# Sunspots Forecast

Luis Pinto

January 21, 2021

**Abstract**

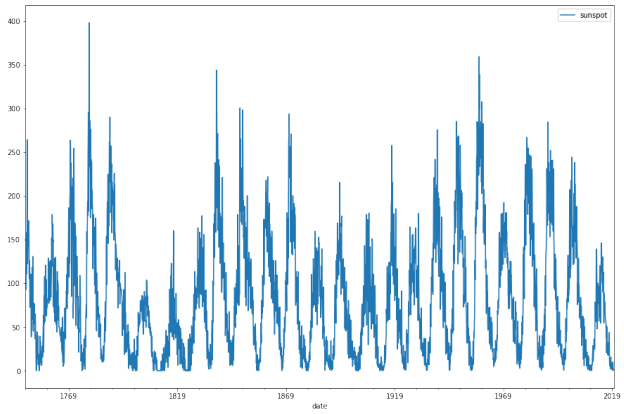
In this paper, two machine learning models consisting of LSTMs and DNNs will be used to predict sunspot numbers using data from 1749/01/01 to 2017/08/31. Hyperparameters are tuned in search of better mean absolute error (MAE). The best model uses the raw features as inputs to the model and gets an 18.3 MAE on the hold-out validation set.

## 1 Introduction

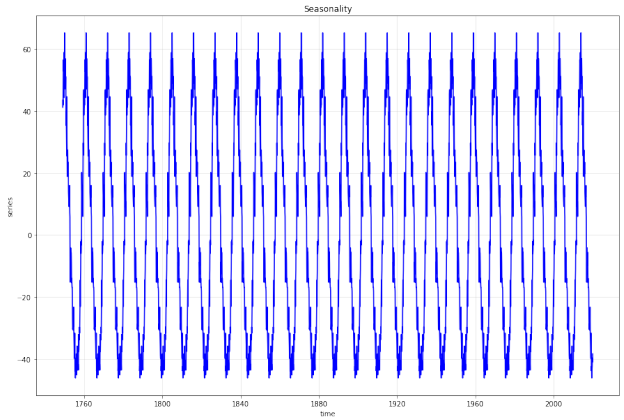
Sunspots are temporary phenomena on the Sun's photosphere that appear as spots darker than the surrounding areas. They are regions of reduced surface temperature caused by concentrations of magnetic field flux that inhibit convection. Sunspots usually appear in pairs of opposite magnetic polarity. Their number varies according to the approximately 11-year solar cycle. In this notebook, we attempt to model and predict the number of sunspots given historical data dating back to 1749. All experiments were performed using the TensorFlow deep learning platform.

## 2 Dataset

This paper will focus on a Kaggle dataset provided by the Solar Influences Data analysis Center (SIDC) - the solar physics research department of the Royal Observatory of Belgium. The data backs from 1749/01/01 to 2017/08/31. Since there is a clear cycle of 11 years, the training set will use twenty 11 year cycles (i.e. until 1969) and the validation set will use the rest.




(a) Sunspot numbers against time.




(b) Seasonality of the series.

