# 1. Genre Prediction

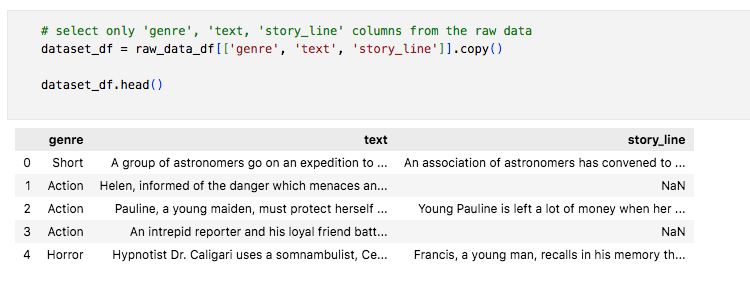
The main goal of this task of the project is to build a classification model which can predict the genre of a movie based on the summary of the movie, obtained from the website IMDB.

The summaries of the movie are paragraphs of sentences and we will apply text analysis techniques on these paragraphs to build a model which can predict the genre of a movie based on the words of the movie descriptions.

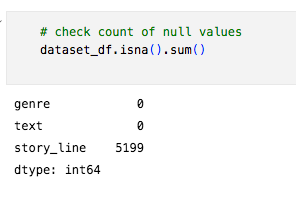
For the purpose of this task, we require only the movie summaries from the scrapped data and this is captured in the column ‘story\_line’ of the dataset. The ‘text’ column in the dataset contains a brief description provided by IMDB for every movie and we will use this column for imputing those records where the ‘storyline’ has missing or invalid data. The ‘genre’ column presents us the class level for each record.

## 1.1 Data Cleaning

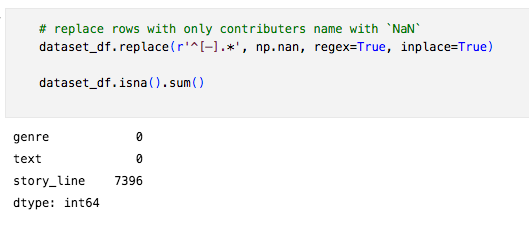
As the first step, we formed a new dataset for the purpose of genre prediction by considering the ‘genre’, ‘storyline’ and ‘text’ columns of the original dataset.



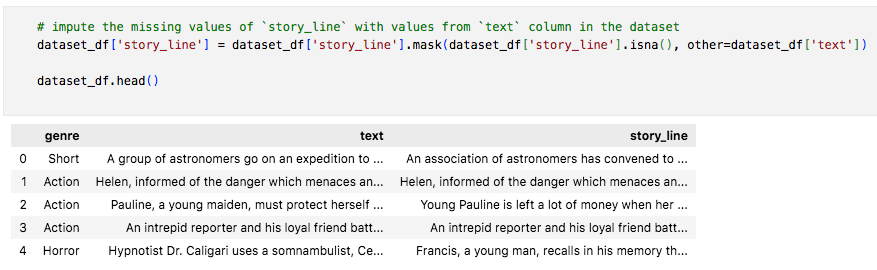
Then we checked for missing values in our dataset and found that there are 5199 missing values for the ‘story\_line’ column.



Additionally, while visually inspecting the dataset, we found many invalid records in the ‘story\_line’ column and all such records had an *Em-Dash* as their first character. So, using regex we identified such records and replaced them with null values. After this, we now had 7396 records with null values.



