UNIVERSITÀ DI PISA

Data Mining Project

Gun Incidents in the USA

MARC LLOBERA VILLALONGA
PATXI JUARISTI PAGEGI

Data Mining (309AA)

Laurea Magistrale in Informatica

08-01-2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

This projects consists of a data analysis based on data mining tools, using Python as programming language, and focusing on four main topics: data understanding and preparation, clustering analysis, predictive analysis and time series analysis, which will be explained in detail in separate sections of this report.

The analysis will focus on gun incidents in the USA, and the most frequent characteristics of such incidents, to later analyze their most common causes, evolution over the years, places with the most frequent incidents, characteristics of the participants... Moreover, to enrich this analysis and look at possible correlations, we will use data from the USA congressional election results and data on poverty percentage for each USA state and year. This data has been obtained from three different datasets that have been provided for this project [1].

2. DATA UNDERSTANDING AND PREPARATION

Data understanding and preparation phase is essential in any data mining project, where the dataset is meticulously explored, cleaned and transformed. In this section, we will explain the steps taken to ensure the accuracy, consistency and relevance of the data to the project objectives.

As mentioned in the introduction, in this project three datasets have been used:

- **Incidents dataset**: Gun incidents in the USA, including data of the date, location and multiple characteristics of the participants and the incident.
- **Poverty rates dataset**: Poverty percentage for each USA state and year.
- **Congressional elections dataset**: Winner of the congressional elections in the USA, for each year, state and congressional district, including the amount of votes obtained.

2.1 Data understanding

First step has been to download and read the datasets. Then, before starting with any task, some basic commands, such as `info()`, `describe()` or `head()` have been used to get an idea of the data content. As a first modification, date values have been changed to *datetime* format, and the numeric values to *numeric*, to then start with more specific adaptations.

2.1.1 Data quality assessment

In this section, tasks performed for data quality assessment will be described.

Missing values: First step has been to handle missing values. It has done only in the dataset of incidents, because the other two do not have missing values.

Rows that do not have information about the incident characteristic of the location, have been removed directly, since they do not have some of the most relevant information. Then, the focus has been to fill somehow the empty values of the age related columns. Using several different techniques, the amount of empty values has been reduced. Similar techniques have been used for the age group and gender.

In the next chart it can be seen the difference of the missing values after applying the modifications. Although there are still many empty values, fields such as `incident_characteristic2` or `notes` are not of much relevance to the analysis. It can also be seen that the age-related values were slightly reduced. They could have been further reduced using more generalist techniques, but

it has been chosen to use only existing values in the data set so as not to distort reality too much with fictitious values.

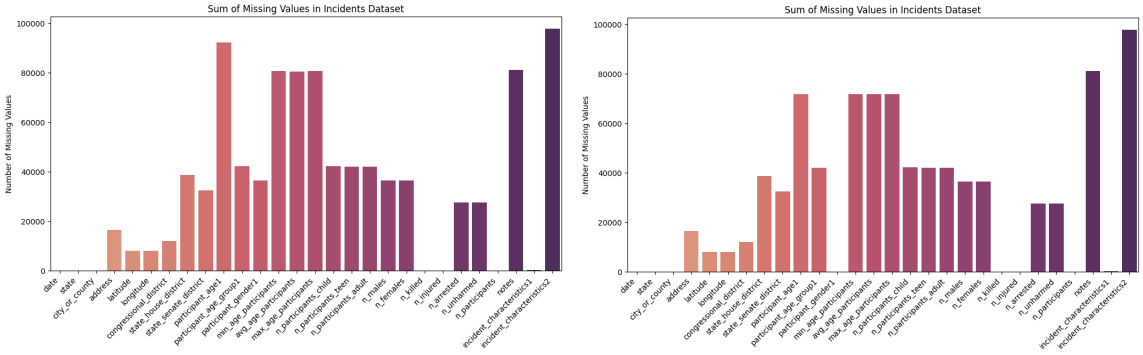

Figure 2.1: Before/after comparison of missing values

Duplicated records: Incident data set had duplicate records, so they were deleted.

2.1.2 Distribution of variables

Evaluation of poverty percentages dataset: The distribution among states has been fairly common, with no notable oddities. Nevertheless, it has been seen that the dataset has a state as called "United States". Since it was not known why that field was there, it was not removed. After all modifications two different dataset have been defined, one including "United States", in case that the purpose is to analyze the general poverty rates, and another with this state removed, if the purpose is to analyze the differences between states for example.

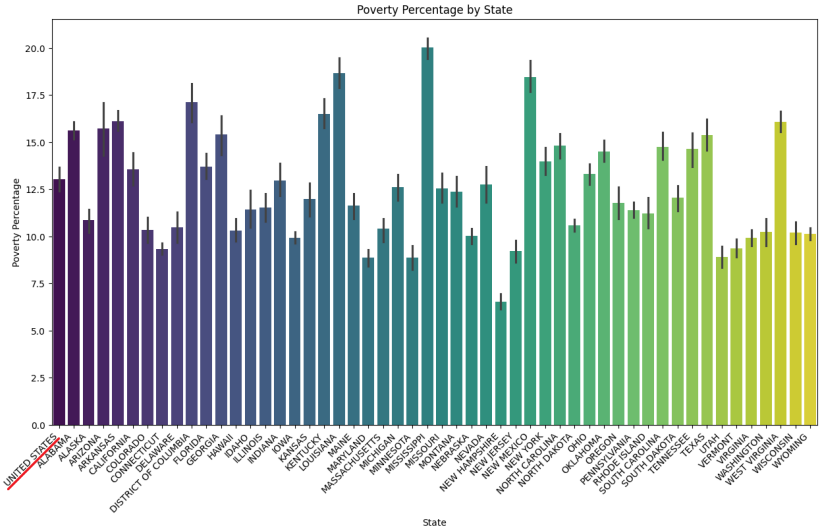

Figure 2.2: Distribution of poverty rates among US states

Referring to the general distribution of the poverty rates throughout all the US, it can be seen that even tough there are some rates that are higher, the distribution is skewed right.

Evaluation of elections dataset: Similar analysis has been performed for the elections dataset. Firstly, the evolution of the amount of votes per year has been checked. In general, it can be said that the participation in the elections has been increasing by the years.

Nevertheless, it is known that the elections are every 4 years, while in this dataset there are entries every 2 years. That's why in the graph we can see the difference in the amount of votes every two years. It has been assumed that this votes correspond to the midterm that the US does for the elections. After finishing with the modifications it has been separated in two datasets: one with all the voting campaigns and the other one just with the elections.

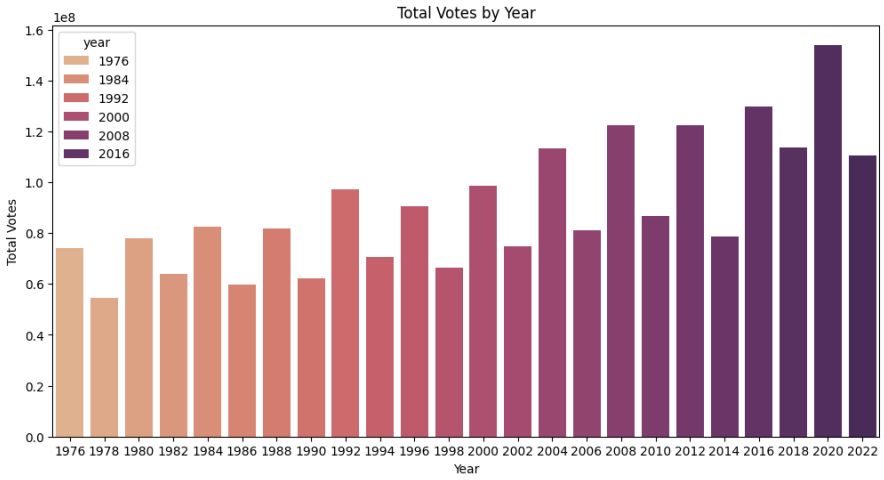

Figure 2.3: Voting evolution by year

With the scatter plot of the relations between candidate votes and total votes, some anomalies in the dataset have been detected. Many values have been located in the (0,0), which means that `candidatevotes` and `totalvotes` are equal and also points can be seen in the top right of the chart with huge values, which are clearly anomalies as well.

To fix it, Interquartile Range (IQR) for both `candidatevotes` and `totalvotes` has been calculated, a measure of statistical dispersion based on quartiles, which is used to define upper and lower bounds for identifying outliers. Outliers have been identified by defining bounds outside 1.5 times the IQR from the quartiles and have been removed from the main dataset, to ensure that the subsequent analyses and visualizations are not influenced by extreme values. The difference between the first and final scatter plot is shown below:

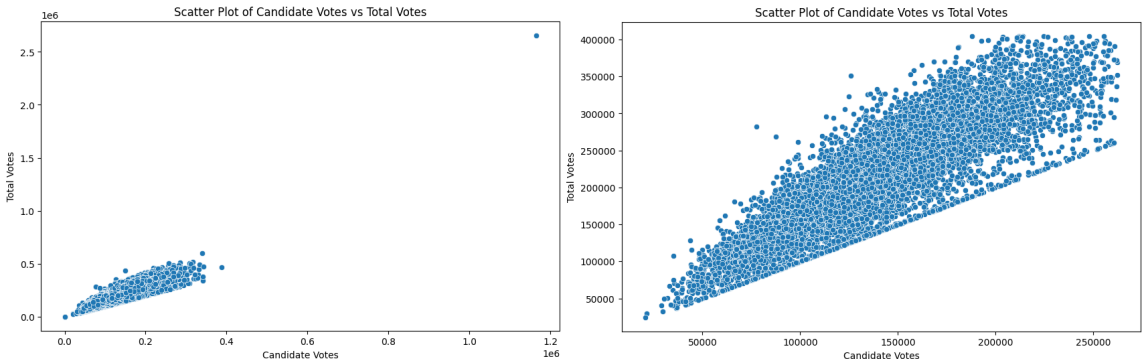

Figure 2.4: Before/after comparison of scatter plot of candidate votes vs total votes

Apart from that, the structure of the dataset has not been very user-friendly, so the decision to adapt it has been taken. The dataset only included data of the winner, and the amount of votes they obtain, but it did not include information about the amount of votes that the rest of the parties obtained.

First, the decision to just keep the republican and democrat parties has been taken. It has been checked the percentage that these little parties occupy in the results and it was less than a 1% of the entire votes during the history. Therefore, these parties votes have been set as republican or democrat depending on each case.

Afterwards, a simple calculation has been performed to set the `candidatevotes` amount to the winner party (republican or democrat) and the subtraction of `totalvotes` minus `candidatevotes` to the looser party. Finally, `candidatevotes` column has been removed, since it is not useful anymore.

To conclude with the modifications, `congressional_district` information has been removed, in order to simplify the dataset and merge the rows that have the same year and state. This way, the information about the votes that each party has obtained will be grouped by year and state, which has been the winner, and the total amount of votes per party and in total.

The final structure of the dataset is this one:

	year	state	totalvotes	republican_votes	democrat_votes	party
0	1976	ALABAMA	984181	315740	666129	DEMOCRAT
1	1976	ALASKA	118208	83722	34141	REPUBLICAN
2	1976	ARIZONA	729002	362192	363365	DEMOCRAT
3	1976	ARKANSAS	336389	74638	260997	DEMOCRAT
4	1976	CALIFORNIA	7442501	3266248	4150218	DEMOCRAT

Evolution of incidents over time: Incidents dataset analysis has begun by examining the temporal evolution of incidents.

The dataset includes values from around 2014 to mid-2018, with a large gap until 2028, which follows with values to 2030. The reason for this gap is unclear. Whether it is an error or a predictive entry. Regardless, the future data is irrelevant for analysis, so it has been excluded.

Moreover, there has been a notable difference between incidents before and after 2014. However, this difference is not due to 2013 being the safest year but rather because there are significantly fewer values for that year in the dataset. To maintain result integrity, entries older than 2014 have been removed.

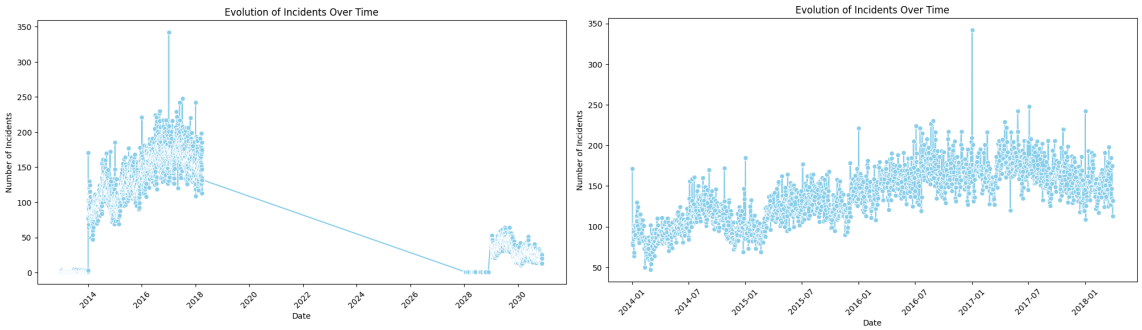

Figure 2.5: Before/after comparison of incidents evolution over time

Geographical distribution of incidents: To analyze the geographical distribution of incidents, we have used `geopandas` library [2].

This tool allows to detect outliers in the geographical data, revealing unexpected incidents around India instead of the expected ones in the USA, as shown in the following image.

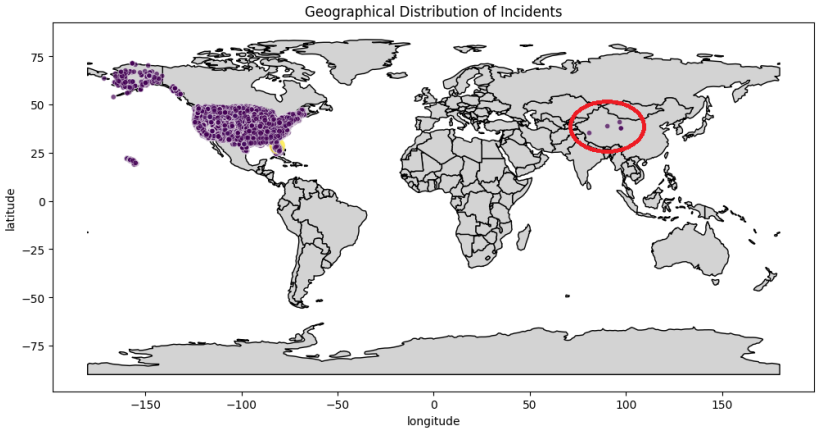

Figure 2.6: Outliers detection in geographical distribution

To fix that, filtering criteria has been refined based on the latitude and longitude values. Latitude and longitude ranges have been set to cover the area of the United States, including Alaska and Hawaii that have different coordinates that the main US region. These adjusted ranges ensure that only incidents falling within the geographical coordinates of the entire United States are retained in the filtered dataset.

Distribution of participant age: As shown in the image below, initial participant age distribution graph appeared odd. This occurred because there were some outliers with extremely high and extremely low values, and when the age values were added to replace the missing values, this outliers made a nonsense distribution. To solve this problem, only values between 0 and 100 years have been taken. With this modification, the distribution that it is shown in the right side of the image has been obtained.

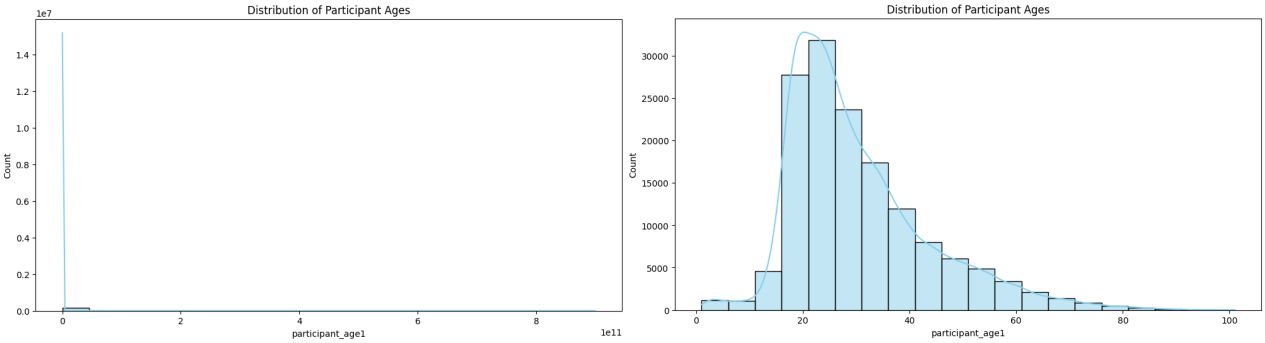

Figure 2.7: Before/after comparison of participant age distribution

Distribution of participant gender: Apart from male and female values, there has been an entry with the gender ”Male, female”. Since it has been just one entry, to simplify the analysis, this entry has been removed, converting it to ”Male”, resulting in just two categories: male and female.

2.1.3 Pairwise correlation

The goal of pairwise correlation analysis is to understand the linear relationship between pairs of variables. The correlation coefficient ranges from -1 to 1, where 1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation and 0 indicates no correlation.

The pairwise correlation matrix for the incidents dataset reveals few noteworthy correlations. While there is a clear positive correlation with variables measuring age, it is considered obvious and not significant. The most interesting finding is the stronger correlation between incident consequences (`n_killed`, `n_arrested`, `n_injured`...) and males compared to females.

Additionally, examining the correlation matrix between incidents and elections, as well as incidents and poverty rates, reveals no significant correlations, as all values are close to 0.

However, a slight correlation can be observed between poverty rates and elections. Even tough it is not a really clear and obvious correlation, it can be observed an interesting difference between the correlation of the poverty percentage and the political party votes. The poverty percentage has a negative correlation of -0.36 with democrat votes, while the correlation with republican votes is close to zero, -0.08. So this indicates that areas with higher poverty rates are more likely to have fewer democrat votes.

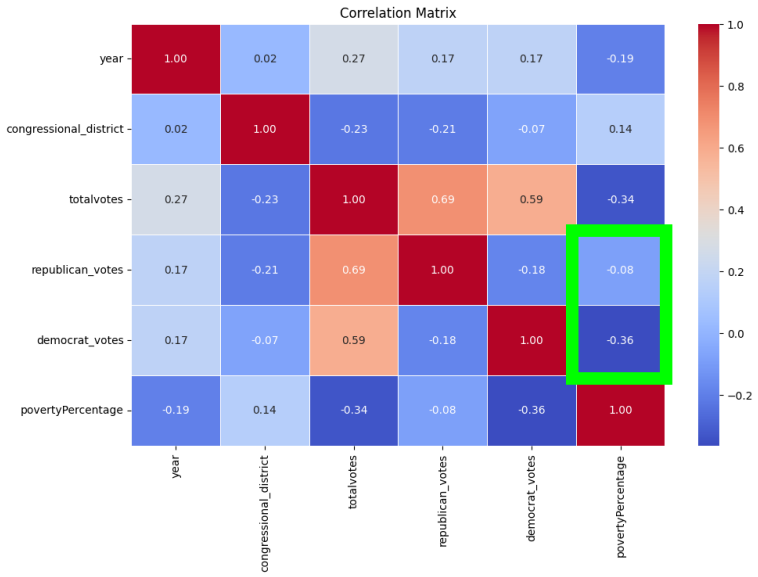

Figure 2.8: Elections and poverty rates correlation matrix

2.2 Data preparation

After extensive dataset modifications, quality of the data has been improved, increasing the knowledge about it in order to extract new interesting features to describe the incidents. To do so, several indicators have been obtained.

Using the year as the period (extracted from the date column) and grouping the data by city and year, the total number of males has been calculated, to later extract the proportion of males involved in incidents for the same city and in the same period.

Next, to assess the proportion of injured and killed individuals in each congressional district within a specific time frame, a similar approach has been used. Group the incidents by congressional

district, calculate the totals, and then obtain the percentages.

Combining the data obtained with new modifications, the evolutions of participant consequences in a specific city has been obtained. Below the example, using Boise as city to test.

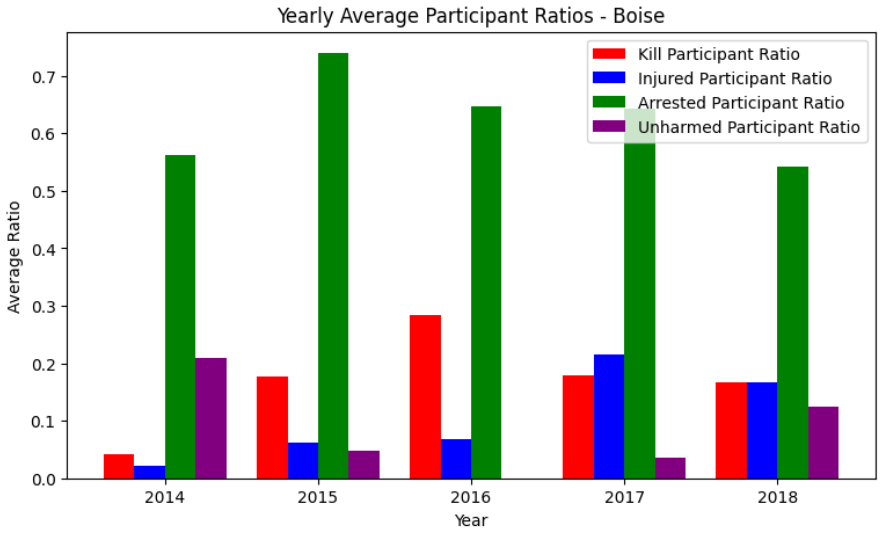

Figure 2.9: Participant consequences evolution in Boise

To conclude the exploration of new characteristics, the average age of incident participants has been calculated for each city or state, to examine how it changes over different time periods. To this approach, incident types have been included, to jointly look at the comparison between the most common incident types. Below it can be seen the chart, using Las Vegas, Orlando and San Francisco as example cities.

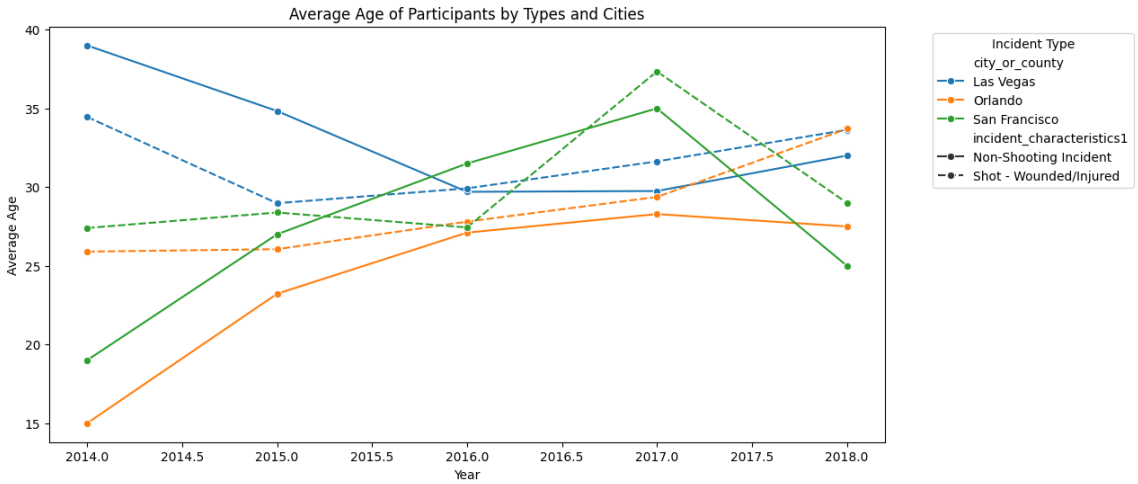

Figure 2.10: Evolution of incidents by average age and type

3. CLUSTERING ANALYSIS

Clustering analysis is a data exploration technique that aims to group similar data points together based on certain features or characteristics. It plays a crucial role in uncovering underlying patterns and structures within large datasets, allowing the identification of distinct subsets or clusters.

3.1 K-Means clustering

First clustering approach used has been clustering analysis by K-means.

3.1.1 Identification of the best k value

To start, *k-means* clustering on the `incidents_dataset` has been performed. Selected features (all columns) have been standardized using `StandardScaler`, and the optimal number of clusters (k) has been determined using `KneeLocator` from `kneed` library, which employs the Elbow Method to do it. In this case, the optimal k value obtained has been 3. After calculating it, a plot has been generated to visualize the ”elbow” point, aiding in the identification of the optimal number of clusters.

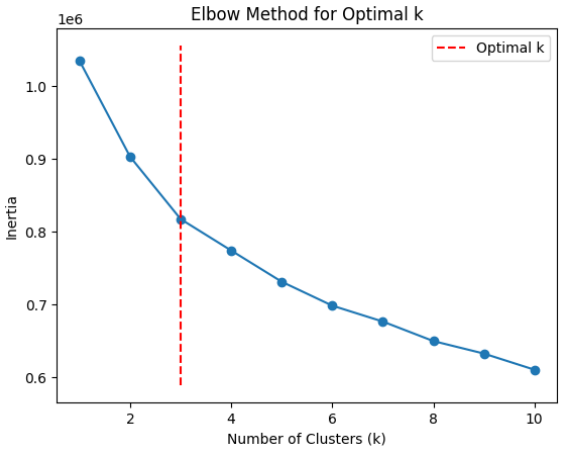

Figure 3.1: Elbow method optimal k value

3.1.2 Cluster generation

After identifying the best k value for clustering, *k-means* clustering on the standardized data has been carried out.

The algorithm assigns each data point to a cluster. After clustering, centroids of each cluster have been calculated, and transformed them back to the original scale using the scaler.

These centroids represent the average values of various features within each cluster resulting from the *k-means* clustering. Each row corresponds to one cluster, and the columns display the centroids of different features.

3.1.3 Cluster evaluation

Obtained three clusters have been evaluated in order to define which is the best. To achieve that two different techniques have been used:

- **Silhouette Score**: This approach measures how well-separated the clusters are. A value close to 1 indicates well-defined, distinct clusters, while a score near 0 suggests overlapping clusters. In this case, a score of 0.160 has been obtained, which indicates slight separation between clusters. However, the clusters may not be perfectly well-defined.
- **Inertia**: It measures cluster compactness, represented by the sum of squared distances between data points and their cluster center. Lower inertia values indicate more compact clusters. The inertia value obtained in this cluster generation, has been relatively high, suggesting that the clusters are not very compact, possibly spread out or loosely grouped.

Interpreting these metrics together, we can consider that the Silhouette Score suggests moderate separation between clusters, but not highly distinct, while the relatively high Inertia value indicates that the clusters are not very compact. In summary, while the clustering is providing some separation, there is room for improvement, so it could be interesting to test other clustering techniques.

3.2 Density-based clustering

To test another clustering approach, density-based clustering has been performed. In this case, a filtered dataset by state (California as the testing state) has been used instead of the entire dataset.

3.2.1 Study of clustering parameters

After having defined the dataset with a unique state, it has been followed with the density-based clustering analysis using DBSCAN. As done before, selected features have been standardized for the state of California using `StandardScaler`. Then, distance to the k^{th} nearest neighbor for each data point has been calculated and visualized the resulting k-distance graph. The code utilizes the `KneeLocator` algorithm to identify the optimal epsilon (neighborhood distance) by finding the knee point in the graph. In this case, the optimal epsilon value obtained has been around 5.2. This

process helps determine a suitable parameter for the DBSCAN algorithm when clustering the data.

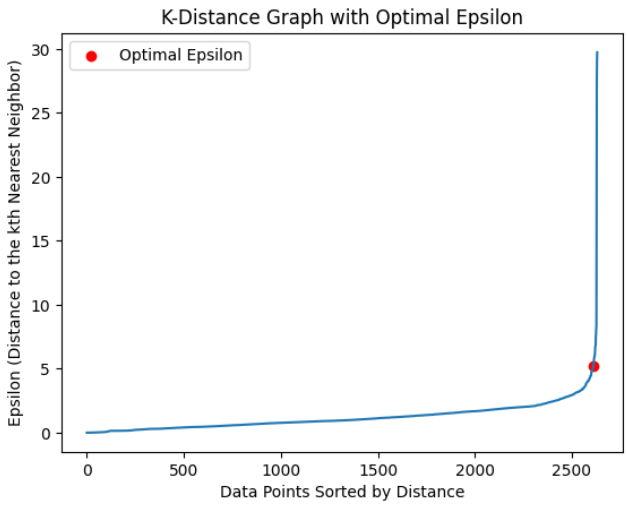

Figure 3.2: K-Distance graph with optimal epsilon

3.2.2 Characterization and interpretation of obtained clusters

With the optimal epsilon value obtained, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) has been conducted. DBSCAN algorithm classifies data points into clusters based on their density within defined neighborhoods.

Resulting clusters are visualized in a scatter plot, where each point represents an incident. The plot uses `avg_age_participants` and `n_participants` as axes, and different colors represent distinct clusters identified by DBSCAN.

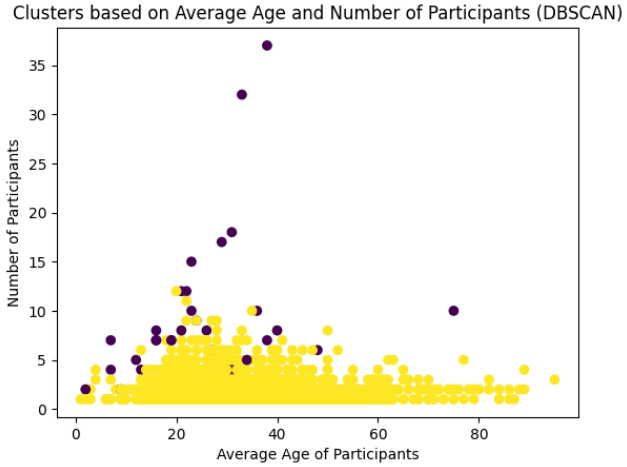

Figure 3.3: Clusters based on average age and number of participants (DBSCAN)

From the results obtained, it can be seen that the clustering creates two clusters, which, looking at the results, it can be affirmed that they are differentiated between the main trend and the less frequent cases. That is, the yellow dots in the graph represent the most common incidents, which are those with the lowest number of participants. On the contrary, the incidents with the highest number of participants are included in the second cluster, represented by the purple dots.

3.3 Hierarchical clustering

To conclude the clustering analysis, an analysis by hierarchical clustering has been performed. The code iterates through the most used linkage methods:

- **Single Linkage**: Measures the shortest distance between points in two clusters. Sensitive to outliers and tends to form elongated clusters.
- **Complete Linkage**: Measures the longest distance between points in two clusters. Tends to produce more compact, spherical clusters.
- **Average Linkage**: Uses the average distance between points in two clusters. Balanced approach, less sensitive to outliers.
- **Ward Linkage**: Minimizes the variance within clusters. Tends to create equally-sized, compact clusters.

As in the previous case, California has been used as the testing state, with a downsampled subset of the incidents dataset to visualize the results more clearly. For each linkage method, dendrogram has been visualized to see the number of clusters formed.

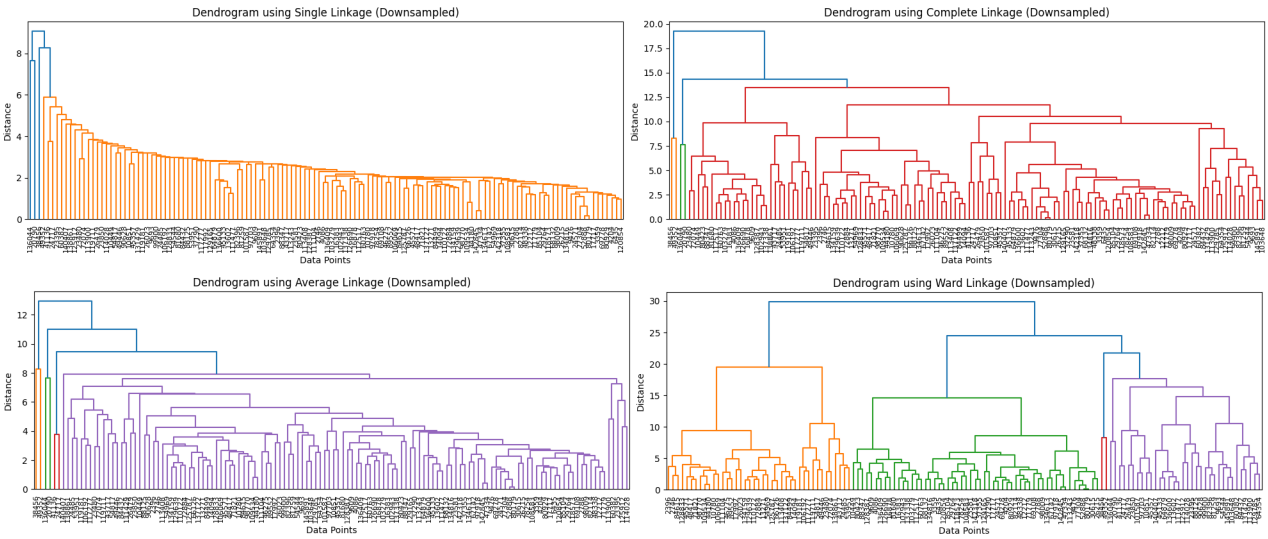

Figure 3.4: Dendrograms comparison between linkage methods

Additionally, Silhouette Score has been calculated for each clustering result. This metric measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation). Higher Silhouette Scores indicate better-defined clusters. In this case, Single Linkage has the highest silhouette score (0.314), suggesting that it produces more distinct clusters.

The number of clusters varies across methods, with Complete and Ward Linkage resulting in higher numbers (28 and 34, respectively) compared to Single and Average Linkage. If the goal is to have a more granular segmentation, Complete or Ward Linkage might be preferred.

Regarding the cluster structure, different linkage methods lead to variations in cluster shapes and sizes. Single Linkage tends to create elongated clusters, while Complete and Ward Linkage aim for more compact clusters. Average Linkage represents a compromise between these extremes.

In conclusion, if the goal is to prioritize well-defined, compact clusters, Ward Linkage might be considered the best in this context. However, it's essential to consider the trade-offs between granularity and interpretability when choosing the appropriate hierarchical clustering approach.

3.4 Evaluation of clustering approaches

To perform a final evaluation and comparison of the clustering approaches (k-means, density-based clustering, and hierarchical clustering), several factors were taken into account, such as, Silhouette Score, interpretability, number of clusters, computational complexity and visualization.

Regarding to the Silhouette score, there has been a clear winner. Density based clustering has a high score, compared with the rest of approaches. On the other hand, the number of clusters is quite diverse, so they have been evaluated more for their interpretability and visualization than for the quantity itself. However, cluster values such as 28 or 34 are too many for this analysis.

Clustering Method	**Silhouette Score**	**Number of Clusters**
K-means	7	3
Density-Based	28	1
Hierarchical - (Single)	0.314	7
Hierarchical - (Complete)	0.164	28
Hierarchical - (Average)	0.186	19
Hierarchical - (Ward)	0.155	34

Table 3.1: Silhouette scores of clustering methods

With respect to the visualization and interpretability, K-means assigns each data point to the cluster whose centroid is nearest. Interpretability is relatively high as the centroid of each cluster can be distinguished and analyzed to understand the average behavior of the points in that cluster.

DBSCAN for his part, identifies clusters based on density-connected regions. In this example, interpretability is not very clear, as the main cluster takes all the relevance, making it difficult to distinguish. It should be noted that it has been tested with a specific state (California), so using it differently, the results may vary.

Lastly, hierarchical clustering organizes data into a tree structure (dendrogram). Choosing ward linkage method, interpretability is quite clear, as the resulting branches and clusters can be observed at different levels of the hierarchy.

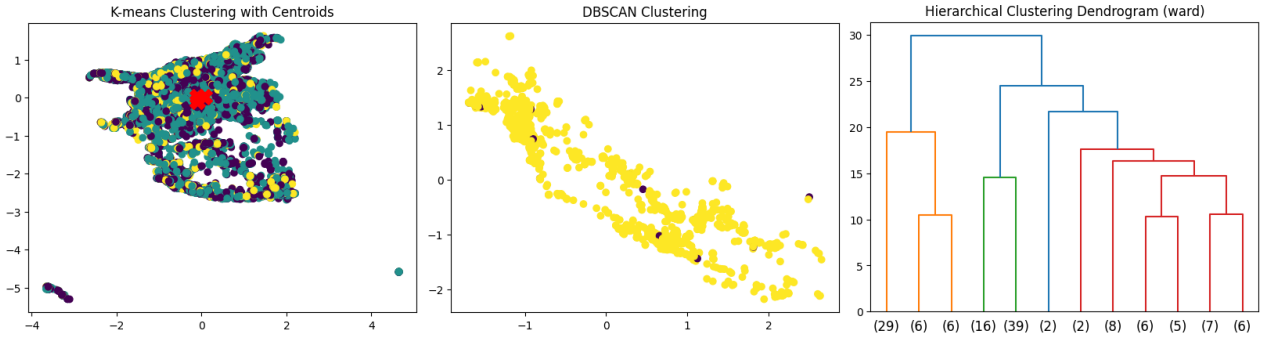

Figure 3.5: Cluster visualizations by method

4. PREDICTIVE ANALYSIS

Following the clustering analysis, a predictive analysis has been conducted, which will be detailed in this chapter. For this predictive analysis, the incidents dataset has been used.

4.1 New feature definition

Firstly, new features related with the time, geographical position and participant features have been defined, which improve the later classification.

Based on the `date` column, month, day of the week and year have been extracted. Moreover, a column was added to determine if the incident day was a weekend or not, and another one to determine in which season it was: spring, summer, autumn or winter.

	date	month	day_of_week	year	is_weekend	season
0	2015-05-02	5	5	2015	1	spring
1	2017-04-03	4	0	2017	0	spring
2	2014-01-18	1	5	2014	1	winter
3	2018-01-25	1	3	2018	0	winter
4	2016-08-01	8	0	2016	0	summer

Then, various new features that take into account the state and city of the incident have been created, including some participant features with them. These include, the total amount of incidents per state and per city, an index of the severity of incidents per city and state, which is obtained by the sum of the killed and injured people in each area, the average age of the participants on incidents per zone and another column to get the female participation in each zone incidents, because the female participation is quite lower than the male one.

4.2 Preprocessing

Preprocessing step is crucial for building and evaluating machine learning models as it ensures that the data is properly formatted and scaled before training the models on it.

To start with the this step, columns that were not going to be used have been removed, in order to reduce the amount of resources that will be used to speed up the analysis and prediction processes. At the same time, missing values have been filled using the average of the dataset.

Subsequently, since the objective has been to predict if in an incident there have been at least a killed person or not, a binary variable has been created, which determines whether there have been deaths or not in the incidents of the dataset. This has been obtained from the variable `n_killed`, if

it is greater than 0 it will be *True*, and if it is not, it will be *False*. The name of the variable has been `people_killed` and it has been the target variable used for the prediction.

After that, the one-hot encoding of the categorical columns has been performed. The encoded categorical columns were `state`, `city_or_country`, `participant_gender1`, `participant_age_group1`, `incident_characteristics1` and `season`.

In order to reduce the resource usage, thresholds have been specified for some columns. For example, for `city_or_country`, since there are thousands of cities, the prediction was incredibly time consuming and memory problems occurred during the analysis, so a limit of 100 has been specified, to just include those that are most frequent. By contrast, columns like state, gender, age group or season do not have any threshold, since all can be included. For instance, age groups are just three, genders are two, and seasons are four, so it is unnecessary to limit the distinct values.

Once the categorical columns have been defined correctly, one-hot encoding has been performed using `get_dummies` function.

Afterwards, feature scaling process has been conducted. As it has been said before, since the objective has been to predict if in an incident there have been at least a killed person or not, we set the `people_killed` variable as target variable `y`, and the rest of the variables as features `X`.

Feature scaling has applied using the `StandardScaler` to standardize the numerical features in the dataset, ensuring that they have zero mean and unit variance. Then, the dataset has been split into training and testing sets using the `train_test_split` function from `scikit-learn`, with 80% of the data used for training and 20% for testing.

4.3 Model selection and evaluation

In the model selection and evaluation process, five distinct classifiers were employed to address the classification task. Used classifiers have been:

- **K-Nearest Neighbors (KNN)**: A non-parametric algorithm that classifies a data point based on the majority class of its k-nearest neighbors.
- **Naive Bayes**: Relies on Bayes' theorem and assumes independence between features, making it computationally efficient.
- **Support Vector Machines (SVM)**: It constructs hyperplanes to separate different classes, aiming to maximize the margin between them.

- **Logistic Regression**: Models the probability of a binary outcome using the logistic function, providing insights into the relationship between features and the target variable.
- **Random Forest**: Is an ensemble method that builds multiple decision trees and combines their predictions, offering robustness and mitigating overfitting.

Using the different classifiers, model has been fitted to the training data (`X_train`, `y_train`) and predictions have been made on the test set (`X_test`). After predicting, with the aim of evaluating the classifiers, various performance metrics have been calculated, including accuracy, precision, recall, F1 score, and area under the ROC curve (AUC-ROC). These metrics are essential for evaluating how well each classifier performs on the test set.

Additionally, cross-validation (`cross_val_score`) has been employed, with a 3-fold validation to assess the models robustly. Although this technique is time-consuming to obtain the results, it helps to estimate the performance on different subsets of the data, providing insights into its generalization ability.

The results for each classifier, including individual metric scores, cross-validation AUC-ROC scores, and the mean cross-validation AUC-ROC, are shown in the following table. Highlighted in blue it is shown the best result for each metric. This comprehensive evaluation allows for a comparison of the classifiers' performance on the given dataset and helps in selecting the most suitable model for the given classification task.

Classifier	Accuracy	Precision	Recall	F1 Score	AUC-ROC	Cross-Validation AUC-ROC	Mean Cross-Validation AUC-ROC
K-Nearest Neighbors	0.9350	0.9471	0.8456	0.8935	0.9116	0.9606	0.9602
Naive Bayes	0.6816	0.5030	**0.9884**	0.6667	0.7620	0.7762	0.7706
Support Vector Machines	0.9811	0.9861	0.9549	0.9703	0.9743	0.9979	0.9980
Logistic Regression	**0.9888**	0.9796	0.9857	**0.9826**	**0.9880**	0.9975	0.9974
Random Forest	0.9845	**0.9926**	0.9591	0.9756	0.9779	**0.9993**	**0.9991**

Table 4.1: Model classifier evaluation table

Analyzing the obtained results, it's evident that K-Nearest Neighbors (KNN) achieved a high accuracy of 93.5%, indicating that the model performed well in correctly classifying instances. The precision score of 94.71% implies that the positive predictions made by KNN were highly accurate, while the recall of 84.56% suggests that the model has not been as effective capturing a substantial portion of the actual positive instances. The F1 score, a balance between precision and recall, is high at 89.35%. The AUC-ROC score of 91.16% and the mean cross-validation AUC-ROC of 96.03% further validate the robustness of KNN in distinguishing between classes. Nevertheless, most of the obtained metrics are lower than the rest of the classifiers.

On the other hand, Naive Bayes exhibits a lower accuracy of 68.16%, indicating that its overall performance is comparatively weaker. The precision score of 50.30% suggests that positive predictions by Naive Bayes may have a higher false positive rate, while the high recall of 98.84% indicates that the model effectively identified the majority of actual positive instances. The AUC-ROC score of 76.20% and the mean cross-validation AUC-ROC of 77.06% highlight the model's ability to discriminate between classes, though not as strongly as other classifiers.

Support Vector Machines (SVM) and Logistic Regression demonstrate exceptional performance, with accuracy values of 98.11% and 98.88%, respectively. Both classifiers showcase high precision, recall, and F1 scores, indicating a well-balanced ability to classify instances. SVM and Logistic Regression achieve AUC-ROC scores of 97.43% and 98.80%, respectively, further affirming their effectiveness. The mean cross-validation AUC-ROC scores for both models consistently exceed 99%, reinforcing their robustness and generalization capabilities.

Finally, Random Forest achieves an accuracy of 98.41%, with a high precision of 99.22% and recall of 95.81%. The F1 score, AUC-ROC, and mean cross-validation AUC-ROC scores all attest to the model's strong discriminatory power. Random Forest's ability to balance precision and recall, along with its high accuracy, underscores its effectiveness in classification tasks.

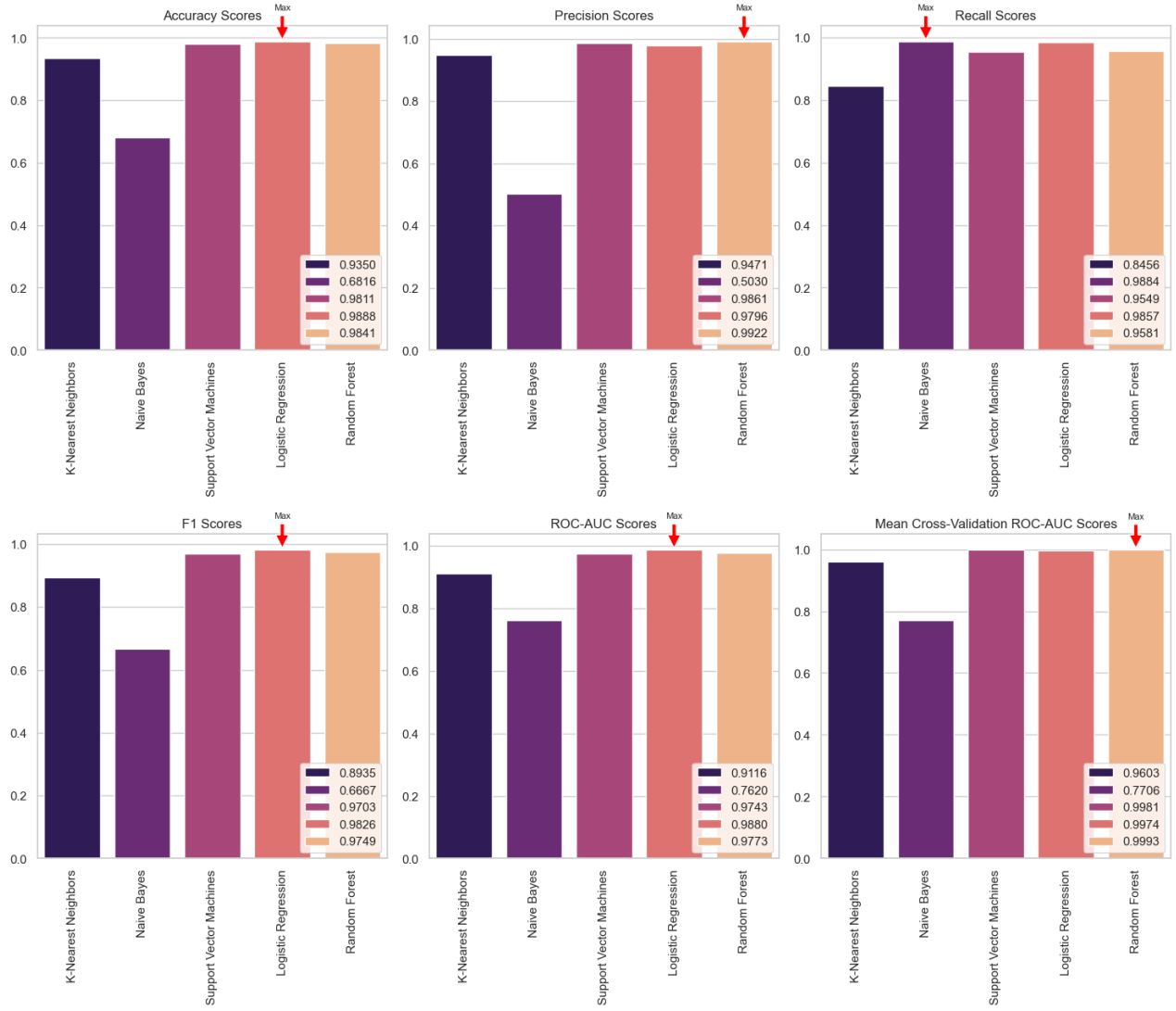

Figure 4.1: Model classifier evaluation

As a conclusion, it's evident that each model excels in different aspects. Logistic Regression and Random Forest showcase high precision, making them suitable for tasks where minimizing false positives is crucial. Naive Bayes, on the other hand, exhibit higher recall, making it favorable when capturing as many positive instances as possible is a priority. Support Vector Machines (SVM) and Logistic Regression stand out for their balanced performance across multiple metrics, making them strong contenders for scenarios where a harmonious blend of precision and recall is desired. In contrast, compared to the other classifiers, K-Nearest Neighbors performed the worst on all metrics.

Analyzing the results obtained in the different metrics, it can be said that the most complete classifier for this case is the Logistic Regression classifier, which should be used in the predictions.

5. TIME SERIES ANALYSIS

This chapter will explain in detail the steps followed to perform the time series analysis, which involves the study and extraction of patterns, trends, and insights from sequential data points recorded over time, enabling the understanding and prediction of future behavior.

The analysis has utilized incidents from 2014 to 2017. Time series scores were calculated for each city weekly over the four years, requiring the addition of a column with a format like *2014-01*, *2014-32*, *2015-7*, *2016-42*, etc., representing the year and week number.

Following, incidents in the same city and week have been grouped, and the total weeks with incidents at each location were calculated. Cities with weeks below 15% of the total 4-year weeks were filtered out from the analysis.

For each week of the four years, a score has been calculated (`city_severity_index`), which is based on a score that was created in the predictive analysis new feature extraction [4.1]. This index, evaluates the severity of the incidents per area. The time series data has this format:

	city_or_county	week	city_severity_index
20480	Knoxville	00-2014	0.008403
11015	Des Moines	00-2014	0.003802
28964	North Charleston	00-2014	0.004717
35642	Saint Paul	00-2014	0.012270
11184	Detroit	00-2014	0.001704

5.1 Clustering

After generating city time series with severity indices, clustering techniques have been applied to uncover patterns and similarities.

Before making the definitive clustering, as it has been done in the clustering analysis [3.1.1] task, a comparison to determine the optimal number of clusters (k) has been performed. To achieve this, K-means clustering algorithm has been used, iterating over a range of potential cluster numbers. Each configuration has been evaluated by calculating both the inertia, and the Silhouette score.

The best Silhouette score has been obtained with 8 clusters, while inertia value was none. This happened because `KneeLocator` might not always provide a clear elbow point, so relying on the silhouette score along with visual inspection of the inertia curve k=8 may be a reasonable choice.

Using the optimal k value, clustering has been performed to then merge the dataframe with city names and their cluster assignments. The overall graph has not shown a clear result, so it has been decided to make a separate representation for each distinct cluster.

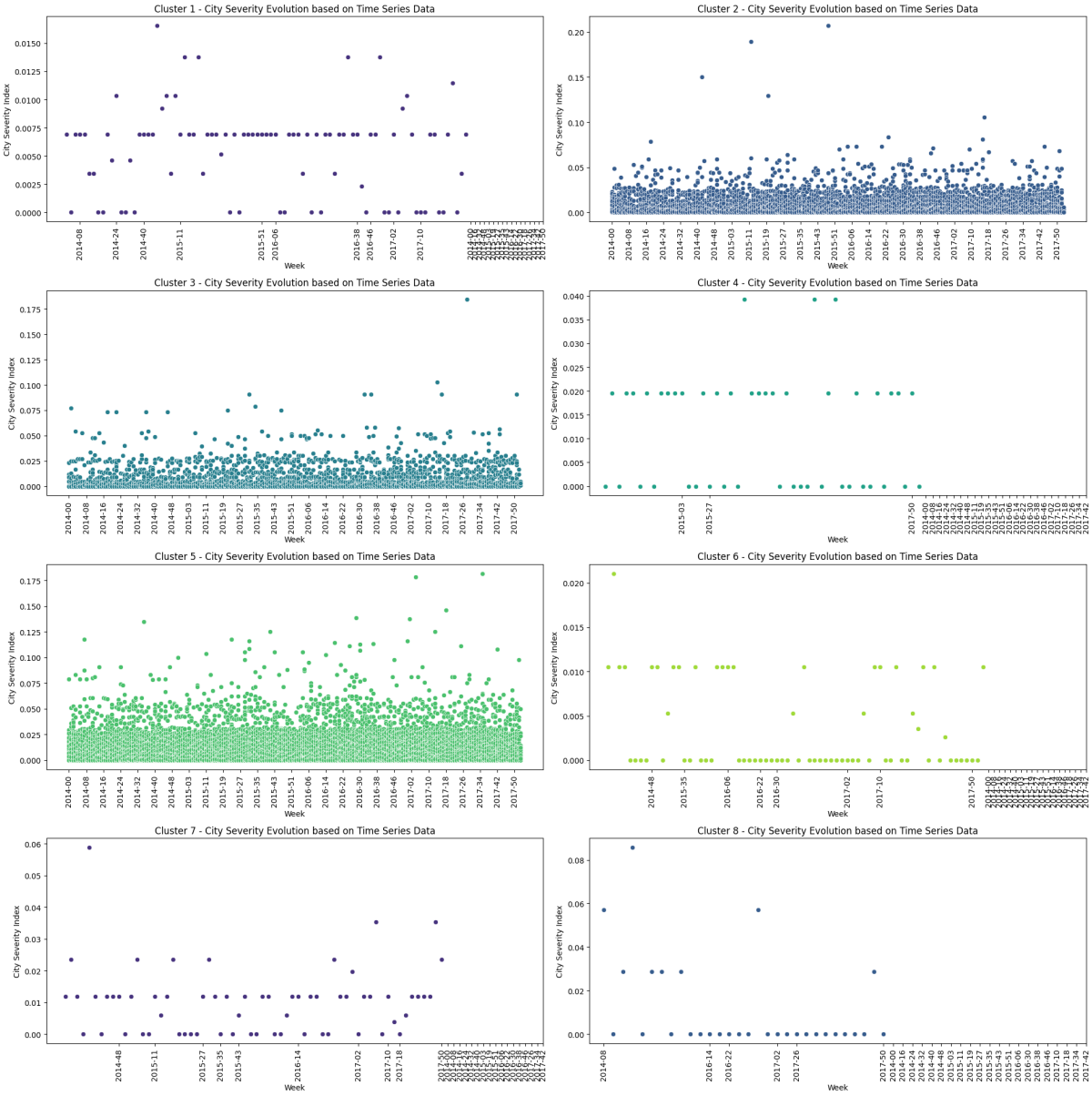

Figure 5.1: K-means clustering of cities based on time series data

There are 3 of them with lot of data, mostly with high scores, while the other 5 have less data with less average score or two defined patters of average date. Due to the constant average score it can be said that cluster 1, 6 and 7 are grouped mostly because of the scoring, while cluster 4 and cluster 8 are probably grouped by a mix of score and week.

5.2 Motif and anomalies extraction

After clustering, attention turned to extracting motifs and anomalies using two parameters: weeks and cities, revealing recurring patterns and exceptional occurrences in the severity index data.

Using the matrix profile technique from the `stumpy` library [3], profiles, motifs and Z-scores for anomaly detection have been computed. Then, the results have been visualized in a plot with motifs (black dots) and anomalies (red dots) within the severity index time series.

