Introduction to Machine Learning (67577)

Exercise 2
Linear Regression

Second Semester, 2023

Contents

1 Submission Instructions

Please make sure to follow the general submission instructions available on the course website. In addition, for the following assignment, submit a single `ex2_ID.tar` file containing:

- An `Answers.pdf` file with the answers for all theoretical and practical questions (include plotted graphs *in* the PDF file).
- The following python files (without any directories): `linear_regression.py`, `polynomial_fitting.py`, `loss_functions.py`, `utils.py`, `house_price_prediction.py`, `city_temperature_prediction.py`

The `ex2_ID.tar` file must be submitted in the designated Moodle activity prior to the date specified *in the activity*.

- Late submissions will not be accepted and result in a zero mark.
- Plots included as separate files will be considered as not provided.
- Do not forget to answer the Moodle quiz of this assignment.

2 Theoretical Part

Let $\mathbf{X}$ be the design matrix of a linear regression problem with m rows (samples) and d columns (variables/features). Let $\mathbf{y} \in \mathbb{R}^m$ be the response vector corresponding the samples in $\mathbf{X}$. Recall that for some vector space $V \subseteq \mathbb{R}^d$ the orthogonal complement of V is: $V^\perp := \left\{\mathbf{x} \in \mathbb{R}^d \middle| \langle \mathbf{x}, \mathbf{v} \rangle = 0 \quad \forall \mathbf{v} \in V\right\}$

2.1 Solutions of The Normal Equations

Based on Lecture 2 and Recitation 3

1. Prove that: $Ker(\mathbf{X}) = Ker\left(\mathbf{X}^\top \mathbf{X}\right)$
2. Prove that for a square matrix A: $Im\left(A^\top\right) = Ker(A)^\perp$
3. Let $\mathbf{y} = \mathbf{X}\mathbf{w}$ be a non-homogeneous system of linear equations. Assume that $\mathbf{X}$ is square and not invertible. Show that the system has ∞ solutions $\Leftrightarrow y \perp Ker\left(\mathbf{X}^\top\right)$.
4. Consider the (normal) linear system $\mathbf{X}^\top \mathbf{X}\mathbf{w} = \mathbf{X}^\top \mathbf{y}$. Using what you have proved above prove that the normal equations can only have a unique solution (if $\mathbf{X}^\top \mathbf{X}$ is invertible) or infinitely many solutions (otherwise).

2.2 Projection Matrices

5. Based on Recitation 1 In this question you will prove some properties of orthogonal projection matrices seen in recitation 1. Let $V \subseteq \mathbb{R}^d$, $dim(V) = k$ and let $\mathbf{v}_1, \ldots, \mathbf{v}_k$ be an orthonormal basis of V. Define the orthogonal projection matrix $P = \sum_{i=1}^k \mathbf{v}_i \mathbf{v}_i^\top$ (notice this is an outer product).

   Prove the following properties in any order you wish:
   (a) Show that P is symmetric.
   (b) Prove that the eigenvalues of P are 0 or 1 and that $\mathbf{v}_1, \ldots, \mathbf{v}_k$ are the eigenvectors corresponding the eigenvalue 1.
   (c) Show that $\forall \mathbf{v} \in V \;\; P\mathbf{v} = \mathbf{v}$.
   (d) Prove that $P^2 = P$.

(e) Prove that $(I-P)P=0$.

2.3 Least Squares

Based on Lecture 2 and Recitation 3 Given a sample $S=\{(\mathbf{x}_i,y_i)\}_{i=1}^m$, the ERM rule for linear regression w.r.t. the squared loss is

$$\hat{\mathbf{w}} \in \underset{\mathbf{w}\in\mathbb{R}^d}{\operatorname{argmin}} \ ||\mathbf{X}\mathbf{w}-\mathbf{y}||^2$$

where $\mathbf{X}$ is the design matrix of the linear regression with rows as samples and y the vector of responses. Let $\mathbf{X}=U\Sigma V^\top$ be the SVD of $\mathbf{X}$, where U is a $m\times m$ orthonormal matrix, Σ is a $m\times d$ diagonal matrix, and V is an $d\times d$ orthonormal matrix. Let $\sigma_i=\Sigma_{i,i}$ and note that only the non-zero σ_i-s are singular values of $\mathbf{X}$. Recall that the pseudoinverse of $\mathbf{X}$ is defined by $\mathbf{X}^\dagger=V\Sigma^\dagger U^\top$ where $\Sigma^\dagger$ is an $d\times m$ diagonal matrix, such that

$$\Sigma^\dagger_{i,i}=\begin{cases}\sigma_i^{-1} & \sigma_i\neq 0\\ 0 & \sigma_i=0\end{cases}$$

6. Show that if $\mathbf{X}^\top\mathbf{X}$ is invertible, the general solution we derived in recitation equals to the solution you have seen in class. For this part, assume that $\mathbf{X}^\top\mathbf{X}$ is invertible.
7. Show that $\mathbf{X}^\top\mathbf{X}$ is invertible if and only if $\operatorname{span}\{\mathbf{x}_1,\ldots,\mathbf{x}_m\}=\mathbb{R}^d$.
8. Recall that if $\mathbf{X}^\top\mathbf{X}$ is not invertible then there are many solutions. Show that $\hat{\mathbf{w}}=\mathbf{X}^\dagger\mathbf{y}$ is the solution whose L_2 norm is minimal. That is, show that for any other solution $\overline{\mathbf{w}}$, $||\hat{\mathbf{w}}||\leq||\overline{\mathbf{w}}||$.

Hints:

- Recall that the rank of $\mathbf{X}$ and the rank of $\mathbf{X}^\top\mathbf{X}$ are determined by the number of singular values of $\mathbf{X}$. If you are not sure why this is true, go over recitation 1.
- Which coordinates must satisfy $\widehat{w}_i=\overline{w}_i$? What is the value of $\widehat{w}_i$ for the other coordinates? If you are not sure, go back to the derivation of $\widehat{\mathbf{w}}$ (see recitation 4).

3 Practical Part

3.1 Fitting A Linear Regression Model

Based on Lecture 2 and Recitation 3 In this question you will have to deal with a real-world dataset and fit a linear regression model to it. As data is noisy, messy and difficult, take the time to "play" and get familiar with it.

Implement the following:

- `mean_square_error` function in the `metrics.loss_functions.py` file.
- `LinearRegression` class in the `learners.regressors.linear_regression.py` file. Follow class and function documentation.
- `split_train_test` function in the `utils.utils.py` file as described in the function documentation.

Then, implement code of following questions in the `exercise2/house_price_prediction.py` file:

1. Split the data frame to a training set (75%) and test set (25%), using `split_train_test` you've implemented.
2. Implement the `preprocess_data` function. The function receives a loaded `pandas.DataFrame` object of the observation matrix, and a `pandas.Series` object of the respone vector, and returns them after preprocessing. Make sure that if you remove a sample, you remove it properly from both the observation matrix as well as from the response vector.
   Explore the data (some information can be found on Kaggle) and perform any necessary preprocessing (such as but not limited to):
   - What sort of values are valid for different types of features? Can house prices be negative? Can a living room size be too small?
   - Some of the features are categorical with no apparent logical order to their values (for example zip-code). Correctly address these features such that it will make sense to fit a linear regression model using them. For assistance you may refer to the following StackOverflow question.
   - Are there any additional features that might be beneficial for predicting the house price and that can be derived from existing features?

   *Notice: Your code should run properly also if it does not receive a response vector (y is None). That is for the case of inference, where you will recieve house features without the label.

   Describe in details the analysis process that lead you to the decisions of:
   - Which features to keep and which not?
   - Which features are categorical how how did you treat them?
   - What other features did you design and what is the logic behind creating them?
   - How did you treat invalid/missing values?
   - Explain any additional processing performed on the data.

   The answers to these question should be added to your `Answers.pdf` file.

3. Basics of feature selection - implement the `feature_evaluation` as specified in the documentation. This function will compute the Pearson Correlation between each of the features and the response:

$$\text{Pearson Correlation: } \rho := \frac{COV(X,Y)}{\sigma_X \sigma_Y}$$

   for X, Y being one of the features and the response.

   R Pearson Correlation: This measure captures the strength and direction of the *linear* relation between the two random variables A and B. Its values range between $[-1,1]$ where a value of 1 indicates a perfect linear relation between A and B, a value of 0 indicates no linear relation between A and B, and a value of -1 indicates a perfect linear anti-relation between A and B (i.e. $A = -B$).

   You are allowed to use functions that calculate the standard deviation and covariance, but not functions that calculate the Pearson correlation itself.

   Choose two features, one that seems to be beneficial for the model and one that does not. In your `Answers.pdf` add the graphs of these two chosen features and explain how do you

conclude if they are beneficial or not.

4. Fit a linear regression model over increasing percentages of the *training set* and measure the loss over the *test set*:
   - Iterate for every percentage $p = 10\%, 11\%, ..., 100\%$ of the training set.
   - Sample $p\%$ of the train set. You can use the `pandas.DataFrame.sample` function.
   - Repeat sampling, fitting and evaluating 10 times for each value of p.

   Plot the mean loss as a function of $p\%$, as well as a confidence interval of $mean(loss) \pm 2 * std(loss)$. If implementing using the `Plotly` library, see how to create the confidence interval in Chapter 2 - Linear Regression code examples.

   Add the plot to the `Answers.pdf` file and explain what is seen. Address both trends in loss and in confidence interval as function of training size. What can we learn about the estimator $\hat{y}_i$ in terms of estimator properties?

3.2 Polynomial Fitting

Based on Lecture 2 and Recitation 4 Implement the `PolynomialFitting` class in the `learners.regressors.polynomi` file as specified in class documentation. Avoid repeating code from the `LinearRegression` class and instead use inheritance or composition patterns. You are allowed to use the `np.vander` function.

In the following questions you will use the Daily Temperature of Major Cities dataset. Notice, that the supplied file is a modified subset of the dataset found on Kaggle containing only 4 countries. You will fit and analyse performance of a polynomial model over the dataset.

1. Implement the `load_data` function in the `city_tempretaure_prediction.py` file.
   - When loading the dataset remember to deal with invalid data.
   - Use the `parse_dates` argument of the `pandas.read_csv` to set the type of the `'Date'` column.
   - Add a `'DayOfYear'` column based on the `'Date'` column. This column will be the feature to be used for the polynomial fitting.

2. Subset the dataset to caintain samples only from the country of Israel. Investigate how the average daily temperature (`'Temp'` column) change as a function of the `'DayOfYear'`.
   - Plot a scatter plot showing this relation, and color code the dots by the different years (make sure color scale is discrete and not continuous). What polynomial degree might be suitable for this data?
   - Group the samples by `'Month'` (have a look at the pandas `'groupby'` and `'agg'` functions) and plot a bar plot showing for each month the standard deviation of the daily temperatures. Suppose you fit a polynomial model (with the correct degree) over data sampled uniformly at random from this dataset, and then use it to predict temperatures from random days across the year. Based on this graph, do you expect a model to succeed equally over all months or are there times of the year where it will perform better than on others? Explain your answer.

   Add both plots and answers to the `Answers.pdf` file.

3. Returning to the full dataset, group the samples according to `'Country'` and `'Month'` and calculate the average and standard deviation of the temperature. Plot a line plot of the average monthly temperature, with error bars (using the standard deviation) color coded by the country. If using `plotly.express.line` have a look at the `error_y` argument.

   Based on this graph, do all countries share a similar pattern? For which other countries is the model fitted for Israel likely to work well and for which not? Explain your answers.
4. Over the subset containing observations only from Israel perform the following:
   - Randomly split the dataset into a training set (75%) and test set (25%).
   - For every value $k \in [1,10]$, fit a polynomial model of degree k using the training set.
   - Record the loss of the model over the test set, rounded to 2 decimal places.

   Print the test error recorded for each value of k. In addition plot a bar plot showing the test error recorded for each value of k. Based on these which value of k best fits the data? In the case of multiple values of k achieving the same loss select the simplest model of them. Are there any other values that could be considered?

5. Fit a model over the entire subset of records from Israel using the k chosen above. Plot a bar plot showing the model's error over each of the other countries. Explain your results based on this plot and the results seen in question 3.