ORF 525: Statistical Foundations of Data Science

Jianqing Fan — Frederick L. Moore'18 Professor of Finance

Problem Set #3 Spring 2021

Due Monday, March 15, 2021.

Choose any 5 problems

1. Suppose a latent variable Z (e.g. severity of autism) follows the linear model

$$Z = \boldsymbol{\beta}^T \mathbf{X} + \varepsilon,$$

where $\mathbf{X}$ is the covariate and ε has a cdf $F(t)$, i.e. $\mathbb{P}(\varepsilon \le t) = F(t)$ for all $t \in \mathbb{R}$.

Rather than observing Z, we observe $Y = 0$ if $Z \le c_1$, $Y = 1$ if $Z \in (c_1, c_2]$ and $Y = 2$ if $Z > c_2$ for some unknown parameters $c_1 < c_2$.

(a) What is the conditional distribution of Y given $\mathbf{X}$?

(b) Suppose we observe a random sample $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ from the above model. Write down the log-likelihood function.

(c) How would you generalize the logistic regression model we learnt in class to categorial data with more than 2 categories? **Hint**: Use multiple vectors of β's.

2. Consider the generalized linear model $f(y|\mathbf{X} = \mathbf{x}) = \exp\{\frac{y\theta - b(\theta)}{\phi} + c(y, \phi)\}$ with the canonical link. Let $\ell_n(\boldsymbol{\beta})$ denote the negative log-likelihood of the data $\{(\mathbf{X}_i, Y_i)\}_{i=1}^n$.

(a) The formula for estimating the variance of the MLE $\widehat{\boldsymbol{\beta}}$ is $\widehat{\mathrm{var}}(\widehat{\boldsymbol{\beta}}) = [\nabla^2 \ell_n(\widehat{\boldsymbol{\beta}})]^{-1}$. Show that

$$\widehat{\mathrm{var}}(\widehat{\boldsymbol{\beta}}) = \phi[\sum_{i=1}^n b''(\widehat{\theta}_i)\mathbf{X}_i\mathbf{X}_i^T]^{-1}, \qquad \text{where} \qquad \widehat{\theta}_i = \mathbf{X}_i^T\widehat{\boldsymbol{\beta}}.$$

(b) Deduce $\widehat{\mathrm{var}}(\widehat{\boldsymbol{\beta}})$ for logistic regression and Poisson regression.

(c) Write the optimization problem for finding the sparsest solution in a high-confidence set.

(d) For logistic regression, what does the formulation above look like?

3. Let $\ell_n(\boldsymbol{\beta}) = n^{-1}\sum_{i=1}^n [b(\mathbf{X}_i^T\boldsymbol{\beta}) - Y_i\mathbf{X}_i^T\boldsymbol{\beta}]$ be the (normalized) negative log-likelihood of the generalized linear model with $\phi = 1$. Consider the penalized likelihood estimator

$$\widehat{\boldsymbol{\beta}} \in \mathrm{argmin}_{\boldsymbol{\beta}}\{\ell_n(\boldsymbol{\beta}) + \lambda_n\|\boldsymbol{\beta}\|_1\}.$$

(a) Show that $\nabla\ell_n(\boldsymbol{\beta}^*) = n^{-1}\sum_{i=1}^n \varepsilon_i\mathbf{X}_i$, where $\varepsilon_i = b'(\mathbf{X}_i^T\boldsymbol{\beta}^*) - Y_i$.

(b) Let $\boldsymbol{\Delta} = \widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*$ where $\boldsymbol{\beta}^*$ is the true parameter. If $\lambda_n \ge 2\|\nabla\ell_n(\boldsymbol{\beta}^*)\|_\infty$, then for any set $\mathcal{S} \subseteq \{1, 2, \cdots, p\}$ we have

$$\|\boldsymbol{\Delta}_{\mathcal{S}^c}\|_1 \le 3\|\boldsymbol{\Delta}_{\mathcal{S}}\|_1 + 4\|\boldsymbol{\beta}^*_{\mathcal{S}^c}\|_1.$$

Hint: Apply Proposition 5.3.

