COMP 530: Data Privacy and Security Course Project: Privacy Preserved Machine Learning on Banking Dataset

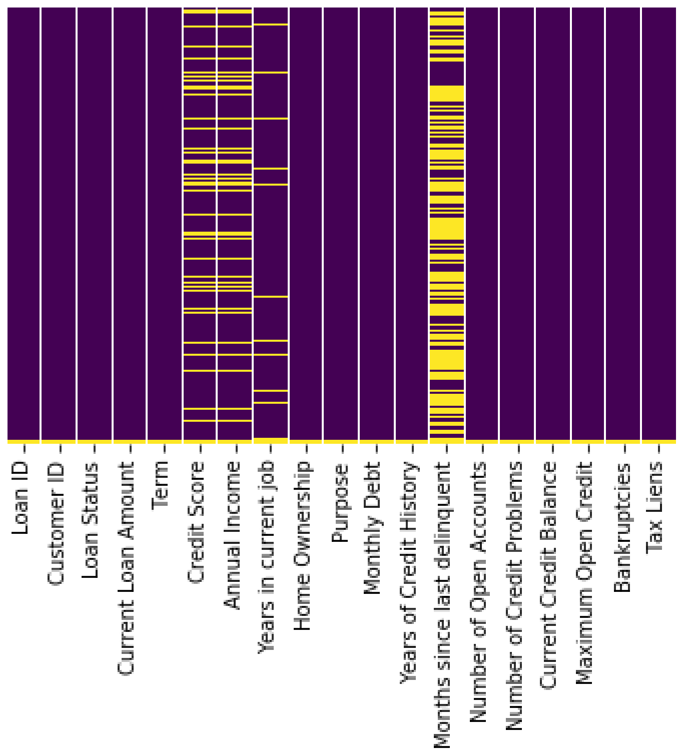
Introduction: Privacy on ML became a serious topic after the use of ML in various applications. ML became a thing we used in our everyday life. To give examples, speech recognition is an ML application. Apple use Siri for speech recognition. Another example is Tesla Car Company. Tesla use Neural Networks for Autopilot. From the privacy perspective, if an adversarial attacks the Tesla's machine learning system, they can make autopilot wrong decisions depending on the attack's power which could cause terrible accidents. An adversarial can attack speech recognition and can attain information which system gets from users. These are some reasons why privacy became an important issue for Machine Learning.

Purpose: Machine learning applications are heavily used in banking sector. For credit analysis, sales forecasting, fraud detection etc. Because they have huge datasets and ML can work very well on huge datasets. Hence, ML is used daily in banking sector. Therefore anonymizing and creating privacy preserved datasets are highly important issue. An adversarial can be both analyst who work in the company or outside attacker. The ML's purpose is to predict the loan status. ML predicts if the user pays or not.

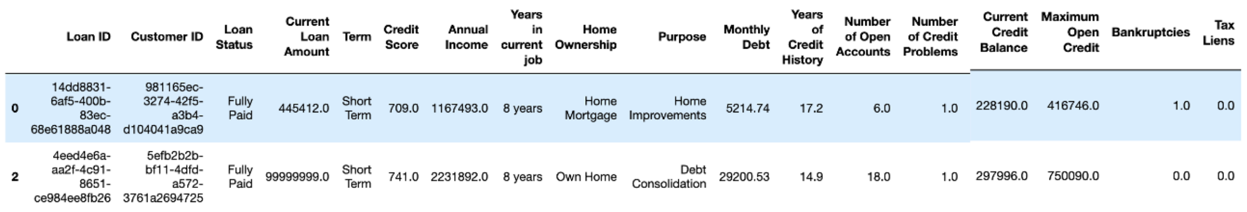
Methods and Materials: After researching several datasets on Kaggle and other platforms. We picked a dataset from Kaggle where the dataset is looking raw (Nearly no privacy methods applied). For anonymization, we did feature based anonymization. For k-anonymity, l-diversity and t-closeness, we implemented algorithm from internet. However design decisions according to our dataset, was necessary for accurate implementation. We did necessary modifications.
Our dataset: https://www.kaggle.com/zaurbegiev/my-dataset
Privacy Algorithms: https://github.com/Nuclearstar/K-Anonymity
Software Used: Python, Jupyter Notebook
Python Packages: Sklearn, Matplotlib, Pandas, Numpy, Seaborn, Scipy

Data Preprocessing & Anonymization:
Data is obtained from Kaggle and Raw data is as like below;
Dataset is imported and for the first thing we wanted to determine null values and our strategy dataset. We saw that feature 'Months since last delinquent' has very high null values. Thus, we dropped this feature from our dataset column wise. For other features we dropped them row wise.

Null values are represented in yellow



Our dataset and our features mainly like this;



	Loan ID	Customer ID	Loan Status	Current Loan Amount	Term	Credit Score	Annual Income	Years in current job	Home Ownership	Purpose	Monthly Debt	Years of Credit History	Number of Open Accounts	Number of Credit Problems	Current Credit Balance	Maximum Open Credit	Bankruptcies	Tax Liens
0	14dd8831-6af5-400b-83ec-68e61888a048	981165ec-3274-42f5-a3b4-d104041a9ca9	Fully Paid	445412.0	Short Term	709.0	1167493.0	8 years	Home Mortgage	Home Improvements	5214.74	17.2	6.0	1.0	228190.0	416746.0	1.0	0.0
2	4eed4e6a-aa2f-4c91-8651-ce984ee8fb26	5efb2b2b-bf11-4dfd-a572-3761a2694725	Fully Paid	99999999.0	Short Term	741.0	2231892.0	8 years	Own Home	Debt Consolidation	29200.53	14.9	18.0	1.0	297996.0	750090.0	0.0	0.0

First we need to introduce our dataset, here our attributes:

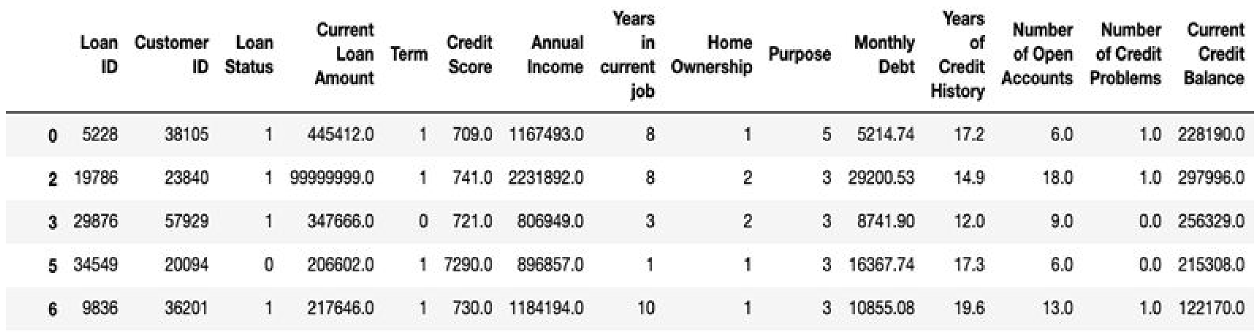
- Unique Identifiers: Loan ID, Customer ID
- Quasi-identifiers: Current Loan Amount, Term, Credit Score, Annual Income, Years in current job, Home Ownership, Purpose, Monthly Debt, Years of Credit History, Number of Open Accounts, Number of Credit Problems, Current Credit Balance
- Sensitive Attribute: Loan Status

After cleaning our dataset, we could start anonymizing our dataset. Now we would like to talk about generalization, generalization techniques achieve anonymization by reducing the detail of the original data. They can be used on numerical, categorical, and date type attributes. we used different types of anonymization while generalizing. For numerical and categorical attributes, generalization means replacing the original value with a discretized version of it. This discretized version is usually a range of values, or a timelapse in date type cases.

Our dataset has many features and all of them need special work on them. That's why we worked on data features one by one. Basically, it was necessary to look at every data feature, for numerical data, we created histograms and searched for the best bin for every feature. For discrete data, we showed the unique values and their distribution and according to this distribution we generalized our dataset.

We don't want to lose information very much, that's why we used 'mostly mean' of the features. For instance, for 'Credit Score', between 700 and 718 we used the mean of these features.

After pre-processing this dataset we get the anonymized below.



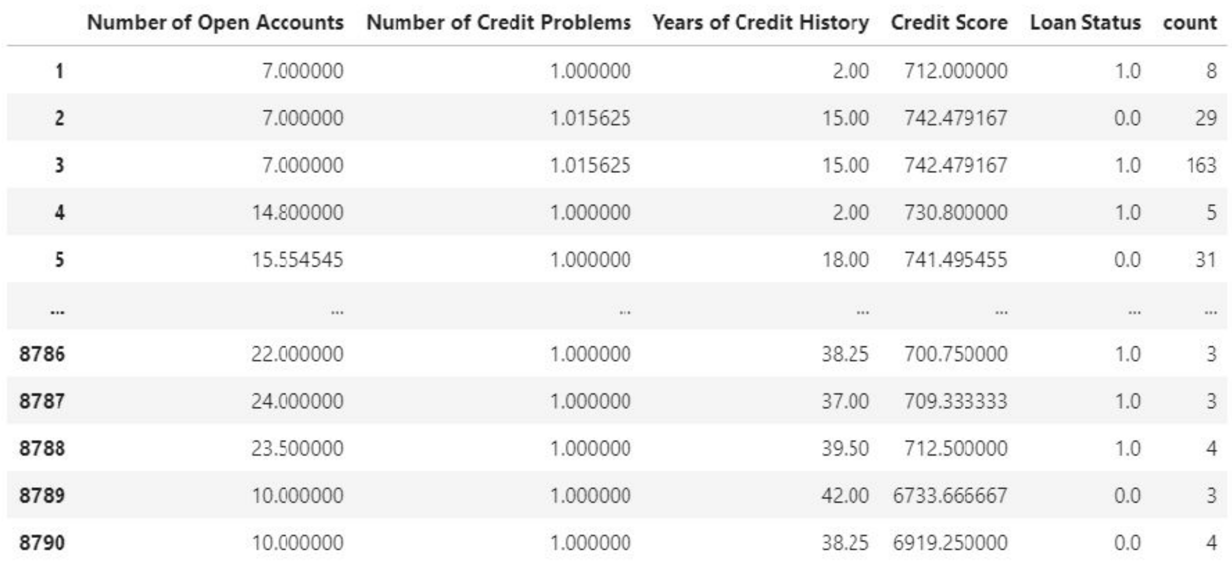
	Loan ID	Customer ID	Loan Status	Current Loan Amount	Term	Credit Score	Annual Income	Years in current job	Home Ownership	Purpose	Monthly Debt	Years of Credit History	Number of Open Accounts	Number of Credit Problems	Current Credit Balance
0	5228	38105	1	445412.0	1	709.0	1167493.0	8	1	5	5214.74	17.2	6.0	1.0	228190.0
2	19786	23840	1	99999999.0	1	741.0	2231892.0	8	2	3	29200.53	14.9	18.0	1.0	297996.0
3	29876	57929	1	347666.0	0	721.0	806949.0	3	2	3	8741.90	12.0	9.0	0.0	256329.0
5	34549	20094	0	206602.0	1	7290.0	896857.0	1	1	3	16367.74	17.3	6.0	0.0	215308.0
6	9836	36201	1	217646.0	1	730.0	1184194.0	10	1	3	10855.08	19.6	13.0	1.0	122170.0

K-anonymity: After doing all these anonymization methods, we need to ask ourselves how secure is the anonymized dataset? To answer this question we calculated the k-anonymity, l-diversity and t-closeness values for each equivalence class. Following graphs are demonstrations of K-anonymous dataset where k is 3.

As we have multiple numeric features anonymizing them is a NP-hard problem. Because when you take all features it is very hard to achieve k-anonymity and as it takes a very long time. So we specified the necessary features that could affect our ML and tried to achieve a best score with these features.

The K-anonymous dataset can be seen below. ‘Count’ feature basically shows how many instances it has in a specific feature.

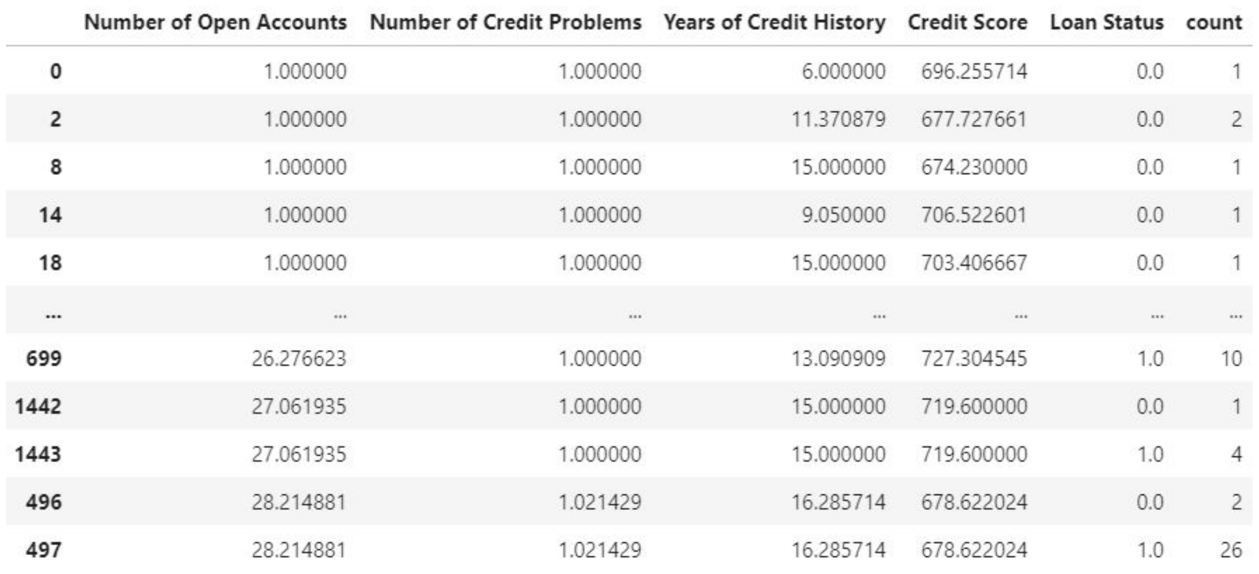
K-anonymous Dataset (k = 3)



	Number of Open Accounts	Number of Credit Problems	Years of Credit History	Credit Score	Loan Status	count
1	7.000000	1.000000	2.00	712.000000	1.0	8
2	7.000000	1.015625	15.00	742.479167	0.0	29
3	7.000000	1.015625	15.00	742.479167	1.0	163
4	14.800000	1.000000	2.00	730.800000	1.0	5
5	15.554545	1.000000	18.00	741.495455	0.0	31
...	...	...	...	...	...	...
8786	22.000000	1.000000	38.25	700.750000	1.0	3
8787	24.000000	1.000000	37.00	709.333333	1.0	3
8788	23.500000	1.000000	39.50	712.500000	1.0	4
8789	10.000000	1.000000	42.00	6733.666667	0.0	3
8790	10.000000	1.000000	38.25	6919.250000	0.0	4

L-diversity: L-diverse dataset as shown below, we wanted to achieve l-diversity on our sensitive attribute ‘Loan Status’.We determined l-valıue of our dataset as 2.

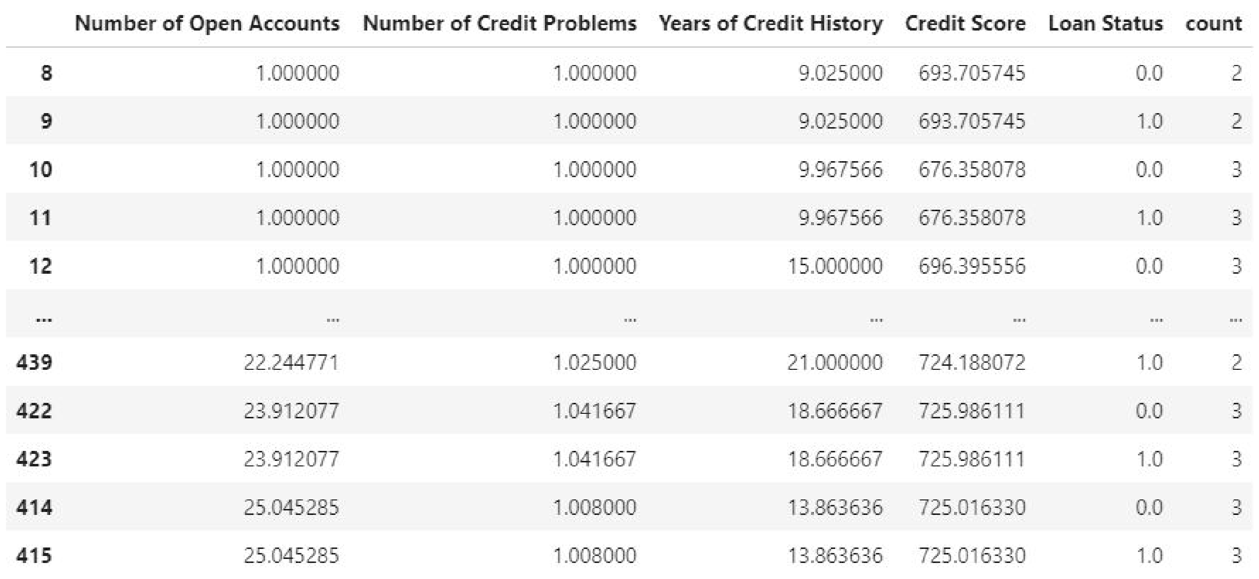
L-diverse Dataset (l = 2)



	Number of Open Accounts	Number of Credit Problems	Years of Credit History	Credit Score	Loan Status	count
0	1.000000	1.000000	6.000000	696.255714	0.0	1
2	1.000000	1.000000	11.370879	677.727661	0.0	2
8	1.000000	1.000000	15.000000	674.230000	0.0	1
14	1.000000	1.000000	9.050000	706.522601	0.0	1
18	1.000000	1.000000	15.000000	703.406667	0.0	1
...	...	...	...	...	...	...
699	26.276623	1.000000	13.090909	727.304545	1.0	10
1442	27.061935	1.000000	15.000000	719.600000	0.0	1
1443	27.061935	1.000000	15.000000	719.600000	1.0	4
496	28.214881	1.021429	16.285714	678.622024	0.0	2
497	28.214881	1.021429	16.285714	678.622024	1.0	26

T-closeness: T-close dataset can be seen in the graph. We created a t-close dataset. To measure diversity, calculate the Kolmogorov-Smirnov distance between the empirical probability distribution of the sensitive attribute over the entire dataset vs. the distribution over the partition.

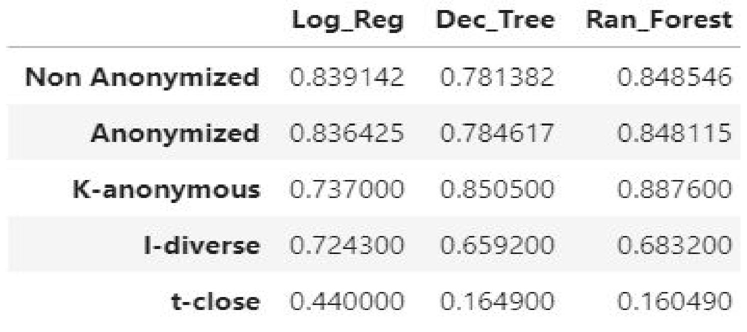
T-close dataset



	Number of Open Accounts	Number of Credit Problems	Years of Credit History	Credit Score	Loan Status	count
8	1.000000	1.000000	9.025000	693.705745	0.0	2
9	1.000000	1.000000	9.025000	693.705745	1.0	2
10	1.000000	1.000000	9.967566	676.358078	0.0	3
11	1.000000	1.000000	9.967566	676.358078	1.0	3
12	1.000000	1.000000	15.000000	696.395556	0.0	3
...	...	...	...	...	...	...
439	22.244771	1.025000	21.000000	724.188072	1.0	2
422	23.912077	1.041667	18.666667	725.986111	0.0	3
423	23.912077	1.041667	18.666667	725.986111	1.0	3
414	25.045285	1.008000	13.863636	725.016330	0.0	3
415	25.045285	1.008000	13.863636	725.016330	1.0	3

Machine Learning: So far, we have introduced the big picture of anonymization, as well as some privacy preservation models. Now, we are ready to ask ourselves: how does data anonymization affect Machine Learning models? How does accuracy change with the use of anonymized training or test data?
Firstly, machine learning models are created and applied for no-privacy preserved ML. After that we implement our algorithms on just generalized data. Then respectively, we implemented these algorithms on k-anonymous, l-diverse and t-close datasets.
For Machine Learning model; 3 different models have been used. They are Logistic Regression, Decision Tree Classifier and Random Forest Classifier.

Model Performances;



	Log_Reg	Dec_Tree	Ran_Forest
Non Anonymized	0.839142	0.781382	0.848546
Anonymized	0.836425	0.784617	0.848115
K-anonymous	0.737000	0.850500	0.887600
l-diverse	0.724300	0.659200	0.683200
t-close	0.440000	0.164900	0.160490

Outcome and Conclusion: Below it shows the results of 3 models on 5 different privacy-preserved datasets. Let's interpret our scores;

1. Anonymized vs Non Anonymized (columns 1 and 3): They both nearly have the same scores. So by anonymizing data carefully without using large bins we can both increase some privacy and we don't lose model accuracy. Hence, we can highly recommend generalization type anonymization instead of using true values.

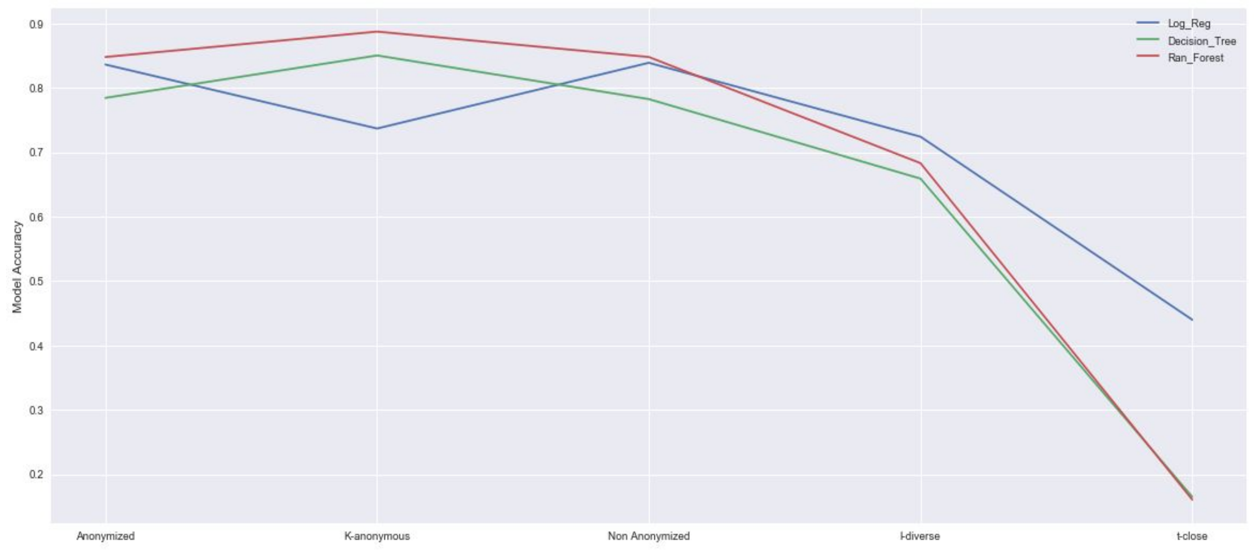
2. Non-anonymized vs K-anonymous: We can see that the score of k-anonymous dataset increases on Random Forest and Decision Tree algorithms but for Logistic Regression it has decreased. The reason behind this probably, Decision Tree and Random Forest models are complex models and at some point they cross the optimum point for bias-variance tradeoff and they start to overfit with big data. With k-anonymity we simply make data more generalized. That reduces overfitting and increases model performance. The highest accuracy values have been seen in k-anonymous data. So we can conclude that for complex models it is highly recommended to use k-anonymity algorithms as they both gain performance and increase privacy.

