Sumptuous Data Sciences

Machine Learning Model with RWD Harmonized by OMOP

AGENDA

__Intro:__ Synopsis

__Data Preparation:__ Handle Missing Data, Outliers, and Inconsistencies

__Machine Learning Model:__ Logisitic Regression & Random Forest

__Conclusion:__ R Markdown

About Me

- **Name:** Jeet Patel
- **Major:** Statistics & Economics at University of Illinois at Chicago
- **Career Goal:** Aspiring Data Analyst
- **Achievement:** Winner of the UIC SparkHacks Hackathon

OMOP
Standardizes healthcare data for consistent analysis across studies

SDTM
Standard format for organizing clinical trial data for regulatory submission

ERD
Diagram showing the relationships between data entities in a database

CDASH

Standardizes how clinical trial data is collected for consistency and analysis.

R

Programming language for statistics, data analysis, and visualization

Key Learning

Synopsis

- **Project Title:** Developing a Machine Learning Model for Risk Prediction Using Real-World Healthcare Data
- **Goal:** Build predictive models to assess clinical risk using healthcare datasets
- **Objectives:**
    - Prepare and clean the dataset (missing data, outliers, and feature engineering)
    - Perform exploratory data analysis (EDA)
    - Develop and compare supervised ML models (logistic regression and random forest)
- **Tools:** R with reproducible documentation RMarkdown
    - Libraries included are "tidymodels" & "tidvyverse"

Background

- Healthcare generates massive amounts of data
- Predicting patient risk is critical for **better outcomes & resource planning**
- Challenge: data is **messy, complex, and high-dimensional**
- Goal: apply ML to improve clinical decision-making

```r
library(haven)
library(tidyverse)
library(tidymodels)
library(vip)
```

PACKAGES

LIBRARY	PURPOSE	FUNCTIONS
haven	read SAS data	read_sas()
tidy verse	data maniuplation and visualization	left_join(), select(), filter()
tidy models	machine learning and workflows	recipe(), logistic_reg(), rand_forest(), fit(), predict(), metrics()
vip	visualize variable importance for random forest	vip()

Phase 1: Data Preparation

- Clinical data often contains **missing values, duplicates, and inconsistencies**
- Before modeling, data must be:
  - **Cleaned** (handle missing/outliers)
  - **Standardized** (consistent variable types)
  - **Labeled** (create outcome/target variable)
- Good preparation ensures **reliable ML results**

Data Preparation

- **Select & rename columns** –
  AGE, SEX, RACE, DCDECOD
  - **Handle missing values**
    - AGE → replace NA with median
    - SEX & RACE → replace NA with "Unknown"
  - **Cap AGE outliers** at 1st & 99th percentile
  - **Remove rows** where DCDECOD is missing
  - **Create target variable** → `completed_flag` (Yes/No)

```r
# 1) Select relevant columns and rename
data <- data[, c("USUBJID", "AGE.y", "SEX.y", "RACE.y", "DCDECOD")]
names(data)[names(data) == "AGE.y"] <- "AGE"
names(data)[names(data) == "SEX.y"] <- "SEX"
names(data)[names(data) == "RACE.y"] <- "RACE"

# 2) Handle missing AGE values by replacing NA with median
median_age <- median(data$AGE, na.rm = TRUE)
data$AGE[is.na(data$AGE)] <- median_age

# 3) Handle missing SEX and RACE by replacing NA with "Unknown"
data$SEX <- fct_na_value_to_level(data$SEX, "Unknown")
data$RACE <- fct_na_value_to_level(data$RACE, "Unknown")

# 4) Cap AGE outliers at 1st and 99th percentile
age_lower <- quantile(data$AGE, 0.01)
age_upper <- quantile(data$AGE, 0.99)
data$AGE[data$AGE < age_lower] <- age_lower
data$AGE[data$AGE > age_upper] <- age_upper

# 5) Remove rows where DCDECOD is missing
data <- data[!is.na(data$DCDECOD), ]

# 6) Create binary target variable
data$completed_flag <- factor(ifelse(data$DCDECOD == "COMPLETED", "Y
```

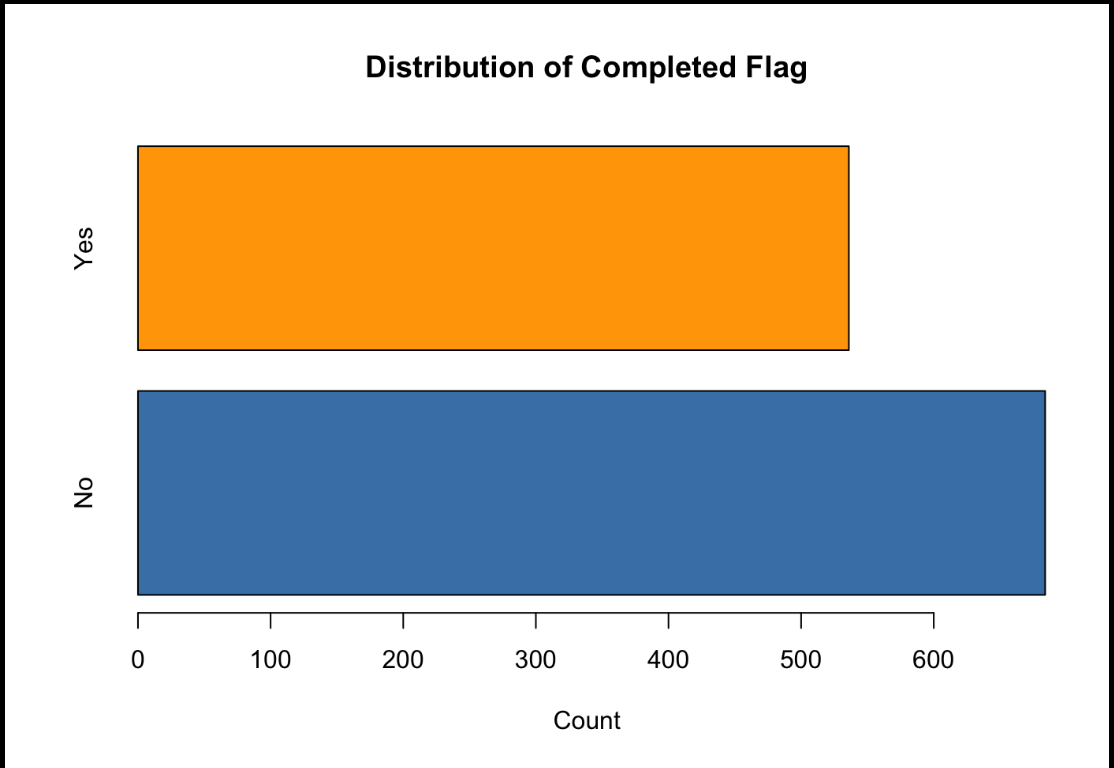
Exploratory Data Analysis

- **Check structure & summary**
- **Class balance** – distribution of `completed_flag`
    - Count & proportion of Yes / No
- **Visualize distribution** – horizontal bar chart

```
str(data)
glimpse(data)
summary(data)
table(data$completed_flag)
prop.table(table(data$completed_flag))

# Count the completed_flag categories
completed_counts <- table(data$completed_flag)

# Horizontal bar chart
barplot(completed_counts,
        horiz = TRUE,
        col = c("steelblue", "orange"),
        main = "Distribution of Completed Flag",
        xlab = "Count")
```

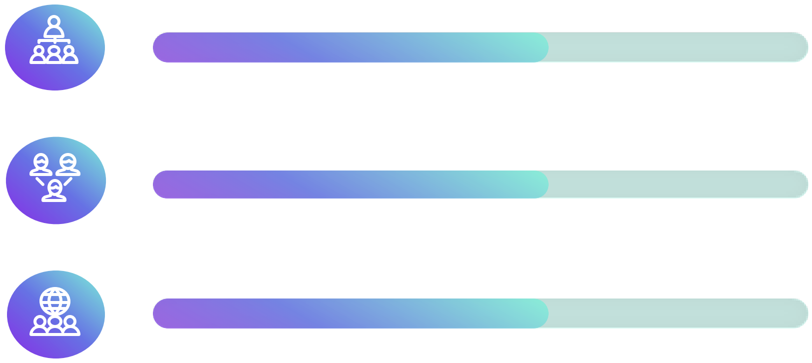
Bar Chart

- Bar chart shows patients who completed vs. did not complete the study
- 56% did not complete, 44% complete
- Target variable is fairly balanced → unlikely to affect ML model performance



Phase 2: Machine Learning

- Machine Learning helps predict **risk/outcomes** from data
- Process:
  - Split data into **train/test sets**
  - **Train models** (e.g., logistic regression, random forest)
  - **Evaluate performance** with metrics (Accuracy, AUC, Precision/Recall)
  - Compare models → pick best balance of **accuracy + interpretability**



Train/Test Split

- **Purpose:** Separate data into training and testing sets for model evaluation.
- **Split:** 80% training, 20% testing
- **Stratification:** Maintain class balance of `completed_flag`

```r
# 9) Split the data into training and testing sets
set.seed(123)  # ensures reproducibility

split <- initial_split(data, prop = 0.8, strata = completed_flag)

train <- training(split)   # training data (80%)
test  <- testing(split)    # testing data (20%)

nrow(train)   # number of rows in training set
nrow(test)    # number of rows in testing set

prop.table(table(train$completed_flag))
```

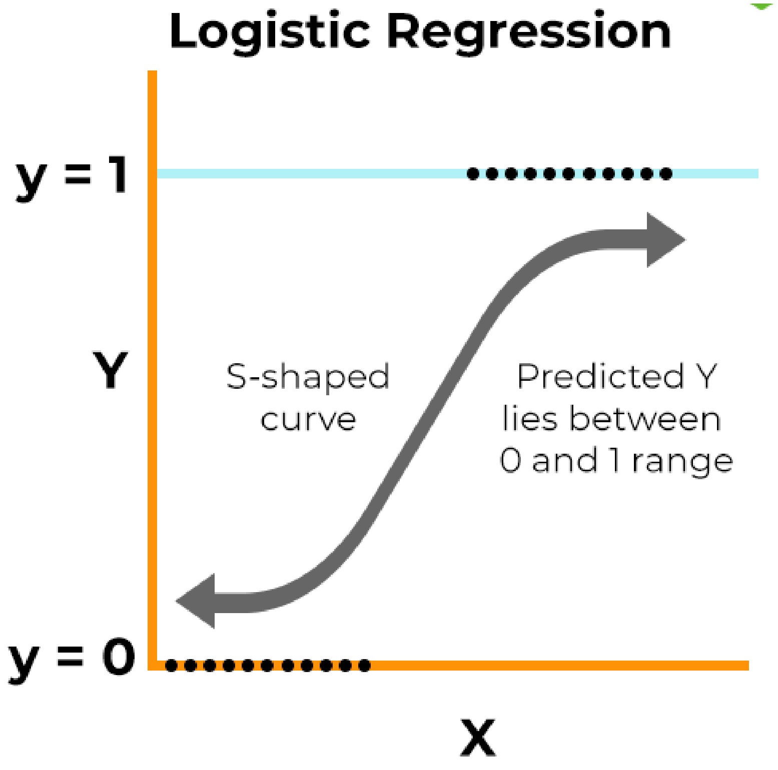
```
> prop.table(table(train$completed_flag))

       No       Yes
0.5610256 0.4389744
```

Preprocessing

- **Create a recipe** for data transformations (`recipe()`)
- **Dummy variables** for categorical predictors (SEX, RACE)
→ `step_dummy()`
- **Normalize numeric predictor** (AGE)
→ `step_normalize()`

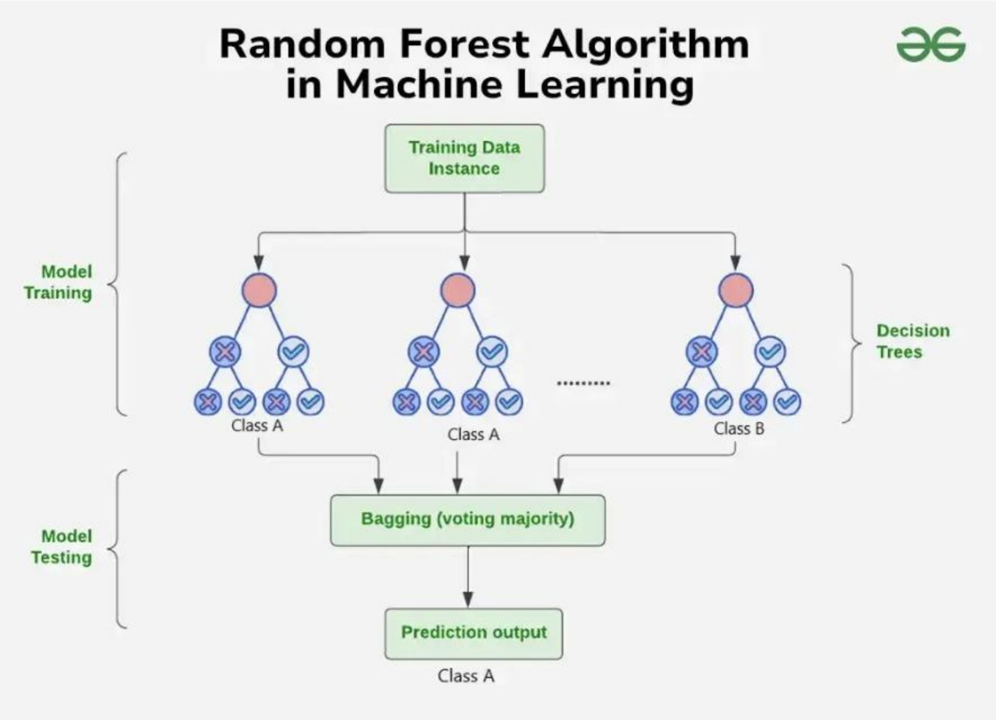
```
log_model <- logistic_reg(mode = "classification")
log_model <- set_engine(log_model, "glm")

log_wf <- workflow()
log_wf <- add_recipe(log_wf, rec)
log_wf <- add_model(log_wf, log_model)

log_fit <- fit(log_wf, data = train)
```

Machine Learning Models



Logistic Regression

- Predicts two outcomes (*Yes/No*) using probabilities from input factors.
- Simple and interpretable, but works best with linear relationships.
- **Limitation**: May underfit when data patterns are complex.



Random Forest

- Predicts outcomes by combining many decision trees, each voting on the result.
- More accurate and handles complex, non-linear relationships.
- **Limitation**: Less interpretable and can be slower to train.

Logistic Regression

- **Model:** `logistic_reg()`
- **Engine:** `glm`
- **Workflow:**
  - Combine recipe + model → `workflow()`, `add_recipe()`, `add_model()`
- Fit model on training data → `fit()`

```r
log_model <- logistic_reg(mode = "classification")
log_model <- set_engine(log_model, "glm")

log_wf <- workflow()
log_wf <- add_recipe(log_wf, rec)
log_wf <- add_model(log_wf, log_model)

log_fit <- fit(log_wf, data = train)
```

Random Forest

- **Model:** `rand_forest()`
- **Engine:** `ranger`
- **Workflow:**
- Combine recipe + model → `workflow()`, `add_recipe()`, `add_model()`
- Fit model on training data → `fit()`

```r
rf_model <- rand_forest(mode = "classification", trees = 500)
rf_model <- set_engine(rf_model, "ranger", importance = "impurity")

rf_wf <- workflow()
rf_wf <- add_recipe(rf_wf, rec)
rf_wf <- add_model(rf_wf, rf_model)

rf_fit <- fit(rf_wf, data = train)
```