# 3 Models

## 3.1 Model Architecture

In this project, we decided to use a sequential model consisting of two bi-directional LSTMs and two Dense layers, with a total of 84 261 parameters. The LSTMs have a 5% dropout and there is a 10% dropout between the two Dense layers to avoid overfitting. Figure 2 shows the architecture.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional (Bidirectional (None, None, 100)         20800
_________________________________________________________________
bidirectional_1 (Bidirection (None, None, 100)         60400
_________________________________________________________________
dense (Dense)                (None, None, 30)          3030
_________________________________________________________________
dropout (Dropout)            (None, None, 30)          0
_________________________________________________________________
dense_1 (Dense)              (None, None, 1)           31
=================================================================
Total params: 84,261
Trainable params: 84,261
Non-trainable params: 0
```

Figure 2: Model Architecture.

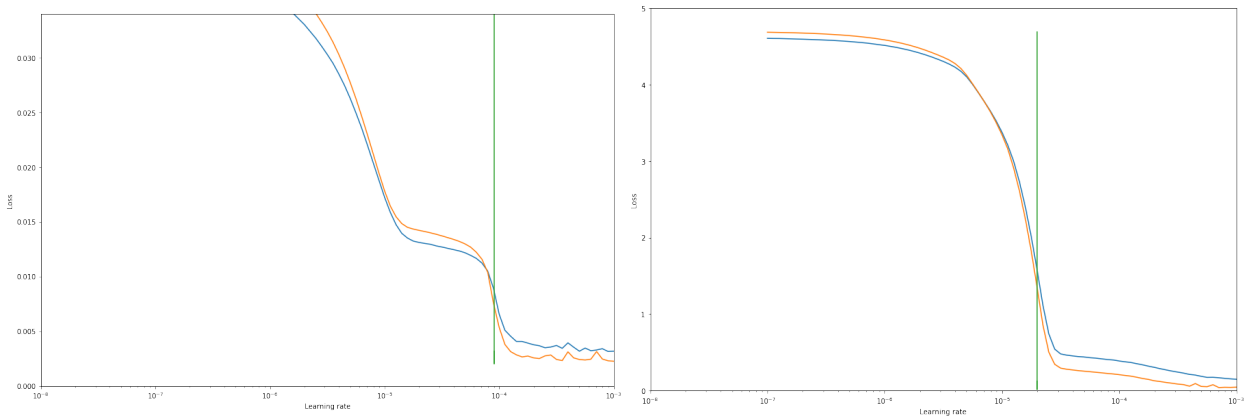
## 3.2 Features

This is where the two models differ, one model will use raw features (i.e. values recorded and stored by the SIDC) and the other model will try to remove the seasonality and use log transformation to decrease the variance.

## 3.3 Hyperparameters

In this study, the only hyperparameter that will be optimized is the learning rate and every other parameter will be kept constant throughout the analysis. Since we are dealing with a time series data, we need a window size to iterate over our series; it will be set to 50. Moreover, a batch size of 100 will be used to train. Also, the two models will be trained for 400 epochs.

The optimization of the learning rate will follow the guidelines of an study by Leslie N. Smith called Cyclical Learning Rates for Training Neural Networks. In this study, it was explained that the ideal learning rate should be the one that gives the highest loss gradient. This can be obtained changing the learning rate as a function of epoch and plotting it against the loss as seen in the figures below. The green lines indicate the approximate ideal learning rate used for each model.




