CS589: Machine Learning - Fall 2025

Homework 1: Classification

Assigned: Thursday, September 10. Due: Wednesday, September 17 at 8:00pm

Getting Started: This assignment consists of written problems and coding problems. Download the assignment archive from Canvas and unzip the file. Starter code for coding problems is provided in the code directory. For general clarification questions, please submit public posts to Campuswire. For questions related to your specific solutions, please submit private posts on Campuswire. In-person help is available through regularly scheduled office hours.

Due Date and Late Work: The assignment is due at 8:00pm ET on September 17th, 2025. Students can submit up to 11:59pm on the due date with no penalty. Work submitted up to 11:59pm on one day after the assignment is due is subject to a penalty of 10%. Work submitted up to 11:59pm two days after the assignment is due is subject to a penalty of 20%. Gradescope will close at 11:59pm two days after the assignment is due and work can not be submitted for credit after this point.

How to Submit: Your written report must be submitted to Gradescope as a PDF file. The maximum length of the report is 5 pages in 11 point font, including all figures and tables. You must select the page on which each answer appears. You are encouraged to typeset your PDF solutions using LaTeX. The source of this assignment is provided to help you get started. You may also submit a PDF containing scans of *clear* hand-written solutions. Work that is illegible will not count for credit. For this assignment, code should be submitted to Gradescope as Python 3.10+ Jupyter Notebooks. Autograding will not be used for this assignment. You may submit both the code and the report as many times as you like before Gradescope closes. Only your final submission will be graded. Any late penalties will be based on the timestamp of your final submission as determined by Gradescope.

Academic Honesty Reminder: Homework assignments are individual work. Being in possession of another student's solutions, code, code output, or plots/graphs for any reason is considered cheating. Sharing your solutions, code, code output, or plots/graphs with other students for any reason is considered cheating. Copying solutions from external sources (books, web pages, etc.) is considered cheating. Collaboration indistinguishable from copying is considered cheating. Posting your code to public repositories like GitHub (during or after the course) is not allowed. Manual and algorithmic cheating detection are used in this class. Any detected cheating will result in a grade of 0 on the assignment for all students involved, and potentially a grade of F in the course.

Generative AI Use Reminder: The use of generative AI tools to help with coding is permitted for programming problems in this course. The use of generative AI for all other work is considered cheating.

1. (*10 points*) **KNN and Data Scaling:** Suppose we have a classification problem where the training data are tuples $(y_i, \mathbf{x}_i)$ with $\mathbf{x}_i = [\mathbf{x}_{i1}, ..., \mathbf{x}_{iD}] \in \mathbb{R}^D$. Suppose we re-scale the feature vectors $\mathbf{x}'_i = [\mathbf{x}_{i1}/s_1, ..., \mathbf{x}_{iD}/s_D]$ using different scale factors $s_d > 0$ for different dimensions d. Is the KNN classification function using standard Euclidean distance invariant to Z-normalization? That is, will it always be the case that $f_{KNN}(\mathbf{x}_i) = f_{KNN}(\mathbf{x}'_i)$? Explain your answer.

2. (*10 points*) **KNN and Missing Data:** Missing data is a common problem in real-world applications of machine learning methods. In the classification case, different feature vectors $\mathbf{x}_i = [\mathbf{x}_{i1}, ..., \mathbf{x}_{iD}] \in \mathbb{R}^D$ can have missing values on any subset of the dimensions. We can keep track of which dimensions have valid data and which have missing data using a binary mask vector $\mathbf{m}_i = [\mathbf{m}_{i1}, ..., \mathbf{m}_{iD}] \in \{0, 1\}^D$. For example, if $\mathbf{x}_1 = [5.0, ?, 1.0]$, then $\mathbf{m}_1 = [1, 0, 1]$. Explain how you could modify the standard KNN classifier so that it can be applied to a data set with missing values represented as a set of tuples $(y_i, \mathbf{x}_i, \mathbf{m}_i)$.

3. (*10 points*) **Probabilistic KNN:** Prove that the conditional probability distribution output by the probabilistic KNN classifier shown below is a valid probability mass function for any input $\mathbf{x}$. Assume that there are C classes, K is the number of neighbors, and $\mathbf{x} \in \mathbb{R}^D$.

$$P_{KNN}(Y = y|\mathbf{X} = \mathbf{x}) = \frac{1}{\epsilon C + K}\left(\epsilon + \sum_{i \in \mathcal{N}_K(\mathbf{x})} \mathbb{I}(y_i = y)\right) \tag{1}$$

4. (*10 points*) **LDA Probabilistic Prediction:** Consider the Linear Discriminant Analysis model defined below for $y \in \{1, ..., C\}$ and $\mathbf{x} \in \mathbb{R}^D$. Derive a formula for $P(Y = y|\mathbf{X} = \mathbf{x})$ in terms of the model parameters π, μ_c, and Σ. Show your work.

$$P(Y = y, \mathbf{X} = \mathbf{x}) = P(Y = y)P(\mathbf{X} = \mathbf{x}|Y = y) \tag{2}$$

$$P(\mathbf{X} = \mathbf{x}|Y = y) = \frac{1}{|2\pi\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu_c)^T\Sigma^{-1}(\mathbf{x} - \mu_c)\right) \tag{3}$$

$$P(Y = y) = \pi_c \tag{4}$$

5. (*10 points*) **Optimization:** Consider the objective function $f(x) = x^4 - 2x^2 + 1$. Use analytic optimization to find all global minimizers of this function.

6. (*50 points*) **Heart Disease Classification:** In this problem, you are going to experiment with KNN, logistic regression and a large language model on the problem of heart disease classification. For this problem, you will use the heart disease data set described here: `https://archive.ics.uci.edu/dataset/45/heart+disease`. Starter Jupyter Notebooks for each method are in the code directory of the assignment package. This code includes methods for downloading and pre-processing the data set that should not be changed. You will answer the questions below in your report. You will also upload the three completed notebooks to Gradescope.

a. (*5 pts*) The data set for this problem has been split into a training and test set and categorical features have been re-encoded for use with KNN and logistic regression. In the `lr.ipynb` notebook, add code to

compute and display the number of training data cases, test data cases, and feature dimensions. List the values you find in your report as the answer to this question.

b. (*10 pts*)In the `lr.ipynb` notebook, add code to learn a standard logistic regression classifier on the heart disease training data set `Xtrain, ytrain` using the scikit-learn implementation of logistic regression[1]. Also add code to compute the training error rate of the learned model using the training data set `Xtrain, ytrain` and the test error rate using the test data set `Xtest, ytest`. As your answer to this question, include the two error rates in your report.

c. (*10 pts*) In the `knn.ipynb` notebook, add code to learn a standard KNN classifier on the heart disease training data set `Xtrain, ytrain` using the scikit-learn implementation of the KNN classifier[2]. Use Euclidean distance. Learn the model using values of K from $\{1, 10, 20, 30, ...\}$, up to the maximum multiple of 10 that is less than the training data set size. For each value of K, compute the training error rate and the test error rate. As your answer to this question, include one figure in your report showing two trend lines: the training error rate vs K, and the test error rate vs K. Make sure to label the two lines as well as the plot axes.

d. (*5 pts*) The `llm.ipynb` notebook includes starter code to configure and run the Falcon-H1-3B-Instruct model on Google Colab. This is a 3 billion parameter instruction tuned LLM (`https://huggingface.co/tiiuae/Falcon-H1-3B-Instruct`). The starter code shows how to construct a simple prompt to use the LLM to make predictions for the heart disease classification problem, but the starter prompt only uses one feature. As a first step, upload the notebook to `colab.google.com` and use "Runtime > Change Runtime Type" in the Colab menu to change the runtime to T4 GPU. Then, run the notebook to configure the model and classify the test data set. It may take 5-10 minutes to download and configure the model. Compute and report the test error rate found.

e. (*10 pts*)Next, in the `llm.ipynb` notebook, experiment with revised prompts that incorporate additional features. You can refer to the data set description and the references provided to learn more about the features to help guide how you present them to the LLM.[3] While experimenting, make sure to write code to use (all or part of) the training data set to evaluate your candidate prompts, not the test set. As your answer to this question, describe the features that you tried and the structure of your prompts. In the `llm.ipynb` notebook, add code to run the LLM on the test data set using the best prompt that you found. Report the test error rate for this prompt.

f. (*10 pts*) The initial prompt structure includes brief instructions to explain the task to the LLM. Since the model we are using is relatively small, it may not have the world knowledge needed to fully understand the problem domain and the features. In this question, try providing the LLM with additional information about the task, definitions of features or anything else that might help improve performance. You can also consider allowing the LLM to provide explanations before deciding on a final classification, but you will need to add additional output parsing. As your answer to this question, explain what you tried and report the test performance of the best prompt structure that you identified.

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html`

[3] `https://archive.ics.uci.edu/dataset/45/heart+disease`

(**g**) If you used generative AI tools to help complete any programming questions on this assignment, please briefly describe which tools you used, how you used them, and for which problems you used them. If you did not use generative AI tools, please indicate that as your response to this question.