qda_final_predictions

December 22, 2022

0.1 FINAL PREDICITONS GENERATION

[48]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

from sklearn.pipeline import make_pipeline
import sklearn.linear_model as skl_lm
import sklearn.discriminant_analysis as skl_da
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler
```

[49]:
```
# Loading the train.csv as the main dataset
data_train = pd.read_csv("../data/train.csv")
data_test = pd.read_csv("../data/test.csv")

# Column Transformation to lowercase and underscored spaces
data_train.columns = data_train.columns.str.replace(' ', '_')
data_train.columns = data_train.columns.str.replace('-', '_')
data_train.columns = data_train.columns.str.lower()

data_test.columns = data_test.columns.str.replace(' ', '_')
data_test.columns = data_test.columns.str.replace('-', '_')
data_test.columns = data_test.columns.str.lower()

X_train = data_train.loc[:, data_train.columns != 'lead']
y_train = data_train.loc[:, data_train.columns == 'lead']

X_test = data_test

# Feature transformations for train data
X_train['lead_words_precentage'] = X_train.number_of_words_lead / X_train.
 ↪total_words
X_train['colead_words_percentage'] = (X_train.number_of_words_lead - X_train.
 ↪difference_in_words_lead_and_co_lead) / X_train.total_words
```

```
X_train['female_words_percentage'] = X_train.number_words_female / X_train.
 ↪total_words

# Feature transformations for test data
X_test['lead_words_precentage'] = X_test.number_of_words_lead / X_test.
 ↪total_words
X_test['colead_words_percentage'] = (X_test.number_of_words_lead - X_test.
 ↪difference_in_words_lead_and_co_lead) / X_test.total_words
X_test['female_words_percentage'] = X_test.number_words_female / X_test.
 ↪total_words

X_train_transformed = X_train[
    [
        'total_words',
        'number_of_male_actors',
        'number_of_female_actors',
        'mean_age_male',
        'mean_age_female',
        'age_lead',
        'age_co_lead',
        'lead_words_precentage',
        'colead_words_percentage',
        'female_words_percentage'
    ]
]
```

FITTING THE MODEL

```
[50]: pipe = make_pipeline(
          StandardScaler(),
          QuadraticDiscriminantAnalysis()
      )

      pipe.fit(X_train_transformed, y_train.to_numpy().reshape(-1, ))
```

```
[50]: Pipeline(steps=[('standardscaler', StandardScaler()),
                      ('quadraticdiscriminantanalysis',
                       QuadraticDiscriminantAnalysis())])
```

```
[51]: from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
       ↪f1_score

      # y_train_true = y_train["lead"].map({'Male': 1, 'Female': 0})
      # y_train_pred = [1 if pred == "Male" else 0 for pred in pipe.
       ↪predict(X_train[selected_features])]

      # recall_score(y_train_true, y_train_pred)
```

```
print('TRAINING SET METRICS:')
print('Accuracy:', accuracy_score(y_train["lead"].map({'Male': 1, 'Female':␣
↪0}), [1 if pred == "Male" else 0 for pred in pipe.
↪predict(X_train_transformed)]))
print('Precision:', precision_score(y_train["lead"].map({'Male': 1, 'Female':␣
↪0}), [1 if pred == "Male" else 0 for pred in pipe.
↪predict(X_train_transformed)]))
print('Recall:', recall_score(y_train["lead"].map({'Male': 1, 'Female': 0}), [1␣
↪if pred == "Male" else 0 for pred in pipe.predict(X_train_transformed)]))
print('F1:', f1_score(y_train["lead"].map({'Male': 1, 'Female': 0}), [1 if pred␣
↪== "Male" else 0 for pred in pipe.predict(X_train_transformed)]))
print('\n')
```

```
TRAINING SET METRICS:
Accuracy: 0.9384023099133783
Precision: 0.9569074778200254
Recall: 0.9617834394904459
F1: 0.9593392630241423
```

FINAL PREDICTIONS

[52]:
```
X_test_transformed = X_test[
    [
        'total_words',
        'number_of_male_actors',
        'number_of_female_actors',
        'mean_age_male',
        'mean_age_female',
        'age_lead',
        'age_co_lead',
        'lead_words_precentage',
        'colead_words_percentage',
        'female_words_percentage'
    ]
]

final_predictions = pipe.predict(X_test_transformed)
```

[53]:
```
final_csv = np.array([1 if pred == "Female" else 0 for pred in␣
↪final_predictions])
```

[54]:
```
final_csv.tofile("../tests/predictions.csv", sep=',')
```

FEATURE TRANSFORMATIONS

[55]:
```
initial_features = [
        'number_words_female',
        'total_words',
        'number_of_words_lead',
        'difference_in_words_lead_and_co_lead',
        'number_of_male_actors',
        'year',
        'number_of_female_actors',
        'number_words_male',
        'gross',
        'mean_age_male',
        'mean_age_female',
        'age_lead',
        'age_co_lead'
]

transformed_features = [
        'total_words',
        'number_of_male_actors',
        'number_of_female_actors',
        'mean_age_male',
        'mean_age_female',
        'age_lead',
        'age_co_lead',
        'lead_words_precentage',
        'colead_words_percentage',
        'female_words_percentage'
]
```