CMPE 462 - Spring 2021
Machine Learning Project
Sentiment Analysis on IMDB User Reviews

Introduction

In this project, you are going to apply machine learning techniques on text data. You will propose and implement a feature extraction/selection and classification model for sentiment analysis on IMDB user reviews. This is a group project and will be submitted in 3 steps. Details of each step, submission and grading are found in the following sections. In the implementation phase, you will be using a conda virtual environment for Phyton. Details of this environment can also be found in the following sections. Set your environment as soon as possible in order to keep up with the project timeline.

Sentiment analysis is the process of understanding the underlying sentiment of a given text, which can be positive, negative or neutral. Sentiment analysis is actually applied and popularly used for analyzing social media messages in order to get an understanding of what the general opinion about a specific subject is. Elections and marketing are two main example areas.

You will implement sentiment analysis on IMDB user reviews. In a film main page on IMDB, you can reach user reviews about that film from the top menu (Figure 1). This page lists the reviews (Figure 2). Each review has a title, text and a corresponding rating score. There is also a user ratings page where you can observe the distribution of the ratings (Figure 3). In this project, you will design and implement a machine learning system that takes a user review as input and decides its sentiment. The target classes are Positive (P, for ratings 7,8,9,10), Negative (N, for ratings 1,2,3) and Neutral (Z, for ratings 4,5,6).

Step1: Data Collection - COMPLETE!

The first step of the project is data collection. You will select reviews with the criteria given, save them and check them.

Each project group will :

- be assigned an English letter, say S.
- collect user reviews for films having (English) names starting with your assigned letter, say Scarface, Star Wars.
- collect 150 user reviews in English: 50 Positive (with rating 7,8,9,10), 50 Negative (with rating 1,2,3), 50 Neutral (with rating 4,5,6)

The IMDB user reviews have three main parts that are important for the project. The first one is the header, the second one is the review text and the third one is the rating. You will save each review in a separate txt file. The first line of the file will be the header of the review. The rest of the file is the review text. There is a naming convention for the file. There are three parts of the file name connected with "_": 1. starting letter, say S, 2. your index of the file, 1:150, 3. class label for the user rating given with the review. For the first review in Figure 2, the corresponding file name will be S_1_P.txt, stating that this is a review for a film starting with S, this is the first of reviews for films starting with S and the review rating is Positive.

Before submitting the review files for Step1 of the project, you will check your files if there exists any special characters that will cause error while reading with Python. You can use the below code and make sure that all your files can be processed without errors.

```
with open(<filename>, 'r') as f:
        lines = f.readlines()
        print(f)
```

If there are errors for some files, contact me and we will analyze and resolve the errors together. There can be situations to ignore the selected review and select a new one. Therefore, apply your checks while you are building your dataset.

Step2: First Run

In this step, you will:

- search literature for the project
- analyze the given data
- if necessary clean data (i.e. remove special characters)
- decide/create your features
- if necessary apply feature selection
- apply your classification model

You will try and compare several methods. You can search similar applications in the literature and make use of their proposed methods with your own implementation. If this is the case, don't forget to cite them in your report. In this First Run, no deep learning models and no use of word-embeddings are allowed.

You will receive two datasets for this step. All the files from all project groups will be added to create the whole project dataset. From this big set, we will select TRAIN and VAL (VALIDATION) sets. You will build your models and train them with the TRAIN set. You will run your pre-trained model on VAL set and compare your models. With this comparison, select your best model. An input dataset is a folder containing txt input files. Input files have a naming convention. There are two parts of the file name connected with "_": 1. the index of the file, 2. class label for the user rating. An example file name will be 1_P.txt, stating that this is the first input file and the rating is Positive. Index starts from 1 for all sets, TRAIN, VAL and TEST.

Report your results on the VAL set by using performance metrics accuracy, precision and recall. Report these metrics for each class and in addition report overall performance as macro-average metrics. Discuss the results on your project report. You can find details of performance metrics in the following sections. Discuss on the results. Try to understand and describe why each model performed that way. Clearly state your best model and discuss why it performed better. Clearly state the improvements in your system.

