



CCMS

**Street Light Centralized Control & Monitoring Systems
Data Analysis**

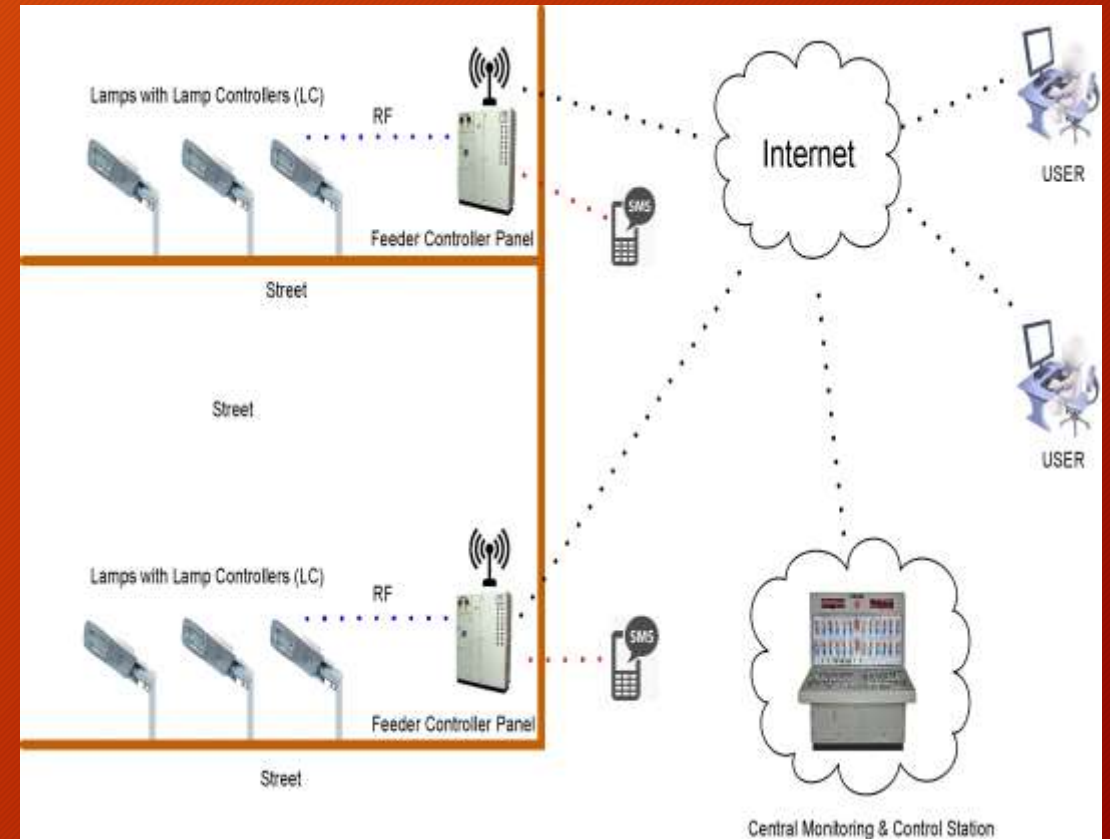
What it Does:-

1. Switch ON and OFF the lights of a particular switching point.
2. Control lights automatically throughout the year
3. Single Switch point can support up to 300 lighting poles
4. GPRS based remote streetlight with self protection from short-circuit
5. Over voltage protection and anti- theft alert.
6. Battery backup of 4 hours.
7. Metal/Polycarbonate/SME enclosure with proper lock arrangement.

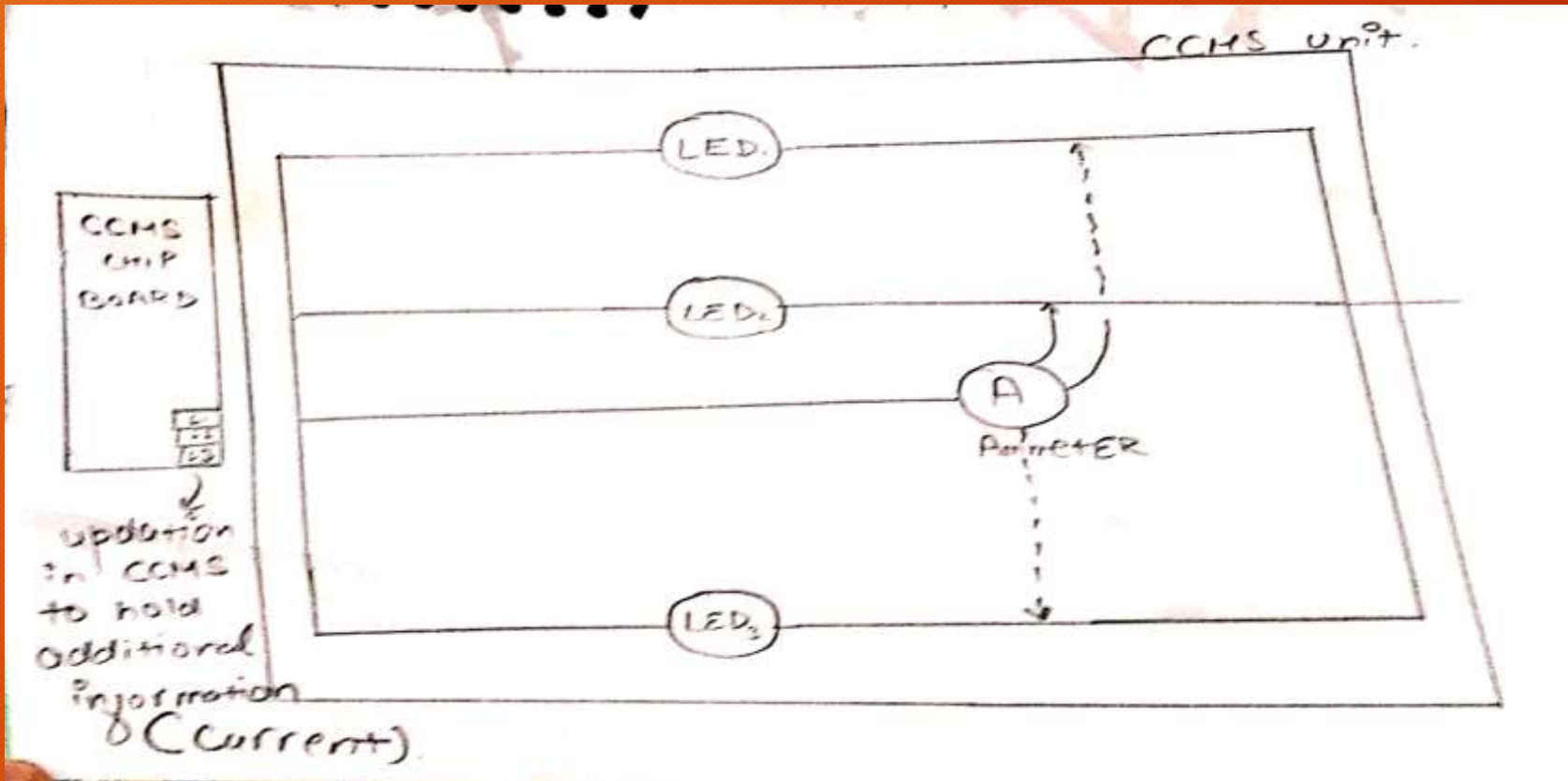


Workflow of CCMS

1. CCMS has a web-server to receive and record all data from the street
2. Displays the power failure details of a particular switching point. Registers all fault conditions like excess
3. Registers all fault conditions like excess
4. It shows the accumulated power consumption of all the switch points
5. It is the real time total power consumption of all the street lights connected to switch points based on selection criteria like State/District/City.



