Assignment 4

STAT 541, Winter 2025

Instructor: Andrew McCormack, mccorma2@ualberta.ca
Due date: March 28th, 11:59 PM.

To receive full marks you must show your work for all derivations.

Problem 1: Classification and Regression Trees

(a) Can a classification tree $\hat{f}$ with 3 leaves (and hence only two splits) for a binary classification problem with a two-dimensional feature space ($p = 2$) fit the training data

$$\begin{aligned}\mathbf{x}^{(1)} &= (0,0),\ y^{(1)} = 0\\ \mathbf{x}^{(2)} &= (1,0),\ y^{(2)} = 1\\ \mathbf{x}^{(3)} &= (0,1),\ y^{(3)} = 1\\ \mathbf{x}^{(4)} &= (1,1),\ y^{(4)} = 1\end{aligned}$$

exactly with zero training error so that $\hat{f}(\mathbf{x}^{(i)}) = y^{(i)}$, $i = 1, 2, 3, 4$? What about for the following training data?

$$\begin{aligned}\mathbf{x}^{(1)} &= (0,0),\ y^{(1)} = 0\\ \mathbf{x}^{(2)} &= (1,0),\ y^{(2)} = 1\\ \mathbf{x}^{(3)} &= (0,1),\ y^{(3)} = 1\\ \mathbf{x}^{(4)} &= (1,1),\ y^{(4)} = 0\end{aligned}$$

(b) A regression tree for a univariate feature space $\mathcal{X} = \mathbb{R}^1$ is simply a piecewise constant function. Suppose that we have determined that the splits for our regression tree should occur at $x = -3, 7$ and so the regression tree has the form

$$f(x) = \beta_1 I(x \leq -3) + \beta_2 I(-3 < x \leq 7) + \beta_3 I(x > 7) \tag{1}$$

where $I(\cdot)$ is an indicator function. It remains to determine the coefficients $\beta_1, \beta_2, \beta_3$ via least squares. Given training data $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$ show that the design matrix $\mathbf{X} \in \mathbb{R}^{n \times 3}$ has columns that are orthogonal to each other. Use this to find a simple expression for the OLS estimates $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$.
Note that this result easily extends to regression trees with features in $\mathbb{R}^p$.

(c) The Gini index at a node R_i for a binary classification problem is given by

$$\text{Gini Index} = \hat{p}_0 \hat{p}_1$$

where $\hat{p}_0, \hat{p}_1$ are the proportion of the observations from class zero and class one in the node R_i. For training observations $(x^{(i)}, y^{(i)})$, $i = 1, \ldots, 5$ equal to $(1,0), (2,0), (3,1), (4,0), (5,0)$ (here the dimension of the feature space is $p = 1$) fit a classification tree using the Gini index as a node impurity measure. How many splits are required to find an exact fit to the training data? Does this procedure make

sensible first split and how is this related to the fact that the splitting procedure is a greedy optimization algorithm?

Recall that each split should be chosen to minimize the sum over each node of the Gini index impurity measure. If there are ties in the impurity measure sum you may choose any split that minimizes this sum.

Problem 2: Bagging and Model Averaging

Bagging is a general technique that can reduce the variance of a prediction algorithm.

(a) Let $\hat{f} = \frac{1}{B}\sum_{i=1}^{B} \hat{f}^{(i)}$ be a bagged prediction function for linear regression where each $\hat{f}^{(i)}(\mathbf{x}) = \mathbf{x}^\top \hat{\boldsymbol{\beta}}^{(i)}$ is a linear regression model. Every $\hat{\boldsymbol{\beta}}^{(i)}$ is the OLS estimate fit using the ith data set $\mathcal{D}^{(i)}$ of size n that is resampled with replacement from the original training data. In what class of functions does $\hat{f}$ lie in? How is $\hat{f}$ related to each of the individual regression functions $\hat{f}^{(i)}$?

(b) We use the `Hartnagel` data in the `carData` package and run linear regressions with `degrees` as a univariate feature and `fconvict` as the response.

(i) Draw $B = 2000$ resampled data sets $\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(2000)}$ where each $\mathcal{D}^{(i)}$ has size $n = 38$ and is sampled from the training data *with replacement*.

(ii) Fit a polynomial regression with a degree 5 polynomial to each of the resampled data sets $\mathcal{D}^{(i)}$. This produces 2000 different estimated regression functions $\hat{f}^{(1)}, \ldots, \hat{f}^{(2000)}$.

(iii) Fix an $x \in \{0, \ldots, 100\}$ and find the mean, median, 90% quantile and 10% quantile of the numbers $\hat{f}^{(1)}(x), \ldots, \hat{f}^{(2000)}(x)$.

(iv) Repeat the above step for every $x \in \{0, \ldots, 100\}$. Plot the curves, as a function of x, of the means (this is bagging estimate), medians, and both quantiles.

(v) Comment on where in the feature space $x \in [0, 100]$ there appears to be the most uncertainty about the predictions obtained by this regression model.

