Detection of Erroneous Weather Data

Srujan Ponnur
North Carolina State University
Raleigh, NC, USA
sponnur@ncsu.edu

Saikaushik Kalyanraman
North Carolina State University
Raleigh, NC, USA
skalyan2@ncsu.edu

Vamsi Madhur Varada
North Carolina State University
Raleigh, NC, USA
vvarada3@ncsu.edu

Bhaskara Bhavani Krishna Kotaru
North Carolina State University
Raleigh, NC, USA
bkotaru@ncsu.edu

1 INTRODUCTION

The NC Climate Office receives measurements from various(23) sensors across 45 stations each day. This amounts to around 544 million samples of data each year. First, these data points get automatically flagged by quality control sensors. Then these flagged samples have to be manually reviewed by experts at the Climate Office to check if any of samples are erroneous(which would mean the sensors need to be looked at). Since the number of samples is huge, this would take a considerable amount of time and effort.

ECONet data are critical to researchers as well as a variety of other sectors such as agriculture and energy. This wider community also utilizes ECONet data for decision-making. Example applications include agriculture irrigation, pest management, and air quality forecasting. Overall, many sectors in North Carolina rely on ECONet data. Thus, it is important to ensure that the highest quality data is available to these various communities. Since the experts have manually collected and flagged the data for the year 2021, we considered this problem an ideal specimen to solve using machine learning methods. This is a supervised, binary classification problem, where we need to predict whether different measures from weather stations are erroneous(instead of manual flagging). A good starting point would be to analyze the process followed by the scientists in manually flagging the data and find out how much each quality control flag influenced the scientists' final decision.

2 RELATED WORK

The data set we are dealing with has imbalanced data, which means it has only a small percentage of 'abnormal' or minority class data. It is also the case that the cost of misclassifying an abnormal (interesting) example as a normal example is often much higher than the cost of the reverse error. [1] Under-sampling of the majority (normal) class or oversampling of the minority(abnormal) class have been proposed as good means of increasing the sensitivity of a classifier to the minority class. In this project, the data imbalance is being addressed in the following ways:

- Random undersampling technique is applied on the majority class.
- Near Miss Undersampling, a K-NN based approach, is applied on the majority class.[5]
- Synthetic Minority Oversampling Technique, or SMOTE for short is applied on the minority class.
- A combination of over and under sampling(SMOTE + Edited Nearest Neighbor method) is applied on the respective parts of the dataset. This is done to increase the model performance even further[2].

[3] proposes the creation of a new attribute from the available 4 Quality control flags

3 APPROACH

First, train data is loaded and pre-processed to convert sensor types('measure' attribute) and station codes('station' attribute) using one-hot encoding. Then, feature engineering is performed to include new(possibly relevant) features and drop certain irrelevant ones. Specifically, the 'Ob' attribute(denotes the timestamp at the time of recording the measure) is split into 4 different attributes, namely 'month', 'day', 'hour', and 'minute'. The possibility of a higher correlation between attributes like 'month' and 'day' with the 'target' attribute, rather than just the timestamp is the rationale behind this step. For example, winter months like December and January usually record lower temperatures, so a high temperature reading during these months is an easy giveaway that the sensor might be faulty.

A new attribute, 'QC_score' is added to assist the model in better predicting the 'target' attribute based on domain knowledge obtained from [3]. QC score is calculated for each datapoint in the dataset, as technicians assess data stations if they receive mediocre QC scores. So, We can see that the QC score influences the decision rather than the individual QC flags.

While generating the QC score for the given data, it is observed that there are about 50% of data points for which QC score is currently defaulted to a score of -1, This shortcoming is observed due to certain combinations of (R_flag, B_flag, I_flag, Z_flag) not being handled in the table 1 present in [3]. The updated table(obtained from domain experts) shown in figure handles many more combinations than its predecessor and it is used to generate the new 'qc_score' values.

Also using the 'measure' attribute, the 'value' attribute is normalized. The data is then sampled to address the data imbalance before any models are fitted on it.

To establish a reference point for more complex models, three baseline models are applied to the pre-processed data, namely Logistic Regression, Decision Tree, and Random Forest. Baseline models are simple to setup and have a reasonable chance of providing decent results. Since experimenting with them is quick and low cost, a few of them were tried out and results were tabulated. A basic train validation split is performed on the pre-processed data and the classification report is generated for both train and validation datasets. These classification reports signify the baseline results, any future model would need to perform better than these baseline models to be considered.

4 THE DATASET

EcoNet Dataset: This dataset has data collected from 45 weather stations across North Carolina at each minute of each day for 2021. Each row in data contains 23 different measurements collected by different sensors. Currently, four different types of automated Quality Control(QC) routines are performed on ECONet data: range check, buddy check, intersensor check, and trend check. The range check(R_flag) uses climatology to test the validity of observations. The buddy check(B_flag) utilizes data from neighboring stations to test whether a value (and its rate of change) are consistent with that of surrounding locations. The intersensor test(I_flag) is run on stations that have co-located parameters to cross check sensor performance. The trend check(Z_flag) uses the values from previous hours to validate the current observation given the present state of the atmosphere.

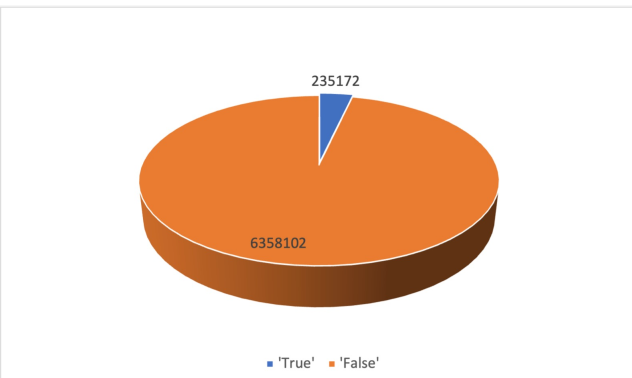

Figure 1: Target variable distribution

The 'station' attribute denotes the respective weather station where the measure was recorded in. The 'Ob' attribute denotes the timestamp when the respective measure was recorded. The 'measure' attribute indicates the type of sensor data(Eg:relative humidity, etc.) being recorded. The 'value' attribute denotes the exact measurement on the respective sensor. Finally, the 'target' is a binary attribute that is labeled True, if the reading was reviewed by a human and found to be likely erroneous, and False if reviewed by a human and found to be likely accurate. The target is also the attribute we need to predict using our machine learning algorithm.

5 HYPOTHESES

As the data set here is highly skewed towards majority class, the performance of the classification model cannot be judged using accuracy alone. The recall of the minority class and the precision of the majority class are more significant in evaluating the performance of a model. Since, reporting a erroneous data point as 'False' means that a faulty sensor wouldn't be detected, this could prove costly. Hence recall of the minority(True) class is the most significant metric to consider while evaluating models.

A precision-recall curve is a plot of the precision(y-axis) and the recall(x-axis) [4]. The performance of the classification model can be improved by adding additional useful attributes. Firstly, a new attribute named 'qc_score' which can be calculated using the automated QC flags(R_flag, I_flag, B_flag, Z_flag). Next, the timestamp attribute('Ob') is split into 4 different attributes to represent 'month', 'day', 'hour' and 'minute' separately.

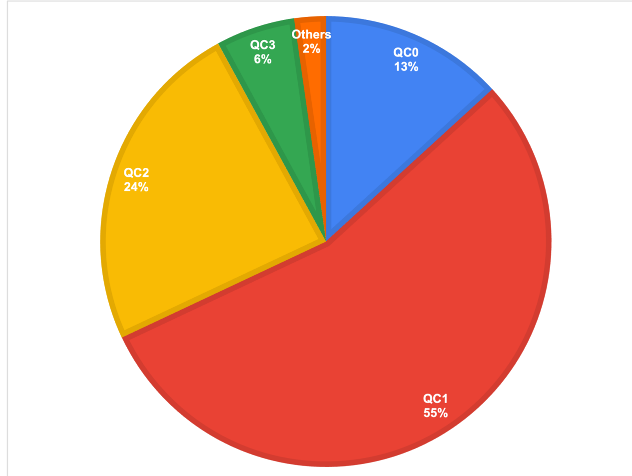

Figure 2: New QC_score distribution

6 EXPERIMENTAL DESIGN

Initially, the 'value' attribute has different ranges of data depending on the sensor. So, in order to transform the attribute to be on a similar scale, normalization is performed after dividing the data using the 'measure' attribute. This is done using StandardScaler.