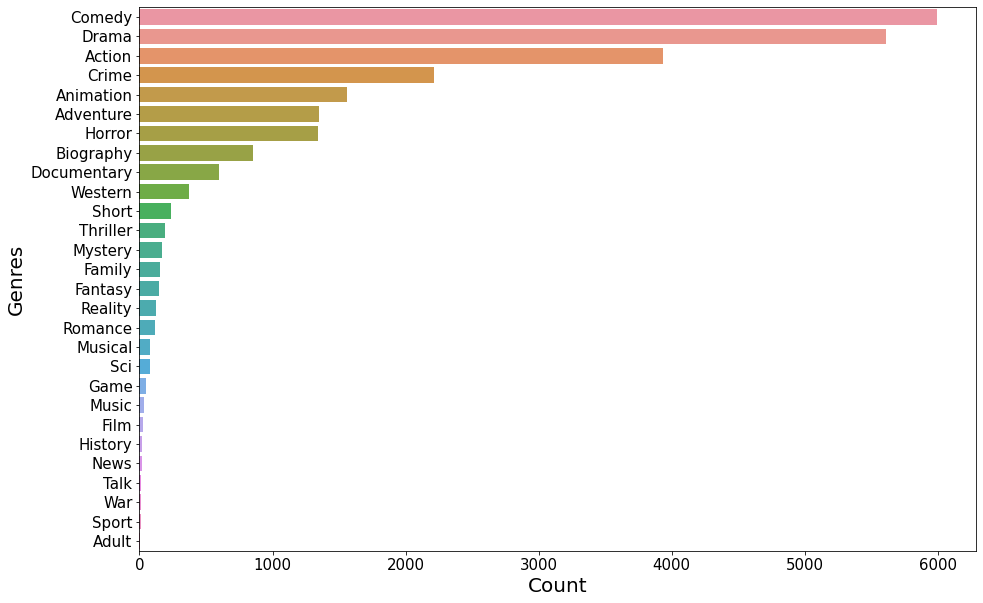
Thereafter as mentioned previously, we imputed the missing records of ‘story\_line’ column with the contents from the ‘text’ column in our dataset.



After imputing the missing values, we removed the ‘text’ column from our dataset. And now our dataset is ready for carrying out text analysis on the ‘story\_line’ column and use the ‘genre’ column as the label for the dataset.

## 1.2 Data Exploration

Before proceeding with the text-analysis, we looked at the distribution of the classes in our dataset. We have 28 unique genres or classes in the dataset and it is a highly imbalanced dataset.



We can see from the above figure that there are about 6000 records of the genre ‘Comedy’ and the count of other genres varies to the single digits in some cases. This shows that the dataset has imbalanced classes.

## 1.3 Data Pre-processing

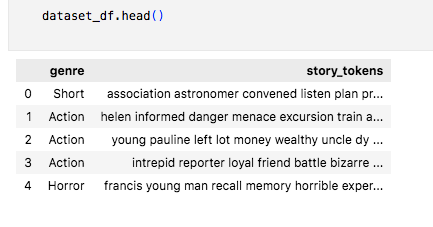
After the exploration, we proceeded with data pre-processing steps to perform text analysis. For the text data in the ‘story\_line’ column, we followed text pre-processing steps discussed in section 1.3.1 and performed encoding of the ‘genre’ column for the classification algorithms.

### 1.3.1 Text Pre-processing

We carried out the below steps to prepare our data for text analysis:

1. **Filter out non-alphabets from the data:** for the purpose of text analysis we considered only alphabets and removed all numbers and special characters from our text data.
2. **Tokenisation:** next we tokenised all the sentences into individual words (tokens).
3. **To lowercase characters:** then we transformed all the tokens into lowercase characters.
4. **Remove Stopwords:** stopwords of English language were then removed as they are not useful for the purpose of text analysis in our case.
5. **Lemmatisation:** Finally, we carried out lemmatisation of the tokens so that all the tokens are transformed to their root word.

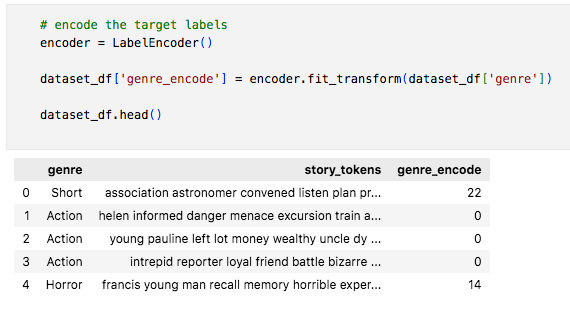
With these above steps, our text data was ready for the analysis. Below is how our data looks after the above transformation steps.



### 1.3.2 Encoding of the class label

We then encoded our class label i.e. the ‘genre’ column into numerical values, so that the machine learning models can understand it. As ‘genre’ is the target or dependent variable in our case, using label-encoding to encode the ‘genre’ column is sufficient.

In label encoding, each unique string of the ‘genre’ column will be assigned a decimal number and we will use this encoded label of the movie genres as the target variable for our classification models.