(c) Let $\mathcal{S}_0 = \text{supp}(\boldsymbol{\beta}^*)$ and $s_n = |\mathcal{S}_0|$. Suppose that $\|\boldsymbol{\beta}^*_{\mathcal{S}_0^c}\|_1 \lesssim \lambda_n$ (here $p_n \lesssim q_n$ means there exists some constant $C > 0$ such that $p_n \leq C q_n$ holds for sufficiently large n), and the restricted strong convexity holds with $\tau_L = 0$ and κ_L being a positive constant. Given $\lambda_n \geq 2\|\nabla \ell_n(\boldsymbol{\beta}^*)\|_\infty$, show that

$$\|\boldsymbol{\Delta}\|_2^2 \lesssim s_n \lambda_n^2 \qquad \text{and} \qquad \|\boldsymbol{\Delta}\|_1 \lesssim s_n \lambda_n.$$

Hint: Apply Theorem 5.8.

4. Suppose that $f(\mathbf{x})$ is smooth and strongly convex in the sense that there exist some $L > \sigma > 0$ such that

$$\mathbf{0} \leq f''(\mathbf{x}) \leq L\mathbf{I}_p \qquad \text{and} \qquad f(\mathbf{x}) \geq f(\mathbf{x}_0) + f'(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0) + \frac{\sigma}{2}\|\mathbf{x} - \mathbf{x}_0\|^2$$

for any $\mathbf{x}_0$ and $\mathbf{x}$. Let

$$f_Q(\mathbf{x}|\mathbf{x}_0) = f(\mathbf{x}_0) + f'(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0) + \frac{1}{2\delta}\|\mathbf{x} - \mathbf{x}_0\|^2$$

be a quadratic majorization at point $\mathbf{x}_0$ with $\delta \leq 1/L$. Let $F(\mathbf{x}) = f(\mathbf{x}) + \lambda\|\mathbf{x}\|_1$ be the objective function to be minimized and $G(\mathbf{x}|\mathbf{x}_0) = f_Q(\mathbf{x}|\mathbf{x}_0) + \lambda\|\mathbf{x}\|_1$ be its penalized quadratic majorization at the point $\mathbf{x}_0$.

(a) Show that $G(\mathbf{x}|\mathbf{x}_0) \leq F(\mathbf{x}) + \frac{1}{2\delta}\|\mathbf{x} - \mathbf{x}_0\|^2$. **Hint**: $f(\mathbf{x}) \geq f(\mathbf{x}_0) + f'(\mathbf{x}_0)^T(\mathbf{x} - \mathbf{x}_0)$.

(b) To find $\mathbf{x}^* = \arg\min F(\mathbf{x})$,[1] consider the iteration

$$\mathbf{x}_i = \text{argmin}\, G(\mathbf{x}|\mathbf{x}_{i-1})$$

whose solution is given by component-wise thresholding. Show that

$$F(\mathbf{x}_i) \leq F(\mathbf{x}_{i-1}) + \min_{0 \leq w \leq 1}\{-w(F(\mathbf{x}_{i-1}) - F(\mathbf{x}^*)) + \frac{w^2}{2\delta}\|\mathbf{x}^* - \mathbf{x}_{i-1}\|^2\}.$$

Hint: Use $F(\mathbf{x}_i) \leq \min_{\mathbf{x}} G(\mathbf{x}|\mathbf{x}_{i-1})$, part (a) and consider minimization only on the line $w\mathbf{x}^* + (1 - w)\mathbf{x}_{i-1}$ for $0 \leq w \leq 1$.

(c) Use the optimality condition of $\mathbf{x}^*$ to show first $F(\mathbf{x}_{i-1}) - F(\mathbf{x}^*) \geq \frac{\sigma}{2}\|\mathbf{x}_{i-1} - \mathbf{x}^*\|^2$.

