Machine learning project

By Nakorn Boonprasong 6510405458

Dataset

Search

FEDESORIANO · UPDATED 4 YEARS AGO

2800 | Code | Download

Heart Failure Prediction Dataset

11 clinical features for predicting heart disease events.

Data Card | Code (1327) | Discussion (28) | Suggestions (0)

About Dataset

Similar Datasets

- Hepatitis C Dataset: LINK
- Body Fat Prediction Dataset: LINK
- Cirrhosis Prediction Dataset: LINK
- Stroke Prediction Dataset: LINK
- Stellar Classification Dataset - SDSS17: LINK
- Wind Speed Prediction Dataset: LINK
- Spanish Wine Quality Dataset: LINK

Usability
10.00

License
Database: Open Database, Cont...

Expected update frequency
Never

Tags

Health | Health Conditions | Classification

Process

- ✔ EDA
- ✔ Data encoding
- ✔ Model training
- ✔ Metrics & Graphs

EDA

Exploring data

```
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Age             918 non-null    int64  
 1   Sex             918 non-null    object 
 2   ChestPainType   918 non-null    object 
 3   RestingBP       918 non-null    int64  
 4   Cholesterol     918 non-null    int64  
 5   FastingBS       918 non-null    int64  
 6   RestingECG      918 non-null    object 
 7   MaxHR           918 non-null    int64  
 8   ExerciseAngina  918 non-null    object 
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object 
 11  HeartDisease    918 non-null    int64  
dtypes: float64(1), int64(6), object(5)
```

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

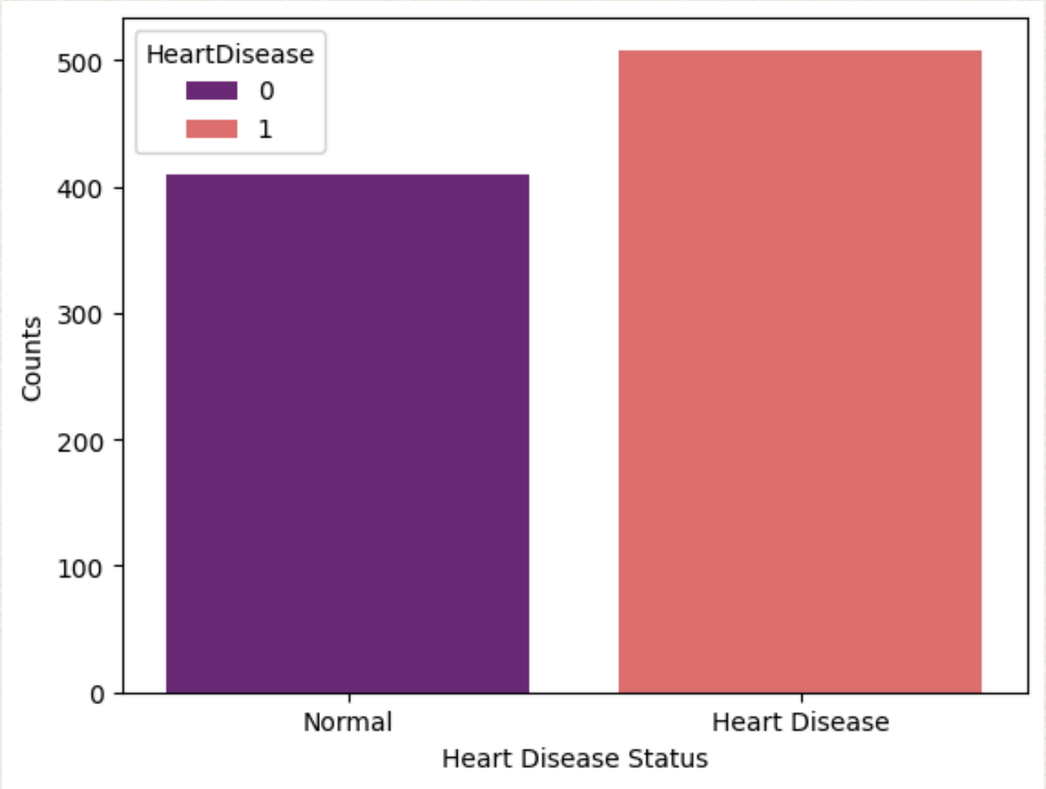
EDA

Pairplot

EDA

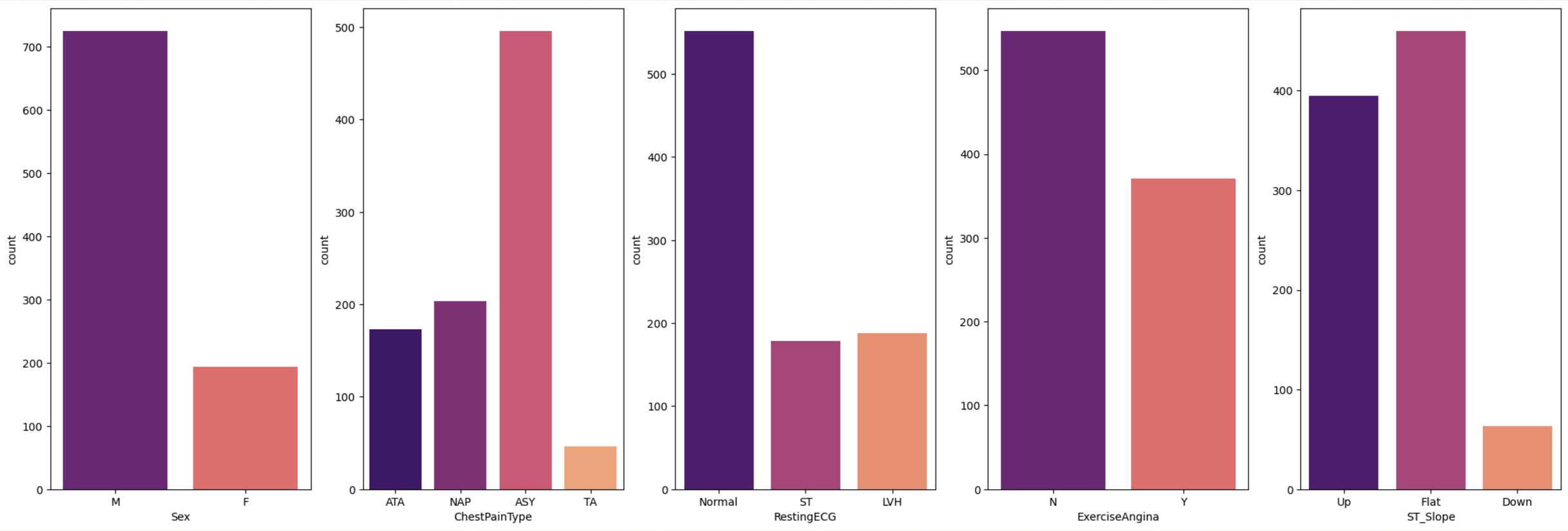
Distribution of Heart Disease Cases

- Class 0 (Normal) has 410 data points
- Class 1 (Heart Disease) has 508 data points



EDA

Distribution of Categorical Features



EDA

correlation matrix

Data Encoding

One-Hot Encoding

- **Perform One-Hot Encoding for categorical features to make data trainable**

Categorical Features

Sex	ChestPainType	RestingECG	ExerciseAngina	ST_Slope
M	ATA	Normal	N	Up
F	NAP	Normal	N	Flat
M	ATA	ST	N	Up
F	ASY	Normal	Y	Flat
M	NAP	Normal	N	Up
...	...	...	...	...
M	TA	Normal	N	Flat
M	ASY	Normal	N	Flat
M	ASY	Normal	Y	Flat
F	ATA	LVH	N	Flat
M	NAP	Normal	N	Up

```python
oneHot = OneHotEncoder(sparse_output=False)

def fit_encoder(df, columns):
    oneHot.fit(df[columns])

    return oneHot

def transform(df, columns, encoder):
    encoded_features = encoder.transform(df[columns])
    encoded_df = pd.DataFrame(encoded_features, columns=encoder.get_feature_names_out()).astype('int')

    new_df = pd.concat([df, encoded_df], axis=1)
    new_df = new_df.drop(columns, axis=1)


    return new_df
```

```python
encoder = fit_encoder(x_train, categorical_columns)

x_train_encoded = transform(x_train, categorical_columns, encoder)
```

