

Robot Learning

Winter Semester 2022/2023, Homework 3

Prof. Dr. J. Peters, N. Funk, T. Schneider, J. Ritter, K. Hansel and D. Rother



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Total points: 39 + 4 bonus

Due date: 23:59, Monday, 12 December 2022

Note: Inside the homework folder, there are:

- A Latex template for this exercise that you can use to fill in your answers and that you can also directly use in sharelatex to collectively write the answers!
- There is also a Conda environment prepared in the "code" folder. To install this, follow the instructions in the README file.

Important: For every answer, please describe precisely how you obtain the results and justify every answer. Otherwise, we cannot give you the points (as in the exam)!

Important: Please, don't forget to submit your code.

Problem 3.1 Machine Learning in a Nutshell [24 Points + 4 Bonus]

For this exercise you will use a dataset, divided into training set and validation set. The first row is the vector \mathbf{x} and the second row the vector \mathbf{y} .

Based on this data, we want to learn a function mapping from \mathbf{x} values to \mathbf{y} values, of the form $\mathbf{y} = \boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x})$.

a) Machine Learning Paradigms [2 Points]

Machine Learning (ML) algorithms permit solving various problems. Name the 5 standard ML paradigms presented in the lecture. To which of those paradigms can the problem above be assigned? Why?

b) ML Problems [2 Points]

Consider the following two problems:

1. You are given the history of the 200-day course of the DAX, and you would like to predict the forecast for the next four weeks.
2. You receive many emails and would like to automatically identify the spam messages.

Answer the questions for each problem: What type of problem is this? Name one method that can solve this problem.

c) Linear Least Squares [Programming Exercise] [4 Points]

Consider the training set above to calculate features $\boldsymbol{\phi}(\mathbf{x})$ of the form $[\sin(2^i x)]_{i=0, \dots, n-1}$. Compute the feature values for $n = 2, 3, 9$ (i.e., when using 2, 3 and 9 features). Use Gauss' principle of least squares, also called linear least squares (LLS) method to predict output values \mathbf{y} for input values $x \in \{0, 0.01, 0.02, \dots, 6\}$ using the different numbers of features. Attach a single plot showing the data points and the three resulting predictions when using 2, 3 and 9 features (i.e., having x and y as axes). What can you observe?

-
- Implement `compute_LLS` in `ml.py`.

d) Training a Model [Programming Exercise] [2 Points]

The Mean Absolute Error (MAE) is defined as $MAE = \frac{1}{N} \sum_{i=1}^N |y_i^{\text{true}} - y_i^{\text{predicted}}|$, where N is the number of data points. Using the LLS algorithm implemented in the previous exercise, train one model per number of features between 1 and 9 (i.e., $[1, 2, 3, \dots, 9]$). For each of these models compute the corresponding MAE for the training set. Attach a plot where the x-axis represents the number of features and the y-axis represents the MAE.

- Implement `train_and_select_model` (part 1) in `ml.py`.

e) Model Selection [Programming Exercise] [4 Points]

Using the models trained in the previous exercise, compute the MAE for each of these models for the validation set.

Compare in one plot the MAE on the training set and on the validation set. How do they differ? Explain possible reasons for these differences. What is the number of features that you should use to obtain a good model?

- Implement **train_and_select_model** (part 2) in **ml.py**.


f) Cross Validation [Programming Exercise] [8 Points]

K -fold cross validation is a common approach to estimate the test error when the dataset is small. The idea is to randomly divide the training set into K different datasets. Each of these datasets is then used as validation set for the model trained from the remaining $K - 1$ datasets. The resulting vector of errors $\mathbf{E} = [e_1 \dots e_K]$ can now be used to compute a distribution (typically by fitting a Gaussian distribution). When K is equal to the number of data points, K -fold cross validation takes the name of leave-one-out cross validation (LOO).

Apply LOO using only the training set and compute the mean/variance of the MAE for the learned models. Repeat for the models with the number of features between 1 and 9 (i.e., $[1, 2, 3, \dots, 9]$).

Attach a plot showing the mean/variance (as a distribution) of the MAE computed using LOO and having on the x-axis the number of features and on the y-axis the MAE. How many features should be used to obtain a good model? What are the advantages and disadvantages of using LOO instead of the simple train and validation split?

