Αναγνώριση Προτύπων και Μηχανική Μάθηση

Εργασία Χειμερινού Εξαμήνου 2024 - 2025

Ομάδα 8

Δεϊρμεντζόγλου Ιωάννης
10015
deirmentz@ece.auth.gr

Οικονόμου Χρήστος
10268
cnoikonom@ece.auth.gr

ΠΕΡΙΕΧΟΜΕΝΑ ΠΑΡΟΥΣΙΑΣΗΣ

A

ΜΕΡΟΣ Α

Ταξινομητής Μέγιστης Πιθανοφάνειας

B

ΜΕΡΟΣ Β

Bayesian Ταξινομητής

C

ΜΕΡΟΣ Γ

Ταξινομητές Δέντρου Απόφασης και Τυχαίου Δάσους

D

ΜΕΡΟΣ Δ

Ανάπτυξη Αλγορίθμου Ταξινόμησης

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

Στο πρώτο μέρος της εργασίας, ζητείται η υλοποίηση ενός ταξινομητή, στον οποίο η άγνωστη παράμετρος θ εκτιμάται μέσω της **μεθόδου Μέγιστης Πιθανοφάνειας (Maximum Likelihood)**.

Για τον σκοπό αυτό, δημιουργείται η κλάση **ML_Classifier**, ώστε να αξιολογήσει αν ο δείκτης από την ανάλυση των μοντέλων είναι αξιόπιστος για την εκτίμηση του επιπέδου στρες χρηστών ενός βιντεοπαιχνιδιού. Ο ταξινομητής έχει σχεδιαστεί ώστε να ταξινομεί δεδομένα σε δύο κλάσεις:

Χωρίς Στρες → Κλάση ω1
Με Στρες → Κλάση ω2

Η πυκνότητα πιθανότητας περιγράφεται από την **κατανομή Cauchy** σύμφωνα με τον τύπο:

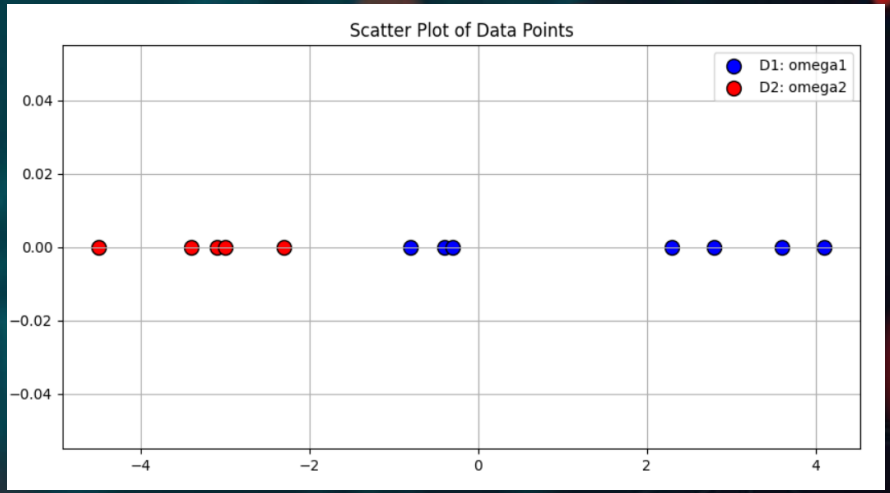
$$p(x|\theta) = \frac{1}{\pi} \cdot \frac{1}{1+(x-\theta)^2}$$

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

Για τους παίκτες που **δεν** ένιωσαν **στρες** και ανήκουν στην κλάση ω1 δίνονται οι εξής τιμές του δείκτη x : **D1 = [2.8,−0.4,−0.8, 2.3,−0.3, 3.6, 4.1]**

Για τους παίκτες που ένιωσαν **στρες** και ανήκουν στην κλάση ω2 δίνονται οι εξής τιμές του δείκτη x: **D2 = [−4.5,−3.4,−3.1,−3.0,−2.3]**

Για την εκτίμηση του θ υπολογίζεται η τιμή του $\widehat{\theta}$ που **μεγιστοποιεί την p(D|$\widehat{\theta}$).**



Παρατηρείται από τα δείγματα που δίνονται από την εκφώνηση είναι ότι οι δύο κλάσεις είναι γραμμικά διαχωρίσιμες. Αυτό σημαίνει ότι μπορεί να βρεθεί μια ευθεία γραμμή (σε μία διάσταση) ή ένα υπερεπίπεδο (σε περισσότερες διαστάσεις) που διαχωρίζει πλήρως τις δύο κατηγορίες **ω1** και **ω2**, χωρίς επικάλυψη στα δεδομένα.

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

Δομή και Λειτουργία του ML_Classifier

Α) Αρχικοποίηση

Η κλάση δέχεται ως είσοδο έναν πίνακα με υποψήφιες τιμές για την παράμετρο θ (thetas), ενώ αρχικοποιούνται οι a-priori πιθανότητες για τις κλάσεις ω1 και ω2 (P_w1 και P_w2 αντίστοιχα) και οι βέλτιστες εκτιμημένες τιμές του θ για κάθε κλάση (theta1 και theta2).

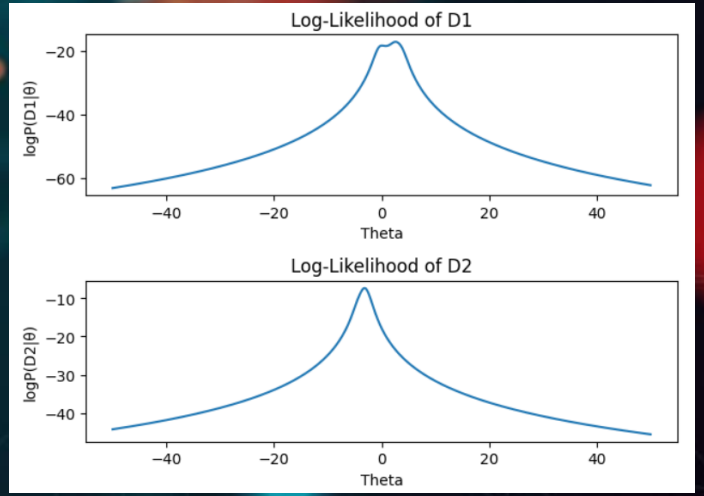
```
class ML_Classifier:
    def __init__(self, thetas):
        """Initialize the classifier with candidate values for theta."""
        self.thetas = thetas
        self.theta1 = None  # Estimated parameter for class omega1
        self.theta2 = None  # Estimated parameter for class omega2
        self.P_w1 = None  # Prior probability for class omega1
        self.P_w2 = None  # Prior probability for class omega2
```

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

Β) Βοηθητικές Συναρτήσεις

Υπολογισμός Πυκνότητας Πιθανότητας p(x|θ):
Η συνάρτηση **likelihood(self, x, theta)** υπολογίζει την υπό συνθήκη πιθανότητα για δεδομένη τιμή x και παράμετρο θ. Η πιθανότητα υπολογίζεται μέσω της **κατανομής Cauchy**, όπως αναφέθηκε προηγουμένως:

$$p(x|\theta) = \frac{1}{\pi} \cdot \frac{1}{1 + (x - \theta)^2}$$



Υπολογισμός Log-Likelihood:

Η συνάρτηση **log_likelihood(self, D, theta)** υπολογίζει τη λογαριθμική πιθανότητα για ένα σύνολο δεδομένων και μια συγκεκριμένη τιμή . Το άθροισμα των λογαρίθμων των πυκνοτήτων πιθανότητας υπολογίζεται για όλα τα δείγματα στο σύνολο δεδομένων και επιστρέφεται ως η συνολική log-likelihood.

ΜΕΡΟΣ Α – ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

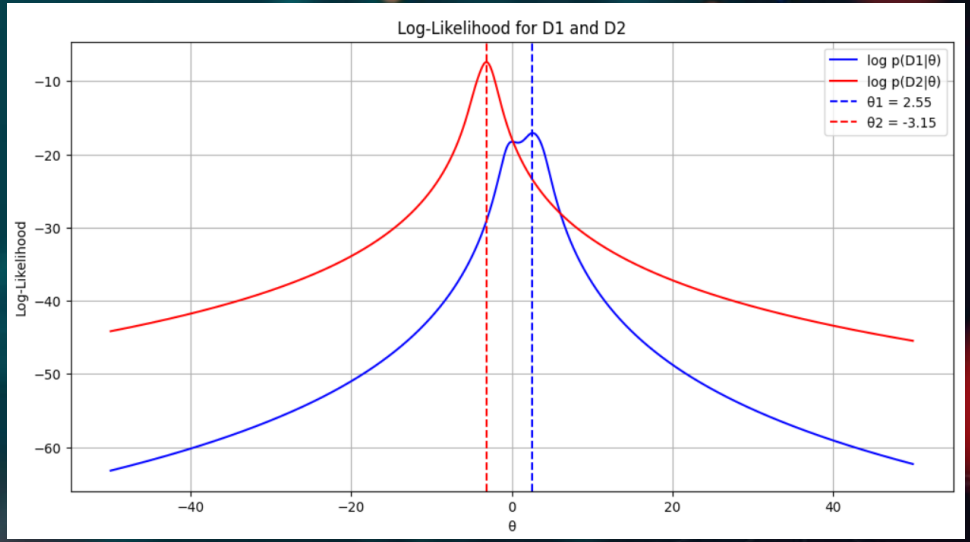
Γ) Εκπαίδευση (fit)

Η μέθοδος **fit(self, D1, D2)** υλοποιεί τη διαδικασία εκτίμησης των παραμέτρων για τις 2 κλάσεις:

Υπολογίζονται οι a priori πιθανότητες και τα ποσοστά των δειγμάτων στις 2 κλάσεις.

Υπολογίζονται οι log-likelihoods για όλες τις υποψήφιες τιμές.

Η τιμή για κάθε κλάση που μεγιστοποιεί την log likelihood (πιο πιθανή τιμή για να περιγράψει τα δεδομένα της κάθε κλάσης) εκτιμάται και αποθηκεύεται.



- Για την κλάση ω1, η log-likelihood μεγιστοποιείται για $\hat{\theta}1=2.55$, που αποτελεί την καλύτερη εκτίμηση για την περιγραφή των δεδομένων της κλάσης με στρες. Αντίστοιχα, για την ω2, η μεγιστη log-likelihood βρίσκεται στη $\hat{\theta}2=-3.15$.

❖ **Συμπερασμα**: Η σημαντική απόσταση μεταξύ $\hat{\theta}1$ και $\hat{\theta}2$ δείχνει ότι οι 2 κλάσεις μπορούν να διαχωριστούν με βάση τον δείκτη x → αξιόπιστος για την ταξινόμηση των χρηστών σε κατάσταση στρες ή μη-στρες.

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

Δ) Πρόβλεψη (predict)

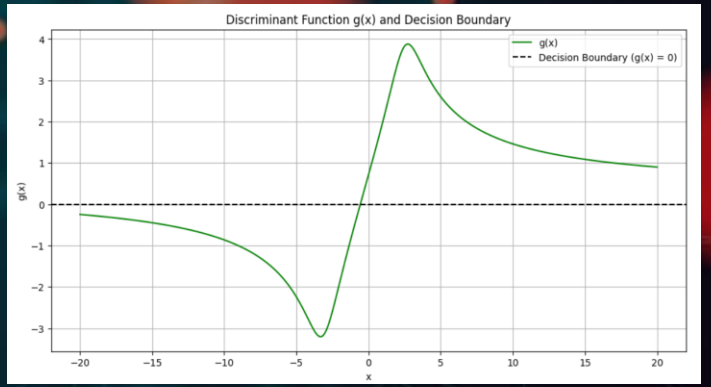
Συνάρτηση Διάκρισης g(x)

Η g(x), που υλοποιείται από τη συνάρτηση **g(self, x),** υπολογίζει τη συνάρτηση διάκρισης, η οποία συγκρίνει τις πιθανότητες των δύο κλάσεων για ένα συγκεκριμένο δείγμα . Η συνάρτηση διάκρισης υπολογίζεται ως:

$$g(x) = \log P(x|\hat{\theta}_1) - \log P(x|\hat{\theta}_2) + \log P(\omega_1) - \log P(\omega_2)$$

Ο κανόνας απόφασης στηρίζεται στη σύγκριση των τιμών logp(x|θ1) και logp(x|θ2) για οποιαδήποτε δεδομένη τιμή x:

Αν **logp(x|θ1) > logp(x|θ2)**, τότε το x ανήκει στην κλάση **ω1**
Αν **logp(x|θ1) < logp(x|θ2)**, τότε το x ανήκει στην κλάση **ω2**



Πρόβλεψη (predict)

Η συνάρτηση **predict(self, X)** ταξινομεί ένα σύνολο δεδομένων X σε μία από τις δύο κλάσεις, χρησιμοποιώντας τη συνάρτηση διάκρισης. Κάθε δείγμα x αντιστοιχίζεται στην κλάση ω1 αν g(x) >0, διαφορετικά αντιστοιχίζεται στην κλάση ω2.