QC_Score is calculated using R, B, I, Z flags. For calculating the QC Score, all the manually curated flag checks are appended in their respective order to generate the QC Flag. This QC Flag can be categorized into different QC Score values (0-3) based on the table[1]. The timestamp attribute 'Ob' is split into 4 different attributes to improve correlation between specific aspects of the timestamp and the dependent variable.

The data is split into train and validation sets. To ensure the similarity in train and validation data sets, stratified flag is used during this split. To overcome the data imbalance, under sampling and oversampling are performed. For Under sampling we used, Random under sampling which randomly chooses majority data points without replacement to match the number of minority data points. Near Miss undersampling which uses a K-NN based approach to choose majority class data points. The version being used is NearMiss-1, which chooses the majority class examples with minimum average distance(Euclidean) to the three closest minority class examples.

For Oversampling we used, SMOTENC oversamples the minority class. It also works on Nominal and Categorical data is suitable for the current dataset as it has 2 categorical attributes. ADASYN works similarly to the borderline SMOTE, but focuses on gaps in

Table 1: Quality control scores

QC0		QC1		QC2		QC3	
U0 (Anywhere in Flag String)						U4 (Anywhere in Flag String) R4 (Anywhere in Flag String unless a U0 precedes it)	
R0	R1B0Z0	R0Z2	R2B0	R0I2	R2B3Z0		
R0Z0	R1B1Z0	R0I1	R2B0Z2	R0I2Z0	R3	R0Z4	R3Z4
R0I0	R2B0Z0	R0I0Z0	R2B1Z0	R0I4	R3Z0	R0I1Z4	R3I4
R0I0Z0	Z0	R0B1	R2B2Z0	R0I4Z0	R3Z2	R0I2Z4	R3B0Z4
R0B0	Z2	R0B2Z0	R3B0Z0	R0B0Z4	R3	R0I4Z4	R3B1Z4
R0B0Z0	I0	R0B3Z0	R3B1Z0	R0B1Z4	R3I0	R0B3Z4	R3B2
R0B1Z0	B0	R1	I1	R0B2	R3B0	R0B4Z0	R3B2Z0
R1Z0	B0Z0	R1B0Z4	B0Z4	R0B2Z4	R3B0Z2	R0B4Z2	R3B2Z2
R1I0	B0Z2	R1I2	B1	R0B3	R3B1	R0B4Z4	R3B2Z4
R1B0	B1Z0	R1B1	B1Z2	R0B3Z2	R3B1Z2	R0B5Z0	R3B3
		R1B2Z0	B2Z0	R1Z4	Z4	R0B5Z2	R3B3Z0
		R2Z0		R1I4	I2	R0B5Z4	R3B3Z2
				R1B1Z4	B1Z4	R1B3	R3B3Z4
				R1B2	B2	R1B3Z4	R3B4
				R1B2Z4	B2Z2	R1B4Z0	R3B4Z0
				R1B3Z0	B2Z4	R1B4Z2	R3B4Z2
				R2	B3	R1B4Z4	R3B4Z4
				R2Z2	B3Z0	R1B5Z0	R3B5
				R2Z4	B3Z2	R1B5Z2	R3B5Z0
				R2B0Z4	B4	R1B5Z4	R3B5Z2
				R2B1Z2	B4Z0	R2B2Z4	R3B5Z4
				R2B1Z4	B4Z2	R2B3	I4
				R2B1		R2B3Z2	B3Z4
				R2B2		R2B3Z4	B4Z4
				R2B2Z2		R2B4Z0	B5
						R2B4Z2	B5Z0
						R2B4Z4	B5Z2
						R2B5Z0	B5Z4
						R2B5Z2	
						R2B5Z4	

the clusters. Borderline SMOTE focuses on increasing the minority class near the decision boundary.

