result-22-05-2024

May 22, 2024

```
[ ]: import sys
     import os

     sys.path.append(os.path.abspath(os.path.join('..')))
```

```
[ ]: import src.data.organize_data as od

     features_list = ['t0', 't1', 't2', 'a0', 'a1', 'a2', 'b0', 'b1', 'b2', 'c0',␣
      ↪'c1', 'c2', 'dicnotch', 'winSys', 'maxAmpl', 'sysTime', 'duration',  'Es',␣
      ↪'As', 'Ed', 'Ad', 'td', 'fd', 'R2_of_fit']
     feature_data_path  = r"C:
      ↪\Users\cical\Documents\GitHub\Repositories\tesina\data\interim\feature_extracted"

     feature_df = od.organize_data2(feature_data_path, features_list)
```

1 Data Pre-Processing

1.1 Split Dataset

```
[ ]: import src.data.organize_data as od

     X_train, X_test, y_train, y_test = od.split_train_test(feature_df, 'Group', 0.2)
```

```
Group
```

1.2 Data selection (outliers and p-value)

```
[ ]: import src.data.data_selection as ds

     X_train, y_train = ds.filter_outliers_by_group(X_train, y_train,␣
      ↪features_to_ignore=['PatientID', 'SignalID', 'R2_of_fit', 'pulse_index'])

     X_train, y_train = ds.filter_fit_value(X_train, y_train, 0.9, 'R2_of_fit')
```

```
INFO:root:Removed 695 rows for label covid_Empoli_60
INFO:root:Removed 987 rows for label healthyControl_Empoli_60
INFO:root:Removed 6257 rows for label mentalDisorders_MIMIC_125
INFO:root:Removed 4779 rows for label sepsis_MIMIC_125
```

1.3 Data Trasformation

1.3.1 Patient median

[ ]:
```
import src.data.data_preprocessing as dp

X_train, y_train = dp.calculate_patient_median(X_train, y_train, 'SignalID',␣
 ↪features_list)
X_test, y_test = dp.calculate_patient_median(X_test, y_test, 'SignalID',␣
 ↪features_list)


X_train.drop(columns=['SignalID',  'R2_of_fit'], inplace=True)
X_test.drop(columns=['SignalID',   'R2_of_fit'], inplace=True)
```

1.3.2 SMOTE+EEN

[ ]:
```
import src.data.data_preprocessing as dp
import src.visualization.visualize as vis

#vis.plot_class_distribution(y_train, title="Distribution before SMOTEEN")

#X_train, y_train = dp.balance_dataset(X_train, y_train)

#vis.plot_class_distribution(y_train, title="Distribution after SMOTEEN")
```

1.3.3 Scaling

[ ]:
```
import src.data.data_preprocessing as dp
import src.visualization.visualize as vis

X_train = dp.scale_numeric_features(X_train)
X_test = dp.scale_numeric_features(X_test)

df = X_train.copy()
df['Group'] = y_train
```

