Approaches Towards Prediction of Weather Forecasting using Machine Learning

---

Student Name: **Kalyan Reddy**

Tutor Name: **skldgjlsjglsdjg**

Module Number: **@2354345**

Table of Content

List of Figures

List of Tables

1 Introduction

1.1 Overview

Nowadays prediction of weather forecasts and analysing is the main task which is faced today. A lot of systems and equipment are already developed for predicting the weather forecast which are sophisticated in its own way. However, these systems can be employed with machine learning algorithms to enhance it towards the computer algorithm implication to improvise the prediction standards. According to Krishnaveni and Padma (2021) the weather forecasting is an emerging domain where the condition of the weather will be predicted for a particular location and particular time. The author had compared the SPRINT algorithm with the existing method naive Bayes algorithm. From this, it is observed that the proposed method provides better efficient accuracy when compared with the existing method. In this work, the author proposed The SPRINT algorithm which mainly works on the principle of decision tree algorithm. Here the present work is carried out by using the climate dataset taken from Kaggle with 1460 rows and 6 columns. However, based upon the features of the dataset the model predicts the weather forecast decision tree and its variant algorithms.

1.5 Research Methodology

Before proposing the algorithm, a lot of research work is done on the different ML approaches. The main method considered here is machine learning for forecasting purposes. Initially collecting the dataset from open-source platform Kaggle and performing some pre-processing, Exploratory data analysis on the obtained dataset. After processing the data and will be split into training and testing sets then the training data is applied to the ML algorithms. Finally, a Decision Tree, Random Forest, and Xgboost algorithms are used. Based on the data the methods considered are:

- Data cleaning
- Data processing
- Feature selection
- Model evaluation through metrics.

1.6 Organization of the Dissertation

In this dissertation thesis, the Chapter 1 explore the importance, role and goal of this work is explained with suitable aim and objectives. To support the research work with adequate materials and previous methods applied in the same context of work is explored by review the existing literature in Chapter 2. The suitable methodology for the present work to predict the weather forecast, a proper approach has been recommended in Chapter 3 with proper justification. Chapter 4 explains the summary of results obtained by implementing the code and allows to conduct an experimental analysis to conclude the overall work in Chapter 5.

2 Literature Review

2.1 Overview

Weather forecasting is the process of predicting the weather conditions for the future. By using the data of temperature, humidity, and pressure using different sensors to predict the weather conditions. In this paper, machine learning techniques are used for data analysis and prediction. Machine Learning doesn't require the present data it can predict by using the past data to predict the future data says Parmar *et al.* (2018).

According to Singh *et al.* (2019), Random Forest classification is a machine learning algorithm that used a learning method in which one or two machine learning methods are combined to form a single learning method. It operates by using multiple decision trees while training the dataset. Cho *et al.* (2020) says that most commonly used method for weather forecasting is Artificial Neural Networks (ANN), which provides higher accuracy than some other techniques like Support Vector Regression (SVR), and Random Forest (RF). According to Scher and Messori.2018, every weather forecast is a certain degree and it has been recognized for many applications. Artificial Neural Networks (ANN) is used in weather forecasting for decades. Recently deep convolutional networks a subclass of artificial neural networks has shown very high skill in image recognition, so which is applied to detect extreme weather conditions using the training dataset containing the image files. Convolutional Neural Networks (CNN) is also applied for atmospheric states to forecast. However, the weather conditions change very fast so it is difficult to propose an algorithm for weather forecasting (Fathi *et al.*2021). A computational process is required with high-tech equipment. The distribution of data related to weather mainly the temperature distribution can be approximated by a Gaussian distribution, this gives a limitation for other weather conditions such as wind speed. The author Sonderby *et al.*2020 suggests that there are many random forests generated from classification and regression trees. The decision trees were obtained by splitting the training data into two groups according to predictors chosen for minimizing the variance of the response variable in the resulting groups. A generalized version of random forest-based quantile based on theoretical considerations has been tested but did not result in improved forecast performance.

