nypd2

December 8, 2021

1 NYPD Allegations

- **See the main project notebook for instructions to be sure you satisfy the rubric!**
- See Project 03 for information on the dataset.
- A few example prediction questions to pursue are listed below. However, don't limit yourself to them!
  - Predict the outcome of an allegation (might need to feature engineer your output column).
  - Predict the complainant or officer ethnicity.
  - Predict the amount of time between the month received vs month closed (difference of the two columns).
  - Predict the rank of the officer.

Be careful to justify what information you would know at the "time of prediction" and train your model using only those features.

2 Summary of Findings

2.0.1 Introduction

In this finding, we will be continue to study on the NYPD data set. More specifically, we will be using machine learning modeling to make predictions on NYPD officer's ranking during incident based on various predictors.

2.0.2 Baseline Model

To start off the research, we will be performing data cleaning and selection on needed data. Cleaning method will inherit from the previoud project by filling in missing values and filtering out `"NaN"` values. After cleaning the data, we will be conducting a base logistic model using the `Sklean` package, and `Pipeline` that helps fitting and transforming the data into our logistic regression model.

- Predictor `"mos_ethnicity"` is used as the baseline model to predict the officer's rankin. It seems intuitively that there is a relationship between officer's ethnicity and its title. Some ethnicity seems to have a higher general ranking than the others, and that is also the reason why we choose to start of the prediction using the `"mos_ethnicity"` variable.

- We know that `"mos_ethnicity"` is an ordinal variable by discovering that it consist various ethnicities such as `Hispanic`, `White`, `Black`, `Asian`. To catergorize this predictor, we decide to use `OrdinalEncoder` to encode the difference in ethnicity, and by fitting it into a pipeline and logistic regression model, we obtained a 68% of accuracy of our model and a 0.6834 R-squared value for this baseline model.

Note: R-squared is a goodness-of-fit measure for linear regression models. It is valued between 0 to 1, the closer the number is getting to 1, means that the better the model is predicted.

2.0.3 Final Model

Although our baseline model has a pretty good prediction on the officer's ranking. We would like to further investigate and want to improve the performance of the prediction. We designed to include feature engineering and predictor searching into different modeling and found the best model for our final model by using the "mos_ethnicity","mos_gender", "mos_age_incident","rank_now" variables. In addition to "mos_ethnicity", the final(best) model has three additional features that strongly helped to predict the officer's title. In the process of searching a related predictor manually by adding new features one by one, and later resulted our final model.

- To fit the predictors into the pipeline, we first transform the column "mos_age_incident" into standardscaler, then apply one-hot-encoder to "mos_gender" and ordinal encoder to the columns "mos_ethnicity" and "rank_now".
- "mos_gender" are categorized by "M" and "F". One-hot encoder will be the most appropreate to the transformation.
- "rank_now" consist different ranking titles, and therfore it is being categorize as "mos_ethnicity" in above for the same reason.

After fitting the predictors into our final model, we obtained a 71% on the model accuracy. By all that means is that we are 71% confident to correctly predict the officer's ranking at the indident given the "mos_ethnicity","mos_gender", "mos_age_incident","rank_now" predictors. Also, this model gives a 0.7076 R-squared value. It is so far the best predicted model that we obtained.

2.0.4 Fairness Evaluation

Lastly, we will be assessing the model through a fairness evaluation, we will be splitting our data and uses permutation to conduct this study. We set a test size to 0.3 and a 42 random state in our splitting so that our data can be shuffled and draw more randomly for the assessment. The observing predictors will remains the same as our final model.

- After splitting, we obtained X_train, X_test, y_train, y_test and ready to fit the data into our modeling.
- We fit the X_train data into the final model pipeline and obtain a predicted train value, same for the X_test data.
- After fitting the two modeling, we can see from the classification report that the two model are having the same accuracy of 71%. However, the f1-score on the X_train set peforms slightly better. Note: The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. (Cited from Wikipedia)
- Since the two models are obtaining a similar accuracy score and f1-score, we can say that we have a decent low false positives and low false negatives, and a true postive and true negative prediction. Therefore, we can say that this model is pretty fair.

