Group File Notebook

April 5, 2024

1 Shinkansen Travel Data Statistics Project

1.1 Data Processing

[1]:
```
options(warn=-1)
```

[2]:
```
install.packages("MASS")
```

```
Installing package into '/home/jupyter/R/x86_64-pc-linux-gnu-library/4.3'
(as 'lib' is unspecified)
```

[3]:
```
library(ggplot2)
library(dplyr)
library(reshape2)
library(glmnet)
library(MASS)
```

```
Attaching package: 'dplyr'


The following objects are masked from 'package:stats':

    filter, lag


The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union


Loading required package: Matrix

Loaded glmnet 4.1-8


Attaching package: 'MASS'
```

```
The following object is masked from ‘package:dplyr’:

    select
```

[4]:
```r
travel <- read.csv("TravelTrain.csv", header=T, sep=",")
survey <- read.csv("SurveyTrain.csv", header=T, sep=",")
full <- merge(survey,travel,by.x="ID",by.y="ID")
nonfactors = c("ID", "Age", "Travel_Distance", "DepartureDelay_in_Mins",␣
 ↪"ArrivalDelay_in_Mins")
factors = -which(names(full) %in% nonfactors)
full[, factors] = lapply(full[, factors], as.factor)
head(full)
```

A data.frame: 6 × 25

	ID <int>	Overall_Experience <fct>	Seat_comfort <fct>	Seat_Class <fct>	Arrival_time_con <fct>
1	98800001	0	need improvement	Green Car	excellent
2	98800002	0	poor	Ordinary	excellent
3	98800003	1	need improvement	Green Car	need improvement
4	98800004	0	acceptable	Ordinary	need improvement
5	98800005	1	acceptable	Ordinary	acceptable
6	98800006	1	need improvement	Ordinary	need improvement

[5]:
```r
num_rows_with_na <- sum(apply(full, 1, function(row) any(is.na(row))))
num_rows_with_na
```

390

[6]:
```r
full_complete <- na.omit(full)
head(full_complete)
```

A data.frame: 6 × 25

	ID <int>	Overall_Experience <fct>	Seat_comfort <fct>	Seat_Class <fct>	Arrival_time_con <fct>
1	98800001	0	need improvement	Green Car	excellent
2	98800002	0	poor	Ordinary	excellent
3	98800003	1	need improvement	Green Car	need improvement
4	98800004	0	acceptable	Ordinary	need improvement
5	98800005	1	acceptable	Ordinary	acceptable
6	98800006	1	need improvement	Ordinary	need improvement

[7]:
```r
num_rows_with_na <- sum(apply(full_complete, 1, function(row) any(is.na(row))))
num_rows_with_na
```

0

[8]:
```r
full_complete <- full_complete[ , !(names(full_complete) %in% c("ID"))]
head(full_complete)
```

A data.frame: 6 × 24

	Overall_Experience <fct>	Seat_comfort <fct>	Seat_Class <fct>	Arrival_time_convenient <fct>	Ca <f
1	0	need improvement	Green Car	excellent	ex
2	0	poor	Ordinary	excellent	po
3	1	need improvement	Green Car	need improvement	ne
4	0	acceptable	Ordinary	need improvement	
5	1	acceptable	Ordinary	acceptable	ac
6	1	need improvement	Ordinary	need improvement	ac

1.2 EDA/Visualizations to check for balanced data

1.2.1 Bar graph to see distribution of response variable (overall experience)

[9]:
```
empty_vals <- sapply(full_complete, function(x) x == "")
colSums(empty_vals)

rows_to_keep <- apply(empty_vals, 1, function(row) !any(row))

shinkansen_data <- full_complete[rows_to_keep, ]

colSums(sapply(shinkansen_data, function(x) x == ""))
```

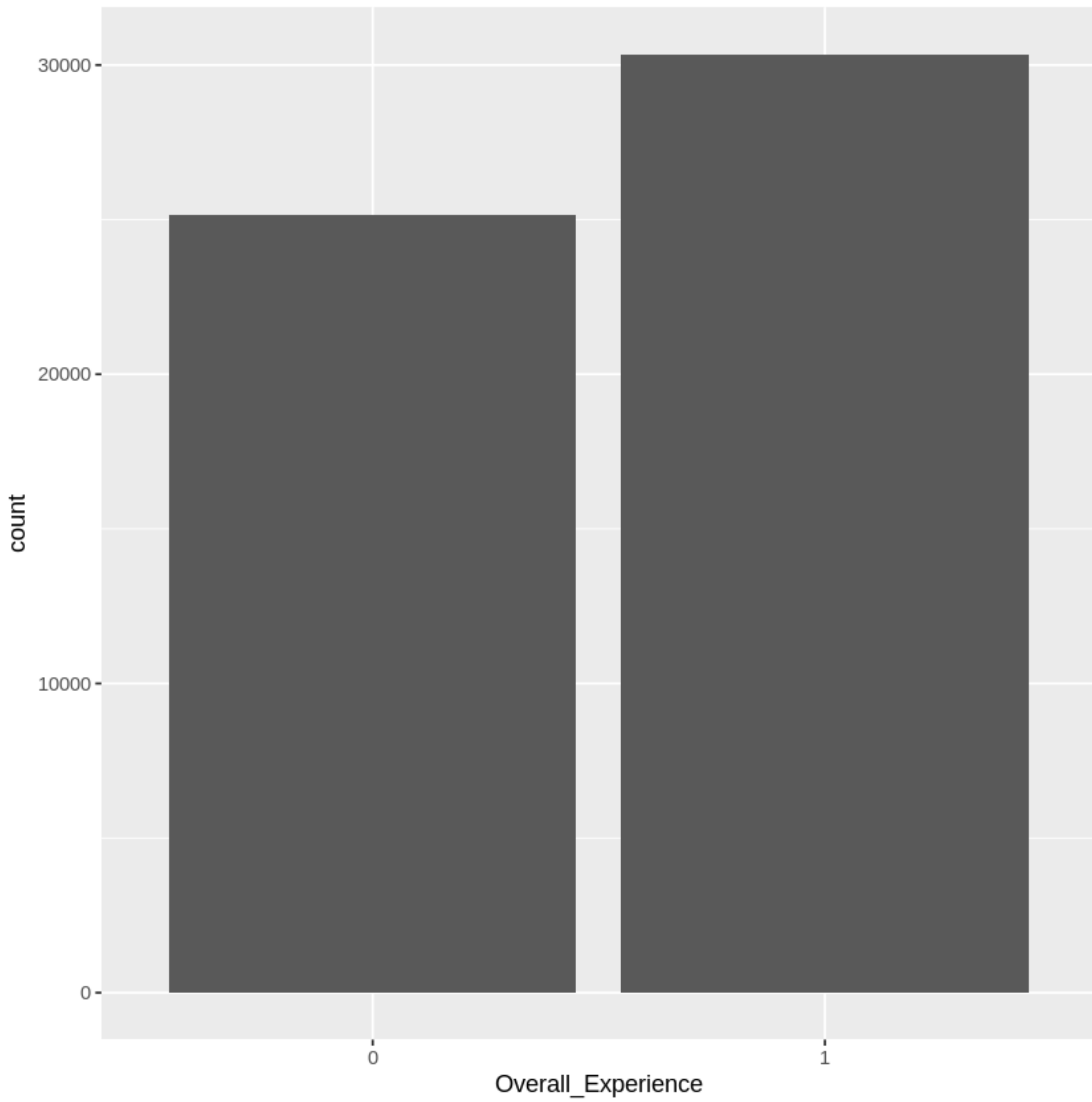
Overall__Experience 0 **Seat__comfort** 61 **Seat__Class** 0 **Arrival__time__convenient** 8891 **Catering** 8702 **Platform__location** 30 **Onboardwifi__service** 30 **Onboard__entertainment** 18 **Online__support** 91 **Onlinebooking__Ease** 73 **Onboard__service** 7569 **Leg__room** 90 **Baggage__handling** 142 **Checkin__service** 77 **Cleanliness** 6 **Online__boarding** 6 **Gender** 47 **CustomerType** 8888 **Age** 0 **TypeTravel** 9161 **Travel__Class** 0 **Travel__Distance** 0 **DepartureDelay__in__Mins** 0 **ArrivalDelay__in__Mins** 0

Overall__Experience 0 **Seat__comfort** 0 **Seat__Class** 0 **Arrival__time__convenient** 0 **Catering** 0 **Platform__location** 0 **Onboardwifi__service** 0 **Onboard__entertainment** 0 **Online__support** 0 **Onlinebooking__Ease** 0 **Onboard__service** 0 **Leg__room** 0 **Baggage__handling** 0 **Checkin__service** 0 **Cleanliness** 0 **Online__boarding** 0 **Gender** 0 **CustomerType** 0 **Age** 0 **TypeTravel** 0 **Travel__Class** 0 **Travel__Distance** 0 **DepartureDelay__in__Mins** 0 **ArrivalDelay__in__Mins** 0

