Estimating Demand with Machine Learning - San Francisco BART

Shafqat Shafiq

2024-12-25

1. Introduction

In the early 1970s, San Francisco was completing a huge new infrastructure project called the Bay Area Rapid Transit (BART) system. The project initially cost about $1.6 billion and included tunneling under the San Francisco Bay. Policy makers were obviously interested in determining how many people would use the new system once it was built. But how do you *predict the demand for a new product that does not exist?*

One possible solution is to ask people through a survey, which is exactly what was done: a survey was conducted of people who were likely to use the new transport system. It asked people detailed questions about their current mode of transport, and asked them whether they would use the new system.

The concern is that it is hard for people to predict how they would use something that does not exist. Berkeley Econometrician Dan McFadden suggested an alternative. Instead of asking people to predict what they would do, McFadden suggested using information on what people *actually do*, and then use economic theory to predict what they *would do.*

2. Revealed Preference

Theory: *If there are two choices, A and B, ad we observe a person choose A, then their utility from A is greater than their utility from B.*

Modeling Demand

Consider a data set where a large number of individuals are observed purchasing either product A or B, at various prices for the two products. Each individual will have some unobserved characteristic, u, which is their *utility.*

We make the following assumptions:

1. The value for an individual for purchasing one of the two products is $u_{Ai} - p_A$ which is the *utility derived* by the i-th individual from purchasing product A minus the *price* of the product.
2. If we observe person i purchase A at price p_A then we know that the following inequality holds:

$$u_{Ai} - p_A > u_{Bi} - p_B$$

$$u_{Ai} - u_{Bi} > p_A - p_B$$

Person i purchases good A if and only if her relative utility for A is greater than the relative price of A. In addition, we often observe the data at the market level rather than the individual level. That is, we see the fraction of individuals that purchase product A. We usually denote this fraction using s; it's the share of individuals that purchase A.

Simulating Demand

Consider the following distribution where the unobserved term (utility) u is drawn from a normal distribution with a mean of `1` and variance `9`: $u \sim \mathcal{N}(1, 9)$. Let's assume that we have data on `1000` people and each of them is described by this u character.

```r
set.seed(1234)

# Number of individuals is 1000
N <- 1000

# Utility for each of these 1000 individuals is drawn from N(1,9) distribution
u <- sort(rnorm(N, mean = 1, sd = 3))
```

We'd like to uncover this distribution of u from observed behavior of the individuals using the **Revealed Preference** theory. That is, we'd like to identify the proportion of individuals who for a given price p will buy a certain product. In probability terms we can postulate it as $P(u > p) = 1 - P(u \leq p)$.

```r
# Suppose the price is p = $2
p <- 2

# The estimated proportion of individuals who valued the product greater
# than p = 2 or the P(u > p = 2) i.e. probability u is greater than 2:
1 - pnorm(p, 1, 3)
```

```
## [1] 0.3694413
```

```r
# Estimate the probability
mean(u - p > 0)
```

```
## [1] 0.35
```

If $p = 2$, then the share of people who purchase `A` is `39%` which is approximately equal to the probability that u is greater than `$2`. Combining the revealed preference assumption with the observed data allows us to uncover the fraction of individuals whose value for the product is greater than `$2`.

Revealing Demand

Let's call this derived value $u = u_{Ai} - p_A$ as the *unobserved characteristic*. If we are able to observe *a large number of prices* then we can use revealed preference to estimate the whole distribution of this *unobserved characteristic*.

We do this by calculating *the share of individuals purchasing product A* at each price. If we observe enough prices, then we can use the observed shares at each price to plot out the demand curve.

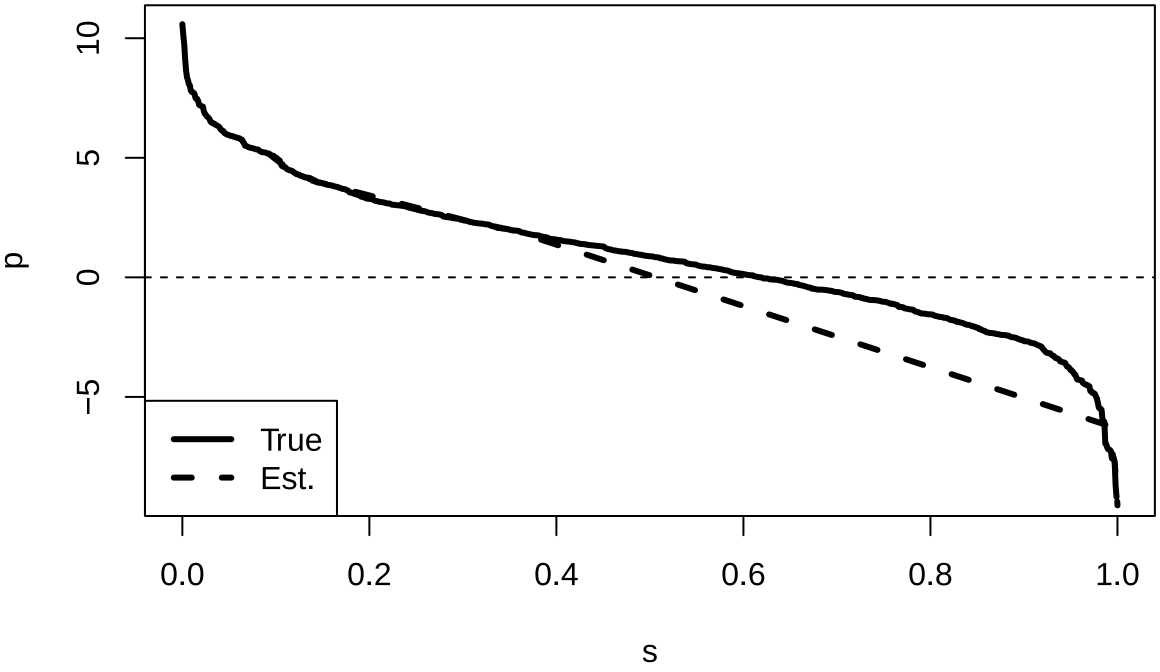
```r
# Generate a random set of 9 prices from U[10, 20]
p <- runif(9, min = -10, max = 10)

# Create a column vector with 9 rows that will hold the estimated
# probability/proportion of individuals who'll buy the product at that price
s <- matrix(NA, length(p), 1)

# Loop over the vector to store the estimated probability
for (i in 1:length(p)) {

  s[i, 1] <- mean(u - p[i] > 0)

}
```

```
print(s)
```

```
##        [,1]
##  [1,] 0.997
##  [2,] 0.987
##  [3,] 0.304
##  [4,] 0.352
##  [5,] 0.999
##  [6,] 0.145
##  [7,] 0.106
##  [8,] 1.000
##  [9,] 0.092
```

Let's now plot the Emperical CDF *(ecdf)* of s against p. `1-ecdf(u)(u)` represents the estimated probabilities of u; `p[order(s)]` arranges the values of vector p in the order of vector s.

```
plot(1-ecdf(u)(u), u, type = "l", lwd = 3, lty = 1, col = 1,
     xlab = "s", ylab = "p", xlim = c(0,1))
lines(sort(s), p[order(s)], type = "l", lwd = 3, lty = 2)
abline(h = 0, lty = 2)
legend("bottomleft", c("True", "Est."), lwd = 3, lty = c(1:2))
```



Simple Discrete Choice Model

Consider the following discrete model. There is some **latent** (hidden) value of the outcome y_i^* where if this value is sufficiently large, we observe $y_i = 1$.

Let's assume that this y_i^* is linearly related to the matrix of variables X_i and we have that

$$y_i^* = \beta \cdot X_i + \epsilon_i$$

Furthermore, the relation between y_i^* and y_i is given as follows:

$$y_i = \begin{cases} 1 & \text{if } y_i^* \geq 0, \\ 0 & \text{if } y_i^* < 0 \end{cases}$$

We can think of y_i^* as the *utility* and y_i representing the *observed demand.* If utility is above a certain threshold, we see an individual demanding/buying that product.

Modeling Discrete Choice

We now have that

$$y_i^* = \beta \cdot X_i + \epsilon_i$$

$$y_i = \begin{cases} 1 & \text{if } y_i^* \geq 0, \\ 0 & \text{if } y_i^* < 0 \end{cases}$$

where X_i is the vector of observed characteristics of the individual i, β is a vector that maps from the observed characteristics to the **latent outcome** and y_i^* is the **unobserved utility/value/latent value** derived by $i - th$ individual from making a choice. We see that $y_i = 1$ if the latent value is greater than `0`, and $y_i = 0$ if it's less than `0`.

