Model

December 3, 2023

[1]:
```
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]:
```
# Read data from Excel file into a Pandas DataFrame
file_path = 'dm_mimic_pathways.csv'
df = pd.read_csv(file_path)
```

[3]:
```
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 = 8
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
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]: 9

[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
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  19071700,40164929
1466              Other           40166274

[350 rows x 2 columns]
```

[14]:
```
# 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)

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

```
Coefficients:
                   Age Group_< 90  Age Group_Eighties  Age Group_Fifties  \
Class
19030580                 1.221850            0.940718          -0.470014
19071700                -1.005623           -0.423091           0.341582
19071700,40164929       -0.716888            0.205089          -0.879696
19071700,40166274       -0.973480           -0.894772          -0.181373
19077638                 0.628918            0.957800           0.508364
40164929                -0.059879            0.166378          -0.003426
40164930                 0.226440            0.242747           0.417392
40166274                 0.460133           -1.179861           0.005963
Other                    0.218528           -0.015007           0.261207

                   Age Group_Forties  Age Group_Seventies  Age Group_Sixties  \
Class
19030580                   -1.245313             0.888297          -0.033777
19071700                    0.482038            -0.481578          -0.228790
19071700,40164929           0.681228             0.685981           1.023210
19071700,40166274           0.625667            -0.672262          -0.469959
19077638                   -0.244680             0.009824          -0.453049
40164929                    0.271014            -0.001081           0.284914
40164930                   -0.562079            -0.165557           0.218812
40166274                   -0.373949            -0.602417          -0.428746
Other                       0.366074             0.338792           0.087385

                   Age Group_Teens  Age Group_Thirties  Age Group_Twenties  \
Class
19030580                 -0.040094           -0.730348           -0.536054
19071700                 -0.117571            0.522569            0.910890
19071700,40164929        -0.021008           -0.614192           -0.364821
19071700,40166274         0.618465            0.890861            1.057943
19077638                 -0.028563           -0.846224           -0.531772
40164929                 -0.106227            0.091291           -0.642507
40164930                 -0.037071            0.242895           -0.582258
40166274                 -0.205722            1.058222            1.267793
Other                    -0.062210           -0.615074           -0.579215

                   Race_Asian  Race_Black  Race_Other  Race_White  \
Class
19030580            -0.570367    0.478417   -0.657893   -0.343456
19071700             0.469175    0.043605   -0.018149   -0.139961
19071700,40164929   -0.669274    0.144343   -0.843712    0.943636
19071700,40166274   -0.477718    0.349604    0.473528   -0.197077
19077638            -1.002655   -0.775394    0.728701    1.136048
40164929             0.297948   -0.081222    0.338239    0.092701
40164930             0.217228   -0.636198    1.151759   -0.685871
40166274             1.381469    0.559056   -1.209698   -0.623458
Other                0.354194   -0.082212    0.037224   -0.182562
```

Class	Race_White/ Hispanic	Gender_Female	Gender_Male	Intercept
19030580	1.088565	0.264068	-0.268802	-1.283450
19071700	-0.354244	0.012536	-0.012110	0.565119
19071700,40164929	0.423910	-0.341379	0.340281	-1.006911
19071700,40166274	-0.147246	-0.009016	0.010108	0.578256
19077638	-0.086083	-0.126295	0.126913	0.016450
40164929	-0.647190	0.052801	-0.052325	0.536362
40164930	-0.045596	-0.015150	0.016472	0.034044
40166274	-0.105954	0.212931	-0.211515	0.501939
Other	-0.126162	-0.050496	0.050978	0.058192