In [36]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [37]:
```
df = pd.read_csv("last_two_years_accidents.csv")
```

In [38]:
```
df.columns
```

Out[38]:
```
Index(['ID', 'Source', 'Severity', 'Start_Time', 'End_Time', 'Start_Lat',
       'Start_Lng', 'Distance(mi)', 'Street', 'City', 'County', 'State',
       'Zipcode', 'Country', 'Timezone', 'Airport_Code', 'Weather_Timestamp',
       'Temperature(F)', 'Humidity(%)', 'Pressure(in)', 'Visibility(mi)',
       'Wind_Direction', 'Wind_Speed(mph)', 'Weather_Condition', 'Amenity',
       'Bump', 'Crossing', 'Give_Way', 'Junction', 'No_Exit', 'Railway',
       'Roundabout', 'Station', 'Stop', 'Traffic_Calming', 'Traffic_Signal',
       'Turning_Loop', 'Sunrise_Sunset', 'Civil_Twilight', 'Nautical_Twilight',
       'Astronomical_Twilight'],
      dtype='object')
```

In [39]:
```
severity_labels = {
    1: 'Not Severe',
    2: 'Not Severe',
    3: 'Severe',
    4: 'Severe'
}
```

In [40]:
```
df['Severity'] = df['Severity'].map(severity_labels)
```

In [41]:
```
size = df['Severity'].value_counts()['Severe']
```

In [42]:
```
df_balanced_severity = pd.DataFrame()
```

In [43]:
```
df_balanced_severity = df.groupby('Severity', group_keys = False).apply(lambda x: x.sample(size, random_state = 30))
```

In [44]:
```
df_balanced_severity['Severity'].value_counts()
```

Out[44]:
```
Severity
Not Severe    128292
Severe        128292
Name: count, dtype: int64
```

In [45]:
```
categorical_features = set(["Weather_Condition", "Civil_Twilight", 'Wind_Direction'])
```

In [46]:
```
for feature in categorical_features:
    df_balanced_severity[feature] = df_balanced_severity[feature].astype("category")
```

In [47]:
```
bool_columns = df_balanced_severity.select_dtypes(include='bool').columns
```

In [48]:
```
df_balanced_severity[bool_columns] = df_balanced_severity[bool_columns].replace({True:1, False:0})
```

In [49]:
```
df2= df_balanced_severity[['Start_Lat','Start_Lng','Distance(mi)', 'Temperature(F)', 'Humidity(%)', 'Pressure(in)',
           'Visibility(mi)', 'Wind_Speed(mph)','Amenity','Bump','Crossing','Give_Way',
           'Junction','No_Exit','Railway','Roundabout','Station','Stop','Traffic_Calming','Traffic_Signal',
           'Civil_Twilight','Weather_Condition','Civil_Twilight',
           'Wind_Direction','Severity']]
```

In [50]:
```
df2 = pd.get_dummies(df2, columns=list(categorical_features), drop_first=True)
```

In [51]:
```
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
```

In [52]:
```
Y = df2['Severity'] # target column
X = df2.drop(columns = ['Severity']) # features
```

In [53]:
```
X_train, X_test, y_train, y_test = train_test_split(X, Y,test_size=0.2, random_state=30)
```

In [54]:
```
sample_1 = df2.sample(5000, random_state=42)
```

In [55]:
```
y_sample1 = sample_1["Severity"]
x_sample1 = sample_1.drop("Severity", axis=1)
```

In [56]:
```
X_train, X_test, y_train, y_test = train_test_split(x_sample1, y_sample1, random_state=42)
```

In [57]:
```
from sklearn import svm
```

Linear Kernel with Costs 0.1, 1, 10

In [58]:
```
svc = svm.SVC(kernel= "linear", C= .2)
svc.fit(X_train, y_train)
```

Out[58]:

SVC

SVC(C=0.2, kernel='linear')

In [59]:
```
print("Train score:", svc.score(X_train, y_train))
print("Validation score:", svc.score(X_test, y_test))
```

```
Train score: 0.6170666666666667
Validation score: 0.5936
```

