In [ ]:
```
# imports
import os
import pandas as pd
import numpy as np

# load data
train = pd.read_csv('../data/processed/train_data_processed.csv')
test = pd.read_csv('../data/processed/test_data_processed.csv')
val = pd.read_csv('../data/processed/val_data_processed.csv')
```

In [ ]:
```
# more feature engineering
# use encoder to encode OCCURRED_ON_DATE column
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
train['OCCURRED_ON_DATE'] = le.fit_transform(train['OCCURRED_ON_DATE'])
test['OCCURRED_ON_DATE'] = le.transform(test['OCCURRED_ON_DATE'])
val['OCCURRED_ON_DATE'] = le.transform(val['OCCURRED_ON_DATE'])
```

In [ ]:
```
train.head()
```

Out[ ]:

	OFFENSE_CODE	OFFENSE_DESCRIPTION	DISTRICT	OCCURRED_ON_DATE	MON
0	520	15	6	247	
1	3821	69	9	316	
2	3114	6	9	253	
3	3801	70	8	244	
4	3502	62	11	116	

In [ ]:
```
# save le
import joblib
joblib.dump(le, '../models/datetime_encoder.pkl')
```

Out[ ]:
```
['../models/datetime_encoder.pkl']
```

In [ ]:
```
test.head()
```

Out[ ]:

	_id	OFFENSE_CODE	OFFENSE_DESCRIPTION	DISTRICT	OCCURRED_ON_DATE
0	20848	801	6	0	0
1	20849	3018	100	0	0
2	20851	801	6	0	0
3	20852	3410	105	5	0
4	20854	724	7	9	1

In [ ]:
```python
#drop _id column

test = test.drop('_id', axis=1)
val = val.drop('_id', axis=1)
```

In [ ]:
```python
# use random forest to predict the target variable
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

# define the target variable
y_train = train['Severe_crimes']
y_test = test['Severe_crimes']
y_val = val['Severe_crimes']

# define the features
X_train = train.drop(['Severe_crimes'], axis=1)
X_test = test.drop(['Severe_crimes'], axis=1)
X_val = val.drop(['Severe_crimes'], axis=1)


# define the number of trees
n_estimators = [200, 500, 1000, 1500, 2000]

# fit the model with the different number of trees
for n in n_estimators:
    rf = RandomForestClassifier(n_estimators=n, random_state=42)
    rf.fit(X_train, y_train)
    print(f'Number of trees: {n}')
    print(f'Train accuracy: {rf.score(X_train, y_train)}')
    print(f'Test accuracy: {rf.score(X_test, y_test)}')
    print(f'Validation accuracy: {rf.score(X_val, y_val)}')
    print('-------------------------------------')
```

```
Number of trees: 200
Train accuracy: 0.9998541139856059
Test accuracy: 0.9952540480178671
Validation accuracy: 0.9939883645765999
-------------------------------------
Number of trees: 500
Train accuracy: 0.9998541139856059
Test accuracy: 0.9955332216638749
Validation accuracy: 0.9940530058177117
-------------------------------------
Number of trees: 1000
Train accuracy: 0.9998541139856059
Test accuracy: 0.9952540480178671
Validation accuracy: 0.9940530058177117
-------------------------------------
Number of trees: 1500
Train accuracy: 0.9998541139856059
Test accuracy: 0.9955332216638749
Validation accuracy: 0.9940530058177117
-------------------------------------
Number of trees: 2000
Train accuracy: 0.9998541139856059
Test accuracy: 0.9955332216638749
Validation accuracy: 0.9940530058177117
-------------------------------------
```

In [ ]:
```
# use 500 as the number of trees and depth of 10
rf = RandomForestClassifier(n_estimators=500, max_depth=10, random_state=42)
rf.fit(X_train, y_train)
```

Out[ ]:
```
RandomForestClassifier(max_depth=10, n_estimators=500, random_state=42)
```

In [ ]:
```
# evaluate the model
from sklearn.metrics import accuracy_score

# save the model
joblib.dump(rf, '../models/rf_model1_week6test.pkl')
```

Out[ ]:
```
['../models/rf_model1_week6test.pkl']
```

In [ ]:
```
# print accuracy
y_pred = rf.predict(X_test)
accuracy_score(y_test, y_pred)
print('Accuracy: ', accuracy_score(y_test, y_pred))
```

```
Accuracy:  0.9952540480178671
```

In [ ]:
```
# cross validation
cross_val_score(rf, X_train, y_train, cv=5, scoring='accuracy').mean()
print('Cross validation: ', cross_val_score(rf, X_train, y_train, cv=5, scoring=

# confusion matrix
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred)
print('Confusion matrix: ', confusion_matrix(y_test, y_pred))
```

```
Cross validation:  0.9940510888363372
Confusion matrix:  [[3367    0]
 [  17  198]]
```