(c) Using the same data as in (b) fit polynomial regression models of degrees $1, 2, 3, 4, 5$ to the full data, giving prediction functions $\hat{g}^{(1)}, \ldots, \hat{g}^{(5)}$. For $x = 0, \ldots, 100$ compute the model average estimate $\hat{g}(x) = \frac{1}{5}\sum_{i=1}^{5} \hat{g}^{(i)}(x)$ and plot the curve $\hat{g}(x)$ for $x \in [0, 100]$. How good is this fit?

(d) What kind of function is $\hat{g}$ in part (c) and how does it relate to the $\hat{g}^{(i)}$'s?

Problem 3: Neural Networks

(a) How many parameters need to be estimated in an a feedforward neural network where: the dimension of the input feature space is p, there are two hidden layers, the first with 4 hidden units, the second with 5 hidden units, every hidden unit is connected to every unit in the previous layer, and each hidden unit along with the output unit contains an intercept parameter?

(b) Even in the relatively simple network in (a) there are a large number of parameters to be estimated. This is usually done by gradient descent (or a stochastic variant of this). Computing the gradients could potentially be very expensive. The technique of *backpropagation* allows us to avoid repeating the same computation when finding the gradients of iterated function compositions and makes fitting neural networks computationally feasible. We examine the general idea of backpropagation in simple case where we have the functions $f(g(h(x))$, $f(g(y))$, $f(z)$ and want to compute the derivative of each function. Define $y_0 = h(x_0)$ and $z_0 = g(y_0)$. Using the chain rule compute

$$\frac{d}{dx} f(g(h(x))|_{x=x_0} \tag{2}$$

$$\frac{d}{dy} f(g(y))|_{y=y_0}. \tag{3}$$

The first step (of the "forward pass") of backpropagation is to compute x_0, y_0, z_0. The next step (the first step of the "backward pass") is to compute $f'(z_0)$. Next we compute (3). Find a simple expression for (3) that uses $f'(z_0)$ which we have already computed. The final step is to compute (2). Find a simple expression for (2) involving (3).

(c) Neural networks can approximate linear functions easily. For simplicity assume that $p = 1$ and as in (c) the neural produces a function of the form

$$f_{\boldsymbol{\beta}}(x) = \beta_3 + \beta_2 \sigma(\beta_1 x + \beta_0).$$

Suppose we want to approximate $g(x) = 3x - 1$. Set $\beta_0 = 0$, $\beta_1 = \frac{1}{k}$, $\beta_2 = 12k$, $\beta_3 = -1 - 6k$. With these choices of the β_i show that $f_{\boldsymbol{\beta}}(0) = -1$ and $f'_{\boldsymbol{\beta}}(0) = 3$ for every k. For $k = 2, 5, 10$ plot this function for $x \in [-10, 10]$ along with $g(x)$.

The idea here is that in a very small region of zero the function $\sigma(z)$ looks linear. For k large $k\sigma(z/k)$ looks like this same linear function but now across a wider range of z. Note that the same idea as above can be generalized to p-dimensional feature spaces where linear functions of the form $\sum_{i=1}^{p} \alpha_i x_i$ can be approximated by a function of the form $\beta_{p+2} + \beta_{p+1}\sigma(\sum_{i=1}^{p} \beta_i x_i + \beta_0)$.

Problem 4: Generalized Additive Models

In this problem we examine the `Soils` dataset in the `carData` package. We will fit the soil `pH` as a function of `Mg`, `Ca`, `P`, and `K`.

(a) Using the `gam` function in the `gam` package fit a GAM to `pH` where the constituent univariate functions in the GAM, f_{Mg}, f_{Ca}, f_P, f_K, are all smoothing splines. To do this you can use `s(Mg),s(Ca),...` etc. in the model formula.

Look at the help documentation ?gam if you need more information. It is good to get experience with deciphering this documentation. Pages 318-321 of ISLR also contain some nice examples.

(b) Repeat step (a) but using univariate functions that are local regressions. To do this you can use `lo(Mg),lo(Ca),...` etc. in the model formula.

(c) Apply `plot` to both of these GAMs. Each GAM will produce plots of the estimated functions $\hat{f}_{Mg}(x_{Mg}), \hat{f}_{Ca}(x_{Ca}), \hat{f}_P(x_P), \hat{f}_K(x_K)$.

(d) For each variable `Mg`, `Ca`, `P`, `K` fit a simple linear regression of pH vs the single variable along with the observed points of pH and this variable (e.g. For Mg fit the linear model pH $\sim$ Mg and plot Mg and pH along with the linear fit). Combine this plot with the previous eight plots into a grid so you can visualize every plot at once. You may want to use the command `par(mfrow = c(3,4), mar = c(0,0,0,0))` for this (If you ever need to reset the plotting parameters `par` then you can use the command `dev.off()` to do this).

(e) What do you notice about how the two GAM fits for each variable compare to the individual linear model fits? Do all of these plots show the same overall trends? Provide an explanation for why the GAM fits behave in this manner. (**Hint:** The GAM plots display fits for each variable when the other variables have *already been included* in the model.)