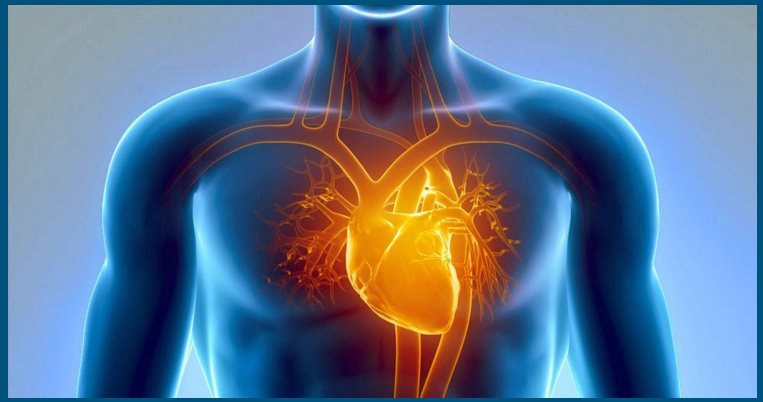
Heart Disease Classification Through Machine Learning Techniques

DS 320 Term Project
By Sarah Petro and Jane Schneider

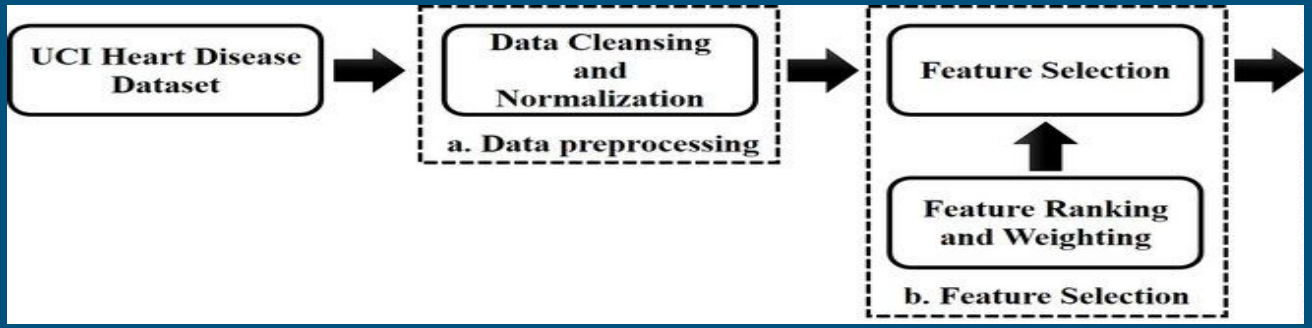
Introduction & Background

- Heart disease is one of the leading causes of death in the United States
- Machine learning techniques to accurately diagnose disease and symptoms
- Complicated disease and symptoms
- Data integration of 3 heart disease datasets
- Tested 4 different ML models



Challenges & Related Work

- Feature selection is the most difficult challenge
- “Classification models for heart disease prediction using feature selection and PCA” by Anna Karen Gárate-Escamila et al. in 2020
- “Heart Disease Classification Using Neural Network and Feature Selection” by A. Khemphila and V. Boonjing in 2011



Data Preprocessing and Integration

- Integrated 3 datasets to make one larger dataset
  - Hungary, Switzerland, and Cleveland
- 300 observations
- Dropped NA values
- Performed one hot encoding on categorical variables for classification

```
#perform one hot encoding on categorical variables and then change numerical 0,1,2,3,4 values into 0/1 binary for classification
dataset['thal'].replace({'fixed defect':'fixed_defect' , 'reversable defect': 'reversable_defect' }, inplace =True)
dataset['cp'].replace({'typical angina':'typical_angina', 'atypical angina': 'atypical_angina' }, inplace =True)


data_tmp = dataset[['age','sex','cp', 'trestbps', 'chol', 'fbs',  'thalch', 'exang', 'oldpeak', 'slope', 'ca', 'thal']].copy()
data_tmp['target'] = ((dataset['num'] > 0)*1).copy()
data_tmp['sex'] = (dataset['sex'] == 'Male')*1
data_tmp['fbs'] = (dataset['fbs'])*1
data_tmp['exang'] = (dataset['exang'])*1

data_tmp.columns = ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure',
              'cholesterol', 'fasting_blood_sugar',
              'max_heart_rate_achieved', 'exercise_induced_angina',
              'st_depression', 'st_slope_type', 'num_major_vessels',
              'thalassemia_type', 'target']
data_tmp.head(15)
```

	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_d
0	63	1	typical_angina	145.0	233.0	1	150.0	0	
1	67	1	asymptomatic	160.0	286.0	0	108.0	1	
2	67	1	asymptomatic	120.0	229.0	0	129.0	1	
3	37	1	non-anginal	130.0	250.0	0	187.0	0	
4	41	0	atypical_angina	130.0	204.0	0	172.0	0	

