Measuring the effect of rain over Car Insurance Claims Frequency in Mexico City

Juan Jose Echevarria

Contents

Problem Statement

The challenge is to develop an analytics model that optimizes the number and geo-location of claim adjusters through the months of June to September, when it rains the most, in Mexico City.

The objective is to identify the effect of rain over the frequency of claims and which parts of town show an uptick on number of claims, so the insurance company can place more claims adjusters near the crashes to improve customer service.

Proposed Work

I propose developing an advanced analytics model that integrates historical claims and rain data to optimize the claims adjusting process during the months of July through September to accommodate the surge in car accidents throughout the city.

The model will consider a base case where no rain is present and the “outlier case” with the increase of frequency when it rains, while trying to predict the part of town where service will be needed based on historic spatial information and current fleet conditions.

Executive Summary

1. 78.18% of Mexico City's rainfall happens between July and October of each year. There is a clear difference in claims' frequency in rainy days.

2. The biggest difference in claims frequency is in the months of September and October.

3. During the COVID19 pandemic the was a general decrease of claims since people didn't leave their homes for an extended period. Frequency hasn't returned to pre-pandemic levels since most companies in Mexico implemented remote work.

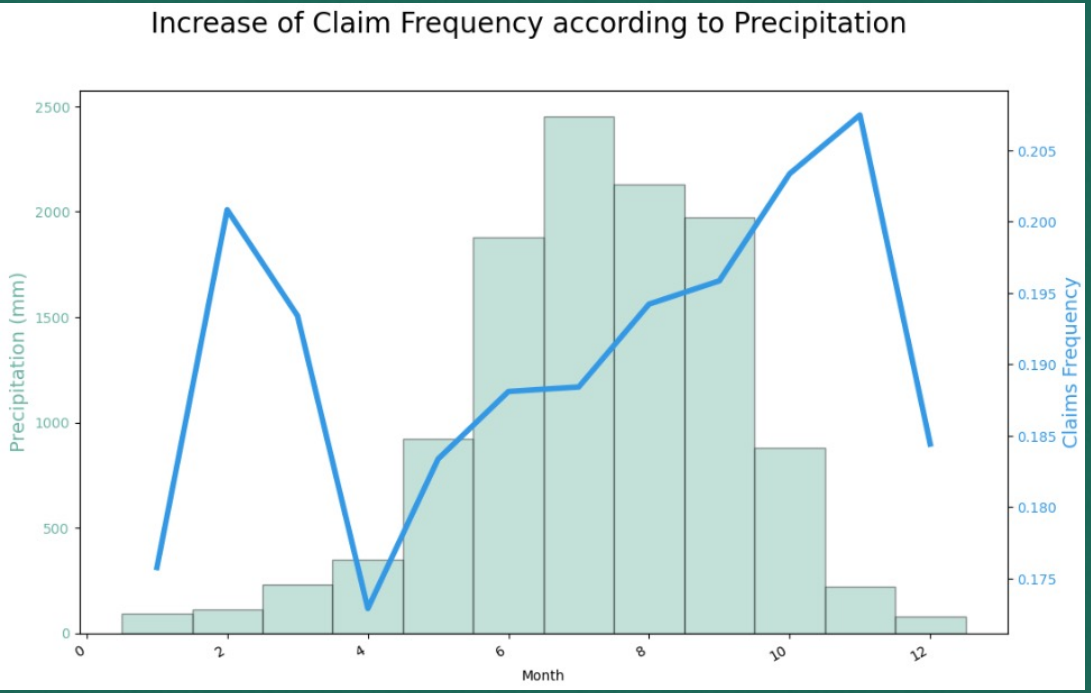
Executive Summary

- Mexico City is divided in 16 “Alcaldias”, like Counties in the U.S. This is also relevant because most of the offices are in certain areas of the city and residential areas are also in different counties.

- With the rainfall, claims and fleet information we have created a model to predict where in the city will most car accidents happen. All you need is the current fleet information and a trusty weather prediction of “Rain-No Rain”

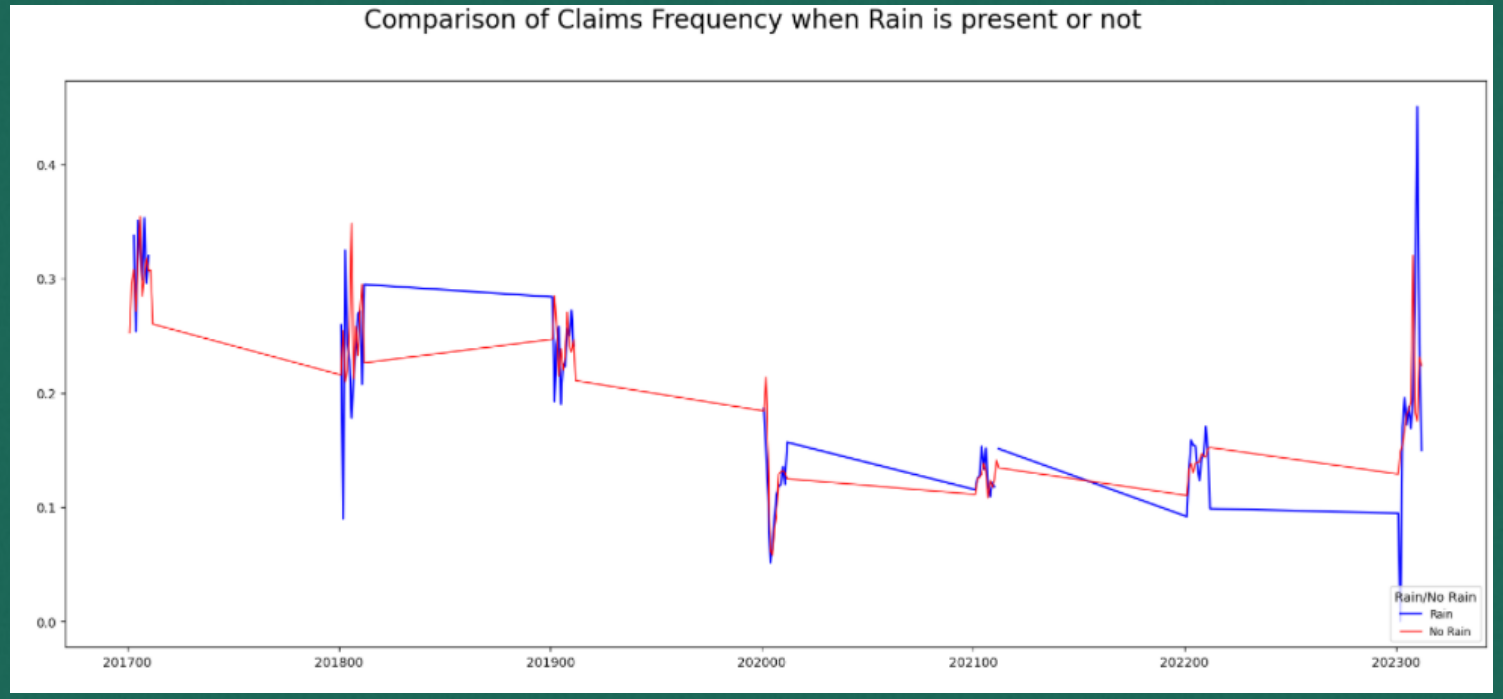
Results

78.18% of Mexico City's rainfall happens between July and October of each year. There is a clear difference in claims' frequency in rainy days. The biggest difference in claims frequency is in the months of September and October.



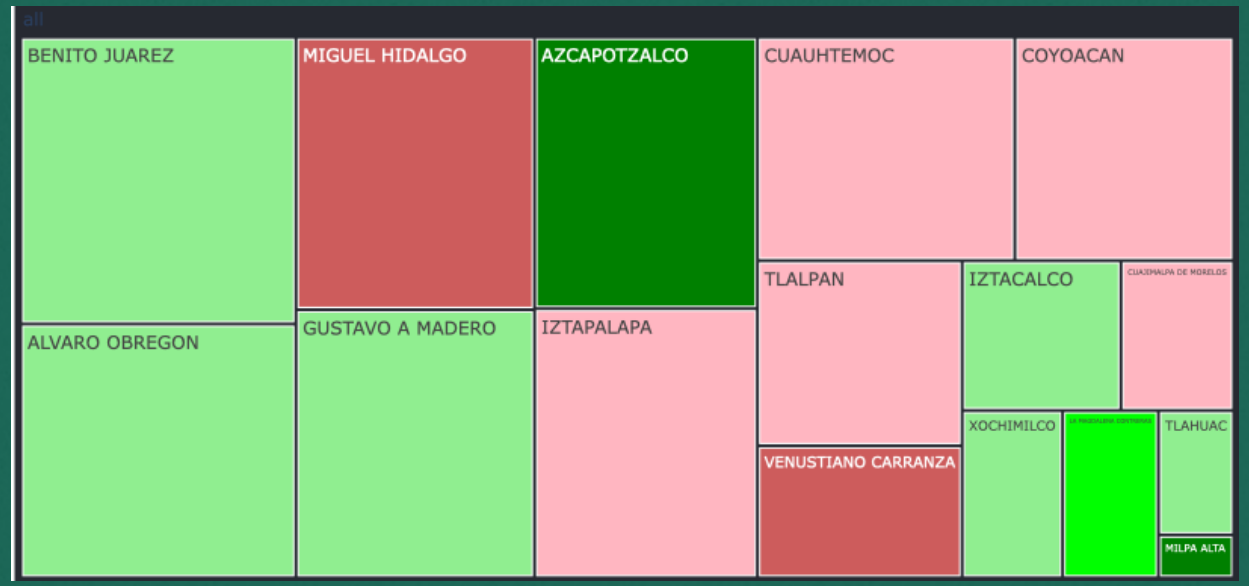
Results

During the COVID19 pandemic the was a general decrease of claims since people didn't leave their homes for an extended period. Frequency hasn't returned to pre-pandemic levels since most companies in Mexico implemented remote work.



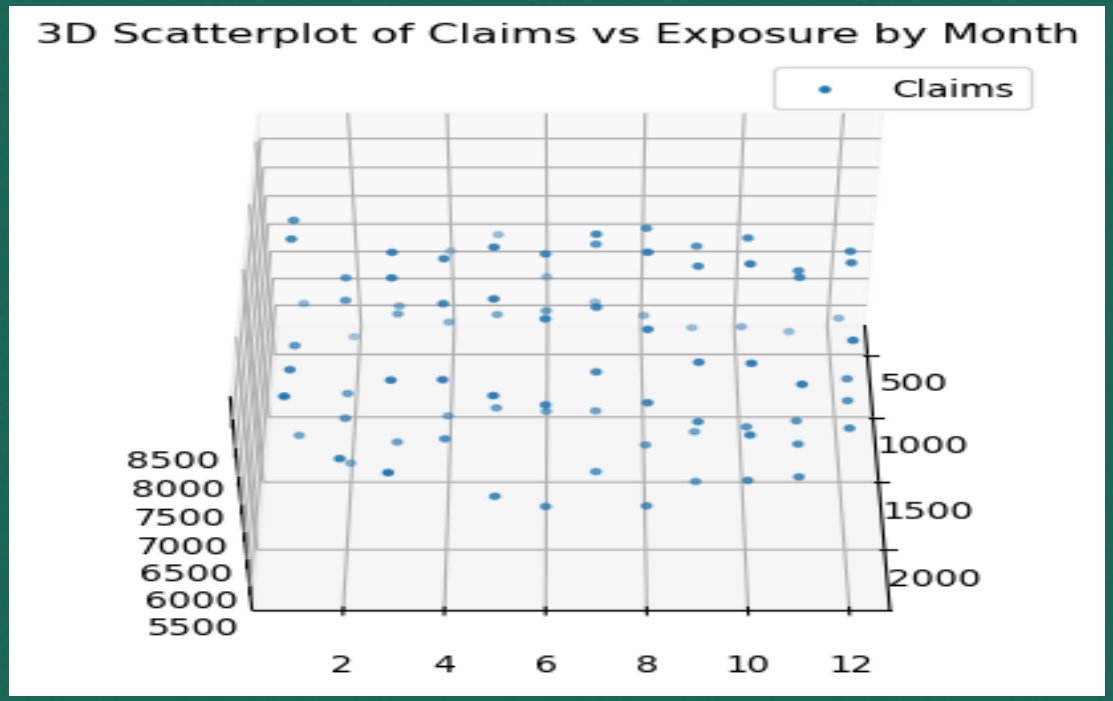
Results

Mexico City is divided in 16 “Alcaldias”, like Counties in the U.S. This is also relevant because most of the offices are in certain areas of the city and residential areas are also in different counties. This shows how claim frequency changes by each Alcaldia.



Results

While we were trying to create our models we tried graphing the data as a scatterplot, just claims versus exposure since that's how frequency is calculated. The problem is that even with the same exposure the claims where all over the place, we introduced a third dimension with the month of claims occurrence, and we got something a lot better:



Results

We created 7 models, they can be used by the Insurance Company depending on the data that they can get, especially rain forecast and fleet exposure.The first one is just using the claims and exposure data, there is no Rain information:

```
In [78]: reg = LinearRegression()
         reg.fit(X, Y)
         predictions1 = model.predict(X)
         print(
           'mean_squared_error : ', mean_squared_error(Y, predictions1))
         print(
           'mean_absolute_error : ', mean_absolute_error(Y, predictions1))
         print(reg.coef_)

         mean_squared_error :  165107.1309923975
         mean_absolute_error :  320.89826735960315
         [10.87344056 -0.07339983]
```

It has a mean error of claims of 320.89, that seem good but we felt it was high.

Results

The second model we tried some Machine Learning, separating into Train and Test subsets and tried the regression again:

```
In [70]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state = 28)

In [76]: model = LinearRegression()
         model.fit(X_train,Y_train)
         predictions = model.predict(X_test)
         print(
           'mean_squared_error : ', mean_squared_error(Y_test, predictions))
         print(
           'mean_absolute_error : ', mean_absolute_error(Y_test, predictions))
         print(model.coef_)

         mean_squared_error :  180646.4538672362
         mean_absolute_error :  350.26342079650954
         [19.0820722  -0.09164548]
```

The mean error went up to 350.26, which is worse from the previous model, it certainly has to do with the almost 2 years of abnormal frequency because of the pandemic, that's why we decide to continuing building from the first model.

Results

We then tried the model with our subset or just rainy days:

```
In [87]: model_rain = pd.read_csv('expuestos_siniestros_rain.csv')
         X_rain = model_rain[['mes', 'expuestos']].to_numpy()
         Y_rain = np.array(model_rain['siniestros'])
         reg_rain = LinearRegression()
         reg_rain.fit(X_rain, Y_rain)
         predictions_rain = reg_rain.predict(X_rain)
         print(
           'mean_squared_error : ', mean_squared_error(Y_rain, predictions_rain))
         print(
           'mean_absolute_error : ', mean_absolute_error(Y_rain, predictions_rain))
         print(reg_rain.coef_)

         mean_squared_error :  41115.129043096946
         mean_absolute_error :  148.21558548405412
         [6.42295346 0.17251117]
```

This came back with a standard error of 148.21, so we are on the right path.

Results

Doing the same with non-rainy days we get a model with a standard error of 286.18, higher than the rainy days model but makes sense since just one third of the days of the year:

In [88]:
```
model_dry = pd.read_csv('expuestos_siniestros_dry.csv')
X_dry = model_dry[['mes', 'expuestos']].to_numpy()
Y_dry = np.array(model_dry['siniestros'])
reg_dry = LinearRegression()
reg_dry.fit(X_dry, Y_dry)
predictions_dry = reg_dry.predict(X_dry)
print(
  'mean_squared_error : ', mean_squared_error(Y_dry, predictions_dry))
print(
  'mean_absolute_error : ', mean_absolute_error(Y_dry, predictions_dry))
print(reg_dry.coef_)
```

```
mean_squared_error :  114725.21200249023
mean_absolute_error :  286.1870061002958
[3.04412632 0.15478095]
```