The probability of observing one of the outcomes $P(y_i = 1)$ is given as follows:

$$P(y_i = 1 \mid X_i) = P(y_i^* > 0) = P(X_i^\top \beta + v_i > 0) = P(v_i > -X_i^\top \beta) = 1 - F(-X_i^\top \beta)$$

Here, $F()$ represents the probability distribution function of the error variable v_i which is also considered to be part of the *unobserved characteristic.* More specifically, v_i is random component, capturing the unobserved or stochastic variation in utility. This could represent individual idiosyncrasies, measurement errors, or other factors not included in X_i.

If we know β, we can determine the distribution of the unobserved term with variation in the Xs. That is, we can determine F.

β is generally the policy parameter we want to estimate. The standard solution is to *assume we know the true distribution* of the unobserved term, $F()$. In particular, the standard assumption is to assume that $F()$ is **standard normal**. Therefore, we can write out the probability of observing $y_i = 1|X_i$ as:

$$P(y_i = 1 \mid X_i) = 1 - \Phi(-X_i^\top \beta)$$

where $\Phi()$ is the *Standard Normal Distribution CDF.*

3. McFadden's Random Utility Model

In order to estimate the impact of BART rail system, McFadden needed a model that captured the current choices and predicted demand for a product that did not exist yet. This can be done either using a *Probit* or a *Logit* estimation model.

Modeling Demand for BART

Consider the following utility model that is linearly related with observed characteristics:

$$U_{ij} = {X^t}_{ij} \cdot \beta + v_i j$$

Person `i's` utility is a function of observable characteristics of both person `i` and the choice `j`, represented by the matrix X_i. These are things such as the person's income and cost of commuting by car. We assume that β does not vary with the product; i.e. individuals weight the characteristics associated with different products the same, irrespective of what they choose.

We can use **Revealed Preference** and **Observed Choices** to make inferences about person `i's` preferences. If observe person " face the choice between two products A and B, and we see them choose A, then we learn that $U_{iA} > U_{iB}$.

$$U_{iA} > U_{iB} = X^t_{iA} \cdot \beta + v_{iA} > X^t_{iB} \cdot \beta + v_{iB} = v_{iA} - v_{iB} > -(X^t_{iA} - X^t_{iB}) \cdot \beta$$

If there's enough variation in the *observed characteristics* of the choices $(X^t_{iA} - X^t_{iB}$, we can then potentially estimate both the unobserved, stochastic components $v_{iA} - v_{iB}$ and β, which are the policy parameters. The X_i vector can contain characteristics such as *price, income, cost of living, density* et al.

Probit and Logit Estimators

Let us simulate a situation where 5000 individuals are choosing between 2 products. Each individual has two observable characteristics, denoted by X_{iA} and X_{iB}. Note that these characteristics (may) vary across the individuals, but the individual preferences do not vary.

Probit

If we assume that $v_{iA} - v_{iB}$ is distributed **standard normal**, we can then use the *probit* model.

```r
set.seed(1234)
N <- 5000

# Create two product characteristic matrices
X_A <- cbind(1, matrix(runif(2*N), nrow = N))
X_B <- cbind(1, matrix(runif(2*N), nrow = N))

# Assume that the Beta coefficients have been estimated; create a Beta vector
beta <- c(1, -2, 3)
```

$$\begin{aligned} P(y_i = 1 | X_{iA}, X_{iB}) &= P(v_{iA} - v_{iB} > -(X_{iA} - X_{iB})^t \cdot \beta) \\ &= 1 - P(v_{iA} - v_{iB} \leq -(X_{iA} - X_{iB})^t \cdot \beta) \\ &= 1 - \Phi(-(X_{iA} - X_{iB})^t \cdot \beta) \\ &= 1 - (1 - \Phi((X_{iA} - X_{iB})^t \cdot \beta)) \\ &= \Phi((X_{iA} - X_{iB})^t \cdot \beta) \end{aligned}$$

Let's estimate probit in R:

```r
v <- rnorm(N)
# Create the unobserved utility function y as a linear combination of X_A and X_B
y <- X_A %*% beta - X_B %*% beta + v > 0 # Unobserved utility
# Fit a probit model using the glm() function
probit <- glm(y ~ I(X_A - X_B), family = binomial(link = "probit"))
```

Logit

On the other hand, if we assume that $v_{iA} - v_{iB}$ is log Weibull/Gumbel distributed, then we use the following *logit link* to associate $P(y_i = 1|X_{iA}, X_{iB})$ with the observed characteristic vectors X_{iA}, X_{iB}.

