Homework Assignment 1

COGS 181: Neural Networks and Deep Learning

Due: January 22, 2017, 11:59pm
Instructions: Please answer the questions below, attach your code, and insert figures to create a pdf file; submit your file to TED (ted.ucsd.edu) by 11:59pm, 01/22/2017. You may search information online but you will need to write code/find solutions to answer the questions yourself. You are free to choose MATLAB, Python, or other preferred programming languages.

Grade: ____ out of 100 points

1 (15 points) Data and Visualization

Download the Iris Data Set from UCI Machine Learning Repository (click ***here***).
http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data).
The dataset description can be found ***here***.
https://archive.ics.uci.edu/ml/datasets/Iris.
Plot two figures to visualize the dataset using: (1) the first 2 feature dimensions, and (2) the first 3 feature dimensions. Some useful instructions are shown below:

MATLAB:

(1) Load $iris.data$.

```
>> [x1,x2,x3,x4,y]=textread('iris.data','%f%f%f%f%s','delimiter',',');
```

(2) Assign each plant with a class label

```
>> Y = grp2idx(categorical(y));
```

(3) Visualize first 2 feature dimensions.

```
>> scatter(x1,x2,15,Y,'filled');
```

(4) Visualize first 3 feature dimensions.

```
>> scatter3(x1,x2,x3,15,Y,'filled');
```

Python:

(1) Import several useful package into Python:

```
import matplotlib.pyplot as plt
from sklearn import datasets
from mpl_toolkits.mplot3d import Axes3D
```

(2) Load Iris dataset into Python:

```
iris = datasets.load_iris()
X = iris.data
Y = iris.target
```

(3) Visualize first 2 feature dimensions:

```
plt.scatter(X[:,0],X[:,1], c=Y, cmap=plt.cm.Paired)
```

(4) Visualize first 3 feature dimensions:

```
fig = plt.figure(1, figsize=(8, 8))
ax = Axes3D(fig, elev=-150, azim=110)
ax.scatter(X[:,0],X[:,1],X[:,2],c=Y,cmap=plt.cm.Paired)
```

2 (10 points) One-hot Encoding

A dataset S is denoted as $S = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$, where each sample refers to the specification of a car.

	Length (inch)	Height (inch)	Color
$\mathbf{x}_1$	182.3	62	Silver
$\mathbf{x}_2$	181	66	Blue
$\mathbf{x}_3$	186	56	Red
$\mathbf{x}_4$	179	59	Blue
$\mathbf{x}_5$	182	50	Black

Please represent this dataset S using a matrix and show it below. **Hint**: for the categorical feature, you may use the one-hot encoding strategy; you can choose either a row vector or a column vector to represent each data sample, but please use one consistently for the entire dataset.

3 (25 points) Linear regression

Download the data file **data.txt** from the course website: https://sites.google.com/site/ucsdcogs181winter2017/classroom-news, and load the data into your environment (MATLAB or Python). The data samples are represented as row vectors; the first column refers to the input (x-axis) and the second column refers to the output (y-axis). We assume that this dataset was generated from a linear model: $y = ax + b$ plus noise, and you need to find the optimal a^* and b^* to fit the data. Now follow the instructions below using the least squre estimation and show the result in your report.

HINT: The closed form solution for the least square estimation is $(X^T X)^{-1} X^T Y$.

MATLAB:

(1) Load the data.

```
>> data = importdata('data.txt');
>> x = data(:,1);
>> y = data(:,2);
```

(2) Plot the data using plot function.

```
>> plot(x,y); grid;
```

(3) Create the matrix

```
>> X = [ones(length(x),1) x.^1]}
```

(4) Compute the least squares line over the given data by:

```
>> sol = inv(X'*X)*X'*y;
```

(5) Overlay the computed least square line over the given data:

```
>> hold on; plot(x, sol(1)+sol(2)*x, '--');
```

(6) Assign a title to this figure:

```
>> title('Least squares line fitting'); xlabel('x'); ylabel('y');
```

(7) Copy and paste the figure in your report.

