Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Πρωτη Σειρα Ασκησεων
Μηχανικη Μαθηση

Αναστασία Χριστίνα Λίβα
03119029
anachriliva@gmail.com

Περιεχόμενα

Άσκηση 1.1: Linear and Ridge Regression

Αρχικά φορτώνω το dataset κανονικοποιώ κατάλληλα τις 3 στήλες που ζητούνται.

```
import pandas as pd

url='https://drive.google.com/file/d/1YDOE_-
    nWe4WFLC2Q_140zFe0J25H57Ks/view?usp=sharing'
file_id=url.split('/')[-2]

dwn_url='https://drive.google.com/uc?id=' + file_id


data = pd.read_csv(dwn_url)

print(data)

data['athletes'] = data['athletes'] - data['athletes'].mean()
data['athletes'] = data['athletes']/(data['athletes'].std())

data['events'] = data['events'] - data['events'].mean()
data['events'] = data['events']/(data['events'].std())

data['medals'] = data['medals'] - data['medals'].mean()
data['medals'] = data['medals']/(data['medals'].std())

print(data)
```

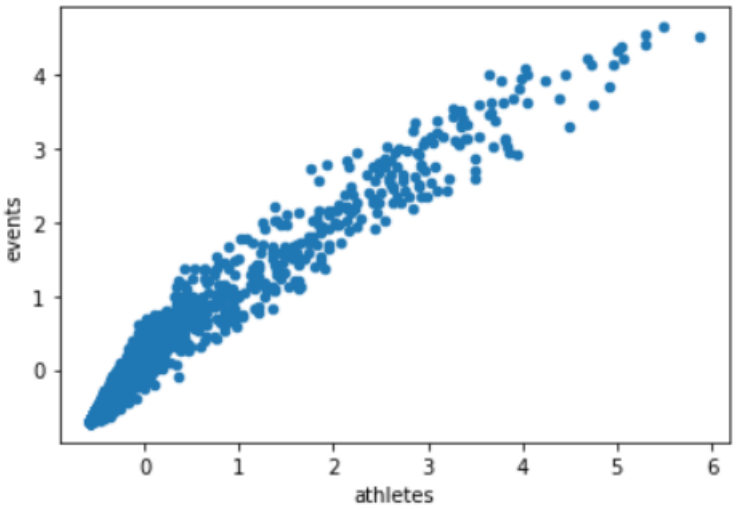
Στην συνέχεια υπολογίζω τη συσχέτιση μεταξύ των στηλών athletes και events:

```
corr = data['athletes'].corr(data['events'])
print("Correlation between Athletes and Events is: ", corr)
```

Οπότε παίρνω αποτέλεσμα: Correlation between Athletes and Events is: 0.9765633245334631

Απεικονίζω το scatter plot για αυτές τις δύο στήλες, το οποίο επιβεβαιώνει το παραπάνω αποτέλεσμα, αφού η σχέση τους είναι πρακτικά γραμμική, χρησιμοποιώντας τον παρακάτω κώδικα:

```
ax1 = data.plot.scatter(x='athletes',
                        y='events')
```



Συνεχίζοντας, δημιουργώ τα X και y και αντίστοιχα προχωρώ στον διαχωρισμό αυτών σε train και test. Παράλληλα μετατρέπω τα dataframes μου σε πίνακες numpy για να μπορώ να τα επεξεργαστώ καλύτερα και να κάνω πράξεις πινάκων με αυτά. Έτσι έχουμε:

```
X = data[['athletes', 'events']]
y = data[['medals']]

from sklearn.model_selection import train_test_split
import numpy as np

RAND_STATE = 0
  # Split into training and testing
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size
   =0.3, random_state=RAND_STATE)

X_train = X_train.to_numpy()
y_train = y_train.to_numpy()
X_val = X_val.to_numpy()
y_val = y_val.to_numpy()
```

Οπότε απομένει να υπολογίσω για την Linear και Ridge Regression τα βάρη και τα RMSE σφάλματα εκπαίδευσης και επαλήθευσης. Αρχικά θα πρέπει να ορίσω τον κατάλληλο υπολογισμό για τις προβλέψεις της Regression:

```
def reg(x1,x2,w1,w2):
  return w1*x1 + w2*x2
```

Το w_0 θα είναι πάντα 0 λόγω της κανονικοποίσης.

Συνεπώς πλέον μπορώ να υπολογίσω τα βάρη και τα RMSE σφάλματα εκπαίδευσης και επαλήθευσης εκτελόντας την παρακάτω διαδικασία.
Αρχικά για τα βάρη ισχύει πως:

$$w = (X^T X + \lambda I)^{-1} X^T y$$

οπότε με έναν απλό υπολογισμό μπορούν να υπολογιστούν. Για τα $RMSE$ βρίσκω τα *predictions* του κάθε Regression και παίρνω την τετραγωνισμένη διαφορά τους από τα ορθά αποτελέσματα. Ύστερα για όλες αυτές τις αποστάσεις θα υπολογίζω την ρίζα της μέσης τιμής τους, που είναι και το $RMSE$.

Ξεκινώντας από την Linear Regression καταλήγω στα εξής αποτελέσματα, με τον παρακάτω κώδικα:

```
weights_linearreg = np.matmul(
                                np.matmul(np. linalg .inv(np.matmul(
    np.transpose(X_train), X_train)), np.transpose(X_train))
                                , y_train)
print("Weights:")
print(weights_linearreg)

#RMSE Linear Regression

y_train_pred_LReg = [reg(X_train[i,0], X_train[i,1],
    weights_linearreg[0], weights_linearreg[1]) for i in range (len(
    y_train))]
LReg_train_errors = [(y_train_pred_LReg[i] - y_train[i])**2 for i
    in range (len(y_train))]
LReg_train_RMSE = np.sqrt(np.mean(LReg_train_errors))
print("Linear Regression Train RMSE: ", LReg_train_RMSE)

y_val_pred_LReg = [reg(X_val[i,0], X_val[i,1], weights_linearreg
    [0], weights_linearreg[1]) for i in range (len(y_val))]
LReg_test_errors = [(y_val_pred_LReg[i] - y_val[i])**2 for i in
    range (len(y_val))]
LReg_test_RMSE = np.sqrt(np.mean(LReg_test_errors))
print("Linear Regression Test RMSE: ", LReg_test_RMSE)

print("\n")
```

```
Weights:
[[ 1.95901654]
 [-1.14081497]]
Linear Regression Train RMSE:  0.497249644514127
Linear Regression Test RMSE:  0.4780379373897748
```

