1η Εργασία
Τεχνικές Εξόρυξης Δεδομένων

Μπαρμπάκος Περικλής : 1115201300111
Ράντος-Χαρισόπουλος Αλέξανδρος : 1115201300147

WORDCLOUDS

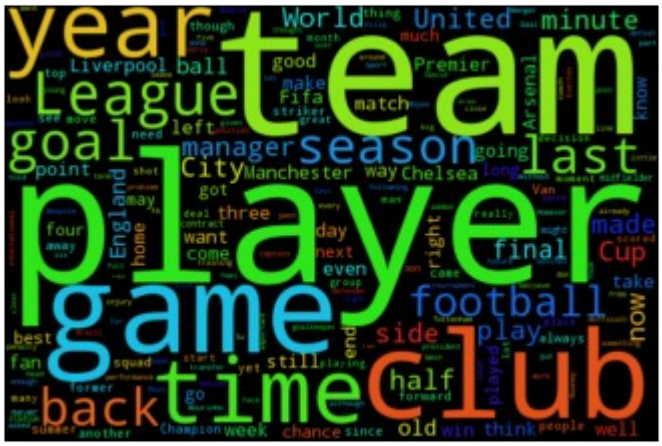
Χρησιμοποιήσαμε τη βιβλιοθήκη wordclould και πιο συγκεκριμένα τα WordCloud,STOPWORDS από αυτήν.Επιπλέον επεκτείναμε τη λίστα των stopwords της βιβλιοθήκης με συγκεκριμένες λέξεις που εμφανίζονταν συχνά σε πολλαπλά WordClouds ενώ δεν πρόσφεραν τίποτα στο περιεχόμενο (πχ say,will,us,also,said etc).Η δημιουργία των WordClouds έγινε από τον Τίτλο και το Περιεχόμενο των άρθρων κάθε κατηγορίας όπως φαίνεται παρακάτω :

Technology

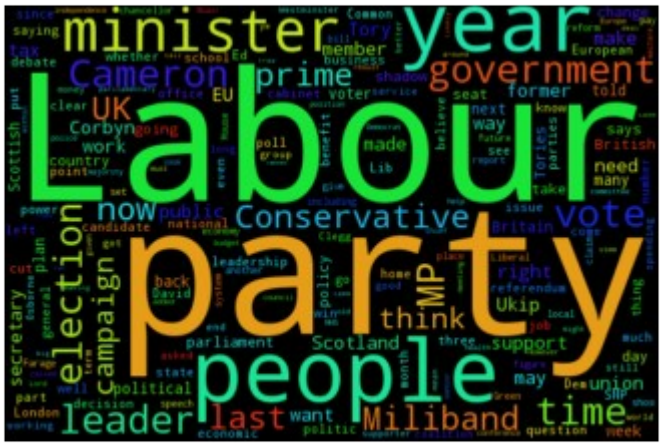
Business

Film

Football



Politics



Clustering

Όπως ζητείται στην εκφώνηση απομονώνουμε από το dataframe το Category ώστε να μην χρησιμοποιηθεί και έπειτα πάμε στις μετατροπές με τον CountVectorizer, τον tfidf-Transformer και τη μείωση διαστάσεων(lsi) με το TruncatedSvd.Η επιλογή μας είναι στα 100 components για να μην είναι πολύ μεγάλα τα sparse matrices που χειριζόμαστε σε σύγκριση με τις άλλες δοκιμές σε μνήμη.
Έπειτα τρέχουμε τον αλγόριθμο K-Means μέσω της βιβλιοθήκης nltk ώστε να χρησιμοποιήσουμε κατάλληλα το cosine similarity. Διαλέγουμε 10 επαναλήψεις ισορροπώντας χρόνο με ακρίβεια. Κάνουμε mapping σε αριθμούς τις κατηγορίες για να μπορέσουμε να τις χρησιμοποιήσουμε σαν indexes μέσα σε κελιά του myarray.Μετράμε τις εμφανίσεις κάθε cluster ως προς κάθε κατηγορία και τον numpy array τον εξάγουμε με την κατάλληλη μορφή σε ένα αρχείο clustering_Kmeans.csv.

Classification

Πλέον κρατάμε μαζί με την πληροφορία του content και αυτή που δίνει ο τίτλος.Ξεκινώντας από τα support vector machines διαπιστώνουμε οτι το LinearSVC δίνει πολύ καλύτερους χρόνους και περίπου στο 0.97 στα περισσότερα measures. Με rbf kernel είναι πολύ κοντά οι μετρήσεις αλλά ο χρόνος διαφέρει χαοτικά,ειδικά όταν κάνουμε GridSearch για τις παραμέτρους C(στο 1 μας δίνει τα best scores) και gamma.Ο κώδικας του rbf φαίνεται παρακάτω :

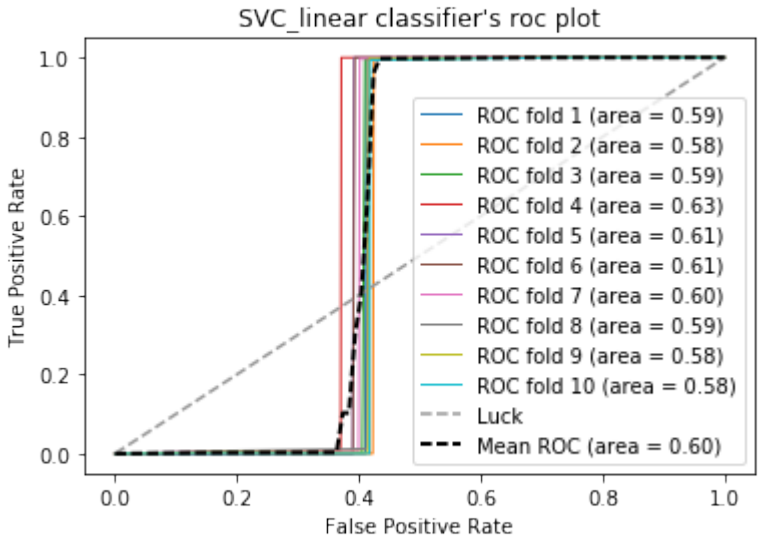
```
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn import metrics
from sklearn.feature_extraction.text import ENGLISH_STOP_WORDS
import numpy as np
stop_words = ENGLISH_STOP_WORDS
count_vect = CountVectorizer(stop_words=stop_words)
kf = ShuffleSplit(n_splits=10, test_size=0.1, random_state=0)
#kf = KFold(n_splits=5)
# kf.get_n_splits(twenty_train.data)
fold = 0
# K-fold diaspame se k kommatia to train_set kai to testaroume.
for train_index, test_index in kf.split(df.Content):
    #dimiourgia dianismatwn twn train + test set sto analogo fold/kommati
    X_train_counts = count_vect.fit_transform(df.Content[train_index])
    transformer = TfidfTransformer(smooth_idf=False)
    X_train_counts=transformer.fit_transform(X_train_counts)
    svd = TruncatedSVD(n_components=500)
    X_train_counts=svd.fit_transform(X_train_counts)
    X_test_counts = count_vect.transform(df.Content[test_index])
    X_test_counts=transformer.transform(X_test_counts)
    X_test_counts=svd.transform(X_test_counts)
    C_range = np.logspace(-2, 10, 13)
    gamma_range = np.logspace(-9, 3, 13)
    param_grid = dict(gamma=gamma_range, C=C_range)
    svr = svm.SVC(kernel='rbf')
    clf = GridSearchCV(svr, param_grid=param_grid)
    clf.fit(X_train_counts, Y[train_index])
```

Για το Cross Validation μοιράζουμε τα δεδομένα με 10-fold μέθοδο(ShuffleSplit ώστε να μοιράσουμε τυχαία τα δεδομένα και όχι με τη σειρά που είναι στο dataframe) και κάνουμε τις ίδιες μετατροπές(vectorizer,lsi klπ.) και στο training kai στο test κομμάτι.Κάνουμε fit την εκάστοτε μέθοδο και έπειτα predict στο κομμάτι που τεστάρουμε(1/10 του συνολικού).
Για τον Naive Bayes αποφασίσαμε ότι ο πιο αποδοτικός και εύχρηστος για το δοθέν dataset ήταν ο GaussianNB.Επίσης δεν βρήκαμε κάποιες συγκεκριμένες μεταβλητές που να τον κάνουν πιο αποδοτικό.Σε παρόμοια πλαίσια κινηθήκαμε και για τη μέθοδο RandomForest.

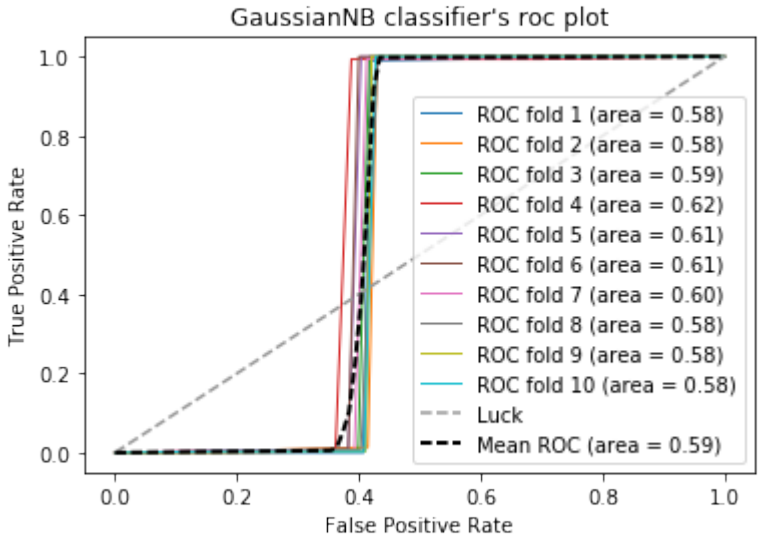
Όσο για το KNN ενώ το υλοποιήσαμε,δεν είχαμε τον απαραίτητο χρόνο να ολοκληρώσουμε το validation του αλλά θεωρούμε ότι η λογική που ακολουθήθηκε είναι η σωστή,αλλά δεν ξέρουμε αν είναι στα πλαίσια που πρέπει από πλευράς χρόνου κλπ.
Οι συναρτήσεις που υλοποιήσαμε είναι:
ο υπολογισμός της ευκλείδιας απόστασης στην ουσία μεταξύ μιας γραμμής του vectorizer του test set και του train set. Στο tuple distance κρατάμε τον αριθμό της γραμμής και την απόσταση που βρήκαμε σε ένα tuple me τη μορφή [row,distance]. Αφού κάνουμε sort με βάση το distance κρατάμε τις k μικρότερες τιμές και επιστρέφουμε τη λίστα των αριθμών των κ γραμμών.Όπου έπειτα τσεκάρουμε τις κατηγορίες τους στο dataframe και επιστρέφουμε αυτή με τις περισσότερες εμφανίσεις ανάμεσα στις 10.

Σε κάθε επανάληψη του k-fold κρατάμε τα κατάλληλα metrics για κάθε μέθοδο που αναφέρθηκε παρακάτω και στο τέλος των K επαναλήψεων τυπώνουμε τις μέσες τιμές του f-score,precision,recall_score,accuracy,AUC καθώς και την γραφική παράσταση roc-plot.Εδώ λάβαμε υπόψην το documentation του sklearn για την υλοποίηση τους.

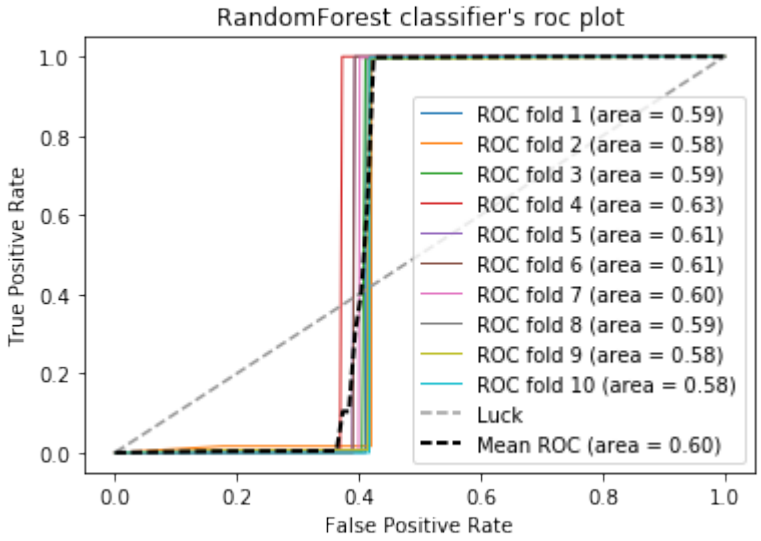
Οι μετρήσεις και τα roc plot κάθε μεθόδου φαίνονται παρακάτω :




```
Average f-score : 0.964047910243
Average precision : 0.965303209989
Average recall_score : 0.96296013989
Average Accuracy : 0.966422167889
Average AUC : 0.595803571553
```




```
Average f-score : 0.879356532218
Average precision : 0.881800629534
Average recall_score : 0.883028066722
Average Accuracy : 0.887367563162
Average AUC : 0.594693052123
```




```
Average f-score : 0.955523931946
Average precision : 0.955949945872
Average recall_score : 0.955269586273
Average Accuracy : 0.958761206194
Average AUC : 0.596414097114
```

Output_Files & Prediction-TestSet

Σε κάθε CrossValidation που κάναμε παραπάνω αποθηκεύαμε σε 2 πίνακες(em,roc10) τις μέσες τιμές κάθε αλγορίθμου για την τελική αναφορά. Τέλος ,βάσει των δεδομένων που συλλέξαμε, δημιουργούμε τα ζητούμενα αρχεία (EvaluationMetric_10fold.csv,roc_10fold.png) όπου στο roc_10 απλά έχουμε τις μέσες τιμές των roc_plot των μεθόδων που υλοποιήσαμε παραπάνω,άρα 4 γραμμές.
Έπειτα κάνουμε την πρόβλεψη μας με την αποδοτικότερη μέθοδο που βρίκαμε (linear_svc) και αποθηκεύουμε τα αποτελέσματα στο αρχείο testSet_categories.csv με τις προϋποθέσεις που ζητά η εκφώνηση.

EXTRA : Πέρα από το README,τα αρχεία εξόδου και το κώδικα έχουμε βάλει σε .pdf ένα αρχείο ‘ASK1 Jupyter Capture’ όπου φαίνεται το παραδοτέο που είχαμε δώσει αν το ανοίξεις με το Jyputer Notebook.