For grading, we will test your system with a separate TEST set. The TEST set will be the same format as TRAIN and VAL sets. We will run your code from start (all steps starting from reading the dataset). We will skip the training part. Therefore, you should submit pre-trained form of your best model (only best model) as a pkl file. Below is an example code about how you can save and load your classification model:

```
import pickle
# Save model to file
pkl_filename = "step2_model_TeamA.pkl"
with open(pkl_filename, 'wb') as file:
        pickle.dump(model, file)
# Load from file
with open(pkl_filename, 'rb') as file:
        pickle_model = pickle.load(file)
```

In addition, you should write a main script that takes a pre-trained model and a dataset as input and applies all your machine learning steps (except training). You should name your script as 462project_step2_<teamname>.py, say 462project_step2_TeamA.py. You can test your main script with VAL set. We will run your script as:

```
python3 462project_step2_TeamA.py step2_model_TeamA.pkl <test-dataset-folder>
```

The output of your script should be a concatenated list of predictions in {P,N,Z} with no space and in the order of the input dataset. The order is the index order in the file names. For example:

```
Given data:
1_P.txt
2_N.txt
3_P.txt
4_Z.txt
5_P.txt

Your method predicts:
file1 N
file2 N
file3 N
file4 Z
file5 P

Then your script outputs as:
NNNZP
```

Step3: Second Run

Details will be announced later.

Project Base Environment

You will be implementing your code with Python 3.6.
You need to create a python virtual environment with Anaconda for your project. After installing Anaconda, a base environment can be created with below commands:

```
conda create -n 462project python=3.6
conda activate 462project
```

While you keep working on your models, you will need to import additional libraries. List these libraries in a requirements.txt file. State any special versions if needed. A sample requirements file can be as below:

```
scikit-learn >= 0.22.2
scipy
pandas
sentencepiece==0.1.91
```

For grading, we will load your requirements with the command below:

```
python3 -m pip install -r requirements.txt
```

Before submission, test your code on a clear new conda environment by installing additional libraries from your requirements file. Because, there will be penalty if your code doesn't run like this.

Performance Metrics

In this project, performance metrics that will be used are accuracy, precision and recall. You will report performance of each class and overall performance as macro-average metrics. Consider the confusion matrix below:

		Predicted	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

Accuracy

Accuracy is the ratio of number of correct predictions to the total number of input samples.

$$accuracy = (TP + TN)/(TP + FP + TN + FN)$$

Precision

Precision measures the percentage of actually positive samples among all positive predicted samples.

$$precision = TP/(TP + FP)$$

Recall

Recall measures the percentage of actually positive samples among all positive samples.

$$racall = TP/(TP + FN)$$

Macro-average

Macro-average is the unweighted average of the class based metrics. For example, we have accuracy values for 3 classes (A,B,C) as acc_A, acc_B and acc_C. The macro-average accuracy for the whole dataset is:

$$acc_{macro} = (acc_A + acc_B + acc_C)/3$$

Grading Details

The project will be graded over 100 points. You will be graded for your code and project reports.

- 20 points for Step 1
- 40 points for Step 2
- 40 points for Step 3

After Step 2 and Step 3, there will be a listing of project groups from highest to lowest accuracy results on TEST set. 3 best performing teams will get 10 bonus points for each step.

- 10 bonus points for Step 2 for 3 teams
- 10 bonus points for Step 3 for 3 teams

Additional details will be announced later.

Submission Details

This is a group project. Your code should be original. Any similarity between submitted projects or to a source from the web will be accepted as cheating.

If you have any further questions, send an e-mail to the course assistant: ozlem.simsek@boun.edu.tr

Step1

- The deadline for submitting Step 1 is **April 20, 2021 - 23:59**.
- For txt files: You should compress all your txt data files in a zip file with name as the assigned capital letter, say S.zip
- For project report:
  - You should submit a detailed project report in pdf format.
  - Clearly state group members and which member did what for this step.
  - You should name your report as step1_report_<teamname>.pdf, say step1_report_TeamA.pdf
- Submit max 2 items in a big zip file: txt files zip and report pdf. Name your submission zip file as step1_<teamname>.zip, say step1_TeamA.zip
- The final zip will be submitted on Moodle. Only one member of each group will make the submission.

