	Name of State / UT	Total Confirmed cases	Cured/Discharged/Migrated	Death
0	Maharashtra	1985	217	149
1	Delhi	1154	27	24
2	Tamil Nadu	1043	50	11
3	Rajasthan	804	21	3
4	Madhya Pradesh	564	0	36
5	Gujarat	516	44	25
6	Telengana	504	43	9
7	Uttar Pradesh	483	46	5
8	Andhra Pradesh	427	11	7
9	Kerala	376	179	2
10	Jammu and Kashmir	245	6	4
11	Karnataka	232	57	6
12	Haryana	185	29	3
13	West Bengal	152	29	7
14	Punjab	151	5	11
15	Bihar	64	19	1
16	Odisha	54	12	1
17	Uttarakhand	35	5	0
18	Himachal Pradesh	32	13	1
19	Chhattisgarh	31	10	0
20	Assam	29	0	1
21	Chandigarh	21	7	0
22	Jharkhand	19	0	2
23	Ladakh	15	10	0
24	Andaman and Nicobar Islands	11	10	0
25	Puducherry	7	1	0
26	Goa	7	5	0
27	Manipur	2	1	0
28	Tripura	2	0	0
29	Arunachal Pradesh	1	0	0
30	Mizoram	1	0	0

Active Cases
1619
1103
982
780
528
447
452
432
409
195
235
169
153
116
135
44
41
30
18
21
28
14
17
5
1
6
2
1
2
1
1

major-project

May 19, 2023

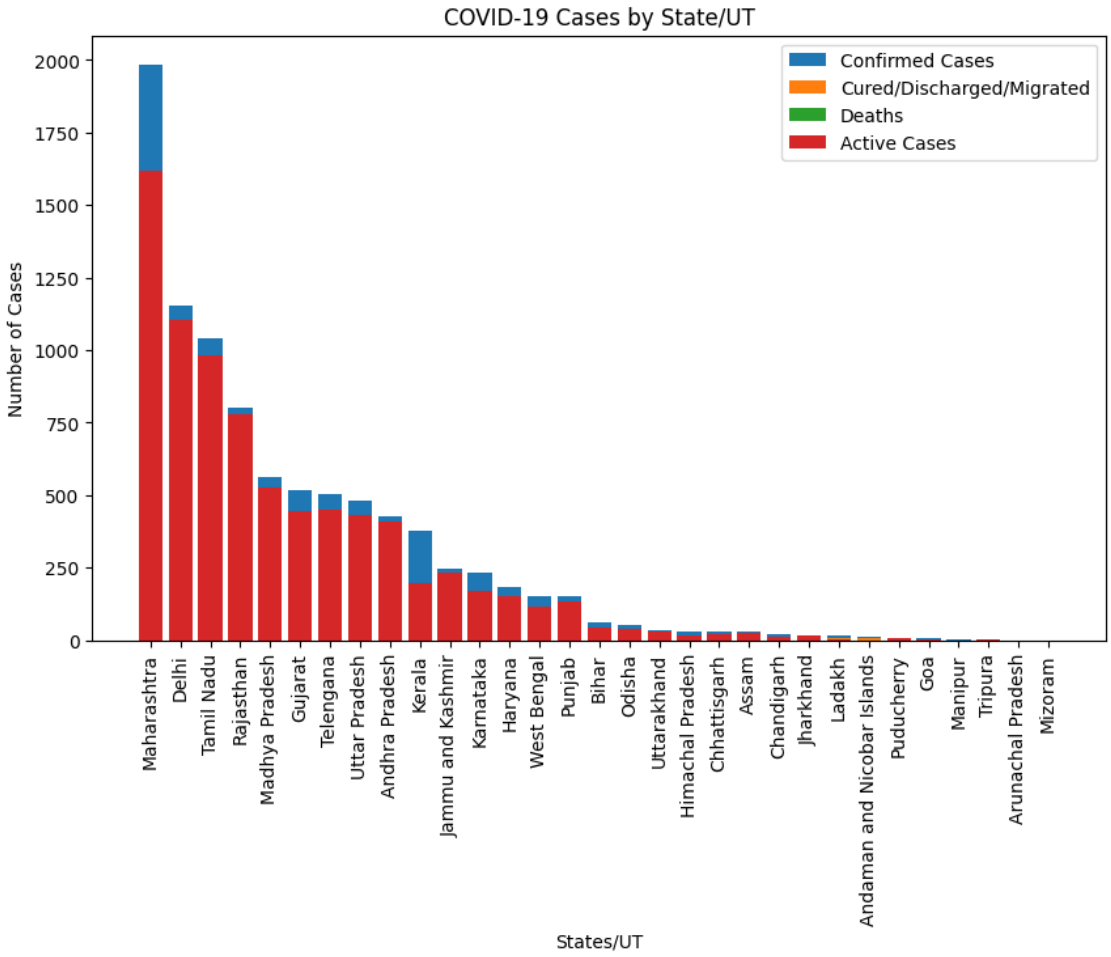
[19]:
```
import pandas as pd

# Load the dataset
data = pd.read_csv('untitled.csv')

# Display the first few rows
print(data.head())
```

```
   Unnamed: 0 Name of State / UT  Total Confirmed cases  \
0           0        Maharashtra                   1985
1           1              Delhi                   1154
2           2         Tamil Nadu                   1043
3           3          Rajasthan                    804
4           4     Madhya Pradesh                    564

   Cured/Discharged/Migrated  Death  Active Cases
0                        217    149          1619
1                         27     24          1103
2                         50     11           982
3                         21      3           780
4                          0     36           528
```

[20]:
```
import matplotlib.pyplot as plt

# Calculate basic statistics
total_cases = data['Total Confirmed cases'].sum()
total_cured = data['Cured/Discharged/Migrated'].sum()
total_deaths = data['Death'].sum()
total_active_cases = data['Active Cases'].sum()

print("Total cases:", total_cases)
print("Total cured:", total_cured)
print("Total deaths:", total_deaths)
print("Total active cases:", total_active_cases)

# Plotting the data
states = data['Name of State / UT']
confirmed_cases = data['Total Confirmed cases']
```

```
cured_cases = data['Cured/Discharged/Migrated']
death_cases = data['Death']
active_cases = data['Active Cases']

plt.figure(figsize=(10, 6))
plt.bar(states, confirmed_cases, label='Confirmed Cases')
plt.bar(states, cured_cases, label='Cured/Discharged/Migrated')
plt.bar(states, death_cases, label='Deaths')
plt.bar(states, active_cases, label='Active Cases')
plt.xticks(rotation=90)
plt.xlabel('States/UT')
plt.ylabel('Number of Cases')
plt.title('COVID-19 Cases by State/UT')
plt.legend()
plt.show()
```

```
Total cases: 9152
Total cured: 857
Total deaths: 308
Total active cases: 7987
```



[25]:
```
import pandas as pd
from sklearn.datasets import make_classification
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

# Select features and target variable
features = ['Total Confirmed cases', 'Death', 'Active Cases']
target = 'Cured/Discharged/Migrated'

X, y = make_classification(random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)

# Initialize the logistic regression model
model = LogisticRegression()

# Train the model
model.fit(X_train, y_train)

# Predict the probabilities of recovery for the test set
y_pred_prob = model.predict_proba(X_test)[:, 1]  # Probability of positive class

# Calculate the accuracy of the model
accuracy = model.score(X_test, y_test)

# Print the accuracy
print(f"Accuracy: {accuracy}")

# Print the predicted probabilities of recovery for the test set
print("Predicted probabilities of recovery:")
for prob in y_pred_prob:
    print(prob)
```

```
Accuracy: 1.0
Predicted probabilities of recovery:
0.15633565159845186
0.998809340778949
0.9786218423371947
0.9661750797254952
5.311210303723623e-05
0.9763647476939868
0.022664928848980855
0.6371190733858195
0.929103634995325
0.4998136067125522
0.4024385778287669
```

0.03975353855548362
0.049673870592165194
0.007566823969582232
0.0011873412036236047
0.19400308204256408
0.9701228765246487
0.07663752929134009
0.020247025867611377
0.00504990494454088
0.04574151421889641
0.8543869451717667
0.9878043796107852
0.8583075735991705
0.006239347683844166