3 Code

[1]:

```
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import seaborn as sns
%matplotlib inline
%config InlineBackend.figure_format = 'retina'  # Higher resolution figures
```

[2]:

```
# Loade the data
df = pd.read_csv('allegations_202007271729.csv')
```

[3]:

```
# Create a copy of the original data
data = df.copy()
```

[4]:

```
# Display the first 5 entries of the data set
data.head()
```

[4]:

```
   unique_mos_id first_name last_name command_now  shield_no  complaint_id  \
0          10004   Jonathan      Ruiz     078 PCT       8409         42835
1          10007       John     Sears     078 PCT       5952         24601
2          10007       John     Sears     078 PCT       5952         24601
3          10007       John     Sears     078 PCT       5952         26146
4          10009      Noemi    Sierra     078 PCT      24058         40253

   month_received  year_received  month_closed  year_closed  …  \
0               7           2019             5         2020  …
1              11           2011             8         2012  …
2              11           2011             8         2012  …
3               7           2012             9         2013  …
4               8           2018             2         2019  …

  mos_age_incident complainant_ethnicity complainant_gender  \
0               32                 Black             Female
1               24                 Black               Male
2               24                 Black               Male
3               25                 Black               Male
4               39                   NaN                NaN

  complainant_age_incident           fado_type                    allegation  \
0                     38.0  Abuse of Authority  Failure to provide RTKA card
1                     26.0         Discourtesy                        Action
2                     26.0  Offensive Language                          Race
3                     45.0  Abuse of Authority                      Question
4                     16.0               Force                Physical force
```

```
  precinct                                contact_reason  \
0     78.0                       Report-domestic dispute
1     67.0                              Moving violation
2     67.0                              Moving violation
3     67.0  PD suspected C/V of violation/crime - street
4     67.0                                Report-dispute

                outcome_description                         board_disposition
0  No arrest made or summons issued  Substantiated (Command Lvl Instructions)
1   Moving violation summons issued                   Substantiated (Charges)
2   Moving violation summons issued                   Substantiated (Charges)
3  No arrest made or summons issued                   Substantiated (Charges)
4    Arrest - other violation/crime      Substantiated (Command Discipline A)

[5 rows x 27 columns]
```

[5]:

```
# Data cleaning on needed to use columns

data['Complaint_ethnicity'] = data['complainant_ethnicity'].replace({'Unknow':␣
↪np.NaN, 'Refused':np.NaN})
data['complainant_gender'] = data['complainant_gender'].replace({'Gender␣
↪non-conforming': np.NaN, 'Not described': np.NaN, 'Transman(FTM)': 'Male',␣
↪'Transwoman (MTF)': 'Female'})
data = data.drop_duplicates()
data = data.dropna()
```

3.0.1 Baseline Model

[6]:

```
import matplotlib.pyplot as plt
import sklearn.preprocessing as pp
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import OrdinalEncoder
from sklearn.compose import ColumnTransformer
from sklearn import metrics
```

[7]:

```
X, y = data[['mos_ethnicity']], data['rank_incident']
```

[8]:

```
#Pipeline for the transformation

pl1 = Pipeline([
    ('ord', OrdinalEncoder()),
    ('log_reg', LogisticRegression())
])
```

[9]:

```
pl1.fit(X, y)
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[9]:

```
Pipeline(steps=[('ord', OrdinalEncoder()), ('log_reg', LogisticRegression())])
```

[10]:

```
y_pred = pl1.predict(X)
y_pred
```

[10]:

```
array(['Police Officer', 'Police Officer', 'Police Officer', …,
       'Police Officer', 'Police Officer', 'Police Officer'], dtype=object)
```

[11]:

```
print(metrics.classification_report(y, y_pred))
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
                  precision    recall  f1-score   support

         Captain       0.00      0.00      0.00       109
Deputy Inspector       0.00      0.00      0.00        75
       Detective       0.00      0.00      0.00      2584
       Inspector       0.00      0.00      0.00        23
      Lieutenant       0.00      0.00      0.00      1018
  Police Officer       0.68      1.00      0.81     18690
        Sergeant       0.00      0.00      0.00      4849
```

```
     accuracy                           0.68     27348
    macro avg       0.10      0.14      0.12     27348
 weighted avg       0.47      0.68      0.55     27348
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[12]:
```
# R^2
pl1.score(X, y) # Ok prediction
```

