CRIME HOTSPOT DETECTION

Presented by,
KARTHIKEYAN R

Introduction

- The "Crime Hotspot Detection" project leverages data-driven insights to improve public safety. By identifying crime hotspots, we aim to empower security solutions and enhance community protection.

- Machine Learning, with its predictive capabilities, offers a transformative approach to understanding and mitigating the challenges posed by crime hotspot detection.

Through data-driven Insights and predictive modelling, this presentation aims to showcase my Machine Learning Capstone Project focused on Predicting crime in Los Angeles.

Abstract

- Crime is one of the most social problems in the country impacting public welfare, children development , and adult socio-economic status. Crime analysis and prevention is a systematic approach for identifying and analyzing trends.

- With the help of crime data analysis , officers can speed up the process of solving crimes and spread awareness in the high prone crime areas.

Objective

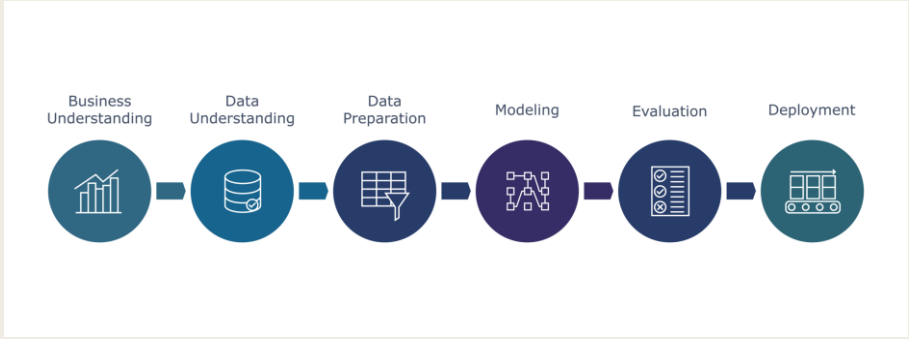
- The objective of this project is to develop a machine learning model to accurately identify crime hotspots in Los Angeles.

- By harnessing LAPD crime data, the model will provide actionable insights, enabling the development of targeted security recommendations in high-risk areas and contribute to a safer environment.

Methodology

- Our methodology encompasses a systematic approach to data processing, feature engineering, model training, evaluation, and visualization.

- This structured process ensures the accuracy and reliability of our crime hotspot detection model.



Data Collection

- I used LAPD crime data for this project. The dataset includes detailed information on various crimes reported in Los Angeles from 2020 to 2024.
- The collected data should include relevant attributes such as the type of crime, Area, date, and time, which are essential for identifying and analyzing crime hotspots.

Data Preprocessing

- Data preprocessing is a crucial step in the data mining process that transforms raw data into a format that can be easily and effectively analyzed.

- This stage ensures that the data is accurate, consistent, and ready for analysis, which is essential for building reliable and effective predictive models.

Exploratory Data Analysis (EDA)

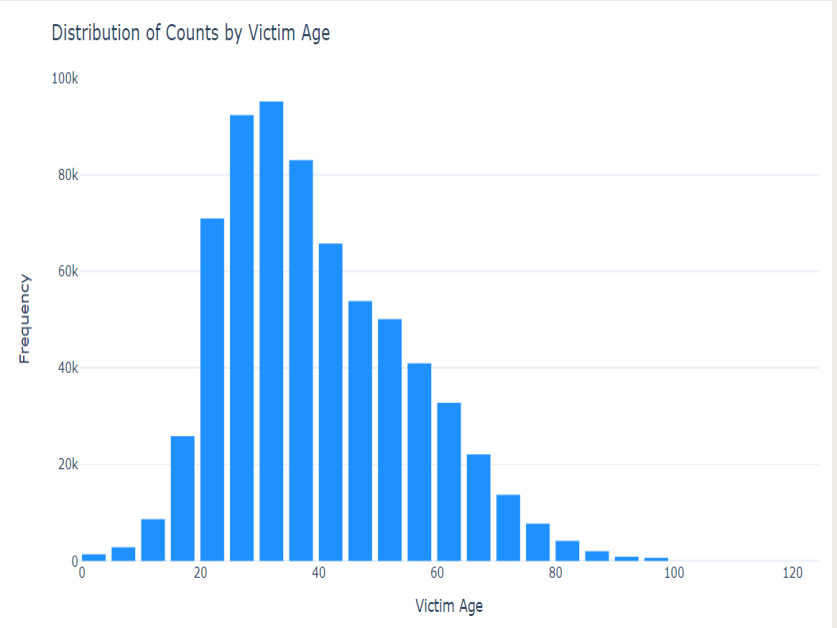
- Exploratory Data Analysis (EDA) is a critical step in the data science process, used to understand the underlying structure, patterns, and relationships within the dataset.

- EDA is essential for uncovering insights that can inform the development of effective crime prevention strategies.

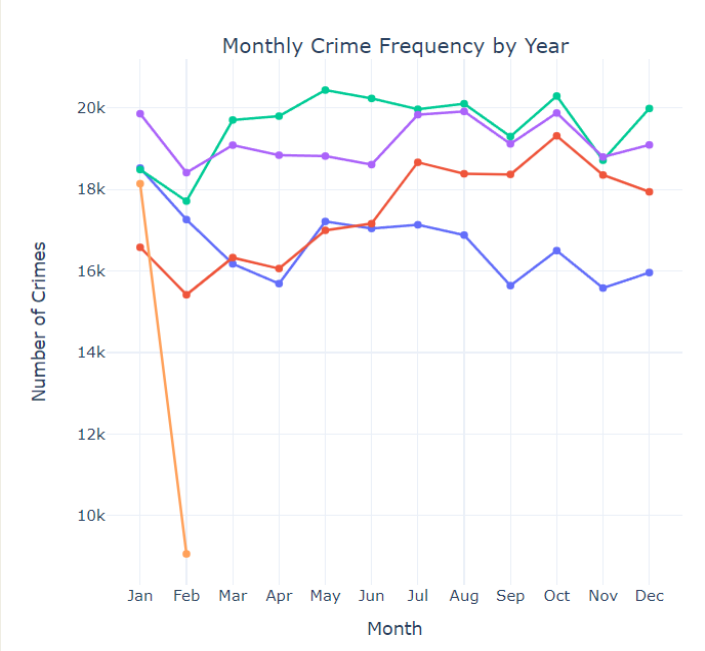
Distribution of counts by Victim Age

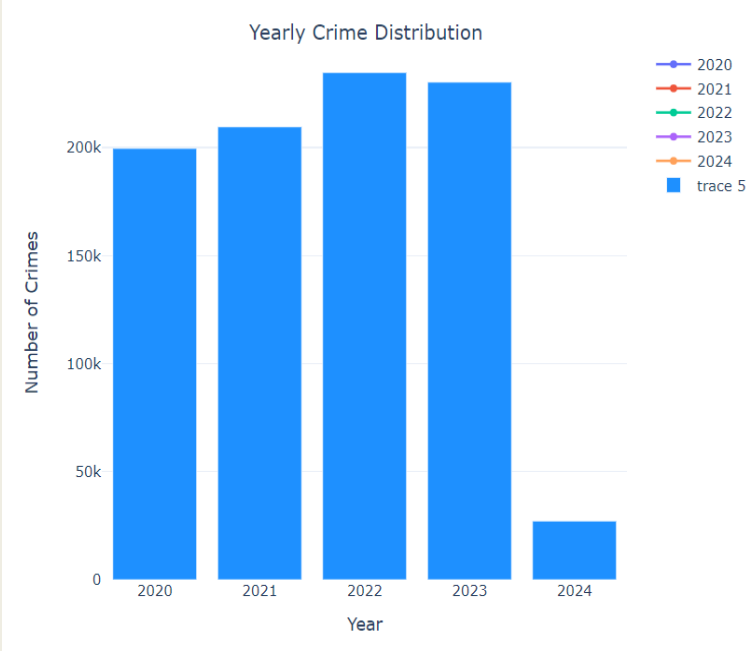
- The histogram presented showcases the distribution of crime victims by age, which is a crucial aspect of our exploratory data analysis (EDA).

- This insights gives the Victim Age are fall under 30 to 34 ages are mostly affected by the crime.



Distribution of Crimes done by Month and Year



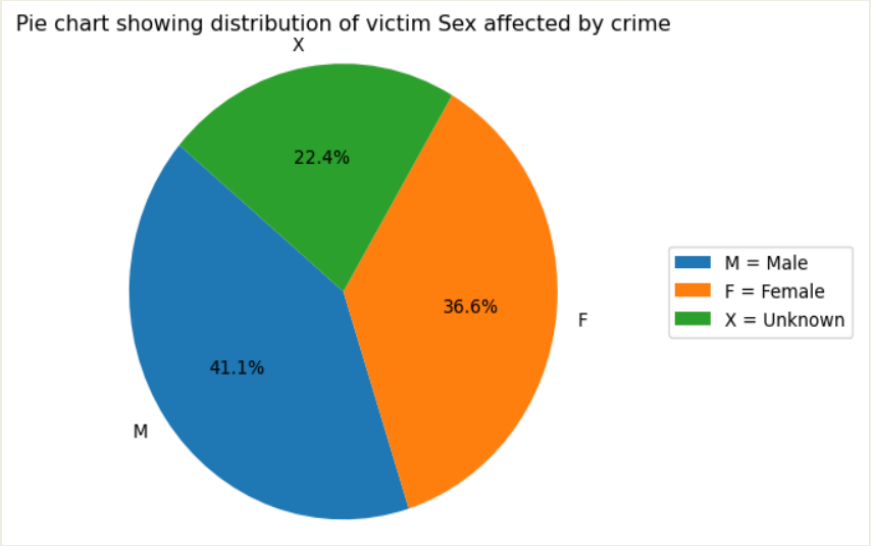


- This insights gives us the most of the crimes are done in the Month of May .
- And the year 2022 which is mostly affected by the crimes compared to all the years.

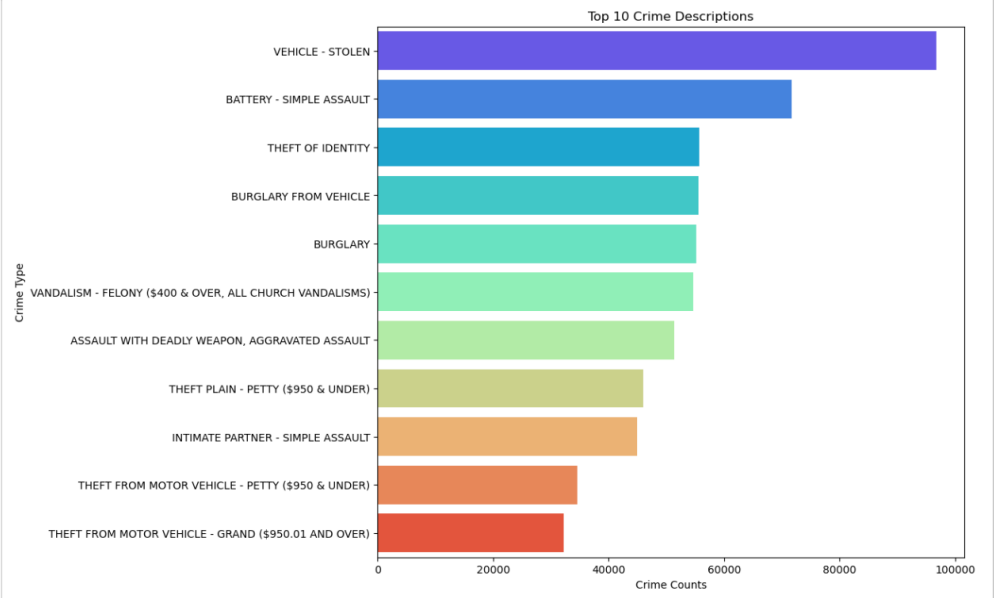
Distribution of Victim Sex affected by crime

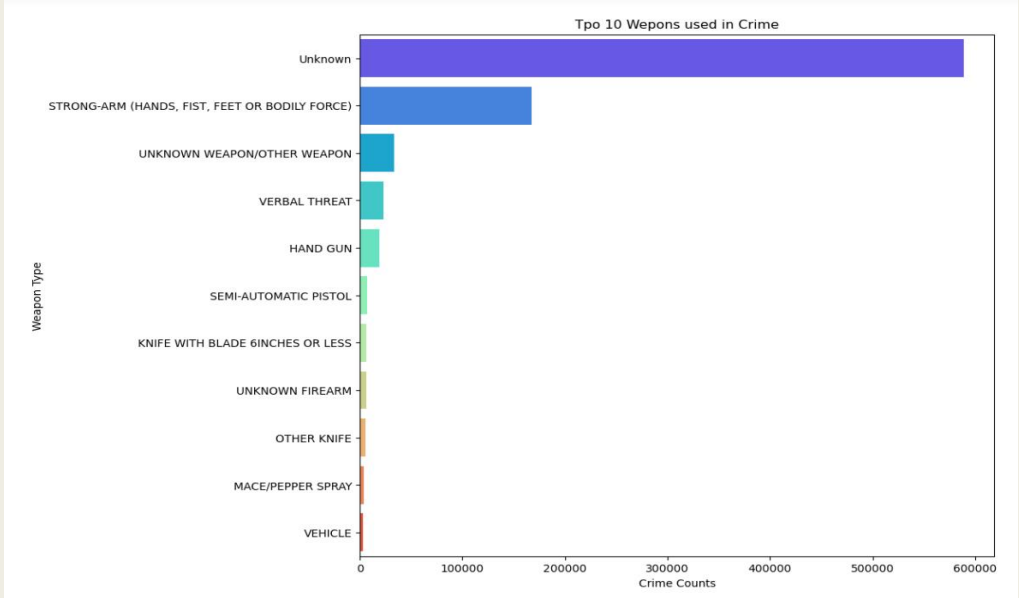
- Pie charts are ideal for comparing the proportions of different categories within a dataset. For instance, if we want to understand the distribution of Victim Sex, a pie chart can visually display which Victim Sex affected by the crimes is most prevalent.

- From this insights we know the most of the Male Victims are affected by Crime



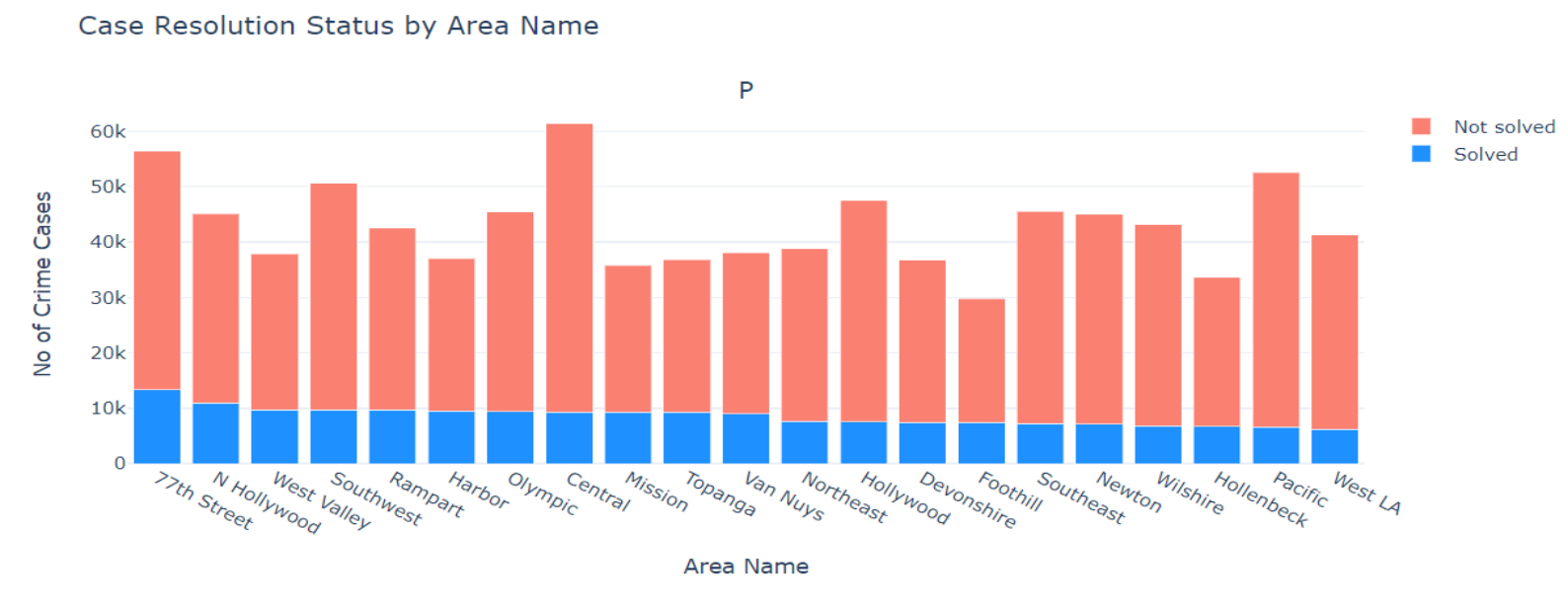
Distribution of Top 10 Crime Descriptions and Weapon used in crime