Python:

(1) Import *matplotlib* and *numpy* package into Python.

```
import matplotlib.pyplot as plt
import numpy as np
```

(2) Load data.

```
data = np.loadtxt('data.txt',dtype='float')
x = data[:,0].reshape(len(data),1)
y = data[:,1].reshape(len(data),1)
```

(3) Plot the data.

```
plt.plot(x,y)
plt.grid()
```

(4) Create the matrix

```
X = np.hstack((np.ones((len(x),1)),np.power(x,1)))
```

(5) Compute the least squares line over the given data by

```
X_t = X.transpose((1,0))
sol = np.dot(np.linalg.inv(np.dot(X_t,X)),np.dot(X_t,y))
```

(6) Plot the least square line, and assign title to the figure

```
plt.plot(x,y)
plt.hold(True)
plt.plot(x,sol[0]+sol[1]*x)
plt.title('Least square line fitting')
plt.xlabel('x')
plt.ylabel('y')
```

4 (25 points) Polynomial Regression

Now use the same data from Q3 but learn a polynomial regressor such as $y = ax^2 + bx + c$ to fit the data using the similar technique (the least squre estimation) as in Q3.

(1) Derive the formulation for the second order least square problem. **Hint:** The matrix X should contain an x^2 term now, for example

```
X = [ones(length(x),1) x.^1 x.^2].
```

(2) Write your code to compute the derived polynomial regressor using the given data (data.txt), and overlay it on the data point in the same figure as in Q3. Please paste your code and figures in the report for this question.

(3) Explain which formulation (line or polynomial) is a better model here with respect to fitting this dataset.

5 (25 points) A Linear Classifier

In a standard binary classification task, $y \in \{-1, +1\}$ is the label for each data sample $\mathbf{x}$. Here we define a linear function:

$$f(\mathbf{x}) = \begin{cases} 1, & ax_1 + bx_2 + c >= 0, \\ -1, & otherwise, \end{cases}$$

where $\mathbf{x} = (x_1, x_2)$ ($\in \mathbb{R}^2$, real numbers) denotes two-dimensional feature, and a, b, and c are the parameters in the linear classifier. When $ax_1 + bx_2 + c >= 0$, the classifier predicts the given $\mathbf{x}$ as a positive sample, which means $f(\mathbf{x}) = 1$; otherwise, the classifier predicts the given $\mathbf{x}$ as a negative sample, which means $f(\mathbf{x}) = -1$.

1. **Data.** Download the data files **train.txt** and **text.txt** from the course website: https://sites.google.com/site/ucsdcogs181winter2017/classroom-news. The data file **train.txt** contains the training set S_{train} with 80 samples, and the data file **test.txt** contains the test set S_{test} with 20 samples. Each data point is represented as a row vector in the data files with the first column being the first feature x_1, the second column being the second feature x_2, and the third column being the class label $y \in \{-1, +1\}$.

2. **Train a linear classifier.** Use the least square estimation to learn a linear classifier. Report the training error of your learned linear classifier on the training set. **HINT:** You can follow the instructions in Q3, but the input is 2-dimensional now. Here is how to create $\mathbf{X}$:

   MATLAB:

   ```
   >> X=[ones(length(x1),1),x1,x2];
   ```

   Python:

   ```
   X=np.hstack((np.ones((len(x1),1)),x1,x2))
   ```

3. **Compute for the test error on the test set.** Report the test error of your learned linear classifier on the test set.

   HINT:

   (1) Training error is defined as $e_{training} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \neq f(\mathbf{x_i}))$,

   (2) Test error is defined as $e_{test} = \frac{1}{q} \sum_{i=1}^{q} \mathbf{1}(y_i \neq f(\mathbf{x_i}))$

   where $\mathbf{1}(z) = \begin{cases} 1, & if\ z = TRUE \\ 0, & otherwise \end{cases}$