Ενώ αντίστοιχα για τις τρεις περιπτώσεις της Ridge Regression:

```
1
2  for l in [1,10,100]:
3    print("For l = ", l)
4    weights_ridgereg =  np.matmul(
5                                np.matmul(np. linalg .inv(np.matmul(
     np.transpose(X_train), X_train) + l*np.identity(2)), np.
     transpose(X_train))
6                                , y_train)
7    print("Weights:")
8    print(weights_ridgereg)
9
10   #RMSE Ridge Regression
11
12   y_train_pred_RReg = [reg(X_train[i,0], X_train[i,1],
      weights_ridgereg[0], weights_ridgereg[1]) for i in range (len(
      y_train))]
13   RReg_train_errors = [(y_train_pred_RReg[i] - y_train[i])**2 for i
       in range (len(y_train))]
14   RReg_train_RMSE = np.sqrt(np.mean(RReg_train_errors))
15   print("Ridge Regression Train RMSE for l = ", l, ": ",
      RReg_train_RMSE)
16
17   y_val_pred_RReg = [reg(X_val[i,0], X_val[i,1], weights_ridgereg
      [0], weights_ridgereg[1]) for i in range (len(y_val))]
18   RReg_test_errors = [(y_val_pred_RReg[i] - y_val[i])**2 for i in
      range (len(y_val))]
19   RReg_test_RMSE = np.sqrt(np.mean(RReg_test_errors))
20   print("Ridge Regression Test RMSE for l = ", l, ": ",
      RReg_test_RMSE)
21   print("\n")
```

```
For l =  1
Weights:
[[ 1.91525931]
 [-1.09715191]]
Ridge Regression Train RMSE for l =  1 :  0.4973436313526021
Ridge Regression Test RMSE for l =  1 :  0.47708833035492465


For l =  10
Weights:
[[ 1.60977308]
 [-0.79290706]]
Ridge Regression Train RMSE for l =  10 :  0.5031949175010084
Ridge Regression Test RMSE for l =  10 :  0.475118406272322


For l =  100
Weights:
[[ 7.95255560e-01]
 [-1.34190165e-04]]
Ridge Regression Train RMSE for l =  100 :  0.5594211834826948
Ridge Regression Test RMSE for l =  100 :  0.5077851547859091
```

Παρουσιάζοντας τα τελικά αποτελέσματα σε έναν πίνακα:

	Linear Regression ($\lambda = 0$)	Ridge Regression ($\lambda = 1$)	Ridge Regression ($\lambda = 10$)	Ridge Regression ($\lambda = 100$)
RMSE (Train Set)	0.497	0.497	0.503	0.559
RMSE (Test Set)	0.478	0.477	0.475	0.508

Από τις 4 περιπτώσεις θα επέλεγα αυτή με τιμή $\lambda = 10$ αφού, παρότι δεν δίνει το καλύτερο RMSE Train δίνει το καλύτερο RMSE Test, πράγμα που το καθιστά πιο αποτελεσματικό πάνω σε μοντέλα τα οποία δεν γνωρίζει.

Άσκηση 2: Gaussian Distribution

Ερώτημα Πρώτο: Εφόσον η τυχαία μεταβλητή x ακολουθεί την κανονική κατανομή με παραμέτρους μ και σ^2, τότε ισχύει

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Επομένως

$$E[x] = \int_{-\infty}^{+\infty} x \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) = \mu$$

$$Var[x] = \int_{-\infty}^{+\infty} (x-\mu)^2 \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Θέτω $u = \frac{x-\mu}{\sigma}$ άρα $du = \frac{dx}{\sigma}$ και τελικά έχω:

$$\int_{-\infty}^{+\infty} (\sigma u)^2 \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) = \sigma$$

Ερώτημα Δεύτερο: Ισχύει

$$(x-\mu)^\tau \Sigma^{-1} (x-\mu) = constant = c$$

Έχω

$$\Sigma^{-1} = \frac{1}{det(\Sigma)} Adj(\Sigma)$$

$$det(\Sigma) = \begin{vmatrix} 2 & 1 \\ 1 & 2 \end{vmatrix} = 2 \cdot 2 - 1 \cdot 1 = 4 - 1 = 3$$

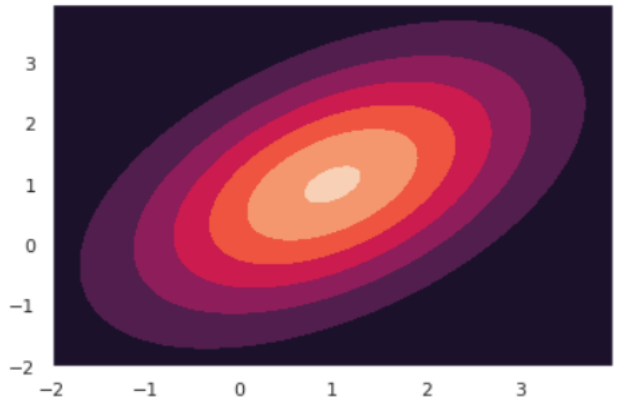
$$Adj(\Sigma) = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

Άρα

$$\Sigma^{-1} = \frac{1}{3} \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

$$[x_1 - 1 \; x_2 - 1] \frac{1}{3} \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} x_1 - 1 \\ x_2 - 1 \end{bmatrix} = 2{x_1}^2 + 2{x_2}^2 - 2x_1 - 2x_2 + 2 = 3c = c'$$

Για τη σχεδίαση των ισοσταθμικών καμπυλών χρησιμοποιώ τον ακόλουθο κώδικα σε python οπότε προκύπτει και η εξής εικόνα:




```python
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal
import numpy as np


mean1= np.array([1, 1])
cov1 = [[2, 1], [1, 2]]
x, y = np.mgrid[-2:4:0.01, -2:4:.01]
pos = np.dstack((x, y))
rv = multivariate_normal(mean1, cov1)
fig2 = plt.figure()
ax2 = fig2.add_subplot(111)
ax2.contourf(x, y, rv.pdf(pos),)
```

Ερώτημα Τρίτο: Έχω

$$p(x_n;\mu) = \frac{1}{|\Sigma|\sqrt{2\pi}} \exp\left(\frac{1}{2}(x_n-\mu)^\tau \Sigma^{-1}(x_n-\mu)\right)$$

Όπως αναφέρεται και στις σημειώσεις του μαθήματος, οι μετρήσεις είναι στατιστικά ανεξάρτητες συνεπώς για N στατιστικά ανεξάρτητες παρατηρήσεις έχω τον εξής λογάριθμος για την από κοινού συνάρτηση πιθανοφάνειας:

$$L(\mu) = ln \prod_{n=1}^{N} p(x_n;\mu) = -\frac{N}{2} ln((2\pi)^l|\Sigma|) \sum_{n=1}^{N} \left(-\frac{1}{2}(x_n-\mu)^\tau \Sigma^{-1}(x_n-\mu)\right)$$

Παραγωγίζω ως προς μ οπότε έχω:

$$\frac{\partial L(\mu)}{\partial \mu} = \sum_{n=1}^{N}((\Sigma^{-1}(x_n-\mu_{ML})$$

Ψάχνω να βρω μέγιστο συνεπώς θέλω μηδενισμό της παραγώγου οπότε προκύπτει:

$$\mu_{ML} = \frac{1}{N}\sum_{n=1}^{N} x_n$$

Άσκηση 3: Bayes Classifier

Ερώτημα Πρώτο: Για να ελαχιστοποιήσω την πιθανότητα σφάλγματος θα πρέπει

$$P(\omega_1)p(x_t|\omega_1) = P(\omega_2)P(x_t|\omega_2) \Rightarrow \frac{1}{2}e^{-|x_t|} = \frac{1}{2}e^{-|x_t-3|} \Rightarrow x_t = \frac{3}{2}$$

Ερώτημα Δεύτερο: Θέλω

$$\lambda_{12}P(x_t|\omega_1) = \lambda_{21}P(x_t|\omega_2) \Rightarrow$$

$$\frac{1}{2}e^{-|x_t|} = e^{-|x_t-3|} \Rightarrow -x_t = ln2 - (-x_t - 3) \Rightarrow x_t = \frac{3 - ln2}{2}$$

Ερώτημα Τρίτο: Έχω

$$g(x) = \frac{1}{2}(x^\tau\Sigma_2^{-1}x - x^\tau\Sigma_1^{-1}x) + \mu_1^\tau\Sigma_1^{-1}x - \mu_2^\tau\Sigma_2^{-1}x - \frac{1}{2}\mu_1^\tau\Sigma_1^{-1}\mu_1 + \frac{1}{2}\mu_2\tau\Sigma_2^{-1}\mu_2 + ln\frac{P(\omega_1)}{P(\omega_2)} + \frac{1}{2}ln\frac{|\Sigma_2|}{|\Sigma_1|} = 0$$

Περίπτωση i:

$$\Sigma^{-1} = \frac{1}{det(\Sigma)}Adj\Sigma = \frac{1}{2.25}\begin{vmatrix} 1.5 & 0 \\ 0 & 1.5 \end{vmatrix}$$

$$\mu_1^\tau\Sigma_1^{-1} = \frac{1}{2.25}\begin{bmatrix} 3 & -3 \end{bmatrix}$$

$$\mu_2^\tau\Sigma_2^{-1} = \frac{1}{2.25}\begin{bmatrix} -1.5 & 3 \end{bmatrix}$$

$$\mu_1^\tau\Sigma^{-1}\mu_1 = \frac{1}{2.25}12$$

$$\mu_2^\tau\Sigma^{-1}\mu_2 = \frac{1}{2.25}7.5$$

Τελικά προκύπτει:

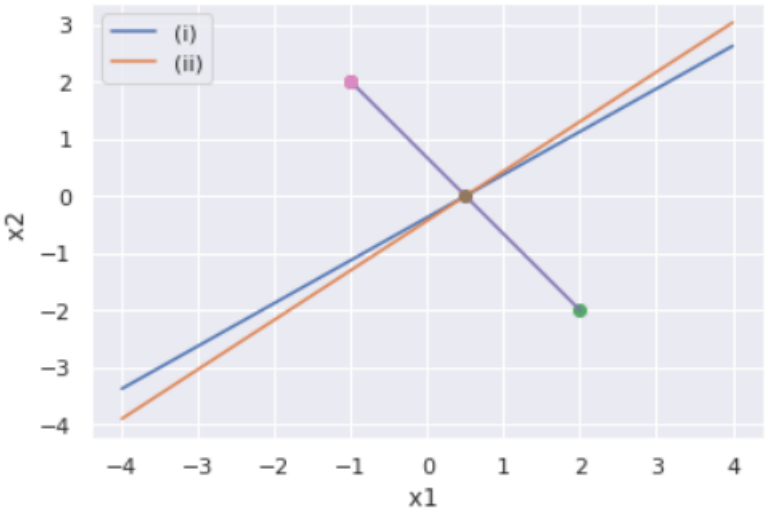
$$6x_1 - 8x_2 - 3 = 0$$

Περίπτωση ii:

$$\Sigma^{-1} = \begin{bmatrix} 0.75 & -0.25 \\ -0.25 & 0.75 \end{bmatrix}$$
$$\mu_1^\tau \Sigma^{-1} = \begin{bmatrix} 2 & -2 \end{bmatrix}$$
$$\mu_1^\tau \Sigma^{-1} = \begin{bmatrix} -1.25 & 1.75 \end{bmatrix}$$
$$\mu_1^\tau \Sigma^{-1} \mu_1 = 8$$
$$\mu_2^\tau \Sigma^{-1} \mu_2 = 4.75$$

Τελικά προκύπτει:

$$26x_1 - 30x_2 - 13 = 0$$



Ερώτημα Τέταρτο: Για $x = [4, 3]^\tau$ έχω:

Περίπτωση i:

$$g(x) = 6 \cdot 4 - 8 \cdot 3 - 3 = -3 < 0$$

Περίπτωση ii:

$$g(x) = 26 \cdot 4 - 30 \cdot 3 - 13 = 1 > 0$$

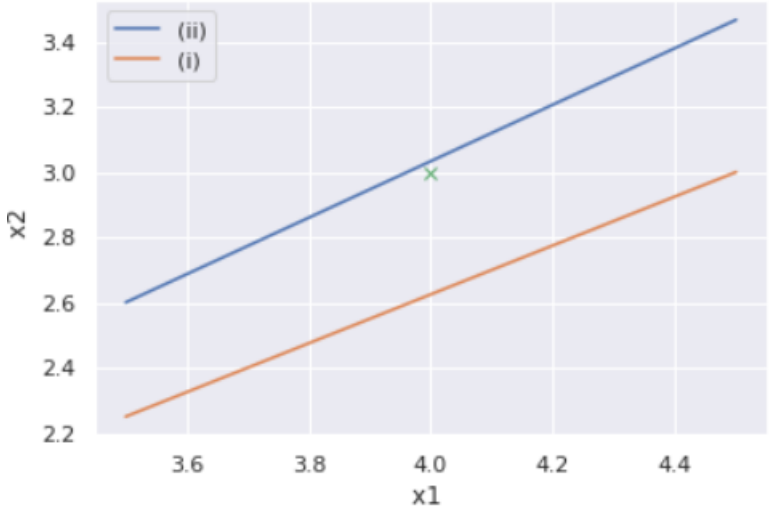
Προκυπτει επίσης και για τις δύο περιπτώσεις

$$g(\mu_1) > 0$$
$$g(\mu_2) < 0$$

.

Συνεπώς το πρότυπο x για την περίπτωση i ταξινομείται στην κλάση ω_2 και για την περίπτωση i ταξινομείται στην κλάση ω_1.

Γεωμετρικά βλέπω πως στην περίπτωση i το x βρίσκεται πάνω από την ευθεία του x_1x_1 επιπέδου, ενώ στην περίπτωση ii κάτω.




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

x1 = np . linspace ( -4 , 4 , 100)
x2_i = 6/8* x1 -3/8
x2_ii = 26/30* x1 -13/30

plt . plot ( x1 , x2_i , label = '(i)')
plt . plot ( x1 , x2_ii , label = '(ii)')
plt . plot ([2] , [ -2] , 'o')
plt . plot ([ -1] , [2] , 'o')
plt . plot ([2 , -1] , [ -2 , 2])
plt . plot (0.5 , 0 , 'o')
plt . xlabel ( 'x1 ')
plt . ylabel ( 'x2 ')
plt . plot ([ -1] , [2] , 'o')
plt . legend ()
plt . show ()
x1 = np . linspace (3.5 , 4.5 , 100)
x2_i = 6/8* x1 -3/8
x2_ii = 26/30* x1 -13/30

plt . plot ( x1 , x2_ii , label = '(ii)')
plt . plot ( x1 , x2_i , label = '(i)')
plt . plot ([4] , [3] , 'x')
```

```
26 plt . xlabel ( 'x1 ')
27 plt . ylabel ( 'x2 ')
28 plt . legend ()
29 plt . show ()
```

Άσκηση 4: Perceptron-Multilayer Perceptron

Ερώτημα Πρώτο: Για το δοθέν νευρωνικό δίκτυο με ένα κρυφό στρώμα έχω:

$$u_1 = w_1 x_1 + w_3 x_2$$

$$u_2 = w_2 x_1 + w_4 x_2$$

Και εξόδους για τους νευρώνες κρυφού στρώματος:

$$y_1 = c(w_1 x_1 + w_3 x_2)$$

$$y_2 c(w_2 x_1 + w_4 x_2$$

Είσοδος στο στρώμα εξόδου:

$$y = c w_5 (w_1 x_1 + w_3 x_2) + c w_6 (w_2 x_1 + w_4 x_2)$$

Η τελική έξοδος:

$$Y = c^2 w_5 (w_1 x_1 + w_3 x_2) + c^2 w_6 (w_2 x_1 + w_4 x_2) =$$

$$c^2 (w_5 w_1 + w_6 w_2) x_1 + c^2 (w_5 w_3 + w_6 w_4) x_2$$

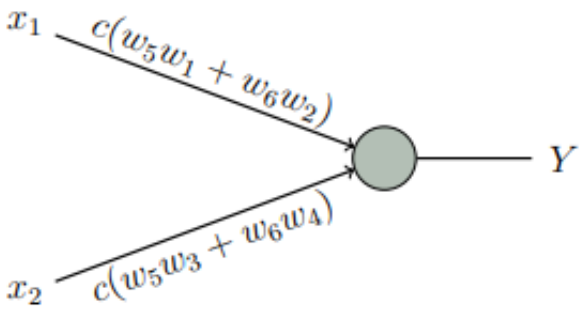
Προκύπτει ότι το νευρωνικό δίκτυο είναι ισοδύναμο με νευρωνικό δίκτυο χωρίς κρυφό στρώμα οπότε θα έχει βάρη:

$$w_1' = c(w_5 w_1 + w_6 w_2)$$

$$w_2' = c(w_5 w_3 + w_6 w_4)$$

Και η συνάρτηση ενεργοποίησης είναι:

$$f(u) = cu$$



Ερώτημα Δεύτερο: Εύκολα παρατηρούμε πως στην έξοδο ενός MLP πάντα θα υπάρχει γραμμικός όρος συναρτήσει των x_1, x_2 με μορφή $Y = f_1(w, c)x_1 + f_2(w, c)x_2$ γεγονός που οφείλεται στις γραμμικές συναρτήσεις ενεργοποίησης που βρίσκονται στην έξοδο. Συνεπώς, ένα multilayer Perceptron μπορεί πάντα να αντικατασταθεί με ισοδύναμο νευρωνικό δίκτυο βαρών $w_1 = f_1(w, c)$ και $w_2 = f_2(w, c)$.

Ερώτημα Τρίτο: Η είσοδος στο νευρώνα εξόδου για το νευρωνικό δίκτυο με τη σιγμοειδή συνάρτηση ενεργοποίησης κρυφού στρώματος είναι:

$$u\frac{w_5}{1 + w^{w_1x_1+w_3x_2}} + \frac{w_6}{1 + e^{w_2x_1+w_4x_2}}$$

Για το αποτέλεσμα της XOR θέλω μηδενική έξοδο για τα ζεύγη τιμών $(0, 0), (1, 1)$ και μονάδα για τα ζεύγη $(1, 0), (0, 1)$. Συνεπώς θέλω:

$(0, 0)$**:**

$$\frac{w_5}{2} + \frac{w_6}{2} < 0$$

$(1, 0)$ **:**

$$\frac{w_5}{1 + e^{-w_1}} + \frac{w_6}{1 + e^{-w_2}} > 0$$

$(0, 1)$**:**

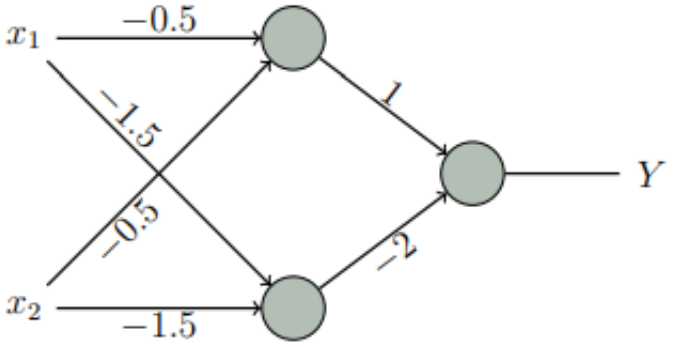
$$\frac{w_5}{1 - e^{-w_3}} + \frac{-w_6}{1 + e^{-w_4}} > 0$$

$(1, 1)$**:**

$$\frac{w_5}{1 + e^{-(w_1+w_3)}} + \frac{w_6}{e^{-(w_2+w_4)}} < 0$$

Επιλεγω $w_6 = -2$ και $w_5 = 1$ οπότε έχω

$$(w_1, w_2, w_3, w_4, w_5, w_6) = (-0.5, -0.25, -0.5, -1.5, 1, -2)$$



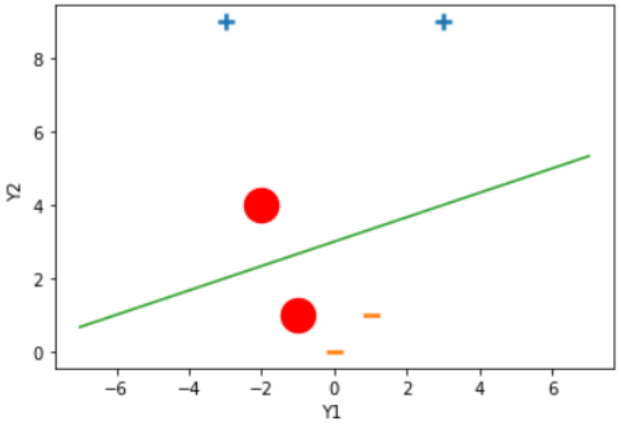
Άσκηση 5: Support Vector Machines-Kernels

Ερώτημα Πρώτο: Για $\Phi(u) = (u, u^2)$

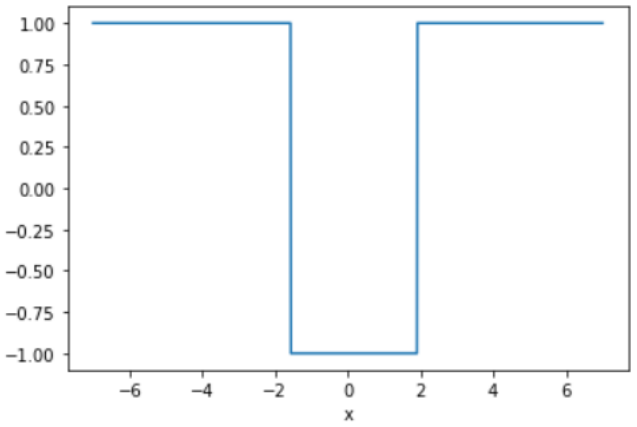
$$k(X_1, X_2) = \Phi(X_1)^T \Phi(Q_2) = X_1 X_1' + (X_1 X_1')^2$$

Ερώτημα Δεύτερο: Το επίπεδο μέγιστου περιθωρίου θα είναι η μεσοκάθετος των σημείων $(-2, 4)(-1, 1)$. Τα σημεία αυτά θα χρησιμοποιηθούν και για τα διανύσματα υποστήριξης. Το ζητούμενο είναι μία ευθεία κάθετη στην ευθεία που ορίζεται από τα προηγούμενα δύο σημεία, ανήκει στο επίπεδο $Y_1 Y_2$ και διέρχεται από το σημείο $(-1.5, 2.5)$. Τελικά προκύπτει η ευθεία $Y_1 - 3Y_2 + 9 = 0$, οπότε $w_1 = 1$, $w_2 = 3$, $c = 9$.

Ερώτημα Τρίτο:



Ερωτημα Τέταρτο: Για το κατώφλι απόφασης χρήσει του Φ^{τ} μελετώ το πρόσημο της $g^*(x) = -3x^2 + x + 9$. Παρατηρώ ότι με το νέο κατώφλι απόφασης οι κλάσεις διαχωρίζονται.



Ερώτημα Πέμπτο: Σύμφωνα με το δυικό πρόβλημα η βέλτιστη διαχωριστική επιφάνεια είνα:

$$g^*(x) = \sum_{n=1}^{|} SVa_n y_n k(x, u_n) + b = 4(a_1 - a_2)x^2 + (-2a_1 + a_2)x$$

Γνωρίζω ήδη τη βέλτιστη διαχωριστική ευθεία οπότε οι πλλαπλασιαστές Lagrange πρέπει να είναι τέτοιοι ώστε η παραπάνω συνάρτηση να ταυτίζεται με εκείνη που βρήκα σε προηγούμενο ερώτημα. Προκύπτει λοιπόν λύνοντας το σύστημα:

$$a_1 = 2$$

$$a_2 = 5$$

$$b = 9$$

Ερώτημα Έκτο: Το $(5, 25)$ στο επίπεδο X_1X_2 απέχει αρκετά από την επιφάνεια απόφασης οπότε τα διανύσματα υποστήριξης θα μείνουν ως έχουν. Κατά συνέπεια μένουν ίδια το περιθώριο και το επίπεδο διαχωρισμού.

Ερώτημα Έβδομο: Αν τα διανύσματα μ, ν προέρχονται από τη χρήση του μετασχηματισμού ϕ_∞ στα σημεια x, y τότε έχω για τον πυρήνα μετασχηματισμού.

$$k(\mu, \nu) = e^{\frac{-(x-y)^2}{2}}$$

Ερώτημα Έβδομο: Οχι ιδιαίτερα, εφόσον κατά τη λύση του προβλήματος τετραγωνικού προγραμματισμού δεν εμφανίζεται ποτέ μόνη της η συνάρτησηϕ_∞, παρά μόνο με τη μορφή του γινομένου $\phi_\infty^\tau \phi_\infty = k(x, y)$. Μάλιστα η συγκεκριμένη συνάρτηση πυρήνα είναι η Γκαουσιανή RBF $e^{\frac{-(x-y)^2}{2\sigma^2}}$

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

Y = np . linspace ( -7 , 7 , 1000)
x1_plus = [ -3 , -2 , 3]
x2_plus = [ x **2 for x in x1_plus ]
x1_minus = [ -1 , 0 , 1]
x2_minus = [ x **2 for x in x1_minus ]
plt . plot ( x1_plus , x2_plus , '+', mew =2.5 , ms =10)
plt . plot ( x1_minus , x2_minus , '_', mew =2.5 , ms =10)
plt . plot ( -2 , 4 , 'ro ', mew =1.5 , ms =20)
plt . plot ( -1 , 1 , 'ro ', mew =1.5 , ms =20)
plt . plot (Y , Y /3+3)

plt . xlabel ( 'Y1 ')
plt . ylabel ( 'Y2 ')
plt . show ()

X = np . linspace ( -7 , 7 , 1000)
classes = -np . sign (X -3* X **2+9)
plt . xlabel ( 'x')
plt . plot (X , classes )
plt . show ()
```