Data Encoding

Data after the encoding process

- **All categorical features are encoded into 0 and 1 values**
- **Convert encoded value into integers**

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestPainType_ASY	ChestPainType_ATA	ChestPainType_NAP
0	61	134	234	0	145	2.6	0	1	0	0	0
1	46	138	243	0	152	0.0	1	0	1	0	0
2	65	145	0	1	67	0.7	0	1	1	0	0
3	43	110	211	0	161	0.0	0	1	1	0	0
4	56	130	167	0	114	0.0	0	1	0	0	1

ChestPainType_TA	RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ExerciseAngina_N	ExerciseAngina_Y	ST_Slope_Down	ST_Slope_Flat	ST_Slope_Up
1	0	1	0	1	0	0	1	0
0	1	0	0	0	1	0	1	0
0	0	0	1	1	0	0	1	0
0	0	1	0	1	0	0	0	1
0	0	1	0	1	0	0	0	1

```
Age                 int64
RestingBP           int64
Cholesterol         int64
FastingBS           int64
MaxHR               int64
Oldpeak           float64
Sex_F               int64
Sex_M               int64
ChestPainType_ASY   int64
ChestPainType_ATA   int64
ChestPainType_NAP   int64
ChestPainType_TA    int64
RestingECG_LVH      int64
RestingECG_Normal   int64
RestingECG_ST       int64
ExerciseAngina_N    int64
ExerciseAngina_Y    int64
ST_Slope_Down       int64
ST_Slope_Flat       int64
ST_Slope_Up         int64
dtype: object
```

Model training

Choosing a classification algorithm

- Choosing random forest as the algorithm can tackles with noisy data(Bagging).
- Tree-based model, friendly to non-linear separable data
- Parameters:
  - n_estimators: 200
  - max feature: square root of dimension number
  - max depth: 4

```
rf = RandomForestClassifier(
                                    n_estimators=200,
                                    max_features='sqrt',
                                    max_depth=4,
                                    random_state=12345,
                                    oob_score=True
)
```

Metrics

K-fold cross validation

- Use cross-validation to measure accuracy
- More reliable than the accuracy score metric
- Performing 5-Fold Cross-Validation
- Yields a mean validation score of 0.88

```
cross_val_scores = cross_val_score(rf, x_train_encoded, y_train, cv=5)

print("cross validation scores:", cross_val_scores)
print("Mean validation score:", f'{cross_val_scores.mean(): .2f}')
```

```
cross validation scores: [0.88461538 0.86538462 0.90384615 0.89102564 0.8525641 ]
Mean validation score:  0.88
```

Metrics

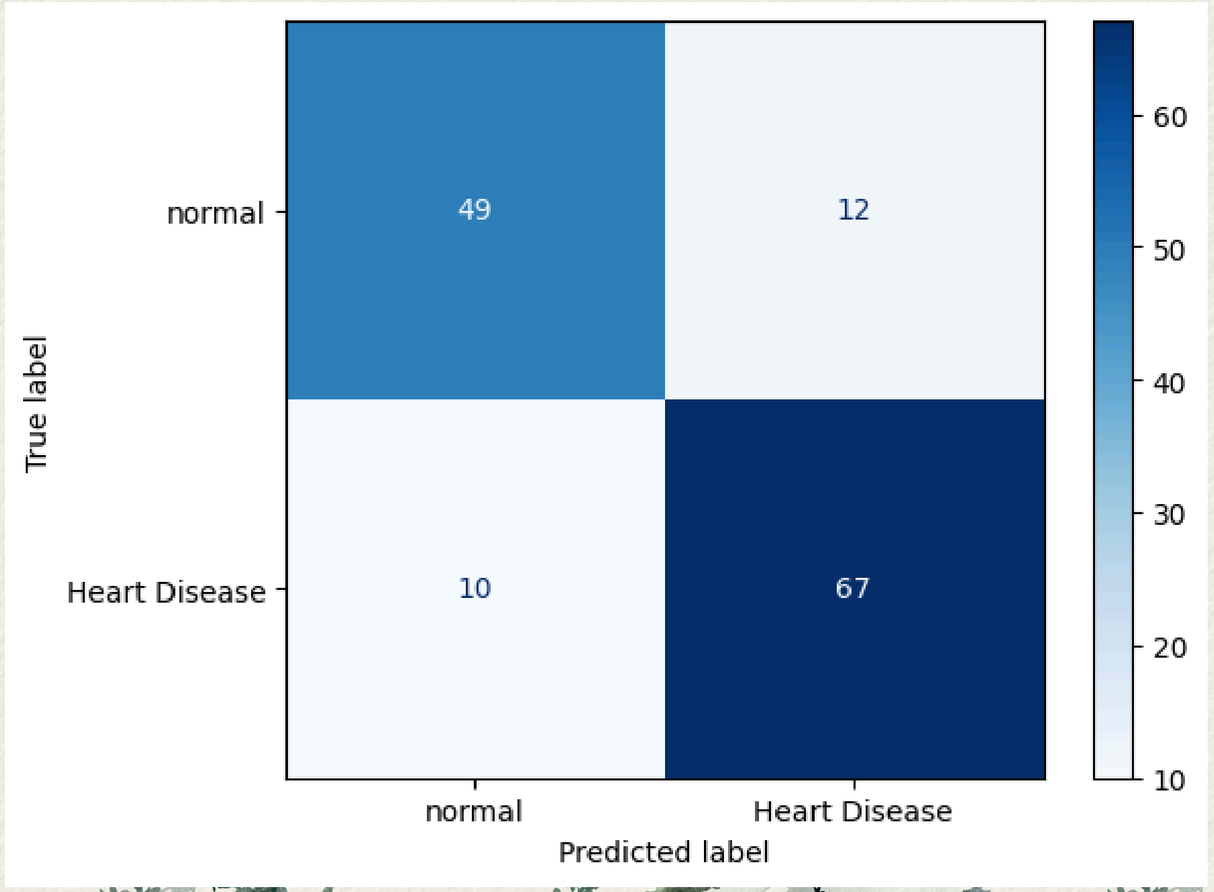
Classification report & O.O.B error

- Mean accuracy on training dataset: 0.89
- Mean accuracy on testing dataset: 0.84
- Accuracy based on Out-of-bag error: 0.88

```
mean accuracy on training dataset:  0.89
mean accuracy on testing dataset:   0.84
O.O.B:                              0.88


              precision    recall  f1-score   support

           0       0.80      0.83      0.82        59
           1       0.87      0.85      0.86        79

    accuracy                           0.84       138
   macro avg       0.84      0.84      0.84       138
weighted avg       0.84      0.84      0.84       138
```

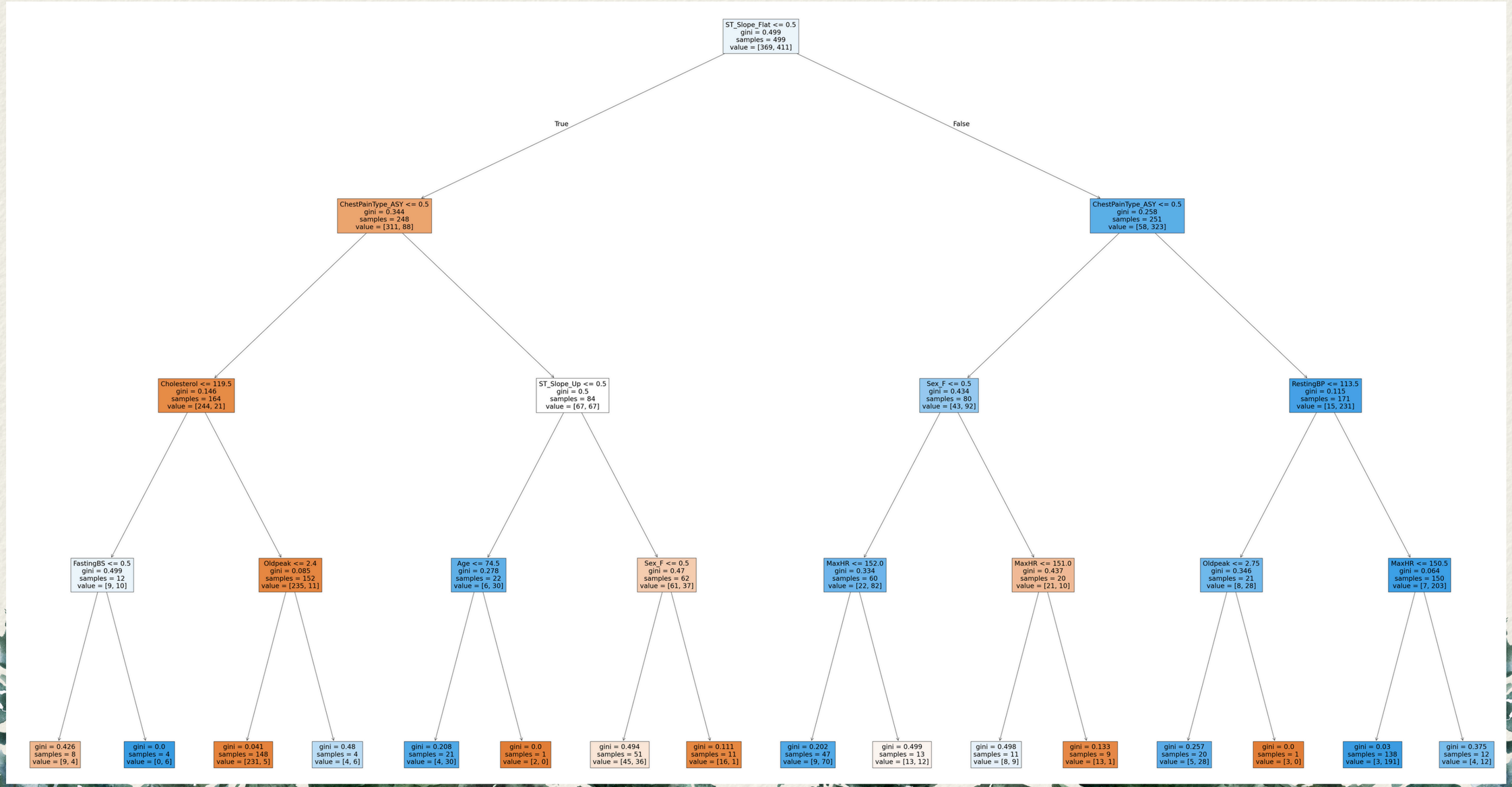
Graphs

Confusion Matrix



Graphs

random forest

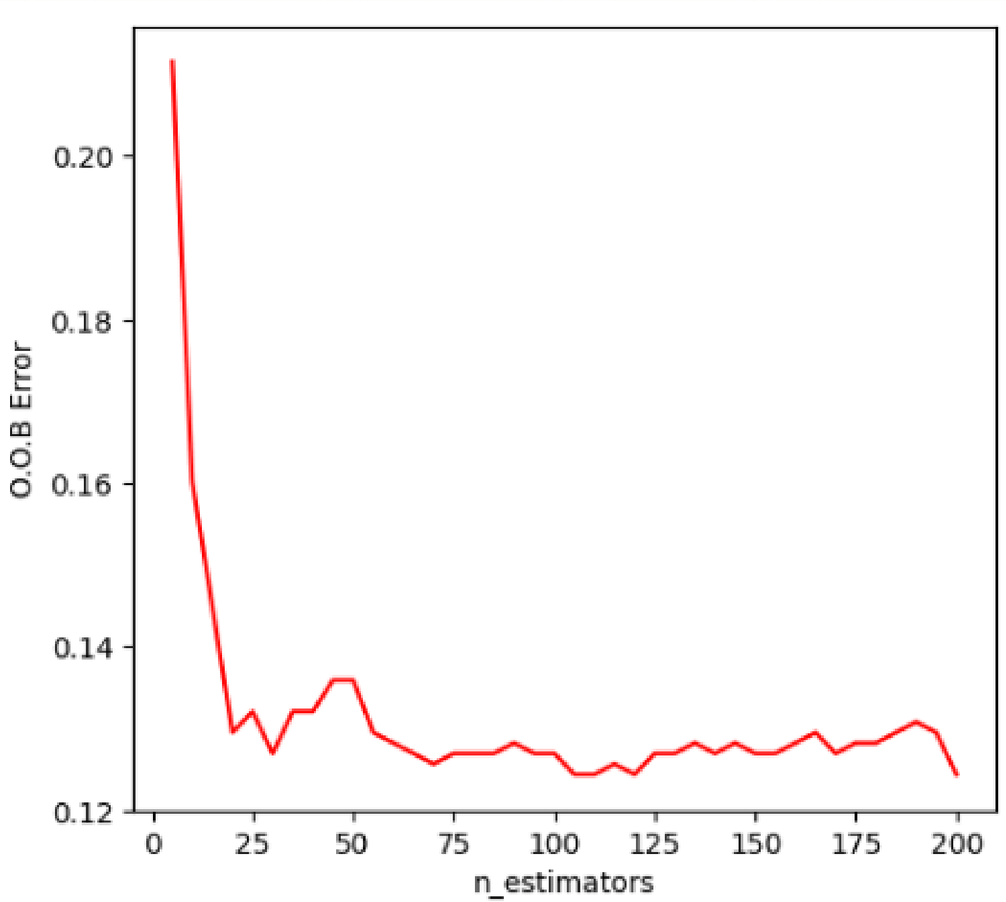


Graphs

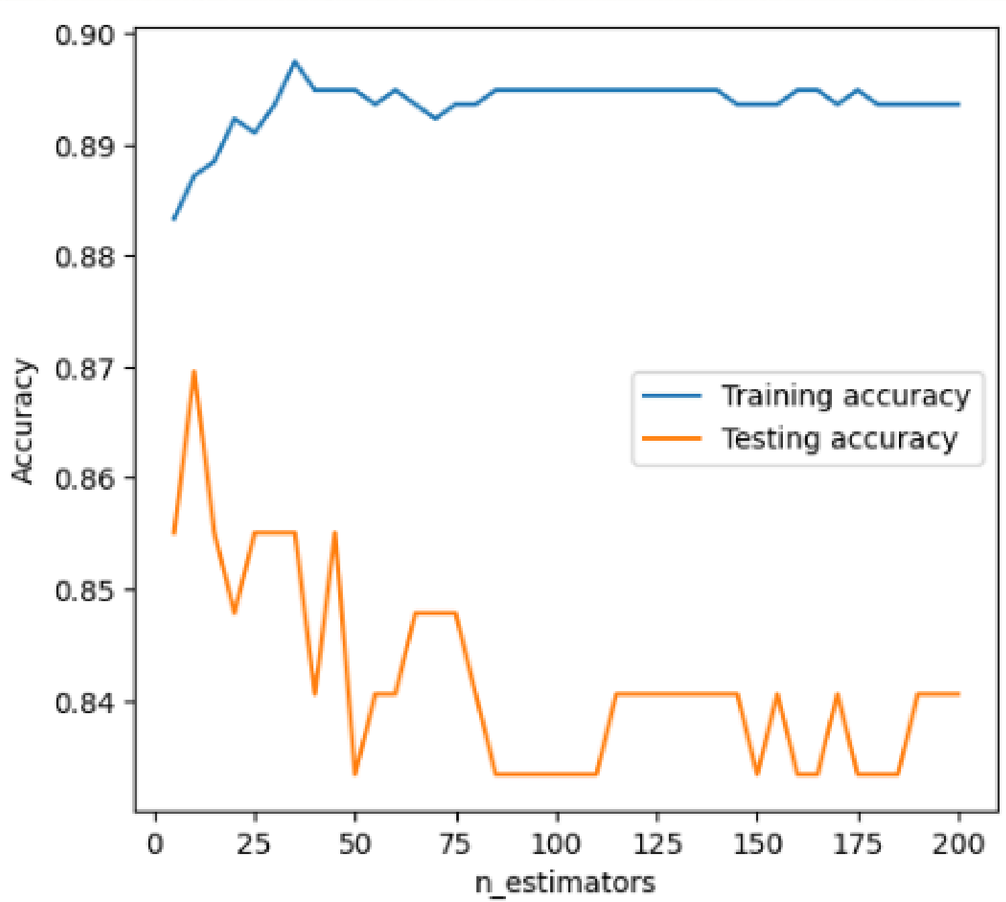
Performance

- performance per n_estimators, starting from 5 to 200

Out-of-bag error



Accuracy on training/testing data




Thank
You