Predicting COVID-19 Using Self-Reported Survey Data

by

Gokulan Vikash Babu

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2021 by the
Graduate Supervisory Committee:

Lalitha Sankar, Chair
Visar Berisha
Ming Zhao
Ni Trieu

ARIZONA STATE UNIVERSITY

May 2021

ABSTRACT

Infectious diseases spread at a rapid rate, due to the increasing mobility of human population. It is important to have a variety of containment and assessment strategies to prevent and limit their spread. In the on-going COVID-19 pandemic, telehealth services including daily health surveys are used to study the prevalence and severity of the disease. Daily health surveys can also help to study the progression and fluctuation of symptoms as recalling, tracking, and explaining symptoms to doctors can often be challenging for patients. Data aggregates collected from the daily health surveys can be used to identify the surge of a disease in a community. This thesis enhances a well-known boosting algorithm, XGBoost, to predict COVID-19 from the anonymized self-reported survey responses provided by Carnegie Melon University (CMU)-Delphi research group in collaboration with Facebook. Despite the tremendous COVID-19 surge in the United States, this survey dataset is highly imbalanced with 84% negative COVID-19 cases and 16% positive cases. It is tedious to learn from an imbalanced dataset, especially when the dataset could also be noisy, as seen commonly in self-reported surveys. This thesis addresses these challenges by enhancing XGBoost with a tunable loss function, α-loss, that interpolates between the exponential loss ($\alpha = 1/2$), the log-loss ($\alpha = 1$), and the 0-1 loss ($\alpha = \infty$). Results show that tuning XGBoost with α-loss can enhance performance over the standard XGBoost with log-loss ($\alpha = 1$).

ACKNOWLEDGMENTS

I would like to express my gratitude to my chair and advisor Professor Lalitha Sankar for her constant support and belief in me. I am grateful to my co-chair Professor Visar Berisha for sharing his thoughts on how to improve the project. I am thankful to other committee members, Professor Ming Zhao and Professor Ni Trieu for their motivation and help with the project. I would also like to thank the collaborators, Tyler Sypherd, for providing a great guidance and sharing his knowledge, and Nathan Stromberg, for helping with conducting experiments. I would like to acknowledge the grant from National Science Foundation (#2031799)and the Google AI for Social Good award for sponsoring this research project. Finally, I am most grateful to my family for their support and encouragement, and to my friends in Tempe for providing a healthy atmosphere.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Chapter 1

INTRODUCTION

COVID-19, a disease caused by a coronavirus, originated in November 2019 and has now reached pandemic levels across the globe [2]. Within a span of one and a half years, the disease has infected around 126M people worldwide, leaving 2.76M reported deaths as of March 26, 2021. In the United States of America, this disease has affected both the personal and professional lives of people with 30.1M cases and around 547K deaths as of March 26, 2021. The current gold standard detection of COVID-19 is through Reverse Transcription Polymerase Chain Reaction (RT-PCR) testing [3]. RT-PCR testing is expensive, time-consuming (takes up to 48 hours to get results) and requires a manual intervention which could violate social distancing. As the pandemic continues to evolve, it is useful to have multiple modalities to assess exposure risk. On university campuses, viral infections can spread very quickly from classrooms to dorm rooms, and eventually to the broader population. At many universities including ASU, students, staff and faculty on-campus are required to report their symptoms daily. These rich symptom survey data can be used to build an additional diagnostic tool to help users evaluate their COVID-19 exposure risk. This Master's thesis is a part of the NSF RAPID project - "Federated Analytics Based Contact Tracing", which involves building machine learning models from user's health survey data to help assess and prevent COVID-19 risk.

Symptom-based survey questions can be converted into diagnostic features and fed into a machine learning model to predict the risk of exposure. Additional forms of surveys such as phonation-based surveys [4] etc. can be used to explain the severity of the disease. Data analytics in the healthcare setting are often built based on linear

models. Given that healthcare datasets contain many continuous and categorical features, it is typical to use simple linear models to allow ease of interpreting results and explainability. Regularized regression, decision trees and random forests lend themselves well to deriving insights and explaining patterns in data. With a large high-dimensional dataset, it is common to use computationally efficient algorithms such as boosting algorithms.

Boosting algorithms train by adding weak learners iteratively, each trying to correct its predecessor. These algorithms are fast and accurate when the dataset is reliable and not noisy. However, in self-reported health survey responses, humans may input error responses either intentionally or unintentionally. Additionally, healthcare datasets may be highly imbalanced especially in settings such as on-going pandemic where a large fraction of the population is still not infected. Thus, machine learning models need to be robust to noise and imbalances in the dataset. Boosting algorithms, such as AdaBoost [5], perform well with imbalanced data but are not noise-resistant [6]. State-Of-The-Art (SOTA) gradient boosting based XGBoost logistic loss algorithm is believed to outperform other linear models due to its robustness, hardware, and software optimizations. These boosting algorithms are dependent on decision trees. Decision trees use tree-like representation, where nodes branch the samples, based on a feature threshold, and the leaves assign a class label to each sample.

With emerging diseases, such as the COVID-19, where testing is still limited with only a few people in the population being exposed to the disease, the survey responses data is inherently imbalanced and could be noisy as well. Thus, it is important to be robust to noise in such health survey datasets. This thesis addresses that problem and enhances XGBoost by incorporating a custom tunable loss function. Recently, custom tunable loss functions were found to suit well for imbalanced and noisy binary classification problems, as they interpolate between exponential loss, log-loss and 0-1

loss [7]. We incorporate α-loss, originally introduced by Liao *et al.* [1], with XGBoost to compare the performance of the native XGBoost log-loss with α-loss.

1.1 Related Work

DeCapprio *et al.* used the medicare data to build a COVID-19 risk index predictor [8]. The risk index is predicted using the hospitalization days as a dependant variable. Menni *et al.* used the top features from an app-based symptom tracker that tracks symptoms from symptomatic and asymptomatic individuals to predict COVID-19 in UK [9]. CMU Delphi survey group conducts health surveys across the US, to identify and track symptoms among healthy and infected individuals [10]. Prior work on α-loss by Sypherd *et al.* studies the performance of α-loss with logistic regression and Convolutional Neural Nets (CNN) in noisy label settings [7], [11]. This thesis extends the previous work on α-loss and studies its performance on a real-world problem.

1.2 Organization of the Thesis

This thesis is organized as follows:

- In **Chapter 2**, we summarize the well-known algorithm of XGBoost as well as its generalization to α-loss. We introduce an enhanced algorithm, called XGBoost.α, that incorporates α-loss with the native XGBoost, thus utilizing the hardware and software optimizations of XGBoost with α-loss.

- In **Chapter 3**, we highlight the transformation and preprocessing strategies applied to the large Facebook-CMU survey dataset.

- In **Chapter 4**, we analyze the outcomes of using *XGBoost.*α in imbalanced and noisy settings.

Chapter 2

XGBOOST.α

Boosting was first introduced by Schapire and Freund [12], [13]. The key idea behind boosting is to combine weak learners (decision trees) iteratively to build a strong linear model. AdaBoost was the boosting algorithm that was first developed [5]. AdaBoost is an exponential loss algorithm. Exponential loss is convex and the loss value grows exponentially for wrong predictions. Although AdaBoost does very well in reliable datasets, its performance degrades with noisy datasets. To tackle this, a class of boosting algorithms such as XGBoost was introduced [14].

2.1 XGBoost Logistic loss

XGBoost is a fast and accurate gradient boosting framework. The XGBoost logistic regression is a powerful solution to classification problems. The combined rule-based logic of many weak learners can detect reasonable and explainable patterns for approaching and solving classification problems. Given a dataset with n samples, the objective value of the dataset ($\mathcal{L}$) is given as the sum of the objective value of each data point ($l(y_i, \hat{y_i})$). The effectiveness of boosting is determined by the loss function, $l(y_i, \hat{y_i})$, that compares the predicted target $\hat{y_i}$ and the ground truth y_i.

$$\mathcal{L} = \sum_{i=1}^{n} l(y_i, \hat{y_i})$$

Boosting is done through boosting rounds, T. In each round, an optimal tree f_t is picked by XGBoost and is added on top of existing $(t-1)$ trees with x_i as input. Mathematically, f_t is represented as,

$$f_t = w_{q(x)}, \;\; f_t \, \epsilon \, R^T \tag{2.1}$$

where w is the leaf weight vector and $q(x)$ is a function that maps each data point to the leaf index. A regularization term, $\Omega(f_k)$, is added to penalize complex individual trees.

$$\mathcal{L} = \sum_{i=1}^{n} l\big((y_i, \hat{y_i}^{(t-1)}) + f_t(x_i)\big) + \sum_{k=1}^{K} \Omega(f_k)$$

XGBoost uses a convex loss function that is twice differentiable in order to optimize the objective using second-order approximation.

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \sum_{k=1}^{K} \Omega(f_k) \tag{2.2}$$

where, g_i and h_i are first and second-order derivatives of the objective function, $l(y_i, \hat{y_i})$. In this iterative process, learning is done over iterations from t = 1 to T. By simplifying 2.2 and rewriting $\tilde{\mathcal{L}}^{(t)}$ in terms of instance sets I_j, we get

$$\tilde{\mathcal{L}} = -\frac{1}{2} \sum_{j=1}^{T} \frac{(\sum_{i \epsilon I_j} g_i)^2}{\sum_{i \epsilon I_j} h_i + \lambda} + \gamma T \tag{2.3}$$

where, γ and λ controls the trade-off between local objective values and model complexity. The logistic regression loss function used in XGBoost is

$$l(y, \hat{y}) = -y \log \sigma(\hat{y} - y) \log(1 - \sigma(\hat{y})) \tag{2.4}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$, is the sigmoid function.

2.2 XGBoost.α: Using α-loss in XGBoost

The class of loss functions α-loss, parametrized by $\alpha \in (0, 1) \cup (1, \infty)$ with continuous extensions at 1 and ∞, was introduced by Liao *et al.* in [1] to quantify a class of adversaries in a data privacy setting. More recently, Sypherd *et al.* has studied in depth the application of α-loss to machine learning, and in particular, to the canonical problem of classification including the problem of optimizing the loss landscape for various learning models. Building on this, we incorporate this loss

family and redefine it in the context of XGBoost logistic regression to create a new class of algorithms, namely, XGBoost.α.

Definition 1 *Let $\mathcal{P}(\mathcal{Y})$ be the set of probability distributions over $\mathcal{Y}$. We define α-loss for $\alpha \in (0,1) \cup (1,\infty)$, $l^\alpha : \mathcal{Y} \times \mathcal{P}(\mathcal{Y}) \to \mathbb{R}_+$ as*

$$l^\alpha(y, \hat{P}) := \frac{\alpha}{\alpha - 1} \left(1 - \hat{P}(y)^{1-1/\alpha}\right), \tag{2.5}$$

and, by continuous extension, $l^1(y, \hat{P}) := -\log \hat{P}(y)$ and $l^\infty(y, \hat{P}) := 1 - \hat{P}(y)$.

Observe that for $(y, \hat{P})$ fixed, $l^\alpha(y, \hat{P})$ is continuous and monotonically decreasing in α. Also note that l^1 recovers log-loss, and plugging in $\alpha = 1/2$ yields $l^{1/2}(y, \hat{P}) := \hat{P}^{-1}(y) - 1$.

The above definition of α-loss presents a class of loss functions whose probability estimate changes as the value of α varies (figure 2.1). As α-loss is continuous, convex,

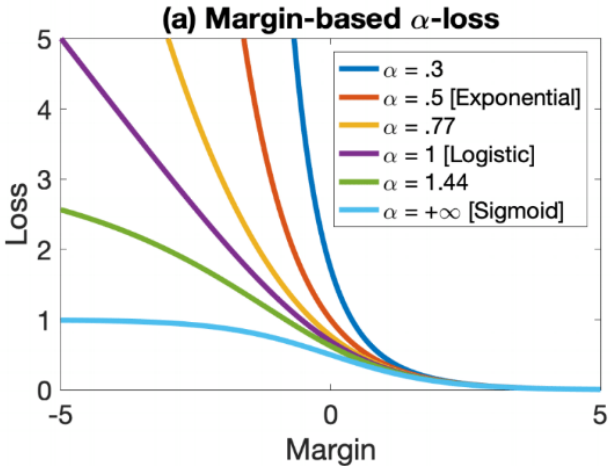

Figure 2.1: α-loss, as a Function of the Probability $p_y(Y)$ for a given $Y \in \mathcal{Y}$. This Figure is Taken from the Original Work of [1]

twice differentiable and decomposed over individual data points, it can be extended to XGBoost by deriving the first and second order derivative as follows:

First derivative:

$$g_i = \frac{\partial}{\partial z} l^{\alpha}(y_i, z)\big|_{z=\hat{y}_i^{t-1}}$$

$$= -y_i \sigma(\hat{y}_i^{(t-1)})^{-1/\alpha}.\sigma(\hat{y}_i^{(t-1)}).\sigma(-\hat{y}_i^{(t-1)}) + (1-y_i)(1-\sigma(\hat{y}_i^{(t-1)}))^{-1/\alpha}.\sigma(\hat{y}_i^{(t-1)}).\sigma(\hat{y}_i^{(t-1)})$$

$$= \sigma'(\hat{y}_i^{(t-1)}).\big[-y_i \sigma(\hat{y}_i^{(t-1)})^{-1/\alpha} + (1-y_i)\sigma(-\hat{y}_i^{(t-1)})^{-1/\alpha}\big] \tag{2.6}$$

Second derivative:

$$h_i = \frac{\partial^2}{\partial^2 z} l^{\alpha}(y_i, z)\big|_{z=\hat{y}_i^{t-1}}$$

$$= (1-y_i).\big[\sigma(\hat{y}_i^{(t-1)})\sigma(-\hat{y}_i^{(t-1)})^{2-1/\alpha} - (1-1/\alpha)\sigma(\hat{y}_i^{(t-1)})^2.\sigma(-\hat{y}_i^{(t-1)})^{1-1/\alpha}\big]$$

$$+ y_i.\big[\sigma(\hat{y}_i^{(t-1)})^{2-1/\alpha}\sigma(-\hat{y}_i^{(t-1)}) - (1-1/\alpha)\sigma(\hat{y}_i^{(t-1)})^{1-1/\alpha}.\sigma(-\hat{y}_i^{(t-1)})^2\big] \tag{2.7}$$

The first and second derivatives are implemented on top of the XGBoost source code and is made available at https://github.com/SankarLab/XGBoostPrivate.

By varying α, the loss function varies between exponential and sigmoid loss. Theoretically, $\alpha < 1$ performs better when there is a high class imbalance, as every wrong prediction on the minority class returns a large loss value, which is a property of the exponential loss. Similarly, $\alpha > 1$, does better with outliers as the loss function becomes quasi-convex. In the quasi-convex region, the loss values for wrong predictions during training is not as high as the exponential loss. Thus, the model is likely to become agnostic to the impact of noises in the data distribution.

Having defined XGBoost.α, we will now apply this enhanced version to the dataset at hand. We will describe the dataset and the data preprocessing strategies in the following chapter.

Chapter 3

DATASET

3.1 COVID-19 Symptom Survey Data

Facebook introduced an online COVID-19 daily health survey, in collaboration with the CMU-Delphi research group [15]. The survey responses dataset was collected by our research team through registering for the symptom data challenge and signing Data Use Agreements protecting the confidentiality of survey responses. To date, this dataset contains about 53,000 daily survey participants and 18,513,000 total responses. The data from the self-reported daily health survey is evolving and is updated regularly to accommodate new survey results that are collected daily. The dataset is anonymized, secure, and provides no means of tracing back the user. The data is also revised each month to include questions that pertain to the on-going trend of the virus spread.

Revisions to the survey questions are deployed in multiple waves, typically at an interval of a month. The first wave was deployed in April 2020 and the most recent wave(wave 10) was deployed on Mar 2, 2021. Additional questions prioritizing the items that have greater efficacy in understanding the human's behavior and the nature of the virus are added in each revision. The recent wave contains information on the symptoms, testing, contacts, risk factors, demographics, and the vaccination preferences. Table 3.1 summarizes the question types and provides a few examples for each question type.

This thesis focuses on using the daily health survey symptoms to build a risk prediction model. The survey dataset has numerous features with large number of

Question Type	Example
Yes/No (19 questions)	*A1_1 - A1_5** : Presence of symptoms such as fever, cough, sore throat, shortness of breath, and breathing difficulty; *C6*: Travelled outside state? *B10*: Tested for COVID-19? *V1*: Had vaccination?
Discrete choice (47 questions)	*C1*: Presence of comorbidities *D1*: Gender *D2*: Age *D7*: Race *C2*: Frequency of wearing masks
Numeric (5 questions)	*C10_1 - C10_4*: Number of people in direct contact with at work, shopping, social gatherings *A2*: Number of sick people in household *A3*: Zip Code

Table 3.1: Question Types with Examples in the Survey Dataset. (*) Represents Arbitrary Labels That Map to Survey Questions

missing values as they were added in the later waves as shown in Figure 3.1. Only a few survey items have lesser missing values as shown in Figure 3.2. Figure 3.3 shows the distribution of the top 5 binary features, that can be reliably used to build a good model. We only retain the questions that are present in all the waves, as the top 8 features, to build a classifier.

3.2 Preprocessing Techniques

The raw survey response data is converted into a classification dataset by filtering out the non-binary COVID-19 test result responses. For the waves that we considered in this dataset, a bulk of the population had not tested for COVID-19, as shown in Figure 3.4. Since we want to use features to predict whether someone is COVID-19 infected or not, we prune the dataset further to include only those responses that have

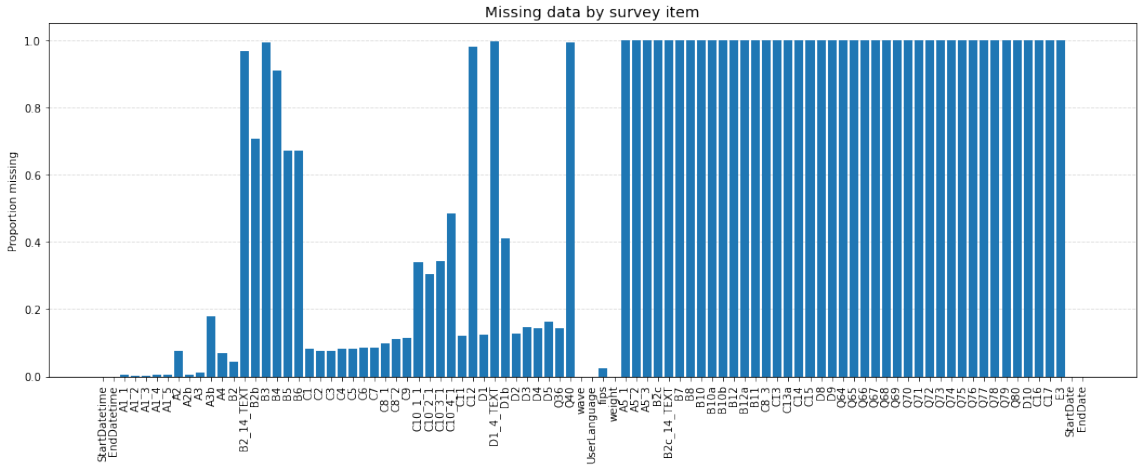

Figure 3.1: Missing Data by Survey Item. X-axis Shows the Survey Item Labels That Are Used for Mapping the Response to the Actual Questions. Note That Most Features Have Many Missing Values as They Were Added in the Later Survey Waves

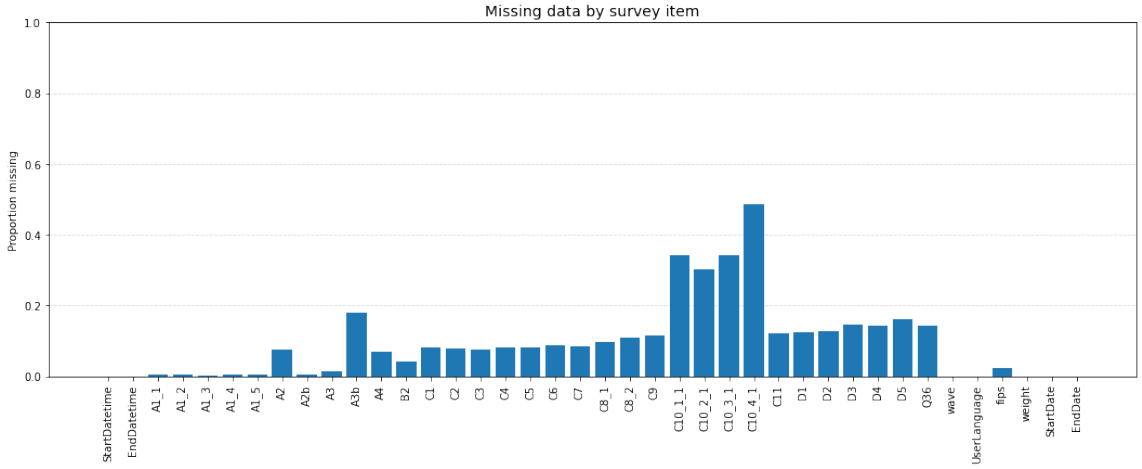

Figure 3.2: Features That Have Fewer Missing Values. These Features Form the Top 28 Features of the Symptom Survey Dataset

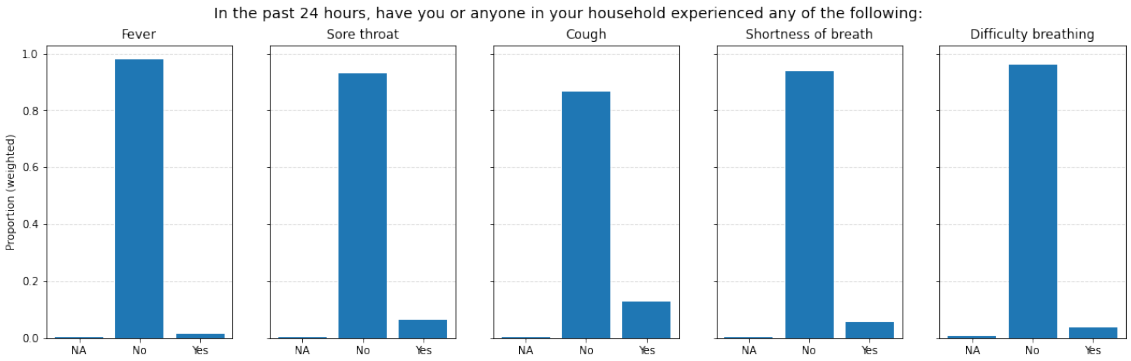

Figure 3.3: Distribution of Top 5 Symptoms from the Self-reported Surveys

tested for COVID-19 and get 951,851 responses. The dataset is split into train and test by taking a stratified split, in which the train and test set have the same amount of imbalance. The assumption here is that the disease is not spreading too fast to make the dataset balanced soon. In a sense, the stratified split mimics the real-world class imbalance in the COVID-19 infected patients. The symptoms from the survey data are converted into binary values to indicate their presence or absence. Discrete choice features that rank the severity of fever, cough, shortness of breath from 1 to 5 are converted into binary values with a mean value threshold of 2.5. If-then type questions are skipped to include only the top-level features. For example, the survey asks for the fever temperature only if the respondent has a fever. In this case, the temperature is ignored and only the presence of fever is used as a symptom feature by the model.

In an effort to remove spurious responses, we filter out a few responses based on the answer value to certain questions. For example, a response stating that there are more than 10 sick people in households is filtered out based on the fact that the average family size in the USA is 3.15. Even a family with three generations would mostly have only 10 people at most. To simplify the dataset, we ignore survey responses showing symptoms more than 24 days as they might contain unrealistically

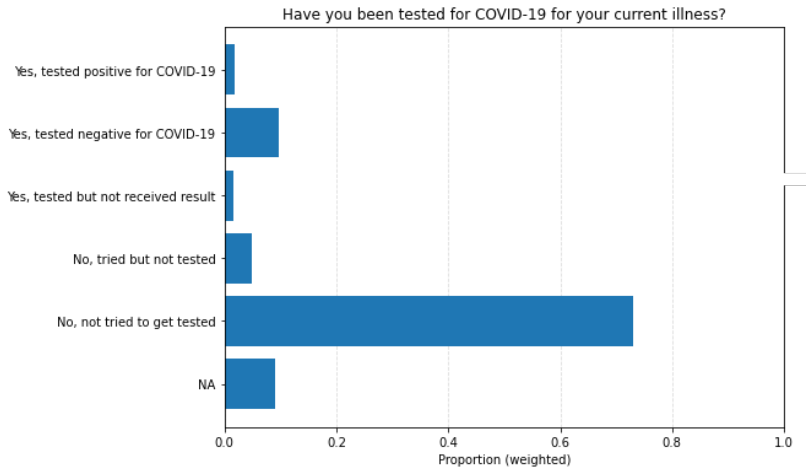

Figure 3.4: Distribution of COVID-19 Rt-pcr Test Result Responses from the Survey

large values. However, only a small portion of the population shows symptoms for more than 24 days. Similarly, we overlook responses that record meeting more than 1000 people in a social gathering. In few cases, the numeric responses contained negative values for "how many" type questions and we ignore those responses as well.

The resultant dataset contains generic user demographics such as name, age, diagnostic information and the COVID prediction label. This pruned dataset contains about 864,154 responses with approximately 84% COVID-19 negative and 16% COVID-19 positive labels.

Chapter 4

EXPERIMENTAL RESULTS

4.1 Applying XGBoost.α to the FB-CMU dataset

This chapter compares α-loss and SOTA XGBoost log-loss results and highlights the effectiveness of α-loss in class imbalance and noisy settings that could be prevalent in an online survey. Given that the dataset is highly imbalanced, the right metrics to compare would be F1 score, Fbeta(beta = 2) score and Area Under the Receiver Operating Characteristic curve (AUROC) score.

The F1 score is measured in terms of precision and recall. Precision quantifies the number of correct positive predictions made. Recall quantifies the number of correct positive predictions made out of all positive predictions that could have been made. As F1 score is the harmonic mean between precision and recall, it punishes the extreme values, such as low precision or low recall.

$$\text{F1} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

We use the Fbeta score to give higher importance to recall directly and to false negatives indirectly. In our scenario, a false negative represents classifying an infected patient as not infected. Fbeta score considers recall β times more important than precision. This chapter uses F2 and Fbeta(beta=2) interchangeably, to mean the same. As far as the COVID-19 survey dataset is concerned, it is important to reduce the false negatives because the model should not predict an individual as not infected when they are infected in reality.

$$\text{Fbeta} = \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{(\beta^2 \times \text{precision}) + \text{recall}}$$

As part of this thesis, we extend the open-sourced XGBoost source code to implement α-loss in C++. Custom loss function implementation in C++ helps tackle the floating point precision limitations with Python. In our experiments with α-loss implementation, in both Python and C++, the latter had a higher precision value and hence gave better results. This new loss function is implemented as per the standards of XGBoost. Testcases were written using the C++ Google tests framework to verify the validity of the loss function. The loss function can be invoked while training by setting the *objective* to *"binary:alpha_logistic"*. The α value is set by *alpha_value* hyperparameter. Implementing the α-loss in XGBoost C++ backend gives the advantage of utilizing the hardware and software optimizations of XGBoost.

The native XGBoost log-loss uses the hyperparameters listed in Table 4.1. With the log-loss, we obtain the best F1 score with a learning rate of 0.71, L2 regularization of 10^{-3} and a maximum tree depth of 2. The optimal learning rate (0.71) was identified from grid search with the search space values ranging from 0.1 to 1.0. a range To provide the fairest comparison, we use the same hyperparameter values with α-loss and only tune over α. In addition to the fixed set of hyperparameters, we use the scale_pos_weight hyperparameter to upweight the minority class samples. As this dataset is highly imbalanced, upweighting aids in giving significant importance to the positive labels.

4.1.1 Comparing Native XGBoost and XGBoost.α in Non-noisy Settings

As described in the previous chapter , we use the preprocessed Facebook-CMU dataset to compare the performance of native XGBoost log-loss and XGBoost.α α-loss. As the reader may recall, the preprocessed dataset uses only the top 8 features, namely, age, gender, presence of fever, cough, shortness of breath, tiredness, aches and loss of smell or taste. In our experiments, we find that $\alpha \in [0.3, 1)$ seems to

Hyperparameter	**Description**	**Value**
objective (logistic regression)	Loss function	binary:logistic
learning_rate	Controls step size	0.71
lambda	L2 regularization	10^{-3}
max_depth	Maximum possible depth of tree	2
eval_metric	Evaluation metric for validation data	aucpr (Area Under Precision Recall curve)
scale_pos_weight	Controls the balance of positive and negative weights	5.09 (optional)

Table 4.1: Optimal Hyperparameters for the Native XGBoost Log-loss

Learning rate	α	F1	F2	AUROC
0.71	1	57.91	50.76	71.97
	0.6	57.91	50.75	71.96

Table 4.2: Comparison of the Performance of Log-loss(α=1) and α-loss($\alpha \neq 1$,) with the Optimal Learning Rate of Log-loss

be doing slightly better than native XGBoost log-loss. However, with an optimal learning rate of 0.71, log-loss has slightly better gains than α-loss as shown in table 4.2. In contrast, if we tune the learning rate with respect to α-loss, $\alpha \neq 1$ performs better as shown in table 4.3. By just varying learning_rate and α, α-loss outperforms log-loss. With other hyperparameter combinations, the gain can potentially be even more significant.

Learning rate	α	F1	F2	AUROC
0.41	0.7	58.08	51.03	72.10
	1	57.86	50.69	71.94

Table 4.3: Comparison of the Performance of Log-loss(α=1) and α-loss($\alpha \neq 1$,) with the Optimal Learning Rate of α-loss

4.2 Comparing Native XGBoost and XGBoost.α in Noisy Settings

Prior work by Sypherd *et al.* [16], who studied α-loss in classification problems, motivates us to restrict $\alpha \in [.8, 4]$ as it is sufficient to handle the label noise. In our exploration, we increase α in a step size of 0.1 for $\alpha \in [.8, 2]$ and a step size of 0.2 for $\alpha \in (2, 4]$. As discussed in Chapter 2, Small α values become sensitive to minority classes due to convexity and large α values become agnostic to the data distribution due to quasi-convexity. We add random noise, with a seed, to the labels and run each experiment 5 times. To generate a fair comparison, the 5 different seed values are repeated across different α to generate the same noise. The test results are averaged across the 5 iterations and the α that generates the maximum score is chosen. Calculation of the relative gain in terms of F1, Fbeta (beta = 2) and AUROC metrics as:

$$\text{relative } X \text{ gain}\% = \frac{|\alpha\text{-loss } X - \text{log-loss } X|}{\text{log-loss } X} \times 100 \tag{4.1}$$

where X can be F1, Fbeta or AUROC metric.

In the following sections, we will dive into the results from different types of experiments.

- **Noisy class labels:** Comparing the results of XGBoost log-loss and XGBoost.α by adding random noise to the train data binary class labels and measure the

performance on clean test data.

  - Stratified Label noise: maintains the imbalance by flipping equal number of positive and negative labels.
  - Non-stratified label noise: improves or worsens the imbalance by randomly flipping the labels.

- **Noisy features:** Comparing the results of XGBoost log-loss and XGBoost.α by adding random noise to the train data binary features train data and measure the performance on clean test data.

4.2.1 Noisy Class Labels

We extend the scenario of adding noise to the class labels, into two sub-categories:

- **Stratified Noising:** The same number of positive and negative labels are flipped in the dataset thus maintaining the original imbalance of 86% negative and 14% positive labels. Since the whole dataset has only 14% positive positive samples, we restrict ourselves to adding noises from 1% to 6% in steps of 1%. In the worst case, the dataset has only 8% real positive labels and 6% noisy positive labels. We randomly select an equal number of train samples from both the positive and negative labels and flip their COVID-19 diagnosis test results. We noise the train data and compare the robustness of log-loss and α-loss with the non-noised test data.

  Theoretically, the conjecture is that $\alpha > 1$ should do better as we tune away from log-loss to make the loss function quasi-convex. In this region, α-loss is less sensitive to outliers/noisy samples. To verify this in practice, we add noise to the COVID-19 survey dataset, as there is a high possibility of encountering spurious responses in a self-reported online survey.

Label Flip %	LL F1	α^* F1	α^*	Rel F1 Gain %
1	57.54	57.58	0.9	0.08
2	56.68	56.80	1.3	0.22
3	55.74	55.76	1.3	0.04
4	54.30	55.27	3.8	1.78
5	51.31	54.26	3.8	5.74
6	43.52	48.75	3.4	12.01

Table 4.4: Stratified Binary Noisy Label Experiment on COVID-19 Survey RT-PCR Results Comparing F1 Metric. Note That Ll F1 and α F1 Stand for Log-loss and α-loss F1 Score for α^*, Respectively. Rel F1 Gain Is Calculated According to Equation 4.1. Also Note That Each Reported F1 Is Averaged over 5 Runs with Different Noise Distributions

In our experiments, we use α^* to denote the optimal α. From tables 4.4, 4.5 and 4.6, it is noticeable that with increase in noise, $\alpha > 1$ values outperform the logistic loss in terms of F1, Fbeta (beta = 2) and AUROC metrics. For very little noise, α slightly less than 1 (0.9) performs better. With an increase in noise, the relative gain increases non-linearly. Fig.4.1 compares the relative gain between the metrics that are used to measure the model's performance. The growth of F2 aligns with our initial goal of reducing the False Negatives by increasing the recall of the model. The gain in α-loss performance is non-monotonic. With more and more noise, the performance of α-loss, or any other loss function for that matter, deteriorates.

$\alpha \neq 1$ favors reducing False Negatives when there is no upweighting on the

Label Flip %	LL F2	α^* F2	α^*	Rel F2 Gain %
1	50.13	50.20	0.9	0.15
2	48.79	48.97	1.3	0.37
3	47.50	47.52	1.3	0.06
4	45.66	47.09	3.8	3.15
5	42.11	45.63	3.8	8.35
6	33.91	39.32	3.4	15.96

Table 4.5: Stratified Binary Noisy Label Experiment on COVID-19 Survey RT-PCR Results Comparing F2 Metric. Note That Ll F2 and α F2 Stand for Log-loss and α-loss Fbeta (Beta = 2) Score for α^*

Label Flip %	LL AUROC	α^* AUROC	α^*	Rel AUROC Gain %
1	71.68	71.71	0.9	0.05
2	71.06	71.14	1.3	0.12
3	70.45	70.46	1.3	0.02
4	69.58	70.23	3.8	0.93
5	67.93	69.57	3.8	2.42
6	64.17	66.62	3.4	3.83

Table 4.6: Stratified Binary Noisy Label Experiment on COVID-19 Survey RT-PCR Results Comparing Auroc Metric. Note That Ll Auroc and α Auroc Stand for Log-loss and α-loss Area under Roc Curve Score for α^*

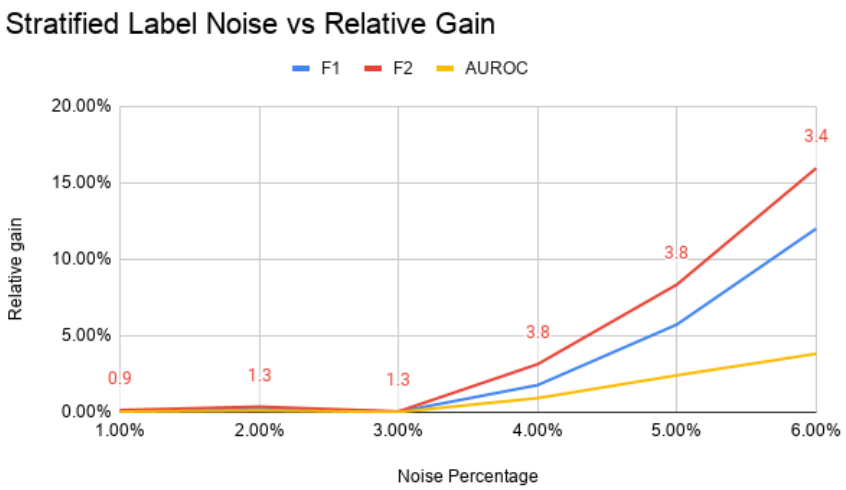

Figure 4.1: Relative Gain Percentage for Metrics F1, F2 and Auroc as a Function of Noise Levels, in a Stratified Noise Setting. Note That the Gain Increases with Increase in Noise

positive classes. With extreme upweighting values, the False positive count on test data is too high and hence the model has a higher alarm rate. With intermediate upweighting values, α favors reducing the False positives, in noisy settings. Table 4.7 compares the relative gain on F1 for both the losses. α^* gives a standard reduced false positive count of 6382 across all noise levels. This is probably the closest that we can get to the optimal Bayes risk with α-loss. Whereas, $\alpha = 1$, gives almost twice that of α^* False Positive count.

- **Non-Stratified Noising:** The labels are chosen and flipped randomly from the whole dataset without maintaining the imbalance. With an increase in noise, the imbalance reduces, as more negative samples tend to be flipped in this highly imbalanced setting. We add noises from 1% to 50%. We randomly select x% of train samples, where x is the noise percentage, and flip the chosen train labels and compare the effectiveness of α-loss with the log-loss. As mentioned

Label Flip %	LL F1	α^* F1	α^*	Rel F1 Gain %
1	57.58	59.37	2.6	3.11
2	57.80	59.37	2.8	2.73
3	57.86	59.37	3.2	2.62
4	57.92	59.37	3.2	2.51
5	58.08	59.37	3.6	2.22
6	58.21	59.37	2.6	2.01
7	58.41	59.37	2.2	1.64

Table 4.7: Stratified Binary Noisy Label Experiment by Upweighting Positive Labels with a Factor of 4

in the stratified noise experiment, the conjecture is that $\alpha > 1$ increases the robustness to noises. From the experiments recorded in tables 4.8, 4.9 and 4.10, similar pattern to that of stratified noise experiments is noticed. The relative gain has a slight fluctuation initially, which could be due to the convoluted case of varying the imbalance and the noise percentage simultaneously. In our stratified noise experiments, only the noise percentage varied, whereas in non-stratified noisy settings, two variables(imbalance and noise) interplay with α to fluctuate the gain. Nevertheless, with more noise, the gain increases as shown in fig.4.2. Similar to the stratified experiment, $\alpha \neq 1$ favors reducing the False positive count with intermediate upweighting, as shown in Table 4.11.

Label Flip %	LL F1	α^* F1	α^*	Rel F1 Gain %
1	57.89	57.91	0.8	0.02
2.5	57.89	58.01	1.2	0.20
5	57.93	58.00	1.3	0.13
7.5	57.88	57.98	1.2	0.17
10	57.87	57.97	1.3	0.17
15	57.86	58.00	1.4	0.25
20	57.79	58.02	1.5	0.40
25	57.83	58.02	1.8	0.32
30	57.84	57.86	0.4	0.03
35	57.77	57.83	1.2	0.10
40	57.64	57.65	1.5	0.01
50	26.22	27.55	3.8	5.09

Table 4.8: Non-stratified Binary Noisy Label Experiment on COVID-19 Survey RT-PCR Results Comparing F1 Metric

4.2.2 Noisy Features

We add noise to the features to mimic the real-world scenario in which survey respondents may enter erratic values to the questions. We use two hyperparameters to control the noise addition:

- row_noise: This hyperparameter controls the number of samples to which noise is added randomly. It varies from 1% to 50%.

- feature_noise: A respondent can answer few or all questions dishonestly. This

Label Flip %	LL F2	α^* F2	α^*	Rel F2 Gain %
1	50.74	50.76	0.8	0.04
2.5	50.74	50.95	1.2	0.43
5	50.79	50.95	1.3	0.32
7.5	50.71	50.93	1.2	0.43
10	50.72	50.89	1.3	0.33
15	50.72	50.95	1.4	0.45
20	50.63	50.99	1.5	0.71
25	50.71	51.11	1.8	0.79
30	50.72	50.76	0.4	0.08
35	50.66	50.76	1.2	0.20
40	50.56	50.58	1.5	0.05
50	38.05	40.14	3.8	5.48

Table 4.9: Non-stratified Binary Noisy Label Experiment on COVID-19 Survey RT-PCR Results Comparing F2 Metric

hyperparameter is a probability measure that controls the addition of noise to each feature. Feature_noise can take values ranging from 0.25 (meaning some features are noisy) to 1 (meaning all features are noisy), representing the probability of flipping each feature.