[10]:
```
experience <- ggplot(shinkansen_data, aes(x = Overall_Experience)) +
  geom_bar()
experience

proportions <- prop.table(table(shinkansen_data$Overall_Experience))
proportions
```

```
        0         1
0.4531106 0.5468894
```

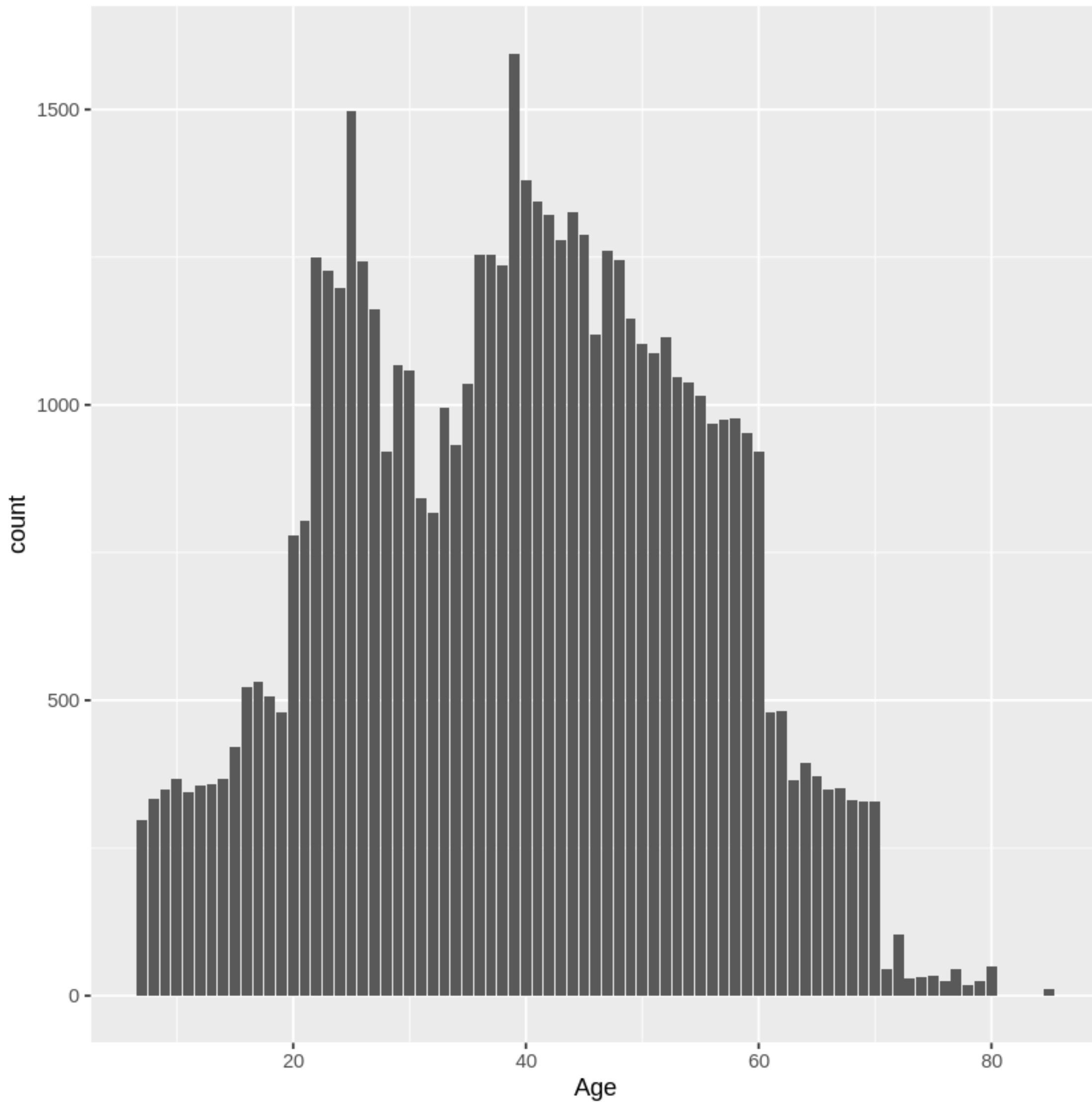


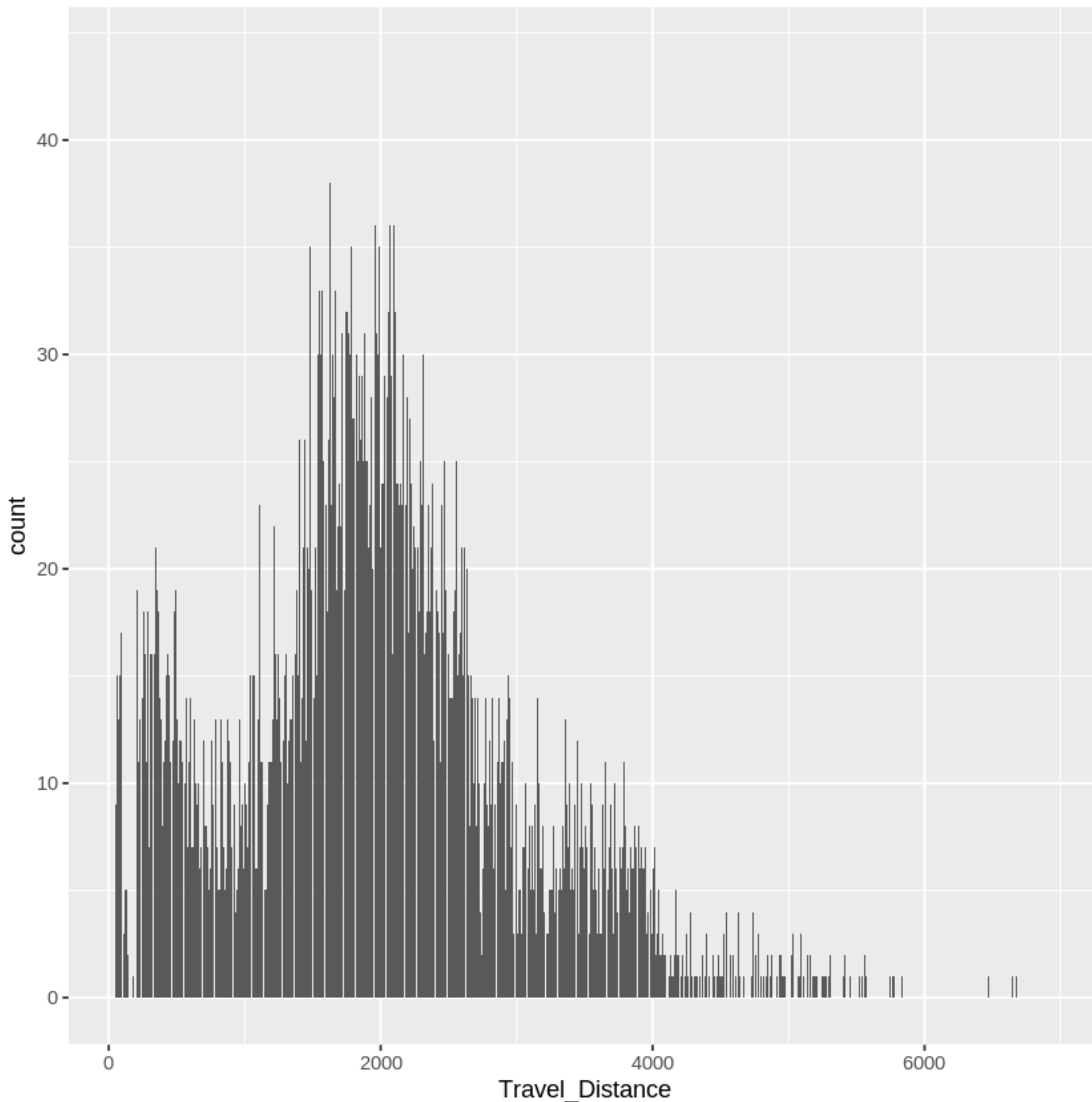
There is a relatively even distribution of overall experiences, with proportions of about 45% and 55%, so the dataset is balanced, and there is no need to undersample/oversample any data

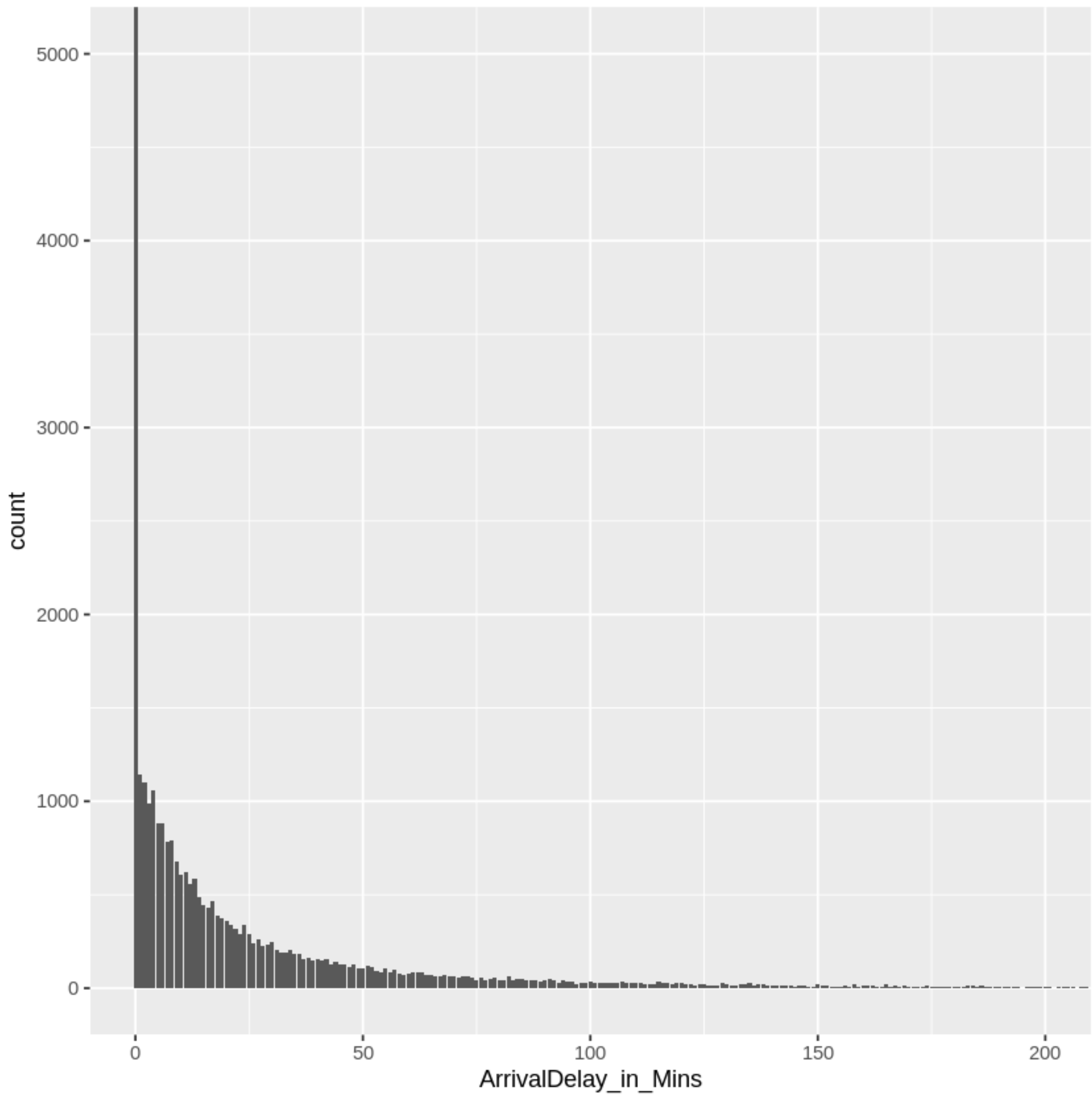
1.2.2 Bar plots to see distributions of numerical variables

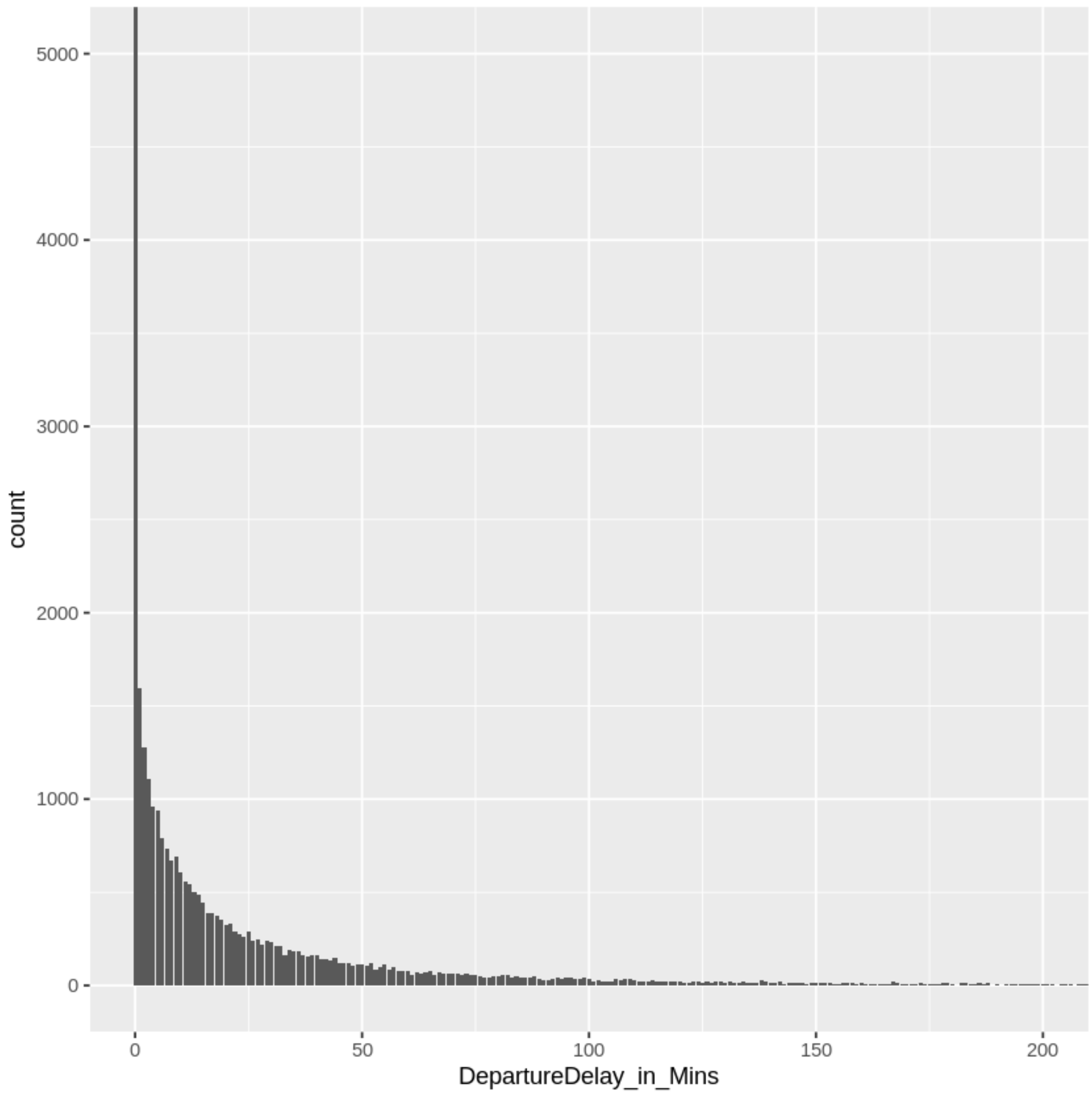
[11]:
```r
plot1 <- ggplot(shinkansen_data, aes(x = Age)) + geom_bar()
plot2 <- ggplot(shinkansen_data, aes(x = Travel_Distance)) + geom_bar()
plot3 <- ggplot(shinkansen_data, aes(x = ArrivalDelay_in_Mins)) +
  geom_bar() +
  coord_cartesian(xlim = c(0, 200), ylim = c(0, 5000))
plot4 <- ggplot(shinkansen_data, aes(x = DepartureDelay_in_Mins)) +
  geom_bar() +
  coord_cartesian(xlim = c(0, 200), ylim = c(0, 5000))
```

```
plot1
plot2
plot3
plot4
```




