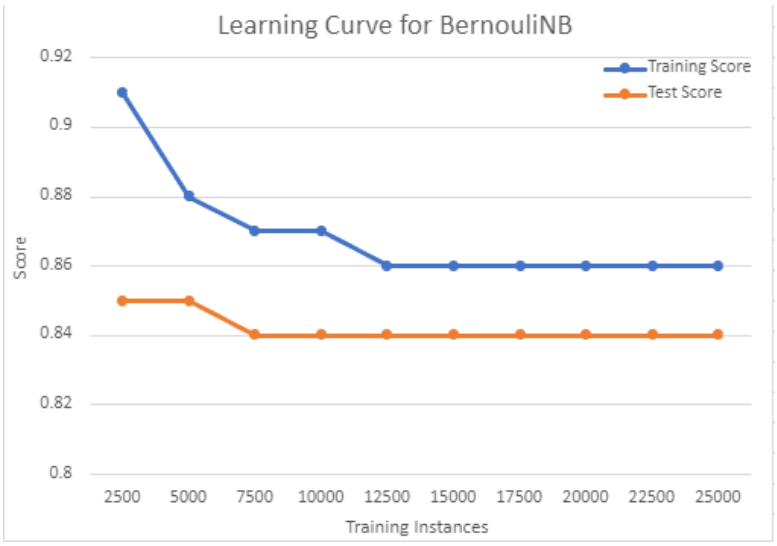
Αποστόλου Ιωάννης 3190013
Καλαντζής Ηλίας 3190068
Κωνσταντίνος Κατσάμης 31900237

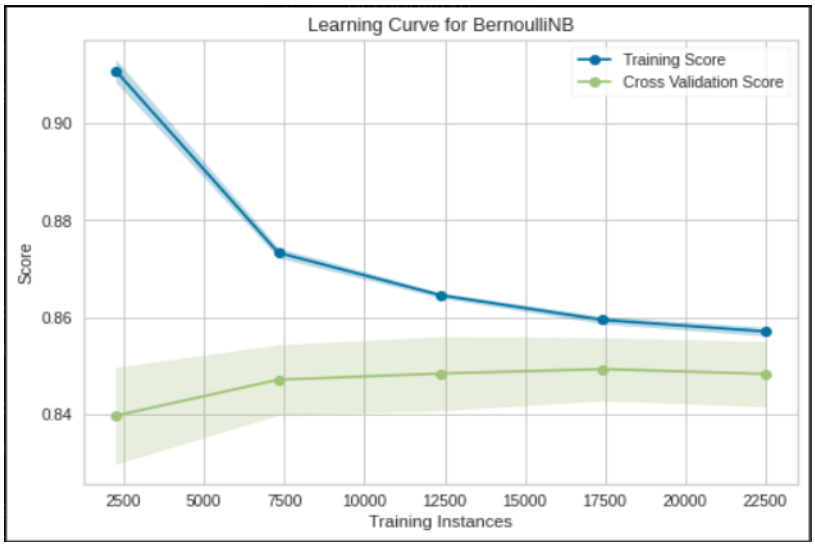
Αφελής ταξινομητής Bayes: Bernoulli

Ο Naive Bayes Bernoulli εκπαιδεύεται υπολογίζοντας την πιθανοτήτων κάθε λέξης του λεξιλογίου που το έχει δοθεί στο στάδιο της εκπαίδευσης σε συνδυασμό με τα δεδομένα εκπαίδευσης. Έτσι όταν του δοθούν νέα δεδομένα υπολογίζει το γινόμενο τών πιθανοτήτων κάθε λέξεις συμφωνα με τις κριτικές που είχε συναντήσει στο παρελθόν και σύμφωνα αυτων των πιθανοτήτων καταλήγει στο αν το νέο σχόλιο που του δόθηκε είναι θετικό η αρνητικό.

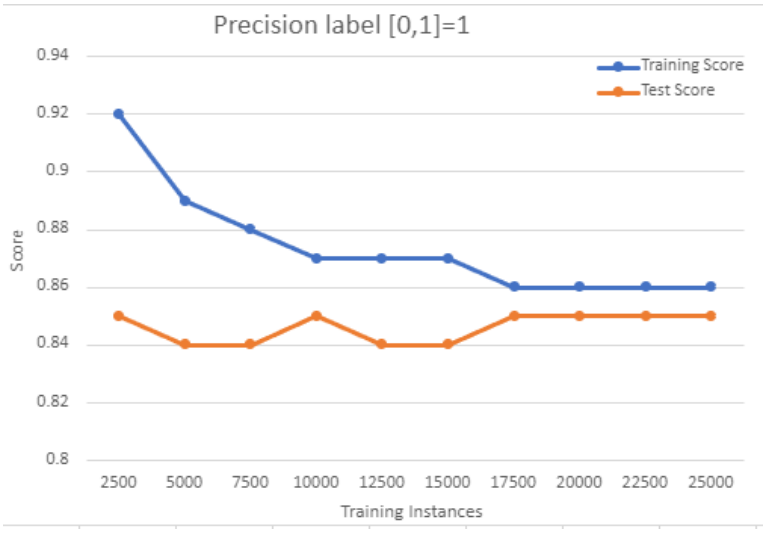
Σταθερές και πίνακες που χρησιμοποιήθηκαν:
Κατα την υλοποίηση μας έπειτα από δοκιμές χρησιμοποιούμε λεξικό M =4000 πιο συχνών λέξεων και δεδομένα ελέγχου ,εκπαίδευσης μεγέθους 25000. Αυτο έχει σαν αποτέλεσμα να έχουμε 4 πίνακες τους:
X_train_binary ,X_test_binary (μεγεθος 25000x4000) οι οποίοι περιέχουν τις κριτικές σε μορφή διανύσματος ιδιοτήτων ενω Y_train ,Y_test (μέγεθος 1x25000) είναι οι πίνακες που περιέχουν τις αποτιμήσεις κάθε κριτικής που απεικονίζουν με 1 τις θετικές και 0 της αρνητικές.
Τα δεδομένα φορτόνονται και μετατρέπονται στην σωστή μορφή απο της load_words. Η εκπαίδευση γίνεται απο την fit και η αποτίμηση απο την predict.

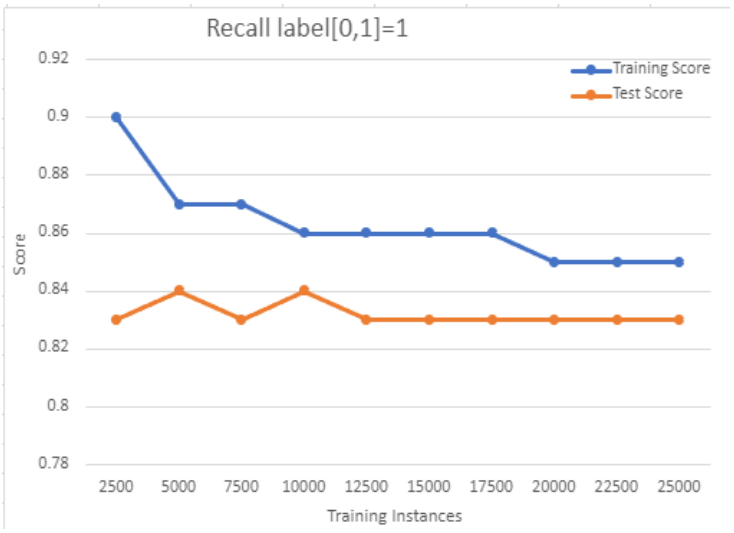
Αποτελέσματα:
Η ακρίβεια μας “σταθεροποιήτε” στα 12500 στοιχεία οπου από εκεί και πέρα παραμένει σταθερή στο 86% ακρίβεια σε νέα δεδομένα και 84% σε δεδομένα εκπαίδευσης . Στα διαγράμματα από κάτω δεξιά έχουμε την δική μας υλοποίηση ενω αριστερά την υλοποίηση της βιβλιοθήκης sklearn. Τα learning curve και των 2 περιπτώσεων είναι πολύ κοντα με την έτοιμη βιβλιοθήκη κοντά στα 12500 στοιχεία να έχει 86.5% ακρίβεια σε νέα δεδομένα και 85% σε στα δεδομένα όπου εκπαιδεύτικε

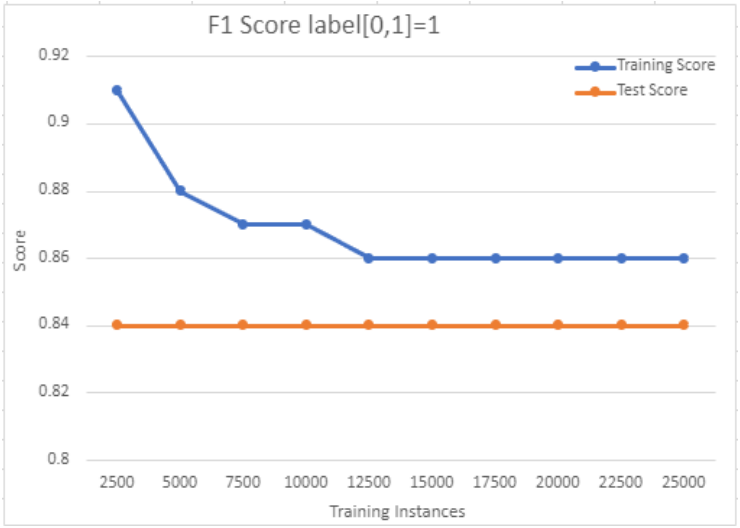




Καμπύλες precision,recall,F1 στην περίπτωση μας τα δεδομένα ελέγχου και εκπαίδευσης μας αποτελούνταν απο 25000 στοιχεια στο κάθε ένα εκ των οποίων τα 12500 ηταν θετικά και τα υπόλοιπα 12500 ήταν αρνητικά. Συνεπώς τα δείγματα μας ηταν χωρισμενα 50-50 με αποτέλεσμα και οι 3 καμπύλες να έχουν σχεδόν τα ίδια αποτελέσματα με την precision στο 86% και 85% για δεδομένα εκπαίδευσης και ελέγχου αντίστοιχα. Την recall με 85% και 83% για δεδομένα εκπαίδευσης και ελέγχου και την F1 όπου είχε 86% και 85%





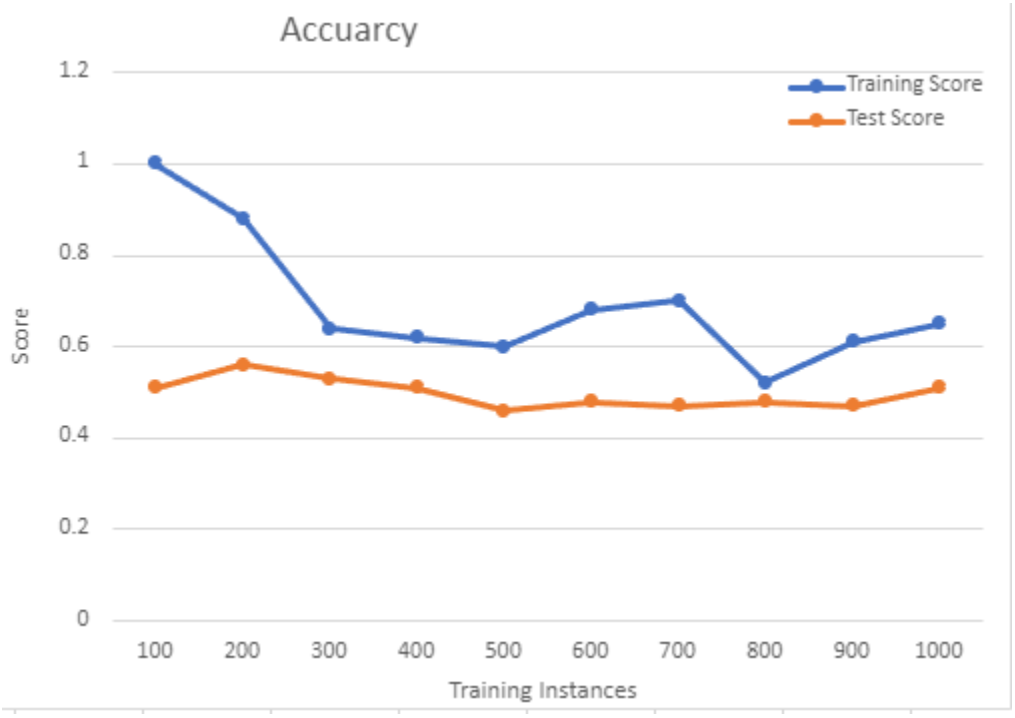


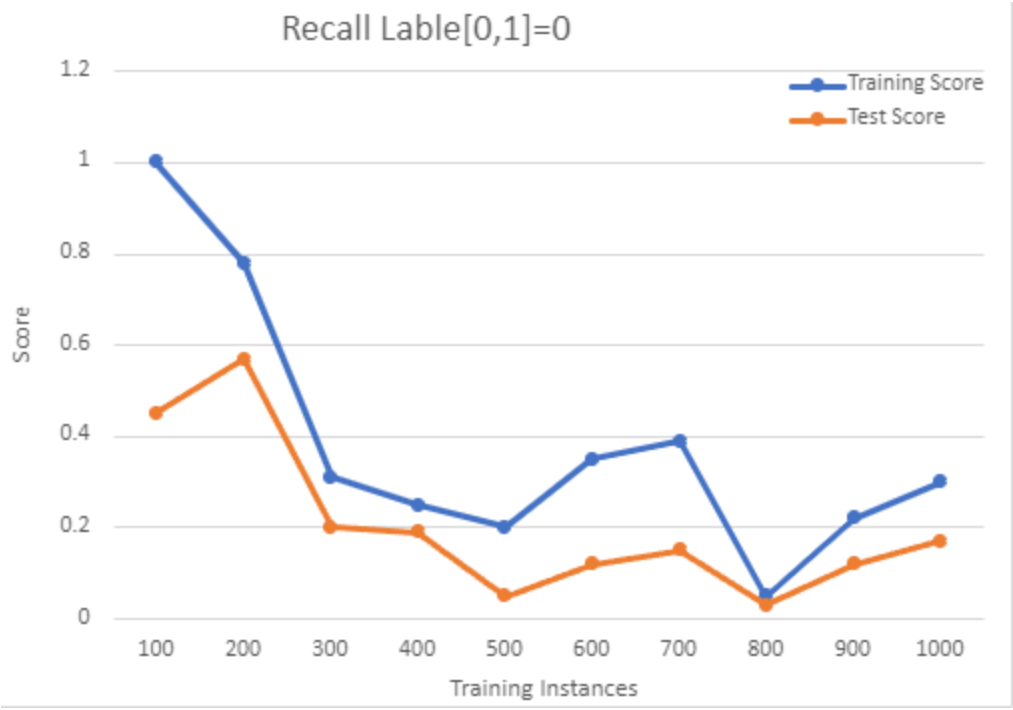
ID3

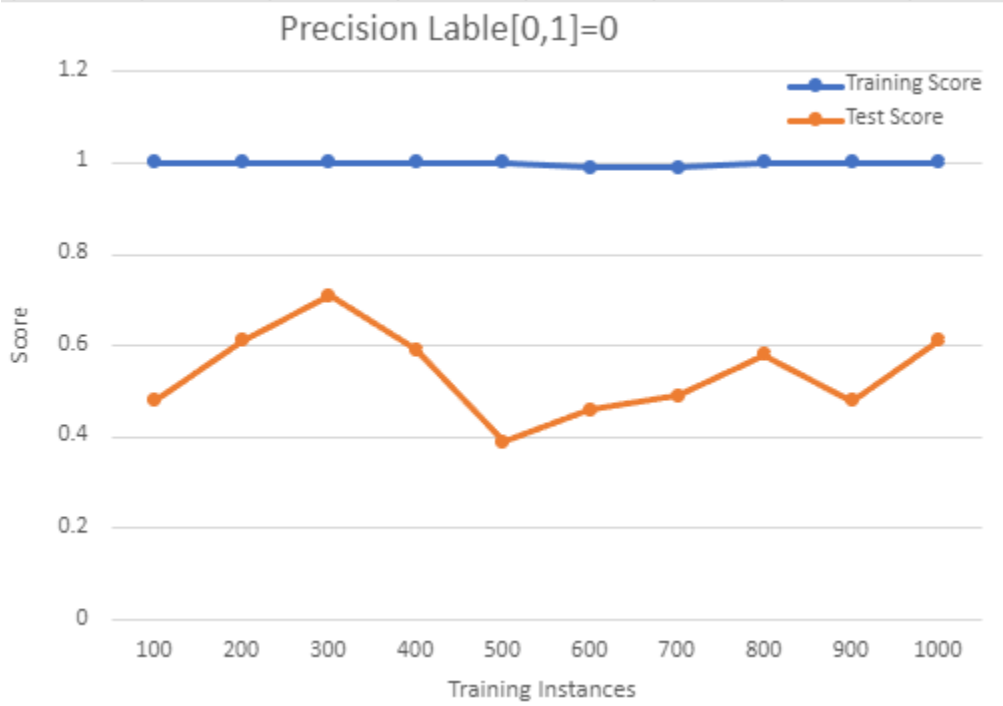
Ο ID3 είναι ένας αλγόριθμος στον οποίο δημιουργούμε ένα δέντρο αποφάσεων. Για να έχουμε σωστές αποφάσεις θα πρέπει να υπάρξει μια σωστή αναλογία μεταξύ βάθους του δέντρου, καθώς οι υπολογιστικοί πόροι δεν είναι απεριόριστοι.

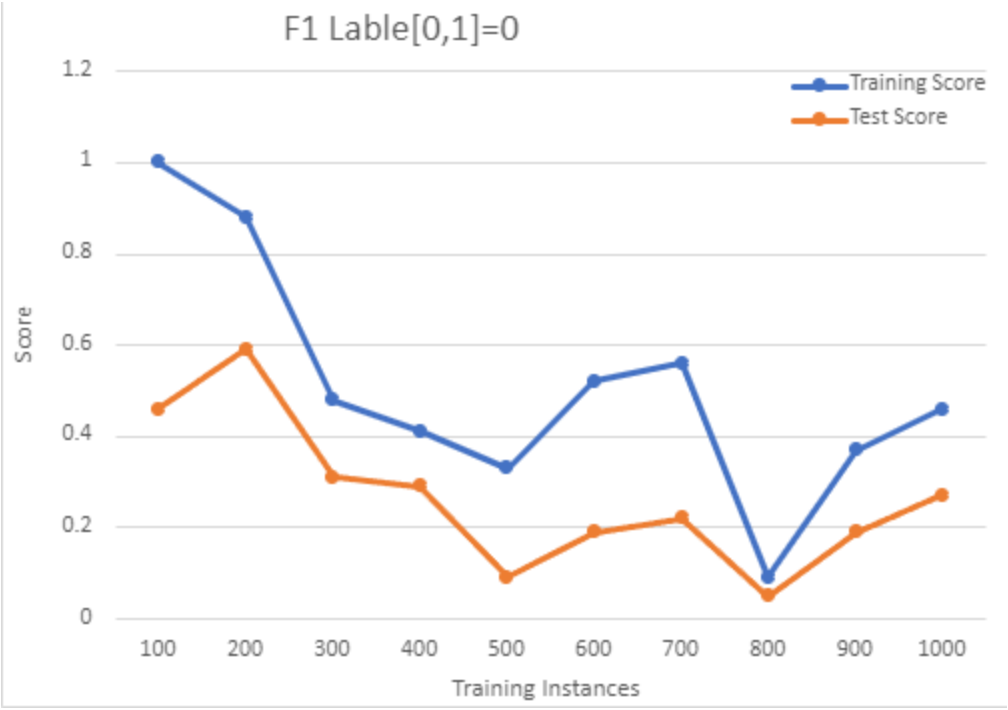
Μεγάλο Βάθος & λίγα δεδομένα:

Όταν έχουμε μεγάλο βάθος και λίγα δεδομένα, καταφέρνει το δέντρο να βγάλει πολύ καλά αποτελέσματα στα δεδομένα που εκπαιδεύτηκε επειδή έχει μεγάλο βάθος, όμως σε καινούργια τα ποσοστά δεν είναι εξίσου καλά. Παράλληλα παρατηρούμε πως υπάρχει overfitting.



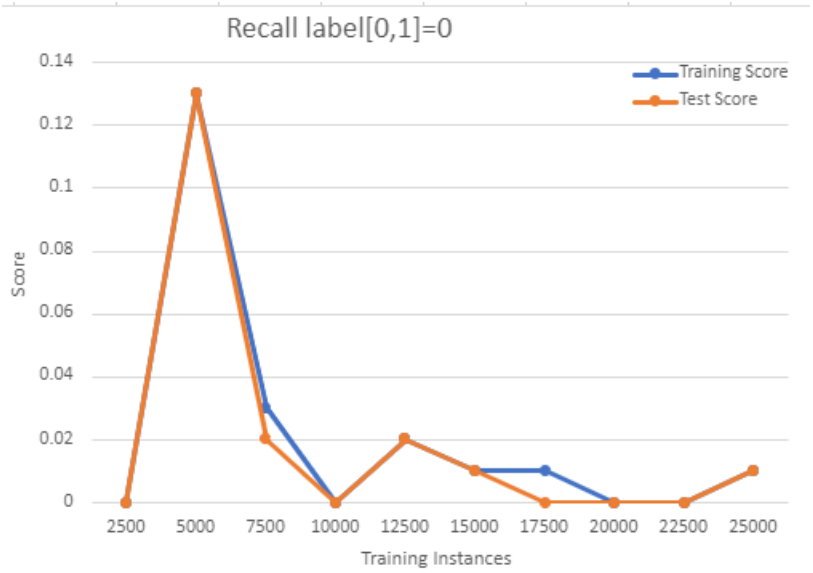

Recall Lable[0,1]=0
Training Score
Test Score
Score
Training Instances
1.2
1
0.8
0.6
0.4
0.2
0
100
200
300
400
500
600
700
800
900
1000


Precision Lable[0,1]=0
Training Score
Test Score
Score
Training Instances
1.2
1
0.8
0.6
0.4
0.2
0
100
200
300
400
500
600
700
800
900
1000

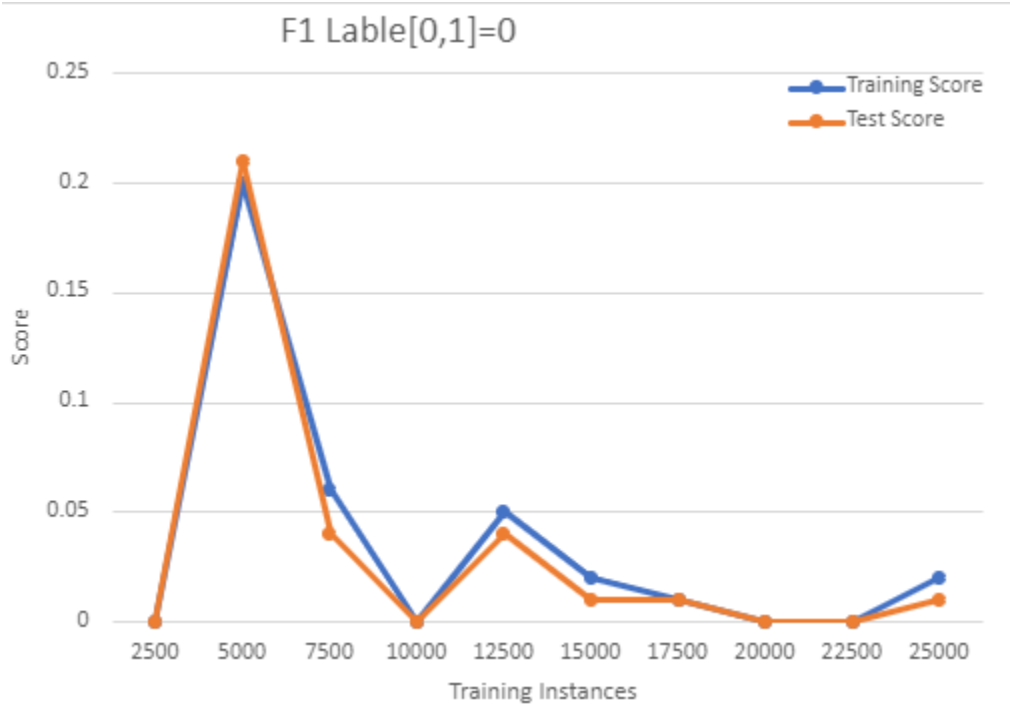


Μικρό Βάθος & πολλά δεδομένα:

Στα πολλά δεδομένα με μικρό βάθος παρατηρούμε πως το δέντρο βαραίνει πάρα πολλή από την μια πλευρά του και τα περισσότερα δεδομένα καταλήγουν σε ένα γενικό κόμβο, όπου είναι τα ποσοστά ενός μεγάλου μέρους των δεδομένων και δεν μπορούμε να βγάλουμε σαφή συμπεράσματα. Για παράδειγμα στα δικά μας δεδομένα που είναι 50-50 οι πιθανότητες, καταλήγει ο μεγαλύτερος όγκος των δεδομένων στο τελευταίο φύλλο όπου έχει την πιθανότητα 0.5-0.5 και άρα δεν βγάζουμε σχεδόν καθόλου σωστά αποτελέσματα.
Εδώ παρακάτω βλέπουμε τα δεδομένα για 25.000 στοιχεία (δλδ όλα τα στοιχεία) και το βάθος 5.




Precision
Training Score
Test Score
1.2
1
0.8
0.6
0.4
0.2
0
Score
2500
5000
7500
10000
12500
15000
17500
20000
22500
25000
Training Instances

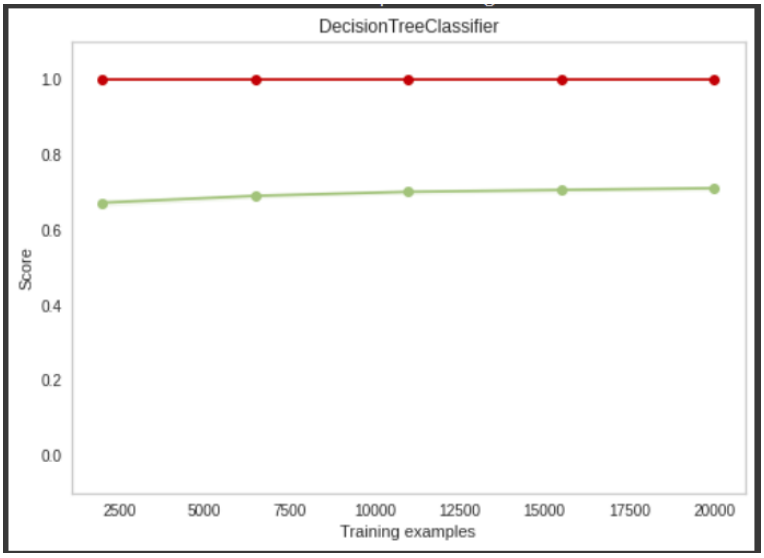

F1 Lable[0,1]=0
Training Score
Test Score
0.25
0.2
0.15
0.1
0.05
0
Score
2500
5000
7500
10000
12500
15000
17500
20000
22500
25000
Training Instances

Random Forest

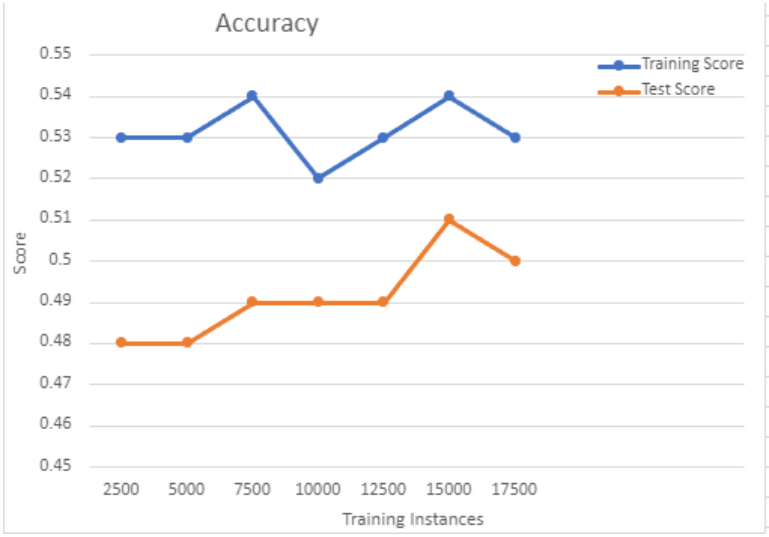
Ο Random Forest, είναι ένας αλγόριθμος που συνδυάζει ταξινομητές που κάνουν διαφορετικά λάθη. Συγκεκριμένα, δημιουργουμε διάφορες παραλλαγες του συνόλου εκπαίδευσης, που έχουν τον ίδιο αριθμό παραδειγμάτων με το αρχικό. Οι επιλογές των παραδειγμάτων γίνονται με επανατοποθέτηση. Χωρίζουμε επίσης το σύνολο ιδιοτήτων σε μικρότερα υποσύνολα, αλλα αυτη τη φορα, κρατάμε ένα μικρότερο αριθμό ιδιοτήτων(ίδιο για όλα τα υποσύνολα) για κάθε υποσύνολο.
Αφού ετοιμάσουμε τα σύνολα που θα χρησιμοποιήσουμε, καλούμε ανάλογα με τα δέντρα που θα φτιάξουμε τόσες φορές τον ταξινομητή, και καθε αποτέλεσμα που προκύπτει το τοποθετούμε σε έναν πίνακα. Έπειτα, αφού τελειώσει η διαδικασια με τον ταξινομητή, παίρνουμε τις επικρατέστερες τιμές από κάθε πίνακα.
Σημαντικές παράμετροι που παίζουν ρόλο στον Random Forest, είναι το πλήθος δέντρων που θα υλοποιήσει, και ο βαθμός στον οποίο θα γίνουν οι τυχαιες επιλογες στα υποσύνολα ετσι ωστε να μην δημιουργούνται τα ίδια μονοπάτια στον ταξινομητή.
Στα παραδείγματα μας, λόγω χρόνου, κρατούσαμε το 30% των παραδειγμάτων και δημιουργουσαμε λιγα δέντρα.