[12]:
```
0.6834137779727951
```

3.0.2 Model Searching for a better model

[13]:
```
X, y = data[['mos_gender','mos_age_incident']], data['rank_incident']
# Numeric columns and associated transformers
num_feat = ['mos_age_incident']
num_transformer = Pipeline(steps=[
    ('scaler', pp.StandardScaler())   # z-scale
])

# Categorical columns and associated transformers
cat_hot_feat = ['mos_gender']
cat_hot_transformer = Pipeline(steps=[
    ('onehot', pp.OneHotEncoder())     # output from Ordinal becomes input to␣
 ↪OneHot
])

# preprocessing pipeline (put them together)
preproc = ColumnTransformer(
    transformers=[
        ('num', num_transformer, num_feat),
        ('hot_cat', cat_hot_transformer, cat_hot_feat),
    ])

pl2 = Pipeline(steps=[('preprocessor', preproc), ('regressor',␣
 ↪LogisticRegression())])

# Fit the model into the pipeline
pl2.fit(X,y)
y_pred = pl2.predict(X)
print(metrics.classification_report(y, y_pred))
```

```python
# R^2
print(pl2.score(X, y)) # A Slightly better prediction
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
                  precision    recall  f1-score   support

          Captain       0.00      0.00      0.00       109
Deputy Inspector       0.00      0.00      0.00        75
        Detective       0.00      0.00      0.00      2584
        Inspector       0.00      0.00      0.00        23
       Lieutenant       0.00      0.00      0.00      1018
   Police Officer       0.72      0.95      0.82     18690
         Sergeant       0.29      0.17      0.21      4849

         accuracy                           0.68     27348
        macro avg       0.14      0.16      0.15     27348
     weighted avg       0.54      0.68      0.60     27348

0.6765759836185461
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
```

```
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[14]:

```
X, y = data[['mos_ethnicity','mos_gender','mos_age_incident']],␣
 ↪data['rank_incident']
# Numeric columns and associated transformers
num_feat = ['mos_age_incident']
num_transformer = Pipeline(steps=[
    ('scaler', pp.StandardScaler())   # z-scale
])

# Categorical columns and associated transformers
cat_hot_feat = ['mos_gender']
cat_hot_transformer = Pipeline(steps=[
    ('onehot', pp.OneHotEncoder())     # output from Ordinal becomes input to␣
 ↪OneHot
])

# Categorical columns and associated transformers
cat_feat = ['mos_ethnicity']
cat_transformer = Pipeline(steps=[
    ('ordin', pp.OrdinalEncoder())     # output from Ordinal becomes input to␣
 ↪OneHot
])

# preprocessing pipeline (put them together)
preproc = ColumnTransformer(
    transformers=[
        ('num', num_transformer, num_feat),
        ('hot_cat', cat_hot_transformer, cat_hot_feat),
        ('cat', cat_transformer, cat_feat)
    ])

pl3 = Pipeline(steps=[('preprocessor', preproc), ('regressor',␣
 ↪LogisticRegression())])

# Fit the model into the pipeline
pl3.fit(X,y)
y_pred = pl3.predict(X)
print(metrics.classification_report(y, y_pred))

# R^2
print(pl3.score(X, y)) # A even slightly better prediction
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
```

```
packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
                  precision    recall  f1-score   support

         Captain       0.00      0.00      0.00       109
Deputy Inspector       0.00      0.00      0.00        75
       Detective       0.00      0.00      0.00      2584
       Inspector       0.00      0.00      0.00        23
      Lieutenant       0.17      0.01      0.02      1018
  Police Officer       0.72      0.95      0.82     18690
        Sergeant       0.29      0.16      0.20      4849

        accuracy                           0.68     27348
       macro avg       0.17      0.16      0.15     27348
    weighted avg       0.55      0.68      0.60     27348

0.6774901272487933
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

3.0.3 Final Model

[15]:
```
X, y = data[['mos_ethnicity','mos_gender','mos_age_incident','rank_now']],␣
 ↪data['rank_incident']
```

[16]:
```
# Numeric columns and associated transformers
num_feat = ['mos_age_incident']
num_transformer = Pipeline(steps=[
    ('scaler', pp.StandardScaler())   # z-scale
])

# Categorical columns and associated transformers
cat_hot_feat = ['mos_gender']
cat_hot_transformer = Pipeline(steps=[
    ('onehot', pp.OneHotEncoder())     # output from Ordinal becomes input to␣
 ↪OneHot
])

# Categorical columns and associated transformers
cat_feat = ['mos_ethnicity', 'rank_now']
cat_transformer = Pipeline(steps=[
    ('ordin', pp.OrdinalEncoder())     # output from Ordinal becomes input to␣
 ↪OneHot
])

# preprocessing pipeline (put them together)
preproc = ColumnTransformer(
    transformers=[
        ('num', num_transformer, num_feat),
        ('hot_cat', cat_hot_transformer, cat_hot_feat),
        ('cat', cat_transformer, cat_feat)
    ])

pl4 = Pipeline(steps=[('preprocessor', preproc), ('regressor',␣
 ↪LogisticRegression())])
```

[17]:
```
pl4.fit(X,y)
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\linear_model\_logistic.py:814: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
```

```
regression
  n_iter_i = _check_optimize_result(
```

```
[17]: Pipeline(steps=[('preprocessor',
                       ColumnTransformer(transformers=[('num',
                                                        Pipeline(steps=[('scaler',
       StandardScaler())]),
                                                        ['mos_age_incident']),
                                                       ('hot_cat',
                                                        Pipeline(steps=[('onehot',
       OneHotEncoder())]),
                                                        ['mos_gender']),
                                                       ('cat',
                                                        Pipeline(steps=[('ordin',
       OrdinalEncoder())]),
                                                        ['mos_ethnicity',
                                                         'rank_now'])])),
                      ('regressor', LogisticRegression())])
```

```
[18]: y_pred = pl4.predict(X)
      y_pred
```

```
[18]: array(['Police Officer', 'Police Officer', 'Police Officer', …,
             'Police Officer', 'Police Officer', 'Police Officer'], dtype=object)
```

```
[19]: print(metrics.classification_report(y, y_pred))
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
                  precision    recall  f1-score   support

         Captain       0.61      0.39      0.47       109
Deputy Inspector       0.30      0.09      0.14        75
       Detective       0.46      0.28      0.35      2584
       Inspector       0.00      0.00      0.00        23
      Lieutenant       0.28      0.03      0.06      1018
  Police Officer       0.74      0.93      0.83     18690
        Sergeant       0.52      0.23      0.32      4849

        accuracy                           0.71     27348
       macro avg       0.42      0.28      0.31     27348
    weighted avg       0.66      0.71      0.66     27348
```

```
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[20]:

```
# R^2
pl4.score(X, y) # Best prediction among the all
```

[20]:

```
0.7076203013017406
```

3.0.4 Fairness Evaluation

[21]:

```
# Recall from the final model that the X,y is:

X,y = data[['mos_ethnicity','mos_gender','mos_age_incident','rank_now']],␣
↪data['rank_incident']
```

[22]:

```
# Split the training and the test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,␣
↪random_state = 42)
```

[23]:

```
# First 5 rows of the train data after splitting
display(X_train.head(5))
```

```
      mos_ethnicity mos_gender  mos_age_incident        rank_now
27319         White          M                28        Sergeant
18133         White          M                34       Detective
4287          White          M                44        Sergeant
22056         White          M                41        Sergeant
30545      Hispanic          M                37  Police Officer
```

[24]:

```
display(X_test.head(5))
```

```
      mos_ethnicity mos_gender  mos_age_incident        rank_now
18946         Black          M                36       Detective
4815          White          M                38       Detective
6235          White          M                27        Sergeant
31397         White          F                39      Lieutenant
31224      Hispanic          M                29  Police Officer
```

[25]:
```
display(y_train.head(5))
```

```
27319    Police Officer
18133    Police Officer
4287           Sergeant
22056          Sergeant
30545    Police Officer
Name: rank_incident, dtype: object
```

[26]:
```
display(y_test.head(5))
```

```
18946         Detective
4815     Police Officer
6235     Police Officer
31397        Lieutenant
31224    Police Officer
Name: rank_incident, dtype: object
```

[27]:
```
pred_train = pl4.predict(X_train)
print(metrics.classification_report(pred_train, y_train))
```

```
                  precision    recall  f1-score   support

         Captain       0.40      0.65      0.50        55
Deputy Inspector       0.10      0.29      0.14        17
       Detective       0.28      0.45      0.35      1103
       Inspector       0.00      0.00      0.00         0
      Lieutenant       0.03      0.32      0.06        75
  Police Officer       0.93      0.74      0.83     16377
        Sergeant       0.23      0.52      0.32      1516

        accuracy                           0.71     19143
       macro avg       0.28      0.43      0.31     19143
    weighted avg       0.83      0.71      0.76     19143

C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Recall
```

```
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[28]:

```
pred_test = pl4.predict(X_test)
print(metrics.classification_report(pred_test, y_test))
```

```
                  precision    recall  f1-score   support

         Captain       0.30      0.43      0.35        14
Deputy Inspector       0.09      0.33      0.14         6
       Detective       0.28      0.47      0.35       468
       Inspector       0.00      0.00      0.00         0
      Lieutenant       0.03      0.22      0.05        41
  Police Officer       0.93      0.74      0.83      7067
        Sergeant       0.22      0.52      0.31       609

        accuracy                           0.71      8205
       macro avg       0.26      0.39      0.29      8205
    weighted avg       0.84      0.71      0.76      8205

C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\linxi\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2
kfra8p0\LocalCache\local-packages\Python39\site-
packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Recall
and F-score are ill-defined and being set to 0.0 in labels with no true samples.
Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```