למידה סטטיסטית מבוססת נתונים - חורף - 096411

HW2

מגישים:

איתי ברקוביץ 039632732

אילן פרנק 043493386

שאלה 2

שלם

יהי: $B = \|w^*\|$, $R = \max \|x_i\|$, $w^* = \arg\min\{\|w\| : \forall i, y_i \langle w, x \rangle \geq 1\}$

א. נוכיח כי לאחר T איטרציות נקבל $\langle w^*, w^{(T+1)} \rangle \geq T$

הוכחה:

- באיטרציה הראשונה : איבר הראשון = $w^{(1)} = (0 \dots 0)$

לכן: $\langle w^*, w^{(1)} \rangle = 0$

- באיטרציה ה-t נעדכן את w^{t+1} ע"י (x_i, y_i):

$$\langle w^*, w^{t+1} \rangle - \langle w^*, w^{(t)} \rangle = \langle w^*, w^{t+1} - w^t \rangle$$
$$= \langle w^*, y_i x_i \rangle = y_i \langle w^*, x_i \rangle$$
$$\geq 1$$

- לכן כאשר נצבר T איטרציות נקבל

$$\langle w^*, w^{(T+1)} \rangle = \sum_{t=1}^{T} \left(\langle w^*, w^{t+1} \rangle - \langle w^*, w^{(t)} \rangle \right) = \sum_{t=1}^{T} \geq 1 \geq T$$

$$\Rightarrow \langle w^*, w^{(T+1)} \rangle \geq T$$

ב. נוכיח שלאחר מעבר T איטרציות מתקיים $\|w^{(T+1)}\|^2 \leq TR^2$

הוכחה:

- עבור איטרציה t, נקבל:

$$\|w^{(t+1)}\|^2 = \|w^{(t)} + y_i x_i\|^2$$
$$= \|w^{(t)}\|^2 + 2y_i \langle w^{(t)}, x_i \rangle + y_i^2 \|x_i\|^2$$
$$\leq \|w^{(t)}\|^2 + R^2$$

- האי שוויון האחרון מתקבל מכיוון שאם עדכנו את w^{t+1}, מהגדרת $y_i \langle w^t, x_i \rangle \leq 0$, והנורמה של x_i הוא לכל היותר R.

מכיוון שהאיבר הראשון = 0 אם נצבר T איטרציות מהגדרת נקבל:

$$\|w^{(t+1)}\|^2 \leq TR^2$$

ג. נראה שלאחר ביצוע T איטרציות נקבל:

$$1 \geq \frac{\langle w^*, w^{(t+1)} \rangle}{\|w^*\| \|w^{t+1}\|} \geq \frac{\sqrt{T}}{RB}$$

- הערך $\frac{\sqrt{T}}{RB}$ מייצג את הזווית בין וקטור w^* לוקטור $w^{(t+1)}$

לכן לפי הגדרה $\max\cos(\alpha) \leq 1$ לכן $1 \geq \frac{\sqrt{T}}{RB}$

- מסעיף ד:

$$\|w^{(t+1)}\|^2 \leq TR^2 \Rightarrow \|W^{(T+1)}\| \leq \sqrt{T}R$$

- ונחבר את המשוואות מסעיפים א+ד ונזכר שנתון $\|W^*\| = B$ נקבל:

$$1 \geq \frac{\langle w^{(T+1)}, w^* \rangle}{\|w^*\| \|w^{(T+1)}\|} \geq \frac{T}{B\sqrt{T}R} = \frac{\sqrt{T}}{BR}$$

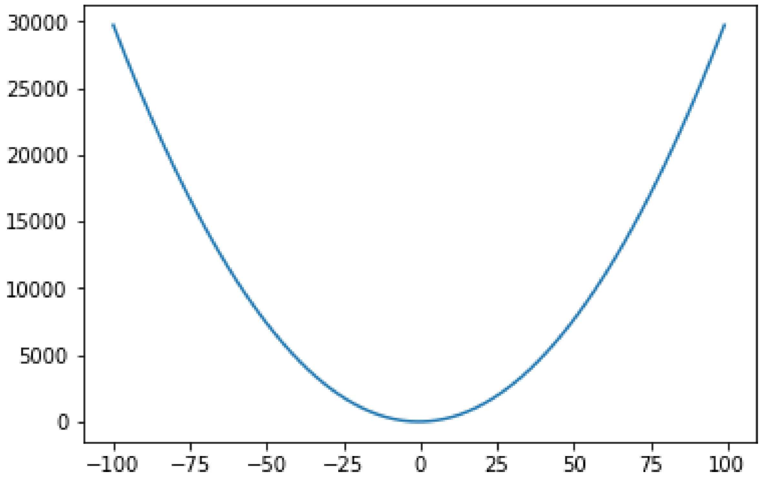
3. מסעיף ג ניתן להסיק:

$$1 \geq \frac{\sqrt{T}}{BR} \rightarrow (BR)^2 \geq T$$

לכן מובטחת התכנסות אחרי מספר איטרציות חסום.

In [6]:

```
#שאלה 2
from utils import load_mnist
from sklearn.model_selection import train_test_split

import numpy as np
import matplotlib.pyplot as plt


#סעיף ב
def func(x):
    x = np.array(x)
    y = eval('3+3*x+3*x**2')
    plt.plot(x, y)
    plt.show()
    pass
x = range(-100, 100)
func(x)
```



In [7]:

```
#סעיף ב
def grad(x):

    return 3+6*x
```

In [8]:

```
#נקודת הקיצון של הפונקציה
x_min = -0.5
```

In [9]:

```
#סעיף ד
def updt(grad, x, a):
    return x - a*grad(x)
```

In [10]:

```
#סעיף ה
t = 0.01
a = 0.1
i = 1
curr_x = x[0]
next_x = updt(grad, curr_x , a)

while abs(next_x-curr_x) > t:
    curr_x = next_x
    next_x = updt(grad, curr_x , a)

print(next_x)
print(f'The value is different because of the convergence condition t={t}.')
print('As t goes to 0, the value we get will be closer to -0.5.')
```

```
-0.50417333248
The value is different because of the convergence condition t=0.01.
As t goes to 0, the value we get will be closer to -0.5.
```

In [14]:

```
#question 3:

#download data:
import requests
import pandas as pd

from sklearn.datasets import fetch_mldata


def load_mnist():
    try:
        mnist = fetch_mldata('MNIST original')

        data_df = pd.DataFrame(mnist['data'])
        label_df = pd.DataFrame(mnist['target'])

        return data_df, label_df
    except requests.exceptions.RequestException:
        print('HTTP exception, check you connection and try again')

data_df , label_df = load_mnist()
```

In [15]:

```
import numpy as np
data_df = np.array(data_df)
label_df = np.array(label_df)
```

In [16]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data_df, label_df, test_size=0.25,
random_state=1000)
```

In [18]:

```
#נירמול הערכים
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
C:\Users\lenovo\Anaconda3\lib\site-packages\sklearn\utils\validation.py:47
5: DataConversionWarning: Data with input dtype uint8 was converted to flo
at64 by MinMaxScaler.
  warnings.warn(msg, DataConversionWarning)
```

In [19]:

```
#סעיף א: התאמת מודל
from sklearn.svm import SVC # "Support vector classifier"
model = SVC(kernel='linear')
model.fit(X_train, y_train.ravel())
```

Out[19]:

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='linear',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

In [20]:

```
#סעיף ב
y_pred = model.predict(X_test)


from sklearn.metrics import confusion_matrix

confusion_matrix(y_pred, y_test)
```

Out[20]:

```
array([[416,   1,   2,   0,   1,   4,   5,   0,   3,   1],
       [  0, 482,  13,   4,   1,   2,   0,   1,   4,   1],
       [  2,   2, 388,   4,   2,   9,   5,   8,   9,   6],
       [  2,   3,   5, 386,   1,  20,   1,   3,  10,   2],
       [  0,   2,   6,   0, 418,   5,   3,   8,   2,  23],
       [  6,   2,   0,  20,   1, 368,   2,   2,  14,  10],
       [  5,   0,   6,   0,   3,   6, 395,   0,   3,   0],
       [  0,   0,   4,   4,   1,   0,   2, 404,   3,  13],
       [  2,   2,   8,   7,   0,  10,   1,   1, 365,   2],
       [  0,   0,   2,   4,  13,   1,   0,  13,   5, 389]], dtype=int64)
```

In [21]:

```
from sklearn import metrics
print (metrics.accuracy_score(y_test,y_pred))
```

```
0.9168
```

In [23]:

```
model.score(X_test,y_test)
```

Out[23]:

```
0.9168
```

In [24]:

```
model.score(X_train,y_train)
```

Out[24]:

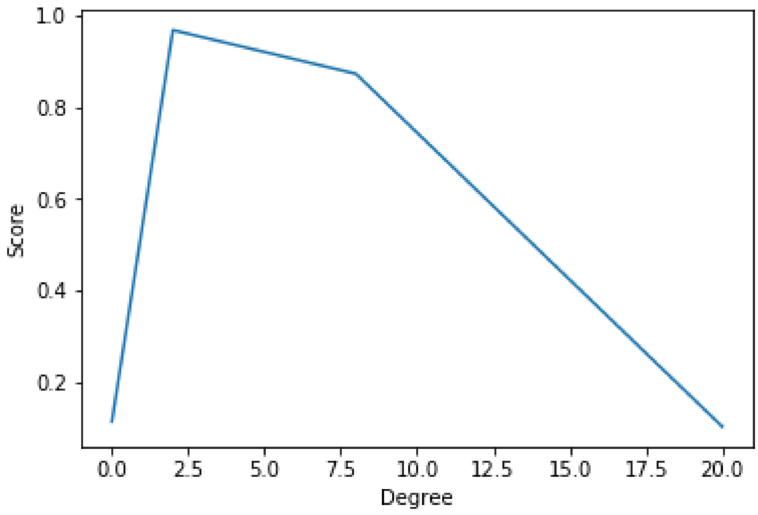
```
1.0
```

In [17]:

```
#סעיף ג
#changing the degress of the polynomial:

from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
import matplotlib.pyplot as plt

ind = [0,1,2,3]
degrees= [0,2,8,20]
score_matrix = np.zeros(4)









for i, deg in zip(ind,degrees):
    model = SVC(kernel = 'poly', degree = deg)
    model.fit(X_train, y_train.ravel())
    y_pred = model.predict(X_test)
    score_matrix[i] = metrics.accuracy_score(y_test,y_pred)






