Movie Genre Classification System

INFO 6105 - DSEM - Project Report

August 13, 2019

Kavya Prakash - 001837570 - prakash.ka@husky.neu.edu
Sreerag Mandakathil Sreenath - 001838559 - mandakathil.s@husky.neu.edu

Table of Content

Introduction & Background

Classifying a movie plot into genres was chosen as it provides a wide range of exploratory paths with data science methods and its application can be found in various sophisticated recommendation engines. The project aims at exploring various classifier algorithms, understanding their behaviors and enhancing the classifier accuracy to predict the genre.

Our goal with the project is to:

1. Conduct EDA on the Data
2. Learn and Apply NLP on the plot content
3. Test out various classic and deep machine learning models
4. Build a pipeline for the best result and pickle the models
5. Develop a simple to use web application as an API for new classification

Data Source and Description

The data is taken from Kaggle data source :
https://www.kaggle.com/jrobischon/wikipedia-movie-plots

Content

The dataset contains descriptions of 34,886 movies from around the world. Column descriptions are listed below:

- Release Year - Year in which the movie was released
- Title - Movie title
- Origin/Ethnicity - Origin of movie (i.e. American, Bollywood, Tamil, etc.)
- Director - Director(s)
- Genre - Movie Genre(s)
- Wiki Page - URL of the Wikipedia page from which the plot description was scraped

- Plot - Long form description of movie plot

	Release Year	Title	Origin/Ethnicity	Director	Cast	Genre	Wiki Page	Plot
0	1901	Kansas Saloon Smashers	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/Kansas_Saloon_Sm...	A bartender is working at a saloon, serving dr...
1	1901	Love by the Light of the Moon	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/Love_by_the_Ligh...	The moon, painted with a smiling face hangs ov...
2	1901	The Martyred Presidents	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/The_Martyred_Pre...	The film, just over a minute long, is composed...
3	1901	Terrible Teddy, the Grizzly King	American	Unknown	NaN	unknown	https://en.wikipedia.org/wiki/Terrible_Teddy,_...	Lasting just 61 seconds and consisting of two ...
4	1902	Jack and the Beanstalk	American	George S. Fleming, Edwin S. Porter	NaN	unknown	https://en.wikipedia.org/wiki/Jack_and_the_Bea...	The earliest known adaptation of the classic f...

Acknowledgements

This data was scraped from Wikipedia

	Genre	Count
116	drama	9487
85	comedy	7320
4	action	5952
426	thriller	3291
337	romance	2639
94	crime	1607
244	musical	951
16	animation	914
73	children	684
131	fantasy	542
245	mystery	481
46	biography	463
47	black	412
149	history	256
368	short	241
114	documentary	131
391	sports	121
357	series	86
6	adult	71
288	political	60
90	costume	49
429	tokusatsu	43
411	supernatural	41
223	masala	41
307	psycho	39

Methodology

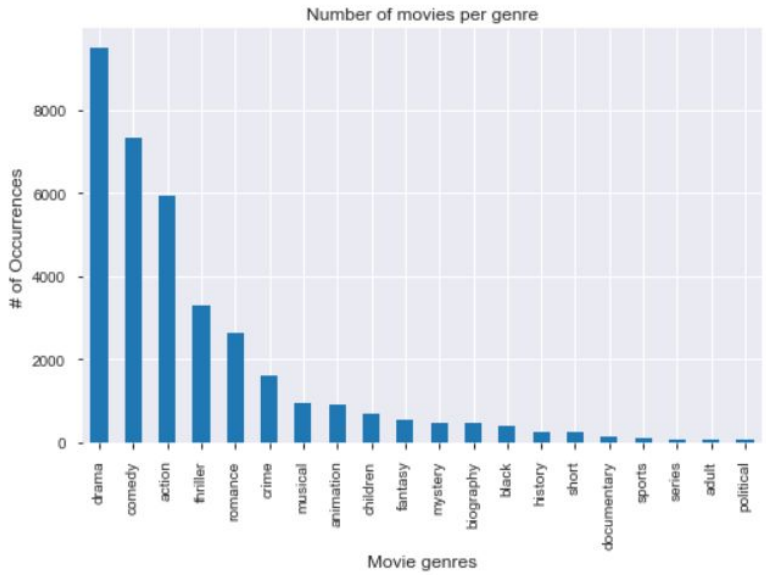
Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an approach for data analysis that employs a variety of techniques (mostly graphical) to :

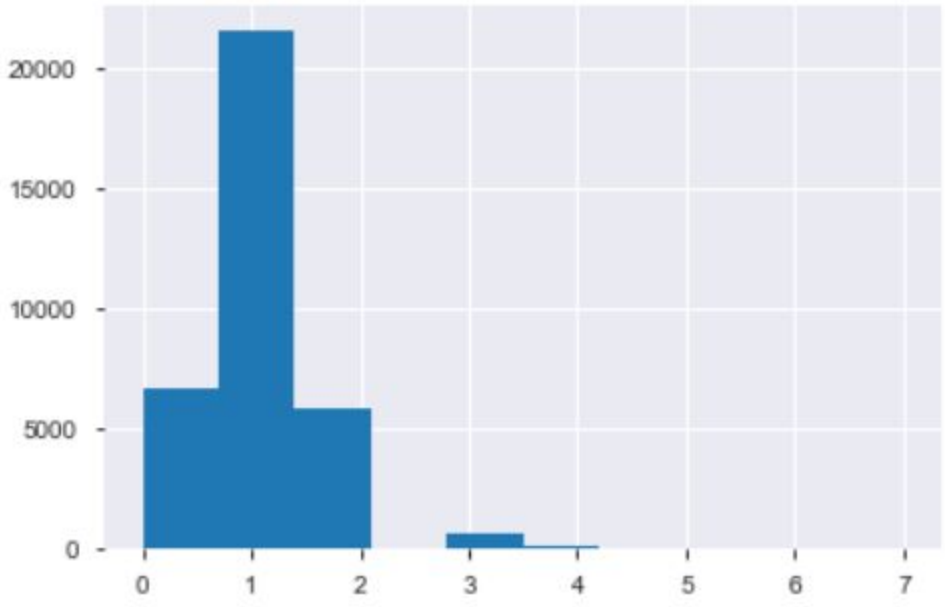
1. maximize insight into a data set
2. uncover underlying structure
3. extract important variables
4. detect outliers and anomalies
5. test underlying assumptions
6. develop parsimonious models
7. determine optimal factor setting

Observations in EDA

Number of movies per top Genre



Number of Genre per movie



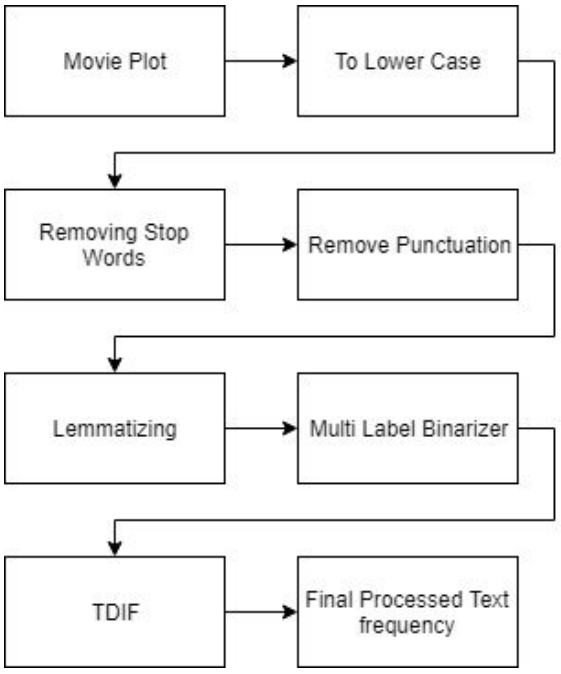
Natural Language Processing (NLP)

Natural language processing (NLP) is a subfield of computer science, information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.

Library Used : Natural Language Toolkit

Steps Followed

1. Convert movie plot into lower case
2. Remove stop words
3. Stemming
4. Lemmatization



Convert movie plot into lower case

The entire plot was made into lowercase as machine thinks there is difference between words which are not of the same case.

Remove stop words

Stop Words are words which do not contain important significance to be used in Search Queries. Usually, these words are filtered out from search queries because they return a vast amount of unnecessary information. Each programming language will give its own list of stop words to use. Mostly they are words that are commonly used in the English language such as 'as, the, be, are' etc.

Stemming

Stemming is the process of reducing inflection in words to their root forms such as mapping a group of words to the same stem even if the stem itself is not a valid word in the Language.

Lemmatization

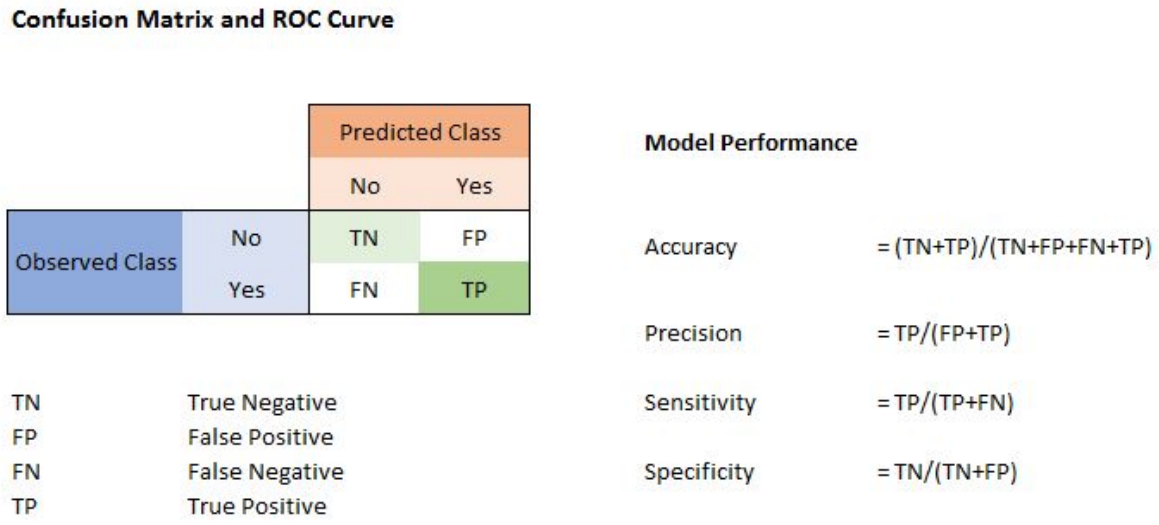
Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language. In Lemmatization root word is called Lemma. A lemma

(plural lemmas or lemmata) is the canonical form, dictionary form, or citation form of a set of words.

Metrics

Confusion Matrix

The Confusion matrix is one of the most intuitive and easiest (unless of course, you are not confused)metrics used for finding the correctness and accuracy of the model. It is used for Classification problem where the output can be of two or more types of classes.



True Positive(TP)

True positives are the cases when the actual class of the data point was 1(True) and the predicted is also 1(True)

True Negatives (TN)

True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False)

False Positives (FP)
False positives are the cases when the actual class of the data point was 0(False) and the predicted is 1(True). False is because the model has predicted incorrectly and positive because the class predicted was a positive one. (1)

False Negatives (FN)
False negatives are the cases when the actual class of the data point was 1(True) and the predicted is 0(False). False is because the model has predicted incorrectly and negative because the class predicted was a negative one. (0)

Accuracy

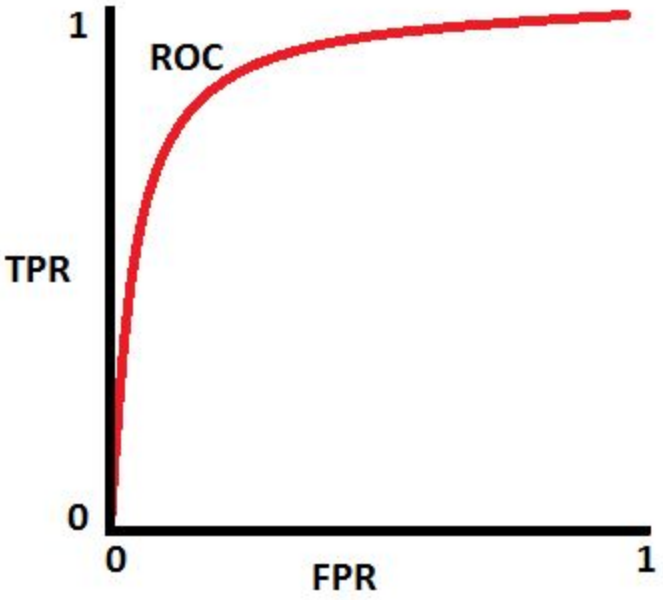
Accuracy is a good measure when the target variable classes in the data are nearly balanced.

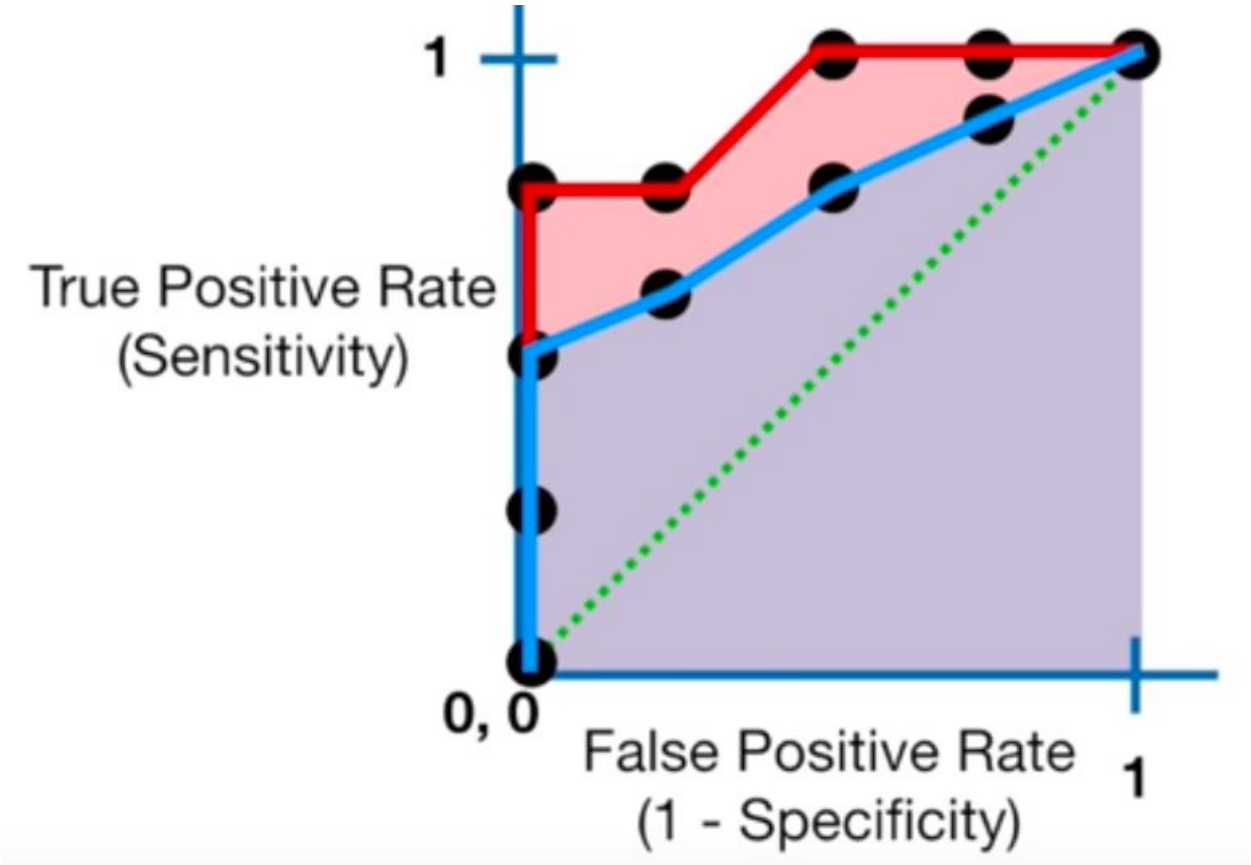
ROC and AUC

A ROC plots the relationship between the true positive rate and the false positive rate.

