2η Εργασία Τεχνητή Νοημοσύνη

Μιχαήλ Πρωτονοτάριος 3200164
Μάριος Γεωργοπετρέας 3200028

Για την επεξεργασία των δεδομένων (reviews) φτιάξαμε μια κλάση tokenizer που ουσιαστικά κάνει το tokenization των reviews σε «λέξεις κλειδία» (features), δηλαδή μετατρέπει κάθε review σε feature vector, όπου έχει 1 στην θέση ενός feature αν αυτό υπάρχει στο κείμενο, αλλιώς 0 αν δεν υπάρχει.
Και για τους 2 αλγορίθμους που είχαμε αναπτύξει, για το φόρτωμα των δεδομένων (πριν κάνουμε initialize τον tokenizer), το πρόγραμμα ζητάει από τον χρήστη να δώσει ως input τα paths των positive και negative training και test data αντίστοιχα.
Χρησιμοποιήσαμε 27 features (λέξεις του λεξιλογίου), φαίνονται κάποιες από αυτές στο παρακάτω pandas dataframe. Καταλήξαμε σε αυτές τις 27 λέξεις φιλτράροντας τα κείμενα από σημεία στίξης, κενά και άλλα άχρηστα strings, έπειτα πήραμε τις 700 πιο συχνές λέξεις από όλα τα κείμενα εκπαίδευσης (αρνητικά και θετικά) και κρατήσαμε αυτές που είχαν information gain μεγαλύτερο ή ίσο του 0.008.
Έχουμε φτιάξει επίσης μια συνάρτηση classification_report που τυπώνει recall, precision, f1-score και accuracy για κάθε κατηγορία, καθώς και το general accuracy του αλγορίθμου (τα τυπώνει όλα ως dataframe).
Τέλος, χρησιμοποιούμε και μία συνάρτηση draw_diagram για να σχεδιάσουμε τα διαγράμματα για ότι χρειαζόμαστε. Η συνάρτηση αυτή δέχεται ως παραμέτρους το object που χρησιμοποιήσαμε για να υλοποιήσουμε τον κάθε αλγόριθμο του οποίου τα στατιστικά θέλουμε να σχεδιάσουμε (πχ nb για naïve bayes ή rf για random forest) και ένα string με το όνομα του στατιστικού που θέλουμε να σχεδιάσει (πχ accuracy, recall, precision κλπ.).

	just	even	well	great	bad	don't	best	plot	love	nothing	...	awful	terrible	perfect	supposed	waste	loved	worse	favorite	horrible	Category
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	0	0	0	0	1
2	0	1	1	1	1	0	1	0	0	0	...	0	0	1	0	0	0	0	0	0	1
3	0	0	0	0	1	0	0	0	1	0	...	0	0	0	0	0	1	0	1	0	1
4	1	1	1	1	0	0	1	0	0	0	...	1	1	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
24995	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
24996	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
24997	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
24998	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
24999	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

25000 rows × 28 columns

Naïve Bayes algorithm:

Για την υλοποίηση του αλγορίθμου αυτού χρησιμοποιήσαμε μια κλάση Probability, η οποία με βάση το παραπάνω dataframe για τα δεδομένα μας δημιουργεί 2 πίνακες 2X2 τους pX_1 και pX_0, όπου για παράδειγμα το pX_1[0][2] περιέχει την πιθανότητα η $3^η$ λέξη να είναι 1 όταν η κατηγορία είναι 0 και αντίστοιχα το pX_0[0][2] περιέχει την πιθανότητα η $3^η$ λέξη να είναι 0 όταν η κατηγορία είναι 0. Αυτό το κάνουμε έτσι ώστε να μην χρειάζεται να υπολογίζουμε κάθε φορά για κάθε νέα πρόβλεψη τις πιθανότητες της κάθε λέξης.
(Λέγοντας η λέξη να είναι 0 εννοούμε ότι δεν υπάρχει στο συγκεκριμένο παράδειγμα και αντίστοιχα για το 1, όπως επίσης κατηγορία 0 είναι η αρνητική κριτική και 1 η θετική)

a)

Stats of our implementation

- Training data report

```
classification_report(y_train, y_pred)
```
[13] ✓ 0.0s

```
positives that didn't find:  0
negatives that didn't find:  65
```

	Recall	Precision	F1-score	Accuracy	Support
0	1.0000	0.994827	0.997407	1.0000	12500.0
1	0.9948	1.000000	0.997393	0.9948	12500.0
general accuracy	0.9974	0.997400	0.997400	0.9974	25000.0

- Test data report

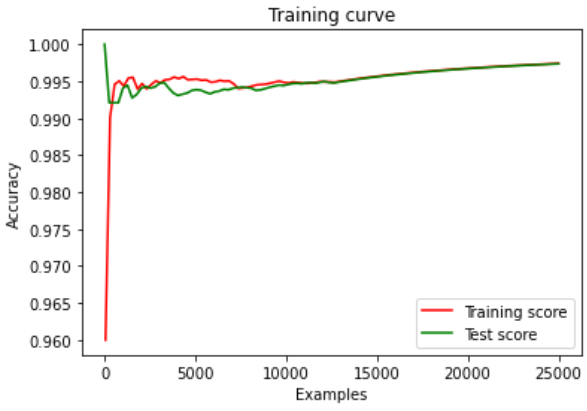
```
classification_report(y_test, y_test_pred)
```
[16] ✓ 0.0s

```
positives that didn't find:  0
negatives that didn't find:  66
```

	Recall	Precision	F1-score	Accuracy	Support
0	1.00000	0.994748	0.997367	1.00000	12500.0
1	0.99472	1.000000	0.997353	0.99472	12500.0
general accuracy	0.99736	0.997360	0.997360	0.99736	25000.0

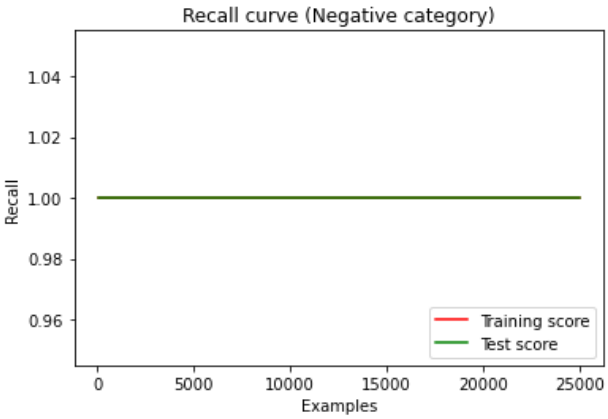
Training Curve (Accuracy)

Αυτή η καμπύλη εκπαίδευσης δείχνει το συνολικό ποσοστό ορθότητας (accuracy) και στις δύο κατηγορίες μαζί (δηλαδή τη συνολική ορθότητα και σε θετικά και σε αρνητικά reviews)



Recall Curve (Negative category)

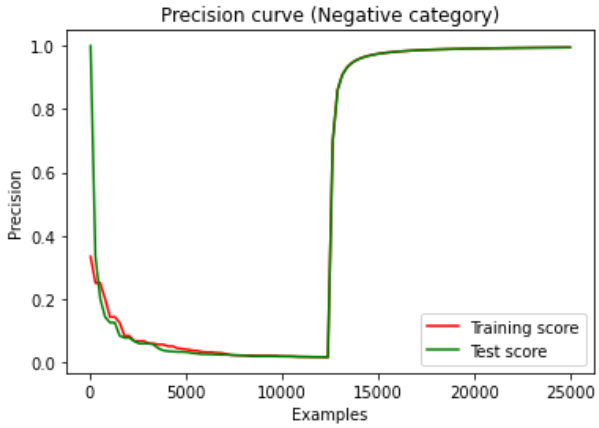
Τα recall των test data και training data ταυτίζονται και είναι σταθερά στο 1

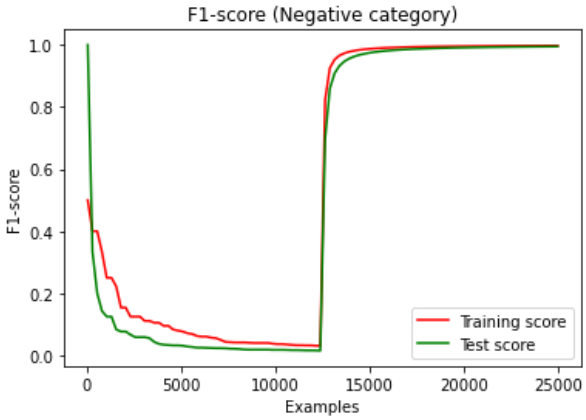


Precision curve (Negative category)

Η καμπύλη έχει αυτήν την μορφή γιατί αρχικά τα πρώτα παραδείγματα που διαβάζουμε είναι positive και στα 12500 όπου τα positive τελειώνουν και ξεκινούν τα negative αρχίζει και ανεβαίνει το precision

F1- Curve (Negative Category)





b)

Σύγκριση των στατιστικών μας με τα στατιστικά της υλοποίησης του naïve bayes algorithm από την βιβλιοθήκη sklearn

Παρατηρούμε ότι τα στατιστικά της υλοποίησης μας σε σχέση με την υλοποίηση της sklearn είναι σχεδόν ίδια και στις δύο κατηγορίες (negative positive) σε όλα τα metrics (precision, recall, f1-score και accuracy)

SKlearn implementation

```
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report as skl_classification_report

nb = GaussianNB()
nb.fit(x_train, y_train)
print(skl_classification_report(y_train, nb.predict(x_train),
                                zero_division=1))
```

[30] ✓ 0.0s

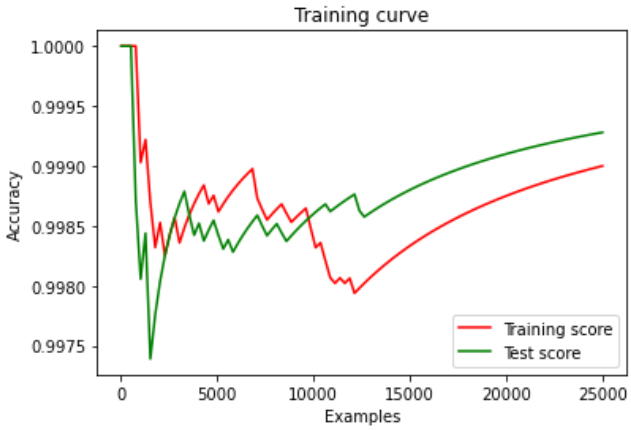
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     12500
           1       1.00      1.00      1.00     12500

    accuracy                           1.00     25000
   macro avg       1.00      1.00      1.00     25000
weighted avg       1.00      1.00      1.00     25000
```

```
print(skl_classification_report(y_test, nb.predict(x_test),
                                zero_division=1))
```

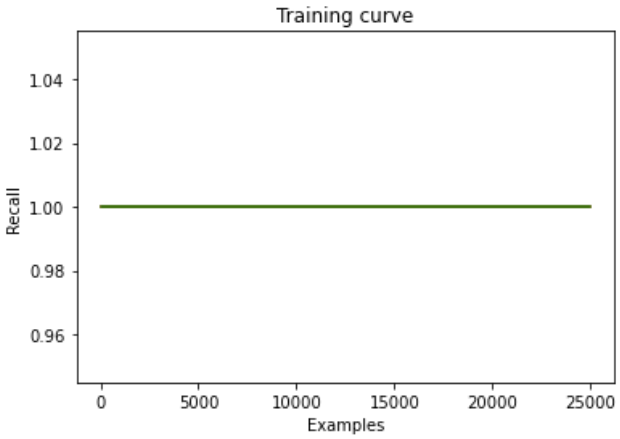
[31] ✓ 0.0s

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     12500
           1       1.00      1.00      1.00     12500

    accuracy                           1.00     25000
   macro avg       1.00      1.00      1.00     25000
weighted avg       1.00      1.00      1.00     25000
```

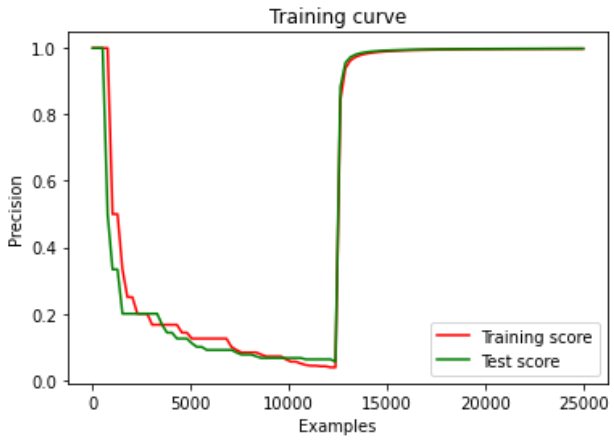
Training curve (Accuracy):



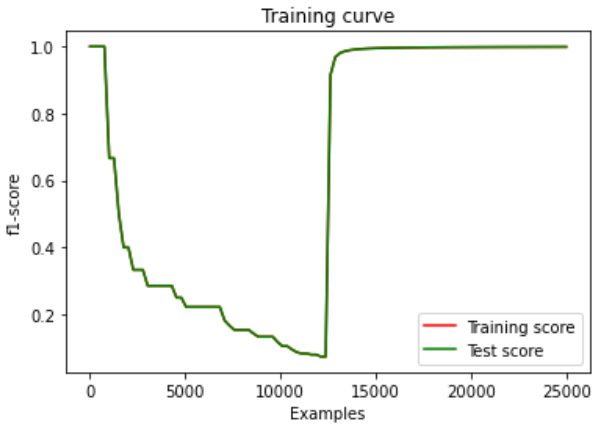
Recall curve:



Precision curve:



F1 curve:



Random Forest algorithm:

Για την υλοποίηση του random forest αρχικά φτιάξαμε μία κλάση decision_tree όπου κατασκευάζει τα δέντρα του δάσους. Έχει μία μέθοδο build_tree η οποία κατασκευάζει το δέντρο λαμβάνοντας υπόψην της το max depth που δίνεται ως είσοδος από τον χρήστη. Η build_tree καλεί την get_best_split η οποία διαλέγει κάθει φορά από την λίστα με τα διαθέσιμα features (τα οποία επιλέγονται τυχαία για κάθε δέντρο), με βάση ποιο feature θα συνεχιστεί η κατασκευή του δέντρου υπολογίζοντας για κάθε διαθέσιμο feature το ig του.
Το random forest λοιπόν κατασκευάζει τόσα δέντρα όσα και η είσοδος που βάζει ο χρήστης και με όσα features θέλει επίσης ο χρήστης για το κάθε δέντρο.

a)

Χρησιμοποιήσαμε τα ίδια 27 features για τα δεδομένα μας (όπως στον naïve bayes παραπάων). Ως υπερπαραμέτρους χρησιμοποιήσαμε 150 δέντρα με 10 features για το κάθε ένα.

Stats of our implementation:

- **Training data report:**

```
classification_report(y_train, y_pred)
```

[16] ✓ 0.0s

	Recall	Precision	F1-score	Accuracy	Support
0	1.00000	0.968092	0.983787	1.00000	12500.0
1	0.96704	1.000000	0.983244	0.96704	12500.0
general accuracy	0.98352	0.983520	0.983520	0.98352	25000.0

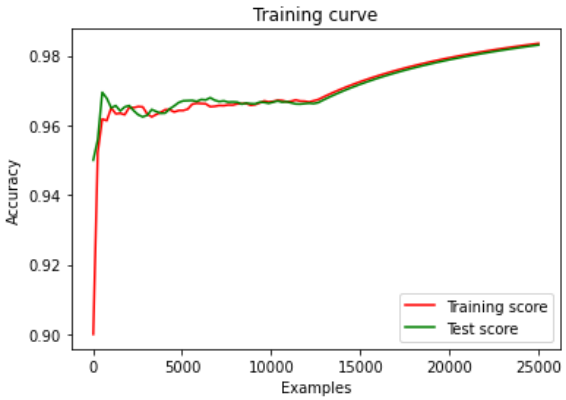
- **Test data report:**

```
classification_report(y_test, y_test_pred)
```

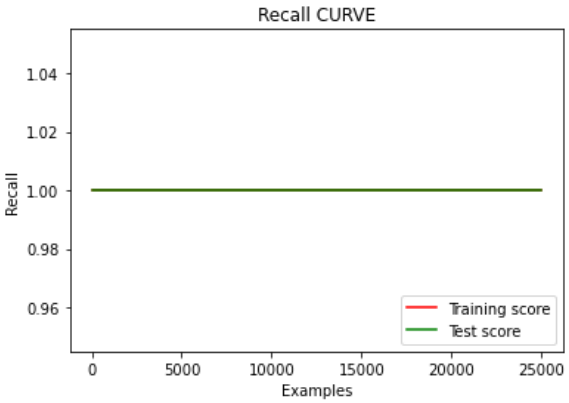
[19] ✓ 0.0s

	Recall	Precision	F1-score	Accuracy	Support
0	1.00000	0.967268	0.983362	1.00000	12500.0
1	0.96616	1.000000	0.982789	0.96616	12500.0
general accuracy	0.98308	0.983080	0.983080	0.98308	25000.0

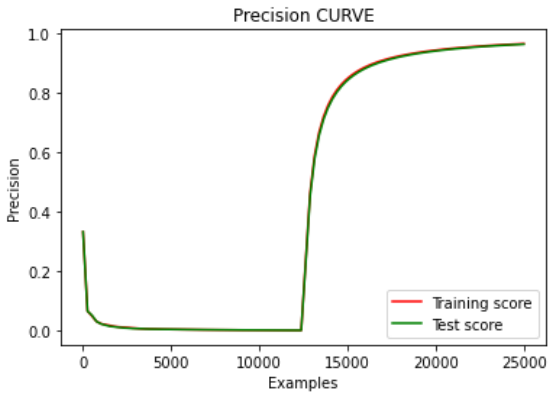
Training curve:



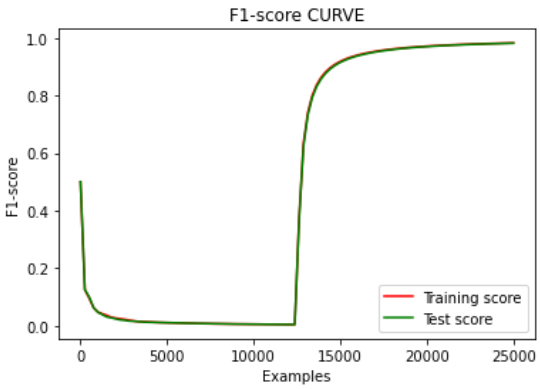
Recall Curve:



Precision Curve:



F1-score Curve:



b)

Σύγκριση των στατιστικών μας με τα στατιστικά της υλοποίησης του random forest algorithm από την βιβλιοθήκη Sklearn

Παρατηρούμε ότι οι διαφορές στα στατιστικά της βιβλιοθήκης Sklearn σε σχέση με την δική μας υλοποίηση είναι πολύ μικρές, με την υλοποίηση της Sklearn να είναι περίπου κατά 2% πιο ακριβής, και στα test data και στα training data

```
classification_report(y_train, rf2.predict(x_train))
```

[25] ✓ 1.6s

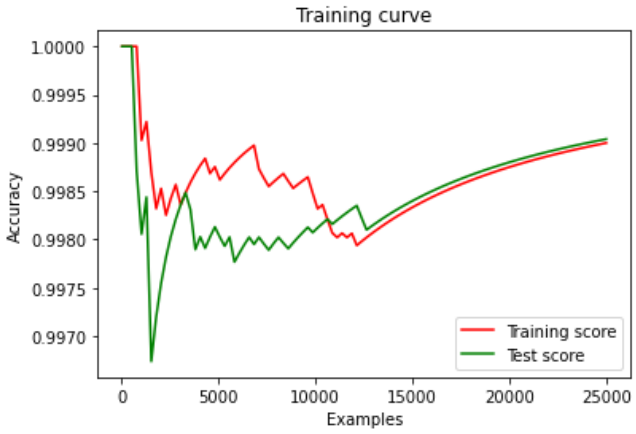
	Recall	Precision	F1-score	Accuracy	Support
0	1.000	0.998004	0.999001	1.000	12500.0
1	0.998	1.000000	0.998999	0.998	12500.0
general accuracy	0.999	0.999000	0.999000	0.999	25000.0

```
classification_report(y_test, rf2.predict(x_test))
```

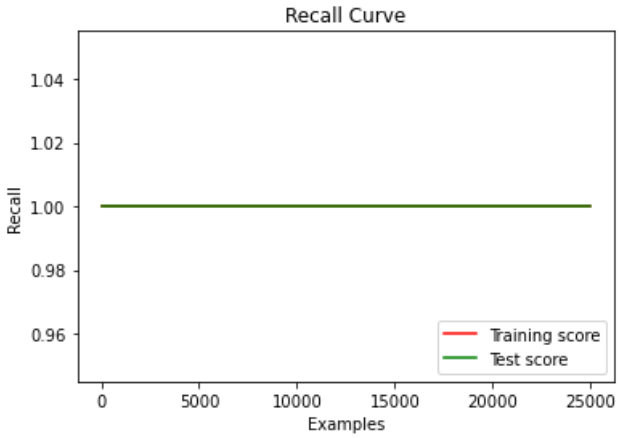
[26] ✓ 0.2s

	Recall	Precision	F1-score	Accuracy	Support
0	1.00000	0.998084	0.999041	1.00000	12500.0
1	0.99808	1.000000	0.999039	0.99808	12500.0
general accuracy	0.99904	0.999040	0.999040	0.99904	25000.0

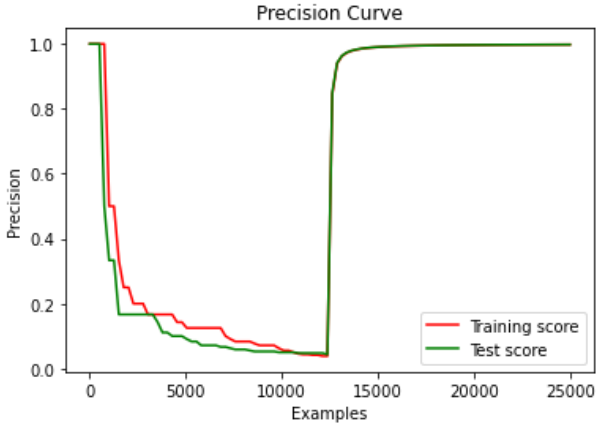
Training Curve (Accuracy):



Recall Curve:



Precision Curve:



F1-score Curve: