CSCE 421 HW1

Sahil Palnitkar

Problem 1

1)

$$\nabla(f) = [\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}]$$

$$\frac{\partial f(x,y)}{\partial x}(x^2 + ln(x) + xy + y^3) = 2x + \frac{1}{x} + y$$

$$\frac{\partial f(x,y)}{\partial y}(x^2 + ln(x) + xy + y^3) = x + 3y^2$$

$$= \begin{bmatrix} \left(2x + \frac{1}{x} + y\right) \\ \left(x + 3y^2\right) \end{bmatrix}$$

Evaluating gradient value at (x,y) = (1, -1), we get

$$= \begin{bmatrix} (2) \\ (4) \end{bmatrix}$$

2)

$$\nabla(f) = [\frac{\partial f(x,y,z)}{\partial x}, \frac{\partial f(x,y,z)}{\partial y}, \frac{\partial f(x,y,z)}{\partial z}]$$

$$\frac{\partial f(x,y,z)}{\partial x}\left(\tanh\left(x^3 y^3\right) + \sin(z)\right) = 3x^2y^3\, sech^2\left(x^3y^3\right)$$

$$\frac{\partial f(x,y,z)}{\partial y}\left(\tanh\left(x^3 y^3\right) + \sin(z)\right) = 3x^3y^2\, sech^2\left(x^3y^3\right)$$

$$\frac{\partial f(x,y,z)}{\partial y}\left(\tanh\left(x^3 y^3\right) + \sin(z)\right) = cos\,(z)$$

Evaluating gradient value at (x,y,z) = (-1, 0, $\frac{\pi}{2}$), we get

$$= \begin{bmatrix} (0) \\ (0) \\ (0) \end{bmatrix}$$

Problem 2

1) $$\begin{pmatrix} 1\cdot 6 + (-1)\cdot 0 + 6\,(-3) + 7\cdot 11 & 1\cdot 2 + (-1)\,(-1) + 6\cdot 0 + 7\cdot 4 \\ 9\cdot 6 + 0\cdot 0 + 8\,(-3) + 1\cdot 11 & 9\cdot 2 + 0\cdot (-1) + 8\cdot 0 + 1\cdot 4 \\ (-8)\cdot 6 + 5\cdot 0 + 2\,(-3) + 3\cdot 11 & (-8)\cdot 2 + 5\,(-1) + 2\cdot 0 + 3\cdot\ 4 \\ 10\cdot\ 6 + 4\cdot\ 0 + 0\cdot(-3) + 1\cdot 11 & 10\cdot 2 + 4\,(-1) + 0\cdot 0 + 1\cdot 4 \end{pmatrix} = \begin{pmatrix} 65 & 31 \\ 41 & 22 \\ -21 & -9 \\ 71 & 20 \end{pmatrix}$$

2) $\begin{pmatrix} 10\cdot 7 & 10\cdot 3 & 10\cdot 0 & 10\cdot 1 \\ 4\cdot 7 & 4\cdot 3 & 4\cdot 0 & 4\cdot 1 \\ (-1)\cdot 7 & (-1)\cdot 3 & (-1)\cdot 0 & (-1)\cdot 1 \\ 8\cdot 7 & 8\cdot 3 & 8\cdot 0 & 8\cdot 1 \end{pmatrix} = \begin{pmatrix} 70 & 30 & 0 & 10 \\ 28 & 12 & 0 & 4 \\ -7 & -3 & 0 & -1 \\ 56 & 24 & 0 & 8 \end{pmatrix}$

3) $(\, 9\,(-3) + (-3)\cdot 4 + 1\cdot(-9) + 6\cdot 0\,) = (-48)$

Problem 3

1) Vector $= \begin{bmatrix} (5-7) \\ (0-9) \\ (-1-5) \\ (4-2) \end{bmatrix} = \begin{bmatrix} (-2) \\ (-9) \\ (-6) \\ (2) \end{bmatrix}$ So we get $\ell_0 = 4$ as there are 4 non-zero terms.

2) Vector $= \begin{bmatrix} (5-7) \\ (0-9) \\ (-1-5) \\ (4-2) \end{bmatrix} = \begin{bmatrix} (-2) \\ (-9) \\ (-6) \\ (2) \end{bmatrix}$ So we get $\ell_1 = |(-2 + -9 + -6 + 2)| = (15)$

3)
$\ell_2 = \sqrt{(5-7)^2 + (0-9)^2 + (-1-5)^2 + (4-2)^2} = \sqrt{(-2)^2 + (-9)^2 + (-6)^2 + (2)^2} = \sqrt{(4}$
$= 11.1803$

4)
Vector $= \begin{bmatrix} (5-7) \\ (0-9) \\ (-1-5) \\ (4-2) \end{bmatrix} = \begin{bmatrix} (-2) \\ (-9) \\ (-6) \\ (2) \end{bmatrix}$ So we get $\ell_\infty = 9$ as this is the absolute value of the max term of the vector.

Problem 4

1) The sample space is $6 \cdot 6 = 36$

2) The sum 10 can be achieved by getting either $(\,4, 6\,)$ or $(\,5, 5\,)$ or $(\,6, 4\,)$. This is 3 ways out of 36. So the probability of getting a sum of 10 when 2 dice are rolled is $\frac{3}{36} = \frac{1}{12}$

3) The sum 6 can be achieved by getting either $(\,1, 5\,)$ or $(\,2, 4\,)$ or $(\,3, 3\,)$ or $(\,4, 2\,)$ or $(\,5, 1\,)$. This is 5 ways out of 36. So the probability of getting a sum 6 when 2 dice are rolled is $\frac{5}{36}$

Problem 5

1) The mean of X will be $\frac{(a+b)}{2}$

2) The standard deviation of X will be $\sqrt{\frac{(b-a)^2}{12}}$

Problem 6

1) The accuracy of the detector will be $\frac{(FP+FN)}{Total} = \frac{(37+55)}{160} = 0.575$

2) The sensitivity of the detector is $\frac{TP}{(TP+FN)} = \frac{37}{(37+45)} = 0.4512$

The specificity of the detector is $\frac{TN}{(FP+TN)} = \frac{55}{(23+55)} = 0.7051$

Balanced accuracy = $\frac{(Sensitivity+Specifity)}{2} = \frac{(0.4512+0.7051)}{2} = 0.5782$

3) The precision of the detector is $\frac{TP}{(TP+FP)} = 0.6167$

4) The recall is the same as the sensitivity which is 0.4512

5) The F-measure of the detector is $\frac{2 \cdot Precision \cdot Recall}{(Precision+Recall)} = 0.5211$

Problem 7

1) ROC value for threshold = 0 will be $(1, 1)$

2) ROC value for threshold = 0.25 will be $(0.4, 0.8)$

3) ROC value for threshold = 0.5 will be $(0.4, 0.6)$

4) ROC value for threshold = 0.75 will be $(0, 0.2)$

5) ROC value for threshold = 1 will be $(0, 0)$

6) AUROC using left endpoint approximation will be

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n-1} x_{i+1} - x_i \cdot f(x_i)$$

In [3]:

```
import pandas as pd
import numpy as np
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
```

In [4]:

```
dataframe = pd.read_csv("Smarket.csv")
```

In [5]:

```
print(dataframe.head())
```

```
   Unnamed: 0  Year   Lag1   Lag2   Lag3   Lag4   Lag5  Volume  Toda
y  \
0           1  2001  0.381 -0.192 -2.624 -1.055  5.010  1.1913  0.95
9
1           2  2001  0.959  0.381 -0.192 -2.624 -1.055  1.2965  1.03
2
2           3  2001  1.032  0.959  0.381 -0.192 -2.624  1.4112 -0.62
3
3           4  2001 -0.623  1.032  0.959  0.381 -0.192  1.2760  0.61
4
4           5  2001  0.614 -0.623  1.032  0.959  0.381  1.2057  0.21
3

  Direction
0        Up
1        Up
2      Down
3        Up
4        Up
```

