Optimización Numérica: Laboratorio 2

rodrigo.mendoza@itam.mx

10 Septiembre 2019

1 Repaso de Análisis Aplicado

Supongamos que tenemos un conjunto de datos $\mathcal{D} = \{(\mathbf{x}_i, y_i) \in \mathbb{R}^p \times \{0,1\} : i \in [m]\}$. El método de *regresión logística* asume que $\Pr[y_i \mid \mathbf{x}_i, \boldsymbol{\beta}] \sim \text{Bernoulli}(\mu_i)$ con $\mu_i = \sigma(\boldsymbol{\beta}^T \mathbf{x}_i)$, donde $\sigma(z) = (1 + \exp(-z))^{-1}$ es la función *sigmoide* y $\boldsymbol{\beta} \in \mathbb{R}^p$. Queremos encontrar el modelo $\hat{\boldsymbol{\beta}} \in \mathbb{R}^p$ que mejor se ajuste a los datos. Si encontramos $\hat{\boldsymbol{\beta}}$ podemos usarlo para modelar $\Pr[y \mid \mathbf{x}, \hat{\boldsymbol{\beta}}]$ y predecir la etiqueta $\hat{y} \in \{0,1\}$ de un nuevo dato $\mathbf{x}$ por medio del método de máxima verosimilitud:

$$\hat{y} = \begin{cases} 1 & \text{si } \sigma(\hat{\boldsymbol{\beta}}^T \mathbf{x}) \geq 0.5 \\ 0 & \text{si } \sigma(\hat{\boldsymbol{\beta}}^T \mathbf{x}) < 0.5 \end{cases} \tag{1}$$

En este laboratorio vamos a encontrar un estimador $\hat{\boldsymbol{\beta}}$ maximizando la verosimilitud de los datos $\mathcal{D}$.

1. La verosimilitud está data por $\prod_{i=1}^m \text{Bernoulli}(\mu_i)$. Demuestre que la *log-verosimilitud negativa* está data por

$$F(\boldsymbol{\beta}) := \text{LVN}(\boldsymbol{\beta}) = -\sum_{i=1} [y_i \log \mu_i + (1 - y_i) \log(1 - \mu_i)] \tag{2}$$

2. Demuestre que $\sigma(-z) = 1 - \sigma(z)$ y que $\sigma'(z) = \sigma(z)(1 - \sigma(z))$.
3. Use el inciso anterior para demostrar que $\nabla F(\beta) = \mathbf{X}^T(\boldsymbol{\mu} - \mathbf{y})$
4. Use esta expresión para implementar la función `grad_F` en Python.

```
def grad_F(X, y, beta):
    '''(X,y):  Datos de entrenamiento [X.shape=(m,p), y.shape=(m,)]
               X,y matrices de numpy
       beta:   Arreglo de numpy beta.shape=(p,)
    '''
    ...
    return g
```

5. Demuestre que $\nabla^2 F(\beta) = \mathbf{X}^T \mathbf{S} \mathbf{X}$ donde $\mathbf{S}$ es una matriz diagonal ¿Cuál es la expresión para $\mathbf{S}_{ii}$?.
6. Comente sobre la convexidad/concavidad de $F(\boldsymbol{\beta})$.
7. Use esta expresión para implementar la función `hess_F` en Python.

```
def hess_F(X, y, beta):
    '''(X,y):  Datos de entrenamiento [X.shape=(m,p), y.shape=(m,)]
               X,y matrices de numpy
       beta:   Arreglo de numpy beta.shape=(p,)
    '''
    ...
    return H
```

8. Los métodos de búsqueda lineal tienen la siguiente forma:
   a) Elegir $\varepsilon > 0$ y $\boldsymbol{\beta}^0 \in \mathbb{R}^p$.
   b) Mientras $\|\nabla F(\boldsymbol{\beta}^k)\| > \varepsilon$:
      i. Encontrar una dirección de descenso $s^k \in \mathbb{R}^p$ tal que $\nabla F(\boldsymbol{\beta}^k)^T s^k < 0$.

ii. Computar α^k sobre s^k tal que $F(\boldsymbol{\beta}^k + \alpha^k s^k) < F(\boldsymbol{\beta}^k)$.

iii. $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\beta}^k + \alpha^k s^k$

iv. $k \leftarrow k + 1$.

