In [1]:
```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report, roc_auc
```

```
/Users/manikshakya/anaconda3/envs/hgp/lib/python3.10/site-packages/scipy/__i
nit__.py:146: UserWarning: A NumPy version >=1.17.3 and <1.25.0 is required
for this version of SciPy (detected version 1.25.0
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:
```
# Read the dataset
df_train = pd.read_csv('mimic_train.csv')
df_test = pd.read_csv('mimic_test.csv')
```

In [3]:
```
# Check for Null values and remove the unnecessary rows from training and te
df_train.drop(['SUBJECT_ID', 'HADM_ID', 'ADMITTIME', 'DISCHTIME', 'DAYS_NEXT
               'ADMISSION_TYPE', 'DEATHTIME', 'DISCHARGE_LOCATION', 'INSURAN
               'DIAGNOSIS', 'GENDER', 'DOB'], axis=1, inplace=True)
df_test.drop(['SUBJECT_ID', 'HADM_ID', 'ADMITTIME', 'DISCHTIME', 'DAYS_NEXT_
              'ADMISSION_TYPE', 'DEATHTIME', 'DISCHARGE_LOCATION', 'INSURANC
              'DIAGNOSIS', 'GENDER', 'DOB'], axis=1, inplace=True)
```

In [4]:
```
# Transform non-numerical features into categorical type
categorical_cols = ['blood', 'circulatory', 'congenital', 'digestive', 'endo
                    'infectious', 'injury', 'mental', 'misc', 'muscular', 'n
                    'pregnancy', 'prenatal', 'respiratory', 'skin']

df_train[categorical_cols] = df_train[categorical_cols].astype('category')
df_test[categorical_cols] = df_test[categorical_cols].astype('category')
```

In [5]:
```
# Select only important features for training and testing purpose
selected_features = ['blood', 'circulatory', 'congenital', 'digestive', 'end
                     'infectious', 'injury', 'mental', 'misc', 'muscular', '
                     'pregnancy', 'prenatal', 'respiratory', 'skin']

X_train = df_train[selected_features]
y_train = df_train['OUTPUT_LABEL']

X_test = df_test[selected_features]
y_test = df_test['OUTPUT_LABEL']
```

In [6]:
```
# Re-scale the data using StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [7]:
```
# Train-test split
X_train_scaled, X_val_scaled, y_train, y_val = train_test_split(X_train_scal
```

In [8]:
```
# Train the models
models = [
    LogisticRegression(),
    RandomForestClassifier(),
    MLPClassifier()
]
```

In [9]:
```
for model in models:
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_val_scaled)
    y_pred_prob = model.predict_proba(X_val_scaled)[:, 1]

    # Evaluate the model
    print(f"Model: {type(model).__name__}")
    print("Confusion Matrix:")
    print(confusion_matrix(y_val, y_pred))
    print("Classification Report:")
    print(classification_report(y_val, y_pred))
    print("AUC Score:", roc_auc_score(y_val, y_pred_prob))
    print("--------------------------------------------------")
```

```
Model: LogisticRegression
Confusion Matrix:
[[142  46]
 [ 95 117]]
Classification Report:
              precision    recall  f1-score   support

           0       0.60      0.76      0.67       188
           1       0.72      0.55      0.62       212

    accuracy                           0.65       400
   macro avg       0.66      0.65      0.65       400
weighted avg       0.66      0.65      0.64       400

AUC Score: 0.7249849458048976
--------------------------------------------------
Model: RandomForestClassifier
Confusion Matrix:
[[121  67]
 [ 65 147]]
Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.64      0.65       188
           1       0.69      0.69      0.69       212

    accuracy                           0.67       400
   macro avg       0.67      0.67      0.67       400
weighted avg       0.67      0.67      0.67       400

AUC Score: 0.7333651144118828
--------------------------------------------------
Model: MLPClassifier
Confusion Matrix:
[[120  68]
 [ 85 127]]
Classification Report:
              precision    recall  f1-score   support

           0       0.59      0.64      0.61       188
           1       0.65      0.60      0.62       212

    accuracy                           0.62       400
   macro avg       0.62      0.62      0.62       400
weighted avg       0.62      0.62      0.62       400

AUC Score: 0.6680048173424327
--------------------------------------------------
```

```
/Users/manikshakya/anaconda3/envs/hgp/lib/python3.10/site-packages/sklearn/n
eural_network/_multilayer_perceptron.py:692: ConvergenceWarning: Stochastic
Optimizer: Maximum iterations (200) reached and the optimization hasn't conv
erged yet.
  warnings.warn(
```

Compare the results and describe which classifier is performing better and why?

Based on the evaluation metrics, RandomForestClassifier outperforms the other classifiers in terms of accuracy, precision, recall, and F1-score. It achieves the highest accuracy of 0.67, indicating that it predicts the output label correctly for approximately 67% of the instances.

In [ ]: