Classification of Movie Genres Through Movie Subtitles

Ali Burak ERDOGAN
Comp. Eng. Dept.
Hacettepe University
Ankara, Turkey
erdoganaliburak@gmail.com

***Abstract*— This study aims to perform classification of movies into movie genres, utilizing the subtitles. The first phase of study reached 63% rate of accuracy at classification of subtitles into 8 movie genres. Our experiments showed that the Logistic Regression algorithm scores the best with TF-IDF unigram text models. The second phase of study focused on the dimensionality reduction techniques, singular value decomposition (SVD) and principal component analysis (PCA). SVD and PCA provided 7-10% increase in accuracy scores of SVM and kNN classifiers and we found that PCA brought slightly better improvements to our classifiers.**

Keywords—text classification, subtitles, movie classification

I. Introduction

The task of text classification is highly in demand in our era due to the vast number of textual data present in the digital world. Many of the examples of this task focus on binary classification such as spam filtering, or sentiment analysis. Our project focuses on a supervised learning task for classification of movies making use of the subtitles. We have collected more than 9,000 movie subtitles belonging to 8 different genres successfully and performed various state-of-the-art machine learning algorithms.

II. Related Work

Text classification has been extensively studied and researched over many years. This task requires some basic steps:

- Data collection: retrieving textual data either manually or via automated programs from Web or other media.
- Text Preprocessing: Tokenizing text, stemming words, removal of stopwords and any other useless texts
- Feature Extraction and Selection: Different approaches such as n-grams or unigrams, TF-IDF or Bag-of-Words etc.
- Model Training: Applying machine learning algorithms to build a model

For the feature selection, numerous techniques have been applied in order to have dimensionality reduction due to the high amount of vocabulary in texts. Studies show that TF-IDF with thresholding can be used for computation efficiency and successful representation of data. [1]

Model training step contains numerous alternative algorithm options. Each can be applied and then be compared in terms of efficiency. For example, SVM provides a binary classification capability, however SVM can also be utilized for multiclass classification by using one-vs-rest approach. Naive Bayes algorithm follows a probabilistic approach based on the Bayes theorem for computation of probability of an item belonging to each class.[2]

Since this project's aim focuses on categorizing movies, we can list some studies made for classification of videos benefiting from subtitles. In a study named VIRUS [3], simultaneous analysis of video, audio and subtitle content is performed for video classification. In another study, an unsupervised approach is used for video classification using WordNet [6] lexical database.

III. Methods

A. Data Collection

More than 9,000 subtitles from 8 movie genres as equally distributed in number (Action, Comedy, Crime, Horror, Musical, Romance, War, Western) have been collected by using OpenSubtitles.org API [4] and ScraPy [5] library for automated download of a big amount of subtitles. The distribution of subtitles is shown in figure below:

```
pprint.pprint(collections.Counter(fullgenre))
```

```
Counter({'Horror': 986,
         'Action': 986,
         'War': 986,
         'Comedy': 986,
         'Romance': 986,
         'Crime': 986,
         'Musical': 986,
         'Western': 985})
```

Figure 1. Equally distributed sets of movie subtitles from various genres

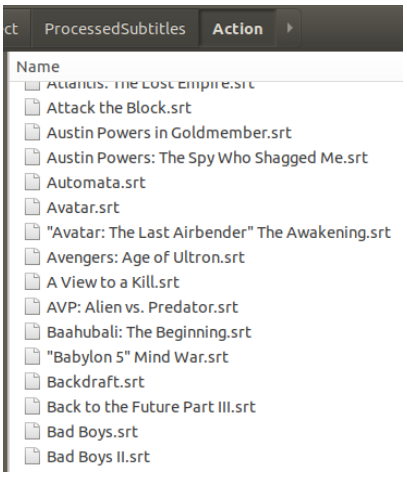


Figure 2. Example view of subtitles data from Action genre

B. Text Preprocessing

Preprocessing subtitle texts is not different than other types of texts except for filtering some subtitle format (.srt) related tags and marks. As can be seen in Figure 3, other than the dialogues, each row includes dialog id, time identifier and some HTML tags for markdown text. Also, there are sometimes sound descriptions enclosed in parentheses for hearing impaired people. During our text preprocessing step, we filter all those out, including time information since we disregard the order of dialogues as well. Next, we apply Porter's stemming algorithm to individual words since words of "loving" and "loves" do not make any difference for our purpose. Next, we filter some common stop words such as "the", "him", "in", "but" by making use of NLTK's stopword list for English. The final output of a preprocessing step can be seen in Figure 4.

```
1
00:00:41,333 --> 00:00:43,586
<i>(rowdy voices and blows landed)</i>

2
00:00:44,002 --> 00:00:46,130
MAN: Come on, put some effort in!

3
00:00:46,838 --> 00:00:48,181
Come on, son!

4
00:00:48,257 --> 00:00:50,305
Right in the kisser!

5
00:00:50,467 --> 00:00:52,515
Keep moving.
That's it, son.
```

Figure 3. Sample subtitle before text preprocessing

```
man
come
put
effort
come
son
right
kisser
keep
move
that
son
```

Figure 4. Sample subtitle after text preprocessing

C. Feature Extraction

1) Bag of Words

Bag-of-words is one of the simplest methods for representing textual data. This approach holds a dictionary of words present in all documents, and stores the occurrences of terms in a given document.

2) TF-IDF

Tf-Idf is another approach for computation of text features in documents. [7] Basically, it's the product of two metrics: term-frequency (tf) and inverse-term-frequency (idf). Tf score is defined as the frequency of a term t in a document d. Idf score of term t is computed as the rareness of it among all of documents D.

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D)$$

The product of those two metrics yields the tf-idf score, and it's a very strong metric for emphasizing important terms with high scores, and decreasing the importance of common words appearing in all classes of texts, for example common verbs like "go", "take".

3) n-gram Model

Two approaches we discussed above do not take word orders into account by default. However, it is possible to try to increase the effectiveness of our models by using n-gram models. For example, instead of computing separate word counts, we could parse the text groups of consecutive two-words, described as a bigram model (2-gram). An example set of bigrams from our preprocessed subtitle data can be seen in Figure 5. As for our dataset consisting of 7887 subtitles, the shape of the matrix for our model is (7887 x 626244) for bigram, and (7887 x 44339) for 1-gram model.

```
vectorizer.vocabulary_.keys()
```

```
'love richard', 'true test', 'like samurai', 'realiz potenti', 'sir t
raffic', 'make flight', 'im broadway', 'andand im', 'make curtain',
'contract need', 'know liter', 'want album', 'progress think', 'la ok
ay', 'okay delay', 'tonight perform', 'mean rachel', 'hold um', 'poss
ibl um', 'spend coupl', 'sidney ive', 'email say', 'tomorrow screw',
'return kind', 'okay sudden', 'heart realiz', 'thing freez', 'use bit
ch', 'screw mean', 'person held', 'huge talent', 'russel crow', 'john
ni carson', 'matter aw', 'santana oh', 'gonna final', 'final alon',
'time couch', 'say june', 'june know', 'know introduc', 'friend ric
h', 'famou way', 'text like', 'hate sweetheart', 'mean ruin', 'green
realli', 'uncertain term', 'finish lie', 'betray im', 'child star',
'star need', 'know mistak', 'broadway legend', 'exist hell', 'great a
udit', 'mean clearli', 'project said', 'want develop', 'oh figur', 'gc
ter', 'celebr friend', 'penlight', 'doj', 'saling', 'epicent', 'unsub
stanti', 'tox', 'criteria', 'alberto', 'ahmet', 'oneminut', 'capitan
o', 'sherwood', 'alitalia', 'real blow', 'open told', 'told bank', 'b
uy 200', 'meet agre', 'ear ye', 'like overnight', 'berlin polic', 'co
me ella', 'odd guy', 'guy foreign', 'tommi close', 'want wit', 'strai
ght voicemail', 'gonna theyv', 'alway theyr', 'gonna stall', 'jesu lo
u', 'lou took', 'tommi dead', 'test neg', 'time immedi', 'immedi minu
```

Figure 5. 2-gram model of a sample subtitle after vectorization

IV. Classification Experiments

A. Selecting the feature model

We used 4 different feature sets and 4 different learning algorithms during our experiments. In order to compare efficiency of n-gram models and make a final decision between Bag-of-Words and TF-IDF approach, we run all learning algorithms with those 4 different feature models. As we can see in Figure 6, **TF-IDF with 1-gram model is the most successful method** in terms of high accuracy. The detailed outputs of those models are listed extensively in Appendix. Another conclusion is that TF-IDF is a more successful method when compared to Bag-of-Words in the majority of cases. This is most probably because TF-IDF takes the inverse-document-frequency into account, which means important words are given more highlighted and resulting in a better classification performance. Another interesting result is that 2-gram models did not increase the accuracy of classification in a significant matter.

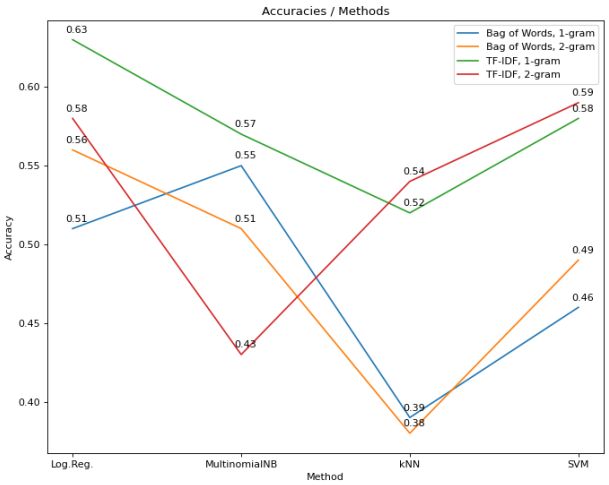


Figure 6. Comparison between feature sets and learning algorithms in terms of accuracy

B. Effectiveness of learning algorithms

Figure 6 shows that Logistic Regression mostly yielded the most accurate classification performance. Since we have 8 classes, the random classification score is 12.5%. Logistic regression with TF-IDF 1-gram model gave 63% accuracy. The second best algorithm is SVM, specifically Linear Support Vector Classifier, with an accuracy score of 58%. Detailed F1-scores with logistic regression and SVM can be observed in figure 7 and 8, respectively.

```
LogisticRegression(max_iter=1000)
time elapsed 24
              precision    recall  f1-score   support

      Action       0.47      0.38      0.42       203
      Comedy       0.44      0.43      0.43       176
       Crime       0.44      0.51      0.48       193
      Horror       0.75      0.78      0.76       198
     Musical       0.69      0.81      0.74       192
     Romance       0.50      0.39      0.44       203
         War       0.76      0.84      0.80       207
     Western       0.88      0.85      0.86       205

    accuracy                           0.63      1577
   macro avg       0.62      0.62      0.62      1577
weighted avg       0.62      0.63      0.62      1577
```

Figure 7. Classification F1-scores of each genre in logistic regression with TF-IDF 1-grams feature set.

```
LinearSVC()
time elapsed 5
              precision    recall  f1-score   support

      Action       0.38      0.35      0.37       203
      Comedy       0.36      0.34      0.35       176
       Crime       0.40      0.45      0.42       193
      Horror       0.70      0.73      0.71       198
     Musical       0.68      0.79      0.73       192
     Romance       0.38      0.33      0.36       203
         War       0.78      0.80      0.79       207
     Western       0.86      0.82      0.84       205

    accuracy                           0.58      1577
   macro avg       0.57      0.58      0.57      1577
weighted avg       0.57      0.58      0.58      1577
```

Figure 8. Classification F1-scores of each genre in SVM with TF-IDF 1-grams feature set.

C. Classification performance of movie genres

Another aspect of our experimental results is the unequal distribution of performance of different genres. As can be seen in Figure 9, F1-scores, which is the harmonic mean of precision and recall scores, are quite high in some genres such as Horror, Musical, War and Western and quite low in some other genres such as Romance, Action, Comedy, Crime.

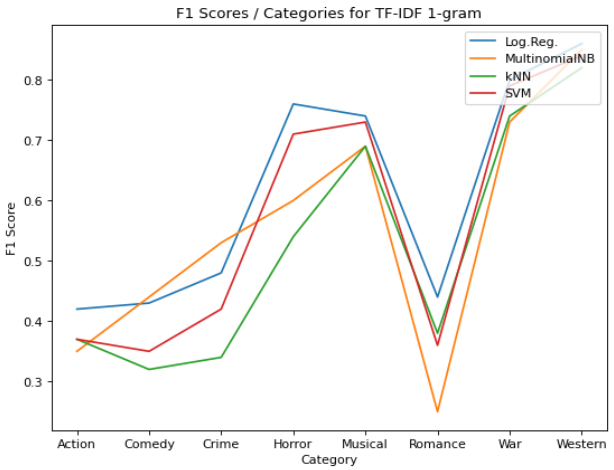


Figure 9. Classification F1-scores of each movie genre

V. Evaluation

To further investigate this situation, we can check the confusion matrix of Logistic Regression with the TF-IDF 1-gram model in Figure 10 since we selected it as the best performing model. Confusion matrices of other models are available in the Appendix.

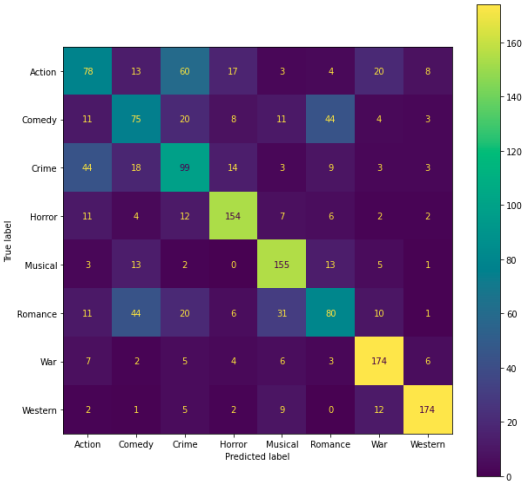


Figure 10. Confusion matrix of Logistic Regression model with TF-IDF 1-gram feature set

Confusion matrix allows us to understand which classes are confused during classification. As we can see Action and Crime genres are the most confused classes. It can be expected due to the common subset of actional words in these two genres. The second most confused classes are Comedy and Romance. We can explain this with the presence of emotional words in those two classes. We further analyzed the presence of common important words among movie genres and generated a heatmap. First, we collected the top-10 words from each movie according to its TF-IDF score. Then, we collected those words into separate lists belonging to specific genres. Then, we compared each genre to another in terms of common words. Figure 11 shows that intersection sets of words are at the highest level in Action-Crime, and Romance-Comedy pairs. Also, we see that the smallest set of common word counts belong to genres Horror, Musical, Western and War. This heatmap has a complete correlation with the F1-score table presented in Figure 7. We conclude that, as the number of common set of words increase in two movie genres, the ability to differentiate them during classification becomes more difficult, hence the F1-score decreases.

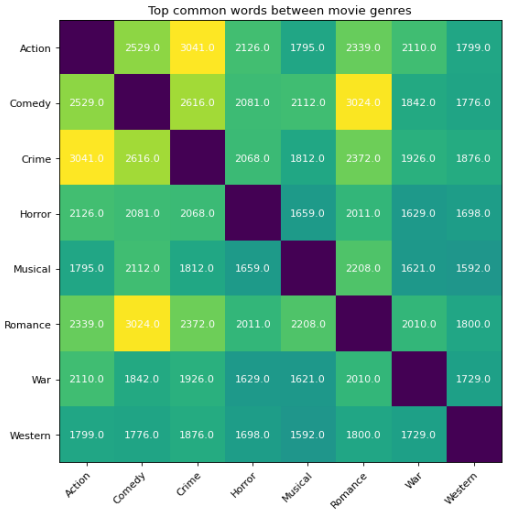


Figure 11. Intersection of most important word lists between movie genres

V. Work Done After Midway Report

In order to make progress on our project, we stated that we plan to apply dimensionality reduction techniques such as PCA and SVD, and compare the results. This section discusses the results and the progress achieved.

A. SVD (Truncated singular value decomposition)

This is a linear dimensionality reduction method mostly used in NLP tasks and also known as *latent semantic analysis (LSA)*. It is generally performed on count/tf-idf vectorizer matrices [8]. It tries to find a linear combination of features in a highly dimensional dataset to obtain an alternative and consistent representation of data. One key difference from the PCA approach is that it is very efficient when working with sparse word-frequency matrices (mostly filled with zero-values) [8]. We applied SVD only to our feature set TF-IDF matrix with 1-grams since it scored the best among others (Bag-of-Words with 2-grams, Bag-of-Words with 1-grams) in our prior experiments. Our tf-idf matrix with 1-grams consisted of 44,339 features, we tried different parameters for feature reduction and the best accuracy score was obtained with 100 features with SVD method, by using Sci-kit library [10]. Figure 12 presents the scores obtained with 100 features only.

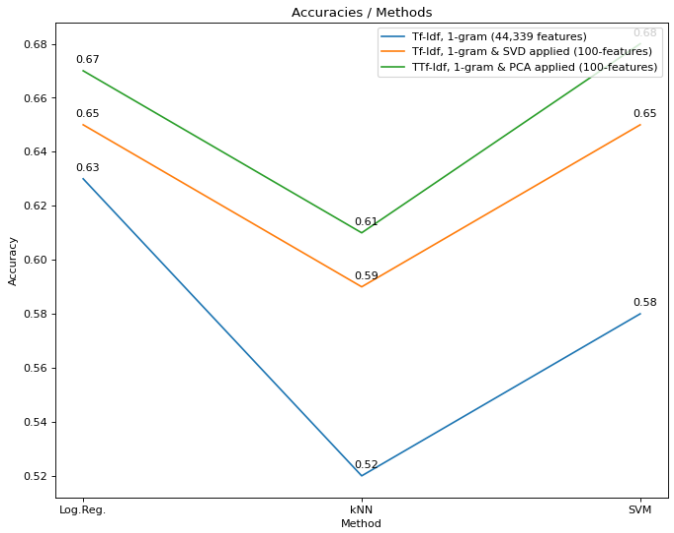


Figure 12. Comparison of accuracy scores when dimensionality reduction methods (SVD or PCA) applied and not applied

In figure 12, it can be seen that multinomial naive bayes scores are omitted. The problem we encountered is that dimensionality reduction methods convert the features into a centered format between 1 and -1 and this situation disallows the performing Multinomial Naive Bayes method.

We observed the highest increase in accuracy with the SVD method in k-Nearest Neighbors algorithm as 7%. SVD also increased the accuracy of SVM by 7%. Logistic regression did not have an improvement as high as others, nevertheless improvement occurred as 2%.

The detailed results of SVD in terms of F1-scores can be seen in figure 13 and 14.

```
LinearSVC()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.57      0.41      0.48       189
      Comedy       0.50      0.44      0.47       204
       Crime       0.55      0.60      0.57       203
      Horror       0.71      0.86      0.78       202
     Musical       0.66      0.88      0.75       172
     Romance       0.55      0.36      0.44       219
         War       0.76      0.85      0.80       209
     Western       0.85      0.91      0.88       179

    accuracy                           0.65      1577
   macro avg       0.64      0.66      0.65      1577
weighted avg       0.64      0.65      0.64      1577
```

Figure 13. Detailed F1-scores of SVM method after SVD applied and number of features were reduced to 100.

```
KNeighborsClassifier()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.46      0.48      0.47       189
      Comedy       0.38      0.44      0.41       204
       Crime       0.50      0.56      0.53       203
      Horror       0.75      0.68      0.71       202
     Musical       0.59      0.78      0.67       172
     Romance       0.36      0.22      0.27       219
         War       0.81      0.74      0.77       209
     Western       0.84      0.90      0.87       179

    accuracy                           0.59      1577
   macro avg       0.59      0.60      0.59      1577
weighted avg       0.58      0.59      0.58      1577
```

Figure 14. Confusion matrix of kNN method after SVD applied and number of features were reduced to 100.

B. PCA (Principal Component Analysis)

This is another linear dimensionality reduction method which is more generally used compared to SVD. PCA is also meant to "reduce dimensionality to increase interpretability but at the same time minimize information loss" [9]. One handicap is to lack the ability to work with sparse matrices which we use as feature sets for word-frequency. We solved this problem by converting our sparse matrix to a dense matrix using NumPy's *todense()* function then applied PCA procedure by using SciKit library [10]. Even though SVD is a specialized algorithm for feature reduction of NLP problems, we have seen a better improvement with the PCA method in our case.

Figure 12 demonstrates that Logistic regression, kNN and SVM methods saw an improvement in accuracy as 4%, 9% and 10%, respectively.

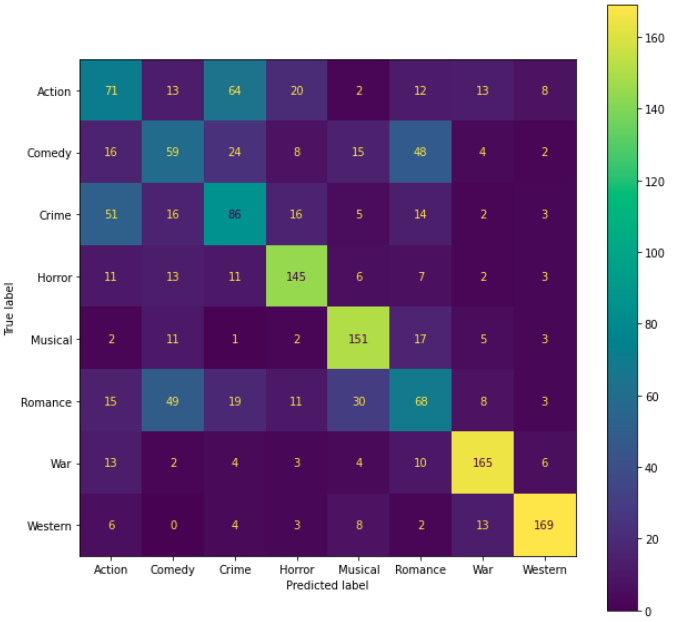


Figure 15. Confusion matrix of SVM method before PCA applied and number of features were reduced to 100.

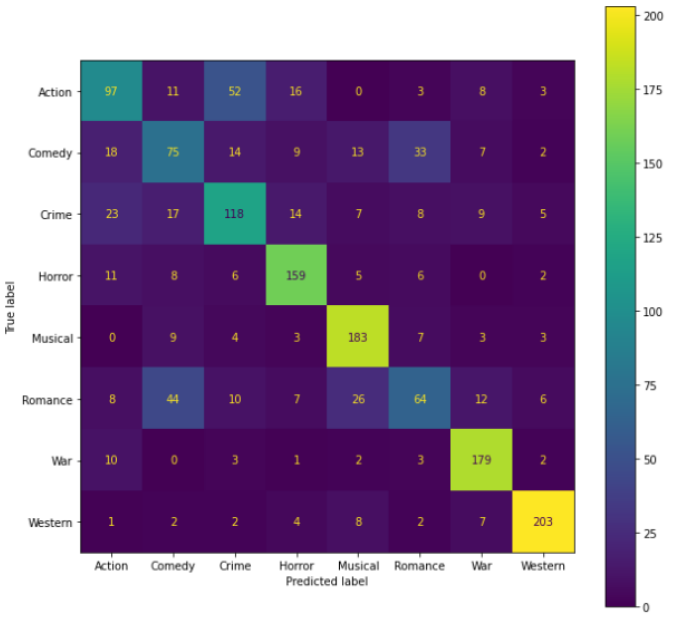


Figure 16. Confusion matrix of SVM method after PCA applied and number of features were reduced to 100.

Confusion matrices of the SVM classifier before and after PCA applied can be seen in Figure 15 and 16 and results can be compared. It is visible that classification confusion between action and crime genres, and comedy and romance genres were decreased due to the PCA method.

VI. Conclusion

In our prior work, we performed the essential parts of text classification on our dataset, and reached 63% of accuracy as the best for classification into 8 categories. We also saw that using n-grams does not increase the overall success rate in a significant manner.

In the remaining part of our study, we applied 2 different dimensionality reduction algorithms on our dataset such as singular value decomposition (SVD) and principal component analysis (PCA). Tf-Idf matrix with 1-grams had 44,309 features in the first place and the number of features were reduced to 100 by those methods. We saw a significant percentage of increase in accuracy, 10% in SVM method and 9% in kNN method due to the PCA. Another technique, SVD, also provided 7% improvement for SVM and kNN methods. It is also worthy to note that logistic regression did not see an improvement as high as kNN and SVM with dimensionality reduction.

VI. Future Work

In our study we did not take into consideration the semantic aspect of texts. There are noteworthy studies which focused on meanings of the words during classification such as synonymity or orderings of the words, for instance using WordNet as a source of semantic information. [6, 11, 12] We leave this as a future work to experiment the effect of semantic analysis in movie genre classification using subtitles.

References

[1] Y. Yang, and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In Proceedings of the Fourteenth International Conference on Machine Learning.. D. H. Fisher, Ed. Morgan Kaufmann Publishers, San Francisco, CA (1997) 412- 420

[2] Katsiouli P., Tsetsos V., Hadjiefthymiades S. 2007. Semantic video classification based on subtitles and domain terminologies Workshop on Knowledge Acquisition from Multimedia Content.