We add noise to each feature depending upon the threshold value of feature_noise hyperparameter M%. We generate a random noise probability value for each feature, say m_1 to m_8, and flip only those features, m_i, whose values are less than M. In our experiment with noisy features, we compare the F1 scores of XGBoost log-loss and

Label Flip %	LL AUROC	α^* AUROC	α^*	Rel AUROC Gain %
1	71.96	71.97	0.8	0.01
2.5	71.96	72.06	1.2	0.14
5	71.98	72.05	1.3	0.10
7.5	71.95	72.04	1.2	0.13
10	71.95	72.03	1.3	0.11
15	71.94	72.05	1.4	0.15
20	71.90	72.07	1.5	0.24
25	71.94	72.11	1.8	0.25
30	71.94	71.96	0.4	0.02
35	71.91	71.96	1.2	0.06
40	71.85	71.86	1.5	0.02
50	51.18	52.68	3.8	2.92

Table 4.10: Non-stratified Binary Noisy Label Experiment on COVID-19 Survey RT-PCR Results Comparing Auroc Metric

Label Flip %	LL F1	α^* F1	α^*	Rel F1 Gain %
1	58.64	59.37	1.9	1.25
2.5	58.47	59.37	2.4	1.54
5	58.15	59.37	2.8	2.10
7.5	57.67	59.37	3.2	2.96

Table 4.11: Non-stratified Binary Noisy Label Experiment by Upweighting Positive Labels with a Factor of 3

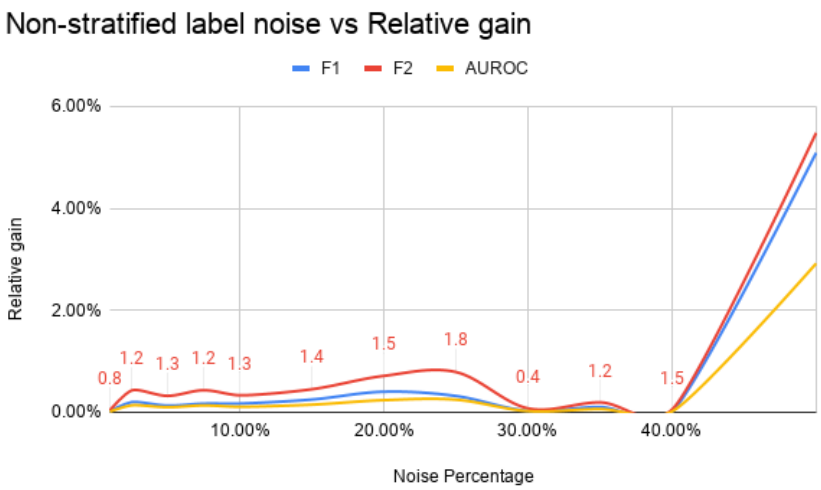

Figure 4.2: Relative Gain Percentage for Metrics F1, F2 and Auroc as a Function of Noise Levels, in a Non-stratified Noise Setting. For Less Noise, the Gain Fluctuates Initially Due to the Interplay of Noise and Imbalance with α. As Noise Increases above 40%, the Relative Gain Increases

α-loss for different feature_noise probability values. From Figures, 4.3, 4.4, 4.5, and 4.6, it is noticeable that with increase in noise, the relative gain in terms of F1 score increases for $\alpha > 1$. It is also important to note, α slightly greater than 1, performs better in most of the cases.

The highlighted experimental results from the imbalance, label noise and feature noise settings suggest that, α-loss yields significantly better test results when compared to the native log-loss. For the noisy label experiments, we find that most $\alpha^* > 1$ give better gains. Similarly, for the noisy feature experiments, we find that all $\alpha^* > 1$ give better gains. In imbalanced setting, we find that most $\alpha^* < 1$ give better gains. Consequently, α-loss can be utilized to build a robust classification algorithm in healthcare online survey scenarios, where the dataset could be noisy and imbalanced.

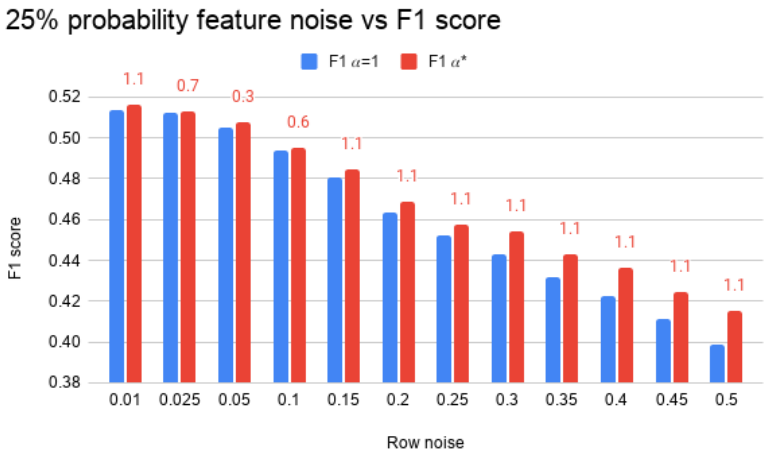

Figure 4.3: Comparison of F1 Scores of the Native XGBoost Log-loss and XGBoost.α for 25% Feature_noise Value. Note That the Gain Increases with Increase in the Amount of Noisy Samples

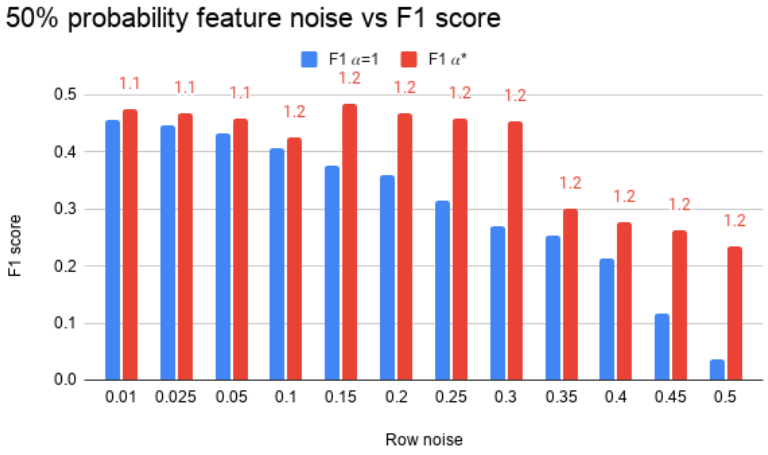

Figure 4.4: Comparison of F1 Scores of the Native XGBoost Log-loss and XGBoost.α for 50% Feature_noise Value

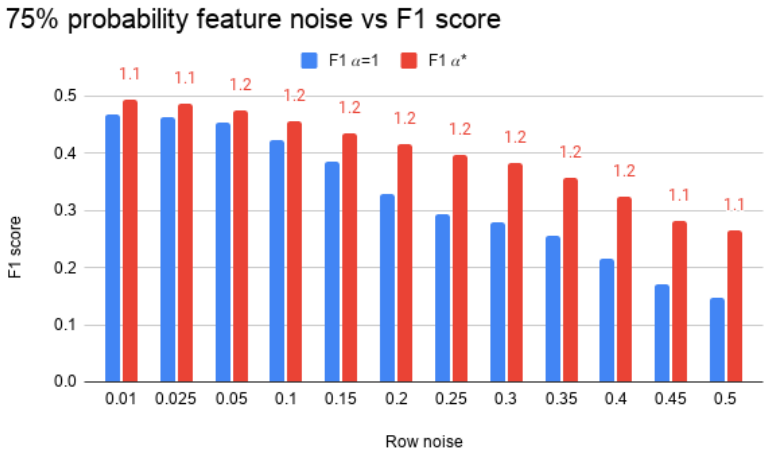

Figure 4.5: Comparison of F1 Scores of the Native XGBoost Log-loss and XGBoost.α for 75% Feature_noise Value

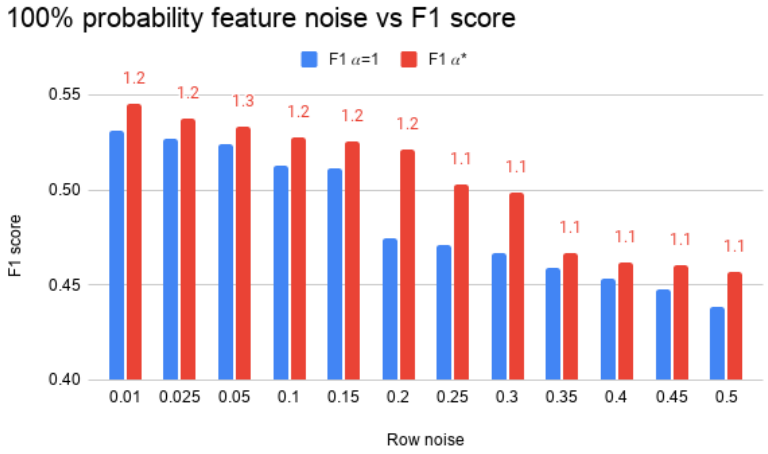

Figure 4.6: Comparison of F1 Scores of the Native XGBoost Log-loss and XGBoost.α for 100% Feature_noise Value

Chapter 5

CONCLUSION AND FUTURE ENHANCEMENTS

This work utilizes the tunable nature of α-loss to improve the robustness and performance of the XGBoost classification algorithm. It is important to note that, just by tuning over α, the results are significantly better, despite using the fixed set of hyperparameters that are optimal to native XGBoost log-loss. By tuning the other hyperparameters together, it is highly possible to get far more better and robust results.

As future work, studying the performance of α-loss in other imbalanced healthcare datasets can be explored. The influence of other XGBoost hyperparameters and α can be studied deeper. The impact of the dimension of the dataset in the loss of accuracy when the noise is introduced should be analyzed. The performance of α-loss with feature noises needs to be explored.

From the experiments, it is evident that the models trained with α-loss can be more robust to outliers than the SOTA XGBoost. Thus, we argue that α-loss can be used in highly imbalanced settings to improve robustness and performance of classification models.

REFERENCES

[1] J. Liao, O. Kosut, L. Sankar, and F. P. Calmon, "A tunable measure for information leakage," in *2018 IEEE International Symposium on Information Theory (ISIT)*, 2018, pp. 701–705.

[2] V. J. Munster, M. Koopmans, N. van Doremalen, D. van Riel, and E. de Wit, "A Novel Coronavirus Emerging in China — Key Questions for Impact Assessment," *New England Journal of Medicine*, vol. 382, no. 8, pp. 692–694, 2020, pMID: 31978293. [Online]. Available: https://doi.org/10.1056/NEJMp2000929

[3] X. Ren, Y. Liu, H. Chen, W. Liu, Z. Guo, y. zhang, C. Chen, J. Zhou, Q. Xiao, G.-M. Jiang, and H. Shan, "Application and optimization of RT-PCR in diagnosis of SARS-CoV-2 Infection (2/25/2020)," 2020.

[4] N. Sharma, P. Krishnan, R. Kumar, S. Ramoji, S. R. Chetupalli, N. R., P. K. Ghosh, and S. Ganapathy, "Coswara — A Database of Breathing, Cough, and Voice Sounds for COVID-19 Diagnosis," *Interspeech 2020*, Oct 2020. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2768

[5] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002200009791504X

[6] P. M. Long and R. Servedio, "Random classification noise defeats all convex potential boosters," *Machine Learning*, vol. 78, pp. 287–304, 2008.

[7] T. Sypherd, M. Diaz, L. Sankar, and P. Kairouz, "A Tunable Loss Function for Binary Classification," in *2019 IEEE International Symposium on Information Theory (ISIT)*, 2019, pp. 2479–2483.

[8] D. DeCapprio, J. Gartner, C. J. McCall, T. Burgess, S. Kothari, and S. Sayed, "Building a COVID-19 Vulnerability Index," *medRxiv*, 2020. [Online]. Available: https://www.medrxiv.org/content/early/2020/03/30/2020.03.16.20036723

[9] C. Menni, A. Valdes, M. Freidin, C. Sudre, L. Nguyen, D. Drew, S. Ganesh, T. Varsavsky, M. Cardoso, J. El-Sayed Moustafa, A. Visconti, P. Hysi, R. Bowyer, M. Mangino, M. Falchi, J. Wolf, S. Ourselin, A. Chan, C. Steves, and T. Spector, "Real-time tracking of self-reported symptoms to predict potential COVID-19," *Nat Med. 2020 Jul;26(7)*, pp. 1037–1040, 2020.

[10] "Delphi Group (2021). COVID Symptom Survey." https://cmu-delphi.github.io/delphi-epidata/symptom-survey/, 2020.

[11] T. Sypherd, M. Diaz, L. Sankar, and G. Dasarathy, "On the -loss Landscape in the Logistic Model," in *2020 IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 2700–2705.

[12] R. E. Schapire, "The strength of weak learnability," *Machine learning*, vol. 5, no. 2, pp. 197–227, 1990.

[13] Y. Freund, "Boosting a weak learning algorithm by majority," *Information and Computation*, vol. 121, no. 2, pp. 256–285, 1995. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0890540185711364

[14] T. Chen and C. Guestrin, "XGBoost," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug 2016. [Online]. Available: http://dx.doi.org/10.1145/2939672.2939785

[15] F. Kreuter, N. Barkay, A. Bilinski, A. Bradford, S. Chiu, R. Eliat, J. Fan, T. Galili, D. Haimovich, B. Kim, S. LaRocca, Y. Li, K. Morris, S. Presser, T. Sarig, J. A. Salomon, K. Stewart, E. A. Stuart, and R. Tibshirani, "Partnering with a global platform to inform research and public policy making," *Survey Research Methods*, vol. 14, no. 2, pp. 159–163, Jun. 2020. [Online]. Available: https://ojs.ub.uni-konstanz.de/srm/article/view/7761

[16] T. Sypherd, M. Diaz, J. K. Cava, G. Dasarathy, P. Kairouz, and L. Sankar, "A Tunable Loss Function for Robust Classification: Calibration, Landscape, and Generalization," 2021, https://arxiv.org/abs/1906.02314.