True positive rate = Recall = Sensitivity = true positive / (true positive + false negative)

False positive rate = 1 – specificity = false positive / (false positive + true negative)


1
ROC
TPR
0
0
FPR
1


1
True Positive Rate
(Sensitivity)
0, 0
False Positive Rate
(1 - Specificity)
1

Algorithms

Logistic Regression

Logistic regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

> In logistic regression, the dependent variable is binary or dichotomous, i.e. it only contains data coded as 1 (TRUE, success, pregnant, etc.) or 0 (FALSE, failure, non-pregnant, etc.).

The goal of logistic regression is to find the best fitting (yet biologically reasonable) model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables. Logistic regression generates the coefficients (and its standard errors and significance levels) of a formula to predict a logit transformation of the probability of presence of the characteristic of interest:

$$logit(p) = b_0 + b_1X_1 + b_2X_2 + b_3X_3 + \ldots + b_kX_k$$

where p is the probability of presence of the characteristic of interest. The logit transformation is defined as the logged odds:

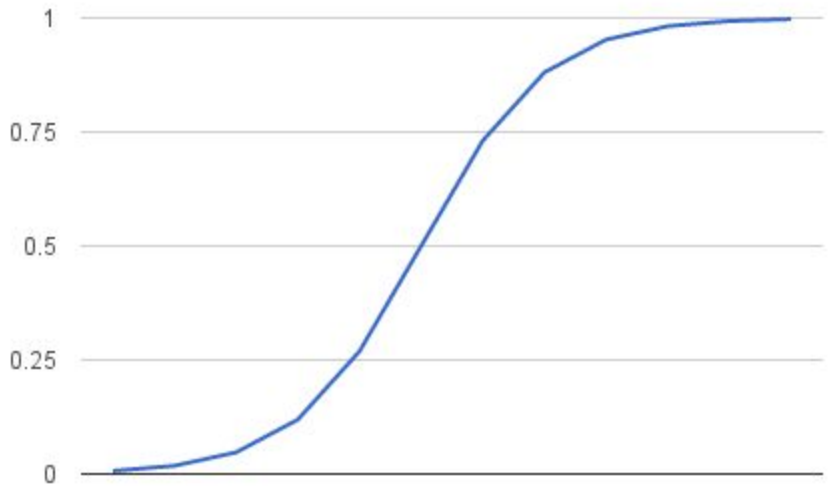
$$odds = \frac{p}{1-p} = \frac{probability\ of\ presence\ of\ characteristic}{probability\ of\ absence\ of\ characteristic}$$

and

$$logit(p) = \ln\left(\frac{p}{1-p}\right)$$

Rather than choosing parameters that minimize the sum of squared errors (like in ordinary regression), estimation in logistic regression chooses parameters that maximize the likelihood of observing the sample values.

Logistic function:

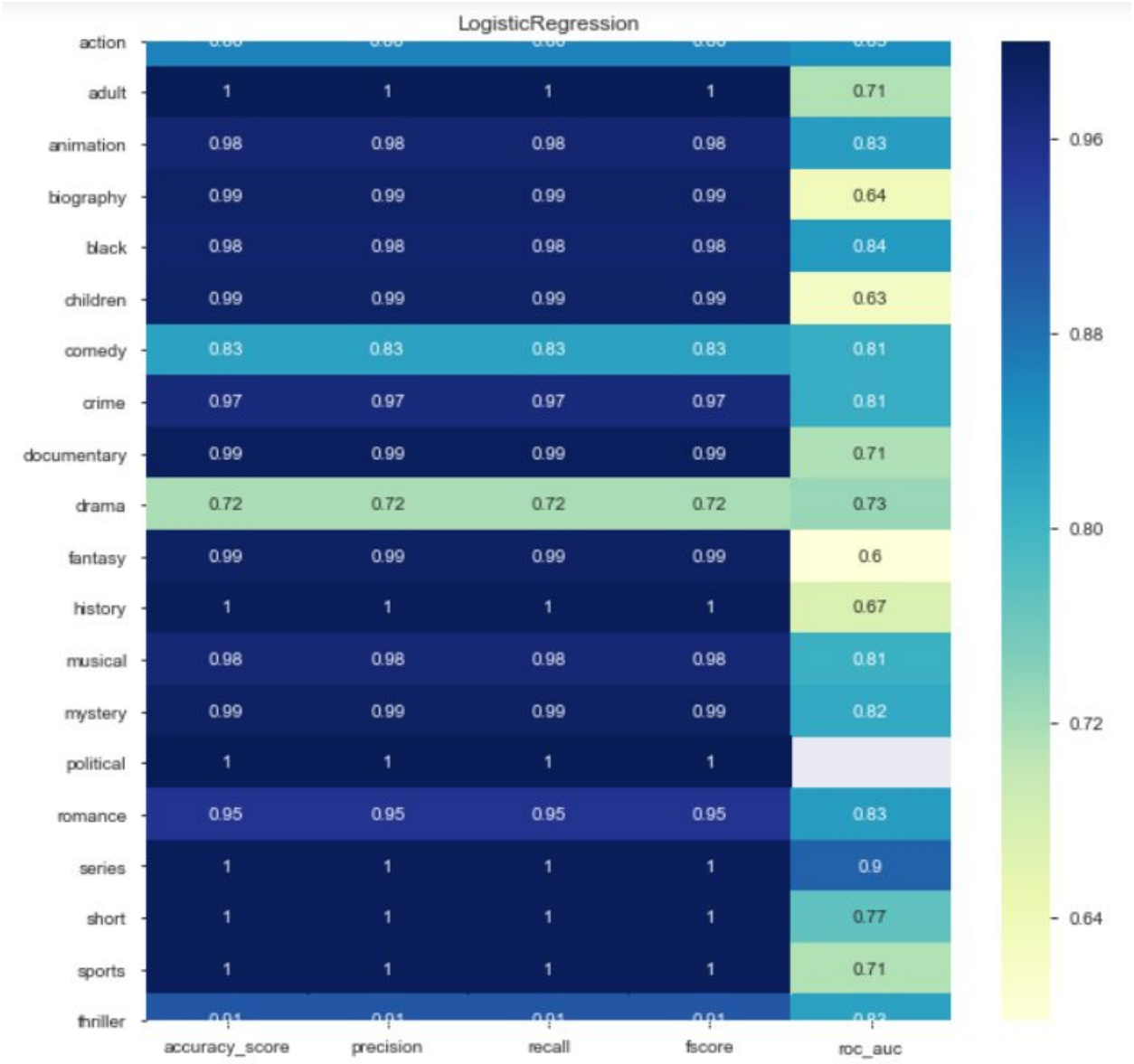


Results

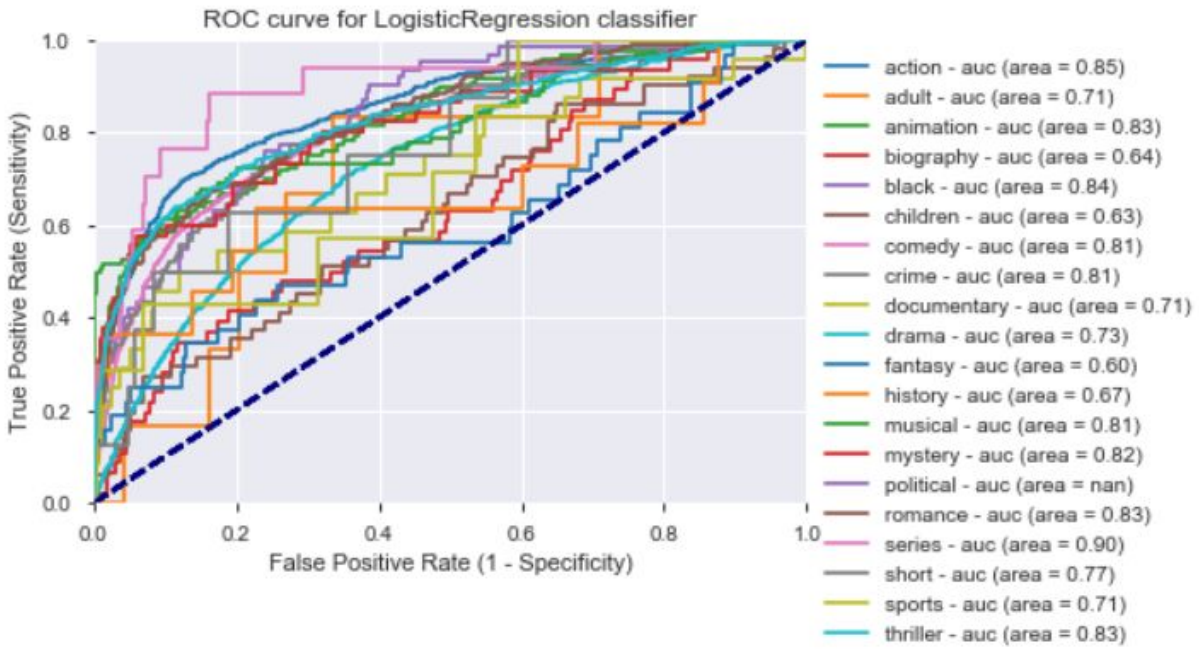
Top Parameters

param_estimator__C	param_estimator__dual	param_estimator__n_jobs	param_estimator__penalty
0.5	False	-1	l2
0.5	True	-1	l2
1	False	-1	l2
1	True	-1	l2
5	True	-1	l2

Accuracy Scores


LogisticRegression
action
adult 1 1 1 1 0.71
animation 0.98 0.98 0.98 0.98 0.83
biography 0.99 0.99 0.99 0.99 0.64
black 0.98 0.98 0.98 0.98 0.84
children 0.99 0.99 0.99 0.99 0.63
comedy 0.83 0.83 0.83 0.83 0.81
crime 0.97 0.97 0.97 0.97 0.81
documentary 0.99 0.99 0.99 0.99 0.71
drama 0.72 0.72 0.72 0.72 0.73
fantasy 0.99 0.99 0.99 0.99 0.6
history 1 1 1 1 0.67
musical 0.98 0.98 0.98 0.98 0.81
mystery 0.99 0.99 0.99 0.99 0.82
political 1 1 1 1
romance 0.95 0.95 0.95 0.95 0.83
series 1 1 1 1 0.9
short 1 1 1 1 0.77
sports 1 1 1 1 0.71
thriller
accuracy_score
precision
recall
fscore
roc_auc
0.96
0.88
0.80
0.72
0.64

ROC AUC Plot



Multinomial NB

The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work.

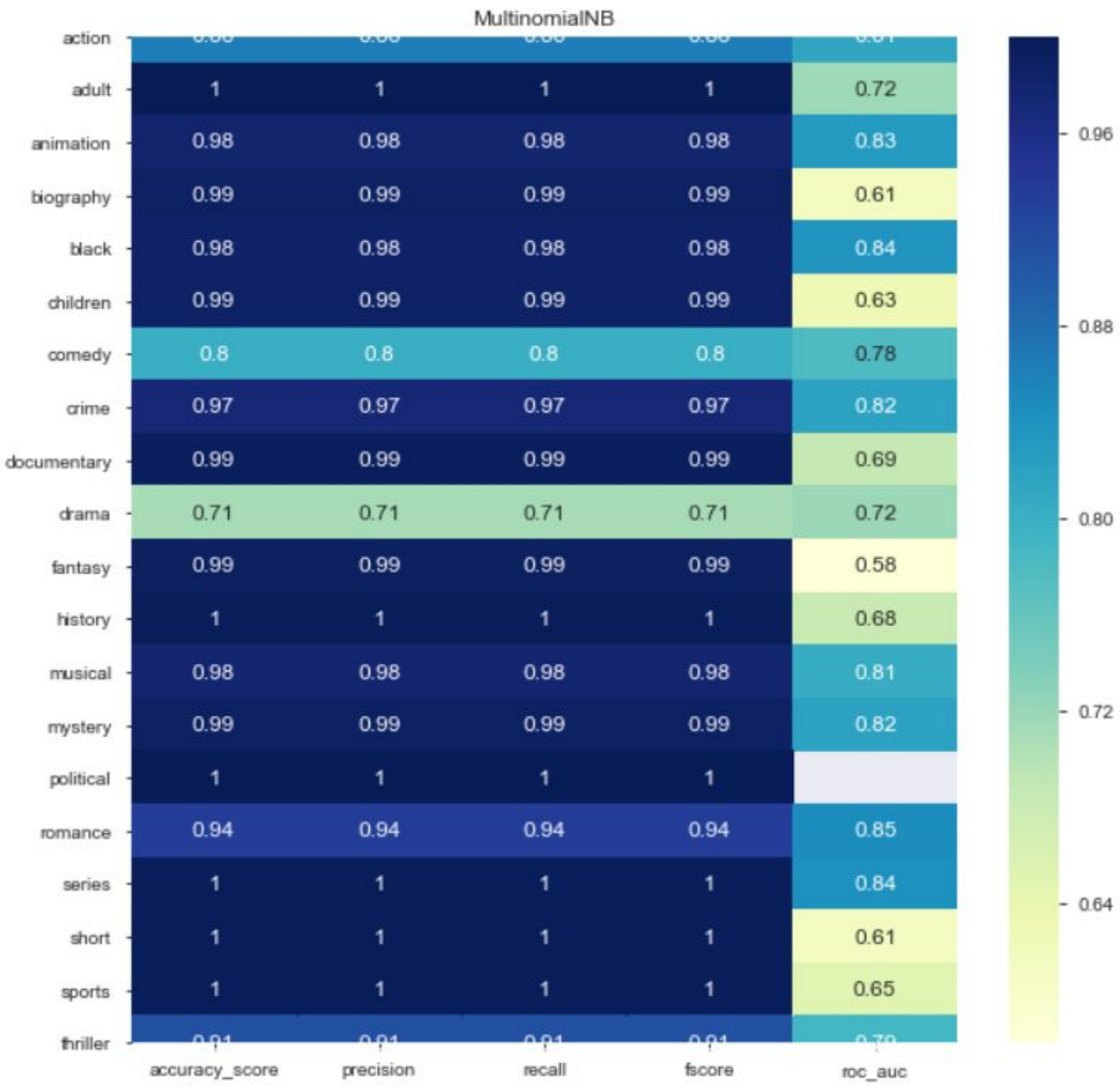
- Suited for classification of data with discrete features (count data)
- Very useful in text processing
- Each text will be converted to a vector of word count
- Cannot deal with negative numbers

Results

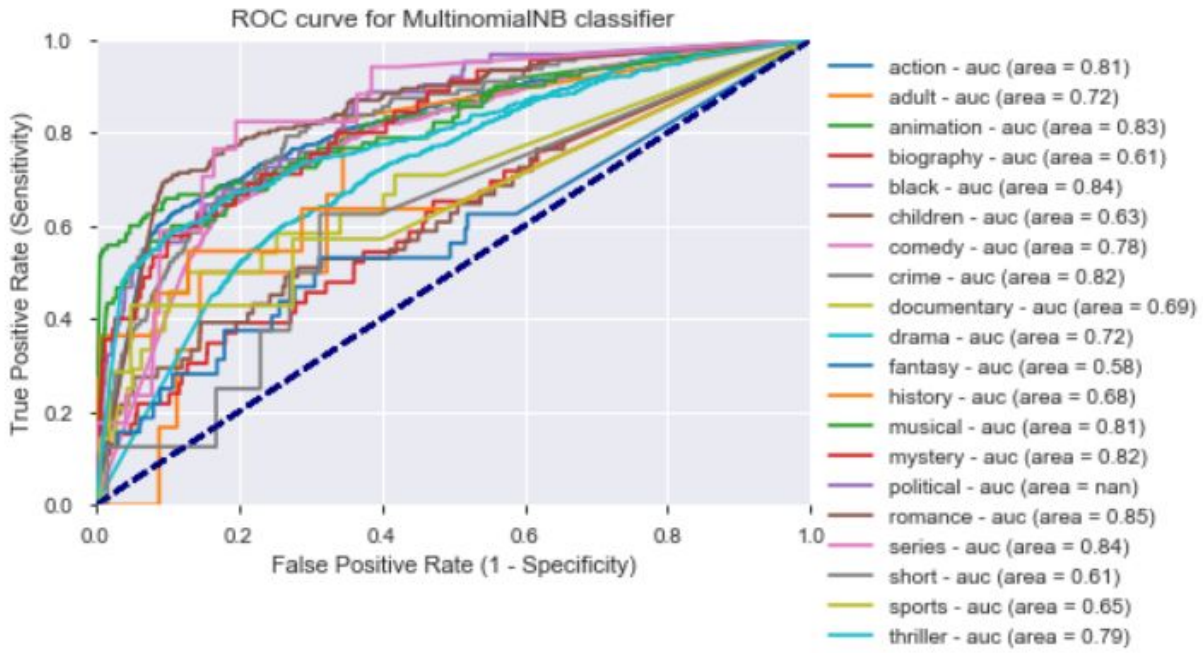
Top Parameters

param_estimator__alpha	param_estimator__fit_prior	param_n_jobs
10	True	-1
10	False	-1
0.01	True	-1
0.01	False	-1
0.001	True	-1

Accuracy Scores


MultinomialNB
action
adult
animation
biography
black
children
comedy
crime
documentary
drama
fantasy
history
musical
mystery
political
romance
series
short
sports
thriller
1 1 1 1 0.72
0.98 0.98 0.98 0.98 0.83
0.99 0.99 0.99 0.99 0.61
0.98 0.98 0.98 0.98 0.84
0.99 0.99 0.99 0.99 0.63
0.8 0.8 0.8 0.8 0.78
0.97 0.97 0.97 0.97 0.82
0.99 0.99 0.99 0.99 0.69
0.71 0.71 0.71 0.71 0.72
0.99 0.99 0.99 0.99 0.58
1 1 1 1 0.68
0.98 0.98 0.98 0.98 0.81
0.99 0.99 0.99 0.99 0.82
1 1 1 1
0.94 0.94 0.94 0.94 0.85
1 1 1 1 0.84
1 1 1 1 0.61
1 1 1 1 0.65
accuracy_score
precision
recall
fscore
roc_auc
0.96
0.88
0.80
0.72
0.64

ROC AUC Plot



Linear SVC

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection.

The advantages of support vector machines are:

Effective in high dimensional spaces.

Still effective in cases where the number of dimensions is greater than the number of samples.

Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

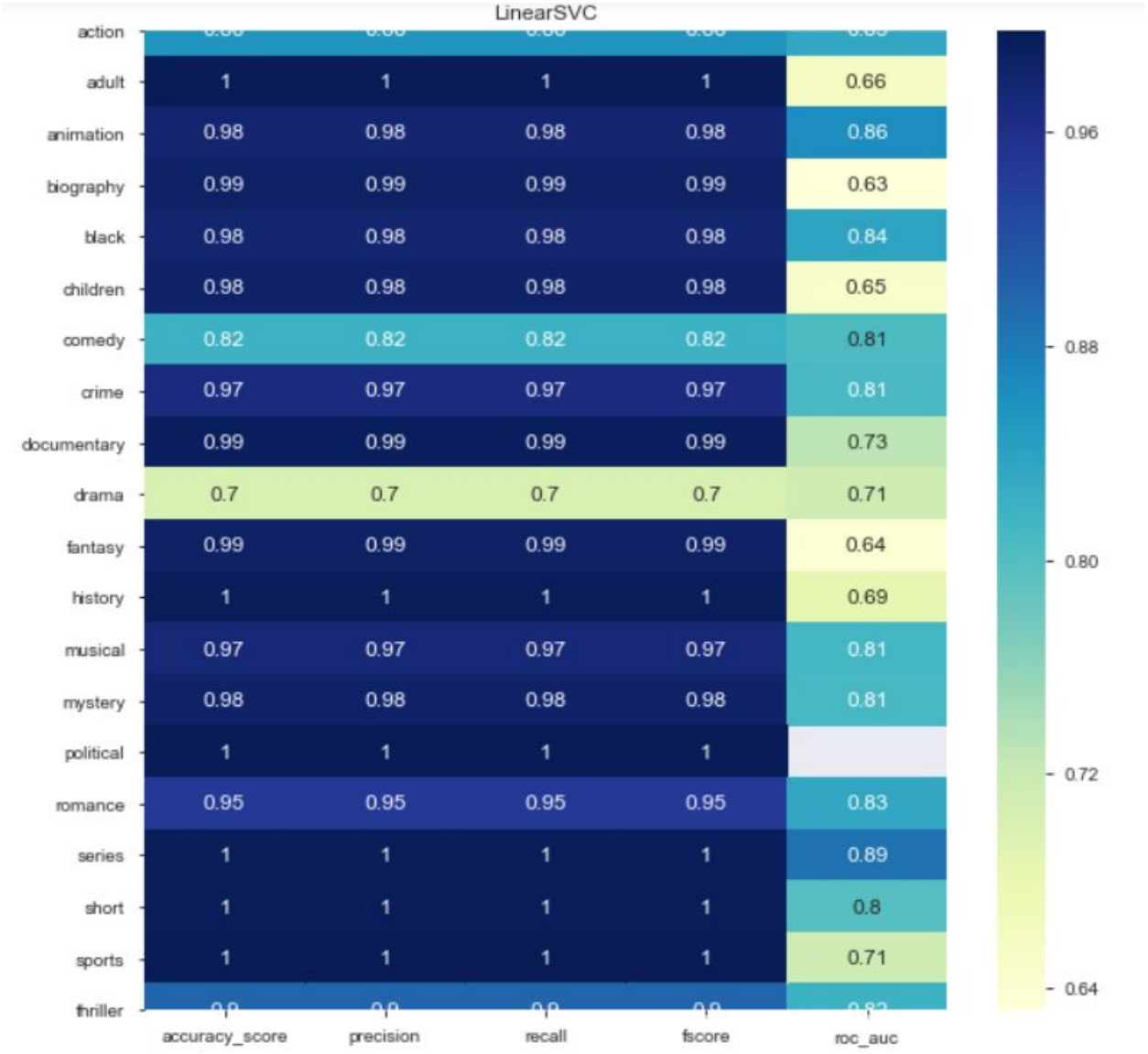
Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

Results

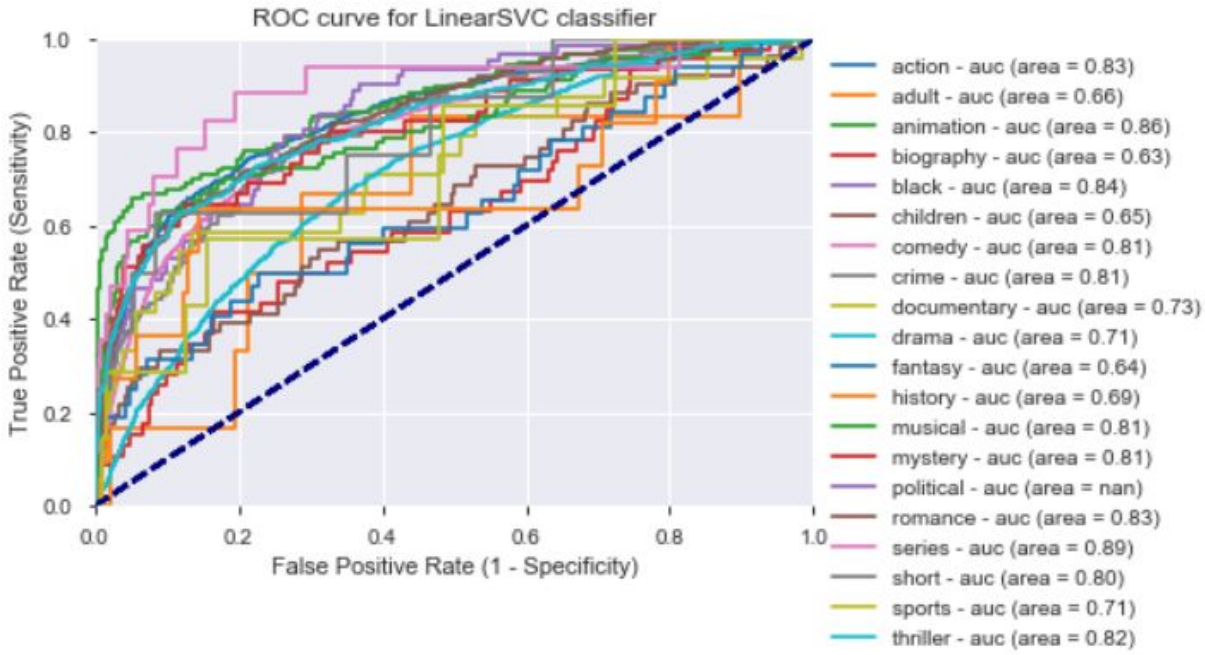
Top Parameters

param_estimator__C	param_estimator__dual	param_estimator__loss	param_estimator__penalty	param_estimator__tol	param_n_jobs	
5	False	squared_hinge	l2	0.01	-1	,
0.5	False	squared_hinge	l2	0.01	-1	,
1	False	squared_hinge	l2	0.01	-1	,
0.5	True	squared_hinge	l2	0.01	-1	,
1	True	squared_hinge	l2	0.01	-1	,

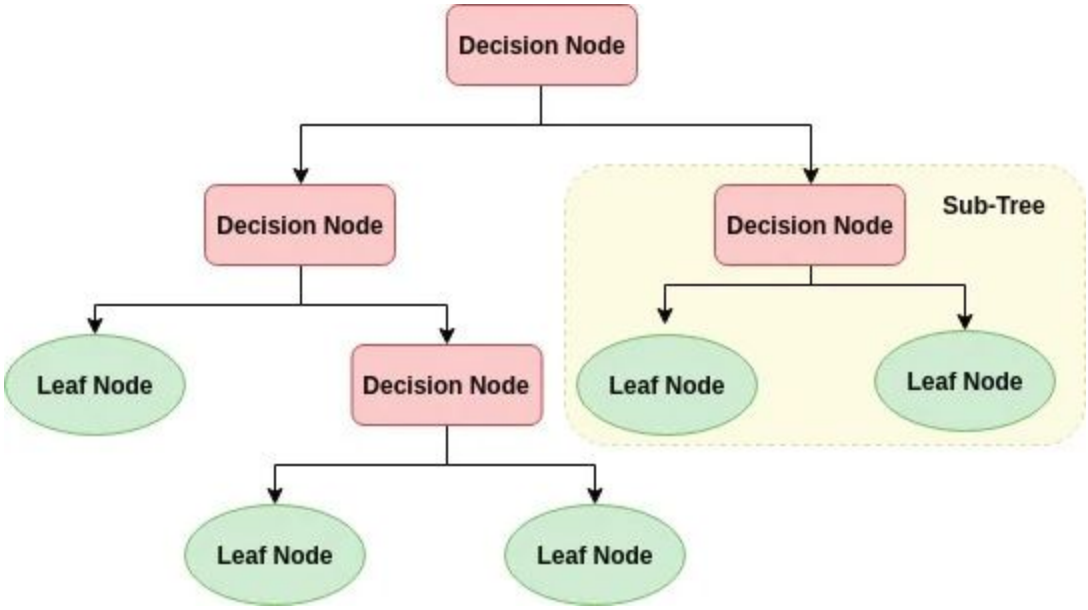
Accuracy Scores


LinearSVC
action
adult 1 1 1 1 0.66
animation 0.98 0.98 0.98 0.98 0.86
biography 0.99 0.99 0.99 0.99 0.63
black 0.98 0.98 0.98 0.98 0.84
children 0.98 0.98 0.98 0.98 0.65
comedy 0.82 0.82 0.82 0.82 0.81
crime 0.97 0.97 0.97 0.97 0.81
documentary 0.99 0.99 0.99 0.99 0.73
drama 0.7 0.7 0.7 0.7 0.71
fantasy 0.99 0.99 0.99 0.99 0.64
history 1 1 1 1 0.69
musical 0.97 0.97 0.97 0.97 0.81
mystery 0.98 0.98 0.98 0.98 0.81
political 1 1 1 1
romance 0.95 0.95 0.95 0.95 0.83
series 1 1 1 1 0.89
short 1 1 1 1 0.8
sports 1 1 1 1 0.71
thriller
accuracy_score
precision
recall
fscore
roc_auc
0.96
0.88
0.80
0.72
0.64

ROC AUC Plot



Decision Tree Classifier



A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making. It's

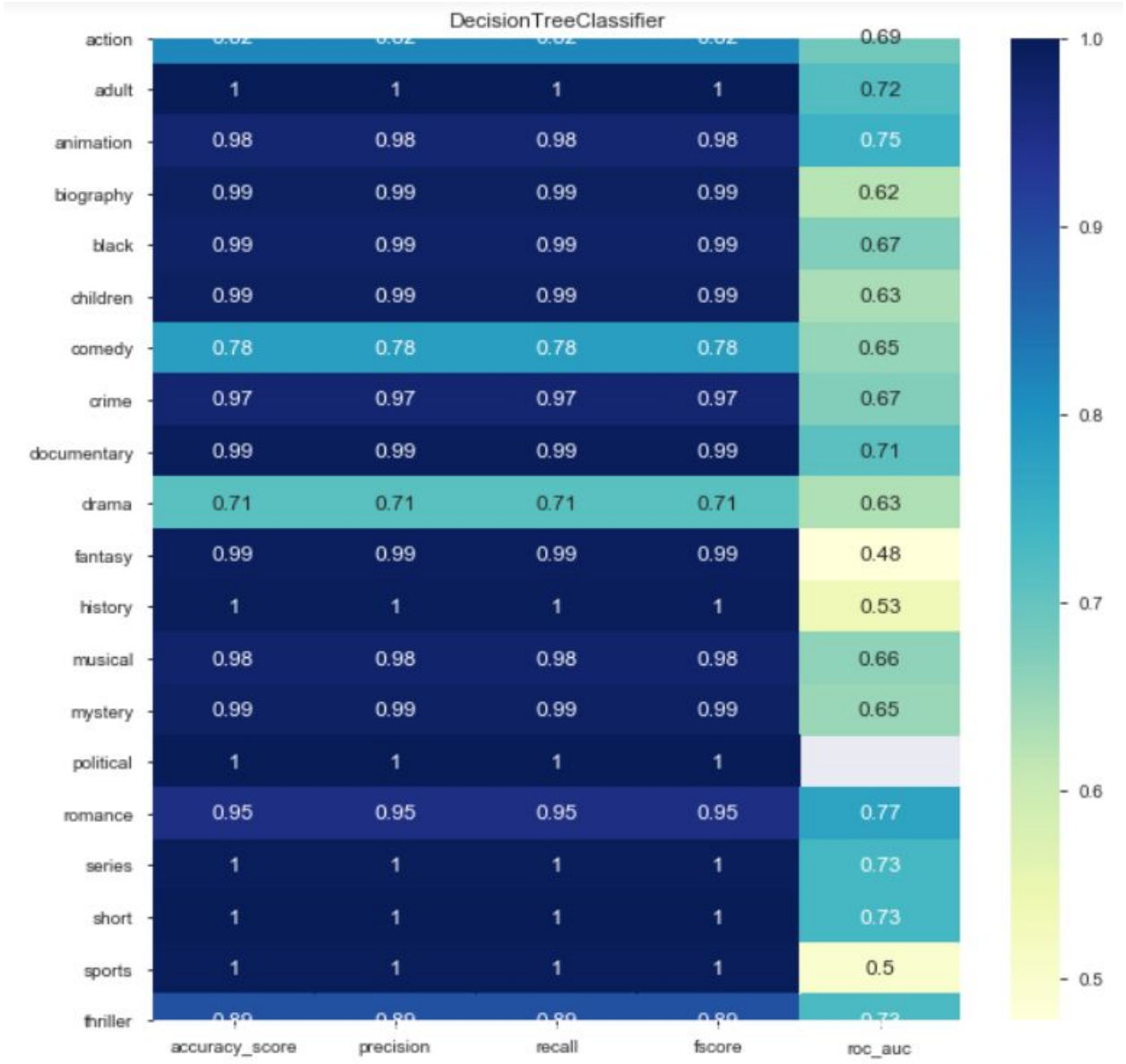
visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.

Results

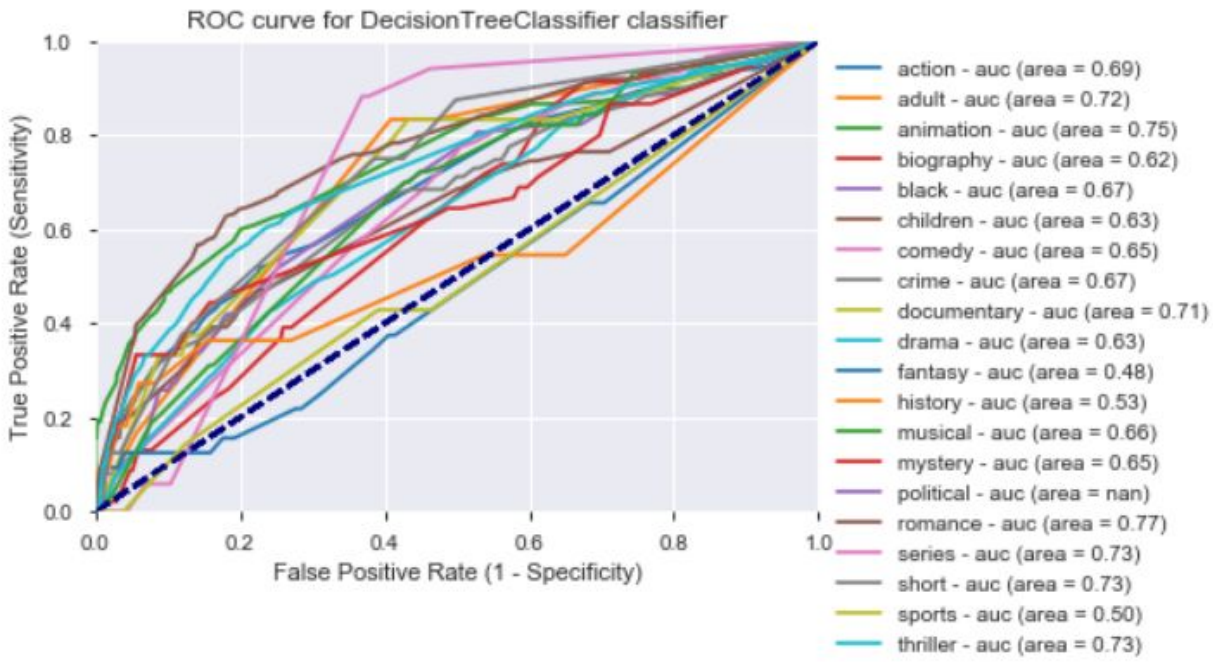
Top Parameters

param_criterion	param_max_depth
entropy	9
gini	9
entropy	8
gini	8

Accuracy Scores

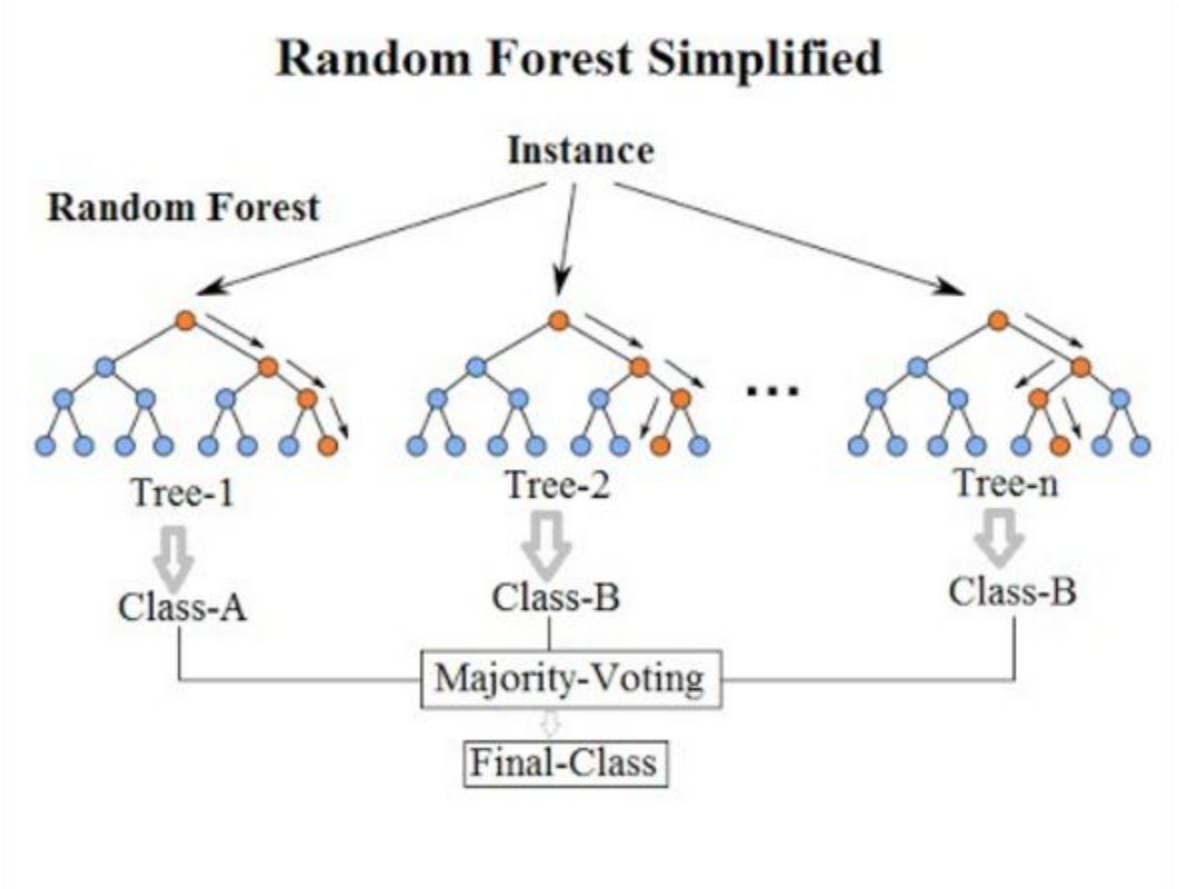

DecisionTreeClassifier
action
adult
animation
biography
black
children
comedy
crime
documentary
drama
fantasy
history
musical
mystery
political
romance
series
short
sports
thriller
accuracy_score
precision
recall
fscore
roc_auc
0.69
1 1 1 1 0.72
0.98 0.98 0.98 0.98 0.75
0.99 0.99 0.99 0.99 0.62
0.99 0.99 0.99 0.99 0.67
0.99 0.99 0.99 0.99 0.63
0.78 0.78 0.78 0.78 0.65
0.97 0.97 0.97 0.97 0.67
0.99 0.99 0.99 0.99 0.71
0.71 0.71 0.71 0.71 0.63
0.99 0.99 0.99 0.99 0.48
1 1 1 1 0.53
0.98 0.98 0.98 0.98 0.66
0.99 0.99 0.99 0.99 0.65
1 1 1 1
0.95 0.95 0.95 0.95 0.77
1 1 1 1 0.73
1 1 1 1 0.73
1 1 1 1 0.5
1.0
0.9
0.8
0.7
0.6
0.5

ROC AUC Plot



Random Forest Classifier

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

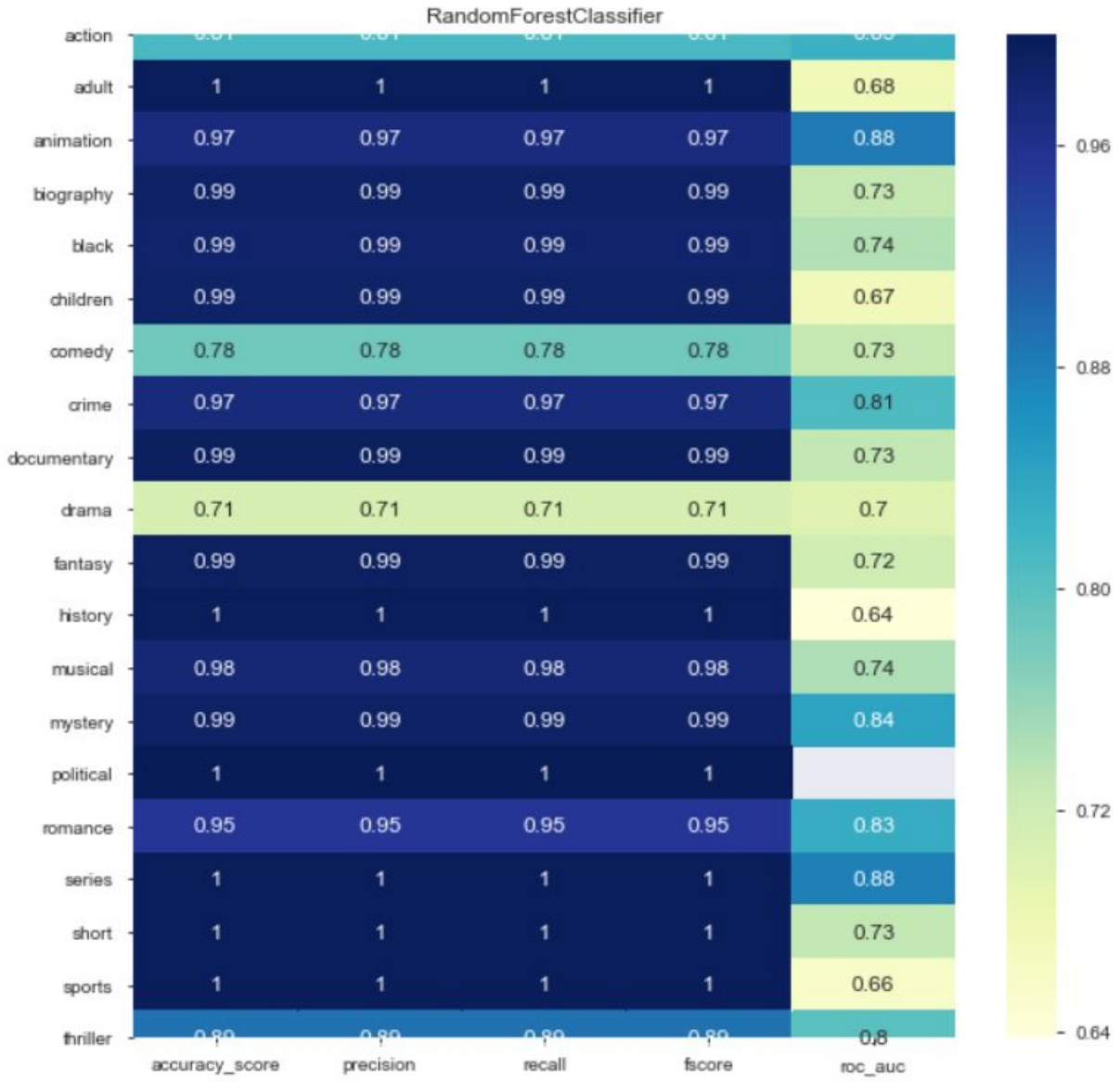


Results

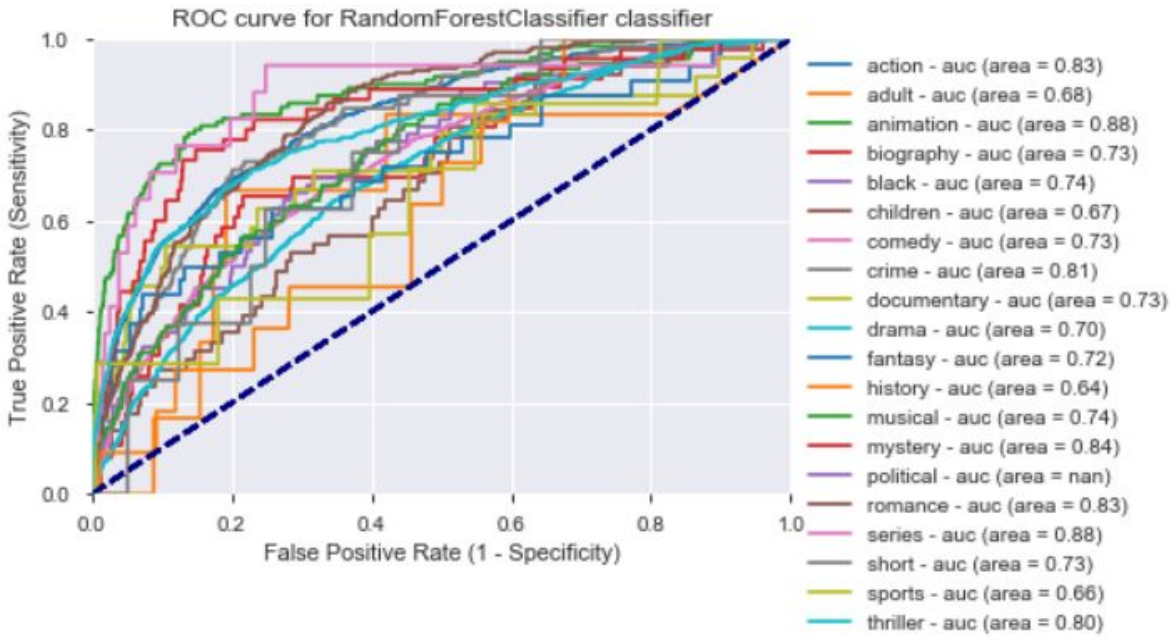
Top Parameters

param_criterion	param_min_samples_leaf	param_min_samples_split	param_n_jobs
gini	19	19	-1
gini	19	20	-1
entropy	19	19	-1
entropy	19	20	-1
gini	20	19	-1