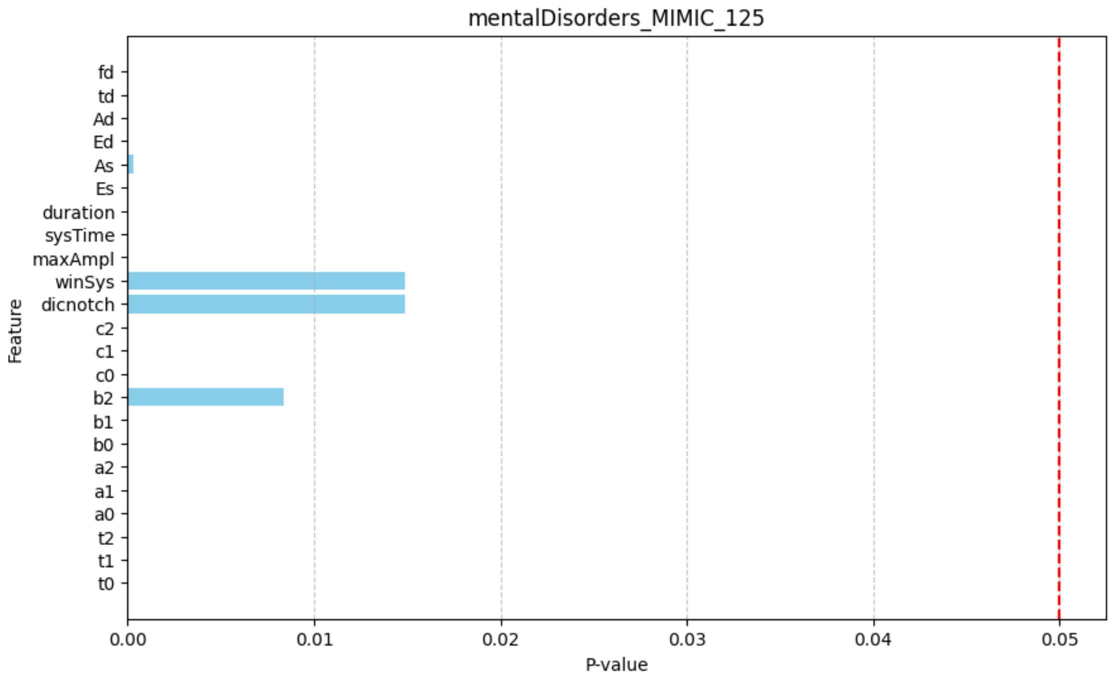
2 Statistical tests

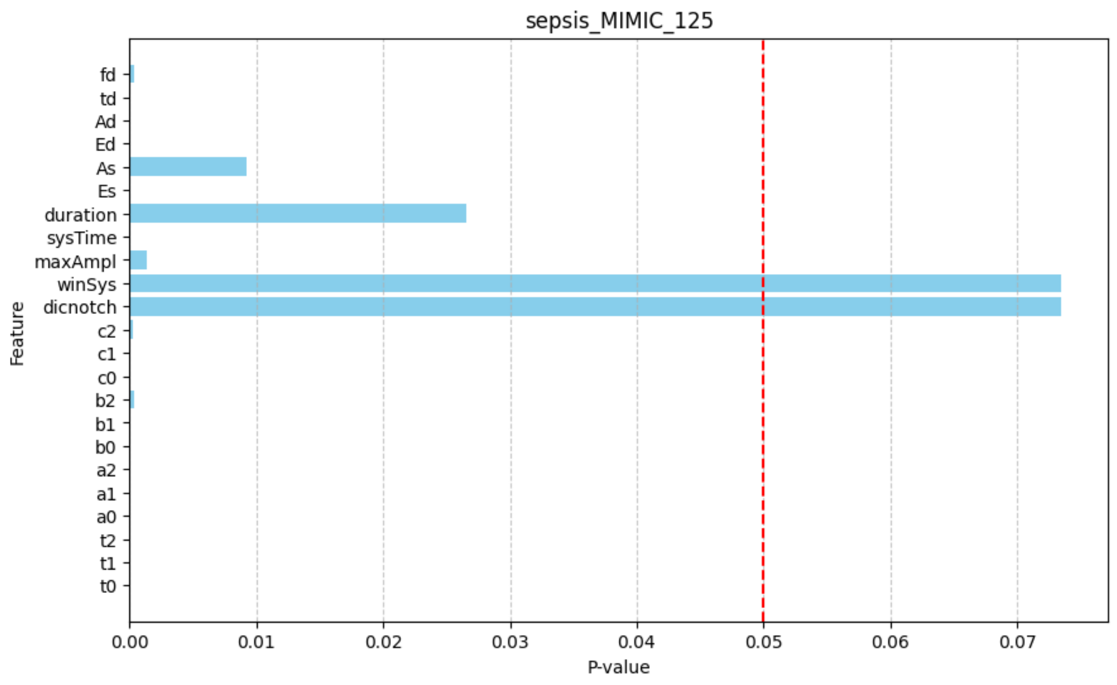
2.1 Shapiro-Wilk test

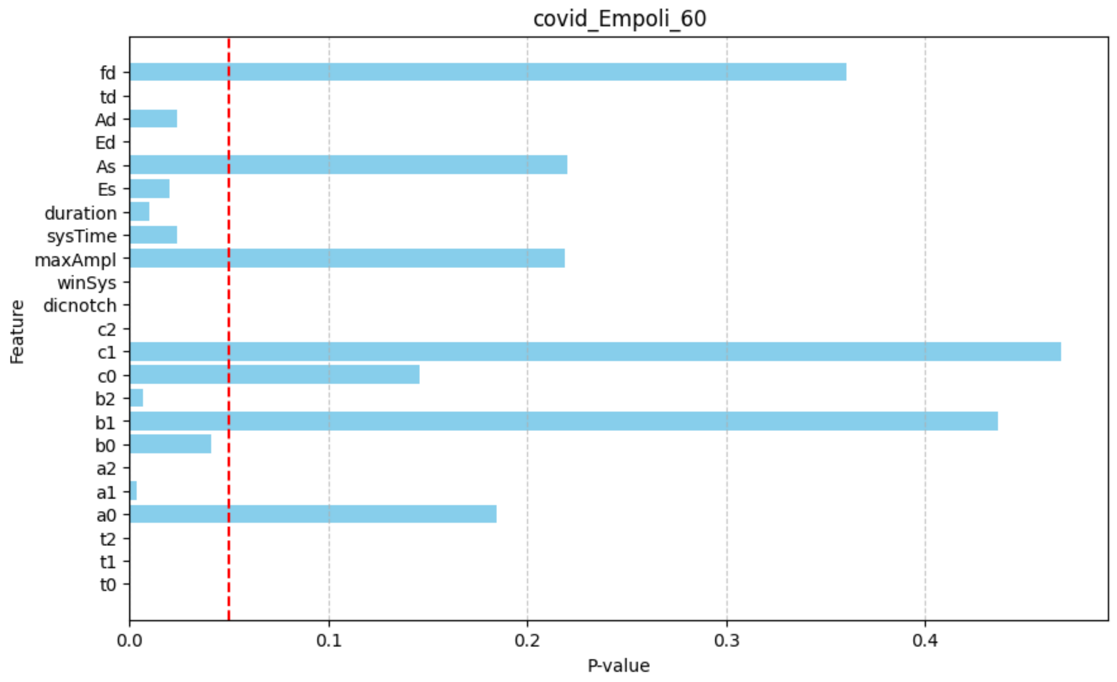
[ ]:
```
import src.statistics.tests_normality as nt
import src.visualization.visualize as vis

normality_test = nt.shapiro_test(X_train, y_train)

for group in normality_test['Group'].unique():
    sub_df = normality_test[normality_test['Group'] == group]
```

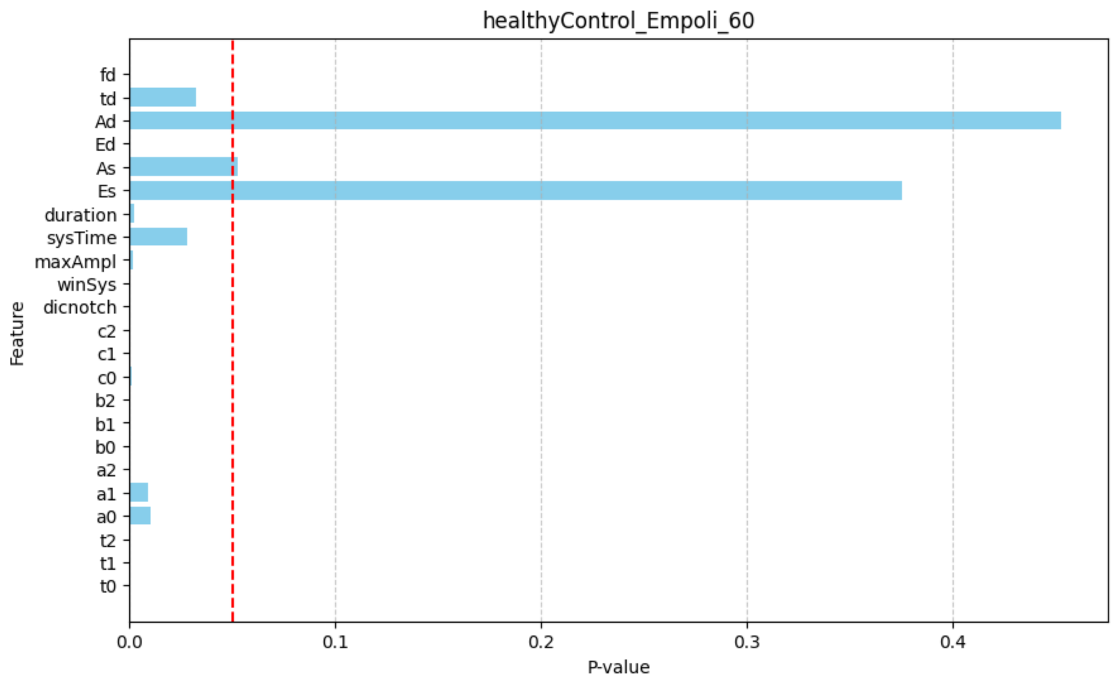
```
vis.plot_pvalues(sub_df, group)
```

```
c:\Users\cical\Documents\GitHub\Repositories\tesina\src\statistics\tests_normali
ty.py:14: FutureWarning: The behavior of DataFrame concatenation with empty or
all-NA entries is deprecated. In a future version, this will no longer exclude
empty or all-NA columns when determining the result dtypes. To retain the old
behavior, exclude the relevant entries before the concat operation.
  result_df = pd.concat([result_df, pd.DataFrame([{'Feature': feature, 'Group':
group, 'Statistic': stat, 'P-value': p_value}])], ignore_index=True)
```




sepsis_MIMIC_125
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
P-value

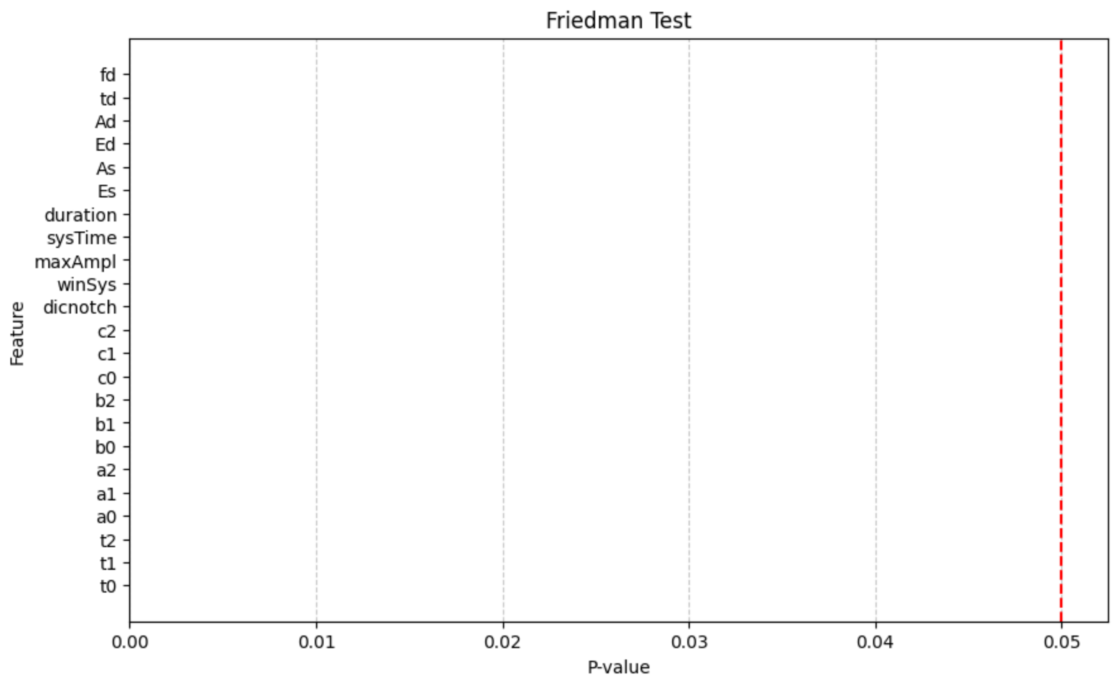

covid_Empoli_60
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.0
0.1
0.2
0.3
0.4
P-value



2.2 Friedman test

```
[ ]: import src.statistics.tests_difference as td
     import src.visualization.visualize as vis

     friedman_test = td.friedman_test(X_train, y_train)

     vis.plot_pvalues(friedman_test, 'Friedman Test')
```

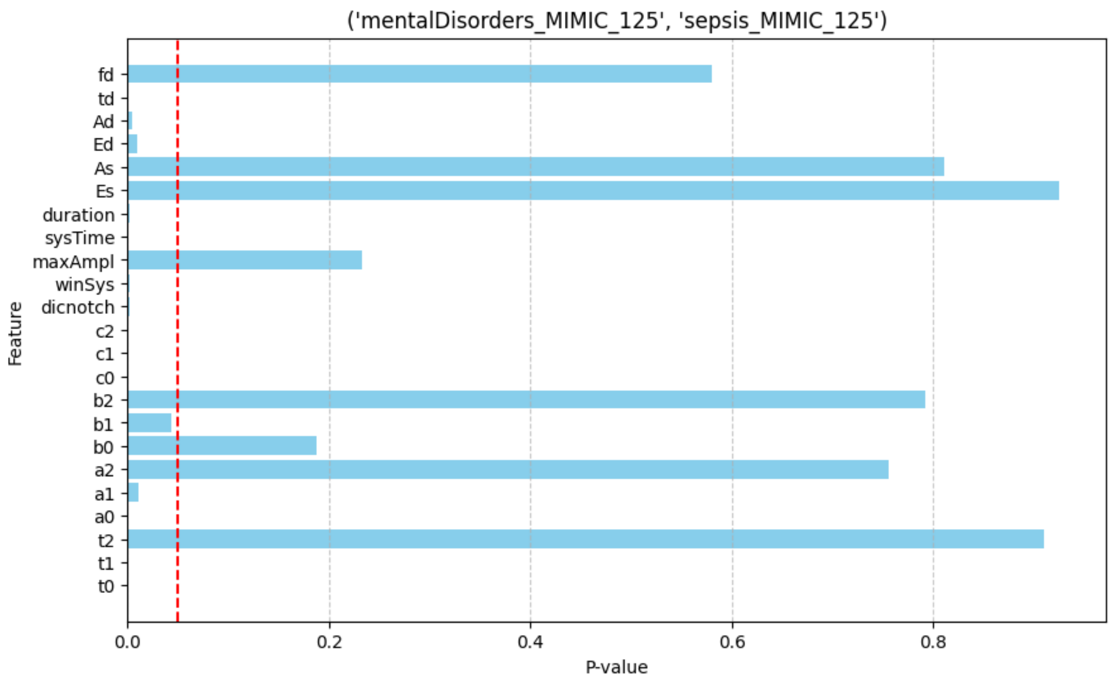
```
c:\Users\cical\Documents\GitHub\Repositories\tesina\src\statistics\tests_differe
nce.py:25: FutureWarning: The behavior of DataFrame concatenation with empty or
all-NA entries is deprecated. In a future version, this will no longer exclude
empty or all-NA columns when determining the result dtypes. To retain the old
behavior, exclude the relevant entries before the concat operation.
  result_df = pd.concat([result_df, pd.DataFrame([{'Feature': feature, 'Group':
group, 'Statistic': f_statistic, 'P-value': p_value}])], ignore_index=True)
```

