**SPOILER ALERT!**

**Applied Machine Learning Project Report Submitted by Group 2**

# **Introduction**

Before we decide to watch a movie, we may look at related online reviews to determine if it would be worthwhile buying the ticket or the DVD. However, reviews that contain spoilers may reduce viewer’s intention from watching or buying the movie, hence adversely affecting the revenue of movie makers. Also, spoilers would prevent viewers from fully enjoying the movie since key plot points were already known beforehand. To deal with spoiler reviews, IMDB currently relied on users to manually tag their reviews as spoiler if their reviews contain key story plots. However, this method could not guarantee the quality control of spoiler posts. Hence, our project focused on auto-detection of spoiler contents in users’ reviews in IMDB via machine learning methods. A potential application would be to tag reviews that contain spoilers automatically and hide it, instead of relying on the users’ discretions.

# **Objective**

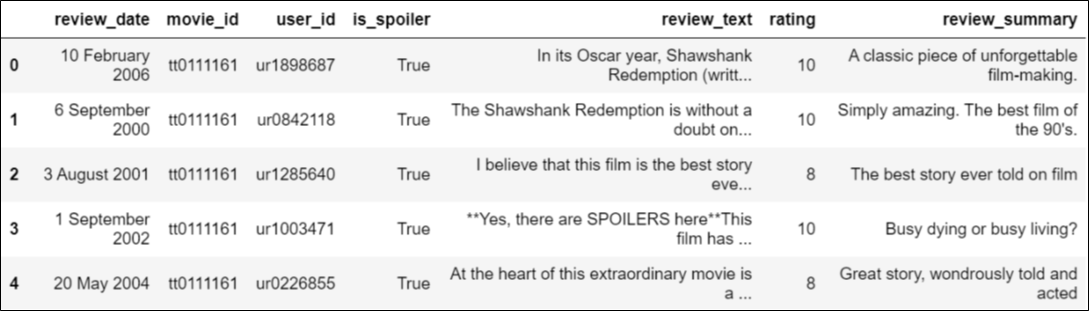
The first objective of our project would be to explore several spoiler detection models given the dataset. The second objective would be to identify the best model and fine-tune it to improve its performance. Lastly, the team would identify the limitations of our model for future improvement.

# **Dataset**

Our data was taken from Kaggle[[1]](#footnote-2). Two datasets were included in our project – the review dataset and the movie dataset. The review dataset contained 573,913 records and **Table 1** described the various attributes. The *is\_spoiler* attribute, which indicated if the review contained spoiler, would be used as labels in our classification models. The *rating* and *review\_text* attributes will be used to build the features for our training.

**Table 1: Review Dataset and Description**

| **Attributes** | **Description** |
| --- | --- |
| review\_date | Date the review was written |
| movie\_id | Unique ID of movie |
| user\_id | Unique ID of the user that wrote the review |
| is\_spoiler | Indicates if the review contains spoiler content |
| review\_text | User review about the movie |
| rating | Rating given by the user on the movie |
| review\_summary | Short summary of the review |

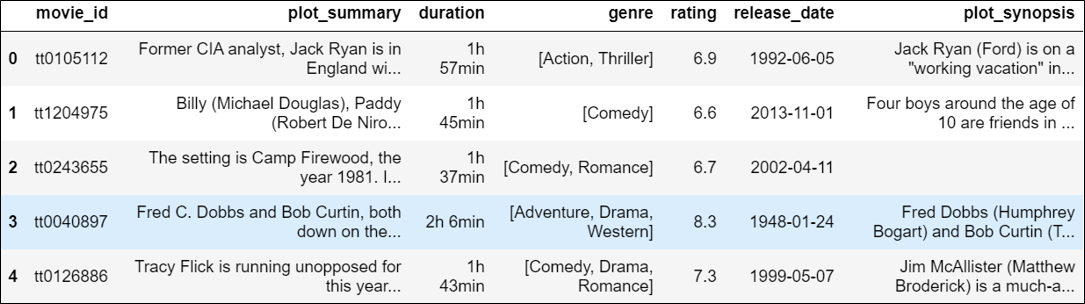
**Figure 1: Preview of Review Dataset** 

The movie dataset contained information of 1,572 movie and **Table 2** described the various attributes. The key attributes were *plot\_synopsis* and *genre*, which would be used to build our training data features as well.

**Table 2: Movie Dataset and Description**

|  |  |
| --- | --- |
| **Attributes** | **Description** |
| movie\_id | Unique ID of movie |
| plot\_summary | Plot summary of the movie |
| duration | Runtime duration |
| genre | Associated genres |
| rating | Overall rating |
| release\_date | The release date of the movie |
| plot\_synopsis | Detail Synopsis of movie's plot |

**Figure 2: Preview of Movie Dataset**



# **Assumptions**

One key assumption was that most movie synopses were available for those movies mentioned in the review dataset. Therefore, we would compare the similarity of the review text against the plot synopsis as one of the features to identify spoiler. Secondly, we assumed that the original spoiler labels were tagged subjectively and correctly. Lastly, we assumed that the reviews were largely written in proper English and that they did not include emoji, other languages and short forms.

# **Overall Methodology**

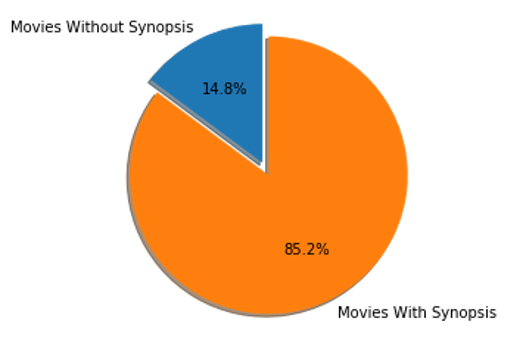
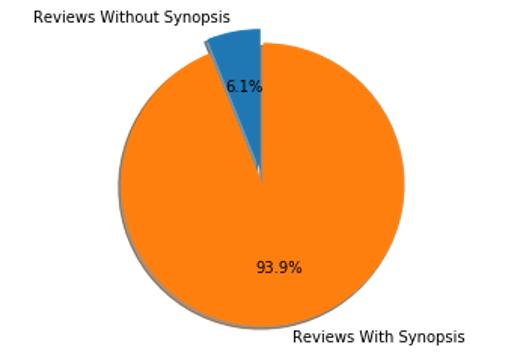
The team would first engineer training data features based on our intuition of how the various parameters could be linked to spoiler. For instance, one intuition was that reviews that were highly similar to the plot synopsis (e.g. high cosine-similarity) would more likely to contain spoilers. The team would then explore the best mixture of engineered features, as well as several models in order to identify the feature and model combination for spoiler detection task.

# **Data Pre-processing**

The pre-processing steps of our data can be summarised as follow:

1. Movie ID Cleaning and Datasets Merging. The team identified and removed unexpected punctuations (e.g. ‘/’) from movie IDs. The review and movie datasets were then merged for ease of comparison.
2. Remove Reviews with Missing Plot Synopsis. Since the comparison of review text and plot synopsis was essential to our feature engineering, those movies without plot synopsis were removed, along with the corresponding reviews. Nonetheless, these only comprised 6.1% of our review data and the team still had approximately 1,339 movies and 539k reviews to work with. (See **Figure 3**)
3. Review Text and Plot Synopsis Pre-processing. The team then perform text tokenisation, removal of punctuations, special characters and numbers, lemmatization, stop words removal and lower casing of​ the review text and plot synopsis. This process ensured that non-essential information were stripped from the data so as to improve the machine learning effectiveness and efficiency.

**Figure 3: Data Breakdown Based on Availability of Corresponding Plot Synopsis**

# **Feature Engineering**

The training features engineered and the intuition behind them were described as follow:

1. Word Vectors. We engineered 4 different word-vector features from the review text and plot synopsis, namely Bag of Words (BOW), Term Frequency Inverse Document Frequency (TFIDF), Document to Vector (Doc2Vec). Both BOW and TFIDF further underwent truncated SVD to reduce the dimensions to 50. As for Doc2Vec, we created two of such features, with dimension 50 as well, using distributed bag of words (DBOW) and distributed memory of mean (DMM).
2. Cosine Similarity. Each of the word-vectors were also used to compute the Cosine Similarity of the review text against the plot synopsis. This would give us 4 different cosine similarity scores. The intuition was that the review text word-vector should be highly similar to the plot synopsis word-vector, if the former contain key words from the plot.
3. Word Count. This feature referred to a simple word count for each review. The intuition was that if the user was writing a spoiler, he would probably need more words to describe the various plot points.
4. Review Rating and Movie Genre. These two features were used “as is” and not an engineered feature per se. The intuition for the selection of review rating was that an user who was discontented with the movie would likely vent his frustration by describing the scenes he/she disliked. This would inevitably reveal key plot points. In terms of movie genre, the team opined that viewers of different genres might have different tendencies of spilling key plots points. For instance, a straight forward action or comedy movie would probably have less in terms of plot points as opposed to those from the thriller genre.

