Mini Project III
Guess the year by using PyTorch

(Baglan Aitu)

In this project, we have the dataset containing the list of songs, each with several attributes (essentially the parameters of timbre and covariance) that help to determine the year they were released.

The dataset (*YearPredictionMSD.csv*) contains data on 515,345 records, with release years ranging from 1922 to 2011.

The task was implemented in Google Colab.

Content:

Data analysis

1. Required libraries.

First, *Pandas* dataframe was used to read the dataset. After that, most of the data analysis was implemented by using the *Pytorch* framework. As it doesn't have functions for shuffling (reducing variance to overfit less) and splitting, the solution was to introduce the *Scikit-learn* library.

2. Checking for categorical features.

Categorical features are the nature of data in the dataset. Normally, they have the same type like numerical, textural, etc. Since we work with numbers, our dataset must have only numerical parameters. We can check it by subtracting the number of features to the total number.

Result: Output is zero which means the dataset contains only numerical features.

3. Checking for missing values

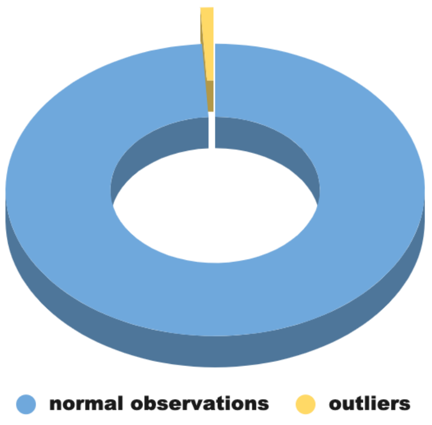
In many datasets, there are null or empty values. These values are useless for training Missing values have to be changed to a numerical value (for example 0, or interpolated from next values in the case of time series) or simply removed.

Result: No missing values.

4. Checking for outliers

Outliers are observations unlike the rest observations. It shifts our output result. To detect them, we set a threshold and decide as an outlier a value which is out of an interval. Then, for each feature we count the percentage of outliers.

Result: Each feature has a different percentage of outliers. They are between 0.5 and 1.87% (which is miserable).



5. Data re-scaling

As it can be shown above, each feature of a given dataset has different scales. It should be rescaled in order to be fed an algorithm for training. Otherwise, the model takes the feature with higher numerical value more important than others. Moreover, it is an important step to improve a model's accuracy.

For this task, it was considered a standardization methodology of rescaling. Its equation:

$$standardized\ value(k) = \frac{feature\ value\ (k) - average\ value}{variance}$$

where, *k* =1:90 (number of features), *average value* – arithmetic mean of all features, *variance* – variance of feature values from its average value.

Result:

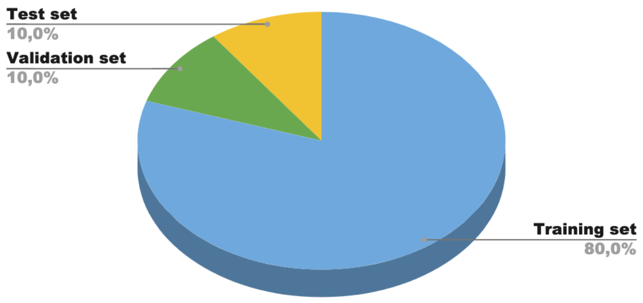
```
max value (before rescaling):  38183.73597
min value (before rescaling):  -11568.50627

max value (after rescaling):  39.58474921381809
min value (after rescaling):  39.58474921381809
```

6. **Splitting the dataset**

We need to shuffle data before splitting. No batching was used.

Result: Data was splitted into datasets for train (80%), validation (10%) and test (10%) respectively.



7. **Data transformation**

Pytorch does not support working directly with pandas datasets, so the values have to be transformed to Pytorch formats.

MODEL

The model has parameters which can be tuned in this task. There are:

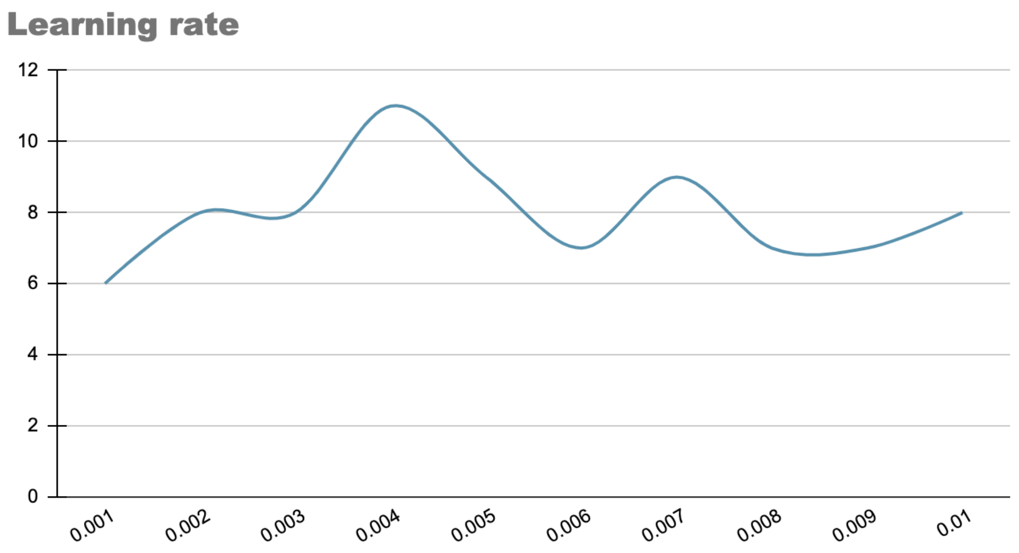
- *Architecture.*

In the paper, it was suggested to use 2-5 hidden layers. I decided to select 3 hidden layers. The depth of layers was recommended to be 5-10 layers. In my case, the model has three hidden layers of depth 100, 50 and 25 units respectively. It takes more time, but more units in layer handle nonlinear tasks better. We will use ReLU as an activation function in the hidden layers and Linear activation in the output layer as it was recommended.

- *Learning rate.*

Learning rate (lr) is necessary to stabilize the process of backpropagation. Its values should vary between 0.1 and 0.0001. The learning process will be better if the rate is low, but it affects the processing time.

I tried different lr for 1000 epochs, unfortunately it gave the random results as can be seen in the graph below where x axis is year difference, y axis is value of leaning rates. Moreover, the results also changeable for each session of Google Collab.



I decided to choose the middle value 0.005.

The following parameters were set according to recommendations in the paper:

- Optimizer = ‘adam’ (Adam algorithm)
- Loss function = ‘MSE’ (mean-squared error)
- Training epoch = 3000

Conclusion

In the dataset:

1. No categorical feature deviation and missing values.
2. There are outliers around **1%** of the dataset. It doesn’t affect the model performance.
3. Standardization decreased the boundaries of feature values to **1000 times.**

The parameters of model:

- Depth: 3 (100 – 50 – 25 units)
- Activation function: ReLU
- Learning rate: 0.005
- Number of epochs: 3000
- Optimizer: adam
- Loss function: mean-squared error

Model was tested several times with above mentioned parameters. It gave different results between 2 and 6 year difference. The best result is a **2 year** difference.

```
Ground truth: 2005.0 Prediction: 2003.466552734375
```