Τεχνητή Νοημοσύνη – 2η Εργασία

ΑΝΑΦΟΡΑ ΠΑΡΑΔΟΣΗΣ

(**Όνομα:** Αναστάσιος Παπαπαναγιώτου, **ΑΜ:** 3200143, **Email:** p3200143@aueb.gr)

(**Όνομα:** Φοίβος Παπαθανασίου, **ΑΜ:** 3200138, **Email:** p3200138@aueb.gr)

Τρόπος Χρήσης:

random_search.py : Τρέχουμε τη συνάρτηση search(...) με παράμετρο τον αριθμό των processes και μέσα στη συνάρτηση επιλέγουμε πλήθος διαφορετικών τιμών των μεταβλητών m και λ.

main.py : Τρέχουμε τη main και δίνουμε τιμές στις μεταβλητές m, n, k.

Αρχεία:

- utils.py

  Συναρτήσεις:

  preprocess_data(n, m, k) – Φορτώνει τα δεδομένα από το imdb dataset και (1)μετατρέπει κάθε review σε διάνυσμα ιδιοτήτων με τιμές 0 ή 1, οι οποίες δείχνουν ποιες λέξεις του λεξιλογίου περιέχει το review. Το λεξιλόγιο περιλαμβάνει τις m συχνότερες λέξεις των δεδομένων εκπαίδευσης, παραλείποντας πρώτα τις n πιο συχνές και τις k πιο σπάνιες λέξεις των δεδομένων εκπαίδευσης.

  Επιστρέφει λίστες x_train, x_test όπου περιέχουν τα reviews σε μορφή λίστας με indexes, επίσης επιστρέφει λίστες y_train, y_test με τιμές 0 ή 1 ανάλογα με το αν το review ήταν θετικό ή αρνητικό, τέλος, επιστρέφει πίνακες διανυσμάτων x_train_binary και x_test_binary όπου περιέχουν τα διανύσματα ιδιοτήτων που προέκυψαν απ'τη μετατροπή (1).

  calculate_metrics(x, y, training, predict_func, clf=None) – Υπολογισμός και επιστροφή σφάλματος (error) για τον υπολογισμό της ορθότητας, υπολογισμός και επιστροφή ακρίβειας (precision) και ανάκλησης (recall) ενός μοντέλου με βάση τα δοθέντα σύνολα δεδομένων.

  display_metrics(metrics, perc, title, model_title) – Υπολογισμός ορθότητας, F1 μοντέλου και κατασκευή και display πίνακα με metrics: accuracy, precision, recall και F1 ενός μοντέλου συναρτήσει του πλήθος των παραδειγμάτων. Εμφάνιση καμπύλων precision και recall μοντέλου συναρτήσει του πλήθους αποτελεσμάτων.

- naive_bayes.py
  Υλοποίηση Naïve Bayes σε multivariable form Bernoulli με εκτιμήτρια Laplace. Κατασκευή accuracy, precision, recall, F1 graphs και αντίστοιχων tables συναρτήσει πλήθους παραδειγμάτων κάθε πειράματος με χρήση του υλοποιημένου μοντέλου και χρήση του έτοιμου μοντέλου της βιβλιοθήκης sklearn.

  Συναρτήσεις:

  evaluate_bayes(training_set, test_set, perc, no_graph=False) – Πραγματοποιεί πειράματα στα σύνολα δεδομένων χρησιμοποιώντας την υλοποίηση Naïve Bayes σε πολυμεταβλητή μορφή Bernoulli.

  Υπολογίζει και κατασκευάζει καμπύλες μάθησης και αντίστοιχους πίνακες που δείχνουν το **ποσοστό σφάλματος** στα δεδομένα εκπαίδευσης και ελέγχου συναρτήσει του πλήθους των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται σε κάθε επανάληψη του πειράματος.

  Επίσης, υπολογίζει και κατασκευάζει καμπύλες και πίνακες με αποτελέσματα **ακρίβειας**, **ανάκλησης**, **F1**, συναρτήσει του πλήθους των παραδειγμάτων εκπαίδευσης.

  Το ποσοστό των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται σε κάθε πείραμα μεταβάλλεται με βάση την παράμετρο perc.

  Παράλληλα, υπολογίζει και κατασκευάζει τα παραπάνω χρησιμοποιώντας την υλοποίηση Naïve Bayes σε πολυμεταβλητή μορφή Bernoulli της βιβλιοθήκης **sklearn** έτσι ώστε να είναι εφικτή η σύγκριση των δύο υλοποιήσεων.

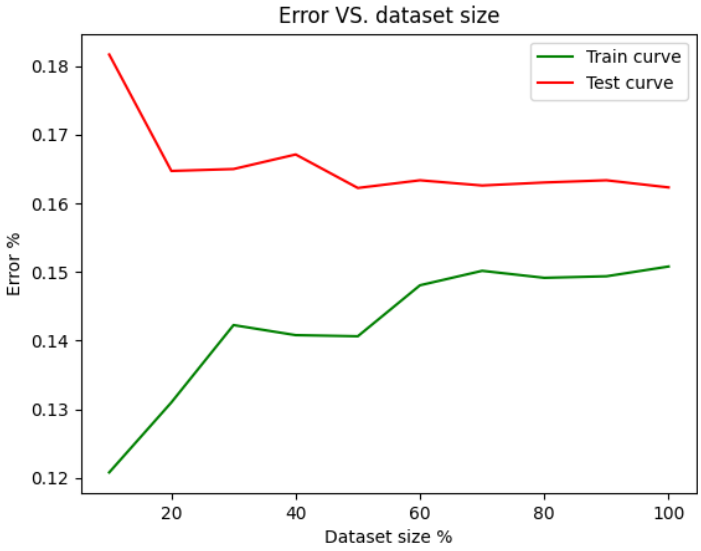
  train_naive_bayes(dataset) – Υλοποίηση Naïve Bayes (πολυμεταβλητή μορφή Bernoulli) με εκτιμήτρια Laplace.

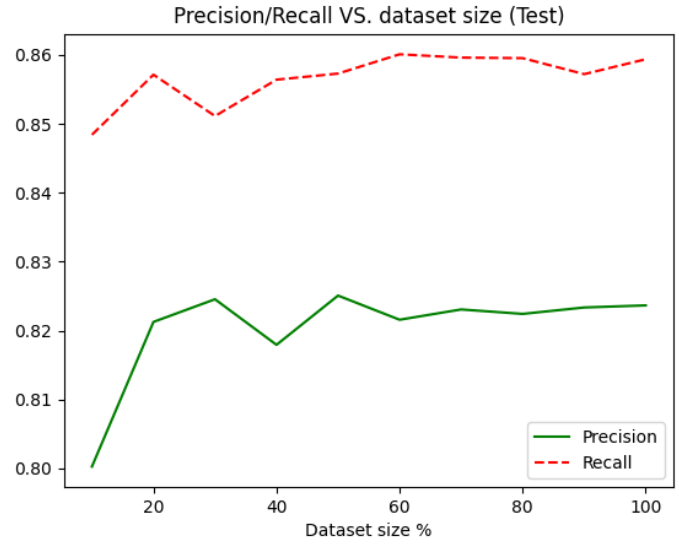
  test_naive_bayes(x, training) – Συνάρτηση ταξινόμησης.

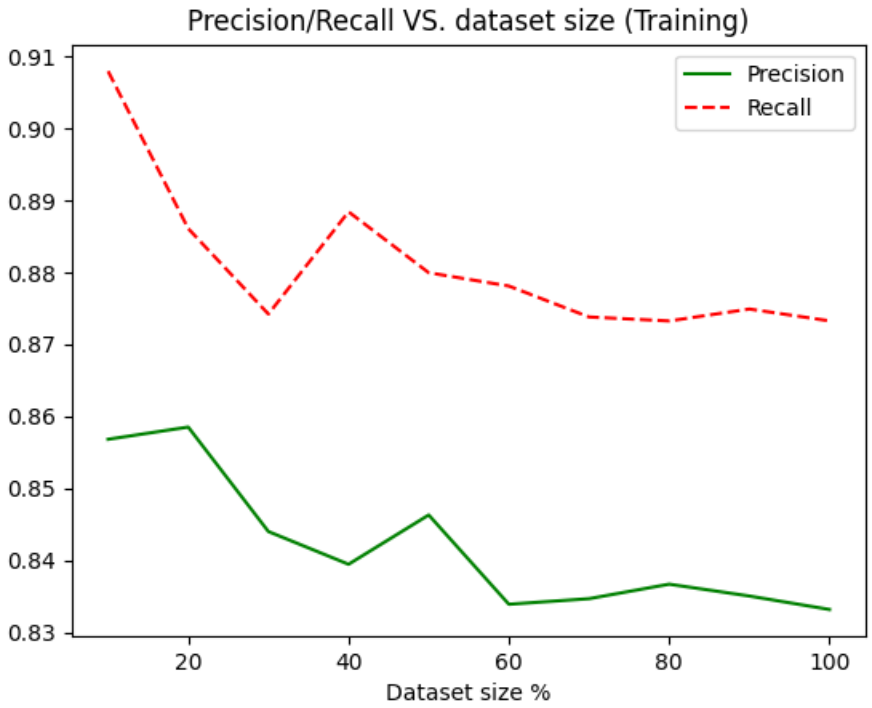
```
=== Metrics for Training ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%      100%
Accuracy  0.879200  0.869000  0.857733  0.859200  0.859360  0.851933  0.849829  0.850850  0.850622  0.849200
Precision 0.856816  0.858516  0.843994  0.839441  0.846295  0.833883  0.834657  0.836674  0.835025  0.833155
Recall    0.907990  0.886111  0.874226  0.888445  0.879975  0.878124  0.873818  0.873269  0.874922  0.873280
F1        0.881661  0.872095  0.858844  0.863248  0.862806  0.855432  0.853789  0.854580  0.854508  0.852746


=== Metrics for Training (sklearn) ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%      100%
Accuracy  0.879200  0.869000  0.857733  0.859200  0.859360  0.851933  0.849829  0.850850  0.850622  0.849200
Precision 0.856816  0.858516  0.843994  0.839441  0.846295  0.833883  0.834657  0.836674  0.835025  0.833155
Recall    0.907990  0.886111  0.874226  0.888445  0.879975  0.878124  0.873818  0.873269  0.874922  0.873280
F1        0.881661  0.872095  0.858844  0.863248  0.862806  0.855432  0.853789  0.854580  0.854508  0.852746


=== Metrics for Test ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%      100%
Accuracy  0.818320  0.835280  0.835000  0.832880  0.837760  0.836640  0.837400  0.836960  0.836640  0.837680
Precision 0.800257  0.821248  0.824537  0.817925  0.825069  0.821565  0.823056  0.822413  0.823344  0.823647
Recall    0.848400  0.857120  0.851120  0.856400  0.857280  0.860080  0.859600  0.859520  0.857200  0.859360
F1        0.823625  0.838801  0.837618  0.836720  0.840866  0.840381  0.840931  0.840557  0.839931  0.841124


=== Metrics for Test (sklearn) ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%      100%
Accuracy  0.818320  0.835280  0.835000  0.832880  0.837760  0.836640  0.837400  0.836960  0.836640  0.837680
Precision 0.800257  0.821248  0.824537  0.817925  0.825069  0.821565  0.823056  0.822413  0.823344  0.823647
Recall    0.848400  0.857120  0.851120  0.856400  0.857280  0.860080  0.859600  0.859520  0.857200  0.859360
F1        0.823625  0.838801  0.837618  0.836720  0.840866  0.840381  0.840931  0.840557  0.839931  0.841124
```

Γραφήματα:

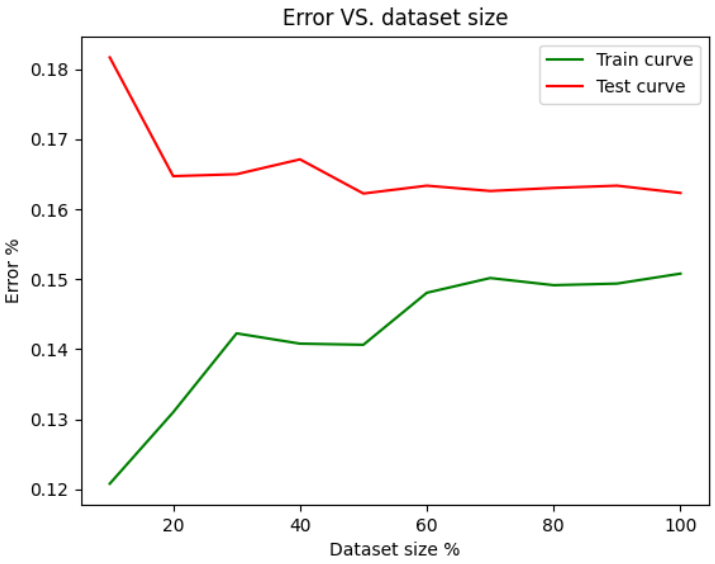
δική μας υλοποίηση:

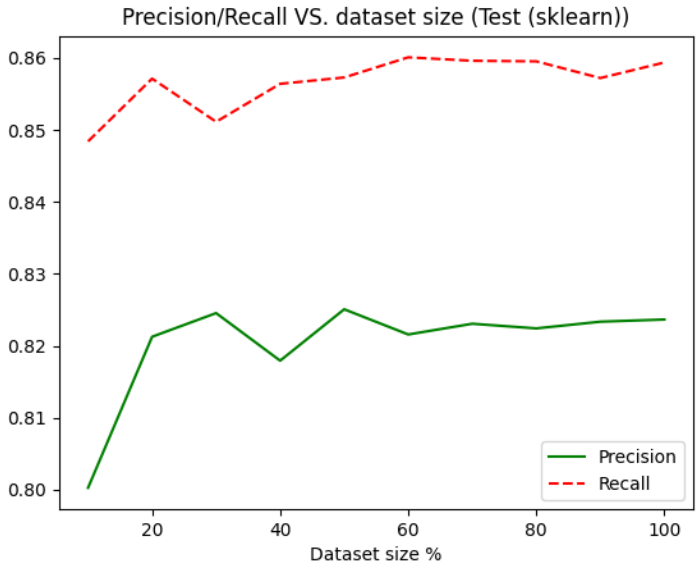


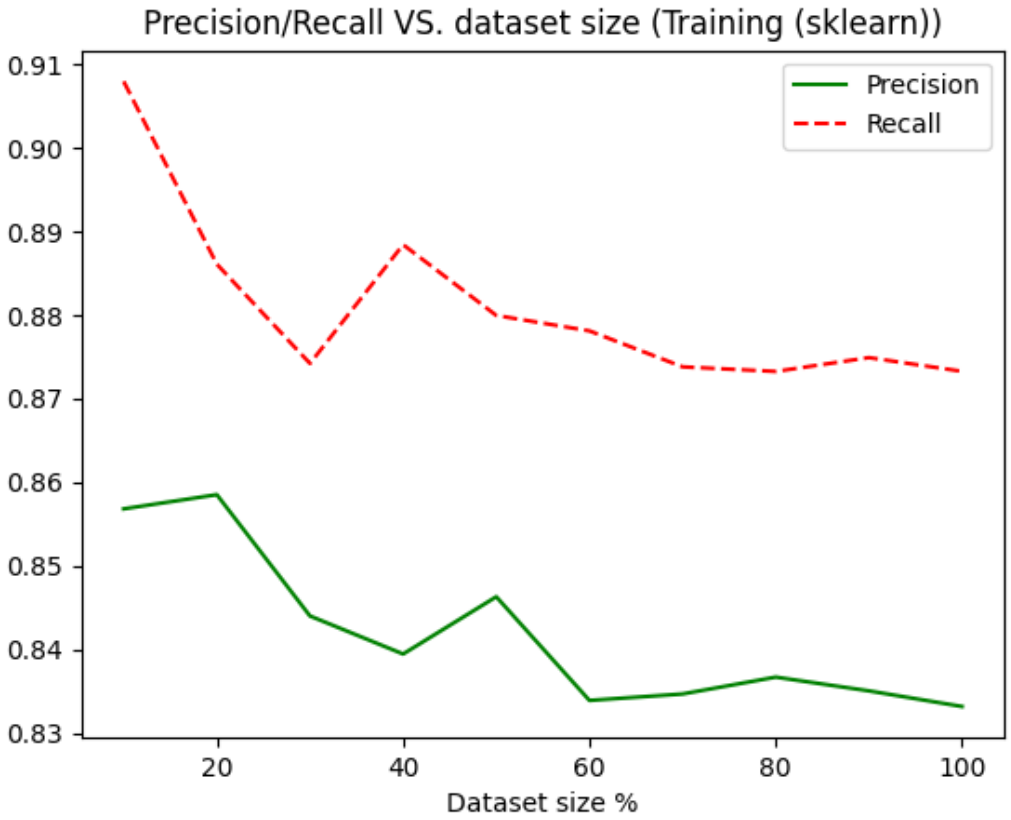




sklearn:


Error VS. dataset size
Train curve
Test curve
Error %
Dataset size %
0.18
0.17
0.16
0.15
0.14
0.13
0.12
20
40
60
80
100


Precision/Recall VS. dataset size (Test (sklearn))
Precision
Recall
Dataset size %
0.86
0.85
0.84
0.83
0.82
0.81
0.80
20
40
60
80
100


Precision/Recall VS. dataset size (Training (sklearn))
Precision
Recall
Dataset size %
0.91
0.90
0.89
0.88
0.87
0.86
0.85
0.84
0.83
20
40
60
80
100

- log_reg.py
  Υλοποίηση Logistic Regression με regularization στο objective function και με stochastic gradient ascent με mini-batches. Κατασκευή accuracy, precision, recall, F1 graphs και αντίστοιχων tables συναρτήσει πλήθους παραδειγμάτων κάθε πειράματος με χρήση του υλοποιημένου μοντέλου και χρήση του έτοιμου μοντέλου της βιβλιοθήκης sklearn.

  Συναρτήσεις:

  evaluate_logistic_regression(training_set, test_set, percentage_increase, max_iter, learning_rate=0.01, lambda_=0.001, no_graph=False) –

  Ομοίως με την evaluate_bayes(...) (βλ. σελ 1) πραγματοποιεί πειράματα στα σύνολα δεδομένων χρησιμοποιώντας την υλοποίηση Λογιστικής Παλινδρόμησης με regularization στην αντικειμενική συνάρτηση και με στοχαστική ανάβαση κλίσης με mini-batches.

  Υπολογίζει και κατασκευάζει καμπύλες μάθησης και πίνακες με αποτελέσματα **ακρίβειας**, **ανάκλησης**, **F1** και **ποσοστό σφάλματος** στα δεδομένα εκπαίδευσης και ελέγχου συναρτήσει του πλήθους των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται σε κάθε επανάληψη του πειράματος.

  Το ποσοστό των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται σε κάθε πείραμα μεταβάλλεται με βάση την παράμετρο percentage_increase.

  Παράλληλα, υπολογίζει και κατασκευάζει τα παραπάνω χρησιμοποιώντας την υλοποίηση Logistic Regression της βιβλιοθήκης **sklearn** έτσι ώστε να είναι εφικτή η σύγκριση των δύο υλοποιήσεων.

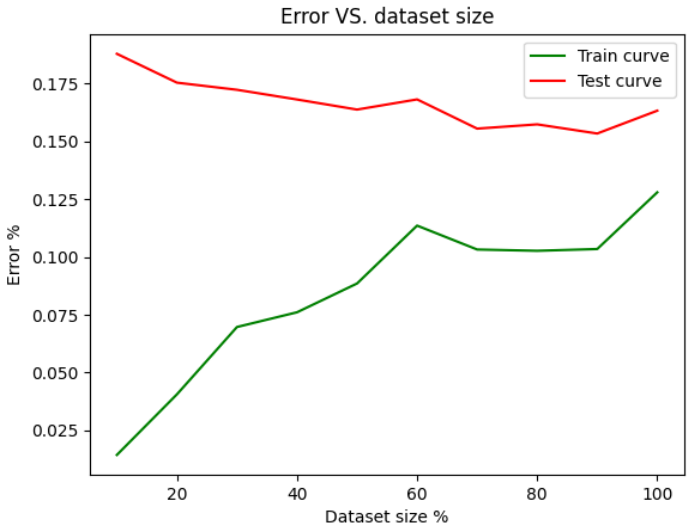
  train(x_train, y_train, batch_size, epochs, learning_rate, lambda_param) – Υλοποίηση Logistic Regression με regularization στην αντικειμενική συνάρτηση και με στοχαστική ανάβαση κλίσης με mini-batches.

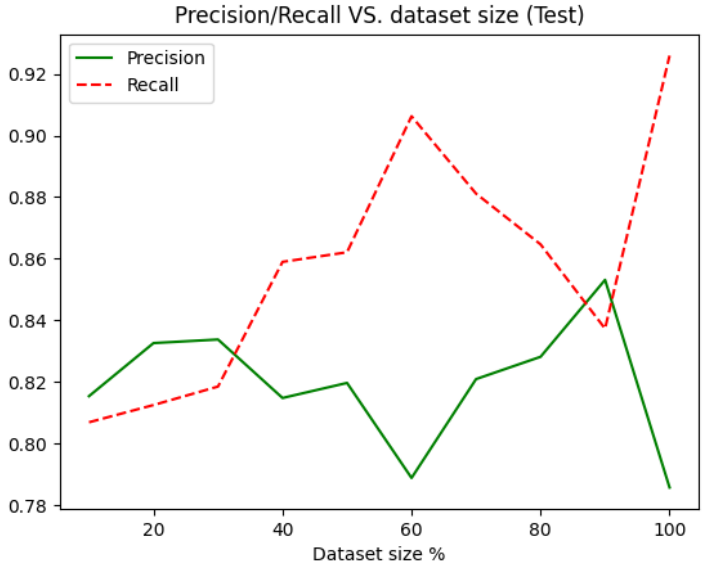
  predict(x, w) – Συνάρτηση ταξινόμησης.

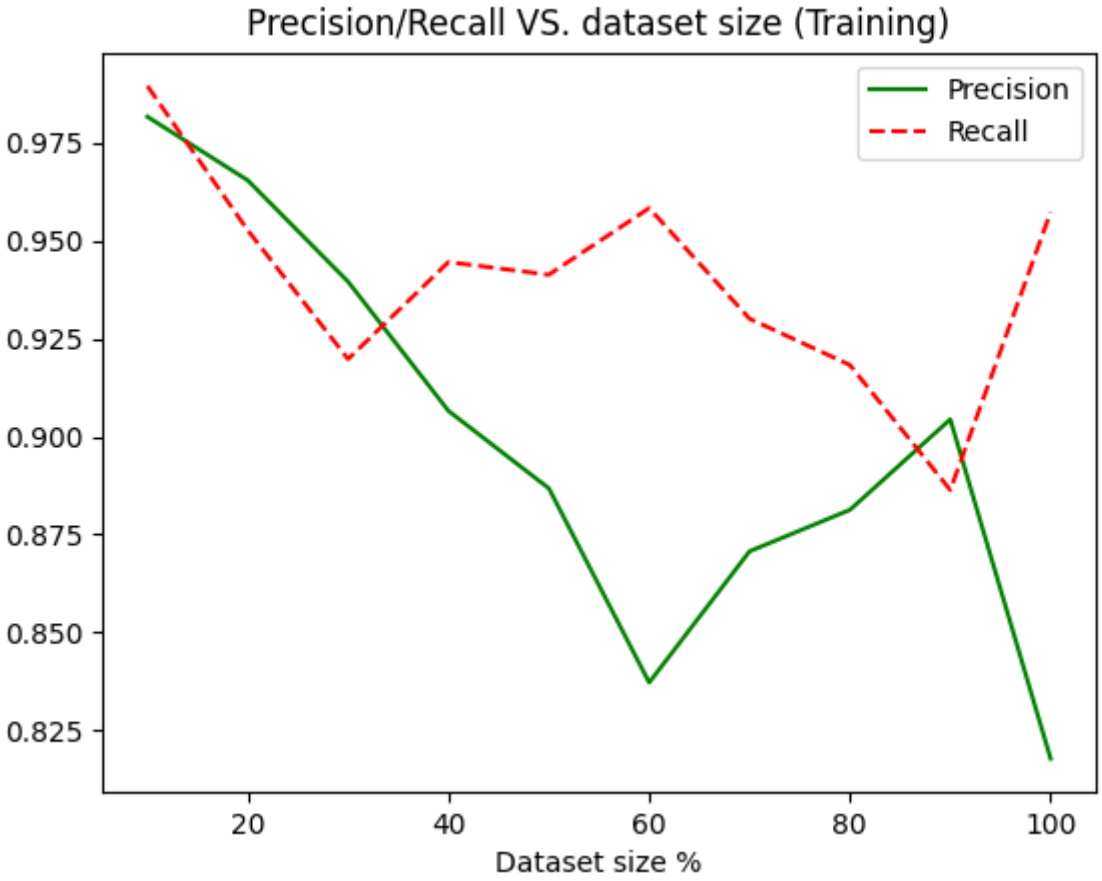
  sigmoid(z) – Παραλείπεται.

Γραφήματα:

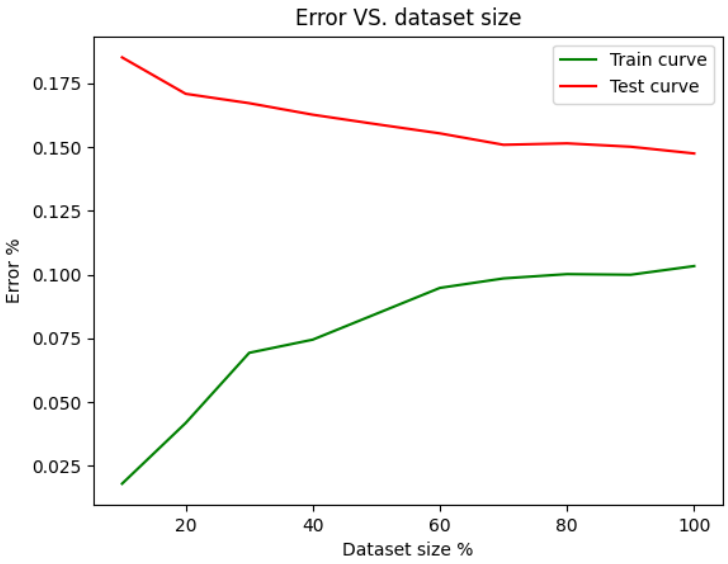
δική μας υλοποίηση:

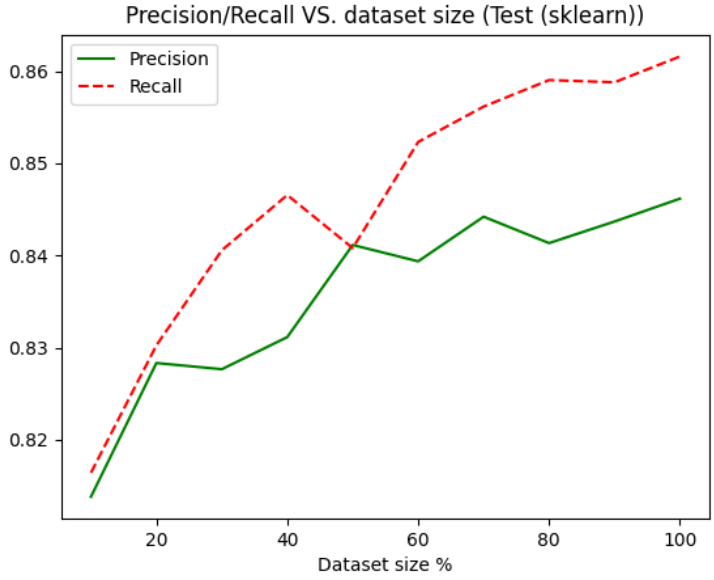


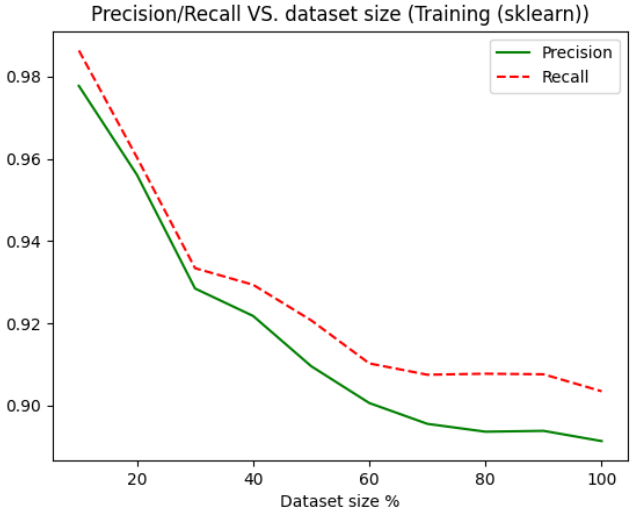




sklearn:








```
=== Metrics for Training ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%       100%
Accuracy   0.985600  0.959400  0.930267  0.923900  0.911440  0.886400  0.896743  0.897300  0.896533  0.872000
Precision  0.981732  0.965489  0.939592  0.906587  0.886810  0.837190  0.870718  0.881309  0.904424  0.817753
Recall     0.989592  0.952724  0.919819  0.944612  0.941376  0.958389  0.930142  0.918316  0.886386  0.957360
F1         0.985646  0.959064  0.929600  0.925209  0.913278  0.893699  0.899449  0.899432  0.895314  0.882067


=== Metrics for Training (sklearn) ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%       100%
Accuracy   0.982000  0.958200  0.930667  0.925500  0.915360  0.905200  0.901486  0.899800  0.900044  0.896640
Precision  0.977778  0.956123  0.928458  0.921775  0.909541  0.900582  0.895514  0.893602  0.893809  0.891318
Recall     0.986389  0.960337  0.933404  0.929360  0.920704  0.910222  0.907469  0.907718  0.907577  0.903440
F1         0.982065  0.958225  0.930925  0.925552  0.915088  0.905377  0.901452  0.900605  0.900641  0.897338


=== Metrics for Test ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%       100%
Accuracy   0.812080  0.824560  0.827640  0.831840  0.836200  0.831800  0.844440  0.842600  0.846560  0.836680
Precision  0.815360  0.832596  0.833754  0.814767  0.819655  0.788803  0.820899  0.828136  0.853114  0.785690
Recall     0.806880  0.812480  0.818480  0.858960  0.862080  0.906240  0.881120  0.864640  0.837280  0.925920
F1         0.811098  0.822415  0.826047  0.836280  0.840332  0.843453  0.849944  0.845994  0.845123  0.850061


=== Metrics for Test (sklearn) ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%       100%
Accuracy   0.814800  0.829080  0.832760  0.837280  0.841000  0.84460  0.849080  0.848520  0.849840  0.852480
Precision  0.813796  0.828318  0.827649  0.831134  0.841136  0.83936  0.844206  0.841338  0.843681  0.846166
Recall     0.816400  0.830240  0.840560  0.846560  0.840800  0.85232  0.856160  0.859040  0.858800  0.861600
F1         0.815096  0.829278  0.834054  0.838776  0.840968  0.84579  0.850141  0.850097  0.851173  0.853813
```

- MLP.py
  Υλοποίηση RNN σε Tensorflow/Keras. Κατασκευή accuracy, precision, recall, F1 graphs και αντίστοιχων tables συναρτήσει πλήθους παραδειγμάτων κάθε πειράματος.

  Συναρτήσεις:

  rnn(train_set, test_set, max_vocab, perc, epochs, no_graph=False) - Πραγματοποιεί πειράματα στα σύνολα δεδομένων χρησιμοποιώντας την υλοποίηση RNN σε Tensorflow/Keras. Οι λέξεις παριστάνονται σαν ενθέσεις λέξεων.

  Υπολογίζει και κατασκευάζει καμπύλες μάθησης και πίνακες με αποτελέσματα **ακρίβειας**, **ανάκλησης**, **F1** και **ποσοστό ορθότητας** στα δεδομένα εκπαίδευσης και ελέγχου συναρτήσει του πλήθους των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται σε κάθε επανάληψη του πειράματος.

  Το ποσοστό των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται σε κάθε πειράμα μεταβάλλεται με βάση την παράμετρο perc.

  create_model(train, max_vocab, epochs) – Υλοποίηση μοντέλου RNN με χρήση Tensorflow/Keras.

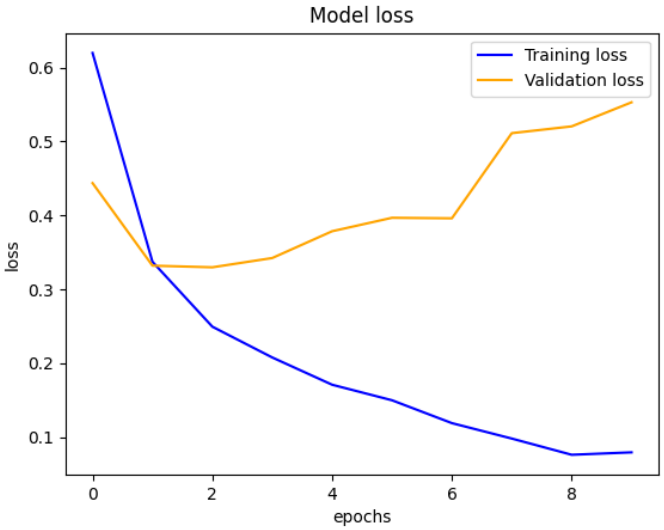
  get_average_length(x_train) – Υπολογίζει και επιστρέφει το μέσο μήκος (πλήθος λέξεων) των reviews που βρίσκονται στη λίστα x_train.

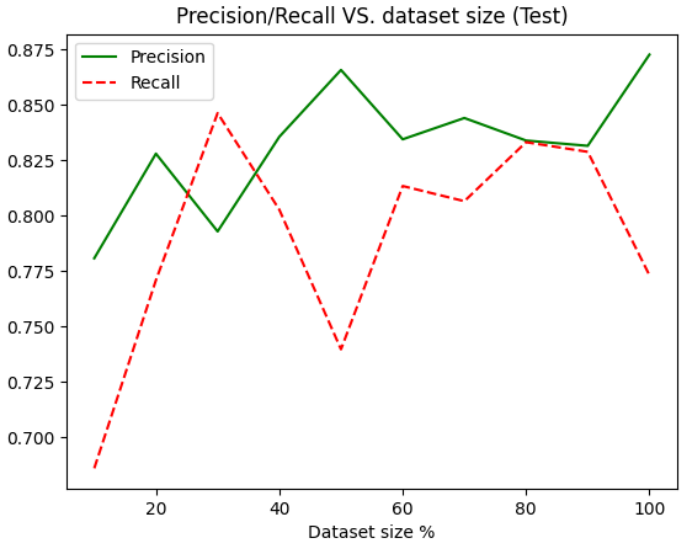
  precision_m(y, y_hat) – Υπολογισμός ακρίβειας μοντέλου RNN συναρτήσει του πλήθους των παραδειγμάτων εκπαίδευσης που χρησιμοποιούνται.

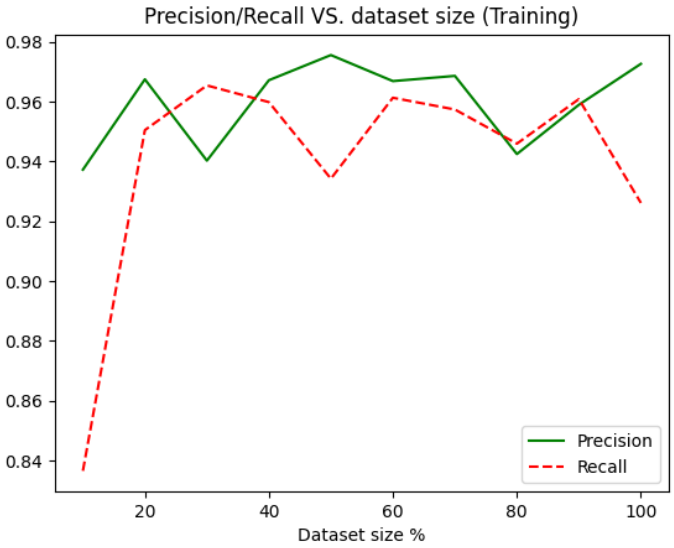
  recall_m(y, y_hat) – Υπολογισμός ανάκλησης μοντέλου RNN συναρτήσει του πλήθους των παραδειγμάτων εκπαίδεσυης που χρησιμοποιούνται.

```
=== Metrics for Training ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%       100%
Accuracy  0.901600  0.960400  0.952133  0.964500  0.955600  0.964200  0.963029  0.945050  0.961156  0.950560
Precision 0.937254  0.967474  0.940272  0.967170  0.975579  0.966873  0.968616  0.942480  0.958960  0.972646
Recall    0.836547  0.950486  0.965427  0.959840  0.934244  0.961310  0.957378  0.945956  0.960851  0.926146
F1        0.884042  0.958904  0.952683  0.963491  0.954464  0.964084  0.962964  0.944215  0.959905  0.948827


=== Metrics for Test ===
              10%       20%       30%       40%       50%       60%       70%       80%       90%       100%
Accuracy  0.746440  0.804560  0.812200  0.822040  0.811600  0.825800  0.828440  0.834040  0.829920  0.829160
Precision 0.780855  0.828037  0.792961  0.835644  0.865796  0.834514  0.844123  0.833956  0.831575  0.872716
Recall    0.686126  0.770987  0.846330  0.802783  0.739807  0.813441  0.806609  0.833211  0.828862  0.773221
F1        0.730432  0.798494  0.818777  0.818884  0.797858  0.823843  0.824940  0.833583  0.830216  0.819961
```

Γραφήματα:







- random_search.py
  Χρησιμοποείται για προσδιορισμό (σχεδόν) βέλτιστον υπερπαραμέτρων. Δοκιμάζει τυχαίους συνδυασμούς υπερπαραμέτρων και επιλέγει των βέλτιστο από αυτούς. Έγινε υλοποίηση με πολυνηματισμό για βελτιστοποίηση ταχύτητας.

  Συναρτήσεις:

generate_random_numbers(n, low, upper, step) – Επιστρέφει αύξουσα ταξινομημένη λίστα με n τυχαίους αριθμούς xi (rounded στα 4 δ.ψ) όπου xi ανήκει στο [low, upper + step) με xi = low + step * k, keZ τ.ω xi < upper+step.

get_combinations(n_m, n_learning_rate, n_lambda) – Επιστρέφει λίστα που περιέχει όλα τα δυνατά tuples που προκύπτουν από το καρτεσιανό γινόμενο των set(n_learning_rate) και set(n_lambda).

thread_search(combinations) – Προσδιορισμός (σχεδόν) βέλτιστων υπερπαραμέτρων με χρήση τεχνικής random search. Υλοποίηση με multithreading για βελτιστοποίηση ταχύτητας (εξετάζονται πολλοί συνδυασμοί υπερπαραμέτρων ταυτόχρονα).

search(num_of_threads) – Παραλείπεται.

- graph.py

  Παραλείπεται.

Προσδιορισμός Υπερπαραμέτρων:

Επιλέξαμε να προσδιορίσουμε τις τιμές των υπερπαραμέτρων m, λ, epochs ενώ τις υπερπαραμέτρους n, k αποφασίσαμε να τις κρατήσουμε σταθερές.

Περιγραφή διαδικασίας: Αρχικά χρησιμοποιήσαμε την τεχνική RandomSearch ώστε να προσδιορίσουμε σχετικά βέλτιστες υπερπαραμέτρους, στη συνέχεια χρησιμοποιώντας τις υπερπαραμέτρους αυτές υπολογίζουμε τα metrics (accuracy, precision, recall, F1) και κατασκευάζουμε γραφήματα ώστε να προσδιορίσουμε μέσω αυτών ακόμα καλύτερες υπερπαραμέτρους.

Προσδιορισμός m: Στο διάγραμμα του error παρατηρήσαμε υψηλό variance και γι'αυτό χρησιμοποιήσαμε information gain ώστε να κρατήσουμε μόνο τις σημαντικές ιδιότητες. Τελικά, μετά το information gain προέκυψε m = 3000.

Προσδιορισμός λ: Αρχικά οι τιμές του σφάλματος είχαν μεγάλη απόκλιση μεταξύ διαδοχικών ποσοστών παραδειγμάτων και γιαυτό μειώσαμε την τιμή της υπερπαραμέτρου λ. Τελικά, θέσαμε λ = 0.001

Προσδιορισμός epochs:

Στο μοντέλο RNN για μεγαλύτερες από 3 εποχές, το validation loss αυξάνεται και άρα δεν έχει νόημα να το τρέξουμε για παραπάνω από 3 εποχές διότι τότε έχουμε overfitting. Συνεπώς epochs = 3.