count
40
30
20
10
0
0
2000
4000
6000
Travel_Distance


5000
4000
3000
2000
1000
0
count
0
50
100
150
200
ArrivalDelay_in_Mins

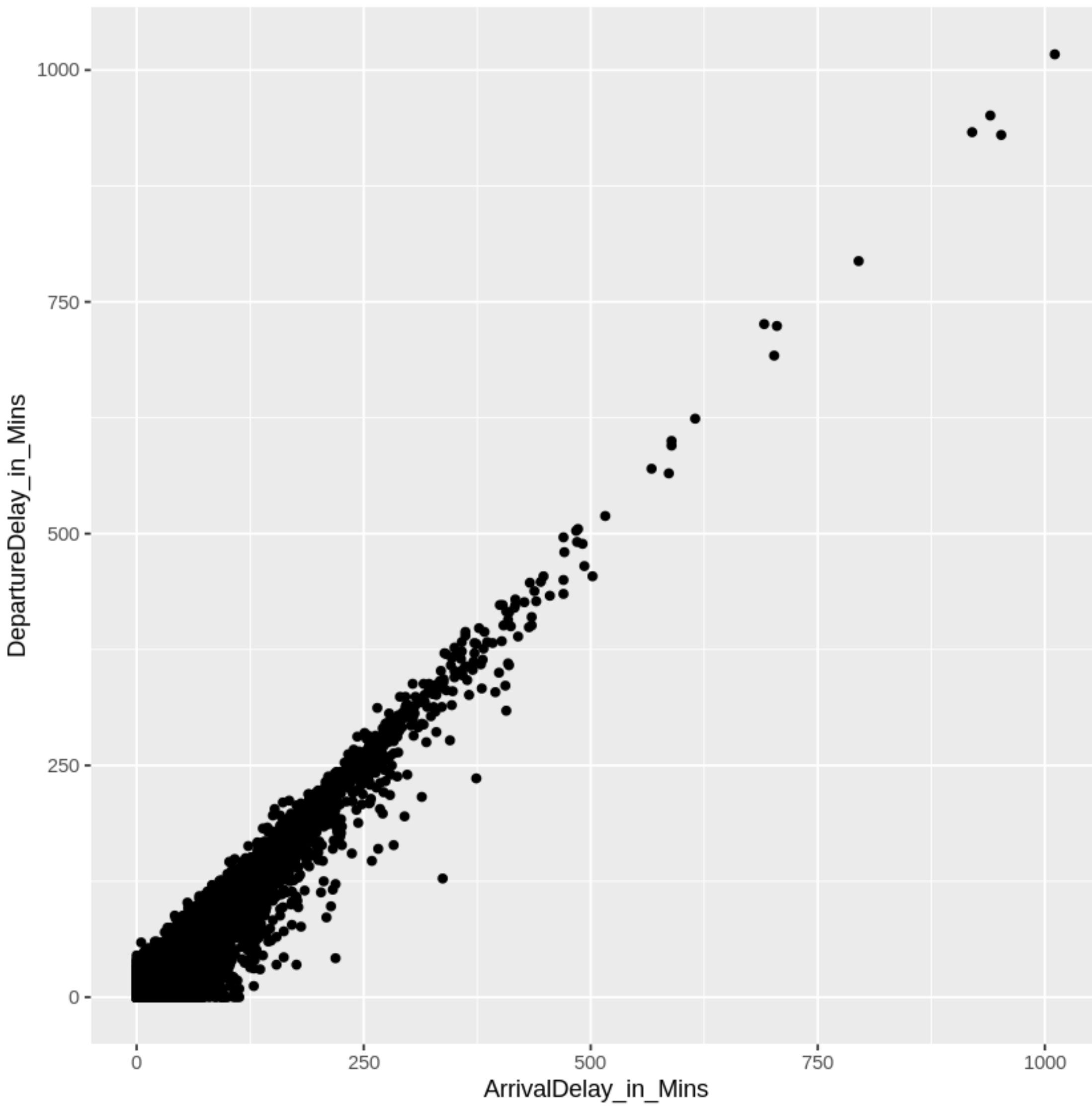


The age and travel distance values follow a roughly normal distribution, with the majority of ages being between 25-60 and the majority of distances being between 1000 and 3000.

The distributions of arrival delay and deparature delay look quite similar, we can plot them against each other to check for a linear relationship

1.2.3 Scatterplot of arrival delay vs departure delay

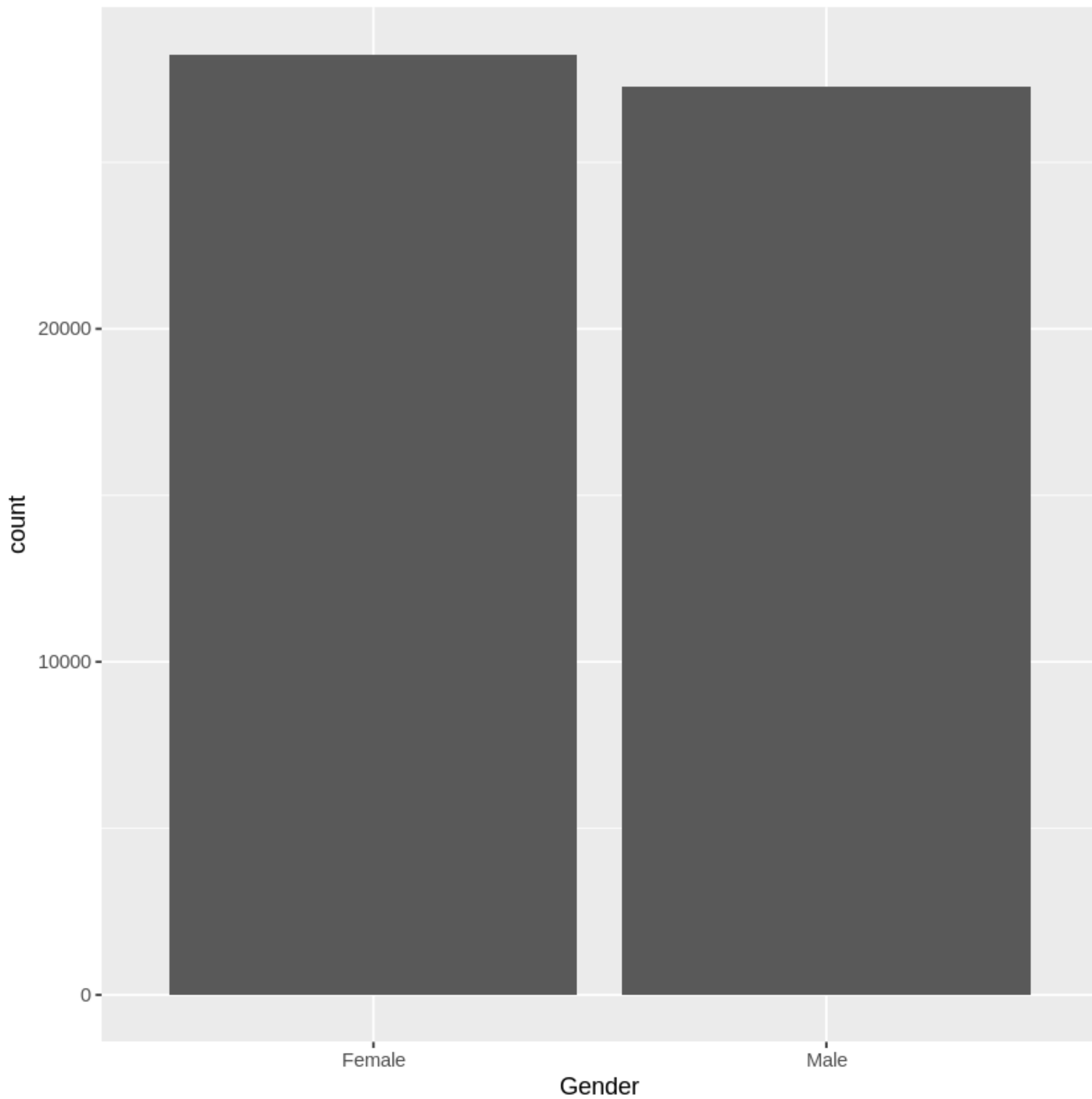
[12]:
```
delays <- ggplot(shinkansen_data, aes(x = ArrivalDelay_in_Mins, y =␣
↪DepartureDelay_in_Mins)) +
  geom_point()
delays
```



These look quite positively correlated, so it is likely we will remove one of the variables during feature selection

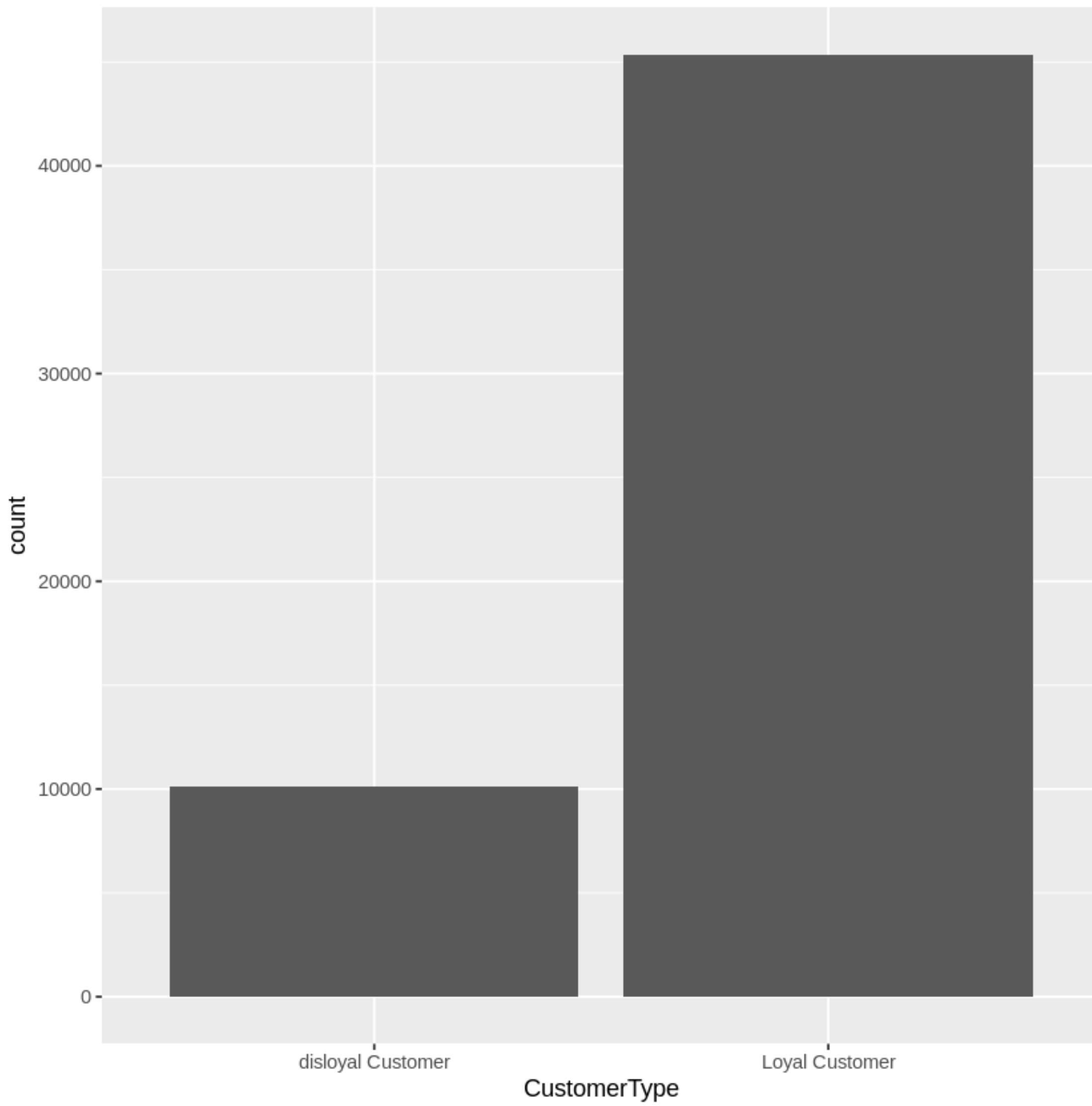
1.2.4 Checking the distribution of the types of customers, to see if data is skewed or not

[13]:
```
# Checking counts of gender
gender <- ggplot(shinkansen_data, aes(x = Gender)) +
  geom_bar()
gender
```



The distribution of Female vs Males seems roughly equal.

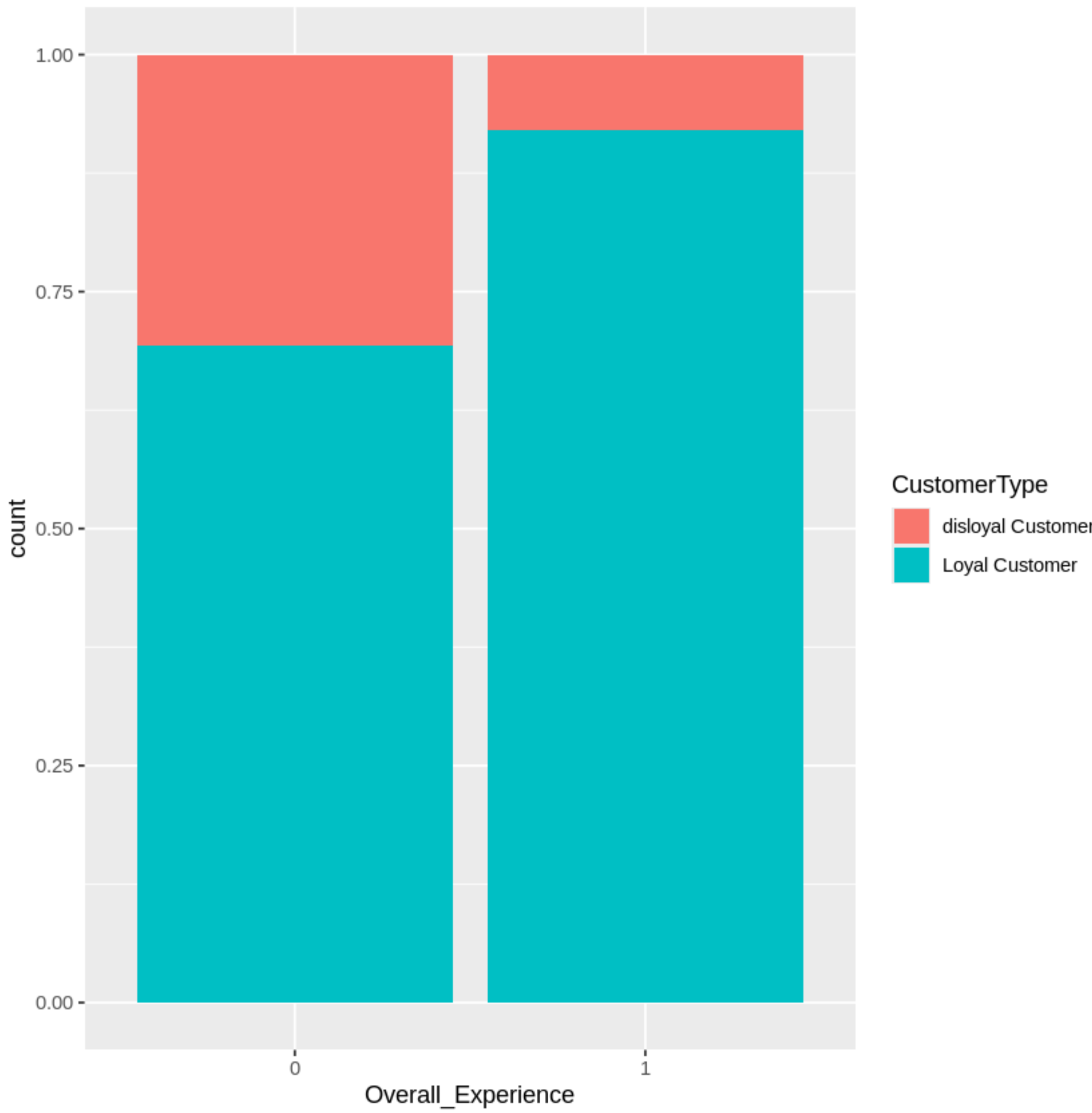
[14]:
```
# checking counts of customer type
customer_type <- ggplot(shinkansen_data, aes(x = CustomerType)) +
  geom_bar()
customer_type
```



There seems to be an imbalance in the type of customer. Having more loyal customers may skew data, in terms of the frequency of these customers giving their surveys, as well as their overall opinion of the system.

We can do a quick check to see if this imbalance is shown in the overall experience (response variable).

[15]:
```
# plotting proportions of loyal and disloyal customers in the responses
ggplot(shinkansen_data, aes(x = Overall_Experience,fill = CustomerType)) +
    geom_bar(position = "fill")
```



It does appear that more loyal customers have a better experience overall, but this isn't an enormous difference, so it appears the imbalance of customer type won't be skewing the results too badly.

1.3 Feature Selection

Our model involves 25 variables (including categorical and continous variables). Of which, the categorical variables involve 5-7 different levels of categories. Our overall data, after processing, involves over 90000 rows. This makes our model significantly complex.

Therefore, we'll first conduct feature selection to reduce model complexity.

But, what should our approach look like?

Understanding Cramer's V Cramer's V is a statistic that will be used to measure the association between two categorical variables, offering a value from 0 to 1. It is calculated from the chi-squared statistic from a contingency table, which assesses the independence of two variables. The formula for Cramer's V is:

$$V = \sqrt{\frac{\chi^2/n}{\min(k-1, r-1)}}$$

where χ^2 is the chi-squared statistic, n is the total number of observations, k is the number of columns, and r is the number of rows in the table. A Cramer's V near 0 signifies a weak association, and one close to 1 indicates a strong associtomer satisfaction.

Cramer's V in the Context of Our Project In our study, we will apply Cramer's V to evaluate the relationship between various categorical predictors and our response variable, `Overall_Experience`. This measure will guide us in understanding the extent to which different factors affect the overall customer experience.

Additionally, we will use Cramer's V for detecting multicollinearity among categorical predictors. Multicollinearity, where predictors are highly inter-correlated, can compromise the integrity of statistical inferences.

Significance of Cramer's V in feature selection process High values of Cramer's V between pairs of variables will highlight redundancies and strategic associations, influencing our decision to remove features that exhibit multicollinearity, and choose variables that best explain `Overall_Experience` of commuters.

[16]:
```
set.seed(123) # For reproducibility
sample_frac <- 0.1
full_sampled <- full_complete[sample(nrow(full_complete), size =
↪floor(nrow(full_complete) * sample_frac)), ]
```

[17]:
```
head(full_sampled)
```

A data.frame: 6 × 24

	Overall_Experience <fct>	Seat_comfort <fct>	Seat_Class <fct>	Arrival_time_convenient <fct>
51883	0	need improvement	Green Car	need improvement
58120	0	good	Ordinary	acceptable
3006	1	good	Green Car	good
30044	1	poor	Green Car	poor
68584	0	good	Ordinary	poor
62823	1	excellent	Green Car	excellent

[18]:
```
# Function to calculate Chi-squared test and Cramér's V for all pairs of
↪categorical variables
association_test <- function(data) {
  cat_vars <- sapply(data, is.factor)
  cat_combinations <- combn(names(cat_vars)[cat_vars], 2)
```

```
  results <- data.frame(Var1 = character(), Var2 = character(), Chi_Squared = numeric(), P_Value = numeric(), Cramers_V = numeric(), stringsAsFactors = FALSE)

  for(i in 1:ncol(cat_combinations)) {
    var1 <- cat_combinations[1, i]
    var2 <- cat_combinations[2, i]

    table <- table(data[[var1]], data[[var2]])
    test <- tryCatch(chisq.test(table), error = function(e) return(e))

    if(!inherits(test, "error")) {
      v <- sqrt(test$statistic / (sum(table) * (min(nrow(table), ncol(table)) - 1)))
    } else {
      v <- NA
    }
    results <- rbind(results, data.frame(Var1 = var1, Var2 = var2, Chi_Squared = if(!is.na(v)) test$statistic else NA, P_Value = if(!is.na(v)) test$p.value else NA, Cramers_V = v))
  }
  return(results)
}

association_results <- association_test(full_sampled)
```

[19]:

```
association_results
```

A data.frame: 190 × 5

	Var1	Var2	Chi_Squared	P_Valu
	<chr>	<chr>	<dbl>	<dbl>
X-squared	Overall_Experience	Seat_comfort	NA	NA
X-squared1	Overall_Experience	Seat_Class	0.3558417	5.50824
X-squared2	Overall_Experience	Arrival_time_convenient	13.5929257	3.45291
X-squared3	Overall_Experience	Catering	618.0773194	2.93779
X-squared4	Overall_Experience	Platform_location	NA	NA
X-squared5	Overall_Experience	Onboardwifi_service	529.2869303	4.11507
X-squared6	Overall_Experience	Onboard_entertainment	3808.0624422	0.00000
X-squared7	Overall_Experience	Online_support	NA	NA
X-squared8	Overall_Experience	Onlinebooking_Ease	1806.3509506	0.00000
X-squared9	Overall_Experience	Onboard_service	NA	NA
X-squared10	Overall_Experience	Leg_room	992.2061091	4.33556
X-squared11	Overall_Experience	Baggage_handling	857.7691208	3.66601
X-squared12	Overall_Experience	Checkin_service	NA	NA
X-squared13	Overall_Experience	Cleanliness	NA	NA
X-squared14	Overall_Experience	Online_boarding	NA	NA
X-squared15	Overall_Experience	Gender	NA	NA
X-squared16	Overall_Experience	CustomerType	718.9507248	7.61787
X-squared17	Overall_Experience	TypeTravel	110.5064868	1.00883
X-squared18	Overall_Experience	Travel_Class	927.8679313	8.58556
X-squared19	Seat_comfort	Seat_Class	NA	NA
X-squared20	Seat_comfort	Arrival_time_convenient	NA	NA
X-squared21	Seat_comfort	Catering	NA	NA
X-squared22	Seat_comfort	Platform_location	NA	NA
X-squared23	Seat_comfort	Onboardwifi_service	NA	NA
X-squared24	Seat_comfort	Onboard_entertainment	NA	NA
X-squared25	Seat_comfort	Online_support	NA	NA
X-squared26	Seat_comfort	Onlinebooking_Ease	NA	NA
X-squared27	Seat_comfort	Onboard_service	NA	NA
X-squared28	Seat_comfort	Leg_room	NA	NA
X-squared29	Seat_comfort	Baggage_handling	NA	NA
X-squared160	Leg_room	TypeTravel	79.29314	5.62869
X-squared161	Leg_room	Travel_Class	210.25995	1.23970
X-squared162	Baggage_handling	Checkin_service	NA	NA
X-squared163	Baggage_handling	Cleanliness	NA	NA
X-squared164	Baggage_handling	Online_boarding	NA	NA
X-squared165	Baggage_handling	Gender	NA	NA
X-squared166	Baggage_handling	CustomerType	76.25177	2.71357
X-squared167	Baggage_handling	TypeTravel	42.39022	6.38977
X-squared168	Baggage_handling	Travel_Class	205.90815	1.54616
X-squared169	Checkin_service	Cleanliness	NA	NA
X-squared170	Checkin_service	Online_boarding	NA	NA
X-squared171	Checkin_service	Gender	NA	NA
X-squared172	Checkin_service	CustomerType	NA	NA
X-squared173	Checkin_service	TypeTravel	NA	NA
X-squared174	Checkin_service	Travel_Class	NA	NA
X-squared175	Cleanliness	Online_boarding	NA	NA
X-squared176	Cleanliness	Gender	NA	NA
X-squared177	Cleanliness	CustomerType	NA	NA
X-squared178	Cleanliness	TypeTravel	NA	NA
X-squared179	Cleanliness	Travel_Class	NA	NA

We see that there are quite a chi-squared tests that failed. We will get rid of the rows that failed. Generally, a Cramer's V association of 0.25 or above shows significant association. We will filter the dataframe to check for this conditoion too.

[20]:
```
association_results <- association_results[!is.
↪nan(association_results$Cramers_V) & association_results$Cramers_V >= 0.25, ]
association_results
```

A data.frame: 16 × 5

	Var1	Var2	Chi_Squared	P_V
	<chr>	<chr>	<dbl>	<db
X-squared3	Overall_Experience	Catering	618.0773	2.93
X-squared6	Overall_Experience	Onboard_entertainment	3808.0624	0.00
X-squared8	Overall_Experience	Onlinebooking_Ease	1806.3510	0.00
X-squared10	Overall_Experience	Leg_room	992.2061	4.33
X-squared11	Overall_Experience	Baggage_handling	857.7691	3.66
X-squared16	Overall_Experience	CustomerType	718.9507	7.61
X-squared18	Overall_Experience	Travel_Class	927.8679	8.58
X-squared54	Arrival_time_convenient	Catering	11754.2662	0.00
X-squared72	Catering	Onboard_entertainment	8717.1552	0.00
X-squared99	Onboardwifi_service	Onboard_entertainment	10907.9394	0.00
X-squared101	Onboardwifi_service	Onlinebooking_Ease	15203.0022	0.00
X-squared123	Onboard_entertainment	Travel_Class	628.0131	2.11
X-squared136	Onlinebooking_Ease	Leg_room	5293.4418	0.00
X-squared137	Onlinebooking_Ease	Baggage_handling	7159.9584	0.00
X-squared154	Leg_room	Baggage_handling	5528.5730	0.00
X-squared189	TypeTravel	Travel_Class	2519.0932	0.00

We can narrow down our search to the 7 predictors that show high association with `Overall_Experience` in the first 7 rows, when categorical variables are concerned.

Upon checking for multicollinearity between any of the 7 predictors - `Catering`, `Onboard_entertainment`, `Onlinebooking_Ease`, `Leg_room`, `Baggage_handling`, `CustomerType`, and `Travel_Class` - we find `Onboard_entertainment`, `Onlinebooking_Ease`, `Baggage_handling`, and `CustomerType` to be the most significant.

`Onboard_entertainment` trumps over `Catering` and `Travel_Class`, due to its stronger association of 0.6365526 with `Overall_Experience`. Similary, `Onlinebooking_Ease` trumps over `Leg_room`.

The chosen 4 predictors are not correlated to each other and demonstrate strong relationships with the response variable, `Overall_Experience`.

Let's check for multicollinearity amongst numerical variables.

