Data Mining and Machine Learning
Fall 2018, Homework 3 [VERSION 2.0]
(due on Sep 23, 11.59pm EST)

Jean Honorio jhonorio@purdue.edu

The homework is based on a total of 10 points. Your code **should be in Python 2.7**. For clarity, the algorithms presented here will assume zero-based indices for arrays, vectors, matrices, etc. Please read the submission instructions at the end. **Failure to comply to the submission instructions will cause your grade to be reduced.**

In this homework, we will focus on rating (ordinal regression). You can use the script **createsepratingdata.py** to create some synthetic separable data:


```
import numpy as np
import scipy.linalg as la
# Input: number of samples n, number of features d, number of labels k
# Output: matrix X of features, with n rows (samples), d columns (features)
#           X[i,j] is the j-th feature of the i-th sample
#         vector y of labels, with n rows (samples), 1 column
#             y[i] is the label (1 or 2 ... or k) of the i-th sample
# Example on how to call the script:
#     import createsepratingdata
#     X, y = createsepratingdata.run(10,2,3)
def run(n,d,k):
  if n < k:
    raise ValueError('n should be at least k')
  X = np.random.random((n,d))
  y = np.zeros((n,1))
  i = 0
  for r in range(1,k+1):
    j = r*n/k
    X[i:j,0] = X[i:j,0] + 1.5*r
    y[i:j] = r
    i = j
  U = la.orth(np.random.random((d,d)))
  X = np.dot(X,U)
  return (X,y)
```

Here are the questions:

1) [4 points] Implement the PRank algorithm for rating (ordinal regression), introduced in Lecture 6. Recall that:

$$s_{tl} = \begin{cases} -1 & \text{if } y_t \leq l+1 \\ +1 & \text{if } y_t > l+1 \end{cases}$$

The algorithm to be implemented is as follows.

Input: number of iterations L, number of labels k, training data $x_t \in \mathbb{R}^d$, $y_t \in \{1, \ldots, k\}$ for $t = 0, \ldots, n-1$
Output: $\theta \in \mathbb{R}^d$, $b \in \mathbb{R}^{k-1}$
$\theta \leftarrow 0$
for $l = 0, \ldots, k-2$ **do**
  $b_l \leftarrow l$
end for
for iter $= 1, \ldots, L$ **do**
  for $t = 0, \ldots, n-1$ **do**
    $E \leftarrow \{l \mid s_{tl}(\theta \cdot x_t - b_l) \leq 0\}$
    if E is not an empty set **then**
      $\theta \leftarrow \theta + \left(\sum_{l \in E} s_{tl}\right) x_t$
      for $l \in E$ **do**
        $b_l \leftarrow b_l - s_{tl}$
      end for
    end if
  end for
end for

The header of your **Python script ratingprank.py** should be:

```
# Input: number of iterations L
#        number of labels k
#        matrix X of features, with n rows (samples), d columns (features)
#            X[i,j] is the j-th feature of the i-th sample
#        vector y of labels, with n rows (samples), 1 column
#            y[i] is the label (1 or 2 ... or k) of the i-th sample
# Output: vector theta of d rows, 1 column
#         vector b of k-1 rows, 1 column
def run(L,k,X,y):
  # Your code goes here
  return (theta, b)
```

2) [2 points] Implement the following rating (ordinal regression) predictor function, introduced in Lecture 6. Here we present k conditionals (if) for clarity. Your implementation should work for any number k.

Input: number of labels k, $\theta \in \mathbb{R}^d$, $b \in \mathbb{R}^{k-1}$, testing point $x \in \mathbb{R}^d$
Output: label $\in \{1, \ldots, k\}$
if $\theta \cdot x \leq b_0$ **then** label $\leftarrow 1$
if $b_0 < \theta \cdot x \leq b_1$ **then** label $\leftarrow 2$
if $b_1 < \theta \cdot x \leq b_2$ **then** label $\leftarrow 3$
$\vdots$
if $b_{k-3} < \theta \cdot x \leq b_{k-2}$ **then** label $\leftarrow k-1$
if $b_{k-2} < \theta \cdot x$ **then** label $\leftarrow k$

The header of your **Python script ratingpred.py** should be:

```
# Input: number of labels k
#        vector theta of d rows, 1 column
#        vector b of k-1 rows, 1 column
#        vector x of d rows, 1 column
# Output: label (1 or 2 ... or k)
def run(k,theta,b,x):
  # Your code goes here
  return label
```


3) [4 points] Now we ask you to implement the following support vector machines for rating (ordinal regression), introduced in Lecture 6. Recall that:

$$s_{il} = \begin{cases} -1 & \text{if } y_i \leq l+1 \\ +1 & \text{if } y_i > l+1 \end{cases}$$

The problem to be implemented is as follows.

$$\begin{aligned} \text{minimize } & \frac{1}{2}\theta \cdot \theta \\ \text{subject to } & s_{il}(x_i \cdot \theta - b_l) \geq 1 \text{ for } i = 0, \ldots, n-1, l = 0, \ldots, k-2 \\ & b_l \leq b_{l+1} \text{ for } l = 0, \ldots, k-3 \end{aligned}$$

First some general notation for clarity: for any integers p and q, let $I_{p\times p} \in \mathbb{R}^{p\times p}$ be the identity matrix with p rows and p columns. Let $0_{p\times q} \in \mathbb{R}^{p\times q}$ be a matrix of zeros, with p rows and q columns. Let $1_{p\times q} \in \mathbb{R}^{p\times q}$ be a matrix of ones, with p rows and q columns.