- From this Insights we know the Vehicle – Stolen crimes are mostly affected from the top 10 crime description
- And we also know that the unknown weapons are used mostly used from the top 10 weapons used in crime.

Distribution of Case Resolution by Area Names

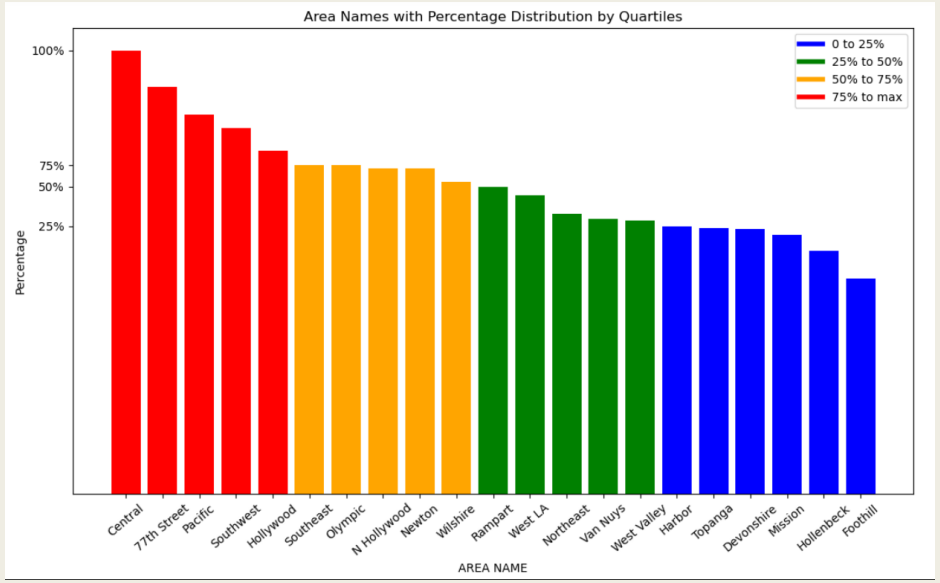


- This Insights gives us the Solved Cases and Unsolved Cases are distributed by Area Names.
- From this visualization we know that the most of the cases are Not Solved in Central Area and the second most cases are Not Solved in 77th street.

Distribution of Area Names by Percentile

- In a bar plot, categories are typically displayed along the x-axis, and the corresponding values are shown on the y-axis. Bars can be oriented either vertically or horizontally.

- From this Insights we know the distribution of Area Names by Percentiles which is fall in 0% Percentile to 100% Percentile and we considered the 0% to 50% as Low Risk and greater then 50% to 100% as High Risk.



Feature Selection

- My Dataset has 9 Lakhs records which cause computational resources and time conception and may cause data leakage for my prediction so I take a sample records for my model prediction.

Taking Sample data for Predection

```
# Sample 300,000 rows from the DataFrame
sampled_df = features.sample(n=300000, random_state=42)
```

```
sampled_df.shape
```

```
(300000, 7)
```

```
X = sampled_df[['AREA', 'Crm Cd', 'Vict Sex', 'Vict Descent', 'Weapon Desc', 'case_solved']]
y = sampled_df['Risk']
```

Scaling the Data

- Scaling is a crucial step in data preprocessing, particularly when preparing data for machine learning models.
- We are scaling all the columns to get a same range.

Scaling the Dataset

```
# import StandarScaler from sklearn
from sklearn.preprocessing import StandardScaler

# initialize the standard scalar
Scale = StandardScaler()

# scale all the numeric variables
# standardize all the columns of the dataframe 'X_Scaled'
X_Scaled = Scale.fit_transform(X)

# create a dataframe of scaled numerical variables
# pass the required column names to the parameter 'columns'
X = pd.DataFrame(X_Scaled, columns = X.columns)
```

Train Test Split

- In this step, We partitioned the dataset into two components : X and y
- The variable X takes all the independent variables, representing the features that contribute to our predictions.
- On the other hand, y takes the dependent variable, serving as the outcome we aim to predict.

Splting Data into Train and Test

```
# split data into train subset and test subset
# set 'random_state' to generate the same dataset each time you run the code
# 'test_size' returns the proportion of data to be included in the testing set
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 21, test_size = 0.2)

# check the dimensions of the train & test subset using 'shape'
# print dimension of train set
print('X_train', X_train.shape)
print('y_train', y_train.shape)

# print dimension of test set
print('X_test', X_test.shape)
print('y_test', y_test.shape)
```

```
X_train (240000, 6)
y_train (240000,)
X_test (60000, 6)
y_test (60000,)
```

Model Prediction

- Importing Algorithm and Training the model

Logistic Regression

```
model = LogisticRegression()
model.fit(X_train,y_train)

# train accuracy
X_train_prediction  = model.predict(X_train)
train_data_accuracy = accuracy_score(X_train_prediction,y_train)

print('Accuracy score of our train data:',train_data_accuracy)

# test accuracy
X_test_prediction  = model.predict(X_test)
train_data_accuracy = accuracy_score(X_train_prediction, y_train)
test_data_accuracy = accuracy_score(X_test_prediction,y_test)

print('Accuracy score of our test data:',test_data_accuracy)
```

Testing Accuracy

```
Accuracy score of our train data: 0.5661125
Accuracy score of our test data: 0.5678666666666666
```

Classification Report

```
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.53      0.33      0.41     27023
           1       0.58      0.76      0.66     32977

    accuracy                           0.57     60000
   macro avg       0.56      0.55      0.53     60000
weighted avg       0.56      0.57      0.55     60000
```

Model Prediction

- Importing Algorithm and Training the model

Random Forest Classifier

```
model_1 = RandomForestClassifier()
model_1.fit(X_train,y_train)

# train accuracy
X_train_prediction  = model_1.predict(X_train)
train_data_accuracy = accuracy_score(X_train_prediction,y_train)

print('Accuracy score of our training data:',train_data_accuracy)

# test accuracy
X_test_prediction  = model_1.predict(X_test)
train_data_accuracy = accuracy_score(X_train_prediction, y_train)
test_data_accuracy = accuracy_score(X_test_prediction,y_test)

print('Accuracy score of our test data:',test_data_accuracy)
```

Testing Accuracy

```
Accuracy score of our training data: 1.0
Accuracy score of our test data: 0.9995666666666667
```

Classification Report

```
Random Forest Classifier Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     27023
           1       1.00      1.00      1.00     32977

    accuracy                           1.00     60000
   macro avg       1.00      1.00      1.00     60000
weighted avg       1.00      1.00      1.00     60000
```

Model Prediction

- To overcome overfitting, I used a hyperparameter tuning technique called Grid Search CV with logistic regression.

- As a result, I achieved better accuracy in the classification report, indicating improved model performance.

Hyper Tuned Logistic Regression classification report

Accuracy: 0.66475

	precision	recall	f1-score	support
0	0.62	0.67	0.65	108605
1	0.71	0.67	0.69	131395
accuracy			0.67	240000
macro avg	0.67	0.67	0.67	240000
weighted avg	0.67	0.67	0.67	240000

	precision	recall	f1-score	support
0	0.62	0.67	0.64	27151
1	0.71	0.66	0.68	32849
accuracy			0.66	60000
macro avg	0.66	0.66	0.66	60000
weighted avg	0.67	0.66	0.67	60000

Conclusion

- In this project, I employed two different predictive models Logistic Regression and Random Forest Classifier to analyze and predict case outcomes.
- The choice of these models was strategic, considering their suitability and performance in classification tasks, especially in scenarios like ours with a mix of categorical and numerical data.
- After rigorous testing and tuning, Logistic Regression emerged as the best performing model for our classification task.
- Our final Logistic Regression model demonstrated robustness and reliability, achieving accuracy and balanced performance across precision, recall, and F1 scores. This makes it a suitable choice for our dataset, providing a model that can effectively handle class imbalance and deliver model predictions.

In conclusion, I want to extend my gratitude for your presence and attention throughout my presentation. Your feedback and support mean a lot to me.