****

**ΤΕΧΝΙΚΕΣ ΕΞΟΡΥΞΗΣ ΔΕΔΟΜΕΝΩΝ**

Εφαρμογή τεχνικών εξόρυξης δεδομένων και αξιολόγηση

WordCloud, Clustering, Classification και Beat the Benchmark

Κοντόπουλος Παναγιώτης ΑΜ: 1115201100124

Τσακριλής Αλέξανδρος-Παναγιώτης

ΑΜ: 1115201100092

ΑΘΗΝΑ 2016

# **Περιεχόμενα**

[Περιεχόμενα 1](#_Toc451785261)

[Δημιουργία WordCloud 2](#_Toc451785262)

[Δομή Κώδικα (+ οδηγίες για εκτέλεση κώδικα) 2](#_Toc451785263)

[Υλοποίηση Συσταδοποίησης (Clustering) 3](#_Toc451785264)

[Δομή Κώδικα (+ οδηγίες για εκτέλεση κώδικα) 3](#_Toc451785265)

[Υλοποίηση κατηγοριοποίησης (Classification) 5](#_Toc451785266)

[Δομή Κώδικα (+ οδηγίες για εκτέλεση κώδικα) 5](#_Toc451785267)

[Δοκιμές 5](#_Toc451785268)

[Συμπεράσματα 5](#_Toc451785269)

# Δημιουργία WordCloud

## Δομή Κώδικα (+ οδηγίες για εκτέλεση κώδικα)

Στον παρακάτω πίνακα φαίνεται η διάρθρωση της εργασίας σε αρχεία και φακέλους.

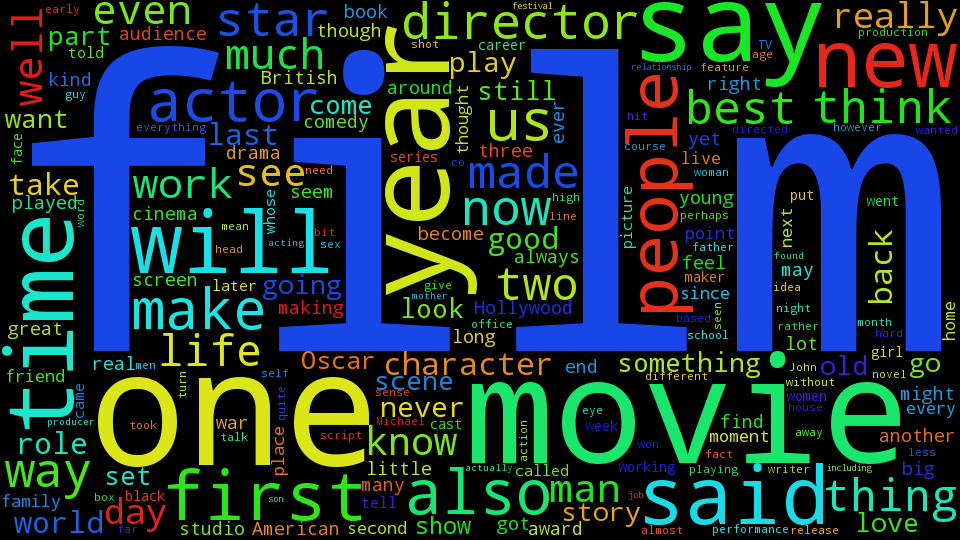
|  |  |  |
| --- | --- | --- |
| ΦΑΚΕΛΟΣ | ΑΡΧΕΙΟ | ΛΕΠΤΟΜΕΡΕΙΕΣ |
| ./ | data\_csv\_functions.py | Αποτελείται από το σύνολο των συναρτήσεων για την εισαγωγή και εξαγωγή δεδομένων από τα αρχεία csv. |
| ./ | data\_wordcloud.py | Αποτελεί την υλοποίηση των συναρτήσεων για την παραγωγή του WordCloud. |

Επίσης στον φάκελο data υπάρχουν τα αρχεία train\_set.csv και test\_set.csv.

**Οδηγίες για την εκτέλεση του προγράμματος**

**Για την εκτέλεση στα μηχανήματα linux της σχολής ο χρήστης τρέχει την εντολή python data\_wordcloud.py path\_to\_file. Ένα παράδειγμα χρήσης του είναι το ακόλουθο: python data\_wordcloud.py ./data/train\_set.csv**

**Αφού εκτελεστεί το πρόγραμμα, θα παραχθούν πέντε .png αρχεία στο φάκελο data και θα εμφανιστούν στην οθόνη, όπου το καθένα περιέχει το wordcloud μίας κατηγορίας.**



Για την παραγωγή του wordcloud χρησιμοποιήθηκε η βιβλιοθήκη: <https://github.com/amueller/word_cloud>

# Υλοποίηση Συσταδοποίησης (Clustering)

## Δομή Κώδικα (+ οδηγίες για εκτέλεση κώδικα)

Στον παρακάτω πίνακα φαίνεται η διάρθρωση της εργασίας σε αρχεία και φακέλους.

|  |  |  |
| --- | --- | --- |
| ΦΑΚΕΛΟΣ | ΑΡΧΕΙΟ | ΛΕΠΤΟΜΕΡΕΙΕΣ |
| ./ | data\_csv\_functions.py | Αποτελείται από το σύνολο των συναρτήσεων για την εισαγωγή και εξαγωγή δεδομένων από τα αρχεία csv. |
| ./ | data\_clustering.py | Αποτελεί την υλοποίηση των συναρτήσεων για την εφαρμογή του clustering με τον αλγόριθμο K-Means. |

Επίσης στον φάκελο data υπάρχουν τα αρχεία train\_set.csv και test\_set.csv.

