**For the TMDB API**

<https://stackoverflow.com/questions/62638591/how-to-extract-data-from-tmdb-using-python>

<https://www.codespeedy.com/fetch-tmdb-movie-data-using-python/>

παράδειγμα χρησιμοποιοντας τον κωδικο ταινιας

<https://www.websitescraper.com/how-to-extract-tv-shows-data-on-tmdb-using-python.php>

extract TV shows αλλα κάτι δεν τρέχει το έκανα για τις ταινίες

<https://www.aionlinecourse.com/blog/how-to-extract-data-from-tmdb-using-python>

<https://blog.jovian.ai/scraping-popular-tv-shows-on-tmdb-using-python-4374295e45f6>

έχει ένα κομμάτι κώδικα που ίσως είναι αυτός με την βιβλιοηθηκη την γνωστη

**ΒΙΒΛΙΟΓΡΑΦΙΑ ΚΑΙ ΧΡΗΣΙΜΑ LINK**

1. Y18-1007

**Predicting the Genre and Rating of a Movie Based on its Synopsis**

1. 4476**…**

**On the Use of Synopsis-based Features for Film Genre Classification**

Σε αυτό το άρθρο εξετάζονται text based features από την σύνοψη του κειμένου με 19 features extraction approaches συνδυασμένα με 4 multilabel classifiers.

implemented and evaluated common text feature extractions techniques and different multi-label classifiers to provide a solid baseline for this task.

Used a total of 19 separate feature extraction approaches:

1. 9 of them based on the usage of tf-idf techniques (term frequency-inverse document frequency)

* <https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089>
* Η μέθοδος
* Παράδειγμα με βήματα

**Cleaning the data before the application of the method**

1.Lowercase

2.Stop wοrds (πιο πριν καλό είναι να κάνω ένα word count για να τα βρω. Ίσως ανά ταινία και εν τέλει να βρω το sum να φτιάξω και ένα διάγραμμα και ανά genre έτσι ώστε να έχω εικόνα)

(Ίσως να χρειαστεί να κάνω σε γκρουπακια κάποιες λέξεις για την διαδικασία που χρειάζομαι)

3.Punctuation

4.Apostrophe&Single characters

5.Lemmatization and then Stemming

6.Converting numbers to words

* <https://www.analyticsvidhya.com/blog/2021/09/creating-a-movie-reviews-classifier-using-tf-idf-in-python/>?
* Η μέθοδος σε κώδικα και παράδειγμα με movie classifier σε positive or negative review πρωτα χρησιμοποιεί την μέθοδο tf-idf και μετά naïve-bayes

1. 10 based on non-sparse word embeddings models.
   * + **Word2Vec🡪 (1.** **CBOW (Continuous Bag of Words), 2.** **Skip Gram)** <https://www.geeksforgeeks.org/python-word-embedding-using-word2vec/>

έχει κάποιον κώδικα

<https://stackabuse.com/implementing-word2vec-with-gensim-library-in-python/>

σημαντικός κώδικας διασικασίας Word2vec και εξήγησης ορολογίας

* + - **FastText**

<https://stackabuse.com/python-for-nlp-working-with-facebook-fasttext-library/>

* + - **Wang2vec**
    - **Glove**

<https://medium.com/analytics-vidhya/basics-of-using-pre-trained-glove-vectors-in-python-d38905f356db>

**NOTE:** Σε αυτό έχει αγνοήσει frequency<5

***Για τους classifiers***: evaluated two separate strategies for multi-label classification

1. based on decision trees (were built using three different algorithms):

* DecisionTreeClassifier

Εδώ ως parameters: quality of each split measured though Gini impurity

(all classes had the same weight)

<https://machinelearningknowledge.ai/decision-tree-classifier-in-python-sklearn-with-example/>

ορισμός και κώδικας

* ExtraTreesClassifier

<https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/>

* RandomForestClassifier

<https://stackabuse.com/random-forest-algorithm-with-python-and-scikit-learn/>

1. based on multi-layer perceptrons (MLP)

* <https://scikit-learn.org/stable/modules/neural_networks_supervised.html>
* <https://analyticsindiamag.com/a-beginners-guide-to-scikit-learns-mlpclassifier/>

**Conclusion:** We obtained the **best results** by using features extracted by **tf-idf** approaches and using a Multilayer Perceptron for the classification.

1. Mangolin2020

**A multimodal approach for multi-label movie genre classification**

labeled each movie of the dataset according to a set of eighteen genre labels

TMDB dataset for synopsis, ID from IMDB

Subtitles were collected from OpenSubtitles

**extracting features:**

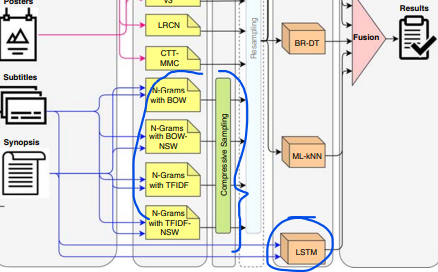
**Subtitles**: Long Short-Term Memory (LSTM), and N-grams with Term Frequency– Inverse Document Frequency (TF-IDF)

**Synopsis**: LSTM and N-grams with TF-IDF.

- LSTM

<https://www.alpha-quantum.com/blog/long-short-term-memory-lstm-with-python/long-short-term-memory-lstm-with-python/>

ορισμός και κώδικας

****

* **TF-IDF features extracted from N-grams**

bag-of-word (BOW)

* **Representation learning approach based on Long Short-Term Memory (LSTM)**

Word2vec as the embedding representation

-max features of this layer as 50,000

**Resampling:**

To deal with the imbalance of our dataset, we included a resampling phase.

Two different resampling algorithms:

**the Multi-Label Synthetic Minority Over-sampling Technique (ML-SMOTE)**

<https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>

**the Multi-Label Tomek Link (MLTL)**

**Late fusion:**

<https://pubmed.ncbi.nlm.nih.gov/25879948/>

ορισμός. Δεν νομίζω να χρειάζεται εδώ καθώς λογικά θα χρησιμοποιήσουμε ένα μόνο classifier.

**CONCLUSION**:

* the data source “synopsis” with the LSTM classifier achieved the best individually performance
* The test indicates that the use of late fusion improved the classification performance

1. Text\_image\_SVM

**Fast Film Genres Classification Combining Poster and Synopsis**

**Feature extraction:**

BOW framework for obtaining text feature

Representing every text document in term of VSM

**Classification:**

SVM clasifier