As we can see from the above figure, each genre in the dataset has been now assigned a decimal number to represent it.

## 1.4 Data Analysis

Now, our dataset was ready for building and training a classification model and predict the genre based on the tokens extracted from the story line of the movies. But before training our classification models, first we split the dataset into training and testing datasets. And after the splitting of the training and testing datasets, we vectorised our text data using different word vectorisation algorithms. Finally, we applied the machine learning algorithms on the vectorised data obtained after word-vectorisation.

### 1.4.1 Splitting the data into training and testing datasets

We began by splitting the dataset into a training and testing dataset. 80% of the dataset was used for training the model and 20% of the dataset was unseen data for the model to test against it. The split was performed in a stratified manner to have equal proportional representation of each class in both training and testing datasets.

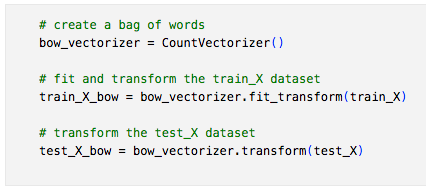


### 1.4.2 Word Vectorisation

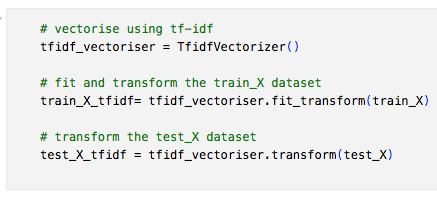
Next, we converted the text data into numerical features so that the algorithms can understand the different tokens. There are different ways to vectorise our text data:

1. Bag of Words (BoW) using Count Vectorizer
2. Term Frequency - Inverse Document Frequency (TF-IDF)

We used the above methods to vectorise our training dataset and compare them later on.

****

Bag of Words using CountVectorizer() from sklearn python library.



TF-IDF using TfidfVectorizer() from sklearn python library.

### 1.4.3 Building Machine Learning models

Finally, we used machine learning algorithms to predict the genre of the movies in the testing dataset. There are different algorithms which we can use for this purpose.

Here we used the following two algorithms:

1. Multinomial Naive Bayes
2. Support Vector Machine (SVM)

## 1.5 Results

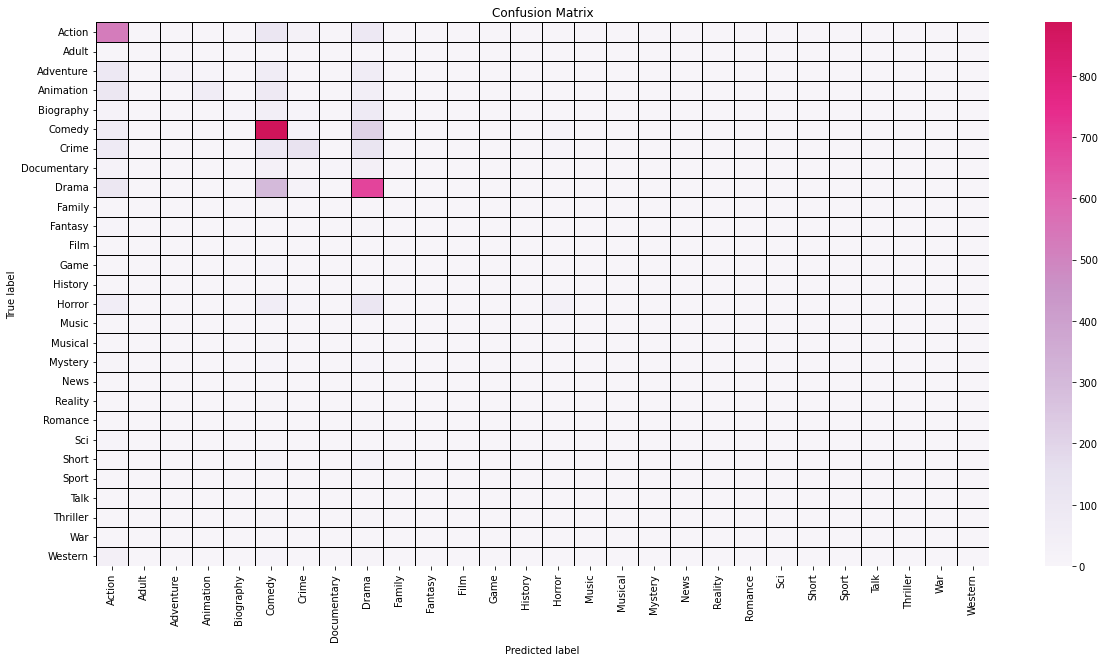
In our project, we applied both the above-mentioned machine learning algorithms on the vectorised data obtained by the two types of word vectorisation algorithm. The following table captures the F1-score of the four models.