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

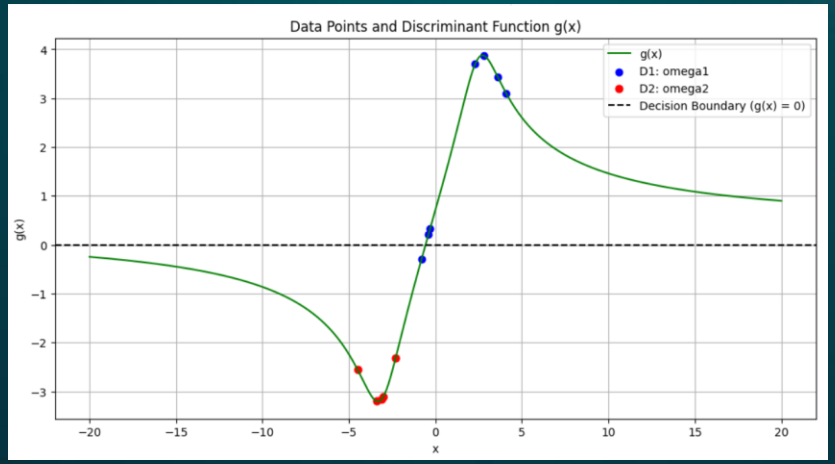
ΑΠΟΤΕΛΕΣΜΑΤΑ ΤΑΞΙΝΟΜΗΣΗΣ

Ταξινόμηση Συνόλου Τιμών D1				Ταξινόμηση Συνόλου Τιμών D2			
x	g(x)	Prediction	Actual Class	x	g(x)	Prediction	Actual Class
2.80	3.87	ω1	ω1	-4.50	-2.56	ω2	ω2
-0.40	0.21	ω1	ω1	-3.40	-3.20	ω2	ω2
-0.80	-0.29	ω2	ω1	-3.10	-3.16	ω2	ω2
2.30	3.70	ω1	ω1	-3.00	-3.10	ω2	ω2
-0.30	0.44	ω1	ω1	-2.30	-2.32	ω2	ω2
3.60	3.44	ω1	ω1				
4.10	3.10	ω1	ω1				

ΜΕΡΟΣ Α - ΤΑΞΙΝΟΜΗΤΗΣ ΜΕΓΙΣΤΗΣ ΠΙΘΑΝΟΦΑΝΕΙΑΣ

ΑΠΕΙΚΟΝΙΣΗ ΟΡΙΟΥ ΑΠΟΦΑΣΗΣ ΚΑΙ ΤΑΞΙΝΟΜΗΣΗΣ

Η διακριτική συνάρτηση g(x) είναι θετική για τις περιοχές που ανήκουν στην κλάση ω_1 και αρνητική για τις περιοχές που ανήκουν στην ω_2. Το όριο απόφασης (**g(x) = 0)** διαχωρίζει τις δύο κλάσεις. Δεδομένα στα δεξιά του ορίου ταξινομούνται ως ω_1, ενώ στα αριστερά ως ω_2.



Παρατήρηση: Η πλειονότητα των δεδομένων της κλάσης ω_1 (μπλε) βρίσκεται σε περιοχές όπου **g(x) > 0**, ενώ όλα σχεδόν τα δεδομένα της ω_2 (κόκκινο) βρίσκονται σε περιοχές όπου **g(x) < 0**. Ένα σημείο της κλάσης ω_1 ταξινομείται λανθασμένα, λόγω της εγγύτητάς του στις τιμές της ω_2.

Συμπέρασμα: Ο διαχωρισμός μεταξύ των κλάσεων είναι γραμμικός, με τη διακριτική συνάρτηση g(x) να αποδίδει αξιόπιστα. Παρά το μικρό σφάλμα ταξινόμησης, ο δείκτης x φαίνεται σχετικά κατάλληλος για τη διάκριση και την ανάλυση του στρες.

ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

Στο δεύτερο μέρος της εργασίας, ζητείται η υλοποίηση ενός ταξινομητή, στον οποίο η άγνωστη παράμετρος θ εκτιμάται μέσω της **μεθόδου Bayes**.

$$P(\omega_i|x) = \frac{p(x|\omega_i)P(\omega_i)}{p(x)}$$

Αντίστοιχα με το πρώτο μέρος, δημιουργείται η κλάση **BayesianClassifier**, ενώ ο ταξινομητής έχει σχεδιαστεί για ακόμη μια φορά έτσι, ώστε να ταξινομεί δεδομένα σε **δύο κλάσεις:**

Χωρίς Στρες → Κλάση ω1
Με Στρες → Κλάση ω2

ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

Δομή και Λειτουργία του BayesianClassifier

Α) Αρχικοποίηση

Η κλάση δέχεται ως είσοδο έναν πίνακα με υποψήφιες τιμές για την παράμετρο θ κατά την αρχικοποίηση (theta_values), καθώς και τις a-priori πιθανότητες για τις κλάσεις ω1 και ω2 (prior_w1 και prior_w2 αντίστοιχα). Αυτές οι τιμές αποθηκεύονται ως attributes και χρησιμοποιούνται για την εκτίμηση των παραμέτρων των δύο κλάσεων.

```
class BayesianClassifier:
    def __init__(self, theta_values, prior_w1, prior_w2):
        """
        Initializes the Bayesian Classifier with given theta values and class priors.

        Parameters:
        - theta_values: Array of candidate parameter values for theta.
        - prior_w1: Prior probability of class omega1.
        - prior_w2: Prior probability of class omega2.
        """
        self.theta_values = theta_values
        self.prior_w1 = prior_w1
        self.prior_w2 = prior_w2
```

ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

Β) Βοηθητικές Συναρτήσεις

- **Υπολογισμός Πυκνότητας Πιθανότητας (prior):** Η συνάρτηση **prior(self, theta)** υπολογίζει την πυκνότητα πιθανότητας για δεδομένο σύνολο τιμών της παραμέτρου θ, χρησιμοποιώντας την σχέση:

$$p(\theta) = \frac{1}{10\pi} \cdot \frac{1}{1 + (\theta/10)^2}$$

- **Υπολογισμός Πιθανότητας p(x|θ)**: Η συνάρτηση **likelihood(self, x, theta)** υπολογίζει την υπό συνθήκη πιθανότητα για ένα σύνολο δεδομένων x και έναν πίνακα με τιμές της παραμέτρου θ. Η πιθανότητα υπολογίζεται μέσω της **κατανομής Cauchy**:

$$p(x|\theta) = \frac{1}{\pi} \cdot \frac{1}{1 + (x - \theta)^2}$$

ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

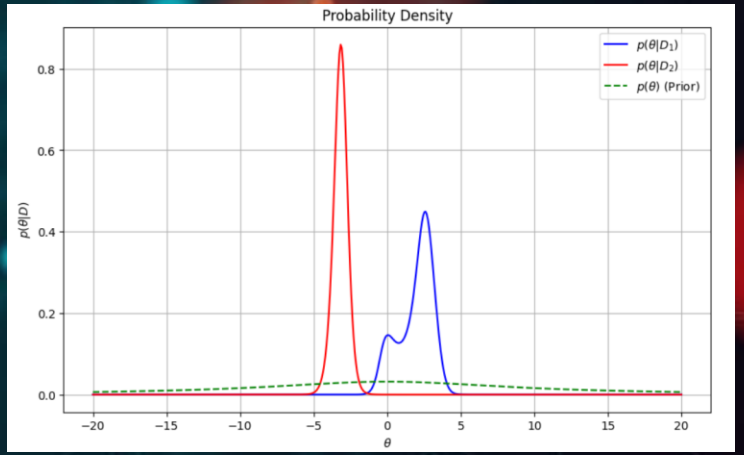
Γ) Εκπαίδευση (fit)

Η συνάρτηση **fit(self, D, theta)** υπολογίζει την εκ των υστέρων πιθανότητα p(θ|D) με βάση ένα σύνολο παρατηρήσεων D και υποψήφιες τιμές για την παράμετρο θ. Αρχικά, υπολογίζει την πιθανότητα p(D|θ) (likelihood) για κάθε τιμή του θ ως το γινόμενο των επιμέρους πιθανοτήτων για τα δεδομένα D:

$$p(D|\theta) = \prod_{i=1}^{N} p(x_n|\theta)$$

Στη συνέχεια, πολλαπλασιάζει την πιθανότητα αυτή με την εκ των προτέρων πιθανότητα p(θ) για να παραχθεί η μη κανονικοποιημένη εκ των υστέρων πιθανότητα. Για να κανονικοποιηθεί το αποτέλεσμα, υπολογίζει μια σταθερά κανονικοποίησης μέσω της αριθμητικής ολοκλήρωσης (με τη **μέθοδο του τραπεζίου**) και επιστρέφει την κανονικοποιημένη εκ των υστέρων πιθανότητα p(θ|D).

$$p(\theta|D) = \frac{p(D|\theta) \cdot p(\theta)}{\int p(D|\theta) \cdot p(\theta) d\theta}$$



ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

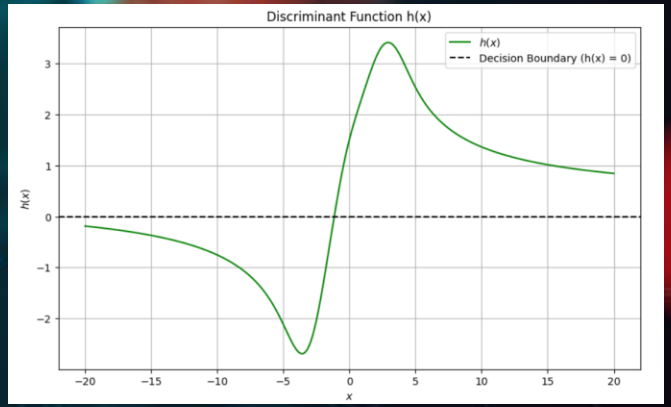
Δ) Πρόβλεψη (predict)

Η συνάρτηση **predict(self, x, theta_values, posterior_D1, posterior_D2)** υπολογίζει τη **διακριτική συνάρτηση h(x)**, η οποία χρησιμοποιείται για την ταξινόμηση ενός δεδομένου σημείου x και δίνεται από τη σχέση:

$$h(x) = logP(x|D_1) - logP(x|D_2) + logP(\omega 1) - logP(\omega 2)$$

Χρησιμοποιεί τις υποψήφιες τιμές της παραμέτρου θ, καθώς και τις εκ των υστέρων πιθανότητες p(θ|D1) και p(θ|D2) για τις δύο κλάσεις ω1 και ω2 αντίστοιχα. Για κάθε κλάση, υπολογίζει την πιθανότητα p(x|Di) ως ολοκλήρωση του γινομένου της συνάρτησης πιθανότητας p(x|θ) με την εκ των υστέρων πιθανότητα p(θ|Di). Στη συνέχεια, υπολογίζει τη διακριτική συνάρτηση h(x) ως τη διαφορά των λογαρίθμων αυτών των πιθανοτήτων, προσαρμοσμένη με τους λογαρίθμους των εκ των προτέρων πιθανοτήτων των δύο κλάσεων. Το πρόσημο της h(x) καθορίζει την ταξινόμηση του x, δηλαδή:

Αν **h(x) > 0**, τότε το x ανήκει στην κλάση **ω1**
Αν **h(x) < 0**, τότε το x ανήκει στην κλάση **ω2**



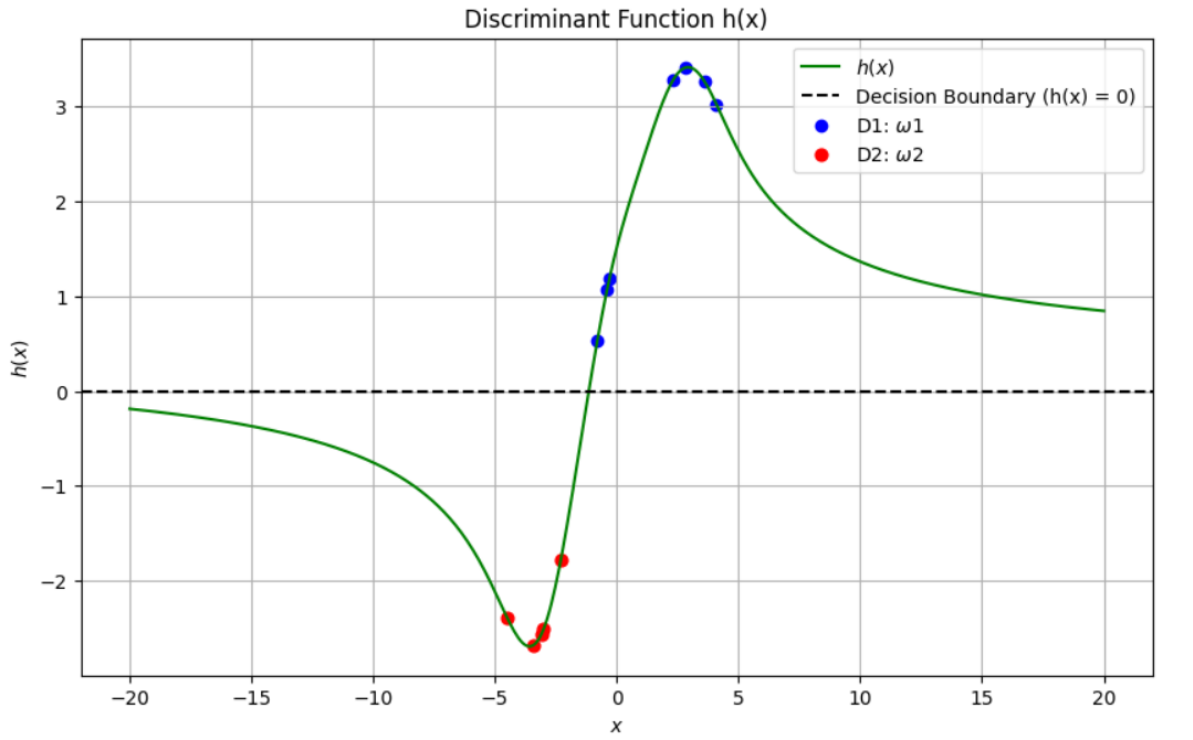
ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

ΑΠΟΤΕΛΕΣΜΑΤΑ ΤΑΞΙΝΟΜΗΣΗΣ

Ταξινόμηση Συνόλου Τιμών D1				Ταξινόμηση Συνόλου Τιμών D2			
x	h(x)	Prediction	Actual Class	x	h(x)	Prediction	Actual Class
2.80	3.41	ω1	ω1	-4.50	-2.39	ω2	ω2
-0.40	1.07	ω1	ω1	-3.40	-2.68	ω2	ω2
-0.80	0.53	ω1	ω1	-3.10	-2.57	ω2	ω2
2.30	3.27	ω1	ω1	-3.00	-2.51	ω2	ω2
-0.30	1.19	ω1	ω1	-2.30	-1.79	ω2	ω2
3.60	3.27	ω1	ω1				
4.10	3.02	ω1	ω1				

ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

ΑΠΟΤΕΛΕΣΜΑΤΑ ΤΑΞΙΝΟΜΗΣΗΣ



ΜΕΡΟΣ Β - BAYESIAN ΤΑΞΙΝΟΜΗΤΗΣ

ΤΕΛΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ

Ο ταξινομητής που υλοποιήθηκε με τη μέθοδο εκτίμησης κατά Bayes είναι πιο ακριβής και αποτελεσματικός από τον ταξινομητή μέγιστης πιθανοφάνειας. Χαρακτηριστικό στοιχείο αιτιολόγησης αποτελεί το γεγονός ότι ο Bayesian ταξινομητής κατάφερε να κατηγοριοποιήσει σωστά όλα τα σημεία των συνόλων D1 και D2, ενώ η μέθοδος μέγιστης πιθανοφάνειας παρουσίασε αστοχία στην κατηγοριοποίηση του σημείου **x = -0.80**.

Η μέθοδος **Μέγιστης Πιθανοφάνειας (ML)** και η **Bayesian Εκτίμηση (BE)** διαφέρουν σε 3 βασικά σημεία:

ΥΠΟΛΟΓΙΣΤΙΚΗ ΠΟΛΥΠΛΟΚΟΤΗΤΑ

Η ML είναι πιο απλή, καθώς απαιτεί μόνο διαφορικό λογισμό ή αναζήτηση μέσω gradient, ενώ η BE περιλαμβάνει πολύπλοκη και πολυδιάστατη ολοκλήρωση

ΕΡΜΗΝΕΥΣΙΜΟΤΗΤΑ

Η ML επιστρέφει 1 μόνο μοντέλο από το σύνολο που παρέχει ο σχεδιαστής, καθιστώντας τη λύση ευκολότερη στην ερμηνεία, ενώ η BE υπολογίζει έναν σταθμισμένο μέσο όρο μοντέλων, γεγονός που μπορεί να οδηγήσει σε μοντέλο που δεν υπήρχε στο αρχικό σύνολο.

ΑΠΟΔΟΣΗ

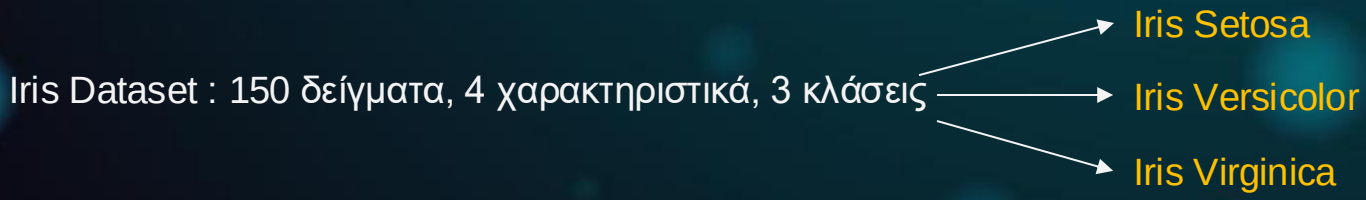
Η ML δεν λαμβάνει υπόψη την εκ των προτέρων κατανομή των παραμέτρων, ενώ η BE αξιοποιεί αυτές τις πληροφορίες, οδηγώντας σε καλύτερες λύσεις.

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Ζήτημα: Στα πλαίσια μιας έρευνας του τμήματος Γεωπονίας ζητείται η αναγνώριση τριών ειδών Ίριδας, που διαφέρουν στο μήκος και πλάτος των σεπάλων και πετάλων.

1. Στην 1η ενότητα, ζητείται η ανάπτυξη ενός Decision Tree Classifier. Ο αλγόριθμος πρέπει να εκπαιδευτεί στο 50% των δεδομένων και να αξιολογηθεί στο υπόλοιπο 50%. Ο ερευνητής καλείται να υπολογίσει το ποσοστό σωστής ταξινόμησης, να προσδιορίσει το βέλτιστο βάθος δέντρου και να απεικονίσει τα όρια απόφασης.

2. Στη 2η ενότητα, ζητείται η δημιουργία ενός Random Forest Classifier με 100 δέντρα, χρησιμοποιώντας τη μέθοδο Bootstrap. Το 50% των δεδομένων που χρησιμοποιήθηκαν για εκπαίδευση στην 1η θα αξιοποιηθεί για τη δημιουργία 100 νέων συνόλων εκπαίδευσης. Η αξιολόγηση θα γίνει στο ίδιο σύνολο που χρησιμοποιήθηκε στην 1η ενότητα.



ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Επεξεργασία και Προετοιμασία Δεδομένων Iris

1) Το σύνολο δεδομένων Iris φορτώνεται από τη βιβλιοθήκη sklearn.datasets.

2) Παρουσιάζονται ενδεικτικά χαρακτηριστικά για το dataset, όπως:

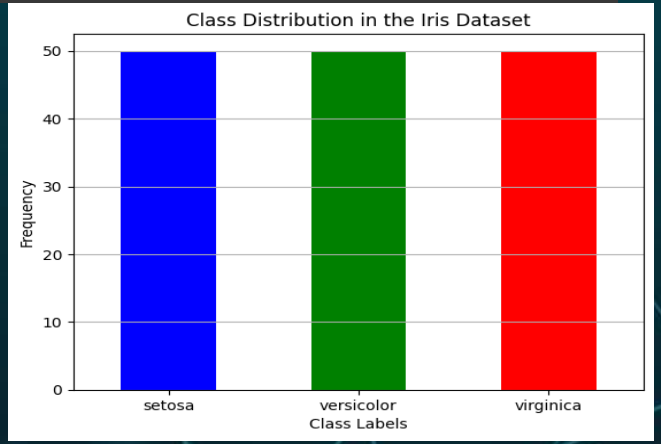
- Καποια τυχαία δεδομένα.
- Στατιστικές περιγραφές.
- Κατανομή των δειγμάτων στις τρεις κλάσεις (Iris setosa, Iris versicolor, Iris virginica).

```
%--- Iris Dataset Information ---%

Summary dataset Statistics:
    sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                 5.1               3.5                1.4               0.2
1                 4.9               3.0                1.4               0.2
2                 4.7               3.2                1.3               0.2
3                 4.6               3.1                1.5               0.2
4                 5.0               3.6                1.4               0.2

Dataset Satistics:
       sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
count         150.000000        150.000000         150.000000        150.000000
mean            5.843333          3.057333           3.758000          1.199333
std             0.828066          0.435866           1.765298          0.762238
min             4.300000          2.000000           1.000000          0.100000
25%             5.100000          2.800000           1.600000          0.300000
50%             5.800000          3.000000           4.350000          1.300000
75%             6.400000          3.300000           5.100000          1.800000
max             7.900000          4.400000           6.900000          2.500000
```

3) Δημιουργείται διάγραμμα για τη **συχνότητα των δειγμάτων ανά κλάση.**



4) Διαχωρισμός dataset και μείωση features

- Τα δεδομένα μειώνονται στα 2 πρώτα χαρακτηριστικά (μήκος και πλάτος σεπάλων).
- Χωρίζονται σε υποσύνολα εκπαίδευσης και ελέγχου (50%-50%) με τη χρήση της συνάρτησης train_test_split από sklearn, με τυχαία δειγματοληψία.

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

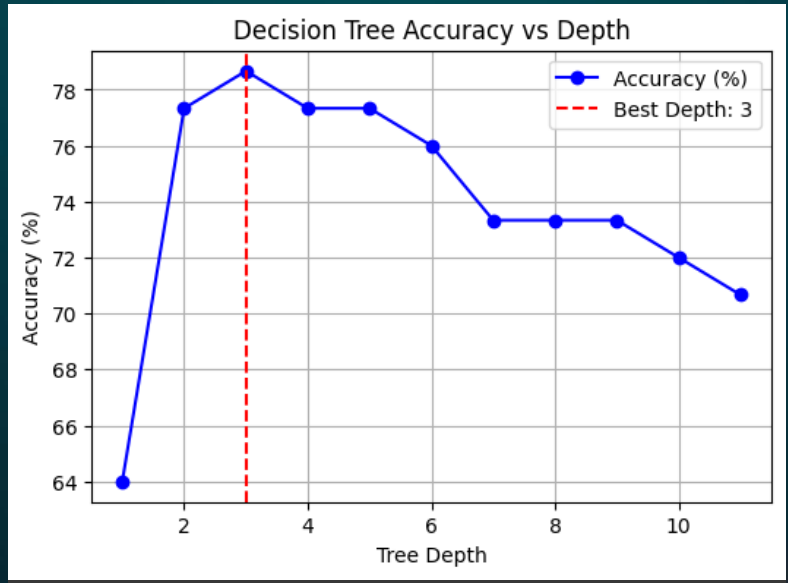
Ανάλυση και Αξιολόγηση Decision Tree Classifier

Λειτουργία της Συνάρτησης tuning_depth_DTC()

Η συνάρτηση εκπαιδεύει έναν **Decision Tree Classifier** για να προσδιορίσει το μέγιστο βάθος του δέντρου. Στη συνέχεια, δοκιμάζει διαφορετικά βάθη (από 1 έως το max), υπολογίζοντας την ακρίβεια για κάθε βάθος. Εντοπίζει το βέλτιστο βάθος (μεγιστοποιεί την ακρίβεια) και επιστρέφει τον αντίστοιχο ταξινομητή. Παράγεται γράφημα που δείχνει την ακρίβεια ανά βάθος, με το βέλτιστο βάθος να επισημαίνεται.

Αποτελέσματα: Η **tuning_depth_DTC** χρησιμοποιείται για τη διερεύνηση της απόδοσης του DTC με διαφορετικά βάθη. Στην συνέχεια, ο ταξινομητής χρησιμοποιείται για την πρόβλεψη των κλάσεων του συνόλου δοκιμής. Υπολογίζεται η ακρίβεια του μοντέλου και το classification report.

Σύγκριση διαφορετικών βαθών: Η ακρίβεια αυξάνεται μέχρι το **βάθος 3** και στη συνέχεια παραμένει σταθερή ή μειώνεται. Αυτό υποδεικνύει ότι μεγαλύτερα βάθη ενδέχεται να οδηγούν σε **υπερπροσαρμογή** του μοντέλου.




```
Most efficient depth of tree: 3
Accuracy at best depth: 78.67%
```

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Συμπεράσματα

Classification Report:

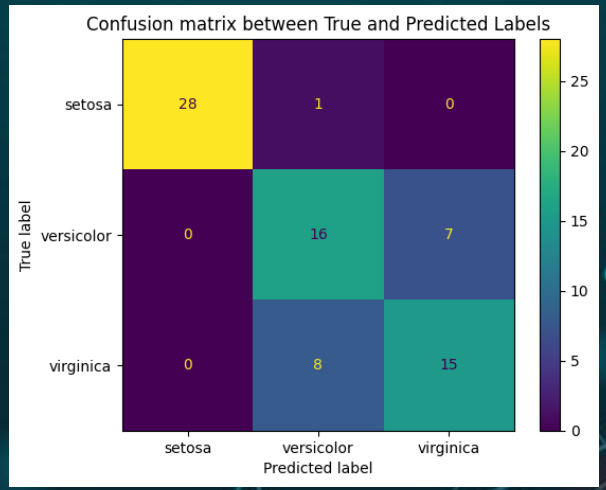
- **Iris setosa**: Πολύ υψηλή απόδοση με ακρίβεια 1.00 και recall 0.97.
- **Iris versicolor και Iris virginica**: Μέτρια απόδοση με ακρίβεια 0.64-0.68 και recall περίπου 0.65-0.70.
- **Συνολική απόδοση**: Η μέση ακρίβεια (weighted avg) είναι 0.79, καταδεικνύοντας καλή αλλά όχι απόλυτη ταξινομητική ικανότητα του μοντέλου.