plt.figure()
plt.plot(degrees, score_matrix)
plt.xlabel('Degree')
plt.ylabel('Score')
```

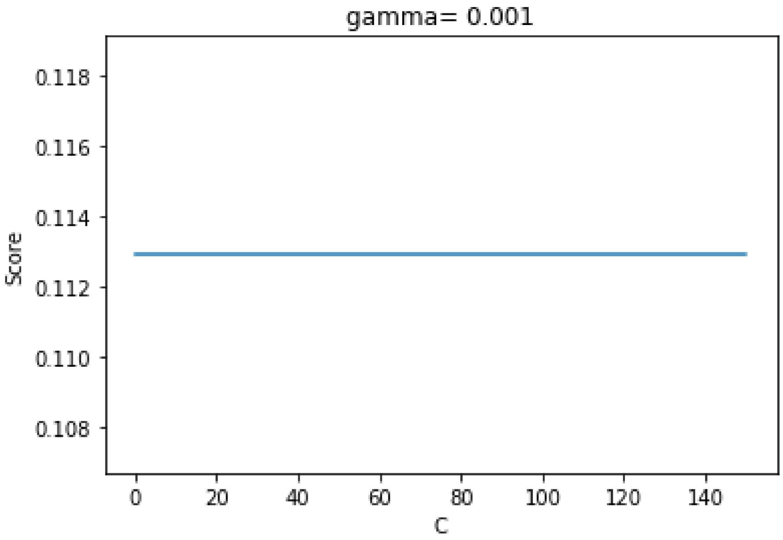
Out[17]:

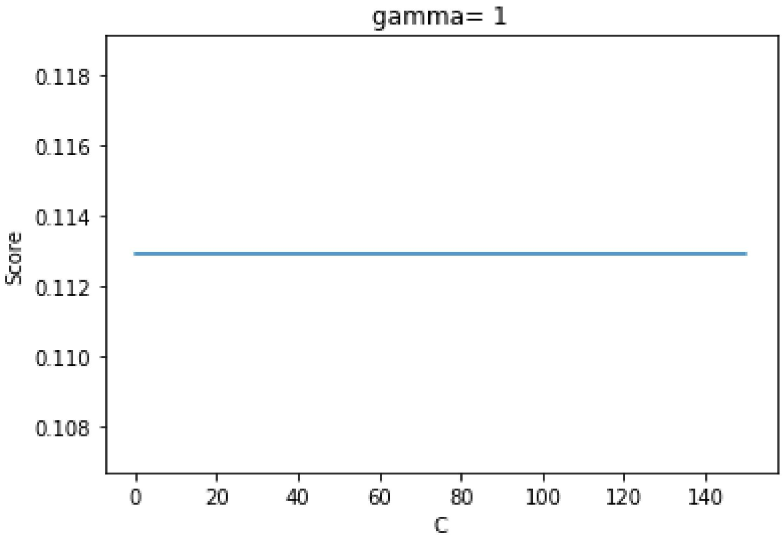
```
Text(0,0.5,'Score')
```

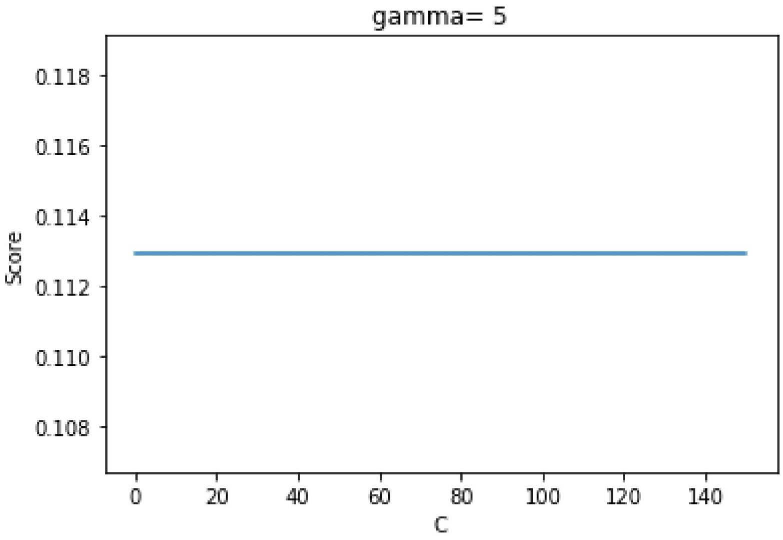


In [24]:

```
#changing the X and gamma parameter for rbf kernel function:
index = [0,1,2]
gamma= [0.001, 1 , 5]
C_parameter = [0.1,1,150]
score_matrix = np.zeros(3)


for this_gamma in gamma[:3]:
    for i, this_C in zip(index,C_parameter):
        model = SVC(kernel = 'rbf', gamma = this_gamma,C = this_C)
        model.fit(X_train, y_train.ravel())
        y_pred = model.predict(X_test)
        score_matrix[i] = metrics.accuracy_score(y_test,y_pred)


