Erasmus University

Erasmus School of Economics

Thesis Business Analytics and Quantitative Marketing

Handling privacy preservation for binary classification data

Author:

D. Veldhuizen (483082)

Supervisors:

U. Karaca

P.J.F. Groenen

July 5, 2020

Abstract

This paper proposes two different privacy preserving algorithms based on logistic regression and tests how well the latter perform. To preserve privacy, the first algorithm adds noise to the estimator. The second algorithm adds noise to the minimisation function. The data consists of three sets. Two of these sets are simulated, with the difference that one is separable and the other one is not. The remaining set is a real data set. The results indicate that there is no superior algorithm in terms of performance. The separable set does not show any difference between the two algorithms, the inseparable set indicates that algorithm 2 gives better predictions and the real data designates smaller errors for algorithm 1. So, depending on the data sets and their characteristics one algorithm is preferred over the other.

Contents

1 Introduction **2**

2 Literature **4**

2.1 Preliminaries 4

2.2 Relevant literature 5

3 Data **5**

4 Methodology **8**

4.1 Regularised logistic regression 9

4.1.1 Logistic regression 9

4.1.2 Estimating parameters with regularisation 9

4.2 Algorithm 1 11

4.2.1 Proof of privacy preservation 11

4.3 Algorithm 2 12

4.3.1 Proof of privacy preservation 12

4.4 k-fold cross-validation 13

4.5 Predicting, calculation of test error and comparison of performance 14

5 Results **15**

5.1 General performance 15

5.2 Learning performance 16

5.3 Privacy and performance 17

6 Conclusion and discussion **18**

7 Appendix **21**

1 Introduction

It seems to be in human nature to keep personal information to one's self. However, in the current state of society, where information is shared by millions of people on a daily basis, it has become difficult to maintain that privacy. Look for example at Google. Many models that were used in their calculations violated privacy. Yet, it is not only inferior for the privacy of its users, but also for the company itself. They got fined 170 million dollars (Singer and Conger, 2019). In this paper the focus lies on how to solve this privacy problem in a efficient and realistic manner. Particularly, the emphasis lies on binary classification data. The research question therefore becomes: "How do we efficiently and realistically preserve privacy for binary classification data?"

The data consists of three different data sets. The first two of those sets are simulated data. They differ in the sense that one is completely linearly separable, whereas the second set is not. The remaining data set is a real data set, retrieved from the UCI machine learning repository (Dua and Graff, 2020). It contains information about the measurements of a discovered abnormal lump found on a women's breast and the diagnosis of breast cancer.

We define privacy according to Dwork et al. (2006). In their research one uses ϵ-differential privacy. This definition states that if at most one element from a data set is replaced or removed, the log-likelihood may only change by a value smaller than or equal to ϵ. If the log-likelihood does not change much, there is no single data point that has a significant influence on the fit of the model, which means privacy is preserved.

The paper contributes to current knowledge and literature, as it gives an idea about how one can incorporate privacy and shows under what circumstances this inclusion works well by extending the work of Chaudhuri and Monteleoni (2008). We do not only look at simulated data, but also take a true data set into account. Simulated data contain characteristics that fit the specific type of research. Put differently, data simulation is an idealisation of data. As a consequence, the results are idealised too. However, if these results differ from a real data set, what is the use? With our research we want to find out the general performance

for more realistic data sets.

Two algorithms that preserve privacy are the foundation of our research. To achieve that preservation, the first algorithm uses the coefficients obtained from regularised regression and adds noise. The second algorithm has a different approach and incorporates noise in the minimisation function. To train and test these algorithms we make use of five fold cross-validation. This splits up the data set into five equal parts. Four parts train the algorithms. The remaining data set tests the performance. This is repeated until all possible data is used as training data and test data. We then calculate the average test error for different batch sizes to not only compare the performance for a large data set, but also to compare the learning curves of both algorithms. Lastly, both algorithms make predictions for one large data set, using different values for ϵ. Doing this shows that less restriction on privacy, leads to an decrease of test errors. This indicates that the algorithms preserve privacy.

The results vary. For the inseparable data, the graphs of the two algorithms lie on top of each other. This indicates they perform equally well. Yet, the computation time of the first algorithm was much shorter than that of algorithm 2. For the inseparable data algorithm 2 outperforms algorithm 1. Standard logistic regression performs well for both of these simulated sets, with a test error of 0 and 0.035, respectively. Logistic regression performs poorly for the real data set, however. Moreover, unlike Chaudhuri and Monteleoni (2008) suggest, algorithm 1 has a better performance for this data set than algorithm 2. In both the case where the batch size increases, and where the restriction on privacy preservation decreases, the graph belonging to the test error of algorithm 1 lies below the one belonging to algorithm 2.

So, depending on the data set the results differ. For the separable data, algorithm 1 is preferred due to similar performance but shorter computation time. For inseparable data we prefer algorithm 2, on account of a better performance. Nevertheless, it is necessary that logistic regression works well for that data. If this it not case, like for the real data set, algorithm 1 is preferred.

The remainder of the report is structured as follows. Section 2 discusses definitions and theory that are important to understand the rest of the paper. Section 3 consists of details about the simulation of the first two data sets and the source of the real data set. In Section 4 we discuss machine learning algorithms and the techniques we use for predicting. Section 5 then examines our main findings. Lastly, in section 6, we conclude and make final remarks about limitations and possibilities regarding the improvement of our research.

2 Literature

In this section we define and explain specific preliminaries and relevant literature used in previously conducted research salient for understanding our research.

2.1 Preliminaries

ϵ-differential privacy Often, data sets consist of data points corresponding to individuals. Using their information to estimate and predict is for many a violation of privacy. Especially, when this information is vital for the model to work properly. The results become too personal. To measure this, Dwork et al. (2006) defines ϵ-differential privacy. Let D and A be two data sets which differ by at most one element. Moreover, let M be a method or mechanism that needs to preserve privacy. Lastly, let t be any outcome of that mechanism. Dwork et al. (2006) then defines ϵ-differential privacy as

$$\frac{P[M(D) = t]}{P[M(A) = t]} \leq e^{\epsilon} \tag{1}$$

This definition states that when information of a single individual is removed or replaced from the original data set, the value of the log-likelihood may only change by a value of ϵ. This means that the information of a single individual is not necessary for the model to work well. Privacy is therefore preserved. The smaller the value of ϵ is, the more strict the privacy restriction. Oppositely, if ϵ is large, there is more freedom regarding privacy. ϵ-differential privacy is what all algorithms in this paper revolve around.

Sensitivity and privacy preservation Sensitivity is the maximum value change of a function, over all possible changes of exactly one input variable x (Chaudhuri and Monteleoni, 2008). For example, the function for an OLS estimator equals $(X'X)^{-1}X'Y$. Changing a single value of a data point changes the value of the different estimated coefficients. The maximum change that can be achieved by this change is the sensitivity. Recall that in the case of this example, each estimated coefficient within the vector has its own sensitivity. The sensitivity of the OLS-estimator is a vector.

2.2 Relevant literature

Machine learning Machine learning algorithms are algorithms that try to pursue human behaviour, by learning from data that is put in, called the training-data. The efficiency and performance of the algorithms are then evaluated using test-data. This data is similar to the training-data set, but not used for the learning process. Machine learning has been used effectively in many different branches such as pattern learning, entertainment and spacecraft engineering (El Naga, 2015). Therefore researchers such as Bradley et al. (2011), focus on privacy use machine learning to make a well-working model to preserve the latter. In our research we make use of these machine learning methods too.

Simulation Simulation is the technique of mimicking a process or system. In Econometrics, this refers to imitating mathematical models or data. In this paper the focus lies on data simulation. Simulated data is convenient when there is need for a character specific data set. It might be the case that a model only works under certain restrictions. By simulating data we can take these restrictions into account. On the other hand, it might not be restrictions, but the poor results of a model that solicits for simulation. Simulation improves and clarifies these results (Maria, 1997). For our research we simulate on account of both reasons.

3 Data

In these section we look at the data we use for our research. The data consists of 3 binary classification data sets. The first two of those sets are simulated data. The third set is real

data with statistics of an abnormal lump in the breast and a binary variable regarding the diagnosis of breast cancer. We use this last real data set to see how well the algorithms work for data that miss some characteristics that we chose in simulation. The simulated sets contain 10 variables and 10000 data points. The real data set contains 5 variables and 565 data points. Let X_i be the vector with these variables, corresponding to observation i. In order for the privacy preserving algorithms to work, the data sets must accede

$$\|X_i\| \leq 1. \tag{2}$$

Normalising every data vector secures this restriction. Moreover, the assumption is made that the best linear classifier goes through the origin. (Chaudhuri and Monteleoni, 2008)

The first data set is a separable simulated data set that is uniformly distributed from the unit 10 dimensional hypersphere. By doing so, we maintain the necessary restriction that the vectors belonging to the data points have a norm smaller than or equal to one. Assuming the linear separator through the origin is the best separator, we choose one of many, along with a band margin of 0.03. A generated point is only added to the simulated data set if it falls outside of this margin. Depending on which side of the band margin a point falls, the point is labelled with either -1 or 1. Figure 1 visualises this two-dimensionally. Here you see the linear separator and its margins. They split up the whole data set in a blue subset, which are the data points labelled with -1, and the green points, which are labelled with 1. The data set is separable.