[3] Langlois, Thibault & Chambel, Teresa & Oliveira, Eva & Carvalho, Paula & Marques, Gonçalo & Falcão, André. (2010). VIRUS: Video information retrieval using subtitles. Proceedings of the 14th International Academic MindTrek Conference: Envisioning Future Media Environments, MindTrek 2010. 197-200. 10.1145/1930488.1930530.

[4] OpenSubtitles API. https://opensubtitles.stoplight.io/docs/opensubtitles-api/open_api.json

[5] ScraPy. https://scrapy.org/

[6] Fellbaum, Christiane (2005). WordNet and wordnets. In: Brown, Keith et al. (eds.), Encyclopedia of Language and Linguistics, Second Edition, Oxford: Elsevier, 665-670.

[7] SPARCK JONES, K. (1972), "A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL", Journal of Documentation, Vol. 28 No. 1, pp. 11-21. https://doi.org/10.1108/eb026526

[8] Scikit-Learn Documentation Web page. Truncated SVD. https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.TruncatedSVD.html#sklearn.decomposition.TruncatedSVD

[9] Jolliffe Ian T. and Cadima Jorge 2016. Principal component analysis: a review and recent developments. Phil. Trans. R. Soc. A.3742015020220150202 http://doi.org/10.1098/rsta.2015.0202

[10] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[11] Sarkar, Koushiki & Law, Ritwika. (2015). A Novel Approach to Document Classification using WordNet.

[12] K. Celik and T. Güngör, "A comprehensive analysis of using semantic information in text categorization," 2013 IEEE INISTA, 2013, pp. 1-5, doi: 10.1109/INISTA.2013.6577651.

Appendix

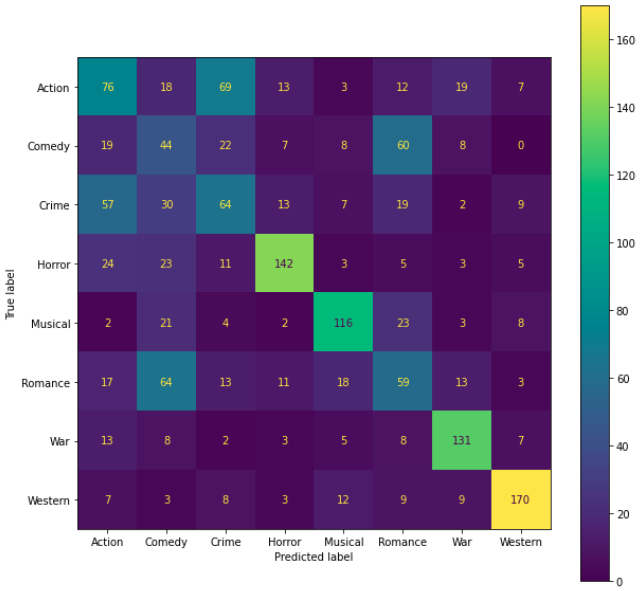
This appendix lists the training and testing results on different models for reference.

Logistic Regression

Bag-of-Words and 1-gram model:

time elapsed 303

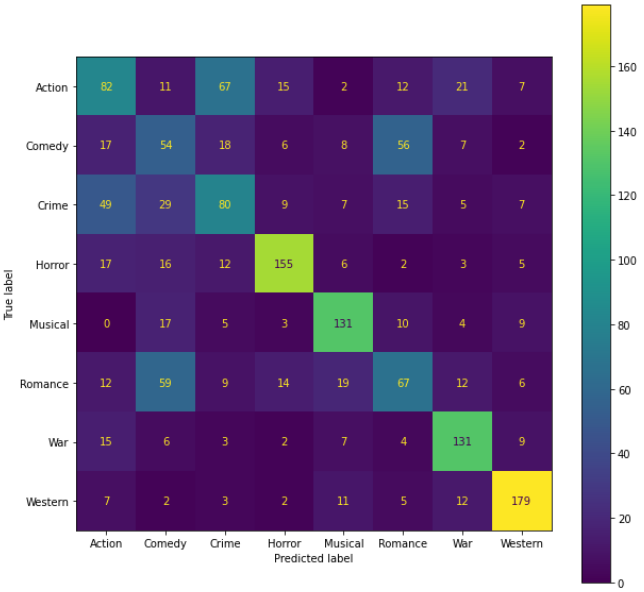
	precision	recall	f1-score	support
Action	0.35	0.35	0.35	217
Comedy	0.21	0.26	0.23	168
Crime	0.33	0.32	0.32	201
Horror	0.73	0.66	0.69	216
Musical	0.67	0.65	0.66	179
Romance	0.30	0.30	0.30	198
War	0.70	0.74	0.72	177
Western	0.81	0.77	0.79	221
accuracy			0.51	1577
macro avg	0.51	0.51	0.51	1577
weighted avg	0.52	0.51	0.51	1577



Bag-of-Words and 2-gram model:

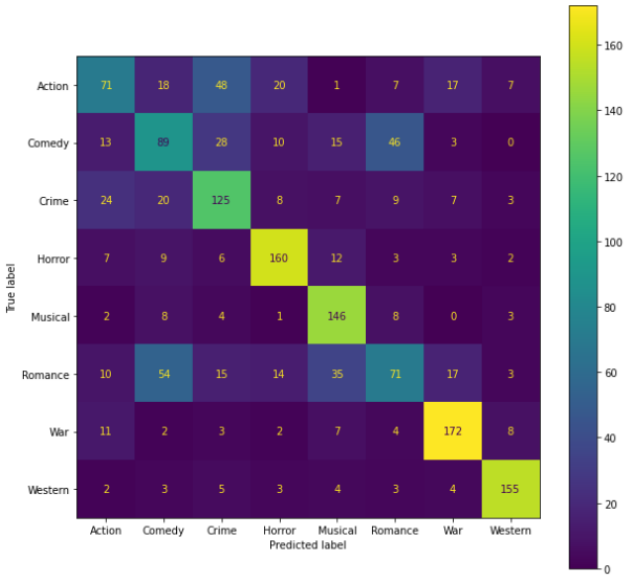
time elapsed 259

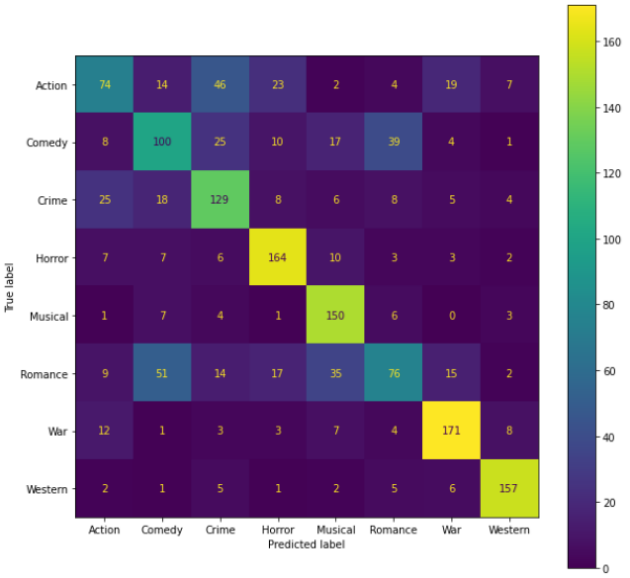
	precision	recall	f1-score	support
Action	0.41	0.38	0.39	217
Comedy	0.28	0.32	0.30	168
Crime	0.41	0.40	0.40	201
Horror	0.75	0.72	0.73	216
Musical	0.69	0.73	0.71	179
Romance	0.39	0.34	0.36	198
War	0.67	0.74	0.70	177
Western	0.80	0.81	0.80	221
accuracy			0.56	1577
macro avg	0.55	0.55	0.55	1577
weighted avg	0.56	0.56	0.56	1577



Tf-Idf and 1-gram model with SVD with 1000-features

```
LogisticRegression(max_iter=1000)
time elapsed 3
              precision    recall  f1-score   support

      Action       0.51      0.38      0.43       189
      Comedy       0.44      0.44      0.44       204
       Crime       0.53      0.62      0.57       203
      Horror       0.73      0.79      0.76       202
     Musical       0.64      0.85      0.73       172
     Romance       0.47      0.32      0.38       219
         War       0.77      0.82      0.80       209
     Western       0.86      0.87      0.86       179

    accuracy                           0.63      1577
   macro avg       0.62      0.64      0.62      1577
weighted avg       0.62      0.63      0.62      1577
```