Test-Train Split

- Implemented test-train split to separate original data for performance evaluation
- X_train, y_train, X_test, y_test
- Min-Max normalization of X_train and X_test

Steps:

- Train the model with X_train and y_train
- Use model to predict y_predict from X_test
- Compare y_predict against y_test for evaluation

```
from sklearn.model_selection import train_test_split
y = data['target']
X = data.drop('target', axis = 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
print(f'Shape of X_train: {X_train.shape}')
print(f'Shape of y_train: {y_train.shape}')
print(f'Shape of X_test: {X_test.shape}')
print(f'Shape of y_test: {y_test.shape}')
```

```
Shape of X_train: (239, 21)
Shape of y_train: (239,)
Shape of X_test: (60, 21)
Shape of y_test: (60,)
```

```
[ ] X_train=(X_train-np.min(X_train))/(np.max(X_train)-np.min(X_train)).values
    X_test=(X_test-np.min(X_test))/(np.max(X_test)-np.min(X_test)).values
    X_test
```

	age	sex	resting_blood_pressure	cholesterol	max_heart_rate_achieved	st_depression
209	0.692308	0.0	0.651163	0.566929	0.622642	0.225806
190	0.384615	1.0	0.406977	0.377953	0.707547	0.000000
12	0.538462	1.0	0.418605	0.614173	0.509434	0.096774
222	0.102564	0.0	0.000000	0.389764	0.858491	0.000000
240	0.153846	1.0	0.186047	0.531496	0.613208	0.000000
137	0.692308	1.0	0.302326	0.712598	0.141509	0.225806

Model Building

- Paid close attention to performance metrics: accuracy, F1-score, precision, and recall
- 4 Models with tuning of hyperparameters::
  - Logistic Regression
  - Decision Tree (Balanced and Unbalanced)
  - Gradient Boosting (ensemble technique)
  - Random Forest (Balanced and Unbalanced)

$$\text{precision} = \frac{\text{tp}}{\text{tp} + \text{fp}}$$

$$\text{recall} = \frac{\text{tp}}{\text{tp} + \text{fn}}$$

$$\text{accuracy} = \frac{\text{tp} + \text{tn}}{\text{tp} + \text{tn} + \text{fp} + \text{fn}}$$

$$\text{F}_1 \text{ score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

Models, Results, and Metrics

Logistic Regression:

- Accuracy: 0.83
- Similar performance to other models

Gradient Boosting Classifier:

- Accuracy: 0.83
- Ensemble learning; best precision, recall, f1-score

Decision Tree:

- Accuracy: 0.78 unweighted, 0.68 weighted
- Performed worst out of all the models

Random Forest:

- Accuracy: 0.83
- High precision

Best Model: Gradient Boosting Classifier

- Highest f1-score; makes up for slightly lower accuracy
- Best evaluation metric: F1 score
  - Best for data with unbalanced classes

```
gradient_booster = GradientBoostingClassifier(learning_rate=0.02, max_depth=3, n_estimators=150)
gradient_booster.fit(X_train, y_train)
y_pred = gradient_booster.predict(X_test)

print('The Accuracy Score is: ', accuracy_score(y_test,y_pred))
print(classification_report(y_test, y_pred))
```

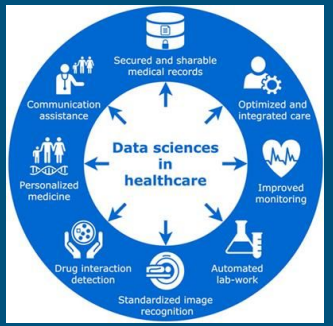
```
The Accuracy Score is:  0.8333333333333334
              precision    recall  f1-score   support

           0       0.83      0.83      0.83        29
           1       0.84      0.84      0.84        31

    accuracy                           0.83        60
   macro avg       0.83      0.83      0.83        60
weighted avg       0.83      0.83      0.83        60
```

Conclusion

- Heart disease classification is inherently difficult to study
  - Genetic and environmental factors difficult to quantify
- Research is ongoing
  - researchers Anna Karen Garate-Escamila et al. have achieved 98%-99% accuracy
- Data Science: the future of medical diagnosis
  - Great impact on medical research
  - Provide mathematical insights that would go unnoticed by human capabilities



Lessons Learned

- Data Integration:
  - Combining 3 smaller datasets (Hungary, Switzerland, Cleveland) into one large dataset
- Data Preprocessing:
  - Removing NA values
  - Encoding of categorical attributes
  - Min-max normalization for training classification models
- Model Building, Tuning, and Evaluation:
  - Test-Train split to measure model performance
  - F1-score for evaluation of models with unbalanced training data