Implemente el método de búsqueda lineal en Python para encontrar $\hat{\boldsymbol{\beta}}$ usando la dirección de descenso $s^k = -\nabla F(\boldsymbol{\beta}^k)$ (En este caso, al método se le conoce como *método del descenso del gradiente*). Encuentre α^k usando la *condición de Armijo*. Esto es, implementando la siguiente subrutina:

a) Elegir $\alpha_0 > 0$, $\tau \in (0, 1)$, $\gamma \in (0, 1)$.

b) Mientras $F(\boldsymbol{\beta}^k + \alpha_i s^k) > F(\boldsymbol{\beta}^k) + \gamma\alpha_i \nabla F(\boldsymbol{\beta}^k)^T s^k$:

i. $\alpha_{i+1} \leftarrow \tau\alpha_i$

ii. $i \leftarrow i + 1$

c) $\alpha^k \leftarrow \alpha_i$

9. Implemente el clasificador de regresión logística (1) para obtener $\hat{y}$.

```
def pred(x, betahat):
    '''x:       Punto a clasificar (vector de numpy)
       betahat: Arreglo de numpy beta.shape=(p,)
       yhat:    Vector binario de predicciones
    '''
    ...
    return yhat
```

10. Implemente la siguiente función de datos (i.e. copie y pegue):

```
import pandas as pd
import numpy
def datos(modo='entrena'):
    gid = 'd932a3cf4d6bdeef36a7230fff959301'
    tail = '64b604aedff376b7757b533d1c93685ce19b2077/bcdata'
    url = 'https://gist.githubusercontent.com/rodrgo/%s/raw/%s' % (gid, tail)
    df = pd.read_csv(url, sep=',')
    df = df.drop(columns=['Unnamed: 32', 'id'])
    var = 'diagnosis'
    df[df[var] == 'M'] = 1
    df[df[var] == 'B'] = 0
    X_cols = [c for c in df.columns if not c is var]
    X, y = df[X_cols].to_numpy(), df[var].to_numpy()
    idx = numpy.random.permutation(X.shape[0])
    train_idx, test_idx = idx[:69], idx[69:]
    idx = train_idx if modo == 'entrena' else test_idx
    return X[idx,:], y[idx]
```

Cree dos conjuntos de datos:

```
X_entrena, y_entrena = datos()
X_prueba, y_prueba = datos(modo='prueba')
```

Use `(X_entrena, y_entrena)` para ajusar un valor $\hat{\boldsymbol{\beta}}$ (`betahat`) usando búsqueda lineal. Use la función `pred` para predecir la etiqueta de cada punto en el conjunto de datos `X_pred`. Compare esta respuesta con `y_prueba` para obtener el error de predicción de su modelo.

11. El *método de Newton* es un método de búsqueda lineal que usa la dirección de descenso $s^k = -(\mathbf{H}^k)^{-1}\mathbf{g}^k$. Implemente en Python el método de Newton para minimizar $F(\boldsymbol{\beta})$.

a) Inicializar $\boldsymbol{\beta}^0$

b) Mientras "no converge"

i. $\mathbf{g}^k = \nabla F(\boldsymbol{\beta}^k)$

ii. $\mathbf{H}^k = \nabla^2 F(\boldsymbol{\beta}^k)$

iii. Resolver $\mathbf{H}^k s^k = -\mathbf{g}^k$

iv. Utilizar Armijo para encontrar una tasa de aprendizaje α^k.

v. $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\beta}^k + \alpha^k \mathbf{d}^k$

12. El método BFGS (Broyden, Fletcher, Goldfarb, Shanno) computa una dirección de descenso $s^k = -(\mathbf{B}^k)^{-1}\nabla F(\boldsymbol{\beta}^k)$ usando una aproximación $\mathbf{B}^k$ de la Hessiana. Implemente este método usando las siguientes ecuaciones para encontrar $\mathbf{B}^k$:

$$\begin{aligned}\mathbf{B}^{k+1} &= \mathbf{B}^k + \frac{\mathbf{w}^k(\mathbf{w}^k)^T}{(\mathbf{w}^k)^T\mathbf{z}^k} - \frac{(\mathbf{B}^k\mathbf{z}^k)(\mathbf{B}^k\mathbf{z}^k)^T}{(\mathbf{z}^k)^T\mathbf{B}^k\mathbf{z}^k}\\ \mathbf{z}^k &= \boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k\\ \mathbf{w}^k &= \nabla F(\boldsymbol{\beta}^{k+1}) - \nabla F(\boldsymbol{\beta}^k)\end{aligned}$$

13. Use los tres métodos para estimar $\hat{\boldsymbol{\beta}}$. Compare el tiempo que requieren para converger. Comente sus resultados.