Part 1: Regression (6 points)

For this part you'll use `data/powerplant.csv`, a dataset from a combined cycle power plant. (All files are available in the github repository.) We consider the problem of predicting y (output power in megawatts) as a function of x (ambient air temperature in centigrades).

Write $(x(i), y(i))$ for our datapoints. Given predictions $\hat{y}(i)$ meant to approximate $y(i)$, the **mean squared error** is

$$L = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}(i) - y(i))^2.$$

When the predictions $\hat{y}(i)$ are linear functions of the model parameters, minimizing the mean squared error is called a **linear least squares problem**.

(a) For the linear model $\hat{y} = \theta_0 + \theta_1 x$, give a formula for the partial derivatives of mean squared error with respect to θ_0 and θ_1. Give a general formula for the optimal parameters (should they exist). Find optimal parameters for the provided dataset and graph the fit. (1 point)

(b) Let $L(\theta)$ be the mean squared error of our linear model with parameter $\theta = (\theta_0, \theta_1)$. Putting $\theta = (0, 0)$, graph the function $L(\theta - \eta\nabla L(\theta))$ as a function of η in some neighborhood of 0. Choose a small enough range to show that $-\nabla L$ is a descent direction. (1 point)

(c) Normalize the powerplant data in some way and fit a linear model using gradient descent on mean squared error. Visualize how the model and the error change during training. (Consider using code from `iml/simple_regression.py`.) (1.5 points)

(d) Now try running gradient descent without first normalizing the power plant data. Explain why the rate of convergence is slower. (1.5 points)

(e) Fit a polynomial model $y = \theta_0 + \theta_1 x + \theta_2 x^2$ using any method you want and graph it. Explain whether fitting this model is possible without using an iterative method like gradient descent. (1 point)

Part 2: Classification (7.5 points)

For this part you'll use the iris dataset. Load it from `sklearn.datasets`.

Given a vector $z \in \mathbb{R}^k$, the **softmax** function is defined by

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}.$$

The entries of this vector are positive and sum to 1, so we can view softmax(z) as a probability distribution over the set $\{1, \ldots, k\}$.

Given a sequence of probability distributions $\hat{y}(i) \in \mathbb{R}^k$ and a sequence of observations $y(i) \in \{1, \ldots, k\}$ they were meant to predict, the **cross-entropy loss** is

$$\frac{1}{N} \sum_{i=1}^{N} - \ln \hat{y}(i)_{y(i)}.$$

When $x(i)$ are samples of some independent variables and each predictive distribution $\hat{y}(i)$ is given by softmax$(Wx(i) + b)$ for some parameters (W, b), minimizing cross-entropy loss is the problem of **multi-class logistic regression**.

(a) Give a formula for the partial derivatives of cross-entropy loss with respect to the parameters (W, b) of a multi-class logistic regression problem. (2 points)

(b) Write a class, roughly with the API of scikit-learn's `LogisticRegression`, that fits a multi-class logistic regression model using gradient descent on the cross-entropy loss. Train your model on the iris dataset and report cross-entropy loss and accuracy. (Consider normalizing your data.) (1.5 points)

(c) Train your logistic regression model with only sepal length and sepal width as input features. Make a scatter plot of these two attributes overlaid with the decision boundaries of your model. (Consider using code from `iml/simple_classifier.py`.) (1 point)

(d) Consider scikit-learn's implementation of k-nearest neighbors and decision tree. Using default hyperparameters, train these two models with sepal length and sepal width as input features. Show their decision boundaries. (1.5 points)

(e) Of the three models you used, which would you choose to predict species as a function of sepal length/width? Support your conclusion with some validation metric. (1.5 points)

Part 3: Unsupervised Methods (6.5 points)

For this part you'll use the iris dataset (from `sklearn.datasets`) and some files from the repository.

(a) Suppose we didn't have access to the species attribute of the iris dataset. Implement the Lloyd-Forgy algorithm for k-means and use it to infer a partition of the iris dataset into three species. Compute the confusion matrix and the accuracy of the partition relative to the true species. (1.5 points)

(b) Consider a vector-valued random variable $X \in \mathbb{R}^n$ and a vector $\theta \in \mathbb{R}^n$. Subject to the constraint $\|\theta\| = 1$, when is θ a strict maximizer for the variance of the inner product $\langle \theta, X \rangle$? Derive a characterization in terms of the covariance matrix of X and illustrate this problem with a two-dimensional dataset of your choosing. (2 points)

(c) Investigate `data/ulu.csv` using PCA and explain how this data was produced. Using scikit-learn, run both k-means clustering and dbscan. Which method produces more reasonable "clusters"? (1.5 points)

(d) `data/low_rank.csv` contains 100 datapoints in 20 dimensions. You can imagine this is sensor data from some physical system sampled over a short period of time. Using PCA, produce a better estimate of the quantity measured by the first sensor. Graph the raw sensor value and your denoised estimate as a function of time. Briefly explain your approach. (1.5 points)

Challenge Problem: Retail Dataset (4 points)

The online retail dataset from the UCI machine learning repository contains about 500,000 transactions from an online retailer in the UK recorded over a span of a year. It's distributed as an Excel sheet, which is a common industry format. You can use either pandas or polars to load and process it.

The challenge problem isn't mandatory to receive full marks for the assignment. If you solve the challenge problem and get more than 20 points in total, your final grade for the assignment will be 20.

(a) Explore the online retail dataset. Train a simple model on some aspect of the data and explain what it tells us. (It should help to put some effort into your preprocessing/exploratory analysis!) (4 points)