In [60]:
```
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, f1_score, roc_curve

y_pred = svc.predict(X_test)
print(classification_report(y_test, y_pred))
```

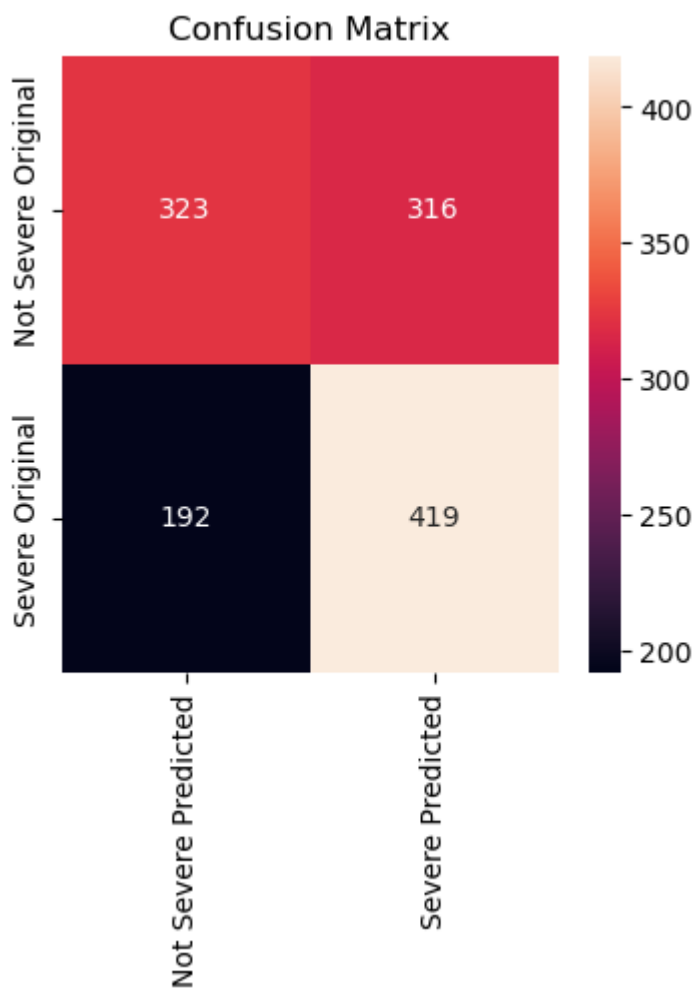
```
              precision    recall  f1-score   support

  Not Severe       0.63      0.51      0.56       639
      Severe       0.57      0.69      0.62       611

    accuracy                           0.59      1250
   macro avg       0.60      0.60      0.59      1250
weighted avg       0.60      0.59      0.59      1250
```

In [61]:
```
y_pred = svc.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)
```

In [62]:
```
index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



In [67]:
```
svm_linear = svm.SVC(kernel= "linear", C= 1)
svm_linear.fit(X_train, y_train)

print("Train score:", svm_linear.score(X_train, y_train))
print("Validation score:", svm_linear.score(X_test, y_test))

y_pred = svm_linear.predict(X_test)


