INTELLIGENCE SURVEILLANCE CAMERA

Submitted By:

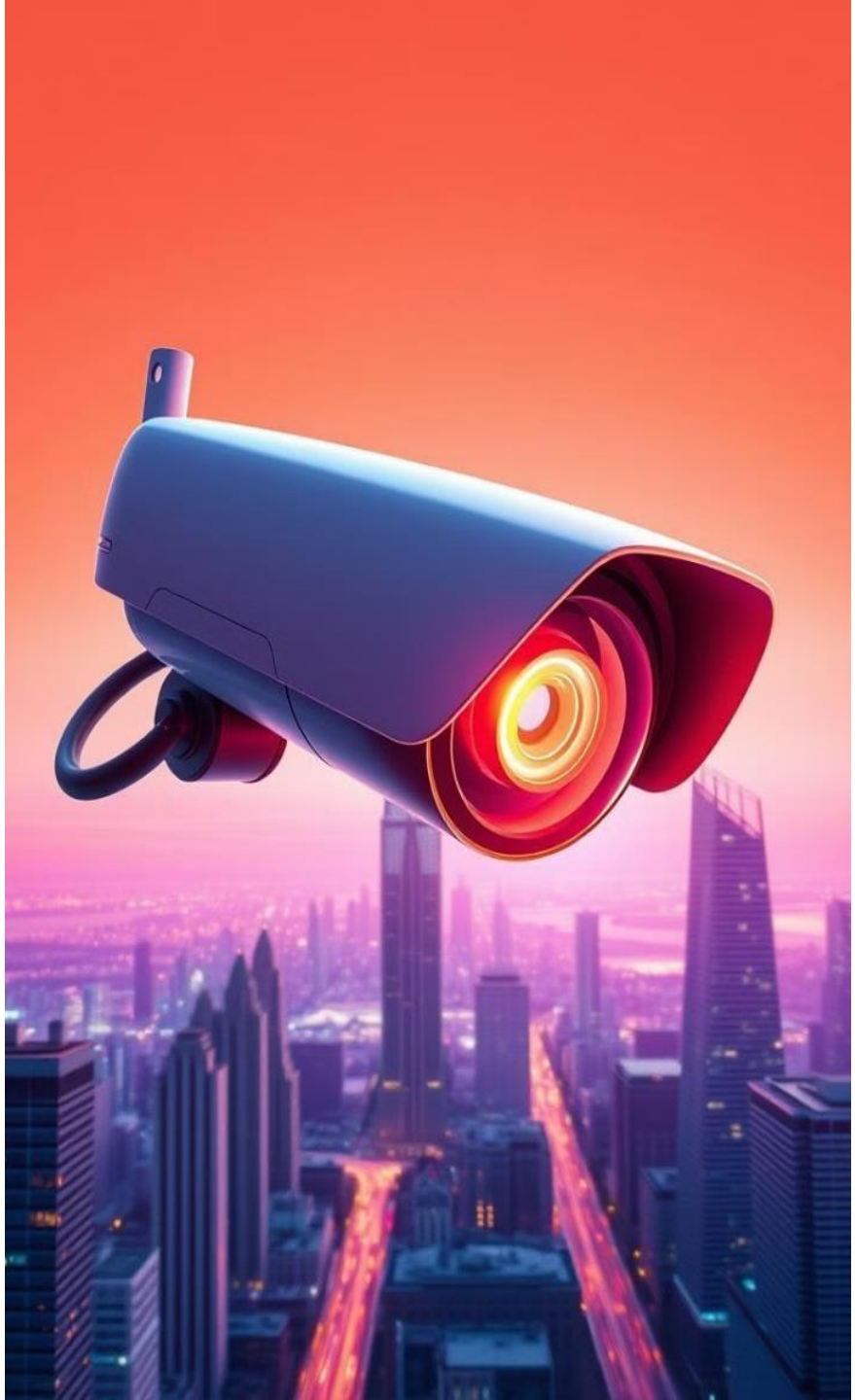
REGIDI. MADHAVI 23MH1A4453

Abstract

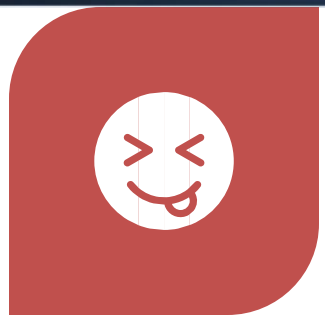
- The project presents the design and implementation of an intelligent surveillance camera system aimed at enhancing security and monitoring capabilities through real-times analysis
- The surveillance systems typically employed are CCTV-based monitoring systems which require an administrator to scan the footage or manual monitoring of the area to detect .

