

**Analyzing Information on Insurance Complaints**

Naquibuddin Shaik (11602741)

Sneha Reddy Nayini (11618990)

Snehitha Donthi (11627234)

Hemanth Narayanan Sathiya (11606107)

Sai Charan Siddagoni (11694074)

ADTA 5340 Section(s) 001 and IPAC 4340 Section(s) 001 (Spring 2024 1)

Professor: Dr. Zeynep Orhan

Spring 2024

## Index

<b>Chapter. No</b>	<b>Chapter Name</b>	<b>Page.No</b>
1	Abstract	3
2	Introduction	4
2.1	Research Question	5
3	Literature Review	6-7
4	Research Design and Methods	7- 37
4.1	Data Collection	8
4.2	Exploratory Data Analysis	9-10
4.3	Dependent and Independent Variables	11-27
4.4	Modelling and evaluation	28- 34
5	Results and Discussion	35-37
6	Conclusion and Recommendations	37
7	Reference	38

**Abstract:**

Our project completely focuses on the analysis of the details related to the insurance data by mainly keeping an objective of understanding and knowing the grievances all over an insurance period, finding few trends, assessing the client demands and also exploring the insurance companies and the attributes related to complaint relationship. We are using the data which was collected from the Texas Department of Insurance (TDI), we finally want to investigate on the insurance complaints all within the state Texas. The dataset which was provided has a very crucial information such as NAIC codes, product categories, company names, complaint dispositions, all the resolution dates, receipt dates, justifications and the complaint IDs. We are using the 'Complaint Reason' as primary interest which has the complete details like what are reasons behind the complaints received from client. The second objective for our team focuses on the resolution and the complaint statuses which offers a very productive insights into the process of complaint resolution.

All by leveraging this dataset, want to provide a lot of valuable of insights for the companies related to insurance and also for the regulatory bodies to make them achieve the enhancement in customer satisfaction and also for promoting the adherence to the regulatory standards. All through the thorough analysis of this dataset, we are aiming to uncover all the patterns, trends within the data and also the correlations within the complaints of insurance, thereby facilitating a very accurate decision-making processes. Finally, our team endeavours to contribute in improvising the overall quality of the services within the insurance companies by focusing on customers grievances very effectively.

**Business Understanding :**

Insurance is necessary for day-to-day operations since it protects both individuals and companies. Because they get complaints, insurance companies are able to look into situations and identify problems that the sector has to fix. The primary goal is to identify the dataset of complaints made against Texas insurance companies. Analysing complaint data can teach insurance firms a lot about areas for improvement, regulatory compliance, and customer satisfaction. The major objective of this inquiry is to acquire a thorough understanding of insurance complaints in Texas. Specifically, the study searches for any patterns or trends in the complaints filed over time regarding insurance. Examine the many requests that customers have made. Look into any connections that might exist between the volume or kind of complaints and insurance firms. Examine the complaints' current state and the average time it takes to resolve them. To increase client satisfaction and compliance, give pertinent data to insurance firms and regulatory agencies. Comprehensive information on every insurance complaint is provided via a range of the collection's criteria. The NAIC code, the company name, the kind of product, the disposition, the date of resolution, the complaint ID, the date of receipt, and the complaint explanation are a few of the essential components. The primary variable under investigation is 'Complaint Reason,' denoting the rationale provided by the customer for lodging the complaint. The analysis of all the different kinds and classifications of complaints made against insurance companies will be done using this variable. The secondary target variables "Disposition" and "Resolution Date" describe the status of the complaint and the timeline for its resolution. The public is not currently able to use data analysis as a method. The study will employ descriptive statistics, exploratory data analysis (EDA), and maybe advanced data analysis methods of study to uncover patterns, trends, and correlations in the dataset. The project documentation will have more information about the course and should be accessible soon. The limits of the data set include challenges

with data quality, underreported complaints, privacy concerns, and regulatory reporting requirements.

**Research Questions :**

Our group is primarily interested in two study questions: first, what kinds of complaints do Texans make to their insurance companies?

Establishing a connection between the time needed to settle complaints and the status of a resolution could shed insight on how effectively Texas' insurance companies communicate with one another.

We can gain a better understanding of customer concerns within the insurance industry by examining data on complaints filed with the Texas Department of Insurance. By starting with patterns and trends in complaints, regulatory agencies and insurance companies may ensure standard compliance, prevent problems, and improve client satisfaction.

The main objectives, data source, variables, target variables, and basic methodology for the analysis of Texas insurance complaints data are outlined in this proposal. Notifying the appropriate parties of the study's findings is necessary once the data is ready for analysis.

This proposal provides an overview of the objectives, variables, target variables, and data source for the investigations of Texas insurance complaints data. It notifies the appropriate parties of the findings and creates a plan of action for additional investigation.

## **Literature review**

The paper mainly focuses on the low-income and middle-income countries related insurance who adopted the community-based health insurance (CBHI) where the primary goal is to reduce both the inequity in the access of healthcare and with a very huge user fees. The CBHI management faced a lot challenges, which includes a very poor relationship with all their members. In the Democratic Republic of the Congo, MHOs managing the schemes of CBHI. The authors wants to know how these MHOs managed well with the member complaints. They used some sequential mixed-methods as their approach, they retrieved lot of valuable insights from almost four types of sources. The source includes 50 relevant documents and also by conducting almost 25 in-depth interviews with the managers of CBHI, also took in-depth interviews with the health facility managers and also conducted 1063 surveys, discussions with 15 groups (Ngo Bebe et al., 2023). These MHO members are from three diff types of MHOs from the capital, Kinshasa. Coming to results they confirmed that only 23% of the MHOs know the complaint related things and almost 84% of the members directly declared that there is usage of the measures at some point. It is proved that most of the members doesn't have any idea about the grievance redressal procedures. Finally the authors say that MHOs should surely improve all the communication with all their members to address the grievances. They should be adapting the well documented and communicated standard procedures in operating

The article “A quick guide to understanding Individual Health Insurance “ completely talks about the reason why it is very important to have good health insurance policy, especially for the youth like those who are single. As per the study they found there are two main types of health insurance: like that can be like individual and family types. Individual is going to

cover only one single person and while the family covers the whole family like group of people. The study explains the complete individual plans that can be like very good for the youth, single ones all because of many reasons like they may get rid of huge rates. They are saying that single people get a lot of benefits if they won't use for one year, they can get a bonus for next year in their policy. They can easily convert the policy to family type if they become to family. The insurance policy companies provide a lot of schemes and that may include like getting a one year plan may decrease the price. Overall they are saying that one should definitely take a policy even if they are young. The complaints may be raised as they face in their life.

The article " Assessment of the satisfaction with public health insurance programs by patients with chronic diseases in China: a structural equation modelling approach" mainly focuses on the landscape of the China's universal health insurance coverage, its primary focus is on the satisfaction of the patients, mainly who are suffering with the chronic diseases. The background study or the previous works have acknowledged the complete significance of the patients satisfaction all for the assessing the complete insurance program effectiveness, definitely there they found some gap in the understanding how the patients who are suffering from the chronic diseases are satisfied with the China's public health insurance policies and also whether this satisfaction varying all over the various types of insurance types. The study is seeking to address this particular gap by evaluating and examining the internal and external factors like that can be patients awareness of the insurance policies, fulfilling all the expectations and also present value of coverages, complaints (Geng et al., 2021). They used the structural equation modelling and conducted surveys in the tertiary hospitals. They finally aimed to study the relationship between these factors and also the satisfaction of the patients, shedding all the light on teh areas for the improvement of policy.

## **Research Design and Methods**

### **Data Collection:**

The TDI that is Texas Department of Insurance handles a lot complaints all against the people and also all the organizations which are licensed by TDI, they can be like companies, agents and also the adjusters. The complete dataset rows contain a each single person and their complaint named with organization. It finally means that we can observe a lot of complaint numbers are included multiple times in the whole dataset. The dataset is very huge one which almost contains 249k rows and 17 columns. Each row in this dataset is a combination of, one with a complaint number, which can be easily shared with the other persons or for any organization which is named in the complaint and also we find a Respondent ID which is a distinct value for each person or the organization named in the specific row. The columns included in the dataset are “complaint number, complaint filed against, complaint filed by, Reason complaint filed, confirm complaint, How resolved, Close date, Complaint type, Coverage level, others involved, Respondent ID, Respondent Role, Respondent type, Complaint type, Keywords and the final one is Received data”. The “complaint number” is a integer data type which contains the number which is assigned for specific complaint and unique. The “Complaint filed against” column contains the data about the names of the person or any organization by whom the complaint was made or filed. The column that is “complaint filed by” shows the complete details about the person who file dthe complaint and for example he can be like insured person, attorney or any relative. The column “ Reason complaint filed” shows all the reasons about the complaint made on and this is a text data type. The column “ Confirmed complaint”, this is the text data type and contains a “Yes” answer which



means the TDI is confirmed the licensed person or the company is in the error. “How Resolve” this is the column which explains the complete details how these complaints got resolved. “Received Date” is the date data type which has the values of complaint received date. Coming to “Closed Date”, it contains the date information when the complaint was closed. “Complaint type” has the details of the complaint variety, which type is that. The “Coverage type” column type has the details regarding the types like Health, accident or any other. The coverage level has the details regarding the coverage levels like is that a property type or causality type of coverage etc. “others involved”, has the details where for this who are all other people covered or involved. The respondent ID, it’s a unique value assigned to each person. For showing the role of the person against the complaint is “Respondent Role”. “Respondent type” can shows us the details regarding the complaint on they filed like on the person or organization. The “complaint type” can be like category of the complaint came from like from a person or org. Finally column “Keywords” shows the complete info about the category or the complaint to help in sorting the common issues and this is a text type of data type. We have lot of missing values and if the data works well we will building the model, if wanted some changes to the data we may choose some random values.

### **Data Description :**

RangeIndex: 249169 entries, 0 to 249168

Data columns (total 17 columns):

#	Column	Non-Null Count	Dtype
0	Complaint number	249169 non-null	int64
1	Complaint filed against	249169 non-null	object

2	Complaint filed by	249169 non-null	object
3	Reason complaint filed	249163 non-null	object
4	Confirmed complaint	249169 non-null	object
5	How resolved	248141 non-null	object
6	Received date	249169 non-null	object
7	Closed date	249169 non-null	object
8	Complaint type	249168 non-null	object
9	Coverage type	249169 non-null	object
10	Coverage level	249169 non-null	object
11	Others involved	220978 non-null	object
12	Respondent ID	249169 non-null	int64
13	Respondent Role	249167 non-null	object
14	Respondent type	249169 non-null	object
15	Complainant type	249169 non-null	object
16	Keywords	199629 non-null	object

### **Exploratory Data Analysis:**

Using the below code, we have tried to know the missing values count.

```
▶ Null_data=Input_Data.isnull().sum()
Null_data
```

```
⦿ Complaint number          0
  Complaint filed against    0
  Complaint filed by         0
  Reason complaint filed     6
  Confirmed complaint        0
  How resolved              1028
  Received date              0
  Closed date                0
  Complaint type             1
  Coverage type              0
  Coverage level             0
  Others involved            28191
  Respondent ID              0
  Respondent Role            2
  Respondent type            0
  Complainant type           0
  Keywords                   49540
  dtype: int64
```

```
Input_Data = Input_Data.dropna(thresh=Input_Data.shape[1]-1)
```

Using the above code, started replacing few missing values in the dataset. After that we got the below results, few missing values got replaced.

```
Null_data=Input_Data.isnull().sum()
Null_data
```

```
Complaint number          0
Complaint filed against    0
Complaint filed by         0
Reason complaint filed     1
Confirmed complaint        0
How resolved              469
Received date              0
Closed date                0
```

```

Complaint type      0
Coverage type      0
Coverage level      0
Others involved     17858
Respondent ID       0
Respondent Role     0
Respondent type     0
Complainant type    0
Keywords            38973
dtype: int64

```

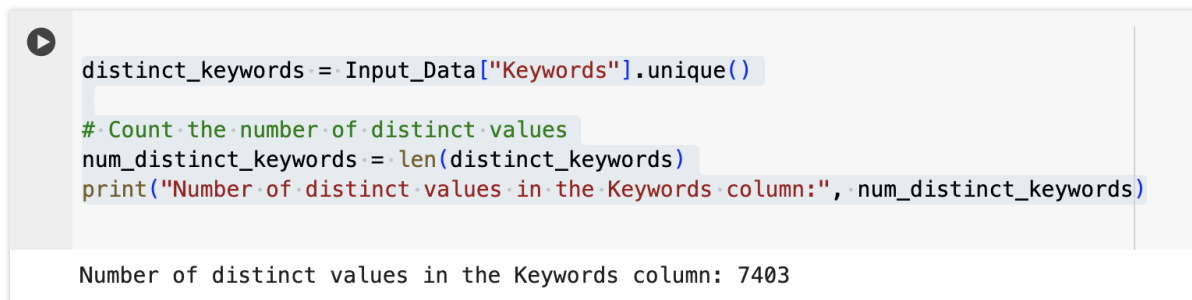
Now using the below code, we started knowing the unique values in the column “Keywords”.

```

distinct_keywords = Input_Data["Keywords"].unique()

# Count the number of distinct values
num_distinct_keywords = len(distinct_keywords)
print("Number of distinct values in the Keywords column:", num_distinct_keywords)

```



```

distinct_keywords = Input_Data["Keywords"].unique()

# Count the number of distinct values
num_distinct_keywords = len(distinct_keywords)
print("Number of distinct values in the Keywords column:", num_distinct_keywords)

```

Number of distinct values in the Keywords column: 7403

The total number of unique values in the column “Keywords” is almost 7403.

Now finding all the unique values in the columns “How resolved”, “others involved” and “keywords”.

```

# Print distinct values and their counts in the 'How resolved' column
resolved_unique_values = Input_Data['How resolved'].unique()
print("Distinct values in 'How resolved' column:")
print(resolved_unique_values)
print("Number of unique values:", len(resolved_unique_values))

# Print distinct values and their counts in the 'Others Involved' column
others_involved_unique_values = Input_Data['Others involved'].unique()
print("\nDistinct values in 'Others Involved' column:")
print(others_involved_unique_values)
print("Number of unique values:", len(others_involved_unique_values))

# Print distinct values and their counts in the 'Keywords' column
keywords_unique_values = Input_Data['Keywords'].unique()
print("\nDistinct values in 'Keywords' column:")
print(keywords_unique_values)
print("Number of unique values:", len(keywords_unique_values))

```

And we found the count in the output.

The number of distinct values in the column “How resolved” is almost 3898. Coming to distinct values count in the column “others involved” is 6841. The count of the unique or distinct values in the column “Keywords” is 7403.

```

# Assuming Input_Data is your DataFrame

# Create a pivot table for the 'How resolved' column
how_resolved_pivot = pd.pivot_table(Input_Data, index='How resolved', aggfunc='size')

# Create a pivot table for the 'Others Involved' column
others_involved_pivot = pd.pivot_table(Input_Data, index='Others involved', aggfunc='size')

# Create a pivot table for the 'Keywords' column
keywords_pivot = pd.pivot_table(Input_Data, index='Keywords', aggfunc='size')

# Print the pivot tables
print("Pivot table for 'How resolved' column:")
print(how_resolved_pivot)
print("\nPivot table for 'Others Involved' column:")
print(others_involved_pivot)
print("\nPivot table for 'Keywords' column:")
print(keywords_pivot)

```

```

▶ Pivot table for 'How resolved' column:
How resolved
Additional Monies Received 4175
Additional Monies Received; Additional Payment Expected 218
Additional Monies Received; Additional Payment Expected; Appraisal Process Invoked 1
Additional Monies Received; Additional Payment Expected; Claim Settled 7
Additional Monies Received; Additional Payment Expected; Company Position Upheld 6
...
Taxes Paid 2
Taxes Paid; Additional Monies Received; Information Furnished 1
Taxes Paid; Claim Settled 1
Taxes Paid; Information Furnished 1
Taxes Paid; Information Furnished; Additional Monies Received 1
Length: 3897, dtype: int64

Pivot table for 'Others Involved' column:
Others involved
Acting Person 6
Acting Person; Adjuster; Insured 1
Acting Person; Associated Subject Agency 1
Acting Person; Associated Subject Agency; Insured 1
Acting Person; Associated Subject Company 2
..
Third Party Admin-Non Licensed 54
URA Contact 1
Utilization Review Agent 7
Workers Comp Network 42
Workers Comp Network Contact 1
Length: 6840, dtype: int64

Pivot table for 'Keywords' column:
Keywords
2012 NORTH TEXAS TORNADOES 5
2012 NORTH TEXAS TORNADOES; ACV; CLAIM EVALUATION; TOTAL LOSS 1
2012 NORTH TEXAS TORNADOES; ADJUSTER'S HANDLING; APPRAISAL; DAMAGE DISPUTE; ROOF 2
2012 NORTH TEXAS TORNADOES; ADJUSTER'S HANDLING; CLAIM EVALUATION; COVERAGE DISPUTE; ROOF 1
2012 NORTH TEXAS TORNADOES; ADJUSTER'S HANDLING; CLAIM EVALUATION; DAMAGE DISPUTE 1
...
UNDERWRITING CRITERIA; WATER DAMAGE 24
VIATICAL SETTLEMENT 56
WATER DAMAGE 254
WIND 2
West Explosion 2013 1
Length: 7402, dtype: int64

```

The code is creating three pivot tables to mainly organize and also to count the data from various other columns in a data frame.

The first pivot table is created for 'How resolved' column, it shows all the count of each resolution type is getting appeared in the whole dataset.

The second pivot table is created with the column 'other Involved' column, which mainly counts the no. of times each type of parties involved.

The third pivot is created with the column 'Keywords' it is mainly counting the occurrences

```
[ ] # Impute missing values in 'How resolved' column with mode
    mode_reason_complaint_filed = Input_Data['How resolved'].mode()[0]
    Input_Data['How resolved'] = Input_Data['How resolved'].fillna(mode_reason_complaint_filed)

    # Impute missing values in 'Others involved' column with mode
    mode_complaint_type = Input_Data['Others involved'].mode()[0]
    Input_Data['Others involved'] = Input_Data['Others involved'].fillna(mode_complaint_type)

    # Impute missing values in 'Keywords' column with mode
    mode_respondent_role = Input_Data['Keywords'].mode()[0]
    Input_Data['Keywords'] = Input_Data['Keywords'].fillna(mode_respondent_role)

    # Impute missing values in 'Reason complaint filed' column with mode
    mode_respondent_role = Input_Data['Reason complaint filed'].mode()[0]
    Input_Data['Reason complaint filed'] = Input_Data['Reason complaint filed'].fillna(mode_respondent_role)
```

Now I'm replacing the missing values like imputing the values with mode.

```
Complaint number      0
Complaint filed against  0
Complaint filed by    0
Reason complaint filed  0
Confirmed complaint    0
How resolved          0
Received date         0
Closed date           0
Complaint type        0
Coverage type         0
Coverage level        0
Others involved       0
Respondent ID         0
Respondent Role       0
Respondent type       0
Complainant type      0
Keywords              0
dtype: int64
```

after replacing, we can see that there are zero missing values for each column.

```
[ ] from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
# Selecting only numerical columns
numerical_columns = Input_Data.select_dtypes(include=['int64', 'float64']).columns

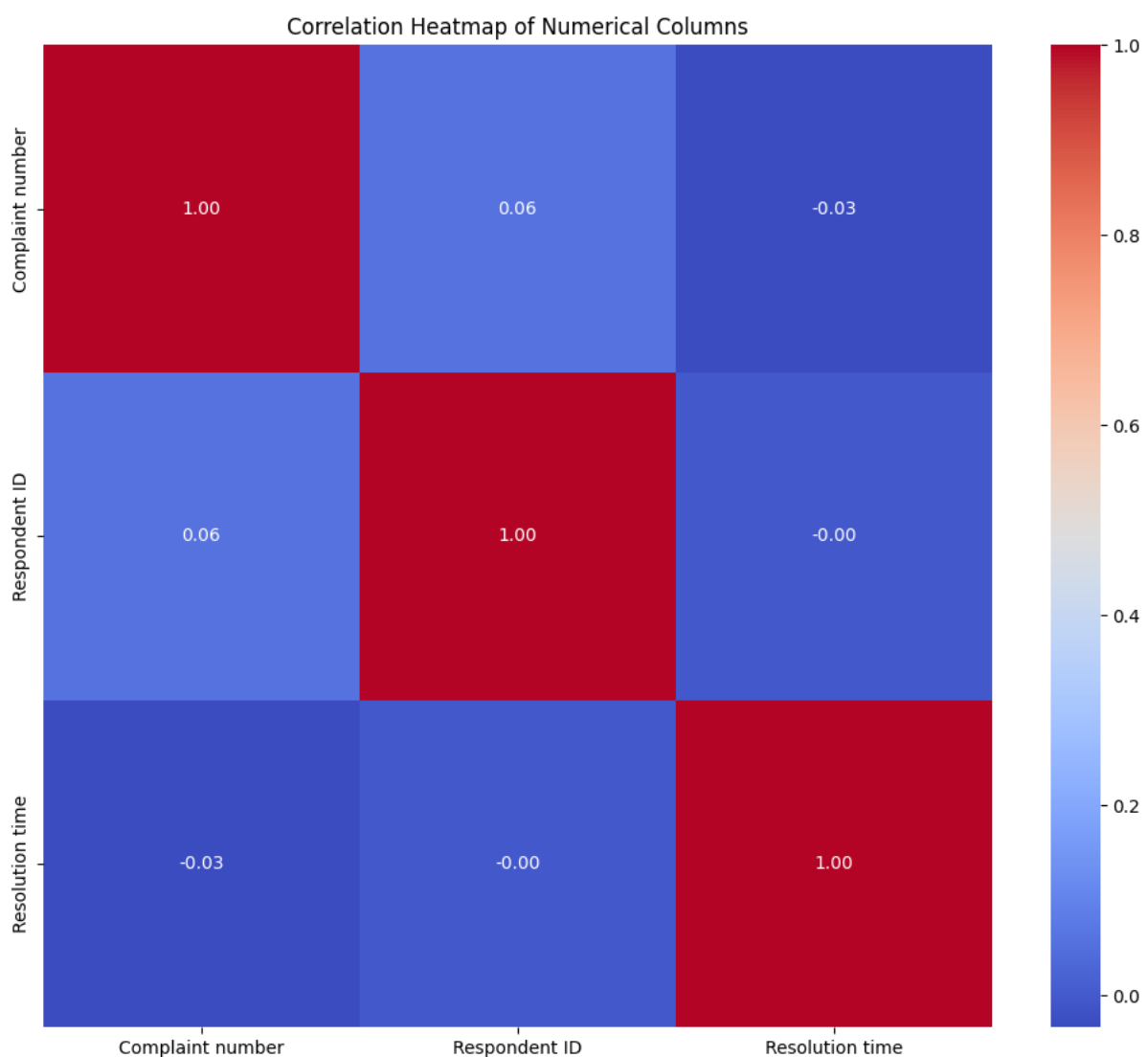
# Plotting the scatter matrix for numerical columns
scatter_matrix(Input_Data[numerical_columns], figsize=(15, 12))
plt.show()

[ ] import seaborn as sns

# Calculating the correlation matrix
correlation_matrix = Input_Data[numerical_columns].corr()

# Plotting the heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Numerical Columns')
plt.show()
```

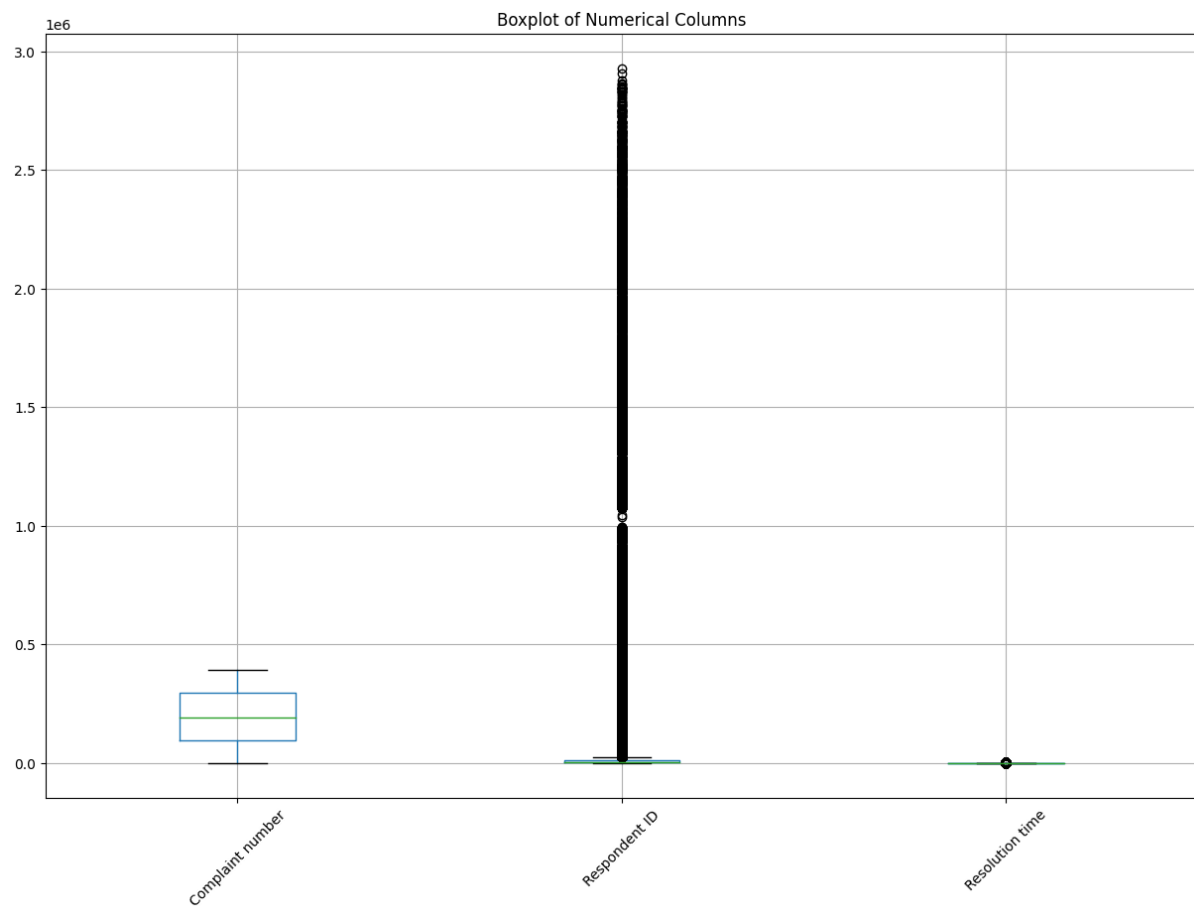
Now using above code we have drawn the scatter matrix. The above code is used to draw the correlation Heatmap of the Numerical columns.



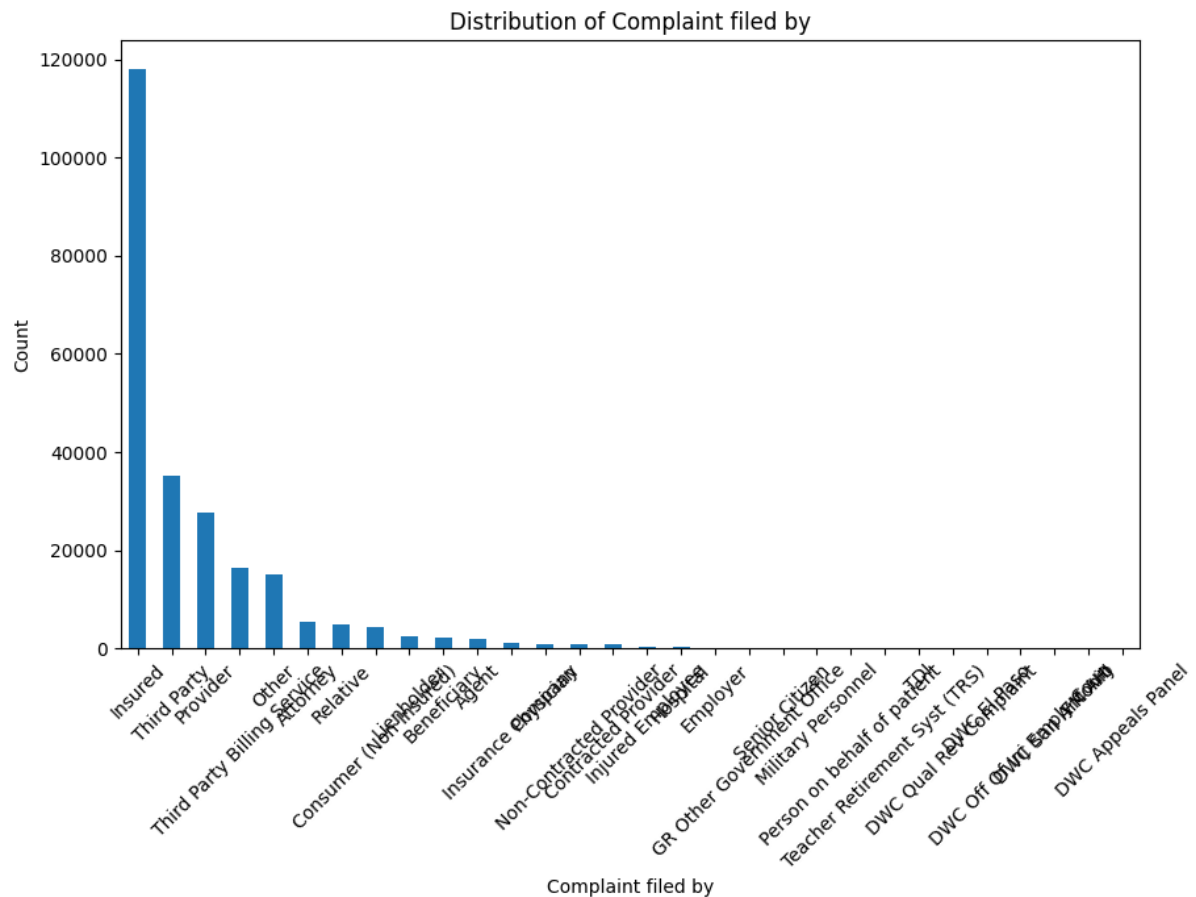


After watching the Correlation map, we can say that there is a high correlation between the columns Respondent ID and Complaint number that is of 0.06, even the columns like Complaint number and Respondent ID has the highest correlation that is of 0.06. The correlation is for the columns 'Resolution time' and 'complaint number' is -0.03 and even the column that is 'complaint number' and 'resolution time'. The columns 'Respondent ID' and 'Complaint number' are dependent on each other very highly.