|  |  |
| --- | --- |
| Model | F1-Score |
| MNB – with BoW | 0.47 |
| MNB – with TF-IDF | 0.40 |
| SVM – with BoW | 0.45 |
| SVM – with TF-IDF | 0.47 |

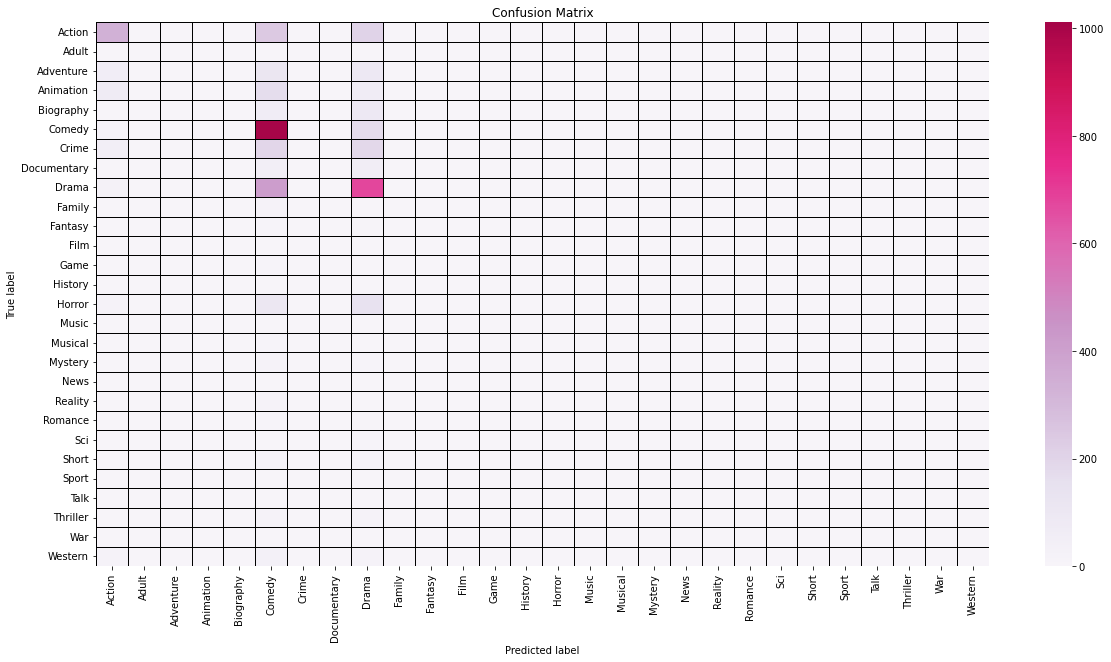
Multinomial Naïve Bayes with Bag-of-Words and Support Vector Machine with TF-IDF models performs the best with the highest F1-score.

And the confusion matrix for the above models are shown below. We can see that all the models perform well in the prediction of only ‘Action’, ‘Comedy’ and ‘Drama’ genres, and these genres of the movies had the highest number of records in the dataset.