# **Models and Performance**

Given the limited time, the team identified four models for exploration. The model choices and their rationale were described below and the performance was summarised in **Table 3**. We chose True F1-score (True referred to a review being a spoiler) as the benchmark metric because it considered both precision and recall score. We also observed that recall score was higher than precision score which means the model could catch 60-70% of actual true labels but predicted more false positive labels. In addition, Doc2Vec-DBOW word vectors tend to give better results.

1. Logistics Regression (LR) and Gaussian Naïve Bayes (GNB). We chose LR and GNB because these were the classic classifiers with reasonable performance and speed. The best True-F1-score[[2]](#footnote-3) obtained using LR and GNB were 52.3% and 47.7% respectively. GNB operated on the assumption that the features were conditionally independent. In our case, however, the features might in fact be correlated because of our features were extracted from the reviews. Therefore, this might be the reason that GNB performed poorer than the LR model.
2. Random Forest (RF). We chose RF as a form of ensemble learning. RF boosted the overall accuracy to about 73%-75%. However, the True F1-scores were only about 30%-34% , making RF the worst performing model. On further reflection, this could be because some of our features, namely word vectors, were not really quantifiable features per se (such as temperature, time interval, weights), but are abstract representation of the review text. Therefore, building decision branches using different column vector in the word vector matrix might not have helped with segregating the spoilers from the non-spoilers.
3. Neural Network (NN). Neural network was selected to determine if a more advance method could perform better than the more traditional machine learnings. Hence, we chose the same set of features for training the NN. Doc2Vec-DBOW set of features produced the best True-F1-score and increased the True F1-score result to 53.6% Since the original data set already got subjective label on movie comment and we already compared the similarity score the post and movie synopsis, we believed that simple structure of NN could output a better result. Therefore, 3 x 224 nodes Dense layers and 1 output layer was implemented.

**Table 3: Model Performance Summary (F1-Score of True Label)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Features Combination** | **Logistic Regression** | **Gaussian Naïve Bayes** | **Random Forest** | **Neural Networks** |
| word\_count, rating, genre, tSVD\_BOW, sim-tsvd\_BOW | 0.493 | 0.406 | 0.308 | 0.505 |
| word\_count, rating, genre, tSVD\_TFIDF, sim-tsvd\_TFIDF | 0.482 | 0.475 | 0.338 | 0.519 |
| word\_count, rating, genre,  doc2vec-DBOW, sim-d2v\_DBOW | **0.523** | 0.477 | 0.337 | **0.536** |
| word\_count, rating, genre, doc2vec-DMM sim-d2v\_DMM | 0.501 | 0.367 | 0.326 | 0.517 |

