

Heart Disease Prediction Model

Cardiovascular Risk Classification using Random Forest

Import Libraries

```
python

import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
import joblib
```

Dataset Overview

The dataset contains anonymized health information for patients, including features such as age, sex, cholesterol levels, and presence of heart disease. It is used to train a binary classification model to predict cardiovascular risk.

Feature Descriptions:

- `(age)` : Age of the patient
- `(sex)` : Gender
- `(ChestPainType)` : Chest pain type
- `(RestingBP)` : Resting blood pressure
- `(Cholesterol)` : Serum cholesterol in mg/dl
- `(FastingBS)` : Fasting blood sugar
- `(RestingECG)` : resting electrocardiogram results
- `(MaxHR)` : Maximum heart rate achieved
- `(ExerciseAngina)` : Exercise induced angina
- `(oldpeak)` : ST depression induced by exercise relative to rest

- `ST_Slope` : the slope of the peak exercise ST segment
 - `target` : 1 = risk, 0 = no risk
-

Data Loading

```
python  
df = pd.read_csv('/content/heart.csv')  
df.head()
```

Output:

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

```
python  
print(df.info())
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
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)
memory usage: 86.2+ KB
None
```

```
python
```

```
df.isnull().sum()
```

Output:

```
Age      0
Sex      0
ChestPainType 0
RestingBP 0
Cholesterol 0
FastingBS  0
RestingECG 0
MaxHR     0
ExerciseAngina 0
Oldpeak   0
ST_Slope   0
HeartDisease 0
dtype: int64
```

Data Preprocessing

```
python
```

```
# Separate features and target
X = df.drop('HeartDisease', axis=1)
y = df['HeartDisease']
```

```
python
```

```
categorical_cols = ["Sex", "ChestPainType", "RestingECG", "ExerciseAngina", "ST_Slope"]
numerical_cols = ["Age", "RestingBP", "Cholesterol", "FastingBS", "MaxHR", "Oldpeak"]
```

```
python
```

```
categorical_transformer = OneHotEncoder(handle_unknown='ignore')
numerical_transformer = StandardScaler()
```

```
python
```

```
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_cols),
        ('cat', categorical_transformer, categorical_cols)
    ]
)
```

```
python
```

```
pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(random_state=42))
])
```

Model Training and Evaluation

```
python
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

pipeline.fit(X_train, y_train)

print(f"Model accuracy: {pipeline.score(X_test, y_test):.2f}")
```

Output:

```
Model accuracy: 0.88
```

```
python
```

```
#f1 score and other metrics
y_pred = pipeline.predict(X_test)
print(classification_report(y_test, y_pred))
```

Output:

	precision	recall	f1-score	support
0	0.85	0.87	0.86	77
1	0.90	0.89	0.90	107
accuracy			0.88	184
macro avg	0.88	0.88	0.88	184
weighted avg	0.88	0.88	0.88	184

Save the Model

```
python

#Saving the Model
joblib.dump(pipeline, "cr_model.pkl")
```

Output:

```
['cr_model.pkl']
```

Conclusion

The trained Random Forest model achieved good accuracy and F1-score. The model is saved for integration into the Streamlit-based CardioRural application.

Next Steps

- Validate the model with external data
- Tune hyperparameters for better performance
- Deploy on low-resource mobile platforms