Step2

- The deadline for submitting Step 2 is **May 17, 2021 - 23:59**.
- For code and pre-trained model:
  - You should submit your code. This includes all your code for this step, your main script and all your code for other models.
  - Your code should be sufficiently commented and indented.
  - You can write a readme file named readme.txt and explain details of your code.
  - State the extra libraries that you use in a requirements.txt file.

  - You should submit your pre-trained model, place it in the same folder as your main script.
  - You should save your pre-trained model as a pkl file with name step2_model_<teamname>.pkl, say step2_model_TeamA.pkl
  - You should compress all your code files, pkl file, requirements and readme in a zip file with name step2_code_<teamname>.zip, say step2_code_TeamA.zip

- For project report:
  - You should submit a detailed project report in pdf format.
  - Clearly state group members and which member did what for this step.
  - Include comparison of the methods you tried. Clarify your comparison with charts and graphics.
  - Clearly state your performance results on VAL set.
  - You should name your report as step2_report_<teamname>.pdf, say step2_report_TeamA.pdf
- Submit 2 items in a big zip file: code and model zip and report pdf. Name your submission zip file as step2_<teamname>.zip, say step2_TeamA.zip
- The final zip will be submitted on Moodle. Only one member of each group will make the submission.
- You should also submit your reports in Turnitin submission on Moodle.

Figures

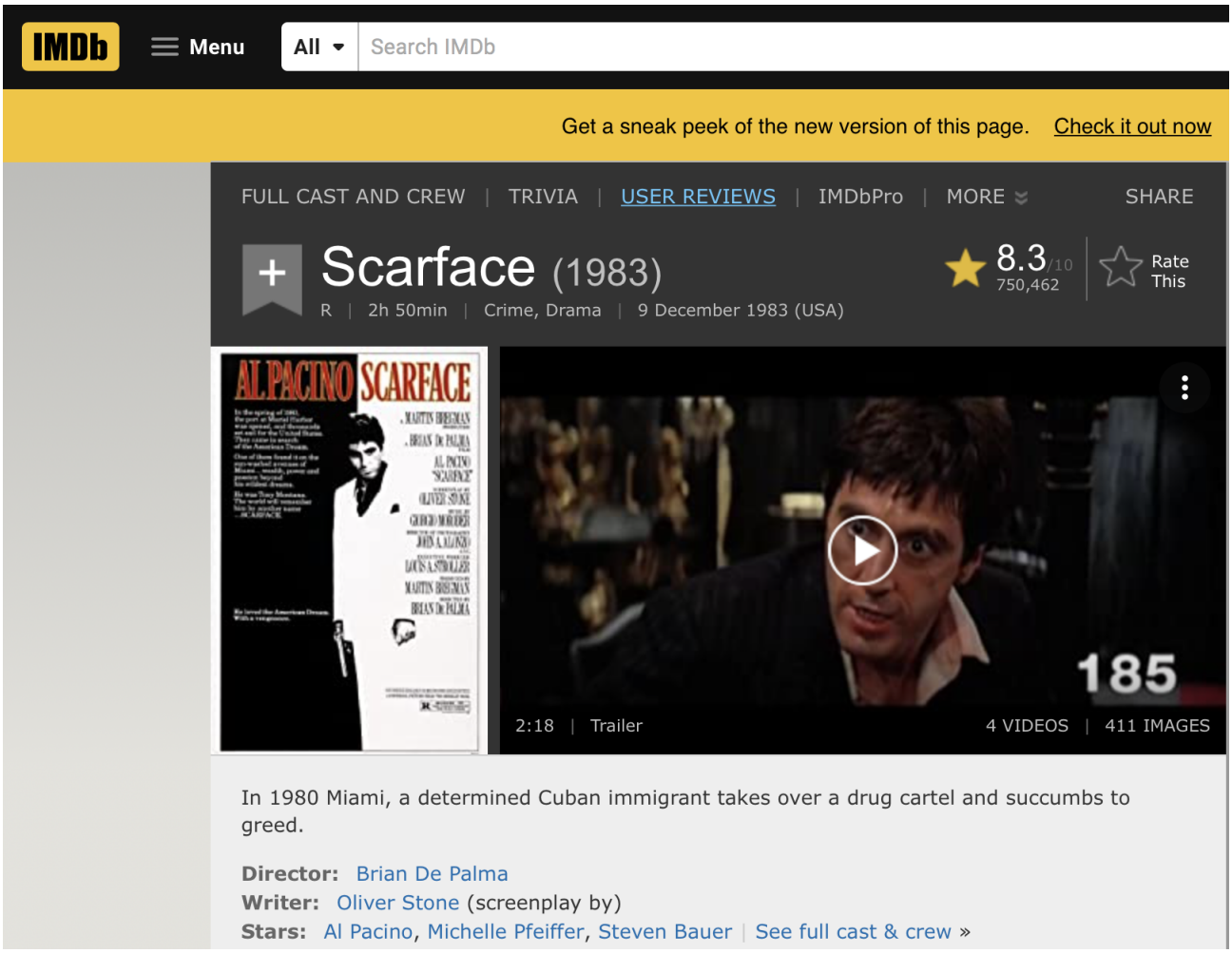

Figure 1: IMDB - film main page

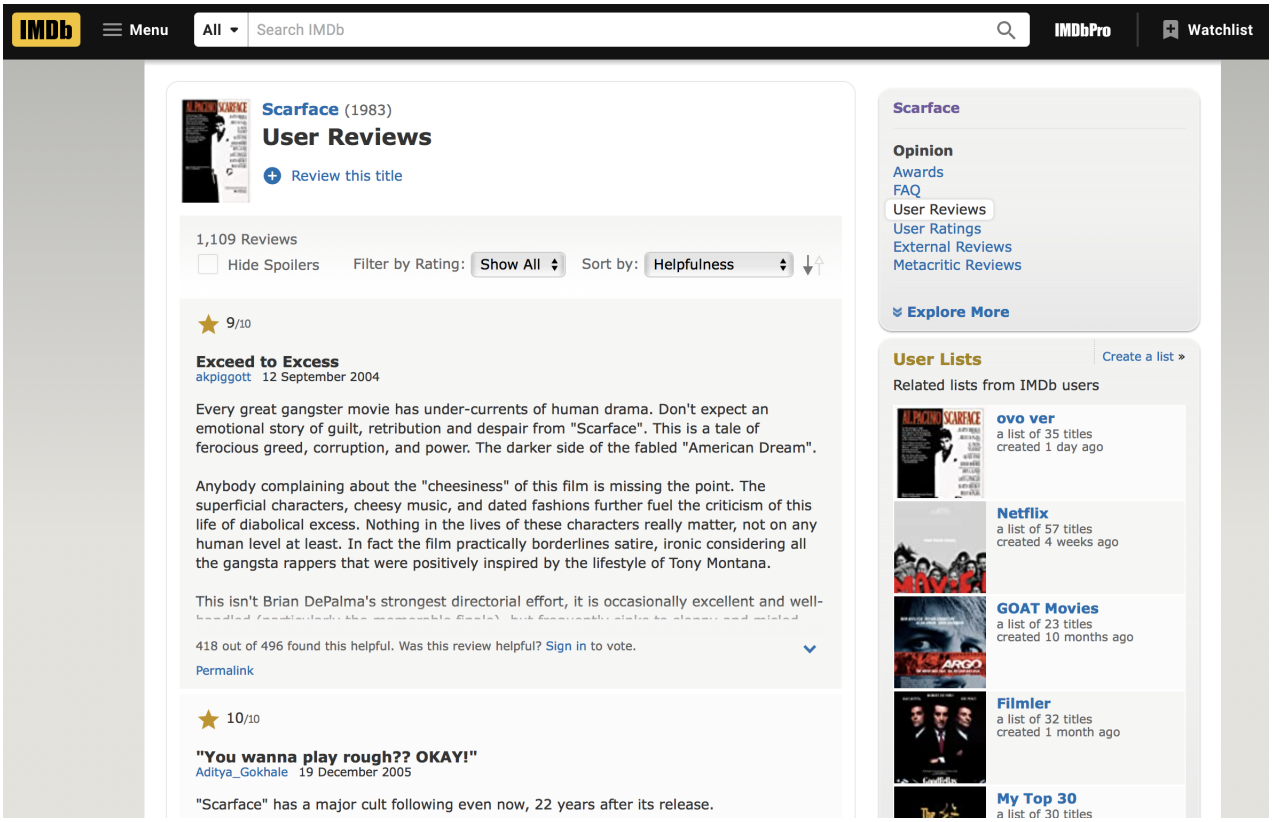