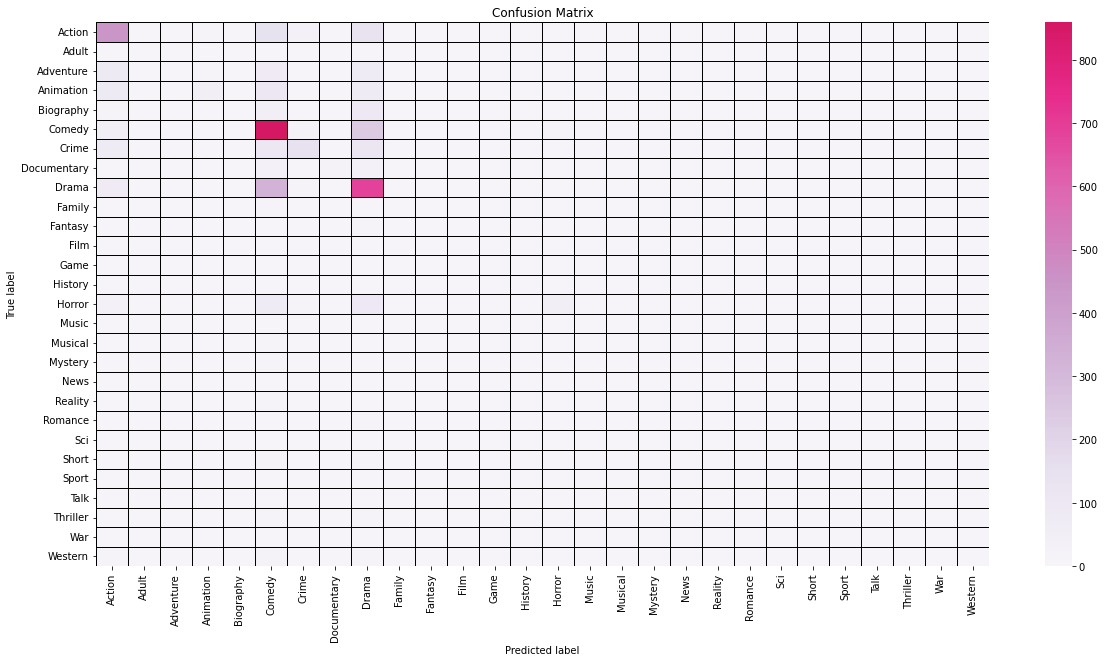
Multinomial Naïve Bayes with Bag-of-Words



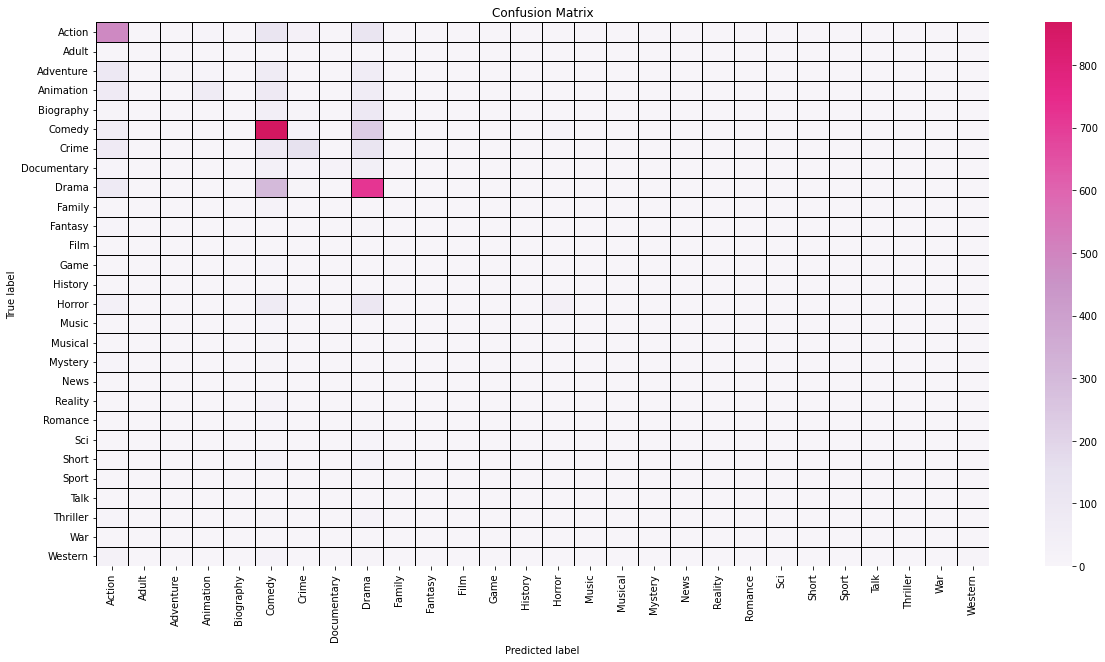
Multinomial Naïve Bayes with TF-IDF



Support Vector Machine with Bag-of-Words



Support Vector Machine with TF-IDF

****