To further increase the performance of models, a combination of under sampling the majority class and oversampling the minority class is also performed on the data set. SMOTEENN(Oversampling using SMOTE and undersampling using Edited Nearest Neighbors).

A few baseline models are applied to the above sampled data.

- Decision tree is applied with a maximum depth of 5. Later this is run on different maximum depth values.
- An ensemble approach RandomForest with different maximum depth values.
- Logistic Regression model with maximum iterations of 10,000.
- KNN model with k value of 5

We also implemented a Deep Neural network with architecture shown in Figure 3. Activation function chosen is Leaky ReLu and output layer has a sigmoid function which gives the results between 0 and 1. This model is ran with BinaryCrossentropy loss and different metrics such as accuracy, AUC. Hyper parameter tuning is performed to get better results.

7 RESULTS

Once the data was pre-processed, four different Machine Learning algorithms(Decision Tree, Logistic Regression, Random Forest and ANN) were run on the resulting data. Each machine learning algorithm underwent hyperparameter tuning(cross-validation) to ensure the best possible results. The training and validation set had an 75-25 split. The ANN model ran for 10 epochs. Each of these models were run, both including and excluding the 'qc_score' column to understand how 'qc_score' impacts the decision making of each model. When using the initial version of the 'qc_score' table to calculate 'qc_score', the models were consistently performing better without the 'qc_score' column present. After updating the 'qc_score' table to table 3, the models(with 'qc_score' present) started performing on par with the models without 'qc_score' present. Further, each model was also trained on both undersampled data(to address the imbalance) and the original data. All the best hyperparameter values are mentioned in the git repository mentioned in the last section(ModelRandomSearch.ipynb)

7.1 Logistic Regression

It can be observed in tables 2 and 3 that even a logistic regression model with the best hyperparameters was under performing on the test and validation data sets. Logistic regression(on undersampled data) has a good class-True recall score in the train data set, but by the poor recall score seen on the test set, it is clear that logistic regression(on undersampled data) overfits on the train data set.

Table 2: Logistic regression on Undersampled data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99745	0.90475	0.94884	6358102
True	0.26688	0.93741	0.41547	235172
Accuracy			0.90592	6593274
Macro Avg	0.63216	0.92108	0.68216	6593274
Weighted Avg	0.97139	0.90592	0.92982	6593274

Test set

AUC _ PR	0.480796
RECALL	0.244350
AUC_ ROC	0.838865
POSITIVE_ F1	0.385592
ACC	0.968662

Table 3: Logistic regression on complete data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.98421	0.99444	0.98930	6358102
True	0.79092	0.56862	0.66159	235172
Accuracy			0.97925	6593274
Macro Avg	0.88756	0.78153	0.82545	6593274
Weighted Avg	0.97731	0.97925	0.97761	6593274

Test set

AUC _ PR	0.625057
RECALL	0.453973
AUC_ ROC	0.953499
POSITIVE_ F1	0.578294
ACC	0.973355

7.2 Decision Tree

Decision Tree classifier clearly performs better when it is trained on undersampled data. As observed in tables 4 and 5, decision tree(on undersampled data) has a slightly low AUC score on the test and validation data sets, hence it slightly overfits on train data.

Though the outcome of decision tree is highly interpretable, there exists a problem of multicollinearity, when two variables both explain the same thing, a decision tree will greedily choose the best one, whereas many other methods will use them both. The column 'Ob' and the columns 'month', 'day', 'hour' and 'minute' both explain the same thing. Similarly the column 'qc_score' is derived from the R, B, I and Z flags. Hence these columns contribute to the problem of multicollinearity.

