<https://datadrivenscience.com/master-movie-genre-prediction-with-nlp-a-comprehensive-guide-to-imdb-dataset-analysis-and-lstm-modeling/>

Dataset

<https://www.kaggle.com/datasets/PromptCloudHQ/imdb-data>

use a [**Long Short-Term Memory (LSTM)**](https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/) model for genre classification. LSTM is a type of [**recurrent neural network (RNN)**](https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/) that can learn and remember long-term dependencies in sequences, making it suitable for text classification tasks.

Once we have the dataset, we will preprocess the movie plot synopses using techniques like tokenization, stopword removal, and lemmatization. These steps help in reducing noise and making the text more suitable for analysis.

<https://becominghuman.ai/mathematical-introduction-to-glove-word-embedding-60f24154e54c>

<https://nlp.stanford.edu/projects/glove/>

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

<https://github.com/wittyicon29/GenreGenie-A-Movie-genre-prediction-model>

Epoch 1/10

WARNING:tensorflow:From C:\magister\venv\lib\site-packages\keras\src\utils\tf\_utils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From C:\magister\venv\lib\site-packages\keras\src\engine\base\_layer\_utils.py:384: The name tf.executing\_eagerly\_outside\_functions is deprecated. Please use tf.compat.v1.executing\_eagerly\_outside\_functions instead.

569/569 [==============================] - 1005s 2s/step - loss: 0.1831 - accuracy: 0.1179 - val\_loss: 0.1754 - val\_accuracy: 0.1212

Epoch 2/10

569/569 [==============================] - 1013s 2s/step - loss: 0.1677 - accuracy: 0.1528 - val\_loss: 0.1656 - val\_accuracy: 0.1854

Epoch 3/10

569/569 [==============================] - 991s 2s/step - loss: 0.1482 - accuracy: 0.2469 - val\_loss: 0.1545 - val\_accuracy: 0.2234

Epoch 4/10

569/569 [==============================] - 990s 2s/step - loss: 0.1311 - accuracy: 0.3086 - val\_loss: 0.1528 - val\_accuracy: 0.2551

Epoch 5/10

569/569 [==============================] - 996s 2s/step - loss: 0.1166 - accuracy: 0.3559 - val\_loss: 0.1528 - val\_accuracy: 0.2738

Epoch 6/10

569/569 [==============================] - 985s 2s/step - loss: 0.1031 - accuracy: 0.4013 - val\_loss: 0.1551 - val\_accuracy: 0.2869

Epoch 7/10

569/569 [==============================] - 986s 2s/step - loss: 0.0906 - accuracy: 0.4397 - val\_loss: 0.1606 - val\_accuracy: 0.2890

Epoch 8/10

569/569 [==============================] - 1037s 2s/step - loss: 0.0798 - accuracy: 0.4629 - val\_loss: 0.1686 - val\_accuracy: 0.3057

Epoch 9/10

569/569 [==============================] - 1003s 2s/step - loss: 0.0697 - accuracy: 0.4845 - val\_loss: 0.1770 - val\_accuracy: 0.3021

Epoch 10/10

569/569 [==============================] - 995s 2s/step - loss: 0.0610 - accuracy: 0.4963 - val\_loss: 0.1911 - val\_accuracy: 0.3072

285/285 - 40s - loss: 0.1911 - accuracy: 0.3072 - 40s/epoch - 139ms/step

Validation Accuracy: 0.3072355389595032