print(classification_report(y_test, y_pred))
```

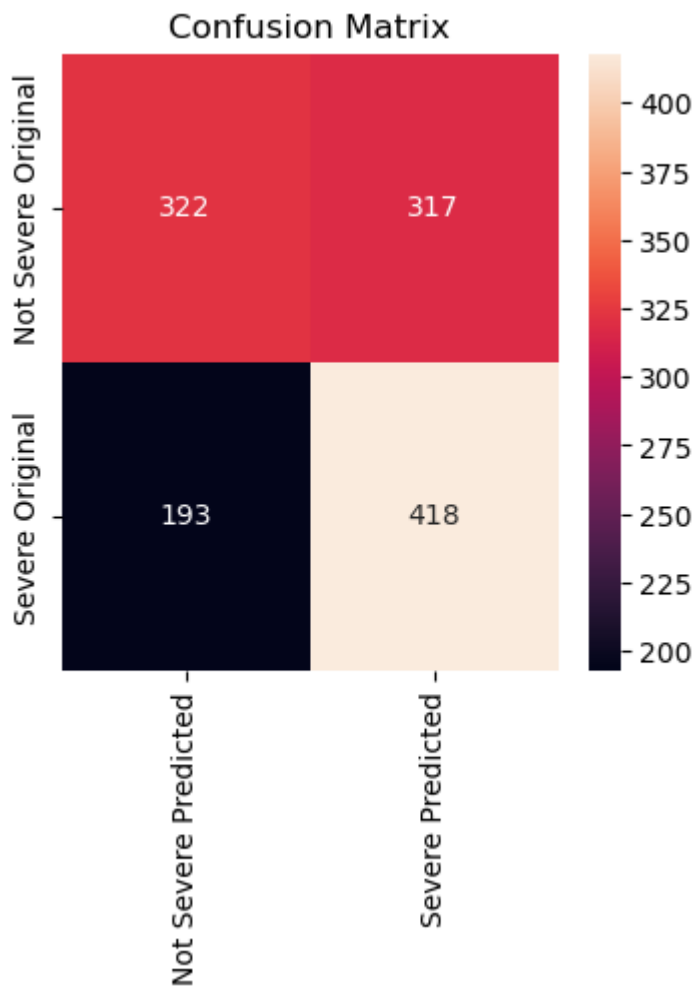
```
Train score: 0.6176
Validation score: 0.592
              precision    recall  f1-score   support

  Not Severe       0.63      0.50      0.56       639
      Severe       0.57      0.68      0.62       611

    accuracy                           0.59      1250
   macro avg       0.60      0.59      0.59      1250
weighted avg       0.60      0.59      0.59      1250
```

In [68]:
```
y_pred = svm_linear.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)
```

```
index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



In [69]:
```
svm_2 = svm.SVC(kernel= "linear", C= 10)
svm_2.fit(X_train, y_train)

print("Train score:", svm_2.score(X_train, y_train))
print("Validation score:", svm_2.score(X_test, y_test))

y_pred = svm_2.predict(X_test)

print(classification_report(y_test, y_pred))
```

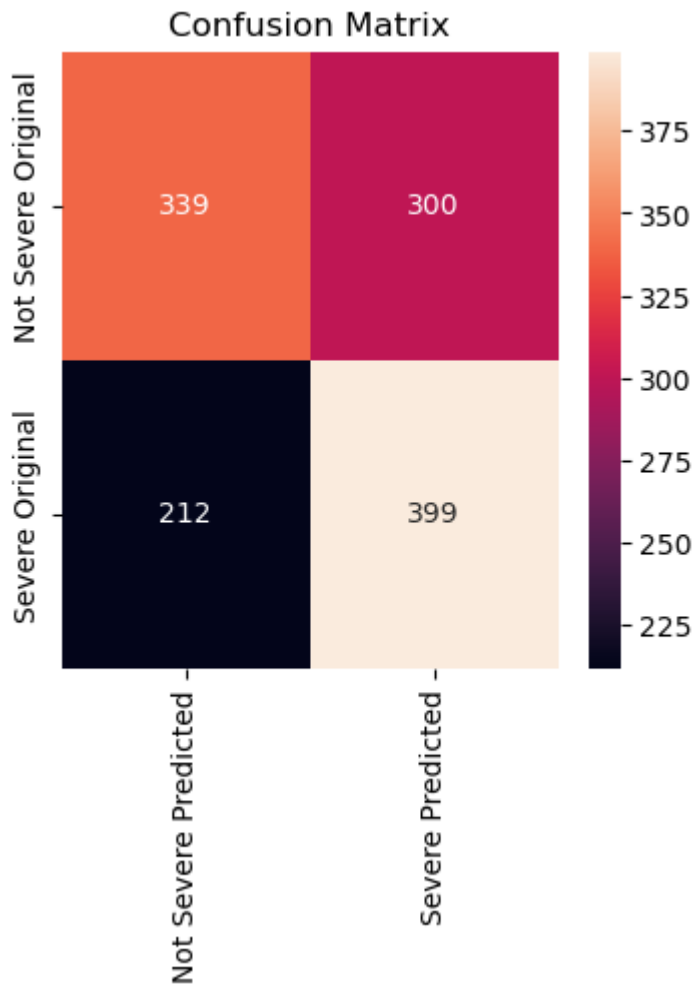
```
Train score: 0.6242666666666666
Validation score: 0.5904
              precision    recall  f1-score   support

  Not Severe       0.62      0.53      0.57       639
      Severe       0.57      0.65      0.61       611

    accuracy                           0.59      1250
   macro avg       0.59      0.59      0.59      1250
weighted avg       0.59      0.59      0.59      1250
```

In [70]:
```
y_pred = svm_2.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



Polynomial Kernel With costs 0.1, 1, 10

In [71]:
```
svm_poly = svm.SVC(kernel= "poly", C= 0.1, degree= 4)
svm_poly.fit(X_train, y_train)

print("Train score:", svm_poly.score(X_train, y_train))
print("Validation score:", svm_poly.score(X_test, y_test))

y_pred = svm_poly.predict(X_test)

print(classification_report(y_test, y_pred))
```

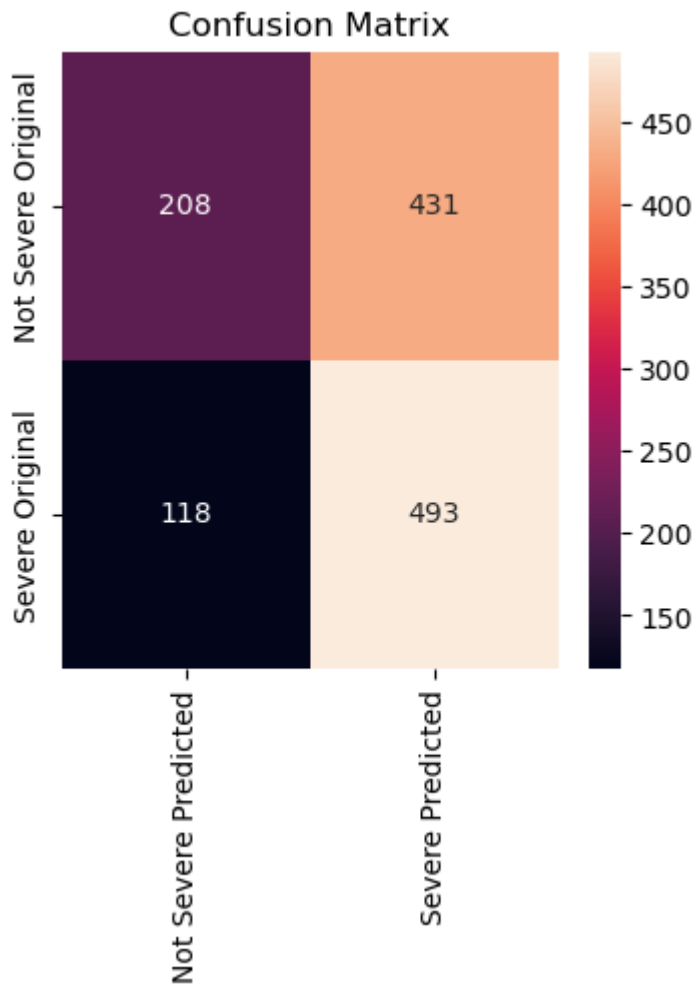
```
Train score: 0.5664
Validation score: 0.5608
              precision    recall  f1-score   support

  Not Severe       0.64      0.33      0.43       639
      Severe       0.53      0.81      0.64       611

    accuracy                           0.56      1250
   macro avg       0.59      0.57      0.54      1250
weighted avg       0.59      0.56      0.53      1250
```

In [72]:
```
y_pred = svm_poly.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



In [75]:
```
svm_poly_2 = svm.SVC(kernel= "poly", C= 1, degree= 4)
svm_poly_2.fit(X_train, y_train)

print("Train score:", svm_poly_2.score(X_train, y_train))
print("Validation score:", svm_poly_2.score(X_test, y_test))

y_pred = svm_poly_2.predict(X_test)

print(classification_report(y_test, y_pred))
```

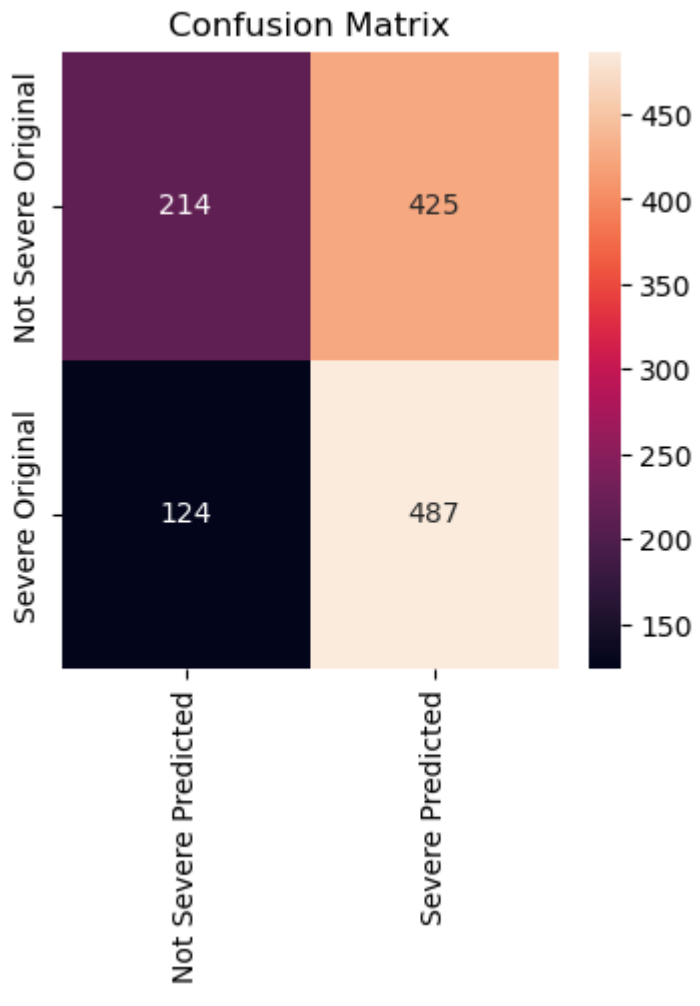
```
Train score: 0.5690666666666667
Validation score: 0.5608
              precision    recall  f1-score   support

  Not Severe       0.63      0.33      0.44       639
      Severe       0.53      0.80      0.64       611

    accuracy                           0.56      1250
   macro avg       0.58      0.57      0.54      1250
weighted avg       0.58      0.56      0.54      1250
```

In [74]:
```
y_pred = svm_poly_2.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



In [76]:
```
svm_poly_3 = svm.SVC(kernel= "poly", C= 10, degree= 4)
svm_poly_3.fit(X_train, y_train)

print("Train score:", svm_poly_3.score(X_train, y_train))
print("Validation score:", svm_poly_3.score(X_test, y_test))

y_pred = svm_poly_3.predict(X_test)