- Implement **k_fold** in **ml.py**.



g) Ridge Regression [2 Points]

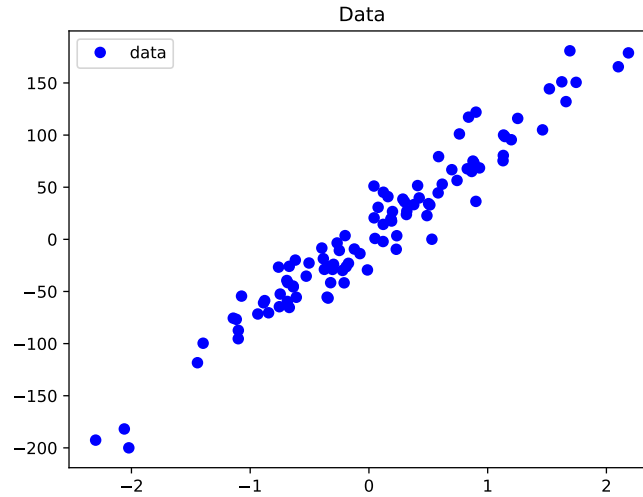
Explain the concept of Ridge Regression and why it is used.

h) Discriminative and Generative models [4 Bonus Points]

Explain the difference between discriminative and generative models and give an example for each case.
Which model category is generally easier to learn and why?

Problem 3.2 Linear Regression [7 Points]

In this exercise, you will implement a type of linear regressors using the `data_ml/data_linear_regression.txt` data file. The file contains noisy observations from an unknown function. Here, the first column represents the input x and the second column represents the output y . Note that when using the function `prepare_data` in `linear_regression.py` X will be a matrix (added columns of 1 for efficient computation).



a) Cost Function [Programming Exercise] [1 Points]

The cost function for the linear regression model is the minimum of the mean squared error of the model obtained by subtracting the predicted values from the actual values. To do this, consider the following objective function:

$$J(w) = \frac{1}{n} \sum_{i=1}^n (h_w(x^{(i)}) - y^{(i)})^2$$

where $h_w(x)$ is the hypothesis given by the linear model $h_w(x) = w^T x = w_1 x_1 + w_0$.

- Implement `cost_function` in `linear_regression.py`.

b) Gradient Descent [Programming Exercise] [4 Points]

In order to minimize the value of the cost function, the batch gradient descent algorithm is used. With each step of the algorithm, the parameters (w_j) get closer to the optimal values until they reach the minimal cost function value. Consider the following gradient descent function:

$$w_j := w_j - \alpha \frac{\partial J}{\partial w_j}$$

where J is the cost function for the linear regression model.

Implement the batch gradient descent algorithm using a learning rate of $\alpha = 0.01$ for 1000 iterations.

- Implement `gradient_descent` in `linear_regression.py`.

c) Fit the Model [Programming Exercise] [2 Points]

A linear regression model uses a straight line to fit the model. This is done using the equation for a straight line, as shown in the equation:

$$y_i = w_1 \cdot x_i + w_0$$

Consider a range for x_i from -3 to 4 to display the data correctly.

- Implement **linear_regression** in **linear_regression.py**.

Problem 3.3 K-Means Clustering [8 Points]

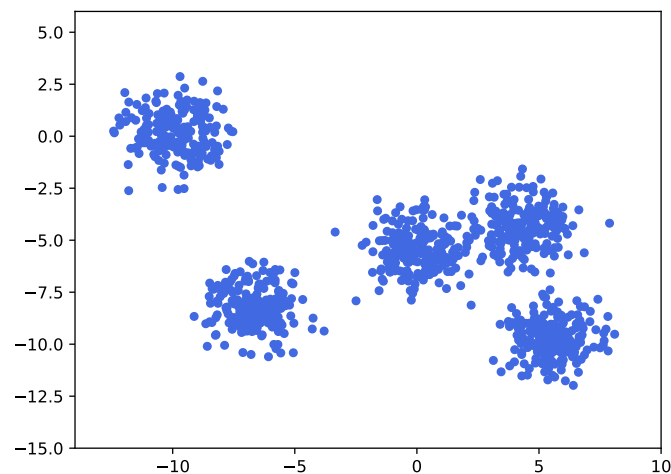
The K-Means algorithm is a computational method that can be used to cluster objects, the so-called cluster analysis. The algorithm is able to determine the respective centers of the clusters from a set of similar objects and a given number of clusters K . The K-Means algorithm is applied to data or objects in an n -dimensional space.

The K-Means algorithm proceeds in the following steps:

- Set the number of clusters K and initialize center points randomly.
- Assign the data points to the different clusters based on the distance (e.g. Euclidean distance) to the center points. Each data point is assigned to the center point with the shortest distance.
- Re-compute the center points by computing the average of all data points of that cluster.
- Repeat step b) and c) until the position of the center points do not change anymore or the maximum number of iterations is reached.

For more information on the K-Means algorithm click [here](#).

In this exercise, you will implement the K-Means algorithm using the `data_ml/data_kmeans.txt` data file.



a) K-Means Algorithm [Programming Exercise] [8 Points]

Implement the K-Means algorithm in `k_means.py` with the given data and the Euclidean distance. We have prepared the data for simplicity and initialized the parameters with 100 as the maximum iterations and 5 clusters. Note the return types in the documentation of function `fit`.

- Implement the K-Means algorithm in the function `fit` in `k_means.py`.

