Differential Privacy for Fraud Detection Machine Learning Models

Bilal Shaw Hamza Saleem Nicolaas Weideman

December 2019

1 Introduction

Fraudulent credit card transactions are a major problem that plague all credit card companies. In an attempt to detect bad transactions, machine learning models predict the likelihood of a transaction being fraudulent. To build robust models that generalize well across credit card company data, fraud platforms such as those of Sift Science, LexisNexus, ID Analytics and others often build consortium models. In such scenarios, these platforms pool the raw data of credit card companies to build a rich training profile of transaction histories. However, it has recently been shown that machine learning models can leak sensitive information [8, 11]. This can be achieved simply by querying the model for a score on test data. This implies that there is serious potential for one credit card company to learn about data of another that may not have been part of the set that it contributed. As credit card transactions contain a lot of sensitive information, the models would have to be hardened against these types of attack before publication.

In this project we assume a threat model where the credit card transaction data are secure with a trusted curator who has access to all the original private data. We then apply differential privacy to the learned model. Building reliable models that do a good job of not only detecting legitimate fraud in transaction data, but are also quite precise is hard. This is due to the nature of the imbalance inherent in the two classes, *fraudulent* and *non-fraudulent* in training data. The proportion of fraudulent transactions typically ranges between 0.01% to 0.4%, whereas the rest of the data are non-fraudulent. In the machine-learning literature this sort of a situation is dubbed as the *class imbalance problem*.

Our main contribution is that the imbalance class problem has not sufficiently been studied in the context of differentially private machine learning models. We were not able to find extensive literature on this idea. It is also not quite clear how the differentially private model would perform if one re-balances the two classes with various class balancing techniques such as SMOTE (Synthetic Minority Over-sampling Technique) [4]. In SMOTE, for example, the algorithm looks at each minority point, and based on the number of neighbors

defined by a parameter k, it synthesizes a number of points on the line connecting the point in question to its neighbors. The number of points synthesized is a second hyperparameter to the algorithm. In this project we use a variant of SMOTE called SMOTE-TOMEK [2]. In this variant, one identifies a link between pairs of minority-majority points which are then marked for elimination if they violate the decision boundary. This is a way of ensuring that the addition of synthetic examples does not radically alter the distribution of the data.

We hope that this small project launches the first of its kind empirical study toward answering the effectiveness of differentially private models on severely imbalanced class problems such as credit card fraud. With data privacy concerns looming large on the global horizon (CCPA in California and GDPR in the EU), it behooves machine learning companies such as the ones cited above to move to a more secure form of model training and deployment that is built on the mathematically rigorous foundations of differential privacy.

2 Related Work

We have not found extensive literature detailing empirical studies on DP-LR and its relation to the class imbalance problem. We think that the reasons might be two-fold. Firstly, DP is still somewhat of a nascent field within computer science and not widely adopted or well understood outside the academic community, although this is starting to change albeit relatively slowly. Secondly, for an industry to adopt a differentially private model to detect and prevent fraud, it has to be incentivized appropriately. In [8, 11], the authors lay the groundwork of why industry should care about differential privacy. However, their results are fairly new, and have not necessarily trickled to all facets of financial institutions. The class imbalance problem on the other hand has been studied for decades because it appears in many real-world problems in finance, physics, biology, economics, marketing research, and even more fields than we can list here. There has been ample research and monetary incentive to make headway into this problem. Researchers who are experts in DP may not be experts in class imbalance and vice versa. We hope that this will change for the better in the coming years.

Nevertheless, we found two papers related to differential privacy and class-imbalance. In [1] the authors provide evidence of how and why DP Stochastic Gradient Descent adversely affects the accuracy of the minority class more than it does the majority class. In other words, they show disparate impact on the minority class as opposed to the majority. The gradients associated with the minority class are larger in value compared to the majority class, and hence clipping and adding noise has a harsher impact on the minority accuracy. In [6] the authors apply federated learning to in a privacy preserving manner to create balanced datasets from imbalance classes. Neither of the papers cover credit card fraud data per se, which is what we shed light on albeit empirically in this report.

3 Experimental Setup

3.1 Dataset

For evaluating our proposed solution experimentally, we used the *Credit Card Fraud* dataset [9]. This dataset contains 284,807 records, each representing a credit card transaction with 30 features and an additional column that indicates if the record is fraudulent or not. The dataset is highly imbalanced with approximately 0.173% of the transactions as fraudulent. The 30 features in this set are principal components derived from a set of underlying engineered features. Hence, these principal components are uncorrelated with each other. Except for "Time" and "Amount" (in dollars), none of the features have a meaningful name. They are simply labelled "V1" to "V28". Moreover, the rest of the features were somewhat normalized with values within a reasonable range. We use z-scaling to normalize "Time" and "Amount" so that each has a mean of 0 and unit standard deviation. Finally, the features have no missing values, so there was no need for imputation. Since this is rarely the case in real-world data, in Section 3.4.3 we investigate the results of our models when using artificially introduced missing value imputation.

Since we did not have access to a similar dataset from which we could learn the sensitivity of the features, we instead uniformly sampled 1% to obtain an estimate for sensitivity (see Section 3.3). The rest of the data, we used to train and test our models. We set aside 70% of the data for training and the rest for testing. We split the two sets uniformly at random each with approximately a fraud rate of 0.2%.

3.2 Models

We chose logistic regression as our supervised algorithm of choice to build our model. There were several motivating factors behind this choice, chief among them being that it is one of the most well understood models often called the "hello world" algorithm of machine learning. Moreover, logistic regression produces a linear model. This makes model interpretation easier compared to algorithms such as neural networks which product non-linear models. The algorithm executes rather quickly with minimal memory impact which allowed us to build models on personal computers rather than a cluster. To measure the performance of this model, we built a base logistic regression model which we designate as non-DP-LR model.

In [3], Chaudhuri, Monteleoni, and Sarwate modify logistic regression to incorporate differential privacy in the algorithm's objective function. They give good theoretical guarantees on the performance of their differentially private algorithm, and back it up with empirical studies. However, they do not study their differentially private logistic regression algorithm in the context of an imbalanced dataset which is what we showcase in this paper. We were also fortunate to find a Python implementation of the algorithm [7]. Using this implementation, we built a differentially private logistic regression model, the

DP-LR model. The non-DP-LR and DP-LR models were trained on both the imbalanced and re-balanced datasets that allows us to measure the impact that the incorporation of differential privacy into a logistic regression machine learning model has on its performance. We refer to the non-DP-LR model as the *base LR* model.

Apart from logistic regression, we also trained a gaussian naive Bayes model for classification. A naive Bayes classifier uses Bayes' Theorem to assign class labels to problem instances. Gaussian Naive Bayes (GNB) is a type of naive Bayes classifier used when the features are continuous and it assumes that these features are normally distributed. We chose a GNB classifier because the classifier works well for many real-world cases despite its simplifying assumption that a value of a particular feature is independent of the value of other features, given the class variable. Moreover, the classifier is highly scalable as it requires parameters linear in the number of features and the training can be performed in linear time using a closed form expression as compared to iterative steps normally required in case of other classifiers.

We first built a non-DP-GNB classifier without incorporating any class balancing techniques to act as a base model for our evaluation. To study the effect of incorporating differential privacy for training on the performance of a GNB classifier, we trained a DP-GNB classifier and compared its performance with the non-DP-GNB base model. In [10] Vaidya et.al. propose a simple technique to incorporate differential privacy in Naive Bayes training by limiting the sensitivity of the mean and the standard deviation of the continuous features in the dataset. However, they do not evaluate the performance of the DP model for imbalanced datasets. The same library we use for the DP-LR model, also has an implementation of DP GNB [7], which is based on the paper [10]. We use this build the DP GNB model for our imbalanced dataset.

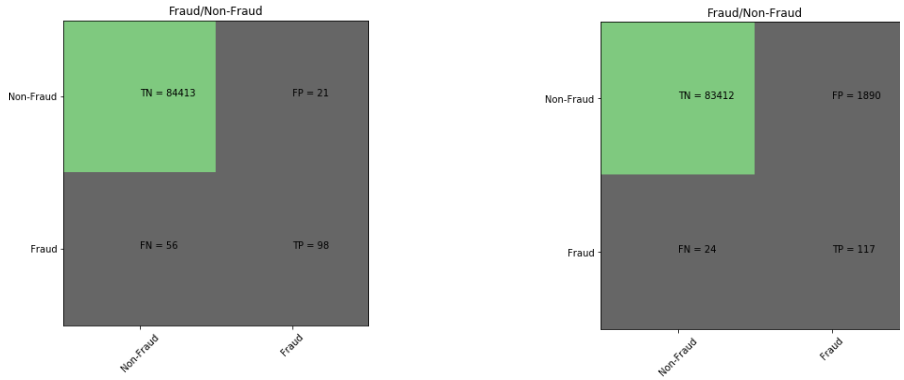

Figure 1: Confusion Matrix for a) Base LR Model b) Base GNB Model

3.3 Limiting Sensitivity

In order to achieve differential privacy for a machine learning model, it is necessary to limit the impact a single data record can have on the trained model. Since in both our non-DP models a single data point can have an arbitrarily large impact on the trained model, it is necessary to place artificial bounds on the features before training the DP models. In order to achieve differential privacy, these bounds cannot depend on the specific dataset that is used. However, as we noted in Section 3.1, we do not know what most of the features in our dataset represent. Therefore, it is impossible to limit the sensitivity in any sensible way. To circumvent this problem, we extracted 1% of the data to use exclusively toward the purpose of estimating an upper bound and lower bound value for each feature. The selected upper and lower bounds are shown in Appendix A in Table 1.

Using these bounds, we truncate the features of both the training and testing data before giving it as input to the model. Since the bounds depend on the dataset, we acknowledge that this introduces additional privacy leakage. We expect, however, that if this model is implemented in practice, the developers will have more knowledge about the data, making it possible to estimate bounds in a manner compliant with differential privacy.

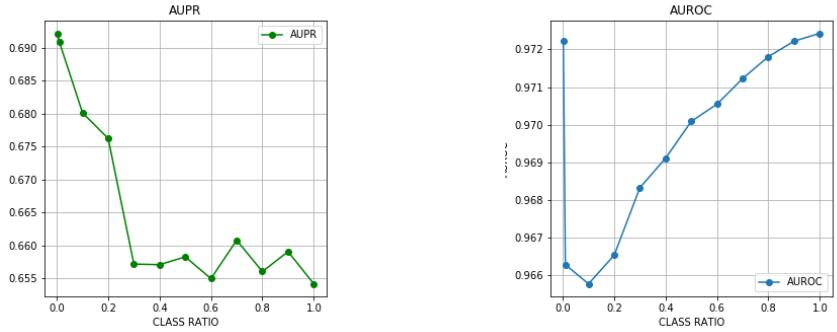

Figure 2: The relationship between the class ratio and a) AUPRC b) AUROC for the base LR model.

3.4 Experimental Results

To evaluate the performance of the models on the test set we chose Area Under the ROC Curve, also known as at the AUROC metric as well as Area Under the Precision-Recall Curve, also known as AUPRC. There is a deep connection between curves in ROC space and those in PR space. For example, a curve that dominates in ROC space also dominates in PR space. However, algorithms that optimize curves in ROC space need not optimize them in the PR space [5]. While AUPRC is a more suitable metric to evaluate models built on class imbalance, we include AUROC as an additional metric to understand how recall changes

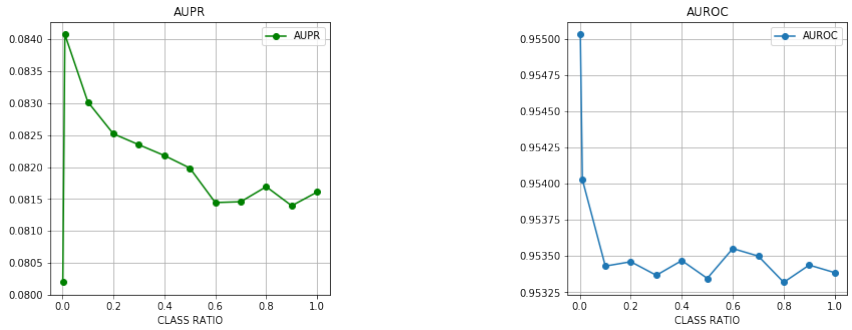

Figure 3: The relationship between the class ratio and a) AUPRC b) AUROC for the base GNB model.

as a function of the false-positive-rate.

3.4.1 Base Models

The AUPRC on our base LR model is approximately 69% whereas the AUROC is 97.2%. The precision is equal to 82.4% whereas the recall is quite low at 63.6%, which indicates that our base fraud model does a poor job of detecting frauds. However, the examples it does detect as fraud, the model is precise approximately 83% of the time. The confusion matrix detailing the performance of this model is captured in Figure 1. Similarly, for our base GNB model, the AUPRC is approximately 8.3% whereas the AUROC is 95.8%. The precision of the model is 5.8% and the recall is 82.9%. The model is thus capable of detecting approximately 83% of the fraud but with low precision. The confusion matrix in Figure 1 details the performance of the base GNB model.

Additionally, we ran 1000 bootstrap samples to create a 95% confidence interval for both AUROC and AUPRC. For the former the interval was [96%,98.9%] in case of the base LR model and [94.7%,97.6%] for the base GNB model, and for the latter it was [70%,81.0%] in case of the base LR model and [7%,9.9%] for the base GNB model. In Figure 2, we document the variation in AUROC and AUPRC for the non-DP-LR model as a function of different class ratios, respectively. Similarly, Figure 3, shows the variation in AUROC and AUPRC for the non-DP-GNB model. Figure 4 captures the relation of precision and recall as a function of varying class ratio for the non-DP-LR model. We note that the model becomes highly imprecise as the class ratio goes up. This is because the false positives goes up dramatically, whereas the false negatives steadily remain low which is reflected by the improved recall performance with increasing class ratio. Similarly, Figure 4 shows the relation of precision and recall as a function of varying class ratio for the non-DP-GNB model.

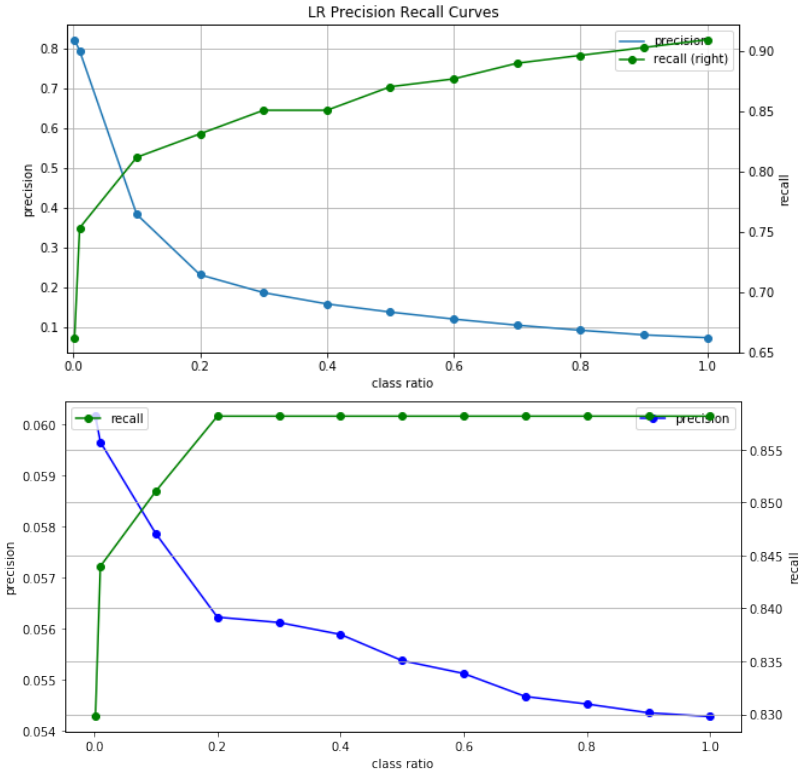

Figure 4: The relationship between precision-recall curves and class ratio for base LR model & base GNB model.

3.4.2 Differentially Private Models

For the DP-LR and DP-GNB models, we wish to understand the relationship between the privacy parameter ε, the class ratio and the performance, AUROC and AUPRC, and how these models compare to their non-DP variants. Using SMOTE-TOMEK, we create re-balanced training datasets for the following ratios

$$\{0.02, 0.01, 0.1, 0.2, \ldots, 1.0\}.$$

For a ratio r, if x is the number of records in the minority class and y is the number of elements in the majority class, in the re-balanced dataset, the relationship will be $r = \frac{x}{y}$. We also define a set of ε values.

For each pair of ε and ratio values, we train 5 of both DP-LR and DP-GNB models on the re-balanced data (after it has been truncated to limit sensitivity). We test the trained models and plot the average performance, AUROC and AUPRC, as a function of ε. Calculating an average for the performance allows us to eliminate some inconsistency introduced by the randomness of differential privacy. Understanding the relationship between the performance and

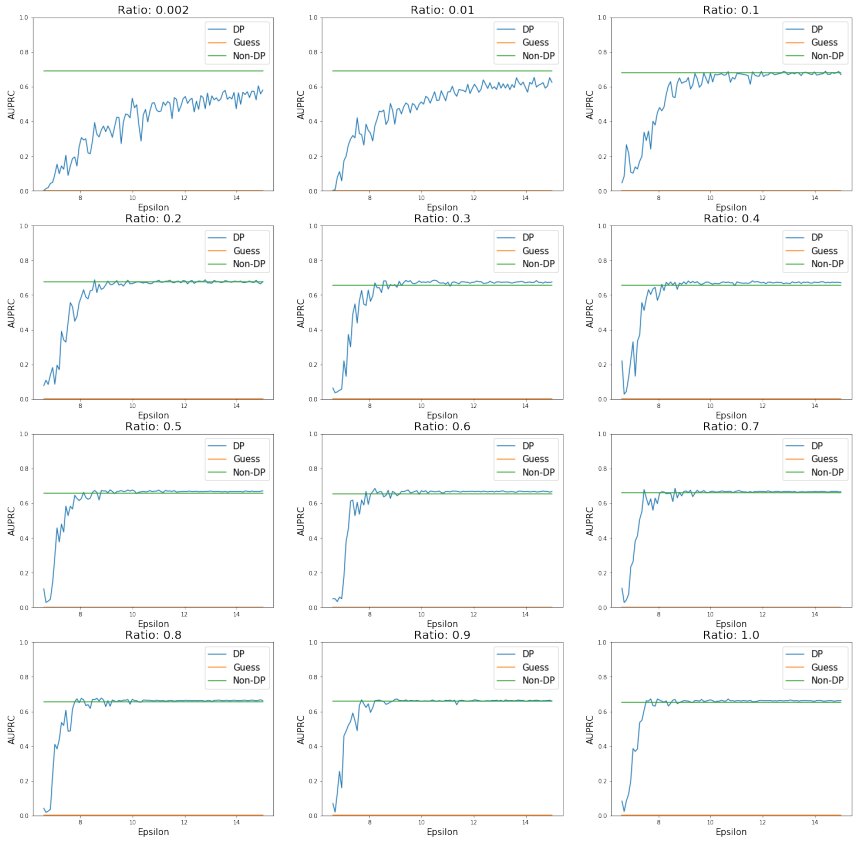

Figure 5: The performance in terms of AUPRC, for different re-balancing ratios and privacy parameters. Each graph shows the performance of the non-DP-LR model, DP-LR model and a model that simply guesses whether a transaction is fraud based on the ratio.

the privacy parameter is key to approximating the utility of the differentially private machine learning models.

Figures 5 and 6 show the results for the DP-LR model whereas Figures 7 and 8 show the results for the DP-GNB model. Each graph contains a plot showing how the performance is affected by the ratio and privacy parameter, as well as two plots for sanity checking the first. For sanity checking, we plot the performance of the non-DP model for a specific ratio, as well as the performance of a model that simply guesses whether each transaction is fraud, depending only on the number of fraud transactions in the dataset. For example, if the ratio is 1.0, it assigns the label *fraud* to every transaction with probability 0.5. Intuitively, we do expect the DP model not to perform worse than the guessing model and no better than the corresponding non-DP model. The figures show that the results correspond with this intuition.

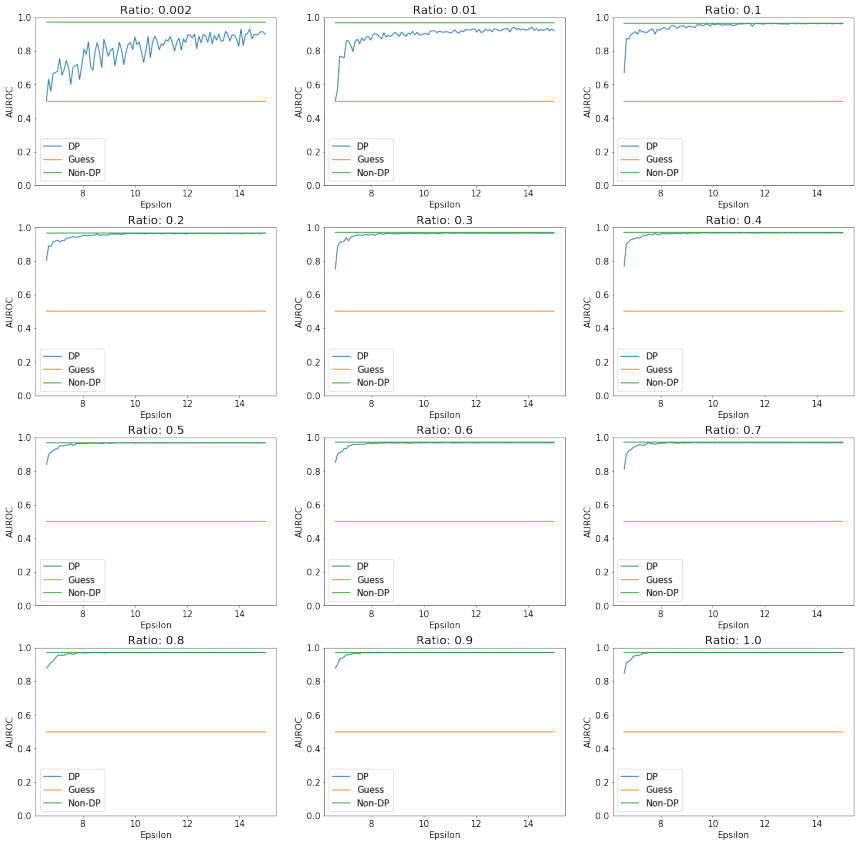

Figure 6: The performance in terms of AUROC, for different re-balancing ratios and privacy parameters. Each graph shows the performance of the non-DP-LR model, DP-LR model and a model that simply guesses whether a transaction is fraud based on the ratio.

In Figures 5 and 6, we plot the AUPRC and AUROC values for the DP-LR models against relatively large privacy parameters, $\varepsilon \geq 6.6$. While experimenting with different privacy parameters, we have noticed that performance falls sharply for a particular value $\varepsilon \approx 6.5$, before it improves again as expected. We believe this phenomenon may be caused by an error in the implementation of the algorithm [7]. This leads us to believe that the results for $\varepsilon \leq 6.5$ are unreliable. The cause of this phenomenon is explained in more detail in Appendix B.

Figures 5 and 6 also show that re-balancing the dataset typically improves the rate at which performance improves with ε. Also, as expected, performance approaches that of the non-DP-LR model as ε increases for any ratio.

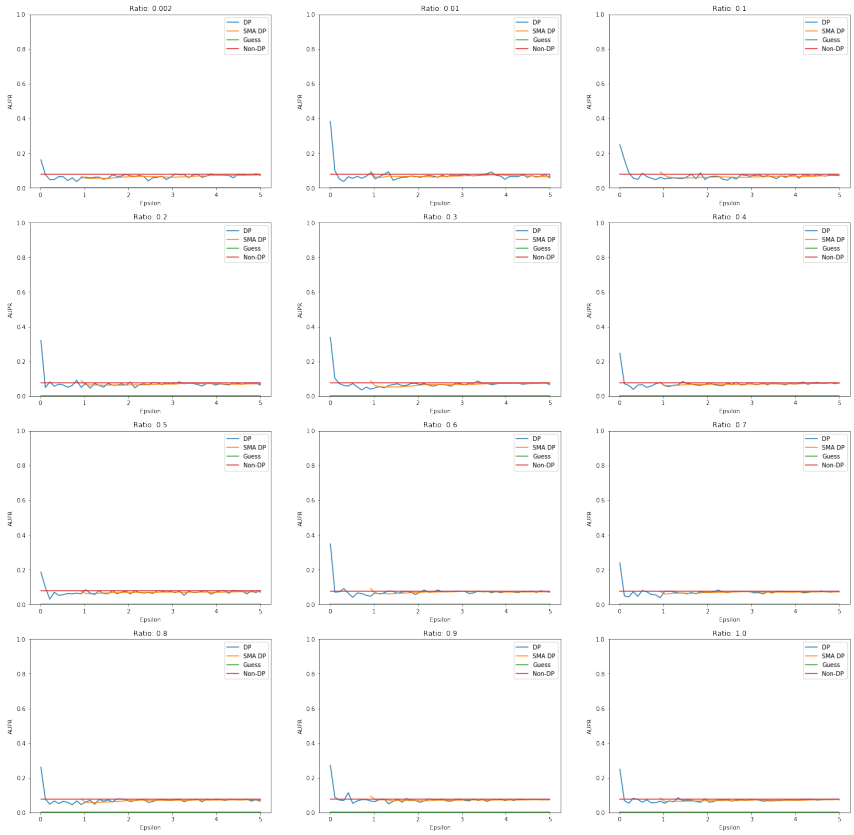

Figure 7: The performance in terms of AUPRC, for different re-balancing ratios and privacy parameters. Each graph shows the performance of the non-DP-GNB model, DP-GNB model and a model that simply guesses whether a transaction is fraud based on the ratio.

For both the DP-LR and DP-GNB models, we were surprised to see that reasonable utility, relative to the non-DP models, can be achieved for relatively small values of ε. Due to time constraints, we where unable to investigate this further.

3.4.3 Performance on Datasets with Missing Values

In the real world we rarely deal with datasets with no missing values. In order to compute realistic performance numbers, we artificially inserted missing values into the features of our current data with rates varying from 0.01 to 0.5. We imputed the missing values for each feature in the train and test sets by their corresponding means. We should mention, however, that there are several real-world problems where data are sparse where one would resort to

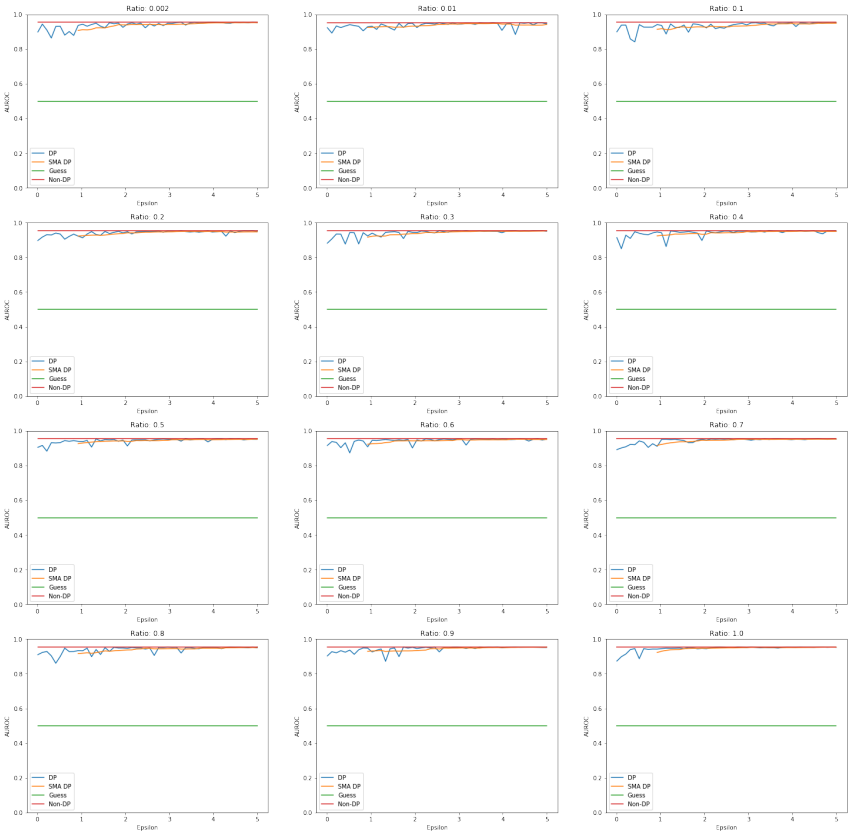

Figure 8: The performance in terms of AUROC, for different ratios and privacy parameters. Each graph shows the performance of the non-DP-GNB model, DP-GNB model and a model that simply guesses whether a transaction is fraud based on the ratio.

a more sophisticated way of imputing the data or move to a low rank matrix representation of the data.

For each missing value dataset, we trained and tested both DP and non-DP logistic regression models. We did similarly for the GNB models. We did not employ any re-balancing techniques in this case.

Figures 10 and 9 show the performance of the non-DP and DP models, for both LR and GNB, as a function of missing data ratios for three different values of epsilon. Each of the three epsilon values were selected from Figures 5, 6 7 and 8 where the corresponding algorithm shows *poor*, *medium* and *good* performance. This allows us to illustrate how missing values affect DP algorithms at various levels of utility. As expected, there is a general downward trend seen for both AUROC and AUPRC metrics with increasing missing rates. As epsilon increases, the gap in performance for the non-DP and DP models for

different missing ratios narrows down. For GNB and epsilon 0.01, the DP model performs better than the non-DP model over the AUPRC metric. The reason is that the non-DP-GNB model, has a low precision because of a large FP's (1890) which causes the AUPRC to degrade. However in case of the DP GNB model, for small values of epsilon like 0.01 the model has very low FP's (a few hundred) and thus the AUPRC is better than the non-DP counter-part. One possible explanation for this might be that for small epsilon values, the noise causes the classifier to classify large number of examples as "Non-Fraud" as the vast majority of examples in the training dataset have a "Non-Fraud" label.

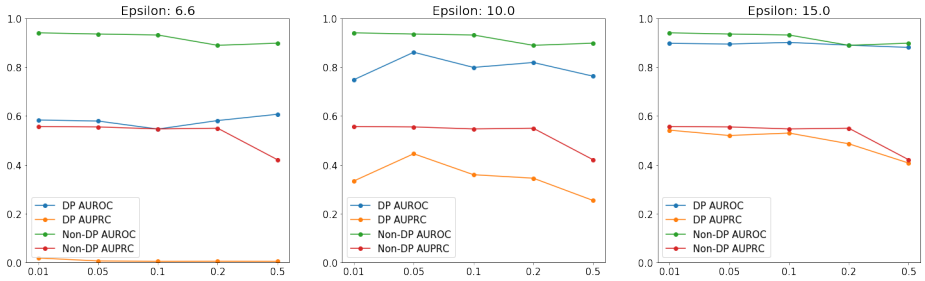

Figure 9: The relationship between performance metrics of the LR models and missing values for three different privacy parameters.

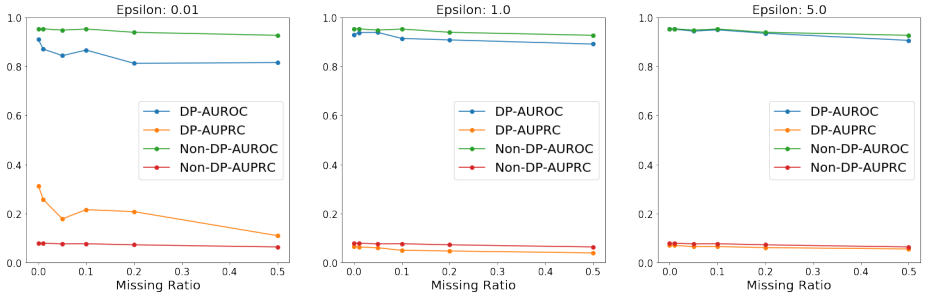

Figure 10: The relationship between performance metrics of the GNB models and missing values for three different privacy parameters.

4 Future Work

Since the dataset we use has mostly unlabeled features, it is difficult to reason about the real-world use of the models implemented in this paper. One avenue for future work would therefore be to obtain a second, similarly imbalanced dataset and validate the results. We believe this is an important direction to pursue, as the DP models showed surprisingly good performance, even for small privacy parameters.

A next step could be to compare the results of DP-LR and DP-GNB directly to determine which model is better suited for task of fraud detection. However, due to the suspected error in the implementation of the DP-LR model, this is currently not possible. To achieve this, analyzing the definition of the algorithm, and the code of its implementation is necessary. We did not succeed in this due to time constraints.

5 Conclusion

In this paper, we set out to kick start an investigation in using differential privacy to protect the sensitive data found in credit card transaction. We have seen that re-balancing techniques can be used effectively along with DP. Moreover, this does not affect performance disproportionately in comparison to the non-DP case, which is usually the case for imbalanced datasets [1].

While our results cannot be used to make any direct claims about real-world application, we believe we have indicated a starting point from where to continue the investigation.

References

[1] Eugene Bagdasaryan and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. *CoRR*, abs/1905.12101, 2019.

[2] Gustavo E. A. P. A. Batista, Ana L. C. Bazzan, and Maria Carolina Monard. Balancing training data for automated annotation of keywords: a case study. In Sérgio Lifschitz, Nalvo F. Almeida Jr., Georgios Joannis Pappas Jr., and Ricardo Linden, editors, *II Brazilian Workshop on Bioinformatics, December 3-5, 2003, Macaé, RJ, Brazil*, pages 10–18, 2003.

[3] Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. Differentially private empirical risk minimization. *J. Mach. Learn. Res.*, 12:1069–1109, 2011.

[4] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002.

[5] Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In William W. Cohen and Andrew W. Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 233–240. ACM, 2006.

[6] Seok-Ju Hahn and Junghye Lee. Privacy-preserving federated bayesian learning of a generative model for imbalanced classification of clinical data, 2019.

[7] IBM. Diffprivlib: The IBM Differential Privacy Library , December 2019.

[8] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 3–18. IEEE Computer Society, 2017.

[9] Machine Learning Group ULB. Credit card fraud detection. https://www.kaggle.com/mlg-ulb/creditcardfraud.

[10] Jaideep Vaidya, Basit Shafiq, Anirban Basu, and Yuan Hong. Differentially private naive bayes classification. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01*, WI-IAT '13, pages 571–576, Washington, DC, USA, 2013. IEEE Computer Society.

[11] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*, pages 268–282. IEEE Computer Society, 2018.

A Feature Bounds

Table 1: The selected upper and lower bounds for each feature used in limiting the sensitivity.

Feature	Lower bound	Upper bound
Amount	-6	12
Time	-7	7
V1	-16	8
V2	-15	11
V3	-12	8
V4	-9	10
V5	-9	10
V6	-9	10
V7	-9	10
V8	-8	8
V9	-8	8
V10	-8	9
V11	-8	8
V12	-9	8
V13	-8	8
V14	-8	8
V15	-8	8
V16	-9	8
V17	-7	8
V18	-8	8
V19	-8	9
V20	-9	9
V21	-8	9
V22	-8	9
V23	-8	9
V24	-7	7
V25	-8	7
V26	-7	7
V27	-8	7
V28	-7	8

B Performance Dips for Differentially Private Logistic Regression

In our experiments for evaluating the impact that differential privacy has on the performance of logistic regression, we have noticed that the implementation [7] yields mysterious performance dips at a specific epsilon value. In this section

we aim to explain the cause of these dips at a high level. First, consider the definition of the Algorithm 2, in [3], from which the implementation has been adapted from. This has been copied into Figure 11 for convenience. In this

Algorithm 2 ERM with objective perturbation

Inputs: Data $\mathcal{D} = \{z_i\}$, parameters ε_p, Λ, c.
Output: Approximate minimizer $\mathbf{f}_{\text{priv}}$.
Let $\varepsilon'_p = \varepsilon_p - \log(1 + \frac{2c}{n\Lambda} + \frac{c^2}{n^2\Lambda^2})$.
If $\varepsilon'_p > 0$, then $\Delta = 0$, else $\Delta = \frac{c}{n(e^{\varepsilon_p/4} - 1)} - \Lambda$, and $\varepsilon'_p = \varepsilon_p/2$.
Draw a vector $\mathbf{b}$ according to (4) with $\beta = \varepsilon'_p/2$.
Compute $\mathbf{f}_{\text{priv}} = \text{argmin} J_{\text{priv}}(\mathbf{f}, \mathcal{D}) + \frac{1}{2}\Delta||\mathbf{f}||^2$.

Figure 11: The algorithm defined in [3] for empirical risk minimization with objective perturbation.

algorithm, ε_p is the privacy parameter, $c = 0.25$, n is the size of the dataset and Λ is a regularization parameter. The variable ε'_p is inversely proportional to the amount of noise required to achieve differential privacy. This variable can take on one of two values, either

$$\varepsilon'_p = \varepsilon_p - \log\left(1 + \frac{2c}{n\Lambda} + \frac{c^2}{n^2\Lambda^2}\right) \tag{1}$$

or, if this value is not strictly positive,

$$\varepsilon'_p = \frac{\varepsilon_p}{2} \tag{2}$$

To understand how the drops in performance occur, we need to understand for which values of ε_p the dependent variable ε'_p is small and therefore, for which the algorithm uses a large amount of noise. The first thing to note is that ε'_p can never be larger than ε_p. Therefore, when ε_p is small, ε'_p will also be small and a large amount of noise is used. This is as you would expect, since any differentially private algorithm given a small privacy parameter uses a lot of noise.

The second case for which ε'_p is small, occurs for a small ε_p that makes Equation 1 strictly positive. Typically, however, ε_p will also be relatively small in this case. If we assume n is large, then

$$-\log\left(1 + \frac{2c}{n\Lambda} + \frac{c^2}{n^2\Lambda^2}\right) \approx 0. \tag{3}$$

Therefore, to achieve poor performance an $\varepsilon_p \approx 0$ is required.

Since both cases for poor performance require a small privacy parameter, we can reason that this does not affect the utility of the algorithm in practice. In the implementation of the algorithm, however, an adaption has been made so that poor performance can be achieved for a larger privacy parameter. Consider the code shown in Figure 12 implementing the algorithm. The `scale` variable is used to set the amount of noise to generate, `c` $= 0.25$ and $\alpha = 1$. The variable

```
epsilon_p = epsilon - 2 * log(1 + c * x / (0.5 * alpha))
delta = 0

if epsilon_p <= 0:
    delta = (c * x / (exp(epsilon / 4) - 1) - 0.5 * alpha)
    epsilon_p = epsilon / 2

scale = epsilon_p / 2 / x
```

Figure 12: The code in [7] implementing Algorithm 2 in [3].

`x` is supplied by the user as a maximum norm for the dataset. This differs from the paper definition, which assumes $\texttt{x} \leq 1$. As with the theoretical definition, we expect poor performance for a small `epsilon` value that makes the first line of Figure 12 positive. Unlike the theoretical definition, however, in the code there is no dependence on the size of the dataset. Moreover, specifying a large value for `x` increases the value of `epsilon` where the poor performance occurs.

With the implementation, we can achieve poor performance for a privacy parameter that is not negligibly small. We suspect this indicates an error in the implementation.

C Non-DP Logistic Regression Model Performance and Class Ratio

Ratio	Non-Fraud	Fraud	TP	TN	FP	FN	AUROC	AURPC
1	197035	197035	140	82663	1771	14	97.25%	65.57%
0.9	197035	177331	139	82843	1591	15	97.21%	65.82%
0.8	197035	157628	138	83073	1361	16	97.13%	65.64%
0.7	197035	137924	137	83262	1172	17	97.14%	66.01%
0.6	197035	118221	135	83445	989	19	97.06%	65.09%
0.5	197035	98517	134	83595	839	20	97.01%	65.73%
0.4	197035	78814	131	83737	697	23	96.91%	65.76%
0.3	197035	59110	131	83863	571	23	96.84%	65.81%
0.2	197035	39407	128	84009	425	26	96.67%	67.65%
0.1	197035	19703	125	84233	201	29	96.49%	67.53%
0.01	197034	1969	116	84404	30	38	96.65%	68.84%
0.002	197019	378	102	84412	22	52	97.48%	69.19%

Table 2: Non-DP Logistic Regression model performance as a function of class ratio.