Data Mining and Machine Learning
Fall 2018, Homework 5
(due on Oct 7, 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.**

Assignment Project Exam Help

For testing your solution for questions 1 and 2, you can use the following script **createsepdata.py** to create some synthetic separable data:

https://powcoder.com

```
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)
```

Add WeChat powcoder

For testing your solution for questions 3 and 4, you can use the following script **createlinregdata.py** to create some synthetic linear regression data:

```
import numpy as np
import numpy.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 scalar values, with n rows (samples), 1 column
#             y[i] is the scalar value of the i-th sample
# Example on how to call the function:
#     import createlinregdata
#     X, y = createlinregdata.run(10,2)
def run(n,d):
  w = 2*np.random.random((d,1))-1
  w = w/la.norm(w)
  X = np.random.normal(0.0,1.0,(n,d))
  y = np.dot(X,w) + 0.25*np.random.normal(0.0,1.0,(n,1))
  return (X, y)
```

Assignment Project Exam Help

https://powcoder.com

Additionally, for questions 3 and 4, you will require a way to solve the linear regression problem, with training data $x_t \in \mathbb{R}^d$, $y_t \in \mathbb{R}$ for $t = 0, \ldots, n-1$.

Add WeChat powcoder

$$\hat{\theta} \leftarrow \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{2} \sum_{t=0}^{n-1} (y_t - \beta \cdot x_t)^2$$

If you assume that $n > d$, a solution to the above problem is given by the following script **linreg.py**:

```
import numpy as np
import numpy.linalg as la
# 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 scalar values, with n rows (samples), 1 column
#             y[i] is the scalar value of the i-th sample
# Output: numpy vector theta, with d rows, 1 column
# Example on how to call the function:
#     import linreg
#     theta = linreg.run(X,y)
def run(X,y):
  return np.dot(la.pinv(X),y)
```

Here are the questions:

1) [3 points] Implement a simplified version of the boosting algorithm (Lecture 8). We will use the exponential loss, and a simple type of weak classifiers, which take the sign of one feature. Recall that y_t is the label of the t-th sample and $x_{t,j}$ is the j-th feature of the t-th sample. Note that $\text{sgn}(z) = 1$ if $z > 0$, and $\text{sgn}(z) = -1$ if $z \leq 0$. Here is the algorithm:

```
**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:** $\alpha \in \mathbb{R}^L$, $\theta \in \{0, \dots, d-1\}^L$
for $t = 0, \dots, n-1$ do
  $W_t \leftarrow 1/n$
end for
for $r = 0, \dots, L-1$ do
  $\theta_r \leftarrow \arg\min_{j \in \{0,\dots,d-1\}} -\sum_{t=0}^{n-1} W_t \, y_t \, \text{sgn}(x_{t,j})$
  $\epsilon \leftarrow \min_{j \in \{0,\dots,d-1\}} -\sum_{t=0}^{n-1} W_t \, y_t \, \text{sgn}(x_{t,j})$
  $\epsilon \leftarrow \min(0.99, \max(-0.99, \epsilon))$
  $\alpha_r \leftarrow \frac{1}{2} \log\left(\frac{1-\epsilon}{1+\epsilon}\right)$
  for $t = 0, \dots, n-1$ do
    $W_t \leftarrow W_t \exp(-\alpha_r \, y_t \, \text{sgn}(x_{t,\theta_r}))$
  end for
  $Z = \sum_{t=0}^{n-1} W_t$
  for $t = 0, \dots, n-1$ do
    $W_t \leftarrow W_t / Z$
  end for
end for
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Since θ_r is a feature index, x_{t,θ_r} denotes taking the feature θ_r from the t-th sample. The header of your **Python script adaboost.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 alpha of weights, with L rows, 1 column
#         numpy vector theta of feature indices, with L rows, 1 column
def run(L,X,y):
  # Your code goes here
  return (alpha, theta)
```