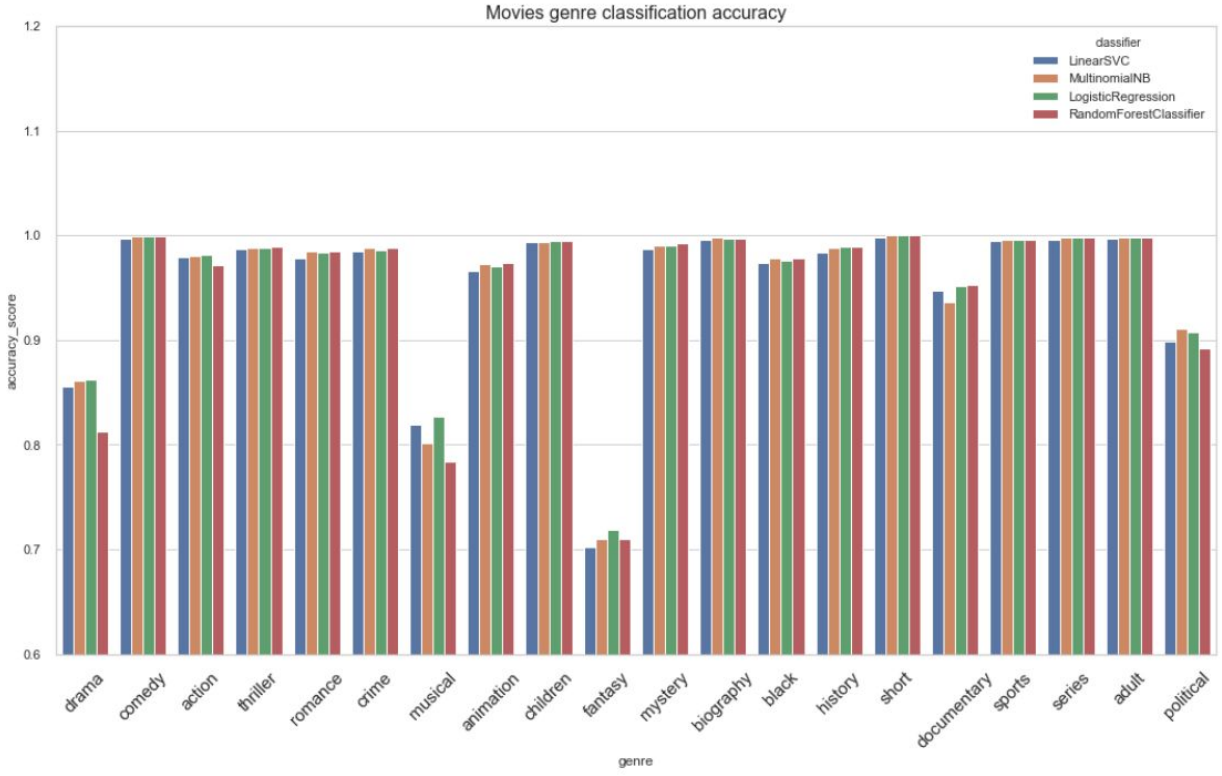
Accuracy Scores


RandomForestClassifier
action
adult 1 1 1 1 0.68
animation 0.97 0.97 0.97 0.97 0.88
biography 0.99 0.99 0.99 0.99 0.73
black 0.99 0.99 0.99 0.99 0.74
children 0.99 0.99 0.99 0.99 0.67
comedy 0.78 0.78 0.78 0.78 0.73
crime 0.97 0.97 0.97 0.97 0.81
documentary 0.99 0.99 0.99 0.99 0.73
drama 0.71 0.71 0.71 0.71 0.7
fantasy 0.99 0.99 0.99 0.99 0.72
history 1 1 1 1 0.64
musical 0.98 0.98 0.98 0.98 0.74
mystery 0.99 0.99 0.99 0.99 0.84
political 1 1 1 1
romance 0.95 0.95 0.95 0.95 0.83
series 1 1 1 1 0.88
short 1 1 1 1 0.73
sports 1 1 1 1 0.66
thriller
0,8
accuracy_score
precision
recall
fscore
roc_auc
0.96
0.88
0.80
0.72
0.64

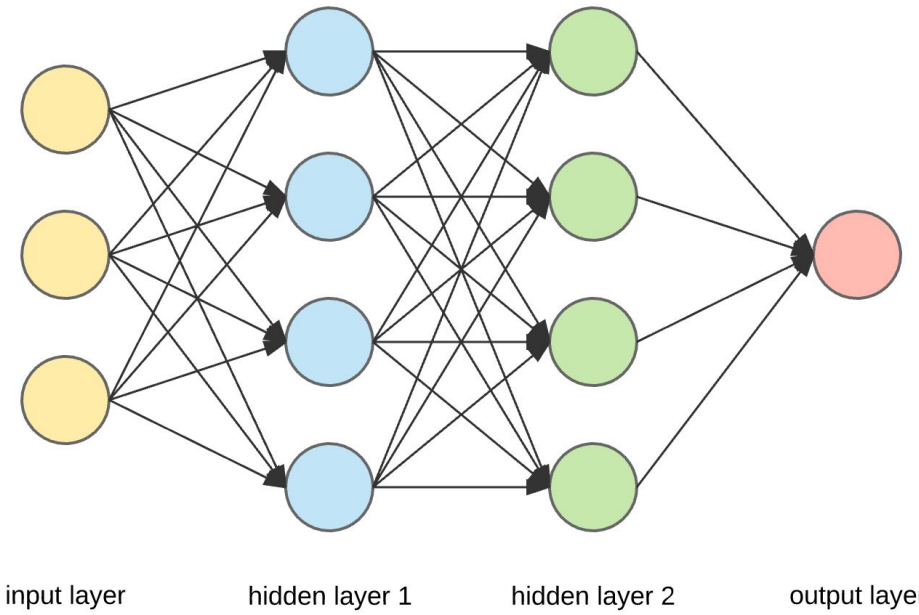
ROC AUC Plot



Comparison of Classic ML Methods



Deep Neural Network



Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on artificial neural networks. Learning can be supervised, semi-supervised or unsupervised.

Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System) and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network mode.

Word Embedding

A word embedding is a class of approaches for representing words and documents using a dense vector representation.

It is an improvement over more traditional bag-of-word model encoding schemes where large sparse vectors were used to represent each word or to score each word within a vector to represent an entire vocabulary. These representations were sparse because the vocabularies were vast and a given word or document would be represented by a large vector comprised mostly of zero values.

Instead, in an embedding, words are represented by dense vectors where a vector represents the projection of the word into a continuous vector space.

The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used.

The position of a word in the learned vector space is referred to as its embedding.

Embedding layer

The Embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset. It is a flexible layer that can be used in a variety of ways, such as: The Embedding layer is defined as the first hidden layer of a network.

Keras offers an Embedding layer that can be used for neural networks on text data.

It requires that the input data be integer encoded, so that each word is represented by a unique integer. This data preparation step can be performed using the Tokenizer API also provided with Keras.

The Embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset.

It is a flexible layer that can be used in a variety of ways, such as:

- It can be used alone to learn a word embedding that can be saved and used in another model later.
- It can be used as part of a deep learning model where the embedding is learned along with the model itself.

- It can be used to load a pre-trained word embedding model, a type of transfer learning.

The Embedding layer is defined as the first hidden layer of a network. It must specify 3 arguments:

It must specify 3 arguments:

- input_dim: This is the size of the vocabulary in the text data. For example, if your data is integer encoded to values between 0-10, then the size of the vocabulary would be 11 words.
- output_dim: This is the size of the vector space in which words will be embedded. It defines the size of the output vectors from this layer for each word. For example, it could be 32 or 100 or even larger. Test different values for your problem.
- input_length: This is the length of input sequences, as you would define for any input layer of a Keras model. For example, if all of your input documents are comprised of 1000 words, this would be 1000.

For example, below we define an Embedding layer with a vocabulary of 200 (e.g. integer encoded words from 0 to 199, inclusive), a vector space of 32 dimensions in which words will be embedded, and input documents that have 50 words each.

```
e = Embedding(200, 32,
input_length=50)
```

The Embedding layer has weights that are learned. If you save your model to file, this will include weights for the Embedding layer.

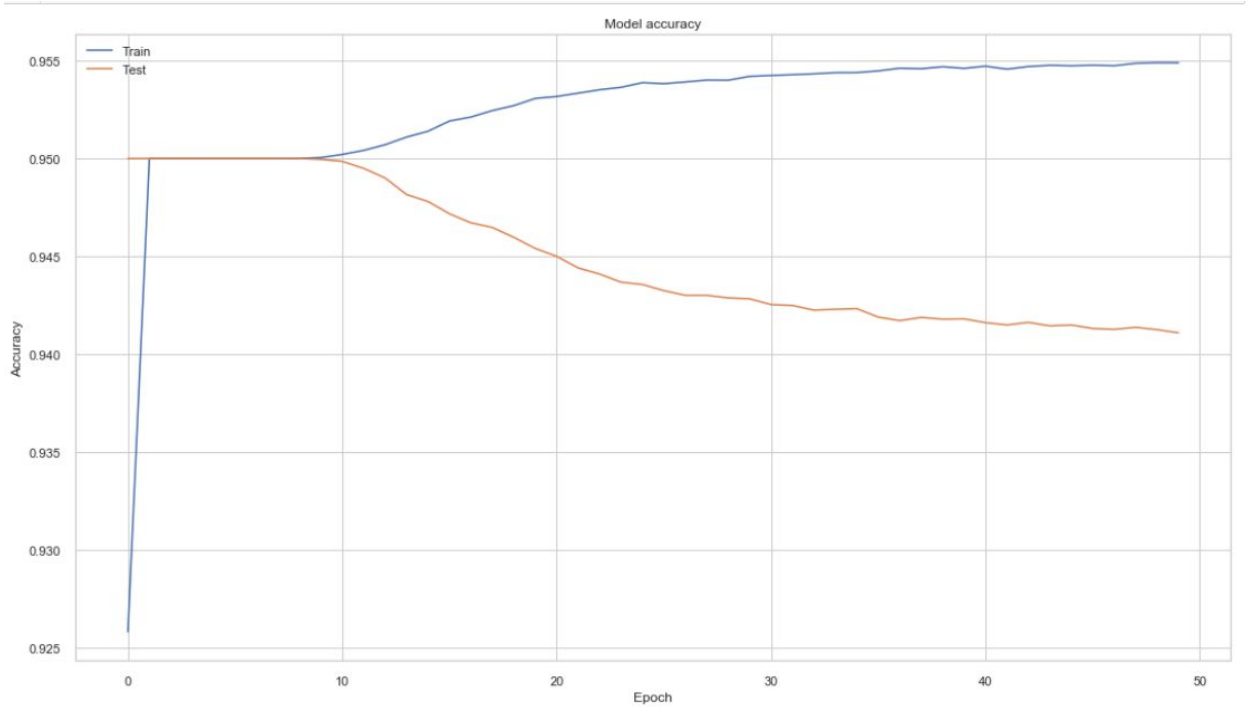
The output of the Embedding layer is a 2D vector with one embedding for each word in the input sequence of words (input document).

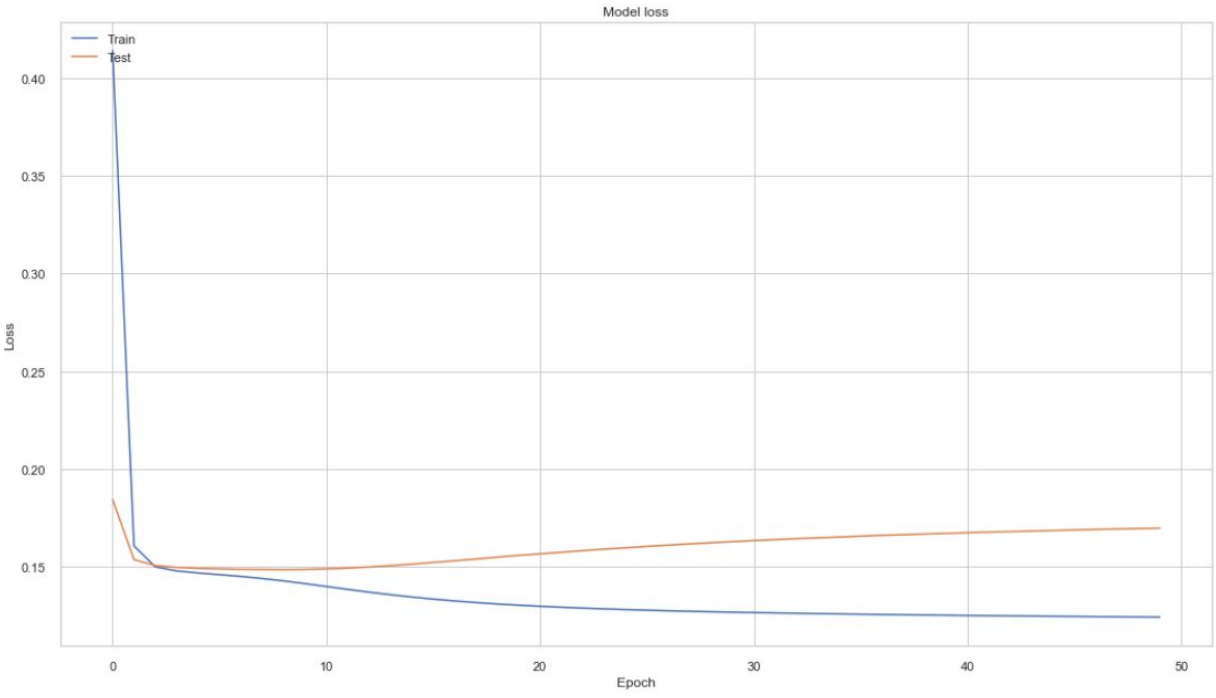
If you wish to connect a Dense layer directly to an Embedding layer, you must first flatten the 2D output matrix to a 1D vector using the Flatten layer.

One Hot Method

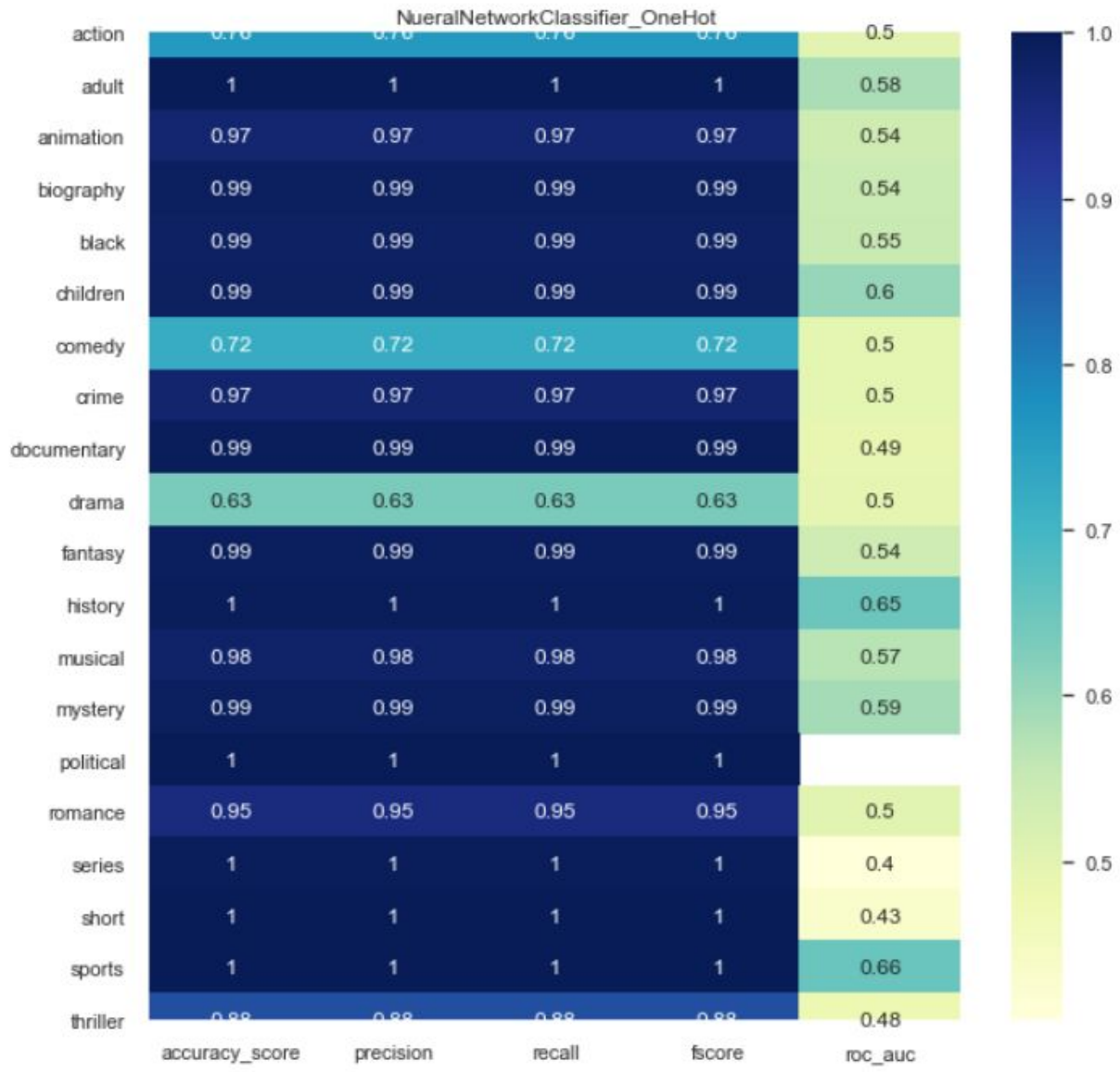
Results

Training & validation - accuracy and loss values

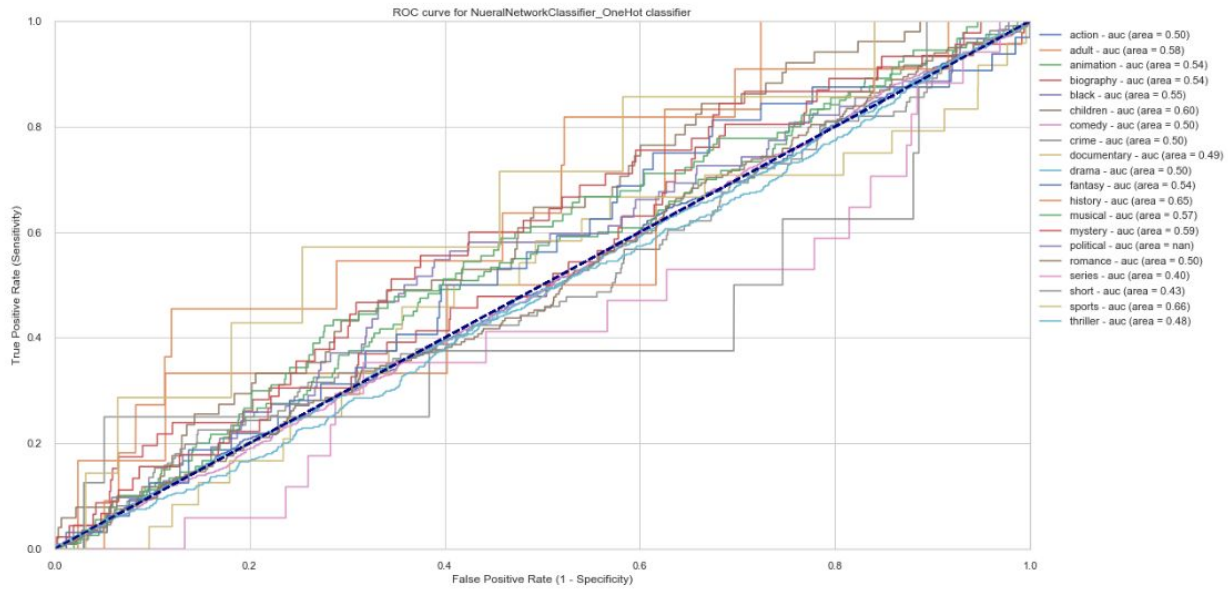




Accuracy Scores


NueralNetworkClassifier_OneHot
action
adult
animation
biography
black
children
comedy
crime
documentary
drama
fantasy
history
musical
mystery
political
romance
series
short
sports
thriller
1 1 1 1 0.58
0.97 0.97 0.97 0.97 0.54
0.99 0.99 0.99 0.99 0.54
0.99 0.99 0.99 0.99 0.55
0.99 0.99 0.99 0.99 0.6
0.72 0.72 0.72 0.72 0.5
0.97 0.97 0.97 0.97 0.5
0.99 0.99 0.99 0.99 0.49
0.63 0.63 0.63 0.63 0.5
0.99 0.99 0.99 0.99 0.54
1 1 1 1 0.65
0.98 0.98 0.98 0.98 0.57
0.99 0.99 0.99 0.99 0.59
1 1 1 1
0.95 0.95 0.95 0.95 0.5
1 1 1 1 0.4
1 1 1 1 0.43
1 1 1 1 0.66
0.48
accuracy_score
precision
recall
fscore
roc_auc
1.0
0.9
0.8
0.7
0.6
0.5

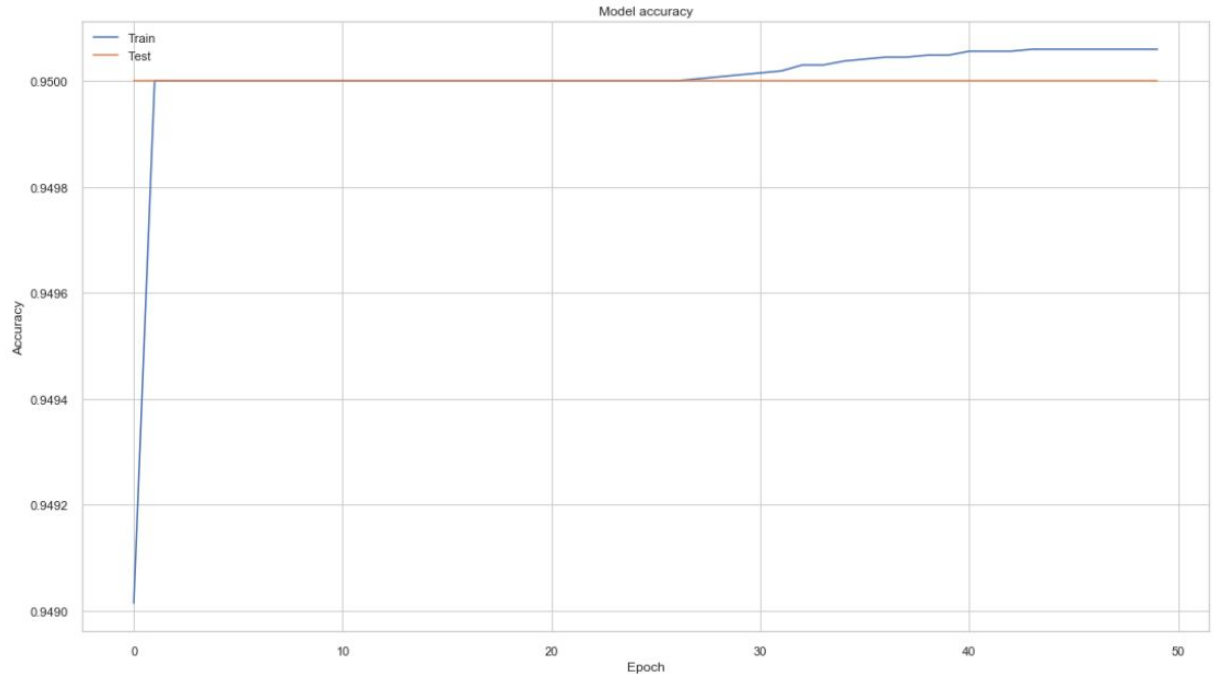
ROC AUC Plot

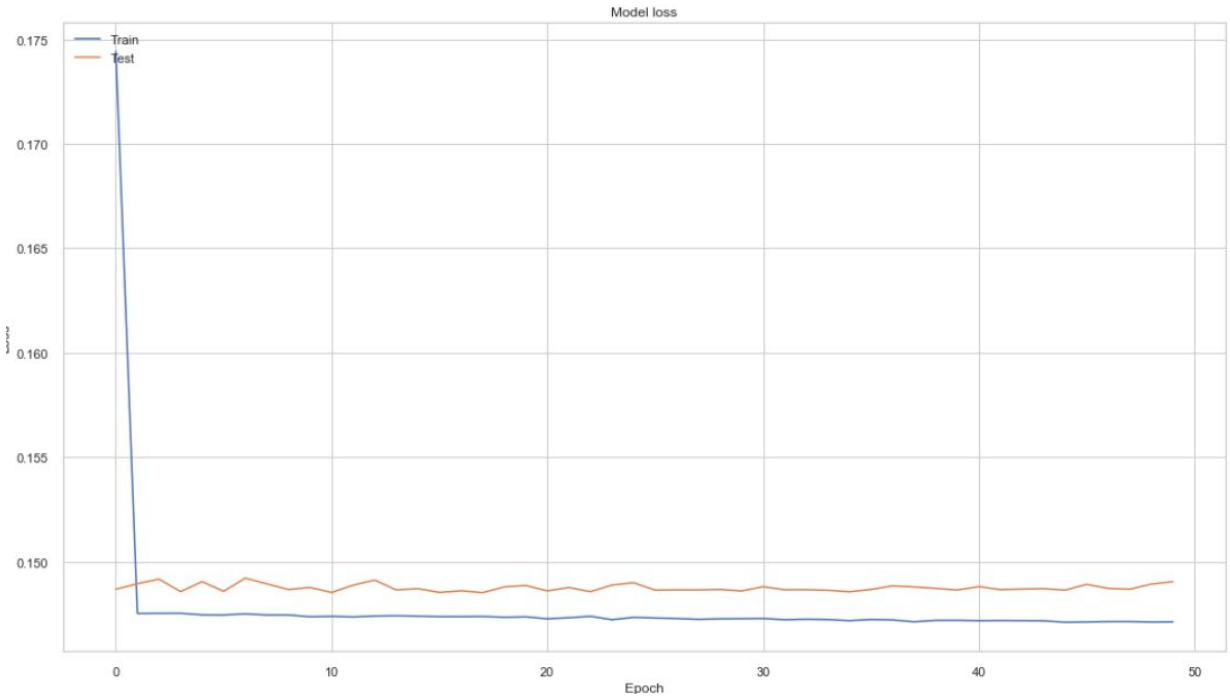


TDIF Method

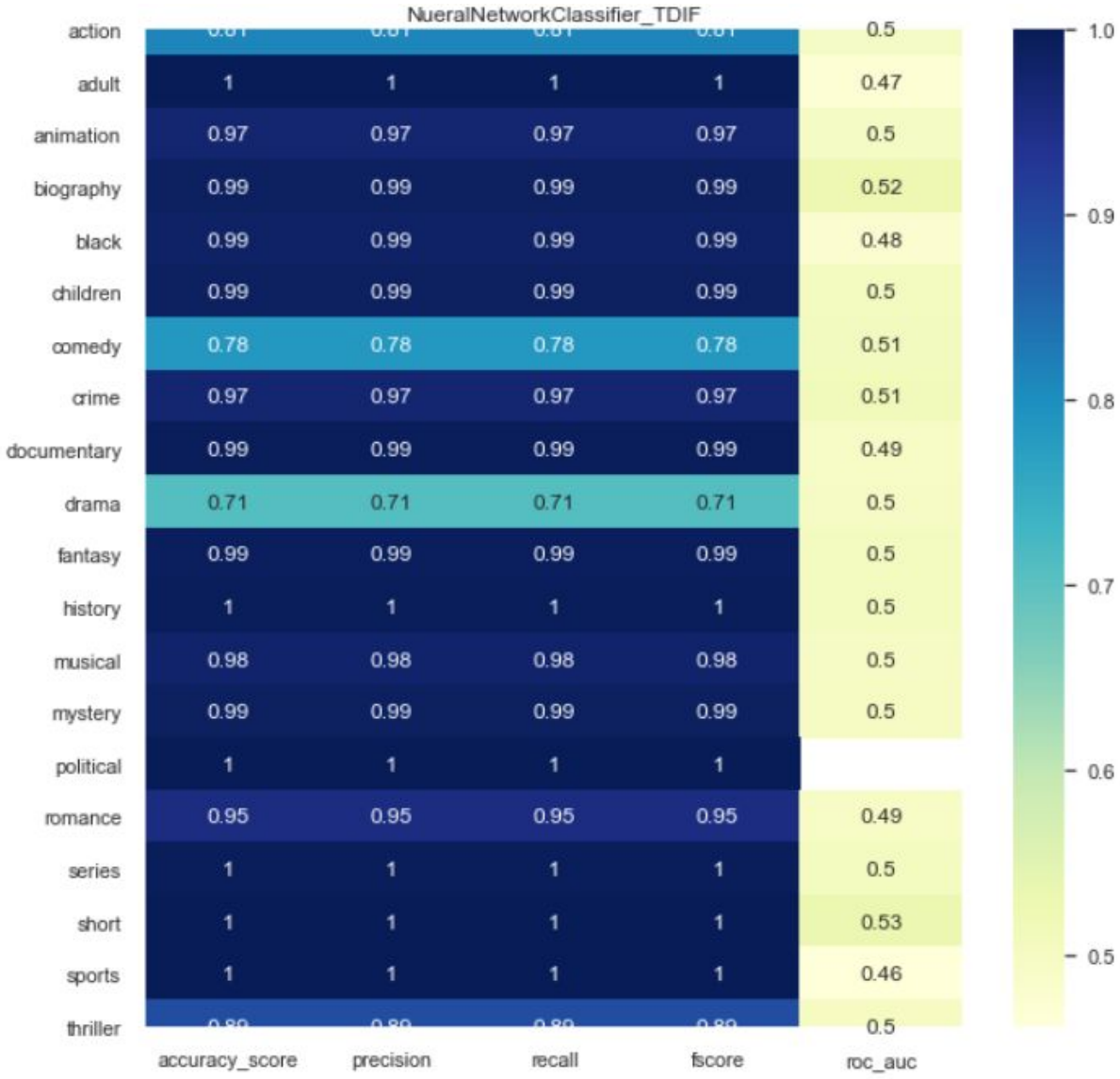
Results

Training & validation - accuracy and loss values

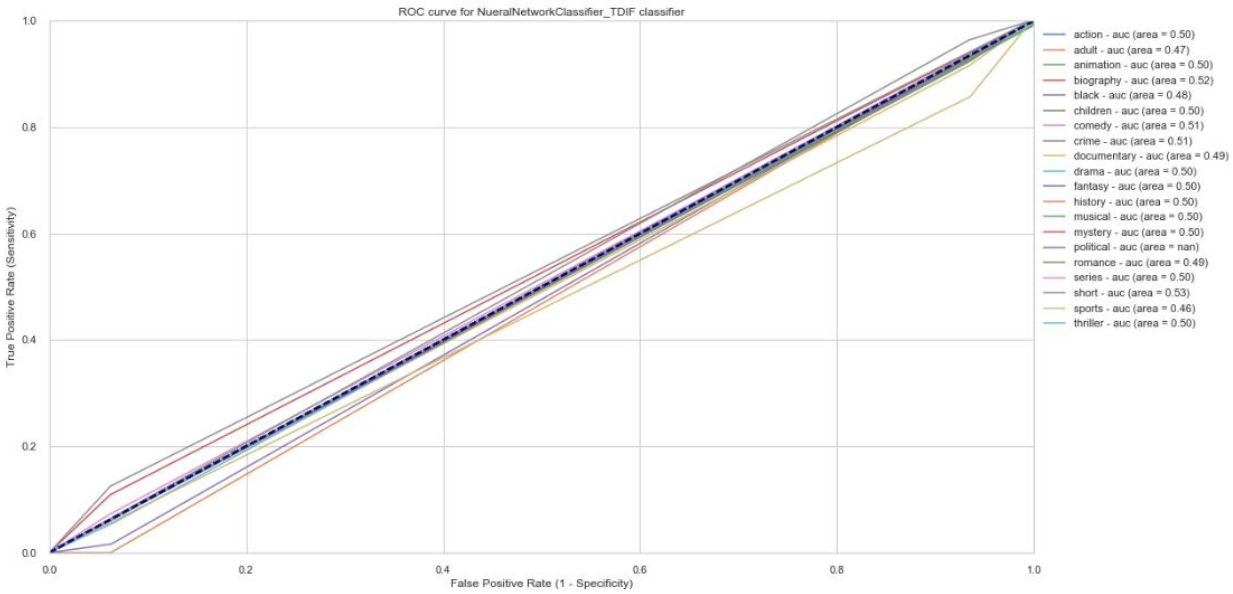




Accuracy Scores


NueralNetworkClassifier_TDIF
action
adult
animation
biography
black
children
comedy
crime
documentary
drama
fantasy
history
musical
mystery
political
romance
series
short
sports
thriller
accuracy_score
precision
recall
fscore
roc_auc
1
0.97
0.99
0.78
0.71
0.98
0.95
0.5
0.47
0.52
0.48
0.51
0.49
0.53
0.46
1.0
0.9
0.8
0.7
0.6
0.5

ROC AUC Plot



Deep Neural Network - Transfer learning

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

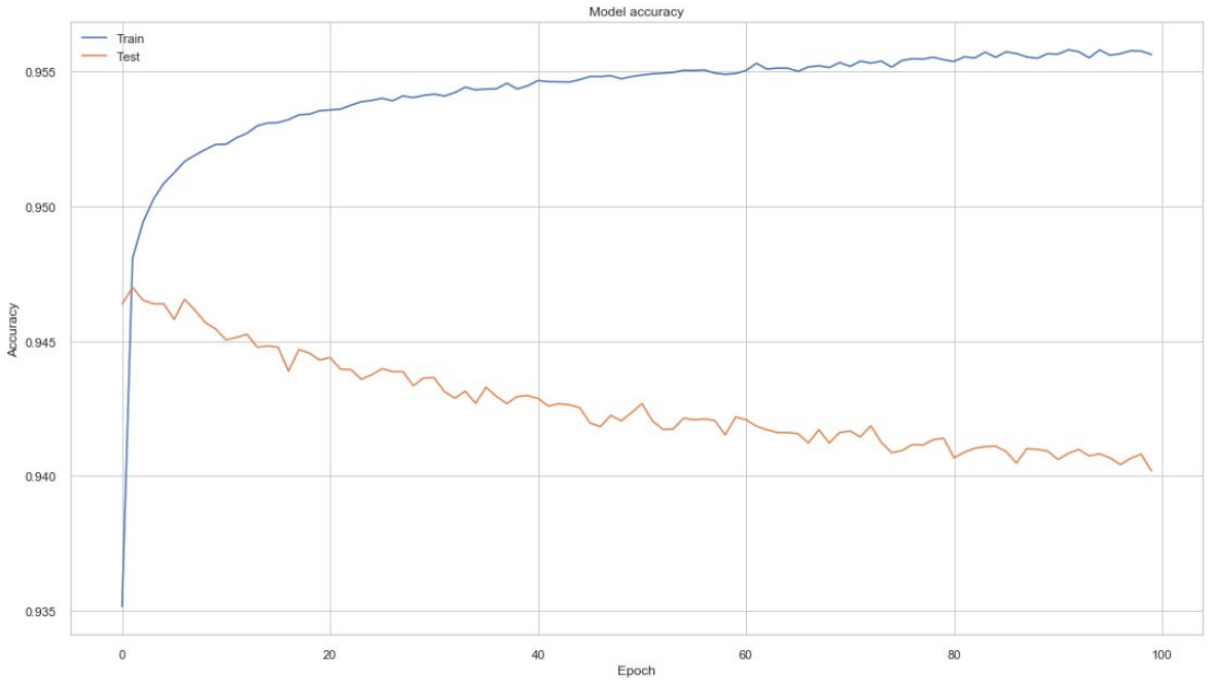
Two popular examples of methods of learning word embeddings from text include:

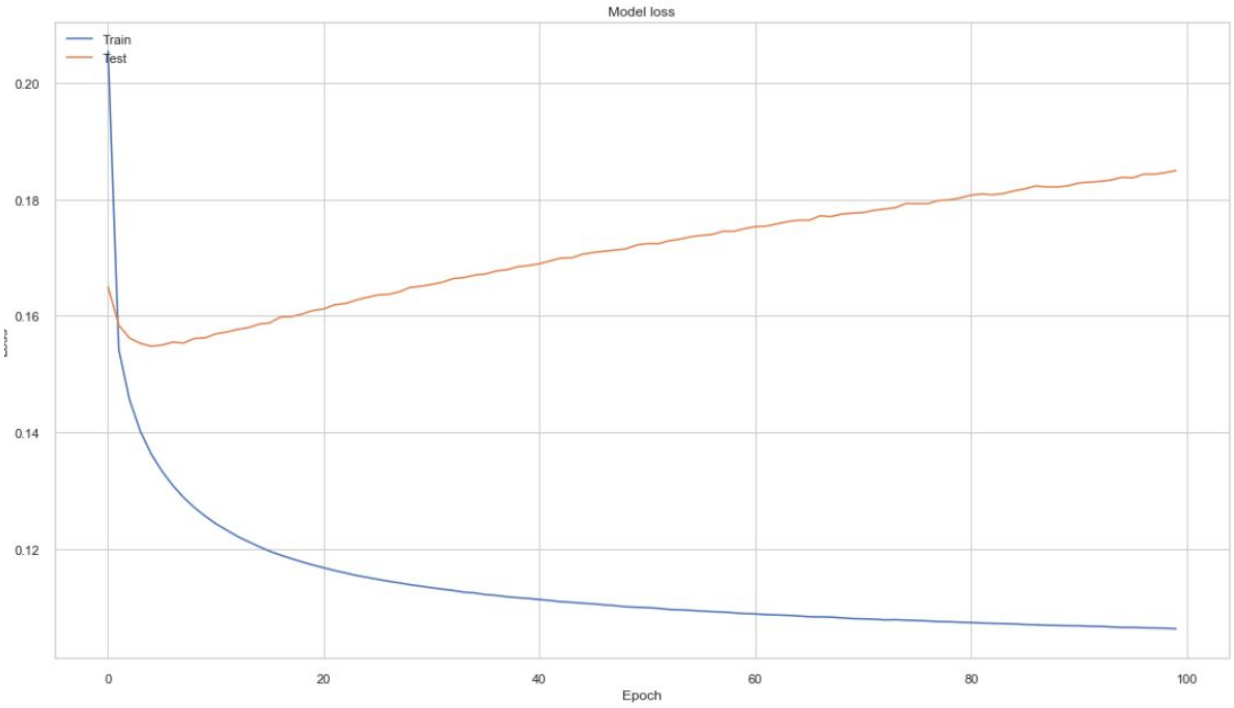
- Word2Vec.
- GloVe.

In addition to these carefully designed methods, a word embedding can be learned as part of a deep learning model. This can be a slower approach, but tailors the model to a specific training dataset.

