תרגיל מספר 2 – רגרסיה לוגיסטית ורגולריזציה

1. (20 נקודות) בתרגיל זה נעשה שימוש באלגוריתם ה- Gradient Descent למימוש רגרסיה לוגיסטית לפתרון בעיית סווג.

נניח כי קבוצת נתוני האימון מייצגת תוצאות מבחני קבלה של 80 סטודנטים שהתקבלו לחוג למדעי המחשב ולחוג לביואינפורמטיקה, מהם 40 שהצליחו בלימודים ו- 40 שלא הצליחו. כל דגימת אימון i (סטודנט) מיוצגת על-ידי $\left(x^{(i)}, y^{(i)}\right)$, כאשר המשתנה הראשון $\left(x^{(i)}\right)$ הוא וקטור המכיל את ציוני שני המבחנים, והמשתנה השני $\left(y^{(i)}\right)$ הוא סקלר – תגית המסמנת אם הסטודנט הצליח בלימודיו בחוג (1) או לא הצליח (0).

בנו מערכת מסווג בינארי באמצעות רגרסיה לוגיסטית המשערכת את סיכויי סטודנטים להצליח בלימודים בחוג על סמך תוצאות שני מבחנים, ומבצעת קבלה על בסיס הסיכוי להצלחה.

נתוני האימון נמצאים באתר הקורס במודל במחיצה Materials for Ex. 2 - 2024 - Logistic Regression (admittance_data.csv).

א. טענו את הנתונים לחלון העבודה באמצעות הפקודות הבאות:

```
# College admittance decision using logistic regression
import numpy as np
import matplotlib.pyplot as plt
import scipy.io as sio
import pandas as pd

Xdata = np.genfromtxt("admittance_data.csv", delimiter = ',',
skip_header = 1)
X_orig = Xdata[:,0:2]
## another option for reading the csv file with pandas.
## You should import pandas before using it: import pandas as pd
#Xdata = pd.read_csv("admittance_data.csv")
#data = Xdata.to_numpy()
#X_orig = data[:,0:2]
y = data[:,2]
m = y.size

# X_orig is the feature matrix (each row represents one student
grades)
# y - the corresponding label (1- admitted, 0 - not admitted)
```

ב. ציור נקודות קבוצת האימון: ציירו את נתוני קבוצת האימון, כאשר נתוני סטודנטים שהצליחו בלימודים מסומנים בעיגול ירוק, וסטודנטים שלא על-ידי x אדום.

```
x1 = X_orig[: , 0]

x2 = X_orig[: , 1]

##### please add here a line to plot the training set.#######

plt.grid(), plt.show()
```

ג. בסעיף זה נכתוב את פונקציה לחישוב פונקציית המחיר J, וכן את הפונקציה grad_descent_logreg המבצעת את אלגוריתם ה- gradient descent.

ראשית כתבו פונקציה למימוש פונקציית הסיגמואיד.

לאחר מכן עליכם לכתוב פונקציה למימוש ה- gradient descent, בה העדכון יתבצע עבור כל דוגמאות האימון (Batch), ועבור כל התכונות באופן מטריצי.

פונקציית המחיר אותה יש למזער היא:

$$J(\theta) = -\frac{1}{m}\Big(y^{(1)} \log(h_\theta(x^{(1)})) + (1-y^{(1)})\log(1-h_\theta(x^{(1)})) + y^{(2)} \log(h_\theta(x^{(2)})) + (1-y^{(2)})\log(1-h_\theta(x^{(2)})) + \dots y^{(m)} \log(h_\theta(x^{(m)})) + (1-y^{(m)})\log(1-h_\theta(x^{(m)}))\Big)$$

או באופן מפורט:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log\left(h_\theta(x^{(i)})\right)+\left(1-y^{(i)}\right)\log\left(1-h_\theta(x^{(i)})\right)\right)$$

נתבונן בביטוי הראשון בכל אחד מהמחוברים:

המשתנה $y^{(i)}$ הוא סקלר, וכן $h_\theta(x^{(i)})$. לכן החישוב של $J(\theta)$ באופן מטריצי (וקטורי) הוא כפל הוקטור y המכיל את כל התגיות בוקטור $h_\theta(x)$, המכיל את פונקציית ההיפותיזה עבור כל דוגמאות האימון:

באופן זה נבצע את חישוב פונקציית המחיר לכל איטרציה באופן הבא:

```
np.dot(y.T, np.log(h_theta)) + np.dot((1-y).T, np.log(1-
h_theta)))
```

כתבו פונקציה שתבצע את חישוב $J(\theta)$ לכל איטרציה. הפונקציה תקבל בכניסה את ערכי theta, X, ו- y, ותחזיר את פונקציית המחיר ואת ערך הגרדיאנט לכל איטרציה.

```
def computeCost(X, y, theta):
    """
COMPUTECOST Compute cost for logistic regression.
Computes the cost of using theta parameters for logistic regression
Input arguments: X -  input matrix (np array), observations
(features) in rows.     y -  (np array) output vector (labels) for
each input sample, theta - (np array) parameters vector, weights of
the measured features
Output arguments:
return - J - the cost function for theta
Usage: J = computeCost(X, y, theta)
    """

    m = y.size
    J = 0
    grad_J = np.zeros(theta.shape)
    Z = np.dot(X,theta)
    h_theta = sigmoid(Z)
    ##### please add here a line to compute J #####
    J = -1 / m * __________
    ##### please add here a line to compute grad_J #####
    grad_J =________________
    return J, grad_J
```

הפונקציה `gradDescent_log` המתוארת בהמשך, תשתמש בפונקציה זו לחישוב הפרמטרים θ.

```
def gradDescent_log(X, y, theta, alpha, num_iters):
    """
    gradDescent - Batch implementation of GD algorithm
    for logistic regression using matrix-vector operations
    Input arguments:
    X - (m, n) numpy matrix - each row is a feature vector,
    y - (m,1) np array of target values,
    theta - (n,1) initial parameter vector,
    alpha - learning rate
    num_iters - number of iterations.
    returns theta - parameters vector and J_iter - (num_iter,1)
    cost function vector.

    """
    J_iter = np.zeros((num_iters, 1))
    for iter in range(num_iters):
        J_iter[iter], grad = computeCost(X, y, theta)
        ####### please add here a line  to update theta ########
        theta =

    plt.plot(J_iter)
    return theta, J_iter
```

ד. עתה נבצע את הסווג ללא נירמול הנתונים:

```
onesvec = np.ones((m ,1))
X = np.concatenate((onesvec, X_orig), axis = 1)
n = X.shape[1]
theta = np.zeros((n,1))
y = y.reshape([y.shape[0], 1])
J, grad_J = computeCost(X, y, theta)
alpha = 0.001
num_iters = 90000
theta, J_iter = gradDescent_log(X, y, theta, alpha, num_iters)
plt.plot(J_iter)
plt.show()
plot_logreg_line(X, y, theta)
```

ה. עתה נבחן כיצד מבוצע הלימוד לאחר נירמול של הנתונים.

לצורך כך יש להפחית מכל אחת מהתכונות בכל דוגמא את ממוצע התכונה בקבוצת האימון, ולחלק בסטיית התקן של התכונה.

האם יש הבדל בין התוצאות לפני ואחרי הנירמול? רשמו את מספר האיטרציות הנדרשות עבור כל שלב.

ו. עתה נרצה לבחון את משטח ההחלטה. במקרה זה וקטור התכונות מכיל רק שתי תכונות (הציונים של שני המבחנים) ולכן משטח ההחלטה הוא עקום על המישור. מאחר ו- $\theta^T X = \theta_0 + \theta_1 x_1 + \theta_2 x_2$, משטח ההחלטה הוא ישר. ההחלטה עבור כל סטודנט או דגימה מקבוצת האימון היא "הצליח" (1) אם:

$$g\left(\theta^T X\right) = g\left(\theta_0 + \theta_1 x_1 + \theta_2 x_2\right) = \frac{1}{1 + \mathrm{e}^{-(\theta_0 + \theta_1 x_1 + \theta_2 x_2)}} > 0.5$$

תנאי זה מתקיים כאשר $\theta^T X = \theta_0 + \theta_1 x_1 + \theta_2 x_2 > 0$.

באופן דומה ההחלטה עבור כל סטודנט או דגימה מקבוצת האימון היא "לא הצליח"(0) אם:

$$g\left(\theta^T X\right) = g\left(\theta_0 + \theta_1 x_1 + \theta_2 x_2\right) = \frac{1}{1 + \mathrm{e}^{-(\theta_0 + \theta_1 x_1 + \theta_2 x_2)}} < 0.5$$

תנאי זה מתקיים כאשר $\theta^T X = \theta_0 + \theta_1 x_1 + \theta_2 x_2 < 0$.

