Plan B (Independent Study) STA 698 Project End - End Explanation:

The project started with the goal of predicting the 10-year coronary heart disease (CHD) risk using patient health data. I used the Framingham Heart Study dataset as the foundation for model development.  
  
First, I wrote an R script to handle data cleaning. In this script, I loaded the dataset using readxl, and carefully checked for missing values. For numeric columns like age, totChol, sysBP, diaBP, BMI, and glucose, I imputed missing values using their respective column means. For categorical variables like currentSmoker and diabetes, I applied mode imputation. After cleaning, I converted important variables (currentSmoker, diabetes, and Sex) into factor types, to prepare the data for modeling.  
  
Following data preparation, I trained four machine learning models in R:  
- Logistic Regression using glm  
- Ridge Regression using glmnet  
- Random Forest using randomForest  
- XGBoost using xgboost  
  
Each model was trained using the cleaned dataset to predict the target variable TenYearCHD. After training, I evaluated them using ROC-AUC metrics. The models performed very well, with Random Forest and XGBoost achieving the highest AUC scores close to 1.0. I saved all the trained models (logistic\_model.rds, ridge\_model.rds, rf\_model.rds, xgb\_model.rds) to disk.  
  
To serve the models for real-time predictions, I created an R Plumber API (api.R). The /predict endpoint was defined to accept user inputs like age, cholesterol, blood pressure, BMI, smoking status, diabetes status, and gender. Inside the API, the input was preprocessed to match the model's expectations, and predictions from all four models were returned in the response.  
  
I then created an R startup script (entrypoint.R) that loads the models and launches the Plumber API when the container starts. It prints loading messages for better debugging visibility.  
  
On the frontend side, I built a Python Flask application (app.py) that collects user input from an HTML form, sends it to the R API, receives predictions, and displays them nicely. I also integrated Chart.js to create a bar graph visualizing the predicted risk percentages from each model.  
  
To package everything together, I wrote a Dockerfile which:  
- Installed R, Python, and necessary system dependencies.  
- Installed R libraries (plumber, randomForest, xgboost, etc.).  
- Installed Python libraries from requirements.txt (like Flask and requests).  
- Copied all project files into the container.  
- Used a custom entrypoint.sh script to launch both the R server and the Flask server inside the container at runtime.  
  
After successfully building the Docker image locally using docker build, I tested the container locally by running it with docker run. Swagger UI was available at localhost:8080/\_\_docs\_\_, and the Flask app ran on port 5000. The communication between Flask and R API was successful, and predictions were displayed properly.  
  
Next, I prepared for deployment. I pushed the Docker image to Google Artifact Registry after configuring Docker authentication using gcloud auth configure-docker. The image was tagged and uploaded correctly.  
  
Then I deployed the container using Google Cloud Run by running gcloud run deploy. I made sure to expose the service publicly without authentication. Once deployed, I received a public URL, where the app became available globally.  
  
Throughout the project, I faced and solved several issues:  
- Errors like "object 'logistic\_model' not found" were solved by proper loading inside entrypoint.R.  
- Handling input data type mismatches (string vs numeric) when sending data between Flask and R.  
- Ensuring the R server starts properly inside the Docker container, before Flask starts making requests.  
- Properly managing port configurations to avoid Cloud Run container failures (ensuring Flask binds to port 8080).  
  
Finally, the app was running live, hosted on Google Cloud Run. It allows any user to input their health parameters, and instantly see predicted 10-year CHD risk scores from four models, with a visual chart highlighting the highest risk model.