# Predicting Heart Disease At Early Stage

Capstone Project – Springboard Data Science Career Track

Hehuan Liao

Palo Alto, CA
Aug. 20, 2020

# Problem Identification

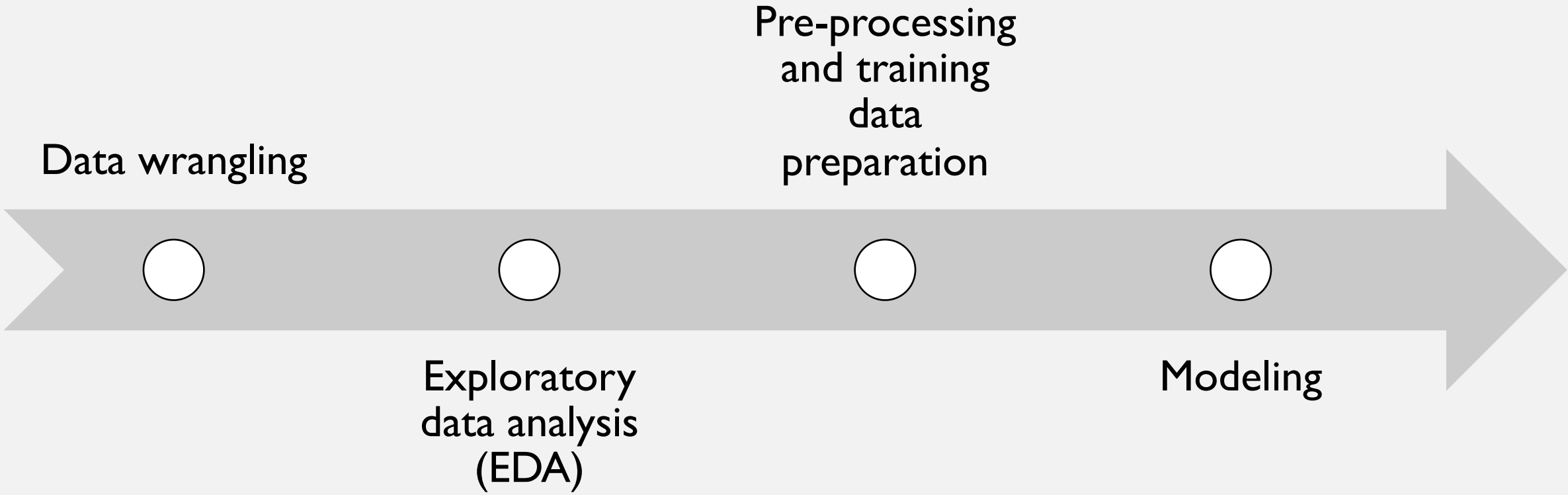
- Heart disease is the leading cause of death
  - > 600,000 / year related deaths in US (1 in every 4 deaths)
- Heart disease produces immense health and economic burdens
  - Costs associated with heath care services, medicines, and lost productivity due to death is about $219 billion / year in US
- Early detection is critical to reduce the mortality associated with heart disease
  - About 1 in 3 deaths is preventable
  - Predictive model based on traditional risk factors remains a rapid, cost-effective, and accurate clinical tool

👉 Given a set of health parameters from routine monitoring, can we robustly predict the risk of heart disease as early as possible?

# Dataset description

- Data source: UCI Machine Learning Repository
- 303 observations of 14 variables
  - Features
    - age, sex
    - chest pain type, fasting blood sugar, resting electrocardiographic results, the slope of the peak exercise ST segment, thalassemia (a blood disorder), and number of major vessels colored by fluoroscopy
    - resting blood pressure, serum cholesterol, maximum heart rate achieved, ST depression induced by exercise relative to rest
  - Target: diagnosis of heart disease (0=no, 1=yes)

# Methodology



# Data wrangling

## Detecting & filling the missing values

```
In [7]: df.isnull().sum().sort_values(ascending=True)

Out[7]: age         0
        sex         0
        cp          0
        trestbps    0
        chol        0
        fbs         0
        restecg     0
        thalach     0
        exang       0
        oldpeak     0
        slope       0
        ca          0
        thal        0
        target      0
        dtype: int64
```

```
In [8]: df.ca.value_counts()

Out[8]: 0.0    176
        1.0     65
        2.0     38
        3.0     20
        ?        4
        Name: ca, dtype: int64
```

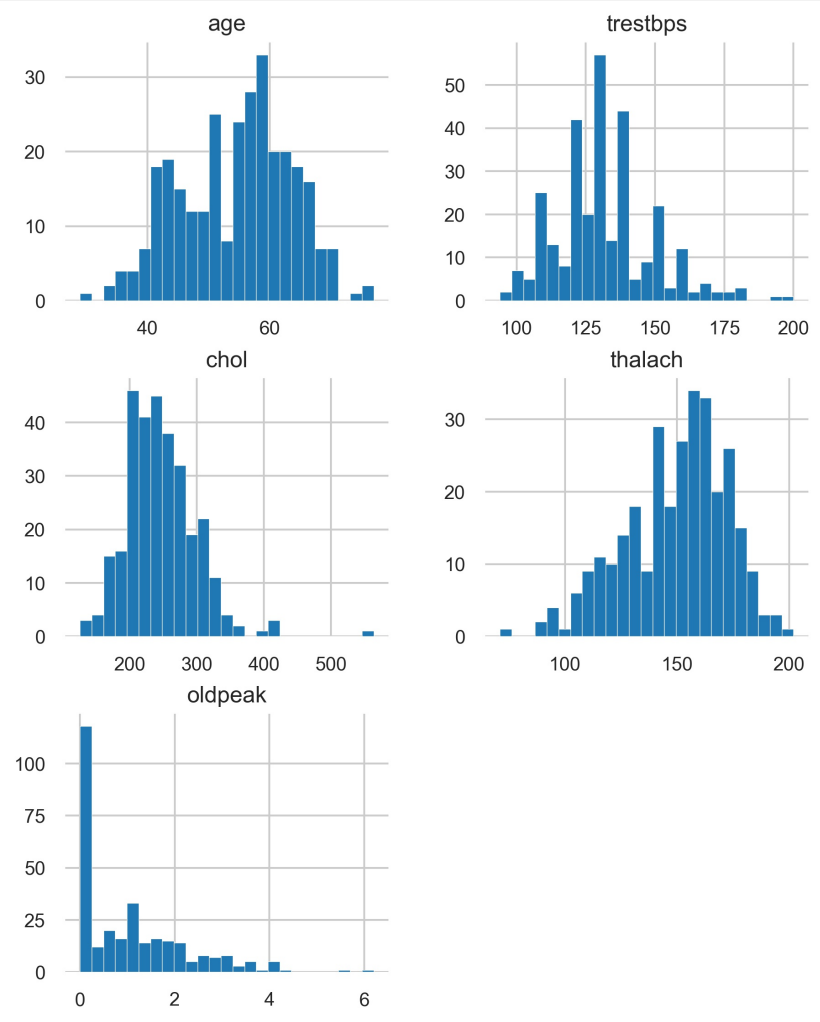
```
In [9]: df.thal.value_counts()

Out[9]: 3.0    166
        7.0    117
        6.0     18
        ?        2
        Name: thal, dtype: int64
```

```
In [10]: # replace the '?' with np.nan
         df.replace('?',np.nan,inplace=True)
```

```
In [11]: #fill na with mode
         df[['ca','thal']] = df[['ca','thal']].fillna(df[['ca','thal']].mode().iloc[0])
```

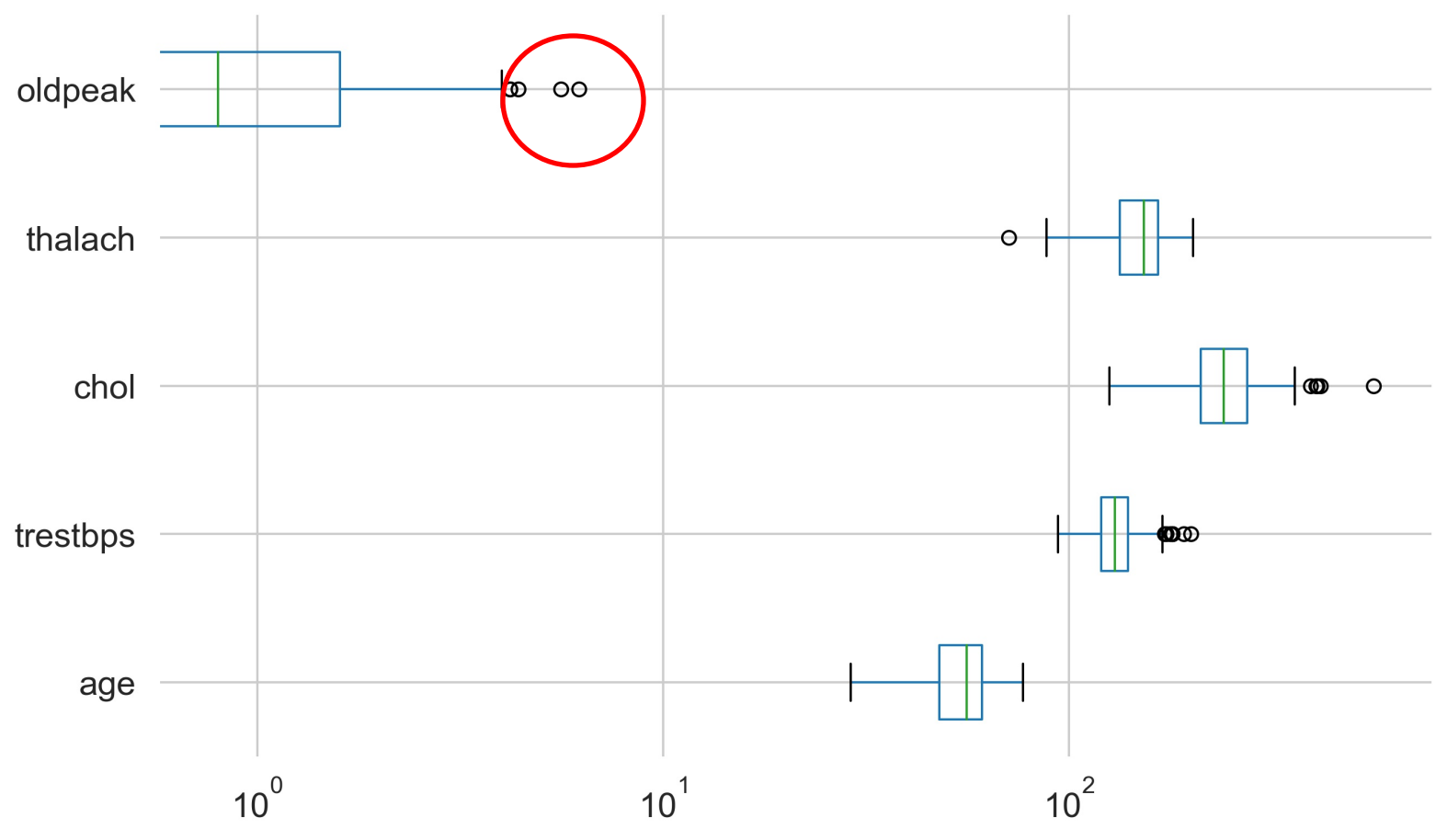
# Exploratory data analysis (EDA)




