Group 26 Final Project Report : EcoNET - True Error Detection

Chaitanya Patel
cpatel3@ncsu.edu
Department of Computer Science
North Carolina State University

Harish Hasti
hhasti@ncsu.edu
Department of Computer Science
North Carolina State University

Lakshmi Swetha Gavini
lgavini@ncsu.edu
Department of Computer Science
North Carolina State University

Subodh Thota
sthota@ncsu.edu
Department of Computer Science
North Carolina State University

1 INTRODUCTION

With the advent of new techniques in Machine Learning, Deep Learning, and advancements in computing power, today's world is data-driven everywhere. North Carolina State Climate Office combines data maintained by the National Weather Service, Federal Aviation Administration, and the US National Resource Conservation Service. They collect data sent by ECONet Stations. The North Carolina Environment and Climate Observing Network (ECONet) currently consists of 43, 10-meter (33-foot) tall aluminium towers across 34 counties in North Carolina and one county in South Carolina. ECONet stations measure a variety of atmospheric and soil parameters. These observations are made on 1 minute intervals and relayed back to our office in Raleigh, NC every 5 minutes.[1]

1.1 Problem Statement

We introduce our problem as a classification task over the EcoNet dataset. The dataset includes 23 measurements from 45 weather stations for each minute of each day for 2021. This approximately translates to 544 Million data points. Unfortunately not all data that is collected is without error. To solve this problem, the NC Climate office runs the data through an automated quality control system to to try and catch erroneous data points that need to be removed. The QC system flags over 8 million readings each year that need to be manually reviewed by experts. The goal of this experiment is to reduce the number of reviews that need to be manually reviewed by predicting each measurements are truly erroneous and which measurements are flagged as false positives. Some other considerations that should be looked at are that each station can be looked at separately because each station will be different in their readings including range and mean. Some stations may be in the mountains and may have different readings compared to stations by the beach.

1.2 Related Work

Severe weather events are can occur more frequently due to climate change and so collecting accurate data is a must. Because weather stations can be costly to install, smaller Automatic Weather Stations (Mini-AWSs) can be deployed to collect data at more frequent times. These systems can have the same shortcomings as bigger systems when it comes to erroneous data being collected when the frequency of collection is increased.

Kim et al. developed Mini-AWSs to produce lower installation and maintenance costs for weather collection. They used several methods to detect erroneous data. Some of the methods include Linear regression, Artificial neural Networks, Support Vector Regression, and Expectation-Maximization Clustering. Their study found that Support Vector Regression (SMOreg) and EM clustering brought the error threshold under the WMO standards.[2]

Lee et al. developed a novel method for determining abnormal values in a meteorological data based on support vector regression (SVR). They use SVR to determine if the difference between an estimated value and the actual observed value is significantly different enough to identify it as erroneous. They found that using SVR reduced the RMSE by an average of 45 percent while also maintaining competitive computing times compared to other estimators.[3]

Braei and Wagner have performed a survey on the State-Of-The-Art on anomaly detection in univariate time-series. Their study showed "that despite the advances in machine learning approaches and deep neural networks, still the statistical methods that rely on the generating process are generally performing best." These models also tend to perform faster and have a very lower training and prediction times.[4]

2 METHODS

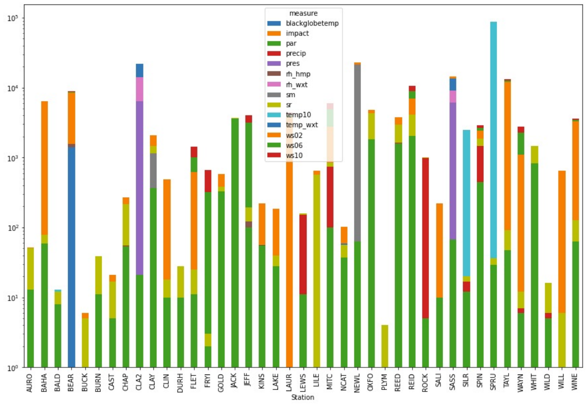
The below section details the our approach we followed along with the justification of the rationale behind them.

2.1 Approach

These are the following steps involved in predicting the erroneous QC data

2.1.1 Exploratory Data Analysis. After the preparation of data, Exploratory Data Analysis (EDA) is performed to get more insights about the data. Also to understand the hidden patterns, this step is important. This step reveals the number of each of the target values present in the dataset, the frequency of each measurement that is marked as the reason for the error target value (Figure 1).

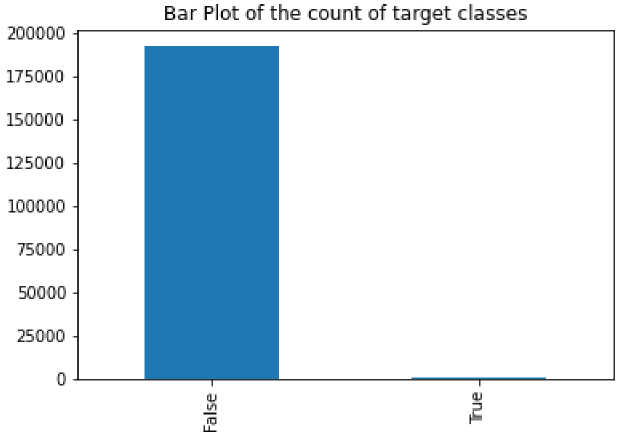
Figure 1 - True Target measurements for each Station (Log Scale)



2.1.2 Data Preparation. Upon performing exploratory data analysis on the training data, the sole use QC flags did not provide any useful insight as to which data points would actually be erroneous, so to get a clearer picture, the use of the full data was included. The full data is first normalized using StandardScalar. For SVMs the use of all of the data is necessary, so both training and test files are used and then merged with the full dataset files from EcoNet with the data('Station', 'Ob') as keys. For MLP classifiers, only the train data was used for training the model.

2.1.3 Class Imbalance. As seen from the Fig 2 of the counts of the 2 classes in the dataset, we can see that the False target label's count is very less compared to the other. This needs to be balanced as we should not make the machine learning model biased towards the majority class. This phenomenon is called class imbalance, we implement techniques like Over Sampling/ Under sampling to handle this imbalance. When we proceed through the experimentation, it will be known if under sampling the majority class will be helpful in getting more recall score.

Figure 2 - Target Values



2.1.4 Data Pre-Processing. The input data in the EcoNet dataset is a mixture of categorical, numerical data. Since the machine learning algorithm cannot process categorical variables, they are converted to one hot encoding. In our project, we are applying the following data pre-processing methods on the EcoNet data:

- Each of the numeric data columns in the training set is first normalized by subtracting the mean of the training data set, then dividing each object by the standard deviation of the set. This is done using the StandardScalar API from sklearn.

2.1.5 Classification Models. Following are the baseline models used to train the data before model evaluation.

SVM - Support Vector Machines(SVM) is a machine learning algorithm where the model tries to find the best hyperplane which divides the two data classes. In general, SVM is used for classification/ regression, but more preferred for classification tasks. Here, in our model, Grid Search cross validation has been used along with SVM to get the better accuracy among the hyper parameter search. The hyper parameters used are C, gamma and the type of kernel.[7]

MLPClassifier - Multi-layer Perceptron classifier is a Neural network that performs the task of classification.[7]. For this dataset, our job is to classify which QC data is considered truly erroneous. When we combine the QC data with the full dataset that includes the reading for that station and date/time, we can train a MLP classifier to determine erroneous data. Some attributes can be removed as they may not contain useful information for the classifier. This includes the Station information, the Date/time, the the measurement/ value that the QC originally flagged. Before training the model, the data is run through StandardScalar to normalize it.[8]

2.1.6 Model Evaluation. There is no need to split the data as the train and test data are prepared separately. Since the test data does not include the target variable, models are to be evaluated only based on train data and the predictions for the test data have to be submitted for review. For the training data, the initial steps is to get the accuracy. Since, False Negatives are more important than the False Positives in our classification task, models which improve the Recall score have to be selected.

2.2 Rationale

2.2.1 Data Preprocessing.

- The Training data that is used only includes 4 flags, the data/time, the station, the measurement type and the actual measurement reading. We believe that this data is not enough to create an efficient model that can be used for prediction. Because of this, data was merged with the full dataset that was available for each station. The date/time was also used for the merge. After this is done, we saved the data so we wouldn't have to redo everything again.
- Normalizing the data is an important step as it will help the machine learning model not confuse about the input variables. When the ranges of the input variables differ,

the gradients take a lot of time and can go back and forth before reaching to the local or global minimum. To avoid this problem, data is normalized before passing into the machine learning model to get the gradients reach local or global minimum faster.

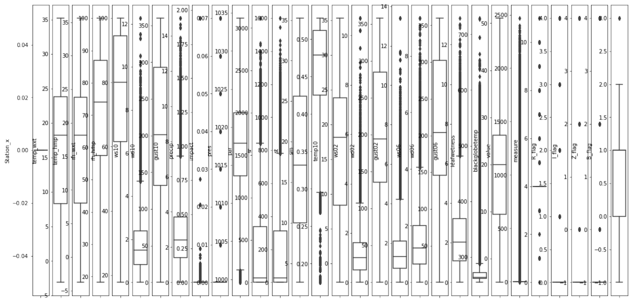
2.2.2 *Exploratory Data Analysis.*

- When the statistics such as mean, median, mode, the percentiles are calculated, it is evident that the range of values for the numeric features is different and hence the data needs to be normalized. This is evident in the Fig 2 - combined box-plot of all features.
- It is also evident from Fig 2 that there are outliers in the data and should be handled in such a way that there is no trade off for accuracy.
- From Fig 3, we can visualize the correlation between the various attributes, and the highest correlation is on the diagonal as it represents the correlation with itself.

2.2.3 *Classification Models.*

- SVM - After going through the SVM, the baseline SVM is built with default parameters.
- MLP Classifier - The baseline classifier consists of parameters: hidden layer sizes = (27,7), random state = 24, early stopping = True

Figure 2 - Combined Box Plot of Numeric Data

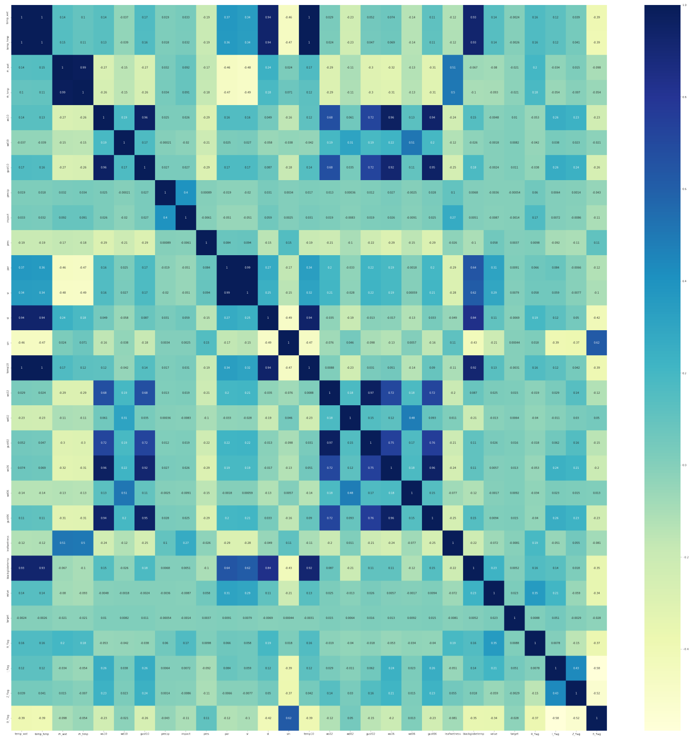


3 EXPERIMENTS

This section covers dataset details, the hypotheses we use, and design specific information we have based our investigation upon.

3.1 Dataset

Figure 3 - Correlation map



We are using the ecoNET dataset provided for the class. This dataset consists of 23 measurements from sensors at 45 weather stations around North Carolina for the year 2021. The attributes are the station where the reading was taken, the date of the reading, the value of the reading, the reading measure, target value set to true if the reading was reviewed by a human and found to be likely erroneous, and to false if reviewed by a human and found to be likely accurate, R/I/Z/B flag where one of 4 automated Quality Control (QC) flags generates based on the reading. Atmospheric and soil parameters measured include

Air Temperature (2 meters)
Relative Humidity (2 meters)
Barometric Pressure (2 meters)
Wind Speed and Direction (3 levels — 2, 6, and 10 meters)
Precipitation (2 levels — 1 meter and 2 meters)
Total Solar Radiation (2 meters)
Photosynthetically Active Radiation (2 meters)
Leaf Wetness (0.6 meters)
Soil Moisture (20 centimeters below the surface)
Soil Temperature (10 centimeters below the surface)
Black Globe Temperature (2 meters at 34 stations)
Air Temperature (9 meters at 35 stations)

3.2 Hypothesis

- Machine and Deep Learning models after handling the imbalance, and outliers should yield better results
- Handling the imbalance of under sampled class should yield better results as model is less biased towards the under sampled class
- Applying Feature engineering on the most correlated features by understanding the relationships between them should yield better results with the test data
- Applying PCA may not work on our problem as there are less pairs of attributes which has high correlation.

3.3 Experimental Design/ Environment

3.3.1 Design.

- The first step is preparing the data by merging the training/testing files with the full Station data using pandas inner join based on the keys = 'Ob' and 'Station' attributes. This ensures that each row gets the full features for that Station and date/time.
- Then StandardScaler from sklearn.preprocessing is used to normalize the data.
- Because there is a class imbalance in this dataset. Oversampling is utilized for the True Target values and brings the Training dataset to 12 million rows. This is too large to train with, so random sampling is used to bring the total data rows back down to 6 million samples.
- Now that the dataset is ready for training our models, it is divided into X and y dataframes.
- The two primary models used for this experiment are Support Vector Machines and MLP classifier. The data is further processed according to each model requirements. The SVM model uses one hot encoding for categorical values nad removes data/time information. The MLP classifier removed certain columns such as Ob, Station, measurement and Value rows as they are either categorical or unnecessary for training the model.
- Testing the validity of the model is done using Cross Validation. The SVM model will use a 3 fold CV to confirm its findings and accuracy. The MLP classifier will use a 4 fold CV for recall and then retrain over the whole training dataset to confirm its findings and metrics.

3.3.2 Experiment Environment.

- The code is written in Jupyter Notebook, a python based coding environment with better visualization of inputs and outputs in a browser.
- The Python libraries which are used are Pandas, Numpy, glob, os, matplotlib, seaborn, sklearn (scikit - learn: This module has many sub-libraries which were used throughout the project)

4 RESULTS

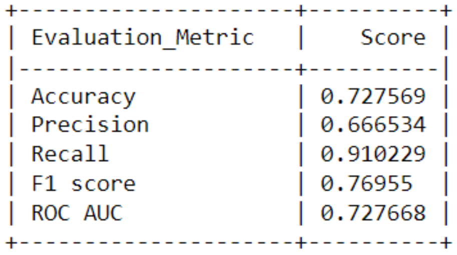
4.1 Results

4.1.1 SVM. Support Vector Machine is used to train the data with 4 fold cross validation after normalizing the data and converting the categorical variables into one hot encoding. The parameter search for SVM are as follows. The values of C used are 0.01, 0.1, 1, 10 and kernel used is 'linear'.

Since the best C and default parameter = 0.01, the performance metrics are same before and after the hyper parameter tuning for this setting.

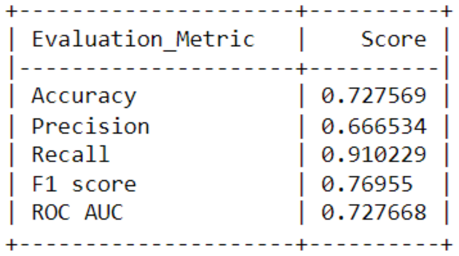
The ROC curve and Precision recall curves are plotted.

Figure 5 - SVM Results - Before Hyper Parameter Tuning



```
+---------------------+----------+
| Evaluation_Metric   |    Score |
|---------------------+----------|
| Accuracy            | 0.727569 |
| Precision           | 0.666534 |
| Recall              | 0.910229 |
| F1 score            | 0.76955  |
| ROC AUC             | 0.727668 |
+---------------------+----------+
```

Figure 6 - SVM Results - After Hyper Parameter Tuning



```
+---------------------+----------+
| Evaluation_Metric   |    Score |
|---------------------+----------|
| Accuracy            | 0.727569 |
| Precision           | 0.666534 |
| Recall              | 0.910229 |
| F1 score            | 0.76955  |
| ROC AUC             | 0.727668 |
+---------------------+----------+
```

