# COMP90049 Assignment 2 Report

**Ruofan Zhang**

## 1 Introduction

The goal of this project is to build machine learning models to predict the genre of a movie based on its audio, visual and textual features.

The data given in this project are derived from the MMTF-14K and MovieLens data sets, which is comprised of three parts: the training set (5240 instances), the validation set (299 instances), and the test set (298 instances).

For each set, there are two files, one describes the features whereas the other one describes the genre labels. Each movie is labeled with a single genre from the following 18 possible ones: action, adventure, animation, children, comedy, crime, documentary, drama, fantasy, film_noir, horror, musical, mystery, romance, sci_fi, thriller, war and western.

As for the features, except for the unique indices movieId and YTId that indicates the movie trailer's URL, they can be generally divided into three categories: metadata features, visual features and audio features.

Metadata features are textual fields that describe the title, year of release and tag of a movie; visual features and audio features are continuous floating point values extracted from the trailer of each movie.

This article consists of 5 sections. Section 2 introduces the feature selection and pre-processing stage of this project, which describes why some features were removed from the raw data, how the textual features were transformed and represented in the numerical format and the discretization of visual and audio features. Section 3 introduces the classifiers implemented in this project, namely Logistic Regression and Neural Network. This section mainly describes the parameter tuning step. Section 4 shows the performance of the trained classifiers on the validation set and re-tuning, and the final result on the test set.

In Section 5, this article recaps the findings and reflects the limitation of this project.

## 2 Pre-processing of features

Feature pre-processing of this project mainly includes two parts: feature selection and transformation. Features were selected based on three criteria: assumptions based on daily life experience, statistical details and techniques such as Principal component analysis (PCA). The transformation includes the matrix representation of the textual features and the discretization of the numerical features.

### 2.1 Metadata features

Metadata features in the data set include 'title', 'year' and 'tag'. Based on daily experience, the year of release can only provide very limited information in predicting the genre of a movie. For example, you can hardly deem a movie an action one simply because it was released in 1995, for other genres of movies were also released in the same year. Thus 'year' was removed from the original data.

'Title' is worth reasoning. Usually the title of a movie tends to be implicit and sometimes they can be misleading. For example, it's difficult to predict the genre of *Forrest Gump* before watching it as it is named after the main character and the title per se is quite neutral. A movie named *Goodfellas* is not necessary about 'good', in fact, it can be a crime movie. To test whether the classifiers can really learn some correlation between the movie title and genre, two training sets were prepared: one with 'title' incorporated and the other without.

'Tag' is an informative feature in predicting the genres. For example, the tag of *Happy Gilmore* includes 'sports', 'comedy', 'hilarious', 'seen_more_than_once', 'clv', which explicitly indicates the genre and this has been verified by its corresponding label. The downside is that it contains multiple redundant words like 'clv', which given the context, should be the abbreviation of Constant Linear velocity. These words were removed by implementing a stop-word list designed for the data.

The pre-processing of 'title' and 'tag' includes 4 steps: a) transformed into lowercase; b) stripped of punctuation marks; c) stripped of numbers and stop words (see Appendix) and d) represented in a binary vector matrix using

Tokenizer from Keras. The matrix of 'title' and 'tag' combined contained 6024 columns while the matrix of 'tag' alone had 189 columns.

### 2.2 Visual and audio features

Values of feature 'avf31', 'avf32' and 'avf104' in the data are all zero. These features were removed.

For the remaining ones, they were discretized to stay consistent with the text matrices using KBinsDiscretizer with kmeans strategy from scikit-learn. Each visual and audio was transformed into a binary form to indicate whether a movie had the feature or not.

The binary-transformed features were then concatenated with the matrix of 'title' and 'tag' combined and the matrix of 'tag' alone to form two training sets and the former had 6148 features while the latter had 313 features.

### 2.3 Dimensionality reduction

Principal component analysis (PCA) was used to implement the dimensionality reduction. Features that explained 95% of the overall variance in the data were kept and the features of the training with 'title' and 'tag' combined were reduced from 6148 to 470 while the ones of the training with 'tag' alone were reduced from 313 to 137.

## 3 Training stage

Logistic Regression and Neural Network were used to build classification models in this project. The training stage mainly involved parameter tuning. GridSearchCV from scikit-learn was used to search for the best parameters.

### 3.1 Logistic regression

The logistic regression model was built by calling LogisticRegression from scikit-learn. 'Penalty' was set to 'l2' and parameters to be searched included 'multi_class' and 'C'. 'Penalty' was set to 'l2' to control the overfitting or underfitting problem. Options for parameter 'multi_class' included 'ovr', which meant 'one vs rest' and 'multinomial' i.e. 'one vs one' to see which strategy worked better in this case. For 'C', it indicates the regularization strength. Options of 'C' included 0.1 and 1. The searching result on the training set with 'title' and 'tag' combined showed that when 'multi_class' was 'ovr' and 'C' was 1, the highest mean accuracy 35.2% was achieved (Figure 1).

| | params | split0_test_score | split1_test_score | split2_test_score | mean_test_score | std_test_score | rank_test_score |
|---|---|---|---|---|---|---|---|
| 0 | {'C': 0.1, 'multi_class': 'ovr'} | 0.365197 | 0.333143 | 0.340206 | 0.346182 | 0.013752 | 3 |
| 1 | {'C': 0.1, 'multi_class': 'multinomial'} | 0.357184 | 0.337722 | 0.343643 | 0.346183 | 0.008146 | 2 |
| 2 | {'C': 1, 'multi_class': 'ovr'} | 0.374928 | 0.333143 | 0.349370 | 0.352480 | 0.017200 | 1 |
| 3 | {'C': 1, 'multi_class': 'multinomial'} | 0.364053 | 0.318832 | 0.341924 | 0.341603 | 0.018463 | 4 |

**Figure 1**- Best parameters on the training set with 'title' and 'tag' combined (Logistic Regression).

The same experiment was run on the training set with 'tag' alone and the result turned out to be the same except that the highest mean accuracy dropped to 33.1% (Figure 2).

| | params | split0_test_score | split1_test_score | split2_test_score | mean_test_score | std_test_score | rank_test_score |
|---|---|---|---|---|---|---|---|
| 0 | {'C': 0.1, 'multi_class': 'ovr'} | 0.341729 | 0.302805 | 0.323024 | 0.322519 | 0.015895 | 3 |
| 1 | {'C': 0.1, 'multi_class': 'multinomial'} | 0.339439 | 0.303377 | 0.323024 | 0.321947 | 0.014742 | 4 |
| 2 | {'C': 1, 'multi_class': 'ovr'} | 0.358901 | 0.313108 | 0.322451 | 0.331487 | 0.019756 | 1 |
| 3 | {'C': 1, 'multi_class': 'multinomial'} | 0.346308 | 0.306239 | 0.316724 | 0.323090 | 0.016966 | 2 |

**Figure 2**- Best parameters on the training set with 'tag' alone (Logistic Regression).

### 3.2 Neural network

This project implemented a one-hidden-layer neural network using MLPClassifier from scikit-learn. 'Learning_rate' was set 'adaptive' and the parameter search space included 'hidden_layer_sizes' (8, 16, 32) and the regularization term 'alpha' (0.1, 1).

The best parameter combination for the training set with 'title' and 'tag' combined was 'hidden_layer_sizes' equaled 8 and 'alpha' equaled 1 and the highest mean accuracy was 34.5% (Figure 3).

| | params | split0_test_score | split1_test_score | split2_test_score | mean_test_score | std_test_score | rank_test_score |
|---|---|---|---|---|---|---|---|
| 0 | {'alpha': 0.1, 'hidden_layer_sizes': (8,)} | 0.344591 | 0.323984 | 0.317869 | 0.328815 | 0.011431 | 4 |
| 1 | {'alpha': 0.1, 'hidden_layer_sizes': (16,)} | 0.333715 | 0.291357 | 0.310997 | 0.312023 | 0.017308 | 5 |
| 2 | {'alpha': 0.1, 'hidden_layer_sizes': (32,)} | 0.321694 | 0.282770 | 0.301833 | 0.302099 | 0.015892 | 6 |
| 3 | {'alpha': 1, 'hidden_layer_sizes': (8,)} | 0.350887 | 0.336577 | 0.348225 | 0.345230 | 0.006214 | 1 |
| 4 | {'alpha': 1, 'hidden_layer_sizes': (16,)} | 0.351460 | 0.330853 | 0.341352 | 0.341221 | 0.008413 | 2 |
| 5 | {'alpha': 1, 'hidden_layer_sizes': (32,)} | 0.353177 | 0.329708 | 0.329897 | 0.337594 | 0.011019 | 3 |

**Figure 3**- Best parameters on the training set with 'title' and 'tag' combined (Neural Network).

The result was the same for the training set with 'tag' alone and the highest mean accuracy declined to 31.9%. (Figure 4)

| | params | split0_test_score | split1_test_score | split2_test_score | mean_test_score | std_test_score | rank_test_score |
|---|---|---|---|---|---|---|---|
| 0 | {'alpha': 0.1, 'hidden_layer_sizes': (8,)} | 0.336005 | 0.295936 | 0.320160 | 0.317367 | 0.016477 | 3 |
| 1 | {'alpha': 0.1, 'hidden_layer_sizes': (16,)} | 0.313681 | 0.283343 | 0.299542 | 0.298855 | 0.012395 | 5 |
| 2 | {'alpha': 0.1, 'hidden_layer_sizes': (32,)} | 0.298798 | 0.269446 | 0.293242 | 0.284162 | 0.016922 | 6 |
| 3 | {'alpha': 1, 'hidden_layer_sizes': (8,)} | 0.333143 | 0.301660 | 0.323597 | 0.319486 | 0.013180 | 1 |
| 4 | {'alpha': 1, 'hidden_layer_sizes': (16,)} | 0.327418 | 0.306239 | 0.321879 | 0.318512 | 0.008968 | 2 |
| 5 | {'alpha': 1, 'hidden_layer_sizes': (32,)} | 0.331998 | 0.298226 | 0.315578 | 0.315267 | 0.013789 | 4 |

**Figure 4**- Best parameters on the training set with 'tag' alone (Neural Network).

Based on the results, it seemed the classifiers did learn a certain correlation between the movie title and its genre as the classifiers tended to have better performance on the training set with 'title' and 'tag' combined. Thus, in the following experiments, the training set used was the one with 'title' and 'tag' combined.

## 4 Validation and test stages

In the validation stage, the trained models in Section 3 were used to perform on the validation set pre-processed following the steps in Section 2 with 'title' incorporated.

To have a baseline to compare against, a dummy classier was implemented. It was built by using DummyClassifier from scikit-learn and the strategy was set 'stratified'. It

predicted the label of a new instance by learning the training set's label distribution. The accuracy on the validation set was 11%.

For the trained logistic regression and neural network models, learning curves were drawn to reflect the performance of them (Figure 5, 6).

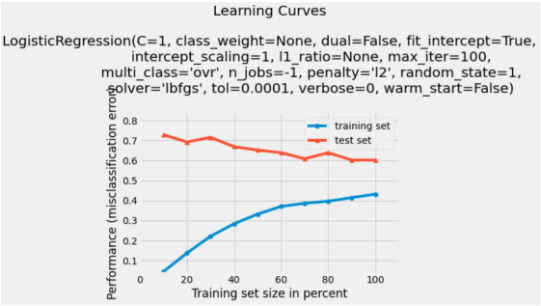


**Figure 5-** Learning curve of the logistic regression model.

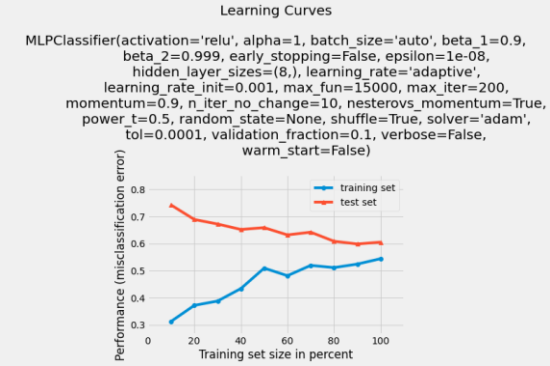


**Figure 6-** Learning curve of the neural network model.

The learning curves showed the sign of underfitting for both models, especially the neural network, as the training error and validation error were both high. To allow the models to fit the training data better, the regularization strength was decreased and maximum iteration rounds were increased. In addition, the layer size was increased for the neural network.

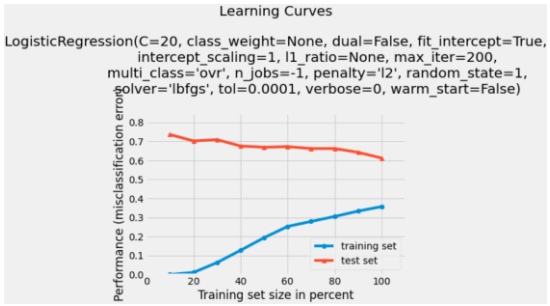


**Figure 7-** Learning curve of the re-tuned logistic regression model.

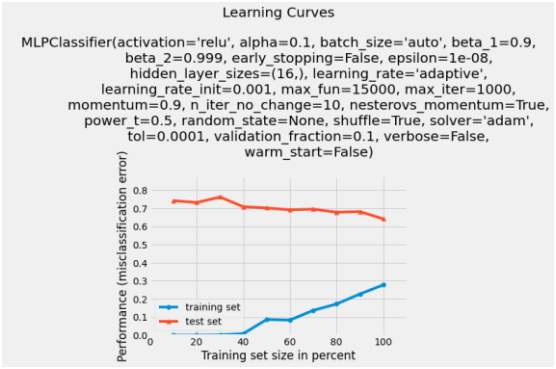


**Figure 8-** Learning curve of the re-tuned neural network model.

After the parameters being re-tuned, the training error of both models declined. Still, the validation error stayed virtually the same.

Pre-processing features had been inspected and different strategies such as keeping the original visual and audio feature values and transforming the texts into tf-idf matrices and scaling them together into the range [0,1] had been tried out. Still, the performance of the classifiers had little improvement.

Using the models trained in Section 3, the validation accuracy of the logistic regression model was 40% and the neural network model scored the same (Figure 9, 10).

The logistic regression model was good at identifying comedy, documentary, science fiction and war movies as the f1-scores of these categories were around or over 0.5, whereas the neural network model was good at identifying comedy, documentary thriller and war movies by the same standard.

They both did not well in predicting action, adventure, children and film noir movies. One reason is that instances of these genres were few in the training set. Another reason might be that it is likely some genres may share common features like action movies and thriller ones. In such a case, a movie can be either tagged 'action' or 'thriller', or 'action' or 'crime'. In other words, the labels are not mutually exclusive and that might be a reason why the performance of the models was limited.

Based on their performance, the logistic regression model was selected as the final classifier to predict the test set and the

Kaggle competition score was 0.3.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Action | 0.00 | 0.00 | 0.00 | 6 |
| Adventure | 0.00 | 0.00 | 0.00 | 2 |
| Animation | 0.33 | 0.33 | 0.33 | 3 |
| Children | 0.00 | 0.00 | 0.00 | 3 |
| Comedy | 0.49 | 0.53 | 0.51 | 38 |
| Crime | 0.25 | 0.40 | 0.31 | 5 |
| Documentary | 0.57 | 0.44 | 0.50 | 18 |
| Drama | 0.35 | 0.53 | 0.42 | 43 |
| Fantasy | 0.55 | 0.33 | 0.41 | 18 |
| Film_Noir | 0.00 | 0.00 | 0.00 | 4 |
| Horror | 0.31 | 0.62 | 0.42 | 8 |
| Musical | 0.14 | 0.10 | 0.12 | 10 |
| Mystery | 0.67 | 0.11 | 0.19 | 18 |
| Romance | 0.36 | 0.41 | 0.38 | 51 |
| Sci_Fi | 0.53 | 0.56 | 0.55 | 16 |
| Thriller | 0.32 | 0.43 | 0.37 | 28 |
| War | 0.67 | 0.38 | 0.48 | 21 |
| Western | 1.00 | 0.14 | 0.25 | 7 |
| accuracy | | | 0.40 | 299 |
| macro avg | 0.36 | 0.30 | 0.29 | 299 |
| weighted avg | 0.43 | 0.40 | 0.38 | 299 |

**Figure 9**- Classification report of the logistic regression model.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Action | 0.00 | 0.00 | 0.00 | 6 |
| Adventure | 0.00 | 0.00 | 0.00 | 2 |
| Animation | 0.00 | 0.00 | 0.00 | 3 |
| Children | 0.00 | 0.00 | 0.00 | 3 |
| Comedy | 0.43 | 0.53 | 0.48 | 38 |
| Crime | 1.00 | 0.20 | 0.33 | 5 |
| Documentary | 0.64 | 0.39 | 0.48 | 18 |
| Drama | 0.34 | 0.60 | 0.44 | 43 |
| Fantasy | 0.47 | 0.39 | 0.42 | 18 |
| Film_Noir | 0.00 | 0.00 | 0.00 | 4 |
| Horror | 0.43 | 0.38 | 0.40 | 8 |
| Musical | 0.33 | 0.10 | 0.15 | 10 |
| Mystery | 1.00 | 0.06 | 0.11 | 18 |
| Romance | 0.33 | 0.37 | 0.35 | 51 |
| Sci_Fi | 0.45 | 0.56 | 0.50 | 16 |
| Thriller | 0.37 | 0.68 | 0.48 | 28 |
| War | 0.80 | 0.38 | 0.52 | 21 |
| Western | 0.00 | 0.00 | 0.00 | 7 |
| accuracy | | | 0.40 | 299 |
| macro avg | 0.37 | 0.26 | 0.26 | 299 |
| weighted avg | 0.44 | 0.40 | 0.37 | 299 |

**Figure 10**- Classification report of the neural network model.

## 5 Conclusion

This project pre-processed the data by removing invalid columns, cleaning the texts and representing them in the binary vector matrix form, discretizing the continuous features also in the binary form and applying PCA to reduce dimension.

Initially there were two training sets prepared with one had 'title', 'tag', visual and audio features combined and the other one without the 'title' to test whether the models can practically learn a certain correlation between a movie title and its genre.

Then this project implemented the logistic regression and neural network models to learn the training sets and find the best parameters. Based on the cross-validation results on the training sets, the models did detect a pattern between the title and the genres as the performance of models tended to be better on the training set with 'title'.

On the validation stage, the learning curves showed the sign of underfitting. After re-tuning of the parameters and re-engineering of the features, the performance still showed little improvement.

The trained logistic regression model was selected as the final classifier to predict the test set and the score was 0.3.

Limitations of this project may include the selection of stop words. The stop words were manually selected by observing the dictionary generated by Tokenizer from Keras. The discretization of the continuous features is worth reasoning although different strategies were also tested in the trial and error stage of this project such as scaling the continuous values with the text matrices together. Still, these didn't work well.

Parameter search space given in Section 3 was also derived from the trial and error stage of this project. And the parameter combination used here might not be optimal as most parameters remained the default.

As mentioned in Section 4, the genre labels are not mutually exclusive in this case and similar genres may share common features like action and thriller movies, which may befuddle the models.

## References


Deldjoo, Y., Constantin, M.G., Ionescu, B., Schedl, M. and Cremonesi, P., 2018, June. MMTF-14K: a multifaceted movie trailer feature dataset for recommendation and retrieval. In *Proceedings of the 9th ACM Multimedia Systems Conference* (pp. 450-455).

Harper, F.M. and Konstan, J.A., 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4), pp.1-19.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1986. Learning representations by back-propagating

errors. *nature*, 323(6088), pp.533-536.

Sammut, C. and Webb, G.I. eds., 2011. Bias–Variance Decomposition. *Encyclopedia of machine learning*. Springer Science & Business Media, pp. 100-101.

Jin, R., Breitbart, Y. and Muoh, C., 2009. Data discretization unification. *Knowledge and Information Systems*, 19(1), p.1.

David W.. Hosmer and Lemeshow, S., 2000. *Applied logistic regression*. New York: Wiley.

Smith, L.I., 2002. *A tutorial on principal components analysis*.

## Appendix

Stop words used in this project:

'bd','from','a','an','with','on','and','for','over','episode','of','in','the','aka','of','clv','dvds','dvd','soundtrack','video','i','ii','iii','iv','imdb','film','more','than','once','betamax','great','seen','top','reviewed','predictable','can\'t','remember','ratings','less','overrated','remake','to','afi','based','true','story','adapted','boring','black','white','national','registry','tumey','erlend','book','classic','ram','franchise','cinematography','world','new','york','city','sequel','england','de','la','cast','ensemble','japan','british','chick','flick','night','original','my','life','big','no','me','last','le','house','multiple','storylines','robert','day','at','all','k','bechdel','test','fail','dialogue','bad','acting','man','serial','narrated','it','american','movie','you','is','who','men','part','little','one','woman','street','mr','out','les','first','that','your','three','living','are','under','days','we','what','or','up','o','good','like','not','two','about','this','how','cat','d','another','don\'t','before','after','oscar','criterion'