A Differential Privacy Algorithm for Real Estate Brokerage Transaction Data

Christopher Eads, Nam Tran, Melissa Wong

October 14, 2020

STATISTICS
TEXAS A&M UNIVERSITY
online learning

Contents

List of Figures

List of Tables

1 Overview

Lone Wolf Technologies, a software company specializing in solutions for real estate brokerages, is interested in providing benchmarking metrics to their clients in the form of boxplots, generated using data from those same clients. There is some concern regarding maintaining confidentiality at the level of the individual brokerage while still providing accurate and meaningful benchmarking data. As such, a project request was made to provide recommendations on how to measure and formalize the level of privacy maintained when providing this benchmarking data to clients.

1.1 Scope

- The project focuses on data from brokerages in California (whereas the full data includes all of the United States and Canada).
- The project implements the Laplace mechanism, a formal privacy method that is general enough that the company can later adapt it on a larger scale, and simple enough that it can be implemented without an advanced data science team after this initial project is complete.
- One of the fundamental concepts of differential privacy is that as the number of queries on the data increases, the amount of privacy loss increases. For this project, it is assumed that the number of queries will be limited to some small number (e.g. 5), mitigating this risk.

1.2 Results

There are four key outcomes for this project:

- We use a toy example to illustrate how non-privatized data can be backed out to obtain confidential information when the results are exact, necessitating some masking mechanism (we use a noise-generating mechanism called the Laplace mechanism). Furthermore, exact results can be backed out even with added noise via averaging over multiple calls to the data, necessitating limits to the number of calls which can be made. Calculating this limit is outside of the scope of this project, as the use case doesn't require multiple calls at this time.
- We implement the Laplace mechanism on the quantiles of the data to produce ’privatized’ boxplot results. To do this, we estimate the sensitivity of the data (i.e., how much difference does the presence / absence of a single row make?) using sampling from a gamma distribution that approximates the distribution of the data.

- We show how the accuracy of the results changes as the privacy loss parameter ϵ is varied and offer general guidance on how to choose ϵ in the future.
- Based on a choice of ϵ, we illustrate how the results might vary from one iteration to the next.

Additionally, while not a key outcome of the paper, we implement a greedy clustering algorithm before any differential privacy mechanism is applied. This gives us samples of the full dataset while also paving a way for dynamically grouping data in the future, should the company decide to use clusters more granular than state-wide.

2 Differential Privacy Summary

Differential privacy (DP) is a broad category of tools and methods used to share aggregated data while still ensuring a specified and measurable level of privacy at the individual level. This is an important requirement when sharing many types of data, as individual records could be confidential or potentially damaging to the individual if made public (e.g., private medical or financial records).

2.1 Privacy Budget and Sensitivity

A core tenet of differential privacy methods is the concept of a measurable, mathematically defined privacy budget, commonly denoted as ϵ. This is a user-specified input in the algorithm that represents the level of privacy to be maintained when the aggregate results are supplied (i.e., the smaller the value of ϵ, the smaller the amount of maximum privacy loss). This ϵ is implemented in different ways for different algorithms. For simple DP mechanisms like noise-generating mechanisms, increasing ϵ means decreasing the amount of noise added to the aggregation function. The choice of ϵ also affects the total number of calls a user is permitted to make on the data to prevent them from simply averaging the noise-added results to find the true result.

Another variable in the DP equation that affects the accuracy of output is the sensitivity. This concept of sensitivity is distinct from the more commonly known sensitivity/specificity definitioins in binary classification. For any given function (e.g., calculating the mean for a group), there is some level of sensitivity, which is a representation of the amount of influence a single record or row has on the output of the function (e.g., how much can the mean of some Group A change as a result of removing any individual member of that group?).

Formally, we can say the sensitivity, Δf, can be defined as:

$$\Delta f = max\|f(D_1) - f(D_2)\|_1$$

where D_1 and D_2 are two datasets that differ in row and f the function applies to the data (e.g. the median)

Thus, while ϵ is a choice made by the provider of the aggregated results, the sensitivity is a quality inherent to the type of aggregation being done.

For some simple cases, sensitivity can be calculated directly. However, for many cases a closed form is not currently known. In such instances, an empirical estimate can be obtained using repeated sampling, resulting in an estimate of ϵ. Thus the distinction is made between ϵ-DP and RDP (Random Differential Privacy). (Rubinstein and Aldà 2017b)

Typically, several resulting datasets are produced with varying levels of ϵ and results are compared. The final choice of ϵ is often more a business intelligence decision than a mathematical one; being able to make this choice is seen as a desirable characteristic of differential privacy methods.

There are many open source differential privacy libaries. All examples shown in this report used the *diffpriv* R package which implements RDP (Rubinstein and Aldà 2017a). For a comparison of several libraries and why we chose *diffpriv* for this project, see Appendix A.

2.2 Mechanisms

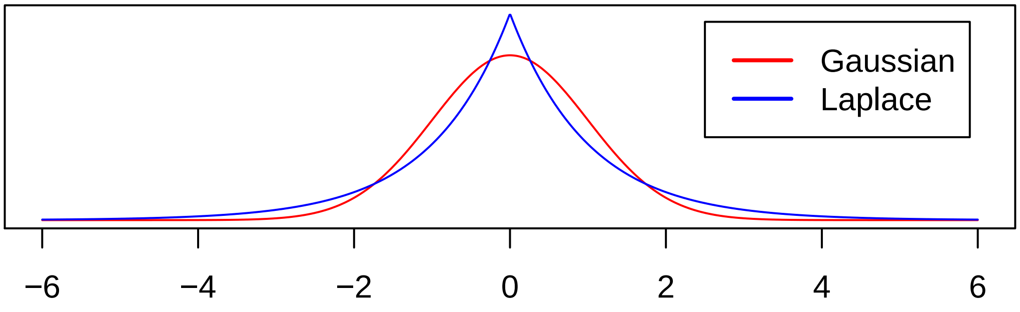


Figure 1: Noise Generating Mechanisms

The differential privacy method used for this project is the Laplace mechanism, which falls into the category of noise-generating mechanisms (or additive noise mechanisms). Some specified level of random noise generated from a Laplace distribution is added to the raw aggregated results, which makes backing out the data more challenging to some measurable degree. A smaller privacy loss budget (higher level of privacy ensured) requires more noise added to the resulting data as it is aggregated. Calculating a function with a higher sensitivity also requires more noise.

The Laplace mechanism is the most commonly used noise-generating mechanism, but the closely related Gaussian mechanism is also commonly used. The Laplace mechanism, also known as the double exponential mechanism, is more heavily peaked than the Gaussian mechanism, and also has heavier tails (Figure 1). Sensitivity in the Laplace mechanism is measured by the l_1 norm while the Gaussian mechanism measures sensitivity with the l_2 norm. This distinction matters when statistical inference and prediction will be done after implementation of the differential privacy method. Since there is no requirement for that with this project, the difference is negligible.

$$M_{\text{Laplace}}(x, f, \epsilon) = f(x) + \text{Lap}(\mu = 0, b = \frac{\Delta f}{\epsilon})$$

We use this same l_1 norm to fit the parameter ϵ, by plotting the Mean Absolute Deviation around the Median (MAD Median), as this metric is the MLE of the scale parameter of the Laplace distribution:

$$\text{MAD median} = \hat{b} = \frac{1}{N}\sum_{i=1}^{N} |x_i - \hat{\mu}|$$

Additionally, we use a scale-free version, which we'll call accuracy:

$$\text{Accuracy} = 1 - \frac{\hat{b}}{\hat{\mu}}$$

This measure of accuracy is only meaningful locally, where $|x_i - \hat{\mu}| < \hat{\mu}$, otherwise accuracy becomes a negative percentage. This restriction is acceptable, however, as we are only interested in values of ϵ that result in an accuracy relatively close to 100%.

2.3 Example

A trivial example will illustrate how differential privacy protects individual data from being extracted from aggregated data. Consider the following set of sales volume data for five brokerages in Table 1.

Table 1: Example Data

Brokerage	Sales Volume
A	23512
B	76543
C	33876
D	56899
E	71432

Assume brokerage B knows there are 5 brokerages that operate in the same region. Futher, brokerage B believes brokerage E is their main competitor and wants to determine E's exact sales volume. Even if only mean sales volume queries are allowed, it would still be possible for B to determine E's sales volume with a pair of queries.

First, query the entire region which results in $SV1 = 52452.4$. Next, query the region but exclude E which results in $SV2 = 47707.5$. Finally, E's sales volume is equal to $5 * SV1 - 4 * SV2 = 71432.0$.

However, if differential privacy is employed when calculating the mean, it is only possible to *approximate* E's sales volume with the same two queries. Note that it is possible to average out the noise with repeated queries; the privacy budget ϵ limits the number of allowable repititions. See Figure 2 as an example of the relationship between ϵ and the number of cumulative repetitions required to average out the added noise.

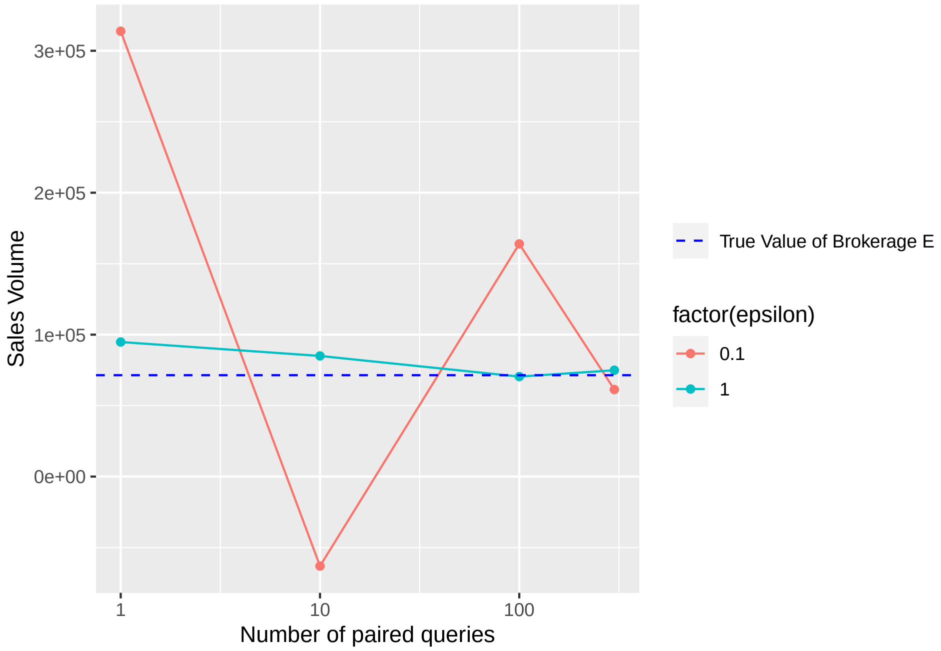


Figure 2: Effect of Privacy Budget on Repeated Query Limit

3 Data

The data consists of deal records from brokerages in California over the last 18 months. Deals consist of two sides (the buyer and the seller), of which the brokerage may represent only one side, or possibly both (a so-called "double ender"). Multiple real estate agents may be on one side of a deal.