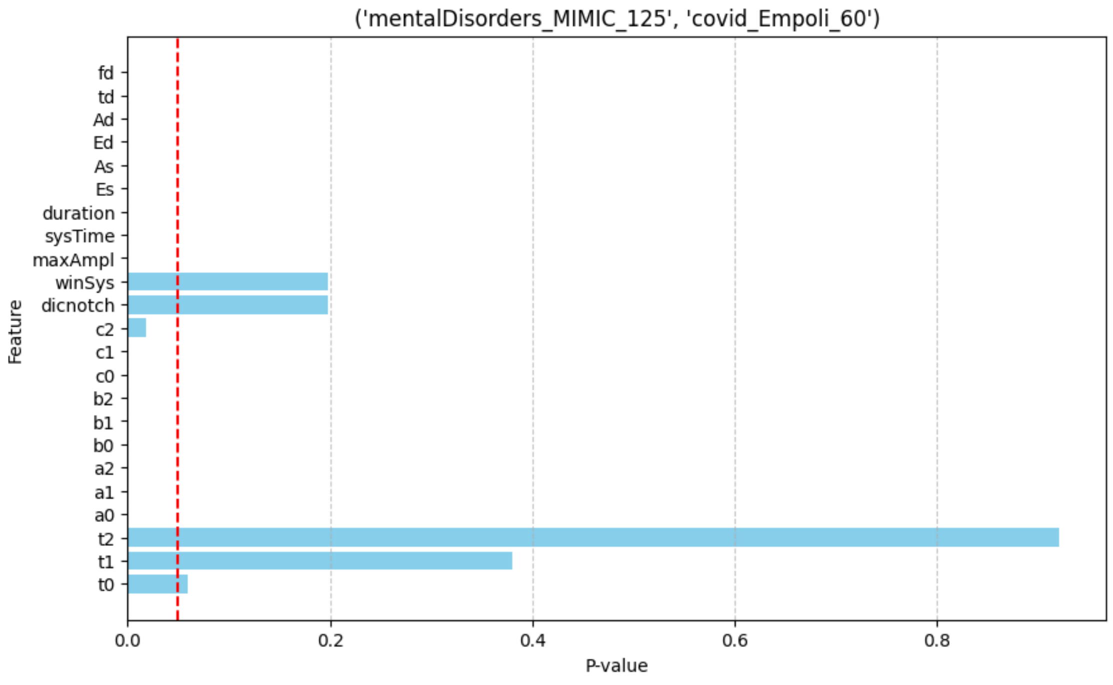


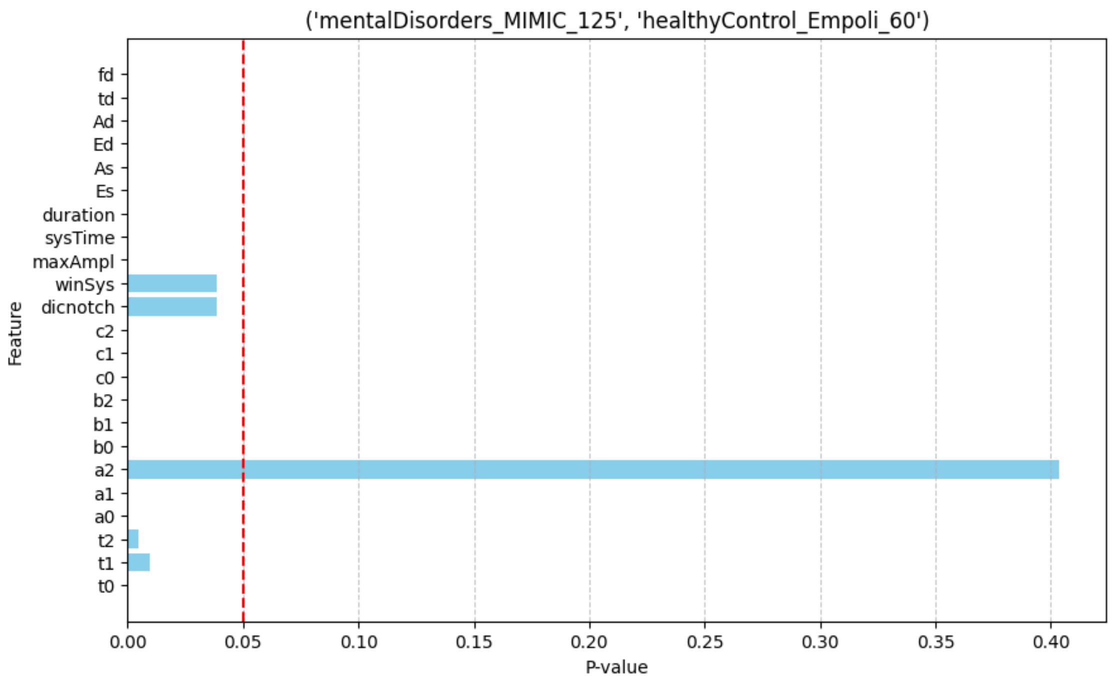
2.3 Post-hoc Mann-Whitney U

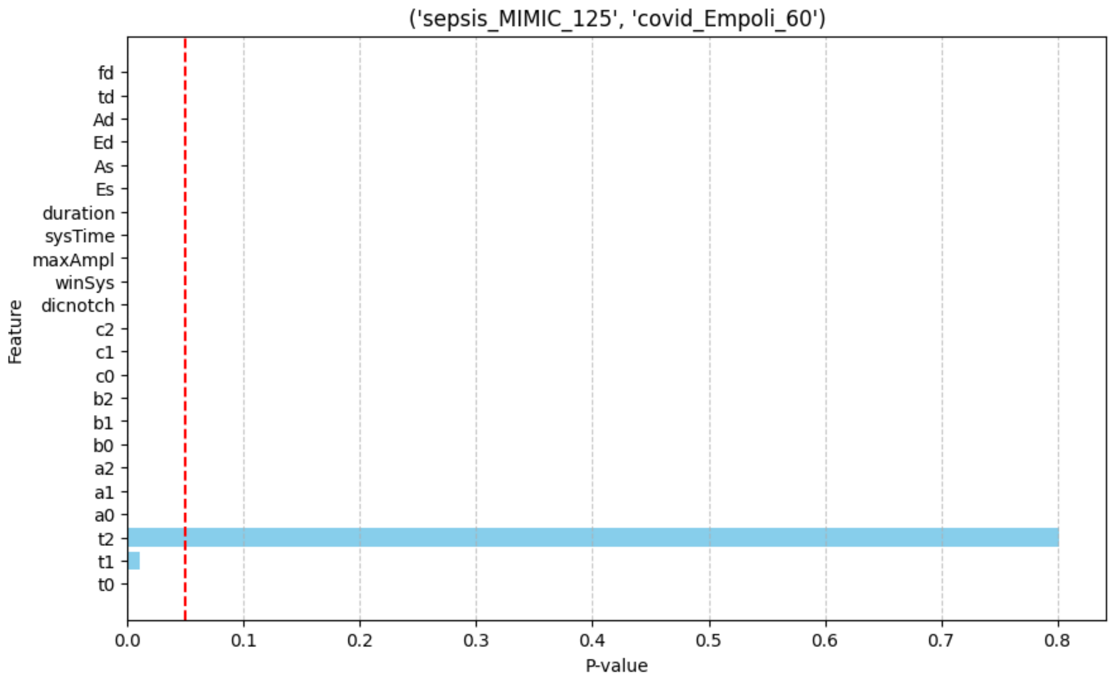
[ ]:
```
import src.statistics.tests_difference as td

# valuto differenze tra gruppi per le features significative del test di␣
↪Friedman
significant_features = friedman_test[friedman_test['P-value'] < 0.
↪05]['Feature'].tolist()
significant_features_dict = td.pairwise_mann_whitney_test(X_train, y_train,␣
↪significant_features=significant_features)
```

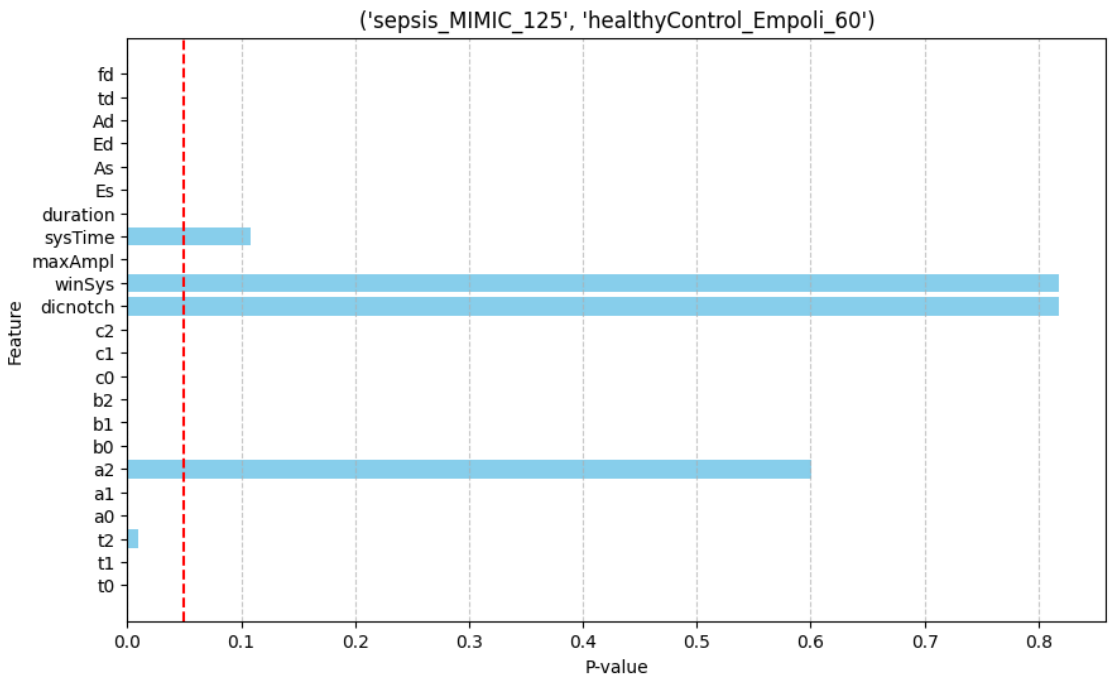
[ ]:
```
import src.visualization.visualize as vis

for group_pair, result in significant_features_dict.items():
    vis.plot_pvalues(result, group_pair)
```

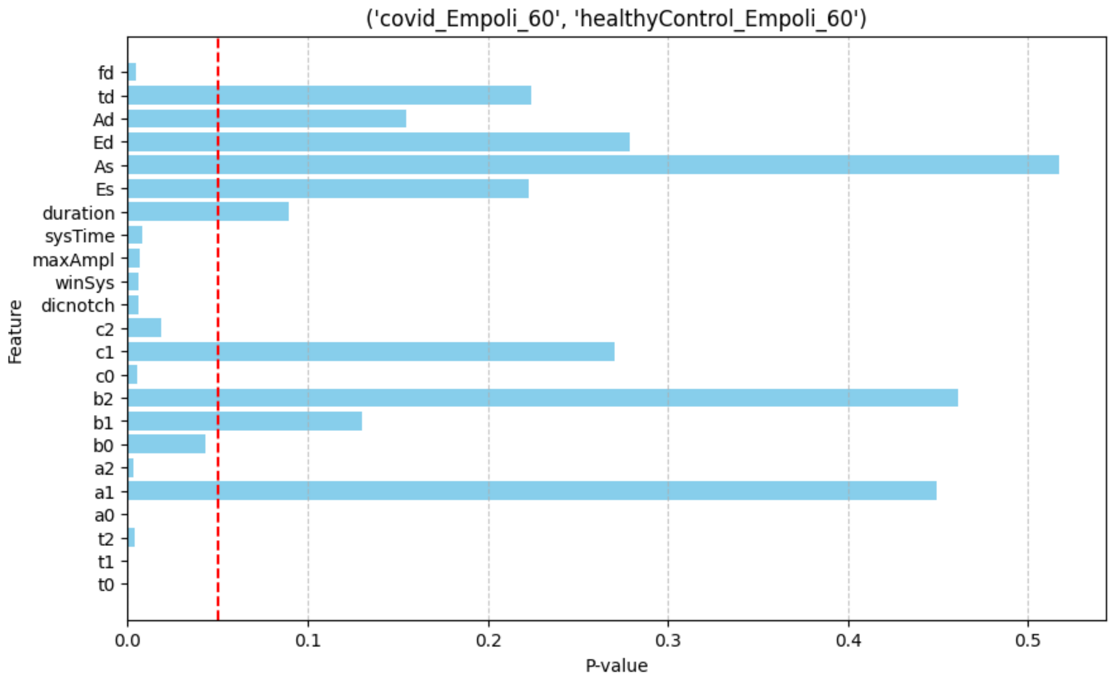

('mentalDisorders_MIMIC_125', 'sepsis_MIMIC_125')
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.0
0.2
0.4
0.6
0.8
P-value


('mentalDisorders_MIMIC_125', 'covid_Empoli_60')
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.0
0.2
0.4
0.6
0.8
P-value


('mentalDisorders_MIMIC_125', 'healthyControl_Empoli_60')
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
P-value


('sepsis_MIMIC_125', 'covid_Empoli_60')
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
P-value


('sepsis_MIMIC_125', 'healthyControl_Empoli_60')
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
P-value

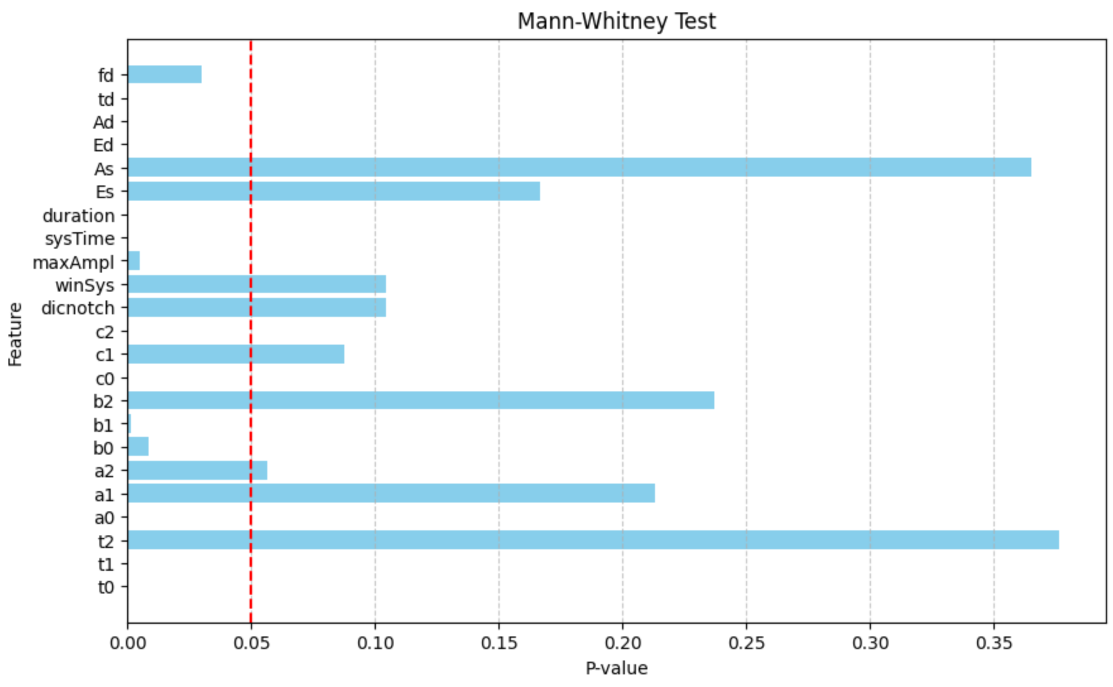

('covid_Empoli_60', 'healthyControl_Empoli_60')
fd
td
Ad
Ed
As
Es
duration
sysTime
maxAmpl
winSys
dicnotch
c2
c1
c0
b2
b1
b0
a2
a1
a0
t2
t1
t0
Feature
0.0
0.1
0.2
0.3
0.4
0.5
P-value

2.4 Unified dataset

Si cercano differenze tra il gruppo di sani (health e soggetti con disturbi mentali) e il gruppo di malati (sepsi e covid)

```
[ ]: import src.statistics.tests_difference as td
     import src.visualization.visualize as vis

     y_train_unified = y_train.copy()

     # Mappa i valori della colonna 'Group' come richiesto
     mapping = {'covid_Empoli_60': "sick", 'sepsis_MIMIC_125': "sick",
                'healthyControl_Empoli_60': "healthy", 'mentalDisorders_MIMIC_125':␣
       ↪"healthy"}
     y_train_unified = y_train_unified.replace(mapping)

     unified_result = td.mann_whitney_test(X_train, y_train_unified)

     vis.plot_pvalues(unified_result, 'Mann-Whitney Test')
```

```
c:\Users\cical\Documents\GitHub\Repositories\tesina\src\statistics\tests_differe
nce.py:62: FutureWarning: The behavior of DataFrame concatenation with empty or
all-NA entries is deprecated. In a future version, this will no longer exclude
empty or all-NA columns when determining the result dtypes. To retain the old
behavior, exclude the relevant entries before the concat operation.
  results_df = pd.concat([results_df, pd.DataFrame([{'Feature': feature,
'Group': group, 'Statistic': u_statistic, 'P-value': p_value}])],
ignore_index=True)
```



3 Training Model

3.1 Primo test (Health vs Ill)

Nel primo test si cerca di addestrare un modello che permetta di identificare tra gruppo di sani (heamth and mental disorders) e patologici (covid o sepsi). A questo scopo vengono utilizzate le features con un valore di p-value al di sotto della soglia impostata del test di mann-Whitney per il dataset unificato.

```
[ ]: features_test_1 = unified_result[unified_result['P-value'] < 0.05]['Feature'].
      ↪tolist()

     X_train_t1 = X_train[features_test_1]
     X_test_t1 = X_test[features_test_1]

     y_train_t1 = y_train.map({'covid_Empoli_60': 'ill', 'sepsis_MIMIC_125': 'ill',
                               'healthyControl_Empoli_60': 'health',␣
      ↪'mentalDisorders_MIMIC_125': 'health'})

     y_test_t1 = y_test.map({'covid_Empoli_60': 'ill', 'sepsis_MIMIC_125': 'ill',
                               'healthyControl_Empoli_60': 'health',␣
      ↪'mentalDisorders_MIMIC_125': 'health'})
```

3.1.1 Dimensionality reduction

```
[ ]: import src.features.dimensionality_reduction as dr

     X_train_t1_reduced, X_test_t1_reduced = dr.reduce_dimensionality(X_train_t1,␣
      ↪X_test_t1, 'PCA', n_components=2)
```

3.1.2 Cross validation

Si effettua una cross validazione con StratifiedKFold (cv=5) e si valutano le performance dei modelli per f1_macro e il coeff. di correlazioen di Matthews. I tre modelli che presentano le prestazioni migliori veranno poi migliorati con un ottizzazione degli iperparametri.

```
[ ]: import src.models.model as models
     import src.models.evaluation as ev

     models = models.define_models()
     metric_results = ev.evaluate_models(X_train_t1_reduced, y_train_t1, models)

     ev.summarize_results(metric_results)
```

```
Models Evaluation with f1_macro: 100%|      | 9/9 [00:07<00:00,  1.22it/s]
Models Evaluation with make_scorer(matthews_corrcoef,
response_method='predict'): 100%|      | 9/9 [00:03<00:00,  2.80it/s]


Metric: f1_macro
Rank=1, Name=catboost, Score=0.580 (+/- 0.017)
Rank=2, Name=svm, Score=0.579 (+/- 0.050)
Rank=3, Name=adaboost, Score=0.578 (+/- 0.011)
Rank=4, Name=nb, Score=0.575 (+/- 0.046)
Rank=5, Name=nc, Score=0.568 (+/- 0.031)
Rank=6, Name=rf, Score=0.564 (+/- 0.023)
Rank=7, Name=gbm, Score=0.564 (+/- 0.035)
Rank=8, Name=mlp, Score=0.561 (+/- 0.027)
Rank=9, Name=dt, Score=0.543 (+/- 0.046)

Metric: make_scorer(matthews_corrcoef, response_method='predict')
Rank=1, Name=svm, Score=0.271 (+/- 0.113)
Rank=2, Name=catboost, Score=0.216 (+/- 0.048)
Rank=3, Name=adaboost, Score=0.213 (+/- 0.023)
Rank=4, Name=mlp, Score=0.211 (+/- 0.082)
Rank=5, Name=nb, Score=0.211 (+/- 0.103)
Rank=6, Name=gbm, Score=0.173 (+/- 0.095)
Rank=7, Name=rf, Score=0.151 (+/- 0.040)
Rank=8, Name=nc, Score=0.139 (+/- 0.059)
Rank=9, Name=dt, Score=0.078 (+/- 0.097)
```