Non-normal distribution

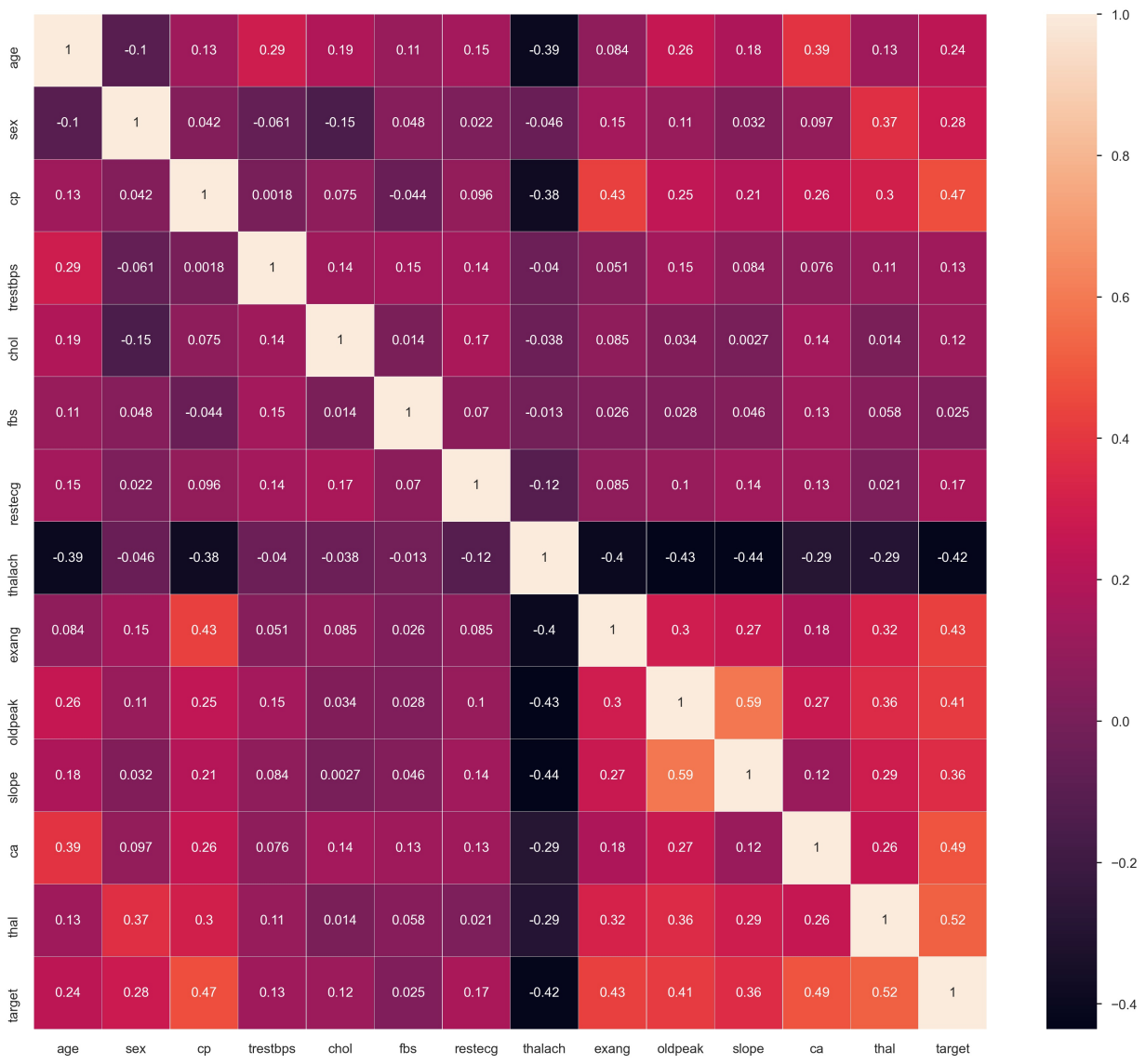

Target relatively balanced

# Exploratory data analysis (EDA)




→ Outliers/extreme values: be careful with models that are sensitive to outliers

# Exploratory data analysis (EDA)




→ Variables are not highly correlated

# Pre-processing and training data preparation

- Create dummy features for categorical variables:
  - sex, cp, restecg, exang, and slope
- Training and test set split:
  - use stratification to maintain the ratio of classes in target
- Standardize the magnitude of numerical features in the training set, and apply the same scaler to the test set
  - Use standard scaler (mean=0, variance=1)

# Modeling

- Supervised learning
  - Binary classification:

    0 = absence of heart disease

    1 = presence of heart disease
  - Binary classifiers →

- Python tools

  Numpy, Pandas

  Matplotlib, Seaborn

  Scikit-learn

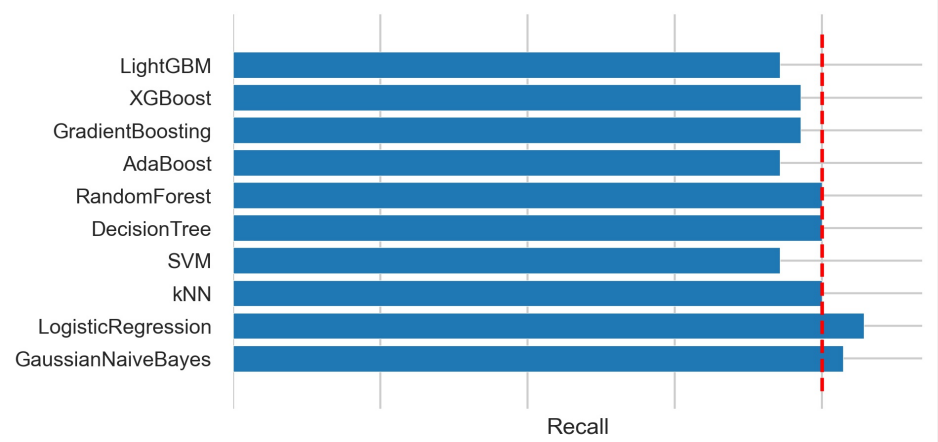
  Plotly, Dash

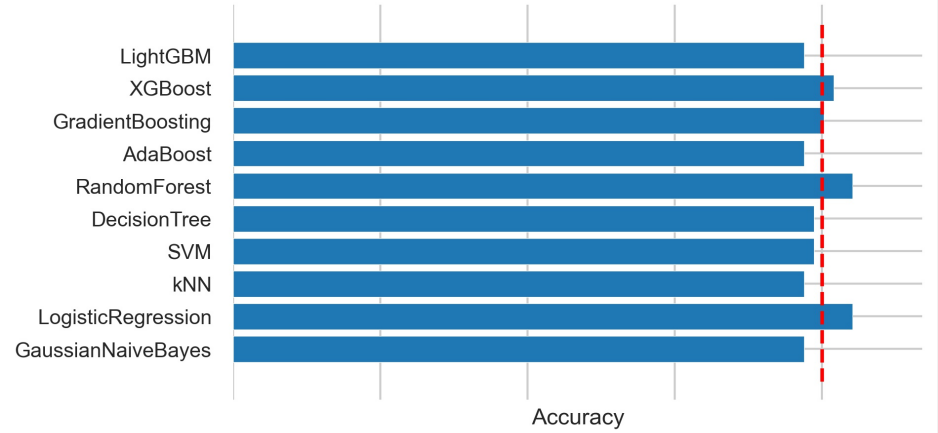
| Binary Classifier | | Advantages | Disadvantages |
|---|---|---|---|
| Naïve Bayes | | Simple, fast, low computation cost, and accurate | Cannot learn interactions between features |
| Logistic Regression | | Lots of ways to regularize the model (e.g. lasso, ridge), and don't have to worry as much about features being correlated, like in Naive Bayes; have a nice probabilistic interpretation; feature scaling is not a requirement | Poor performance on non-linear data (e.g. images) |
| k-Nearest Neighbors | | No assumptions about data; simple and intuitive, relatively high accuracy; constantly evolving model | Curse of dimensionality; feature scaling is an absolute must; does not perform well on imbalanced data; sensitive to outliers; slow for large dataset |
| Support Vector Machine | | Good performance over high-dimension data (e.g. images), and is not sensitive to outliers | Poor performance with overlapping classes, and is sensitive to the type of kernel used; hyperparameter tuning is important |
| Decision Tree | | Feature scaling is not needed; Easy to explain and visualize | Prone to overfitting |
| Ensemble - bagging | Random Forest | Easy to interpret and explain; can handle feature interactions, non-parametric; fast and scalable | Don't support online learning |
| Ensemble - boosting | AdaBoost | Low generalization error, easy to implement, and works with many classifiers | Sensitive to outliers |
| | Gradient boosting | High accuracy and flexibility | Sensitive to outliers and computationally expensive |
| | XGBoost | Feature scaling is not needed; computational efficiency and often better model performance | Difficult to interpret and visualize; hard to tune (a lot hyperparameters) |
| | LightGBM | high speed, high accuracy, can use categorical features as input directly | Prone to overfitting |