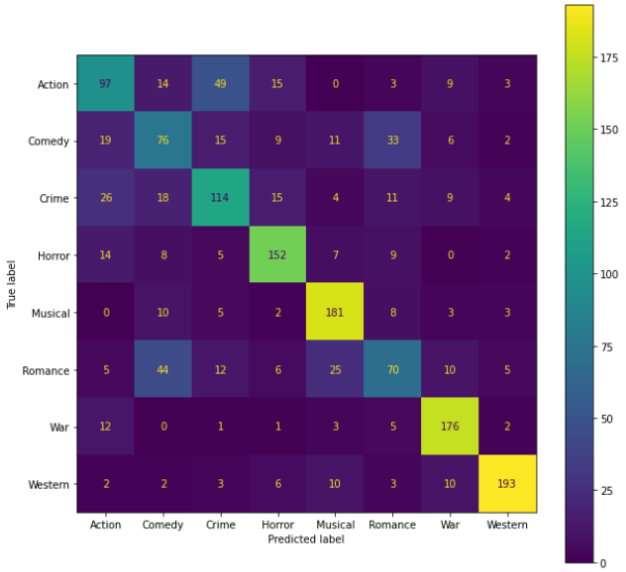


Tf-Idf and 1-gram model with PCA with 100-features:

```
LogisticRegression(max_iter=1000)
time elapsed 1
              precision    recall  f1-score   support

      Action       0.55      0.51      0.53       190
      Comedy       0.44      0.44      0.44       171
       Crime       0.56      0.57      0.56       201
      Horror       0.74      0.77      0.75       197
     Musical       0.75      0.85      0.80       212
     Romance       0.49      0.40      0.44       177
         War       0.79      0.88      0.83       200
     Western       0.90      0.84      0.87       229

    accuracy                           0.67      1577
   macro avg       0.65      0.66      0.65      1577
weighted avg       0.67      0.67      0.67      1577
```

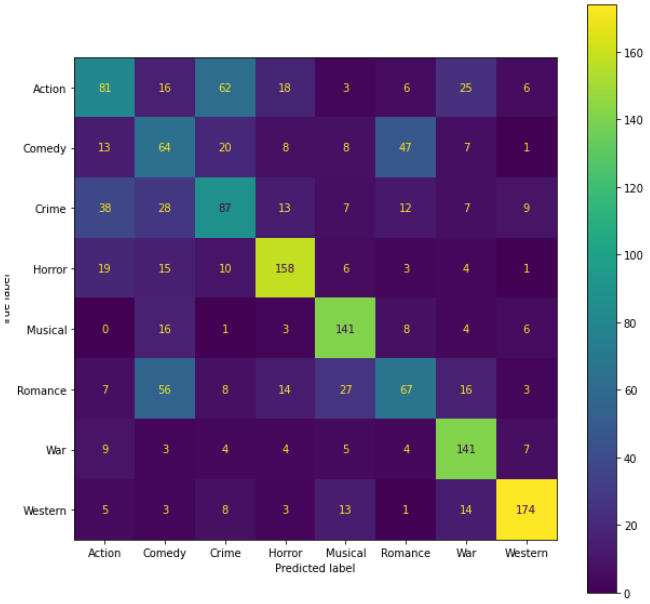
Tf-Idf and 1-gram model with SVD with 100-features:

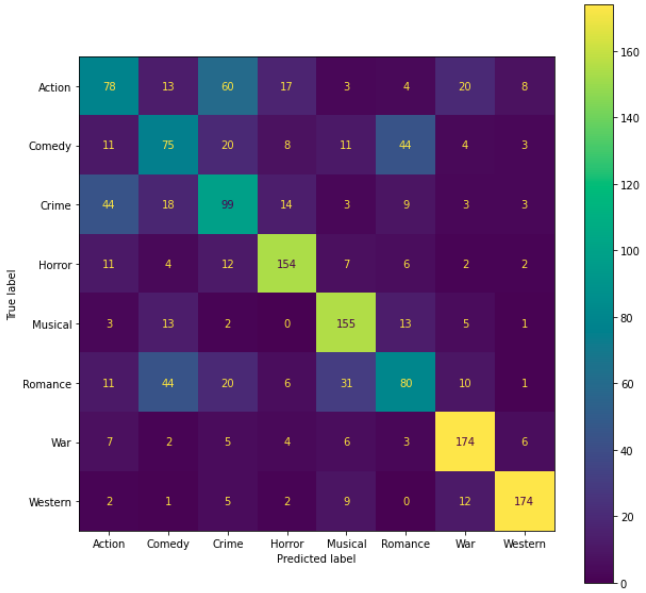
```
LogisticRegression(max_iter=1000)
time elapsed 0
              precision    recall  f1-score   support

      Action       0.54      0.39      0.45       189
      Comedy       0.50      0.49      0.50       204
       Crime       0.56      0.64      0.59       203
      Horror       0.72      0.81      0.76       202
     Musical       0.66      0.87      0.75       172
     Romance       0.52      0.35      0.42       219
         War       0.77      0.82      0.79       209
     Western       0.85      0.88      0.87       179

    accuracy                           0.65      1577
   macro avg       0.64      0.66      0.64      1577
weighted avg       0.64      0.65      0.64      1577
```



Tf-Idf and 1-gram model:

```
LogisticRegression(max_iter=1000)
time elapsed 24
              precision    recall  f1-score   support

      Action       0.47      0.38      0.42       203
      Comedy       0.44      0.43      0.43       176
       Crime       0.44      0.51      0.48       193
      Horror       0.75      0.78      0.76       198
     Musical       0.69      0.81      0.74       192
     Romance       0.50      0.39      0.44       203
         War       0.76      0.84      0.80       207
     Western       0.88      0.85      0.86       205

    accuracy                           0.63      1577
   macro avg       0.62      0.62      0.62      1577
weighted avg       0.62      0.63      0.62      1577
```





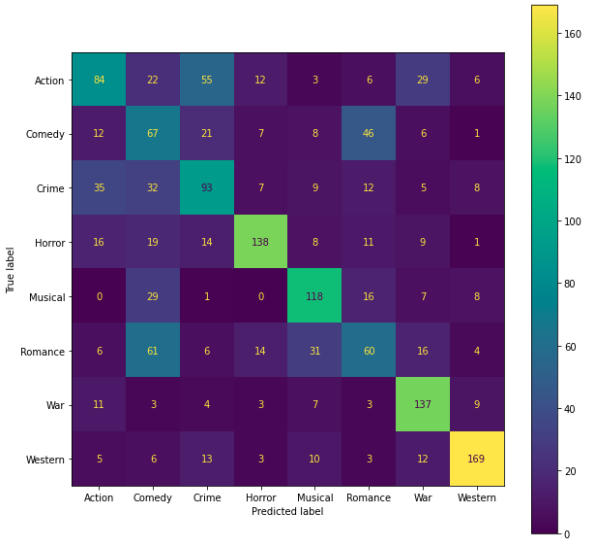
Multinomial Naive Bayes

Bag-of-Words and 1-gram model:

```
MultinomialNB()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.50      0.39      0.44       217
      Comedy       0.28      0.40      0.33       168
       Crime       0.45      0.46      0.46       201
      Horror       0.75      0.64      0.69       216
     Musical       0.61      0.66      0.63       179
     Romance       0.38      0.30      0.34       198
         War       0.62      0.77      0.69       177
     Western       0.82      0.76      0.79       221

    accuracy                           0.55      1577
   macro avg       0.55      0.55      0.55      1577
weighted avg       0.56      0.55      0.55      1577
```

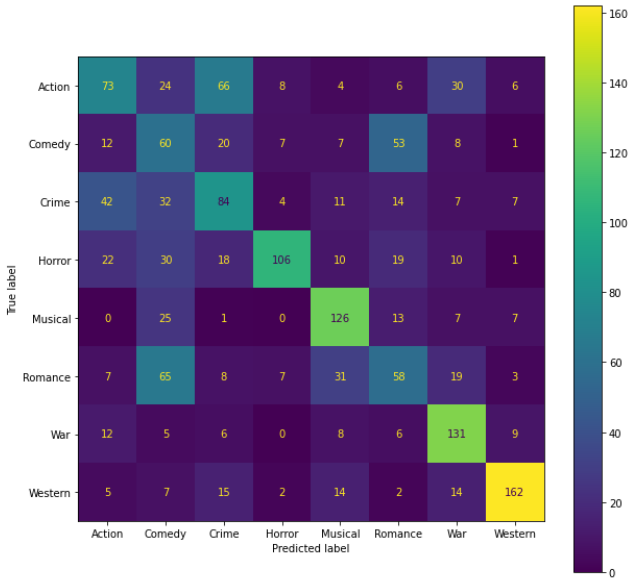
Tf-Idf and 2-gram model:

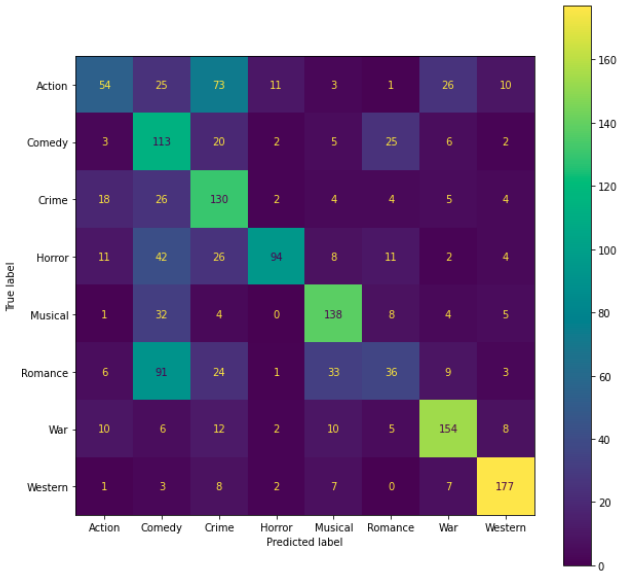
```
LogisticRegression()
time elapsed 158
              precision    recall  f1-score   support

      Action       0.47      0.37      0.42       217
      Comedy       0.32      0.38      0.35       168
       Crime       0.43      0.43      0.43       201
      Horror       0.71      0.73      0.72       216
     Musical       0.67      0.79      0.72       179
     Romance       0.45      0.34      0.39       198
         War       0.65      0.80      0.71       177
     Western       0.84      0.79      0.81       221

    accuracy                           0.58      1577
   macro avg       0.57      0.58      0.57      1577
weighted avg       0.58      0.58      0.57      1577
```



Bag-of-Words and 2-gram model:

```
MultinomialNB()
time elapsed 1
              precision    recall  f1-score   support

      Action       0.42      0.34      0.37       217
      Comedy       0.24      0.36      0.29       168
       Crime       0.39      0.42      0.40       201
      Horror       0.79      0.49      0.61       216
     Musical       0.60      0.70      0.65       179
     Romance       0.34      0.29      0.31       198
         War       0.58      0.74      0.65       177
     Western       0.83      0.73      0.78       221

    accuracy                           0.51      1577
   macro avg       0.52      0.51      0.51      1577
weighted avg       0.53      0.51      0.51      1577
```



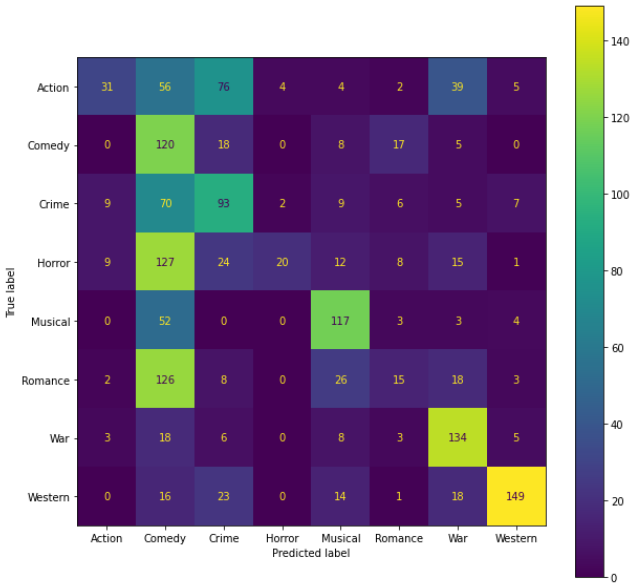


Tf-Idf and 2-gram model:

```
MultinomialNB()
time elapsed 1
              precision    recall  f1-score   support

      Action       0.57      0.14      0.23       217
      Comedy       0.21      0.71      0.32       168
       Crime       0.38      0.46      0.41       201
      Horror       0.77      0.09      0.17       216
     Musical       0.59      0.65      0.62       179
     Romance       0.27      0.08      0.12       198
         War       0.57      0.76      0.65       177
     Western       0.86      0.67      0.75       221

    accuracy                           0.43      1577
   macro avg       0.53      0.45      0.41      1577
weighted avg       0.54      0.43      0.40      1577
```

Tf-Idf and 1-gram model:

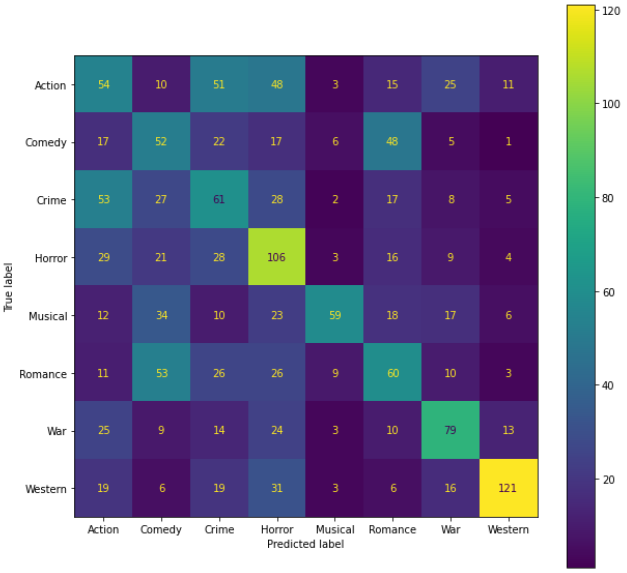
```
MultinomialNB()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.52      0.27      0.35       203
      Comedy       0.33      0.64      0.44       176
       Crime       0.44      0.67      0.53       193
      Horror       0.82      0.47      0.60       198
     Musical       0.66      0.72      0.69       192
     Romance       0.40      0.18      0.25       203
         War       0.72      0.74      0.73       207
     Western       0.83      0.86      0.85       205

    accuracy                           0.57      1577
   macro avg       0.59      0.57      0.56      1577
weighted avg       0.60      0.57      0.56      1577
```

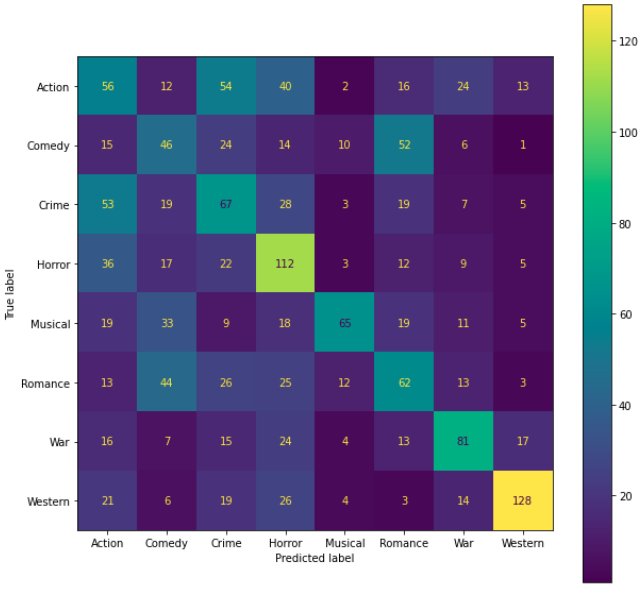


K-Nearest Neighbors

Bag-of-Words and 1-gram model:

```
KNeighborsClassifier()
time elapsed 9
              precision    recall  f1-score   support

      Action       0.24      0.26      0.25       217
      Comedy       0.25      0.27      0.26       168
       Crime       0.28      0.33      0.31       201
      Horror       0.39      0.52      0.45       216
     Musical       0.63      0.36      0.46       179
     Romance       0.32      0.31      0.31       198
         War       0.49      0.46      0.47       177
     Western       0.72      0.58      0.64       221

    accuracy                           0.39      1577
   macro avg       0.42      0.39      0.39      1577
weighted avg       0.42      0.39      0.40      1577
```