Results

Finally, we tried the model introducing the "Alcaldia" feature, this should give us a better result. We tried with the full dataset:

```
In [90]: model_alcaldia_tot = pd.read_csv('alcaldia_expuestos_siniestros_total.csv')
         X_alcaldia_tot = model_alcaldia_tot[['mes', 'alcaldia','expuestos']].to_numpy()
         Y_alcaldia_tot = np.array(model_alcaldia_tot['siniestros'])
         reg_alcaldia_tot = LinearRegression()
         reg_alcaldia_tot.fit(X_alcaldia_tot, Y_alcaldia_tot)
         predictions_alcaldia_tot = reg_alcaldia_tot.predict(X_alcaldia_tot)
         print(
           'mean_squared_error : ', mean_squared_error(Y_alcaldia_tot, predictions_alcaldia_tot))
         print(
           'mean_absolute_error : ', mean_absolute_error(Y_alcaldia_tot, predictions_alcaldia_tot))
         print(reg_alcaldia_tot.coef_)

         mean_squared_error :  2002.1088427942814
         mean_absolute_error :  34.44245836857302
         [0.58573569 0.8870982  0.16144312]
```

The standard mean error improves to 34.44, since it's a monthly measure there is only a 1.13 claim difference by day and by Alcaldia since there are roughly 30.4 days a month.

Results

Doing the same with the “Rain” subset and the “Alcaldia” feature:

```
In [91]: model_alcaldia_rain = pd.read_csv('alcaldia_expuestos_siniestros_rain.csv')
         X_alcaldia_rain = model_alcaldia_rain[['mes', 'alcaldia','expuestos']].to_numpy()
         Y_alcaldia_rain = np.array(model_alcaldia_rain['siniestros'])
         reg_alcaldia_rain = LinearRegression()
         reg_alcaldia_rain.fit(X_alcaldia_rain, Y_alcaldia_rain)
         predictions_alcaldia_rain = reg_alcaldia_rain.predict(X_alcaldia_rain)
         print(
           'mean_squared_error : ', mean_squared_error(Y_alcaldia_rain, predictions_alcaldia_rain))
         print(
           'mean_absolute_error : ', mean_absolute_error(Y_alcaldia_rain, predictions_alcaldia_rain))
         print(reg_alcaldia_rain.coef_)

         mean_squared_error :  356.8908084125096
         mean_absolute_error :  12.076309177563537
         [0.40275291 0.34752271 0.17230134]
```

The standard error goes to 12.07 on average by month and “Alcaldia” since there are 9.47 rainy days a month we could say that the model has a 1.27 claim error for each Alcaldia.

Results

And lastly, the same exercise but with the “Non-Rain” or “Dry” subset:

```
In [92]: model_alcaldia_dry = pd.read_csv('alcaldia_expuestos_siniestros_dry.csv')
         X_alcaldia_dry = model_alcaldia_dry[['mes', 'alcaldia','expuestos']].to_numpy()
         Y_alcaldia_dry = np.array(model_alcaldia_dry['siniestros'])
         reg_alcaldia_dry = LinearRegression()
         reg_alcaldia_dry.fit(X_alcaldia_dry, Y_alcaldia_dry)
         predictions_alcaldia_dry = reg_alcaldia_dry.predict(X_alcaldia_dry)
         print(
           'mean_squared_error : ', mean_squared_error(Y_alcaldia_dry, predictions_alcaldia_dry))
         print(
           'mean_absolute_error : ', mean_absolute_error(Y_alcaldia_dry, predictions_alcaldia_dry))
         print(reg_alcaldia_dry.coef_)

         mean_squared_error :  1101.6864842643924
         mean_absolute_error :  23.93021933935371
         [0.26709369 0.78843495 0.16708333]
```

The standard error falls to 23.93 or 1.14 claims by Alcaldia worse, since there are 20.93 Non-Rain days in Mexico City by month in average.

Future Work

- Include in the model the day of the week, this also has a social element to it, it has been proven that claims occur different by day of the week.

- Check on the result of frequency of February and March, there's something there that we can't identify, at least in this exercise.

- Move preprocessing to Python (Jupyter Notebook)

Timeline

Problem Understanding: Understand the project requirements, goals, and constraints. Define the scope of the analytics model and establish key objectives.

Data Collection and Preparation: Gather historical claims, rain and fleet data. Clean, preprocess, and organize the data for analysis.

Model Development and Testing: Develop the analytics model that predicts claims surges based on historical data. Divide data into 2 segments, learn and test data. Test the model's accuracy and recommendations against historical events.

Timeline

Validation and Real-world Applicability: Validate the model's recommendations against actual claims in number and location.

Visualization and Reporting: Create clear and informative visualizations to present the model's insights and recommendations. Compile a comprehensive report outlining the methodology, findings, and implications of the analytics model.