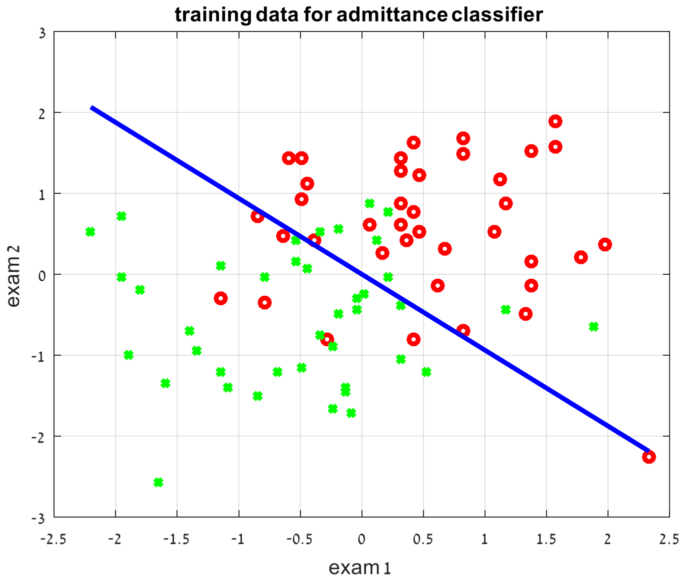
משוואת הישר המפריד היא אם כן: $\theta_0 + \theta_1 x_1 + \theta_2 x_2 = 0$, ולכן:

$$x_2 = \frac{-\theta_0 - \theta_1 x_1}{\theta_2}.$$

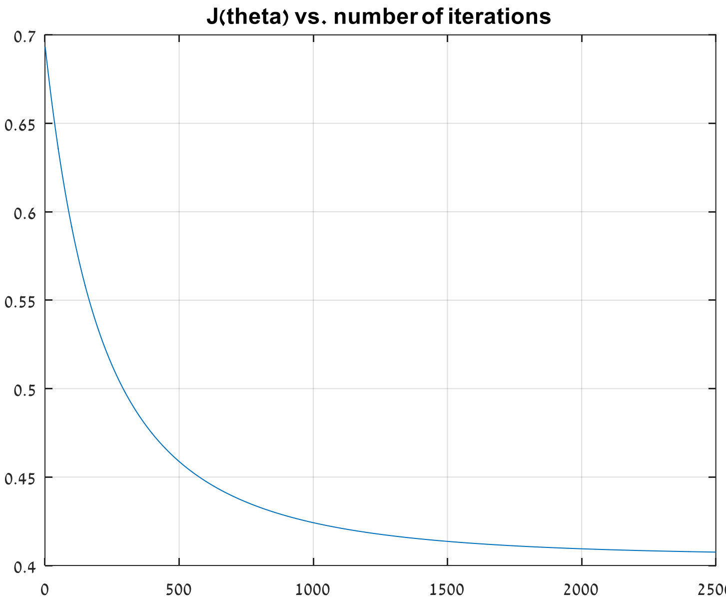
נצייר את משטח ההפרדה על-ידי :

```
def plot_logreg_line(X, y, theta):
    """
    plot_reg_line plots the data points and regression line for logistic regrssion
    Input arguments: X - np array (m, n) - independent variable.
    y - np array (m,1) - target variable
    theta - parameters
    The function is for 2-d input - x2 = -(theta[0] + theta[1]*x1)/theta[2]
    """
    ind = 1
    x1_min = 1.1*X[:,ind].min()
    x1_max = 1.1*X[:,ind].max()
    x2_min = -(theta[0] + theta[1]*x1_min)/theta[2]
    x2_max = -(theta[0] + theta[1]*x1_max)/theta[2]
    x1 = X
    x1lh = np.array([x1_min, x1_max])
    x2lh = np.array([x2_min, x2_max])

    x1 = X[: , 1]
    x2 = X[: , 2]
    ### please add here a line to plot the points and the decision boundary
    plt.plot(…
    plt.xlabel('x1'), plt.ylabel('x2')
    plt.title('data')
    plt.grid(),   plt.show()
```



ז. נבחן האם אלגוריתם ה- gd מתכנס וכן האם הפרמטרים (alpha, num_iters) מתאימים על-ידי ציור פונקציית המחיר J כתלות במספר האיטרציות.