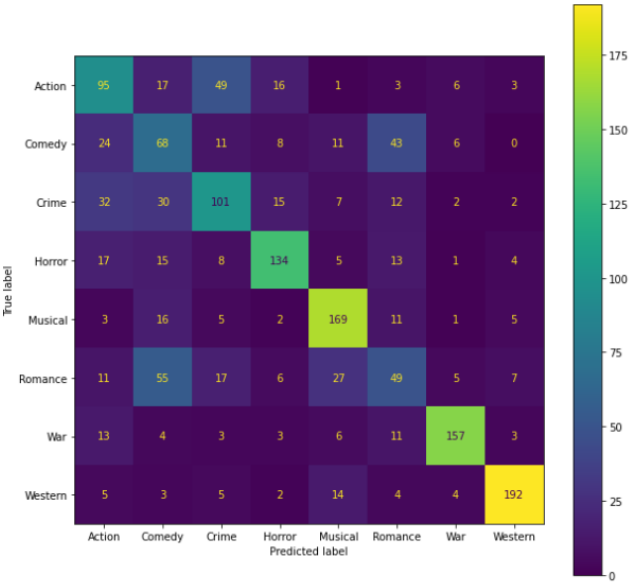


Bag-of-Words and 2-gram model:

```
KNeighborsClassifier()
time elapsed 12
              precision    recall  f1-score   support

      Action       0.25      0.25      0.25       217
      Comedy       0.25      0.31      0.27       168
       Crime       0.26      0.30      0.28       201
      Horror       0.35      0.49      0.41       216
     Musical       0.67      0.33      0.44       179
     Romance       0.32      0.30      0.31       198
         War       0.47      0.45      0.46       177
     Western       0.74      0.55      0.63       221

    accuracy                           0.38      1577
   macro avg       0.41      0.37      0.38      1577
weighted avg       0.41      0.38      0.38      1577
```

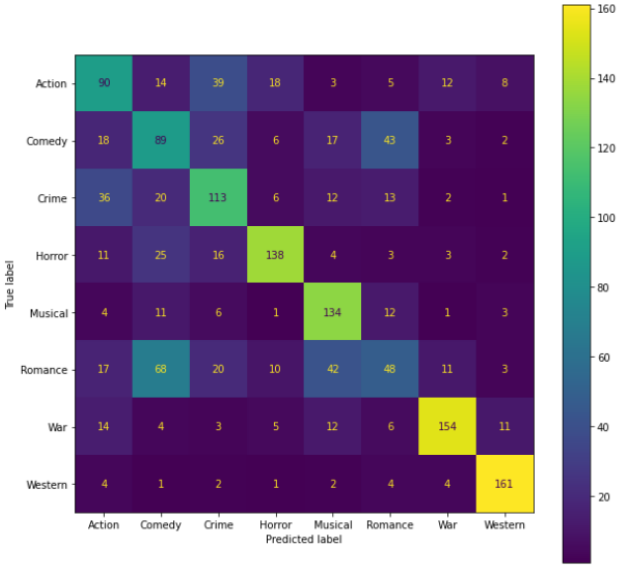
Tf-Idf and 1-gram model with PCA with 100-features:

```
KNeighborsClassifier()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.47      0.50      0.49       190
      Comedy       0.33      0.40      0.36       171
       Crime       0.51      0.50      0.51       201
      Horror       0.72      0.68      0.70       197
     Musical       0.70      0.80      0.75       212
     Romance       0.34      0.28      0.30       177
         War       0.86      0.79      0.82       200
     Western       0.89      0.84      0.86       229

    accuracy                           0.61      1577
   macro avg       0.60      0.60      0.60      1577
weighted avg       0.62      0.61      0.61      1577
```



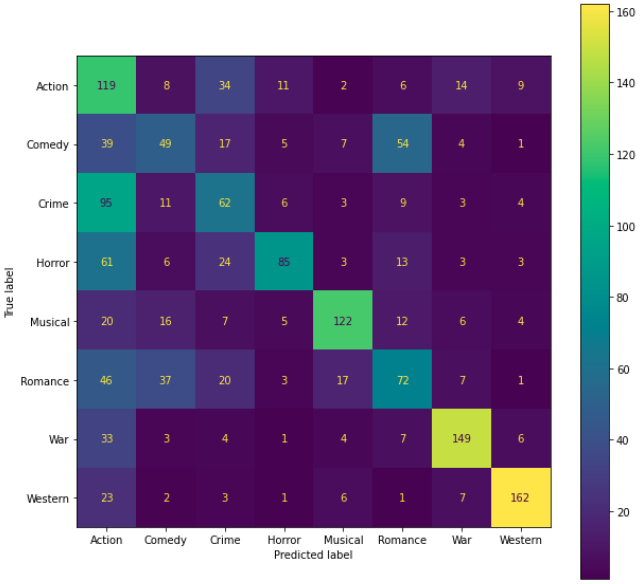
Tf-Idf and 1-gram model with SVD with 100-features:

```
KNeighborsClassifier()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.46      0.48      0.47       189
      Comedy       0.38      0.44      0.41       204
       Crime       0.50      0.56      0.53       203
      Horror       0.75      0.68      0.71       202
     Musical       0.59      0.78      0.67       172
     Romance       0.36      0.22      0.27       219
         War       0.81      0.74      0.77       209
     Western       0.84      0.90      0.87       179

    accuracy                           0.59      1577
   macro avg       0.59      0.60      0.59      1577
weighted avg       0.58      0.59      0.58      1577
```



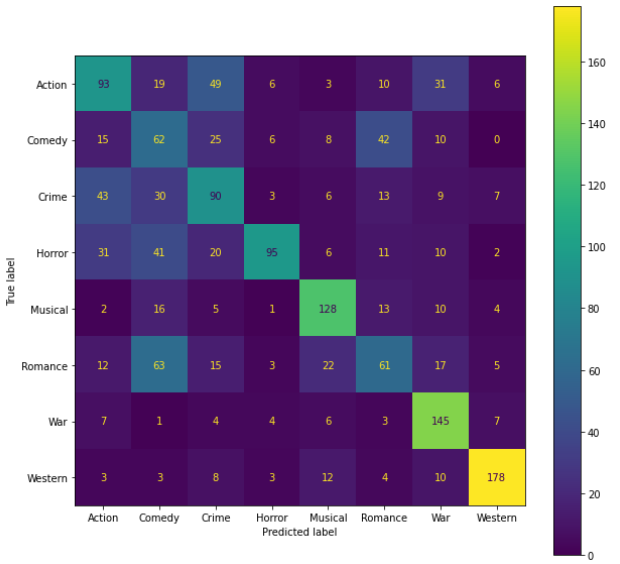
Tf-Idf and 1-gram model:

```
KNeighborsClassifier()
time elapsed 10
              precision    recall  f1-score   support

      Action       0.27      0.59      0.37       203
      Comedy       0.37      0.28      0.32       176
       Crime       0.36      0.32      0.34       193
      Horror       0.73      0.43      0.54       198
     Musical       0.74      0.64      0.69       192
     Romance       0.41      0.35      0.38       203
         War       0.77      0.72      0.74       207
     Western       0.85      0.79      0.82       205

    accuracy                           0.52      1577
   macro avg       0.56      0.51      0.53      1577
weighted avg       0.57      0.52      0.53      1577
```



Tf-Idf and 2-gram model:

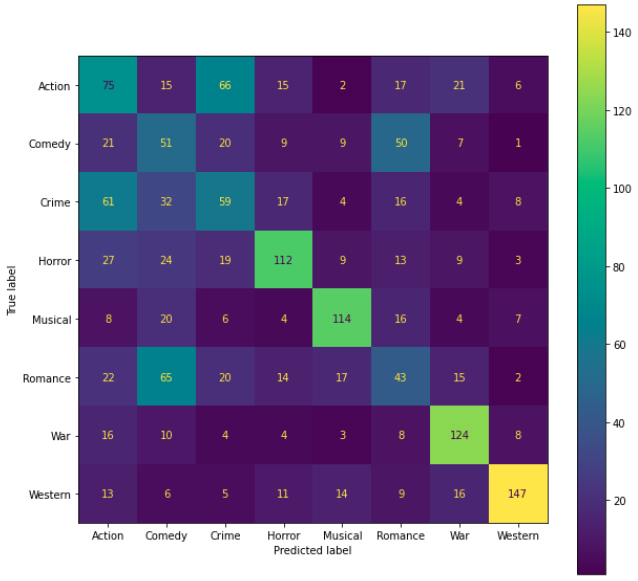
```
KNeighborsClassifier()
time elapsed 11
              precision    recall  f1-score   support

      Action       0.45      0.43      0.44       217
      Comedy       0.26      0.37      0.31       168
       Crime       0.42      0.45      0.43       201
      Horror       0.79      0.44      0.56       216
     Musical       0.67      0.72      0.69       179
     Romance       0.39      0.31      0.34       198
         War       0.60      0.82      0.69       177
     Western       0.85      0.81      0.83       221

    accuracy                           0.54      1577
   macro avg       0.55      0.54      0.54      1577
weighted avg       0.56      0.54      0.54      1577
```