Of primary interest are the performance of (1) individual real estate agents, and (2) of offices as a whole. As such, each row of data has performance metrics for a specific agent on a specific side of any given deal. As a result, a deal will spread across multiple rows of data if there are multiple agents and/or multiple sides on the deal for a given brokerage.

The specific metrics of interest are Sales Volume (SV) and Gross Commission Income (GCI), both of which are used to measure agent performance. SV is a "credit" metric used to split up the sale price of the deal among all of the agents on the deal. Similarly, GCI is a "credit" metric used to split up the total commission on the deal among agents (note that this doesn't necessarily reflect the commission that individual agents actually receive; it is simply a way to assign credit).

- Total SV per office
- Total SV per agent
- Total GCI per office

- Total GCI per agent

For this project, we focus exclusively on grouping offices (and employees-within-office) by region (see Appendix B for details on the grouping algorithm). Brokers are interested in seeing how their office compares to other offices in their region, and how their agents compare to other agents in their region. However, as stated above, the goal is to implement a general enough privacy algorithm that it can be adapted to other grouping types (e.g, brokerage size, market type) in the future with simple feature engineering.

One other important note is that there are many agents in the data who have \$0 for Sales Volume, GCI, or both. Including these agents in the analysis doesn't work well, as it often results in the median being equal to the minimum. However, it is common practice in the real estate industry to exclude these \$0 agents and only view 'productive' agents. Thus, for the purposes of this study, we excluded the \$0 agents.

4 Results

In this section, we focus solely on Sales Volume. Similar results for GCI are provided in Appendix C. We use two clusters of offices, one small (5 offices, 112 agents) and one large (21 offices, 374 agents), to highlight how results vary as n changes.

4.1 Evaluating MAD and Accuracy as Epsilon Varies

The client seeks to display boxplots of privatized data. Thus, we use the Laplace mechanism on five quantiles (minimum, 25%, median, 75%, maximum) to produce boxplots. Implementation of such an algorithm requires tuning in several areas. We add a constraint that the resulting quantile values be monotonically non-decreasing. We tune the sensitivity using Random Differential Privacy based on samples from a $gamma(\alpha = 3, \theta = mean(X)^{3/4})$, where the number of samples corresponds to the number of records in the dataset. The gamma distribution was chosen for several reasons:

- The sales data is strictly positive and heavy-right-tailed, which the gamma distribution easily approximates.
- Choosing a shape parameter of 3 yields a moderately 'non-informative' distribution. It allows for a non-zero maximum while still being relatively right-skewed.
- Choosing the scale parameter equal to the mean of the data to the $\frac{3}{4}$ power accomodates the fact that sensitivity is scale-variant.

Since our model is not an exact ϵ-DP model, but rather an RDP model, we simply require a randomness generator that generally approximates the sensitivity of the dataset; it is not expected to be exact. Further tuning can be addressed by modifying the privacy loss parameter ϵ. This can be done using charts which show how Mean Absolute Deviation around the median (MAD Median) varies with the choice of ϵ (Figure 3).

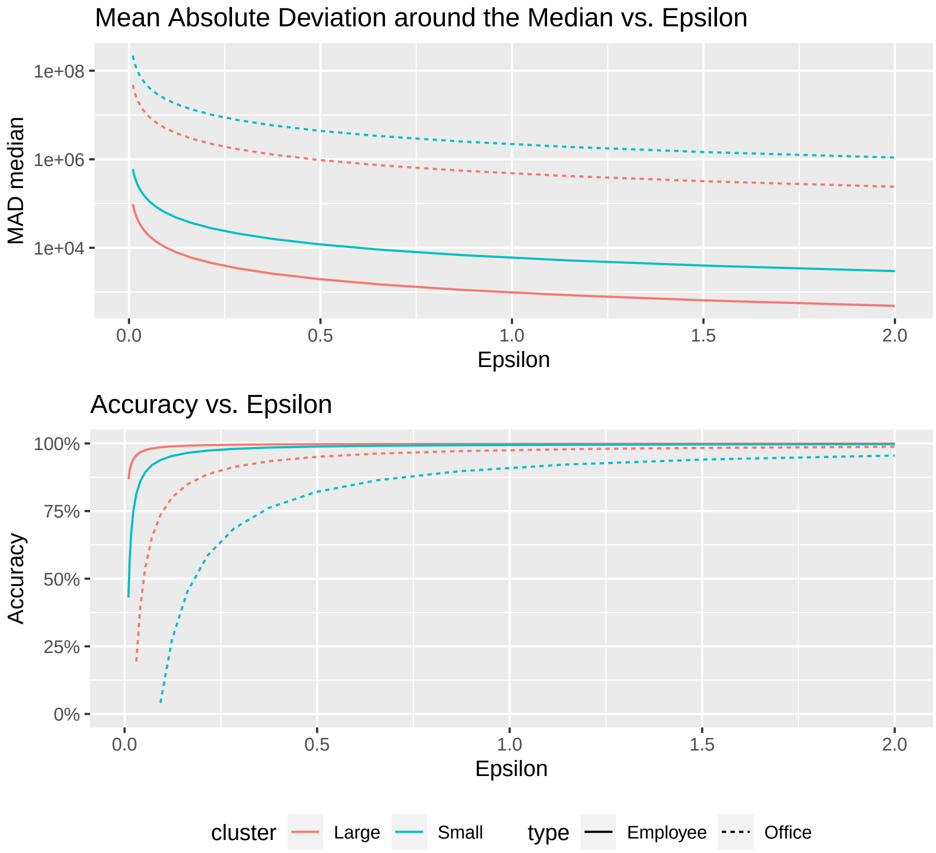


Figure 3: Plotting Mean Absolute Deviation around the Median as a function of Epsilon

Calculating MAD Median and Accuracy for these four groups highlights several important results.

- For a given value of ϵ, the absolute amount of noise added varies with the estimated sensitivity of the data.
- For a given value of ϵ, the relative amount of noise added varies with the estimated sensitivity *as well as* with the overall magnitude.
- A given value of ϵ may not yield sufficiently accurate results in some cases, and may yield overly accurate results in others, where the desired level of accuracy is more a abusiness intelligence question than a statistical question.

For the office-aggregated results of the smaller cluster with only ($n = 5$ offices, there is much more privacy loss for a similar amount of accuracy (or conversly much lower accuracy for a similar amount of privacy loss). Table 2 summarizes the results for a vertical slice at $\epsilon = 0.5$. Notice the significant loss in accuracy for the office grouping of the small cluster. This is due to the data having much higher sensitivity as a result of the much smaller n.

Given the current scope restriction that the end user will not be able to query the data repeatedly, a larger ϵ could be chosen that gives an accuracy for all four of these groups. If the query limit changes, a more formal calculation of the privacy loss for the full continental data set is recommended.

Table 2: MAD Median and Accuracy at Epsilon=0.5

MAD	accuracy	cluster	type
1965.606	0.9973483	Large	Employee
11992.099	0.9885227	Small	Employee
966371.955	0.9503936	Large	Office
4405143.454	0.8200538	Small	Office

Similarly, note how much ϵ varies for an accuracy of 95% in Table 3.

Table 3: MAD Median and Epsilon at Accuracy=0.95

epsilon	MAD	cluster	type
0.0305090	31956.91	Large	Employee
0.1230160	48353.67	Small	Employee
0.4960161	966371.95	Large	Office
2.0000000	1092511.07	Small	Office

4.2 Boxplots and the Variation Between Iterations

Based on the above results, we chose $\epsilon = 0.5$ for all of subsequent boxplots. The office results will be less accurate, on average, than the employee results. Similarly, the small cluster results will be less accurate, on average, than the large cluster results. This makes sense, as n is smaller for the office groupings than for the employee groupings, and smaller for the small cluster than for the large cluster. These plots show the sort of variation we would expect to see.

4.2.1 Large Cluster, Grouped by Employee

Table 4: DP Iterations on Sales Volume by Employee, Large Cluster

iteration	min	q1	median	q3	max
1	41059.32	374604.3	740684.5	1572674	43691915
2	72477.52	275854.5	716368.4	1539029	43664087
3	140061.00	361119.9	816863.1	1469412	43688770
4	61869.29	360669.3	719497.5	1571726	43703662
5	56938.88	336891.7	742378.2	1549709	43748699
True	40000.00	375000.0	741250.0	1575500	43700000

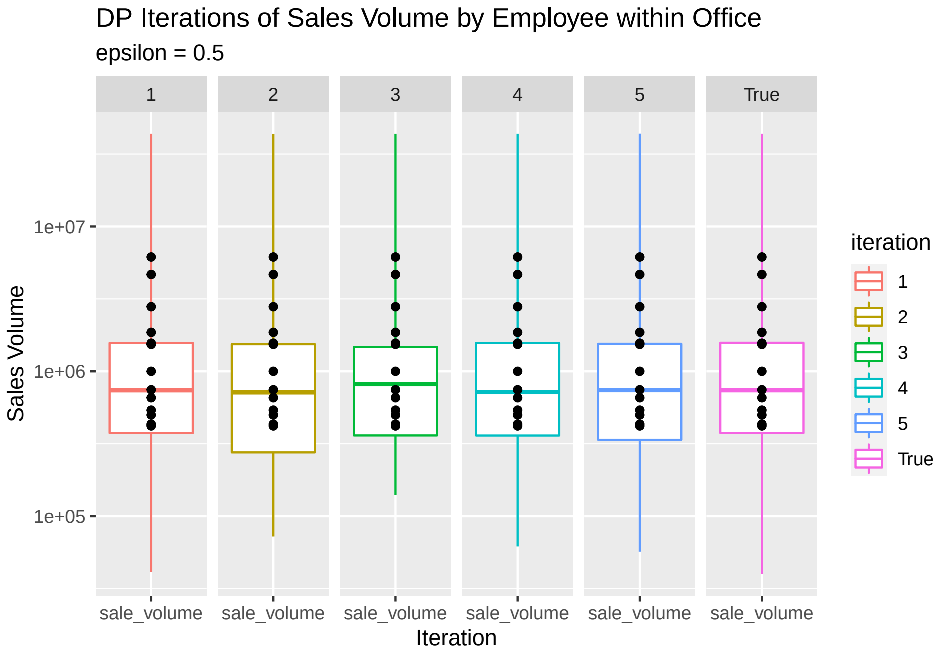


Figure 4: DP Iterations on Sales Volume by Employee, Large Cluster

4.2.2 Small Cluster, Grouped by Employee

Table 5: DP Iterations on Sales Volume by Employee, Small Cluster

iteration	min	q1	median	q3	max
1	143585.0	307160.4	1097811.4	2006706	12270177
2	229196.8	409036.5	1277023.2	1991945	12287267
3	253504.1	332712.6	1442997.1	2757911	13451326
4	184435.4	259795.7	1081095.9	2675549	12229188
5	454012.1	720219.7	935246.3	1610412	12164343
True	41400.0	460500.0	1044850.0	2008425	12269403

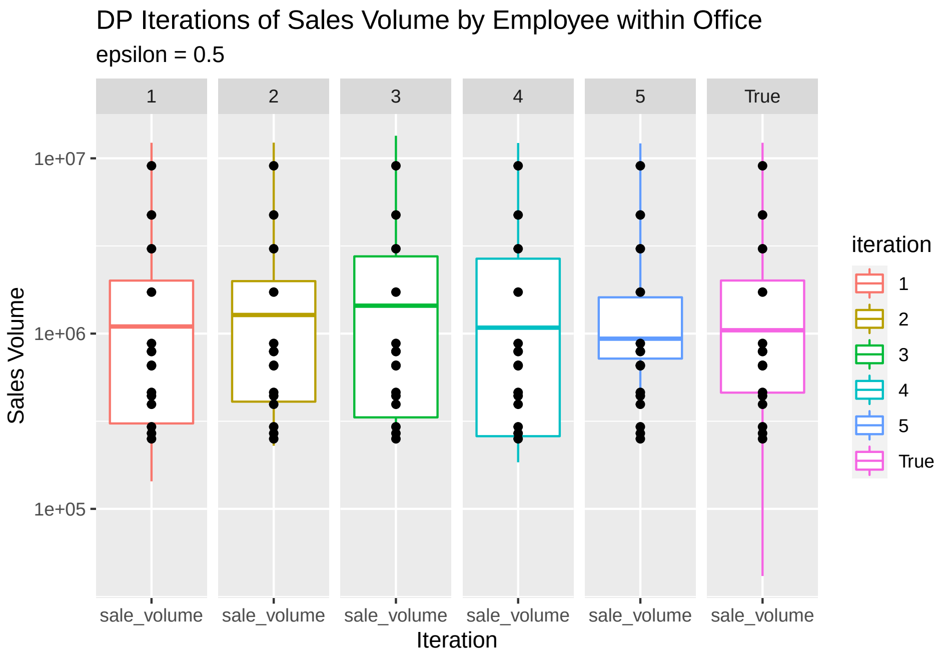


Figure 5: DP Iterations on Sales Volume by Employee, Small Cluster

4.2.3 Large Cluster, Grouped by Office

Table 6: DP Iterations on Sales Volume by Office, Large Cluster

iteration	min	q1	median	q3	max
1	2611728	10168737	14529550	28929906	75142305
2	5201715	5518318	20020744	29325564	80544019
3	3535887	6082981	25262923	31150163	66116131
4	4311825	4828337	18748025	25832476	72092156
5	3141042	4808126	6334092	29847446	66851617
True	1398000	5263087	19480791	29973000	74218892

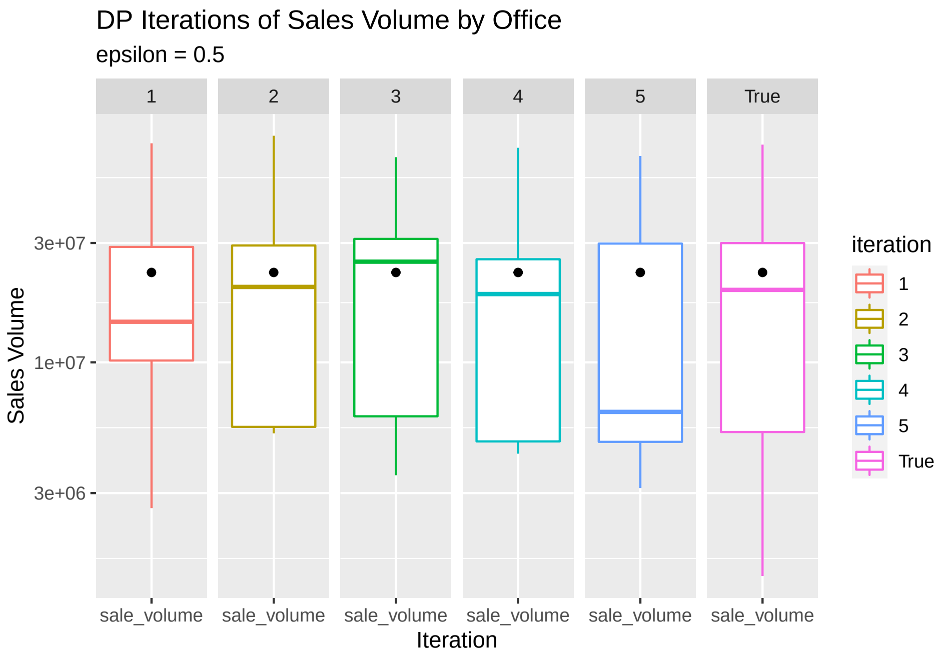


Figure 6: DP Iterations on Sales Volume by Office, Large Cluster

4.2.4 Small Cluster, Grouped by Office

Table 7: DP Iterations on Sales Volume by Office, Small Cluster

iteration	min	q1	median	q3	max
1	2668339	7288111	26957530	31605211	102934076
2	4443379	8119854	23727025	71945809	117394159
3	9831516	15103771	27306614	65945235	104637445
4	1559647	9174315	27212794	58241024	100809986
5	4620231	11054249	22618444	59850447	91302232
True	540000	4690000	24480331	58409943	105538078

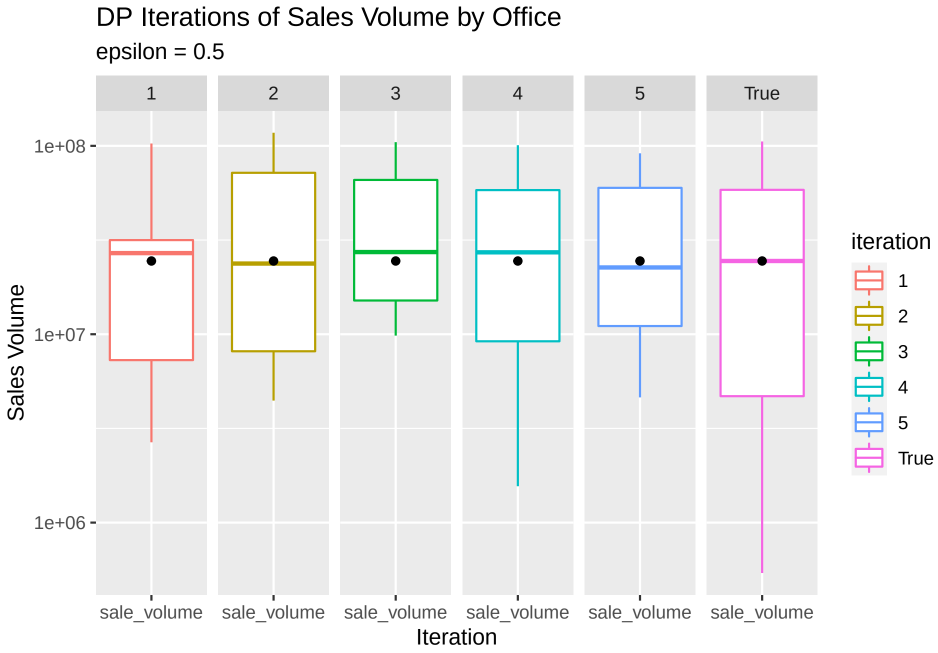


Figure 7: DP Iterations on Sales Volume by Office, Small Cluster by Office

5 Conclusion

In this paper, we implemented a differential privacy mechanism on the quantiles of Sales Volume and GCI data from the real estate market. The results still approximate the true values in the data while also generating a controllable amount of noise. This noise fluctuates as expected with n, with the overall sensitivity of the data, and with the user-controlled privacy loss parameter ϵ. Additionally, we demonstrated how this DP mechanism helps to preserve privacy and offered guidance on how to balance privacy with accuracy in the output.

5.1 Next Steps

Several interesting and important topics of exploration remain for further study.

- Currently, our model uses sensitivity sampling on a gamma distribution with a scale parameter proportional the median. A recommended addition to this approach might be to also adjust the shape parameter such that the variance of the sampling distribution more closely approximatates the variance of the data.
- Another approach to estimating sensitivity would be to sample from a function or functions of the data itself. While more computationally intensive, this may also yield a more accurate estimation of the data's sensitivity. For example, sampling separately for the sensitivity of minimum, median and maximum would allow for more flexibility in the model, as each quantile could have a separate sensitivity estimate, and a separate resulting amount of noise added.
- The Differential Privacy libraries produced by Google and IBM are more robust and have better documentation. A larger scale implementation of DP on this data might benefit from the features that come with those libraries.
- The problem of the ”overdispersed” data when \$0 agents are included remains unsolved; when the \$0 agents are not filtered out of the data, then the median of the data often equals the minimum, leading to poor DP results. A simple enough solution would be to simply display a 0 in the output when there is a 0 present in the original data, and only calculate DP estimates for the non-zero values. This makes sense, since 0 is the lower bound; if the minimum and the 25th percentile are both 0, then under certain assumptions, there would be no privacy loss by simply displaying a 0 for the minimum instead of adding noise via a DP mechanism. Assumptions might include some minimum number of non-zero offices or agents in the data set to ensure privacy.

A Open Source Differential Privacy Libraries

Below is a summary of the open source differential privacy libraries that we evaluated as part of this study.

Source	Algorithms	Supported Languages
Google https://github.com/google/differential-privacy	Count Sum Mean Variance/Std. Deviation Order Statistics Gaussian Mechanism Laplacian Mechanism	C++ Java
Uber (deprecated) https://github.com/uber-archive/sql-differential-privacy	SQL Queries	Scala
IBM https://github.com/IBM/differential-privacy-library	Histograms Gaussian Naive Bayes, Logistic Regression Linear Regression K-means PCA Standard Scaler Laplacian Mechanism Exponential Mechanism Binary Mechanism Staircase Mechnaism Uniform Mechanism Wishart Mechanism	Python
diffpriv https://cran.r-project.org/web/packages/diffpriv/index.html	Laplacian Mechanism Exponential Mechanism Gaussian Mechanism	R
OpenDP (under development) https://projects.iq.harvard.edu/opendp	TBD	TBD

For more context, we focus on three libraries we actively tried and some of our thoughts regarding them,

- `IBM's Differential Privacy Library (IBM DP)`
  - Updated as recently as July 2020.
  - Python Support Only
  - A multitude of Python notebooks demonstrating various features under the lens of pedagogy.
- `OpenDP`
  - Extremely new project that is being actively developed.
  - Core system written in Rust with R, Python, and Rust bindings.
  - Attempted to use R bindings, which involved compiling the core Rust system from scratch, which wasn't able to be done successfully.
- `diffpriv`
  - Not actively developed, with last commit from July 2017.
  - R Support Only.
  - Relatively sparse examples in R vignette as well as less of a pedagogical bent.
  - Could be arguably seen as a proof of concept and less "for production" relative to the aforementioned libraries.

On a high level,

- `IBM DP`, `OpenDP` are both more "for production" ready as well as have a multitude of examples to help drive development.
- `diffpriv` is very barebones, has minimal included examples, and has more of a "proof-of-concept" flavor.

We actively preferred `IBM DP`, but due to making the decision to use `R`, we ruled out that library. We attempted to compile `OpenDP` and use its R bindings, but we encountered insurmountable compilation errors. Ultimately, we chose `diffpriv` as the last remaining option, albeit we feel comfortable in its "optimized for correctness" approach even at the expense of pedagogy.

B Grouping Algorithm

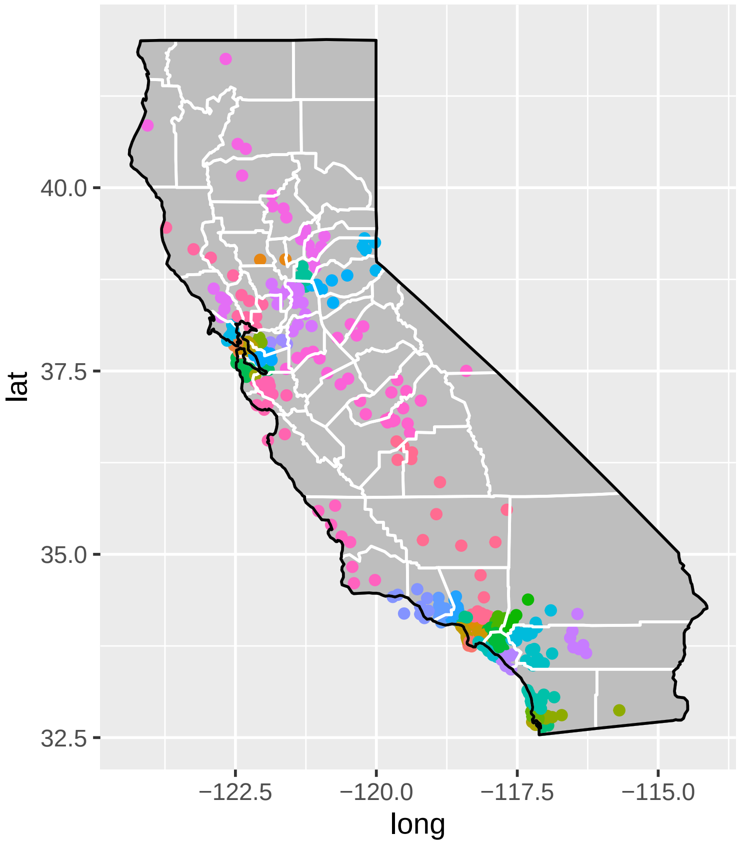
One reasonable grouping criteria is distance given the size and diversity of communities in California. For example, real estate data in highly urban areas such as Los Angelels or San Francisco differs significantly from data in more rural areas such as Bakersfield. We implemented a simple greedy algorithm that groups offices based on the minimum geodetic distance between zip codes. We choose the minimum cluster size as seven offices, but this is a parameter which can be easily changed. An R implementation of the algorithm is shown below.

