TTIC 31020: Introduction to Machine Learning
Autumn 2019

Problem Set #1

Was due: Friday October 18

Solution for Problem 1 [10 points]

$$\begin{aligned}
\mathbf{w}' &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}' && (1)\\
&= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top(a\mathbf{y}+\mathbf{b}) \qquad \text{where } \ \mathbf{b} = [b\ b\ \cdots\ b]^\top && (2)\\
&= a(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} + (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{b} && (3)\\
&= a\mathbf{w}^* + (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{b} && (4)
\end{aligned}$$

Seems like we cannot simplify the expression $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{b}$ anymore and will have to look at the data.

But let's take a closer look at this term. Given that the first column of $\mathbf{X}$ is one, we can express $\mathbf{b}$ as:

$$\mathbf{b} = \mathbf{X}\mathbf{b}' \qquad \text{where } \ \mathbf{b}' = [b\ 0\ 0 \cdots\ 0]^\top$$

Thus,

$$\begin{aligned}
\mathbf{w}' &= a\mathbf{w}^* + (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{X}\mathbf{b}' && (5)\\
&= a\mathbf{w}^* + \mathbf{b}' && (6)
\end{aligned}$$

In other words, we scale all the entries of $\mathbf{w}^*$ by a and add b to just the bias term w_0. An intuitive explanation for this dependence on the terms can be thought of as:

- *Additive term* (b): We are shifting all the targets by b. This shift can be accounted by shifting the bias term of the weight vector by the same amount.
- *Multiplicative term* (a): Scaling the targets can be matched by scaling the weight vector.

There is an alternative, perhaps more intuitive way to see what $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{b}$ is: this is simply the parameter vector for least squares regression fit to data consisting of $\mathbf{X}$ and the label vector $\mathbf{b}$, i.e., all the examples have the exact same label b! Such a "regression" can be trivially shown to produce a constant function $\widehat{y}(\mathbf{x}) = b$, which really ignores $\mathbf{X}$.

End of solution, Problem 1

Solution for Problem 2 [10 points]

The scaling of the features can be expressed by the following matrix operation:

$$\widetilde{\mathbf{X}} = \mathbf{X}\mathbf{C} \quad \text{where } \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & c_1 & 0 & \cdots & 0 \\ 0 & 0 & c_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & c_d \end{bmatrix} \tag{7}$$

(The 1 in the first column is for the bias feature.)

If we assume that the c_j's are non-zero, i.e., we are not eliminating any features, $\mathbf{C}$ is invertible. Then, $\mathbf{w}'$ can be computed as:

$$\begin{aligned}
\mathbf{w}' &= (\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^\top\mathbf{y} && (8) \\
&= (\mathbf{C}^\top\mathbf{X}^\top\mathbf{X}\mathbf{C})^{-1}\mathbf{C}^\top\mathbf{X}^\top\mathbf{y} && (9) \\
&= (\mathbf{C}\mathbf{X}^\top\mathbf{X}\mathbf{C})^{-1}\mathbf{C}\mathbf{X}^\top\mathbf{y} \qquad \text{(transpose of a diagonal matrix)} && (10) \\
&= \mathbf{C}^{-1}\mathbf{X}^{-1}(\mathbf{X}^\top)^{-1}\mathbf{C}^{-1}\mathbf{C}\mathbf{X}^\top\mathbf{y} && (11) \\
&= \mathbf{C}^{-1}(\mathbf{X}^\top\mathbf{X})^{-1}(\mathbf{C}^{-1}\mathbf{C})\mathbf{X}^\top\mathbf{y} && (12) \\
&= \mathbf{C}^{-1}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} && (13) \\
&= \mathbf{C}^{-1}\mathbf{w}^* && (14)
\end{aligned}$$

The inverse of a diagonal matrix is another diagonal matrix with the diagonal entries being the reciprocals of the original matrix. Hence,

$$\mathbf{w}' = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{c_1} & 0 & \cdots & 0 \\ 0 & 0 & \frac{1}{c_2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{c_d} \end{bmatrix} \quad \tilde{\mathbf{w}} = \begin{bmatrix} w_0 \\ \frac{w_1}{c_1} \\ \frac{w_2}{c_2} \\ \vdots \\ \frac{w_d}{c_d} \end{bmatrix} \tag{15}$$

If any of the c_j is zero, then $\mathbf{C}$ is not invertible. This case means that one or more features have been removed (set to zero in all examples). Is it still possible to compute $\tilde{\mathbf{w}}$ from $\mathbf{w}^*$ without looking at the data in this case? One can construct a counter-example showing that it is not possible.