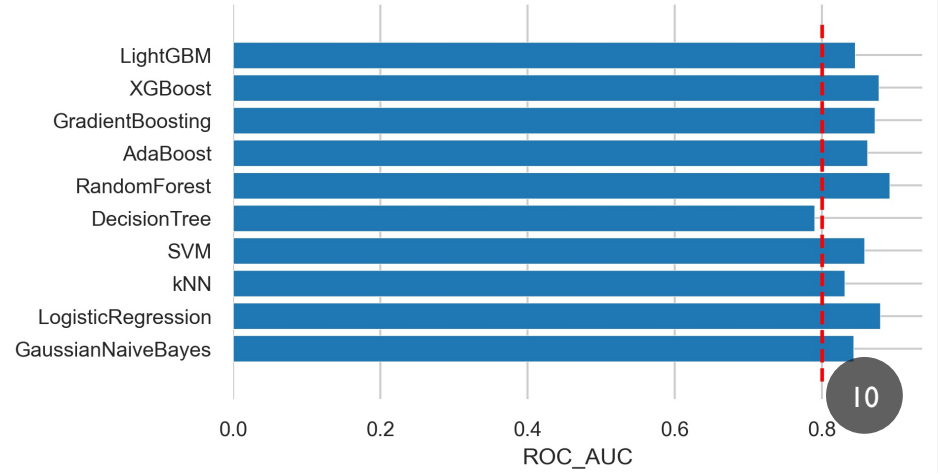
# Modeling: evaluation

- Train the 10 models with default settings using the training set, and evaluate the model performance using the test set

- Logistic regression, random forest were selected for optimization with grid search



# Modeling: hyperparameter tuning

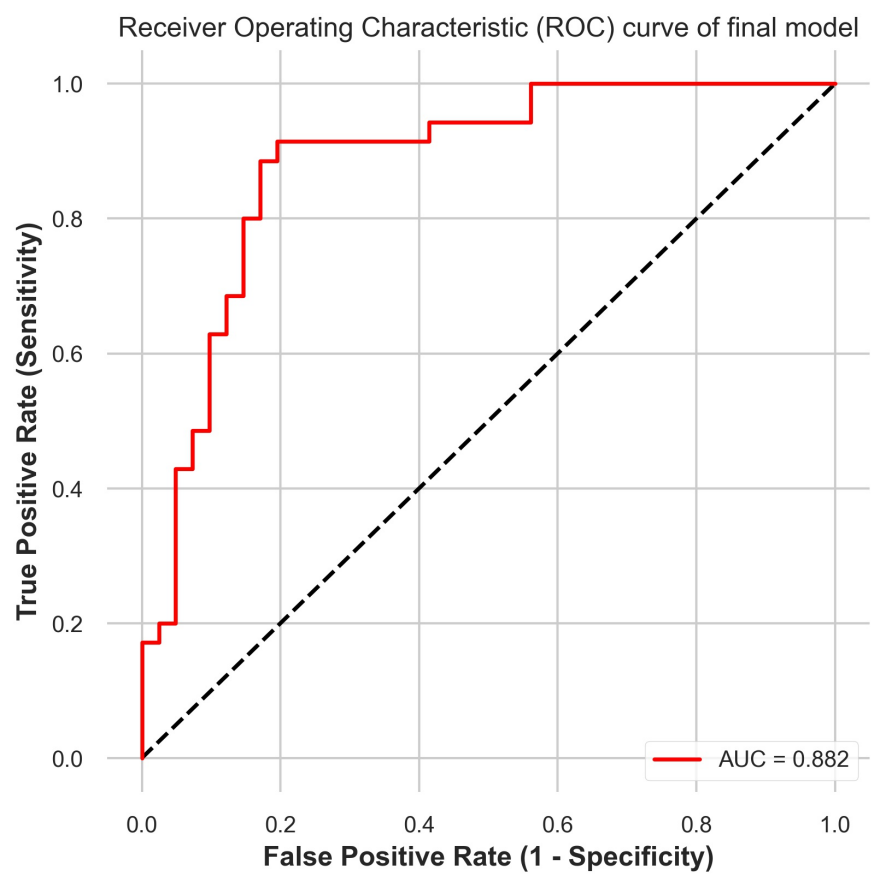
| model | precision | recall | accuracy | mcc | roc_auc |
|---|---|---|---|---|---|
| LogisticRegression | 0.811 | 0.857 | 0.842 | 0.685 | 0.879 |
| LogisticRegression_tuned | 0.811 | 0.857 | 0.842 | 0.685 | 0.882 |
| RandomForest | 0.848 | 0.8 | 0.842 | 0.682 | 0.892 |
| RandomForest_tuned | 0.853 | 0.829 | 0.855 | 0.708 | 0.886 |

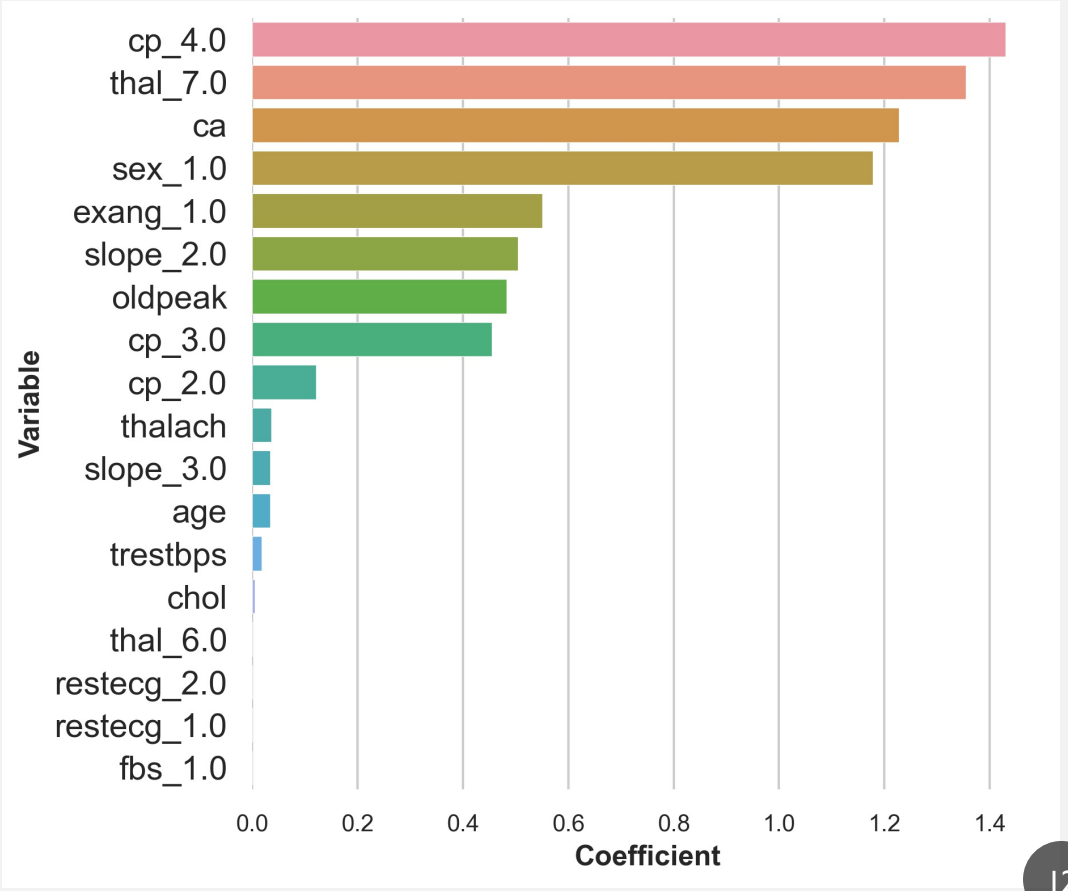
→ Hyperparameter tuning further increased the performance of random forest

→ In contrast, the performance of the tuned logistic regression model has only slightly increased in terms of roc-auc score.

# Modeling: final model

Lr_final = LogisticRegression (C=1.0, penalty='L1', solver='liblinear')