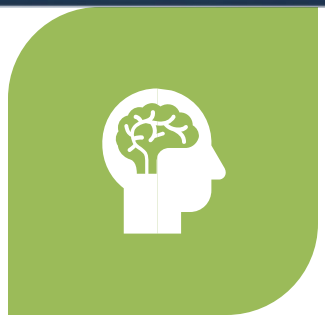
Introduction

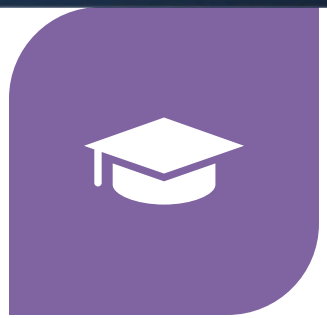
- The intelligent surveillance system uses advanced video analysis technologies, such as object detection, behavior recognition, face recognition, etc., to automatically identify and analyze targets in the surveillance scene.



Problem Statement







PROBLEM FACED IN AI BIAS MISIDENTIFICATION ?

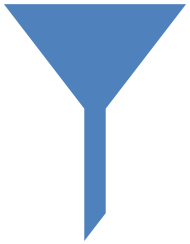
DATA SECURITY RISK AND CYBER SECURITY RISK?

DATA COLLECTION AND AND TRACKING?

System Requirements

- Hardware: 4GB RAM, 2GHz CPU
- Software: Windows/Linux, Python 3.x, Jupyter Notebook
- Libraries: pandas, numpy, matplotlib, seaborn.

DATA SOURCE



Dataset downloaded Fields include: Daily Activity Trends,Peak from Kaggle. hour analysis,high risk activity,location wise security

📦 *Data Source in the Notebook*

The data source is explicitly defined in the following line:

```
python
df = pd.read_csv('Intelligent_Surveillance_Dataset.csv')
```

✅ Details:

Attribute	Description
Format	CSV (Comma-Separated Values)
Filename	Intelligent_Surveillance_Dataset.csv
Access Method	Loaded using pandas.read_csv()
Location	Expected to be in the same directory as notebook
Tool Used	Python pandas library

📌 Summary:

* The dataset is likely a structured table with columns such as:

  * Timestamp
  * People_Count
  * Motion_Score
  * Alert_Flag
  * Activity_Type
  * Location

Architecture

1. DATA 2. DATA 3. DATA 4. QUERY COLLECTION CLEANING EXPLORATION ANALYSIS

5. VISUALIZATION

6. CONCLUSION & REPORTING.

Data Cleaning

Handled missing values & duplicates.

Verified data types and ensured proper data formatting for analysis.

1. *Date-Time Conversion*

```
python
df['Timestamp'] = pd.to_datetime(df['Timestamp'])
```

* Converts timestamp to a proper datetime format.

---

2. *Feature Engineering from Timestamp*

```
python
df['Hour'] = df['Timestamp'].dt.hour
df['Day'] = df['Timestamp'].dt.day
df['DayOfWeek'] = df['Timestamp'].dt.day_name()
df['Date'] = df['Timestamp'].dt.date
```

3. *Categorization and Mapping*

* *Day/Night Classification*

```
python
 df['Time_Period'] = df['Hour'].apply(categorize_time)
```

* *Weekend Identification*

```
python
 df['IsWeekend'] = df['DayOfWeek'].isin(['Saturday', 'Sunday'])
```

* *Crowd Size Labeling*

```
python
 df['Crowd_Size'] = df['People_Count'].apply(categorize_crowd)
```

* *Activity Type Mapping*

```
python
 df['Activity_Category'] = df['Activity_Type'].apply(lambda
x: ...)
```

- Peak Hours Analysis - When is the facility busiest?
- Location-wise Security Analysis
- High-Risk Activity Detection
- Temporal Pattern Analysis - Day vs Night Activity
- Crowding Events Analysis
- Motion Score Anomaly Detection
- Daily Activity Trends
- Alert Correlation Analysis
- Loitering Incidents Analysis

Sample Queries (1-10)

- Running Activity Security Assessment
- Weekend vs Weekday Patterns
- Low-Occupancy High-Alert Events
- Activity Type Transition Patterns
- Motion Score vs People Count Relationship
- Location Security Risk Ranking
- Time-based Alert Clustering
- Standing vs Moving Activity Analysis

- Temporal Alert Prediction Indicators

- COMPREHENSIVE SECURITY DASHBOARD SUMMARY

References

Kaggle Dataset

Python Libraries Documentation

Impact of intelligence surveillance camera on society.