(d) Use (b) and (c) to show the linear rate convergence: $F(\mathbf{x}_i) - F(\mathbf{x}^*) \leq (1 - \frac{\delta\sigma}{4})[F(\mathbf{x}_{i-1}) - F(\mathbf{x}^*)]$.

5. Let us consider the 128 macroeconomic time series from Jan. 1959 to Dec. 2018, which can be downloaded from the course website (see the "transformed macroeconomic data" at the bottom of the page `https://fan.princeton.edu/fan/classes/525.html`). As before, we extract the data from Jan. 1960 to Oct. 2018 (in total 706 months) and remove the feature named "sasdate" and the features with missing entries. Different from homework #2, suppose that we only observe the binary response $Y = I(\text{UNRATE} \geq 0)$ rather than "UNRATE" (up or down rather than the actual unemployment rate changes).

[1] Please verify by yourself that the minimizer exists and is unique. You don't need to prove this.

(a) In this sub-problem, we are going to study which macroeconomic variables are leading indicators for driving future unemployment rates up and down. To do so, we will pair each row of predictors with the indicator of the next row of response. The last row of predictors and the first element in the response are hence discarded. Do the following steps for Lasso (using R package `glmnet`) and SCAD (using R package `ncvreg`): Set a random seed by `set.seed(525)`; Plot the regularization paths as well as the prediction error estimated by 10-fold cross-validation; Choose a model based on cross-validation, report the model, and point out two most important macroeconomic variables (largest coefficients in the standard variables) that are correlated with the current change of unemployment rate.

(b) Consider the setting in (a) and set a random seed by `set.seed(525)`. Let us take the variables selected by lasso with absolute coefficient > 0.01 (under standardized variables), called it model 1. Run the logistic regression and summarize the fit. Now pick the significant variables whose absolute z-statistics is larger than 1.96. Run the model again with the subset of variable, called this model 2. Are models 1 and 2 statistically significant by running a likelihood ratio test? Are two models practically different by plotting the residuals of model 1 against model 2. Also, plot the residuals of model 2 to see if there are any patterns. (You can use the function `glm` and `anova`. It is a good practice to do anova part of calculation by hand for understanding)

(c) Consider the setting of (a). Leave the last 120 months as testing data and use the rest as training data. Set a random seed by `set.seed(525)`. Run Lasso and SCAD on the training data using `glmnet` and `ncvreg`, respectively, and choose a model based on 10-fold cross-validation. Compute the out-of-sample proportion of prediction error.

6. Upright Human Detection in Photos

Go to the instructor's class website, download the image data `pictures.zip` and its associated preliminary codes `human.r`. In this problem, we are going to create a human detector that tells us whether there is a upright human in a given photo. We treat this as a classification problem with two classes: having humans or not in a photo. You are provided with two datasets `POS` and `NEG` that have photos with and without upright humans respectively.

(a) Load Pictures and Extract Features (The code has already written for you)

The tutorial below that explains the data loading and feature extraction in `human.r`. If you do not want to read, you can just execute the code to get the extracted features. Remember to install the package `png` by using `install.packages("png")` and to change the working directory to yours.

i. Install the package `png` by using `install.packages("png")`, and use the function `readPNG` to load photos. The function `readPNG()` will return the grayscale matrix of the picture.

ii. Use the function `grad` to obtain the gradient field of the central 128×64 part of the grayscale matrix.

iii. Use the function `hog` (Histograms of Oriented Gradient) to extract a feature vector from the gradient field obtained in the previous step. Your feature vector should have 96 components. Please see the appendix for parameter configuration of this function.