Now, recall that we have n training samples and that $x_i \in \mathbb{R}^d$ for $i = 1, \ldots, n$. Let $H = \begin{bmatrix} I_{d\times d} & 0_{d\times(k-1)} \\ 0_{(k-1)\times d} & 0_{(k-1)\times(k-1)} \end{bmatrix} \in \mathbb{R}^{(d+k-1)\times(d+k-1)}$. Let $f = 0_{(d+k-1)\times 1}$. Let $A \in \mathbb{R}^{(n(k-1)+k-2)\times(d+k-1)}$ defined as follows. (Recall that $x_{i,j}$ is the j-th

feature of the i-th sample.)

$$A = \begin{bmatrix}
-s_{0,0}x_{0,0} & -s_{0,0}x_{0,1} & \dots & -s_{0,0}x_{0,d-1} & s_{0,0} & 0 & 0 & \dots & 0 & 0 \\
-s_{0,1}x_{0,0} & -s_{0,1}x_{0,1} & \dots & -s_{0,1}x_{0,d-1} & 0 & s_{0,1} & 0 & \dots & 0 & 0 \\
\vdots & & & & & & & & & \\
-s_{0,k-2}x_{0,0} & -s_{0,k-2}x_{0,1} & \dots & -s_{0,k-2}x_{0,d-1} & 0 & 0 & 0 & \dots & 0 & s_{0,k-2} \\
\vdots & & & & & & & & & \\
\vdots & & & & & & & & & \\
\vdots & & & & & & & & & \\
-s_{n-1,0}x_{n-1,0} & -s_{n-1,0}x_{n-1,1} & \dots & -s_{n-1,0}x_{n-1,d-1} & s_{n-1,0} & 0 & 0 & \dots & 0 & 0 \\
-s_{n-1,1}x_{n-1,0} & -s_{n-1,1}x_{n-1,1} & \dots & -s_{n-1,1}x_{n-1,d-1} & 0 & s_{n-1,1} & 0 & \dots & 0 & 0 \\
\vdots & & & & & & & & & \\
-s_{n-1,k-2}x_{n-1,0} & -s_{n-1,k-2}x_{n-1,1} & \dots & -s_{n-1,k-2}x_{n-1,d-1} & 0 & 0 & 0 & \dots & 0 & s_{n-1,k-2} \\
0 & 0 & \dots & 0 & 1 & -1 & 0 & \dots & 0 & 0 \\
0 & 0 & \dots & 0 & 0 & 1 & -1 & \dots & 0 & 0 \\
\vdots & & & & & & & & & \\
0 & 0 & \dots & 0 & 0 & 0 & 0 & \dots & 1 & -1
\end{bmatrix}$$

Let $c = \begin{bmatrix} -1_{(n(k-1))\times 1} \\ 0_{(k-2)\times 1} \end{bmatrix} \in \mathbb{R}^{n(k-1)+k-2}$. Since $\theta \in \mathbb{R}^d$ and $b \in \mathbb{R}^{k-1}$, lets define $z = \begin{bmatrix} \theta \\ b \end{bmatrix} \in \mathbb{R}^{d+k-1}$ and we can rewrite the rating SVM problem as:

$$\begin{aligned} &\text{minimize } \frac{1}{2} z^T H z + f^T z \\ &\text{subject to } Az \le c \end{aligned}$$

Fortunately, the package **cvxopt** can solve exactly the above problem by doing:

```
import cvxopt as co
z = np.array(co.solvers.qp(co.matrix(H,tc='d'),co.matrix(f,tc='d'),
                           co.matrix(A,tc='d'),co.matrix(c,tc='d'))['x'])
```

The header of your **Python script ratingsvm.py** should be:

```
# Input: number of labels k
#        matrix X of features, with n rows (samples), d columns (features)
#            X[i,j] is the j-th feature of the i-th sample
#        vector y of labels, with n rows (samples), 1 column
#            y[i] is the label (1 or 2 ... or k) of the i-th sample
# Output: vector theta of d rows, 1 column
#         vector b of k-1 rows, 1 column
def run(k,X,y):
  # Your code goes here
  return (theta, b)
```

Notice that for prediction you can reuse the **ratingpred.py** function that you wrote for question 2.

SOME POSSIBLY USEFUL THINGS.
Python 2.7 is available at the servers antor and data. From the terminal, you can use your Career account to start a ssh session:

```
ssh username@data.cs.purdue.edu
OR
ssh username@antor.cs.purdue.edu
```

From the terminal, to start Python:

```
python
```

Inside Python, to check whether you have **Python 2.7**:

```
import sys
print (sys.version)
```


Inside Python, to check whether you have the package **cvxopt**:

```
import cvxopt
```


From the terminal, to install the Python package **cvxopt**:

```
pip install --user cvxopt
OR
python -m pip install --user cvxopt
```


More information at `https://cvxopt.org/install/index.html`

SUBMISSION INSTRUCTIONS.
Your code **should be in Python 2.7**. We **only need** the Python scripts (.py files). We **do not need** the Python (compiled) bytecodes (.pyc files). You will get 0 points if your code does not run. You will get 0 points in you fail to include the Python scripts (.py files) even if you mistakingly include the bytecodes (.pyc files). We will deduct points, if you do not use the right name for the Python scripts (.py) as described on each question, or if the input/output matrices/vectors/scalars have a different type/size from what is described on each question. Homeworks are to be solved individually. We will run plagiarism detection software.
Please, submit a single ZIP file **through Blackboard**. Your Python scripts (**ratingprank.py**, **ratingpred.py**, **ratingsvm.py**) should be directly inside the ZIP file. **There should not be any folder inside the ZIP file**, just Python scripts. The ZIP file should be named according to your Career account. For instance, if my Career account is jhonorio, the ZIP file should be named **jhonorio.zip**