Figure 7 - SVM Classifier Results - ROC

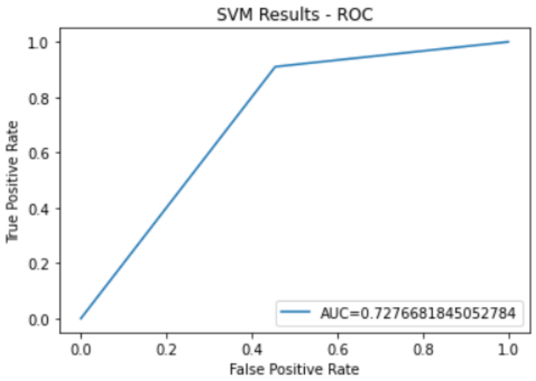
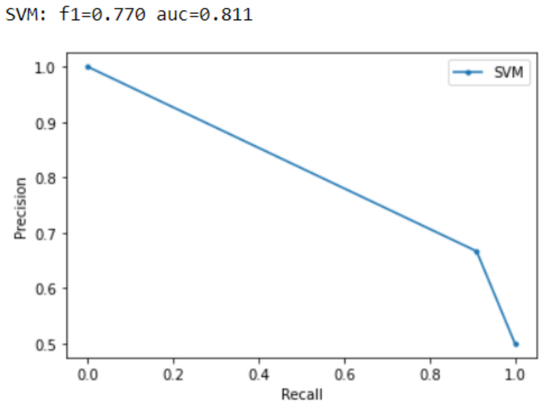


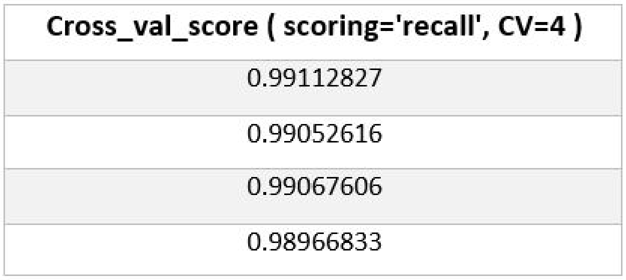
Figure 8 - SVM Classifier Results - Precision Recall Curve



```
SVM: f1=0.770 auc=0.811
```

4.1.2 MLP Classifier. MLP Classifier parameters were chosen using GridSearchCV, the returned optimized model should have activation: 'tanh', and hidden_layer_sizes: '(27, 6)' with early_stopping enabled. The model was then trained with the full standardized training dataset that also over-sampled the minority class. A cross validation (cv=4) was done on the model to ensure it would not overfit the data. The CV scoring used was recall.

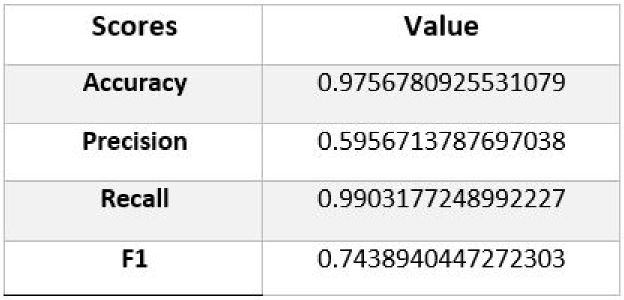
Figure 9 - MLP Classifier CV Results - Processed Training data



Cross_val_score (scoring='recall', CV=4)
0.99112827
0.99052616
0.99067606
0.98966833

After the cross validation is done, used the original training data to check the validity of the model. The Accuracy, Precision, Recall, F1 are calculated. The most important metric for this dataset is recall, which this model gives us 99%. The ROC and Precision Recall Curve graphs are also calculated.

Figure 10 - MLP Classifier Results - Original Training data



Scores	Value
Accuracy	0.9756780925531079
Precision	0.5956713787697038
Recall	0.9903177248992227
F1	0.7438940447272303

4.1.3 XgBoost Classifier. XgBoost Classifier is used to fit the training data with the default parameters. XgBoosting is an optimization of gradient boosting algorithm[9]. A cross fold of 4 is used and the results are as follows.

After the cross validation, the training data is used to calculate the performance metrics such as accuracy, precision, recall, f1-score and confusion matrix. Since the recall is the most important metric, and received 0.996 recall.

The training data is used to plot the Reciever Operator Charecterestics curve, Precesion Recall curve. From these two plots, it is evident that the XgBoost performed well on the training data.