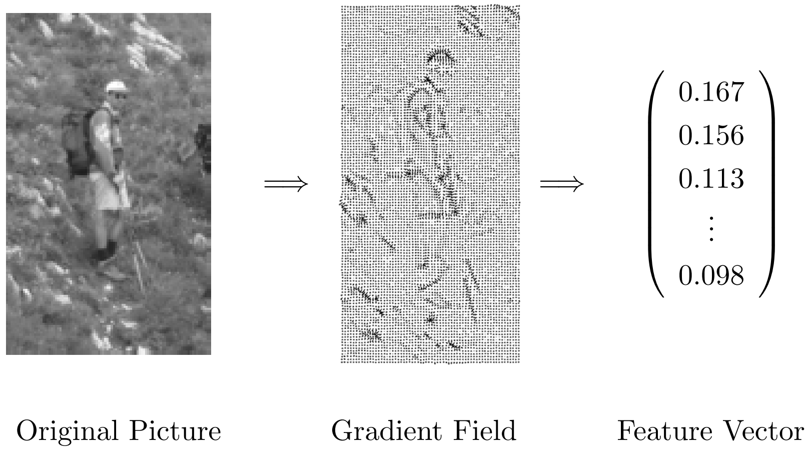

Figure 1: Illustration of feature extraction for a positive example from `POS`.

(b) Logistic Regression (You are responsible for writing the code).

In this question, we will apply logistic regression to the data and use the cross validation to test the classification accuracy of the fitted model. Please finish the following steps:

i. Set the random seed to be 525, i.e., type in `set.seed(525)` in your code. Randomly divide your whole data into five parts. Let the first four parts be the training data and the rest be your testing data.

ii. Feed the training data to the function `glm` and store the fitted model as `fitted1` in R.

iii. Use the function `predict` to apply your fitted model `fitted1` to do classification on the testing data and report the misclassification rate (prediction error).

iv. Let us now select the features using the stepwise selection, by issuing the R-function `step(fitted1)`. Let us call the selected model (the last model in the output) `fitted2`. Report the model `fitted2` and the misclassification rate of `fitted2`.

v. Report the misclassification rate by adding a Lasso penalty with λ chosen by 10 fold cross-validation. (You can use the function `glmnet`.)

Appendix

A. Histogram of Oriented Gradient

Here we give a brief introduction of what `hog(xgrad, ygrad, hn, wn, an)` does. First of all, it uniformly partitions the whole picture into `hn*wn` small parts with `hn` partitions on the height and `wn` partitions on the width. For each small part, it counts the gradient direction whose angle falls in the intervals $[0, 2\pi/\texttt{an}), [2\pi/\texttt{an}, 4\pi/\texttt{an}), ..., [2(\texttt{an}-1)\pi/\texttt{an}, 2\pi)$ respectively.

So `hog` can get `an` frequencies for each small picture. Applying the same procedure to all the small parts, `hog` will have `hn*wn*an` frequencies that constitute the final feature vector for the given gradient field.

B. Useful Functions

(a) `crop.r(X, h, w)` randomly crops a sub-picture that has height h and width w from `X`. The output is therefore a sub-matrix of `X` with `h` rows and `w` columns.

(b) `crop.c(X, h, w)` crops a sub-picture that has height h and width w at the center of `X`. The output is therefore a sub-matrix of `X` with `h` rows and `w` columns. This function helps the `hog(...)` function. For, the cropping in your assignment, use `crop.r()`.

(c) `grad(X, h, w, pic)` yields the gradient field at the center part of the given grayscale matrix `X`. The center region it examines has height `h` and width `w`. It returns a list of two matrices `xgrad` and `ygrad`. The parameter `pic` is a boolean variable. If it is `TRUE`, the generated gradient filed will be plotted. Otherwise the plot will be omitted.

(d) `hog(xgrad, ygrad, hn, wn, an)` returns a feature vector in the length of `hn*wn*an` from the given gradient field. `(xgrad[i,j], ygrad[i,j])` gives the grayscale gradient at the position `(i,j)`. `hn` and `wn` are the partition number on height and width respectively. `an` is the partition number on the angles (or the interval $[0, 2\pi)$ equivalently).