Classification Report:

	precision	recall	f1-score	support
setosa	1.00	0.97	0.98	29
versicolor	0.64	0.70	0.67	23
virginica	0.68	0.65	0.67	23
accuracy			0.79	75
macro avg	0.77	0.77	0.77	75
weighted avg	0.79	0.79	0.79	75

Ο **Decision Tree Classifier (DTC)** παρουσιάζει εξαιρετική ακρίβεια για την κατηγορία **Setosa** (28/29 σωστές προβλέψεις), ενώ δυσκολεύεται να διακρίνει τις κατηγορίες **Versicolor** και **Virginica**, με 7 και 8 λάθη αντίστοιχα λόγω επικάλυψης χαρακτηριστικών.

Το μοντέλο είναι κατάλληλο για της Ίριδας, αλλά η επίδοση του επηρεάζεται από την επιλογή του βάθους. Η συνολικά μέτρια απόδοση σε versicolor και virginica δείχνει περιθώρια βελτίωσης.

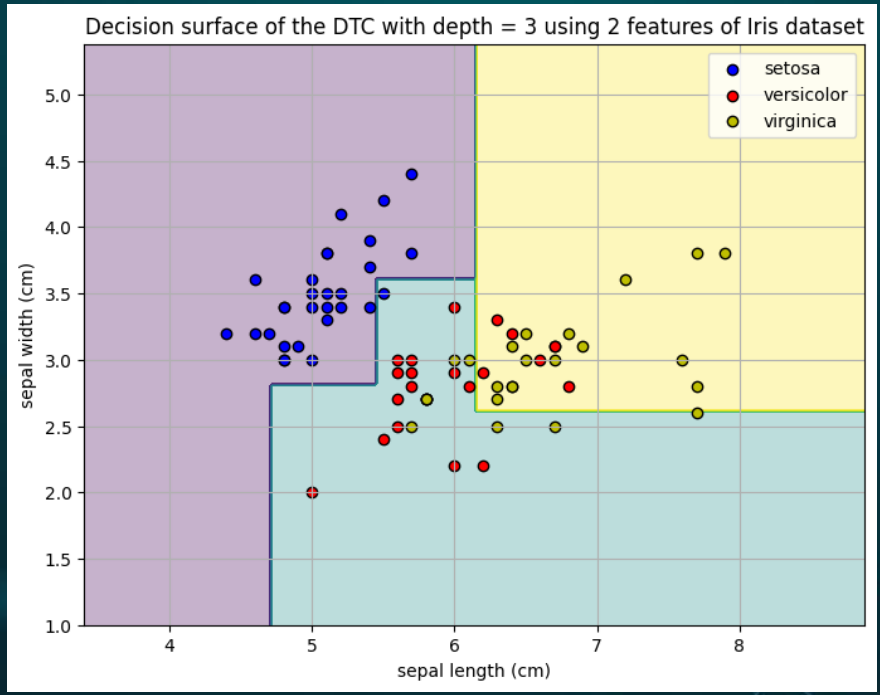


ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Απεικόνιση των ορίων απόφασης του DTC

- **Συνάρτηση plot_boundaries_DCT**: Δημιουργεί τα όρια απόφασης για τον καλύτερο ταξινομητή που προέκυψε από τη διαδικασία βελτιστοποίησης βάθους. Τα όρια αυτά ορίζουν τις περιοχές όπου το μοντέλο αποφασίζει για την ταξινόμηση κάθε κατηγορίας.

Η οπτικοποίηση των ορίων απόφασης παρέχει πληροφορία για την κατανόηση του τρόπου με τον οποίο το μοντέλο χωρίζει τον χώρο χαρακτηριστικών και αναδεικνύει πιθανές αδυναμίες του σε πιο περίπλοκα προβλήματα ταξινόμησης.



ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Ερμηνεία Διαγράμματος Ορίων Απόφασης DCT

Διαχωρισμός της κατηγορίας Setosa: Η κατηγορία Setosa (μπλε) είναι απολύτως διαχωρίσιμη από τις άλλες 2, γεγονός που δείχνει ότι το μοντέλο καταφέρνει να εντοπίσει αποτελεσματικά τα χαρακτηριστικά που την διαφοροποιούν από τις άλλες.

Επικάλυψη των κατηγοριών Versicolor και Virginica: Παρατηρείται σημαντική επικάλυψη μεταξύ των κατηγοριών Versicolor (κόκκινη) και Virginica (κίτρινη). Αυτό υποδεικνύει ότι οι 2 κατηγορίες έχουν παρόμοιες τιμές για τα χαρακτηριστικά που χρησιμοποιήθηκαν (μήκος και πλάτος σεπάλων), με αποτέλεσμα την αυξημένη πιθανότητα λανθασμένων ταξινομήσεων μεταξύ τους.

Απλοποιημένα Όρια Απόφασης: Το μοντέλο με μέγιστο βάθος 3 παράγει σχετικά απλοποιημένα όρια απόφασης, πράγμα που μειώνει την πιθανότητα υπερπροσαρμογής (overfitting), αλλά οδηγεί σε μέτρια απόδοση σε περιοχές όπου τα δεδομένα επικαλύπτονται.

Χαρακτηριστικά που χρησιμοποιήθηκαν: Το μήκος και το πλάτος των σεπάλων δεν επαρκούν για την αποτελεσματική διάκριση μεταξύ Versicolor και Virginica. Ίσως η χρήση περισσότερων χαρακτηριστικών να οδηγούσε σε καλύτερα αποτελέσματα.

Συμπερασματικά, το μοντέλο λειτουργεί καλά για την κατηγορία Setosa αλλά παρουσιάζει δυσκολίες στη διάκριση μεταξύ Versicolor και Virginica, γεγονός που υποδεικνύει την ανάγκη για βελτιώσεις στη μέθοδο ή την επιλογή χαρακτηριστικών.

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Ενότητα 2η : Κλάση myRandomForestClassifier

Συνάρτηση bootstrapMethod(): Υλοποιεί επαναδειγματοληψία bootstrap, δημιουργώντας τυχαία υποσύνολα δεδομένων βάσει της παραμέτρου gamma. Αυτά τα υποσύνολα χρησιμοποιούνται για την εκπαίδευση των δέντρων. Επιστρέφει υποσύνολα δεδομένων (features και labels).

`class myRandomForestClassifier`

Η κλάση αυτή είναι μια προσαρμοσμένη υλοποίηση ενός **ταξινομητή τυχαίου δάσους (Random Forest)**, η οποία περιλαμβάνει δύο συναρτήσεις, την **fit** για την εκπαίδευση και τη μέθοδο **predict** για την πρόβλεψη. Κατά την αρχικοποίηση, ορίζεται ο αριθμός των δέντρων, το ποσοστό των δεδομένων που θα χρησιμοποιείται σε κάθε επαναδειγματοληψία (bootstrap), ο ελάχιστος αριθμός δειγμάτων που απαιτούνται για να γίνει διαχωρισμός σε έναν κόμβο και το μέγιστο βάθος των DTC.

Η συνάρτηση **fit** εκπαιδεύει το μοντέλο δημιουργώντας διαδοχικά δέντρα απόφασης. Για κάθε δέντρο, εφαρμόζεται η συνάρτηση **bootstrap** και εκπαιδεύεται στα υποσύνολα αυτά με βάση τις προκαθορισμένες παραμέτρους και αποθηκεύεται στη λίστα forest. Με τον τρόπο αυτό, δημιουργείται ένα σύνολο DTC που έχουν εκπαιδευτεί σε διαφορετικά δείγματα δεδομένων.

Η συνάρτηση **predict** χρησιμοποιεί το εκπαιδευμένο δάσος για να προβλέψει τις ετικέτες για ένα νέο σύνολο δεδομένων. Κάθε DTC στο δάσος προβλέπει την ετικέτα για κάθε δείγμα. Οι προβλέψεις από όλα τα δέντρα συνδυάζονται μέσω ψηφοφορίας πλειοψηφίας για να καθοριστεί η τελική πρόβλεψη.

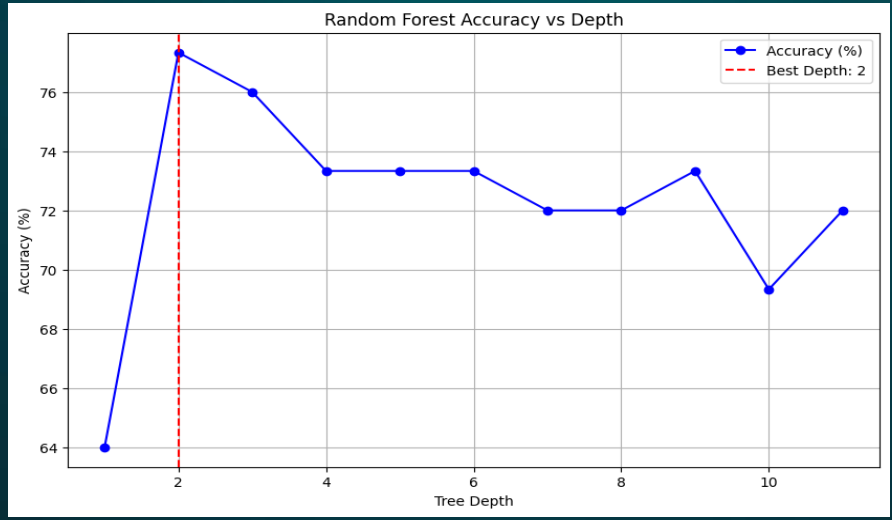
ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Ανάλυση και Βελτιστοποίηση Βάθους για RFC

Συνάρτηση tuning_depth_RFC: Ανάλυση της απόδοσης του myRandomForestClassifier για διαφορετικά βάθη DTC. Η διαδικασία ξεκινά με την εκπαίδευση διαδοχικών μοντέλων RF για διαφορετικές τιμές βάθους, από 1 μέχρι το καθορισμένο maxDepth και υπολογίζεται η ακρίβεια τους. Μόλις ολοκληρωθεί η εκπαίδευση για όλα τα βάθη, η συνάρτηση εντοπίζει το βέλτιστο βάθος του δάσους (εκείνο με την υψηλότερη ακρίβεια).

Παράμετροι εισόδου : σύνολα δεδομένων εκπαίδευσης και δοκιμής (**XTrain, XTest, yTrain, yTest**), το μέγιστο βάθος δέντρων (**maxDepth**) και το ποσοστό των χαρακτηριστικών που χρησιμοποιούνται κατά τη διαδικασία επαναδειγματοληψίας (**gamma**).

Επιστρέφει το **μοντέλο με το καλύτερο βάθος**, ένα **dictionary αποτελεσμάτων**, και τις **προβλέψεις** του καλύτερου μοντέλου.




```
Most efficient depth of tree: 2
Accuracy at best depth: 77.3333%
```

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

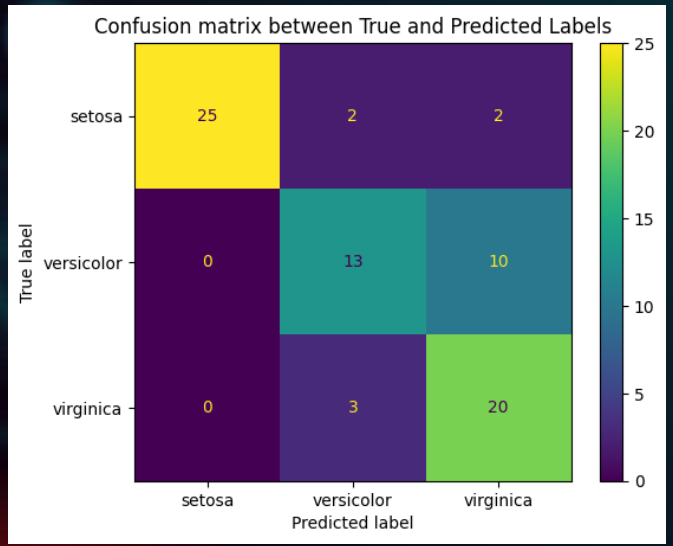
Συμπεράσματα

Setosa: Εξαιρετική απόδοση με απόλυτη ακρίβεια (**precision = 1.00**), ενώ το **recall= 0.86** (όλα τα δείγματα που προβλέφθηκαν ως Setosa είναι σωστά, αλλά χάθηκαν ορισμένα δείγματα πραγματικής Setosa).

Versicolor: Το **precision** είναι **0.72** (αρκετές προβλέψεις για Versicolor σωστές). Το **recall** είναι χαμηλότερο, στο **0.57** (αρκετά πραγματικά Versicolor λάθος ταξινόμηση). **f1-score = 0.63**, (μέτρια απόδοση).

Classification Report:

	precision	recall	f1-score	support
setosa	1.00	0.86	0.93	29
versicolor	0.72	0.57	0.63	23
virginica	0.62	0.87	0.73	23
accuracy			0.77	75
macro avg	0.78	0.77	0.76	75
weighted avg	0.80	0.77	0.78	75



Virginica: Το **precision** είναι **0.62**, ενώ το **recall** είναι υψηλότερο, στο **0.87**. Αυτό δείχνει ότι αν και πολλές προβλέψεις Virginica ήταν λανθασμένες, ο αλγόριθμος αναγνώρισε σωστά τα περισσότερα πραγματικά δείγματα αυτής της κατηγορίας. **f1-score =0.73**.

Confusion Matrix

- **Setosa** : σχεδόν τέλεια πρόβλεψη (25 σωστές προβλέψεις και μόνο 4 λάθη).
- **Versicolor**: 10 δείγματα ταξινομήθηκαν λανθασμένα ως Virginica.
- **Virginica**: 3 δείγματα ταξινομήθηκαν λανθασμένα ως Versicolor.

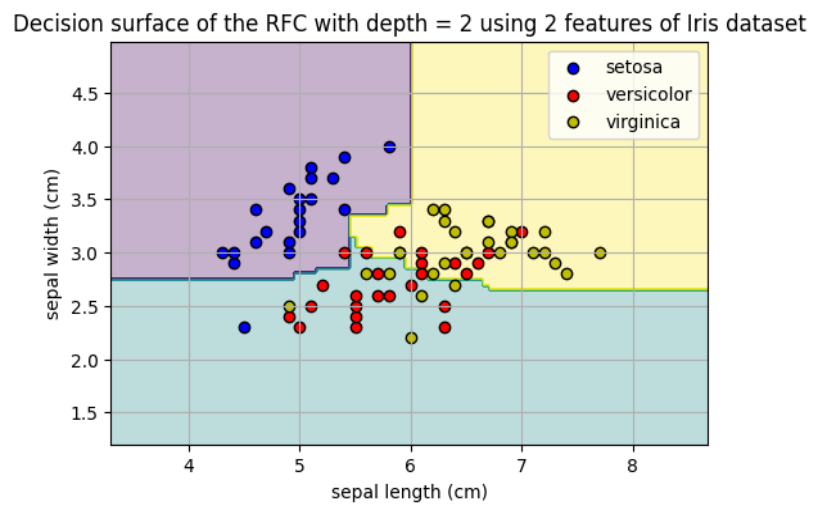
ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Απεικόνιση των ορίων απόφασης του ταξινομητή RF

Συνάρτηση plot_boundaries_RFC: Δημιουργεί το διάγραμμα των ορίων απόφασης ενός Random Forest Classifier (RFC).

- **Παράμετροι εισόδου**: μέγιστο βάθος του μοντέλου, προβλέψεις, δεδομένα εκπαίδευσης και το gamma.

Αρχικά, υπολογίζει τον αριθμό των κλάσεων. Στη συνέχεια, εκπαιδεύει το RFC με τις συγκεκριμένες παραμέτρους. Μετά, η συνάρτηση δημιουργεί ένα πλέγμα σημείων στον χώρο χαρακτηριστικών και προβλέπει την κλάση για κάθε σημείο, δημιουργώντας με αυτόν τον τρόπο το διάγραμμα των ορίων απόφασης.



Αποτελέσματα και σχολιασμός

Ο διαχωρισμός για την κατηγορία Setosa είναι ξεκάθαρος και καλά ορισμένος.

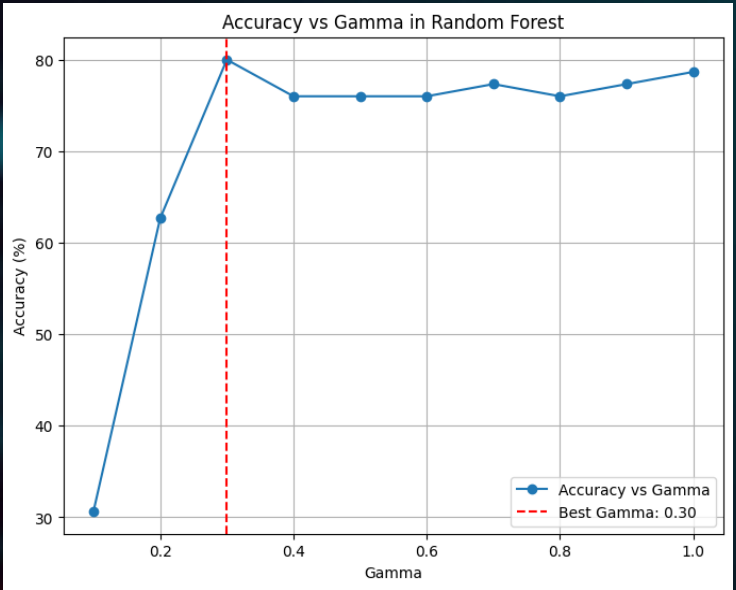
Οι κατηγορίες **Versicolor** και **Virginica** εμφανίζουν μεγαλύτερη επικάλυψη. Στο όριο μεταξύ αυτών των δύο κατηγοριών, διακρίνονται πιο σύνθετες γραμμές διαχωρισμού, που δείχνουν ότι το μοντέλο παλεύει να διαχωρίσει τα δεδομένα. Ωστόσο, ίσως υπάρχει και μεγαλύτερος κίνδυνος για εμφάνιση overfitting.

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Μελέτη της επιρροής του συντελεστή γ

Επιλέχτηκε ως παράδειγμα να δοκιμαστεί ο ταξινομητής Random Forest με:

- **Συγκεκριμένο βάθος**, το οποίο επιλέχθηκε να είναι το βέλτιστο βάθος που υπολογίστηκε στα παραπάνω ερωτήματα
- Σε ένας εύρος **από 10% μέχρι 100% με αυξητικό βήμα 10%** κατά επανάληψη



Αξιολογείται η απόδοση του αλγορίθμου **myRandomForestClassifier** για διαφορετικές τιμές της παραμέτρου γ, η οποία ελέγχει το ποσοστό του dataset που χρησιμοποιείται στη δημιουργία του κάθε δέντρου. Πιο συγκεκριμένα, ο αλγόριθμος βρίσκει την τιμή του γ που επιτυγχάνει την υψηλότερη ακρίβεια.

Η μέγιστη ακρίβεια επιτυγχάνεται όταν το γ είναι 0.3. Η ακρίβεια αυξάνεται αρχικά καθώς το γ μεγαλώνει, αλλά όταν το **γ = 0.3** και μετά, η απόδοση αρχίζει να σταθεροποιείται και παρουσιάζει μικρές διακυμάνσεις.

Φαίνεται ότι η υπερβολική ή πολύ μικρή δειγματοληψία μπορεί να επηρεάσει αρνητικά την ικανότητα γενίκευσης του ταξινομητή.

ΜΕΡΟΣ Γ: DECISION TREE CLASSIFIER ΚΑΙ RANDOM FOREST CLASSIFIER

Γενικό Συμπέρασμα για επιρροή γ και Bonus

Συμπέρασμα

Όταν το γ είναι πολύ μικρό, ο ταξινομητής δεν χρησιμοποιεί αρκετά δεδομένα για κάθε δέντρο, με αποτέλεσμα κακή απόδοση. Από την άλλη, για τιμές γ μεγαλύτερες του 0.30, η αύξηση της δειγματοληψίας δεν προσφέρει περαιτέρω βελτίωση, καθώς τα δέντρα αρχίζουν να μοιάζουν περισσότερο μεταξύ τους λόγω της επικάλυψης των δειγμάτων. Σε σύγκριση με τον απλό ταξινομητή Decision Tree της πρώτης ενότητας, το Random Forest παρουσιάζει καλύτερη απόδοση και μειώνει το φαινόμενο του overfitting μέσω της του bootstrap sampling.

Bonus

Στο notebook που παραδόθηκε επισυνάπτονται επίσης υλοποιήσεις και για τις δύο ενότητες του μέρους Γ υλοποιήσεις με έτοιμες συναρτήσεις όπως η **RandomForestClassifier()** και η **GridSearchCV** για το tuning των υπερπαραμέτρων με σκοπό τα αποτελέσματα που θα προκύψουν να συγκριθούν με τα αποτελέσματα των δικών μας implementations.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Στο τέταρτο μέρος της εργασίας, ζητείται η ανάπτυξη ενός αλγορίθμου ταξινόμησης με χρήση οποιασδήποτε μεθόδου της αρεσκείας μας. Η επιλογή της καταλληλότερης μεθόδου θα πραγματοποιηθεί έπειτα από δοκιμές διαφόρων ταξινομητών, χρησιμοποιώντας το σύνολο δεδομένων **datasetTV.csv** ως training set. Μετά την αξιολόγηση των μοντέλων, το καλύτερο από αυτά θα εφαρμοστεί στα δεδομένα του **datasetTest.csv**, τα οποία θα χρησιμοποιηθούν ως test set, με σκοπό την εξαγωγή ενός διανύσματος ετικετών.

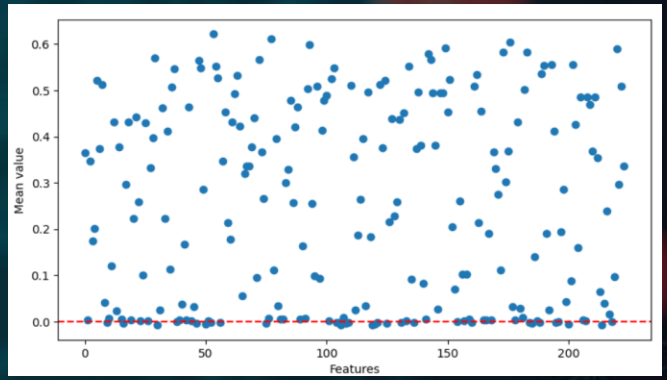
Προετοιμασία του training dataset για χρήση σε μοντέλο μηχανικής μάθησης

Αρχικά, το dataset μετατρέπεται σε numpy πίνακα και διαχωρίζονται τα **χαρακτηριστικά (features) X** από τις **ετικέτες (labels) y**, που αποτελούν την τελευταία στήλη του set. Οι ετικέτες κωδικοποιούνται σε αριθμητική μορφή (αν δεν είναι ήδη) και, στη συνέχεια, πραγματοποιείται διαχωρισμός σε **σύνολο εκπαίδευσης (training)** και **δοκιμής (testing)**, μέσω της συνάρτησης **train_test_split** της βιβλιοθήκης **sklearn**. Οι ετικέτες μετατρέπονται σε one-hot encoding για πολυκατηγορική ταξινόμηση. Τα δεδομένα και οι ετικέτες επαναδιαμορφώνονται και **κανονικοποιούνται** σε τιμές μεταξύ 0 και 1 για να διευκολυνθεί η εκπαίδευση του μοντέλου. Ο κώδικας διασφαλίζει ότι τα δεδομένα έχουν σωστή μορφή και είναι έτοιμα για εισαγωγή σε αλγόριθμο.

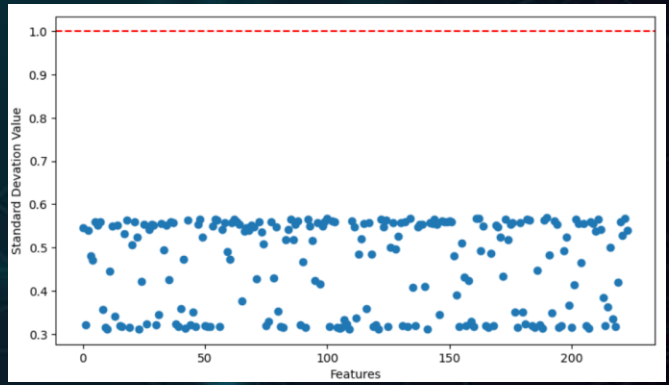
ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Απεικόνιση των Μέσων Τιμών και των Τυπικών Αποκλίσεων των Χαρακτηριστικών

Παρότι υπάρχει σημαντική ποικιλία στις μέσες τιμές των χαρακτηριστικών, η πλειονότητά τους συγκεντρώνεται κοντά στο μηδέν, υποδεικνύοντας ότι πολλά χαρακτηριστικά έχουν χαμηλές μέσες τιμές. Κάποια χαρακτηριστικά με υψηλότερες μέσες τιμές μπορεί να είναι πιο σημαντικά για το πρόβλημα ή να έχουν διαφορετική κλίμακα.



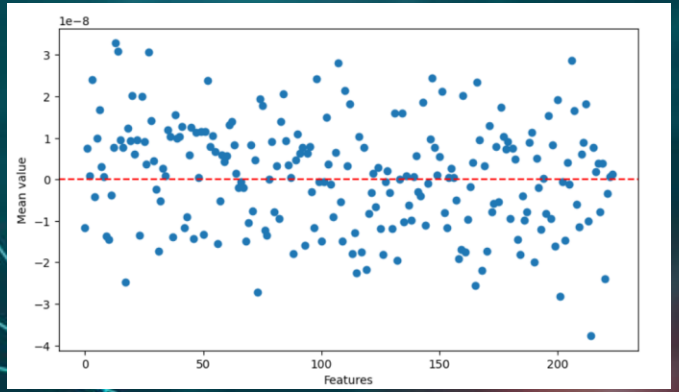
Παρατηρούμε πως τα χαρακτηριστικά του dataset έχουν αρκετά χαμηλές και διαιρεμένες τιμές τυπικής απόκλισης. Αυτό μπορεί να αποτελέσει πρόβλημα κατά την χρήση του για την αξιολόγηση των ταξινομητών, καθώς υπάρχει περίπτωση να αποδοθούν **διαφορετικά βάρη** στα χαρακτηριστικά, με αποτέλεσμα να υπάρχει κατευθυνόμενη αποδοτικότητα. Για τον λόγο αυτό, θα προβούμε σε **κανονικοποίηση των χαρακτηριστικών**.

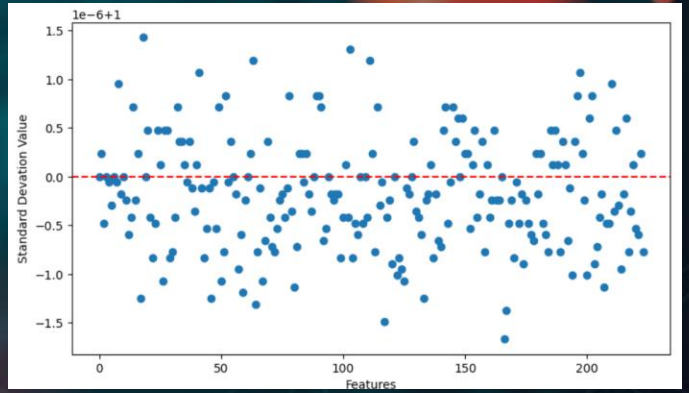


ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Κανονικοποίηση Χαρακτηριστικών (Feature Normalization)

Χρησιμοποιείται ο **StandardScaler**, ώστε να μετασχηματιστούν τα δεδομένα εκπαίδευσης και δοκιμής (XTrain και XTest), με αποτέλεσμα να έχουν **μηδενική μέση τιμή** και **τυπική απόκλιση 1**.





Πλέον, όπως φαίνεται από τα παραπάνω διαγράμματα, όλα τα χαρακτηριστικά έχουν μέση τιμή κοντά στο μηδέν και τυπική απόκλιση κοντά στη μονάδα. Συνεπώς, έχουμε ένα **normalized training dataset**, γεγονός που θα μας επιτρέψει να κάνουμε πιο "δίκαιες" αξιολογήσεις των μοντέλων που πρόκειται να μελετηθούν.

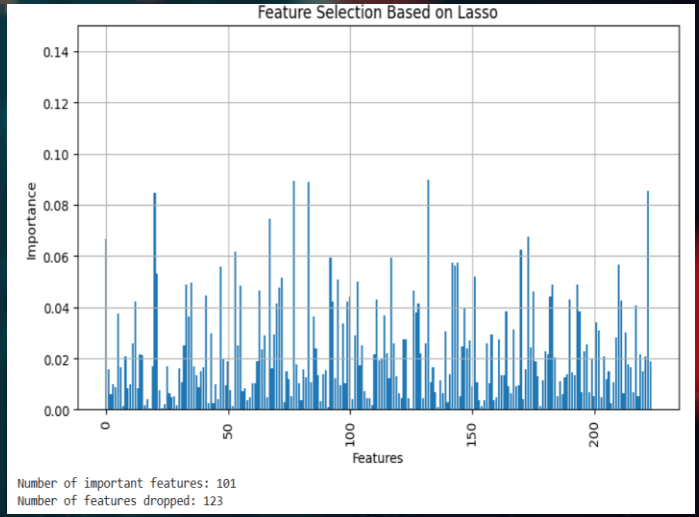
ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Βελτιστοποίηση Υπερπαραμέτρων - Επιλογή Χαρακτηριστικών

Μοντέλο Lasso

Βελτιστοποίηση υπερπαραμέτρων: Με τη χρήση του **GridSearchCV**, εκτελείται αναζήτηση πλέγματος (grid search) για να βρεθεί η καλύτερη τιμή της παραμέτρου α για το μοντέλο Lasso, χρησιμοποιώντας **5-fold cross-validation**. Με την καλύτερη τιμή της α που προκύπτει, δημιουργείται ένα βέλτιστο μοντέλο Lasso Regression, το οποίο εκπαιδεύεται στο κανονικοποιημένο σύνολο εκπαίδευσης.

Επιλογή χαρακτηριστικών: Οι συντελεστές του μοντέλου Lasso (coef) λαμβάνονται, μετατρέπονται σε απόλυτες τιμές, και χρησιμοποιούνται για να αξιολογηθεί η σημασία των χαρακτηριστικών. Ένα χαρακτηριστικό θεωρείται σημαντικό αν ο συντελεστής του είναι μεγαλύτερος από ένα κατώφλι (0.018).



PCA: Principle Component Analysis

Με ανάλυση κυρίων συνιστωσών, μειώνεται ο αριθμός των features, ωστε να διατηρείται το 90%, ή αντίστοιχα το 95 % της διακύμανσης των αρχικών δεδομένων. Γενικά, στόχος είναι να δημιουργηθούν νέα χαρακτηριστικά, που να προκύπτουν από το αρχικό σύνολο (224 για το συγκεκριμένο dataset), με σκοπό να διατηρηθεί όσο περισσότερη πληροφορία γίνεται σε πολύ μικρότερο αριθμό features.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

ΣΥΓΚΡΙΣΗ ΤΑΞΙΝΟΜΗΤΩΝ

Classifier	Initial Dataset	Reduced Dataset (Lasso)	Reduced Dataset (PCA - 90%)
Naive Bayes	0.699	0.694	0.753
LDA	0.767	0.752	0.763
Gaussian Process	0.209	0.285	0.209
QDA	0.807	0.807	0.807
AdaBoost	0.620	0.609	0.658
Random Forest	0.806	0.805	0.782
XGBoost	**0.840**	0.815	0.822
CatBoost	0.824	0.811	0.798
k-NN (k=1)	0.787	0.779	0.791
k-NN (k=3)	0.803	0.792	0.804
k-NN, $k \in [1, 30]$	0.834	0.825	0.832
Nearest Centroid (Euclidean Distance)	0.804	-	-
Nearest Centroid (Manhattan Distance)	-	-	0.804

Μετά το πέρας των δοκιμών, παρατηρείται πως η χρήση των dataset μειωμένης διαστασιμότητας δεν έχει ουσιαστική επίδραση στα αποτελέσματα και στην ακρίβεια της εκάστοτε ταξινόμησης. Για τον λόγο αυτό, στα επόμενα βήματα της εργασίας θα χρησιμοποιηθεί το αρχικό training dataset.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Support Vector Machines for Multiclass Classification

Τύποι kernel

- **Linear** : Εσωτερικό γινόμενο μεταξύ των 2 vectors εισόδου.
- **Gaussian/RBF**: Τιμή του gaussian πυρήνα για τα 2 vectors εισόδου υπολογίζοντας την ευκλείδεια απόσταση μεταξύ τους και βάζοντας τα στην εκθετική συναρτηση.
- **Polynomial**: Τιμή του πολυωνυμικού πυρήνα για δύο vectors εισόδου βάσει της εκθετικής αύξησης στο εσωτερικό γινόμενο τους και του βαθμού του πολυωνύμου.

Κλάση SVM : Παράμετροι

- **kernel** (πυρήνας): Τύπος πυρήνα (μετασχηματισμός των δεδομένων).
- **C**: Παράμετρος κανονικοποίησης, που πολλαπλασιάζεται με τις μεταβλητές χαλαρότητας ξi στο τροποποιημένο πρόβλημα του QP. Είναι το βάρος τους κόστους των λάθος ταξινομήσεων. Για τιμή C=0, αγνοούνται τελείως οι παραμέτροι χαλαρότητας και οι λάθος ταξινομήσεις. Αν έχει μεγάλη τιμή, δίνεται μεγαλύτερη σημασία στη σωστή ταξινόμηση των προτύπων.
- **gamma**: Παράμετρος πυρήνα γκαουσιανής συνάρτησης.
- **degree** : Βαθμός του πυρήνα πολυωνυμικής συνάρτησης.
- **constant** : Σταθερά πολυωνυμικού πυρήνα.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Support Vector Machines for Multiclass Classification

Κλάση SVM : Συναρτήσεις

- **fit() :** Εκπαίδευση του μοντέλου. Αν >2 κλάσεις το πρόβλημα αντιμετωπίζεται ως multiclass με one-vs-one στρατηγική οπου δημιουργείται ένα ξεχωριστό δυαδικό μοντέλο για κάθε πιθανό ζεύγος κλάσεων. Για κάθε ζεύγος δημιουργείται ένα νέο SVM μοντέλο και εκπαιδεύεται μέσω της binary_fit() και υπολογίζεται η ακρίβεια.

- **binary_fit()**: Εκπαίδευση SVM για πρόβλημα δυαδικής ταξινόμησης:
  - Υπολογίζει τον γραμμικό πίνακα (Gram matrix) από τα δεδομένα εκπαίδευσης (επιλεγμένο kernel).
  - Δημιουργούνται οι πίνακες για το πρόβλημα βελτιστοποίησης Quadratic Programming(P, q, A, b).
  - Ορίζονται οι πίνακες περιορισμών (constraints) ανάλογα με το C. Ο πίνακας G περιέχει τους περιορισμούς του προβλήματος QP και ο h τις αντίστοιχες τιμές στις συνθήκες περιορισμού.
  - Οι συντελεστές Lagrange υπολογίζονται από τη λύση του QP προβλήματος και με βάση αυτούς προκύπτουν τα support vectors
  - Υπολογίζεται το σταθερό όριο (bias).

- **accuracy_calculator():** Υπολογίζει την ακρίβεια του μοντέλου συγκρίνοντας τις προβλέψεις του με τα πραγματικά labels. Χρησιμοποιεί την binary_predict για να παράγει προβλέψεις και υπολογίζει το ποσοστό των σωστών προβλέψεων.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Support Vector Machines for Multiclass Classification

Κλάση SVM : Συναρτήσεις

- **prepare_binary_data()** : Προετοιμάζει τα δεδομένα εκπαίδευσης για ένα δυαδικό πρόβλημα ταξινόμησης. Αρχικά, γίνεται διαχωρισμός δειγμάτων από δύο συγκεκριμένες κλάσεις, που ορίζονται από τις class1(λαμβάνει label +1) και class2(-1).

- **predict()** : Πραγματοποιεί προβλέψεις για τόσο δυαδική όσο και multiclass ταξινόμηση. Στην περίπτωση multiclass κάθε ζεύγος κλάσεων εκπαιδεύει ένα μοντέλο και οι προβλέψεις καταγράφονται ως ψήφοι(1vs1). Μετά τη συγκέντρωση των ψήφων από όλα τα μοντέλα, η τελική ετικέτα κάθε δείγματος καθορίζεται με βάση την πλειοψηφία των ψήφων. Στη δυαδική ταξινόμηση, χρησιμοποιεί απευθείας τη μέθοδο binary_predict για πρόβλεψη.

- **binary_predict()** : Υλοποιεί την πρόβλεψη του μοντέλου SVM ανάλογα με τον τύπο του kernel που χρησιμοποιείται στο test set και επιστρέφει ως έξοδο τις προβλέψεις (-1 ή 1 για κάθε test sample).

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Αποτελέσματα για όλους τους τύπους Kernel

- Πραγματοποιήθηκαν αρκετές δοκιμές για την επιλογή των παραμέτρων του κάθε πυρηνα και παρουσιάζονται κάποιες ενδεικτικά.

Τύπος Kernel	Parameters	Test Accuracy	F1-Score weighted avg	Recall weighted avg	Precision weighted avg
Linear	C = 1	0.7878	0.79	0.79	0.79
Gaussian/RBF	C = 1, gamma =0.1	0.8107	0.82	0.81	0.81
Gaussian/RBF	C = 1, gamma = 1	0.20	0.04	0.21	0.07
Polynomial	C=1,degree=2, constant=10	0.8461	0.85	0.85	0.85
Polynomial	C=0.1,degree=2, constant=10	0.8461	0.85	0.85	0.85
Polynomial	C=1,degree=2, constant=1	0.8490	0.85	0.85	0.85
Polynomial	C=1,degree=2, constant=1	0.8553	0.86	0.86	0.86

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Αποτελέσματα για όλους τους τύπους Kernel

- Από τη σύγκριση των αποτελεσμάτων προκύπτει ότι ο **Polynomial** πυρήνας προσφέρει την καλύτερη απόδοση για το συγκεκριμένο πρόβλημα. Αυτό μπορεί να αποδοθεί στη δυνατότητά του να μοντελοποιεί μη γραμμικές σχέσεις στα. Επιτυγχάνει την υψηλότερη ακρίβεια **(85.53%**) και εξαιρετικά F1-Score, Recall και Precision (**0.86**). Αυτό υποδεικνύει ότι το συγκεκριμένο πρόβλημα ταξινόμησης παρουσιάζει μη γραμμικά όρια απόφασης που μπορεί να μοντελοποιήσει αποτελεσματικά.

