Anomalous Climate Pattern Detection in ECONet - CSC522 Group P09

Vignesh Muthukumar
vmuthuk3
NCSU
vmuthuk3@ncsu.edu

Atharva Risbud
ayrisbud
NCSU
ayrisbud@ncsu.edu

Suyog Dharmadhikari
sdharma2
NCSU
sdharma2@ncsu.edu

ABSTRACT

Maintaining a healthy and resilient mesoscale environmental observing network necessitates the process of verifying quality observations. The ECONet dataset is a quality control dataset that is used to identify and correct faulty sensors and measurements. The QC dataset is imbalanced, with low erroneous data and a majority of data that has been correctly flagged. This study presents with a novel approach of incorporating season-specific data and resampling the provided dataset to classify the erroneous data points. Since the distribution of erroneous points are very less, they pose a challenge of being considered as an outlier which is handled effectivelt in this study to correctly classify such imbalanced dataset.

KEYWORDS

Anomaly Detection, Weather Data, Climate Data, Imbalanced Classification, Random Forest, Under sampling

1 INTRODUCTION

The North Carolina Environment Climate Observing Network is a sensor network that collects data from 43 weather stations to assess 23 weather measure values. The EcoNET dataset is currently being used to analyze and enhance severe weather forecasting, energy planning, and natural resource management. An automatic Quality assurance (QA) system flags the measurements in the dataset, and they are manually inspected for mistakes. When compared to proper data, the QC dataset is unbalanced and contains very little incorrect data. Because only a few measures are erroneously forecasted, manually assessing the measures with false flags is time demanding and a waste of human resources. The problem statement focuses on efficiently detecting the erroneous data points and flagging them accordingly. A robust classification based algorithm is experimented to work on the time series based imbalanced dataset. Since this is a weather dataset, there is a strong dependency of how different weather measure changes with time. This poses a novel challenge of wrapping the measure values across each weather station over time periods to generate a predictive model. Through this approach, it is expected to save human efforts and time and make sure to provide a high quality data for further predictions.

2 RELATED WORK

There has been significant work done on performing imbalanced class classification. The authors of [2] specifies about how spacial and temporal analysis would be helpful in detecting anomalies in weather data. This serves as an important motivation to consider the temporal aspect in our work. To extend on the temporal nature, the studies performed by [3] explains about using ARIMA models for improving the predictions. The paper [5] defines about usage of XGBoost for performing imbalanced class classification and the work in [4] explains about the rational of using Random Forest for higher dimensional dataset. Both being tree based models have performed significantly well on larger dataset with sparse features. This propelled us to experiment these methods in our implementation. Another novel method for detecting anomalies in climate data using **Stacked and Densely Connected Long Short-Term Memory Model** suggested by [7] will be experimented as LSTM has been shown to increase anomaly detection.

3 DATASET PREPARATION

3.1 Dataset

The dataset used for studying this project is the North Carolina Environment Climate Observing Network known as the ECONet Dataset obtained from The State Climate Office of North Carolina (SCO). The Environment and Climate Observing Network (ECONet) dataset is provided by The State Climate Office of North Carolina (SCO). The dataset contains data from 43 distinct research weather stations located around North Carolina. The study has been provided with training and testing dataset individually. The training dataset consists of 3 weeks of data from each month and testing data consists of data of last week of every month. The training dataset is highly imbalanced with 6,358,102 correctly marked in the QC checks and only 235,172 being anomalous. The test data set contains 1,856,106 entries. The features of the training and test dataset is defined as follows:

- **Station**: The ID of research weather station where the readings were taken.
- **Ob**: It is the timestamp i.e the date and time at which the readings were taken.
- **Value**: The value of the reading in the respective units depending on the value of the measure attribute.
- **Measure**: It is attribute that tells us about different types of parameters measured with the accompanying sensor(s) that collect(s) the data. There are a total of **16** measure values in the training dataset and **20** value in the test dataset. Some of the values of measures include wind speed, soil moisture, temperature, precipitation, atmospheric pressure etc.
- **Target**: Values assigned by human reviewers, if it is erroneous it is True or else it is False.
- **R_flag**: It is a range check flag which uses climatology to test the validity of the readings. If the flag is 0 then it is considered to be passed with highest level. The flag is 4 if it fails to tests the validity. The flag values 1,2,3 are based on PAR to SR ratio.

- **I_flag**: It is a *intersensor check* flag. Basically, it is a parameter specific check for radiation, precipitation and wind speed. This flag takes values of 0,2 and 4.
- **Z_flag**: It is *trend check* flag which compares current values to longer period of time that the values being reported is valid given the current state. This flag takes values 0,2 and 4
- **B_flag**: It is a*buddy check* flag which compares the readings of each stations with the neighboring stations. It takes values from 0 which means the check is passed and 4 which means the check is failed.

The Target attribute accepts binary values (TRUE / FALSE), with TRUE indicating ANOMALOUS data entry and FALSE indicating NORMAL / ACCURATE data entry. Experts validated the instances of the training data and labeled them suitably.

3.2 Data Preparation

The training dataset is initially checked for occurrences of NaN or empty characters across all the attributes. This is very important to analyse because in such a vast dataset that is dependent on time, having null values can themselves can hamper the distribution and the meaning of the data. Hence, as an initial step all the null values are cleaned from the dataset.

Then we performed statistical methods like Inter-quartile Range to understand how each measure value has been distributed. The minimum and maximum range is set as 25% and 75% to clearly distinguish the anomalous points that serve as outliers in this distribution. To mention in detail about a comparison of two measures, we studied how the value of *temp_wxt* changed over time. It was evident that during the time periods of hotter months of the year, the temperature value was by itself high where as it was low during the colder period of months. Similarly, the soil moisture and leafwetness were lesser during hotter months and greater during the humid months. This gave us an understanding about seasonal impact on measure which led us to divide the dataset based on seasons. Similarly there were many stations that were geographically closer to each other and a few flags had relationship between the readings of sensors between the stations.

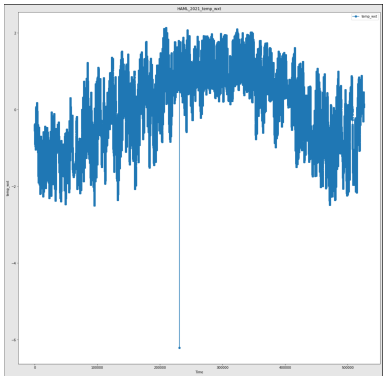

Figure 1: Temperature metric over the entire time period for Weather Station HAML

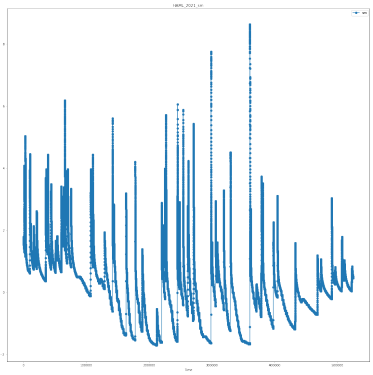

Figure 2: Soil Moisture metric over the entire time period for Weather Station HAML

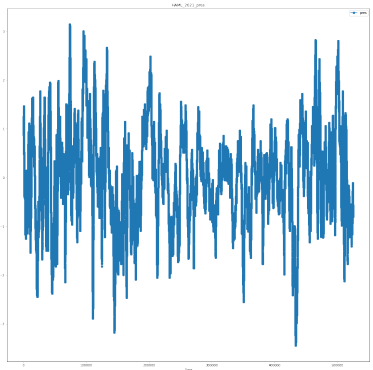

Figure 3: Pressure metric over the entire time period for Weather Station HAML

Once smaller chunks of data was obtained, we examined the correlation between the measure attributes using *Pearson's Correleation Coefficient*. This gives a clear picture about collinearlity of attributes and how they can be combined together.

The data is based on time series and it is critical to evaluate the impact of time on the overall performance. The timestamp appears as a string in this dataset, from which the actual time is retrieved and the time delta is computed and added as a new feature. This displays the influence that has built over time in a rolling time span.

The data was highly imbalanced which led to experiment with resampling methods. SMOTE was used to perform oversampling and it caused the model to overfit. Hence this approach was dropped. By performing undersampling using TomekLinks a great deal of majority data points were removed along with maintaining the proportional distribution.

Then, the categorical values in the dataset was encoded using Dummy Encoding. This is better than Categorical encoding and very similar to one hot encoding as it does not induce any bias based on the numerical weightage given to the category and each category is at a distance of *sqrt(2)* from the other. Following this, the measure values were normalized using *Z-Score Normalization*, *Min-Max Normalization* and *Robust Scaler Normalization*. The data

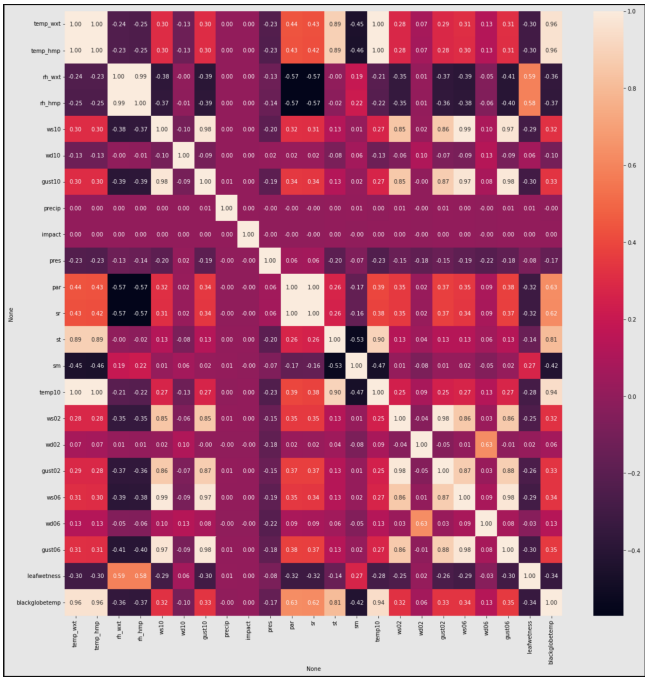

Figure 4: Correlation Matrix over all measure attributes for Weather station HAML over entire time period

was highly imbalanced which led to experiment with resampling methods. The SMOTE [1] algorithm was implemented to oversample the minority classes and the distribution then fared to have both TRUE and FALSE classes as 6,358,102. This drastically increased the size of the dataset. On the other hand, undersampling method like TOMEKLinks [6] proportionally undersampled the majority classes to roughly eliminate 5% of the datapoints without affecting the distribution boundary of the dataset. The timestamp which is defined in the *Ob* value is encoded to get as the total minutes referring in a particular day.

4 RESEARCH QUESTION

Some of the hypothesis made about the dataset preparation and methodology approaches are :

- RQ1) Can there be any pattern found in the data for effective regrouping?
- RQ2) Can time delta values extracted from the time stamps define more about the time dependency of the dataset?
- RQ3) Can oversampling and undersampling strategies resample the data with expected distributions?
- RQ4) Can XGboost and Random Forest classifiers performs better than other classifiers like Logistic Regression and K-NN?

5 METHODOLOGY

We design our approach for implementation of the classification algorithm on top of the season wise divided dataset. Each of the dataset is loaded and the train and validation sample is divided using *train_test_split* method with a division of 80% of training data and 20% of test data. The split was done with a statified sample technique based on the class labels to ensure proportional distribution of class labels over training and test dataset.

The substantial work in [5] highlights the use of Extreme Gradient Boosting (XGBoost) with integrated weighted and focused loss to address class imbalance, which inspired us to use XGBoost in our study. We performed GridSearchCV on hyper paramters such as **eta** amd **max_depth** with a cross validation count of 3 and the scoring metric against *aucpr*. The *eta* defines the step size shrinkage which prevents overfitting. After each boosting step, we can directly get the weights of new features, and eta shrinks the feature weights to make the boosting process more conservative. *max_depth* corresponds to the maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit. The different values of hyper parameters tuned are param_grid = 'eta':[0.01, 0.05, 0.07, 0.1, 0.15, 0.2, 0.25], 'max_depth' : [10, 30, 50, 75]. Being a light weight multi threaded algorithm, XGBoost was really effective in generating models for multiple season-wise values quickly and also the tree based ensemble models were packed with compressed weight distribution.

Another method that was implemented in this research was the Random Forest Classifier. As the name implies, a random forest is made up of a huge number of individual decision trees that work together as an ensemble. Each tree in the random forest produces a class prediction, and the class with the most votes becomes the prediction of our model. The low correlation between models is the key where uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. While some trees may be incorrect, many others will be correct, allowing the trees to move in the correct direction as a group. Some of the important hyper parameters tuned for Random Forest Classifier are *n_estimators*, *max_features*, *max_depth*, *min_samples_split*. *min_samples_leaf*, *bootstrap*, *criterion*. param_grid = 'n_estimators': [400, 800, 1200], 'max_features': ['auto'], 'max_depth': [50, 75, 100, 150, None], 'min_samples_split': [2,5,10], 'min_samples_leaf': [1,2,4], 'bootstrap': [True, False], 'criterion': ['gini', 'entropy']

Once both the models with the best set of hyper parameters were run for 6 season wise generated files, the best model was chosen based on the highest value of F-1 score that they produced on the vaidation set. The weights of the models were saved using the *pickle* method supported by Scikit learn library.

On the test dataset, the categorical measures were also equivalently encoded and the measure values were fit on the same scaler that was generated with the train dataset which ensured that both train and test measures had been preprocessed over the same mean and standard deviations. Then the test data was also split into seasons using the time stamps and each of the season wise subset was predicted on the respective season wise model. Finally the predictions were extracted using the *predict_proba* method with the probability of the minority class being predicted rightly. The reference to our implementation can be found on the Github Link **https://github.com/NCSU-CSC522-Spring2022/Econet.git**

6 EXPERIMENTAL DESIGN

6.1 RQ1) Can there be any pattern found in the data for effective regrouping?

The main takeaway from visualizing the data using a Correlation Matrix was that there was a significant level of seasonal and measure-wise connection. Each measure value had similar distribution over the hotter (summer), colder(winter) and windier(fall, spring) seasons of the year. Similarly values over the daytime period is also different from the values in the night period. Hence by using the timestamp information available in the dataset, we had split the training dataset into sub datasets by basing them into seasons namely (Fall, Summer, Spring and Winter). To add more detailing about the seasonal distribution, Summer and Fall were further subdivided into Summer1, Summer2, Fall1 and Fall2 respectively.

- Spring - 1743720 - 2021-03-01 to 2021-05-31
- Summer 1 - 890325 - 2021-06-01 to 2021-07-15
- Summer 2 - 1021725 - 2021-07-16 to 2021-08-31
- Fall 1 - 609004 - 2021-09-01 to 2021-10-15
- Fall 2 - 676800 - 2021-10-16 to 2021-11-30
- Winter - 1651700 - 2021-12-01 to 2021-02-28

Similarly the correlation matrix, delineates clearly about the correlation between the measures. This allows us to club highly correlated features and generate new features in the dataset. Upon using on all the measure attributes, there were group of attributes that showed high positive correlation which we broadly grouped them by their parent attributes,

- **Wind Attributes** : *ws10, ws02, ws06*
- **Temperature Attributes** : *temp_wxt, temp10, blackglobetemp, st*
- **Humidity Attributes** : *rh_hmp, rh_wxt, leafwetness*
- **Radiation Attributes** : *par, sr*

.

6.2 RQ2) Can time delta values extracted from the time stamps define more about the time dependency of the dataset?

The time stamps are as such datetime values which themselves do not tell much about the derivable factors. Also since the dataset contains details about time values over a year, things that remain comparable will be the time of the day at which the data is extracted. Hence the time stamps can be converted to time delta representing the minute value of a day ranging from 0 to 1440. The timestamp is initially converted to the dataTime format and the hours and minutes values were extracted. With this the exact minute reference for the day is computed and added as a new attribute. For multiple measure values on similar time stamps over a range of days, this can help highly generalize the predictions. This creates a strong temporal relationship which can be further extended using algorithms like Stacked LSTM and Recurrent Neural Networks.