print(classification_report(y_test, y_pred))
```

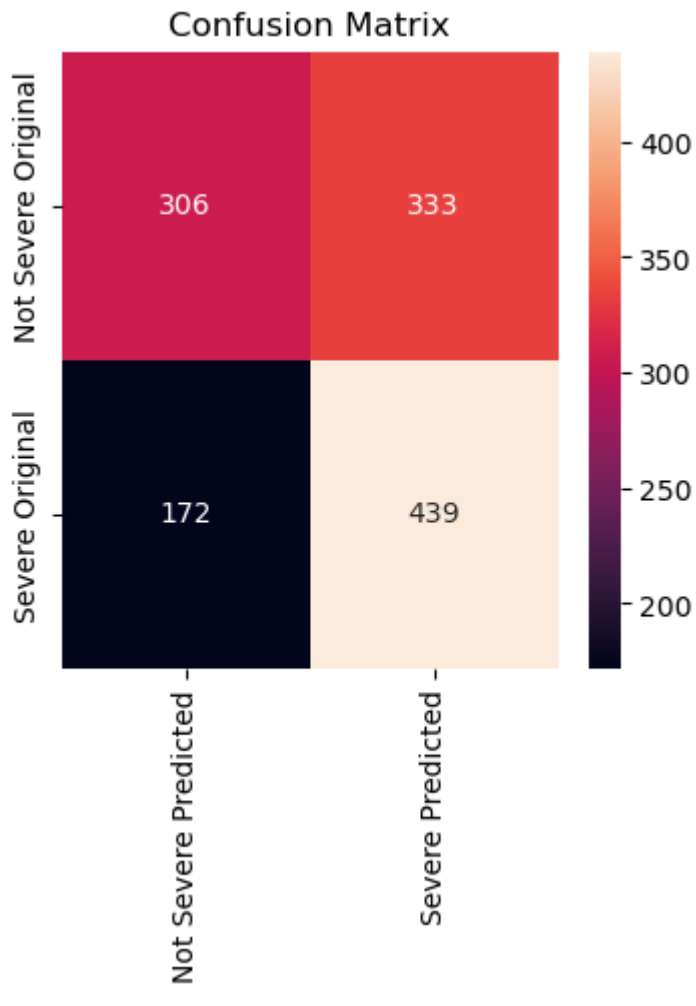
```
Train score: 0.6056
Validation score: 0.596
              precision    recall  f1-score   support

  Not Severe       0.64      0.48      0.55       639
      Severe       0.57      0.72      0.63       611

    accuracy                           0.60      1250
   macro avg       0.60      0.60      0.59      1250
weighted avg       0.61      0.60      0.59      1250
```

In [77]:
```
y_pred = svm_poly_3.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



RBF Guassian Kernel With Costs 0.1, 1, 10

In [78]:
```
svm_rbf = svm.SVC(kernel= "rbf", C= 0.1)
svm_rbf.fit(X_train, y_train)

print("Train score:", svm_rbf.score(X_train, y_train))
print("Validation score:", svm_rbf.score(X_test, y_test))

y_pred = svm_rbf.predict(X_test)

print(classification_report(y_test, y_pred))
```

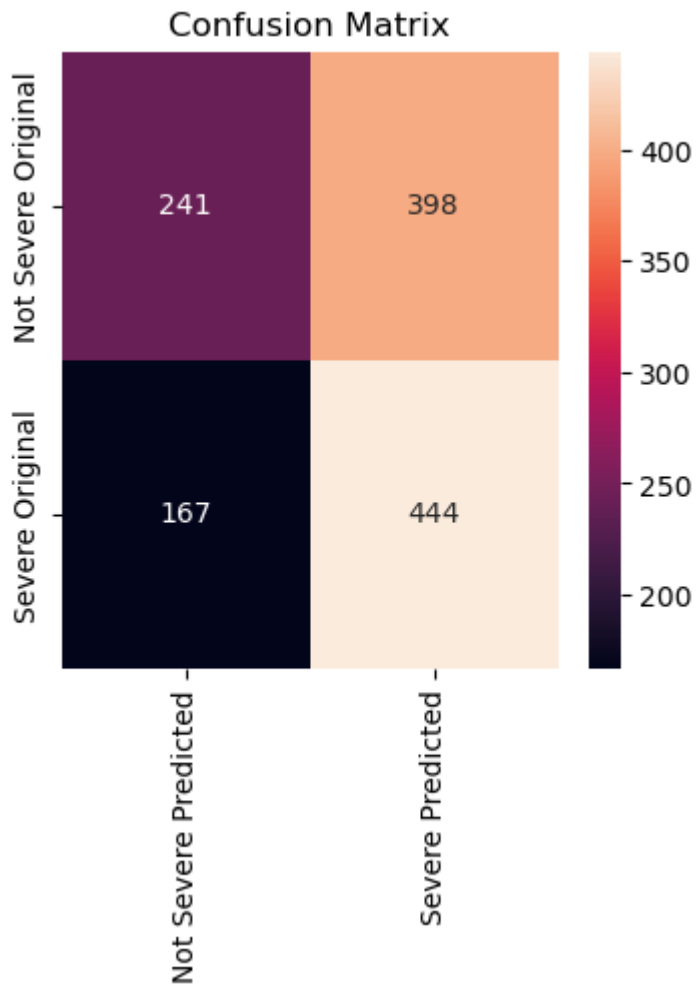
```
Train score: 0.5421333333333334
Validation score: 0.548
              precision    recall  f1-score   support

  Not Severe       0.59      0.38      0.46       639
      Severe       0.53      0.73      0.61       611

    accuracy                           0.55      1250
   macro avg       0.56      0.55      0.54      1250
weighted avg       0.56      0.55      0.53      1250
```

In [79]:
```
y_pred = svm_rbf.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



In [80]:
```
svm_rbf_2 = svm.SVC(kernel= "rbf", C= 1)
svm_rbf_2.fit(X_train, y_train)

print("Train score:", svm_rbf_2.score(X_train, y_train))
print("Validation score:", svm_rbf_2.score(X_test, y_test))

y_pred = svm_rbf_2.predict(X_test)

print(classification_report(y_test, y_pred))
```

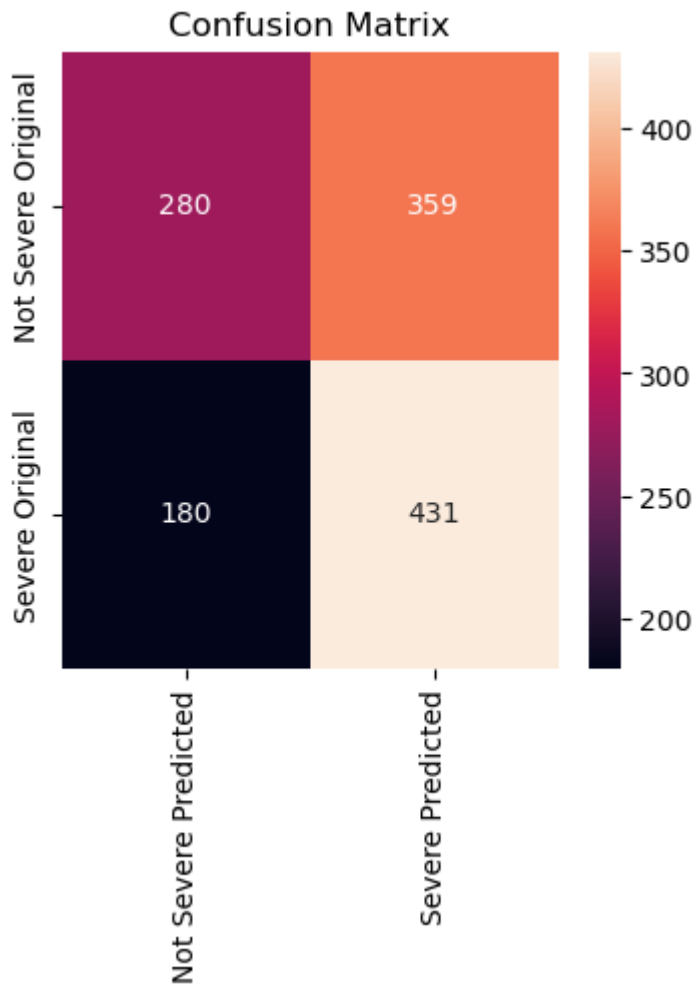
```
Train score: 0.5813333333333334
Validation score: 0.5688
              precision    recall  f1-score   support