**Οδηγίες για την εκτέλεση του προγράμματος**

**Για την εκτέλεση στα μηχανήματα linux της σχολής ο χρήστης τρέχει την εντολή python data\_clustering.py path\_to\_file. Ένα παράδειγμα χρήσης του είναι το ακόλουθο: python data\_clustering.py ./data/train\_set.csv**

**Αφού εκτελεστεί το πρόγραμμα, θα παραχθεί το αρχείο clustering\_KMeans.csv, το οποίο περιέχει τα ποσοστά των δεδομένων κάθε κατηγορίας.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Politics | Football | Technology | Film | Business |
| Cluster1 | 0.09 | 0.00 | 0.01 | 0.00 | 0.90 |
| Cluster2 | 0.01 | 0.00 | 0.90 | 0.01 | 0.08 |
| Cluster3 | 0.00 | 0.99 | 0.00 | 0.00 | 0.00 |
| Cluster4 | 0.01 | 0.00 | 0.02 | 0.96 | 0.01 |
| Cluster5 | 0.97 | 0.00 | 0.00 | 0.00 | 0.02 |

Για την υλοποίηση του αλγορίθμου χρησιμοποιήθηκε κώδικας από:

The Data Science Lab: <https://datasciencelab.wordpress.com/2013/12/12/clustering-with-k-means-in-python/>

Scikit-learn: <http://scikit-learn.org/dev/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html>

# Υλοποίηση κατηγοριοποίησης (Classification)

## Δομή Κώδικα (+ οδηγίες για εκτέλεση κώδικα)

Στον παρακάτω πίνακα φαίνεται η διάρθρωση της εργασίας σε αρχεία και φακέλους.

|  |  |  |
| --- | --- | --- |
| ΦΑΚΕΛΟΣ | ΑΡΧΕΙΟ | ΛΕΠΤΟΜΕΡΕΙΕΣ |
| ./ | data\_classification.py | Αποτελεί την υλοποίηση των συναρτήσεων για την εφαρμογή του classification. |

Επίσης στον φάκελο data υπάρχουν τα αρχεία train\_set.csv και test\_set.csv.

**Οδηγίες για την εκτέλεση του προγράμματος**

**Για την εκτέλεση στα μηχανήματα linux της σχολής ο χρήστης τρέχει την εντολή python data\_classification.py path\_to\_file. Ένα παράδειγμα χρήσης του είναι το ακόλουθο: python data\_clustering.py ./data/train\_set.csv**

**Αφού εκτελεστεί το πρόγραμμα, θα παραχθούν τα αρχεία:**

* **EvaluationMetric\_10fold.csv, το οποίο περιέχει τον πίνακα με τις ακρίβειες και τα αποτελέσματα του ROC plot.**
* **roc\_10fold.png, το οποίο περιέχει τα αποτελέσματα του ROC plot.**
* **testSet\_categories.csv, το οποίο περιέχει τις κατηγορίες των άρθρων που περιέχονται στο Test Set.**

## Δοκιμές

Στις δοκιμές μας χρησιμοποιήσαμε το 67% του train\_set για train των αλγορίθμων και το υπόλοιπο 33% σαν test\_set, ώστε να ελέγξουμε την απόδοση των αλγορίθμων και να βρούμε τις βέλτιστες ρυθμίσεις.

## Μέθοδοι

* **NaiveBayes – Multinumial**: Σε αυτόν αλλάζαμε τιμές στο όρισμα α (The smoothing priors \alpha \ge 0 accounts for features not present in the learning samples and prevents zero probabilities in further computations. Setting \alpha = 1 is called Laplace smoothing, while \alpha < 1 is called Lidstone smoothing.), ώστε να μελετήσουμε τις διαφορές για τους περιορισμούς. Στο δικό μας train\_set παρατηρήσαμε ότι για την τιμή 0.05 τα αποτελέσματα, εν συγκρίσει με την default τιμή 1, ήταν καλύτερα. Επομένως η περιοχή Lidstone smoothing από 0 ως 1, είχε τα βέλτιστα αποτελέσματα.
* **NaiveBayes – Binomial (Bernoulli)**: Σε αυτόν αλλάζαμε τιμές στο όρισμα α, όπως και πριν οι αλλαγές είναι παρόμοιες.
* **K-Nearest Neighbor**: Σε αυτόν αλλάζαμε το όρισμα Κ ώστε να επηρεάσουμε τον αριθμό των γειτόνων. Σε αυτή την περίπτωση μετά από δοκιμές η βέλτιστη τιμή στο set που είχαμε ήταν το 9 (Larger values of k generalize better, and smaller values may tend to overfit.).
* **Support Vector Machines (SVM)**: Σε αυτή την περίπτωση επιλέξαμε την default τιμή με C=1.
* **Random Forests**: Σε αυτή την περίπτωση επιλέξαμε ως τιμή του ορίσματος n\_estimators το 1000, καθώς μετά από δοκιμές για τιμές πολύ κοντά στο >1000 και <1000 υπήρχαν ελάχιστες διαφορές προς το χειρότερο.

## Συμπεράσματα