Homework 4

1. *Constant predictors.* Consider a constant predictor of a scalar y, of the form $g(\theta) = \theta$. (We remove the dependence of g on the feature vector x, since there isn't a feature vector in this case.) You will find the outcome vector `y` for this problem in `constant_predictors.json`.

   (a) Find the constant predictor parameter θ^{mse} that minimizes mean square error on the given datatset. Report θ^{mse} up to two decimal points.

   (b) Find the constant predictor parameter θ^{aae} that minimizes average absolute error on the given datatset. Report θ^{aae} up to two decimal points.

   (c) *Comparison.* Give a table that shows the root mean square (RMS) and average absolute error (AAE) for the two constant predictors with parameters θ^{mse} and θ^{aae} on the given dataset. Report these numbers up to two decimal points. (Your table will contain 4 numbers.)

   Hint. Consider plotting or viewing the entries of `y`.

2. *Fitting the leaf values in a tree predictor.* We consider a decision tree predictor for a scalar outcome y. Suppose the decision tree is fixed, *i.e.*, for each non-leaf vertex, we fix the feature to split on, and we fix the threshold. To fully specify the decision tree predictor, we need to give the value of $\hat{y}$ for each leaf vertex. We denote the leaf values as θ_j, for $j = 1, \dots, p$, where p is the number of leaves in the decision tree. We collect these leaf values into a parameter vector $\theta \in \mathbf{R}^p$, and focus on how to choose θ using ERM, given the data $y^1, \dots, y^n$.

   We let $\mathcal{L}(x) \in \{1, \dots, p\}$ denote the leaf that the feature $x \in \mathbf{R}^d$ falls in. Let $\mathcal{I}_j = \{i \mid \mathcal{L}(x^i) = j\}$ denote the set of indices of our data set for which the feature lies in leaf j, for $j = 1, \dots, p$. We will assume that each of these sets is nonempty, *i.e.*, at least one feature in our data set falls in each of the leaves.

   (a) Explain how to choose θ using ERM with quadratic loss.

   (b) Explain how to choose θ using ERM with absolute loss.

   For both cases, you can give your answer in English, and without justification.

3. *Sequential outlier removal.* We consider the problem of fitting data corrupted with outliers, using a simple sequential outlier removal method. In `outlier_rem.json`, you will find a 250×10 matrix `U_train` and a 250-vector `v_train` consisting of raw training input and output data, and a 250×10 matrix `U_test` and a 250-vector `v_test` consisting of raw test input and output data, respectively. We will work with input and output embeddings $x = \phi(u) = u$ and $y = \psi(v) = v$. and you will use a simple linear predictor (without a constant feature) with square loss to fit the model. We will judge model performance using the RMS error on the test set.

A number of the output data entries in the training set have been corrupted (but in a non-obvious way). You do not know which data points have been corrupted, or how many, but you can assume no more than 50. You will explore a simple sequential method to remove the corrupted data points and form a prediction model.

Repeat the following for 50 iterations:

- Create a linear predictor from the training data set.
- Find the data point in the training data set with the largest prediction error.
- Remove the data point from the training data set.

This results in 50 predictors. Plot the test RMS error for them, versus the number of points removed. Give a guess as to how many of the data points were corrupted, with justification.

Julia hint. To remove row `i` from a matrix `X` and a vector `y`, use `X[setdiff(1:end, i), :]` and `y[setdiff(1:end, i)]`, respectively.