# Film Classification

**Abstract**

NLP is a field that emphasises practice and application, where innovations can be new algorithms, tasks, applications, data, discoveries, etc., and its impact depends on how it promotes the development of the field. In the field of artificial intelligence, natural language processing belongs to a relatively small branch, but it serves a very important role. Chatbots with intelligent replies designed through a large amount of data analysis, navigation assistants with different voices and even local dialects in China, knowing the attitude of Twitter users towards a certain news by analysing their positive and negative emotions, and knowing the attitude of Twitter users towards a certain news through a large number of readings of high-frequency vocabulary of spam text messages , automatically filtering spam content.

The usefulness of NLP goes far beyond this, with the emergence of ChatGPT, it is also possible to be a 'screenwriter' yourself, by inputting a film genre and keywords to ChatGPT, for example, input the film genre, comedy film, and then input the keywords, Japan, school, ghosts, etc., and ChatGPT will automatically generate a film based on that content and by combining it with a web A large amount of film content, automatically generate a film. In order to figure out the principle and study the related NLP modules, for this purpose I worked on a film classification application, this dataset contains different types of films and film lines, the original dataset was a time-limited competition in Kaggle with a total number of 22580 entries, in order to run the code in a more portable way I removed the number of datasets in the train to 15471 entries. I use, and make predictions for film lines. Using numpy, os, I trained the dataset using LSTM, CNN and MultinomialNB to train the dataset model for the main training, predicted the accuracy of CNN, and predicted the text of the movie lines using MultinomialN, and at the end, I used IF - IDF, LDA, sentence\_transform package to read the code and produce new model data.

**Data Cleaning and Preprocessing**

Firstly, the dataset is divided into three categories: 'train', 'test', and 'Val', and in the subsequent training, since these three datasets are too large, the following datasets are added: 'val1' with only a thousand or so data. 'val1' has only a thousand or so pieces of data, after Create the environment, set numpy, pandas , os, matplotlib, seaborn as sns %matplotlib inline as the basis of the run of the environment code module import. Defined as df, using the panda module to read the training set/movie\_train.csv, read to learn that there are a total of three columns, respectively 'id, 'text', 'genre', the 15470 film data. Open train.csv to confirm that the reading is correct.

图形用户界面, 文本, 应用程序

描述已自动生成文本

描述已自动生成

'id' represents the serial number of the film, 'text' represents a part of the lines of this film, 'genre' represents the type of the film, in the visualised database. Calculate the number of 'genre', set the chart size figsize=(14,10), y-axis is the number, x-axis is the name of the genre, use %matplotlib to show plt.show(). Yields a total of nine types of films, 'action', 'adventure', 'comedy', 'drama', 'horror', 'other', 'romance', 'sci-fi', 'thriller', with drama having the highest number of 6100, and the second highest number of film types being thriller with about 4750 data and the least number of films is 'romance' with only about six per cent of the 15740 films.

图表, 条形图

描述已自动生成

Next the nine film genres are defined as VALUES as 'other': 0, 'action': 1, 'adventure': 2, 'comedy':3, 'drama':4, etc., and 'genre' is changed from ' thiller', 'comdy' to '8', '3' defined respectively. Prepare for the next cleaning data, data preprocessing.

文本, 信件

描述已自动生成

表格

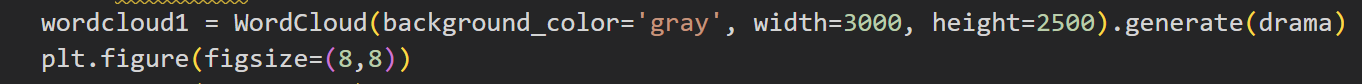
描述已自动生成

Installation of the NLTK library. The NLTK library provides a rich set of natural language processing functions and tools, offering a variety of text preprocessing tools, including text cleaning, text normalisation, word splitting, etc. These tools can help users quickly transform raw text data into a data format that can be used for further analysis. Using the nltk library's 'stopwords', stopwords are common words that are ignored in text processing, such as "the", "a", "an", etc. to clean the text, remove special characters from dialogues/scripts, convert the whole dialogues/scripts to lower case, tag dialogues/scripts by words, remove stop words, stem words, join words in the stem, and create a corpus.

Install the wordcould library to generate word cloud images from text, the wordcloud library treats the word cloud as a WordCloud object and needs to be installed via the pip command:

图片包含 应用程序

描述已自动生成

Select 'drama' 'action' 'comedy' to make a wordcloud image, create a WordCloud object, use the .generate() method to load the text. Calculate the word frequency, output the wordcloud file and display the image using pyplot. 文本

描述已自动生成一些文字和图案

描述已自动生成图片包含 日历

描述已自动生成

**LSTM**

Text data processing

Tokenizer is a class for vectorizing text, or converting text into a sequence (i.e. a list consisting of the subscripts of words in a dictionary, counting from 1).list of sequences, each sequence in the list corresponds to a piece of input text. padding sequences pad\_sequences converts sequences of length nb\_smaples to (nb\_samples,nb\_timesteps)2Dnumpy attay. if maxlen is supplied, nb\_timesteps=maxlen。

**Simple RNN**

Import the keras model, first construct an RNN class, add the first RNN and Dropout to prevent overfitting. If return\_sequences is True, it means there is another RNN upstream and return, (batch\_size,time\_step,units) shape matrix, otherwise return (batch\_size,units) matrix. Setting Embeding, embedding is an idea to compress a high dimensional sparse tensor into a low dimensional dense tensor for easy representation and computation.



文本

低可信度描述已自动生成图形用户界面, 表格

描述已自动生成

*Total params: 20330881 (77.56 MB) Trainable params: 20330241 (77.55 MB) Non-trainable params: 640 (2.50 KB)*

The accuracy of the Validation dataset is highest at epoch 6, and the accuracy of the Training datase is highest at around epoch 10, and then both gradually decrease together.The Model Loss curve is similar for both datasets, with a gradual decrease.

图表, 折线图

描述已自动生成

**Sklearn-Naive Bayes**

This is a probabilistic model based on Bayes' theorem, which assumes that all features are conditionally independent from each other. This assumption makes the Naive Bayes model computationally and learning efficient, and also makes it perform well in natural language processing tasks such as text classification and sentiment analysis.

Fitting Naive Bayes to the training set and predicting the results of the test set, the Accuracy score is: 89.01% using sklearn.metrics module. 图形用户界面, 文本, 应用程序, 聊天或短信

描述已自动生成

The confusion matrix then aggregates the records in the dataset in a matrix form according to two criteria: the true category and the category judgment predicted by the classification model. The rows of the matrix represent the true values and the columns of the matrix represent the predicted values. Accuracy is calculated using a test set and Accuracy score is: 89.01%. Import the confusion matrix and evaluate it. Get the following parameters, which are derived from the plt export graph.

图片包含 表格

描述已自动生成

日历

中度可信度描述已自动生成

*Adjusting the hyperparameters of Naive Bayes classifier, the adjusted Accuracy score for alpha=0.1 is: 91.18%*

文本

描述已自动生成

**Predictions**

Word set the basic requirements, imported the random module and validated the test set with the model. Randomly generate textual content and film types.

文本, 信件

描述已自动生成

**LDA**

LDA (Latent Dirichlet Allocation) is a document topic generation model, also known as a three-layer Bayesian probabilistic model, which contains three layers of words, topics, and documents. The so-called generative model, that is, we believe that each word in an article is obtained through a process of "selecting a certain topic with a certain probability, and selecting a certain word from this topic with a certain probability". Documents to topics follow a polynomial distribution, and topics to words follow a polynomial distribution.

图表, 折线图

描述已自动生成

The 'stop word' package is called on the data to remove irrelevant high-frequency but not very semantic words, and the Gensim package is called for text mining. Semantic topics can be extracted from documents efficiently and automatically. Call Gensim in LDA to do text mining, build a document-term matrix. obfuscation model, calculate relevance, and come up with the following picture:

文本

描述已自动生成

*Character ratios from LDA training*

图形用户界面, 应用程序

描述已自动生成

*Visualisation image topic.html*

**Comparison between the bag-of-words model and TF-IDF**

The main idea of TF-IDF: If a word or phrase appears in an article with a high Term Frequency (TF), and has a high word frequency and rarely appears in other articles, it is considered that this word or phrase has a good ability to distinguish between categories and is suitable for classification. The bag-of-words approach does not take into account the order of words, which simplifies the complexity of the problem and provides an opportunity for model improvement. Each document represents a probability distribution consisting of some topics, and each topic represents a probability distribution consisting of many words.

图片包含 表格

描述已自动生成

*bag-of-words model*

图片包含 日历

描述已自动生成

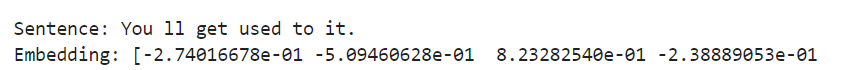
*TF-IDF Model*

**Sentence Transformers**

Sentence Transformers is a Python library that can be used for sentence, text and image embeddings. Text embeddings can be computed for more than 100 languages and they can be easily used for common tasks such as semantic text similarity, semantic search, and synonym mining. Inference using Pretrained Model to obtain embedding vectors ... Select three sentences with similar subject, predicate based on three partial film lines, two dramas and one comedy as predicted by plain Bayes. And obtain the distance of similarity as 0.76, 0.55,0.51.

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

文本

描述已自动生成

**Conclusion**

With the development of Artificial Intelligence, Natural Language Processing will be everywhere in the future, RNN and LSTM as well as Naive Bayes, in my opinion, the most accurate one is the Simple Bayes, which learns the lines of different movies by using obfuscated computation, and it is more accurate in predicting the lines of the movies, and it can get more than fifty-five percent similarity by using SentenceTransformers to analyse the similarity of the underlying semantic text. The semantic text similarity of the base of the analysis can also be obtained more than fifty-five per cent similarity.

LDA is suitable for text cleaning, call different packages in LDA, such as genism, history, etc., to analyse what is the commonality of high-frequency words, and if possible, import into Naive Bayes, use sklearn and other modules for deep learning, and then use high-frequency vocabulary to predict new movie text.