A. Machine Learning – Background

Machine Learning has played a main role in physics for over a decade. it tells whether a given set of particles is associated with a bottom quark, which is four times heavier than a proton

(Schwartz, 2021). Machine learning is used in physics for several years the traditional machine learning is replaced with modern machine learning. the most valuable in the data is a network that is to be free to find (Schwartz, 2021). The Pile Up Mitigation with Machine Learning (PUMML) builds to find the intensity of the pixel of an image. For PUMML three images are constructed one is for charged particles from the primary collision and second from secondary collisions points and the third is from neutral particles. Machine Learning has developed an entirely different purpose for particle physics applications.

The author Benning *et al.* (2022) states that intelligent agents are described by the term

- The work also progressed towards utilising the known abilities of hyperparameter tuning on the machine learning algorithms (DeCastro-García *et al.* 2019), the decision tree algorithm was optimised using the Grid search CV method of tuning. The tuning method selection was explained and justified by showing that the optimisation has caused an increase in accuracy which was 79% after tuning and 76% before tuning. The

aim here was to show the effectiveness of tuning, but however, through the results it was made clear that XGBoost shows better results.

- Hence, these were the identified potential methods an ideas form eth literature that made author to carry out the code and research work collectively. However, the work was aimed to make predictions through XGBoost and use hyper-parameter tuning to justify the enhancements that it can make when implemented. Finally, the work was narrowed down through an comparative analysis and concluding the discussions on the best performed model based on the accuracy. Moreover, other important metrics like precision, recall and confusion matrix are evaluated to make better discussions and justify the model used on the data selected. However, the study finds it future scope and implications through adding the over sampling technique and increase the number of instances and then make predictions.

3 Methodology

3.1 Overview

Here in this chapter the algorithms used as a part of methodology is discussed and reviewed with working and summary. The Decision tree, Random Forest and XGBoost algorithm is used here to implement the results and use dataset. these algorithms are compared based on accuracy. The below sections are elaborated to discussed to understand the working and implementation and to be used in the work.

3.2 Working of Decision Tree and Random Forest models in Machine learning

A Decision Tree is a supervised learning technique that is used for both classification and

3.5 Summary

According to Konstantinov and Utkin (2021), the stacked ensemble typically consists of base models and meta models. The base models, also known as Level-0 models, fit the training data and predict the new incoming data, whereas, the meta models, also known as Level-1 model, fits the model from base models' prediction and learns how to obtain the best predictions. A stacked ensemble involves steps such as implementing K-Fold cross validation to separate the data into K different folds and then keeping one of the folds separate and then train multiple independent base models on the other folds. Upon training the base models, it is modelled upon the fold which was held out earlier. This process of separating one-fold and training the rest on the base models is done for all the K folds. Then the predictions made by the base models are then fed into a meta model to the obtain the final output. Authors Smeden *et al.* (2019) and Caigny *et al.* (2018) say that the meta model typically contains models such as Linear regression and Logistic Regression, that used for regression and classification tasks respectively. The author is also convinced that ensemble techniques because of the special methods involved can handle the fluctuations and provide a better performance in comparison with the ordinary standalone machine learning algorithms.

4 Results and Discussions

Following a discussion of the work's approach, the data set is subjected to decision tree, random forest, and XG boost analysis. The data set used in this analysis was obtained from the open-source kaggle. The data set is mounted to the colab notebook so that it may be accessed. The Colab notebook comes with python3 as a compute backend, 12 GB of free RAM, and 187 GB of storage space. As a result, it's simple to utilise virtually. The notebook style is identical to that of the Jupyter notebook. With only an internet connection, the Notebook may be browsed, downloaded, saved, and linked via the drive.

4.1 Overview on the dataset

There are 1461 instances and 6 columns in the data set. Date, precipitation, temperature maximum, temperature minimum, wind, and lastly the independent variable weather are the columns. There are five unique values in the independent variable weather. There are rain, fog, drizzle and snow.

```
data.head()
```

	date	precipitation	temp_max	temp_min	wind	weather
0	2012-01-01	0.0	12.8	5.0	4.7	drizzle
1	2012-01-02	10.9	10.6	2.8	4.5	rain
2	2012-01-03	0.8	11.7	7.2	2.3	rain
3	2012-01-04	20.3	12.2	5.6	4.7	rain
4	2012-01-05	1.3	8.9	2.8	6.1	rain

Figure 4.1 Head of the dataset

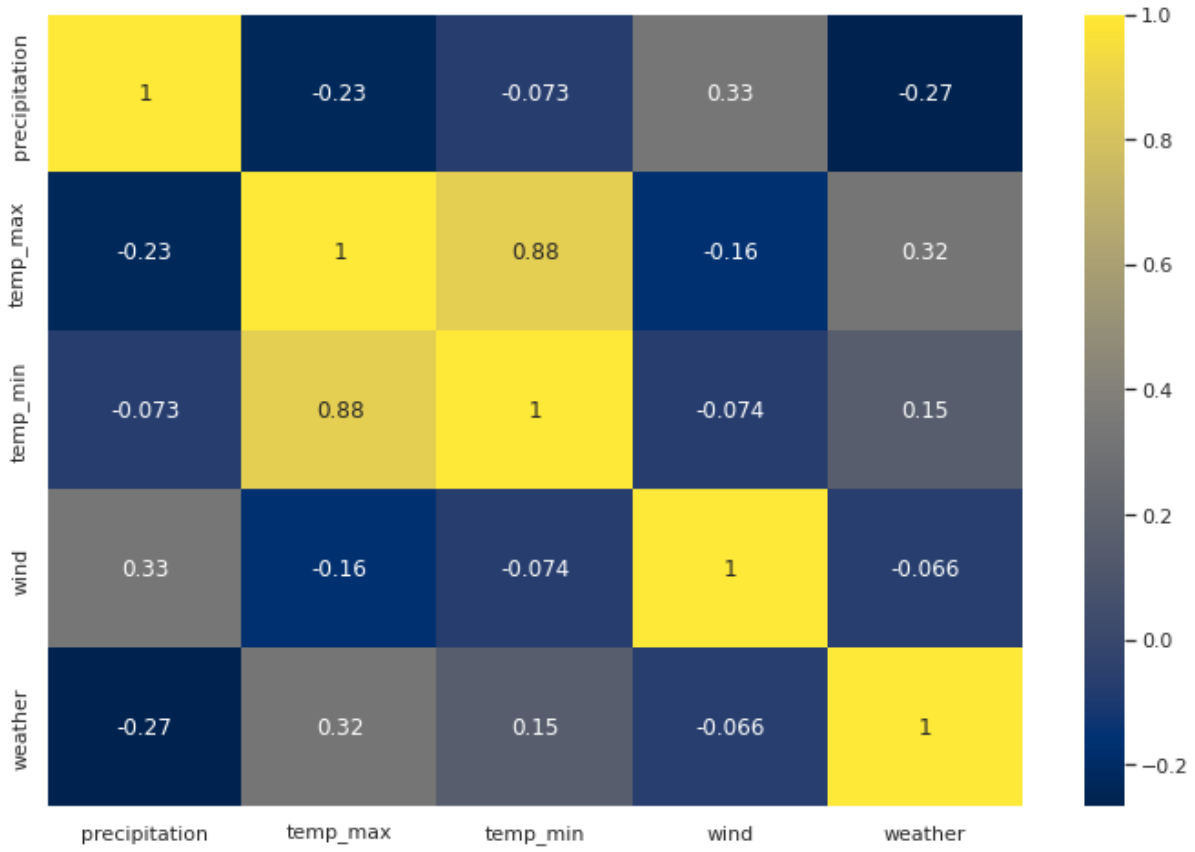
```
rain       641
sun        640
fog        101
drizzle     53
snow        26
Name: weather, dtype: int64
```

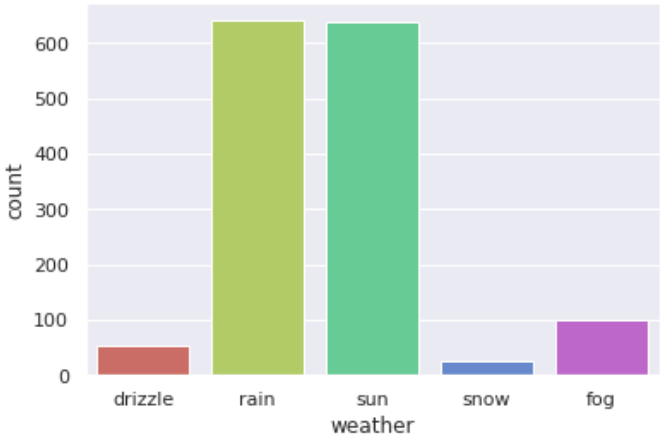
```
data.info()
```

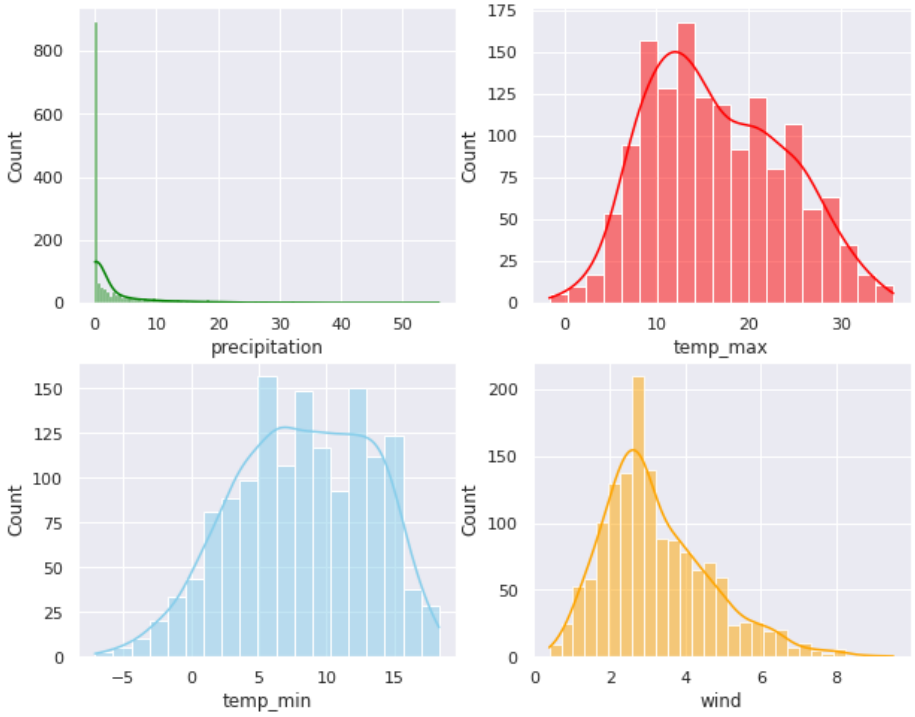
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1461 entries, 0 to 1460
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   precipitation  1461 non-null   float64
 1   temp_max       1461 non-null   float64
 2   temp_min       1461 non-null   float64
 3   wind           1461 non-null   float64
 4   weather        1461 non-null   object 
dtypes: float64(4), object(1)
memory usage: 57.2+ KB
```

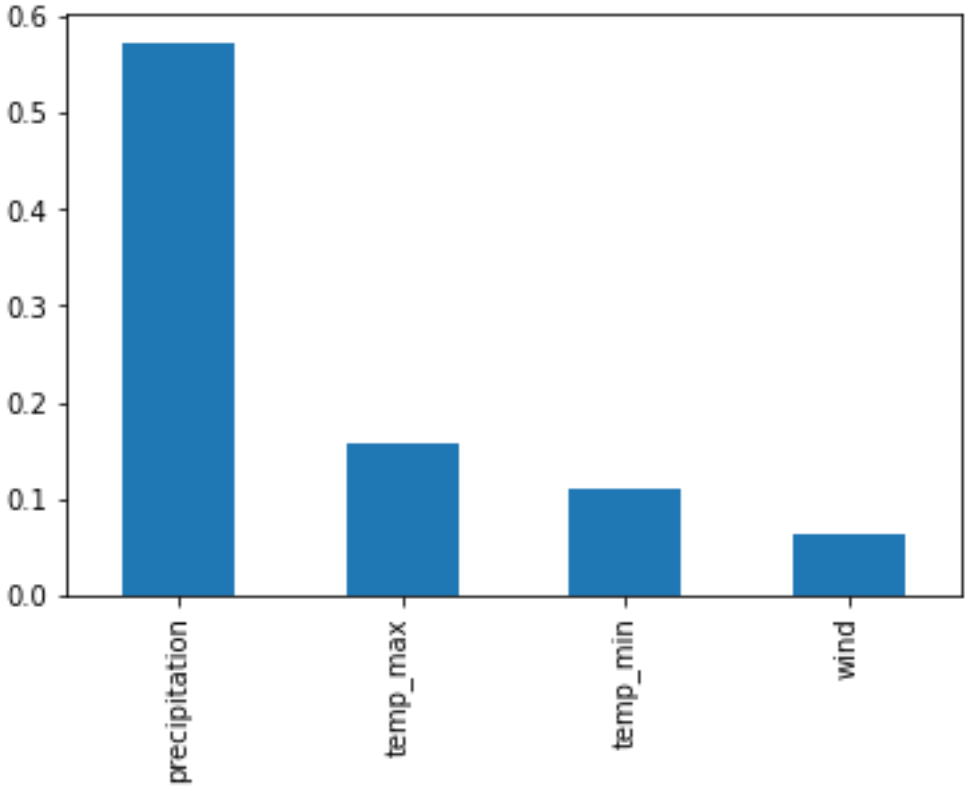
```
data.isnull().sum()
```

```
precipitation    0
temp_max         0
temp_min         0
wind             0
weather          0
dtype: int64
```


precipitation
temp_max
temp_min
wind
weather
1
-0.23
-0.073
0.33
-0.27
-0.23
1
0.88
-0.16
0.32
-0.073
0.88
1
-0.074
0.15
0.33
-0.16
-0.074
1
-0.066
-0.27
0.32
0.15
-0.066
1
precipitation
temp_max
temp_min
wind
weather
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


600
500
400
300
200
100
0
count
drizzle
rain
sun
snow
fog
weather


800
600
400
200
0
Count
0
10
20
30
40
50
precipitation
175
150
125
100
75
50
25
0
Count
0
10
20
30
temp_max
150
125
100
75
50
25
0
Count
−5
0
5
10
15
temp_min
200
150
100
50
0
Count
0
2
4
6
8
wind


0.6
0.5
0.4
0.3
0.2
0.1
0.0
precipitation
temp_max
temp_min
wind

```
features
```

```
precipitation    0.574609
temp_max         0.166616
temp_min         0.064076
wind             0.076185
dtype: float64
```

```
Classification report -
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        15
           1       0.24      0.30      0.26        30
           2       0.90      0.93      0.92       192
           3       0.22      0.40      0.29         5
           4       0.83      0.74      0.78       197

    accuracy                           0.77       439
   macro avg       0.44      0.47      0.45       439
weighted avg       0.79      0.77      0.78       439
```

```
cm
```

```
array([[  0,   3,   1,   0,  11],
       [  2,   9,   5,   0,  14],
       [  1,   1, 179,   7,   4],
       [  0,   0,   3,   2,   0],
       [ 12,  21,   9,   0, 155]])
```

	precision	recall	f1-score	support
0	0.93	0.93	0.93	15
1	1.00	0.97	0.98	30
2	1.00	0.99	1.00	192
3	1.00	1.00	1.00	5
4	0.99	1.00	0.99	197
accuracy			0.99	439
macro avg	0.98	0.98	0.98	439
weighted avg	0.99	0.99	0.99	439

	precision	recall	f1-score	support
0	1.00	0.07	0.12	15
1	1.00	0.03	0.06	30
2	0.99	0.97	0.98	192
3	0.80	0.80	0.80	5
4	0.80	1.00	0.89	197
accuracy			0.89	439
macro avg	0.92	0.57	0.57	439
weighted avg	0.91	0.89	0.85	439

```
Model_Tuning.best_params_
```

```
{'max_depth': 70,
 'max_features': 'log2',
 'max_leaf_nodes': 50,
 'min_samples_leaf': 30,
 'splitter': 'best'}
```

Figure 4.14 Parameters from the grid search CV

Hence from the Figure 4.14, the overall best parameters obtained are 70 as the maximum depth, log2 functionality in defining the features, 50 leaf nodes and 30 samples per leaf with best functionality as the splitter. However, it can be seen that there are only 4 features, and when the model was implemented with the parameters suggested by the grid search it still showed less accuracy. However, when the parameter the maximum features was changed to 3 and maximum depth was put 60. The model shows a better accuracy. However, the 3 features were selected from the Fig. 4.8 and Fig. 4.9. It can be concluded that the model after hyper parameter tuning and utilising the feature selection have improved with the respect to the accuracy. Apart from that the model was evaluated with the whole data to be trained on and predictions were made on the test set. The accuracy of the model is 98%. The classification report of the model is seen in the Fig. 4.15.

	precision	recall	f1-score	support
0	1.00	0.93	0.97	15
1	0.97	0.97	0.97	30
2	0.98	0.99	0.99	192
3	1.00	1.00	1.00	5
4	1.00	0.99	0.99	197
accuracy			0.99	439
macro avg	0.99	0.98	0.98	439
weighted avg	0.99	0.99	0.99	439

Figure 4.15 Classification report of tuned Decision tree

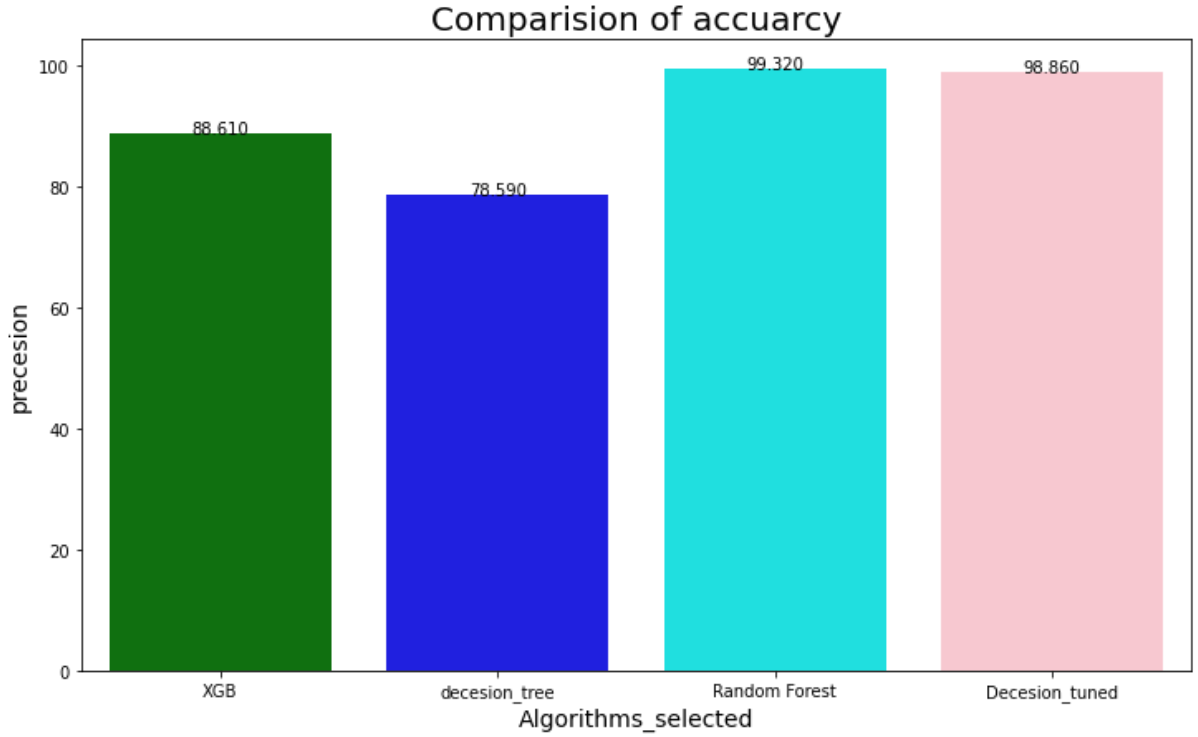
Hence, the enhancements can be seen when the model was tuned as well as when the training set was increased. The precision values from the previous results of the decision tree has been improved.

5 Conclusions and Recommendations

5.1 Conclusions

The present work is python and some of the important specifications found in the study based on code and methods are:

- The data set is exposed to tree structure model decision tree, XG boost and random forest algorithms after a summary of the work's strategy.
- The data set for this research came from the accessible source Kaggle platform.
- The data set is stored on the colab notebook and may be viewed from there. The Colab notebook has a computing backend of python 3, 12 Gigabytes of available Memory, and 187 Gb internal memory.
- As a consequence, using it virtually is straightforward. The notebook is designed in the same way as the Jupyter notebook. The Notebook may be accessed, downloaded, stored, and linked via the disc with simply an internet connection.


Comparision of accuarcy
precesion
Algorithms_selected
XGB
decesion_tree
Random Forest
Decesion_tuned
88.610
78.590
99.320
98.860
0
20
40
60
80
100

is a lack of data, machine learning algorithms discover their own unique strategies to fulfil the predicted outcomes. As a result, these were the recognised prospective approaches and ideas from the literature that prompted the author to collaborate on the coding and research work. However, the goal of the project was to utilise XGBoost to create predictions and employ hyper-parameter tweaking to explain the improvements that it can make once deployed.