6.3 RQ3) Can oversampling and undersampling strategies resample the data with expected distributions?

Since the data is highly imbalanced with the distibution being 95% of FALSE (Normal readings) class and 5% of TRUE (Erroneous reading), it is worthwhile to experiment resampling methods. We began by oversampling the minority class using the SMOTE algorithm which generates synthetic points of the minority class randomly all over the space. This gave both the majority and minority class to be of the same size. While training and evaluating the oversampled dataset, the Precision value dropped drastically from 70% to 20% because it no more maintains the inclination of the majority class and there is a great chance of majority class labels being misclassified also. As an effect the overall F1-score also dropped under 40% which made us not proceed with this algorithm.

TOMEKLinks was used to undersample the data. This is a distance based undersampling algorithm that removes the majority points that are very closer to the minority points to expand the decision margin between them for effective classification. Upon performing TOMEKLinks on season wise segregated data, there was a 5% decrease in the count of majority classes. This increased the recall of the minority class to 77.7% without affecting the precision of the majority class. The overall F-1 score was 85.15%.

6.4 RQ4) Can XGboost and Random Forest classifiers performs better than other classifiers like Logistic Regression and K-NN?

Since the dataset has large number of features, a classifier algorithm that can generalize the features is very effective. KNN is not appreciable method on this dataset because the number of training datapoints is roughly 6M and the algorithm computes distance between K neighbours for N such points which grows exponentially. Logistic Regression being a simple binary classifier is also not very effective because the data is imabalanced and the decision boundary is not distinct. In tree based algorithms like Random Forest and XGBoost, the tree can expand to maximum depth generalizing the features while ensembling the features. This creates a boosting based algorithm which improves the F1-Score extremely high. When negative loss is observed, XGBoost divides the nodes to the maximum depth and then prunes the tree backwards to eliminate the splits. When the model is used in parallel processing, the learning rate of the model is boosted. Overfitting of data is also avoided by regularizing the aim, which is one of the most crucial things to keep in mind while working with an unbalanced dataset.

7 RESULTS

Upon evaluating the performance of the algorithms on the validation set, all of the seasonal dataset produced better results on the XGBoost algorithm except the Fall-1 model that fared well on the Random Forest algorithm.

For Fall-1 dataset, RandomForest produced an F-1 Score of 0.78 with a recall of 0.71 and precision of 0.87 for the minority class. On the other hand XGBoost produced an F-1 score of 0.78 with recall of 0.67 and precision of 0.92. This shows that XGBoost did

```
Running  randomForest  for the file  fall1
[[121278     50]
 [   139    334]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    121328
        True       0.87      0.71      0.78       473

    accuracy                           1.00    121801
   macro avg       0.93      0.85      0.89    121801
weighted avg       1.00      1.00      1.00    121801
```

Figure 5: Confusion Matrix and Classification Report for Random Forest Algorithm on Fall-1 dataset

```
Running  xgBoost  for the file  fall1

[[121300     28]
 [   154    319]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    121328
        True       0.92      0.67      0.78       473

    accuracy                           1.00    121801
   macro avg       0.96      0.84      0.89    121801
weighted avg       1.00      1.00      1.00    121801
```

Figure 6: Confusion Matrix and Classification Report for XGBoost Algorithm on Fall-1 dataset

comparatiely better with precision although random forest had a better recall value.

```
Running  randomForest  for the file  fall2
[[134149    122]
 [   165    924]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    134271
        True       0.88      0.85      0.87      1089

    accuracy                           1.00    135360
   macro avg       0.94      0.92      0.93    135360
weighted avg       1.00      1.00      1.00    135360
```

Figure 7: Confusion Matrix and Classification Report for Random Forest Algorithm on Fall-2 dataset

```
Running  xgBoost  for the file  fall2

[[134153    118]
 [   163    926]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    134271
        True       0.89      0.85      0.87      1089

    accuracy                           1.00    135360
   macro avg       0.94      0.92      0.93    135360
weighted avg       1.00      1.00      1.00    135360
```

Figure 8: Confusion Matrix and Classification Report for XGBoost Algorithm on Fall-2 dataset

For Fall-2, Random Forest had 0.88 precision, 0.85 recall and 0.87 as F-1 while XGBoost had 0.89 precision. 0.85 recall and 0.87 F-1 which shows that both Random Forest and XGBoost performed equally well.