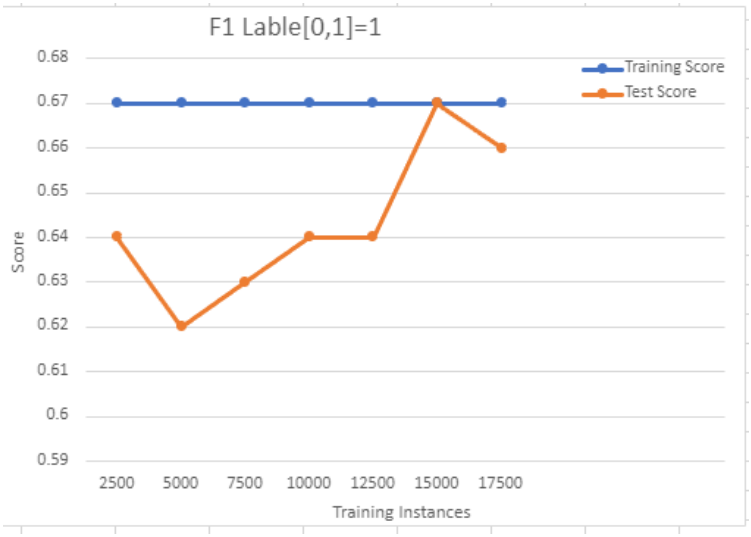
Αποτελέσματα:
Χρησιμοποιώντας τον ID3 που υλοποιήσαμε, η καλύτερη ακρίβεια που πετύχαμε είναι 54% για 7500 στοιχεία στα δεδομένα εκπαίδευσης. Στα νέα δεδομένα, η καλύτερη ακρίβει είναι 71% για 12000 στοιχεία.
Παρακάτω, φαίνονται οι καμπύλες ορθότητας, ακρίβειας και ανάκλησης.
Στην πρώτη εικόνα, φαίνεται το score σε συναρτηση με τα παραδείγματα του Random Forest με τον έτοιμο ID3 της βιβλιοθήκης.
Χρησιμοποιώντας τον έτοιμο ID3, η καλύτερη ακρίβεια που πετύχαμε είναι 0.52% για 22500 στοιχεία.

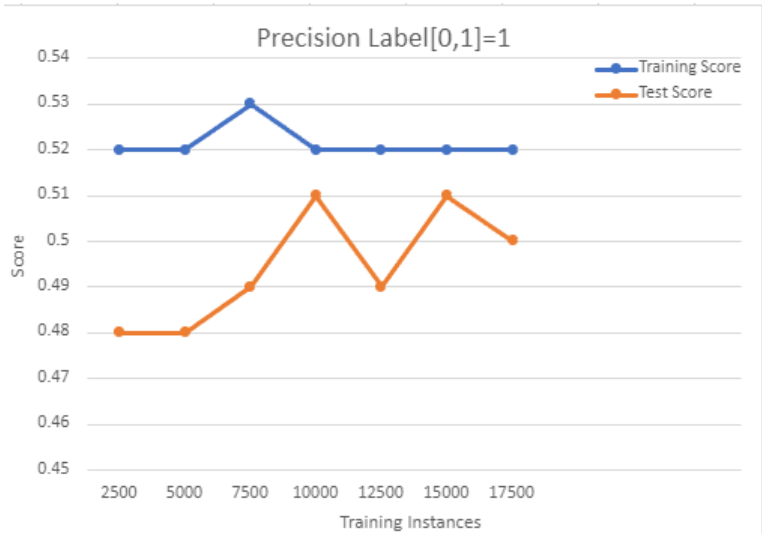
Ακολουθεί η καμπύλη του έτοιμου ID3:

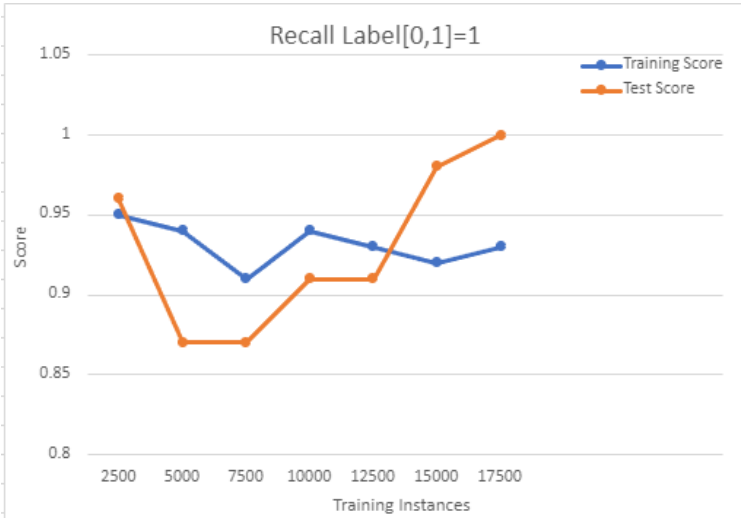


Ακολουθούν οι καμπύλες στα νεα δεδομένα, του Random Forest με τον δικό μας ID3:

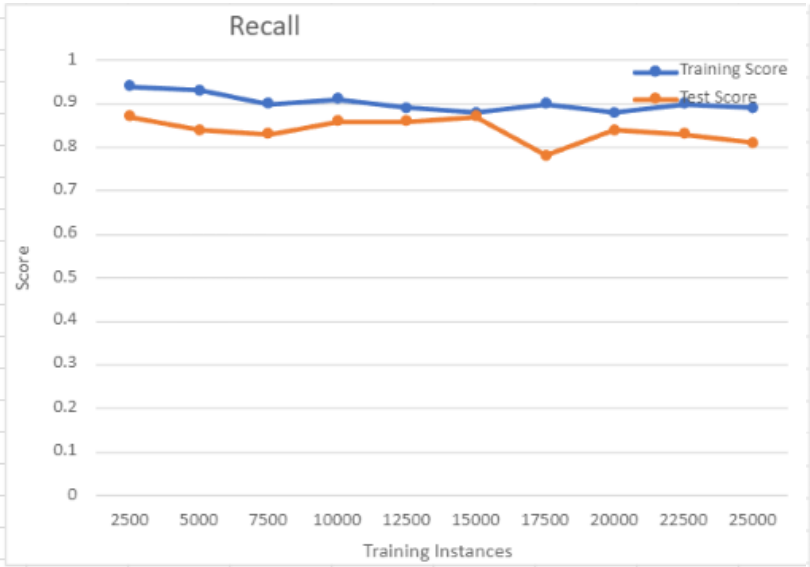


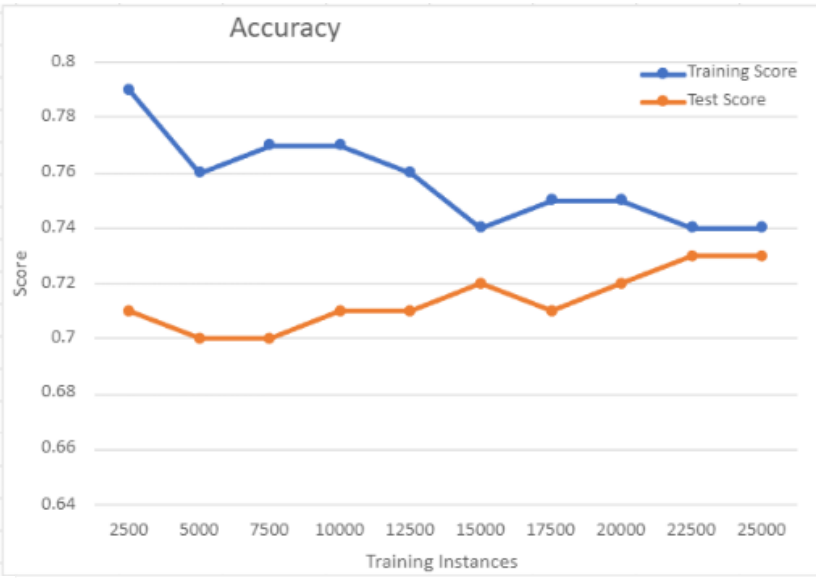


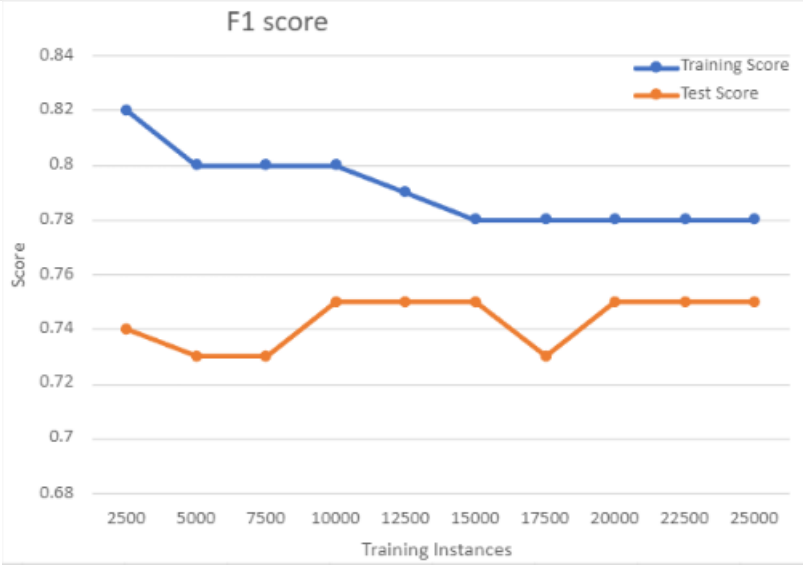




Ακολουθούν οι καμπύλες στα νέα δεδομένα με τον έτοιμο ID3:







Παρατήρηση: Στην υλοποιηση του Random Forest με την έτοιμη ID3 (Decision Tree) παρατηρούμε αύξηση στην ακρίβεια 10%

Σημείωση: Για να γίνει import ο id3 μεσα στον κωδικα του Random Forest, θα πρεπει να γινει στο colab κατα τη διαρκεια εκτελεσης του προγραμματος. Δεν γινεται αυτοματα. Γι' αυτόν τον λόγο αντιγράψαμε τον id3 μέσα στο αρχείο.