$$P(y_i = 1|X_{iA}, X_{iB}) = \frac{\exp((X_{iA} - X_{iB})^t \cdot \beta)}{1 + \exp((X_{iA} - X_{iB})^t \cdot \beta)}$$

Let's estimate Logit in R:

```r
v_A <- log(rweibull(N, shape = 1)); v_B <- log(rweibull(N, shape = 1))
y <- (X_A%*%beta - X_B%*%beta) + (v_A - v_B) > 0
logit <- glm(y ~ I(X_A - X_B), family = binomial(link = "logit"))
```

Let us combine both `Logit` and `Probit` models and compare them:

```r
# Tidy up the results
probit_results <- tidy(probit)
logit_results <- tidy(logit)

# Add model type to the results
probit_results$model <- "Probit"
logit_results$model <- "Logit"

# Combine the results
combined_results <- rbind(probit_results, logit_results)

# Reorder columns to have model information first
combined_results <- combined_results[, c("model","term",
                                         "estimate", "std.error", "statistic", "p.value")]

# View the combined results
print(combined_results)
```

```
## # A tibble: 8 x 6
##   model  term          estimate std.error statistic    p.value
##   <chr>  <chr>            <dbl>     <dbl>     <dbl>      <dbl>
## 1 Probit (Intercept)    -0.0138    0.0226    -0.611  5.41e-  1
## 2 Probit I(X_A - X_B)1  NA        NA          NA     NA
## 3 Probit I(X_A - X_B)2  -2.00      0.0678   -29.5    2.61e-191
## 4 Probit I(X_A - X_B)3   3.01      0.0812    37.0    5.29e-300
## 5 Logit  (Intercept)     0.0525    0.0332     1.58   1.14e-  1
## 6 Logit  I(X_A - X_B)1  NA        NA          NA     NA
## 7 Logit  I(X_A - X_B)2  -1.91      0.0897   -21.2    4.40e-100
## 8 Logit  I(X_A - X_B)3   2.94      0.101     29.0    3.09e-185
```

As one can see, the estimates (except for the intercept) for both `Probit` and `Logit` are not that far off from the true β values.

4. Demand for Rail in San Francisco

In McFadden's analysis, policy makers were interested in how many people would use the new **BART** rail system. Would there be enough user to make such a large infrastructure project worthwhile? This is a question that remains relevant for major cities across the world. Many large cities in the U.S. such as New York, Boston, and Chicago have major rail infrastructure while smaller cities do not. For these smaller cities, the question is whether building a robust rail system lead to an increase in public transportation usage.

To answer this question, we can use data from the **National Household Travel Survey**, particularly the publicly available **Household Component** of said survey. The data provides information on what mode of transport a given household uses most days; *cars, bus, or trains.* It also contains demographic information such as *home ownership, income, rural/urban residence, density of rail networks in a given location.*

We use the *logit* model to identify the factors influencing demand for transportation in **rail** and **non-rail** cities, focusing on estimating odds ratios. We begin by fitting logistic regression and random forest models for the **rail** cities, evaluating their accuracy to determine which model performs best. Once the best-performing model is identified, we apply it to predict the demand for rail services in **non-rail** cities.

National Household Travel Survey

In this analysis, we utilize the National Household Travel Survey (NHTS), focusing specifically on the 2017 dataset. The NHTS is a comprehensive survey conducted in the United States that collects detailed information on the travel behavior of households, including data on the frequency, duration, and purpose of trips made by various modes of transportation.

The 2017 dataset includes valuable demographic information, such as household size, income, and urbanization level, which are crucial for understanding travel patterns. This dataset allows us to examine how different factors influence transportation choices, particularly the use of public transit options like trains.

For our analysis, we make necessary adjustments to the raw data to ensure it is suitable for our modeling efforts. These adjustments may include filtering for specific variables of interest, addressing any missing or inconsistent data, and transforming categorical variables into appropriate formats for analysis. By carefully processing the 2017 NHTS dataset, we aim to derive meaningful insights into travel behavior and the potential impacts of transportation infrastructure changes.

```r
df <- read.csv("hhpub.csv")
```

