Assignment 2: Linear Regression and Classification

COMP 551 Winter 2025, McGill University
Contact TAs: Zahra TehraniNasab and Yijie Zhang

Released on February 4 midnight
Due on February 26 midnight

Please read this entire document before beginning the assignment.

Preamble

- This assignment is **due on February 26th at 11:59pm (EST, Montreal Time).**
- For late submission, 2^k percent will be deducted per k days of the delay. Late submission requests are handled case by case by the lead TA, Huiliang Zhang. You may post private message on Ed with title "Late submission request for Assignment 1" and select the "Assignment" tag.
- This assignment is to be completed in groups of three. All members of a group will receive the same grade except when a group member is not responding or contributing to the assignment. If this is the case and there are major conflicts, please reach out to the contact TA or instructor for help and flag this in the submitted report. Please note that it is not expected that all team members will contribute equally. However every team member should make integral contributions to the assignment, be aware of the content of the submission and learn the full solution submitted. See more solutions on Frequently Asked Questions on Groups and Assignments
- You will submit your assignment on MyCourses as a group. You must register your group on MyCourses and any group member can submit. See MyCourses or here for details.
- We recommend to use **Overleaf** for writing your report and **Google colab** for coding and running the experiments. The latter also gives access to the required computational resources. Both platforms enable remote collaborations.
- You should use Python for this and all assignments. You are free to use libraries with general utilities, such as matplotlib, numpy and scipy for Python, unless stated otherwise in the description of the task. In particular, in most cases you should implement the models and evaluation functions yourself, which means you should not use pre-existing implementations of the algorithms or functions as found in SciKit learn, and other packages. The description will specify this in a per case basis.

Synopsis

In this assignment, you will implement linear regression, logistic regression, and multiclass classification and evaluate these algorithms on two tabular datasets. The goal is to gain experience implementing these algorithms from scratch and to get hands-on experience evaluating their performances. You have two dataset options for each classification type. You can choose either dataset based on your preference

1 Task 1: Acquire, preprocess, and analyze the data

1.1 Binary Logistic Regression Datasets

Choose one of the two datasets:

- Option 1 - The Breast Cancer Wisconsin dataset:
  `https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic`
  The dataset contains 30 continuous features to predict binary `Diagnosis` target variable of malignant (1) or benign (0) tumor. There are 569 examples. The 30 numeric features were derived from digitized images of fine needle aspirate (FNA) of breast masses, such as radius, texture, smoothness, and compactness.
- Option 2: Parkinson's Disease Dataset:
  `https://archive.ics.uci.edu/dataset/174/parkinsons`
  The dataset contains biomedical voice measurements from 197 individuals with or without Parkinson's disease with the target variable `status` equal to 0 or 1, respectively. The dataset consists of 22 numerical features derived from various speech signals, such as fundamental frequency, jitter, shimmer, and harmonic-to-noise ratio.

1.2 Multiclass Classification Datasets

Choose one of the two datasets:

- Option 1 - Palmer Penguins Dataset:
  `https://www.kaggle.com/code/parulpandey/penguin-dataset-the-new-iris/input?select=penguins_size.csv`
  This is the same dataset used in Assignment 1. The task is to predict the "species" class labels (i.e., Chinstrap, Adélie, or Gentoo) based on 4 continuous features. Remove the "island" and "sex" feature before use.
- Option 2 - Wine Recognition Dataset:
  `https://archive.ics.uci.edu/dataset/109/wine`
  The task is to predict the three types of wines (i.e., `class` target variable) based on the quantities of 13 continuous features (e.g., the concentration of chemical constituents).

The essential subtasks are:

1. Load the datasets into NumPy or Pandas objects in Python.

2. Use simple regression (Module 4.1) to separately compute feature importance for each input variable. Because our target variables are discrete (binary or one-hot encoded multi-class), we can standardize them and treat them as "continuous variable" for the simple regression. Also, you may standardize the continuous input features. After that, you may compute simple regression coefficient for each feature d by $w_d = \mathbf{x}_d^\top \mathbf{y}/N$. Or equivalently (and efficiently), using matrix computation, we can compute $\mathbf{w} = \mathbf{X}^\top \mathbf{y}/N \in \mathbb{R}^{D\times 1}$ for binary and $\mathbf{W} = \mathbf{X}^\top \mathbf{Y}/N \in \mathbb{R}^{D\times C}$ for multi-class classification of C classes all in one go.
3. For binary classification, you should obtain D simple regression coefficients for D features; for multi-class of C classes, you should obtain $D \times C$ simple regression coefficients. Rank the features per class in decreasing order and examine them by horizontal barplot (e.g., `matplotlib.pyplot.barh`), where the x-axis are the regression coefficient and y-axis are the feature names. Therefore, the most positive and negative features will appear at the top and bottom of your barplot, respectively.
4. Do some simple research to make overall sense of the top/bottom features.
5. Remove any unnecessary features from the dataset (i.e. features that do not relate to the prediction). Again, same as A1, if you choose the Penguin dataset make sure you also remove the "island" feature.

