Binary Classification from Positive-Confidence Data

Group 7:

Hardik Chugh, Shashank Dubey, Samuel Matloob, Matthew Weston

***Abstract* - Ishida et al. [1] presented a method of modeling binary classification models using only positive data with confidence. Using their positive-confidence model, we attempted to replicate some of their experiments, modify their code to have better general use, and see how it performs with other datasets. We tested the effectiveness of Pconf classifier for large training datasets, minimum confidence values, and high dimensional data. In this report, we'll summarize our findings and provide some advantages and disadvantages to using the Pconf classifier, providing results of a few experiments, some of which replicated the experiments in [1].**

I. Introduction

Machine learning requires labeled data to successfully build models and accurately classify data. However, in the real world, it becomes expensive or impractical to fully label datasets. Weakly supervised learning is a field of study meant to address this problem [2]. By reducing the amount of labeling needed to build a model, the cost to make an accurate model goes down.

Ishida et al [1] proposed a weak supervised model called the Positive-Confidence classification model (Pconf) which builds a binary classifier using only positive data equipped with a confidence value. This model would allow data processing in scenarios in which gathering data from one side of the binary classifier is impractical, impossible, or illegal. For example, a company wanting to predict the likelihood a customer would buy a product from them rather than their competitor would only have access to their own data. Another example would be an app developer needing to predict whether a user will continue to use their app or not. Depending on privacy and data laws, the developers would not be able to legally use data collected from people who are no longer using their app.

II. Related Work

Pconf classification, according to Ishida et al. [1], is related to one-class classification. Many researchers have studied the one-class classification and tackled it from different angles. For example, Tax and Duin [3] suggested a novel approach to identify and exclude outliers using support vector domain description. Similarly, Schölkopf et al. [4] used an extension of the support vector algorithm to detect and exclude anomalies in the data. On the other hand, Sugiyama et al. [5] used a computational approach to maximize information based on squared-loss variants of mutual information, while Breunig et al. [6] used a density-based method for finding outliers.

Furthermore, Pconf classification can be related to Positive-Unlabeled (PU) classification, in which the dataset has some labeled positive points and others that are not labeled. Plessis et al. [7, 8, 9] and Blanchard et al. [10, 11] have studied PU from the perspective of use and implementation difficulties. However, according to [1], what differentiates Pconf classification from the PU classification is that the former does not need to estimate the class-prior probability. The idea of using confidence scores is not a new concept. Bach et al. [12], Gandrabur et al. [13], and many others have used confidence scores to improve the model's effectiveness in language processing and other fields. However, what stood out to us, was Liu et al. work [14], in which they tried to filter out label outliers by attaching a confidence score to each label that they calculated through their suggested framework.

III. Objective

In this project, we studied how Pconf works, attempted to replicate some of the experiments done by Ishida et al. [1], ran experiments that used different datasets, and compared the Pconf classifier accuracy with other known classifiers, such as Random Forest.

We started our work using the authors' code in [15], which included a function to generate uniformly randomized data, which was used in [1] and, hence in our first experiments as well.

To give Pconf classifier a good test, we needed to train the Pconf classifier on a large dataset with bigger dimensionality than the authors of [1] have used in their experiment (their dataset included only two features). Therefore, in later experiments, we used the Spaceship Titanic dataset [16] (from competition-2) and a phishing dataset [17], measured the accuracy of Pconf classifier, and compared it with other classifiers such as Random Forest and Adaboost.

IV. DEFINING CONFIDENCE

Sometimes, we can only get a dataset that includes one label (positive label), for example, if we're building a dog/wolf classifier but only have dog pictures. In such scenarios, we can assign a number to each data point, between zero and one, that reflects how confident we are, given the features, to give the data point a positive label. That number is the confidence score that plays a critical role in training the Pconf classifier.

We can obtain the confidence score differently, depending on the situation. For example, we can survey customers and ask them to rate a product, and we can use the rating as the confidence score. We can assign the confidence score manually in the dog/wolf example. We also can use statistical methods if we have enough data. Ishida et al. [1] used a statistical approach to calculate the confidence score for each data point in their experiments. They fed their algorithm with the desired covariance value for each feature to generate a random uniformed dataset. The covariance values would then be used for the confidence score calculations alongside the probability of positive data in the generated data.

In our experiments (for example, the Spaceship Titanic experiment), since we already had a data set, we calculated the covariance values for each feature and used them for the confidence score calculation.

V. TECHNICAL DETAILS

With ordinary binary classifiers, positive and negative data are given to the model to learn the pattern and to be able to classify/label unseen data in the future. However, it's not always possible to collect data that includes all the possible labels to train the model, which is where the Pconf model comes into play. The Pconf classifier is a binary classification that accepts one labeled data with a confidence score associated with each datapoint.

The goal of Pconf is to train a binary classifier g(x) which reduces classification risk. Normally, one would use the following formula:

$$R(g) = \mathrm{E}_{p(x,y)}[\ell(yg(x))]$$

where $g(x)\colon \mathbb{R}^d \to \mathbb{R}$ (is the classifier function), $x \in \mathbb{R}^d, y \in \{+1, -1\}$, $\mathrm{E}_{p(x,y)}$ is the expectation over $p(x,y)$ and $\ell(z)$ is the loss function. However, because Pconf only uses positive data with confidence, this equation does not work (y is always +1).

Instead, the following Empirical Risk Minimization (ERM) framework was proposed [1]:

$$\min_{g} \sum_{i=1}^{n} [\ell(g(x_i)) + \frac{1-r_i}{r_i}\ell(-g(x_i)]$$

where x_i is an independently drawn positive pattern and r_i is a confidence score given by $r_i = p(y = +1|x_i)$. The ERM function finds g that gives the minimal loss given the confidence scores, which is why the model works.

VI. Overview of Proposed Approach

A. Experiment replication

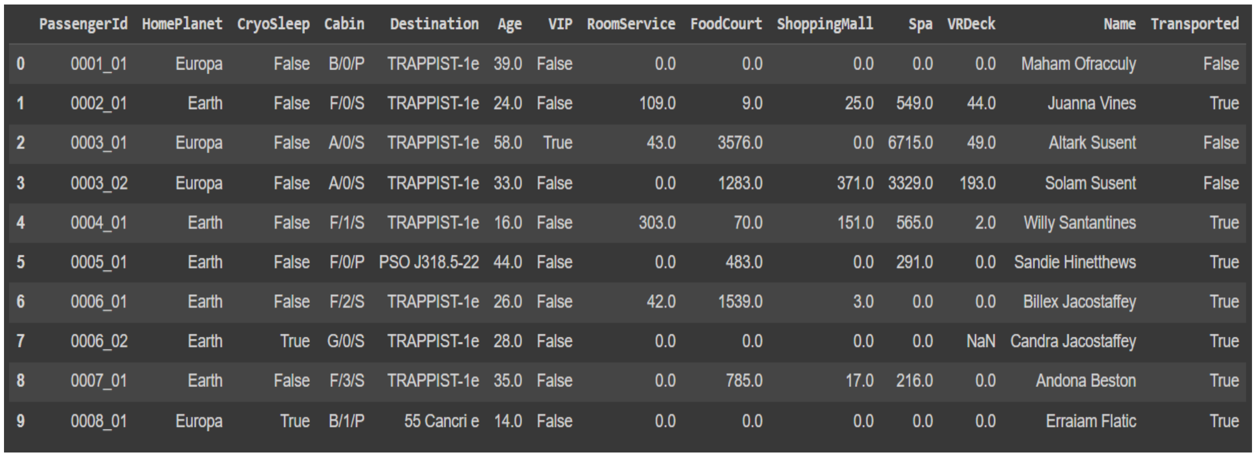
We first attempted to replicate the experiments from Ishida et al. [1] using randomized data. In their experiment, they generated four different two-dimensional Gaussian distributions with means μ_+ and μ_- and covariance matrices Σ_+ and Σ_- for p(x|y = 1) and p(x|y = -1) respectively (Table 1). The parameters for each setup were included in the paper and we used the same parameters.

	μ_+	μ_-	Σ_+	Σ_-
Setup A	$[0 \quad 0]^T$	$[-2 \quad 5]^T$	$\begin{bmatrix} 7 & -6 \\ -6 & 7 \end{bmatrix}$	$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$
Setup B	$[0 \quad 0]^T$	$[0 \quad 4]^T$	$\begin{bmatrix} 5 & 3 \\ 3 & 5 \end{bmatrix}$	$\begin{bmatrix} 5 & -3 \\ -3 & 5 \end{bmatrix}$
Setup C	$[0 \quad 0]^T$	$[0 \quad 8]^T$	$\begin{bmatrix} 7 & -6 \\ -6 & 7 \end{bmatrix}$	$\begin{bmatrix} 7 & 6 \\ 6 & 7 \end{bmatrix}$
Setup D	$[0 \quad 0]^T$	$[0 \quad 4]^T$	$\begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Table 1: Parameters used to generate randomized data

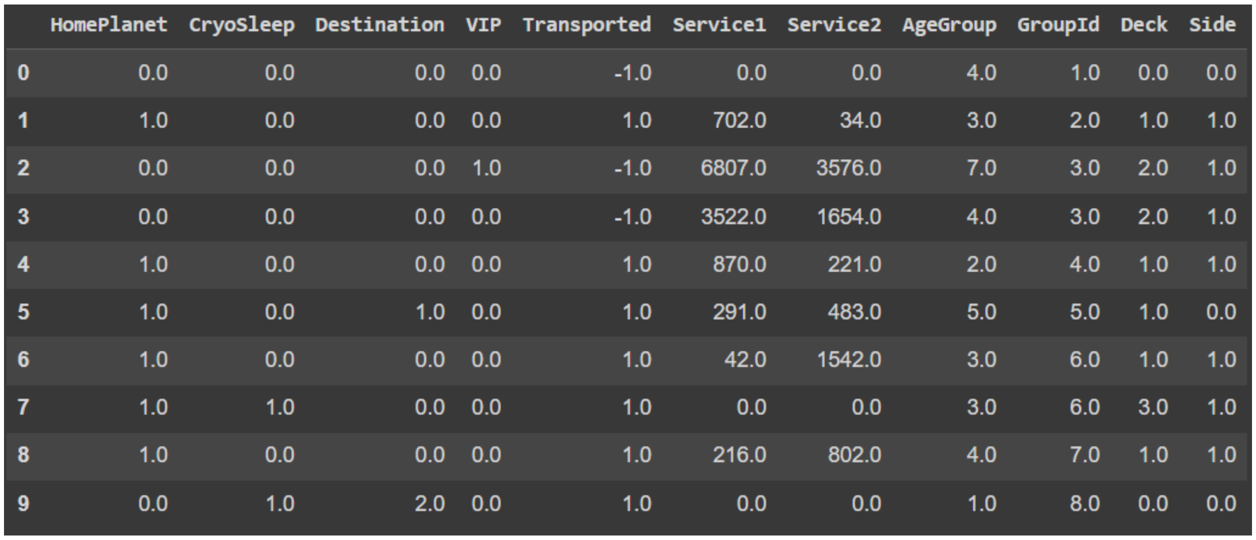
B. Spaceship Titanic dataset

The Spaceship Titanic [15] dataset is a machine learning challenge offered by Kaggle (see figure 1). It involves building a binary classifier, meaning Pconf should be able to build an accurate model. The training dataset contains ~8,700 entries which would allow us to test whether Pconf can handle high datasets when building a model. Each entry also has 14 entries, with some fields being blank. After processing it, we had ~4,500 positive entries (Positive being defined as Transported = true) for the Pconf model while keeping the same number of entries for the other models. Each entry had 10 parameters. The code used to process it can be found at https://github.com/sm1248/CAP5610/blob/main/PConf_SpaceTitanic (see figure 2).



	PassengerId	HomePlanet	CryoSleep	Cabin	Destination	Age	VIP	RoomService	FoodCourt	ShoppingMall	Spa	VRDeck	Name	Transported
0	0001_01	Europa	False	B/0/P	TRAPPIST-1e	39.0	False	0.0	0.0	0.0	0.0	0.0	Maham Ofracculy	False
1	0002_01	Earth	False	F/0/S	TRAPPIST-1e	24.0	False	109.0	9.0	25.0	549.0	44.0	Juanna Vines	True
2	0003_01	Europa	False	A/0/S	TRAPPIST-1e	58.0	True	43.0	3576.0	0.0	6715.0	49.0	Altark Susent	False
3	0003_02	Europa	False	A/0/S	TRAPPIST-1e	33.0	False	0.0	1283.0	371.0	3329.0	193.0	Solam Susent	False
4	0004_01	Earth	False	F/1/S	TRAPPIST-1e	16.0	False	303.0	70.0	151.0	565.0	2.0	Willy Santantines	True
5	0005_01	Earth	False	F/0/P	PSO J318.5-22	44.0	False	0.0	483.0	0.0	291.0	0.0	Sandie Hinetthews	True
6	0006_01	Earth	False	F/2/S	TRAPPIST-1e	26.0	False	42.0	1539.0	3.0	0.0	0.0	Billex Jacostaffey	True
7	0006_02	Earth	True	G/0/S	TRAPPIST-1e	28.0	False	0.0	0.0	0.0	0.0	NaN	Candra Jacostaffey	True
8	0007_01	Earth	False	F/3/S	TRAPPIST-1e	35.0	False	0.0	785.0	17.0	216.0	0.0	Andona Beston	True
9	0008_01	Europa	True	B/1/P	55 Cancri e	14.0	False	0.0	0.0	0.0	0.0	0.0	Erraiam Flatic	True

Figure 1: Shape of Training Data before preprocessing - (8693, 14)



	HomePlanet	CryoSleep	Destination	VIP	Transported	Service1	Service2	AgeGroup	GroupId	Deck	Side
0	0.0	0.0	0.0	0.0	-1.0	0.0	0.0	4.0	1.0	0.0	0.0
1	1.0	0.0	0.0	0.0	1.0	702.0	34.0	3.0	2.0	1.0	1.0
2	0.0	0.0	0.0	1.0	-1.0	6807.0	3576.0	7.0	3.0	2.0	1.0
3	0.0	0.0	0.0	0.0	-1.0	3522.0	1654.0	4.0	3.0	2.0	1.0
4	1.0	0.0	0.0	0.0	1.0	870.0	221.0	2.0	4.0	1.0	1.0
5	1.0	0.0	1.0	0.0	1.0	291.0	483.0	5.0	5.0	1.0	0.0
6	1.0	0.0	0.0	0.0	1.0	42.0	1542.0	3.0	6.0	1.0	1.0
7	1.0	1.0	0.0	0.0	1.0	0.0	0.0	3.0	6.0	3.0	1.0
8	1.0	0.0	0.0	0.0	1.0	216.0	802.0	4.0	7.0	1.0	1.0
9	0.0	1.0	2.0	0.0	1.0	0.0	0.0	1.0	8.0	0.0	0.0

Figure 2: Shape of Training Data after preprocessing - (8693, 11)

While building a model, we discovered that entries that generated extremely low confidence scores reduced the accuracy or even made it impossible to make a classification model with Pconf. We therefore further filtered out training data with confidence < .085, leaving us with a training dataset with ~3,400 entries.

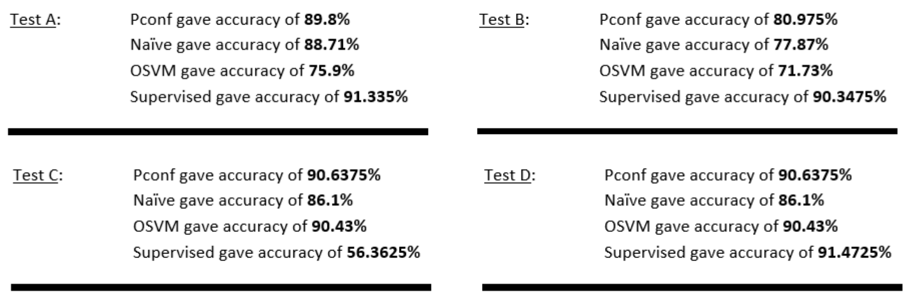
C. Phishing website dataset

The phishing dataset [16] contains 8,000 entries with 48 parameters. 4,000 entries are phishing websites, while the other 4,000 are non-phishing. The parameters include characteristics like URL length, existence of certain characters in the URL, and popup windows in the site. Of the 48 parameters, 29 were kept after preprocessing. The Pconf training data set retained ~2,000 entries after removing negative data and data with low confidence. Positive data was defined as the website being a phishing site.

The code we used to preprocess can be found at https://github.com/Har1743/P_Conf_Project_Machine_Learning/blob/main/Phishing_web.ipynb.

D. Results

We first compared Pconf's accuracy using the **randomized datasets** to Naïve, OSVM, and Fully- Supervised models.



Test A: Pconf gave accuracy of **89.8%**
Naïve gave accuracy of **88.71%**
OSVM gave accuracy of **75.9%**
Supervised gave accuracy of **91.335%**

Test B: Pconf gave accuracy of **80.975%**
Naïve gave accuracy of **77.87%**
OSVM gave accuracy of **71.73%**
Supervised gave accuracy of **90.3475%**

Test C: Pconf gave accuracy of **90.6375%**
Naïve gave accuracy of **86.1%**
OSVM gave accuracy of **90.43%**
Supervised gave accuracy of **56.3625%**

Test D: Pconf gave accuracy of **90.6375%**
Naïve gave accuracy of **86.1%**
OSVM gave accuracy of **90.43%**
Supervised gave accuracy of **91.4725%**

Figure 3: Accuracy of each model

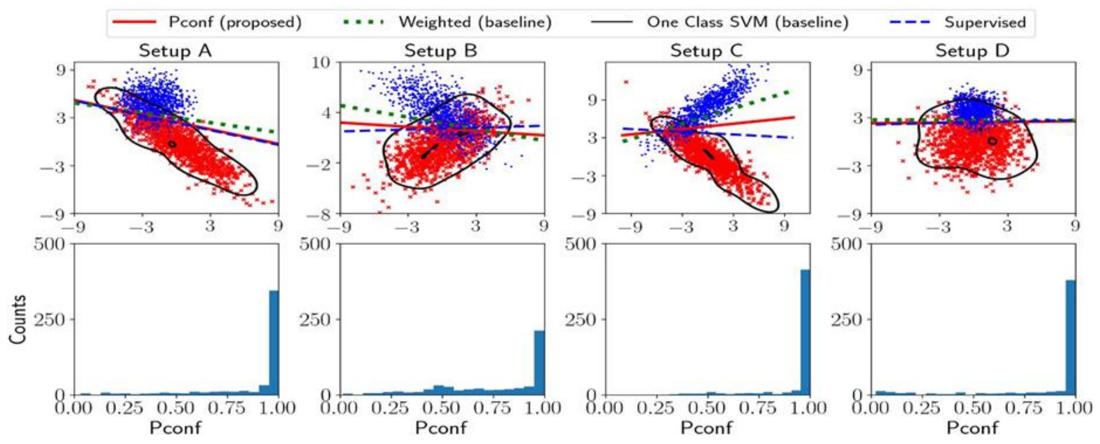


Figure 4: Visual Display of the models

As seen in Fig. 4 and Fig. 5, Pconf was highly accurate and consistent across each setup. Pconf's results are comparably equal to other classifier models as well.