Figure 2: IMDB - user reviews

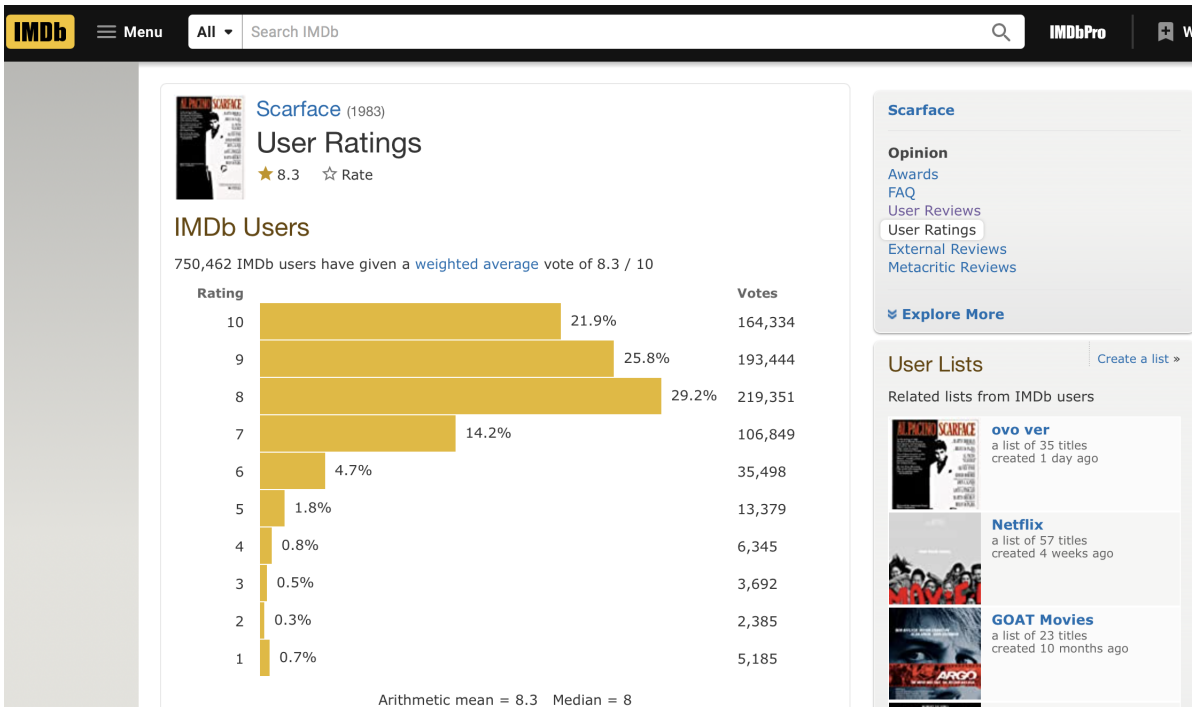

Figure 3: IMDB - user ratings