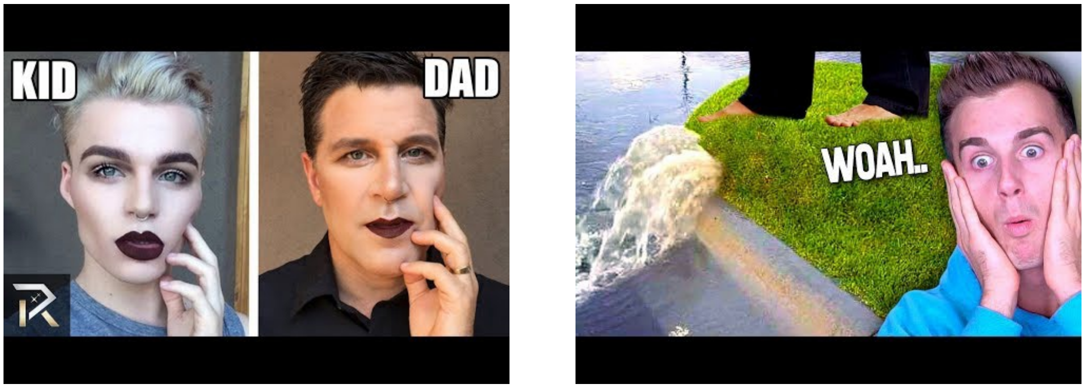
DS 310 Machine Learning

Project #1: Clickbait Thumbnail Classification

Fall 2020 / Amulya Yadav

NOTE: This is a two-member team project.

Task: The aim of this project is to build an accurate binary classification model that can discriminate clickbait YouTube video thumbnails from legitimate ones with high F1-measure. Here is a Wikipedia definition for clickbait: `https://en.wikipedia.org/wiki/Clickbait`. As an example, below are two thumbnails, their titles of YouTube videos, and their corresponding verdicts (i.e., clickbait or non-clickbait). Sometimes, a thumbnail in question is obviously clickbait-y, trying to lure users to click. However, often, it is unclear to determine if a thumbnail in question is clickbait or not, requiring judges to examine the titles and other metadata, or even watch the whole video clip.



(a) 10 Parents Who HILARIOUSLY Embarrassed Their Kids Online

(b) GIANT Grass Bubble POPPING Compilation (Oddly Satisfying)

Figure 1: Clickbait Thumbnail vs. Non-clickbait Thumbnail

Dataset: You can download the dataset on Canvas.

The dataset consists of the following items, after unzipping:

- `train.csv` : a training data in the CSV format

- `test_1.csv` : a test data in the CSV format
- `baseline1.ipynb` : a baseline1 model in Jupyter Notebook
- `baseline2.ipynb` : a baseline2 model in Jupyter Notebook
- `video_download.py` : a python script to download all videos
- `thumbnails` : a folder containing all thumbnail images

The provided training data, `train.csv`, has about 7,000 instances of thumbnails, each of which contains the following features that we have extracted:

- `ID` : a unique identifier of each YouTube video
- `thumbnails/ID.jpg` : a corresponding thumbnail of each YouTube video
- `title` : a title of each YouTube video
- `description` : a description text of each YouTube video
- `timestamp` : when each YouTube video was published
- `viewCount` : # of views of each YouTube video
- `likeCount` : # of Thumbs-Up of each YouTube video
- `dislikeCount` : # of Thumbs-Down of each YouTube video
- `commentCount` : # of user comments of each YouTube video
- `user_comment_1` : top-1 user comment (sorted by YouTube)
- `...`
- `user_comment_10` : top-10 user comment (sorted by YouTube)
- `URL` : URL to each YouTube video
- `class` : **True**=clickbait, **False**=non-clickbait

Using only these features, one can build a reasonably accurate classification model. However, teams are welcomed (and challenged) to think of extra features to improve the accuracies of the model. For instance, if teams want to use YouTube video contents as additional features, they can download videos using the provided script, `video_download.py`. To run the script, note that both `train.csv` and `test_1.csv` files must be in the same folder as the script, `video_download.py`.

Your ultimate task is to build the best learning model using the training data, apply the model to the test data, `test_1.csv`, and generate the output file, `submission.csv`, in the following format of (ID, class), for every instance in the test data:

```
ZluOUS_46ZM, False
svG4y4SsTwl, True
1k0rGopJk38, False
...
```

That is, in this example, a team's best learned model predicts the thumbnail with ID=`ZluOUS_46ZM` as non-clickbait, but the thumbnail with ID=`svG4y4SsTwl` as clickbait, and so on (note that this is a fake example). To help teams, we released two baseline models in Jupyter notebook — `baseline1` and `baseline2` — with varying F1 scores of accuracies.

Kaggle In-class Competition

We set up a Kaggle-based competition environment for only DS310 students below:
https://www.kaggle.com/t/52f64d06ae544978ad94cfb17154bcfe

After teams generate the prediction results for the test data, `test_1.csv`, in the `submission.csv`, they may post it to the Kaggle get the feedback on the accuracy of their model, and their relative ranking via the leaderboard. Note that each team can post *up to 5 times per day to the Leaderboard*.

Kaggle Team Formation

Each student first registers in the Kaggle site and follows the following three steps to team up with the other member as follows:

1. Go to the "Team" tab;
2. Create your own team name;
3. Team up by requesting the "merge" with the other team member.

Each team must consist of a maximum of 2 students.

Canvas Team Formation

Within the Canvas, also form a 2-member team, same as Kaggle team.

Deliverables: Each team needs to accomplish the following tasks:

1. Kaggle Leaderboard: At the end, we take the best ranked entry of each team's submission against the 1st test file, `test_1.csv`, in the leaderboard as the team's final submission result. If a team has not posted any submission to the leaderboard, the team receives 0 point. The higher ranking a team's final submission obtains, the more points are to be awarded.

2. Jupyter notebook: Each team posts its final model in Jupyter notebook, named as `submission.ipynb`, to Canvas. If a team does not post its `submission.ipynb` file at all, the team receives 0 point. If the submitted `submission.ipynb` file does not execute properly, the team receives the penalty score according to the severities of bugs.

3. Project Report: Project report in PDF that contains all the details of the major steps of the project such as:

- Cover page with the **Kaggle team name and member names**;
- Data pre-processing (if any);

- Feature engineering (if any additional features) with explanation;
- Feature engineering (if any additional features) with explanation;
- Explanation of the chosen ML model with rationale;
- Any hyperparameter setting (if any);
- Performance evaluation in training (e.g., show the best F1 score from training data);
- Screenshot of the leaderboard, showing the best F1 score of your submission.

Make your report self-contained such that after reading your report, a student in a data science program should be able to replicate your results. Page is limited to up to 10 pages using reasonable formatting (including cover and all appendix). As such, within the limit, provide all important details of your project as clear/concise as possible. Do not make your report unnecessarily long with figures or codes (i.e., no need to have 10-page report).

Grading Rubric (100 points)

- 80 points: performance (65 points) and ranking (15 points) in the Kaggle Leaderboard, measured in F1-measure, using each team's best `submission.csv` file. 50 points for baseline performance and 15 points for better performance.
- 10 points: submitted Jupyter notebook, `submission.ipynb`.
- 10 points: quality of report.

Grading Exceptions

The goal of this project is to encourage you to emerge yourself in a real-life data science process to learn the materials better in a hands-on fashion. **DO NOT CHEAT IN ANY WAYS.** If you do not attempt to get your hands dirty in this project, you will not learn much.

All team members need to participate in the project and contribute as much as he/she could. Each is expected to bring different skills to the table, i.e, some are good in programming, while others in analytics, etc. Find out what you can contribute within the team, and play the team work. While both members get the same points by default, I will intervene in an exceptional situation (e.g., one member does not participate at all).