HEART DISEASE REFERRAL APPLICATION

Terryann Barnett
Osman Dumbuya
Michael Roberts

Data Analytics Boot Camp Program
Columbia University School of Engineering
December 14, 2023

INTRODUCTION

- This is a supervised machine learning project which uses Python (Pandas, NumPy, Scikit-learn, Flask, Joblib) HTML, CSS, and JavaScript technologies.

- The project aims to build a binary classifier, which performs with at least 90% accuracy, to make predictions about the risk of heart disease.

- The project features a full stack-front and back end development tools-to create a functional application that satisfies modern industry standards.

BACKGROUND AND PURPOSE

- Heart Disease is among the most prevalent chronic diseases in the United States.
- It claims roughly **647,000** lives each year!
- NuCare Health Insurance Company is concerned about 2 phenomena which occurred in 2015:
  - (i) 80% of their cardiology specialist referrals were for cases that had low risk for heart disease.
  - (ii) The expenditure for cardiology speciality care was 40% more than the previous year.
- The National Health Insurance Equity Board is looking to lower the number of "No Referral Needed from Primary Care" appointments to make more specialist care spots available to patients who need them the most.
- Our organization is contracted to develop an application which can be used as a baseline for primary care doctors to use to generate codes to make cardiology referrals.

FEATURES

- ETL
- RANDOM FOREST BINARY CLASSIFIER
- BACKEND DEVELOPMENT
- FRONT END DEVELOPMENT

WORKFLOW: Extract, Transform, and Load (ETL)

- Convert 500mb size dataset of more than 300 columns and 440k rows to 67k columns and 14 rows
- Utilized dataset documentation (Codebook) and advise of medical clinician to determine columns in the dataset that illustrated health conditions that correlate strongly with heart disease
- Rename columns to make them more readable and drop data rows with null (empty) values
- Ensure that columns had binary values (0 or 1) to match expected input data
- Separate the dataset into labels (heart disease) and features (13 categories) in preparation to develop training and testing data to use with a Binary Classifier Model

WORKFLOW: Binary Classifier Model Build - EVALUATION

Balanced Accuracy: 0.8461485786755695

	PRECISION	RECALL	F1-SCORE	SUPPORT
LOW RISK	.93	.99	.96	10767
HIGH RISK	.96	.70	.81	2794
ACCURACY			.93	13561
MACRO AVG	.94	.85	.88	13561
WEIGHTED AVG	.93	.93	.93	13561

MODEL ANALYSIS

- **94% ACCURACY**-only 6% of all predictions of both classes are incorrect

- **Low Risk: 0.93 Precision**-is not correct only 7% of the time when low risk is predicted

  1.00 Recall-correctly identifies 100% of all instances of low risk

- **High Risk: 1.00 Precision**-is 100% correct when high risk is predicted

  0.70 Recall-misses 30% of the instances of high risk

- **Recommendation:** The physician considers **prevalence** and **presenting clinical data**

WORKFLOW: FLASK

- Set up Flask
- Load the Random Forest model with Joblib **load** method
- Define a **function** to **preprocess** the input data from the **formWizard**
- Define the **home route** to render the **index.html** to receive input data
- Define the **prediction route** with **POST** method to submit data to be processed
- Ensure that **preprocessing** creates an **array** and **processing** creates a **dictionary (request.form.to_dict)**
- Random Forest model's **predict** method makes the prediction
- **get_random_code** function generates a code for a prediction of 1 (High Risk)
- **result.html** is rendered with the appropriate message

WORKFLOW: HTML/JavaScript

index.html

- Set up **head** with CSS Link
- Set up **navbar** with **container** for buttons to navigate to external sites
- **Div** for the **formWizard** being careful with setting up the **form steps** and **button** functionality and also turning of the **auto-complete dropdown** to avoid crowding and input errors
- Script tag for **jQuery** for document traversal/functionality/event handling
- JavaScript code to handle the **form steps**

WORKFLOW: HTML/JavaScript

result.html

- Set up **head** with CSS link
- Set up **navbar** with **container** for buttons to navigate to external sites
- **Div** for the prediction which includes **if...else…endif** code block, and the statements, and random code if needed, which are to be printed when a condition is met.

WORKFLOW: CSS

- **index_style.css/result_style.css:** scripts provide styling for the **index.html** and **result.html**
- **navbar:** with title and functioning buttons
- **body:** background picture with **no repeat**
- **formWizard:** size, position, padding, linear gradient color, shadow

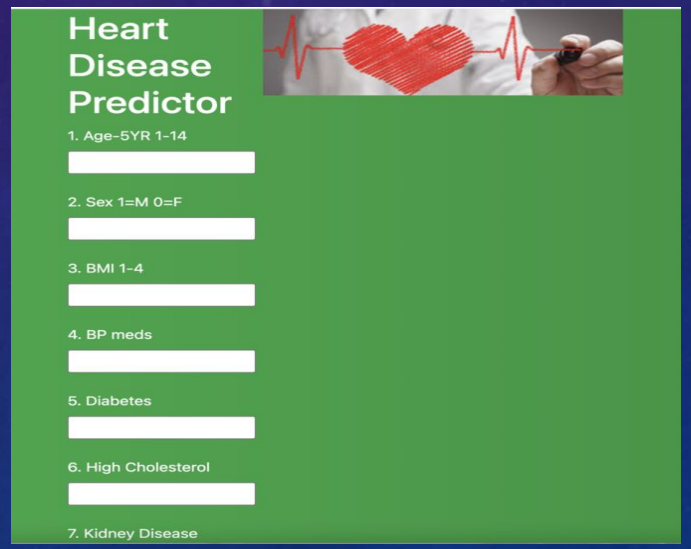
KEY CHALLENGES

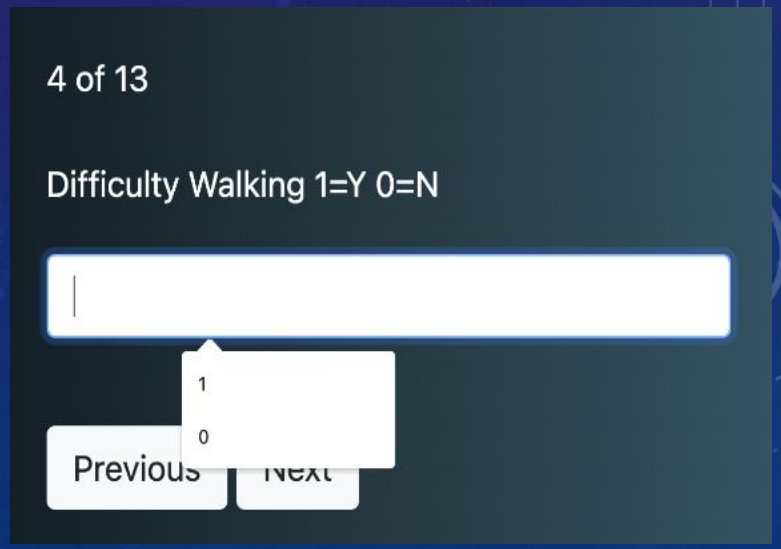
- **DATA ANALYSIS**
  Large Data Set (440,000 rows by 330 columns
  Domain Knowledge - BRFSS Codebook/clinician feedback
  Locate and remove nulls/missing values/blanks/extraneous data

- **DATA SCIENCE**
  Encoding: Binary structure of 1 & 2/Categorical coding (0 to 14) for BMI and age/Extraneous values (7 and 9)
  Logistic Regression Model not converging
  Weakness in model especially High Risk recall score persists at 70%

- **WEB DEVELOPMENT**
  Flask (return.html) failure
  Navigation Bar
  formWizard

- **USER EXPERIENCE**
  Modern
  Interactive
  One-page vs carousel
  Turn off auto complete
  Links





REFERENCES

1. Kaggle Dataset: Behavioral Risk Factor Surveillance System (2015)

   https://www.kaggle.com/datasets/cdc/behavioral-risk-factor-surveillance-system?select=2015.csv

1. Behavioral Risk Factor Surveillance System 2015 Codebook Report

   https://www.cdc.gov/brfss/annual_data/2015/pdf/codebook15_llcp.pdf

LIVE DEMONSTRATION OF THE APPLICATION!

- **Terryann:** *Project Structure*
- **Micha**el: *formWizard*
- Osman: *OnClickSubmit*

RECOMMENDATIONS FOR IMPROVEMENT

- Leverage feature engineering
- Optimize the model (SVM, Keras Tuner, TensorFlow)
- Train the model on a larger data set
- Develop application for mobile devices
- Develop application for patient use

QUESTIONS/COMMENTS