Table 4: Decision tree classifier on Undersampled data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99788	0.96171	0.97946	6358102
True	0.47715	0.94476	0.63406	235172
Accuracy			0.96110	6593274
Macro Avg	0.73751	0.95323	0.80676	6593274
Weighted Avg	0.97931	0.96110	0.96714	6593274

Test set

AUC _ PR	0.758338
RECALL	0.893877
AUC_ ROC	0.959684
POSITIVE_ F1	0.320388
ACC	0.847389

Table 5: Decision tree classifier on complete data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.97449	0.99849	0.98634	6358102
True	0.87797	0.29325	0.43965	235172
Accuracy			0.97334	6593274
Macro Avg	0.92623	0.64587	0.71300	6593274
Weighted Avg	0.97104	0.97334	0.96684	6593274

Test set

AUC _ PR	0.480796
RECALL	0.244350
AUC_ ROC	0.838865
POSITIVE_ F1	0.385592
ACC	0.968662

7.3 Random Forest

Ensemble methods such as random forests can negate the problem of multicollinearity to a certain extent. Hence, in tables 6 and 7 it has much better AUC scores on test and validation sets than the decision tree. If, just the AUC score is to be prioritized, this random forest classifier(on the complete data) is the best performing model. The random forest classifier, excluding the 'qc_score' at table 9 performs slightly better than when the 'qc_score' is included.

7.4 ANN

If recall is to be prioritized, ANN(on the complete data) is the best performing model. ANN performs better as we are back propagating based on the loss which is calculated by AUC Score metric. So, it distinguish between both the minority and majority class better. So, we get a good recall when compared to other models.

Table 6: Random Forest on Undersampled data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99996	0.99598	0.99797	6358102
True	0.90198	0.99900	0.94801	235172
Accuracy			0.99609	6593274
Macro Avg	0.95097	0.99749	0.97299	6593274
Weighted Avg	0.99647	0.99609	0.99619	6593274

Test set		
	AUC _ PR	0.920489
	RECALL	0.874277
	AUC_ ROC	0.987346
	POSITIVE_ F1	0.878216
	ACC	0.990242

Table 7: Random Forest on complete data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99974	0.99998	0.99986	6358102
True	0.99951	0.99287	0.99618	235172
Accuracy			0.99973	6593274
Macro Avg	0.99962	0.99643	0.99802	6593274
Weighted Avg	0.99973	0.99973	0.99973	6593274

Test set		
	AUC_ PR	0.949204
	RECALL	0.770403
	AUC_ ROC	0.988582
	POSITIVE_ F1	0.855728
	ACC	0.989546

Table 8: Random Forest on Undersampled data without QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99999	0.99662	0.99830	6358102
True	0.91614	0.99974	0.95612	235172
Accuracy			0.99673	6593274
Macro Avg	0.95807	0.99818	0.97721	6593274
Weighted Avg	0.99700	0.99673	0.99680	6593274

Test set		
	AUC_ PR	0.918565
	RECALL	0.834221
	AUC_ ROC	0.987539
	POSITIVE_ F1	0.865579
	ACC	0.989573

Table 9: Random forest on complete data without QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99996	1.00000	0.99998	6358102
True	0.99991	0.99900	0.99946	235172
Accuracy			0.99996	6593274
Macro Avg	0.99994	0.99950	0.99972	6593274
Weighted Avg	0.99996	0.99996	0.99996	6593274

Test set		
	AUC _ PR	0.949279
	RECALL	0.773187
	AUC_ ROC	0.992072
	POSITIVE_ F1	0.858489
	ACC	0.989742

Table 10: Neural network on Undersampled data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99561	0.99274	0.99418	211529
True	0.99277	0.99563	0.99420	211780
Accuracy			0.99419	423309
Macro Avg	0.99419	0.99419	0.99419	423309
Weighted Avg	0.99419	0.99419	0.99419	423309

Test set		
	AUC _ PR	0.873319
	RECALL	0.811730
	AUC_ ROC	0.963237
	POSITIVE_ F1	0.807552
	ACC	0.984430

Table 11: Neural network on Undersampled data without QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99875	0.98228	0.99045	211457
True	0.98260	0.99878	0.99062	211852
Accuracy			0.99054	423309
Macro Avg	0.99068	0.99053	0.99054	423309
Weighted Avg	0.99067	0.99054	0.99054	423309

Test set		
	AUC _ PR	0.918434
	RECALL	0.922968
	AUC_ ROC	0.982541
	POSITIVE_ F1	0.765848
	ACC	0.977287

8 CONCLUSION

In this report, firstly, the advantages of undersampling on a skewed data set can be seen. In most of the results, applying a model to the undersampled data produces better class - True(minority class) recall values. Another observation is that even the best random forest model(with its high AUC scores) under performs in terms of class - True(minority class) recall values in comparison to the deep neural network(ANN model). As it has good scores in all the metrics(both recall and AUC score) this ANN is the most well

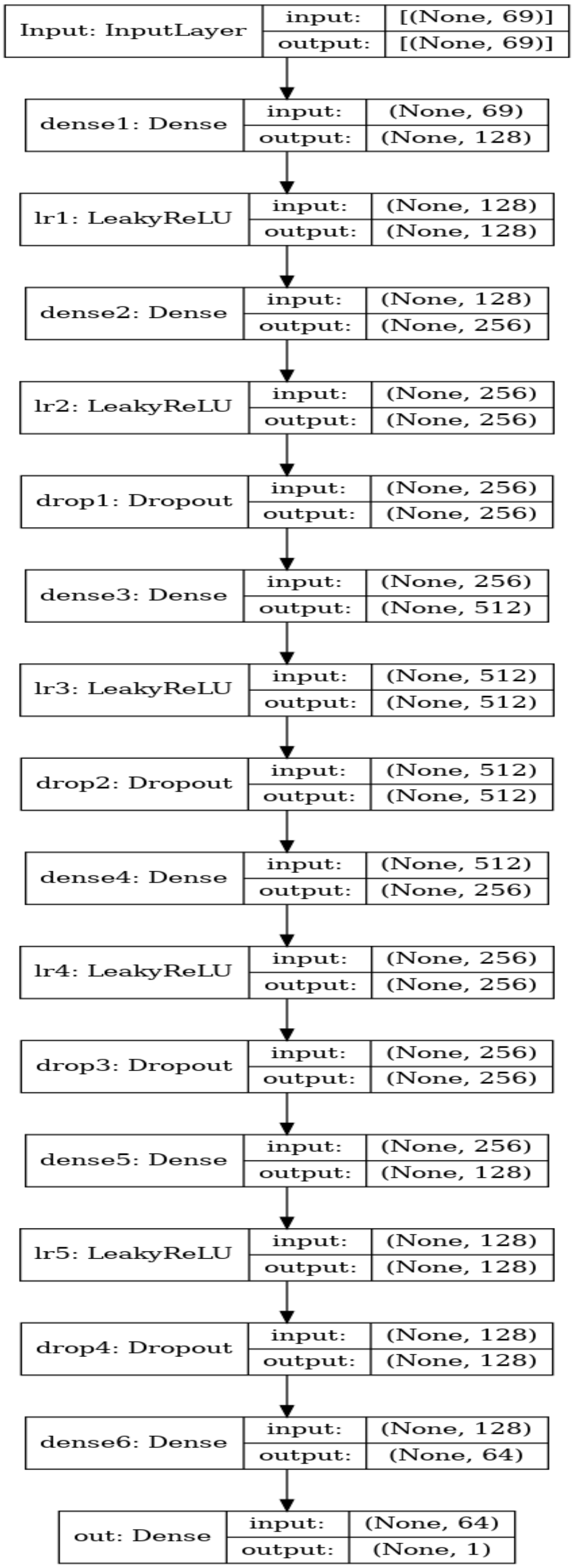

Figure 3: Deep Neural Network Architecture

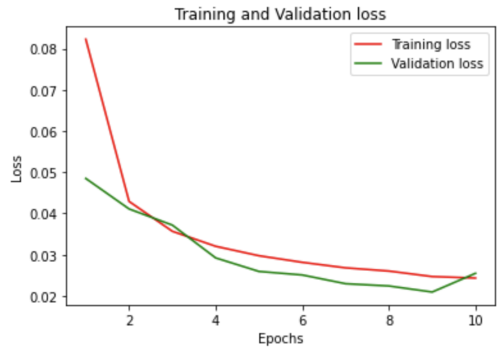

Figure 4: Training and Validation Loss in Neural Network for each epoch

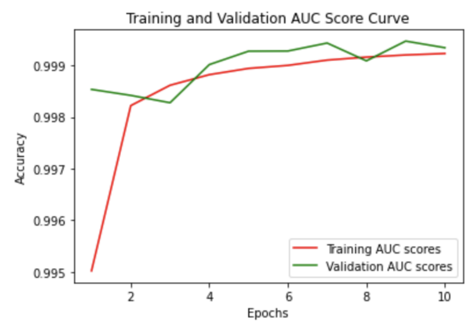

Figure 5: Training and Validation AUC Score in Neural Network for each epoch

Table 12: Neural network on complete data with QC_ score

Training set

	Precision	Recall	F-1 score	Support
False	0.99833	0.99453	0.99643	5722314
True	0.86590	0.95500	0.90827	211632
Accuracy			0.99312	5933946
Macro Avg	0.93212	0.97477	0.95235	5933946
Weighted Avg	0.99361	0.99312	0.99328	5933946

Test set	AUC _ PR	0.922705
	RECALL	0.867115
	AUC_ ROC	0.952825
	POSITIVE_ F1	0.869606
	ACC	0.989535

rounded model out of the ones experimented on. Observing the results of undersampling, applying oversampling methods could prove to be a game changer in combination with ANNs, which currently was computationally expensive. This could open up an avenue for future experimentation. Further, the complete data set comprised of not just the train and test sets, but also a bunch of station specific data sets. Using the additional data to make the

Dates	Agenda	Attendance	Time
03/25/2022	Overall project plan	All members were present	9:00AM-11:00AM
04/07/2022	QC Score & Sampling	All members were present	9:00AM-11:00AM
04/10/2022	Base line Models	All members were present	9:00AM-11:00AM
04/11/2022	Midway Project Report	All members were present	6:00PM-11:59PM
04/15/2022	Random Forest Model	All members were present	6:00PM-10:00 PM
