

Open in app ↗

Medium

 Search Write

M

★ Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#)



Machine learning project: predicting song genre from lyrics

Martin Kostrubanič · [Follow](#)

5 min read · Dec 30, 2021



2



Introduction

As my semester project, I have decided to build a model, which would be able to predict a song genre from its lyrics. It is not an easy problem, because some genres are quite similar lyricwise and some songs can belong to more genres. I have chosen 5 genres to classify between: pop, rock, country, electronic and hip-hop. I have compared 4 models: DNN(deep neural network), XGB(tree classifier) RNN(recurrent neural network) and CNN(convolutional neural network). DNN and XGB used tf-idf text representation and RNN and CNN used word embedding. The project is available on my [Gitlab page](#).

Data

I used dataset from Kaggle. It consists of almost 300 000 songs with lyrics and genre label. The genre labels were obtained thanks to Spotify library. To each artist was assigned his most common genre. The dataset contains a lot of pop and rock songs, but sadly not so many songs from other genres. I had to make the dataset balanced, so there was the same number of songs of each genre. After balancing only over 13 000 songs left.

I removed all diacritic and useless symbols from lyrics and turned it all to lowercase. Next, I tokenized the data, this means I split it into individual words. After that I removed stopwords. Stopwords are words, which occur in almost every text, like *I*, *with* or *do*. These words aren't helpful with classification. Then I did stemming. Stemming is a technique, which turns words into their stems. For example word *presentation* is turned into *present*, *seeing* is turned into *see* etc. It reduces dimensionality of the data. In the end of data preparation I split the data into train, validation and test subsets.

Text representation

Tf-idf

Term frequency-Inverse document frequency is a simple text representation. Each document is represented as a vector of length of the vocabulary of the dataset. Each index of the vector represents one word and is equal to $tf \cdot \log(N/d)$, where tf is number of occurrences of the given word in the given document divided by length of the document, N is total number of documents in the dataset and d is number of documents, which contain the given word. Thanks to second part of the multiplication common words are not so important and specific words, which can help with classification, get more importance. Tf-idf doesn't take into account similar words and order of the words.

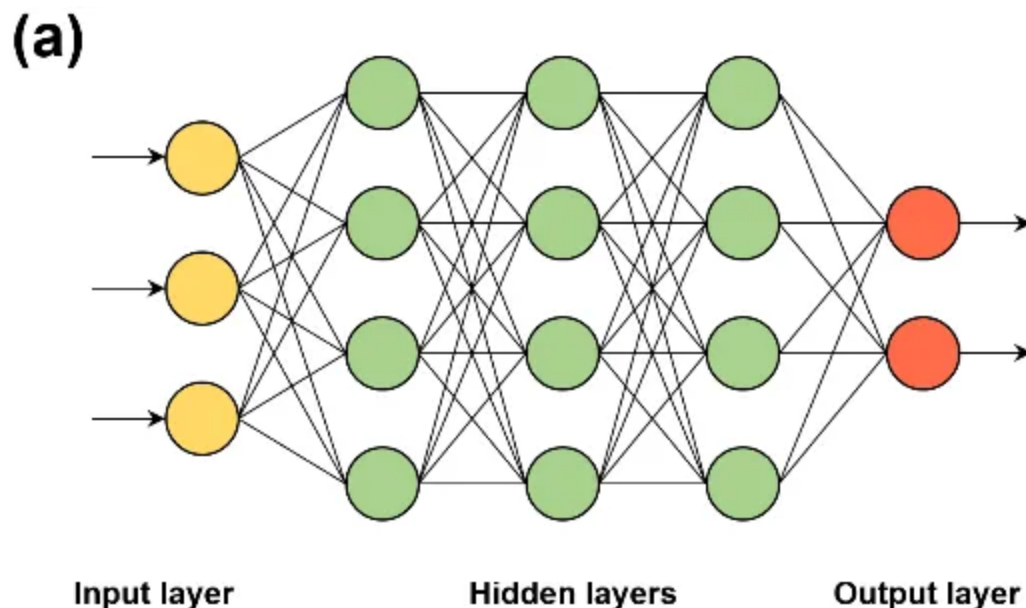
Word embedding

More sophisticated text representation is word embedding. This technique represents each word as a vector. The length of the vector can vary, but it is usually in tens or hundreds. Important is, that similar words are represented by similar vectors, that make a small angle. A document is then represented by vector of vectors. This way, order of the words is taken into account.

Models

DNN

Deep neural network is a network of neurons. The neurons are connected into layers. The input layer is a representation of a document. Each neuron in the output layer stands for one class. Other layers are hidden. Training of DNN is driven by gradient descent, which optimizes weights of connections between neurons.



Source: [ResearchGate](#)

XGB

XGB is a tree classifier. Tree classifiers send data points into one of two branches in each node depending on value of one feature. Data point ends in a leaf, which is assigned to a class. Training of tree classifier is done by choosing right feature and deciding value for each node. XGB uses more of such tree classifiers. These classifiers usually have quite a small depth.

RNN

Recurrent neural networks care about the order of given data, that's why they are commonly used for text classification. To prevent problems of exploding and vanishing gradient, long short-term memory(LSTM) is being used. LSTM is a special type of RNN. It has a similar structure, but uses some gates to control, how much information is being memorized.

CNN

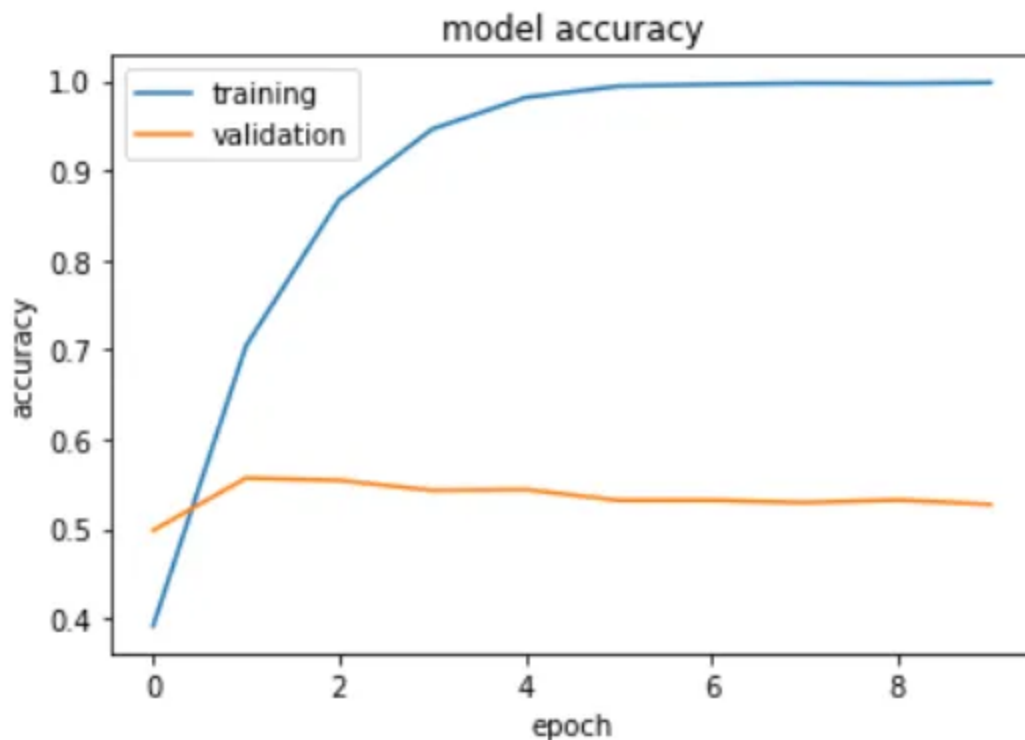
Convolutional neural network was originally built for image processing, but is commonly used for text classification as well. It consists of several filters, which extract patterns from given data. The filters create layers, which are stacked, to extract even pretty complex patterns. To reduce the computational complexity, CNN uses pooling to reduce the size of the output from one layer to the next layer in the network.

Results

Model	Validation accuracy
DNN	55.77 %
XGB	53.20 %
RNN	51.1 %
CNN	48.58 %

The best result was achieved by DNN with tf-idf. I was expecting RNN and CNN with word embedding to perform better, but it is probably because for this task, the order of the words is not really important, but their appearance is.

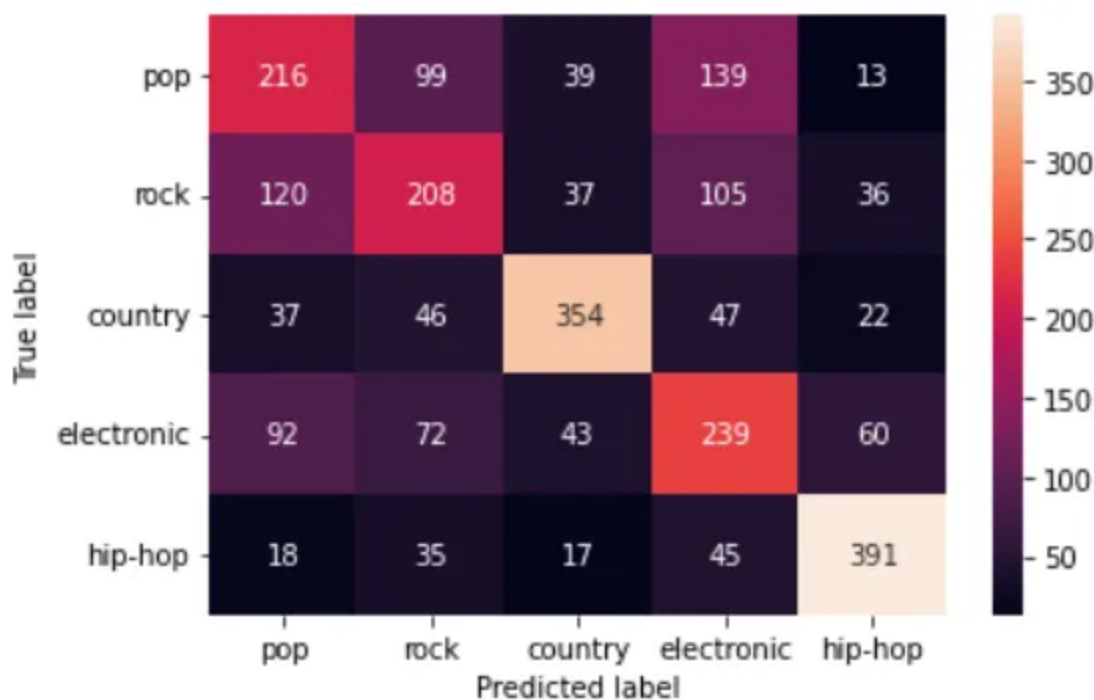
Lets have a closer look at the best performing model. I experimented with different structures of the DNN, but differences were quite small. In the end, I built a network with 2 hidden layers with 1024 neurons each. I also added a dropout layer with value 0.2, which turns off some of the neurons. I trained the model for 10 epochs.



After 2 epochs, the DNN started to overfit. The thing that helped against overfitting the most, was balancing the dataset. Without balancing, the model only predicted pop or rock and had validation accuracy only around 30 %. Adding a dropout layer also helped. Turning off some neurons makes

the network generalize more. Removing stopwords and stemming helped a little bit as well.

I printed a confusion matrix of the validation data.



As we can see, model predicts country and hip-hop quite well. The problematic genres are pop, rock and electronic, which the DNN confuses sometimes. That is understandable, because lyrics of these genres are often quite similar.

Lastly, I let this model predict the test data. It achieved accuracy of 54.82 %.

Conclusion

The results aren't the greatest, but given the similarity of some genres, they aren't the worst either. I think the main problem was the small amount of data, respectively small amount of electronic, country and hip-hop songs.

Data Science

Machine Learning

Text Classification

**Written by Martin Kostrubanič**

5 Followers · 1 Following

Follow

More from Martin Kostrubanič

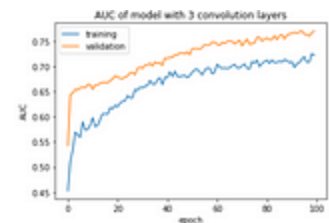


Martin Kostrubanič

Graph classification using DGCNN

As my semester project I have chosen graph classification on molhiv dataset. Graph classification is an important problem,...