Figure 11 - MLP Classifier Results - ROC

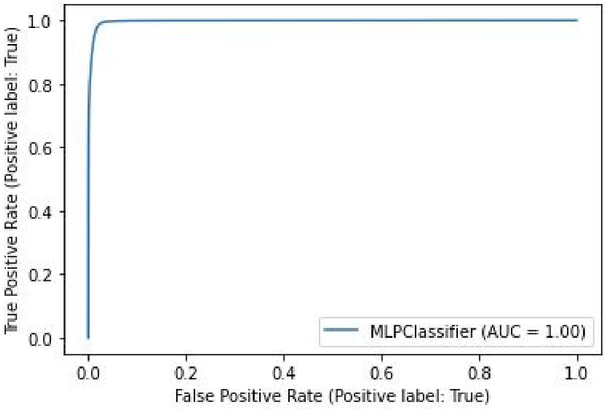


Figure 12 - MLP Classifier Results - Precision Recall Curve

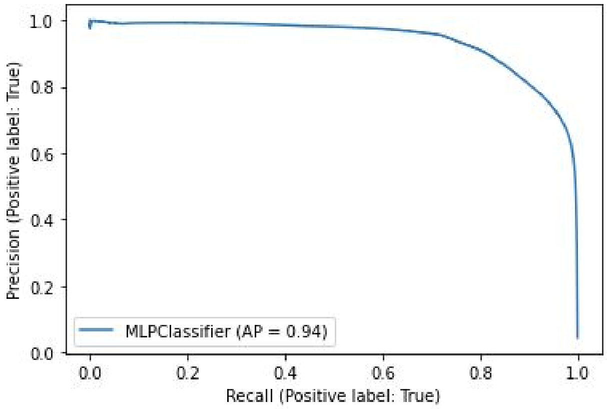
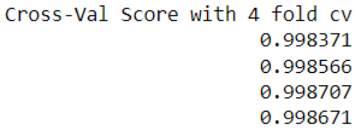
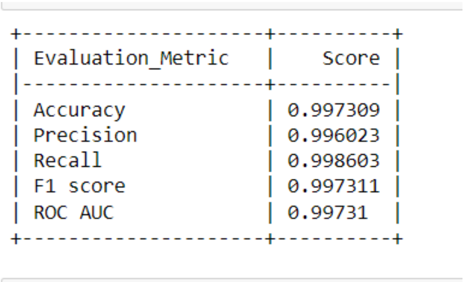


Figure 13 - Xg Boost Classifier CV Results - Processed Training data



```
Cross-Val Score with 4 fold cv
                         0.998371
                         0.998566
                         0.998707
                         0.998671
```

Figure 14 - Xg Boost Classifier Results - Processed Training data



```
+--------------------+----------+
| Evaluation_Metric  |    Score |
|--------------------+----------|
| Accuracy           | 0.997309 |
| Precision          | 0.996023 |
| Recall             | 0.998603 |
| F1 score           | 0.997311 |
| ROC AUC            | 0.99731  |
+--------------------+----------+
```

Figure 15 - Xg Boost Classifier Results - ROC

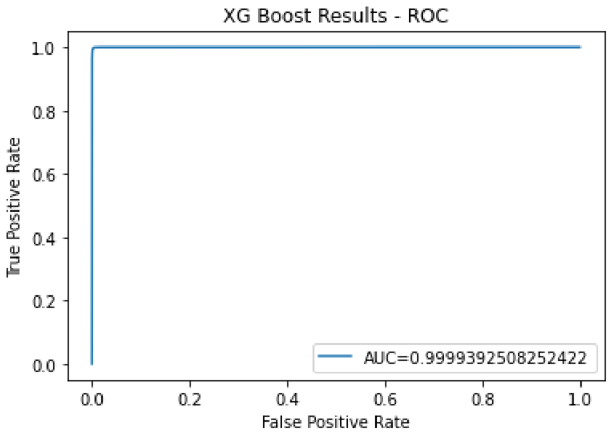
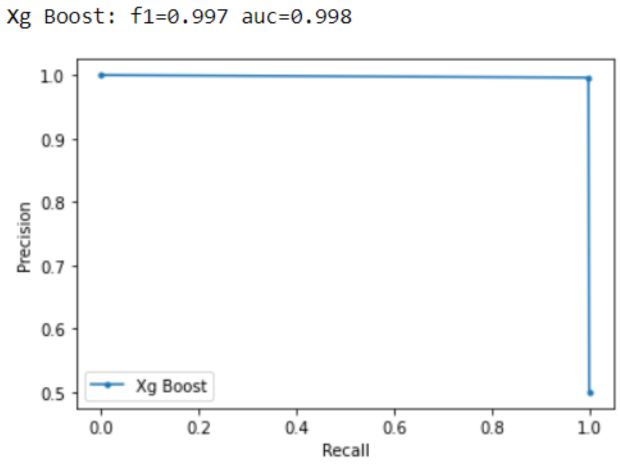