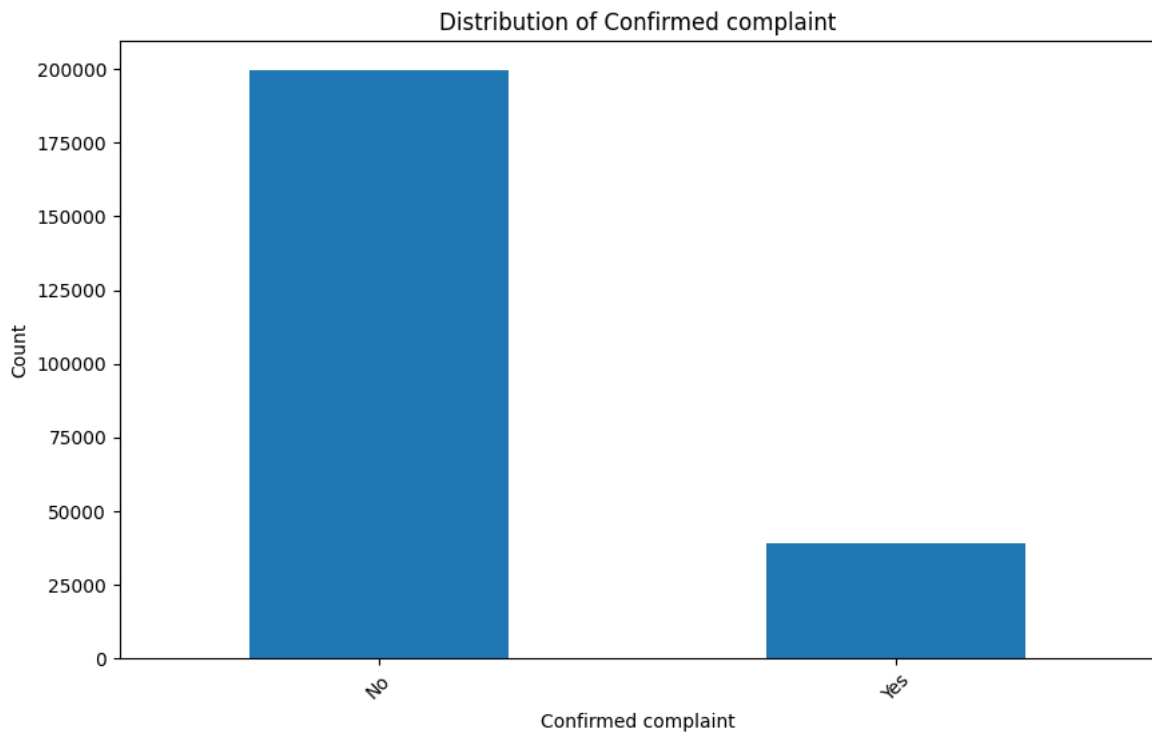
The below image is the Boxplot of the Numerical columns.



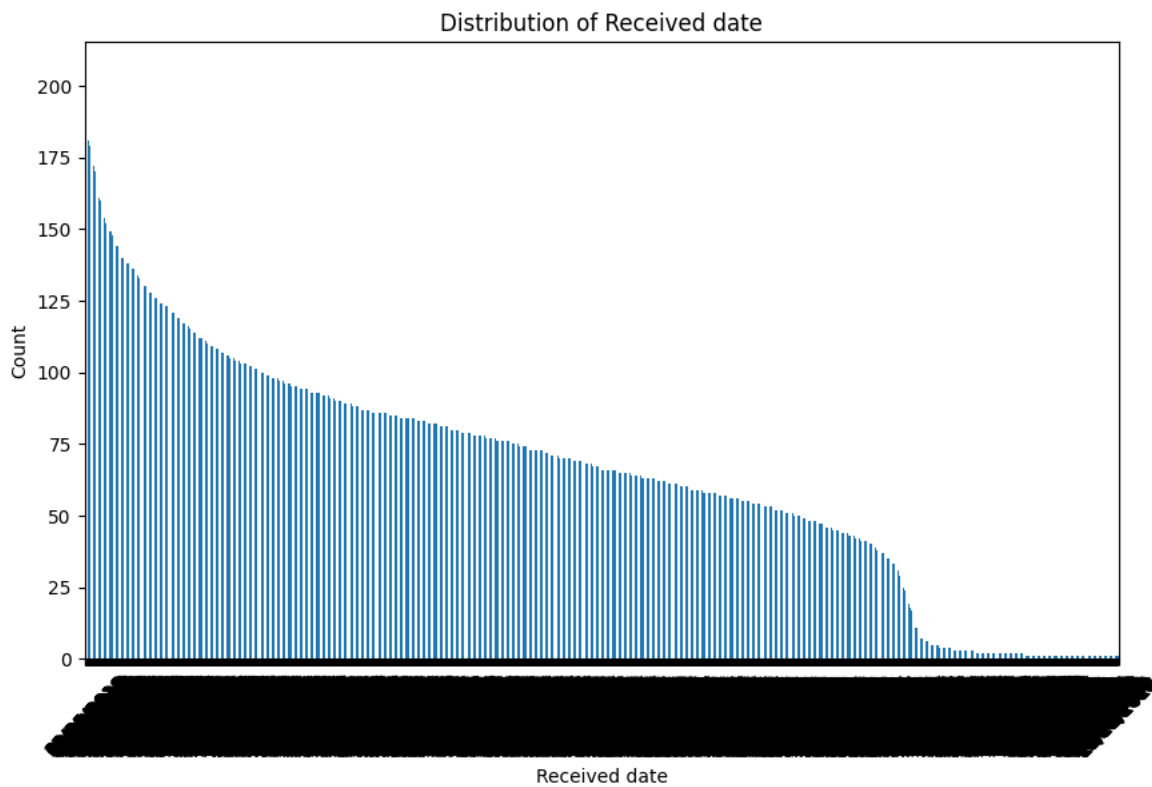
The Above boxplot displays the outliers present in the numerical columns, Complaint number, Respondent ID and Resolution time. Here we see Respondent ID is having outliers obviously as its ID and other variables has no outliers.

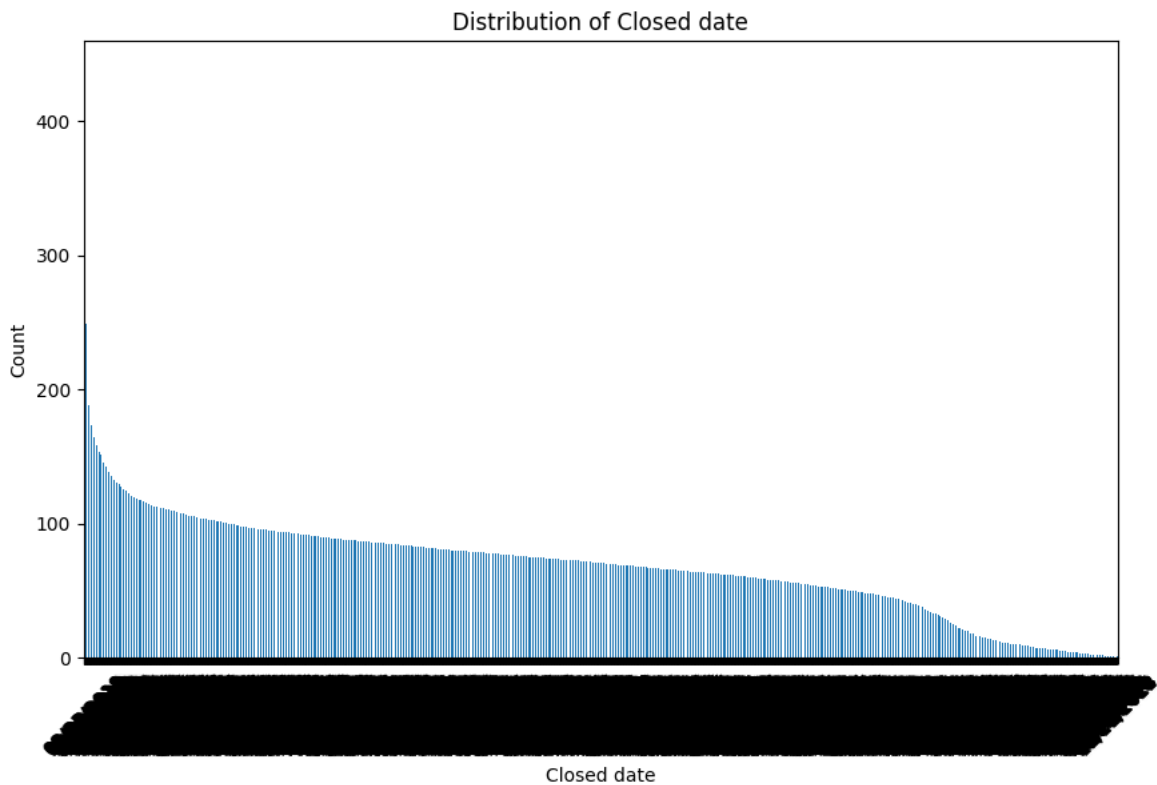


The above chart is a bar graph, which finally shows there is huge distribution for most of the columns. Like the Insured type of people had filed a lot of complaints as per the data. Next most of the complaints are filed by Third Party and then Provider comes next. Almost 1,10,000 of complaints are from Insured parties only.

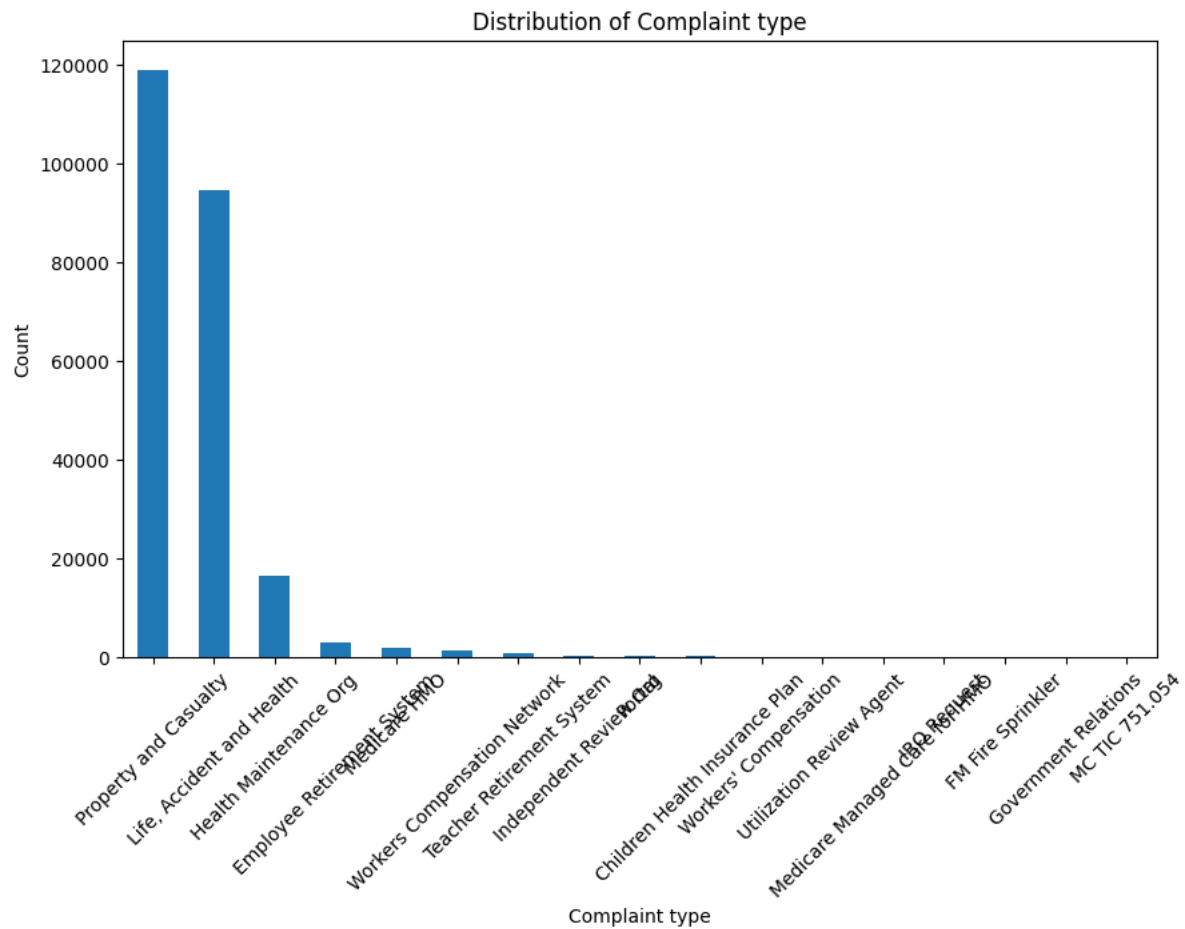


The most of the confirmed complaints in the data is not confirmed one. We see a highest percentage of not confirmed cases as per the distribution made in this above bar graph.

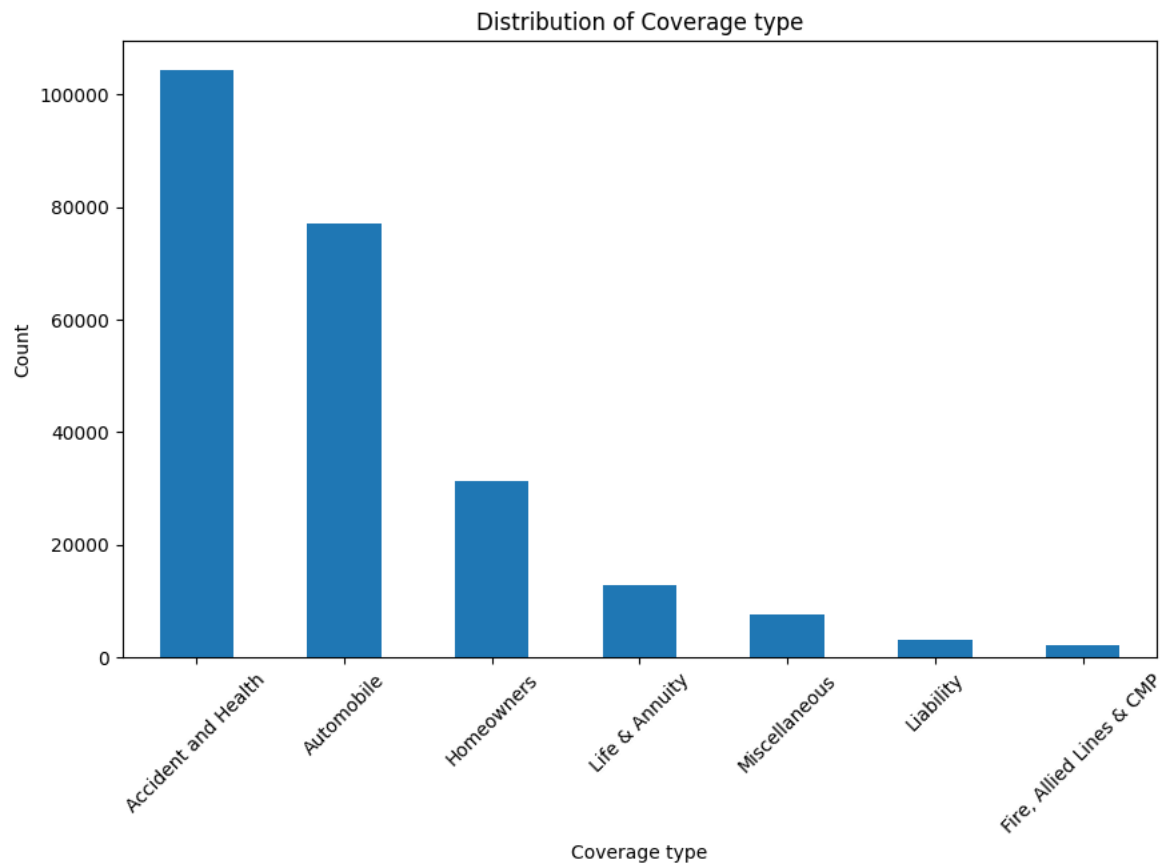




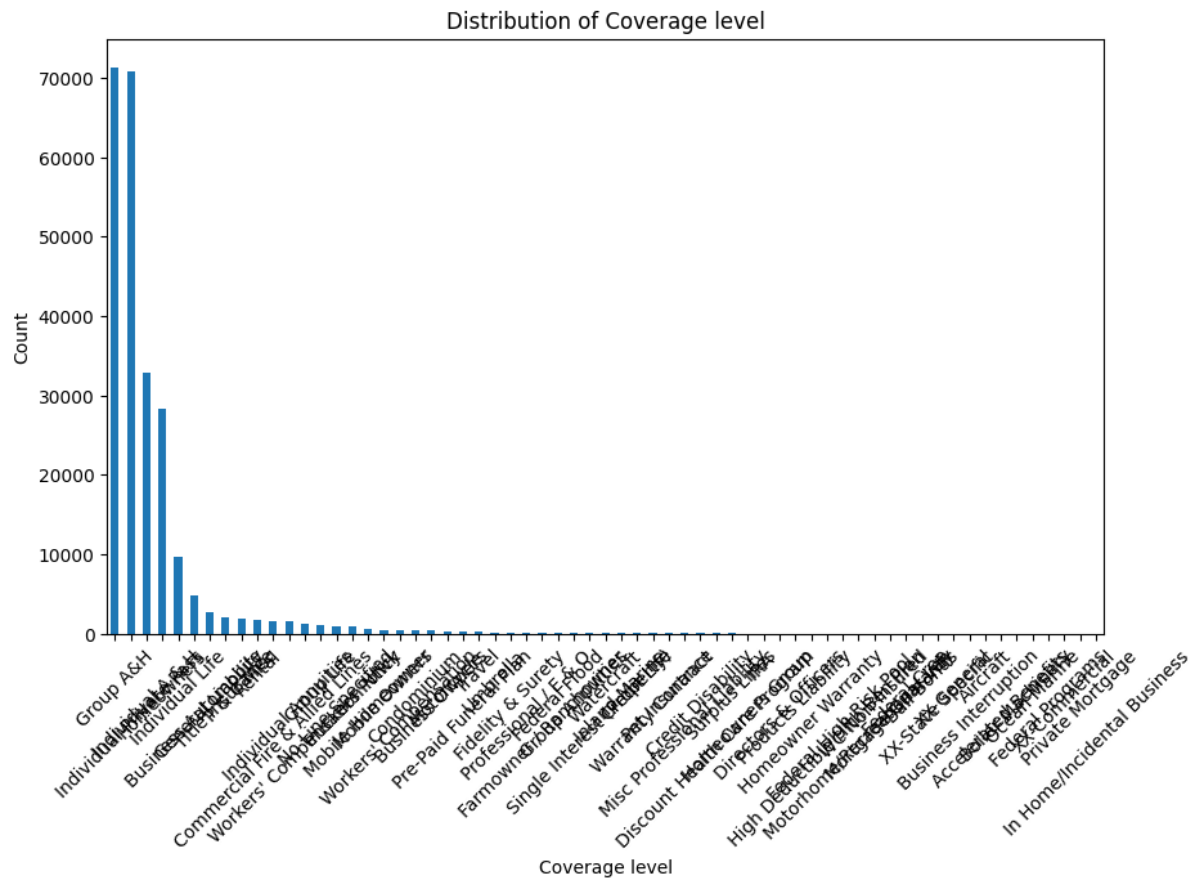
Here we are trying to find the distribution of the complaint received date and closed date, as it's important to know how the received and closed date are related as it would strengthen our knowledge on the depth of problem type and how long are they taking to be resolved. As we see, both the graphs are similar and left-skewed, which states much time has not been taken to solve each problem.



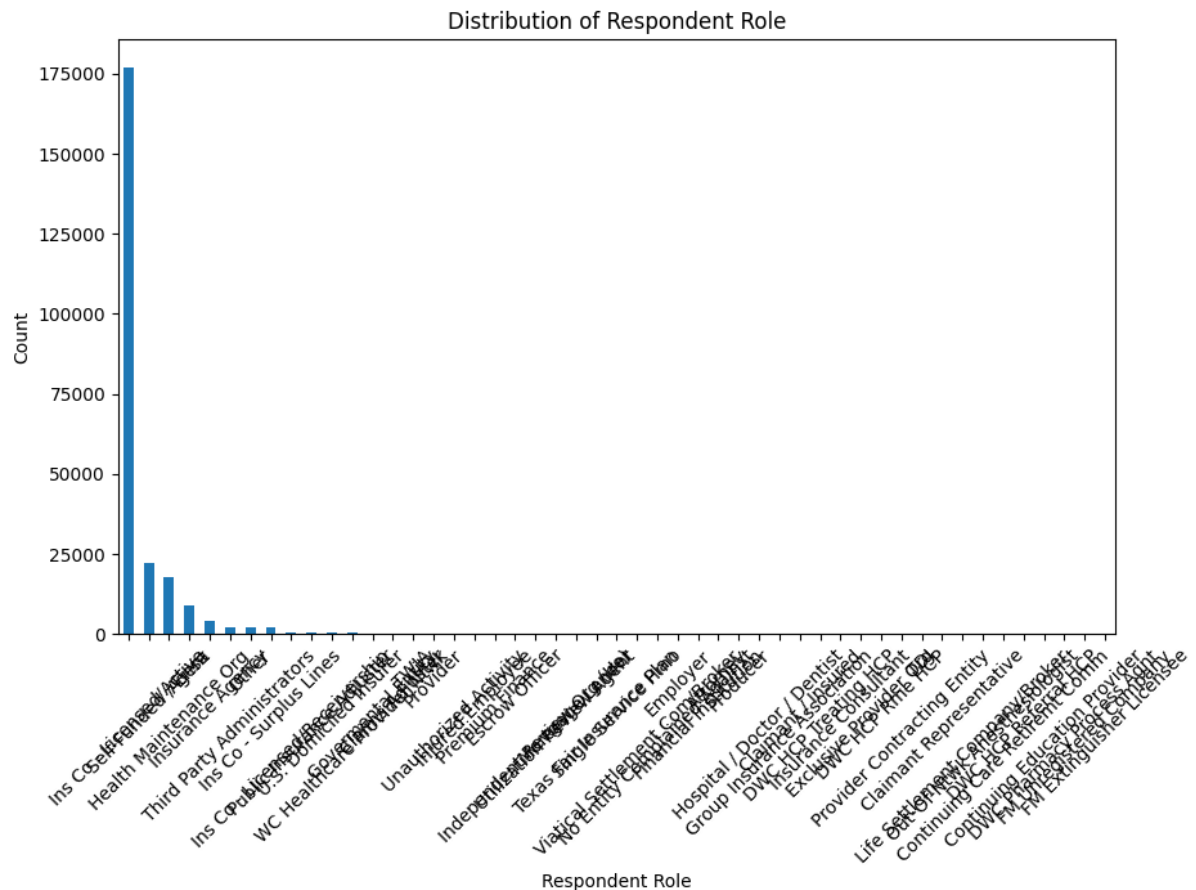
As per the bar graph here we can see that the highest distribution of the complaint types is by property and causality related only. Then next comes the Life, accident and Health related complaints.



The coverage type is very high for the Accident and health and then comes the automobile. The least coverage type is from the Fire, Allied Lines and the CMP. The highest one has the value of almost 1,20,000. The x axis has the Coverage type and the Y axis has the count.

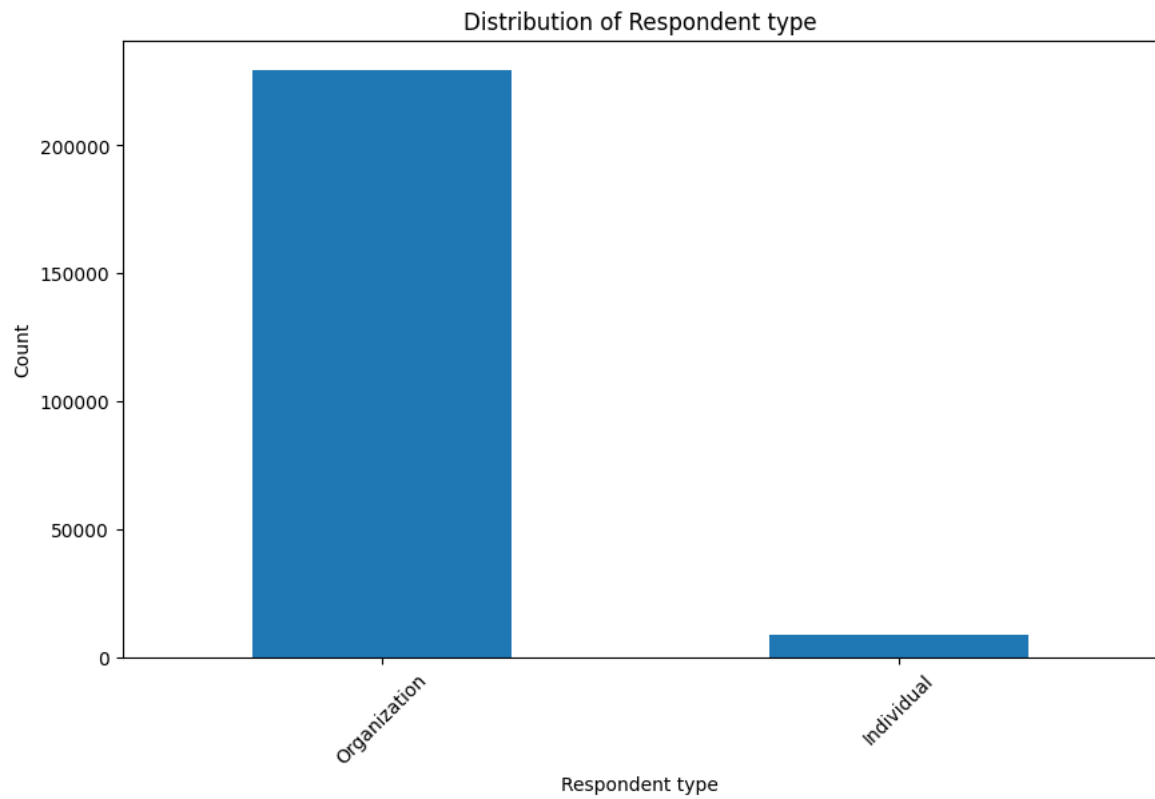


Now coming to the Coverage type bars, we found that the highest reading is for the Group A&H. and we can see that almost with the same level or with almost same count we have the individual type of coverage level distribution.

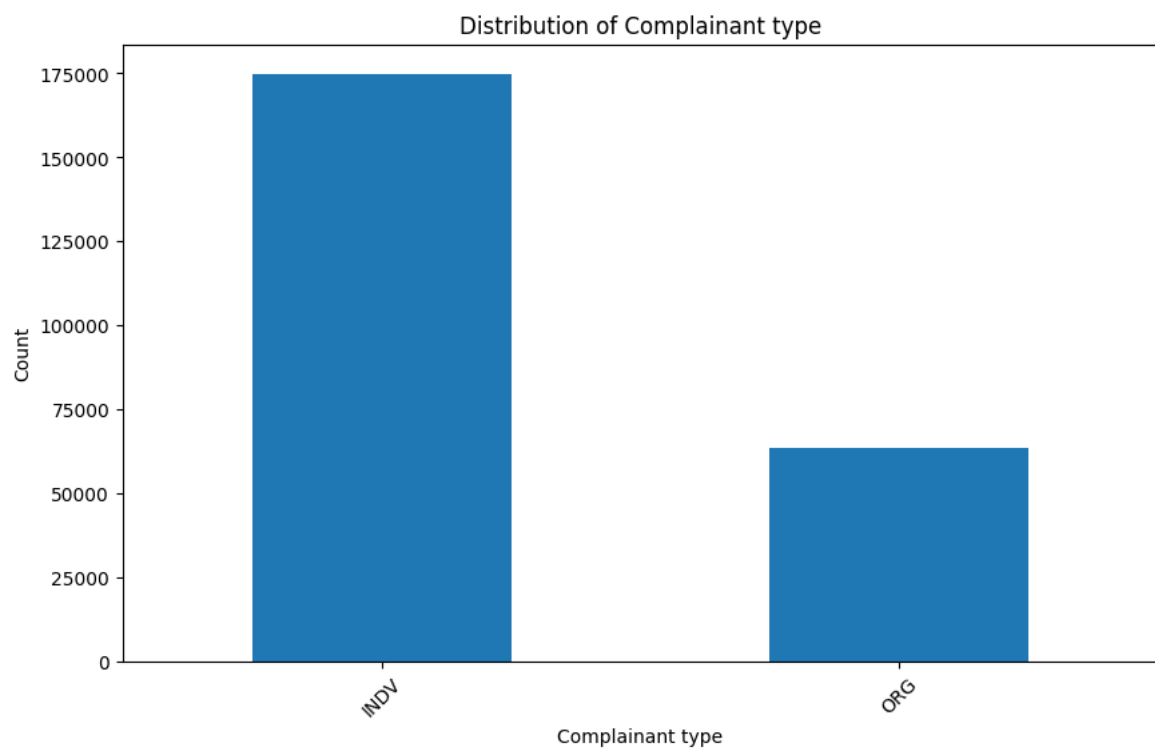


Now we can see the Distribution of Respondent Role, here we only few roles with the highest count and rest of them completely disappeared in the graph, that means they are not distributed as much as these columns in the first or starting of the graph like Health Maintenance Org, Insurance Agency etc.

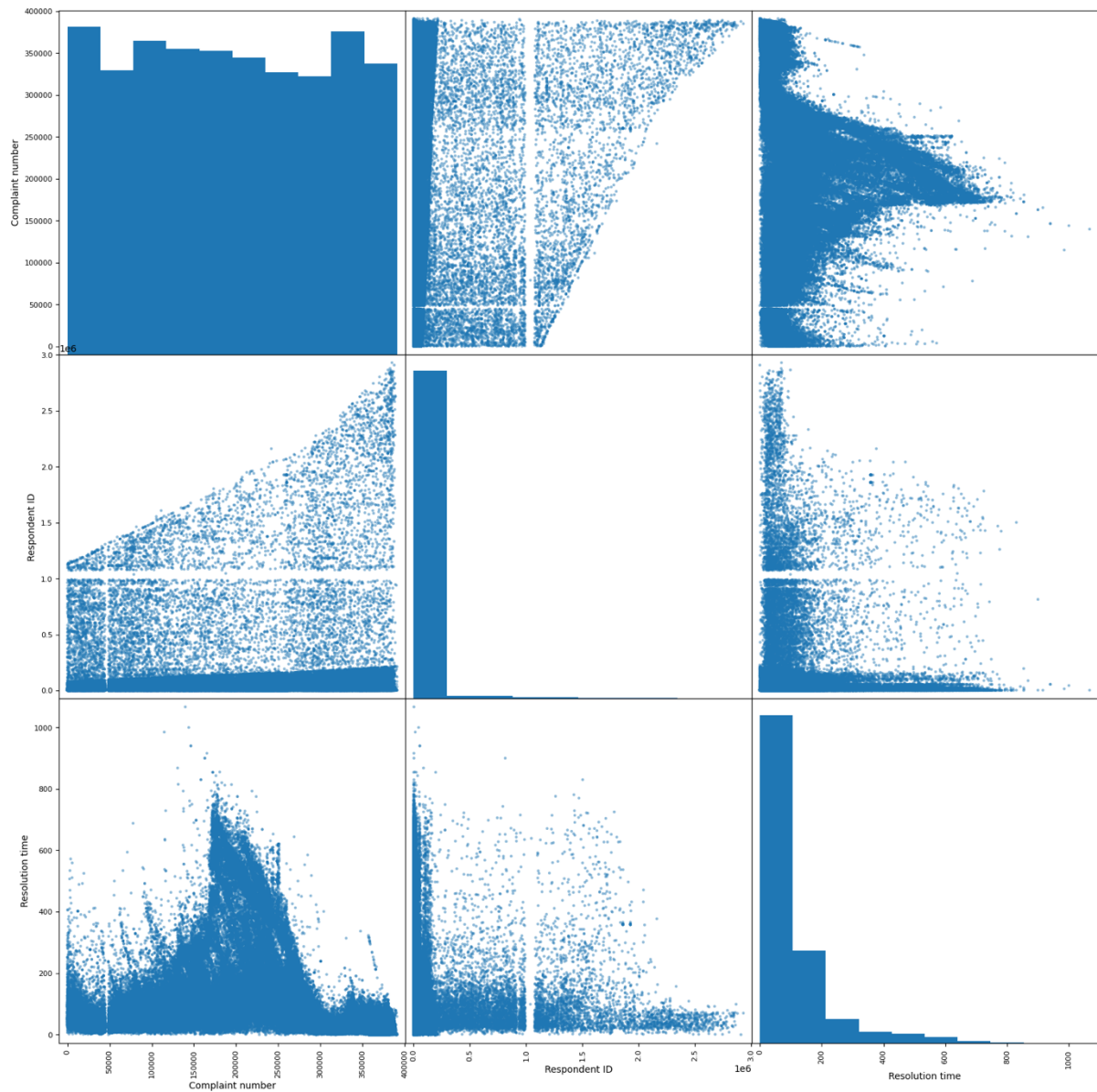




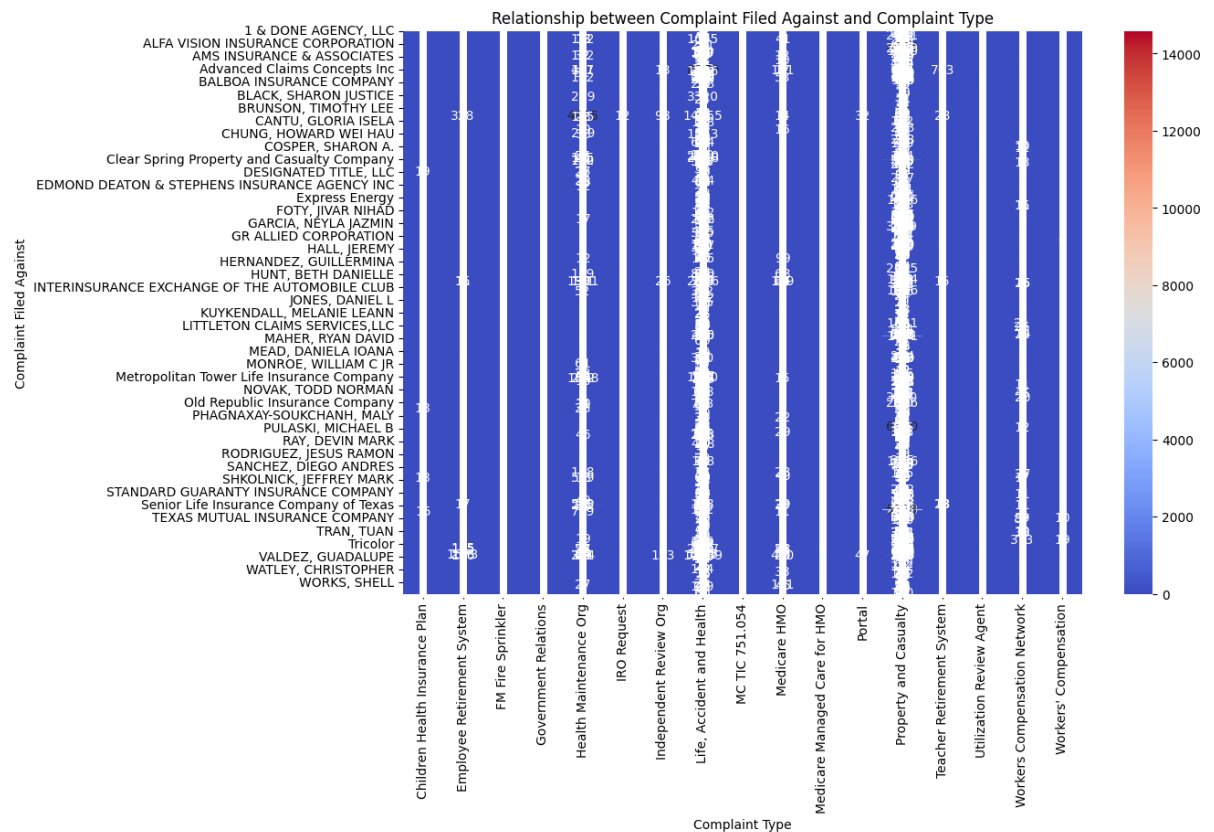
Now coming to the Distribution of Respondent type, we can see more count for the Organization type of Respondents. The count of Individual is very low.



The Distribution of the Complainant type, here complaints type are from mainly from the individual side. The organization has the least count like there complaints are very low.



Here we are trying to concentrate on each numerical variable is related to the other numerical variables. If we observe, for Respondent Id and complaint number the data points greatly increase from left to right which states the complaint numbers are increasing gradually. Where as complaint number and resolution time the distribution is normal as we see. The resolution time and Respondent Id the points are randomly scattered.



This is the Heat Map, with the Distribution of 'Complaints Filed against', which shows the details of the companies against whom the complaints are filed and The complaint type. This plot is created to understand if there exists any pattern in the company name and type of complaints. Which didn't show much relation as for the figure.

```
from datetime import datetime

Input_Data["Received date"] = pd.to_datetime(Input_Data["Received date"])
Input_Data["Closed date"] = pd.to_datetime(Input_Data["Closed date"])

# Calculate frequency of complaints filed against each company
complaint_frequency = Input_Data["Complaint filed against"].value_counts()

# Calculate average resolution time for each company
Input_Data["Resolution time"] = (Input_Data["Closed date"] - Input_Data["Received date"]).dt.days
average_resolution_time = Input_Data.groupby("Complaint filed against")["Resolution time"].mean()

# Print the results
print("\nFrequency of complaints filed against each company:")
print(complaint_frequency)
print("\nAverage resolution time for each company:")
print(average_resolution_time)
```

Using the code we converting the "Received date" and "Closed date" columns into the datetime format for the best calculations. this can calculate the frequency of the complaints all filed against by each company. then it is computing the avg resolution time for each company in the data. now this is printing the results showing all the complaints filed often against each company and how these complaints are taking time to get resolved.

We got the results as below:

Frequency of complaints filed against each company:

Complaint filed against

Blue Cross and Blue Shield of Texas, A Division of Health Care Service Corporation  
20056

UnitedHealthcare Insurance Company 15245

Aetna Life Insurance Company 9246

PROGRESSIVE COUNTY MUTUAL INSURANCE COMPANY

6667

State Farm Mutual Automobile Insurance Company 5400

...

Texas Farmers Insurance Company, Inc. 1

SCHKADE-HILL, KIMBERLEY ANN 1

WEHRLY, JOSEPH WILLIAM 1

THOMAS, FRED ALAN 1

WELLINGTON INSURANCE SERVICES, INC. 1

Name: count, Length: 15626, dtype: int64

Average resolution time for each company:

Complaint filed against

1 & DONE AGENCY, LLC 130.000000

1 STOP FINANCIAL SERVICE CENTERS OF AMERICA LLC 44.750000

1-Stop Auto Insurance 85.000000

1908 Advisors, LLC 29.000000

1Dental 288.000000

...

prime seamless gutters & roofing, llc 76.000000

sneed, Taylor brooke 118.000000

usautoinsurance.com 68.000000

webTPA Employer Services, LLC 134.783333

williams, alfred 22.000000

Name: Resolution time, Length: 15626, dtype: float64

### Modelling and evaluation:

In this case we have first tried training the model with 2 lakh datapoints however it took larger times and due to more noise didn't show any expected results. This might be due to the more number of categorical variables, which takes longer time for encoding those many records. Hence we have implemented the sampling techniques to select around 1 lakh samples randomly using sample function to get better and efficient results.

Now for the best modelling and predicting in a accurate way, we are choosing almost 1,00,000 random records from the data. For that we used the below code.

```
[ ] # Randomly sample 100,000 records from Input_Data
    sampled_df = Input_Data.sample(n=10000, random_state=42)

    # Save the sampled DataFrame to a new variable or file if needed
```

The model we have tried to predict the 'Reason complaint filed' is “RandomForestClassifier”. Checking the accuracy for this model. The RandomForestClassifier is a model which is like a team making some decision in a expertise way like woking all together then trying to solve a problem. Each expert or in the group who is working on this problem is (tree) looks at a very random part of the data and try to make a decision on it. Then all the decisions are together combined to make some final prediction like making it accurately in a proper way. This is a powerful model its because it combines all diverse opinions and reduces the chances of making a casual mistake in the process. it is mainly used for tasks where there is a need of prediction that can be like finding a email which is either genuine one or spam. finding whether the disease is severe or normal in stage. RandomForestClassifier is the most popular one all because of its robust nature and this handles a large datasets in a well manner. This is very less prone to overfitting. The model can memorises the complete data instead learning all form it.

This is the code used to design the model and then finding the accuracy.

```

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder
import pandas as pd

# Copy the Input_Data dataframe to avoid modifying the original data
data = sampled_df.copy()

# Encode categorical columns using LabelEncoder
label_encoders = {}
for column in data.select_dtypes(include='object').columns:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])

# Split the data into features (X) and target variable (y)
X = data.drop(columns=['Reason complaint filed'])
y = data['Reason complaint filed']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Instantiate the RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier
classifier.fit(X_train, y_train)

# Predict on the test set
y_pred = classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

```

Accuracy: 0.267

The code is training the program to predict the reasons all behind the customer complaints using the random data. It started with preparing the data, encoded the categorical info into numbers for making the model understand. Then the splitting is done like one is for training the model and one is for testing the complete accuracy of the results or the model. Now building the Random Forest Classifier, teaching the whole thing how to predict the results using the training data. Now the program finally predicts the complaint related reasons using the test data. evaluation of the model performance is done by comparing the predictions with the actual complaint reasons. This model is providing an accuracy about its predictions and that is almost 0.267. The model accuracy is very low.

```

from sklearn.metrics import accuracy_score, precision_score, f1_score, classification_report

precision = precision_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

# Convert scores to percentages
accuracy_percent = accuracy * 100
precision_percent = precision * 100
f1_percent = f1 * 100

print("Accuracy:", accuracy_percent, "%")
print("Precision:", precision_percent, "%")
print("F1 Score:", f1_percent, "%")

# Additional: Classification report
print("\nClassification Report:\n", classification_report(y_test, y_pred))

```

Accuracy: 26.700000000000003 %  
 Precision: 20.39458694960614 %  
 F1 Score: 22.447379137794456 %

Converting the results to percentages, then we can see that the results are:

Accuracy: 26.700000000000003 %

Precision: 20.39458694960614 %

F1 Score: 22.447379137794456 %

So we can surely say that the model is not so good for predictions.

Now we are finding the highest and most correlated columns. Using that columns, we start training the model and start predicting. Using this method can reduce the columns count and also can remove the unwanted columns. The most correlated columns can be used for designing our model. Below is the code used for finding the most correlated columns for the target variable that is 'Reason complaint filed'. Printing the correlation\_data.

```

import pandas as pd
import numpy as np
from scipy.stats import chi2_contingency
# from scipy.stats import cramers_v

# Load your dataset
# data = pd.read_csv("your_dataset.csv")

# Select categorical columns and target variable
categorical_columns = ['Complaint filed against', 'Complaint filed by', 'How resolved', 'Complaint type', 'Coverage type', 'Coverage level', 'Others involved', 'Respondent Role', 'Respondent type',

# Remove 'Reason complaint filed' from the list of categorical columns
categorical_columns.remove('Reason complaint filed')

# Create a DataFrame to store correlation values
correlation_data = pd.DataFrame(index=categorical_columns, columns=['Chi-Square P-value', "Cramer's V"])

# Calculate correlation for each categorical variable
for column in categorical_columns:
    # Create a contingency table
    contingency_table = pd.crosstab(Input_Data[column], Input_Data['Reason complaint filed'])

    # Chi-square test for independence
    chi2, p, _, _ = chi2_contingency(contingency_table)

    # Cramer's V statistic
    num_categories = contingency_table.shape[0]
    cramer_v = np.sqrt(chi2 / (Input_Data.shape[0] * (min(contingency_table.shape) - 1)))

    # Store the results in the correlation DataFrame
    correlation_data.loc[column, 'Chi-Square P-value'] = p
    correlation_data.loc[column, "Cramer's V"] = cramer_v

# Print correlation data
print(correlation_data)

```

The results shows that most correlated data with the target variable is:

Chi-Square	P-value	Cramer's V
Complaint filed against	0.0	0.297292
Complaint filed by	0.0	0.495497
How resolved	0.0	0.308368
Complaint type	0.0	0.441097
Coverage type	0.0	0.446813
Coverage level	0.0	0.280307
Others involved	0.0	0.246934
Respondent Role	0.0	0.318062
Respondent type	0.0	0.688168
Complainant type	0.0	0.488514
Keywords	0.0	0.251631

After checking out the complete results we found these are the variables mostly correlated with the target variable and they are "Complaint filed by", "Complaint type", "Coverage type", "Complainant type", "Reason complaint filed".

Using these variables, we started building the model that is neural network model. And trained training on these highly correlated columns.



```

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.utils import to_categorical

# Load the dataset

# Select only the effective columns
selected_columns = ["Complaint filed by", "Complaint type", "Coverage type", "Complainant type", "Reason complaint filed"]
data = Input_Data[selected_columns]

# Drop rows with any missing values
data.dropna(inplace=True)

# Encode categorical variables
label_encoders = {}
for column in ["Complaint filed by", "Complaint type", "Coverage type", "Complainant type", "Reason complaint filed"]:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])

# Split the data into features (X) and target variable (y)
X = data.drop(columns=["Reason complaint filed"])
y = data["Reason complaint filed"]

# Convert target variable to one-hot encoded vectors
y = to_categorical(y)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Build the neural network model
model = Sequential()
model.add(Dense(64, input_dim=X_train.shape[1], activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(32, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(y_train.shape[1], activation='softmax'))

# Compile the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, epochs=3, batch_size=64, validation_split=0.2)

# Evaluate the model on the test set
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {accuracy * 100:.2f}%")

```

As we selected only few columns to predict the ‘Reason complaint filed’, now started building the neural network model. The code is building and training a neural network model for the prediction of reasons behind the complaints from the customers. It started loading the data and then used the selected columns which are relevant to the target variable. encoded the data for converting the categorical variables and now splitting the data. here in this code we converted the target variable into one-hot encoded vectors for the reason of classification. Training and testing data got split up. As the neural network model is constructed in a way like having multiple layers, all by including the dropout layers for preventing the overfitting. After the compilation of the model, now it got trained with the few epoch. The model performance is evaluated at the last and finding the accuracy of the model. The results show Test Accuracy: 16.04%.

now converting them to percentages, we have

Accuracy: 15.5 %

Precision: 4.8467122283655835 %

F1 Score: 6.675425368887638 %

As the accuracy is very low for the model, we can say that the predicted values are not so perfect after comparing them with the actual reasons for the complaints by the customers.

The neural network didn't suit for the predictions.

Now choosing an another model that is

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import CategoricalNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.preprocessing import OneHotEncoder

# Load the dataset
# data = pd.read_csv("your_dataset.csv")

# Select the input features and target variable
X = sampled_df.drop(columns=["Complaint filed against"])
y = sampled_df["Complaint filed against"]

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# One-hot encode categorical variables with handling unknown categories
encoder = OneHotEncoder(handle_unknown='ignore')
X_train_encoded = encoder.fit_transform(X_train)
X_test_encoded = encoder.transform(X_test)

# Convert sparse matrix to dense array
X_train_encoded_dense = X_train_encoded.toarray()
X_test_encoded_dense = X_test_encoded.toarray()

# Train the Naive Bayes classifier
classifier = CategoricalNB()
classifier.fit(X_train_encoded_dense, y_train)

# Predict on the test set
y_pred = classifier.predict(X_test_encoded_dense)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

The results for this model is:

Accuracy: 0.1165

Precision: 0.06417685443220268

Recall: 0.1165

F1 Score: 0.047139643599793044

```

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.preprocessing import LabelEncoder
import pandas as pd

# Copy the Input_Data dataframe to avoid modifying the original data
data = sampled_df.copy()

# Encode categorical columns using LabelEncoder
label_encoders = {}
for column in data.select_dtypes(include='object').columns:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])

# Split the data into features (X) and target variable (y)
X = data.drop(columns=['Complaint filed against'])
y = data['Complaint filed against']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Instantiate the RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier
classifier.fit(X_train, y_train)

# Predict on the test set
y_pred = classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("\nClassification Report:\n", classification_report(y_test, y_pred))

```

Accuracy: 0.6515

accuracy		0.65	2000
macro avg	0.14	0.16	0.14 2000
weighted avg	0.59	0.65	0.61 2000

after converting them to percentage wise, we can see the results as

Accuracy: 65.14999999999999 %  
Precision: 58.617715919854554 %  
F1 Score: 60.5210195565312 %

The code encodes the categorical columns using the LabelEncoder for converting the categorical data to the numerical data type of format, by that the model can understand that. The data is split into X and Y, where X has the features and the Y has the target variable that is " Complaint filed against" . Now the dataset got split with training set and testing set. Then the "RandomForestClassifier" model is built and trained with some training data. The model is evaluated by checking out the accuracy rate, F1 score etc. By comparing all the results of each model we built, we found the highest accuracy, precision and F1 score for the model

that is “Random Forest Classifier”. The model is predicting the target variable with more accuracy and precision. This shows that the results are almost accurate when compared with the actual reasons for the complaints from the customers.

### **Results and Discussion**

The accuracy of the model is almost 65%, that means mostly it is predicting the party all against which the complaints are being filed about. However, the precision that is measuring how many positive cases have been predicted correctly, that is around 58% and coming to F1 score, which has the purpose of balancing the precision and the recall has only 60%. These are the scores for each and finally suggest that model is somewhat predicting the target variable with good accuracy but we feel it could be improved in the terms of finding or accurately identifying all the parties involved in the complaints. As we observe the results we have known that the results were not efficient and requires more study to improve the performance of model. The main causes for these outputs and loss in the performance metrics is found to be due to the dataset. As we know the model performance mainly depends on the quality of the dataset we are using. Here in case of our dataset, though we have larger number of records, and not so bad number columns, Most of the columns are with non-collinearity, which shows very lesser dependence of one variable on the other variable, which makes the model hard to find the deeper connections between the variables. One other disadvantage is the presence of more categorical values, as we see we have limited number of numerical variables this is the other reason.



### **Conclusion and Recommendations**

For the future studies, we feel to explore more on the additional features which may be so better for capturing all the nuances of the complaints and the involved entities. This may involve like collecting more data or may be changing the machine learning model which can finally handle the complexity in predicting the target variable more accurately. Data pre-processing is also very important, we feel more efforts should be kept on that which makes the model more accurate by giving it right data. We feel regular evaluation is also good recommendation for us and refining the model using techniques like cross-validation or any hyperparameter tuning. All these measures may be very beneficial in ensuring all its effectiveness for accurately predicting the complaints outcomes.

## References

- A quick guide to understanding Individual Health Insurance - Comprehensive coverage at a minimal premium. (2023, May 24). *BioSpectrum*, <https://libproxy.library.unt.edu/login?url=https://www.proquest.com/magazines/quick-guide-understanding-individual-health/docview/2818878548/se-2>
- Geng, J., Chen, X., Shi, J., Bao, H., Chen, Q., & Yu, H. (2021, October 19). Assessment of the satisfaction with public health insurance programs by patients with chronic diseases in China: a structural equation modeling approach. *BMC Public Health*, 21(1). <https://doi.org/10.1186/s12889-021-11947-7>
- Ngo Bebe, D., Kwilu, F. N., Mavila, A., Mafuta, E. M., Mangalu, J. M., Jessani, N. S., & Criel, B. (2023, September). Making health insurance responsive to citizens: the management of members' complaints by mutual health organisations in Kinshasa, Democratic Republic of Congo. *BMJ Global Health*, 7(Suppl 6), e011438. <https://doi.org/10.1136/bmjgh-2022-011438>