Eventually, the work was distilled down through a comparison study, with the talks focusing on the model that performed best in terms of accuracy. Other essential metrics including as accuracy, recall, and the confusion matrix are also analysed in order to improve conversations and validate the model employed on the data. However, by using the Over sampling approach and increasing the number of cases, the study discovers its future breadth and ramifications, allowing it to make predictions.

References

Ahmad, T. and Aziz, M.N., 2019. Data preprocessing and feature selection for machine learning intrusion detection systems. ICIC Express Lett, 13(2), pp.93-101.

Amr, T., 2020. *Hands-On Machine Learning with scikit-learn and Scientific Python Toolkits: A practical guide to implementing supervised and unsupervised machine learning algorithms in Python*. Packt Publishing Ltd.

Benning, L., Peintner, A. and Peintner, L., 2022. Advances in and the Applicability of Machine Learning-Based Screening and Early Detection Approaches for Cancer: A Primer.

Victoria, A.H. and Maragatham, G., 2021. Automatic tuning of hyperparameters using Bayesian optimization. Evolving Systems, 12(1), pp.217-223.

Visser, L., AlSkaif, T. and Van Sark, W., 2019, June. Benchmark analysis of day-ahead solar power forecasting techniques using weather predictions. In *2019 IEEE 46th Photovoltaic Specialists Conference (PVSC)* (pp. 2111-2116). IEEE.

Wang, B., Lu, J., Yan, Z., Luo, H., Li, T., Zheng, Y. and Zhang, G., 2019, July. Deep uncertainty quantification: A machine learning approach for weather forecasting. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 2087-2095).

Wen, Z., Shi, J., He, B., Chen, J., Ramamohanarao, K. and Li, Q., 2019. Exploiting GPUs for efficient gradient boosting decision tree training. IEEE Transactions on Parallel and Distributed Systems, 30(12), pp.2706-2717.

Xuan, S., Liu, G., Li, Z., Zheng, L., Wang, S. and Jiang, C., 2018, March. Random forest for credit card fraud detection. In 2018 IEEE 15th international conference on networking, sensing and control (ICNSC) (pp. 1-6). IEEE.

Yang, L. and Shami, A., 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, *415*, pp.295-316.

Yang, L. and Shami, A., 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing, 415, pp.295-316.

Yu, T. and Zhu, H., 2020. Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689*

Zahedi, L., Mohammadi, F.G., Rezapour, S., Ohland, M.W. and Amini, M.H., 2021. Search algorithms for automated hyper-parameter tuning. arXiv preprint arXiv:2104.14677.

Zhang, L. and Wang, Y., 2021. Transformer Fault Diagnosis Based on Stacking-Ensemble Meta-Algorithms. Advances in Computer, Signals and Systems, 5(1), pp.42-47.