Second, we used the processed **Spaceship Titanic dataset** and compared Pconf's results to Random Forest's over several trials.

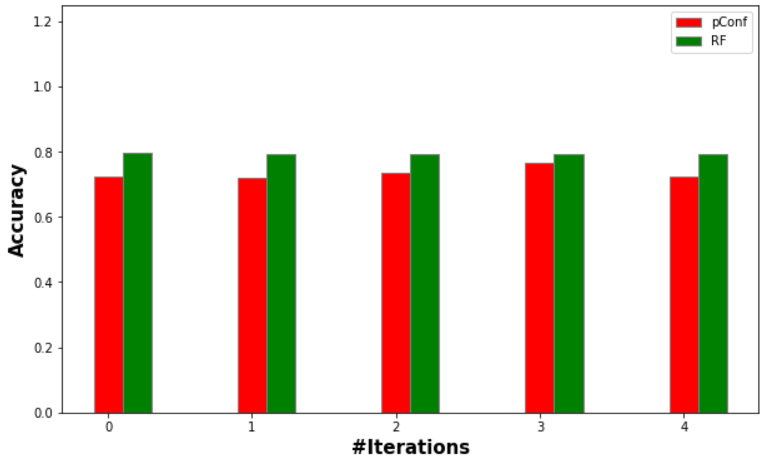
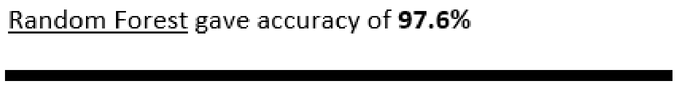


Figure 5: Pconf and Random Forest over several trials

While Random Forest tended to perform better (79.5%), Pconf was able to generate similarly accurate models a 76.5% (Figure 6).

Finally, we used the **phishing dataset** to test Pconf against Random Forest.

Pconf gave accuracy of **74.25%**


Random Forest gave accuracy of **97.6%**

In this case, Random Forest performed far better than Pconf. One issue we experienced with Pconf however is the generated models could vary greatly in accuracy (Figure 8). We speculated this could be because the dataset has a high number of parameters, so we reduced the phishing training dataset to have only 10 parameters. With 10 parameters, Pconf was able to generate models with more consistent accuracy (Figure 7). Therefore, we conclude Pconf is less stable as the number of parameters increases.

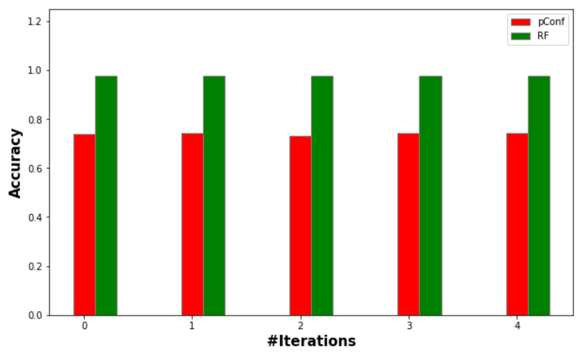


Figure 7: PConf vs Random Forest with 10

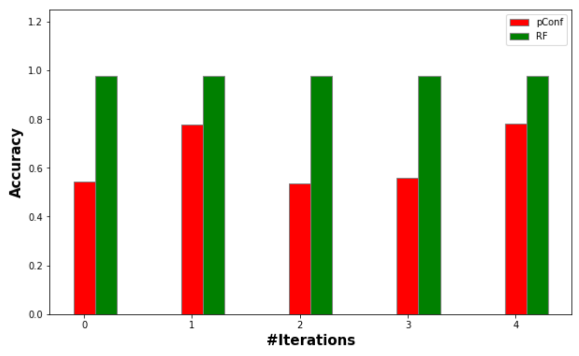


Figure 6: Pconf vs Random Forest with 29

VII. Conclusions

Confidence scores and dimensionality require special consideration when building a model, while dataset size is not an issue. The model will not function correctly if the confidence scores in the training set are too low (typically less than .01), and the model will be less consistent as parameters increase. High dimensionality would have to be fixed in preprocessing, but we enhanced the original source code to protect against low confidence scores.