End of solution, Problem 2

Solution for Problem 3 [10 points]

$$P(\mathbf{X}, \mathbf{y}, \mathbf{w}) = \prod_{i=1}^{n} \frac{e^{-\frac{(y_i - \mathbf{w}^\top \mathbf{x}_i)^2}{2\sigma^2_{\mathbf{x}_i}}}}{\sqrt{2\pi\sigma^2_{\mathbf{x}_i}}} \tag{16}$$

using log likelihood,

$$\log P(\mathbf{X}, \mathbf{y}, \mathbf{w}) = \sum_{i=1}^{n} (-\frac{(y_i - \mathbf{w}^\top \mathbf{x}_i)^2}{2\sigma^2_{\mathbf{x}_i}} - 1/2 \log 2\pi - \log \sigma_{\mathbf{x}_i}) \tag{17}$$

only the first term is relevant to $\mathbf{w}$, so it is equivalent to maximizing the sum of the first term,

$$\log P(\mathbf{X}, \mathbf{y}, \mathbf{w}) = -\sum_{i=1}^{n} \frac{(y_i - \mathbf{w}^\top \mathbf{x}_i)^2}{2\sigma^2_{\mathbf{x}_i}} \tag{18}$$

calculate and set the derivative to zero

$$\frac{d \log P(\mathbf{X}, \mathbf{y}, \mathbf{w})}{d\mathbf{w}} = \sum_{i=1}^{n} \frac{2(y_i - \mathbf{w}^\top \mathbf{x}_i)\mathbf{x}_i}{2\sigma^2_{\mathbf{x}_i}} = 0 \tag{19}$$

$$\sum_{i=1}^{n} \frac{\mathbf{w}^\top \mathbf{x}_i \mathbf{x}_i}{\sigma^2_{\mathbf{x}_i}} = \sum_{i=1}^{n} \frac{y_i \mathbf{x}_i}{\sigma^2_{\mathbf{x}_i}} \tag{20}$$

$$\tag{21}$$

Let

$$\mathbf{\Sigma} = \begin{bmatrix} 1/\sigma^2_{\mathbf{x}_1} & 0 & 0 & \dots & 0 \\ 0 & 1/\sigma^2_{\mathbf{x}_2} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1/\sigma^2_{\mathbf{x}_N} \end{bmatrix} \tag{22}$$

We can rewrite the previous equation as,

$$(\mathbf{X}^\top \mathbf{\Sigma} \mathbf{X})\mathbf{w} = \mathbf{X}^\top \mathbf{\Sigma} \mathbf{y} \tag{23}$$

$$\mathbf{w} = (\mathbf{X}^\top \mathbf{\Sigma} \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{\Sigma} \mathbf{y} \tag{24}$$

Hence if we do not have the exact values of $\sigma^2_{\mathbf{x}}$, $\mathbf{w}$ is not solvable.

End of solution, Problem 3

Solution for Problem 4 [10 points]

If we do know all the values of $\sigma^2_{\mathbf{x}}$, we can plug those into (24) and obtain the closed-form solution (or use gradient descent). Looking at the objective in (18), we can develop a simple interpretation: in this objective, the squared loss terms are *weighted* according to inverse of $\sigma^2_{\mathbf{x}_i}$. This makes sense: for larger $\sigma^2_{\mathbf{x}_i}$, the noise component is more likely to be of larger magnitude; therefore, the observed values of y_i are more likely to differ significantly from those predicted by the optimal model; therefore, we should "trust" them less and give them less power in determining the regression model than we give to examples with smaller $\sigma^2_{\mathbf{x}_i}$.

End of solution, Problem 4

Solution for Problem 5 [15 points]

See the notebook.

End of solution, Problem 5

Solution for Problem 6 [15 points]

See the notebook.

End of solution, Problem 6

Solution for Problem 7 [15 points]

See the notebook for code/typical results.

The main observation here: a model trained with a certain loss is expected to optimize that particular loss. Hence, there is no reason that the model trained with symmetric squared loss would do well on asymmetric loss, and

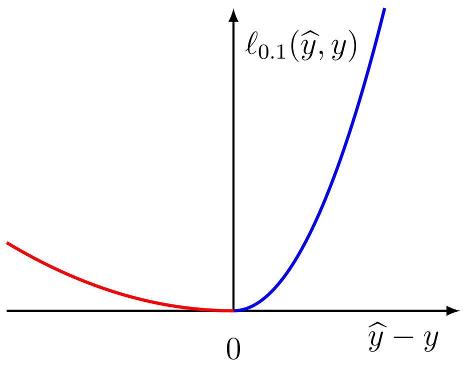

Figure 1: Plot of asymmetric loss, $\alpha = 0.1$; red part corresponds to loss on underpredicted values, blue part to loss on over-predicted values.

vice versa. If you observed such behavior, it is most likely due to failure to optimize one of the models properly. (Note that the objectives in both cases are convex; the plot of asymmetric loss in Figure 1 demonstrates this. Hence we can expect to optimize those objectives nearly perfectly, up to numerical issues and optimization algorithm behavior.)

Given the motivation for the use of asymmetric loss, we should choose the model that does better with respect to that loss on the validation set. The performance on the "normal", symmetric squared loss is irrelevant, since this is not the quantity we care about in the end when we deploy our predictor.

End of solution, Problem 7