**Tasks for Saturday morning:**

1. Data processing/prep: combine data tables into one table while maintaining available data in files. - Vi Anh
2. Assemble Oscar data into table -Yansy
3. Look into complexity of generating AI-generated review - Sergei
4. Look through previous notebooks, review basic methods we are able to apply. What kind of problems can they solve? - Tianyu
5. Look into NLP models that can assess sentiment. -Kerry
6. Come up with the idea(s) by 10 am - Together

**Tasks for Saturday afternoon:**

1. Data processing: finish combining reviews before Oscar nominations announcement date for each movie.
2. Sentiment Analysis: compute Vader score for combined review strings. Difficulty: time complexity. ~~Possible solutions: assign to each teammate, design code that does parallel computing, restrict size of set that we are studying, for example only 2016 and 2017.~~ Current solution: Compute average score for each review, instead of the score for combined strings.
3. Feature Extraction trial: It is going to learn from the combined strings (that was not used in Sentiment Analysis).

**Ideas**

1. Assign genre tags to a movie based on synopsis and/or users’ written reviews.
2. Spoiler Detector. What are the top words in spoilers? Which types of movies get the most spoilers?
3. Predict imdb score on reviews. Build a model to predict the numerical movie score based on a user’s text review.
4. Use reviews to predict which movie will win Best Pictures - compared to the list of movies that actually won Best Pictures.
5. Classify good reviews. Do they relate to spoilers? What words are frequently used? Are they positive or negative according to the library?
6. AI-generated review based on real reviews
   1. Summarize reviews with a predictive model.
7. Classify keywords for movies based on review.

**Timeline**

1. Saturday
   1. Flesh out our project details and delegate final duties by 10:00 AM.
   2. Finish at least one question by the end of the work day.
2. Sunday
   1. Finish presentation by 10:00 AM.

**Updates**

1. First question: Predict Oscar Best Picture nominations based on reviews before nomination announcement. Possible follow-up question: Predict winners based on reviews between nomination announcement and ceremony.
2. Data Processing: Adding dates, and attach Oscar information to original data. Then technical processing: There are multiple reviews for each movie. How do we combine them as one (or more, but we typically wish to keep each movie having features of the same dimension?) features for each movie. Trivial method: Just connect all reviews as one paragraph, possibly use the number of reviews as another factor?
3. NLP Algorithms: Sentiment Analysis (library can provide a score for each review. Then combine scores and do multilinear regression: Ridge vs Lasso) and Feature Extraction (learning algorithms: Logistic Regression and K-Nearest Neighbors). Note that spoilers and non-spoilers can always be treated separately.
4. **\***Combine user reviews into one text for each movie. Use Vader to provide a sentiment score for each movie. (Will it perform better than averaging sentimental scores of each review?) Is there a correlation between sentiment scores and movie score? Are there other features that can relate to the overall sentiment score?
   1. **\*note:** User reviews were combined into a single string for each movie. These strings were long and computationally heavy for the sentiment scoring package. Instead, we will focus on computing the sentiment score for each review and average them across movie title.
   2. Sentiment scores are being computed for each user. We will need to know the sentiment score for users prior to the announcement of Oscar nominations and post Oscar nominations.
5. Features we do not consider include plot summary, review summary, user review date, user review id, and movie release date (keep year). User review date will be used to compute a 0 or 1: 0 if released in the same year as the film being reviewed and 1 otherwise.

Table of thing we can work on

| NLP Algorithms | Learning Algorithms | Features | Predictions for Oscar Best Pictures | |
| --- | --- | --- | --- | --- |
| Sentiment Analysis | Logistic vs KNN | Vader scores | Predict nominations from all movies | Predict winners from all nominations |
|
| Feature Extraction | Combined string |
| Train all year, predict all year | Train old year, predict new year |
| Non-NLP features | Optional for each algorithm | IMDB score, review count, etc |
|

~~Assignment for computing Vader scores for 1572 movies on each laptop, may take 7-8 hours (Tianyu: possibly develop parallel scripts)!~~

~~Vi Anh: 0-314~~

~~Yansy: 315-629~~

~~Kerry: 630-944~~

~~Sergei: 945-1259~~

~~Tianyu: 1260-1572~~

# To read data, try the format below. If your data file is not in current directory, please edit the first “\*” or move the file.

df = pd.read\_json(open("IMDB\_movie\_details.json", "r", encoding="utf8"),lines=True)

Sergei: Working on filtering reviews

Tianyu: Working on Feature Extraction, including combining reviews as strings (done). If regression takes too long, I will only perform predictions from 2016 to 2017.

Vi Anh: Working on the correlation between genres and nominations/ wins.

Yansy: Working on binary multiple linear regression - nominated vs. not nominated, sentiment score vs. user rating, win vs not win.

Data Description:

File **feb4v3.csv**

Columns:

*movie\_id* - movie identification code from the original data

*user\_id* - user identification code from original data

*review\_text* - text of the review from original data

*review\_rating* - rating given by this user

*oscar\_year* - year this movie could get an oscar (if any). For example movies from

1999 could get an oscar in 2000

*date\_flag* - 0 if review was published before nomination, 1 if after nomination but before

ceremony

*oscar\_nom* - 0 if was not nominated for the Oscar, 1 if was

*oscar\_win* - 0 if did not win the Oscar, 1 if did

*duration* - duration of the movie

*genre* - list of genres for this movie

*avg\_rating* - avg rating by all reviews published before ceremony

*avg\_rating\_before\_ann* - avg rating by reviews published before nominations

*summary\_sentiment* - sentiment based on summary of the comment

*review\_sent* - sentiment of comment

Summary of Feature Extraction (by Tianyu)

1. For each movie, combine reviews before Oscar nominations into one string. Here is the bottle-neck of time complexity — it may take 30 minutes.
2. Extract frequencies from text via TF-IDF Vectorizer (similar to workshop 3).
3. Since nominated movies account for a single subset of movies with a small size, Logistic Regression is preferred over KNN. Results: 92.16% on training, 89.78% on testing.
4. Remaining task: Compare performance to Sentiment Analysis.
5. Extra task: Specifically predict Oscar 2017 and 2018 via previous data.
6. Issue: I found actually no winner in our original data. Is that true?

| Accuracy | Train (70%) | Test (30%) |
| --- | --- | --- |
| Nomination | 91.36% | 86.80% |
| Winner | 86.30% | 75% |