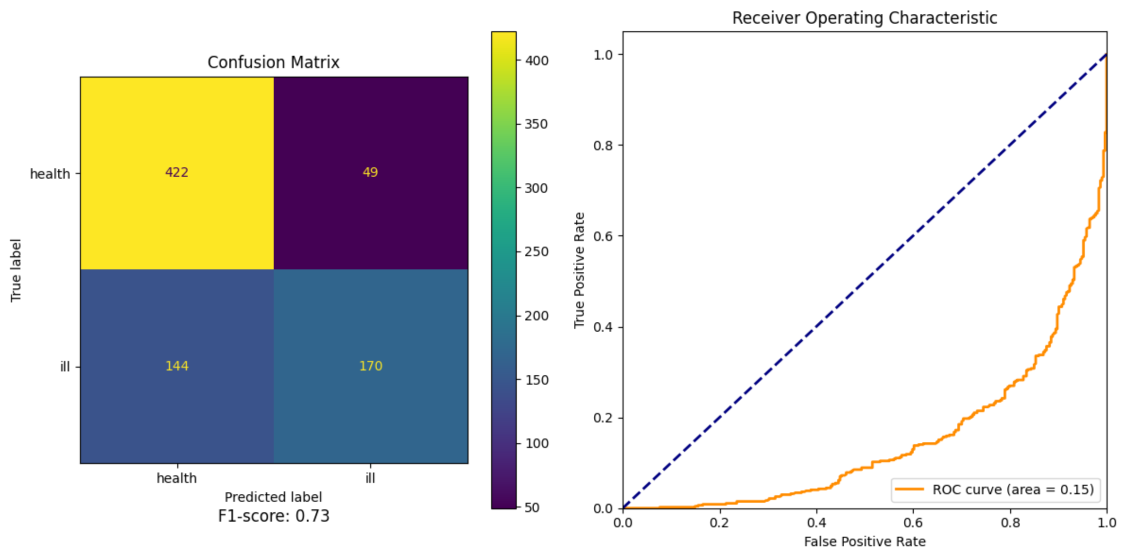
[ ]:
```
import src.models.evaluation as ev

optimal_parameters_t1 = ev.evaluate_optimized_models(X_train_t1_reduced,␣
 ↪y_train_t1, metric_results, metric='f1_macro')
```

```
Best hyperparameters for model catboost: {'iterations': 200, 'learning_rate':
0.2}
Model catboost - Score=0.541 (+/- 0.033)
---------------------------------------------------------------
Best hyperparameters for model svm: {'C': 1, 'gamma': 0.1, 'kernel': 'rbf',
'probability': True}
Model svm - Score=0.580 (+/- 0.050)
---------------------------------------------------------------

c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\ensemble\_weight_boosting.py:519: FutureWarning: The SAMME.R
algorithm (the default) is deprecated and will be removed in 1.6. Use the SAMME
algorithm to circumvent this warning.
  warnings.warn(
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\ensemble\_weight_boosting.py:519: FutureWarning: The SAMME.R
```

```
algorithm (the default) is deprecated and will be removed in 1.6. Use the SAMME
algorithm to circumvent this warning.
  warnings.warn(

Best hyperparameters for model adaboost: {'learning_rate': 0.5, 'n_estimators':
200}
Model adaboost - Score=0.570 (+/- 0.011)
---------------------------------------------------------------
```

3.1.3 Training and test fine tuned model

```
[ ]: import src.models.model as md
     import src.visualization.visualize as vis

     trained_model_t1 = md.train_model_with_optimal_params('svm',␣
      ↪optimal_parameters_t1, X_train_t1_reduced, y_train_t1)

     vis.plot_model_performance(trained_model_t1, X_test_t1_reduced, y_test_t1,␣
      ↪'f1-score')
```



3.2 Secondo test (ill)

In questo caso viene addestrato un modello che deve riconoscere i pazienti affetti da covid da quelli affetti da sepsi

```
[ ]: import src.data.data_selection as ds

     try:
```

```
    result_statistical_ill = significant_features_dict[('covid_Empoli_60',␣
  ↪'sepsis_MIMIC_125')]
except KeyError:
    result_statistical_ill = significant_features_dict[('sepsis_MIMIC_125',␣
  ↪'covid_Empoli_60')]

features_test_2 = result_statistical_ill[result_statistical_ill['P-value'] < 0.
  ↪05]['Feature'].tolist()

#features_test_2 = list(set(unified_result[unified_result['P-value'] > 0.
  ↪05]['Feature'].tolist()).intersection(features_test_2))

X_train_t2 = X_train[features_test_2]
X_test_t2 = X_test[features_test_2]

# si rimuovono le righe relative ai gruppi 'healthyControl_Empoli_60' e␣
  ↪'mentalDisorders_MIMIC_125'
target_values = ['covid_Empoli_60', 'sepsis_MIMIC_125']
X_train_t2, y_train_t2 = ds.filter_rows_by_values(X_train_t2, y_train,␣
  ↪target_values)
X_test_t2, y_test_t2 = ds.filter_rows_by_values(X_test_t2, y_test,␣
  ↪target_values)
```

3.2.1 Dimensionality Reduction

```
[ ]: import src.features.dimensionality_reduction as dr

     X_train_t2_reduced, X_test_t2_reduced = dr.reduce_dimensionality(X_train_t2,␣
       ↪X_test_t2, 'PCA', n_components=2)
```

3.2.2 Cross-validation

```
[ ]: import src.models.model as models
     import src.models.evaluation as ev

     models = models.define_models()
     metric_results = ev.evaluate_models(X_train_t2_reduced, y_train_t2, models)

     ev.summarize_results(metric_results)
```

```
Models Evaluation with f1_macro: 100%|      | 9/9 [00:02<00:00,  4.20it/s]
Models Evaluation with make_scorer(matthews_corrcoef,
response_method='predict'): 100%|      | 9/9 [00:02<00:00,  4.28it/s]


Metric: f1_macro
Rank=1, Name=mlp, Score=0.878 (+/- 0.044)
Rank=2, Name=rf, Score=0.875 (+/- 0.073)
```

```
Rank=3, Name=adaboost, Score=0.871 (+/- 0.097)
Rank=4, Name=catboost, Score=0.860 (+/- 0.073)
Rank=5, Name=svm, Score=0.856 (+/- 0.078)
Rank=6, Name=nb, Score=0.856 (+/- 0.075)
Rank=7, Name=gbm, Score=0.829 (+/- 0.103)
Rank=8, Name=dt, Score=0.828 (+/- 0.084)
Rank=9, Name=nc, Score=0.808 (+/- 0.054)

Metric: make_scorer(matthews_corrcoef, response_method='predict')
Rank=1, Name=adaboost, Score=0.755 (+/- 0.181)
Rank=2, Name=mlp, Score=0.753 (+/- 0.102)
Rank=3, Name=rf, Score=0.752 (+/- 0.149)
Rank=4, Name=catboost, Score=0.734 (+/- 0.137)
Rank=5, Name=svm, Score=0.731 (+/- 0.145)
Rank=6, Name=nb, Score=0.729 (+/- 0.139)
Rank=7, Name=gbm, Score=0.674 (+/- 0.191)
Rank=8, Name=dt, Score=0.664 (+/- 0.169)
Rank=9, Name=nc, Score=0.632 (+/- 0.108)
```

```
[ ]: import src.models.evaluation as ev

     optimal_parameters_t2 = ev.evaluate_optimized_models(X_train_t2_reduced,␣
      ↪y_train_t2, metric_results, metric='f1_macro', cv=5)
```