Week based

```
Weeks with anomalies: ['2014-41' '2015-23' '2015-29']
Weeks with motifs: ['2017-15' '2014-09' '2015-23']
```

The weeks with anomalies have a clearly high or low value compared with the trend. The values with high severity scores may be due to certain American holidays, special days that occur on those dates or some specific event that occurred and triggered the number of participants in the accidents at those times. By contrast, the low severity value, may be due to some act that reduced, for example, crime and, consequently, the incidents that occurred.

Regarding to motifs, even tough there are marked with black dots there is not any sequence that repeats itself clearly enough to affirm that it is a pattern, therefore it could be said that all the cases throughout the year are random and have no relation to the season of the year..

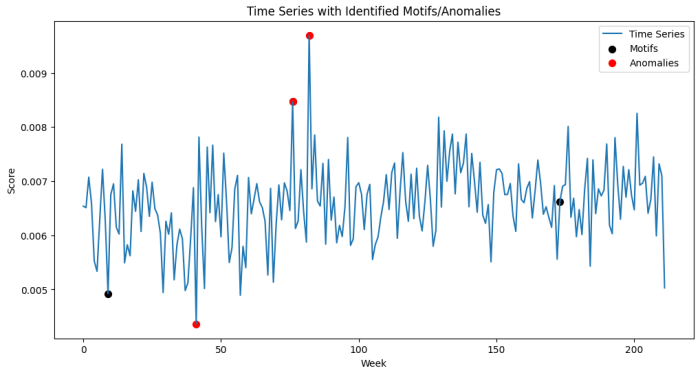

Figure 5.2: Time series with identified motifs/anomalies (week based)

City based

```
Cities with anomalies: ['Providence' 'Baton Rouge' 'Saint Petersburg']
Cities with motifs: ['Tucson' 'Long Beach' 'Springfield']
```

It can be seen that there are three marked anomalies, which clearly have a high severity index value compared to the rest of the cities. This could be an anomaly or simply that these three

cities (”Providence”, ”Baton Rouge” and ”Saint Petersburg”) are more likely to have higher severity incidents than the average of the rest of the cities.

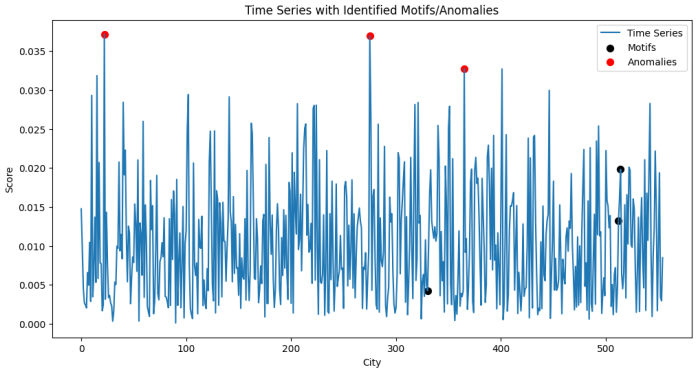

Figure 5.3: Time series with identified motifs/anomalies (city based)

To check for anomalies, time series has been display including just the cities with anomalies. This way, it can be checked if there are possible anomalies, or there are just more dangerous cities than the average. Observing the graph, it can be seen clearly that there is a considerable anomaly in the case of Saint Petersburg, where the value of week 2014-07 is huge compared with the trend.

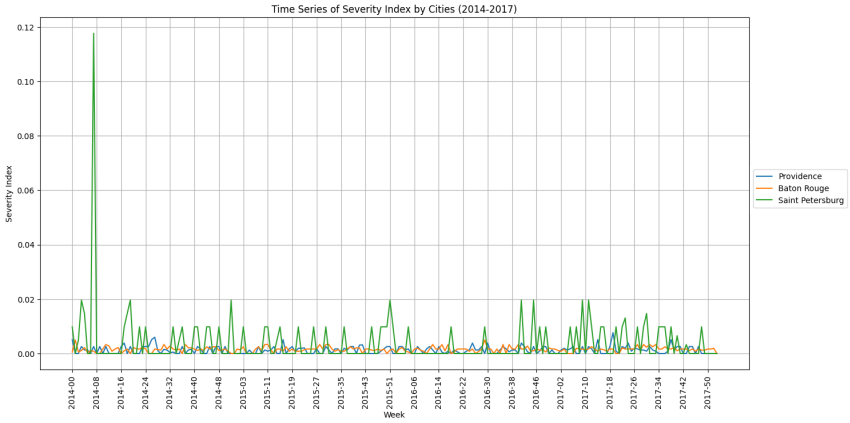

Figure 5.4: Time series of severity index (Providence, Baton Rouge and Saint Petersburg)

Anyway, after removing this particular line, the same graph has been shown again, and the values for Saint Petersburg are still strange compared to the other two cities. Therefore, this city has been removed from the dataset.

Then, to conclude, we compare the two cities marked as anomalies with Detroit, which is considered one of the most dangerous cities in the USA. In the comparison, it can not be detected any remarkable difference, so the city values have been maintained.

Similarly, the analysis process for cities marked as motifs was conducted. Analyzing the resulting graph, despite some repeating small patterns, it can not be said that there is a really clear pattern that should be considered as motif.

5.3 Shapelet extraction

This last section will be focused in the shapelet extraction from the previous time series, according to the class of the binary variable `people_killed`.

Initially, the score for the time series and the `n_killed` variable must be unrelated. As the earlier score has not been unsuitable, a new severity score has been computed by combining several metrics. With the score calculated and the time series recalculated, shapelet extraction began. Data has been normalized, and a shapelet model has been trained using `tslearn` library [4]. The shapelet model extracts discriminative subsequences from the time series data, known as shapelets, which can provide insights into patterns associated with the binary variable `people_killed`.

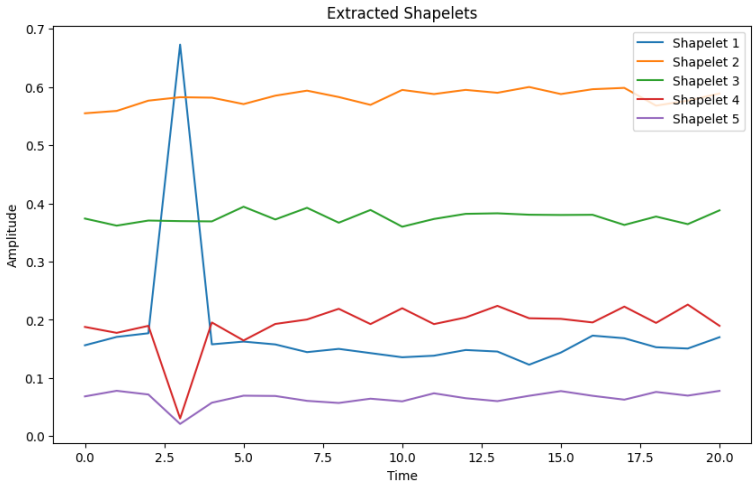

Figure 5.5: Extracted shapelets from the time series

It can be seen that the amplitude is quite constant in almost all the shapelets, instead for the shapelet 1. A constant amplitude suggests that the pattern represented by the shapelet remains relatively stable, while variations may indicate changes in the underlying data.

The peak in the shapelet and also the little trough in shapelet 4 might indicate important events or anomalies in the time series data. Moreover, we can observe similarities in the patterns of some of the shapelets. Similarities may indicate common patterns across multiple cities, while differences might highlight unique characteristics.

BIBLIOGRAPHY

[1] **Project Datasets**. UNIPI Didawiki. Data Mining 2023/2024 *(Accessed on 16-11-2023)* `http://didawiki.cli.di.unipi.it/lib/exe/fetch.php/magistraleinformatica/dmi/gun-data.zip`.

[2] **Geopandas**. Geopandas Library Documentation *(Accessed on 16-11-2023)* `https://geopandas.org/en/stable/docs.html`.

[3] **The Matrix Profile**. Stumpy. The Matrix Profile v1.12.0 *(Accessed on 03-01-2024)* `https://stumpy.readthedocs.io/en/latest/Tutorial_The_Matrix_Profile.html`.

[4] **tslearn**. Shapelets. tslearn v0.6.3 *(Accessed on 03-01-2024)* `https://tslearn.readthedocs.io/en/stable/user_guide/shapelets.html`.