- Ο **Gaussian/RBF** πυρήνας, ο οποίος βασίζεται σε μία εκθετική συνάρτηση και έχει τη δυνατότητα να απομονώνει μη γραμμικά μοτίβα, εμφανίζει καλή απόδοση για **γ = 0.1**.

- Ο **γραμμικός** πυρήνας, από την άλλη, παρουσιάζει σαφώς χαμηλότερη ακρίβεια, γεγονός που υποδηλώνει ότι οι σχέσεις μεταξύ των χαρακτηριστικών και των κλάσεων είναι μη γραμμικές και δεν μπορούν να περιγραφούν ικανοποιητικά από μία απλή γραμμική συνάρτηση.

- **Συμπερασματικά**, η εξήγηση για την υπεροχή του Polynomial πυρήνα έγκειται στην ικανότητά του να συλλαμβάνει πολύπλοκες μη γραμμικές σχέσεις στα δεδομένα. Ο γραμμικός πυρήνας, λόγω της περιορισμένης ευελιξίας του, δεν αποτελεί κατάλληλη επιλογή για το συγκεκριμένο σύνολο δεδομένων.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

SVM : Hyperparameter Tuning

Μετά από δοκιμές για κάποιες υπερπαραμετρους γίνεται απόπειρα εύρεσης των καταλληλότερων με τα εξής βήματα:

- Ενωση train και test sets, προκειμένου να χρησιμοποιηθεί η τεχνική **cross validation**.
- Για κάθε τύπο kernel, **grid search** για τις αντίστοιχες υπερπαραμετρους. Για κάθε παράμετρο ελέγχονται όλοι οι δυνατοί συνδυασμοί, με 3-fold Cross validation για να έχει αξιοπιστία η τελική επιλογή.
- Επιλογή μοντέλου με την καλύτερη ακρίβεια.

Αποτελέσματα: Μετά από δοκιμές διαπιστώθηκε ότι ο πολυωνυμικός πυρήνας πετυχαίνει την καλύτερη ακρίβεια. Οι τιμές για τις οποίες γίνεται το grid search είναι : **C = [0.1 , 1, 10, 100], degree = [2, 3, 4] , constant = [0 , 0.1, 1, 2].**

```
Optimal parameters for the model:
C = 1, Degree = 2, Constant = 0, Accuracy: 0.8507718696397941
```

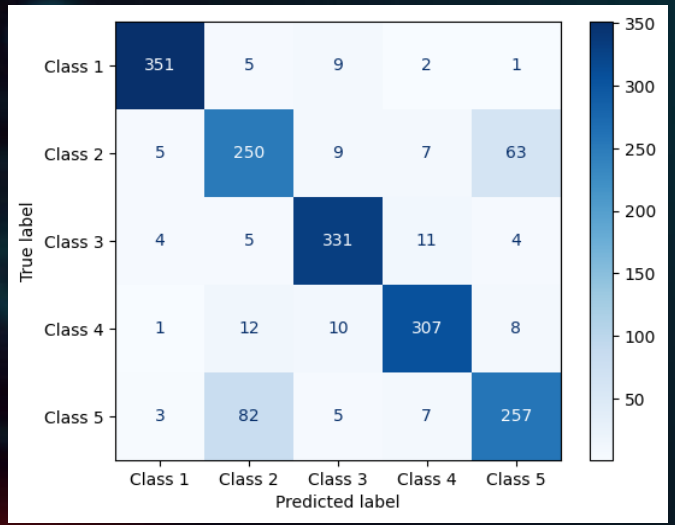
ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Αποτελέσματα SVM με Πολυωνυμικό Πυρήνα

Classification Report Βέλτιστου Μοντέλου

	precision	recall	f1-score	support
1	0.96	0.95	0.96	368
2	0.71	0.75	0.73	334
3	0.91	0.93	0.92	355
4	0.92	0.91	0.91	338
5	0.77	0.73	0.75	354
accuracy			0.86	1749
macro avg	0.85	0.85	0.85	1749
weighted avg	0.86	0.86	0.86	1749

- Πετυχαίνει συνολική ακρίβεια (accuracy) 86%. Η **κλάση 1** παρουσιάζει την καλύτερη απόδοση, με F1-Score 96%, ενώ η **κλάση 2** και η **κλάση 5** εμφανίζουν χαμηλότερη απόδοση, με F1-Score 73% και 75% αντίστοιχα. Φαίνεται ότι το μοντέλο δυσκολεύεται περισσότερο στη διάκριση αυτών των 2 κλάσεων.



Confusion Matrix

Για την κλάση 2, 63 δείγματα ταξινομήθηκαν λανθασμένα ως 5, υποδεικνύοντας ισχυρή σύγχυση μεταξύ τους. Αντίστοιχα, για την κλάση 5, 82 δείγματα ταξινομήθηκαν ως κλάση 2. Από την άλλη πλευρά, οι υπόλοιπες κλάσεις, όπως η 1, παρουσιάζουν ελάχιστη σύγχυση, καθώς οι περισσότερες προβλέψεις είναι σωστές(351 σωστά ταξινομημένα δείγματα).

Περιθώρια βελτίωσης στη διαχωριστική ικανότητα του SVM για τις κλάσεις 2 και 5 : **data augmentation** για τις κλάσεις **2** και **5**

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Classification με Multi-Layer Perceptron (Fully Densed)

Εισαγωγή

Για την συνέχεια, θεωρείται πως η χρήση αλγορίθμων βαθιάς μάθησης είναι η πιο κατάλληλη προσέγγιση, καθώς αναμένεται να αποδώσει τα καλύτερα δυνατά αποτελέσματα.

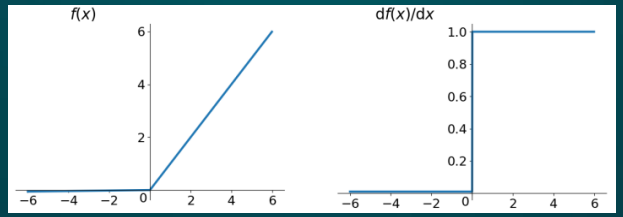
Η χρήση ενός νευρωνικού δικτύου προκρίνεται ως μία ακόμα επιλογή, λόγω της ικανότητάς του να προσαρμόζεται σε σύνθετα προβλήματα και να παρέχει υψηλή απόδοση. Δεδομένου, ωστόσο, ότι ένα νευρωνικό δίκτυο αποτελεί μια γενική τεχνολογία με μεγάλο πλήθος παραμέτρων, αποφασίστηκε να περιοριστεί ο χώρος αναζήτησης στις αρχιτεκτονικές πλήρως συνδεδεμένων νευρωνικών δικτύων, προκειμένου να επιτευχθεί η βέλτιστη λύση.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

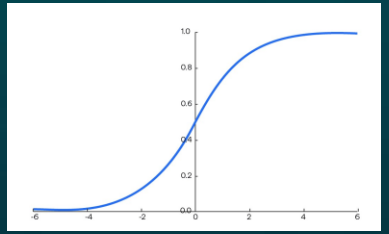
Classification με Multi-Layer Perceptron (Fully Densed)

- Για κάποιες παραμέτρους επιλέχθηκαν συγκεκριμένες τιμές , οι οποίες δεν αλλάζουν κατά την διάρκεια της εργασίας

- Ως συνάρτηση ενεργοποίησης των hidden layers επιλέγεται η **ReLU** (Rectified Linear Unit).Η επιλογή αυτή είναι η πλέον διαδεδομένη για MLP και CNN.



- Ως συνάρτηση ενεργοποίησης του στρώματος εξόδου επιλέγεται η **softmax**, η οποία μετατρέπει ένα διάνυσμα K πραγματικών αριθμών σε κατανομή πιθανότητας K πιθανών αποτελεσμάτων και μπορεί να θεωρηθεί ως πιθανότητες με τις οποίες ανήκει η είσοδος σε κάθε κλάση.



- Ως συνάρτηση κόστους επιλέγεται η **categorical cross-entropy**, ειδικά για classification tasks που περιλαμβάνουν πολλαπλές κλάσεις. Εφαρμόζει λογαριθμικές συναρτήσεις στις προβλεπόμενες πιθανότητες, τιμωρώντας πιο σημαντικά σφάλματα, (βοηθά στη σύγκλιση κατά τη διάρκεια του training). Στην συγκεκριμένη περιπτωση , η συναρτηση θα είναι :

$$CE\ Loss = -\sum_{i=1}^{number\ of\ Classes=10} y_{test} \times log(y_{prediction})$$

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Multilayer Perceptron Neural Network

- **Συνάρτηση MLP_Model() :** Δημιουργεί MLP NN με εισόδους : το μέγεθος των χαρακτηριστικών του dataset (input_shape), αριθμό κλάσεων εξόδου (numClasses), τύπο του optimizer (adam ή sgd), αριθμός epochs (numEpochs) και αρχικό learning rate (initial_learning_rate). Περιλαμβάνει ένα layer εισόδου (Dense) με ενεργοποίηση ReLU, κανονικοποίηση batch (BatchNormalization) και dropout για την αποφυγή υπερεκπαίδευσης. Ακολουθούν αρκετά κρυφά layers με συνάρτηση ενεργοποίησης ReLU, κανονικοποίηση batch και dropout. Το layer εξόδου αποτελείται από ένα 5 νευρώνες(αριθμός κλάσεων) και Softmax (Σ.Ε.). Ο ρυθμός εκμάθησης μέσω ενός μηχανισμού Exponential Decay, μειώνεται σταδιακά κατά τη διάρκεια της εκπαίδευσης.

- **Εκπαίδευση με SGD (Stohastic Gradient Descent) :**

  Κατά τη διάρκεια κάθε εποχής, το δίκτυο εκπαιδεύεται σε υποσύνολα δεδομένων (mini-batches) ώστε να προσαρμόζει σταδιακά τις παραμέτρους του. Σε κάθε βήμα, το μοντέλο εκτελεί forward pass για να υπολογίσει τις προβλέψεις του και συγκρίνει αυτές τις προβλέψεις με τις πραγματικές ετικέτες, υπολογίζοντας μια τιμή απώλειας. Στη συνέχεια, με τη χρήση της μεθόδου backpropagation, υπολογίζονται οι παράγωγοι της απώλειας ως προς τις παραμέτρους του μοντέλου και χρησιμοποιούνται από τον αλγόριθμο βελτιστοποίησης (optimizer) για την ενημέρωση των βαρών του μοντέλου, με στόχο τη βελτίωση της απόδοσής του.

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Δομή MLP NN

Dropout

Εφαρμογή σε κάθε layer : μία τιμή **0 < p < 1** που καθορίζει το ποσοστό των νευρώνων του επιπέδου που θα απενεργοποιηθούν με τυχαία επιλογή για κάθε επανάληψη . Η τεχνική του dropout περιορίζει περιπτώσεις οπού υπάρχουν κόμβοι που διορθώνουν λάθη προηγουμένων που οδηγούν σε αδυναμία γενίκευσης. Μετα από κάποιες δοκιμές για το συγκεκριμένο μοντέλο επιλέγεται η τιμή 0.4 για το p .

Batch Normalization

Γίνεται υπολογισμός της μέσης τιμής και διακύμανσης του διανύσματος εισόδου, κατά μήκος του batch, εκτελείται κανονικοποίηση πριν την εφαρμογή της συνάρτησης ενεργοποίησης. Βοηθάει ιδιαίτερα στην διαδικασία εκπαίδευσης του μοντέλου. Δίνεται η δυνατότητα να χρησιμοποιηθούν υψηλότεροι lr, επιτρέποντας την ταχύτερη σύγκλιση.

Layer (type)	Output Shape
dense_6 (Dense)	(None, 1024)
batch_normalization_5 (BatchNormalization)	(None, 1024)
dropout_5 (Dropout)	(None, 1024)
dense_7 (Dense)	(None, 512)
batch_normalization_6 (BatchNormalization)	(None, 512)
dropout_6 (Dropout)	(None, 512)
dense_8 (Dense)	(None, 400)
batch_normalization_7 (BatchNormalization)	(None, 400)
dropout_7 (Dropout)	(None, 400)
dense_9 (Dense)	(None, 256)
batch_normalization_8 (BatchNormalization)	(None, 256)
dropout_8 (Dropout)	(None, 256)
dense_10 (Dense)	(None, 128)
batch_normalization_9 (BatchNormalization)	(None, 128)
dropout_9 (Dropout)	(None, 128)
dense_11 (Dense)	(None, 5)

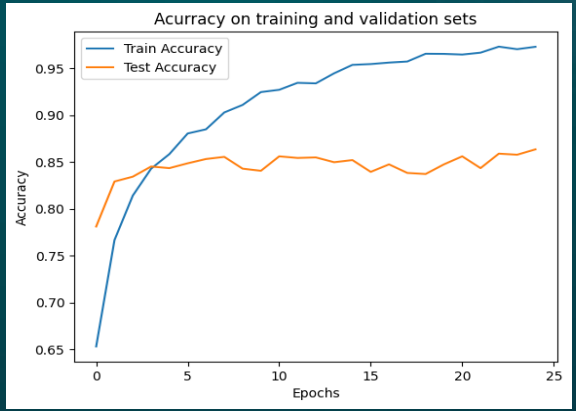
ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

MLP Neural Network : Αποτελέσματα Εκπαίδευσης

❖ Accuracy

Training set : Σταθερή αύξηση και σταθεροποίηση γύρω στο 98%.