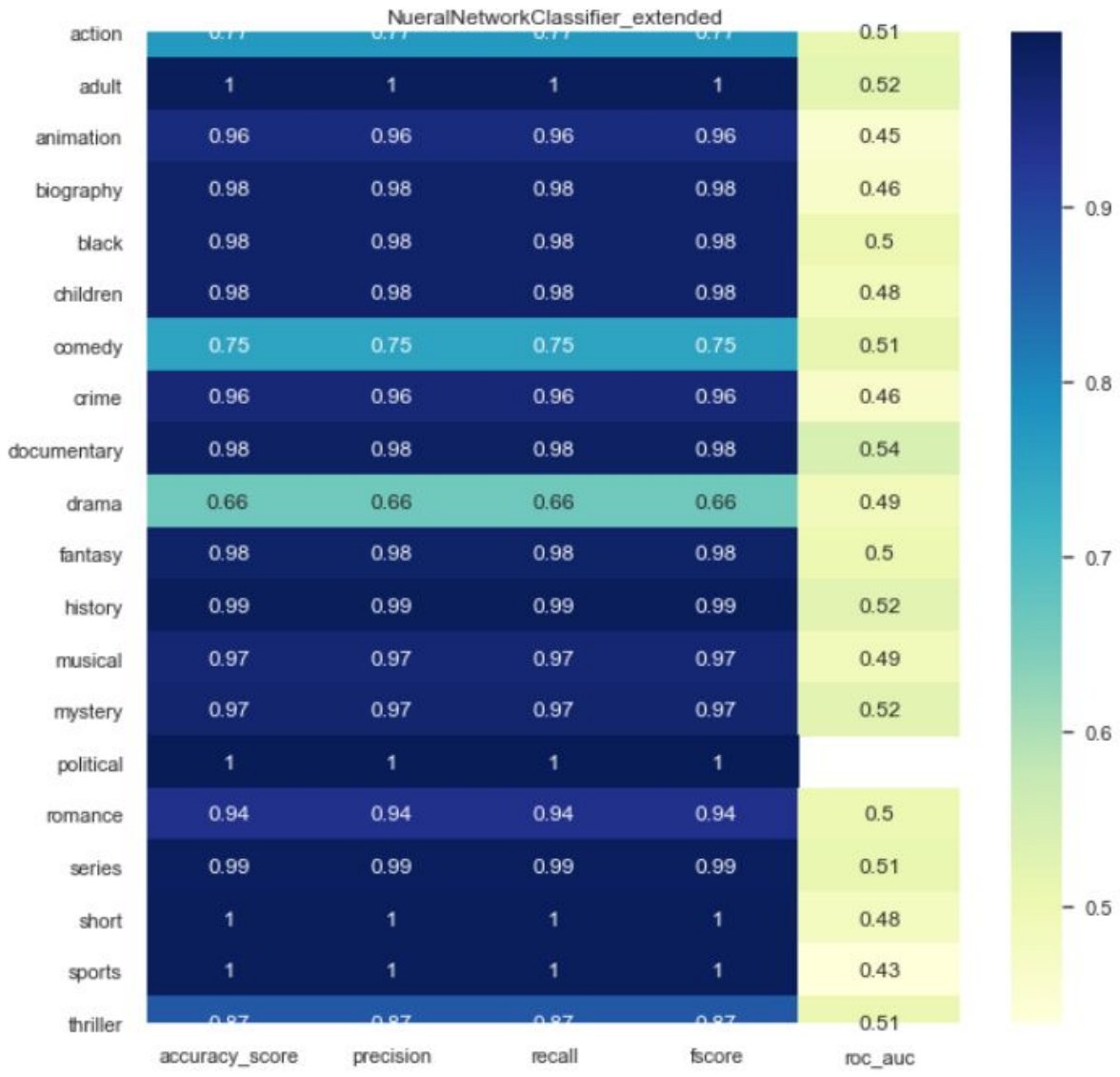
Results

Training & validation - accuracy and loss values

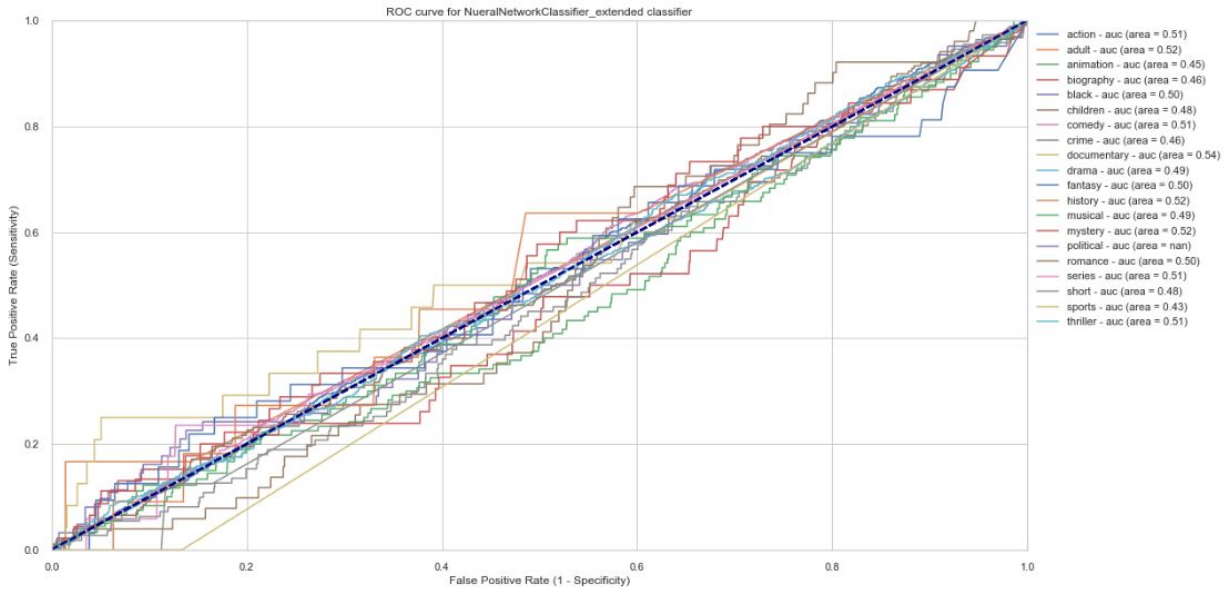




Accuracy Scores

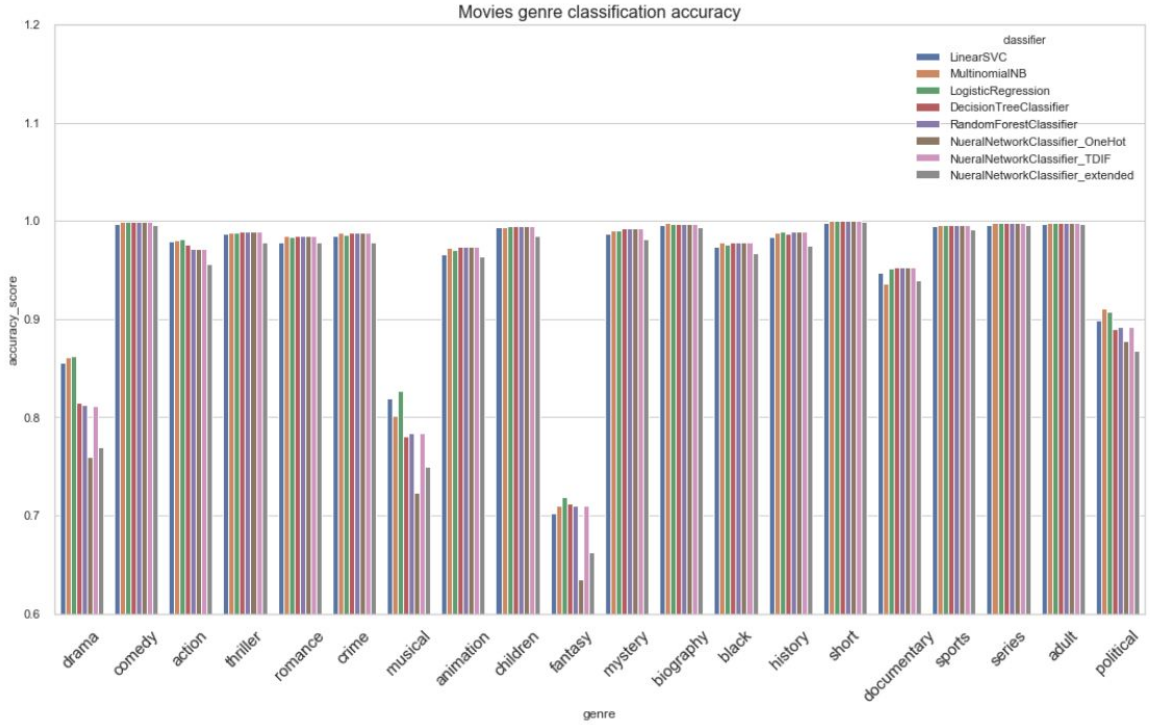

NueralNetworkClassifier_extended
action
adult
animation
biography
black
children
comedy
crime
documentary
drama
fantasy
history
musical
mystery
political
romance
series
short
sports
thriller
0.51
1 1 1 1 0.52
0.96 0.96 0.96 0.96 0.45
0.98 0.98 0.98 0.98 0.46
0.98 0.98 0.98 0.98 0.5
0.98 0.98 0.98 0.98 0.48
0.75 0.75 0.75 0.75 0.51
0.96 0.96 0.96 0.96 0.46
0.98 0.98 0.98 0.98 0.54
0.66 0.66 0.66 0.66 0.49
0.98 0.98 0.98 0.98 0.5
0.99 0.99 0.99 0.99 0.52
0.97 0.97 0.97 0.97 0.49
0.97 0.97 0.97 0.97 0.52
1 1 1 1
0.94 0.94 0.94 0.94 0.5
0.99 0.99 0.99 0.99 0.51
1 1 1 1 0.48
1 1 1 1 0.43
0.51
accuracy_score
precision
recall
fscore
roc_auc
0.9
0.8
0.7
0.6
0.5

ROC AUC Plot



Result

Performance of the models



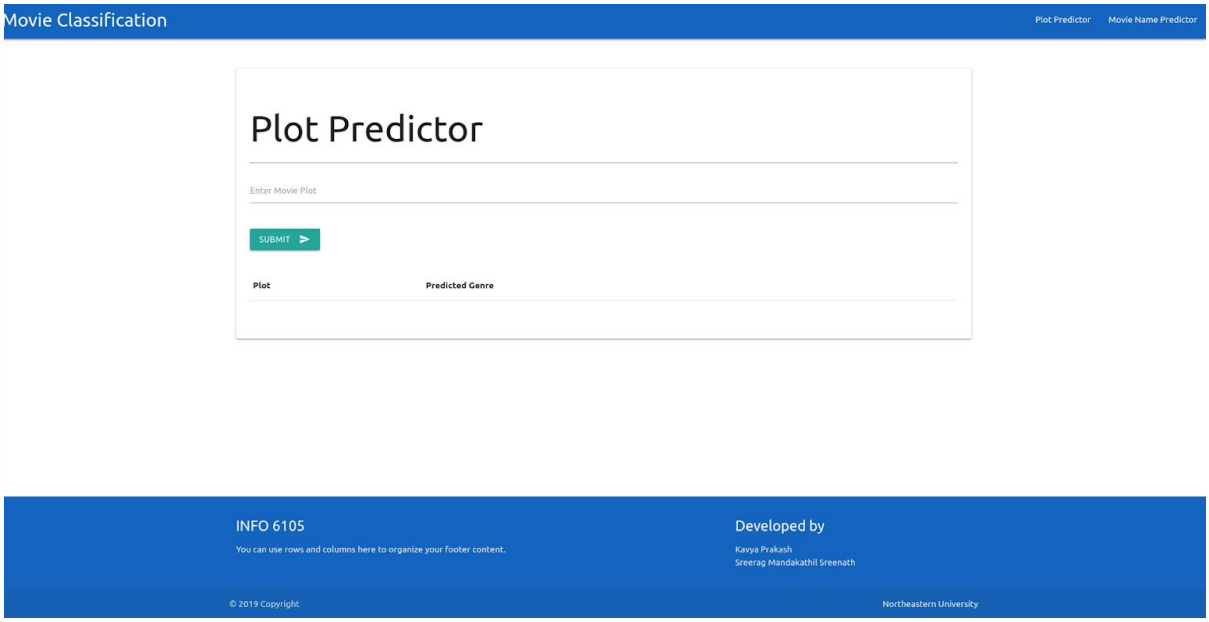
Web Application

Technologies

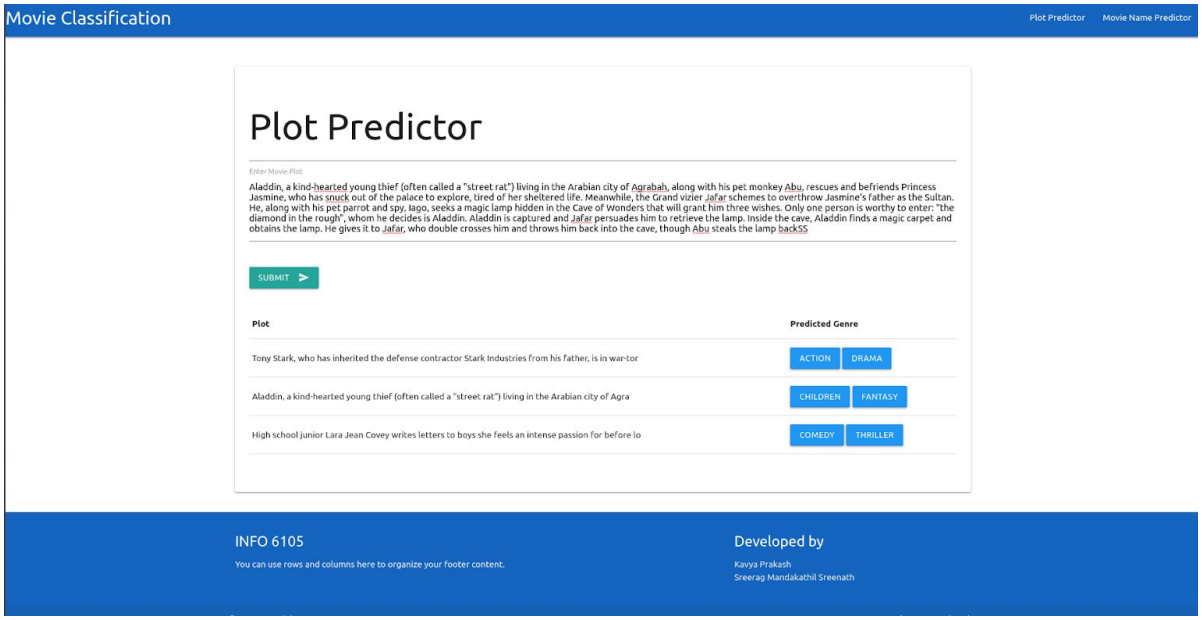
Sno.	Library	Purpose
1	Flask	Web Server
2	Scikit	Model Predictor
3	Pickle	Load/Save Model

Function

Movie Plot Prediction

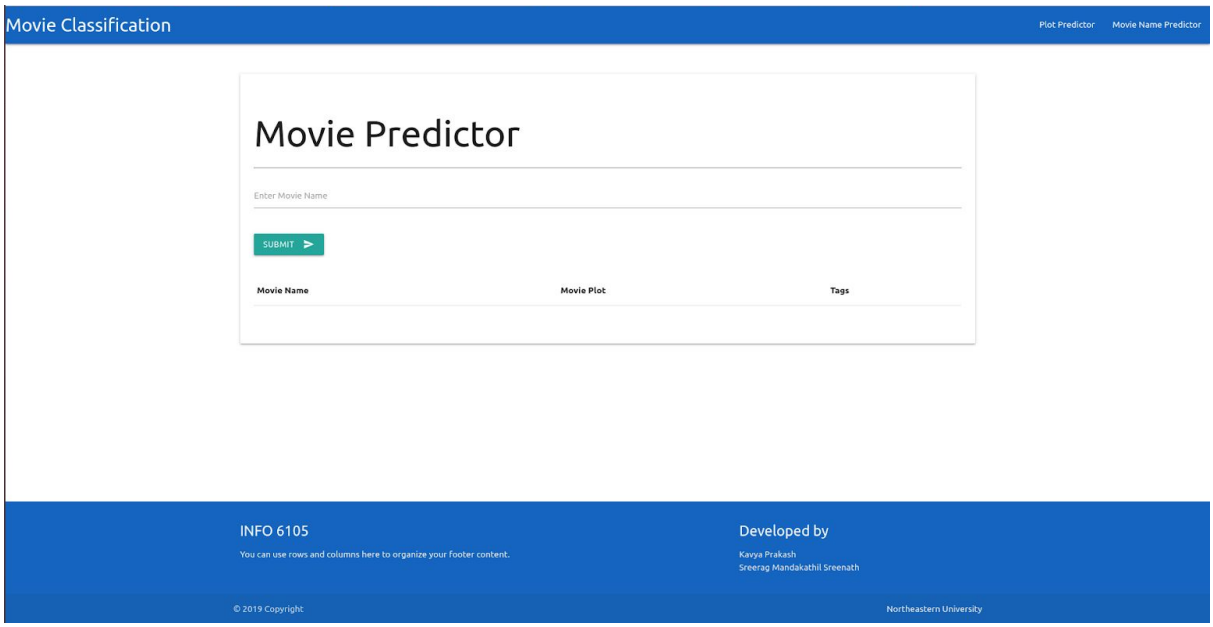


Before Plot entry

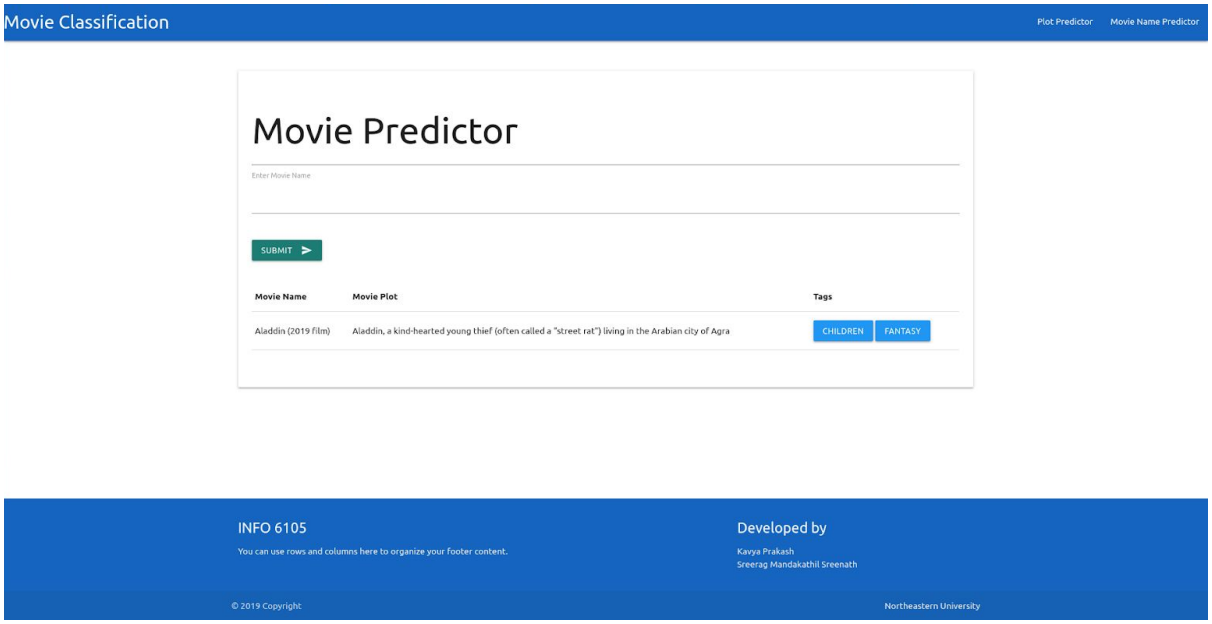


After Plot entry

Movie Title Prediction



Before movie name entry



After Movie name entry it searches for the movie in wikipedia and give their genre

Github Repository

https://github.com/sreeragsreenath/info6105_project

Conclusion

1. Conducted EDA on the Data
2. Learned and Applied NLP on the plot content
3. Tested out various classic and deep machine learning models
4. Built a pipeline for the best result and pickle the models
5. Developed a simple to use web application as an API for new classification

From the above we can see that LogisticRegression, MultinomialNB, RandomForestClassifier, NueralNetworkClassifier_OneHot and DecisionTreeClassifier were been able to have higher accuracies in many genres.

Random Forest seems to have an over better results than most of the classifiers

We finally used a combination of LogisticRegression, MultinomialNB, RandomForestClassifier, NueralNetworkClassifier_OneHot and DecisionTreeClassifier for the final web application.
All the results of the model comparison are store into classifier_result.csv