
CS689: Machine Learning - Fall 2019

Homework 1

Sankaran Vaidyanathan (svaidyanatha@cs.umass.edu)

1. Optimal Predictions for Zero-One Loss

a. Using the given loss criterion, we can find the optimal prediction function f_* as follows:

$$\begin{aligned} f_* &= \arg \min_f \mathbb{E}_{P(x,y)} [L_{01}(y, f(x))] \\ &= \arg \min_f \int_{x \in R} \sum_{y \in \{0,1\}} L_{01}(y, f(x)) P(X = x, Y = y) dx \\ &= \arg \min_f \mathbb{E}_{P(x)} \left[\mathbb{E}_{P(y|x)} [L_{01}(y, f(x))] \right] \\ &= \mathbb{E}_{P(x)} \left[\arg \min_f \mathbb{E}_{P(y|x)} [L_{01}(y, f(x))] \right] \end{aligned}$$

Here, the last step is a result of the x 's being i.i.d. Considering the optimal prediction for a given x , we can write the objective as follows:

$$\begin{aligned} f_*(x) &= \arg \min_f \mathbb{E}_{P(y|x)} [L_{01}(y, f(x))] \\ &= \arg \min_f \sum_{y \in \{0,1\}} L_{01}(y, f(x)) P(Y = y|X = x) \end{aligned}$$

The above function is the optimization objective. Here, $L_{01}(y, f(x)) = \begin{cases} 0 & \text{if } y = f(x) \\ 1 & \text{if } y \neq f(x) \end{cases} = 1 - [y = f(x)]$

b. We have:

$$\begin{aligned} f_*(x) &= \arg \min_f \sum_{y \in \{0,1\}} L_{01}(y, f(x)) P(Y = y|X = x) \\ &= \arg \min_f \sum_{y \in \{0,1\}} (1 - [y = f(x)]) P(Y = y|X = x) \\ &= \arg \min_f \left\{ \sum_{y \in \{0,1\}} P(Y = y|X = x) - \sum_{y \in \{0,1\}} [y = f(x)] P(Y = y|X = x) \right\} \\ &= \arg \min_f \left\{ 1 - \sum_{y \in \{0,1\}} [y = f(x)] P(Y = y|X = x) \right\} \\ &= \arg \min_f \left\{ \sum_{y \in \{0,1\}} [y = f(x)] - \sum_{y \in \{0,1\}} [y = f(x)] P(Y = y|X = x) \right\} \end{aligned}$$

$$\begin{aligned}
&= \arg \min_f \left\{ \sum_{y \in \{0,1\}} [y = f(x)] (1 - P(Y = y|X = x)) \right\} \\
&= \arg \max_f \left\{ \sum_{y \in \{0,1\}} [y = f(x)] P(Y = y|X = x) \right\} \\
&= \arg \max_{y \in \{0,1\}} P(Y = y|X = x)
\end{aligned}$$

The last expression appears because the term in the sum goes to zero if $y \neq f(x)$; when $y = f(x)$, the term is maximized by the value of y with highest conditional probability $P(Y = y|X = x)$.

1. (40 points) Logistic Regression with Centering: In the standard binary logistic regression model, the probability of the class variable $y \in \{-1, 1\}$ is modeled as a logistic function of the input \mathbf{x} and the parameters $\theta = [\mathbf{w}, b]$:

$$P(Y = y|\mathbf{X} = \mathbf{x}, \theta) = \frac{1}{1 + \exp(-y(\mathbf{w}\mathbf{x}^T + b))}$$

A common pre-processing approach when applying machine learning models is to pre-center the inputs so the mean on each dimension is 0. We can instead learn an optimal centering of the inputs by augmenting the logistic regression model with a vector of centering parameters \mathbf{c} . The augmented parameter set is $\theta = [\mathbf{w}, \mathbf{c}, b]$ and the augmented model is:

$$P(Y = y|\mathbf{X} = \mathbf{x}, \theta) = \frac{1}{1 + \exp(-y(\mathbf{w}(\mathbf{x} - \mathbf{c})^T + b))}$$