Logistic Regression & Random Forest

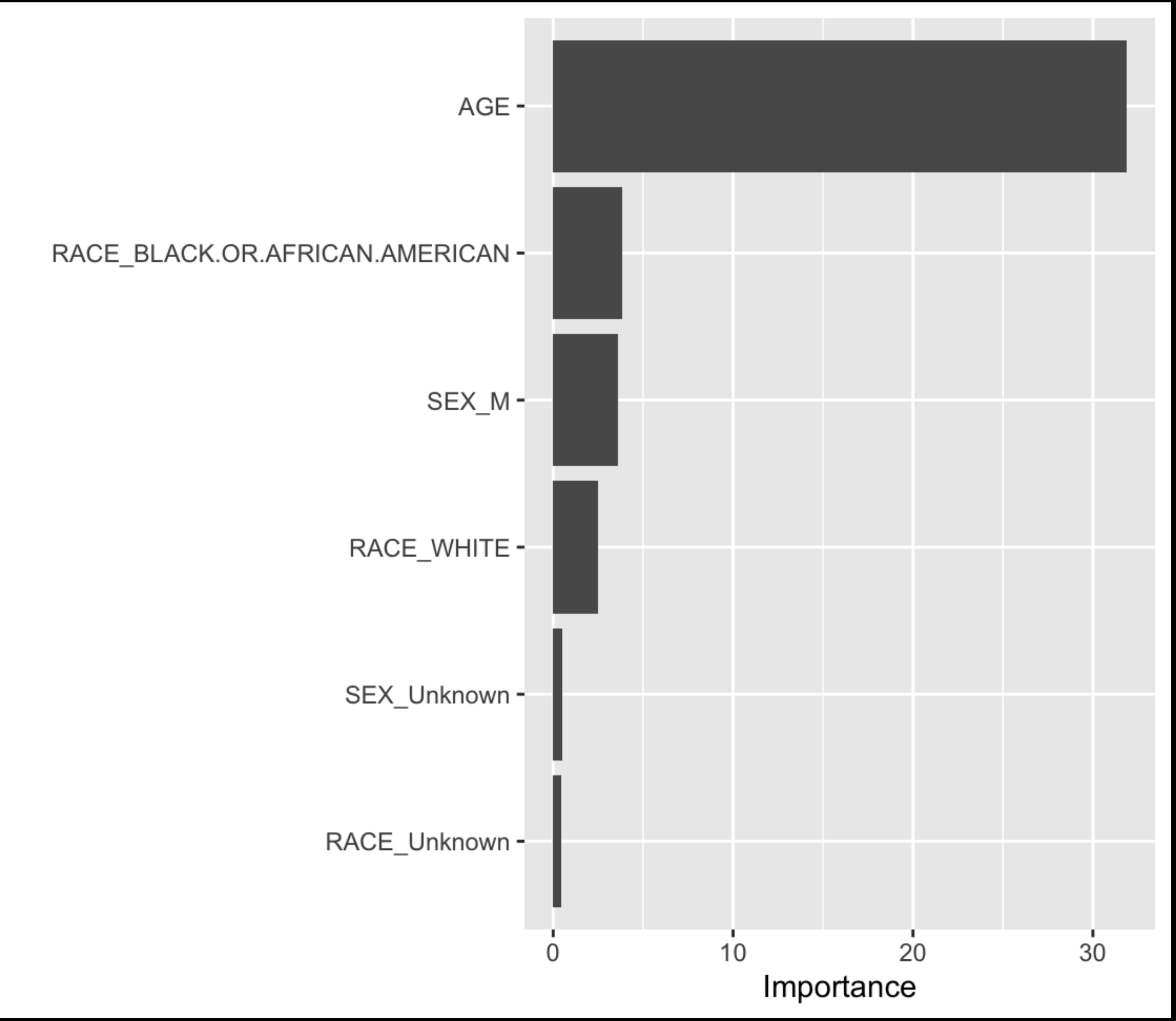
- **Predictions:** `predict()` on test set
- **Metrics:** `metrics()` → accuracy, sensitivity, specificity
- **Confusion matrix:** `conf_mat()` → shows true positives, true negatives, false positives, false negatives

```
log_preds <- predict(log_fit, test)
log_preds <- bind_cols(log_preds, select(test, completed_flag))

log_metrics <- metrics(log_preds, truth = completed_flag, estimate = .pred_class
log_metrics

log_cm <- conf_mat(log_preds, truth = completed_flag, estimate = .pred_class)
log_cm
```

```
rf_preds <- predict(rf_fit, test)
rf_preds <- bind_cols(rf_preds, select(test, completed_flag))

rf_metrics <- metrics(rf_preds, truth = completed_flag, estimate = .pred_class)
rf_metrics

rf_cm <- conf_mat(rf_preds, truth = completed_flag, estimate = .pred_class)
rf_cm
```

Variable Importance Plot

- Shows which variables contributed most to predictions in the Random Forest model.
- **AGE** is the most important predictor of study completion.
- Other factors (e.g., sex, race) had smaller influence.



Conclusion

Figures

```
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.514
2 kap      binary        -0.0479
> log_cm
          Truth
Prediction  No Yes
       No  108  90
       Yes  29  18
```

```
  .metric  .estimator .estimate
  <chr>    <chr>          <dbl>
1 accuracy binary         0.645
2 kap      binary         0.226
> rf_cm
          Truth
Prediction  No Yes
       No  129  79
       Yes   8  29
```

Statistics

Logistic Regression's Accuracy

51.4%

Random Forest Accuracy

64.5%

Random Forest's Kappa

22.6%

Age's Importance

32%

Result

Random Forest outperformed Logistic Regression

Key Takeaway

Machine learning helps identify patient risk factors and supports clinical trial planning

Future Work

Test on larger datasets and include additional predictors for improved prediction

Recap

Data Preparation

Merge data, replace missing data, and cap outliers

Exploratory Data Analysis

Inspect class distribution & bar graph

Train/Test Split

80% training & 20% testing

Feature Engineering

Make a recipe

Recap

Logistic Regression/ Random Forest

Select a mode & engine

Evaluation

Metrics determine accuracy & confusion metrics notice false positives/negatives

Interpretation

Identify which features influences the predicter

Future

More machine learning models

Sumptuous Data Sciences

Thank You!

Any Questions?