As a final point, it's important to note that the experiments conducted in [1] and by us may not fully exploit the potential of the Pconf classifier. When the datasets contained both positive and negative data, we showed that Pconf and several other classifier models performed comparably. However, Pconf doesn't require negative data to build a reliable model. It’s possible to take the training dataset used to build a Random Forest or OSVM binary classifier and convert it into a training dataset for a Pconf classifier, but the opposite is not true.

Our updated version of Pconf can be accessed at: https://github.com/sm1248/CAP5610/blob/main/PConf_UpdatedCode.

References

[1] Ishida, T., Niu, G., & Sugiyama, M. (2018). Binary classification from positive-confidence data. *Advances in neural information processing systems*, *31*.

[2] Zhou, Zhi-Hua (2018). A Brief Introduction to Weakly Supervised Learning. *National Science Review*. Volume 5, Issue 1, January 2018, Pages 44–53, https://doi.org/10.1093/nsr/nwx106

[3] D. Tax and R. Duin. Support vector domain description. In *Pattern Recognition Letters,* 1999.

[4] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural computation, 13*(7), 1443-1471.

[5] Sugiyama, M., Niu, G., Yamada, M., Kimura, M., & Hachiya, H. (2014). Information-maximization clustering based on squared-loss mutual information. *Neural Computation*, *26*(1), 84-131.

[6] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. *SIGMOD Rec. 29, 2* (June 2000), 93–104. https://doi.org/10.1145/335191.335388

[7] Marthinus C. du Plessis, Gang Niu, and Masashi Sugiyama. 2014. Analysis of learning from positive and unlabeled data. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'14). MIT Press, Cambridge, MA, USA, 703–711.

[8] Marthinus Christoffel Du Plessis, Gang Niu, and Masashi Sugiyama. 2015. Convex formulation for learning from positive and unlabeled data. In Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15). JMLR.org, 1386–1394.

[9] Ryuichi Kiryo, Gang Niu, Marthinus C. du Plessis, and Masashi Sugiyama. 2017. Positive-unlabeled learning with non-negative risk estimator. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 1674–1684.

[10] Scott, C., & Blanchard, G. (2009, April). Novelty detection: Unlabeled data definitely help. In *Artificial intelligence and statistics* (pp. 464-471). PMLR.

[11] Blanchard, G., Lee, G., & Scott, C. (2010). Semi-supervised novelty detection. *The Journal of Machine Learning Research*, *11*, 2973-3009.

[12] Nguyen Bach, Qin Gao, and Stephan Vogel. 2008. Improving word alignment with language model based confidence scores. In Proceedings of the Third Workshop on Statistical Machine Translation (StatMT '08). Association for Computational Linguistics, USA, 151–154.

[13] Simona Gandrabur, George Foster, and Guy Lapalme. 2006. Confidence estimation for NLP applications. ACM Trans. Speech Lang. Process. 3, 3 (October 2006), 1–29. https://doi.org/10.1145/1177055.1177057

[14] Liu, Y. P., Xu, N., Zhang, Y., & Geng, X. (2021, January). Label distribution for learning with noisy labels. In Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence (pp. 2568-2574).

[15] Takashiishida. (n.d.). Takashiishida/PCONF: Code for the paper "binary classification from positive-confidence data". Retrieved November 30, 2022, from http://github.com/takashiishida/Pconf

[16] Kaggle. Spaceship Titanic. Retrieved November 2022 from https://www.kaggle.com/competitions/spaceship-Titanic/data

[17] Tiwari, S. 2020. Phishing Dataset for Machine Learning. Retrieved November 2022 from https://www.kaggle.com/datasets/shashwatwork/phishing-dataset-for-machine-learning