In [6]:

```
print(dataframe.shape)
```

```
(1250, 10)
```

In [7]:

```
X = dataframe[['Lag1','Lag2']]
y = dataframe['Direction']
print(X)
print(y)
```

```
        Lag1   Lag2
0      0.381 -0.192
1      0.959  0.381
2      1.032  0.959
3     -0.623  1.032
4      0.614 -0.623
5      0.213  0.614
6      1.392  0.213
7     -0.403  1.392
8      0.027 -0.403
9      1.303  0.027
10     0.287  1.303
11    -0.498  0.287
12    -0.189 -0.498
13     0.680 -0.189
14     0.701  0.680
15    -0.562  0.701
16     0.546 -0.562
17    -1.747  0.546
18     0.359 -1.747
19    -0.151  0.359
20    -0.841 -0.151
21    -0.623 -0.841
22    -1.334 -0.623
23     1.183 -1.334
24    -0.865  1.183
25    -0.218 -0.865
26     0.812 -0.218
27    -1.891  0.812
28    -1.736 -1.891
29    -1.851 -1.736
...      ...    ...
1220   0.179 -0.385
1221   0.941  0.179
1222   0.440  0.941
1223   0.527  0.440
1224   0.508  0.527
1225   0.347  0.508
1226   0.209  0.347
1227  -0.851  0.209
1228   0.002 -0.851
1229  -0.636  0.002
1230   1.216 -0.636
1231   0.032  1.216
1232  -0.236  0.032
1233   0.128 -0.236
1234  -0.501  0.128
1235  -0.122 -0.501
1236   0.281 -0.122
1237   0.084  0.281
1238   0.555  0.084
1239   0.419  0.555
1240  -0.141  0.419
1241  -0.285 -0.141
1242  -0.584 -0.285
1243  -0.024 -0.584
1244   0.252 -0.024
1245   0.422  0.252
1246   0.043  0.422
1247  -0.955  0.043
1248   0.130 -0.955
```

```
1249 -0.298  0.130

[1250 rows x 2 columns]
0         Up
1         Up
2       Down
3         Up
4         Up
5         Up
6       Down
7         Up
8         Up
9         Up
10      Down
11      Down
12        Up
13        Up
14      Down
15        Up
16      Down
17        Up
18      Down
19      Down
20      Down
21      Down
22        Up
23      Down
24      Down
25        Up
26      Down
27      Down
28      Down
29      Down
        ...
1220      Up
1221      Up
1222      Up
1223      Up
1224      Up
1225      Up
1226    Down
1227      Up
1228    Down
1229      Up
1230      Up
1231    Down
1232      Up
1233    Down
1234    Down
1235      Up
1236      Up
1237      Up
1238      Up
1239    Down
1240    Down
1241    Down
1242    Down
1243      Up
1244      Up
1245      Up
1246    Down
```

```
1247      Up
1248    Down
1249    Down
Name: Direction, Length: 1250, dtype: object
```

In [8]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random
_state=42)
```

In [9]:

```
plot_y = []
plot_x = []
for i in range(1,11):
    plot_y.append(i)
    print(str(i) + ":")
    neigh = KNeighborsClassifier(n_neighbors=i)
    neigh.fit(X_train,y_train) #Trains model
    score_test = neigh.score(X_test, y_test) #.score uses .predict as an underly
ing function
    plot_x.append(score_test)
    print("test_" + str(i) + ":")
    print(score_test)
```

```
1:
test_1:
0.5108958837772397
2:
test_2:
0.5302663438256658
3:
test_3:
0.46246973365617433
4:
test_4:
0.48668280871670705
5:
test_5:
0.4721549636803874
6:
test_6:
0.4552058111380145
7:
test_7:
0.48426150121065376
8:
test_8:
0.4745762711864407
9:
test_9:
0.46246973365617433
10:
test_10:
0.4891041162227603
```

In [10]:

```
plt.plot(plot_x)
plt.ylabel(plot_y)
plt.show()
```