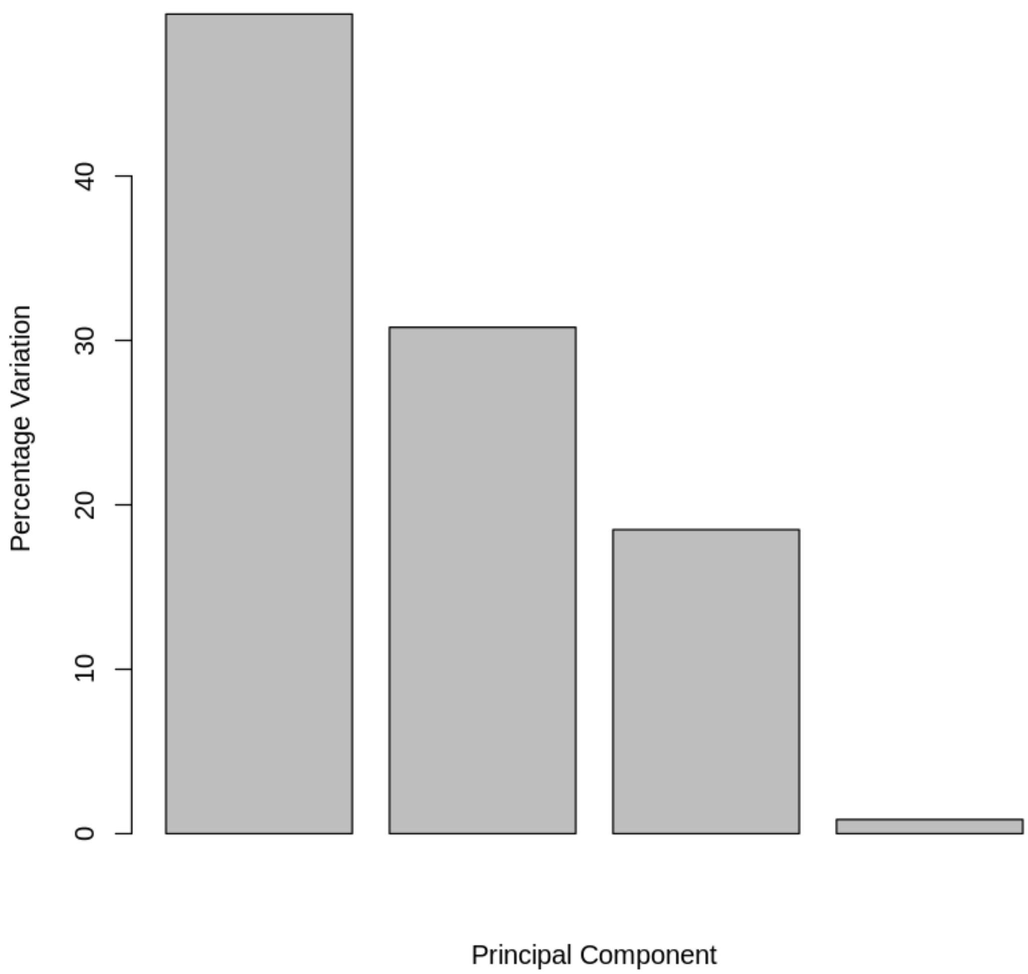
[21]:
```
numerical_data <- full_sampled[c("Age", "Travel_Distance",␣
↪"DepartureDelay_in_Mins", "ArrivalDelay_in_Mins")]
correlation_matrix <- cor(numerical_data, use="complete.obs")
correlation_matrix
```

A matrix: 4 × 4 of type dbl

	Age	Travel_Distance	DepartureDelay_in_M
Age	1.000000000	-0.2553502	0.003081416
Travel_Distance	-0.255350203	1.0000000	0.107836205
DepartureDelay_in_Mins	0.003081416	0.1078362	1.000000000
ArrivalDelay_in_Mins	0.005747269	0.1051325	0.967571942

1.4 Principal Component Analysis

[22]:
```
numcols = full_complete[, c("Age", "Travel_Distance", "DepartureDelay_in_Mins",␣
↪"ArrivalDelay_in_Mins")]
pca <- prcomp(numcols, scale=T)
pca.var <- pca$sdev^2
pca.var.per <- round(pca.var/sum(pca.var) * 100, 2)
barplot(pca.var.per, xlab="Principal Component", ylab="Percentage Variation")
```



We are more concerned about the `Overall_Experience` of onboarding passengers. Hence, we'll neglect `ArrivalDelay_in_Mins` since the survey collects this information after the passengers off-board.

1.5 Model Selection and Model Fitting

Now that we have successfully reduced data complexity. We can go ahead and fit a logistic regression on our population data with our chosen variables. And we continue our analysis to find the 3 best predictors.

1.5.1 Fitting a Logistic Regression Model

[23]:
```
full_model <- glm(Overall_Experience ~ Onboard_entertainment +
 ↪Onlinebooking_Ease + Baggage_handling + CustomerType + Age + Travel_Distance
 ↪+ DepartureDelay_in_Mins, data = full_complete, family = binomial())
summary(full_model)
```

```
Call:
glm(formula = Overall_Experience ~ Onboard_entertainment + Onlinebooking_Ease +
    Baggage_handling + CustomerType + Age + Travel_Distance +
    DepartureDelay_in_Mins, family = binomial(), data = full_complete)

Coefficients:
                                        Estimate Std. Error z value Pr(>|z|)
(Intercept)                            9.757e-01  6.337e-01   1.540 0.123630
Onboard_entertainmentacceptable       -1.754e+00  5.361e-01  -3.272 0.001068
Onboard_entertainmentexcellent         2.510e+00  5.368e-01   4.676 2.93e-06
Onboard_entertainmentextremely poor    7.476e-01  5.382e-01   1.389 0.164801
Onboard_entertainmentgood              3.849e-01  5.359e-01   0.718 0.472648
Onboard_entertainmentneed improvement -1.841e+00  5.363e-01  -3.433 0.000596
Onboard_entertainmentpoor             -1.607e+00  5.366e-01  -2.995 0.002743
Onlinebooking_Easeacceptable          -1.458e+00  3.725e-01  -3.915 9.05e-05
Onlinebooking_Easeexcellent           -6.004e-01  3.724e-01  -1.612 0.106868
Onlinebooking_Easeextremely poor      -1.183e+01  4.563e+01  -0.259 0.795515
Onlinebooking_Easegood                -5.206e-01  3.722e-01  -1.399 0.161811
Onlinebooking_Easeneed improvement    -1.834e+00  3.726e-01  -4.923 8.53e-07
Onlinebooking_Easepoor                -2.511e+00  3.732e-01  -6.727 1.73e-11
Baggage_handlingacceptable            -2.771e-01  2.696e-01  -1.028 0.304112
Baggage_handlingexcellent              1.203e+00  2.693e-01   4.468 7.91e-06
Baggage_handlinggood                   5.610e-01  2.690e-01   2.085 0.037070
Baggage_handlingneed improvement       1.402e-01  2.702e-01   0.519 0.604004
Baggage_handlingpoor                   2.606e-01  2.713e-01   0.961 0.336735
CustomerTypedisloyal Customer         -1.077e+00  3.906e-02 -27.566  < 2e-16
CustomerTypeLoyal Customer             2.443e-01  3.174e-02   7.697 1.39e-14
Age                                    4.780e-03  6.545e-04   7.303 2.81e-13
Travel_Distance                       -2.546e-05  9.749e-06  -2.611 0.009021
```

```
DepartureDelay_in_Mins                -4.527e-03  2.611e-04 -17.341  < 2e-16

(Intercept)
Onboard_entertainmentacceptable       **
Onboard_entertainmentexcellent        ***
Onboard_entertainmentextremely poor
Onboard_entertainmentgood
Onboard_entertainmentneed improvement ***
Onboard_entertainmentpoor             **
Onlinebooking_Easeacceptable          ***
Onlinebooking_Easeexcellent
Onlinebooking_Easeextremely poor
Onlinebooking_Easegood
Onlinebooking_Easeneed improvement    ***
Onlinebooking_Easepoor                ***
Baggage_handlingacceptable
Baggage_handlingexcellent             ***
Baggage_handlinggood                  *
Baggage_handlingneed improvement
Baggage_handlingpoor
CustomerTypedisloyal Customer         ***
CustomerTypeLoyal Customer            ***
Age                                   ***
Travel_Distance                       **
DepartureDelay_in_Mins                ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 129475  on 93988  degrees of freedom
Residual deviance:  71414  on 93966  degrees of freedom
AIC: 71460

Number of Fisher Scoring iterations: 10
```

Our categorical variables have various levels. It seems difficult to understand which global predictors are essential.

1.5.2 Model Selection using Lasso, AIC and Backward Selection

We'll fit a Lasso regression, and gradually increase the strength of the regularization parameter. This way, we'll attempt to visualize more significant parameters.

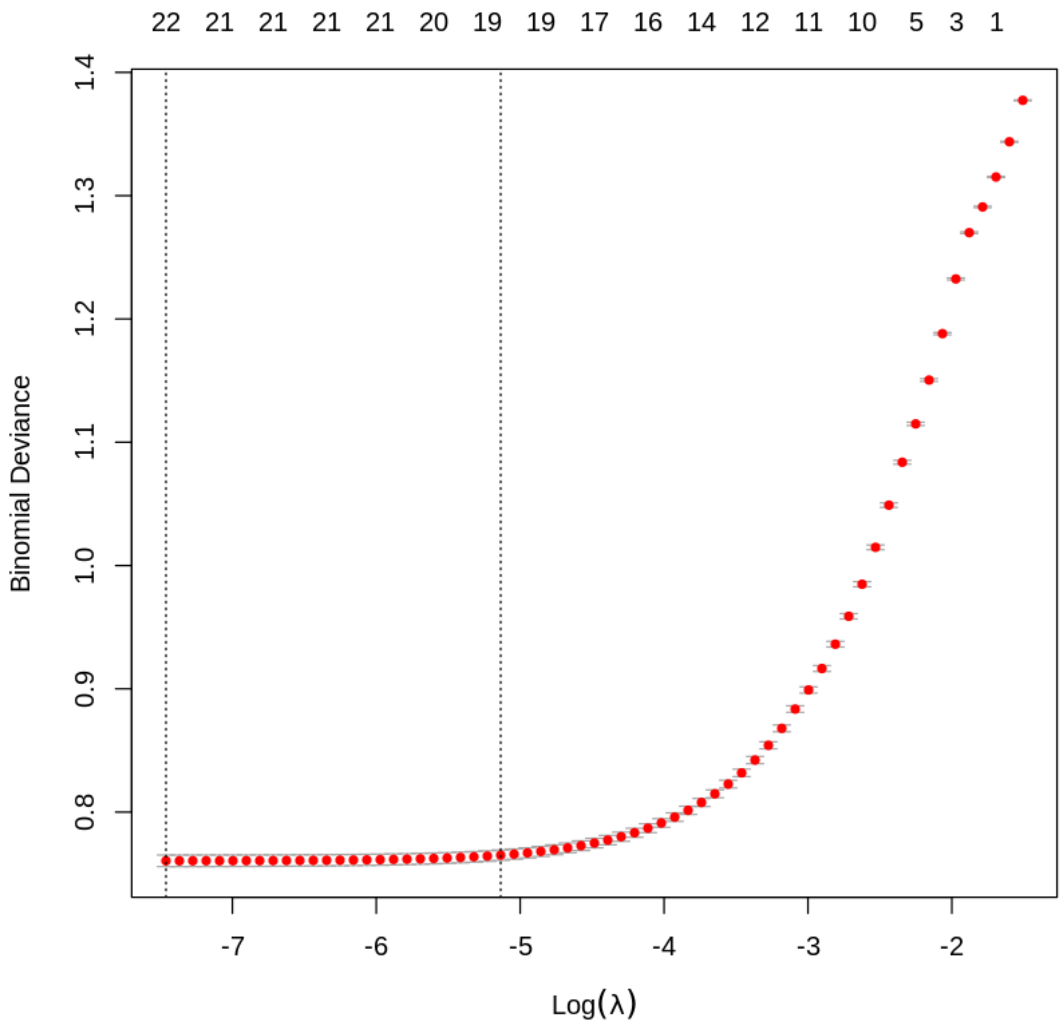
[24]:

```
```

```r
# Lasso Regression
x <- model.matrix(Overall_Experience ~ Onboard_entertainment +
  Onlinebooking_Ease + Baggage_handling + CustomerType + Age + Travel_Distance
  + DepartureDelay_in_Mins - 1, data=full_complete)
y <- full_complete$Overall_Experience

cv_fit <- cv.glmnet(x, y, family="binomial", alpha=1)
plot(cv_fit)
coef(cv_fit, s="lambda.1se")
```

```
24 x 1 sparse Matrix of class "dgCMatrix"
                                               s1
(Intercept)                           0.088100967
Onboard_entertainment                 .
Onboard_entertainmentacceptable      -1.674902229
Onboard_entertainmentexcellent        2.181797525
Onboard_entertainmentextremely poor   0.433887323
Onboard_entertainmentgood             0.292962353
Onboard_entertainmentneed improvement -1.747612091
Onboard_entertainmentpoor            -1.485675173
Onlinebooking_Easeacceptable          .
Onlinebooking_Easeexcellent           0.766934408
Onlinebooking_Easeextremely poor      .
Onlinebooking_Easegood                0.867598163
Onlinebooking_Easeneed improvement   -0.331090251
Onlinebooking_Easepoor               -0.923401067
Baggage_handlingacceptable           -0.623110349
Baggage_handlingexcellent             0.648756855
Baggage_handlinggood                  0.039625096
Baggage_handlingneed improvement     -0.196384595
Baggage_handlingpoor                 -0.037479102
CustomerTypedisloyal Customer        -0.972206758
CustomerTypeLoyal Customer            0.187027309
Age                                   0.002356687
Travel_Distance                       .
DepartureDelay_in_Mins               -0.003143383
```



We already observe `Travel_Distance` being forced to zero. We'll increase the strength parameter a bit further to observe its effects.

[25]:
```
incremented_lambda <- cv_fit$lambda.1se + 0.03
coef(cv_fit, s=incremented_lambda)
```

```
24 x 1 sparse Matrix of class "dgCMatrix"
                                           s1
(Intercept)                        -0.2817019
Onboard_entertainment               .
Onboard_entertainmentacceptable    -0.7193269
Onboard_entertainmentexcellent      2.0032472
Onboard_entertainmentextremely poor .
Onboard_entertainmentgood           0.7006682
```

```
Onboard_entertainmentneed improvement -0.7581119
Onboard_entertainmentpoor             -0.4145802
Onlinebooking_Easeacceptable           .
Onlinebooking_Easeexcellent            0.4512165
Onlinebooking_Easeextremely poor       .
Onlinebooking_Easegood                 0.5478915
Onlinebooking_Easeneed improvement    -0.1118917
Onlinebooking_Easepoor                -0.4348406
Baggage_handlingacceptable            -0.2683590
Baggage_handlingexcellent              0.3613980
Baggage_handlinggood                   .
Baggage_handlingneed improvement       .
Baggage_handlingpoor                   .
CustomerTypedisloyal Customer         -0.6373816
CustomerTypeLoyal Customer             .
Age                                    .
Travel_Distance                        .
DepartureDelay_in_Mins                 .
```

Now we observe `Age` and `DepartureDelay_in_Mins` parameters to also have null coefficients, along with `Travel_Distance`. We can remove these from our analysis too.

We can also observe some critical information over here. For example, `Onboard_entertainmentexcellent` has the highest postitive regeression coefficient of 2.0032472. With a targeted strategic approach, Shinkansen trains maintenance team can work to ensure better onboard entertainment experience of its passengers to significantly improve their `Overall_Experience`.

Note: We decided to not perform stepwise selection (in any direction) due to the complex nature of our data, and the computational demands of running this algorithm. Instead of performing an exhaustive search and checking for all model combinations (even if redundant), we decided to optimize our search process.

Below, we attempt to replicate the stepwise selection process by simulating a function with a simple for loop that fits a logistic regression for some parameters and returns the models AIC values. The catch here is that we only perform this function for a parameter size of 3 predictors since that's what we are interested in. Then, we simply find the model with the lowest AIC value, and its predictor variable combination.

[26]:
```
predictors <- c("Onboard_entertainment", "Onlinebooking_Ease",␣
↪"Baggage_handling",
                "CustomerType")
combinations <- combn(predictors, 3)

aic_values <- sapply(1:ncol(combinations), function(i) {
  formula_str <- paste("Overall_Experience ~", paste(combinations[, i],␣
↪collapse = " + "))
  formula <- as.formula(formula_str)
  model <- glm(formula, data = full_complete, family = "binomial")
```

```
  AIC(model)
})
```

[27]:
```
# Find the index of the combination with the lowest AIC
best_model_index <- which.min(aic_values)
```

[28]:
```
best_combination <- combinations[, best_model_index]
best_combination
```

1. 'Onboard_entertainment' 2. 'Onlinebooking_Ease' 3. 'CustomerType'

1.6 Conclusion

From our analysis, the three variables that contribute the most to a person having a positive experience on a Shinkansen is the customer's loyalty, the quality of the onboard entertainment, and the ease of online booking. This insight can be used to help increase the amount of customers having a positive experience by creating stratetgies to push more people to become loyal customer and improving onboard entertainment as well as the online booking experience.

One thing that could have been done to improve the study would be to make use of the training and testing splits given on kaggle. Instead of only using the training data and fitting models on itself, we could have used a training/testing split to ensure the model is more suitable in general and lower the risk of overfitting. Additionally, given the large number of variables, we could have chosen more variables instead of the best three; there could be other variables that are also significant but are ignored due to choosing only three.