3. L-diversity vs K-anonymity: We set l-diversity equal to 2 and calculate our scores for 3 models. There is nearly a 20% accuracy decrease. It can be discussed to achieve l-diversity as model performance decreases 20% and it is not a value to overlook. Even very little decreases in ML models can be crucial. From my perspective trade-off is large in case of utility.

4. T-closeness vs L-diversity: As we can see below, model performance with achieving t-closeness values are the lowest. We can confidently say, for ML models it's not appropriate to achieve t-closeness with low values like 0.16 for ML accuracy.

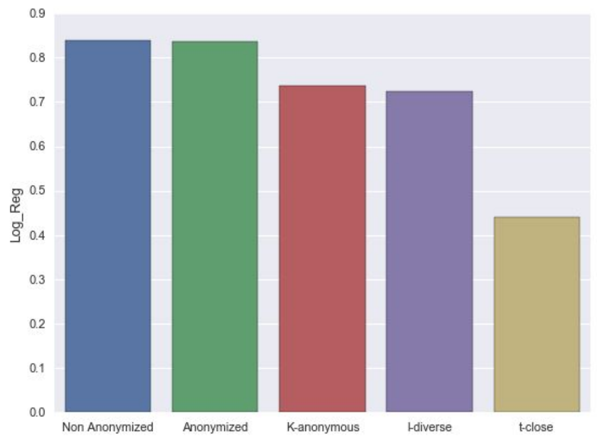
Conclusion and Feature Work: In conclusion, we applied privacy techniques and rendered five ML models. Hence, for some results; we are surprised and for some results it's exactly like we expected. For instance it was surprising to see accuracy increase while we increased privacy at K-anonymity. For future work, models can be optimized with different methods. For different k-anonymity values models can be run and with performance comparison we can achieve the best k value for this particular dataset.

3 model performance at 5 situation

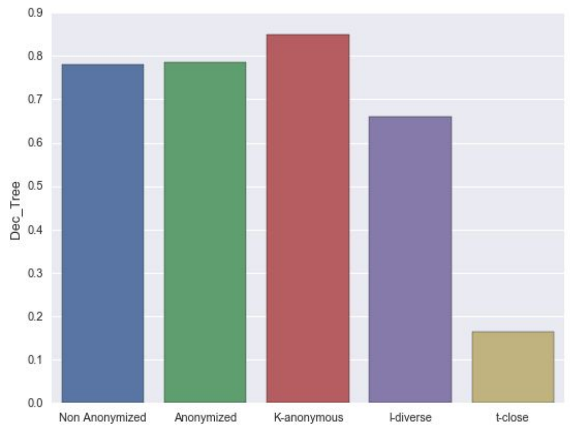


Three model performances individually can be seen below.

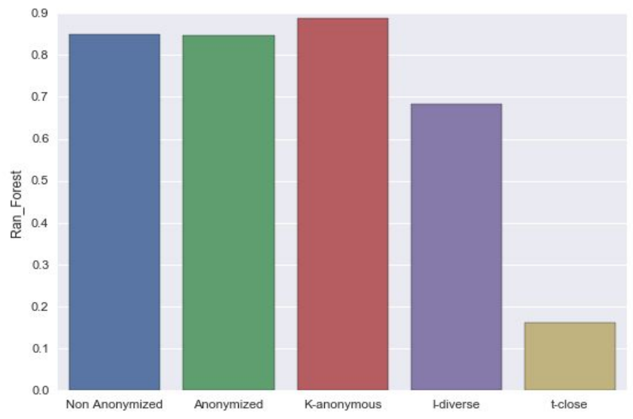
Logistic Regression Performance



Decision Tree Performance



Random Forest Performance


Ran_Forest
Non Anonymized
Anonymized
K-anonymous
l-diverse
t-close