Support Vector Machines

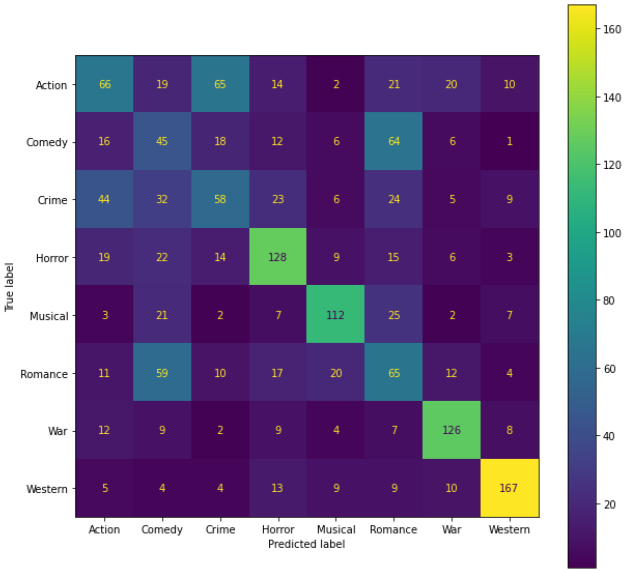
Bag-of-Words and 1-gram model:

```
time elapsed 77
              precision    recall  f1-score   support

      Action       0.31      0.35      0.33       217
      Comedy       0.23      0.30      0.26       168
       Crime       0.30      0.29      0.30       201
      Horror       0.60      0.52      0.56       216
    Musical        0.66      0.64      0.65       179
     Romance       0.25      0.22      0.23       198
         War       0.62      0.70      0.66       177
     Western       0.81      0.67      0.73       221

    accuracy                           0.46      1577
   macro avg       0.47      0.46      0.46      1577
weighted avg       0.48      0.46      0.47      1577
```



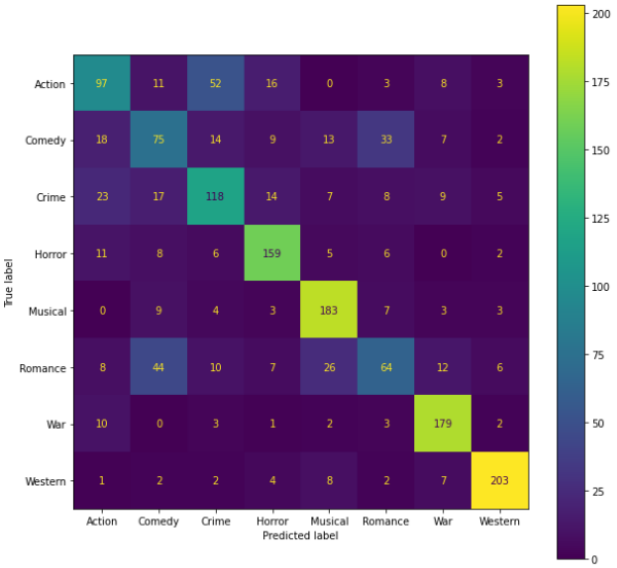
Bag-of-Words and 2-gram model:

```
time elapsed 466
              precision    recall  f1-score   support

      Action       0.38      0.30      0.34       217
      Comedy       0.21      0.27      0.24       168
       Crime       0.34      0.29      0.31       201
      Horror       0.57      0.59      0.58       216
    Musical        0.67      0.63      0.65       179
     Romance       0.28      0.33      0.30       198
         War       0.67      0.71      0.69       177
     Western       0.80      0.76      0.78       221

    accuracy                           0.49      1577
   macro avg       0.49      0.48      0.49      1577
weighted avg       0.49      0.49      0.49      1577
```



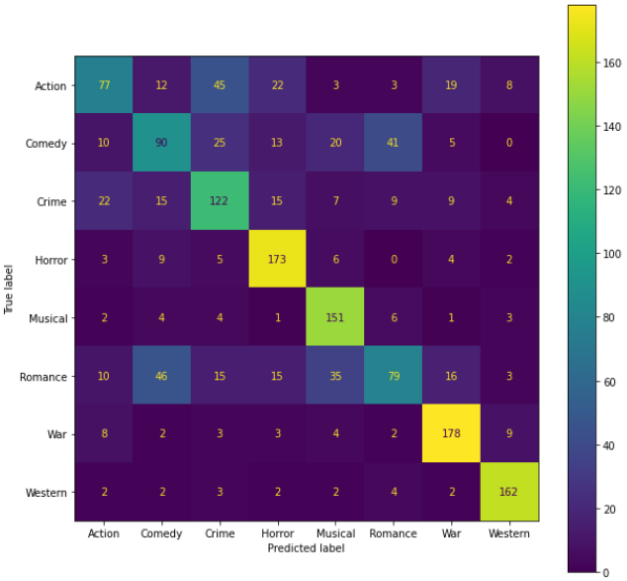
Tf-Idf and 1-gram model with PCA with 100-features:

```
LinearSVC()
time elapsed 1
              precision    recall  f1-score   support

      Action       0.58      0.51      0.54       190
      Comedy       0.45      0.44      0.45       171
       Crime       0.56      0.59      0.58       201
      Horror       0.75      0.81      0.78       197
    Musical        0.75      0.86      0.80       212
     Romance       0.51      0.36      0.42       177
         War       0.80      0.90      0.84       200
     Western       0.90      0.89      0.89       229

    accuracy                           0.68      1577
   macro avg       0.66      0.67      0.66      1577
weighted avg       0.67      0.68      0.68      1577
```



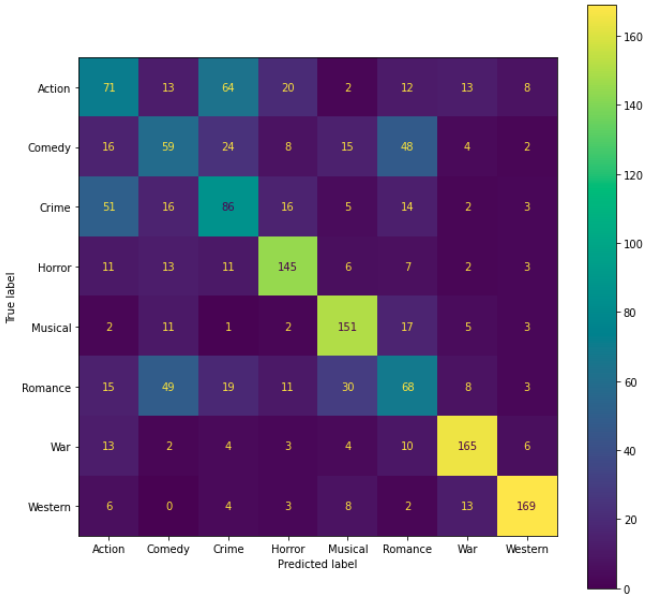
Tf-Idf and 1-gram model with SVD with 100-features:

```
LinearSVC()
time elapsed 0
              precision    recall  f1-score   support

      Action       0.57      0.41      0.48       189
      Comedy       0.50      0.44      0.47       204
       Crime       0.55      0.60      0.57       203
      Horror       0.71      0.86      0.78       202
     Musical       0.66      0.88      0.75       172
     Romance       0.55      0.36      0.44       219
         War       0.76      0.85      0.80       209
     Western       0.85      0.91      0.88       179

    accuracy                           0.65      1577
   macro avg       0.64      0.66      0.65      1577
weighted avg       0.64      0.65      0.64      1577
```



Tf-Idf and 1-gram model:

```
LinearSVC()
time elapsed 5
              precision    recall  f1-score   support

      Action       0.38      0.35      0.37       203
      Comedy       0.36      0.34      0.35       176
       Crime       0.40      0.45      0.42       193
      Horror       0.70      0.73      0.71       198
     Musical       0.68      0.79      0.73       192
     Romance       0.38      0.33      0.36       203
         War       0.78      0.80      0.79       207
     Western       0.86      0.82      0.84       205

    accuracy                           0.58      1577
   macro avg       0.57      0.58      0.57      1577
weighted avg       0.57      0.58      0.58      1577
```



Tf-Idf and 2-gram model:

```
LinearSVC()
time elapsed 36
              precision    recall  f1-score   support

      Action       0.48      0.36      0.42       217
      Comedy       0.32      0.33      0.32       168
       Crime       0.43      0.43      0.43       201
      Horror       0.73      0.79      0.76       216
     Musical       0.68      0.82      0.74       179
     Romance       0.42      0.34      0.38       198
         War       0.69      0.82      0.75       177
     Western       0.85      0.82      0.83       221

    accuracy                           0.59      1577
   macro avg       0.58      0.59      0.58      1577
weighted avg       0.58      0.59      0.58      1577
```