In [221…
```
# ! pip install -q -U keras-tuner
```

In [222…
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from datetime import time
# preprocessing and pipeline
from sklearn.model_selection import train_test_split, cross_val_score, GridSearc
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, LabelEncoder, StandardScaler
from sklearn.pipeline import Pipeline
from outlier_cleaner import OutlierCleaner



# sklearn libraries
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.svm import SVC, SVR
from sklearn.tree import DecisionTreeRegressor, DecisionTreeClassifier
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.ensemble import (
    RandomForestRegressor, GradientBoostingRegressor, AdaBoostRegressor,
    VotingRegressor, StackingRegressor, RandomForestClassifier, VotingClassifier
)

# sklearn metrics
from sklearn.metrics import r2_score, mean_squared_error, accuracy_score, confus
from extended_sklearn_metrics import evaluate_model_with_cross_validation

# extra model
import xgboost as xgb

import joblib

# deep learning
from imblearn.over_sampling import SMOTE
import tensorflow as tf
from tensorflow.keras import layers, models
import keras_tuner as kt
from tensorflow import keras
```

In [223…
```
df = pd.read_csv('heart_attack_prediction_indonesia.csv', nrows=30000)
df1 = df.copy()
```

Feature Engineering (higher values = more risk)

In [224…
```
df['alcohol_consumption'] = df['alcohol_consumption'].fillna('None')
stress_mapping = {
    'Low': 1,
    'Moderate': 2,
    'High': 3
}
df['stress_level'] = df['stress_level'].map(stress_mapping)
smoking_mapping = {
```

```
    'Never': 1,
    'Past': 2,
    'Current': 3
}
df['smoking_status'] = df['smoking_status'].map(smoking_mapping)
alcohol_consumption_mapping = {
    'None': 1,
    'Moderate': 2,
    'High': 3
}
df['alcohol_consumption'] = df['alcohol_consumption'].map(alcohol_consumption_ma
physical_activity_mapping = {
    'Low': 1,
    'Moderate': 2,
    'High': 3
}
df['physical_activity'] = df['physical_activity'].map(physical_activity_mapping)
dietary_habits_mapping = {
    'Unhealthy': 1,
    'Healthy': 2,
}
df['dietary_habits'] = df['dietary_habits'].map(dietary_habits_mapping)
air_pollution_mapping = {
    'Low': 1,
    'Moderate': 2,
    'High': 3
}
df['air_pollution_exposure'] = df['air_pollution_exposure'].map(air_pollution_ma
income_level_mapping = {
    'Low': 3,
    'Middle': 2,
    'High': 1
}
df['income_level'] = df['income_level'].map(income_level_mapping)
df['age_group'] = np.where(
    df['age'] < 40, 1, np.where(
        df['age'] < 55, 2, np.where(
            df['age'] < 65, 3, 4
        )
    )
)
df.age_group.unique()
```

Out[224… `array([3, 2, 4, 1])`

In [225…

```
df.columns.tolist()
```

Out[225…
```
['age',
 'gender',
 'region',
 'income_level',
 'hypertension',
 'diabetes',
 'cholesterol_level',
 'obesity',
 'waist_circumference',
 'family_history',
 'smoking_status',
 'alcohol_consumption',
 'physical_activity',
 'dietary_habits',
 'air_pollution_exposure',
 'stress_level',
 'sleep_hours',
 'blood_pressure_systolic',
 'blood_pressure_diastolic',
 'fasting_blood_sugar',
 'cholesterol_hdl',
 'cholesterol_ldl',
 'triglycerides',
 'EKG_results',
 'previous_heart_disease',
 'medication_usage',
 'participated_in_free_screening',
 'heart_attack',
 'age_group']
```

In [226…
```
df['health_risk_score'] = (
    df['hypertension'] +
    df['diabetes'] +
    df['obesity'] +
    df['family_history'] +
    df['smoking_status'] +
    df['alcohol_consumption'] +
    (1 - df['physical_activity']) +
    (1 - df['dietary_habits']) +
    df['air_pollution_exposure'] +
    df['stress_level'] +
    df.income_level
)

# df['cholesterol ratio'] = df['cholesterol_hdl']/df['cholesterol_ldl']
# df = df.drop(labels=['cholesterol_hdl', 'cholesterol_ldl'], axis=1)
df['obesity_risk_score'] = np.where(df['gender'] == 'Male' , df['obesity'] * 1 +
# df = df.drop(labels=['obesity', 'waist_circumference'], axis=1)
# df['pulse_pressure'] = df['blood_pressure_systolic'] - df['blood_pressure_dias
# df = df.drop(labels=['blood_pressure_systolic', 'blood_pressure_diastolic'], a
df['stress_to_sleep_ratio'] = df['stress_level'] / df['sleep_hours']
# df = df.drop(columns=['stress_level', 'sleep_hours'])
df['mean_arterial_pressure'] = (2* df['blood_pressure_systolic'] + df['blood_pre
# df = df.drop(columns=['blood_pressure_systolic', 'blood_pressure_diastolic'])
df['triglyceride-hdl-ratio'] = df['triglycerides'] / df['cholesterol_hdl']
# df = df.drop(columns=['triglycerides', 'cholesterol_hdl'])
```

In [227…
```
# df_risk_score = df[['health_risk_score','hypertension', 'diabetes', 'obesity_r
# df_risk_score.corr().sort_values(by='health_risk_score', ascending=False).styl
```

In [228…
```
# df = df.drop(columns=['physical_activity', 'dietary_habits'])
```

In [229…
```
df.isna().any()
```

Out[229…
```
age                                False
gender                             False
region                             False
income_level                       False
hypertension                       False
diabetes                           False
cholesterol_level                  False
obesity                            False
waist_circumference                False
family_history                     False
smoking_status                     False
alcohol_consumption                False
physical_activity                  False
dietary_habits                     False
air_pollution_exposure             False
stress_level                       False
sleep_hours                        False
blood_pressure_systolic            False
blood_pressure_diastolic           False
fasting_blood_sugar                False
cholesterol_hdl                    False
cholesterol_ldl                    False
triglycerides                      False
EKG_results                        False
previous_heart_disease             False
medication_usage                   False
participated_in_free_screening     False
heart_attack                       False
age_group                          False
health_risk_score                  False
obesity_risk_score                 False
stress_to_sleep_ratio              False
mean_arterial_pressure             False
triglyceride-hdl-ratio             False
dtype: bool
```

In [230…
```
# df.health_risk_score.unique()
```

Encoding

In [231…
```
## Encoding
cat_df = df.select_dtypes(include='object')
num_df = df.select_dtypes(exclude='object')
encoder = LabelEncoder()
for cols in cat_df:
    cat_df[cols+'_encoded'] = encoder.fit_transform(cat_df[cols])
cat_df = cat_df.select_dtypes(exclude='object')
df = pd.concat([cat_df, num_df], axis=1)
```

Cleaning Outliers

In [232…
```
cleaner = OutlierCleaner(df, preserve_index=True)
cleaned_df, info = cleaner.clean_columns(
    method='zscore',
    show_progress=True
)
df = cleaned_df
```

```
Cleaning columns: 100%|██████████| 34/34 [00:00<00:00, 188.79it/s]
```

Visualization of correlation of features with heart attack

In [233…
```
df.waist_circumference.min(), df.waist_circumference.max()
# df.waist_circumference.unique()
```

Out[233… (45, 142)

In [234…
```
df.columns
```

Out[234…
```
Index(['gender_encoded', 'region_encoded', 'EKG_results_encoded', 'age',
       'income_level', 'hypertension', 'diabetes', 'cholesterol_level',
       'obesity', 'waist_circumference', 'family_history', 'smoking_status',
       'alcohol_consumption', 'physical_activity', 'dietary_habits',
       'air_pollution_exposure', 'stress_level', 'sleep_hours',
       'blood_pressure_systolic', 'blood_pressure_diastolic',
       'fasting_blood_sugar', 'cholesterol_hdl', 'cholesterol_ldl',
       'triglycerides', 'previous_heart_disease', 'medication_usage',
       'participated_in_free_screening', 'heart_attack', 'age_group',
       'health_risk_score', 'obesity_risk_score', 'stress_to_sleep_ratio',
       'mean_arterial_pressure', 'triglyceride-hdl-ratio'],
      dtype='object')
```

In [235…
```
results = []
for i in df1.columns:
    results.append({
        f'{i}': df1[i].unique()
    })
results
```

Out[235…

```
[{'age': array([60, 53, 62, 73, 52, 64, 49, 61, 57, 32, 34, 48, 42, 58, 44, 38,
72,
        55, 37, 56, 41, 59, 47, 51, 77, 54, 40, 31, 39, 63, 46, 67, 33, 50,
        66, 71, 25, 45, 65, 84, 68, 81, 43, 36, 70, 35, 87, 90, 82, 80, 30,
        76, 74, 29, 69, 79, 78, 27, 75, 28, 85, 86, 83, 26, 88, 89],
       dtype=int64)},
 {'gender': array(['Male', 'Female'], dtype=object)},
 {'region': array(['Rural', 'Urban'], dtype=object)},
 {'income_level': array(['Middle', 'Low', 'High'], dtype=object)},
 {'hypertension': array([0, 1], dtype=int64)},
 {'diabetes': array([1, 0], dtype=int64)},
 {'cholesterol_level': array([211, 208, 231, 202, 232, 238, 165, 186, 121, 196,
190, 234, 193,
        125, 134, 271, 185, 230, 132, 163, 200, 191, 219, 142, 180, 205,
        228, 265, 177, 192, 176, 207, 174, 225, 170, 130, 251, 201, 159,
        172, 153, 258, 221, 189, 214, 105, 255, 149, 128, 199, 131, 139,
        133, 168, 188, 285, 252, 216, 220, 212, 116, 182, 250, 175, 246,
        215, 226, 240, 210, 147, 254, 227, 243, 223, 146, 241, 173, 256,
        244, 198, 247, 187, 164, 217, 218, 152, 161, 303, 203, 206, 245,
        151, 249, 181, 178, 183, 166, 184, 162, 204, 156, 179, 171, 158,
        270, 222, 136, 194, 154, 253, 167, 236, 209, 266, 263, 148, 242,
        195, 239, 235, 118, 113, 229, 123, 117, 197, 233, 273, 224, 272,
        150, 310, 298, 169, 257, 268, 155, 145, 213, 248, 237, 160, 141,
        259, 111, 275, 144, 127, 264, 138, 277, 305, 157, 280, 119, 103,
        260, 135, 261, 279, 140, 137, 274, 114, 115, 122, 107, 100, 278,
        124, 283, 318, 143, 306, 288, 287, 120, 276, 299, 269, 129, 325,
        262, 106, 319, 112, 293, 284, 301, 281, 295, 302, 296, 292, 126,
        289, 108, 282, 109, 267, 102, 290, 300, 286, 308, 312, 326, 309,
        291, 104, 311, 330, 316, 334, 294, 304, 315, 317, 341, 110, 335,
        323, 307, 297, 324, 321, 101, 314, 313, 336, 320, 331, 339, 350,
        327, 349, 337, 328, 329], dtype=int64)},
 {'obesity': array([0, 1], dtype=int64)},
 {'waist_circumference': array([ 83, 106, 112,  82,  89,  81,  91,  72, 115,  8
8,  99,  80,  70,
         74,  77,  63, 116,  95, 101,  84,  73, 113, 132, 104, 100,  90,
        117, 110,  92,  94, 105,  97,  76,  98, 108, 122,  93,  86,  60,
        124, 109,  75, 148,  66,  67, 111,  87,  71,  96, 107, 114,  69,
         61, 102,  85,  62, 103,  68,  78,  58,  79,  65, 130, 119, 121,
        123, 129, 120, 135,  59,  57,  53,  44,  50, 127, 118, 131, 126,
         54, 143,  55, 134, 128,  64,  56, 133, 139,  43,  52,  51, 125,
         49,  46,  48, 136,  45,  42, 138, 141, 140, 137, 173, 144, 142,
        151,  47, 146,  34,  27, 149, 145,  37, 155,  40,  41,  33, 147,
        152, 150,  31, 154,  20], dtype=int64)},
 {'family_history': array([0, 1], dtype=int64)},
 {'smoking_status': array(['Never', 'Past', 'Current'], dtype=object)},
 {'alcohol_consumption': array([nan, 'Moderate', 'High'], dtype=object)},
 {'physical_activity': array(['High', 'Moderate', 'Low'], dtype=object)},
 {'dietary_habits': array(['Unhealthy', 'Healthy'], dtype=object)},
 {'air_pollution_exposure': array(['Moderate', 'High', 'Low'], dtype=object)},
 {'stress_level': array(['Moderate', 'High', 'Low'], dtype=object)},
 {'sleep_hours': array([5.97060316, 5.64381314, 6.33619667, ..., 6.71533855, 6.
44848813,
        5.8914506 ])},
 {'blood_pressure_systolic': array([113, 132, 116, 136, 127, 131, 128, 109, 15
0, 142,  98, 143, 146,
        145, 110, 138, 129,  99, 122, 123, 117, 137, 121, 130, 106, 141,
        118,  90,  85, 100, 162, 104, 126, 144, 166, 125, 114, 135, 140,
        148, 107, 139, 155, 151, 149, 124, 115, 103, 105, 119, 152, 102,
        147, 161, 134, 112,  78, 153, 169, 133, 120, 111, 101, 159, 164,
         83,  88, 108,  92,  94,  82, 157, 170,  97,  96, 154, 163, 165,
```

```
         91,  89, 160,  95, 156, 158,  93, 168, 187,  81, 167, 173,  74,
        171,  72, 174, 180,  87,  86, 175, 176, 172,  77,  79,  84,  80,
        182, 181,  73, 179,  76, 178,  75, 177, 190, 183,  71], dtype=int64)},
 {'blood_pressure_diastolic': array([ 62,  76,  74,  65,  75,  71,  97,  83,  8
7,  98, 103,  77,  54,
         64,  81,  49,  79,  93,  67,  70,  89,  66,  85,  78,  80,  86,
        105,  82,  72,  68,  73,  92,  90,  96,  88,  61,  84,  69,  91,
         95,  94,  63, 102, 101,  59,  99,  60,  58, 108, 106, 107,  56,
        100,  51, 104,  55,  53,  57,  52, 109, 113, 111,  48,  45,  46,
         50, 114, 110,  40,  43, 112,  47, 115, 116,  44, 127], dtype=int64)},
 {'fasting_blood_sugar': array([173,  70, 118,  98, 104, 129,  88, 112, 147,  7
9, 105, 113, 126,
         86,  83,  84, 141, 145, 101, 114, 102, 157, 127, 117, 107,  89,
        109, 153, 108,  80, 140, 168,  73,  75,  94, 123,  77, 144, 119,
         81,  92, 116, 125, 100, 135,  71, 165,  85, 121, 111, 120,  90,
        137, 164, 156, 131, 103, 134, 161, 158, 110,  76,  74, 142, 115,
        143, 146,  91, 162, 148, 154,  93, 167, 132, 130,  72, 160,  96,
        124,  97, 122,  95, 133, 170, 151, 150, 149, 136, 176,  99, 182,
         78,  82, 187, 178, 106, 172, 155, 152, 138, 186, 207, 177, 128,
        169, 139,  87, 174, 166, 195, 185, 171, 159, 163, 181, 179, 184,
        210, 203, 220, 180, 175, 194, 196, 216, 212, 183, 190, 188, 202,
        201, 192, 193, 199, 197, 215, 198, 191, 189, 200, 204, 206, 214,
        208, 209], dtype=int64)},
 {'cholesterol_hdl': array([48, 58, 69, 52, 59, 34, 40, 47, 46, 38, 62, 39, 44,
61, 55, 50, 56,
        35, 49, 45, 66, 51, 53, 33, 37, 57, 64, 54, 63, 43, 32, 60, 19, 72,
        27, 41, 30, 67, 28, 36, 31, 23, 42, 24, 68, 73, 65, 76, 20, 74, 70,
        26, 21, 71, 25, 29, 81, 77, 75, 18, 78, 79, 17, 22, 15, 83, 80, 16,
        85, 87, 86, 82, 13, 88, 89, 14], dtype=int64)},
 {'cholesterol_ldl': array([121,  83, 130,  85, 127, 148, 128, 100, 157,  93, 1
09, 172,  89,
         60, 185, 175, 126, 158, 144, 203, 205, 141, 132,  72, 116, 110,
         53, 140, 161, 107,  33, 165, 159,  97, 177, 152, 195, 117,  99,
         82, 193, 171, 108, 137, 162, 119, 150, 215, 153, 134,  84, 131,
        135, 142, 123,  90, 155,  71, 115,  77, 105, 106, 166, 133, 145,
        101, 118, 149, 111, 183, 187,  95,  98, 103, 180, 163, 173, 129,
        156, 167,  57, 188, 112, 113, 189, 169,  88, 124, 114,  74, 219,
        138, 160, 104,  70, 164,  76, 184, 122, 186, 170,  81, 182, 226,
        143, 146, 210, 154,  66,  62, 102,  96,  87, 125, 181, 197,  80,
        139,  51, 147, 120, 204, 174,  42, 168, 178, 136,  79,  67, 199,
         73, 179,  86, 208, 194,  92,  65,  94,  91, 176, 235,  64,  78,
        207, 191,  52, 196,  61,  45, 198, 151, 240,  55,  59,  58, 221,
         54,  39, 201,  46, 200, 192,  69, 206,  21, 209,  63,  47,  49,
         34,  68, 222,  75, 202, 216,  24, 214,  30, 213,  56, 233,  38,
        190,  43,  50, 211,  35,  36, 275,  37,  48,  31,  20,  40, 230,
        232, 241, 238, 223, 225,  16, 252, 218,  28, 244, 256,  15, 237,
        231, 217, 212, 262,  44,  32, 220,  26, 261,  27,  17, 228,  29,
        224,  41, 243,  22, 229, 227,   9, 258,  11, 260, 239,  19, -13,
        234,  12,   3, 246,  -7,  23,   8, 253,   1,  13, 236, 250, 247,
        242], dtype=int64)},
 {'triglycerides': array([101, 138, 171, 146, 139, 191, 167,  50, 198, 164, 14
5, 148,  92,
        186, 202, 176, 157, 233, 155, 170, 183, 188, 211, 152, 185, 154,
        147, 247, 161, 192, 133, 205, 112,  78, 141, 212, 162, 196, 184,
        110, 125, 163,  76, 230, 215,  74, 119, 187, 168, 190, 178,  89,
         87, 156, 201, 159, 172, 200, 140,  84, 102, 126, 239,  85, 197,
        222, 169, 223, 107, 275, 254, 165,  80, 179, 111, 244, 127, 144,
        130, 181, 182, 153, 136, 241,  79, 149, 137, 124, 108, 113, 150,
         86, 189, 135, 105,  72,  73, 194, 174, 142,  70, 242, 121, 166,
        177, 214, 118, 206, 158, 240,  97, 122, 238,  57, 160,  98, 100,
```

```
       131, 128,  60, 123, 199, 104, 106,  83, 143, 300,  94, 151, 116,
        88, 204, 226, 117, 234, 209, 180,  59, 245, 268, 216,  71,  77,
        64, 224, 250, 208,  91, 132,  93, 263,  75, 262, 227,  95, 173,
       203, 236, 269,  67, 134,  99, 258, 252, 175,  96, 129, 115, 253,
       114, 120, 249, 259, 193, 195, 109, 219, 213, 103,  62, 225,  81,
       243, 282, 237, 228, 217,  54,  58, 264,  69, 207, 229, 218,  66,
       246,  61, 232, 220, 248,  55,  56, 274, 221, 278, 235,  65,  90,
        52, 270, 210, 265,  82, 297,  63, 251,  51, 295, 279,  68, 255,
       231, 266, 323, 257, 298, 267, 261, 288, 276,  53, 285, 272, 307,
       271, 256, 286, 316, 291, 281, 318, 296, 283, 340, 294, 303, 305,
       273, 260, 301, 308, 317, 309, 290, 299, 284, 277, 287, 334, 342,
       293, 314, 330, 306, 329, 322, 302, 320, 280, 289, 319, 327, 349],
      dtype=int64)},
 {'EKG_results': array(['Normal', 'Abnormal'], dtype=object)},
 {'previous_heart_disease': array([0, 1], dtype=int64)},
 {'medication_usage': array([0, 1], dtype=int64)},
 {'participated_in_free_screening': array([0, 1], dtype=int64)},
 {'heart_attack': array([0, 1], dtype=int64)}]
```

In [236…

```
results = []
for i in df.columns:
    results.append({
        f'{i}': df[i].unique()
    })
results
```

Out[236…

```
[{'gender_encoded': array([1, 0])},
 {'region_encoded': array([0, 1])},
 {'EKG_results_encoded': array([1, 0])},
 {'age': array([60, 53, 62, 73, 52, 64, 49, 61, 57, 32, 34, 48, 42, 58, 38, 72,
55,
         37, 56, 41, 59, 47, 51, 77, 54, 40, 31, 39, 63, 46, 67, 33, 50, 66,
         44, 25, 45, 65, 84, 68, 71, 81, 43, 36, 70, 35, 87, 90, 82, 80, 30,
         76, 74, 29, 69, 79, 78, 27, 75, 28, 86, 85, 83, 26, 88, 89],
        dtype=int64)},
 {'income_level': array([2, 3, 1], dtype=int64)},
 {'hypertension': array([0, 1], dtype=int64)},
 {'diabetes': array([1, 0], dtype=int64)},
 {'cholesterol_level': array([211, 208, 231, 202, 232, 238, 165, 186, 121, 196,
190, 234, 193,
         125, 134, 271, 185, 132, 163, 200, 191, 219, 142, 180, 205, 228,
         265, 177, 192, 176, 207, 174, 225, 170, 130, 251, 201, 159, 172,
         153, 258, 221, 189, 214, 105, 255, 149, 128, 199, 131, 139, 133,
         168, 285, 252, 216, 220, 212, 116, 182, 250, 175, 246, 226, 240,
         210, 147, 254, 227, 243, 223, 146, 241, 173, 256, 244, 198, 247,
         187, 164, 217, 218, 152, 161, 303, 203, 206, 245, 151, 181, 183,
         166, 184, 162, 204, 156, 179, 171, 158, 270, 222, 136, 230, 194,
         154, 253, 167, 236, 209, 266, 263, 148, 242, 195, 239, 235, 113,
         229, 123, 117, 197, 233, 273, 224, 272, 215, 150, 310, 249, 298,
         169, 257, 268, 155, 145, 213, 248, 237, 160, 141, 259, 111, 275,
         144, 127, 264, 188, 178, 138, 277, 305, 157, 280, 119, 103, 260,
         135, 261, 279, 140, 137, 274, 114, 115, 122, 107, 278, 118, 283,
         318, 143, 306, 288, 287, 120, 276, 299, 100, 269, 129, 262, 106,
         112, 293, 284, 301, 281, 295, 302, 296, 292, 126, 289, 108, 282,
         109, 267, 102, 290, 300, 124, 286, 308, 312, 309, 291, 104, 311,
         316, 294, 304, 317, 110, 307, 297, 315, 101, 314, 313], dtype=int64)},
 {'obesity': array([0, 1], dtype=int64)},
 {'waist_circumference': array([ 83, 106, 112,  82,  89,  81,  91,  72, 115,  8
8,  99,  80,  70,
          74,  77,  63,  95, 101,  84,  73, 113, 132, 104, 100,  90, 117,
         110,  92,  94, 105,  97,  76,  98, 108, 122,  93,  86,  60, 124,
         109,  75,  66,  67, 111,  87,  71,  96, 107, 114,  69,  61, 102,
          85,  62, 103,  68,  78,  58,  79, 130, 119, 121, 123,  65, 129,
         120, 135,  59,  57,  53, 116, 127, 118, 131, 126,  54,  55, 134,
         128,  64,  56, 133,  52,  51, 125,  49,  46,  48, 136,  45,  50,
         138, 141, 140, 137, 139, 142,  47], dtype=int64)},
 {'family_history': array([0, 1], dtype=int64)},
 {'smoking_status': array([1, 2, 3], dtype=int64)},
 {'alcohol_consumption': array([1, 2, 3], dtype=int64)},
 {'physical_activity': array([3, 2, 1], dtype=int64)},
 {'dietary_habits': array([1, 2], dtype=int64)},
 {'air_pollution_exposure': array([2, 3, 1], dtype=int64)},
 {'stress_level': array([2, 3, 1], dtype=int64)},
 {'sleep_hours': array([5.97060316, 5.64381314, 6.33619667, ..., 6.71533855, 6.
44848813,
         5.8914506 ])},
 {'blood_pressure_systolic': array([113, 132, 116, 136, 127, 131, 128, 109, 15
0, 142,  98, 143, 146,
         145, 110, 138, 129,  99, 123, 117, 137, 121, 130, 106, 141, 118,
          90,  85, 100, 162, 104, 126, 144, 166, 125, 114, 135, 140, 148,
         122, 107, 139, 155, 151, 149, 124, 115, 103, 105, 119, 152, 102,
         147, 161, 134, 112, 153, 169, 120, 111, 101, 159, 133, 164, 108,
          92,  94, 157,  97,  88,  96, 154, 163, 165,  91,  89, 160,  95,
         156, 158,  93, 168, 167, 170, 171,  87,  86, 172, 174, 173],
        dtype=int64)},
 {'blood_pressure_diastolic': array([ 62,  76,  74,  65,  75,  71,  97,  83,  8
```

```
7,  98, 103,  77,  54,
         64,  81,  79,  93,  67,  70,  89,  66,  85,  78,  80,  86, 105,
         82,  72,  68,  73,  92,  90,  96,  88,  61,  84,  69,  91,  95,
         94,  63, 102, 101,  59,  99,  60,  58, 108, 106, 107,  56, 100,
         51, 104,  55,  53,  57,  52, 109,  50], dtype=int64)},
 {'fasting_blood_sugar': array([173,  70, 118,  98, 104, 129,  88, 112, 147,  7
9, 105, 113, 126,
         86,  83,  84, 145, 101, 114, 102, 157, 127, 117, 107,  89, 109,
        153, 108,  80, 140, 168,  73,  75,  94, 123,  77, 144, 119,  81,
         92, 116, 100, 135,  71, 165,  85, 121, 111, 120,  90, 137, 164,
        156, 131, 103, 134, 141, 161, 110,  76,  74, 142, 125, 115, 143,
        146,  91, 162, 158, 148, 154, 167, 132,  72, 160,  96, 124,  97,
        130, 122,  95, 133, 170, 151,  93, 150, 136, 176,  99, 182,  78,
         82, 187, 178, 106, 172, 155, 138, 186, 177, 128, 169, 139,  87,
        174, 166, 149, 185, 171, 152, 159, 163, 181, 179, 184, 180, 175,
        183, 190, 188, 192, 193, 191, 189], dtype=int64)},
 {'cholesterol_hdl': array([48, 58, 69, 52, 59, 34, 40, 47, 46, 38, 62, 39, 44,
61, 55, 50, 56,
        35, 49, 45, 66, 51, 53, 33, 37, 57, 64, 54, 63, 43, 32, 60, 72, 41,
        30, 67, 28, 36, 23, 42, 24, 31, 68, 73, 65, 76, 74, 70, 27, 26, 21,
        71, 25, 29, 77, 75, 78, 20, 79, 22], dtype=int64)},
 {'cholesterol_ldl': array([121,  83, 130,  85, 127, 148, 128, 100, 157,  93, 1
09, 172,  89,
         60, 185, 175, 126, 144, 203, 205, 141, 132,  72, 116, 110,  53,
        140, 161, 107,  33, 165, 159,  97, 177, 152, 195, 117,  99,  82,
        193, 171, 108, 137, 162, 119, 150, 215, 158, 153, 134,  84, 131,
        135, 142, 123,  90, 155,  71, 115,  77, 105, 106, 166, 133, 145,
        101, 118, 149, 111, 183, 187,  95,  98, 103, 180, 163, 173, 129,
        156, 167,  57, 188, 112, 113, 189, 169,  88, 124, 114,  74, 219,
        138, 160, 104,  70, 164,  76, 184, 122, 186, 170,  81, 182, 226,
        143, 146, 210, 154,  66,  62, 102,  96,  87, 125, 181, 197,  80,
        139,  51, 147, 120, 204, 174,  42, 168, 178, 136,  67, 199,  73,
        179,  86, 208, 194,  92,  65,  94, 176, 235,  64,  78, 207, 191,
         52, 196,  61,  45, 198, 151,  55,  79,  59,  91,  58, 221,  54,
         39, 201,  46, 200, 192,  69, 206, 209,  63,  47,  49,  34,  68,
        222,  75, 202, 216, 214,  30, 213,  56, 233,  38, 190,  50, 211,
         35,  36,  37,  48,  31,  40, 230,  43, 223, 225, 218,  28, 231,
        217, 212,  32, 220,  26,  44,  27, 228,  29, 224,  41, 229, 227,
        232, 234], dtype=int64)},
 {'triglycerides': array([101, 138, 171, 146, 139, 191, 167,  50, 198, 164, 14
5, 148,  92,
        186, 202, 176, 233, 155, 170, 183, 188, 211, 152, 185, 154, 147,
        247, 161, 192, 133, 205, 112,  78, 141, 212, 162, 196, 184, 110,
        125, 163,  76, 230, 215,  74, 119, 187, 168, 190, 178,  89,  87,
        156, 201, 159, 172, 200, 140,  84, 102, 126, 239,  85, 197, 222,
        169, 223, 107, 275, 165,  80, 179, 111, 244, 127, 144, 130, 181,
        182, 153, 136, 241,  79, 149, 137, 124, 108, 113, 150,  86, 189,
        135, 105,  72,  73, 174, 142,  70, 242, 121, 166, 177, 214, 118,
        206, 158, 240,  97, 122, 238,  57, 160,  98, 100, 131, 128,  60,
        123, 104, 106,  83, 143,  94, 151, 116,  88, 204, 226, 117, 234,
        209, 180,  59, 245, 268, 216,  71,  77,  64, 199, 224, 250, 208,
         91, 132,  93, 263,  75, 262, 227,  95, 173, 203, 236, 194,  67,
        134,  99, 258, 252, 175,  96, 129, 115, 253, 157, 114, 120, 249,
        259, 193, 195, 109, 219, 213, 103,  62, 225,  81, 243, 282, 237,
        228, 217,  54,  58,  69, 207, 229, 218,  66, 246,  61, 232, 220,
        248,  55,  56, 221, 278, 235,  65, 264,  90,  52, 270, 210, 265,
        269,  82, 297,  63, 251,  51, 279,  68, 231, 266, 257, 298, 267,
        276, 254,  53, 272, 271, 256, 295, 255, 261, 291, 281, 288, 296,
        283, 294, 273, 260, 290, 284, 274, 277, 287, 285, 293, 286, 280,
        289], dtype=int64)},
```

```
 {'previous_heart_disease': array([0, 1], dtype=int64)},
 {'medication_usage': array([0, 1], dtype=int64)},
 {'participated_in_free_screening': array([0, 1], dtype=int64)},
 {'heart_attack': array([0, 1], dtype=int64)},
 {'age_group': array([3, 2, 4, 1])},
 {'health_risk_score': array([ 7, 11,  9, 12, 10, 14,  8, 13,  6,  5,  4, 15],
dtype=int64)},
 {'obesity_risk_score': array([0. , 0.5, 1.5, 1. ])},
 {'stress_to_sleep_ratio': array([0.33497453, 0.53155551, 0.15782338, ..., 0.29
782564, 0.31015022,
        0.33947497])},
 {'mean_arterial_pressure': array([ 96.        , 113.33333333, 102.        , 11
2.33333333,
         109.66666667, 111.        , 117.66666667, 100.33333333,
         129.        , 123.66666667,  98.        , 129.66666667,
         123.        , 114.66666667,  94.66666667, 119.66666667,
         113.66666667,  93.        , 118.33333333, 109.        ,
         120.66666667, 121.        , 101.        , 102.66666667,
         115.        , 104.66666667, 111.33333333,  88.66666667,
         132.33333333,  83.66666667, 107.33333333,  91.33333333,
         133.33333333, 125.66666667,  94.        , 119.33333333,
         118.66666667, 110.33333333,  98.33333333, 127.33333333,
          92.66666667, 113.        , 139.66666667, 105.66666667,
         124.        , 102.33333333, 126.33333333,  98.66666667,
         121.33333333, 100.66666667, 105.33333333, 116.33333333,
         101.66666667,  96.33333333, 131.33333333, 126.66666667,
         121.66666667, 106.33333333, 112.        , 108.66666667,
         126.        , 106.66666667, 120.        , 129.33333333,
         110.        , 112.66666667, 118.        , 103.        ,
         125.        , 107.        , 103.66666667, 108.        ,
         107.66666667,  96.66666667, 101.33333333, 108.33333333,
         130.        , 127.        ,  93.66666667, 128.66666667,
         117.        , 127.66666667, 128.        , 114.        ,
          97.66666667, 123.33333333, 131.        , 116.66666667,
         143.66666667, 124.33333333, 109.33333333, 135.33333333,
         115.33333333, 119.        , 100.        ,  91.        ,
         122.33333333, 117.33333333, 124.66666667, 137.        ,
          97.        ,  90.        , 115.66666667, 122.        ,
         114.33333333, 103.33333333, 110.66666667, 122.66666667,
         139.33333333,  99.        ,  87.66666667, 116.        ,
         111.66666667,  88.        , 120.33333333, 133.        ,
          90.33333333,  87.33333333,  97.33333333,  88.33333333,
          92.        , 105.        , 131.66666667, 134.33333333,
         128.33333333, 104.        ,  99.33333333, 135.        ,
          99.66666667,  95.        , 138.66666667,  95.33333333,
          94.33333333,  95.66666667, 138.33333333, 139.        ,
          89.        , 132.66666667, 104.33333333, 136.33333333,
         133.66666667, 106.        ,  89.66666667, 130.33333333,
          85.66666667,  86.33333333, 137.33333333, 134.        ,
          86.        ,  92.33333333, 125.33333333, 140.        ,
         136.        , 134.66666667, 136.66666667, 137.66666667,
         132.        ,  93.33333333,  91.66666667, 130.66666667,
          84.33333333,  90.66666667, 142.33333333, 138.        ,
          87.        ,  82.66666667,  89.33333333, 140.66666667,
         135.66666667, 142.        , 141.66666667, 141.33333333,
         143.33333333,  85.33333333,  85.        , 144.        ,
          83.33333333,  83.        ,  82.        ,  86.66666667,
         141.        ,  84.66666667, 142.66666667, 140.33333333,
          82.33333333,  81.66666667,  84.        , 143.        ])},
 {'triglyceride-hdl-ratio': array([2.10416667, 2.37931034, 2.47826087, ..., 1.0
```

```
46875  , 1.703125  ,
        1.10869565])}]
```

In [237…

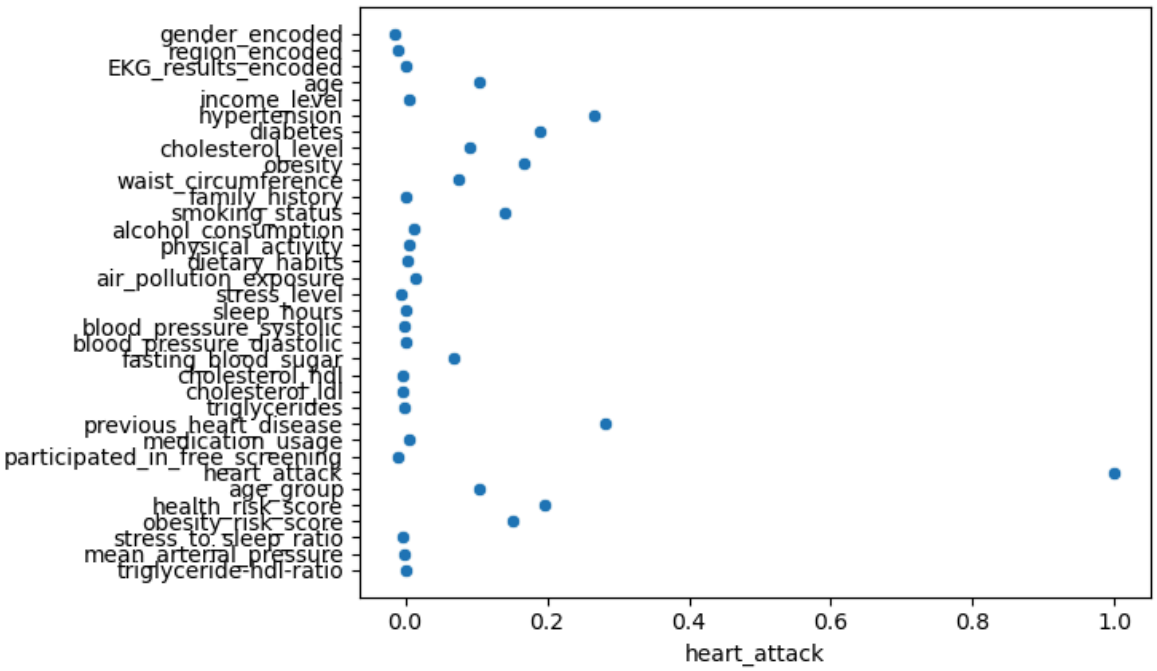
```
corr_df = df.corr()[['heart_attack']].sort_values(by='heart_attack', ascending=F
corr_df.style.background_gradient(cmap='coolwarm', axis=None)
```

Out[237...

	heart_attack
heart_attack	1.000000
previous_heart_disease	0.281527
hypertension	0.266837
health_risk_score	0.197037
diabetes	0.188859
obesity	0.166300
obesity_risk_score	0.152269
smoking_status	0.141086
age	0.103794
age_group	0.103686
cholesterol_level	0.090968
waist_circumference	0.074709
fasting_blood_sugar	0.068222
air_pollution_exposure	0.013178
alcohol_consumption	0.011258
income_level	0.005869
medication_usage	0.004564
physical_activity	0.004182
dietary_habits	0.003435
triglyceride-hdl-ratio	0.001366
family_history	0.001147
blood_pressure_diastolic	0.000394
EKG_results_encoded	0.000035
sleep_hours	-0.000019
mean_arterial_pressure	-0.001625
blood_pressure_systolic	-0.001841
triglycerides	-0.002392
cholesterol_ldl	-0.004101
cholesterol_hdl	-0.004503
stress_to_sleep_ratio	-0.004751
stress_level	-0.006581
participated_in_free_screening	-0.009895
region_encoded	-0.010467

	heart_attack
gender_encoded	-0.014083

In [238…
```
sns.scatterplot(data=df.corr(), x='heart_attack', y=df.columns.tolist())
plt.show()
```

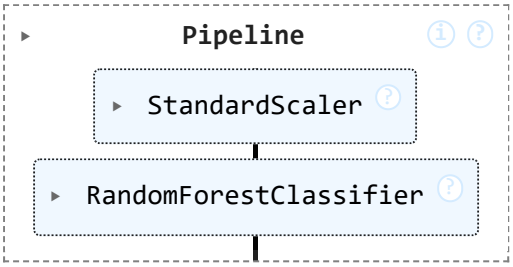


Modelling

In [239…
```
x = df.drop('heart_attack', axis=1)
y = df['heart_attack']
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_
# scaler = StandardScaler()
# X_train_scaled = scaler.fit_transform(X_train)
# X_test_scaled = scaler.fit_transform(X_test)
# X_train_scaled, X_train_scaled[0]
SEED = 42
rf =  RandomForestClassifier(
    criterion='entropy',
    max_depth=5,
    max_features='sqrt',
    min_samples_leaf=1,
    min_samples_split=3,
)
# param_grid = {
#     'classifier__criterion': ['gini', 'entropy', 'log_loss'],
#     'classifier__max_depth': [2,4,5],
#     'classifier__min_samples_split': [2, 3, 4],
#     'classifier__min_samples_leaf': [1, 2, 3],
#     'classifier__max_features': ['sqrt', 'log2'],
# }
rf_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', rf),
])
# grid_search = GridSearchCV(
```

```
#     estimator = rf_pipeline,
#     param_grid = param_grid,
#     scoring = 'accuracy',
#     n_jobs = None,
#     cv = 5,
#     verbose = 2,
# )
# grid_search.fit(X_train, y_train)
# scaler = StandardScaler()
# X_train = scaler.fit_transform(X_train)
# X_test = scaler.transform(X_test)
# rf.fit(X_train, y_train)
rf_pipeline.fit(X_train, y_train)
```

Out[239…



In [240…

```
x.columns
```

Out[240…

```
Index(['gender_encoded', 'region_encoded', 'EKG_results_encoded', 'age',
       'income_level', 'hypertension', 'diabetes', 'cholesterol_level',
       'obesity', 'waist_circumference', 'family_history', 'smoking_status',
       'alcohol_consumption', 'physical_activity', 'dietary_habits',
       'air_pollution_exposure', 'stress_level', 'sleep_hours',
       'blood_pressure_systolic', 'blood_pressure_diastolic',
       'fasting_blood_sugar', 'cholesterol_hdl', 'cholesterol_ldl',
       'triglycerides', 'previous_heart_disease', 'medication_usage',
       'participated_in_free_screening', 'age_group', 'health_risk_score',
       'obesity_risk_score', 'stress_to_sleep_ratio', 'mean_arterial_pressure',
       'triglyceride-hdl-ratio'],
      dtype='object')
```

In [241…

```
y_pred = rf_pipeline.predict(X_test)
print(classification_report(y_test,y_pred))
```

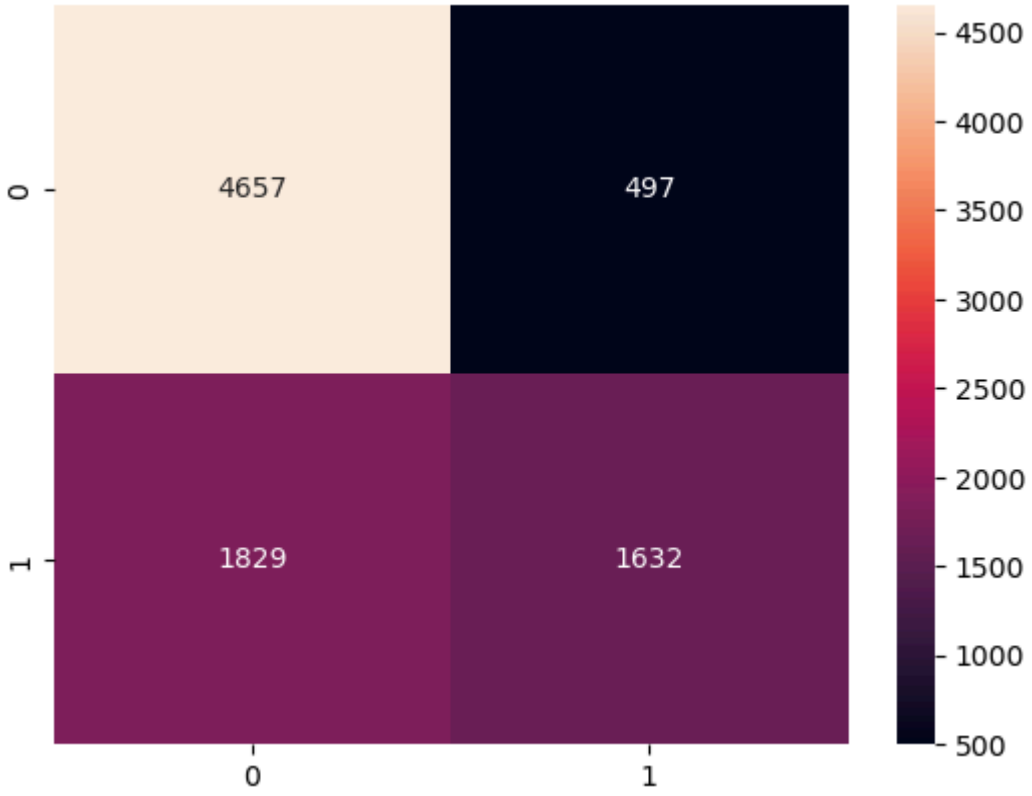
```
              precision    recall  f1-score   support

           0       0.72      0.90      0.80      5154
           1       0.77      0.47      0.58      3461

    accuracy                           0.73      8615
   macro avg       0.74      0.69      0.69      8615
weighted avg       0.74      0.73      0.71      8615
```

In [242…

```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True, fmt='d')
plt.show()
```



Saving model to joblib file

In [243…
```
# joblib.dump(rf_pipeline, 'heart_attack_prediction_model.joblib')
```

Save clean df as CSV file

In [244…
```
df.to_csv('clean_hap.csv', index=False)
```

In [ ]:
```
```