architecture to endorse our solutions:-



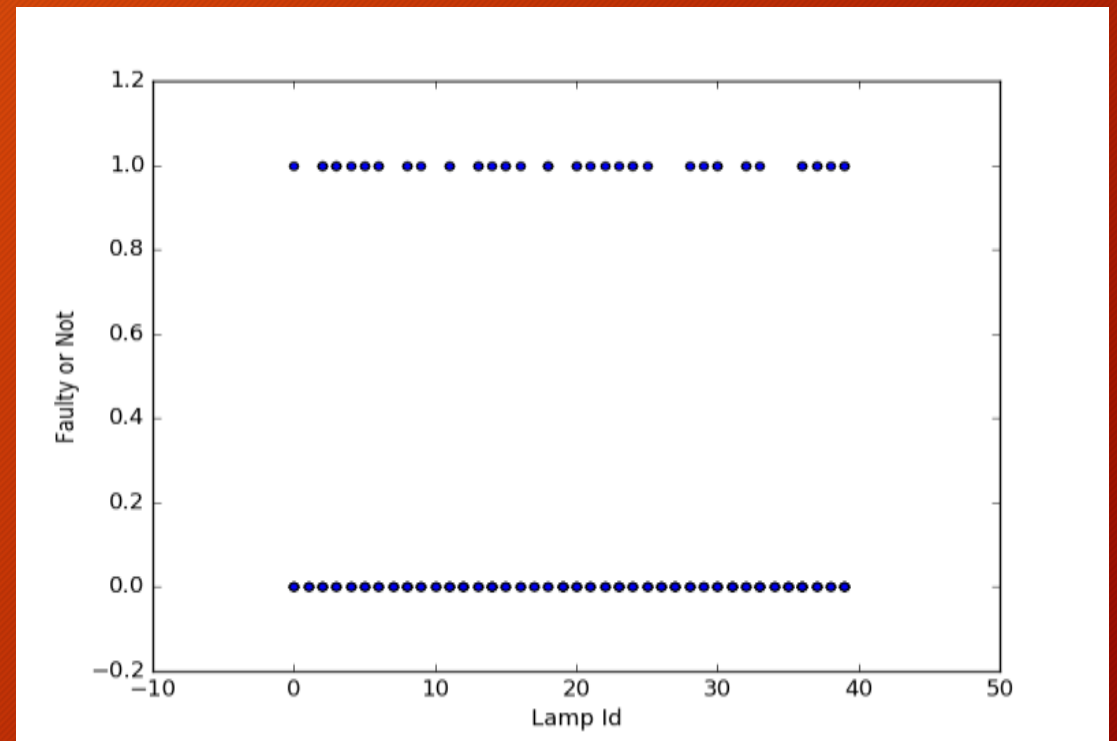
Solution to Problem Statement #1

- We have tried to come up with a solution by using hashing to identify different LED lamps and then the data has been cleansed and thoroughly visualized to see various correlations between the different attributes in the datasets.



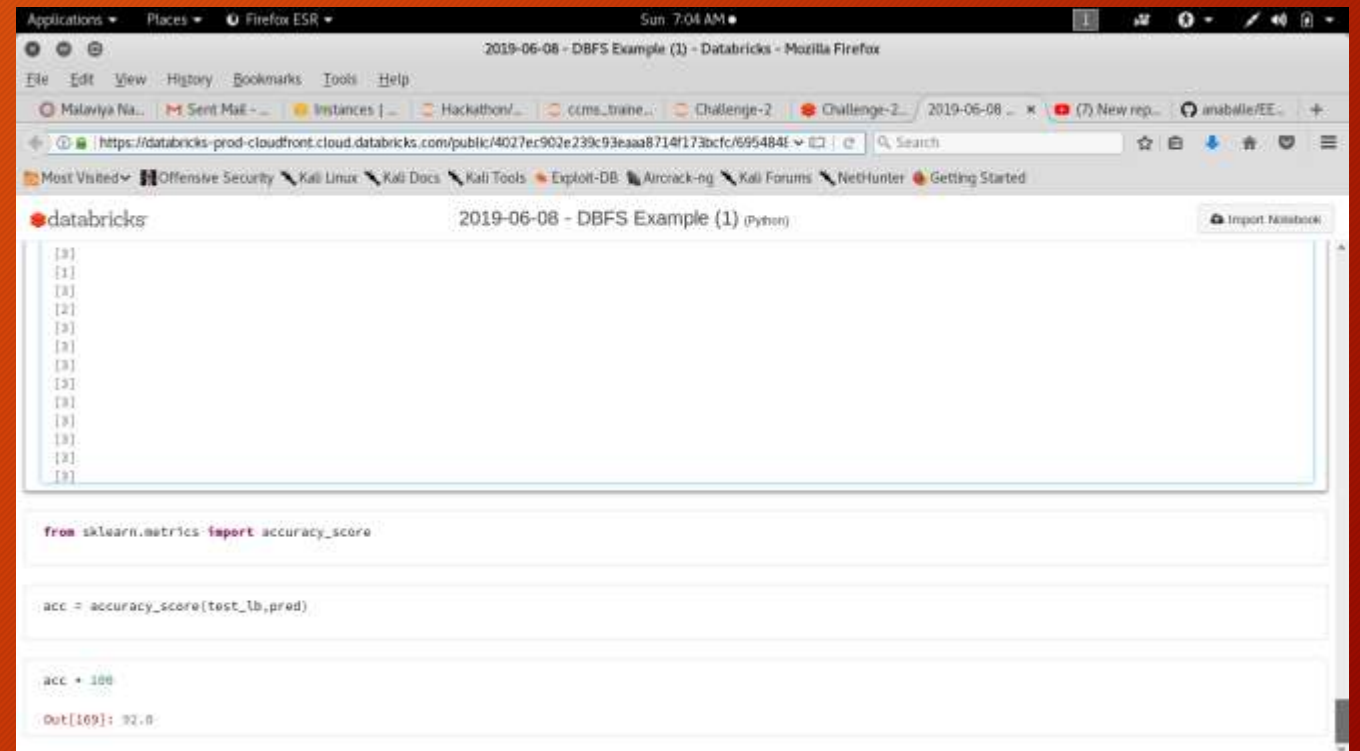
Problem Statement #1 prototype

- Then we have used Decision Tree to predict the faulty LED lamp by building a model using Sci-kit.



Solution to Problem Statement #2

- Here we have build again a model using the famous Support Vector Machine(SVM) algorithm. Different attributes have been evaluated based on their correlation factors and then a model with an accuracy of 92% has been achieved.

