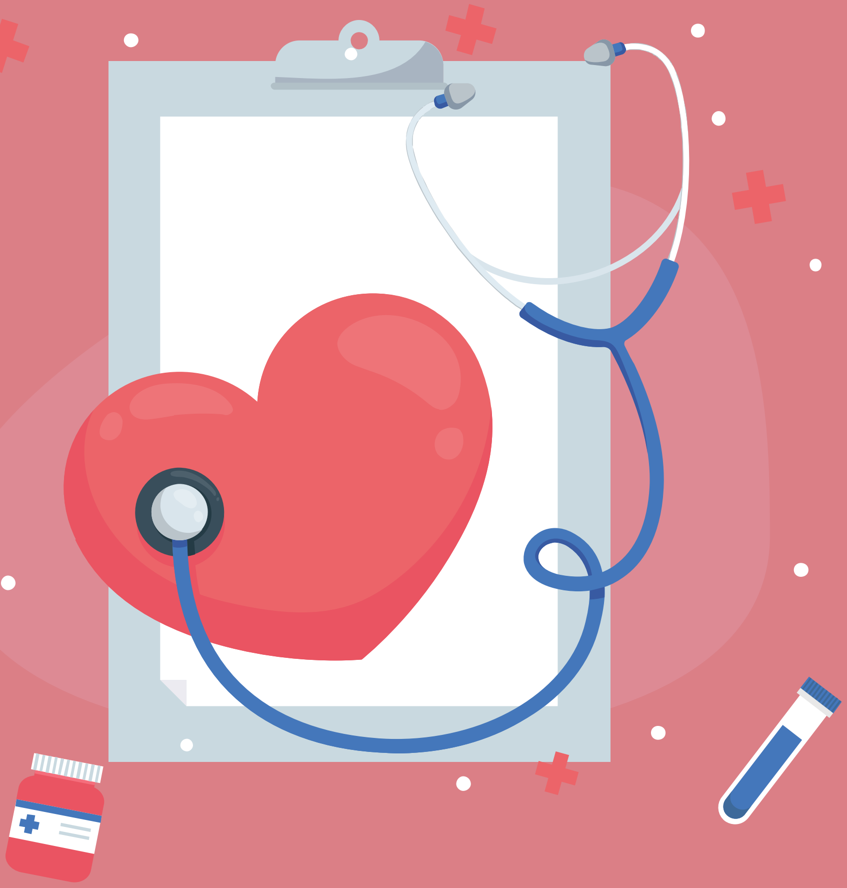
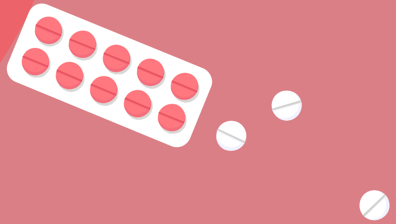
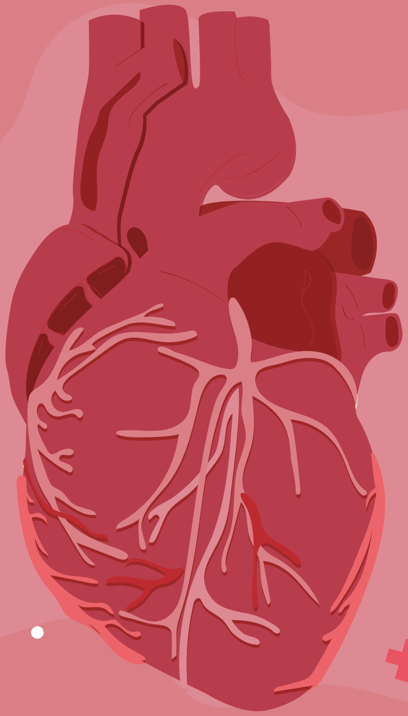
AI Model for Cardiovascular Disease Prediction







Cardiovascular Disease

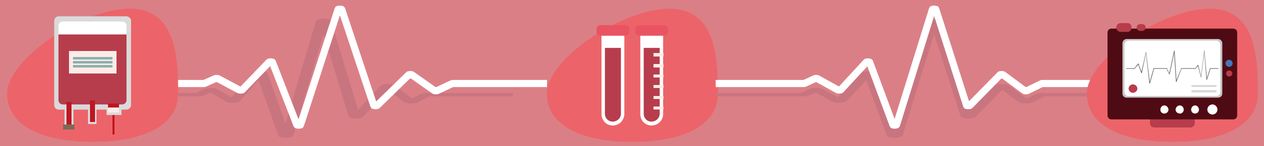
Any disease involving the heart or blood vessels, including coronary artery disease, stroke, heart failure, and more. Accounts for approximately 31% of global deaths. Early detection is key in halting progression of the disease and improving patient outcomes.

PREV NEXT

Goal

Create a machine learning model that can identify patients at high risk for cardiovascular disease using basic biometric data

Data Collection and Preparation



Dataset Selection & Cleaning

Our dataset has 64,000 unique patient records available to train and test our model. Data cleaning involved removing errors and outliers.

Feature Selection

The dataset had a total of 11 features available:

Objective Features: age, height, weight, gender

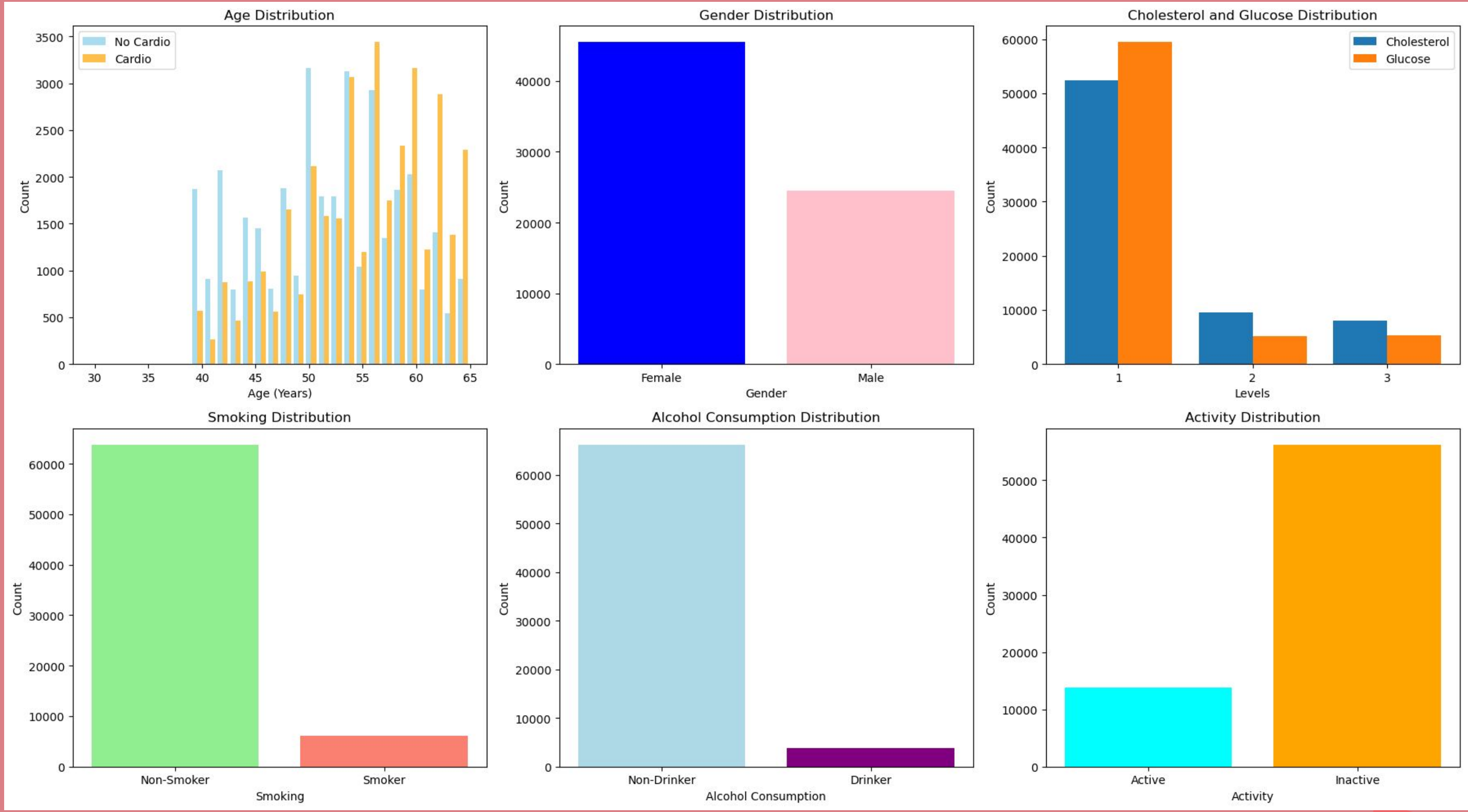
Examination Features: blood pressure (ap_hi, ap_lo), cholesterol level, glucose level

Subjective Features: drinking, smoking, activity level

Target Selection

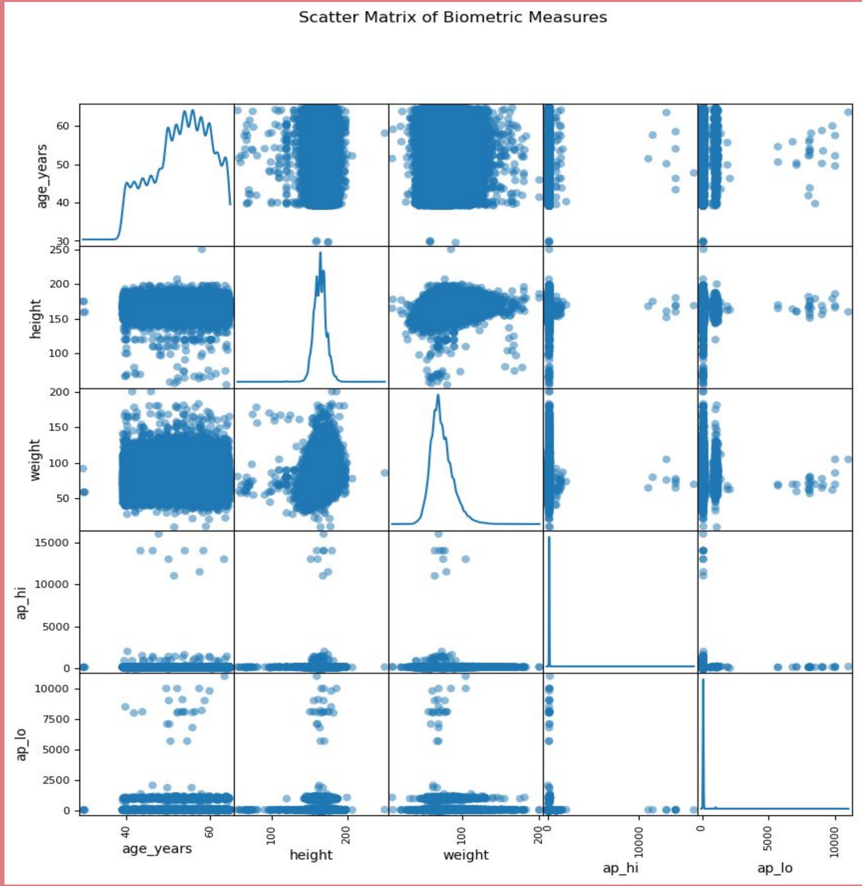
Our target (cardio) indicates the absence (0) or presence (1) of cardiovascular disease in the patient

The Dataset



Age Distribution
No Cardio
Cardio
Count
Age (Years)
0
500
1000
1500
2000
2500
3000
3500
30
35
40
45
50
55
60
65
Gender Distribution
Count
Gender
Female
Male
0
10000
20000
30000
40000
Cholesterol and Glucose Distribution
Cholesterol
Glucose
Count
Levels
1
2
3
0
10000
20000
30000
40000
50000
60000
Smoking Distribution
Count
Smoking
Non-Smoker
Smoker
Alcohol Consumption Distribution
Count
Alcohol Consumption
Non-Drinker
Drinker
Activity Distribution
Count
Activity
Active
Inactive

The Dataset



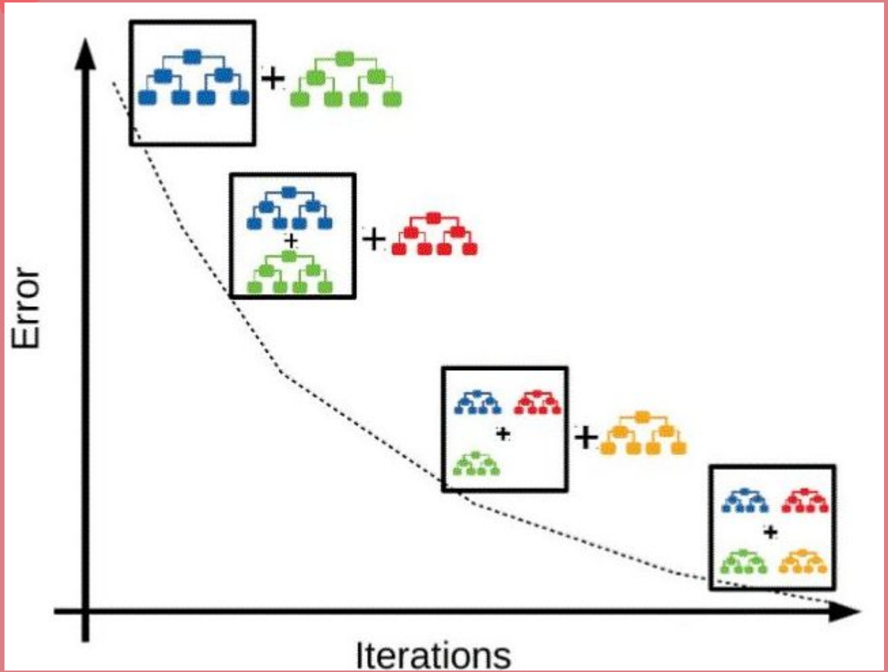
Scatter Matrix of Biometric Measures
age_years
height
weight
ap_hi
ap_lo

Model Selection: PyCaret

best = compare_models()

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
gbc	Gradient Boosting Classifier	0.7299	0.7963	0.6866	0.7504	0.7170	0.4597	0.4613	4.8000
lightgbm	Light Gradient Boosting Machine	0.7284	0.7947	0.6826	0.7501	0.7148	0.4567	0.4585	2.4790
catboost	CatBoost Classifier	0.7281	0.7941	0.6867	0.7473	0.7157	0.4560	0.4575	12.3960
ada	Ada Boost Classifier	0.7239	0.7903	0.6491	0.7618	0.7009	0.4476	0.4525	1.1860
xgboost	Extreme Gradient Boosting	0.7236	0.7884	0.6816	0.7427	0.7108	0.4470	0.4485	3.5890
ridge	Ridge Classifier	0.7212	0.0000	0.6491	0.7571	0.6989	0.4422	0.4468	0.0880
lda	Linear Discriminant Analysis	0.7212	0.7872	0.6491	0.7571	0.6989	0.4422	0.4468	0.1790
nb	Naive Bayes	0.7134	0.7827	0.5997	0.7746	0.6760	0.4265	0.4377	0.1670
rf	Random Forest Classifier	0.7132	0.7737	0.6906	0.7220	0.7059	0.4264	0.4268	6.2280
qda	Quadratic Discriminant Analysis	0.7053	0.7687	0.6187	0.7467	0.6767	0.4102	0.4164	0.0900
et	Extra Trees Classifier	0.7032	0.7625	0.6849	0.7096	0.6970	0.4063	0.4065	6.3030
lr	Logistic Regression	0.6977	0.7522	0.6496	0.7173	0.6817	0.3951	0.3969	0.6490
knn	K Neighbors Classifier	0.6337	0.6726	0.5852	0.6464	0.6142	0.2671	0.2683	0.5740
dt	Decision Tree Classifier	0.6284	0.6284	0.6287	0.6270	0.6278	0.2569	0.2569	0.4020
svm	SVM - Linear Kernel	0.5616	0.0000	0.4451	0.6945	0.4118	0.1225	0.1815	2.1570
dummy	Dummy Classifier	0.5015	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.1010

Gradient Boosting Classifier·



Gradient Boosting Classifier - a special type of Ensemble Learning technique that works by combining several weak learners (predictors with poor accuracy) into a strong learner (a model with strong accuracy). Each new model takes a step in the direction that minimizes prediction error.

Gradient Boosting Classifier

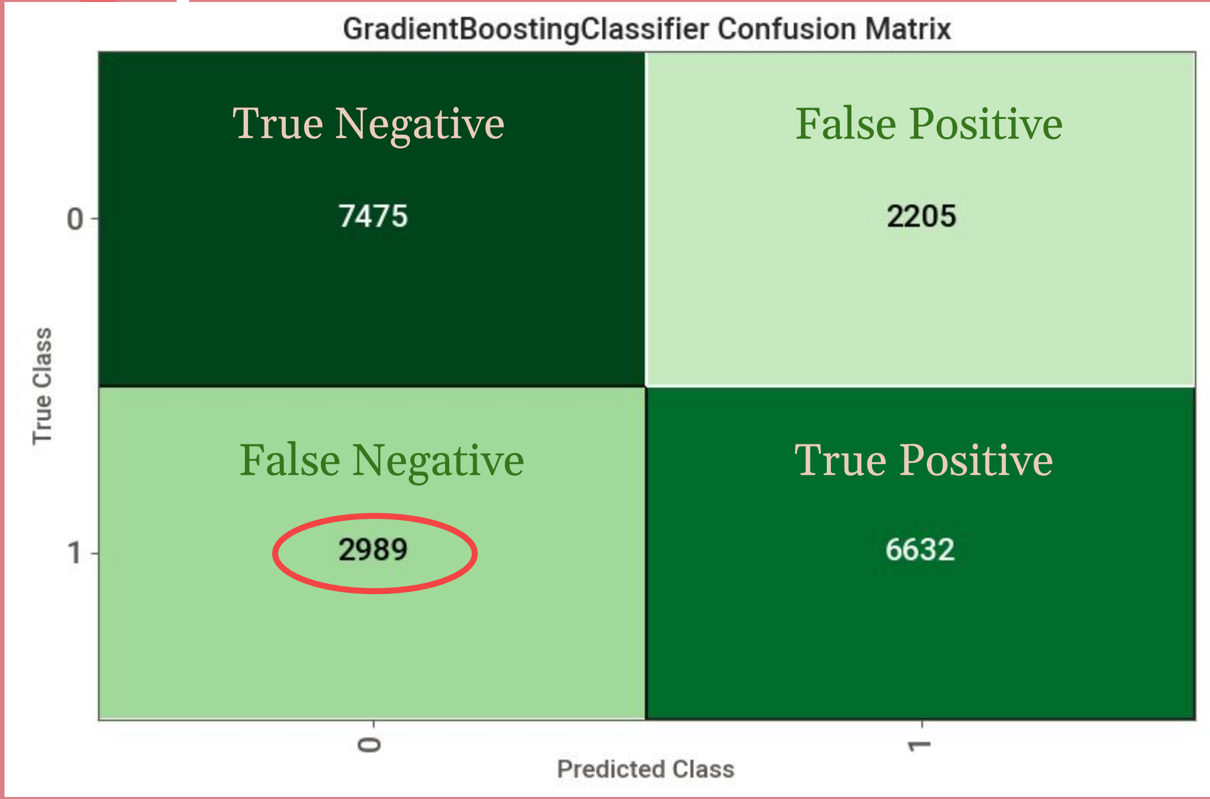
Hyperparameters of the best model

	Parameters
ccp_alpha	0.0
criterion	friedman_mse
init	None
learning_rate	0.1
loss	log_loss
max_depth	3
max_features	None
max_leaf_nodes	None
min_impurity_decrease	0.0
min_samples_leaf	1
min_samples_split	2
min_weight_fraction_leaf	0.0
n_estimators	100
n_iter_no_change	None
random_state	123
subsample	1.0
tol	0.0001
validation_fraction	0.1
verbose	0
warm_start	False

Hyperparameters changed for tuning the model

Max_depth
N_iter
Min_samples_leaf
N-estimators
Optimization (AUC, recall)

Evaluation: Confusion Matrix



Recall / Precision:

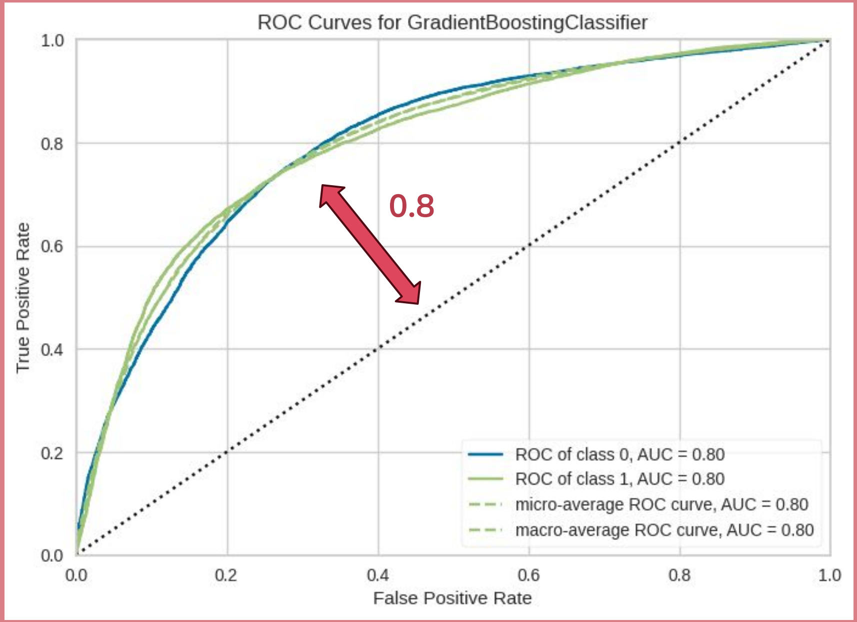
True Negative - 39%

True Positive - 34%

False Positive - 11%

False Negative - 15%

Evaluation: Area Under Curve Accuracy



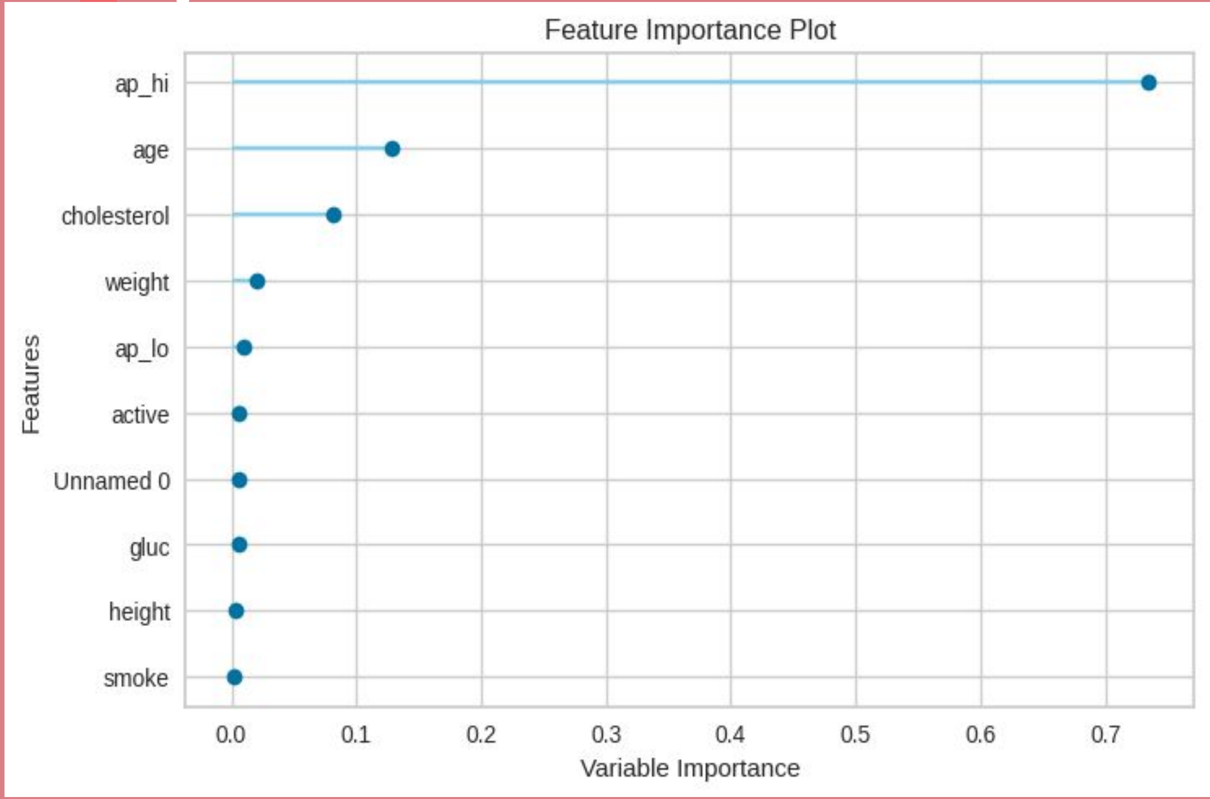
AUC - ROC Curve

AUC - Area Under Curve

ROC - Receiver Operating Characteristics

AUC of 0.8 to 0.9 is excellent.

Evaluation: Feature Importance



Feature Importance:

1. Systolic Blood Pressure
2. Age
3. Cholesterol Level

Tensorflow Neural Network Model

```python
#initiate sequential model
model = tf.keras.models.Sequential()

# input layer
model.add(tf.keras.layers.Dense(units=X.shape[1], activation='relu', input_shape=(11,)))

# additional layers
model.add(tf.keras.layers.Dense(units=30, activation='relu'))
model.add(tf.keras.layers.Dense(units=40, activation='relu'))
model.add(tf.keras.layers.Dense(units=50, activation='relu'))
model.add(tf.keras.layers.Dense(units=40, activation='relu'))
model.add(tf.keras.layers.Dense(units=30, activation='relu'))
model.add(tf.keras.layers.Dense(units=40, activation='relu'))

# Output layer
model.add(tf.keras.layers.Dense(units=2, activation='softmax'))

# Compile the model
model.compile(loss=tf.keras.losses.CategoricalCrossentropy(from_logits=False),
              optimizer='adam',
              metrics=[tf.keras.metrics.CategoricalAccuracy()])

# Early stopping
early_stopping = tf.keras.callbacks.EarlyStopping(patience=10, restore_best_weights=True)

# Train the model
history = model.fit(X_train_scaled, y_train_one_hot, epochs=100, batch_size=64,
                    validation_data=(X_test_scaled, y_test_one_hot),
                    callbacks=[early_stopping])
```