2 Task 2: Implement linear regression models

As baseline models, implement multiple linear regression on the two classification tasks. Similar to the simple regression, you may standardize the binary and the one-hot-encoded class labels and treat them as continuous response variables. This enables us to train multiple linear regression $\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}$ for binary classification and multivariate and multiple linear regression for multi-class classification $\mathbf{Y} = \mathbf{X}\mathbf{W} + \mathbf{E}$ by minimizing the sum of squared errors (SSE). You may implement closed-form solutions for both models (i.e., $\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}$ for univariate and $\mathbf{W} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{Y}$ for multivariate regression), coordinate descent, or gradient descent to fit the two models.

3 Task 3: Implement Logistic and Multiclass classifiers

Following the equations presented in the lecture slides in Module 4.2 and 4.3, implement the models from scratch (i.e., you cannot use scikit-learn or any other pre-existing implementations of these methods). However, you are free to implement these models based on the code provided in the Colab notebook as you see fit. Using the numpy package, however, is allowed and encouraged.

In particular, your two main tasks are to:

1. Implement Logistic Regression using the dataset you selected in Section 1.1.
2. Implementing Multiclass Logistic Regression using the dataset you selected in Section 1.2.

Regarding the implementation, we recommend the following approach:

- Implement the Logistic Regression and Multiclass Classification models as Python classes. You should use the constructor for the class to initialize the model parameters as attributes, as well as to define

other important properties of the model.

- Your model class for each algorithm should have (at least) two functions:
  - Define a `fit` function, which takes the training data (i.e., $\mathbf{X}$ and $\mathbf{y}$)—as well as other hyperparameters (e.g., the learning rate and/or number of gradient descent iterations)—as input. This function should train your model by modifying the model parameters.
  - Define a `predict` function, which takes a set of input points (i.e., $\mathbf{X}$) as input and outputs predictions (i.e., $\hat{\mathbf{y}}$) for these points.

To verify your implementations are correct, you must provide your own code for the two tasks below so that the TAs can validate the findings you write in the report:

1. Compare the analytical gradients in your implementations with the approximate gradients computed by the small perturbation approach covered in Module 4.2 and 4.3. The difference between the analytical gradients and the approximate gradients should be very small (e.g., 10^{-6}); otherwise you have a bug.
2. Plot cross entropy loss on the training data as a function of iterations. We expect a smooth decreasing trend for both logistic regression models.

4 Task 4: Run experiments

The goal of this task is to have you explore linear classification and examine feature importance by the linear coefficients. *Evaluate the performance using AUROC for binary classification and accuracy for multi-class classification.* You are welcome to perform any experiments and analyses you see fit, **but at the minimum, you must complete the following experiments in the order stated below**:

1. Rank and barplot the features from the most positive coefficients to the most negative coefficients using simple linear regression as detailed in Task 1 (Section 1).
2. Implement from scratch or use the course colab code and train your:
   (a) **multiple linear regression** and **binary logistic regression** on the Breast Cancer or Parkinson's Disease datasets.
   (b) **multivariate and multiple linear regression** and **multi-class logistic regression** on the Palmer Penguins or Wine Recognition datasets.
3. For the logistic regression models, verify your implementation by gradient checking and training cross entropy as detailed in Task 3 (Section 3)
4. Split the data into training, validation and test using appropriate proportions as you see fit. Choose the model at the best iteration where the validation AUROC or the validation accuracy is the highest to go forward for the evaluation on the test data
5. Compare the linear regression models with the above logistic regression models in terms of AUROC and prediction accuracy on the same test set. For linear regression models, you can transform the predicted continuous values to class probabilities via sigmoid and softmax functions, respectively.

6. On the same plot, draw ROC curves and report the AUROC values of the binary logistic regression and linear regression on the binary classification task.
7. Report the test accuracies of multivariate linear regression and multi-class logistic regression on the multi-class classification test data.
8. Compare the binary regression coefficients learned by your multiple logistic regression with those simple and/or multiple linear regression coefficients in task 1 and/or task 2, respectively. You may compare the horizontal barplot from the two sets of coefficients side by side. Do the top features change and why?
9. Plot a $D \times C$ heatmap for the multi-class logistic coefficients similar to the Iris flower heatmap shown in Module 4.3. Compare this heatmap with the heatmap based on the simple and/or multiple linear regression as you did in Task 1 and/or 2, respectively. Do the top feature per class change and why?