04/19/2022	ANN Model and Tuning	All members were present	6:00PM-10:00 PM
04/25/2022	QC Score V2 & Hyper Parameter Tuning	All members were present	6:00PM-11:59 PM
04/26/2022	Final Report	All members were present	6:00PM-11:59 PM

Table 13: Meeting times, Agenda and Attendance.

current train(and test) data more robust could be another avenue to explore in the future.

9 MEETINGS

We had a total of 8 Meetings. One meeting to discuss about the project, One while coming up with the project proposal and the remaining discussions were mainly on implementing our approach, getting the status on the due tasks and finally, generating the mid-way and final project reports. Every team member has attended all the meetings and contributed equally to this project.

10 CODE REFERENCE

GitHub Repo Link

REFERENCES

[1] Chris Drummond and Robert Holte. 2003. *C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats OverSampling.* Retrieved April 10, 2022 from https://www.researchgate.net/publication/245593532_C45_Class_Imbalance_and_Cost_Sensitivity_Why_Under-Sampling_beats_OverSampling

[2] Mouna Lamari, Nabiha Azizi, Nacereddine Hammami, Assia Boukhamla, Soraya Cheriguene, Nadjette dendani, and Nacer Eddine Benzebouchi. 2020. *SMOTE-ENN Based Data Sampling and Improved Dynamic Ensemble Selection for Imbalanced Medical Data Classification.* Retrieved April 10, 2022 from https://www.researchgate.net/publication/342571467_SMOTE-ENN_Based_Data_Sampling_and_Improved_Dynamic_Ensemble_Selection_for_Imbalanced_Medical_Data_Classification

[3] John A. McGuire Heather A. Dinon Aldrige Ryan P. Boyles Sean P. Heuser*, Aaron P. Sims. 2021. *Quality Control Methods for the North Carolina Environment and Climate Observing Network.* Retrieved April 10, 2022 from https://econet.climate.ncsu.edu/wp-content/uploads/2021/05/QC-paperV4_NoLN.pdf

[4] Christopher K I Williams. 2021. *The Effect of Class Imbalance on Precision-Recall Curves.* Retrieved April 26, 2022 from https://arxiv.org/pdf/2007.01905

[5] Mani I. Zhang, J.P. 2003. *KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction.* Retrieved April 10, 2022 from https://www.site.uottawa.ca/~nat/Workshop2003/jzhang.pdf