Musk and Non-Musk Classification

Project Report

Submitted By-

Prashant Patel

ABSTRACT

Musk is a class of aromatic substances commonly used as base notes in perfumery. This Project deals with the classification of the organic chemical compounds into the classes 'Musk' and 'Non-Musk' based on their chemical features, isomeric conformation and names. The classification is based on the Artificial Neural Network (ANN) Deep Learning model. Model is evaluated by using graphs of Training and Validation Accuracy and Loss, Confusion matrix, Precision and Recall.

1. Data

This dataset describes a set of 102 molecules of which 39 are judged by human experts to be musks and the remaining 63 molecules are judged to be non-musks. The goal is to learn to predict whether new molecules will be musks or non-musks. However, the 166 features that describe these molecules depend upon the exact shape, or conformation, of the molecule. Because bonds can rotate, a single molecule can adopt many different shapes. To generate this data set, all the low-energy conformations of the molecules were generated to produce 6,598 conformations.

	ID	molecule_name	conformation_name	f1	f2	f3	f4	f5	f6	f7	...	f158	f159	f160	f161	f162	f163	f164	f165	f166	class
0	1	MUSK-211	211_1+1	46	-108	-60	-69	-117	49	38	...	-308	52	-7	39	126	156	-50	-112	96	1
1	2	MUSK-211	211_1+10	41	-188	-145	22	-117	-6	57	...	-59	-2	52	103	136	169	-61	-136	79	1
2	3	MUSK-211	211_1+11	46	-194	-145	28	-117	73	57	...	-134	-154	57	143	142	165	-67	-145	39	1
3	4	MUSK-211	211_1+12	41	-188	-145	22	-117	-7	57	...	-60	-4	52	104	136	168	-60	-135	80	1
4	5	MUSK-211	211_1+13	41	-188	-145	22	-117	-7	57	...	-60	-4	52	104	137	168	-60	-135	80	1

2. Pre-processing

Data is pre-processed by passing through the following steps.

2.1 Dropping the Columns

Data set contains features which are not important for training the model e.g. ID and conformation_name. These features can be removed from the data set.

2.2 Encoding the Categorical Features

Data set contains feature “molecule_name” which is a categorical variable. Since machine learning models are all about numbers and calculations, these categorical variables need to be coded into numbers. Label Encoder is used to encode the “molecule_name”.

2.3 Scaling the Features

The scale and distribution of the data drawn from the domain may be different for each variable. Unscaled input variables can result in a slow or unstable learning process, whereas unscaled target variables on regression problems can result in exploding gradients causing the learning process to fail. In this project StandardScaler is used to scale the input to a common scale.

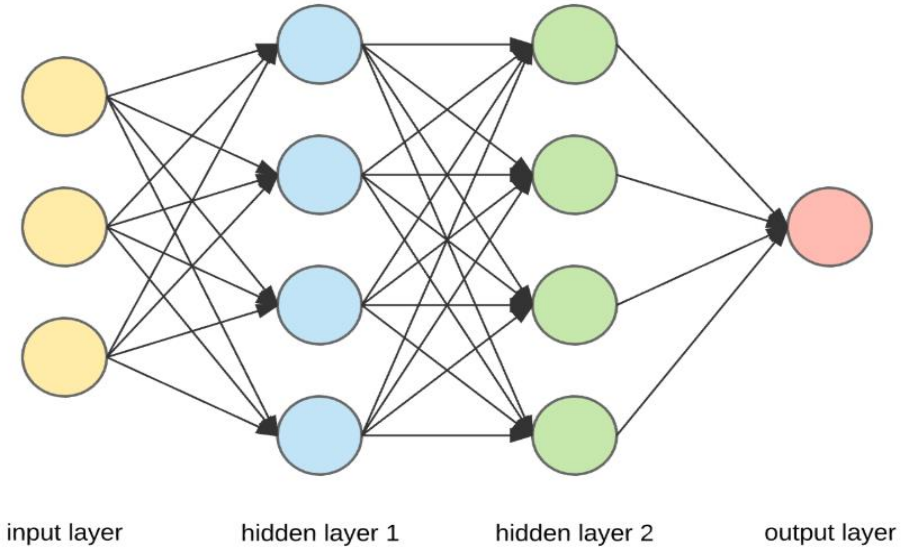
2.4 Splitting the Data Set into Training and Test Set

Machine learning models require us to provide a training set for the machine so that the model can train from that data to understand the relations between features and can predict for new observations. It is a good idea to split the dataset into to two, a training set and a test set, so that we can test our model after its been trained with the training set. In this project data is splitted in 80:20 ratio for training and validation datasets.

3. Artificial Neural Networks

Artificial Neural Networks (ANN) are multi-layer fully-connected neural nets that look like the figure below. They consist of an input layer, multiple hidden layers, and an output layer. Every node in one layer is connected to every other

node in the next layer. We make the network deeper by increasing the number of hidden layers.



3.1 Initializing and Adding layers to the Network

In this project I have used three layers to build the network. There are two hidden layers and one output layer. There are various parameters are used for initializing the layer these parameters are:

- **units:** It is the number of nodes present in the layers. In this project 84 nodes are used for both hidden layers and one node for output layer because we have to classify the data into binary classes.
- **kernel_initializer:** Random weights are initialized according to algorithm specified by this parameter.
- **activation:** In ANN we do the sum of products of inputs(**X**) and their corresponding Weights(**W**) and apply a Activation function **f(x)** to it to get the output of that layer and feed it as an input to the next layer. In this project Rectified Linear Unit (ReLU) activation function is used for hidden layers and Sigmoid is used for output layer.

