Data Mining and Machine Learning
Fall 2018, Homework 1
(due on Sep 4, 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 classification for separable data. You can use the following script **createsepdata.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
# Output: numpy matrix X of features, with n rows (samples), d columns (features)
#             X[i,j] is the j-th feature of the i-th sample
#         numpy vector y of labels, with n rows (samples), 1 column
#             y[i] is the label (+1 or -1) of the i-th sample
# Example on how to call the script:
#     import createsepdata
#     X, y = createsepdata.run(10,3)
def run(n,d):
  y = np.ones((n,1))
  y[n/2:] = -1
  X = np.random.random((n,d))
  idx_row, idx_col = np.where(y==1)
  X[idx_row,0] = 0.1+X[idx_row,0]
  idx_row, idx_col = np.where(y==-1)
  X[idx_row,0] = -0.1-X[idx_row,0]
  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 following perceptron algorithm, introduced in Lecture 1.

```
Input: number of iterations $L$, training data $x_t \in \mathbb{R}^d$, $y_t \in \{+1, -1\}$ for
$t = 0, \dots, n-1$
Output: $\theta \in \mathbb{R}^d$
$\theta \leftarrow 0$
for iter $= 1, \dots, L$ do
  for $t = 0, \dots, n-1$ do
    if $y_t(\theta \cdot x_t) \leq 0$ then
      $\theta \leftarrow \theta + y_t x_t$
    end if
  end for
end for
```

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

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

2) [2 points] Implement the following linear predictor function, introduced in Lecture 1.

```
Input: $\theta \in \mathbb{R}^d$, testing point $x \in \mathbb{R}^d$
Output: label $\in \{+1, -1\}$
if $\theta \cdot x > 0$ then
  label $\leftarrow +1$
else
  label $\leftarrow -1$
end if
```

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

```
# Input: numpy vector theta of d rows, 1 column
#        numpy vector x of d rows, 1 column
# Output: label (+1 or -1)
def run(theta,x):
  # Your code goes here
  return label
```

3) [4 points] Now we ask you to implement the following *primal* support vector machines (PSVM) problem, introduced in Lecture 2.

$$\begin{aligned} &\text{minimize } \frac{1}{2}\theta \cdot \theta \\ &\text{subject to } y_i(x_i \cdot \theta) \geq 1 \text{ for } i = 0, \dots, n-1 \end{aligned}$$

Let $H \in \mathbb{R}^{d \times d}$ be the identity matrix with d rows and d columns. Let $f = (0, 0, \dots, 0)^{\mathrm{T}} \in \mathbb{R}^d$ be a d-dimensional vector of zeros. Let $A \in \mathbb{R}^{n \times d}$ be a matrix of n rows and d columns, where $a_{i,j} = -y_i x_{i,j}$ for all $i = 0, \dots, n-1$ and $j = 0, \dots, d-1$. (Recall that y_i is the label of the i-th sample and $x_{i,j}$ is the j-th feature of the i-th sample.) Let $b = (-1, -1, \dots, -1)^{\mathrm{T}} \in \mathbb{R}^n$ be an n-dimensional vector of minus ones. Since $\theta \in \mathbb{R}^d$, we can rewrite the PSVM problem as:

$$\begin{aligned} &\text{minimize } \frac{1}{2}\theta^{\mathrm{T}} H \theta + f^{\mathrm{T}}\theta \\ &\text{subject to } A\theta \leq b \end{aligned}$$

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

```
import cvxopt as co
theta = np.array(co.solvers.qp(co.matrix(H,tc=’d’),co.matrix(f,tc=’d’),
                               co.matrix(A,tc=’d’),co.matrix(b,tc=’d’))[’x’])
```

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

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

Notice that for prediction you can reuse the **linpred.py** script 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
```

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 (**linperceptron.py**, **linpred.py**, etc.) 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**