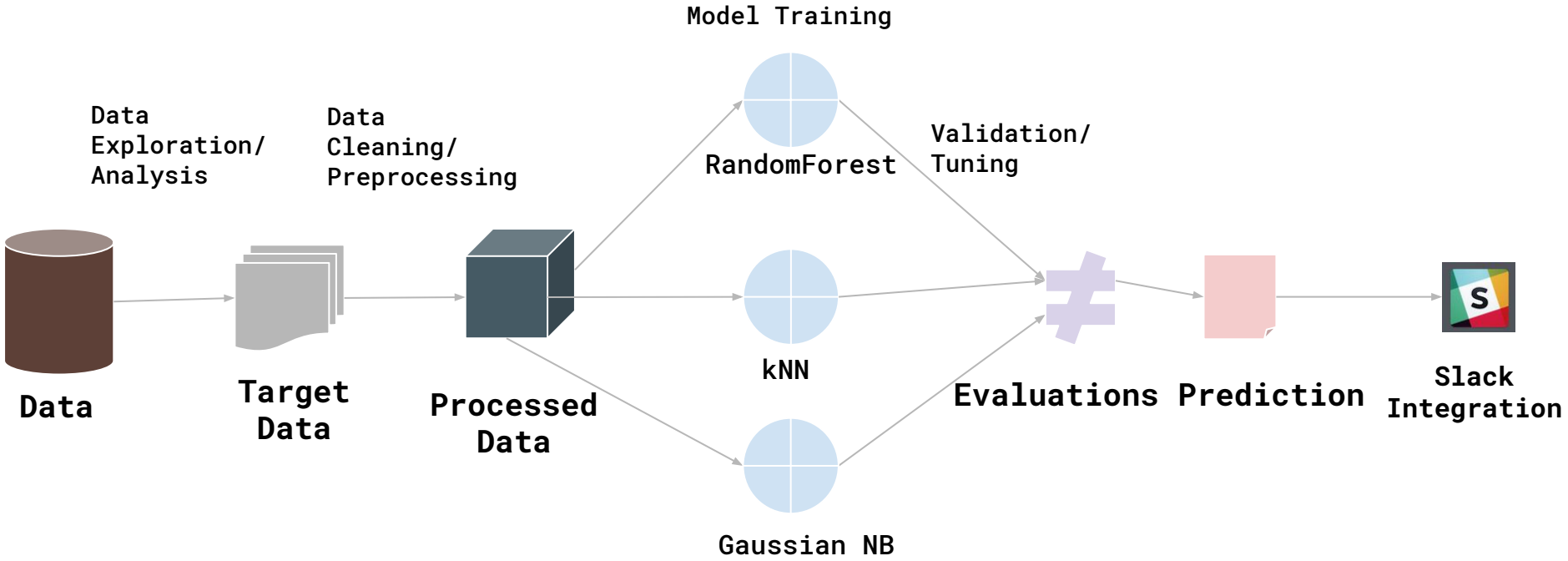
Crime Vigilance - A Data Mining Approach

Team: Vigilante
Aashish Subramanian
Chintan Vachhani
Sudheer Sammeta

Introduction

- Crime prevention and resolution are major issues faced by governments and law enforcement agencies around the world
- Vigilant citizens can save themselves and others if they are better informed
- Our idea is to use Data Mining to provide critical insight with simple parameters to gauge the threat level at a particular location and a particular time
- This would help the not just individuals, but law enforcement to maintain law and order

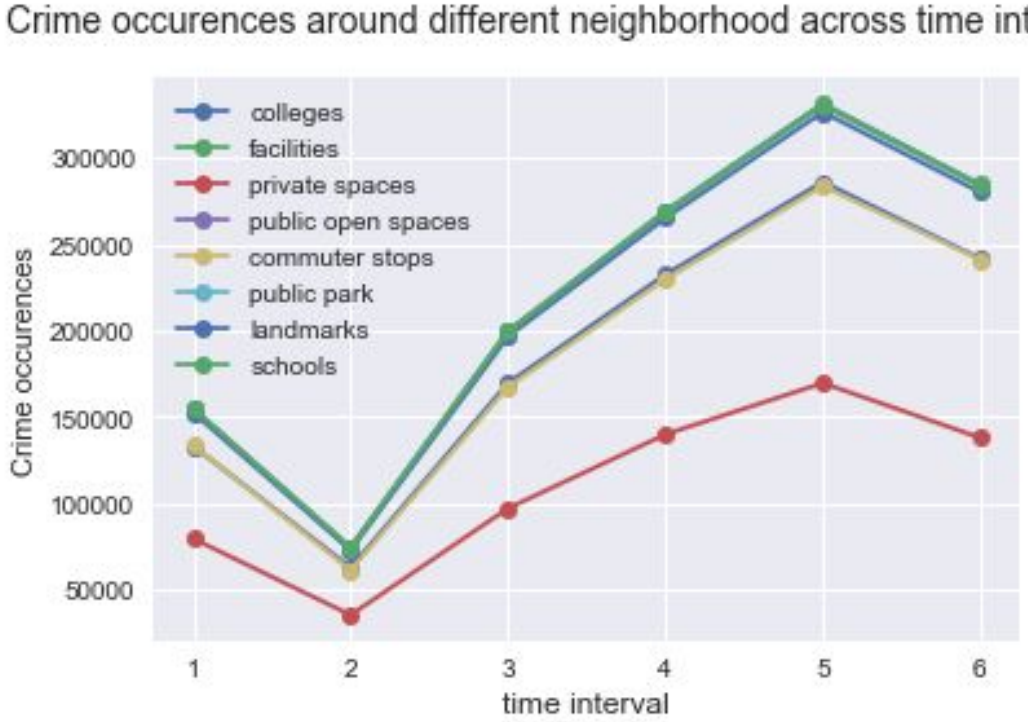
Process


Model Training
Data Exploration/ Analysis
Data Cleaning/ Preprocessing
RandomForest
Validation/ Tuning
Data
Target Data
Processed Data
kNN
Evaluations
Prediction
Slack Integration
Gaussian NB

Dataset/Preprocessing

- A variety infrastructure related data was collected for San Francisco
- Colleges and schools, Facilities and landmarks in the city, Commuter shuttle stops, Public and private open spaces, Crime data with category of crimes
- Primary features of interest - location and name
- Final dataset consists of 2.1M rows
- Preprocessing steps included elimination of NaN, modification of data types, binarization, label encoding, feature decomposition, feature integration

Analysis



and many more..

Model

- Data was split into training and testing using StratifiedKFold strategy.
- Fit the data using
- KNN Classifier
- Gaussian Naive Bayes
- Random Forest algorithms

Tuning/Evaluation

Gridsearch [Tuning] and F1-Score [Evaluation]

```
[011447696@c4 vigilante]$ python hpc/randomForest_model_training.py

Tuning the model.
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='sqrt', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=-1,
            oob_score=False, random_state=0, verbose=0, warm_start=False)

Evaluating the model.
             precision    recall  f1-score   support

       high       0.78      0.86      0.82    262807
        low       0.52      0.45      0.48     89111
   moderate       0.29      0.13      0.18     27233

avg / total       0.68      0.71      0.69    379151
```

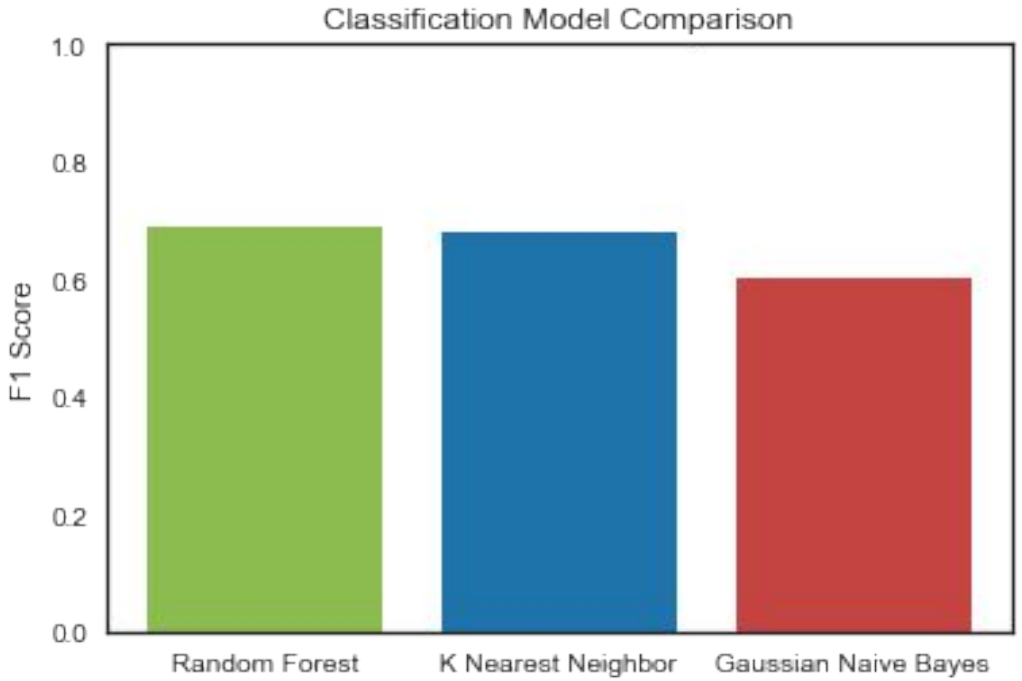
Tuning/Evaluation

```
[011447696@c1 vigilante]$ python hpc/kNN_model_training.py

Tuning the model.
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
           metric_params=None, n_jobs=-1, n_neighbors=100, p=2,
           weights='distance')

Evaluating the model.
             precision    recall  f1-score   support

       high       0.77      0.88      0.82    262807
        low       0.52      0.39      0.45     89111
   moderate       0.30      0.11      0.17     27233

avg / total       0.67      0.71      0.68    379151
```

Tuning/Evaluation

```
[0114476960c2 vigilante]$ python hpc/gaussianNB_model_training.py

Tuning the model.
GaussianNB(priors=None)

Evaluating the model.
             precision    recall  f1-score   support

       high       0.70      0.95      0.81    262807
        low       0.49      0.09      0.16     89111
   moderate       0.08      0.02      0.03     27233

avg / total       0.61      0.68      0.60    379151
```

Results


Classification Model Comparison
F1 Score
1.0
0.8
0.6
0.4
0.2
0.0
Random Forest
K Nearest Neighbor
Gaussian Naive Bayes

Conclusion

- Given the context of the problem, some of the data sets collected proved to be ineffective
- Other datasets needed to be cleaned and transformed to be useful
- Evaluation metrics were identified after data analysis
- Three algorithms were used to develop the model and RandomForest achieved the best among them

DEMO

Q & A

THANK YOU!