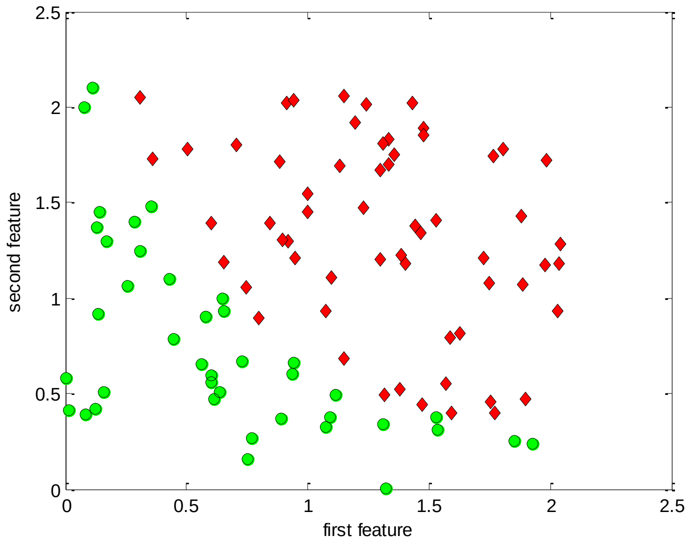
ח. עתה נבחן האם הסטודנט יתקבל באופן אוטומטי (כלומר על-פי החלטת המסווג) אם ציוני המבחנים שלו 65 במבחן הראשון ו- 41 במבחן השני. מהי ההסתברות של סטודנט להצליח בלימודים? מהי ההסתברות של סטודנטית להצליח בלימודים אם ציון המבחן הראשון שלה הוא 53 והשני 85?

2. (20 נקודות) בתרגיל זה עליכם ליצור מערכת לומדת שתסווג דואר נכנס כמכתב אמיתי או כספאם (spam). בדרך כלל התכונות המאפשרות לזהות מכתב כספאם או כמייל הן השכיחויות של מלים מסויימות מתוך מילון של מלים בשפה. מלים מסויימות הן אופייניות לספאם ונדירות במכתב רגיל, וההפך.

בתרגיל נבנה מערכת לומדת מסוג logistic regression עבור קבוצת הלימוד הנתונה ב- email_data_1 (ראו ב- Materials for Ex. 2 2024 - Logistic Regression).

כל דוגמת לימוד מכילה וקטור של שתי תכונות (שורה במטריצה X) והתגיות המתאימות נמצאות בוקטור y.

א.   (2) ציירו את הנתונים עם צבעים וסימנים שונים לדוגמאות המכתבים (y=1) (צבע ירוק, עיגול) וה- spams (y=0) (צבע אדום)   כמו בציור 1.



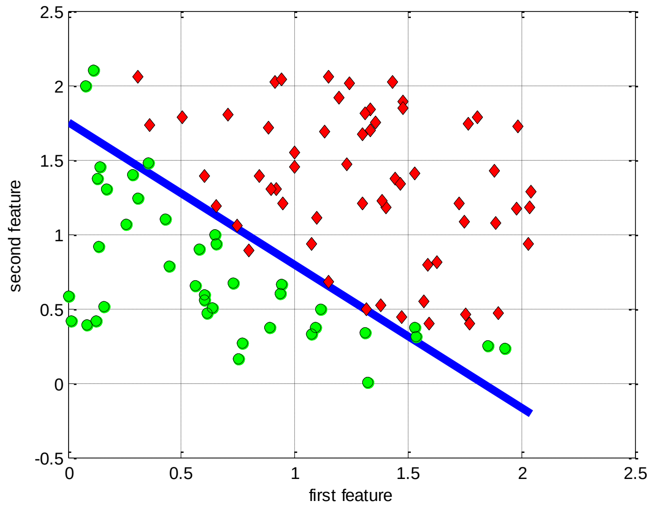
ציור 1

ב.   כתבו מימוש עבור הפונקציות הבאות וצרפו אותן לפתרון :

- חישוב פונקציית המחיר $J(\theta)$.

[J,grad]=costF_logreg(theta,X1,y)

- פונקציה למימוש אלגוריתם ה- gd.
- פונקציה לציור משטח או גבול ההפרדה (כלומר במקום השורות ב- script המממשות את הציור, כתבו פונקציה שתקבל את X ו- θ, ותצייר את משטח ההפרדה יחד עם דוגמאות הלימוד כמו בציור 2.