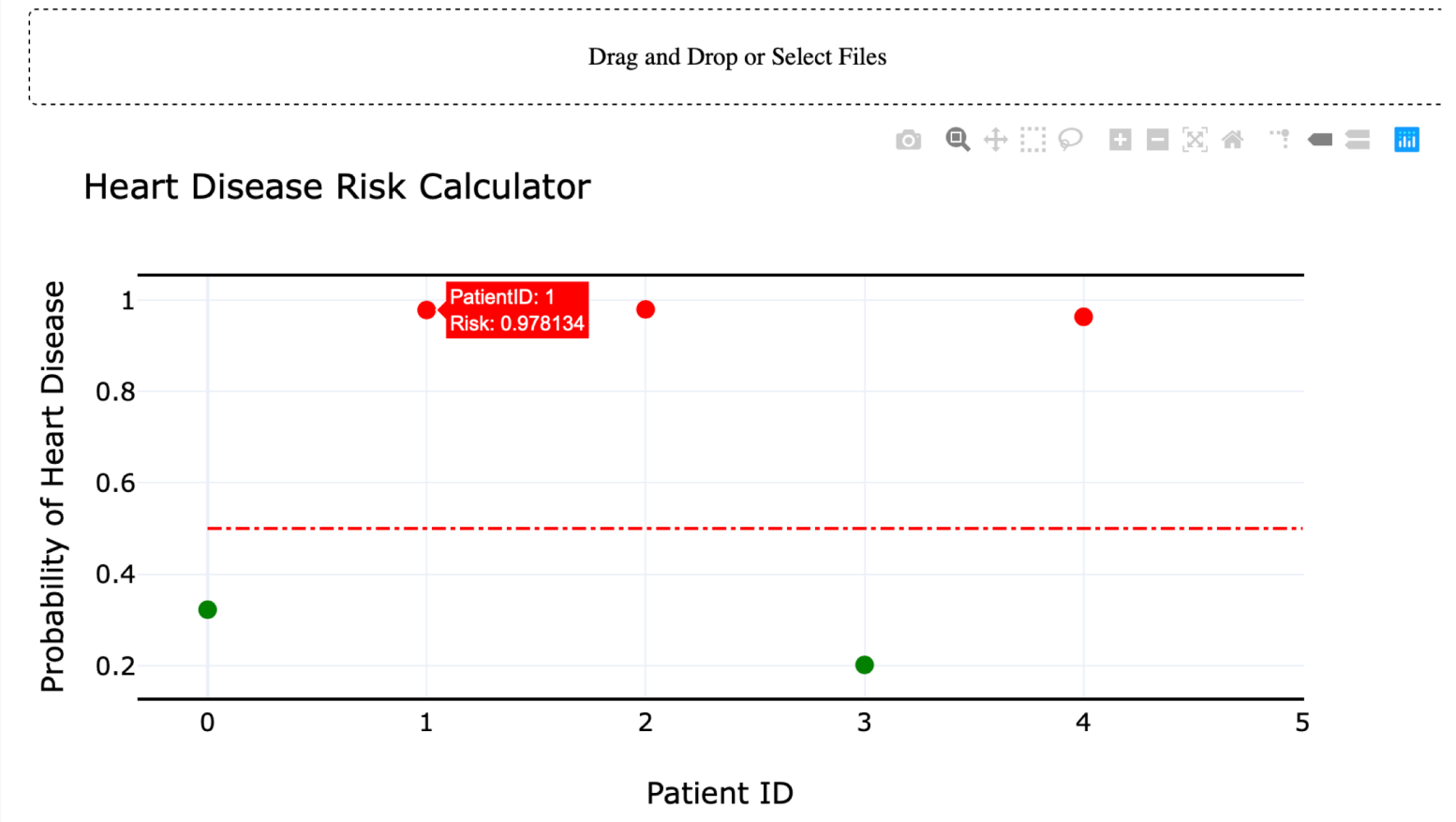



# Conclusions

Heart disease is the leading cause of death in the US and worldwide, which produces immense health and economic burdens. Identifying those at increased risk for heart disease at earliest stage is critical to reduce the mortality

- While advanced 'omics' technologies have emerged as promising ways of understanding risk of disease mechanistically, predictive model based on traditional risk factors remains an important clinical tool that is rapid, cost-effective, and can be as accurate as molecular methods
- A comparison of 10 binary classification algorithms suggested that logistic regression and random forest were among the models with top performance, which were further optimized via hyperparameter tuning
- Logistic regression with tuned hyperparameters (C=1.0, penalty='l1', solver='liblinear') was selected as the final model
  - Highest recall score – minimizing false negatives
  - Substantially faster
  - Easier to incorporate more training data once they become available.
- Future improvements
  - Additional feature engineering
  - Developing a user-friendly interactive dashboard tool

# Work in Progress:



# Acknowledgements


- Data source: UCI Machine Learning repository
- Springboard team
  - Dipanjan Sarkar
  - Kenneth Gil-Pasquel


The entire project is available:
https://github.com/hehuanliao/Springboard/blob/master/Capstone2/heart_disease_ML.ipynb