Spring dataset also showed similar results on both random forest and XGBoost with comparable ranges of precision of 0.96 and 0.98, recall of 0.94 and 0.92, F-1 of 0.95 and 0.95 respectively. Spring had a very high value of precision, recall and F-1 compared to fall and summer because the variance of the measure value was very less in this subset and the number of data points were also high which helped in better generalization while validating and testing.

Summer-1 performed averagely with an F-1 of 0.80 on Random Forest and 0.81 on XGBoost. The recall remained same for both with 0.78 while the precision changed a bit with 0.81 for RF and

```
Running  randomForest  for the file  spring
[[334627    572]
 [   769  12776]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    335199
        True       0.96      0.94      0.95     13545

    accuracy                           1.00    348744
   macro avg       0.98      0.97      0.97    348744
weighted avg       1.00      1.00      1.00    348744
```

Figure 9: Confusion Matrix and Classification Report for Random Forest Algorithm on Spring dataset

```
Running  xgBoost  for the file  spring
[[334965    234]
 [  1044  12501]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    335199
        True       0.98      0.92      0.95     13545

    accuracy                           1.00    348744
   macro avg       0.99      0.96      0.97    348744
weighted avg       1.00      1.00      1.00    348744
```

Figure 10: Confusion Matrix and Classification Report for XGBoost Algorithm on Spring dataset

```
Running  randomForest  for the file  summer1
[[175206    442]
 [   520   1897]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    175648
        True       0.81      0.78      0.80      2417

    accuracy                           0.99    178065
   macro avg       0.90      0.89      0.90    178065
weighted avg       0.99      0.99      0.99    178065
```

Figure 11: Confusion Matrix and Classification Report for Random Forest Algorithm on Summer-1 dataset

```
Running  xgBoost  for the file  summer1
[[175299    349]
 [   542   1875]]
              precision    recall  f1-score   support

       False       1.00      1.00      1.00    175648
        True       0.84      0.78      0.81      2417

    accuracy                           0.99    178065
   macro avg       0.92      0.89      0.90    178065
weighted avg       0.99      0.99      0.99    178065
```

Figure 12: Confusion Matrix and Classification Report for XGBoost Algorithm on Summer-1 dataset

0.84 for XGBoost. Overall by a small margin XGBoost performed better than Random Forest.

Summer-2 had performed poorly amongst all the subsets with F-1 score of just 0.72 on both the Random Forest and XGBoost algorithm. Recall was 0.67 while precision was 0.78 which gave a strong sign for improving the data preprocessing better on such widely varied data points.

Winter had a massive chunk of data distribution which helped us get an high value of F-1 score, Recall and Precision of 0.96 on both Random Forest and XGBoost algorithm respectively.

While running through the test dataset, we predicted each of the seasonal subset on the respective season wise generated best model to get the best prediction probability for the minority class. Upon submitting the results over the online platform CodaLab **https://codalab.lisn.upsaclay.fr/competitions/4076** setup as a part of the research study, it proved that the predictions generated by the Random Forest Algorithm outscored the XGBoost algorithm. The submission made through XGBoost algorithm had an overall AUC-Precision-Recall score of 82.297 while that of Random Forest

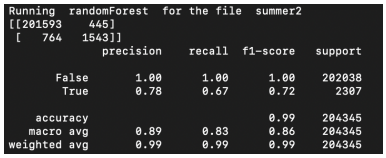

Figure 13: Confusion Matrix and Classification Report for Random Forest Algorithm on Summer-2 dataset

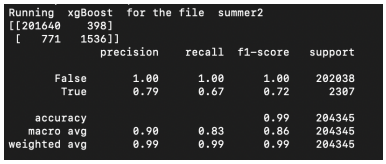

Figure 14: Confusion Matrix and Classification Report for XGBoost Algorithm on Summer-2 dataset