Άσκηση 6: Decision Trees

Ερώτημα Πρώτο: Έχω:

$$ig(t) = E(t) - E(a)$$

$$E(t) = \sum_{i \in \text{t.labels}} -p_i log_2 p_i$$

$$E(\alpha) = \sum_{v \in \alpha.\text{values}} \frac{|\alpha = v|}{|\alpha|} E(\alpha = v)$$

$$E(\alpha = v) = \sum_{i \in \text{a.labels}} -p_i log_2 p_i$$

Ισχύει:

$$\sum_V P(V) = 1$$

Οπότε:

$$
\begin{aligned}
ig(t) = E(t) - E(\alpha) &= -\sum_{X \in \text{t.labels}} P(X) log_2 P(X) + \sum_V P(V) \sum_X P(X|V) log_2 P(X|V) \\
&= -\sum_V \sum_X P(X,V) log_2 P(X) + \sum_V P(V) \sum_X P(X|V) log_2 P(X|V) \\
&= \sum_V \sum_X P(X,V)(log_2 P(X|V) - log_2 P(X)) \\
&= \sum_V \sum_X P(X,V) \cdot log_2 \frac{P(X|V)}{P(X)} \\
&= -\sum_V \sum_X P(X,V) \cdot log_2 \frac{P(X)}{P(X|V)} \\
-ig(t) = \sum_V \sum_X P(X,V) \cdot log_2 \frac{P(X)}{P(X|V)} &\leq log_2 \sum_V \sum_X \frac{P(X,V)P(X)}{P(X|V)} \Rightarrow \\
-ig(t) \leq log_2 \sum_V \sum_X \frac{P(X|V)P(V)P(X)}{P(X|V)} &= log_2 \sum_V \sum_X P(X)P(V) \\
&= log_2 \sum_V P(V) \sum_X P(X) \leq log_2(1) = 0
\end{aligned}
$$

Για κάθε κόμβο δέντρου έχω $ig(t) \geq 0$

Από την ανισότητα του Jensen $\sum_i p_i log(x_i) \leq log_2\left(\sum_i p_i x_i\right)$ που χρησιμοποίησα η ισότητα ισχύει όταν όλοι οι x_i είνια ίσοι, οπότε συνεπάγεται ότι $ig(t) = 0$ όταν στον κόμβο του χαρακτηριστικού η κατανομή των ετικετών μένει όμοια με την κατανομή στον κόμβο-ρίζα

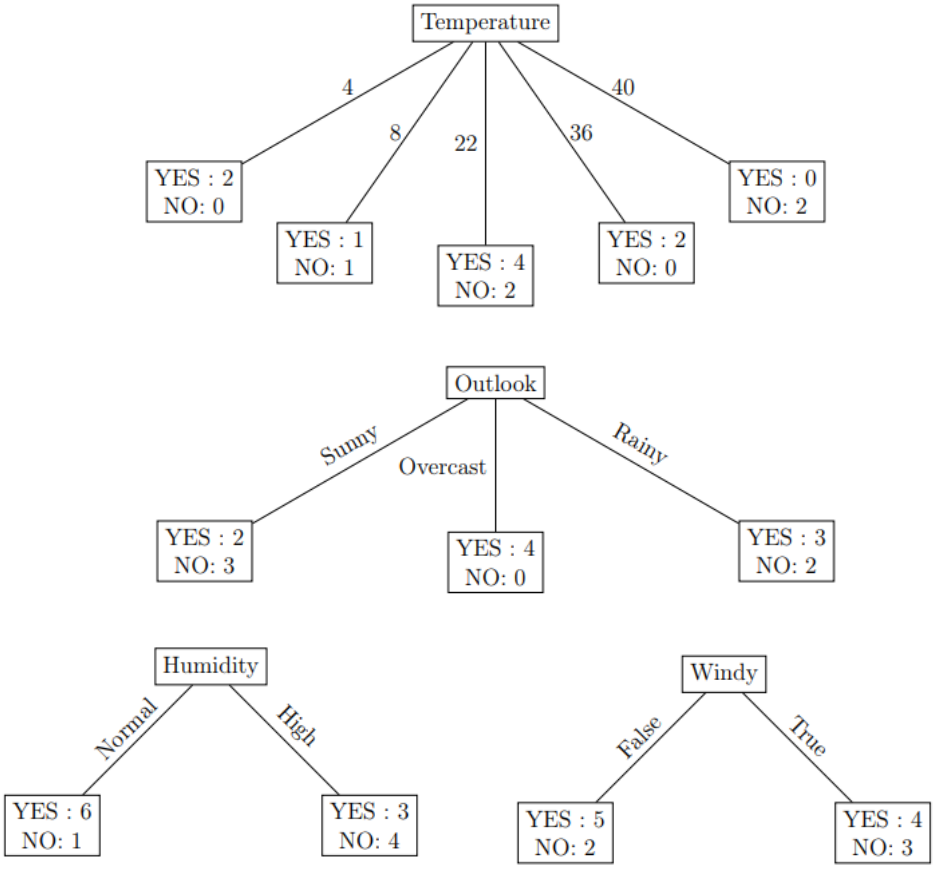
Κατά την κατασκευή του δέντρου με τον αλγόριθμο ID3 για έκαστο κόμβο έχω δύο περιπτώσεις: $i)$ $E(t_j) = 0$, οπότε όλα τα δεδομένα του υποσυνόλου που αντιστοιχεί σε αυτόν τον κόμβο έχουν την ίδια ετικέτα, και άρα ο αλγόριθμος θα κάνει τον αλγόριθμο t_j φύλλο και δεν θα υφίσταται κέρδος πληροφορίας γι'αυτόν, και $ii) E(t_j) > 0$, οπότε συνυπάρχουν δεδομένα με πολλές ετικέτες και ο $ID3$ θα ξαναεκτελεστεί για το νέο υποσύνολο.