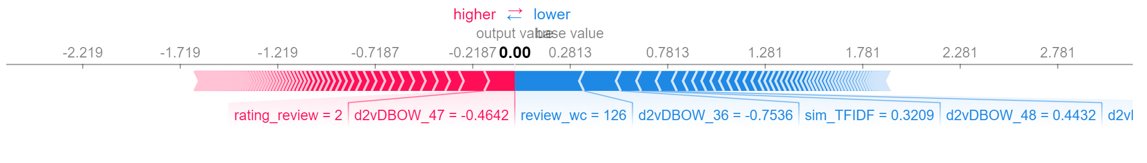
# **Feature Analysis**

**Figure 4: SHAP Value Explanation on Features Importance**



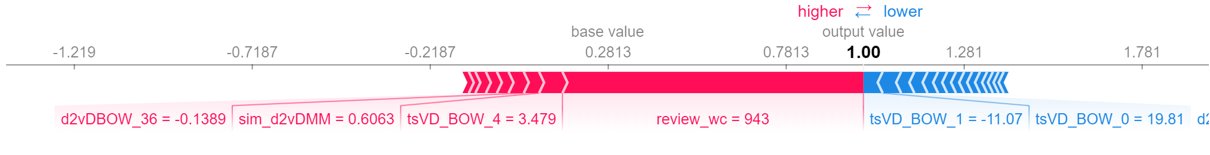
We also performed feature analysis on their importance using Shapley values. We trained a logistic regression classifier using all the features and used 10,000 test data for SHAP Analysis. **Figure 4** showed that review word count was the most important feature to predict the spoiler comment. Attributes from d2vDBOW vectors also featured prominently. This further supported our earlier finding where doc2vec-DBOW give better prediction result (**Table 3**). Although we compared the review similarity with synopsis, this feature was not the top 5 important ones. For the number of word count, when the number is higher, the model learned higher positive impact on predicting the true spoiler. When the number of word count around the mean, the impact was ambiguous.

**Figure 5: Force Plot of Important Features for Predicting a False Label**



**Figure 5** showed the important features to predict the False spoiler. Review word count, d2vDBOW\_47 and d2vDBOW\_36 contributed significantly to the predicted result.

**Figure 6: Force Plot of Important Features for Predicting a True Label**



**Figure 6** showed the important features in predicting True spoiler label. Review word count had a weight of 943, indicating its impact on the model in accurately predicting spoilers. In fact, the word count feature had over 70% contribution to spoiler detection.

# **Further Exploration**

The team had a hypothesis that some users might not have correctly labelled their reviews. If this hypothesis was correct, then by correcting those mislabelled records, the model prediction should improve. Therefore, the team undertook the manual re-labelling of 1000 reviews. To avoid bias during the manual relabelling process, the team members were not shown the *is\_spoiler* tag and had to relabel the reviews as objectively as possible. **Table 4** showed the number of reviews altered as a result of the re-labelling process.

**Table 4: Summary of Relabelling Result**

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | **Original Label** | **Our Label** | **Count** |
| No Change after Relabelling | Spoiler | Spoiler | 145 |
| Non-Spoiler | Non-Spoiler | 546 |
|  | **Sub-Total** | **691** |
| Relabelled Reviews | Spoiler | Non-Spoiler | 137 |
| Non-Spoiler | Spoiler | 172 |
|  | **Sub-Total** | **309** |

From **Table 4**, one observed that there were 309 reviews that that team opined was mislabelled. Our model then made a prediction based on these 1000 relabelled reviews using LR and NN. In both cases, we observed a remarkable improvement to the f1-score of the True label (i.e. spoilers) by 13% and 14% for the LR and NN models respectively. This exercise demonstrated that the team’s hypothesis was correct and that the model did predict spoiler better than what the original label seemed to suggest.

# **Model Limitations and Future Improvements**

There were two main limitations of our models. Firstly, the models could only handle reviews written in English. This was because translation of non-English reviews would be non-trivial as sentiments and nuances relayed in non-English languages might be lost during translation. Secondly, our models were unable to fully handle the myriad of nuances in human language, for instance sarcasm and ambiguity. See **Figure 7** for an example. Even though the user obviously hinted at the death of the dog at the end of the movie, the language used was ambiguous and our model predicted that this review does not contain spoiler.

**Figure 7: Ambiguity in Movie Reviews**



# **Conclusion**

Automatic spoiler detection remains a challenging and relatively unexplored field. In our attempt, the team managed to obtain a reasonable model performance[[3]](#footnote-4) when benchmarked with other similar works (See Reference 1 and 2 at the end), given the limited time and data we had. We have also explored various combinations of engineered features and models. Specifically, we identified word count and word vectors constructed using Doc2Vec-DBOW as engineered features that worked well with LR and NN models.

# **References**

[1] Wan M.T., Misray R., Nakashole N., McAuley J. (2019). Fine-Grained Spoiler Detection from Large-Scale Review Corpora. University of California, San Diego, Amazon.com, Inc.

[2] Boyd-Graber J., Glasgow K., Zajac J. S. (2013). Spoiler Alert: Machine Learning Approaches to Detect Social Media Posts with Revelatory Information. University of Maryland, UMD iSchool.

1. https://www.kaggle.com/rmisra/imdb-spoiler-dataset [↑](#footnote-ref-2)
2. The True-F1-score refers to the F1-score of the predictions made for those reviews that are spoilers. [↑](#footnote-ref-3)
3. The model’s performance was considered reasonable when benchmarked against recent model, such as SpoilerNet in 2019, which obtained an accuracy of 73.7%. [↑](#footnote-ref-4)