    plt.figure()
    plt.plot(C_parameter, score_matrix)
    plt.title('gamma= {}'.format(this_gamma))
    plt.xlabel('C')
    plt.ylabel('Score')
```

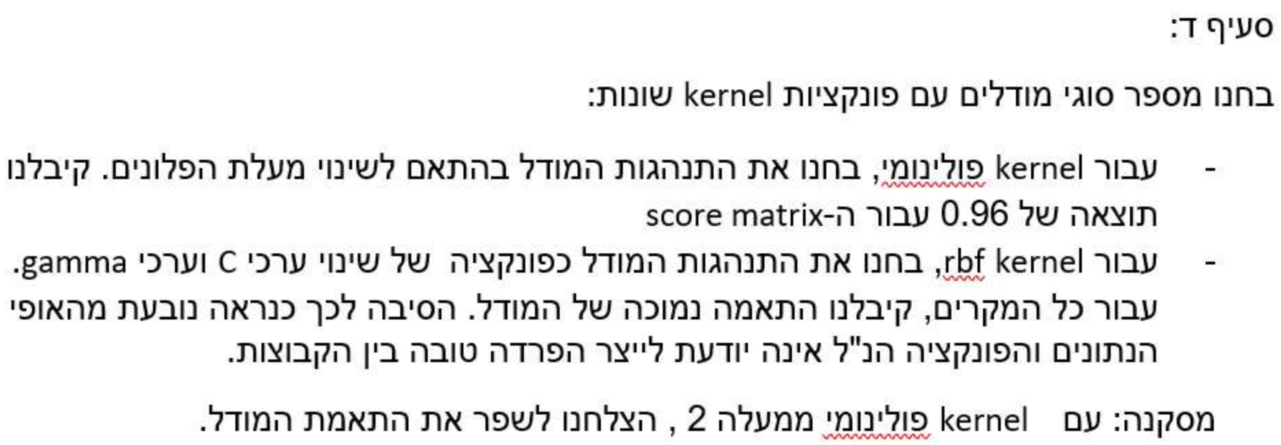

gamma= 0.001
Score
0.118
0.116
0.114
0.112
0.110
0.108
0
20
40
60
80
100
120
140
C


gamma= 1
Score
0.118
0.116
0.114
0.112
0.110
0.108
0
20
40
60
80
100
120
140
C


gamma= 5
Score
0.118
0.116
0.114
0.112
0.110
0.108
0
20
40
60
80
100
120
140
C

In [26]:

```
model = SVC(kernel = 'rbf', gamma = 0.1, C = 2)
model.fit(X_train, y_train.ravel())
y_pred = model.predict(X_test)
score_matrix = metrics.accuracy_score(y_test,y_pred)
```



סעיף ד:

בחנו מספר סוגי מודלים עם פונקציות kernel שונות:

- עבור kernel פולינומי, בחנו את התנהגות המודל בהתאם לשינוי מעלת הפלונים. קיבלנו תוצאה של 0.96 עבור ה-score matrix
- עבור rbf kernel, בחנו את התנהגות המודל כפונקציה של שינוי ערכי C וערכי gamma. עבור כל המקרים, קיבלנו התאמה נמוכה של המודל. הסיבה לכך כנראה נובעת מהאופי הנתונים והפונקציה הנ"ל אינה יודעת לייצר הפרדה טובה בין הקבוצות.

מסקנה: עם kernel פולינומי ממעלה 2 , הצלחנו לשפר את התאמת המודל.

In [11]:

```
#4 שאלה


import datetime
from sklearn.metrics import confusion_matrix
from utils import load_mnist
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.datasets.samples_generator import make_blobs
import numpy as np
from sklearn.datasets import load_iris

def ex4(data):
    p_x = data["X"]
    p_y = data["y"]
    input_dim = len(p_x[0])
    w = np.zeros(input_dim )
    max_reps = 1000

    for t in range(max_reps):
        indicator = 1
        for i in range(len(p_x)):
            if np.dot(w, p_x[i])*p_y[i] <= 0:
                w +=p_x[i]*p_y[i]
                indicator = 0
                break
        if indicator == 1:
            return (w.transpose(),t)

#בדיקת האלגוריתם
def test_perceptron():

    data = load_iris()
    p_x = np.array(data.data)
    p_y = np.array(data.target)
    # only the first class is separable
    selected_class = 0
    p_y[p_y != selected_class] = -1
    p_y[p_y == selected_class] = 1
    w,t = ex4({"X": p_x, "y": p_y})
    print(w,t)
test_perceptron()
```

```
[ 1.3  4.1 -5.2 -2.2] 5
```