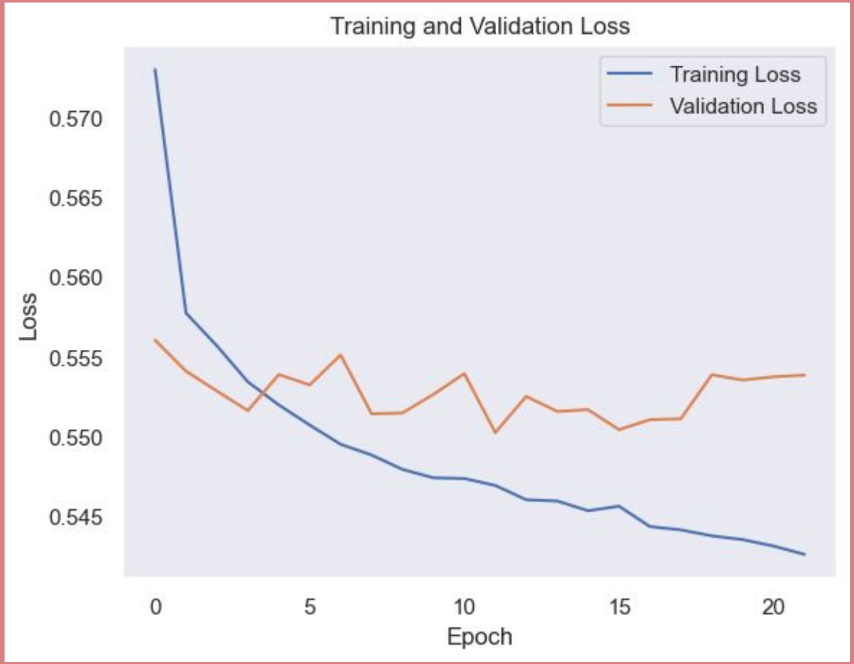
Accuracy: 73%

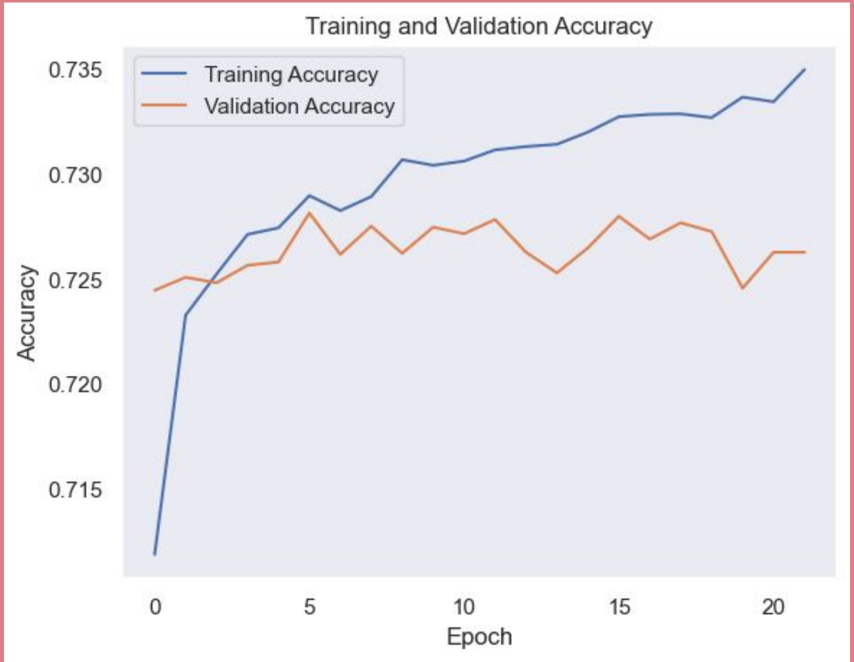
Val Accuracy: 72%

Loss: 54%

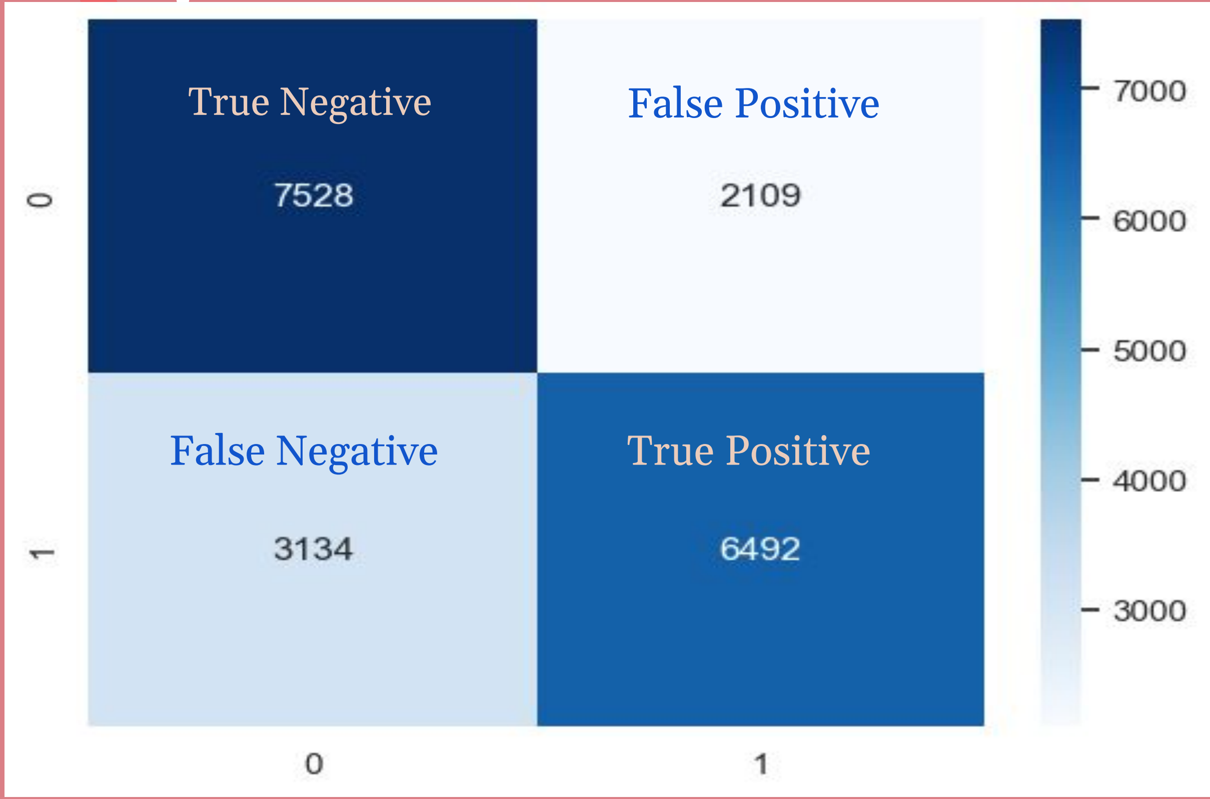
Val Loss: 55%

Evaluation: Training and Validation Loss and Accuracy





Evaluation: Confusion Matrix



Recall / Precision:

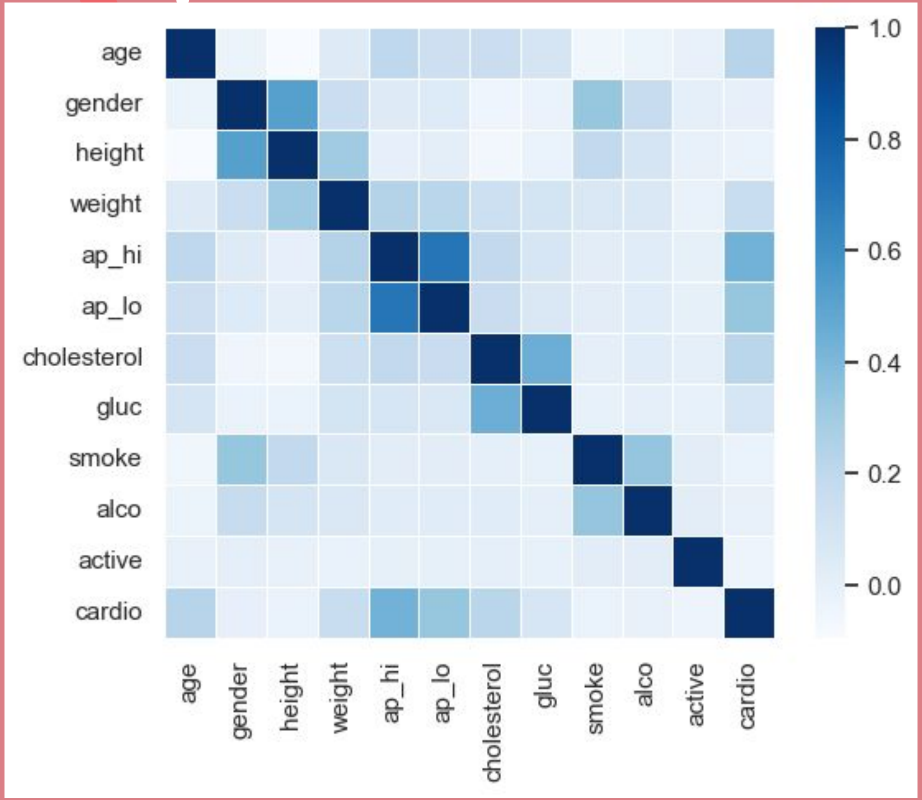
True Negative - 39%

True Positive - 33%

False Positive - 11%

False Negative - 17%

Evaluation: Correlation Matrix



Best Correlations:

Systolic Blood Pressure

Age

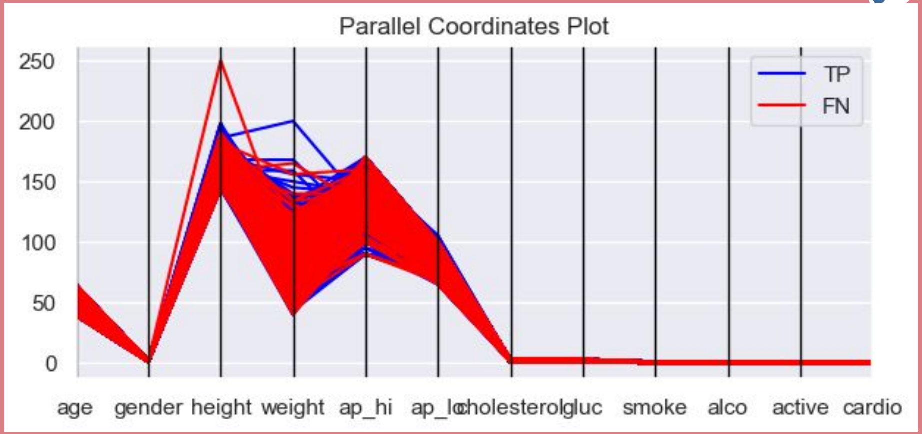
Cholesterol

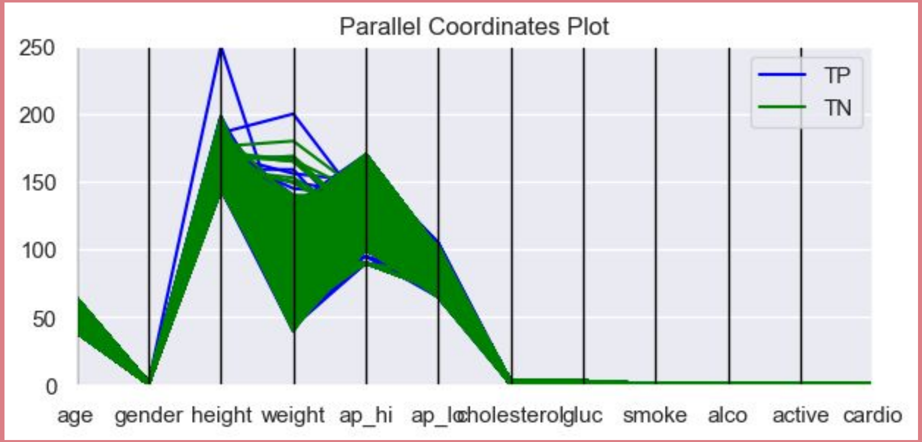
Overall Correlation:

Poor

Evaluation: Parallel Coordinates Plots

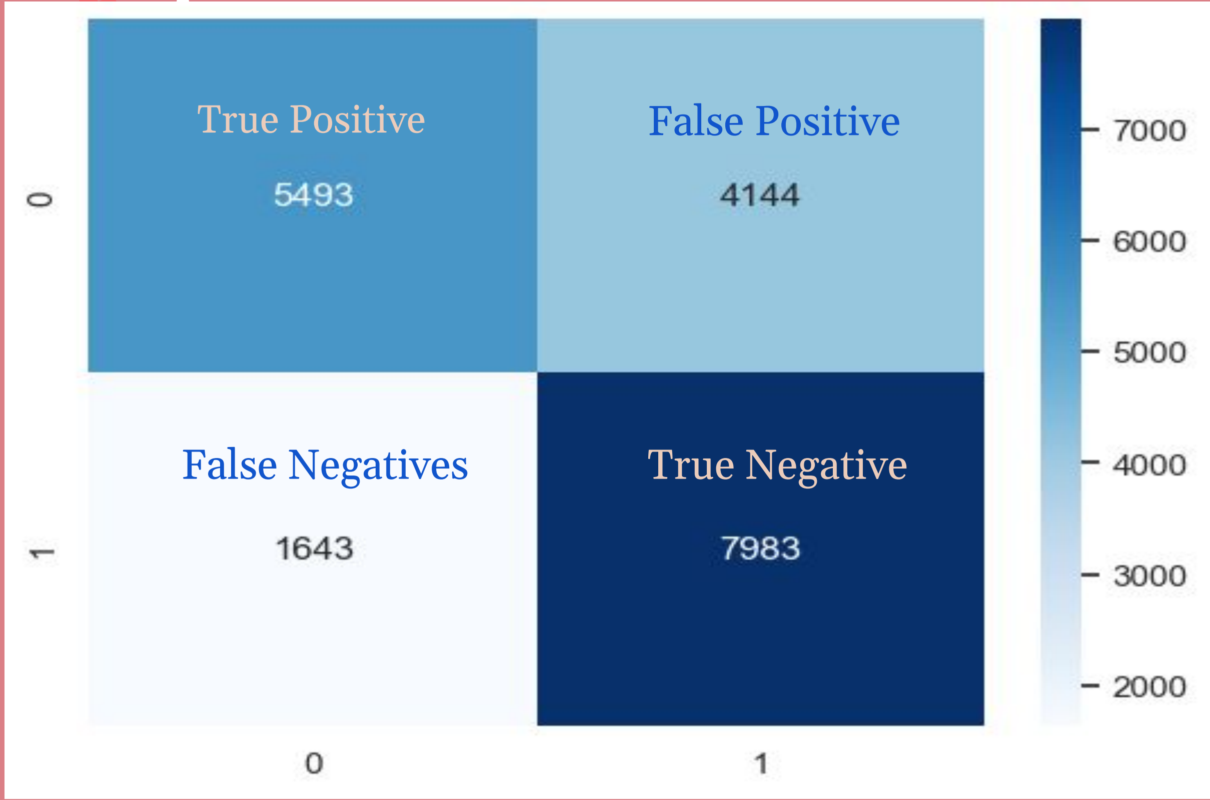
True Positive vs False Negative





True Positive vs True Negative

Adjusting Threshold for Classification



Recall / Precision:

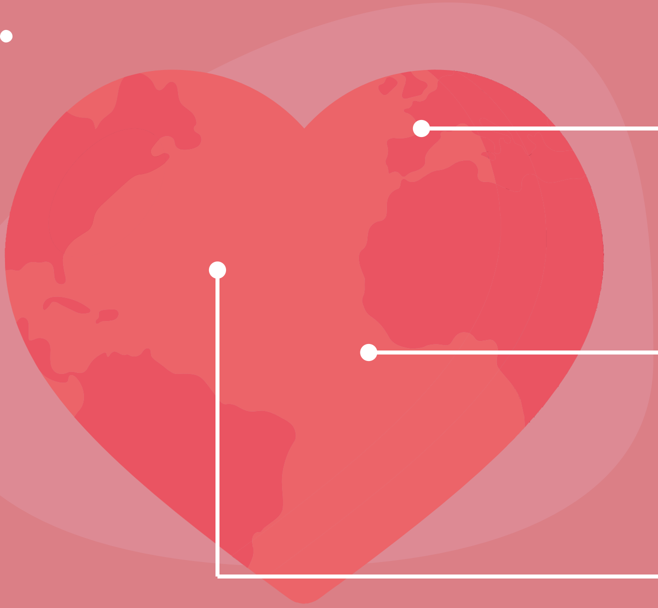
True Negative - 28%

True Positive - 41%

False Positive - 21%

False Negative - 10%

Limitations



1. Simplistic data
2. Overall poor correlations
3. Minimal room for optimization

Optimization

Higher Quality Data	Statistical Analysis	Hyperparameter Tuning
Our dataset is simplistic and user-friendly. More scientific data could improve accuracy.	More advanced statistical analysis methods can be used to guide feature weighting.	It fixes everything, right?

Conclusion

Keep your
hearts healthy!