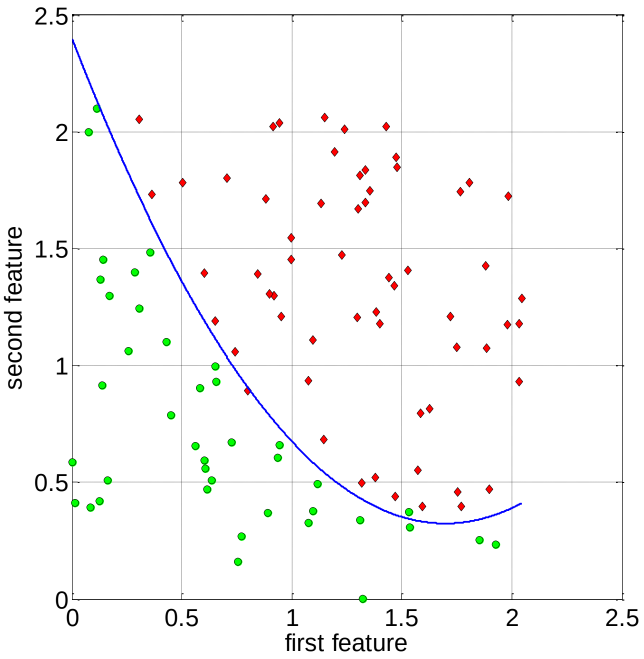


ציור 2

ג. הפכו את משטח ההפרדה לקוודרטי וציירו את התוצאה (הדרכה: כתבו פונקציה לציור משטח הפרדה ריבועי כך שהמשתנה z הוא מהצורה

$(z=\theta_0+\theta_1 x_1+\theta_2 x_2+\theta_3 x_1^2$

העזרו בציור 3.



ציור 3.

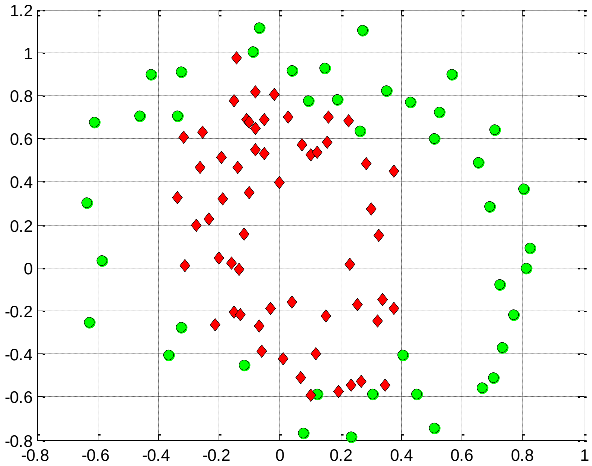
ד. ציירו את פונקציית המחיר כתלות במספר האיטרציות, ובחנו ערכי α שונים כפי שנלמד בהרצאה. כתבו בפתרון מה דעתכם, כלומר איזה ערכים בחנתם ועבור איזה ערך ההתכנסות היא הכי טובה.

ה. עבור קבוצת דוגמאות המבחן השמורה ב- email_data_test_2024.csv, חשבו ורשמו כמה דוגמאות יסווגו נכון עבור כל אחד מהפתרונות (הליניארי והריבועי), כאשר ytest היא התגית y המתאימה, וחשבו את אחוז הזיהוי הנכון.

3. (20 נקודות) כדי לבנות מערכת לומדת לזיהוי דואר spam נעשו שתי מדידות שונות, x1 ו – x2, באמצעותן רוצים לסווג את הדואר הנכנס לשרת e-mail למכתבים אמיתיים ול- spams.

לרשותכם קבוצת אימון של שתי דוגמאות, עבורן לכל אחת שתי תכונות, וידוע האם כל דוגמא היא מכתב או spam, כלומר הדוגמאות מתוייגות.

הנתונים מוצגים באמצעות דיאגרמת הפיזור הבאה:



א. הנתונים נמצאים במחיצת Materials for Ex. 2_2024 - Logistic Regression ב-
.email_data_2_2021

טענו את הנתונים של קבוצת האימון על-ידי

```
Xdata = pd.read_csv("email_data_2.csv("
data = Xdata.to_numpy()
X_orig = data[:,0:2[
y = data[:,2[
m = y.size
```

יצרו דיאגרמת פיזור בדומה לציור למעלה, בו כל נקודה מיוצגת על-ידי שתי התכונות x1 ו- x2, ומסומנת על-ידי עיגול ירוק אם הדואר תקין y=1 ו- x אדום עבור spam (y=0).