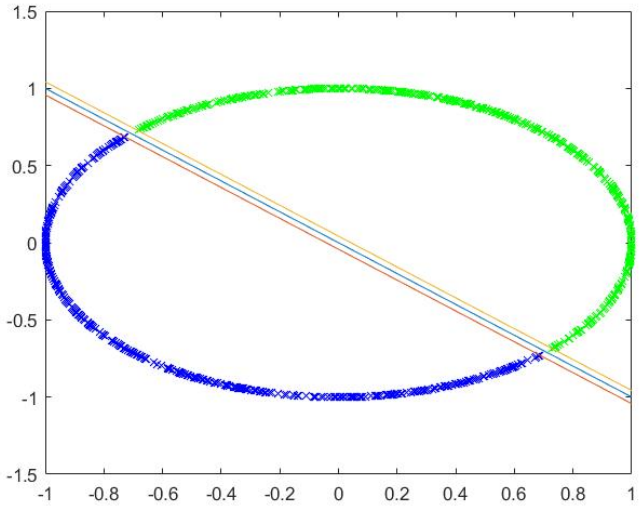

Figure 1: Separable data with margin of 0.03.

The second data set is an inseparable simulated data set. In this set too, it is only allowed to have data vectors that have a norm smaller than or equal to one. We therefore first copy the simulation techniques to create a similar separable data set. The separator chosen here is once again one of many linear separators and therefore might not be the optimal linear separator. The optimal separator maximises the band margin. A method called the support vector machine algorithm (El Naga, 2015) we estimate this perfect linear separator. The reason we do this is because of the next step. The optimised margin is expanded. By using the perfect linear separator instead of the initial separator we are sure that points are going to fall in this expanded margin. In this case, we set the margin to 0.1, similar to Chaudhuri and Monteleoni (2008). All points that fall inside this new margin are given an opposite label, with a probability of 0.2. This probability is also called the flipping probability. For instance, if a point had an original label of 1 in the separable data set, but now falls into the expanded margin, it gets the label -1. By doing so, we create an inseparable data set. We visualise this Figure in 2. To make a more clear distinction between the separable and inseparable data we increase the flipping probability for this plot from 0.2 to 0.6. The red and yellow lines indicate the borders of the expanded margin. All the rest of the colouring is equal to that of separable data. We see that some points fall inside this larger margin, which results in opposite labelling and a change of colour. No longer can a linear line split the full

data set. The data is inseparable.

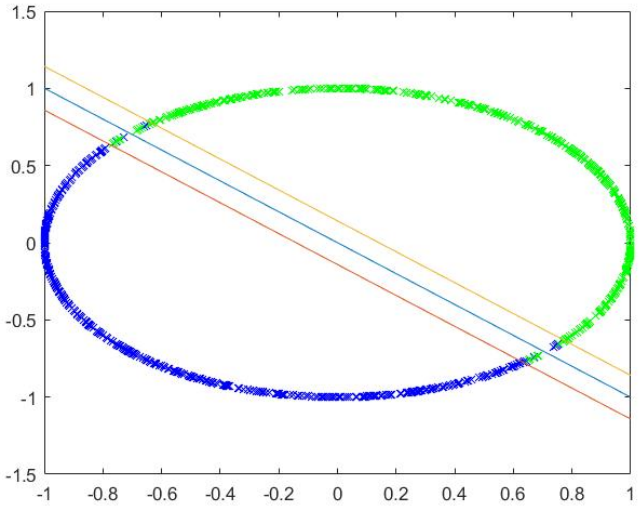

Figure 2: Inseparable data with expanded margin of 0.1 and flipping probability of 0.6.

The last data set we use is a real normalised data set regarding the diagnosis of breast cancer (Dua and Graff, 2020). It contains five measurement variables, such as the area and the perimeter of an abnormal lump. The binary variables equals one if there is a diagnosis of cancer, and zero if this is not the case.

4 Methodology

This section explains all the methods of our research. The section is split up in five different parts. Three of those contain information about the models and algorithms we use to estimate parameter values and the other two contain information about predicting and the comparison of these models. All programming is done in MATLAB (2018).

4.1 Regularised logistic regression

4.1.1 Logistic regression

Due to the binary nature of the data, standard regression models such as OLS do not fit. This is because OLS assumes the dependent variable is continuous. Obviously, binary data is not. To solve this, one can use the logit model. This model is based on probabilities, rather than linear relations between the dependent and explanatory variables. Specifically, it makes use of the sigmoid function. We define this function as follows. Let $i \in \{1, 2, ..., N\}$ be one of the observations from the data set. In the simulation data there is no real specification of this observation. In the real data set i corresponds to an individual. Furthermore let β be the vector of coefficients belonging to the explanatory variable vector X_i. The length of both vectors equal M. The probability that the value of the dependent variable Y_i equals one is

$$P[Y_i = 1] = \frac{1}{1 + e^{-X_i\beta}}. \tag{3}$$

Recall that this vector of coefficients is equal over the different observations. We use this probability specification to estimate the values for β.

4.1.2 Estimating parameters with regularisation

To estimate the parameter vector β we look at the standard maximum likelihood specification. To define this, we first determine the probability function for every observation i. Every observation either has the value one or minus one for Y_i. Depending on this value, the probability function either becomes $P[Y_i = 1]$ or $P[Y_i = -1]$. To incorporate this in a single function we use the indicator function. The probability function f_i of observation i becomes

$$f_i(\beta|X_i, Y_i) = P[Y_i = 1]^{I[Y_i=1]} P[Y_i = -1]^{1-I[Y_i=1]}. \tag{4}$$

Notice that f_i is a function of β, as it is the parameter vector we need to estimate. The values for X_i are known. We define f_i as the likelihood contribution of i. The maximum

likelihood function is the product of these likelihood contributions and therefore becomes

$$\mathcal{L}(\beta|\mathbf{X},\mathbf{Y}) = \prod_{i=1}^{N} f_i(\beta|X_i, Y_i) \tag{5}$$

Here, $\mathbf{X},\mathbf{Y}$ are the complete data matrix $\mathbf{X}$ and the corresponding dependent variable vector $\mathbf{Y}$, respectively. The product within the function makes it a rather complex function. Therefore it is convenient to take the logarithm, creating the log-likelihood.

$$l(\beta|\mathbf{X},\mathbf{Y}) = \sum_{i=1}^{N} \ln(f_i(\beta|X_i, Y_i)) \tag{6}$$

Standard logistic regression would maximise this function in order to get a estimation for β. A problem occurs here, however. Maximum likelihood bases its estimation on the given data set. This means the estimation can become too data-specific and coefficients tend to be much smaller or larger than needed. If one then uses this result to test other data sets the results become poor. Regularised logistic regression takes care of this. This method minimises

$$g(\beta|\mathbf{X},\mathbf{Y}) = -l(\beta|\mathbf{X},\mathbf{Y}) + \lambda \sum_{j=1}^{M} |\beta_j|. \tag{7}$$

This function consists of two parts. First all, there is the negative of the log-likelihood, also called the loss function. Where the value of the log-likelihood needs to be as large as possible, the opposite holds for the loss function. The second part is the regularisation part that solves the problem of data-specific estimation. Here, j displays the variable β_j corresponds to. It penalises a high absolute value of a coefficient and therefore restrains the estimation. The higher the value of λ, the larger that restraint is. In other words, the higher the value of λ, the less data-specific your estimation becomes. This method is also called Lasso-regularisation (Bradley et al., 2011).

One thing that we should mention is the difference between the loss function mentioned above and the one used by Chaudhuri and Monteleoni (2008). Their loss function equals

$$\sum_{i=1}^{N} \ln(1 + e^{-Y_i X_i \beta}). \tag{8}$$

This is similar to the one we specify and the outcomes of both minimisation problems are comparable. We see however, that this version of the loss function comes back in algorithm 2, which is why we mention it.

4.2 Algorithm 1

The first algorithm estimates the values for the coefficient vector β while preserving ϵ-differential privacy (Dwork et al., 2006). The algorithm makes use of Lasso-regularisation, but adds noise to the estimated coefficients. The algorithm is as follows:

- Apply regularised Lasso logistic regression on data set $\mathbf{X}$ to obtain coefficient estimator w.
- Pick a noise vector η from the Laplace distribution with mean 0 and standard deviation $\frac{2}{N\epsilon\lambda}$. This is done by picking a vector from the d-dimensional unit hypersphere and multiplying this vector with the norm $\|\eta\| \sim \text{Gam}(\text{d}, \frac{2}{N\epsilon\lambda})$. Here, d and N are the amount of variables and the amount of observations in data set $\mathbf{X}$, respectively.
- $\beta = w + \eta$

4.2.1 Proof of privacy preservation

The proof of privacy preservation is based on 2 theorems from other papers.

Firstly, let X_i be the vector of explanatory variables for observation i. Moreover let N be the total amount of observations. If $\|X_i\| \leq 1$, then the sensitivity SS of regularised logistic regression with regularisation parameter λ is at most $\frac{2}{\lambda N}$ (Chaudhuri and Monteleoni, 2008). Secondly, let $f(x)$ be the outcome of a general function. Adding noise from a Laplace

distribution with a mean of zero and a standard deviation of $\frac{SS}{\epsilon}$ preserves ϵ-differential privacy. This also holds for multivariate cases (Nissim et al., 2007).

The first step of algorithm 1 is regularised logistic regression with outcome β. We assume $\|X_i\| \leq 1$ and therefore the sensitivity of this estimator is at most $\frac{2}{\lambda \epsilon N}$. Combining this result with the second theorem, suggests that adding noise from a Laplace distribution with mean zero and standard deviation $\frac{2}{N\epsilon\lambda}$ preserves ϵ-differential privacy. This is exactly what happens in the second step of algorithm 1.

4.3 Algorithm 2

The second algorithm does not add noise to the estimated coefficients, but adds noise to the minimisation problem to preserve ϵ-differential privacy. The algorithm is as follows:

- Pick a noise vector η from the Laplace distribution with mean 0 and standard deviation $\frac{2}{\epsilon}$. This is done by picking a vector from the d-dimensional unit hypersphere and multiplying this vector with the norm $\|\eta\| \sim \text{Gam}(\text{d}, \frac{2}{\epsilon})$. Here, d is the amount of variables in data set $\mathbf{X}$.
- $\beta = \text{argmin}_w \frac{w^T w \lambda}{2} + \frac{b^T w}{N} + \frac{\sum_{i=1}^{N} \ln(1+e^{-Y_i X_i \beta})}{N}$

As one can see, the last sum of the minimization problem looks similar to the loss function in equation 8. The function is convex and we therefore expect that the solving time is similar to that of logistic regression. Theoretically, the second algorithm does not only have a better general performance, but also a better learning performance. This means that predicting using this algorithm should be more precise, and needs less data to perform well. (Chaudhuri and Monteleoni, 2008)

4.3.1 Proof of privacy preservation

Let w^* be the outcome that minimises the function in the second step of algorithm 2. Moreover, let **(X,Y)** and **(X',Y')** be two different inputs, where the only difference is exactly one observation. Let the values of this observation for the two inputs be (a, y) and (a', y'), respectively. Here too we assume $\|X_i\| \leq 1$. Due to the fact that the minimisation function

is differentiable everywhere, there is a unique value for b that matches the input **(X,Y)** with output w^*. Let b_1 and b_2 be the unique values for **(X,Y)** and **(X',Y')**, respectively. In both cases w^* is the solution to minimising the function in the second step of algorithm 2. Therefore, taking the derivative and filling in this value, results in a value of 0. This means that when doing this, the derivatives for the different inputs can be set equal, resulting in equation

$$b_1 - \frac{ya}{1 + e^{yw^{*T}a}} = b_2 - \frac{y'a'}{1 + e^{y'w^{*T}a'}}. \tag{9}$$

Because $\|X_i\| \leq 1$ it must hold that $\|a\| \leq 1$ and $\|a'\| \leq 1$. Besides, $\frac{1}{1+e^{yw^{*T}a}} \leq 1$ and $\frac{1}{1+e^{y'w^{*T}a'}} \leq 1$. Therefore $\|b_1 - b_2\| \leq 2$. Using the triangle inequality this means $-2 \leq \|b_1\| - \|b_2\| \leq 2$. From this it result follows that the ratio

$$\frac{P[\mathbf{(X,Y)}]}{P[\mathbf{X',Y'}]} = e^{-\frac{\epsilon}{2}(\|b_1\| - \|b_2\|)} \tag{10}$$

is at most e^{ϵ}. ϵ-differential privacy is therefore preserved.

4.4 k-fold cross-validation

To train and test the machine learning algorithms we make use of k-fold cross validation. Specifically, we make use of 5-fold cross validation. This method shuffles the complete data set and splits it up into five equal parts. Four of these data parts train the machine learning models. This subset is also called the batch. These are the red blocks in Figure 3. The remaining part tests the performance. We repeat this for every combination we can make with the five different data parts. Here this equals five times. The test errors of the different cross validations are added up and divided by five to get a mean test error over 5-fold cross validation. This whole process is repeated 200 times due to the fact that each algorithm has a factor of randomness.

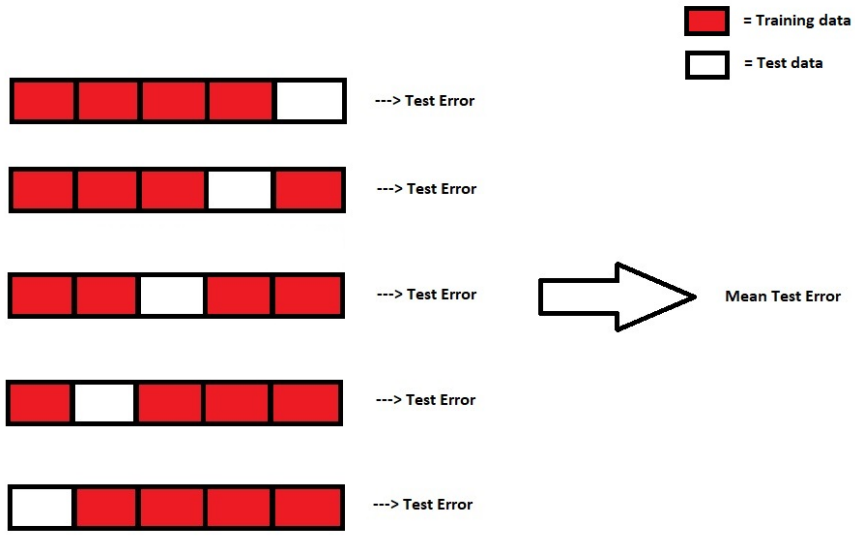

Figure 3: 5-fold cross-validation

4.5 Predicting, calculation of test error and comparison of performance

We predict according to Chaudhuri and Monteleoni (2008), using the sign-function $SGN(z)$. This function equals one if z is positive and minus one if the value is negative. In our case the function becomes $SGN(X_i\beta)$. If the sum of coefficients multiplied with their corresponding variables is negative, we predict 1 for observation i. Is this not the case, we predict -1. The test error we use is the fraction of correct predictions. Let $\hat{Y}_i$ be the prediction and $I[a = b]$ be the indicator function. We then calculate the test error with