In this question, we will implement and analyze this augmented model. As the learning objective function, we will minimize the regularized negative conditional log likelihood:

$$\mathcal{L}(\mathcal{D}, \theta) = \sum_{n=1}^N \log(1 + \exp(-y_n(\mathbf{w}(\mathbf{x}_n - \mathbf{c})^T + b))) + \lambda \|\mathbf{w}\|_2^2 + \lambda \|\mathbf{c}\|_2^2 + \lambda \cdot b^2$$

a. (5 pts) Find the gradient of $\mathcal{L}(\mathcal{D}, \theta)$ with respect to \mathbf{w} , \mathbf{c} , and b . Show your work.

b. (20 pts) Starting from the provided template, implement a Scikit-Learn compatible class for this model. For learning, use the `fmin_l_bfgs_b` optimizer from `scipy.optimize` with the `disp=10` option to see output from the optimizer. Initialize all model parameters to 0. As your answer to this question, describe any issues you had implementing the model and what approach you used to fix them.

c. (5 pts) Learn the model on the provided training data set with $\lambda = 1e - 6$, and evaluate it by computing its average prediction error on the test set. The entire learning procedure should take a few seconds. Report the average test error that you find as your answer to this question.

d. (5 pts) Using the provided data, learn a standard logistic regression model using scikit-learn's `linear_model.LogisticRegression` implementation with $C = 10^6$ and `solver='lbfgs'`. Leave the other model hyper-parameters at their default values. Evaluate the standard model by computing its average prediction

error on the test set. Report the average test error that you find as your answer to this question.

e. (5 pts) If your implementation is correct, you should find that the augmented and standard models perform nearly identically. Explain why you think this is the case, providing as much detail as possible.

2. (45 points) Regression with Outliers: One important problem with standard least-squares linear regression is that outliers can significantly corrupt learning. One way to combat this problem is with regression loss functions that are less sensitive to larger errors. Consider the robust prediction loss function shown below where $\delta > 0$ is a parameter of the loss.

$$L_\delta(y, y') = \delta^2(\sqrt{1 + (y - y')/\delta^2} - 1)$$

Using this robust loss for a linear regression model $f(\mathbf{x}_n, \theta) = \mathbf{w}\mathbf{x}_n^T + b$ results in the learning objective

$$\mathcal{L}_\delta(\mathcal{D}, \theta) = \sum_{n=1}^N L_\delta(y_n, f(\mathbf{x}_n, \theta))$$

Use this model, objective function, and loss to answer the following questions. Assume that $\mathbf{x} \in \mathbb{R}^D$ and $y \in \mathbb{R}$ in your derivations and code.

a. (5 pts) Produce a plot of the robust loss function $L_\delta(y, y')$ for $\delta \in \{0.1, 1, 10\}$ where the x-axis corresponds to the prediction error $\epsilon = y - y'$ and the y-axis corresponds to the value of the loss. In the same figure, also include a similar plot of the standard squared loss $L_{sq}(y, y') = (y - y')^2$. Use these plots to explain why the loss $L_\delta(y, y')$ could result in less sensitivity to outliers than the squared loss $L_{sq}(y, y')$.

b. (5 pts) Find the gradient of $\mathcal{L}_\delta(\mathcal{D}, \theta)$ with respect to \mathbf{w} and b . Show your work.

c. (20 pts) Starting from the provided template, implement a Scikit-Learn compatible class for this model. For learning, use the `fmin_l_bfgs_b` optimizer from `scipy.optimize` with the `disp=10` option to see output from the optimizer. Initialize all model parameters to 0. As your answer to this question, describe any issues you had implementing the model and what approach you used to fix them.

d. (5 pts) Using the provided data set, apply the model using $\delta = 1$. Also apply Scikit-Learn's standard least-squares linear regression model `linear_model.LinearRegression`. Report the MSE achieved by both models on the train data set.

e. (5 pts) Provide a single figure that includes a scatter plot of the training data, the regression line found using the robust model (with $\delta = 1$), and the regression line found using the standard least-squares model.

f. (5 pts) Based on this plot, which model do you think would have better generalization performance on future test data? Explain your answer.