  Not Severe       0.61      0.44      0.51       639
      Severe       0.55      0.71      0.62       611

    accuracy                           0.57      1250
   macro avg       0.58      0.57      0.56      1250
weighted avg       0.58      0.57      0.56      1250
```

In [81]:
```
y_pred = svm_rbf_2.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```



In [82]:
```
sv_rbf_3 = svm.SVC(kernel= "rbf", C= 10)
sv_rbf_3.fit(X_train, y_train)

print("Train score:", sv_rbf_3.score(X_train, y_train))
print("Validation score:", sv_rbf_3.score(X_test, y_test))

y_pred = sv_rbf_3.predict(X_test)

print(classification_report(y_test, y_pred))
```

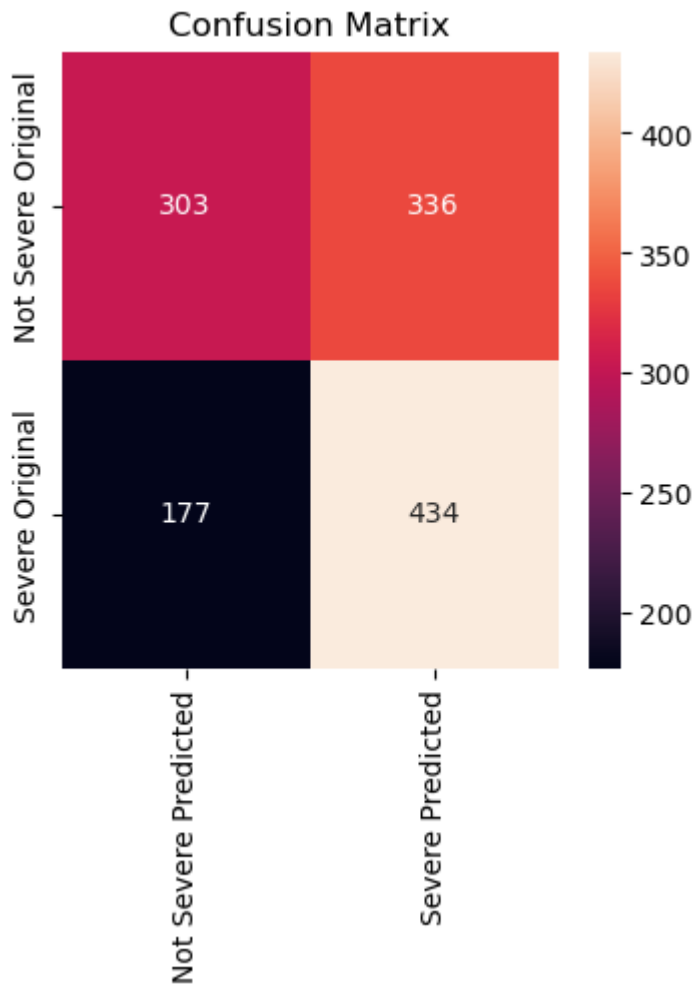
```
Train score: 0.6128
Validation score: 0.5896
              precision    recall  f1-score   support

  Not Severe       0.63      0.47      0.54       639
      Severe       0.56      0.71      0.63       611

    accuracy                           0.59      1250
   macro avg       0.60      0.59      0.59      1250
weighted avg       0.60      0.59      0.58      1250
```

In [83]:
```
y_pred = sv_rbf_3.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

index = ["Not Severe Original", "Severe Original"]
columns = ["Not Severe Predicted", "Severe Predicted"]
conf_matrix = pd.DataFrame(data=confmat, columns=columns, index=index)
plt.figure(figsize=(4,4))
sns.heatmap(conf_matrix, annot=True, fmt="d")
plt.title("Confusion Matrix")
plt.show()
```


Confusion Matrix
Not Severe Original
Severe Original
303
336
177
434
Not Severe Predicted
Severe Predicted
400
350
300
250
200