Figure 16 - Xg Boost Classifier Results - Precision Recall Curve



5 DISCUSSION

Performing EDA helps us understand more about the dataset that is being looked at. The data is highly imbalanced, it mostly contains false labels at a ratio of about 30 to 1 for False to True. The dataset also had a wide varieties of stations that they were collected from and each station works in its own way. The standardization of data collected would need to be validated. To solve this issue of imbalance, oversampling methods like SMOTE or simple oversampling were used to try an get a balance between True and False data. The original Training dataset is not enough for creating a model that can predict erroneous data. This can be confirmed by looking at the relations between the QC flags and class labels to check for any correlation between them. The solve for this, the full data for each station that is provided is used with the Training data by merging. This gives each element in the training dataset a reference as to what the full data looked like when it was collected.

The dataset that we are working with is collected by a automated system that liberally marks any data it would suspect of being erroneous. This means that if the 4 flags may not show that it is erroneous, the QC program may still mark the data. This may explain as to why there is such an imbalance in the training data. Since the point of this project is to find out which of the elements in the dataset are actually erroneous, the use of recall as the metric means that out of all the data that is True, how many times did we predict that it was actually true. This recall metric is more important than accuracy because the True elements in the dataset have to be hand picked by people and so if a model can collect all the true values with high recall, it would lower the amount of work humans would have to do.

Since the dataset has more features and many of them are of the form numeric. Considering the imbalance, recall is the most important metric. Support Vector Classifier was used as one of the baseline model as SVM's tend to learn the dividing plane better compared to other machine learning models. The training time of SVM classifier with kernels other than linear take lot of time and hence linear kernel is used for my modelling. With the default parameters(C=0.01), a recall of 0.91 is achieved on training data. In the next steps, hyper parameter tuning was performed to get the best classification model. The hyper parameter values of C which are used - 0.01, 0.1, 1, 10 and the best C is 0.01. Then the two plots - Roc curve, precision - recall curve plot is plotted where it is evident that model performed well on the training data.

Since this dataset is in a tabular form, there are a number of methods that could be deployed to model the data. The first method explored is using ANNs. The data that is merged with the Training set is all numerical data, this is good because we can easily feed that into a neural network at try to model the dataset. The simplest model to use is a MLP (Multi-layer Classifier) neural network. The output of this classifier is a probability for True or false. The hyper-parameters for our model were chosen using a GridsearchCV and the best hyper-parameters included a hidden layer of (27, 6) meaning 6 layers of fully connected 27 nodes each, the use of a tanh activation function and also early stopping. After deciding out our hyper-parameters, we have to cross validate the model, so Cross_val_score is used with the scoring results for recall, because it is the metric we want to maximize. When the results for the Cross validation look promising, the model is trained on the full dataset and again used to predict the classes for the original Training dataset, that does not over-sample the imbalance data. The use of ROC and PR-curve[10] help use visualize how well our model is doing. Boosting Algorithms are machine learning models which aggregate weak learners and give better results. Hence, to our machine learning model. Because of the high amount of data, XG Boost (Extreme Boosting) algorithm, which is optimising the boosting algorithm with parallelism is used to fit the training data with 4 fold cross validation. A recall of 0.998 is achieved after training. The precision-recall and ROC curve is almost perfect for XgBoost algorithm when plotted with the training data.

After trying the various machine learning models and uploading the predictions in the competition website, MLP Classifier gave the best results on testing data. The performance metrics are as follows - Accuracy: 0.9398; Avg Prec: 0.3419; Recall: 0.5455.

6 CONCLUSION

We understand from the baseline models used to train the original unbalanced dataset before model evaluation and comparing undersampling and oversampling as methods with class imbalance, that an highly imbalanced dataset negatively affects the model's learning. This situation has high chances of overfitting or generalization of the model in regard to the minority class. From the exploratory data analysis, one should avoid removing important information during undersampling the majority class as it helps our model to learn dominant features. In order for the learning to be executed fairly and without any bias, it is good to go with oversampling techniques to sync up with the count of minority classes with majority.

The baseline models which are trained on training data have shown different outcomes with decent results, but when predictions are uploaded to check the scores, MLP Classifier performed well on the test data after hyper-parameter tuning. Even though there is almost same performance between Xg Boost and MLP classifier with training data, MLP Classifier with the architecture presented had clear edge over the test data.

Since our data resembles the time series data, statistical models such as simple moving average, ARIMA can be performed to check for performance.

7 MEETING SCHEDULE

The overall schedule till date is as follows - All the dates mentioned below are via Zoom and were attended by all the teammates during Tuesday/Thursday/Saturday - 1pm to 3pm

- 03/19/2022 Data Set and Problem Statement conversations.
- 03/31/2022 Rough idea on model selection, based on exploratory data analysis.
- 04/05/2022 Baseline model discussion, different model implementation and evaluation.
- 04/07/2022 Started working on the report and finalized it on 04/09/2022.
- 04/14/2022 Handling class imbalances using various sampling techniques and make sure the data is balanced - Swetha and Subodh.
- 04/16/2022 SVRs and ANNs training and evaluation - Chaitanya.
- 04/19/2022 Random Forest, XGBoost, Support Vector Machines(SVMs) taining and evaluation - Harish
- 04/21/2022 Change the performance metrics and draw insights from our model - Swetha, Subodh, Chaitanya, Harish.
- 04/23/2022 - Discussion on the evaluations of models - ANN'S, XGBoost, SVMs and dropping off SVR's, Random Forests - Swetha, Subodh, Chaitanya, Harish.
- 04/23/2022 - Finalizing on the performance metrics and tabulating them for all the models - Swetha, Subodh, Chaitanya, Harish.
- 04/26/2022 - Additions, modifications and improvements to the report

8 GITHUB LINK

The project code is present in the following GitHub repository: https://github.ncsu.edu/hhasti/EcoNet-Team-26

REFERENCES

[1] https://econet.climate.ncsu.edu/about/

[2] Min-Ki Lee, Seung-Hyun Moon, Yourim Yoon, Yong-Hyuk Kim, and Byung-Ro Moon. 2018. [*Detecting anomalies in meteorological data using support vector regression. Advances in Meteorology 2018 (2018), 1–14*]

[3] Ji-Hun Ha, Yong-Hyuk Kim, Hyo-Hyuc Im, Na-Young Kim, Sangjin Sim, and Yourim Yoon. 2018. Error correction of meteorological data obtained with Mini-awss based on machine learning. Advances in Meteorology 2018 (2018), 1–8. DOI:http://dx.doi.org/10.1155/2018/7210137

[4] Mohammad Braei and Sebastian Wagner. 2020. Anomaly detection in Univariate Time-Series: A Survey on the state-of-the-art. (April 2020). Retrieved April 23, 2022 from https://arxiv.org/abs/2004.00433

[5] https://analyticsindiamag.com/using-near-miss-algorithm-for-imbalanced-datasets/

[6] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: Synthetic minority over-sampling technique. J Artif Intell Res 16: 321–357. DOI:https://doi.org/10.1613/jair.953

[7] TY - JOUR AU - Zanaty, E. PY - 2012/11/01 SP - 177–183 T1 - Support Vector Machines (SVMs) versus Multilayer Perception (MLP) in data classification VL - 13 DO - 10.1016/j.eij.2012.08.002 JO - Egyptian Informatics Journal ER DOI: https://doi.org/10.1016/j.eij.2012.08.002

[8] H. Bourlard and C. Wellekens, "Links between markov models and multilayer perceptrons," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 12, pp. 1167 –1178, Dec. 1990. DOI: https://doi.org/10.1109/34.62605

[9] Tianqi Chen and Carlos Guestrin, "XGBoost: A Scalable Tree Boosting System," KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. DOI: https://dl.acm.org/doi/10.1145/2939672.2939785

[10] Jesse Davis and Mark Goadrich, "The Relationship Between Precision-Recall and ROC Curves," Appearing in Proceedings of the 23 rd International Conference on Machine Learning, Pittsburgh, PA, 2006. DOI: https://www.biostat.wisc.edu/~page/rocpr.pdf

[11] Evgeniou, Theodoros Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-712.