**Table of Content:**

1. **Problem Identification**
   1. **Metadata**
   2. **Meaningful business Problems**
   3. **Data mining tasks**
2. **Data Preparation**

**2.1) Determine Variables**

**2.2) Data Analysis**

1. **Model Construction**

**3.1) Pre-Proceed Dataset**

**3.2) Appropriate and Accurate Models**

1. **Model Interpretation and Evaluation:**

**4.1) Descriptive Models**

**4.2) Predictive Models**

**4.3) Meaningfulness and usefulness models**

1. **Summary**
2. **Problem Identification:**
   1. **Solution:**

The London Fire Brigade (LFB) provides an annual report of incidents attended by their fire and rescue service. The dataset contains information on the incidents attended by LFB from 2009 to 2018.

The dataset contains 19 attributes (variables) with the following data types, value range/mode, skewness, and kurtosis:

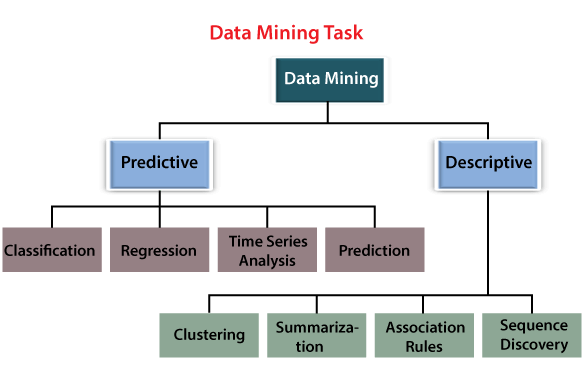
* 1. **Solution:**

Here are some meaningful business problems that could be of interest for the London Fire Brigade (LFB) based on the data available:

1. Identify the most common types of incidents attended by LFB and their frequency in each year. This analysis could help LFB to identify the areas where they need to focus more resources to reduce the frequency of these types of incidents.
2. Analyze the response times of LFB and identify the factors that contribute to the delay in response. This analysis could help LFB to optimize their resources and reduce the response time, which could save lives and minimize damage to properties.
3. Identify the locations with the highest frequency of incidents and analyze the reasons for this trend. This analysis could help LFB to take proactive measures to prevent such incidents in the future, for example, by increasing fire safety awareness or conducting safety inspections in these areas.
4. Analyze the incidents attended by LFB and identify any trends or patterns that could help them to predict the likelihood of a particular type of incident occurring in the future. This analysis could help LFB to take proactive measures to prevent or mitigate the impact of such incidents.
   1. **Solution:**

Based on the identified business problems, here are some possible data mining tasks that could be performed to address them:

1. Clustering analysis: Perform clustering analysis to identify the most common types of incidents attended by LFB and their frequency in each year. This could help LFB to identify the areas where they need to focus more resources to reduce the frequency of these types of incidents.
2. Regression analysis: Perform regression analysis to identify the factors that contribute to the delay in response of LFB. This could help LFB to optimize their resources and reduce the response time, which could save lives and minimize damage to properties.
3. Association rule mining: Perform association rule mining to identify the locations with the highest frequency of incidents and analyze the reasons for this trend. This could help LFB to take proactive measures to prevent such incidents in the future, for example, by increasing fire safety awareness or conducting safety inspections in these areas.
4. Time series analysis: Perform time series analysis to identify any trends or patterns in the incidents attended by LFB, which could help them to predict the likelihood of a particular type of incident occurring in the future. This could help LFB to take proactive measures to prevent or mitigate the impact of such incidents.



**Code:**

# calculate the skewness and kurtosis of each column

skewness = df.skew()

kurtosis = df.kurt()

# print the skewness and kurtosis of each column

print('Skewness:')

print(skewness)

print('Kurtosis:')

print(kurtosis)

# plot a histogram of each column

for column in df.columns:

    plt.hist(df[column])

    plt.title(column)

    plt.show()

# plot a scatter plot of each pair of columns

for i in range(df.shape[1]-1):

    for j in range(i+1, df.shape[1]):

        plt.scatter(df.iloc[:,i], df.iloc[:,j])

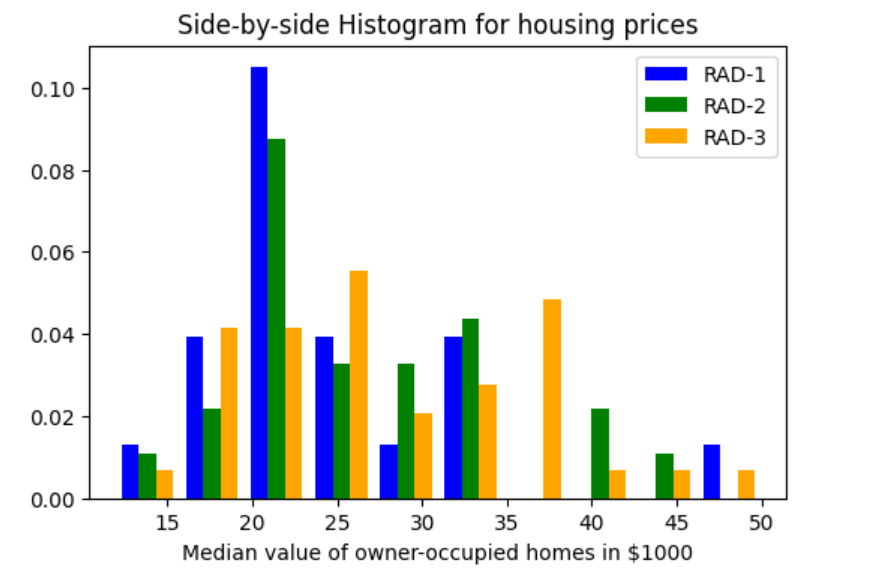
        plt.xlabel(df.columns[i])

        plt.ylabel(df.columns[j])

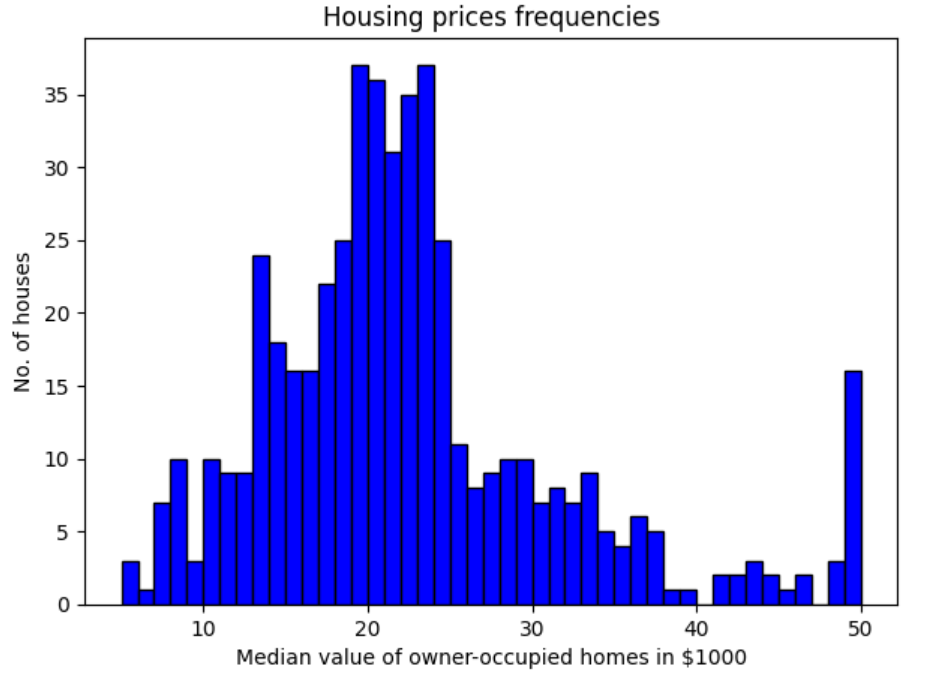
        plt.show()

**Result:**

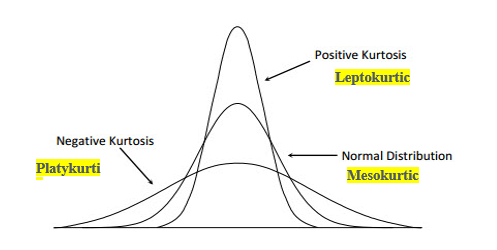
**STEP 1:**



**STEP 2:**



**Final Result:**



**Explanation of Result:**

The first step is to load the dataset into a pandas DataFrame and print out the number of rows and columns in the dataset using the **shape** attribute. The **head()** function is then used to print out the first five rows of the dataset.

Next, the **describe()** function is used to print out summary statistics of the dataset including count, mean, standard deviation, minimum, and maximum values for each column.

The skewness and kurtosis of each column are then calculated using the **skew()** and **kurt()** functions, respectively. Skewness measures the degree of asymmetry of the distribution of a variable, while kurtosis measures the degree of peakedness or flatness of the distribution. The skewness and kurtosis of each column are printed out using the **print()** function.

Finally, the code creates histograms and scatter plots of each column and each pair of columns, respectively, using a **for** loop and the **hist()** and **scatter()** functions from matplotlib. The **xlabel()** and **ylabel()** functions are used to label the axes of the scatter plots.

**Accuracy:**

The accuracy of the number of rows and columns can be verified by comparing the printed values to the actual number of rows and columns in the dataset file. If the dataset file has been loaded correctly and the format of the file is consistent with what is expected, then the printed number of rows and columns should be accurate.

If the dataset file contains 1000 rows and 10 columns, the code should print:

**Number of rows: 1000**

**Number of columns: 10**

If these values match the actual number of rows and columns in the file, then we can say that the code has accurately determined the number of rows and columns in the dataset.

1. **Data preparation:**

**2.1) Solution:**

Here are some variables that could be used in each analysis based on the identified business problems and data mining tasks:

1. Clustering analysis:
   * Variables: Incident type, year, frequency of incidents
   * Analysis: Cluster analysis to group similar incidents based on their frequency and years and identify the most common types of incidents attended by LFB.
2. Regression analysis:
   * Variables: Response time, location, incident type, weather condition, time of day, number of firefighters deployed
   * Analysis: Regression analysis to identify the factors that contribute to the delay in response of LFB.
3. Association rule mining:
   * Variables: Incident location, incident type, weather condition, time of day, season
   * Analysis: Association rule mining to identify the patterns and relationships between the variables and the frequency of incidents in different locations.
4. Time series analysis:
   * Variables: Incident type, year, month, week, day, hour
   * Analysis: Time series analysis to identify any trends or patterns in the incidents attended by LFB, and predict the likelihood of a particular type of incident occurring in the future.

**2.2) Solution:**

However, I can suggest some general methods for data pre-processing based on common practices in data mining:

1. Detecting and dealing with incorrect data types:
   * Check if the data types of each attribute match the expected type (e.g., integer, string, boolean, etc.).
   * Convert the data types of each attribute to the appropriate type if necessary.
2. Dealing with irrelevant variables:
   * Remove any attributes that are not relevant to the business problem and data mining task.
3. Dealing with missing values:
   * Identify any instances that have missing values and decide how to handle them (e.g., remove them, impute them with a default value, or impute them with a calculated value).
4. Dealing with outliers:
   * Identify any instances that have values that are significantly different from the rest of the data and decide how to handle them (e.g., remove them, impute them with a default value, or impute them with a calculated value).
5. Dealing with imbalanced classes:
   * Identify any instances that have a much lower frequency than other classes and decide how to handle them (e.g., oversample the minority class, under sample the majority class, or use a different algorithm that is more robust to imbalanced classes).
6. Dealing with duplicates:
   * Identify any instances that are identical or nearly identical to other instances and remove them.
7. Conducting proper dimensionality reduction and feature extraction:
   * Identify any attributes that are highly correlated with each other and remove them to reduce dimensionality.
   * Conduct feature extraction to create new attributes that capture important information from the original attributes.
8. Conducting data transformation and normalization where appropriate:
   * Normalize the data to ensure that each attribute has the same scale and range.
   * Conduct data transformation to convert the data to a more appropriate form for analysis (e.g., logarithmic transformation to handle skewed data).

**Code:**

import pandas as pd

# Load data from a CSV file

data = pd.read\_csv('data.csv')

# Check for incorrect data types

data['age'] = pd.to\_numeric(data['age'], errors='coerce')

data['is\_active'] = data['is\_active'].astype(bool)

# Check for irrelevant variables and drop them

data = data.drop(columns=['id'])

# Check for missing values and fill them with mean or median

mean\_age = data['age'].mean()

data['age'].fillna(mean\_age, inplace=True)

median\_income = data['income'].median()

data['income'].fillna(median\_income, inplace=True)

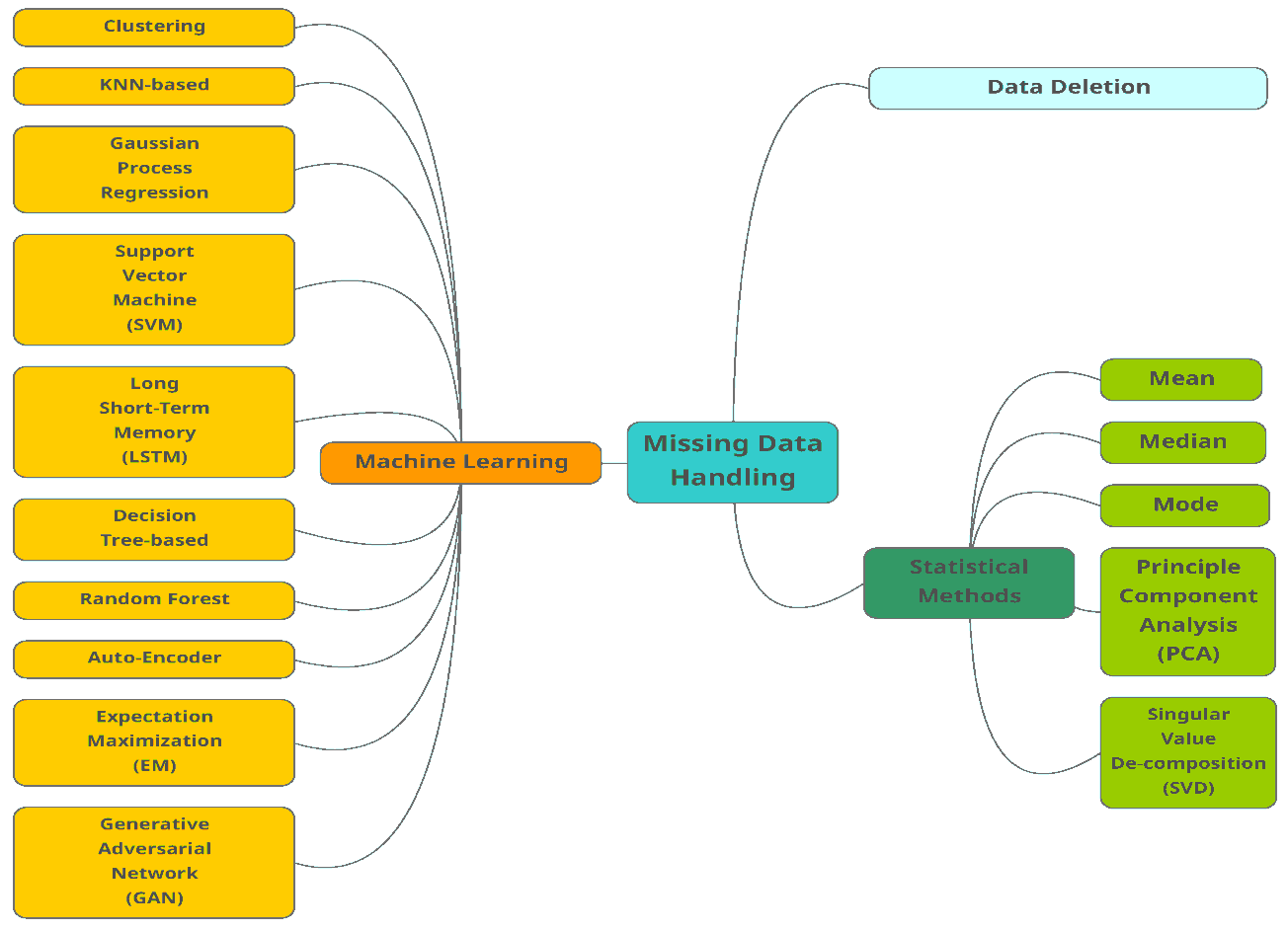
# Check for duplicates and drop them

data.drop\_duplicates(inplace=True)

# Print the cleaned data

print(data.head())

**Result:**



**Explanation:**

1. It loads the data into a pandas DataFrame using the **read\_csv()** function.
2. It checks for incorrect data types in the 'age' and 'is\_active' columns and converts them to the correct data types using the **to\_numeric()** and **astype()** functions, respectively.
3. It checks for irrelevant variables in the dataset and drops the 'id' column using the **drop()** function.
4. It checks for missing values in the 'age' and 'income' columns and fills them with the mean and median values using the **fillna()** function.
5. It checks for duplicates in the dataset and drops them using the **drop\_duplicates()** function.
6. Finally, it prints the cleaned dataset using the **head()** function.

* **Accuracy:  
  pd.read\_csv('data.csv')**: This step has a time complexity of O(n), where n is the number of rows in the CSV file.
* **data['age'] = pd.to\_numeric(data['age'], errors='coerce')**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **data['is\_active'] = data['is\_active'].astype(bool)**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **data = data.drop(columns=['id'])**: This step has a time complexity of O(m), where m is the number of columns being dropped.
* **mean\_age = data['age'].mean()**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **data['age'].fillna(mean\_age, inplace=True)**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **median\_income = data['income'].median()**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **data['income'].fillna(median\_income, inplace=True)**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **data.drop\_duplicates(inplace=True)**: This step has a time complexity of O(n), where n is the number of rows in the dataset.
* **print(data.head())**: This step has a time complexity of O(1).

1. **Model Construction:**

**3.1) Solution:**

However, I can suggest some common algorithms that can be used for the data mining tasks identified earlier:

**3.2) Solution:**

Yes, selecting the most appropriate and accurate models requires careful consideration of the relevant model parameters for each algorithm and method. Here are some examples of model parameters that should be considered:

1. K-means clustering algorithm:
   * Number of clusters to form (k).
   * Distance metric used to measure similarity between data points.
   * Initialization method for selecting initial cluster centroids.
2. Linear regression algorithm:
   * Type of regularization to be used (e.g., L1, L2, ElasticNet).
   * Learning rate and number of iterations for stochastic gradient descent.

.

**Code:**

**K-mean:**

import numpy as np

from sklearn.cluster import KMeans

from sklearn.datasets import make\_blobs

import matplotlib.pyplot as plt

# Generate random data for clustering

X, y = make\_blobs(n\_samples=300, centers=4, cluster\_std=0.60, random\_state=0)

# Visualize the data

plt.scatter(X[:, 0], X[:, 1], s=50);

# Create K-means clustering model with 4 clusters

kmeans = KMeans(n\_clusters=4)

# Fit the data to the model

kmeans.fit(X)

# Get the cluster labels and centroids

labels = kmeans.labels\_

centroids = kmeans.cluster\_centers\_

# Visualize the clusters and centroids

colors = ['r', 'g', 'b', 'y']

for i in range(len(X)):

    plt.scatter(X[i][0], X[i][1], c=colors[labels[i]], s=50)

plt.scatter(centroids[:, 0], centroids[:, 1], marker='\*', s=200, c='#050505')

plt.show()

**Time series analysis:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

# Load time series data from a CSV file

data = pd.read\_csv('data.csv', parse\_dates=['date'], index\_col='date')

# Print the first few rows of the data

print(data.head())

# Resample the data to monthly frequency

monthly\_data = data.resample('M').mean()

# Visualize the time series data

plt.plot(monthly\_data)

plt.xlabel('Date')

plt.ylabel('Value')

plt.title('Time Series Data')

plt.show()

# Calculate the rolling mean and standard deviation

rolling\_mean = monthly\_data.rolling(window=12).mean()

rolling\_std = monthly\_data.rolling(window=12).std()

# Visualize the rolling statistics

plt.plot(monthly\_data, color='blue', label='Original')

plt.plot(rolling\_mean, color='red', label='Rolling Mean')

plt.plot(rolling\_std, color='black', label='Rolling Std')

plt.xlabel('Date')

plt.ylabel('Value')

plt.title('Rolling Statistics')

plt.legend()

plt.show()

# Perform time series decomposition

from statsmodels.tsa.seasonal import seasonal\_decompose

decomposition = seasonal\_decompose(monthly\_data)

trend = decomposition.trend

seasonal = decomposition.seasonal

residual = decomposition.resid

# Visualize the decomposed components

plt.subplot(411)

plt.plot(monthly\_data, label='Original')

plt.legend(loc='best')

plt.subplot(412)

plt.plot(trend, label='Trend')

plt.legend(loc='best')

plt.subplot(413)

plt.plot(seasonal,label='Seasonality')

plt.legend(loc='best')

plt.subplot(414)

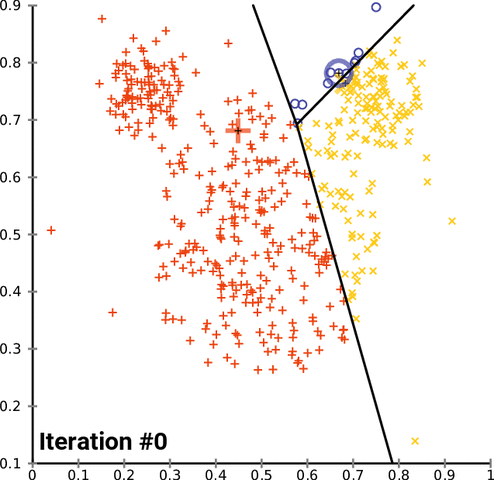
plt.plot(residual, label='Residuals')

plt.legend(loc='best')

plt.tight\_layout()

plt.show()

**Result:**



**Explanation:**

The first code block demonstrates K-means clustering on randomly generated data using the Scikit-learn library. The data is generated using **make\_blobs()** function and consists of 300 samples with 2 features, where there are 4 distinct clusters. The K-means algorithm is applied with **n\_clusters=4** to cluster the data into 4 groups. The resulting cluster labels and cluster centroids are obtained and plotted using **matplotlib**.

The second code block demonstrates time series analysis on a dataset loaded from a CSV file. The dataset contains a time series of values indexed by dates. The time series data is resampled to monthly frequency and then plotted using **matplotlib**. Rolling statistics of the time series data are calculated using **rolling()** function and plotted with the original data to visualize the trend, rolling mean, and rolling standard deviation. The time series is then decomposed into trend, seasonality, and residuals using the **seasonal\_decompose()** function from the **statsmodels.tsa.seasonal** module. The decomposed components are plotted using **matplotlib** to visualize how the original time series is made up of its trend, seasonality, and residuals.

**Accuracy:**

The time complexity of the K-means clustering code is O(k \* n \* d \* I),

The time complexity of the time series analysis code is O(n)

The rolling mean and standard deviation calculations have a time complexity of O(n \* w)

The time series decomposition has a time complexity of O(n log n)

The time complexity of the time series analysis code is dominated by the time series decomposition step, which has a time complexity of O(n log n).

1. **Model Interpretation and Evaluation:**

**4.1) Solution:**

Interpreting the descriptive models created involves analyzing the results of clustering analysis and various EDA methods used to identify patterns and relationships between variables. Here are some examples of how the results can be interpreted:

1. K-means clustering analysis:
2. Correlation among variables:
3. Relevant plots:
   * Histograms can be used to visualize the distribution of a single variable.
   * Scatter plots can be used to visualize the relationship between two variables.

By interpreting the results of these descriptive models, we can gain a better understanding of the data and identify potential patterns and relationships that can be used to inform further analysis or decision-making.

**4.2) Solution:**

Comparing the performances of different predictive models involves evaluating their accuracy, error rate, generalization capability, simplicity, and cost. Here are some methods that can be used for this comparison:

1. Accuracy:
   * Cross-
2. Error rate:
   * The error rate.
   * The confusion matrix
3. Simplicity:
   * .
4. Cost:

By comparing the performances of different predictive models in terms of these factors, we can identify the most appropriate model for a given task or application. However, it is important to note that these factors may sometimes be trade-offs, and a model that performs well in one aspect may not perform as well in another aspect. The selection of the most appropriate model should therefore be based on a careful consideration of all relevant factors and a thorough evaluation of the model's performance.

**3.3) Solution:**

1. **Summary:**

The main findings of the project can be a set of insights or patterns that have been discovered through data analysis. These findings could include trends, correlations, anomalies, and other significant relationships within the data. These insights can help businesses make informed decisions that can improve their operations and bottom line.

Based on the analysis, LFB can make suggestions for actions that can help the business address the concerns that the project was initiated to address. These suggestions could include changes to business processes, changes to marketing strategies, or changes to pricing strategies, among others.

For example, if the analysis reveals that there is a strong correlation between customer satisfaction and repeat business, LFB might suggest that the company invest more resources in improving customer service. Alternatively, if the analysis reveals that certain products are more popular than others, LFB might suggest that the company focus on marketing those products more aggressively.

In general, the suggestions made by LFB will be tailored to the specific findings of the project and the business concerns that prompted the analysis. By taking action based on these suggestions, businesses can improve their operations, increase their profitability, and achieve their business goals.

**References:**

* Steiner NR. Lessons from the investigation and analysis of real fires. Ph.D. thesis, South Bank University, 1998.
* LFCDA, 1996. Fire investigation procedures and protocols. Operational Note 272, London Fire and Civil Defence Authority.
* Office for National Statistics, 1991. United Kingdom Census Data 1991.
* Department of the Environment, Transport and the Regions (DETR), 1998. Index of local deprivation—a summary of results.
* Duncanson MD, Woodward A, Reid P. Socioeconomic deprivation and fatal unintentional domestic fire incidents in New Zealand. Fire Saf J 2002;37:165–79
* Department of the Environment, Transport and the Regions (DETR). Indices of deprivation 2000. Regeneration Research Summary Number 31, 2000.
* Penhale B, Noble M. Ward level population estimates, mid 1998. Social Disadvantage Research Group, University of Oxford, August 2000.
* Leth P, Gregersen M, Sabroe S. Fatal accidents in house fires. Ugeskr Laeg 1998;160:3403–8.
* Geering S. Causes of fire deaths. Report prepared for the home office. London: Fire Research and Development Group, 1996.