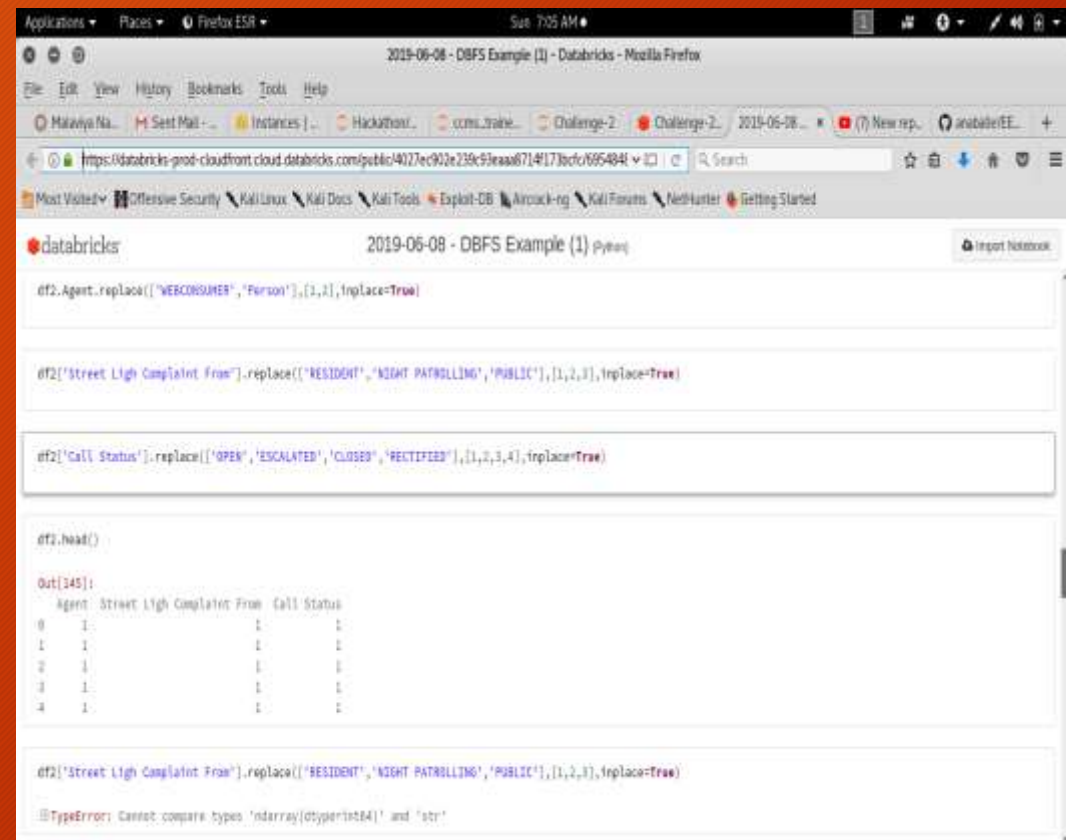


The screenshot shows a web browser window displaying a Databricks notebook. The browser's address bar shows a URL from databricks.com. The notebook interface includes a header with the title '2019-06-08 - DBFS Example (1) (Python)' and an 'Import Notebook' button. The main area contains a list of 15 numbers: [9], [1], [8], [2], [9], [9], [9], [9], [9], [9], [9], [9], [9], [9], [9]. Below this list, there is a Python code snippet that imports 'accuracy_score' from 'sklearn.metrics', calculates the accuracy for a test set, and prints the result. The output of the code is 'Out[169]: 0.92'.

```
[9]  
[1]  
[8]  
[2]  
[9]  
[9]  
[9]  
[9]  
[9]  
[9]  
[9]  
[9]  
[9]  
[9]  
[9]  
  
from sklearn.metrics import accuracy_score  
  
acc = accuracy_score(test_lb, pred)  
  
acc * 100  
  
Out[169]: 92.0
```

Problem Statement #2 prototype

- A permanent link to the notebook can be found at <https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaa8714f173bcfc/6954848893521581/1283534314062248/2558959897262863/latest.html>



The screenshot shows a Databricks notebook interface with the title "2019-06-08 - DBFS Example (1)". The code cell contains three PySpark replace operations on a DataFrame 'df2' and a head() call. The output shows a DataFrame with 5 rows and 4 columns: Agent, Street Light Complaint From, and Call Status. The first three rows have values, and the fourth row is empty. Below the output, there is an error message: "TypeError: Cannot compare types 'ndarray(dtype=int84)' and 'str'".

```
df2.Agent.replace(['WEBCONSUMER', 'Person'], [1, 2], inplace=True)

df2['Street Light Complaint From'].replace(['RESIDENT', 'NIGHT PATROLLING', 'PUBLIC'], [1, 2, 3], inplace=True)

df2['Call Status'].replace(['OPEN', 'ESCALATED', 'CLOSED', 'RECTIFIED'], [1, 2, 3, 4], inplace=True)

df2.head()
```

	Agent	Street Light Complaint From	Call Status
0	1	1	1
1	1	1	1
2	1	1	1
3	1	1	1
4	1		

```
df2['Street Light Complaint From'].replace(['RESIDENT', 'NIGHT PATROLLING', 'PUBLIC'], [1, 2, 3], inplace=True)

TypeError: Cannot compare types 'ndarray(dtype=int84)' and 'str'
```


Some Problems and Effective solutions:-

- Group the different problems based on there states and try to find core relation between status of complain and the agent.
- Use this agent to successfully monitor no. of complain coming from your locality.
- Find co-relation between different attributes and also solve problems on bases of time duration.

• •

- Since the state of complain are categories in 4 categories.
OPEN | CLOSE | ESCULATED | RECTIFIED
- We will categories problems based on LIMITS ,DURATION ,TYPE co-relation with there states.
- These can be used fundamentally to dose find an opinion.
- Solution based on some raw data provides to us weather we can just find the possibility to problem accuracy.

REFERENCE

- <http://scikit-learn.org/stable/documentation.html>
- <https://www.incubateind.com/inspirehackathon/>
- <https://docs.databricks.com/>
- <https://stackoverflow.com/>