ב. השתמשו ברגרסיה לוגיסטית כדי להפריד בין שתי הקבוצות (דואר רגיל ודואר spam). מהי מסקנתכם? כתבו באופן מפורש.

ג. כדי להפריד בין דוגמאות האימון של שתי הקבוצות נשתמש ברגרסיה לוגיסטית עם פולינום מסדר גבוה יותר. נבצע התאמת מודל מסדר חמישי במקום המודל הליניארי . נשתמש בפונקציה mapFeature באופן הבא:

```
x1 = X[: , 0]
x2 = X[: , 1]
X = map_feature(x1, x2)
```

כאשר הפונקציה map_feature נמצאת במחיצת החומרים לתרגיל זה.

וקטור התכונות הוא עתה: $z = \left(1 \ \ x_1 \ \ x_2 \ \ x_1^{\ 2} \ \ x_1 x_2 \ \ x_2^{\ 2} \ \ x_1^{\ 3} \ \ldots \ \ x_1 x_2^{\ 5} \ \ x_2^{\ 6}\right)^T$

כתבו פונקציה שתבצע את פעולת המערכת הלומדת באמצעות פונקציית המחיר (עם רגולריזציה) הבאה:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2.$$

וכן הגרדיאנט:

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j \qquad \text{for } j \ge 1$$

הדרכה: 1. כתבו פונקציה שתחשב את פונקציית המחיר והגרדיאנט עם רגולריזציה.
2. כתבו פונקציה למימוש אלגוריתם ה- gd.

```
theta, J_iter = gd_reg(X, y, theta, alpha,
num_iters,lambda)
```

צרפו את הפונקציות לפיתרון.

ד. הפעילו את הפונקציות שכתבתם על הנתונים בשלב הראשון עם ערך $\lambda = 0$.

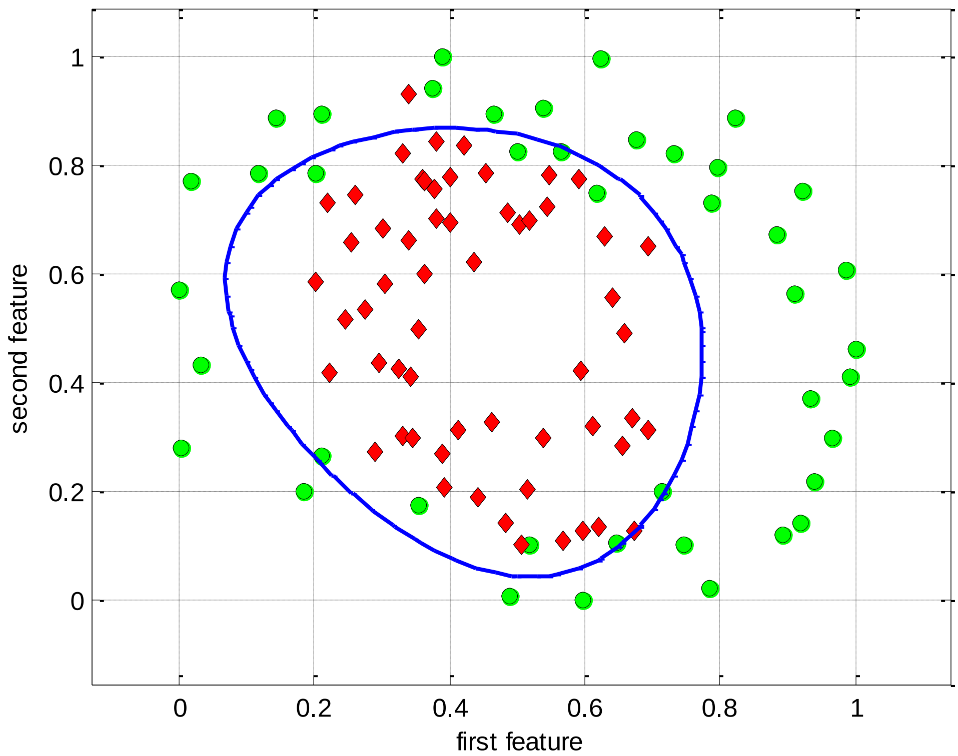
ציירו את משטח ההחלטה שקיבלתם על-ידי

plotDecisionBoundary(theta, X, y)

(plotDecisionBoundary נמצא בתיקיה Materials for Ex. 2_2024 - Logistic Regression)

ה. בצעו שוב את סעיף ד' והשתמשו ברגולריזציה כדי לשפר את התוצאה. בחנו ערכי λ שונים (על לפחות 7 ערכים, לדוגמא λ =0.01, 0.05, 0.1, 0.5, 1, 5, 10. כתבו מה ההשפעה של ערכי λ על גבול ההחלטה.