5 Optional tasks

1. Experiment non-linear feature transformation (Module 4.1) with d-degree polynomial, Gaussian, or sigmoid basis functions. Note that both linear regression and logistic regression can operate on the transformed features.
2. Experiment regularized regression such as Ridge and LASSO. You are free to implement them yourself or use any Python library that has their implementation (e.g. `https://scikit-learn.org/0.15/modules/classes.html#module-sklearn.linear_model`).
3. Compare with other models such as KNN and Decision Trees from scikit-learn `https://scikit-learn.org/stable/modules/tree.html` for both binary and multi-class classification tasks.

Deliverables

You must submit two separate files to MyCourses (**using the exact filenames and file types outlined below**):

1. `assignment2_group-k.ipynb`: Your data processing, classification, and evaluation code should be all in one single Jupyter Notebook. Your notebook should reproduce all the results in your reports. The TAs may run your notebook to confirm your reported findings.
2. `assignment2_group-k.pdf`: Your (max 8-page) assignment write-up as a pdf (details below).

where k is your group number.

Project write-up

Your team must submit a project write-up that is a maximum of **8 pages** (single-spaced, 11pt font or larger; minimum 0.5 inch margins, an extra page for references/bibliographical content can be used). We highly recommend that students use LaTeX to complete their write-ups. You have some flexibility in how you report your results, but you must adhere to the following structure and minimum requirements:

Abstract (5+ sentences) Summarize the project task and your most important findings. For example, include sentences like "In this project we investigated the performance of linear classification models on two benchmark datasets", "We found that the logistic/multiclass regression approach achieved worse/better accuracy than multiple linear regression (and decision tree) and was significantly faster/slower to train."

Introduction (5+ sentences) Summarize the project task, the two datasests, and your most important findings. This should be similar to the abstract but more detailed. You should include background information and citations to relevant work (e.g., other papers analyzing these datasets).

Datasets (5+ sentences) Briefly describe the datasets and how you processed them. Describe the new features you come up with in detail. Present the exploratory analysis you have done to understand the data, e.g. simple regression feature weights.

Results (10+ sentences) Describe the results of all the experiments mentioned in Task 3 (at the minimum) as well as any other interesting results you find. You must report the following results:

1. Display horizontal bar plots from the Simple linear regression in Task 1 on the datasets you choose with the coefficient weight as the x-axis and the feature names as the y-axis
2. For binary and multi-class logistic regression models, report the numerical difference between your analytical gradients and the approximate gradients via numerical perturbation (Task 3)
3. Convergence plot on how the logistic and multiclass classification converge given a reasonably chosen learning rate (Task 3)
4. A single plot containing two ROC curves of logistic regression and multiple linear regression on the chosen binary classification dataset (Task 4).
5. Report the classification accuracies of multiclass classification and multivariate multiple regression on the test data (Task 4)
6. A horizontal bar plot showing the features from the logistic regression on the chosen dataset with the coefficient as the x-axis and the feature names as the y-axis
7. A heatmap showing the relationship between features (rows) and classes (columns) in the multiclass logistic classification.
8. Discussion on the findings from conducting some of the optional tasks

Discussion and Conclusion (5+ sentences) Summarize the key takeaways from the project. Discuss the reason of performance differences between linear regression and logistic regression. For example, which models are more efficient? which loss function is more appropriate for the classification task? Compare the regression coefficients between linear regression and logistic regression. Do both models rank features differently and why? Briefly investigate the meaning of the top features – do they make sense in real world applications?

Statement of Contributions (1-3 sentences) State the breakdown of the workload across the team members.

Evaluation

The assignment is out of 100 points, and the evaluation breakdown is as follows:

- Completeness (20 points)
  - Did you submit all the materials?
  - Did you run all the required experiments?
  - Did you follow the guidelines for the project write-up?
- Correctness (40 points)
  - Are your models implemented correctly?
  - Are your reported accuracies close to the reference solutions?
  - Do your selected top features actually improve performance compared to randomly chosen features in the data?
  - Do you observe the correct trends in the experiments (e.g., comparing training size)?
- Writing quality (25 points)
  - Is your report clear and free of grammatical errors and typos?
  - Did you go beyond the bare minimum requirements for the write-up (e.g., by including a discussion of related work in the introduction)?
  - Do you effectively present numerical results (e.g., via tables or figures)?
- Originality / creativity (15 points)
  - Did you go beyond the bare minimum requirements for the experiments?
  - **Note:** Simply adding in a random new experiment will not guarantee a high grade on this section! You should be thoughtful and organized in your report.

Final remarks

You are expected to display initiative, creativity, scientific rigour, critical thinking, and good communication skills. You don't need to restrict yourself to the requirements listed above - feel free to go beyond, and explore further.

You can discuss methods and technical issues with members of other teams, but **you cannot share any code or data with other teams.**