Model

December 3, 2023

[1]:
```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score
import pandas as pd
```

[2]:
```python
# Read data from Excel file into a Pandas DataFrame
file_path = 'dm_mimic_pathways.csv'
df = pd.read_csv(file_path)
```

[3]:
```python
column_name_mapping = {'person_id': 'Person',
                       'race_concept_id': 'Race',
                       'gender_concept_id':'Gender',
                       'age_group':'Age Group',
                       'pathways':'Treatment Regimen'}

race_mapping = {8527: 'White/ Hispanic',
                8516: 'Black',
                8515: 'Asian',
                0:'Unknown',
                38003592: 'Asian',
                4077359: 'Other',
                4218674: 'Unknown',
                4188159: 'White/ Hispanic',
                38003599: 'Black',
                38003574: 'Asian',
                4212311: 'Asian',
                38003600: 'Black',
                8557: 'Other',
                38003584: 'Asian',
                38003578: 'Asian',
                4087921: 'Other',
                38003615: 'Other',
                38003581: 'Asian',
                8657: 'Other',
                38003579: 'Asian',
```

```
                 38003605: 'Black',
                 38003614: 'White',
                 4213463: 'White'}

gender_mapping = {8507: 'Male',
                  8532: 'Female'}

age_mapping = {'10 - 19': 'Teens',
               '20 - 29': 'Twenties',
               '30 - 39': 'Thirties',
               '40 - 49': 'Forties',
               '50 - 59': 'Fifties',
               '60 - 69': 'Sixties',
               '70 - 79': 'Seventies',
               '80 - 89': 'Eighties',
              '> 90': 'Nineties'}
```

[4]:
```
df = df.rename(columns=column_name_mapping)
df['Race'] = df['Race'].replace(race_mapping)
df['Gender'] = df['Gender'].replace(gender_mapping)
df['Age Group'] = df['Age Group'].replace(age_mapping)
df['Age Group'].fillna('Unknown', inplace=True)
```

[5]:
```
df = df[(df['Age Group'] != 'Unknown') & (df['Race'] != 'Unknown')]
```

[6]:
```
print(len(df))
n = 9
values_to_preserve = df['Treatment Regimen'].value_counts().head(n)
print(values_to_preserve)
```

```
1746
Treatment Regimen
19071700             463
19071700,40166274    197
40164929              73
40164930              62
40166274              61
19071700,40164929     47
19077638              45
19030580              24
19077682              19
Name: count, dtype: int64
```

[7]:
```
def preserve_or_change(value, value_set, replacement_value):
    return value if value in value_set else replacement_value
```

[8]:
```
df['Treatment Regimen'] = df['Treatment Regimen'].apply(lambda x:␣
↪preserve_or_change(x, values_to_preserve, 'Other'))
df.head(5)
len(df['Treatment Regimen'].unique())
```

[8]: 10

[9]:
```
X = df[['Age Group', 'Race', 'Gender']]
y = df['Treatment Regimen']
```

[10]:
```
preprocessor = ColumnTransformer(
    transformers=[
            ('cat', OneHotEncoder(), ['Age Group', 'Race', 'Gender'])
        ],
        remainder='passthrough'

)
pipeline = Pipeline([
    ('preprocessor', preprocessor),
    ('classifier', LogisticRegression(multi_class='multinomial', class_weight =␣
↪'balanced'))
])
```

[11]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[12]:
```
# Train the model
pipeline.fit(X_train, y_train)
```

[12]:
```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('cat', OneHotEncoder(),
                                                  ['Age Group', 'Race',
                                                   'Gender'])])),
                ('classifier',
                 LogisticRegression(class_weight='balanced',
                                    multi_class='multinomial'))])
```

[13]:
```
# Make predictions on the test set
y_pred = pipeline.predict(X_test)

# Evaluate the accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')

# Create a DataFrame with actual and predicted values
```

```python
df_predictions = pd.DataFrame({
    'Actual': y_test,
    'Predicted': y_pred
})

print("Actual vs Predicted:")
print(df_predictions)
```

```
Accuracy: 0.05
Actual vs Predicted:
                    Actual          Predicted
408               19071700           19030580
387               19071700           40164929
803               19071700           19077638
81                   Other           40164929
942               19071700  19071700,40164929
…                        …                  …
596      19071700,40166274           40166274
1710                 Other           19030580
894      19071700,40164929           19030580
1226                 Other           19077682
1466                 Other           40166274

[350 rows x 2 columns]
```

