THE REPORT

Architecture

The model is a series of linear layers composed of different nodes to eventually output 8 digits (batch_size, output_size) and combine them into real date as the input is shape of (batch_size, 10 - features).

Each layer followed by a batch normalization, dropout and finally activation function.

(batch_size, 10) >> |

=> **32** node => **128** node => **512** node => **64** node => **32** node => **8** node =>

| >> (batch_size, 8)

The model is trained with nn.MSELoss function and nn.L1Loss function in different experiments, but nn.MSELoss function seems giving better results. Here we've used Adam optimizer along with momentum, adaptive learning rate and L2 regularization some runs switched on / off interchangeably.

The model takes [[day condition] [month condition] [leap year condition] [decade condition]] as inputs and outputs a predicted date then export them in 2 variants files ['output_file.txt', 'output_file_smote.txt']. The 2 output files will be in the format as [[day condition] [month condition] [leap year condition] [decade condition] [date]].

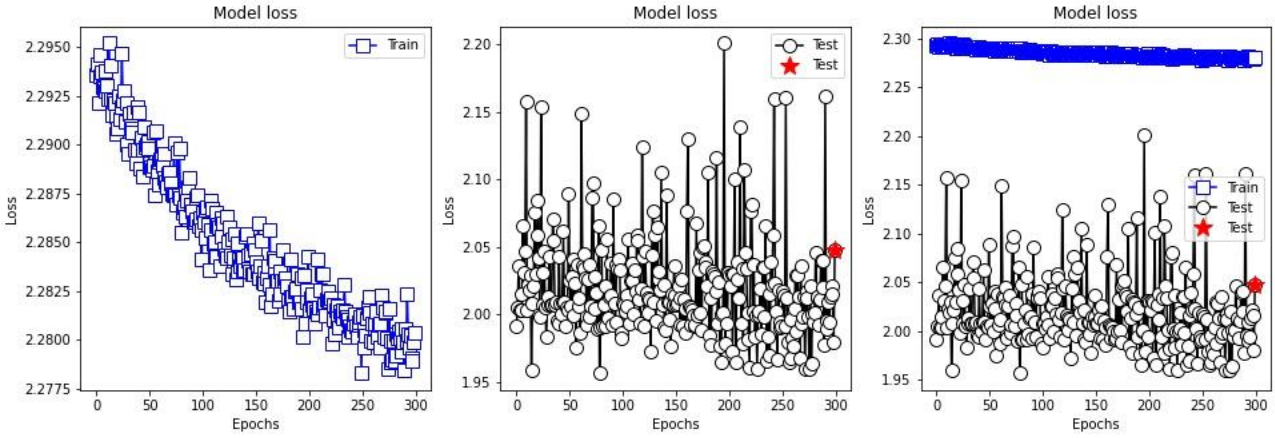
The prediction is 8 integer numbers each in range [0, 9]. Then convert them into date format.

Input features: ['day_scaled', 'day_sin', 'day_cos', 'month_scaled', 'month_sin', 'month_cos', 'leap_year', 'decade_scaled', 'decade_sin', 'decade_cos'].

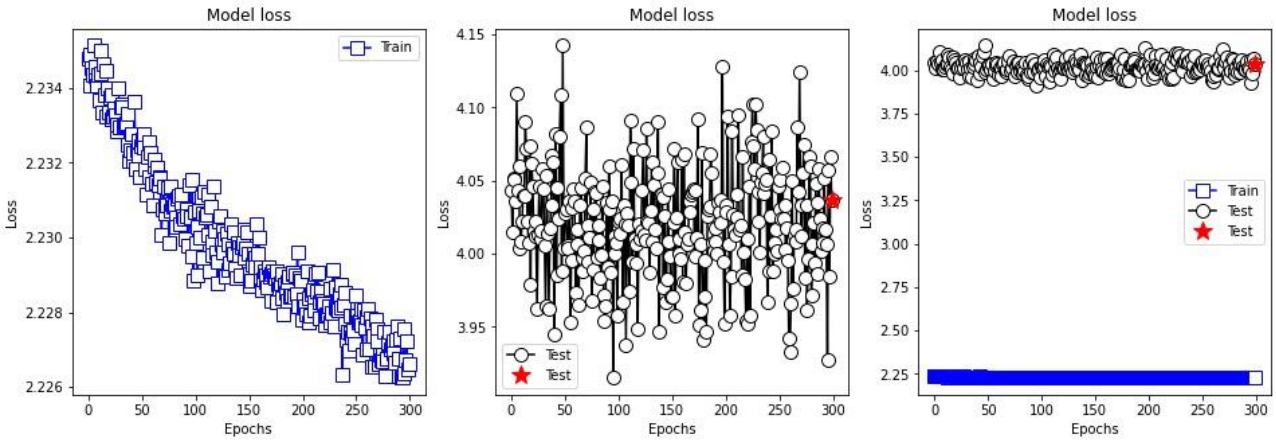
Target: ['date_int'].

Loss:

Loss from Model:-

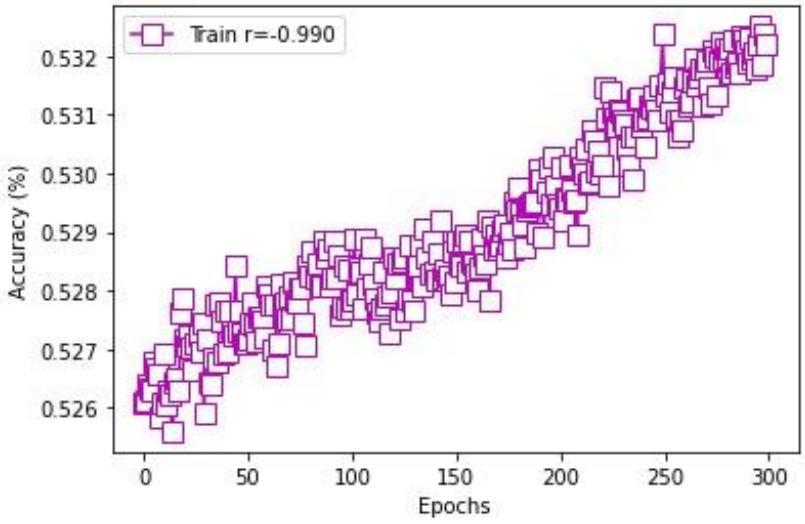


Loss from Model SMOTE:-

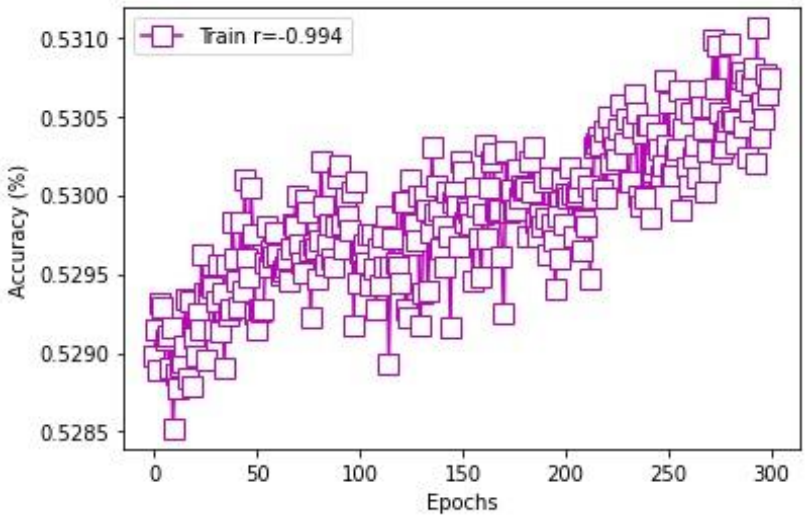


Accuracy:

Accuracy from Model:-



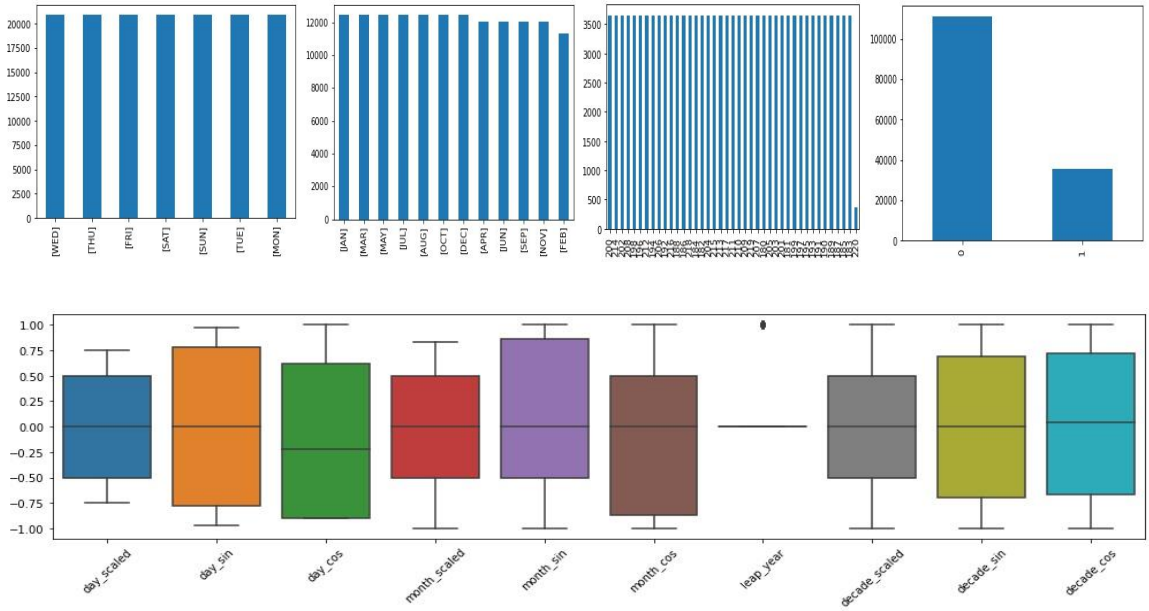
Accuracy from Model SMOTE:-



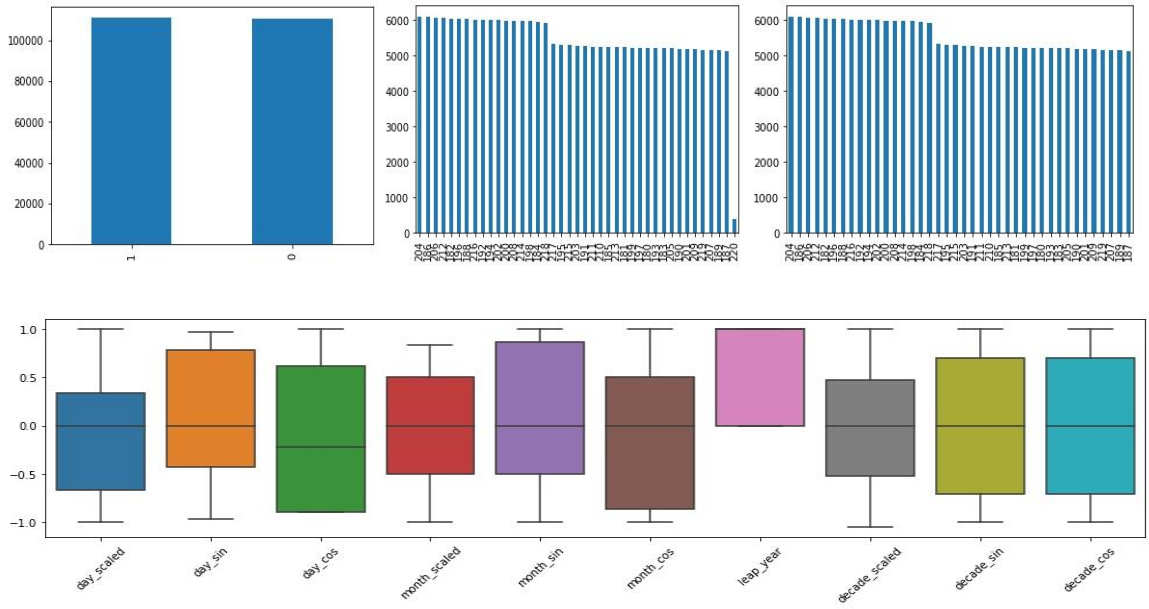
EDA and tokenization:

Using pandas to read and parse the data, firstly trying to infer date and then into numpy array 2d shape as target. Going through parsing input conditions into a side pandas dataframe then encode into number, scale them with scikit learn scaler and calculate

sine and cosine on them (since as a pair it’s hard to repeat ending as unique pairs).



It’s been used SMOTE technique to solve the imbalance data [‘leap_year’] column and training the model again using new balanced data.



your analysis of the outputs:

- it's quite impressive and unexpected to predict date in only 4 features and does quite well.

make sure to reflect on your choices and analysis briefly:

- I've tried different arcatures and techniques like normalization and scaling to come up with convenient results. There are mainly 2 variants of models. One output as predict a continues number then convert back to datetime object. The other - winner – model predicts a single numbers range [0, 9] then concatenate them into an integer then datetime object in `repo/model/8/MSE3/`

- I see that we can solve that problem as classification problem and I think it would give a much higher results. We can discuss that together.

Can we make the same conclusions you made, fast ?

- I think it could be. First thing first use GPU to run the experiment faster. Try to switch off L2 regularization (weight decay) and turn off momentum and adaptive learning rate.

Can we visually and logically validate that your results are ok ? Provide some output examples, maybe also provide examples where your model failed and your reflection on them.

- Might be by compare the predicted date and real date, though extract the 4 features from the predicted date and real features, compare the right predicted features with features of predicted date features individually one by one, see the correlation and covariance and use regression evaluation metrics to check out the model performance in total.

- For sure the model that trained on SMOTE data predicted higher date than exceeds the Range which unexpected e.g. '2305-11-15' which is far beyond out of bounds.