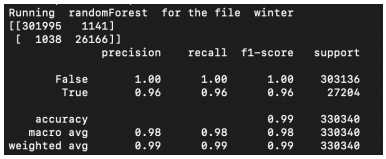

Figure 15: Confusion Matrix and Classification Report for Random Forest Algorithm on Winter dataset

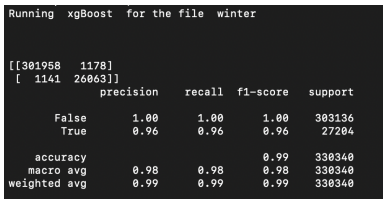

Figure 16: Confusion Matrix and Classification Report for XGBoost Algorithm on Winter dataset

had a score of 90.99. This is because, since Random Forest being itself an ensemble based model that selects the best class predictions from the set of trees, had generalized well on the test dataset.

8 DISCUSSION

While performing data preprocessing and training the models, there were a few limitations that were encountered. Initially the time distribution over the dataset is not really continuous as the training data had the first 3 weeks' data in a month and the last one weeks' data was placed as test data. This had a visible break in the time pattern which made us explore along the horizons of wrapping over the available time period over a different window size. Also there were a few stations that did not have measures captured. Since our approach had derived measure wise correlation, the missing measures were balanced out in the measure group and the predictions were produced accurately. For example ws06 and ws10 correlated maximum and were grouped as wind attributes. If there were a few stations that did not have measure values for ws10, they were captured using the dependency of wind attributes and predictions were done accordingly.

Although the XGBoost algorithm performed well on all the season wise dataset while validating, Random Forest emerged to produce the best prediction on the test dataset which was different than we expected. This later was understood because of the fact that XGBoost gives importance to functional space to reduce the cost of the model while Random forest gives more importance to hyper parameters to optimize the model.

This paves way for further discussion in future to explore on other tree based and boosting based algorithms on such time series based imabalanced data. Some other scope of work in future is to try temporal neural network architectures like RNN, LSTM, Bidirectional LSTM and Stacked LSTM to completely conceive the time perception and tune the weights of the layers accordingly.

9 CONCLUSION

It is critical to detect incorrect data in the quality control dataset, which needs the highest level of classification algorithm accuracy. In this paper, we used randomforest and xGBoost classifiers to determine one such classifier for classification on an imbalanced dataset. Because the imbalanced data had a season-by-season distribution, we opted to divide it down by seasons. On this season-by-season dataset, the performance of each classifier was evaluated using a variety of evaluation metrics. On the test dataset, the Random Forest classifier has an AUC-PR of 90.99%, a Positive F-1 of 84.15%, and an AUC-ROC of 96.27%. The results of this metric were superior to those of the xGBoost classifier. As a result, when it comes to detecting erroneous values in the Econet QC dataset, the Random forest classifier works best with season-wise data.

REFERENCES

[1] Nitesh Chawla, Kevin Bowyer, Lawrence Hall, and W. Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res. (JAIR)* 16 (06 2002), 321–357. https://doi.org/10.1613/jair.953

[2] Mahashweta Das and Srinivasan Parthasarathy. 2009. Anomaly detection and spatio-temporal analysis of global climate system. In *Proceedings of the third international workshop on knowledge discovery from sensor data.*

[3] M. Gundreddy, V. Shah, and E. Suess. 2018. Spatial and Temporal Trends in Weather Forecasting and Improving Predictions with ARIMA Modeling. In *In Proceedings of the Joint Statistical Meeting, Vancouver, BC, Canada.*

[4] Mohammed Khalilia, Sounak Chakraborty, and Mihail Popescu. 2011. Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making 11.1* (2011).

[5] C. V. Priscilla and D. P. Prabha et al. 2020. Influence of Optimizing XGBoost to handle Class Imbalance in Credit Card Fraud Detection. *Third International Conference on Smart Systems and Inventive Technology* (2020).

[6] I. Tomek. 1976. Two modifications of CNN. *BIn Systems, Man, and Cybernetics, IEEE Transactions* 6 (1976).

[7] Wahyono, Teguh, and et al. 2020. Anomaly Detection in Climate Data Using Stacked and Densely Connected Long Short-Term Memory Model. *Journal of Computers* 31, 4 (2020), 42–53.