Note that we implemented this as a way to simulate different “states” in our data, since we only used data from California. However, this could be implemented as a way of presenting more granular geographical benchmarking metrics to clients while still maintaing an appropriate minimum cluster size.

```r
library(geosphere)
library(sp)

# Number of offices and agents per zip code
df_clean_loc <- df_clean %>%
  group_by(office_postal_code, lat, lng) %>%
  summarise(office_count = n_distinct(office),
            agent_count = n_distinct(employee)) %>%
  mutate(cluster = NA)

# Matrix to hold distances between offices
dist <- SpatialPointsDataFrame(matrix(c(df_clean_loc$lng,df_clean_loc$lat),
                                      ncol=2),
                                data.frame(ID=seq(1:nrow(df_clean_loc))),
                                 proj4string=CRS("+proj=longlat
                                                 +ellps=WGS84
                                                 +datum=WGS84"))


nzips <- length(df_clean_loc$office_postal_code)
mdist <- distm(dist)
diag(mdist) <- NA

# Greedy clustering based on distance
min_cluster_size = 7
```

```r
# Find zip codes that already meet min_cluster_size
idx <- which(df_clean_loc$office_count >= min_cluster_size)
df_clean_loc$cluster[idx] <- seq(1, length(idx))

mdist[idx,] <- NA
#mdist[,idx] <- NA

while (any(is.na(df_clean_loc$cluster)))
{

  min_dist <- min(mdist, na.rm=TRUE)
  idx <- which(mdist == min_dist)
  ridx <- (idx[1] %/% nzips) + 1
  cidx <- idx[1] %% nzips
  if (cidx == 0)
  {
    ridx <- ridx - 1
    cidx <- nzips
  }
  mdist[ridx, cidx] <- NA
  mdist[cidx, ridx] <- NA

  if (is.na(df_clean_loc$cluster[ridx]) & is.na(df_clean_loc$cluster[cidx]))
  {
    # New Cluster
    cluster_id <- max(df_clean_loc$cluster, na.rm=TRUE) + 1
    df_clean_loc$cluster[ridx] <- cluster_id
    df_clean_loc$cluster[cidx] <- cluster_id
  }
  else if (is.na(df_clean_loc$cluster[ridx]))
  {
    # Merge existing cluster(s)
    cluster_id <- df_clean_loc$cluster[cidx]
    df_clean_loc$cluster[ridx] <- cluster_id
  }
  else if (is.na(df_clean_loc$cluster[cidx]))
  {
    # Merge existing cluster(s)
    cluster_id <- df_clean_loc$cluster[ridx]
    df_clean_loc$cluster[cidx] <- cluster_id
  }
  else
  {
    row_cluster <- df_clean_loc$cluster[ridx]
    col_cluster <- df_clean_loc$cluster[cidx]
    cluster_id <- max(c(row_cluster, col_cluster), na.rm=TRUE)
    df_clean_loc$cluster[df_clean_loc$cluster == row_cluster] <- cluster_id
    df_clean_loc$cluster[df_clean_loc$cluster == col_cluster] <- cluster_id
  }

  cluster_size <- sum(df_clean_loc$office_count[df_clean_loc$cluster
                                                == cluster_id], na.rm=TRUE)
  if(cluster_size >= min_cluster_size)
  {
    cluster_rows <- which(df_clean_loc$cluster == cluster_id)
    mdist[cluster_rows,] <- NA
  }
}
```

The result was a total of 48 clusters; the smallest clusters contain 7 offices each and the largest cluster contains 38 offices. A map of the offices color coded by cluster is shown below.


40.0
37.5
35.0
32.5
lat
−122.5
−120.0
−117.5
−115.0
long

C Results for GCI

Results were similar for GCI calculations, as seen in the plots below. One notable difference is that a different (larger) ϵ would likely be chosen to obtain the desired level of accuracy; likely closer to 1.0 or 1.5.

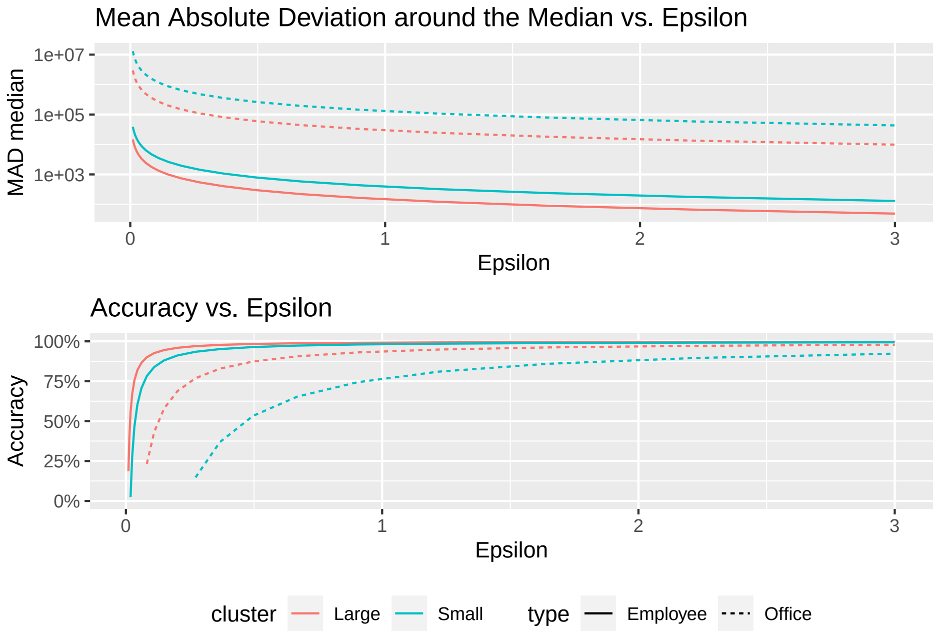


Figure 8: Plotting Mean Absolute Deviation around the Median as a function of Epsilon for GCI

Table 8: MAD Median and Accuracy at Epsilon=1.25 for GCI

MAD	accuracy	cluster	type
121.0950	0.9933264	Large	Employee
320.3386	0.9854114	Small	Employee
24316.5490	0.9485192	Large	Office
106445.6283	0.8099700	Small	Office

Table 9: MAD Median and Epsilon at Accuracy=0.9 for GCI

epsilon	MAD	cluster	type
0.0817756	1805.092	Large	Employee
2.2220123	58395.344	Small	Office

D References

Rubinstein, Benjamin I. P., and Francesco Aldà. 2017a. "Diffpriv: An R Package for Easy Differential Privacy." *Journal of Machine Learning Research* 18. https://cran.r-project.org/web/packages/diffpriv/vignettes/diffpriv.pdf.

———. 2017b. "Pain-Free Random Differential Privacy with Sensitivity Sampling." *CoRR* abs/1706.02562. http://arxiv.org/abs/1706.02562.