Στη γενική περίπτωση που δεν υπάρχουν δεδομένα στο σύνολο που έχουν ίδιες τιμές σε όλα τα χαρακτηριστικά αλλά διαφορετική ετικέτα, τότε θα επιλεχθεί κάποιο χαρακτηριστικό που θα ξεχωρίσει περαιτέρω τις ετικέτες στο υποσύνολο και θα προσφέρει το μέγιστο κέρδος πληροφορίας, οπότε θα οδηγηθώ σε ελαχιστοποίηση της εντροπίας

οπότε $ig(t) > 0$. Αν ωστόσο υπάρχουν έστω δύο πρότυπα με ίδια χαρακτηριστικά αλλά διαφορετική ετικέτα, τότε υπάρχει περίπτωση να βρίσκονται αυτά μόνο 15 τα πρότυπα στο παρόν υποσύνολο. Έτσι, όποιο χαρακτηριστικό απόφασης και να επιλεχθεί η αναλογία των ετικετών θα παραμείνει ίδια, και άρα πάλι δεν θα υπάρξει κέρδος πληροφορίας, οπότε θα είναι $ig(t) = 0$ αφού αυτό θα είναι και το μέγιστο κέρδος πληροφορίας από οποιοδήποτε χαρακτηριστικό.

Ερώτημα Δεύτερο:

Θεωρώ το temperature κατηγορικό οπότε εκτελώ τους παρακάτω υπολογισμούς:



Για χάριν συντομίας παραλείπονται οι πράξεις οπότε έχω:

$$gini(Root) = 0.46$$

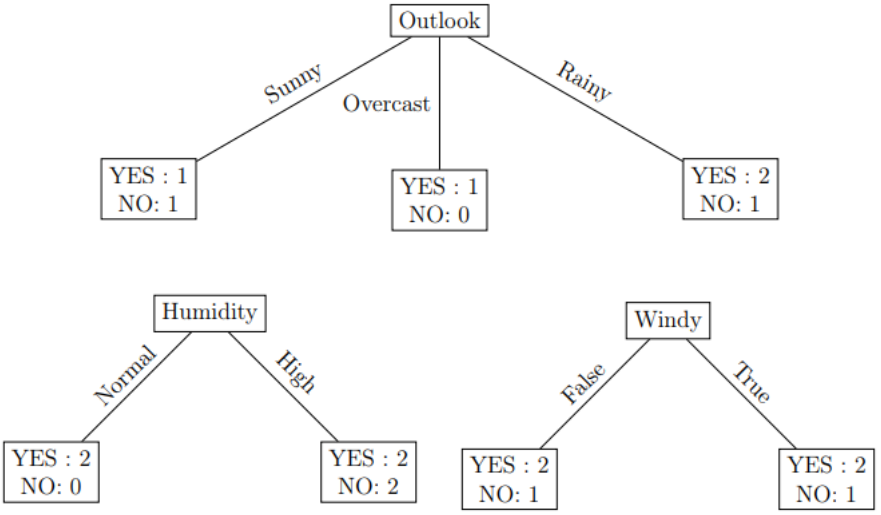
$$gini(Temperature) = \frac{1}{7}gini(4C)+\frac{1}{7}gini(8C)+\frac{3}{7}gini(22C)+\frac{1}{7}gini(36C)+\frac{1}{7}gini(40C) = 0.262$$

$$gini(Outlook) = \frac{5}{14}gini(Sunny) + \frac{2}{7}gini(Overcast) + \frac{5}{14}gini(Rainy) == 0.343$$

$$gini(Humidity) = \frac{1}{2}gini(Normal) + \frac{1}{2}gini(High) = 0.367$$

$$gini(Windy) = \frac{1}{2}gini(False) + \frac{1}{2}gini(True) = 0.449$$

Μέγιστο κέρδος πληροφορίας $ig(Temperature) = gini(Root) - gini(Temperature) = 0.398 - 0.262 = 0.136$ έχω επιλέγοντας το Temperature που έχει τον μικρότερο δείκτη gini. Τα υποσύνολα με θερμοκρασία ίση με 4, 36, και 40 έχουν έκαστο ίδια ετικέτα για όλα τους τα στοιχεία, οπότε σύμφωνα με τον αλγόριθμο CART μετατρέπονται σε φύλλα με το όνομα της ετικέτας. Για το υποσύνολο με θερμοκρασία ίση με 8 διαχωρίζω τα δείγματα ανάλογα την τιμή του χαρακτηριστικού Windy για να καταλήξω σε φύλλα και για το υποσύνολο με Τεμπερατυρε= 22 θα πρέπει να εκτελεστεί ξανά ο αλγόριθμος (στο υποσύνολο της ¨νέας ρίζας' τώρα έξω 4 στοιχεία με ετικέτα YES και 2 με ετικέτα NO):