[14]:

```python
# Access the one-hot encoder from the pipeline
encoder = pipeline.named_steps['preprocessor'].named_transformers_['cat']

# Get feature names after one-hot encoding
feature_names_after_encoding = list(encoder.get_feature_names_out(X.
 ↪select_dtypes(include=['object']).columns))

# Concatenate feature names with numeric features
all_feature_names = X.select_dtypes(include=['number']).columns.tolist() +␣
 ↪feature_names_after_encoding

# Access the model from the pipeline
model = pipeline.named_steps['classifier']

# Get coefficients
coefficients = model.coef_

# Display coefficients in a DataFrame
df_coefficients = pd.DataFrame(coefficients, columns=all_feature_names)
df_coefficients['Intercept'] = model.intercept_
df_coefficients['Class'] = model.classes_
df_coefficients.set_index('Class', inplace=True)
```

```python
print("Coefficients:")
print(df_coefficients)
```

```
Coefficients:
                   Age Group_< 90  Age Group_Eighties  Age Group_Fifties  \
Class
19030580                 1.057495            0.993000          -0.506694
19071700                -1.062094           -0.394247           0.287233
19071700,40164929       -0.786606            0.214666          -0.931846
19071700,40166274       -1.041874           -0.856634          -0.227869
19077638                 0.423980            0.998270           0.459400
19077682                 1.143511           -0.339185           0.419265
40164929                -0.167551            0.195660          -0.050051
40164930                 0.062006            0.282591           0.372680
40166274                 0.340145           -1.130294          -0.030394
Other                    0.030988            0.036174           0.208276

                   Age Group_Forties  Age Group_Seventies  Age Group_Sixties  \
Class
19030580                   -1.109323             0.764416          -0.037044
19071700                    0.607928            -0.572851          -0.257324
19071700,40164929           0.791166             0.612755           0.984975
19071700,40166274           0.760653            -0.753746          -0.494474
19077638                   -0.110128            -0.055596          -0.455475
19077682                   -1.181039             0.802798           0.162804
40164929                    0.392291            -0.093075           0.257548
40164930                   -0.429023            -0.245508           0.214260
40166274                   -0.243459            -0.704831          -0.447023
Other                       0.520934             0.245638           0.071754

                   Age Group_Teens  Age Group_Thirties  Age Group_Twenties  \
Class
19030580                 -0.036366           -0.650484           -0.478298
19071700                 -0.105416            0.576978            0.920434
19071700,40164929        -0.017279           -0.544165           -0.324017
19071700,40166274         0.575426            0.958790            1.080776
19077638                 -0.026310           -0.756649           -0.476477
19077682                 -0.019234           -0.617050           -0.375570
40164929                 -0.094075            0.145520           -0.585676
40164930                 -0.032849            0.302301           -0.524890
40166274                 -0.188145            1.118980            1.286711
Other                    -0.055754           -0.534222           -0.522993

                   Race_Asian  Race_Black  Race_Other  Race_White  \
Class
19030580            -0.520262    0.454524   -0.591591   -0.305473
19071700             0.500714   -0.011704    0.047290   -0.094208
```

19071700,40164929	-0.603336	0.037359	-0.757630	0.975153
19071700,40166274	-0.431053	0.290041	0.531206	-0.153384
19077638	-0.932758	-0.789786	0.774966	1.146076
19077682	-0.373462	0.476923	-0.606681	-0.397857
40164929	0.334909	-0.145247	0.404472	0.141463
40164930	0.242341	-0.686904	1.209257	-0.620564
40166274	1.390883	0.505909	-1.119381	-0.562404
Other	0.392022	-0.131116	0.108091	-0.128802

	Race_White/ Hispanic	Gender_Female	Gender_Male	Intercept
Class				
19030580	0.959504	0.272659	-0.275956	-1.126304
19071700	-0.441451	0.055182	-0.054539	0.697715
19071700,40164929	0.348102	-0.313966	0.313615	-0.883987
19071700,40166274	-0.235761	0.030496	-0.029447	0.705601
19077638	-0.197483	-0.088559	0.089575	0.150103
19077682	0.897378	-0.316279	0.312579	-1.213826
40164929	-0.735007	0.093454	-0.092864	0.667195
40164930	-0.142563	0.027065	-0.025497	0.159378
40166274	-0.213319	0.257665	-0.255975	0.643686
Other	-0.239401	-0.017715	0.018510	0.200439