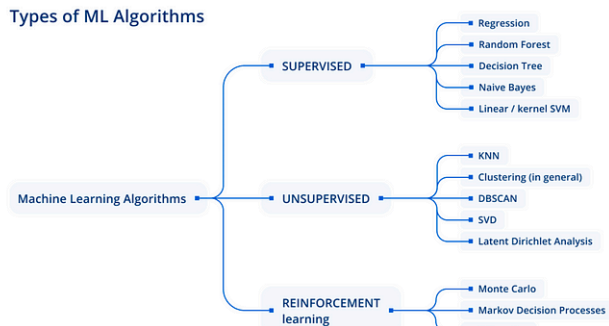
May 5, 2022 🖱 1



See all from Martin Kostrubanič

Recommended from Medium

Types of ML Algorithms



John Vastola

10 Must-Know Machine Learning Algorithms for Data Scientists

Machine learning is the science of getting computers to act without being explicitly...



Dec 6, 2022



928



12



In Stackademic by Abdur Rahman

Python is No More The King of Data Science

5 Reasons Why Python is Losing Its Crown



Oct 22



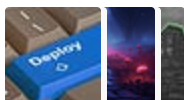
9K



34



Lists



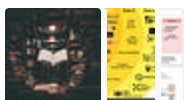
Predictive Modeling w/ Python

20 stories · 1701 saves



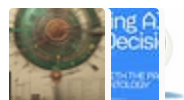
Practical Guides to Machine Learning

10 stories · 2073 saves



Natural Language Processing

1845 stories · 1468 saves



data science and AI

40 stories · 296 saves

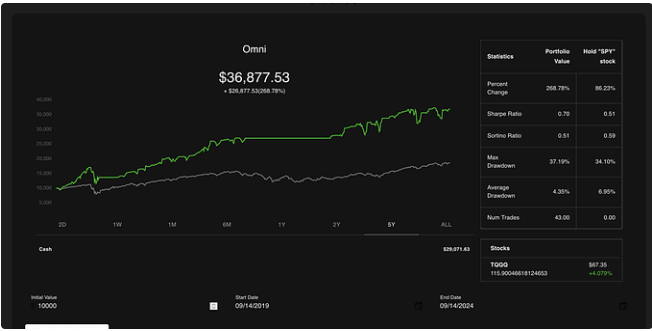


 In Towards AI by Shenggang Li

Dynamic Weight Models: Bridging GLM and Neural Networks

Introduction

★ 5d ago 🖱️ 229 💬 5  ⋮




 In DataDrivenInvestor by Austin Starks

I used OpenAI's o1 model to develop a trading strategy. It is...

It literally took one try. I was shocked.

★ Sep 15 🖱️ 6.8K 💬 180  ⋮

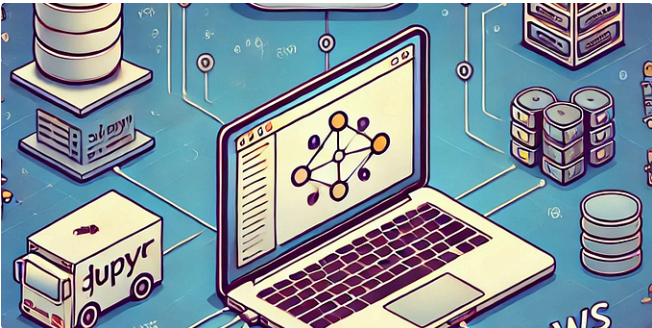



 Rahul Kumar

NLP Hands-On with Text Classification

This post is a part of the NLP Hands-on series and consists of the following tasks: 1. Text...

★ Oct 27  ⋮



 In Artificial Intelligence in Plain En... by Ritesh Gu...

From Jupyter to Production: Deploying Machine Learning...

Turn your Jupyter Notebook experiments into production-ready applications with this...

★ Oct 24 🖱️ 374 💬 2  ⋮

See more recommendations