HOMEWORK 2

CSC2515 Fall 2022

- **Deadline:** Friday, November 4, 2022 at **16:59**.
- **Submission:** You need to submit two files through MarkUs. One is a PDF file including all your answers and plots. The other is a source file (Python script or Jupyter Notebook) that reproduces your answers. You can produce the file however you like (e.g. LaTeX, Microsoft Word, etc) as long as it is readable. Points will be deducted if we have a hard time reading your solutions or understanding the structure of your code. If the code does not run, you may lose most/all of your points for that question. Your report must include all the relevant figures and graphs. We may not look at your code or Jupyter Notebook.
- **Late Submission:** 10% of the marks will be deducted for each day late, up to a maximum of 3 days. After that, no submissions will be accepted.
- **Collaboration:** You can discuss the assignment with up to two other students (group of three). You can work on the code together. But each of you need to **write your homework report individually**. You must mention the name of your collaborators clearly in the report and the source code.

1. Bias and Variance Decomposition for the ℓ_2-regularized Mean Estimator – 25pts.

This exercise helps you become more comfortable with the bias and variance calculations and decomposition. It focuses on the simplified setting of the mean estimator.

Consider a r.v. Y with an unknown distribution p. This random variable has an (unknown) mean $\mu = \mathbb{E}[Y]$ and variance $\sigma^2 = \mathrm{Var}[Y] = \mathbb{E}\left[(Y-\mu)^2\right]$. Consider a dataset $\mathcal{D} = \{Y_1, \dots, Y_n\}$ with independently sampled $Y_i \sim p$.

(a) **[3pt]** Show that the sample average estimator $h_{\text{avg}} = \frac{1}{n}\sum_{i=1}^{n} Y_i$ is the solution of the following optimization problem:

$$h_{\text{avg}}(\mathcal{D}) \leftarrow \operatorname*{argmin}_{m\in\mathbb{R}} \frac{1}{n}\sum_{i=1}^{n} |Y_i - m|^2.$$

Recall that we have the following bias-variance decomposition for the mean estimator $h(\mathcal{D})$:

$$\mathbb{E}_{\mathcal{D}}\left[|h(\mathcal{D}) - \mu|^2\right] = \underbrace{|\mathbb{E}_{\mathcal{D}}[h(\mathcal{D})] - \mu|^2}_{\text{bias}} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[|h(\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[h(\mathcal{D})]|^2\right]}_{\text{variance}}.$$

(b) **[2pt]** Compute the bias and variance of $h_{\text{avg}}(\mathcal{D})$.

(c) **[5pt]** Consider the ℓ_2-regularized mean estimator $h_\lambda(\mathcal{D})$ defined for any $\lambda \geq 0$.

$$h_\lambda(\mathcal{D}) \leftarrow \operatorname*{argmin}_{m\in\mathbb{R}} \frac{1}{n}\sum_{i=1}^{n} |Y_i - m|^2 + \lambda|m|^2.$$

Notice that this is similar to the ridge regression, but only for a single random variable. Provide an explicit formula for the estimator $h_\lambda(\mathcal{D})$ in terms of the sample average and λ.

(d) **[6pt]** Compute the bias and variance of this regularized estimator.

(e) **[5pt]** Visualize $\mathbb{E}_{\mathcal{D}}\left[|h_\lambda(\mathcal{D}) - \mu|^2\right]$, the bias, and the variance terms for a range of λ. As a starting point, you can choose $\mu = 1$, $\sigma^2 = 9$, and $n = 10$.

(f) **[4pt]** Explain the visualization from the previous part, in terms of the effect of bias and variance on the expected squared error.

2. Different Flavours of Regression – 20 pts. We have seen in the lecture how to solve the regression problem with the linear models and the squared error loss. In this question, you learn two other regression methods: *Poisson regression* and *locally weighted regression*. You will derive the associated loss functions and solutions for these methods.

Poisson regression: Recall that the squared error has a probabilistic justification. If we assume that the target values are contaminated with a zero-mean Gaussian noise, the maximum likelihood estimate is the same as the solution of the minimizer of the least squares loss.

We may use probabilistic models other than the Gaussian distribution to model the targets. One such choice is the *Poisson* distribution, which is a discrete probability distribution that can be used to model the number of times a certain event has happened. The probability mass function of a random variable Z with a Poisson distribution with the *rate* $\lambda \in (0, \infty)$ is

$$\mathbb{P}\{Z = k; \lambda\} = p(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}. \tag{2.1}$$

This is the probability that the number of events Z is $k \in \{0, 1, \dots\}$. It can be shown that the expected value and the variance of this random variable are both λ, that is, $\mathbb{E}[Z] = \lambda$ and $\mathrm{Var}[Z] = \lambda$.