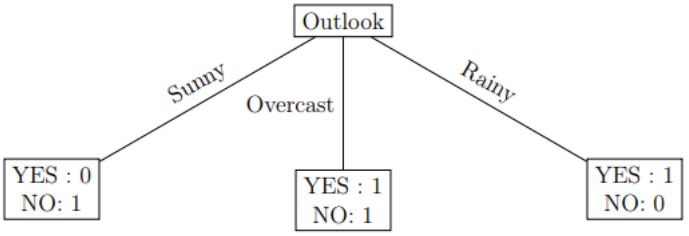
$$gini(Root) = 0.444$$

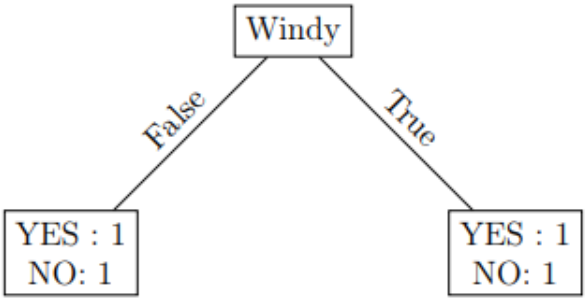
$$gini(Outlook) = 0.389$$

$$gini(Humidity) = 0.333$$

$$gini(Windy) = 0.444$$

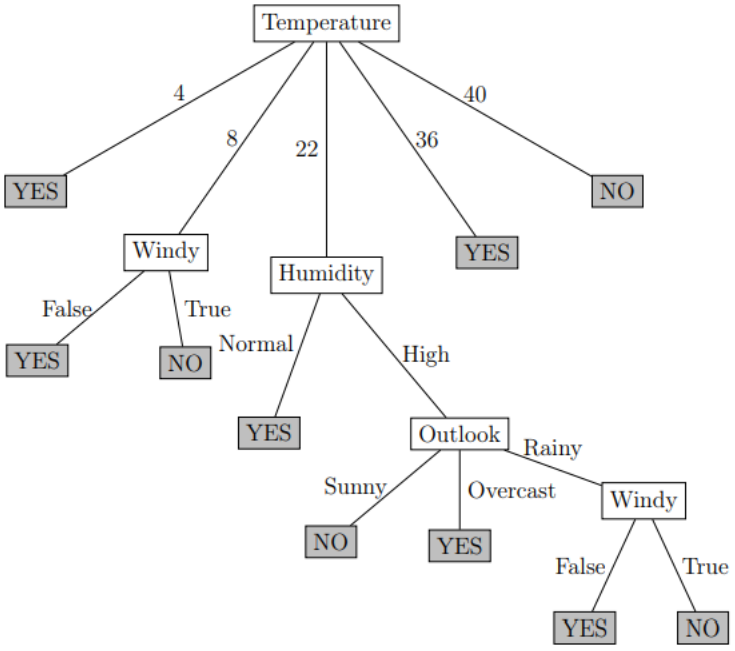
Επιλέγω το humidity αφού προσφέρει το μεγαλύτερο κέρδος πληροφορίας, και δημιουργείται ένα φύλλο για τα δεδομένα με $Temperature = 22$ και $Humidity = Normal$. Για τα δεδομένα που απομένουν έχω:





$$gini(Root) = 0.5$$
$$gini(Outlook) = 0.25$$
$$gini(Windy) = 0.5$$

Επιλέγεται το Outlook για όσα σημεία δεν έχουν διαχωρίσει το *windy* οπότε



Τώρα θα αντιμετωπίσω το χαρακτηριστικό Temperature ως αριθμητικό. Αυτό θα το κάνω ταξινομώντας τις τιμές της θερμοκρασίας στο σύνολο δεδομένων και μετά επιλέγοντας από τις υποψήφιες τιμές για όριο απόφασης αυτήν που θα μου προσφέρει το μεγαλύτερο κέρδος πληροφορίας. Οι υποψήφιες τιμές θα είναι τα μέσα κάθε δύο διαδοχικών τιμών στο σύνολο δεδομένων, δηλαδή στη περίπτωσή μας που έχω τις τιμές $Temperature$ (4,8,22,36,40) θα είναι (6,15,29,38). Υπολογίζω για καθεμία από αυτές τις τιμές T_0 το $gini(T_0)$

και επιλέγω σαν όριο απόφασης αυτό με το μικρότερο gini καθώς μου δίνει τη μεγαλύτερη πληροφορία . Πρόκειται για το $T_0 = 38$ που έχει $gini(6) = 0.322$. Τα $gini$ των υπολοίπων χαρακτηριστικών και της ρίζας είναι όπως είχαν υπολογιστεί πριν:

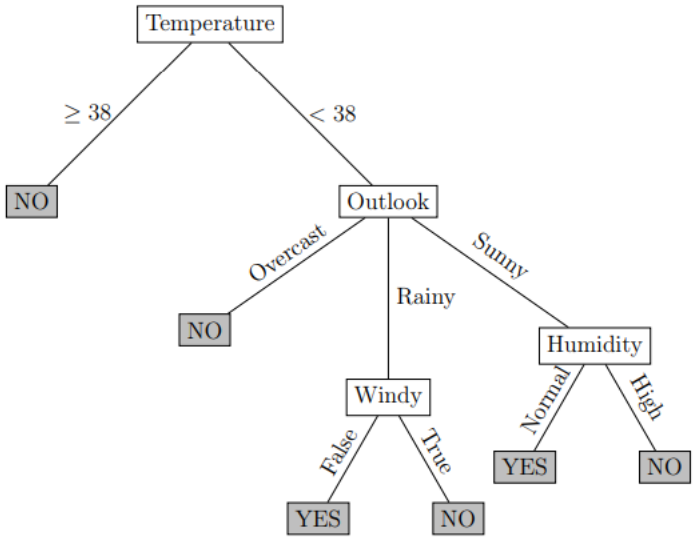
$$gini(Root) = 0.46$$

$$gini(Outlook) = 0.357$$

$$gini(Humidity) = 0.367$$

$$gini(Windy) = 0.449$$

Αρχικά παίρνω το χαραλτηριστικό θερμοκρασίας με όριο απόφασης 38 και θα συνεχίσω στο υποσύνολο με $temperatute < 38$ αν τα στοιχεία y εχουν αρνητική ετικέτα. Τελικά:



Παρόλο που φαίνεται να παίρνω περισσότερη πληροφορία αντιμετωπίζοντας τη θερμοκρασία κατηγορικά, προτιμώ το δέντρο όπου το $Temperature$ αντιμετωπίζεται ως αριθμητικό χαρακτηριστικό διότι θεωρώντας τη θερμοκρασία αριθμητικό δεδομένο έχω τη δυνατότητα να ταξινομήσω οποιδήποτε πρότυπο με οποιαδήποτε τιμή στο χαρακτηριστικό της θερμοκρασίας. Ειδάλλως μπορώ να ταξινομήσω μόνο όσα πρότυπα που στην τιμή της θερμοκρασίας έχουν τιμή που υπάρχει και στο σύνολο εκπαίδευσης, οπότε μπορεί να οδηγηθώ σε overfitting.