Homework 3

Due April 23, 2020 by 11:59pm

Instructions: This homework consists of a reading assignment, one mathematical exercises, and one coding exercises. Please submit your solutions via Gradescope. Solutions should consist of three files: a PDF containing your solutions to the *non-coding questions*; a Jupyter notebook (`.ipynb` file) and an html print-out (`.html` file) with your solution to the *coding exercise*. **All coding exercises must be completed in Python.** Please be sure to comment the code appropriately. Students are encouraged to discuss homework problems, particularly on Canvas and in the TA hours, but must submit their own solutions.

Reading Assignments

- Review Lecture 3.
- Review Lab 3 (available from the Course Materials page on Canvas).
- Read and explore `distill.pub/2017/momentum/`.

Problems

Please submit your solutions as a single PDF file under the `Homework 3 - Problems` assignment in Gradescope.

1 Exercise 1

Suppose we estimate the regression coefficients in a logistic regression model by minimizing

$$F(\beta) := \frac{1}{n}\sum_{i=1}^{n} \log\left(1 + \exp(-y_i\, x_i^T \beta)\right) \; + \lambda \|\beta\|_2^2$$

for a particular value of λ in the interval $(0, \lambda_{\max}]$, with $\lambda_{\max} \gg 0$. For parts (a) through (e), indicate which of (i) through (v) is correct. Justify your answer.

(a) As λ decreases from $\lambda_{\max}$ to 0, the misclassification error on the training set will:

(i) Steadily increase.

(ii) Steadily decrease.

(iii) Increase initially, and then eventually start decreasing in an inverted U shape.

(iv) Decrease initially, and then eventually start increasing in a U shape.

(v) Remain constant.

(b) Suppose now that we compute the misclassification error on a large test set consisting of previously-unseen observations drawn from the same distribution as the training set. Which of the patterns (i) – (v) do we expect to observe as λ decreases from $\lambda_{\max}$ to 0? Again, justify your answer.

Coding

Please submit your solutions the coding exercise below as a Jupyter notebook (`.ipynb` file) and an html print-out (`.html` file) under the `Homework 3 - Coding` assignment in Gradescope. **Please run all cells in your notebook prior to submission, so we can view their output.**

2 Exercise 2

In this exercise, you will implement in **Python** a first version of *your own fast gradient algorithm* to solve the ℓ_2^2-regularized logistic regression problem. Recall from the lectures that the logistic regression problem writes as

$$\min_{\beta \in \mathbb{R}^d} F(\beta) := \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \, x_i^T \beta)\right) \; + \lambda \|\beta\|_2^2 \, . \tag{1}$$

We use here the machine learning convention for the labels that is $y_i \in \{-1, +1\}$. The fast gradient algorithm is outlined in Algorithm 1. The algorithm requires a subroutine that computes the gradient for any β.

Algorithm 1 Fast Gradient Algorithm

input step-size η_0, target accuracy ε
initialization $\beta_0 = 0$, $\theta_0 = 0$
repeat for $t = 0, 1, 2, \ldots$
Find η_t with backtracking rule
$\beta_{t+1} = \theta_t - \eta_t \nabla F(\theta_t)$
$\theta_{t+1} = \beta_{t+1} + \frac{t}{t+3}(\beta_{t+1} - \beta_t)$
until the stopping criterion $\|\nabla F\| \leq \varepsilon$.

(a) Assume that $d = 1$ and $n = 1$. The sample is then of size 1 and boils down to just (x, y). The function F writes simply as

$$F(\beta) = \log(1 + \exp(-yx \, \beta) \; + \lambda \beta^2 \, . \tag{2}$$

Compute and write down the gradient ∇F of F.

(b) Assume now that $d > 1$ and $n > 1$. Using the previous result and the linearity of differentiation, compute and write down the gradient $\nabla F(\beta)$ of F.

(c) Consider the smarket dataset from *Introduction to Statistical Learning*. Download the data:

```
import pandas as pd
file = 'https://raw.githubusercontent.com/JWarmenhoven/ISLR-python/master/Notebooks/Data/Smarket.csv'
smarket = pd.read_csv(file, sep=',', header=0, index_col=0)
```

This dataset contains trading information for the S&P 500 over 1250 days from 2001 to 2005. For each date we have the percent return from the previous 5 days (the `Lag` features), the volume of shares traded (in billions), the percent return on the date itself (`Today`), and whether the market moved up or down (`Direction`). We will apply our gradient descent and fast gradient descent algorithms to fit a logistic regression model for the binary outcome `Direction` based on the features `Lag1`, `Lag2`, and `Volume`.

(d) Construct the matrix of features and the response. Transform the response to a vector with entries in $\{+1, -1\}$, corresponding to whether `Direction` is 'Up' or 'Down', respectively. Split the data into train and test sets (80/20 split) and standardize the features.

(e) Write a function *computegrad* that computes and returns $\nabla F(\beta)$ for any β.

(f) Write a function *backtracking* that implements the backtracking rule.

(g) Write a function *graddescent* that implements the gradient descent algorithm with the backtracking rule to tune the step-size. The function *graddescent* calls *computegrad* and *backtracking* as subroutines. The function takes as input the initial point, the initial step-size value, and the target accuracy ε. The stopping criterion is $\|\nabla F\| \leq \varepsilon$.

(h) Write a function *fastgradalgo* that implements the fast gradient algorithm described in Algorithm 1. The function *fastgradalgo* calls *computegrad* and *backtracking* as subroutines. The function takes as input the initial step-size value for the backtracking rule and the target accuracy ε. The stopping criterion is $\|\nabla F\| \leq \varepsilon$.

(i) Initialize the step size to $\eta_0 = 0.1$. Set the target accuracy to $\varepsilon = 1 \times 10^{-5}$. Run *graddescent* and *fastgradalgo* on the training set of the smarket dataset for $\lambda = 0.5$. Plot the curve of the objective values $F(\beta_t)$ for both algorithms versus the iteration counter t (use different colors). What do you observe?

(j) Denote by β_T the final iterate of your fast gradient algorithm. Compare β_T to the $\beta^\star$ found by scikit-learn's `LogisticRegression()`. Compare the objective value for β_T to the one for $\beta^\star$. What do you observe? **Note:** Remember that scikit-learn penalizes the logistic regression objective differently from our formulation above. You will need to find the setting of their C parameter that corresponds to a given choice of λ in our definition.

(k) Run cross-validation on the training set of the smarket dataset using *scikit-learn* to find the optimal value of λ (see `sklearn.linear_model.LogisticRegressionCV`). Again, note that scikit-learn's penalty C is not the same as our λ. Find the value λ^* corresponding to the best choice C^* obtained with this method. Run *graddescent* and *fastgradalgo* to optimize the objective with $\lambda = \lambda^*$. Plot the curve of the objective values $F(\beta_t)$ for both algorithms versus the iteration counter t. Plot the misclassification error on the training set for both algorithms versus the iteration counter t. Plot the misclassification error on the test set for both algorithms versus the iteration counter t. What do you observe?