3.2 Compiling the Network

Compile function bind all the layers in the network and specifies the loss and optimizer to be used in model. Adam optimizer is used in this model with the learning rate of 0.0001.

3.3 Fitting the Network

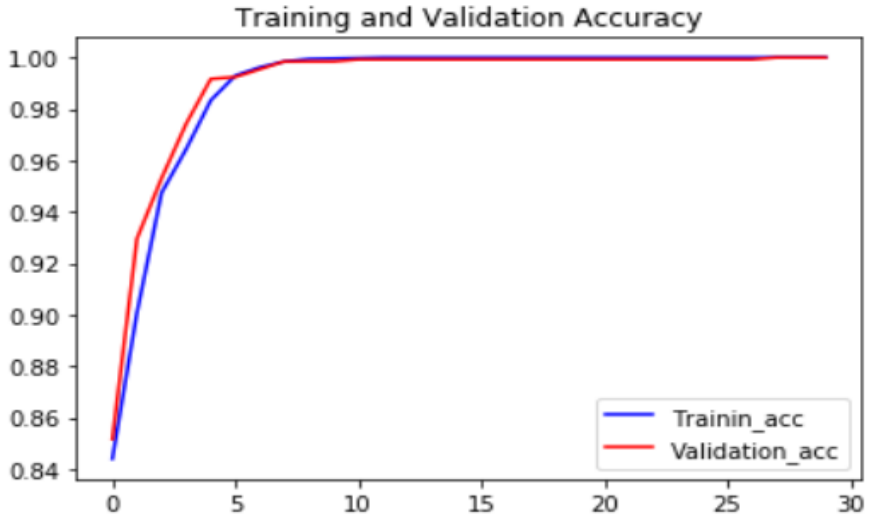
The model is trained for 30 epochs and batch size of 16 on the training set and validated on the test set.

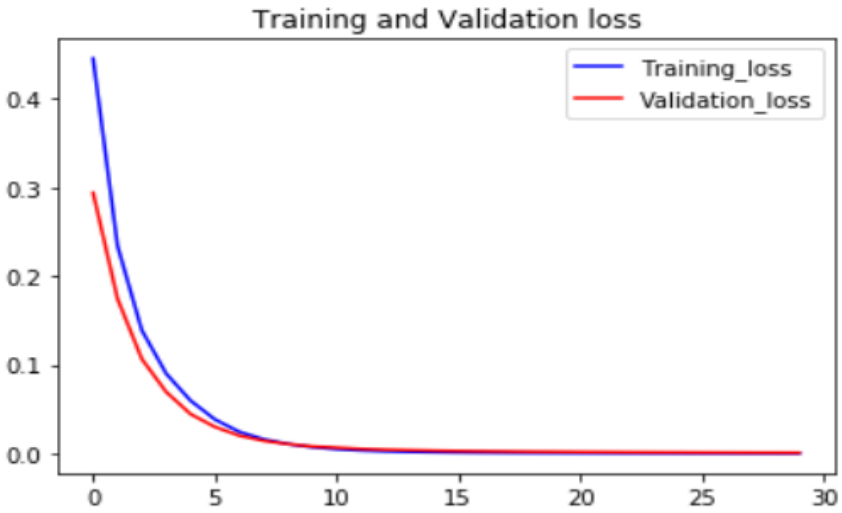
```
Epoch 26/30
5278/5278 [==============================] - 1s 113us/step - loss: 8.9666e-05 - accuracy: 1.0000 - val_loss: 8.7618e-04 - val_a
ccuracy: 0.9992
Epoch 27/30
5278/5278 [==============================] - 1s 110us/step - loss: 7.2065e-05 - accuracy: 1.0000 - val_loss: 8.1030e-04 - val_a
ccuracy: 0.9992
Epoch 28/30
5278/5278 [==============================] - 1s 112us/step - loss: 5.8466e-05 - accuracy: 1.0000 - val_loss: 7.3094e-04 - val_a
ccuracy: 1.0000
Epoch 29/30
5278/5278 [==============================] - 1s 112us/step - loss: 4.7873e-05 - accuracy: 1.0000 - val_loss: 6.9809e-04 - val_a
ccuracy: 1.0000
Epoch 30/30
5278/5278 [==============================] - 1s 112us/step - loss: 3.8764e-05 - accuracy: 1.0000 - val_loss: 6.0184e-04 - val_a
ccuracy: 1.0000
```

4. Performance Evaluation

I have achieved 100% accuracy by training the model for 30 epochs with learning rate 0.0001

4.1 Accuracy and Loss graphs





4.1 Confusion matrix

```
[[1120    0]
 [   0  200]]
```

4.2 Validation Accuracy

```
accuracy_score(y_test,y_pred)
```

```
1.0
```

4.3 Classification Report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1120
1	1.00	1.00	1.00	200
accuracy			1.00	1320
macro avg	1.00	1.00	1.00	1320
weighted avg	1.00	1.00	1.00	1320