ו. עבור המודל שקיבלתם, מה תהיה שגיאת החיזוי עבור נתוני המבחן הנמצאים במחיצת Materials for Ex. 2_2024 - Logistic Regression ב- email_data_3_2024.csv?



4. (20 נקודות) בתרגיל זה נבצע מימוש של אלגוריתם רגרסיה לוגיסטית לסווג רב-מחלקתי multi-class logistic regression. נשתמש ב- dataset ידוע מאוד המכונה Fisher Iris dataset (ראו גם ב- https://en.wikipedia.org/wiki/Iris_flower_data_set). המאגר כולל נתוני מדידות אורך ורוחב של עלי הכותרת ועלי הגביע של מספר צמחים – מסוג אירוס – Virginica, Versicolor ו- Setosa, כאשר סה"כ יש במאגר 150 צמחים, 50 פרטים מכל מין.

האיור הבא מציג את שלושת מיני הצמחים:

איור 1: שלושת מיני האירוס: Virginica, Versicolor ו-Setosa

מטרת התרגיל היא לזהות באופן אוטומטי את מין הצמח או הפרח לפי פרמטרים אלה:
אורך ורוחב עלי הכותרת (petal), אורך ורוחב עלי הגביע (sepal).

דוגמא לנתונים מוצגת באיור הבא:

Dataset Order	Sepal length	Sepal width	Petal length	Petal width	Species
1	5.1	3.5	1.4	0.2	*I. setosa*
2	4.9	3.0	1.4	0.2	*I. setosa*
3	4.7	3.2	1.3	0.2	*I. setosa*
4	4.6	3.1	1.5	0.2	*I. setosa*
5	5.0	3.6	1.4	0.3	*I. setosa*
51	7.0	3.2	4.7	1.4	*I. versicolor*
52	6.4	3.2	4.5	1.5	*I. versicolor*
53	6.9	3.1	4.9	1.5	*I. versicolor*
54	5.5	2.3	4.0	1.3	*I. versicolor*
55	6.5	2.8	4.6	1.5	*I. versicolor*
101	6.3	3.3	6.0	2.5	*I. virginica*
102	5.8	2.7	5.1	1.9	*I. virginica*
103	7.1	3.0	5.9	2.1	*I. virginica*
104	6.3	2.9	5.6	1.8	*I. virginica*
105	6.5	3.0	5.8	2.2	*I. virginica*
106	7.6	3.0	6.6	2.1	*I. virginica*
107	4.9	2.5	4.5	1.7	*I. virginica*

איור 2: טבלה חלקית המציגה את נתוני אורך ורוחב עלי הגביע ועלי הכותרת של מיני אירוס שונים.

א. טענו את הנתונים על-ידי הפקודה:

```
import matplotlib.pyplot as plt
from sklearn import datasets

iris = datasets.load_iris()
X = iris.data[:, 1:3]  # we only take the first two features.
y = iris.target
```

```python
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5

plt.figure(2, figsize=(8, 6))
plt.clf()

# Plot the training points
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Set3,
            edgecolor='k')
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
```

ציירו דיאגרמת פיזור של אורך עלי הגביע כתלות באורך עלי הכותרת עבור קבוצת האימון המכילה את הפרטים 1-35, 51-85, ו- 101-135, כלומר 35 הדוגמאות הראשונות מכל אחד מהמינים. השתמשו בצבעים אדום עבור Setosa, ירוק עבור Versicolor וכחול עבור Virginica (ראו איור 3 עבור אורך ורוחב עלי הכותרת).

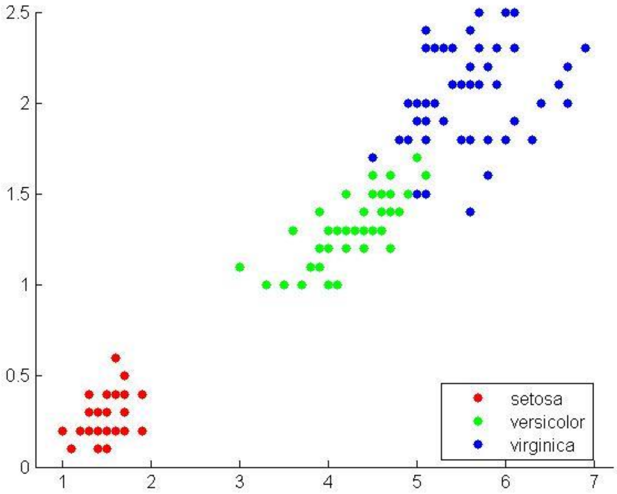
ב. בסעיף זה נשתמש ברגרסיה לוגיסטית עם סווג רב מחלקתי לסווג המינים על-פי עמודות 2 ו- 3 בלבד, כלומר אורך עלי הכותרת ורוחב עלי הגביע מקבוצת המדידות.

הדרכה: כפי שראינו בהרצאה, כדי לבצע סווג רב-מחלקתי נשתמש בשיטת one-versus-all classification. נאמן שלושה מסווגים בינאריים, כאשר כל אחד מהמסווגים מיועד ליצור גבול הפרדה בין אחת הקבוצות לאיחוד של שתי הקבוצות האחרות. לדוגמא: מסווג אחד יבצע הפרדה בין קבוצה אחת נניח Setosa, וקבוצה שניה הנוצרת מאיחוד של Versicolor ו- Virginica. באותו אופן נבצע אימון עבור שלושה מסווגים, כאשר קבוצת האימון מוגדרת בסעיף א׳. השתמשו בפונקציה אותה כתבתם בשאלה 1.

ג. ציירו על דיאגרמת הפיזור מסעיף א׳ את גבולות ההחלטה של המסווגים הבינאריים.

ד. לאחר מציאת שלושה מסווגים בינאריים בצעו סווג לקבוצת המבחן (פרטים 36-50, 86-100, וכן 136-150). כתבו פונקציה המקבלת כקלט את שלושת המסווגים מסעיף ב׳, את קבוצת המבחן, ואת התיוגים הנכונים של קבוצת המבחן, ומחזירה את התיוגים החזויים predicted values של קבוצת המבחן ואת אחוז השגיאה, כלומר מה מספר הזיהויים הנכונים ביחס לסה״כ הפרטים בקבוצת המבחן.

ה. חזרו על סעיף ב׳ ו-ד׳ תוך שימוש בכל הנתונים בקבוצות האימון והמבחן, כלומר אורך ורוחב עלי הגביע ואורך ורוחב עלי הכותרת של פרחי האירוס.



איור 3: דיאגרמת פיזור של נתוני האירוס עבור אורך ורוחב של עלי הכותרת.

5. (20 נקודות). בתרגיל זה נבצע מימוש של פרספטרון כמסווג ליניארי דרך הראשית.

א. (10 נקודות) טענו את הנתונים מ- Perceptron_exercise_2.npz ל- python באמצעות:

```
npzfile = np.load("Perceptron_exercise_2.npz")
sorted(npzfile.files)
X = npzfile['arr_0']
y = npzfile['arr_1']
```

השלימו את השורות החסרות עבור הפונקציות: perceptron_train ו- plot_points_and_boundary ב- perceptron_home_exercise_2.py (נמצא ב- Materials for Ex. 2_2024 - Logistic Regression) והריצו את הפרספטרון עבור נתוני האימון. רשמו את הערך של theta, את מספר האיטרציות k וצרפו ציור של גבול ההחלטה ונקודות הנתונים.

ב. (10 נקודות) הכלילו את הפונקציה perceptron_train כך שתוכל לסווג גם נקודות אימון פרידות אך לא רק סביב הראשית.

הדרכה: בכל איטרציה מעדכנים את ה- bias בנוסף לעדכון theta על-ידי: $b = b + yt$.

בחנו את הפונקציה המעודכנת על נתונים פרידים ליניארית אך לא דרך הראשית. אפשר להשתמש בקוד הנמצא בהערה משורה 102 ועד 110 ב- script. רשמו את הערך של theta, את מספר האיטרציות k וצרפו ציור של גבול ההחלטה ונקודות הנתונים.