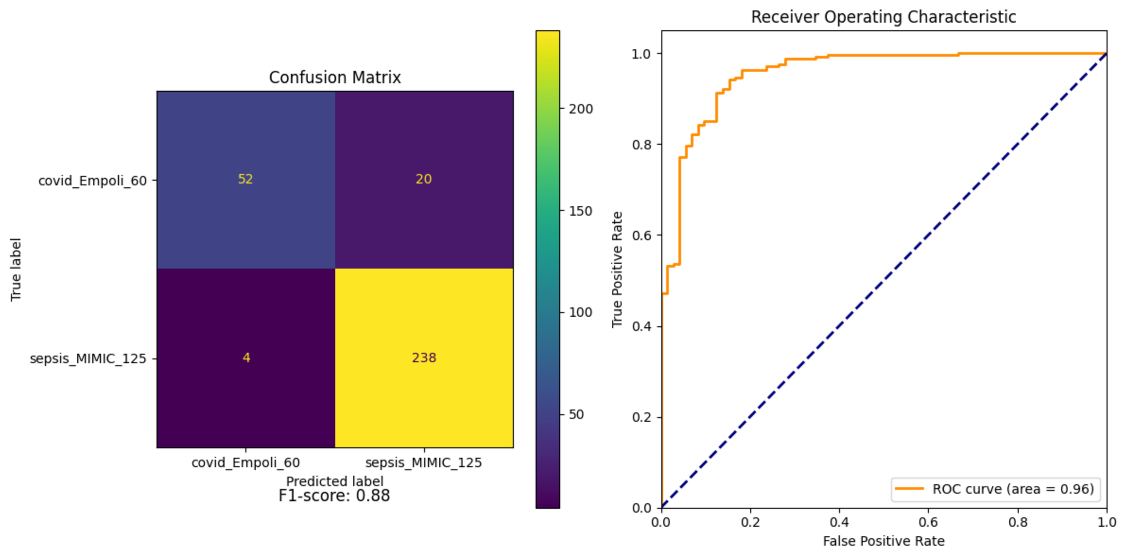
```
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(

Best hyperparameters for model mlp: {'alpha': 0.001, 'hidden_layer_sizes':
(50,)}
Model mlp - Score=0.877 (+/- 0.044)
---------------------------------------------------------------
Best hyperparameters for model rf: {'max_depth': 10, 'min_samples_leaf': 2,
'min_samples_split': 5, 'n_estimators': 100}
Model rf - Score=0.861 (+/- 0.088)
---------------------------------------------------------------

c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\ensemble\_weight_boosting.py:519: FutureWarning: The SAMME.R
algorithm (the default) is deprecated and will be removed in 1.6. Use the SAMME
algorithm to circumvent this warning.
```

```
  warnings.warn(

Best hyperparameters for model adaboost: {'learning_rate': 0.5, 'n_estimators':
50}
Model adaboost - Score=0.871 (+/- 0.084)
---------------------------------------------------------------

c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\ensemble\_weight_boosting.py:519: FutureWarning: The SAMME.R
algorithm (the default) is deprecated and will be removed in 1.6. Use the SAMME
algorithm to circumvent this warning.
  warnings.warn(
```

3.2.3 Training and test fine tuned model

```
[ ]: import src.models.model as md
     import src.visualization.visualize as vis

     trained_model = md.train_model_with_optimal_params('mlp',␣
      ↪optimal_parameters_t2, X_train_t2_reduced, y_train_t2)

     vis.plot_model_performance(trained_model, X_test_t2_reduced, y_test_t2,␣
      ↪'f1-score')
```

```
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(
```



3.3 Test 3 (Health)

Viene addestrato un modello con lo scopo di identificare i pazienti sani da quelli con disturbi mentali

```
[ ]: import src.data.data_selection as ds

result_statistical_health =␣
 ↪significant_features_dict[('mentalDisorders_MIMIC_125',␣
 ↪'healthyControl_Empoli_60')]
features_test_3 =␣
 ↪result_statistical_health[result_statistical_health['P-value'] < 0.
 ↪05]['Feature'].tolist()

#features_test_3 = list(set(unified_result[unified_result['P-value'] > 0.
 ↪05]['Feature'].tolist()).intersection(features_test_3))

X_train_t3 = X_train[features_test_3]
X_test_t3 = X_test[features_test_3]

# si rimuovono le righe relative ai gruppi 'covid_Empoli_60' e␣
 ↪'sepsis_MIMIC_125'
target_values = ['mentalDisorders_MIMIC_125', 'healthyControl_Empoli_60']
X_train_t3, y_train_t3 = ds.filter_rows_by_values(X_train_t3, y_train,␣
 ↪target_values)
X_test_t3, y_test_t3 = ds.filter_rows_by_values(X_test_t3, y_test,␣
 ↪target_values)
```

3.3.1 Dimensionality reduction

```
[ ]: import src.features.dimensionality_reduction as dr

X_train_t3_reduced, X_test_t3_reduced = dr.reduce_dimensionality(X_train_t3,␣
 ↪X_test_t3, 'PCA', n_components=2)
```

3.3.2 Cross-validation

```
[ ]: import src.models.model as models
import src.models.evaluation as ev

models = models.define_models()
metric_results = ev.evaluate_models(X_train_t3_reduced, y_train_t3, models)

ev.summarize_results(metric_results)
```

```
Models Evaluation with f1_macro: 100%|      | 9/9 [00:02<00:00,  4.10it/s]
Models Evaluation with make_scorer(matthews_corrcoef,
response_method='predict'): 100%|      | 9/9 [00:02<00:00,  4.24it/s]
```

```
Metric: f1_macro
Rank=1, Name=svm, Score=0.932 (+/- 0.022)
Rank=2, Name=mlp, Score=0.913 (+/- 0.025)
Rank=3, Name=catboost, Score=0.911 (+/- 0.028)
Rank=4, Name=gbm, Score=0.906 (+/- 0.032)
Rank=5, Name=rf, Score=0.892 (+/- 0.039)
Rank=6, Name=dt, Score=0.878 (+/- 0.030)
Rank=7, Name=nc, Score=0.875 (+/- 0.044)
Rank=8, Name=adaboost, Score=0.870 (+/- 0.062)
Rank=9, Name=nb, Score=0.835 (+/- 0.021)

Metric: make_scorer(matthews_corrcoef, response_method='predict')
Rank=1, Name=svm, Score=0.868 (+/- 0.044)
Rank=2, Name=mlp, Score=0.844 (+/- 0.055)
Rank=3, Name=catboost, Score=0.828 (+/- 0.058)
Rank=4, Name=gbm, Score=0.816 (+/- 0.063)
Rank=5, Name=dt, Score=0.796 (+/- 0.032)
Rank=6, Name=rf, Score=0.783 (+/- 0.063)
Rank=7, Name=nc, Score=0.762 (+/- 0.087)
Rank=8, Name=adaboost, Score=0.746 (+/- 0.124)
Rank=9, Name=nb, Score=0.679 (+/- 0.049)
```

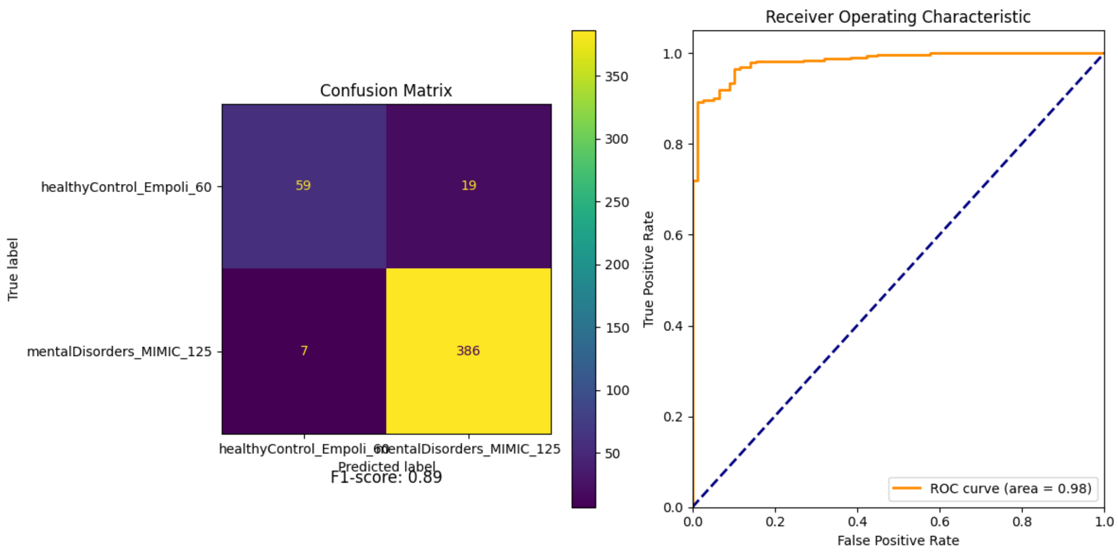
```
[ ]: import src.models.evaluation as ev

     optimal_parameters_t3 = ev.evaluate_optimized_models(X_train_t3_reduced,␣
      ↪y_train_t3, metric_results, metric='f1_macro', cv=5)
```

```
Best hyperparameters for model svm: {'C': 1, 'gamma': 0.1, 'kernel': 'rbf',
'probability': True}
Model svm - Score=0.932 (+/- 0.022)
---------------------------------------------------------------

c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(

Best hyperparameters for model mlp: {'alpha': 0.0001, 'hidden_layer_sizes':
(50,)}
Model mlp - Score=0.921 (+/- 0.027)
---------------------------------------------------------------
```

```
Best hyperparameters for model catboost: {'iterations': 100, 'learning_rate':
0.01}
Model catboost - Score=0.913 (+/- 0.020)
---------------------------------------------------------------
```