Perform some data cleaning and wrangling:

```r
# Create a new variable called "Choice" that corresponds to the transport mode
df <- df %>%
  mutate(
    choice = case_when(
      CAR == 1 ~ "car",
      BUS == 1 ~ "bus",
      TRAIN == 1 ~ "train",
      TRUE ~ NA_character_  # Default case for missing values
    )
  )

df$car1 <- df$choice=="car"
df$train1 <- df$choice=="train"

# Adjusting variables to account for missing data

# Home ownership
df$home <- ifelse(df$HOMEOWN == 1, 1, NA)
```

```
df$home <- ifelse(df$HOMEOWN > 1, 0, df$home)

# Household income
df$income <- ifelse(df$HHFAMINC > 0, df$HHFAMINC, NA)

# Population density; dividing by 1000 makes the results look nicer
df$density <- ifelse(df$HTPPOPDN == -9, NA, df$HTPPOPDN)/1000


# Create a binary indicator to indicate whether household is in urban/rural
df$urban1 <- df$URBAN == 1

# Limit to households that may commute and those that live in some type of city
# Create a new variable 'y' such that it filters according to the following
# conditions:
# WRKCOUNT > 0 filters rows where the household has at least one person working
# (df$MSACAT == 1 | df$MSACAT == 2) further filters rows where MSACAT == 1 or 2
# MSACAT is Metropolitan Statistical Area and represents city type.
y <- df[df$WRKCOUNT > 0 & (df$MSACAT == 1 | df$MSACAT == 2), ]

# Create a binary indicator for rail access/an MSA with rail
y$rail <- y$RAIL == 1

# Drop missing values from key columns in y
y <- y %>% drop_na(car1, train1, home, HHSIZE, income, urban1, density, MSACAT, rail)
```

A fundamental question that we'd like to ask is, how different are **rail** cities from **non-rail** cities? The plan is to use demand estimates for **cars, bueses and rail** in cities with rail networks, to predict demand for rail in other cities.

However, cities with and without rail may differ in a variety of ways, which may lead to different demand for rail between them.

```
vars <- c("car1", "train1", "home", "HHSIZE", "income", "urban1", "density")

summ_tab <- matrix(NA, length(vars), 2)

for (i in 1:length(vars)){

  summ_tab[i, 1] <- mean(y[y$rail==1, colnames(y) == vars[i]])
  summ_tab[i, 2] <- mean(y[y$rail == 0, colnames(y) == vars[i]])

}

row.names(summ_tab) <- vars
colnames(summ_tab) <- c("Rail", "No Rail")

print(summ_tab)
```

```
##               Rail     No Rail
## car1    0.91990709 0.987109148
## train1  0.04361666 0.001471966
## home    0.73012732 0.749230563
## HHSIZE  2.48786992 2.462420269
## income  7.51884033 7.054551943
## urban1  0.91027185 0.869753334
## density 7.55907605 4.661936304
```

We see that in cities with rail networks, about 4% of the population uses trains most days, while it is only 0.1% for cities without a dense rail network. The two types of cities also differ in terms of income, household size, home ownership, and population density, however those differences are not that stark.

Q: What would happen to the demand for rail in a non-rail city, were the city to build a rail-network? Would demand increase to 4%? or would the demand be different due to the characteristic differences between the cities?

Demand for Cars

We begin by looking at how the demand for cars varies between the two types of cities. We do this by fitting 2 different GLMs; one on cities with rail and another on cities without.

```
# With rail
y_r <- y[y$rail==1, ]
glm_r <- glm(car1 ~ home + HHSIZE + income + urban1 + density,
             data = y_r, family = binomial(link = "logit"))
summary(glm_r)
```

```
##
## Call:
## glm(formula = car1 ~ home + HHSIZE + income + urban1 + density,
##     family = binomial(link = "logit"), data = y_r)
##
## Coefficients:
##              Estimate Std. Error z value             Pr(>|z|)
## (Intercept)  3.638290   0.360744  10.086 < 0.0000000000000002 ***
## home         0.807866   0.089283   9.048 < 0.0000000000000002 ***
## HHSIZE       0.242916   0.036661   6.626      0.0000000000345 ***
## income       0.001157   0.016879   0.069                0.945
## urban1TRUE  -0.339608   0.345093  -0.984                0.325
## density     -0.146297   0.004252 -34.409 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6486.1  on 11623  degrees of freedom
## Residual deviance: 4265.2  on 11618  degrees of freedom
## AIC: 4277.2
##
## Number of Fisher Scoring iterations: 7
```

```
# Without rail
y_nr <- y %>% filter(rail == 0)
glm_nr <- glm(car1 ~ home + HHSIZE + income + urban1 + density,
             data = y_nr, family = binomial(link = "logit"))
summary(glm_nr)
```

```
##
## Call:
## glm(formula = car1 ~ home + HHSIZE + income + urban1 + density,
##     family = binomial(link = "logit"), data = y_nr)
##
```

```
## Coefficients:
##             Estimate Std. Error z value            Pr(>|z|)    
## (Intercept)  3.12449    0.44300   7.053     0.00000000000175 ***
## home         1.22215    0.14474   8.444 < 0.0000000000000002 ***
## HHSIZE       0.16817    0.05307   3.169              0.00153 ** 
## income       0.31283    0.02803  11.160 < 0.0000000000000002 ***
## urban1TRUE  -1.25184    0.42092  -2.974              0.00294 ** 
## density     -0.06787    0.01013  -6.702     0.00000000002062 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 3089.3  on 22418  degrees of freedom
## Residual deviance: 2576.9  on 22413  degrees of freedom
## AIC: 2588.9
## 
## Number of Fisher Scoring iterations: 9
```

In order to interpret the coefficients, we need to exponentiate them to get the *Odds-Ratios*.

```
or_r <- exp(coef(glm_r))
or_nr <- exp(coef(glm_nr))

odds_ratios <- cbind(or_r, or_nr)

print(odds_ratios)
```

```
##                   or_r      or_nr
## (Intercept) 38.0267635 22.7481955
## home         2.2431154  3.3944903
## HHSIZE       1.2749615  1.1831395
## income       1.0011579  1.3672919
## urban1TRUE   0.7120494  0.2859768
## density      0.8639009  0.9343825
```

Interpretation of Odds Ratios for Car Demand

The odds ratios (OR) for the demand for cars are presented for two models: cities with rail networks (`or_r`) and cities without rail networks (`or_nr`). Below is a summary of the key findings:

Variable	Odds Ratio (With Rail)	Odds Ratio (Without Rail)	Interpretation
Intercept	38.03	22.75	The baseline odds of choosing a car are higher in cities with rail networks.
Home	2.24	3.39	In cities with rail, home ownership increases car odds by 124%. In cities without rail, the increase is 239%.
HHSIZE	1.27	1.18	Larger households are more likely to choose cars, but this effect is slightly weaker in cities without rail.
Income	1.00	1.37	Income has a negligible impact on car demand in cities with rail but increases the odds by 37% in cities without rail.
Urban1TRUE	0.71	0.29	Living in urban areas reduces car odds by 29% with rail, and by 71% without rail.
Density	0.86	0.93	Higher population density decreases the likelihood of choosing a car by 14% with rail and by 7% without rail.

Key Insights

The presence of a rail network significantly influences car demand. In cities without rail, reliance on cars is more pronounced, as seen in higher odds ratios for variables like `home` and `income`. Urban living and density reduce car demand more strongly in cities without rail, suggesting better accessibility to alternative transport options.

Estimating Demand for Rail

We can set up the McFadden demand model for cars, buses and trains. The utility of car and train is relative to bus. We note that the value of train is assumed to be a function of density. The assumption is that *trains have fixed station locations* and in more dense cities, *these locations are likely to be more easily accessible to the average person.*

We undertake the following steps:

1. **Fit models for cities with train services**: We evaluate both Logistic Regression and Random Forest models.
2. **Apply the selected model to comparable cities without train services**.
3. **Calculate the mean of predicted train users**: This will provide us with the estimated proportion of individuals likely to use trains.

```
# Create a dataframe where rail == TRUE
y_rail <- y %>% filter(rail == TRUE)

# Filter columns relevant to model building
y_rail <- y_rail %>% dplyr::select(c("home","HHSIZE","income", "urban1","density", "train1"))

head(y_rail)
```

```
##   home HHSIZE income urban1 density train1
## 1    1      2      5   TRUE    30.0  FALSE
## 2    0      2      4   TRUE    30.0  FALSE
## 3    1      4      8  FALSE     0.3  FALSE
## 4    0      1      5   TRUE    17.0  FALSE
## 5    1      2      9   TRUE     0.3  FALSE
## 6    1      3     10  FALSE     0.3  FALSE
```

```
# Split the data into test and train proportions

set.seed(1234)

#
train_indices <- sample(nrow(y_rail), size = 0.7 * nrow(y_rail))  # 70% for training

# Split the data
train_data <- y_rail %>% slice(train_indices)  # Training set
test_data <- y_rail %>% slice(-train_indices) # Testing set

# Check the sizes of the splits
nrow(train_data) # Should be approximately 70% of the data
```

```
## [1] 8136
```

```
nrow(test_data)  # Should be approximately 30% of the data
```

```
## [1] 3488
```

Now that we have split the data into test (70%) and train (30%) components, we look to fit two different models and evaluate them. The models we'll look to implement are *Logistic Regression* and *Random Forest Model*, and our target variable will be `train1`, since we are interested in predicting whether an individual `i` is going to use the train, based on their individual characteristics. This set of characteristics include `home, HHSIZE, income, urban1, density`.

We begin by fitting a *Logistic Regression* model:

1. Logistic Regression

```
# Train the logistic model
train1_logistic <- glm(train1 ~ home + HHSIZE + income +
                         urban1 +
                         density,
                 data = train_data,
                 family = binomial(link = "logit"))

# Predict probabilities on the test dataset
predicted_probs <- predict(train1_logistic, newdata = test_data, type = "response")

# Convert probabilities to TRUE/FALSE values using a threshold (e.g., 0.5)
predicted_logistic_train1 <- predicted_probs >= 0.4
# This will give TRUE if the probability is >= 0.4

# Actual labels from the test dataset (assuming the actual variable is a binary factor)
test_data$predicted <- predicted_logistic_train1

# Evaluate the model using specificity and sensitivity and confusion matrix

# Create confusion matrix
confusion_matrix <- table(Actual = test_data$train1,
                          Predicted = test_data$predicted)

# Calculate sensitivity (True Positive Rate)
sensitivity <- confusion_matrix["TRUE", "TRUE"] /
  sum(confusion_matrix["TRUE", ])

# Calculate specificity (True Negative Rate)
specificity <- confusion_matrix["FALSE", "FALSE"] /
  sum(confusion_matrix["FALSE", ])

# Print results
print(confusion_matrix)
```

```
##        Predicted
## Actual  FALSE TRUE
##   FALSE  3286   45
##   TRUE    111   46
```

```
cat("Sensitivity:", sensitivity, "\n")
```

```
## Sensitivity: 0.2929936
```

```
cat("Specificity:", specificity, "\n")
```

```
## Specificity: 0.9864905
```

Interpreting the results

Sensitivity: We see that the model has a *True Positive* value of 0.293, which means that out of all actual train users (those for whom `train1` is `TRUE`), your model correctly identifies approximately 29.3% of them as train users. In other words, the model is missing a significant portion of true train users, indicating that it may not be very effective at identifying individuals who actually use trains.

Specificity: We see that the model has a *True Negative* value of .9865, which indicates that out of all individuals who do not use trains (those for whom `train1` is `FALSE`), your model correctly identifies about 98.65% of them as non-users. This high specificity suggests that the model is effective at correctly classifying individuals who do not use trains.

Overall: The model has high specificity, meaning it's good at identifying *non-users*, but relatively low sensitivity, indicating it struggles to correctly identify *actual users of the train.* This might imply that the model is biased toward predicting non-users or that there may be features that are not adequately capturing the factors influencing train usage.

In order to overcome issues of data-imbalance in our classification problem, we can make use of the *Random Forest* modeling technique.

2. Random Forest

```r
# Load necessary library
library(randomForest)

# Set seed for reproducibility
set.seed(123)

# Ensure train1 is a factor
train_data$train1 <- as.factor(train_data$train1)

# Fit the Random Forest model with class weights
rf_model <- randomForest(train1 ~ home + HHSIZE + income + urban1 + density,
                         data = train_data,
                         ntree = 500,
                         mtry = 2,
                         importance = TRUE,
                         classwt = c("FALSE" = 1, "TRUE" = 2))
# Assign higher weight to TRUE class

# Predict on test data
predicted_rf <- predict(rf_model, newdata = test_data)
# Predictions will be factors

# Create confusion matrix for Random Forest
confusion_matrix_rf <- table(Actual = test_data$train1,
                             Predicted = predicted_rf)

# Calculate sensitivity (True Positive Rate) for Random Forest
sensitivity_rf <- confusion_matrix_rf["TRUE", "TRUE"] /
                  sum(confusion_matrix_rf["TRUE", ])

# Calculate specificity (True Negative Rate) for Random Forest
specificity_rf <- confusion_matrix_rf["FALSE", "FALSE"] /
                  sum(confusion_matrix_rf["FALSE", ])
```

```
# Print results for Random Forest
print(confusion_matrix_rf)
```

```
##        Predicted
## Actual  FALSE TRUE
##   FALSE  2606  725
##   TRUE     19  138
```

```
cat("Sensitivity (RF):", sensitivity_rf, "\n")
```

```
## Sensitivity (RF): 0.8789809
```

```
cat("Specificity (RF):", specificity_rf, "\n")
```

```
## Specificity (RF): 0.7823476
```

Interpreting the results

Sensitivity: We see that the model has a *Sensitivity Score* of `0.878`. This means that approximately 87.8% of the actual positive cases (train users) were correctly identified by the model. This is a strong sensitivity score, indicating that the model is effective at predicting the TRUE class, once adjusted for the data imbalance.

Specificity: The *Specificity Score* of the model is `0.782`. This indicates that about 78.2% of the actual negative cases (non-train users) were correctly classified. While this specificity score is lower than ideal, it reflects the imbalance in the dataset and suggests the model is better at identifying positive cases compared to negative ones.

Overall: Overall, the model shows a strong ability to detect users likely to use the train (high sensitivity), but there's room for improvement in correctly identifying non-users (specificity).

3. Choosing the correct model

Primary Goal: If your main objective is to maximize the detection of train users (high sensitivity), then Random Forest is the better choice. If minimizing false positives is more critical, then Logistic Regression would be preferable.

Data Imbalance: Given that your data is imbalanced, consider if the cost of false negatives (failing to identify a train user) is higher than the cost of false positives (incorrectly identifying a non-user as a train user). If you prioritize capturing more train users, go with RF.

Model Complexity: Random Forest is a more complex model, which might require more resources for tuning and can be less interpretable than Logistic Regression. If interpretability is a key requirement, LR might be more suitable despite its lower sensitivity.

Since we are looking to get accurate predictions, we'll use the `rf_model` and apply it on the non-rail dataset to predict what percentage of users are likely to use train, once it is built.

```
# Create a dataframe where rail == FALSE
y_not_rail <- y %>% filter(rail == FALSE)

# Filter columns relevant to model building
y_not_rail <- y_not_rail %>% dplyr::select(c("home","HHSIZE","income", "urban1","density", "train1"))

head(y_not_rail)
```

```
##   home HHSIZE income urban1 density train1
## 1    1      2      8  FALSE     0.3  FALSE
## 2    0      3      5   TRUE     3.0  FALSE
## 3    0      2      7   TRUE     7.0  FALSE
## 4    1      3     11   TRUE     1.5  FALSE
## 5    1      2      7   TRUE     1.5  FALSE
## 6    1      4      8   TRUE     3.0  FALSE
```

```
# Predict using the existing Random Forest model on the non-rail dataset
predicted_rf_not_rail <- predict(rf_model, newdata = y_not_rail)

# Calculate the mean of predicted TRUE values
mean_true <- mean(predicted_rf_not_rail == "TRUE")

# Calculate the mean of the original train1 column in y_not_rail
mean_train1 <- mean(y_not_rail$train1 == "TRUE")

# Print the results for comparison
cat("Mean of predicted TRUE values:", mean_true, "\n")
```

```
## Mean of predicted TRUE values: 0.1219501
```

```
cat("Mean of original train1 values:", mean_train1, "\n")
```

```
## Mean of original train1 values: 0.001471966
```

Results

- **Mean of Predicted TRUE Values**: 12.18%

- **Mean of Original train1 Values**: 0.15%

Interpretation

The analysis reveals the following:

1. The mean predicted train usage for cities without rail networks is approximately **12.18%**.
2. In contrast, the current train usage in these cities stands at approximately **0.15%**.

5. Conclusion

Based on these findings, we can conclude that establishing a rail network could significantly increase train usage in these cities. Specifically, the usage rate is projected to increase from **0.15%** to **12.18%**, suggesting a potential increase by a factor of about **81 times**.

Policy Implications

These insights are valuable for policymakers considering investments in rail infrastructure, indicating that such investments could substantially enhance public transportation options and increase ridership.

Considerations

- **Assumptions**: The results are based on the assumptions and patterns learned from the data. Actual impacts may vary based on various factors such as location, population density, existing transportation options, and socioeconomic conditions.
- **Further Analysis**: Additional analyses, such as sensitivity analyses or employing different predictive models, may be beneficial to validate the robustness of these predictions.

6. References

1. Faraway, J. (2016). *Extending the Linear Model with R* (2nd ed.). Chapter 2 & 3.

2. Adams, C. P. (2019). *Learning Microeconometrics with R.* Chapter 5: Estimating Demand.

3. Vinod, H. D. (2020). *Hands-On Intermediate Econometrics using R* (2nd ed.).