, batch size, number of epochs. Try to achieve the best validation accuracy you can. In this subproblem, part of your grade depends on how well your model performs. You should at least get 70% on the validation set to obtain points greater than 0. You can get the full points when you are close to the maximal performance, $\sim 97\%$, on the validation set (i.e. $\geq 90\%$ should be fine to obtain the full points). Report the best values for each hyperparameter.

   (h) (5 points) Plot the loss computed over the training set and over the validation set. In addition, plot the classification accuracy computed over the two sets. Choose a stopping criteria for your training. State and justify your stopping criteria. Include your answer and the plots in your report.

   (i) (5 points) Report the final accuracy obtained and the confusion matrix on the training and validation data sets.