Validation set : Αυξάνεται αρχικά αλλά σταθεροποιείται γύρω στο 0.85, χωρίς να βελτιώνεται σημαντικά μετά τις πρώτες 5-7 εποχές. Αυτό υποδηλώνει ότι το μοντέλο πιθανώς έχει φτάσει στο μέγιστο της απόδοσής του στο σετ δοκιμής.



❖ Loss : Categorical Cross-entropy

Training Set : Συνεχής μείωση, φτάνοντας σε πολύ χαμηλά επίπεδα (<0.1), υποδεικνύοντας ότι το μοντέλο μαθαίνει καλά το train set.

Validation Set : Αρχική μείωση αλλά στη συνέχεια παρουσιάζει αστάθεια και ελαφρά αύξηση μετά από περίπου 5-7 εποχές, κάτι που υποδηλώνει πιθανή υπερπροσαρμογή (overfitting).

Επίδοση στο test set

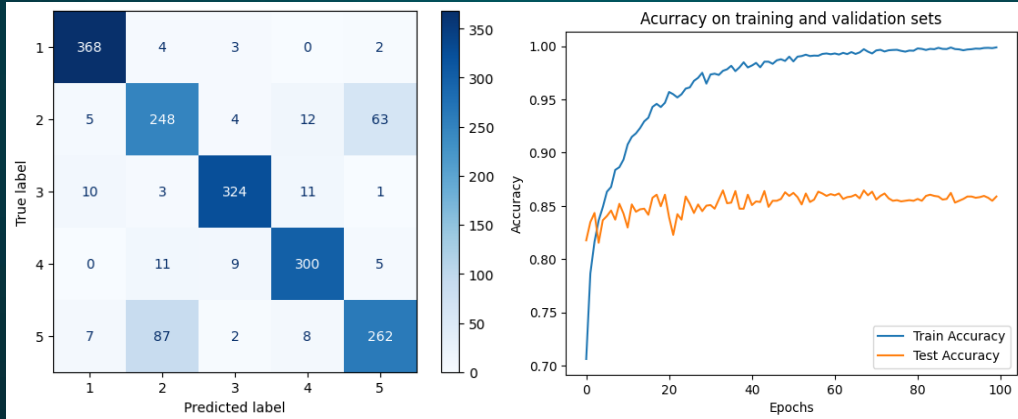
```
Test accuracy: 0.8634
Test F1 Score: 0.8651
```

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

MLP Neural Network : Αποτελέσματα Εκπαίδευσης (2)

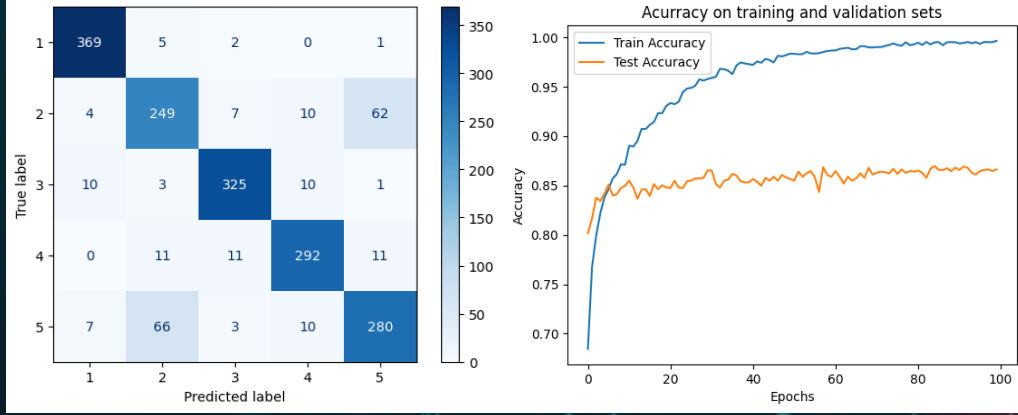
- Με αύξηση των epochs = 100 και των νευρώνων

```
Test accuracy: 0.8588
Test F1 Score: 0.8596
```



- Με αύξηση των epochs = 100 και μείωση των νευρώνων

```
Test accuracy: 0.8662
Test F1 Score: 0.8660
```



ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

MLP: Hyparameter Tuning

Σε αυτό το στάδιο της εργασίας θα γίνει προσπάθεια εύρεσης των καταλληλότερων για το dataset. Γενικά έγιναν αρκετές δοκιμές με **αριθμούς hidden layers, αριθμό νευρώνων, learning rates, batch sizes** και από παρατηρήσεις προέκυψαν οι πιθανές βέλτιστες υπερπαραμετροι που εξετάζονται.

- **Συνάρτηση buildModel() :** Δημιουργεί ένα πλήρως συνδεδεμένο NN (MLP) με τη χρήση υπερπαραμέτρων. Στόχος είναι να δοκιμαστούν διαφορετικοί συνδυασμοί υπερπαραμέτρων:
  - **Πιθανότητα Dropout (p)**: 0.2 και 0.4.
  - **Αριθμός Νευρώνων στο 1ο hidden layer**: **2048, 1024, 512, 400, 256 και 128**.
  - **Αριθμός Νευρώνων στο 2ο hidden layer**: **1024, 512, 400, 256 και 128**.
  - **Αριθμός Νευρώνων στο 3ο hidden layer: 512, 400, 256 και 128.**
  - **Ρυθμός Μάθησης (learning_rate): 0.001, 0.01 και 0.1**.

- Για την επιλογή των υπερπαραμετρων χρησιμοποιείται ο **RandomSearch** tuner που ελέγχει επιλεκτικά κάποιους πιθανούς συνδυασμούς από όλο το grid (αντί grid search στο οποίο θα ελεγχθούν όλοι οι πιθανοί συνδυασμοί, και επομένως το πλήθος των συνδυασμών θα αυξάνονταν εκθετικά για κάθε υπερπαράμετρο που θα προστίθονταν).

  Κάθε συνδυασμός:
  - **training 3 φορές (executions_per_trial)**/ τυχαιότητα στο τελικό μοντέλο
  - **25 epochs.**

ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Αποτελέσματα Tuning Υπερπαραμέτρων

Βέλτιστο Μοντέλο

```
Trial 106 Complete [00h 03m 01s]
val_accuracy: 0.8541815678278605

Best val_accuracy So Far: 0.8596616784731547
Total elapsed time: 03h 19m 32s
Total time: 11972.505 seconds

Optimal number of neurons in 1st layer: 1024
Optimal number of neurons in 2nd layer: 512
Optimal number of neurons in 3rd layer: 256
Optimal value of p for dropout: 0.4
Optimal value of learning rate: 0.001
```

❖ Classification Report Βέλτιστου Μοντέλου

Η συνολική ακρίβεια του μοντέλου είναι **84%**. Οι κλάσεις 0 και 2 έχουν τις καλύτερες επιδόσεις, με υψηλές τιμές για precision, recall, και f1-score (πάνω από 90%). Οι 1 και 4 παρουσιάζουν χαμηλότερη απόδοση, με το f1-score να φτάνει το 0.70 και 0.75 αντίστοιχα, γεγονός που υποδεικνύει δυσκολία στη σωστή ταξινόμησή τους.

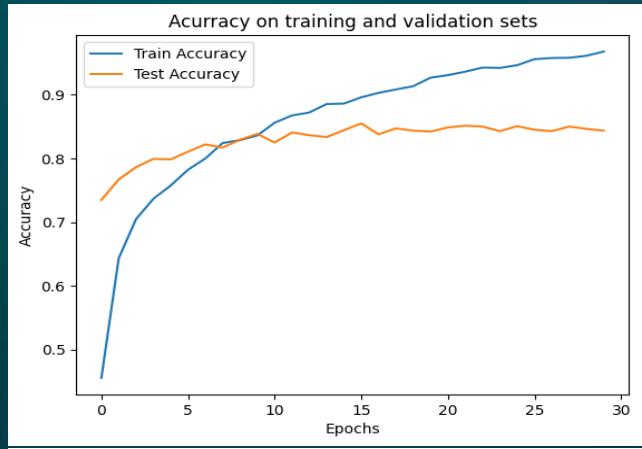
	precision	recall	f1-score	support
0	0.93	0.98	0.96	377
1	0.70	0.70	0.70	332
2	0.92	0.91	0.91	349
3	0.89	0.88	0.88	325
4	0.76	0.74	0.75	366
accuracy			0.84	1749
macro avg	0.84	0.84	0.84	1749
weighted avg	0.84	0.84	0.84	1749

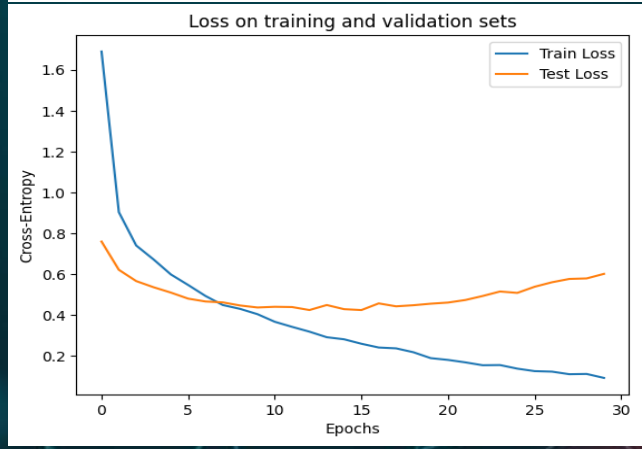
ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Αποτελέσματα Βέλτιστου Μοντέλου

- Παρατηρείται ότι η ακρίβεια στο **σύνολο εκπαίδευσης** αυξάνεται περαιτέρω φτάνοντας περίπου το **97%**, αλλά στο **σύνολο δοκιμών** παραμένει σταθερή γύρω στο **85%**, χωρίς σημαντική βελτίωση. Η απώλεια στο σύνολο εκπαίδευσης συνεχίζει να μειώνεται, ενώ στο σύνολο δοκιμών παρουσιάζει διακυμάνσεις και ελαφρά αυξητική τάση, υποδεικνύοντας υπερπροσαρμογή (overfitting). Η εκπαίδευση για περισσότερες εποχές (30 αντί 25) δεν βελτίωσε την απόδοση στο σύνολο δοκιμών.

- Με τη χρήση τεχνικών όπως το early stopping, την περαιτέρω μείωση του learning rate ή την εκπαίδευση για περισσότερες εποχές, είναι πιθανό η ακρίβεια να προσεγγίσει ακόμη και το **90%**.



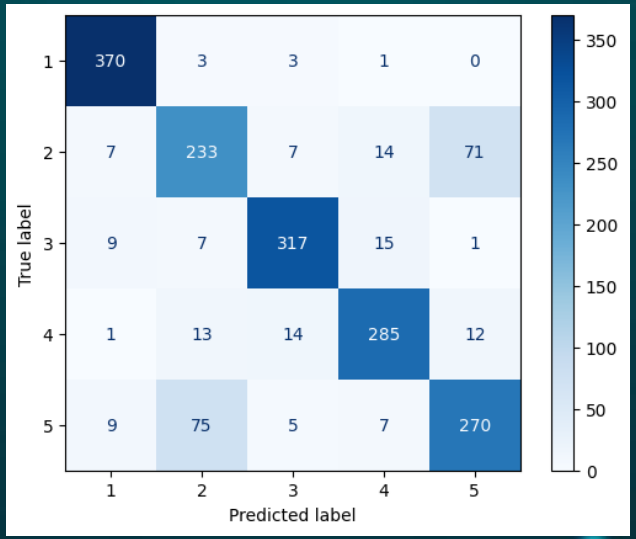


ΜΕΡΟΣ Δ - ΑΝΑΠΤΥΞΗ ΑΛΓΟΡΙΘΜΟΥ ΤΑΞΙΝΟΜΗΣΗΣ

Αποτελέσματα Βέλτιστου Μοντέλου

Confusion Matrix

Το μοντέλο αποδίδει ικανοποιητικά, με υψηλή ακρίβεια για την πλειονότητα των κλάσεων, ιδιαίτερα για την 1 και την 3 με 370 και 317 σωστές προβλέψεις αντίστοιχα. Ωστόσο, υπάρχει σημαντική σύγχυση μεταξύ της 2 και της 5, με 71 δείγματα από την 2 να ταξινομούνται λανθασμένα ως κλάση 5 και 75 δείγματα από την 5 να ταξινομούνται ως κλάση 2. Η συγκεκριμένη αστοχία ενδέχεται να υποδηλώνει την ύπαρξη κοινών χαρακτηριστικών ανάμεσα σε αυτές τις δύο κλάσεις. Επιπλέον, παρατηρείται μικρότερης κλίμακας σύγχυση μεταξύ της κλάσης 4 και της κλάσης 3, με 14 δείγματα να ταξινομούνται λανθασμένα.



Συνολικά, το μοντέλο φαίνεται να χρειάζεται βελτίωση στη διακριτική του ικανότητα μεταξύ συγκεκριμένων κλάσεων.