(a) Learning rate against loss using raw features.

(b) Learning rate against loss using pre-processed features

# 4 Results

We trained our two models and compared their mean absolute error in Table 1. The two models use the same architecture but differ in the pre-processing. The learning rate was the only hyperparameter that was optimized for each model. As it can be seen in the figures below, the two algorithm do a good job at predicting the peaks and troughs and overall pattern. Nevertheless, by looking at the error metrics reported on Table 1, the model with raw features performed better with 19.934 training mean absolute error and 18.363 validation mean absolute error. This can be explained by the fact that neural networks are good at catching seasonality and other characteristics in data.

| **Model** | **Train MAE** | **Evaluation MAE** |
|---|---|---|
| Raw features | 19.934 | 18.363 |
| With pre-processing | 26.947 | 23.169 |

Table 1: Performance of variations of the model.

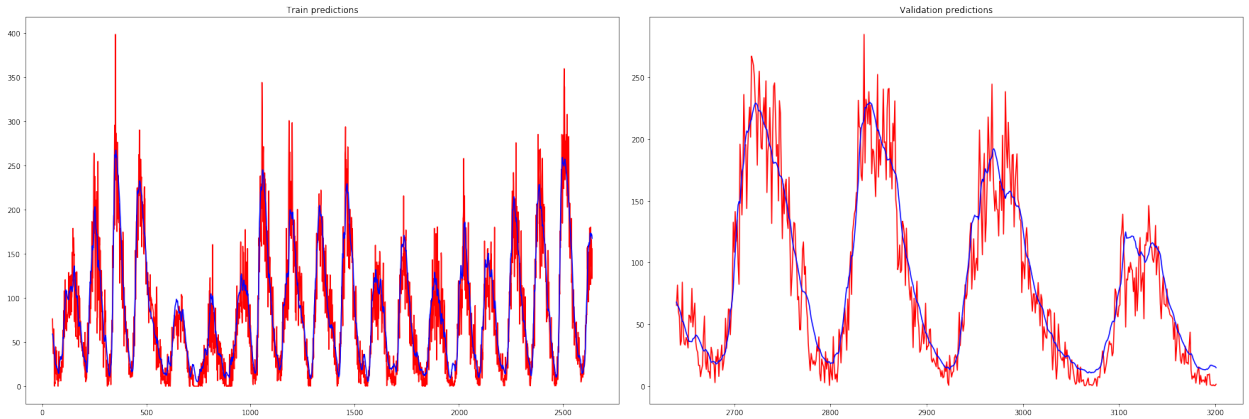



Figure 4: Original data in red and predictions in blue for the model using raw features. The x-axis was changed from real dates to time steps (1 day = 1 unit).

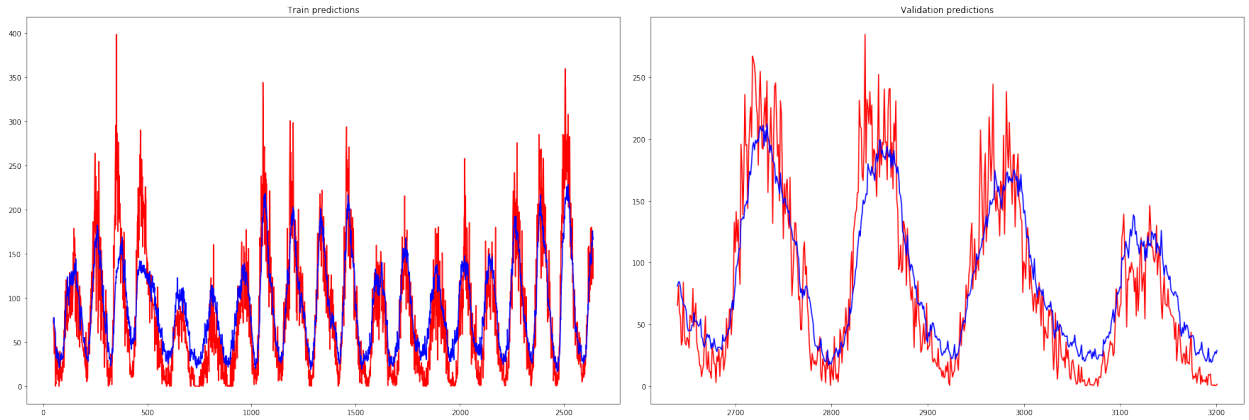



Figure 5: Original data in red and predictions in blue for the model using pre-processed features. The x-axis was changed from real dates to time steps (1 day = 1 unit).

# 5 Discussion and Conclusion

In this work, we present two variations of the same sequential neural network that use raw features and pre-processed features respectively. The learning rates of both models were optimized using a technique outlined by Leslie N. Smith in the paper Cyclical Learning Rates for Training Neural Networks. Every other hyperparameter was kept constant for both models. Both models perform well as seen in the figures and table but more works needs to be done to get better results.

To further improve the mean absolute error of our two models, experiments could be done using different model architectures. Moreover, the addition of recurrent dropout on the LSTM layers can be beneficial since this univariate model is prone to overfit. The optimization of training hyperparameters such as the batch size and the window size also need to be studied to further improve the error metrics.