2) [1 point] Implement the Adaboost predictor function. Note that $\text{sgn}(z) = 1$ if $z > 0$, and $\text{sgn}(z) = -1$ if $z \leq 0$.

```
**Input:** $\alpha \in \mathbb{R}^L$, $\theta \in \{0, \ldots, d-1\}^L$, testing point $x \in \mathbb{R}^d$
**Output:** label $\in \{+1, -1\}$
label $\leftarrow \text{sgn}\left( \sum_{r=0}^{L-1} \alpha_r \, \text{sgn}(x_{\theta_r}) \right)$
```

Since θ_r is a feature index, x_{θ_r} denotes taking the feature θ_r from the testing point x. The header of your **Python script adapred.py** should be:

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

Assignment Project Exam Help

3) [3 points] Implement k-fold cross validation (Lecture 9) with linear regression. (The function $\lfloor w \rfloor$ denotes the largest integer less than or equal to $w \in \mathbb{R}$, i.e., the “floor” function.)

https://powcoder.com

Add WeChat powcoder

```
**Input:** number of folds $k$, data $x_t \in \mathbb{R}^d$, $y_t \in \mathbb{R}$ for $t = 0, \ldots, n-1$
**Output:** mean squared error $z \in \mathbb{R}^k$
**for** $i = 0, \ldots, k-1$ **do**
  $T \leftarrow \{\lfloor n\, i/k \rfloor, \ldots, \lfloor n(i+1)/k - 1 \rfloor\}$
  $S \leftarrow \{0, \ldots, n-1\} - T$
  $\widehat{\theta} \leftarrow \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{2} \sum_{t \in S} (y_t - \beta \cdot x_t)^2$
  $z_i \leftarrow \frac{1}{|T|} \sum_{t \in T} (y_t - \widehat{\theta} \cdot x_t)^2$
**end for**
```

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

```
# Input: number of folds k
#        numpy matrix X of features, with n rows (samples), d columns (features)
#        numpy vector y of scalar values, with n rows (samples), 1 column
# Output: numpy vector z of k rows, 1 column
def run(k,X,y):
  # Your code goes here
  return z
```

4) [3 points] Implement bootstrapping (Lecture 9) with linear regression.

```
**Input:** number of bootstraps $B$, data $x_t \in \mathbb{R}^d$, $y_t \in \mathbb{R}$ for $t = 0, \ldots, n-1$
**Output:** mean squared error $z \in \mathbb{R}^B$
for $i = 0, \ldots, B-1$ do
  $u \leftarrow (0, \ldots, 0)$ (an array of $n$ zeros)
  $S \leftarrow$ emptyset
  for $j = 0, \ldots, n-1$ do
    choose $k$ uniformly at random from $\{0, \ldots, n-1\}$
    $u_j \leftarrow k$ (repeated elements are allowed in the array $u$)
    $S \leftarrow S \cup \{k\}$ (repeated elements are not allowed in the set $S$)
  end for
  $T \leftarrow \{0, \ldots, n-1\} - S$ (repeated elements are not allowed in the set $T$)
  $\widehat{\theta} \leftarrow \arg\min_{\beta \in \mathbb{R}^d} \frac{1}{2} \sum_{j=0}^{n-1} (y_{u_j} - \beta \cdot x_{u_j})^2$
  $z_i \leftarrow \frac{1}{|T|} \sum_{t \in T} (y_t - \widehat{\theta} \cdot x_t)^2$
end for
```

Assignment Project Exam Help

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

```
# Input: number of bootstraps B
#        numpy matrix X of features, with n rows (samples), d columns (features)
#        numpy vector y of scalar values, with n rows (samples), 1 column
# Output: numpy vector z of B rows, 1 column
def run(B,X,y):
  # Your code goes here
  return z
```

https://powcoder.com

Add WeChat powcoder

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)
```

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 (**adaboost.py**, **adapred.py**, **kfoldcv.py**, **bootstrapping.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**

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder