ΘΕΩΡΙΑ ΑΠΟΦΑΣΕΩΝ
ΕΞΑΜΗΝΙΑΙΑ ΕΡΓΑΣΙΑ 2020-2021

Τμήμα Μηχανικών Η/Υ και Πληροφορικής

Ελευθεριάδης Πέτρος 1041741 (6044)
Βασιλείου Χαράλαμπος 1043757 (6277)

-Ομάδα 07-

Εισαγωγή

Το project που επιλέξαμε είναι η εργαστηριακή άσκηση 07 και έχει σκοπό τη κατηγοριοποίηση σχολίων χρηστών απο social media σε θετικά και αρνητικά. Αυτό θα επιτευχθεί με τη χρήση μοντέλων μηχανικής μάθησης πάνω σε δεδομένα, τα οποία πρώτα έχουν προεπεξεργαστεί. Μιας και τα δεδομένα μας είναι έγγραφα κειμένου θα πρέπει να τα μετατρέψουμε σε κάποια αναπαράσταση από αριθμούς ώστε να μπορέσουμε να τα εισάγουμε σε ένα τέτοιο μοντέλο.

Το κώδικα της εργασίας μας καθώς και την εξέλιξη της μπορείτε να βρείτε στο ακόλουθο git repository: https://github.com/Petros11888/Sentiment-Analysis.

Προεπεξεργασία Δεδομένων

Εισαγωγή δεδομένων

Έχουμε στη διαθεση μας δυο labeled datasets, το SocialMedia_Positive.csv και το SocialMedia_Negative.csv, τα οποία περιέχουν θετικά και αρνητικά σχόλια αντίστοιχα. Τα τοποθετήσαμε ολα σε ενιαίο dataset all_data.csv και ύστερα μετατρέψαμε τη 3η στήλη των δεδομένων σε ακέραιο με την εξής αντιστοίχιση: ‘positive’->0 & negative->1. Έπειτα αυτό το αρχείο εισάγεται στο πρόγραμμα με τη χρήση της read_csv που προσφέρει η βιβλιοθήκη pandas. Έτσι έχουμε ένα dataframe και απο αυτο θα παίρνουμε τα δεδομένα που θέλουμε.

Επεξεργασία δεδομένων

Τα δεδομένα που θα επεξεργαστούμε βρίσκονται στη 2η στήλη του dataframe και είναι μορφής text. Μεταφέρουμε τη στήλη στη λίστα ‘sentences’ και στη συνέχεια, έχοντας παρατηρήσει το dataset, “καθαρίζουμε” το κείμενο από άχρηστη πληροφορία.

Συγκεκριμένα:

Αφαιρούμε μονούς χαρακτήρες πχ ‘s’

Αφαιρούμε τα links πχ “http://t.co/LpHI2B55vp”

Αφαιρούμε τα tags πχ “@geoX9”

Αφαιρούμε τα hashtags πχ “#iphone”

Αντικαθιστούμε τα πολλά κενά με ένα.

Μια σημαντική βιβλιοθήκη που χρησιμοποιήσαμε είναι η NLTK. Με τη συνάρτηση word_tokenize της NLTK διαχωρίσαμε τα κείμενα σε tokens. Από το nltk.corpus κάναμε import το stopwords ώστε να απομακρύνουμε άχρηστη πληροφορία από τα δεδομένα καθώς και τα σημεία στίξης και έπειτα μετατρέψαμε όλες τις λέξεις σε πεζά γράμματα. Τέλος, από το nltk.stem.wordnet εισήγαμε τη WordNetLemmatizer για να κρατήσουμε το λήμμα από κάθε λέξη για μεγαλύτερη απόδοση.

Δοκιμάσαμε την επιλογή του PorterStemmer όμως παρατηρήσαμε πως είχε λίγο χειρότερη απόδοση. Επίσης δοκιμάσαμε lemmatization με postags, αλλα ούτε αυτό είχε καλύτερα αποτελέσματα οπότε επιλέξαμε σκέτο lemmatization.

Μοντέλα Μηχανικής Μάθησης

Πριν προχωρήσουμε στα μοντέλα που χρησιμοποιήσαμε, πρέπει να αναφέρουμε πως θα μετατρέψουμε δεδομένα τύπου string (λέξεις) σε αριθμούς οι οποίοι είναι κατανοητοί από τον αλγόριθμο μηχανικής μάθησης. Αυτό το πετύχαμε με τη χρήση της συνάρτησης TFidfVectorizer από τη βιβλιοθήκη της sklearn. Έτσι, κάθε token μετατρέπεται σε έναν αριθμό που το αντιπροσωπεύει και μπορεί να χρησιμοποιηθεί ως είσοδο σε ένα μοντέλο μηχανικής μάθησης. Τέλος, με τη χρήση της συνάρτησης toarray, μετατρέψαμε τα δεδομένα εισόδου σε numpy array.

Με τη χρήση της train_test_split από τη βιβλιοθήκη sklearn κάναμε train το 80% των δεδομένων και το υπόλοιπο 20% κρατήσαμε για testing.

KNN

Ο πρώτος αλγόριθμος που δοκιμάσαμε ήταν ο K Nearest Neighbor, ο οποιος κάνει τις επιλογές του υπολογίζοντας τις ευκλείδειες αποστάσεις των δεδομένων στην είσοδο. Όσον αφορά τον αριθμό K του KNN δοκιμάσαμε διάφορα νούμερα, και για το συγκεκριμένο dataset καταλήξαμε στο K=30. Η υλοποίηση του αλγορίθμου εγινε με τη χρήση του KNeighborsClassifier απο την sklearn.

Αποτελέσματα

Έχοντας τρέξει τον αλγόριθμο 10 φορές μπορούμε να συμπεράνουμε πως το accuracy κυμαίνεται απο 81% εως 88%. Όμως αυτο δεν είναι σταθερό καθώς υπήρξαν και φορές που πέτυχε accuracy μικρότερο του 70%

```
[[72 33]
 [ 0 96]]
              precision    recall  f1-score   support

           0       1.00      0.69      0.81       105
           1       0.74      1.00      0.85        96

    accuracy                           0.84       201
   macro avg       0.87      0.84      0.83       201
weighted avg       0.88      0.84      0.83       201

0.835820895522388
```

Naive Bayes

Ένας άλλος αλγόριθμος που δοκιμάσαμε είναι ο Naive Bayes classifier ο οποίος κάνει χρήση του θεωρήματος Bayes για να κάνει προβλέψεις. Η υλοποίηση του αλγορίθμου έγινε με τη χρήση του MultinomialNB απο την sklearn.

Αποτελέσματα

Εδώ τα αποτελέσματα του accuracy κυμαίνονται από 83%-88%. Δεν έχει μεγάλη βελτίωση απο τον αλγόριθμο KNN παρά μόνο στη σταθερότητα της τιμής.

```
[[92 24]
 [ 3 82]]
              precision    recall  f1-score   support

           0       0.97      0.79      0.87       116
           1       0.77      0.96      0.86        85

    accuracy                           0.87       201
   macro avg       0.87      0.88      0.87       201
weighted avg       0.89      0.87      0.87       201

0.8656716417910447
```

SVM

Το τελευταίο μοντέλο που χρησιμοποιήσαμε ηταν το Support Vector Machine (SVM), το οποίο ψάχνει να βρει το hyperplane που κατηγοριοποιεί καλύτερα τα δεδομένα. Η υλοποίηση του αλγορίθμου έγινε με τη χρήση του SVC απο την sklearn.

Αποτελέσματα

Εδώ τα αποτελέσματα του accuracy κυμαίνονται από 85%-92%. Πέρα από το accuracy, που το SVM πετυχαίνει καλύτερα νούμερα, έχουμε και μεγαλύτερη σταθερότητα.

```
[[ 66  18]
 [  1 116]]
              precision    recall  f1-score   support

           0       0.99      0.79      0.87        84
           1       0.87      0.99      0.92       117

    accuracy                           0.91       201
   macro avg       0.93      0.89      0.90       201
weighted avg       0.92      0.91      0.90       201

0.9054726368159204
```

Συμπεράσματα

Έχοντας δοκιμάσει τους 3 αυτούς αλγορίθμους καταλήγουμε στο συμπέρασμα πως ο SVM έχει τη καλύτερη απόδοση σε σχέση με τα άλλα καθώς πετυχαίνει σταθερά καλύτερο accuracy.

Ένα αξιοσημείωτο γεγονός που παρατηρούμε σε όλους τους αλγορίθμους είναι το εξής. Παρατηρούμε υψηλό precision και χαμηλό recall για τα θετικά κείμενα και το ακριβώς αντίθετο στα αρνητικά κείμενα. Αυτό φαίνεται και από το confusion matrix, του οποίου η τιμή πάνω από τη διαγώνιο είναι αυξημένη ενώ η τιμή κάτω από τη διαγώνιο τείνει στο 0.

ΤΕΛΟΣ ΑΝΑΦΟΡΑΣ