The Poisson distribution can be used to model the number of events in a given amount of time or space. Some examples are the number of calls to a call centre in an hour, network packages arriving at a router in a minute, meteorites hitting Earth each year, goals or scores in a soccer or hockey match, bikes used at a bicycle sharing facility per-hour, and "soldiers killed by horse-kicks each year in the Prussian cavalry".

We may use the Poisson distribution to model a regression problem when the target has a count interpretation. In that case, instead of having a fixed rate $\lambda \in (0, \infty)$, the rate λ is a function of the input, which means that $\lambda : \mathcal{X} \to (0, \infty)$. This modelling might be more natural for some problems than using a zero-mean Gaussian noise contamination.

(a) **[5pt]** Suppose that we are given a dataset of $\{Z_1, \dots Z_N\}$ all independently drawn from a Poisson distribution (2.1) with unknown parameter λ. Derive the maximum likelihood estimate of the λ. Show the individual steps and note where you use the data assumptions (identically and independently distributed).
Hint: Use the standard approach to derive the MLE via $\operatorname{argmax}_\lambda \log \prod p(Z_i; \lambda)$.

(b) **[5pt]** The Poisson regression model is based on considering the λ parameter of the Poisson distribution a function of $\mathbf{x}$ and a weight $\mathbf{w}$, which should be determined. The relation is specified by

$$\log \lambda(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \mathbf{x}.$$

Here the logarithm of the rate has a linear model, as opposed to the rate itself. This ensures that the rate is always non-negative.
This rate function leads to the following statistical model for target $y \in \{0, 1, \dots\}$ conditioned on the input $\mathbf{x}$:

$$p(y|\mathbf{x}; \mathbf{w}) = \frac{\exp(y\mathbf{w}^\top \mathbf{x}) \cdot \exp(-e^{\mathbf{w}^\top x})}{y!}.$$

Derive the maximum likelihood function of this model given a dataset consisting of i.i.d. input and response variables $\{(\mathbf{x}^{(1)}, y^{(1)}), \dots, (\mathbf{x}^{(N)}, y^{(N)})\}$. Note that the resulting estimator does not have a closed form solution, so you only have to simplify the resulting loss function as far as possible.

Locally Weighted Regression is a non-parametric algorithm, that is, the model does not learn a fixed set of parameters as is done in regression with a linear model. Rather, parameters are computed individually for each data point $\mathbf{x}$. The next two questions help you derive the locally weighted regression.

(c) **[5pt]** The *weighted* least squares cost uses positive weights $a_1, \dots, \alpha_N$ per datapoint to construct a parameter estimate of the form:

$$\mathbf{w}^* \leftarrow \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{N} a_i^{(i)} (y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)})^2.$$

Show that the solution to this optimization problem is given by the formula

$$\mathbf{w}^* = \left(\mathbf{X}^\top \mathbf{A} \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{A} \mathbf{y},$$

where $\mathbf{X}$ is the design matrix (defined in class) and $\mathbf{A}$ is a diagonal matrix where $\mathbf{A}_{ii} = a^{(i)}$

(d) **[5pt]** Locally weighted least squares combines ideas from k-NN and linear regression. For each query $\mathbf{x}$, we first compute distance-based weights for each training example

$$a^{(i)}(\mathbf{x}) = \frac{\exp\left(-\frac{\left\|\mathbf{x}-\mathbf{x}^{(i)}\right\|^2}{2\tau^2}\right)}{\sum_{j=1}^{N} \exp\left(-\frac{\left\|\mathbf{x}-\mathbf{x}^{(j)}\right\|^2}{2\tau^2}\right)}$$

for some temperature parameter $\tau > 0$. We then construct a local solution

$$\mathbf{w}^*(\mathbf{x}) \leftarrow \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{N} a^{(i)}(\mathbf{x}) \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)}\right)^2.$$

The prediction then takes the form $\hat{y} = \mathbf{w}^*(\mathbf{x})^\top \mathbf{x}$. How does this algorithm behave as the temperature $\tau \to 0$? What happens as $\tau \to \infty$? What is the disadvantage of this approach compared to the least squares regression in terms of computational complexity?

3. Implementing Regression Methods in Python – 55 pts. This question will take you step-by-step through implementing and empirically studying several regression methods on the **Capital Bikesharing dataset**. We are interested in predicting the number of used bikes on a per-hour basis based on calendar features (e.g., time, weekday, holiday) and environmental measurements (e.g., humidity, temperature).

We ask that you provide us with **both** a Jupyter notebook source file as well as an exported PDF file of the notebook. Your code needs to be functional once downloaded. Non-functional code will result in losing all marks for this question. If your code is non-modular or otherwise difficult to understand, you risk losing a significant portion of the marks, even if the code is functional.

Environment setup: For this question you are strongly encouraged to use the following Python packages:

- `sklearn`
- `matplotlib`
- `numpy`
- `pandas`
- `autograd`

Note that you may **NOT** use the `sklearn` implementations for least squares regression, Poisson regression, and locally weighted least squares regression since that would trivialize the exercise. You may however use your code from the gradient descent tutorial.

3.1. *Initial data analysis – 15 pts.*

(a) **[0pt]** Load the Capital Bikesharing Dataset from the downloaded `hour.csv` file as a pandas dataframe.

(b) **[2pt]** Describe and summarize the data in terms of number of data points, dimensions, and used data types.

(c) **[5pt]** Present a single grid containing plots for each feature against the target. Choose the appropriate axis for dependent vs. independent variables.
Hint: use the `pyplot.tight_layout` function to make your grid readable.

(d) **[5pt]** Perform a correlation analysis on the data and plot the correlation matrix as a colored image. State which feature is the most positively, most negatively, and least correlated with the target column `cnt`.

(e) **[1pt]** Drop the following columns from the dataframe: `instant`, `atemp`, `registered`, `casual`, `dteday`.

(f) **[2pt]** Shuffle the dataframe's rows using `sklearn.utils.shuffle` with random state 0. Split the data into a training set and a test set on index 10000.

3.2. *Regression implementations – 40 pts.*

(a) **[5pt]** Implement the ordinary least squares regression algorithm as discussed in class. You are free to implement the closed form solution.

(b) **[3pt]** Fit the ordinary least squares regression model to the pre-processed data. Report the coefficient of determination, also known as the R^2 score, of your model. The R^2 score between the correct target labels y and the predicted target labels $\hat{y}$ can be calculated as:

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - (\sum_{i=1}^{n} y_i))^2}$$

Hint: you may use `sklearn.metrics.r2_score`.

(c) **[5pt]** You will find that the fit of the model is not very good. This is in part due to the fact that the dataset contains categorical input variables. So far, your implementation uses these categorical features as continuous inputs, which leads to not so good performance. Instead, it is advised to explicitly encode these input variables as categorical variables. Recall that one way of dealing with categorical variables is to replace them with 1-hot-encoded vectors. The following columns in the dataframe are known to be categorical: `season`, `mnth`, `hr`, `weekday`, `weathersit`. Substitute these features with 1-hot-encoded vectors in the dataframe. Use this dataframe for all upcoming questions. Make sure that you split the dataframe as described in Question 3.1 (f).
Hint: use `pandas.get_dummies`.

(d) **[2pt]** Re-fit the model with the new data and report the updated R^2 score.

(e) **[5pt]** Implement the locally weighted regression algorithm as described in Question 2.

(f) **[5pt]** Fit the locally weighted regression model to the data with $\tau = 1$. Report the R^2 score of your model. Verify whether and describe how the expected behaviour for $\tau \to 0$ and $\tau \to \infty$ as described in your answer to Question 2 holds.

(g) **[2pt]** Plot a histogram of the target variable. What distribution does the target follow?

(h) **[5pt]** Implement the Poisson regression algorithm as describe in Question 2.
Since the maximum likelihood estimate is not solvable in closed form, you will need to implement gradient descent. You may use autograd to compute the gradient of the loss function, but implement the gradient descent procedure by hand as presented in the tutorial.

(i) **[3pt]** Fit the Poisson model to the data. Report the fraction of explained Tweedie deviance, also known as the D score, of your model. The D score between the correct target labels y and the predicted target labels $\hat{y}$ can be calculated as:

$$D(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n} 2(y_i \log(y_i/\hat{y}_i) + \hat{y}_i - y_i)$$

Hint: you may use `sklearn.metrics.d2_tweedie_score` with `power=1`.

(j) **[5pt]** For all three models, report the final weights. Which are the most and least significant features in each model? Justify your answer. If the feature is categorical, report both the feature name, as well as the 1-hot index.