$$\frac{\sum_{i=1}^{N} I[\hat{Y}_i \neq Y_i]}{N}. \tag{11}$$

The optimal value for regularisation parameter λ minimises this test error for regularised logistic regression.

To compare the general performance of the two algorithms, we look at the means over 5 fold cross-validation, averaged over 200 restarts. This, however, does not say anything about the learning performance. To compare and visualise this, we use plots. In the first plot

we calculate the test error for different batch-sizes of the data, increasing the training data with 1000 observations for the simulated data after every calculation. For the real data, this increment is 50. This results in a sequence of test errors belonging to different batch-sizes. The corresponding graphs reflect the learning performance. The lower the graph lies, and the faster it decreases, the better learning the performance is.

In addition, we show that both algorithms preserve privacy. We do this by calculating the test error using different sizes of ϵ. As stated in the literature section, this is the parameter that defines how much privacy has to be preserved. We show that an increase of ϵ, decreases the test error for both algorithms.

5 Results

In this section we go through the main results of our research. The computer we used for the computation of these results is a HP ENVY x360 Convertible 15-bp0xx. Table 1 displays the system details of this computer.

Table 1

Information type	**Details**
Processor:	Intel®, Core™, i7-7500U CPU @ 2.70 GHz 2.90 GHz
System type:	64-bit Operating System, x64-based processor
Software:	Windows 10
Memory:	8 GB

5.1 General performance

By minimising the test error for regularised logistic regression, we obtain an optimal value for λ of 0.01 for all data sets. We utilise this for the estimation of all coefficients.

First of all we look at Table 2 , displaying the means over 5-fold cross validation. Standard logistic regression makes perfect predictions for the inseparable data. The test error is zero. For the two privacy preserving algorithms this is different. Their mean errors differ from zero,

showing preserving privacy is at the cost of performance. Secondly, the size of the test errors is larger for the inseparable data than for the inseparable data. In other words, regardless of the method, the separable data predictions are more precise.

Even though proof by Chaudhuri and Monteleoni (2008) states that algorithm 2 has better performance the general results show that the average test errors for algorithm 1 and 2 are very similar for separable data. Their values are 0.029 and 0.031, respectively. For the inseparable data we do see a smaller test error for algorithm 2 than for algorithm 1, however.

Looking at the row of the real data we see the test errors have increased approximately with a factor of 10, compared to the simulated data. The test error of logistic regression closes down to 0.4. Nevertheless, privacy preservation causes a reduction of precision: The test errors of both algorithms are lower than that of logistic regression. Lastly, we see that algorithm 1 has a better general performance than algorithm 2, with a test error of 0.395 compared to 0,461 for algorithm 2.

Table 2

Data	**Logistic regression**	**Algorithm 1**	**Algorithm 2**
Separable data:	0	0.029	0.031
Inseparable data:	0.035	0.062	0.051
Real data:	0.391	0.395	0.461

5.2 Learning performance

Figure 4 and 5 display the learning performance of the two algorithms for the simulated data. Figure 4 displays the results for the separable data. Figure 5 displays the results for the inseparable data. As the number of observations in the training data increases, we see that the test errors of both algorithms decline, regardless of the data set we use. This means that both algorithms tend to learn better from more data. In the case of the separable data, we see the graphs of both algorithms lie on top of each other. Their performance for inseparable data seems the same, regardless of the batch size we use. So, even though Chaudhuri and Monteleoni (2008) prove algorithm 2 has a better performance guarantee than algorithm 1,

we do not conclude this here. For the inseparable data this is the case, however. Here we see the learning curve of algorithm 2 lying underneath the curve of algorithm 1, meaning that the test error of algorithm 2 is smaller. Moreover, similar to the results in Table 2, the test errors for the separable data is smaller than those of the inseparable data set.

Figure 6 shows the results for the real data set. The green graph indicates standard logistic regression. We see that for this data set, logistic regression does not work well. That is, using a small data set of 50 observations logistic regression has a test error of 0.86. In other words, averaged over 5 fold cross validation, only 14% of the predictions were predicted correctly. This test error decreases as the batch-size increases indicating logistic regression learns more from a larger data set. The learning curve is more or less decreasing for both algorithms. As the increment for the batch size of the real data is only 50, they do not decrease as fast as they do in the simulated data, however. Besides, we see that for all three methods, the test errors are larger than for the simulated data, most of the time even lying above 0.5. Nevertheless, the aspect that is truly different from the simulated data, is the fact that overall, the learning curve of algorithm 1 lies above the learning curve of algorithm 2. This means that for this data set algorithm 1 has a better performance than algorithm 2.

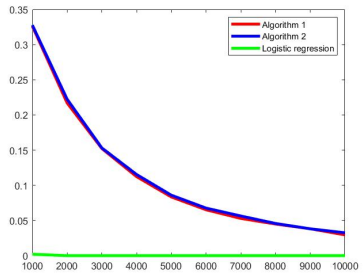

Figure 4: Test error of separable data over increments of batch-size

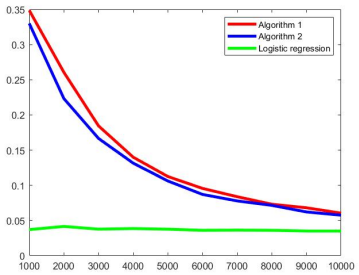

Figure 5: Test error of inseparable data over increments of batch-size

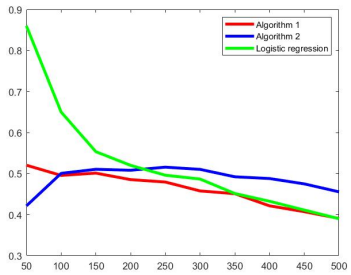

Figure 6: Test error of the real data set over increments of batch-size

5.3 Privacy and performance

Figures 7 and 8 show what happens to the test errors of the two algorithms when the values of ϵ vary. As the value of ϵ increases, the test errors become smaller. This results in a downward sloping graph. In other words, the less restriction there is on privacy, the better

the test results are. This accounts for both algorithms. It is not necessarily the case one algorithm improves more as ϵ increases. For separable data the curves are once again similar for both algorithms. For inseparable data we see the curve of algorithm 2 is the lowest. In other words, no matter how tight the restriction of privacy preservation is, algorithm 2 makes better predictions than algorithm 1 for the inseparable data.

Figure 9 displays the test errors of the real data set for different values of ϵ, averaged over 5-fold cross-validation. Similar to the simulated data, both curves decrease as ϵ increases. Yet, the values of the test errors are larger than for the simulated data. Moreover, the curve of algorithm 1 lies above algorithm 2 for all values of ϵ. So, no matter how tight the restriction of privacy preservation is, algorithm 1 makes better predictions than algorithm 2 for this real data.

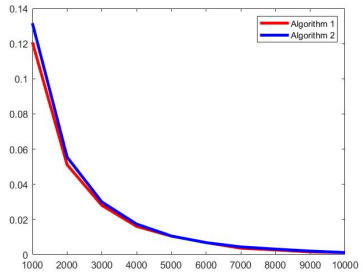

Figure 7: Test error of separable data over increments of ϵ

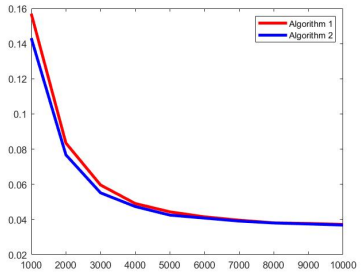

Figure 8: Test error of inseparable data over increments of ϵ

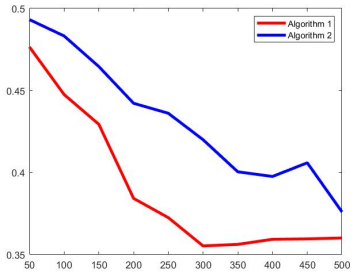

Figure 9: Test error of the real data set over increments of ϵ

6 Conclusion and discussion

The main research question of this research is: "How do we efficiently and realistically preserve privacy for binary classification data?". After defining what privacy is, we focus on two privacy preserving algorithms (Chaudhuri and Monteleoni, 2008). For all data sets we use, the results show that less restrictive ϵ-differential privacy, leads to an increase of performance. This indicates that the algorithms indeed take privacy into account. Moreover, standard logistic regression gives very precise predictions for the simulated data. But which of the two algorithms is better? For separable data both algorithms work equally well. No

clear distinction can be made between the test errors. However, in this case we do prefer algorithm 1 as the computation time was much less than that of algorithm 2.

For the inseparable data there was a performance difference. Using a batch size of 10.000, the average test error over 5-fold cross validation was smaller for algorithm 1 than for algorithm 2. Yet, the results show that this is not only the case for the latter. For the sequence of 10 incrementing batch sizes, the test error of algorithm 2 is smaller than that of algorithm 1. This suggests that algorithm 2 should be used for inseparable data, for which logistic regression performs well. This is in agreement with Chaudhuri and Monteleoni (2008).

Lastly, the research tests the algorithms using a real data set. The results show that standard logistic regression performs poorly. We see that its performance outclasses the performance of algorithm 2. For the whole sequence of different batch-sizes, the test errors of algorithm 1 are smaller than those of algorithm 2. This suggest that algorithm 1 should be used for real data sets where standard logistic regression performs poorly.

Our research, however, has some limitations. First of all, we came across some interpretation problems. The first part of our paper consists of the replication of Chaudhuri and Monteleoni (2008). Generally, this paper is complex, but short in terms of explanation. It was therefore really difficult to understand how their simulation was conducted. Due to this complication, we possibly simulated the data sets in a different manner than in the paper. This explains why our learning curves differ from Chaudhuri and Monteleoni (2008). Where we were supposed to see a clear difference between algorithm 1 and 2, there was none. Our simulated data has probably become "too separable" in the sense that is very easy to predict with that data. This why both algorithms work well and no clear difference between the two can be made. Nevertheless, it does not contradict the results by Chaudhuri and Monteleoni (2008) and it shows how sensitive the results for both algorithms are.

Secondly, the real data set caused some comparison problems. First of all, the data set is rather small. Therefore it becomes really difficult to compare the results from this set with the simulated data, which was 20 times as large. For example, increments of 1000 data points

were chosen for the simulated data set. Due to the dimensions of the real data set, we had to use a smaller step-size of 50. We think this also contributes to the odd results regarding logistic regression and the algorithms. In future research one could try and find larger data set, to prevent these struggles from occurring.

Thirdly, the results regarding the real set seem off. We expected that the test errors of logistic regression would always be better than those of the two algorithms, as these algorithms are more restrictive due to privacy preservation. However, we clearly see this is not the case. We think this is caused by a combination of the small sample size, the sign function used to predict and the poor performance of logistic regression on this data set. When logistic regression does not work well on a data set, the estimations for the coefficients are incorrect. In our case, these wrong estimates resulted in the same sign value for every observation. In other words, the model predicted the same value for every observation. Using only 50 observations, this lead to a very poor performance, as most of these observations contained the opposite label of what was predicted. Adding noise dials down these wrong coefficients. In this way, the sign function does no longer give one single value. This is why we think the performances of the two algorithms are better. This however, soon disappears as the batch-size increases.

Lastly, there were some problems with the computation time of algorithm 2. Even though in theory, this computation time should be close to that of logistic regression (Chaudhuri and Monteleoni, 2008), this was not the case in practice. It took much longer than expected. In future research one could use different programs other than MATLAB (2018) to see if the computation is shorter there, or try to improve the written MATLAB (2018) code, to decrease the latter.

References

Bradley, J., Kyrola, A., Bickson, D., and Guestrin, C. (2011). Parallel coordinate descent for l1-regularized loss minimization. *International Conference on Machine Learning*.

Chaudhuri, K. and Monteleoni, C. (2008). Privacy-preserving logistic regression. *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems*.

Dua, D. and Graff, C. (2020). Uci machine learning repository. `http://archive.ics.uci.edu/ml`. University of California, Irvine, School of Information and Computer Sciences.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. *Theory of Cryptography Conference*, pages 265–284.

El Naga, I. (2015). *Machine Learning in Radiation Oncology*. Springer.

Maria, A. (1997). Introduction to modeling and simulation. *Proceedings of the 1997 Winter Simulation Conference*.

MATLAB (2018). *Matlab R2018a*. The MathWorks Inc., Natick, Massachusetts.

Nissim, K., Raskhodnikova, S., and Smith, A. (2007). Smooth sensitivity and sampling in private data analysis. *D. S. Johnson and U. Feige*, pages 75–84.

Singer, S. and Conger, K. (2019). Google is fined 170 million for violating children's privacy on youtube. *New York Times*.

7 Appendix

Here one can find short descriptions for the codes that were used for the research. The complete codes can be found in a zip-file, which comes with the paper.

optParam5fold This code performs regularised logistic regression for several different values of the regularisation parameter lambda. Moreover it calculates the test error over 5 fold cross-validation and selects the value of lambda that gives the lowest value of this test error.

Lambda Gives a sequence of 25 different values of the regularisation vector lambda.

Algorithm 1 This code executes algorithm 1.

Algorithm 2 This code executes algorithm 2.

Noise This code gives the noise vector from a Laplace distribution.

minfunAlgo2 This code resembles the summation of logarithms of the minimisation function of algorithm 2.

ErrorAlgo1 Uses the results from Algorithm 1 to predict and calculate the test error over 5 fold cross-validation.

ErrorAlgo2 Uses the results from Algorithm 2 to predict and calculate the test error over 5 fold cross-validation.

randsphere Gives uniformly distributed random vectors that lie on the unit sphere.

SeparableSimulatedData Code that simulates separable data.

InseparableSimulatedData Code that simulates inseparable data.

Split 5 This code splits up the input data set in five equal parts. It then creates 5 training and test sets.