3.3.3 Training e test fine tuned model

```
[ ]: import src.models.model as md
     import src.visualization.visualize as vis

     trained_model = md.train_model_with_optimal_params('svm',␣
      ↪optimal_parameters_t3, X_train_t3_reduced, y_train_t3)

     vis.plot_model_performance(trained_model, X_test_t3_reduced, y_test_t3,␣
      ↪'f1-score')
```



3.4 Test 4 (All)

```
[ ]: features_test_4 = friedman_test[friedman_test['P-value'] < 0.05]['Feature'].
      ↪tolist()

     X_train_t4 = X_train[features_test_4]
     X_test_t4 = X_test[features_test_4]

     y_train_t4 = y_train.copy()
     y_test_t4 = y_test.copy()
```

3.4.1 Dimensionality Reduction

```
[ ]: import src.features.dimensionality_reduction as dr

     X_train_t4_reduced, X_test_t4_reduced = dr.reduce_dimensionality(X_train_t4,␣
      ↪X_test_t4, 'PCA', n_components=2)
```

3.4.2 Cross-validation

```
[ ]: import src.models.model as models
     import src.models.evaluation as ev

     models = models.define_models()
     metric_results = ev.evaluate_models(X_train_t4_reduced, y_train_t4, models)

     ev.summarize_results(metric_results)
```

```
Models Evaluation with f1_macro: 100%|      | 9/9 [00:03<00:00,  2.30it/s]
Models Evaluation with make_scorer(matthews_corrcoef,
response_method='predict'): 100%|      | 9/9 [00:03<00:00,  2.32it/s]


Metric: f1_macro
Rank=1, Name=mlp, Score=0.461 (+/- 0.057)
Rank=2, Name=catboost, Score=0.448 (+/- 0.047)
Rank=3, Name=nc, Score=0.439 (+/- 0.073)
Rank=4, Name=nb, Score=0.427 (+/- 0.040)
Rank=5, Name=rf, Score=0.426 (+/- 0.036)
Rank=6, Name=svm, Score=0.425 (+/- 0.030)
Rank=7, Name=adaboost, Score=0.424 (+/- 0.014)
Rank=8, Name=gbm, Score=0.401 (+/- 0.025)
Rank=9, Name=dt, Score=0.399 (+/- 0.030)

Metric: make_scorer(matthews_corrcoef, response_method='predict')
Rank=1, Name=mlp, Score=0.350 (+/- 0.029)
Rank=2, Name=svm, Score=0.327 (+/- 0.034)
Rank=3, Name=nb, Score=0.269 (+/- 0.044)
Rank=4, Name=catboost, Score=0.239 (+/- 0.038)
Rank=5, Name=rf, Score=0.235 (+/- 0.014)
Rank=6, Name=adaboost, Score=0.215 (+/- 0.055)
Rank=7, Name=gbm, Score=0.200 (+/- 0.050)
Rank=8, Name=nc, Score=0.190 (+/- 0.086)
Rank=9, Name=dt, Score=0.172 (+/- 0.058)
```

```
[ ]: import src.models.evaluation as ev
```

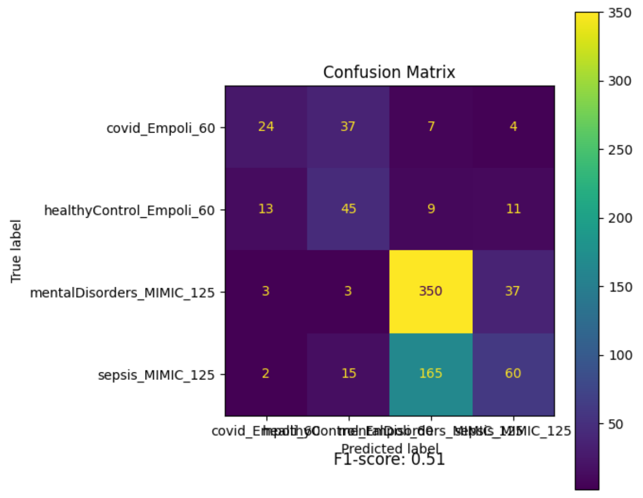
```
optimal_parameters_t4 = ev.evaluate_optimized_models(X_train_t4_reduced,␣
↪y_train_t4, metric_results, metric='f1_macro', cv=5)
```

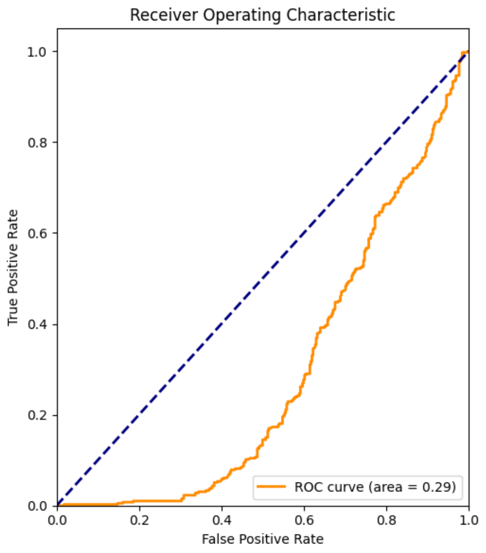
```
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(
c:\Users\cical\Documents\GitHub\Repositories\tesina\venv\Lib\site-
packages\sklearn\neural_network\_multilayer_perceptron.py:691:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  warnings.warn(
```

```
Best hyperparameters for model mlp: {'alpha': 0.001, 'hidden_layer_sizes':
(100,)}
Model mlp - Score=0.467 (+/- 0.066)
---------------------------------------------------------------
Best hyperparameters for model catboost: {'iterations': 100, 'learning_rate':
0.2}
Model catboost - Score=0.475 (+/- 0.044)
---------------------------------------------------------------
Model nc does not have hyperparameters for fine tuning
Model nc - Score=0.439 (+/- 0.073)
---------------------------------------------------------------
Best hyperparameters for model svm: {'C': 1, 'gamma': 0.1, 'kernel': 'rbf',
'probability': True}
Model svm - Score=0.432 (+/- 0.026)
---------------------------------------------------------------
Model nb does not have hyperparameters for fine tuning
Model nb - Score=0.427 (+/- 0.040)
---------------------------------------------------------------
```

3.4.3 Training and test fine tuned model

[ ]:
```
import src.models.model as md
import src.visualization.visualize as vis

trained_model = md.train_model_with_optimal_params('catboost',␣
↪optimal_parameters_t4, X_train_t4_reduced, y_train_t4)

vis.plot_model_performance(trained_model, X_test_t4_reduced, y_test_t4,␣
↪'f1-score')
```


Confusion Matrix
True label
covid_Empoli_60
healthyControl_Empoli_60
mentalDisorders_MIMIC_125
sepsis_MIMIC_125
24 37 7 4
13 45 9 11
3 3 350 37
2 15 165 60
Predicted label
F1-score: 0.51
350
300
250
200
150
100
50


Receiver Operating Characteristic
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (area = 0.29)