KHAFAJI_FA5

Titanic Dataset Classification via Logistic Regression

Data Preprocessing

Loading Dataset

First, let's load our data. The online data set is already separated into training and testing data sets. However, the test data set is missing the "survived" column, and so for the intent of our study, we will mostly be looking at the training data set.

However, any change in the structure of the data set, e.g. one-hot encoding, will be done on both.

```
titanic_test_data <- read_csv('test.csv')
```

```
## Rows: 418 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): Name, Sex, Ticket, Cabin, Embarked
## dbl (6): PassengerId, Pclass, Age, SibSp, Parch, Fare
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
cat("\n\n")
```

```
titanic_train_data <- read_csv('train.csv')
```

```
## Rows: 891 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): Name, Sex, Ticket, Cabin, Embarked
## dbl (7): PassengerId, Survived, Pclass, Age, SibSp, Parch, Fare
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
titanic_data <- bind_rows(titanic_train_data, titanic_test_data)

head(titanic_data)
```

```
## # A tibble: 6 x 12
##   PassengerId Survived Pclass Name    Sex     Age SibSp Parch Ticket  Fare Cabin
##         <dbl>    <dbl>  <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <chr>  <dbl> <chr>
## 1           1        0      3 Braund~ male     22     1     0 A/5 2~  7.25 <NA>
## 2           2        1      1 Cuming~ fema~    38     1     0 PC 17~ 71.3  C85
## 3           3        1      3 Heikki~ fema~    26     0     0 STON/~  7.92 <NA>
## 4           4        1      1 Futrel~ fema~    35     1     0 113803 53.1  C123
## 5           5        0      3 Allen,~ male     35     0     0 373450  8.05 <NA>
## 6           6        0      3 Moran,~ male     NA     0     0 330877  8.46 <NA>
## # i 1 more variable: Embarked <chr>
```

The data set has the following features:

Pclass: Passenger class (1st, 2nd, 3rd)

Sex: Gender of the passenger

Age: Age of the passenger

SibSp: Number of siblings/spouses aboard

Parch: Number of parents/children aboard

Fare: Ticket fare

Embarked: Port of embarkation

Survived: Target variable (0 = No, 1 = Yes)

Quick Exploratory Data Analysis

Let's do a quick exploratory data analysis using the skimr package

```
titanic_data %>% skim()
```

Table 1: Data summary

Name	Piped data
Number of rows	1309
Number of columns	12
Column type frequency:	
character	5
numeric	7
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Name	0	1.00	12	82	0	1307	0
Sex	0	1.00	4	6	0	2	0
Ticket	0	1.00	3	18	0	929	0
Cabin	1014	0.23	1	15	0	186	0
Embarked	2	1.00	1	1	0	3	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
PassengerId	0	1.00	655.00	378.02	1.00	328.0	655.00	982.00	1309.00	
Survived	418	0.68	0.38	0.49	0.00	0.0	0.00	1.00	1.00	
Pclass	0	1.00	2.29	0.84	1.00	2.0	3.00	3.00	3.00	
Age	263	0.80	29.88	14.41	0.17	21.0	28.00	39.00	80.00	
SibSp	0	1.00	0.50	1.04	0.00	0.0	0.00	1.00	8.00	
Parch	0	1.00	0.39	0.87	0.00	0.0	0.00	0.00	9.00	
Fare	1	1.00	33.30	51.76	0.00	7.9	14.45	31.27	512.33	

```
titanic_quanti_only <- titanic_data %>% select(c("Pclass", "Age", "SibSp", "Parch", "Fare"))
```

We can see that the feature "Cabin" is missing for most of the observations, with 687 missing instances, or 22.9% completeness rate.

Given that The categorical variable "Embarked" have 2 missing observations, we would be looking to simply delete these 2 instances, as their deletion would unlikely to change the overall effect to the model.

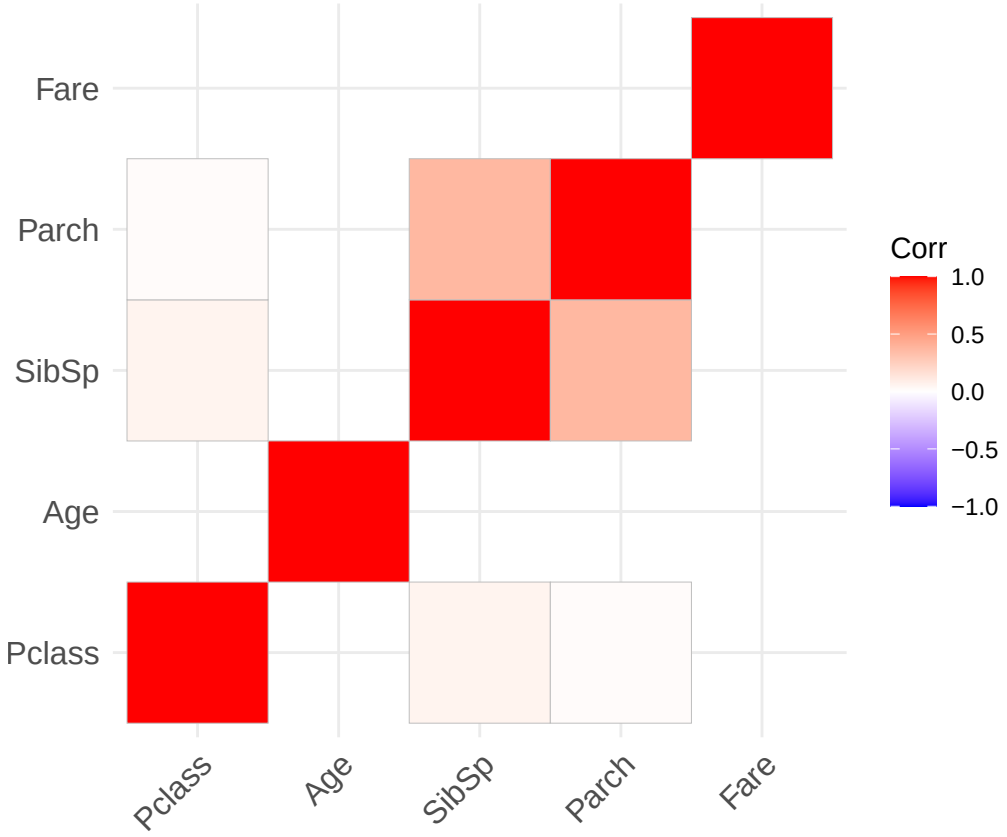
Fare also has one missing observation, and we should also drop that row.

Age is missing 263 observations, or being only 80% complete. For the last two variables, we would be looking to impute the missing values using a multivariate imputation.

Notice that survived is missing for 400 observations, this came from the test data set, and we believe it is better to leave them there, given that they are blank by design.

Let's then check the correlation matrix of the quantitative columns

```
cor_matrix_titanic <- cor(titanic_quanti_only)
ggcorrplot(cor_matrix_titanic)
```



As we can see, the Parch and SibSp correlated, although only moderately. Thus, we assume that there would be no multicollinearity problems.

Data Cleaning:

First, let's filter out the columns we won't need for building the model, i.e. the columns that only serve to identify the given entry. We included tickets here because, although not necessarily specific to only that observation, it is not feasible to add in our model with 681 unique observations out of 891 observations.

We have also taken the liberty of converting the "survived" and "Pclass" features into a factor data type.

```
titanic_data <- titanic_data %>% mutate(across(c("Survived", "Pclass"), as.factor)) %>% select(-c("Name

str(titanic_data)
```

```
## tibble [1,309 x 9] (S3: tbl_df/tbl/data.frame)
##  $ Survived: Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex     : chr [1:1309] "male" "female" "female" "female" ...
##  $ Age     : num [1:1309] 22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp   : num [1:1309] 1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch   : num [1:1309] 0 0 0 0 0 0 0 1 2 0 ...
##  $ Fare    : num [1:1309] 7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin   : chr [1:1309] NA "C85" NA "C123" ...
##  $ Embarked: chr [1:1309] "S" "C" "S" "S" ...
```

```
set.seed(12345)

titanic_data <- titanic_data %>% select(-c("Cabin")) %>% drop_na(c("Embarked", "Fare"))


titanic_imputed <- mice(titanic_data, maxit = 0, print=FALSE)
```

```
## Warning: Number of logged events: 2
```

```
meth = titanic_imputed$method
pred_matrix <- titanic_imputed$predictorMatrix

meth[!names(meth) %in% c("Age")] <- ""
pred_matrix[, "Survived"] <- 0

titanic_imputed <- mice(titanic_data, method = meth, predictorMatrix = pred_matrix ,printFlag = FALSE)

titanic_data <- complete(titanic_imputed, 1)
```

Note that we excluded the target variable, survived, from the multivariate imputation, both as a column to be imputed and as a predictor.

```
titanic_data %>% skim()
```

Table 4: Data summary

Name	Piped data
Number of rows	1306
Number of columns	8
Column type frequency:	
character	2
factor	2
numeric	4
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Sex	0	1	4	6	0	2	0
Embarked	0	1	1	1	0	3	0

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Survived	417	0.68	FALSE	2	0: 549, 1: 340
Pclass	0	1.00	FALSE	3	3: 708, 1: 321, 2: 277

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	29.28	14.20	0.17	20.12	28.00	38.00	80.00	
SibSp	0	1	0.50	1.04	0.00	0.00	0.00	1.00	8.00	
Parch	0	1	0.39	0.87	0.00	0.00	0.00	0.00	9.00	
Fare	0	1	33.22	51.77	0.00	7.90	14.45	31.27	512.33	

Next, we want to create one hot encoding of the following categorical features:

```
dummy_model <- dummyVars( ~ Pclass + Embarked, data = titanic_data, fullRank = TRUE)
df_encoded <- predict(dummy_model, newdata = titanic_data)

titanic_data_final <- cbind(titanic_data[, !(names(titanic_data) %in% c("Pclass", "Embarked"))], df_enc
  mutate(across(c("Pclass.2", "Pclass.3", "EmbarkedQ", "EmbarkedS"), as.factor)) %>%
  mutate(across(where(~ is.factor(.) && nlevels(.) == 2),
                ~ factor(., levels = levels(.), labels = c("No", "Yes")))) %>%
  mutate(across("Sex", as.factor))


head(titanic_data_final)
```

```
##   Survived    Sex Age SibSp Parch    Fare Pclass.2 Pclass.3 EmbarkedQ EmbarkedS
## 1       No   male  22     1     0  7.2500       No      Yes        No       Yes
## 2      Yes female  38     1     0 71.2833       No       No        No        No
## 3      Yes female  26     0     0  7.9250       No      Yes        No       Yes
## 4      Yes female  35     1     0 53.1000       No       No        No       Yes
## 5       No   male  35     0     0  8.0500       No      Yes        No       Yes
## 6       No   male  30     0     0  8.4583       No      Yes       Yes        No
```

```
titanic_data_final %>% skim()
```

Table 8: Data summary

Name	Piped data
Number of rows	1306
Number of columns	10
Column type frequency:	
factor	6

numeric	4
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Survived	417	0.68	FALSE	2	No: 549, Yes: 340
Sex	0	1.00	FALSE	2	mal: 842, fem: 464
Pclass.2	0	1.00	FALSE	2	No: 1029, Yes: 277
Pclass.3	0	1.00	FALSE	2	Yes: 708, No: 598
EmbarkedQ	0	1.00	FALSE	2	No: 1183, Yes: 123
EmbarkedS	0	1.00	FALSE	2	Yes: 913, No: 393

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	29.28	14.20	0.17	20.12	28.00	38.00	80.00	
SibSp	0	1	0.50	1.04	0.00	0.00	0.00	1.00	8.00	
Parch	0	1	0.39	0.87	0.00	0.00	0.00	0.00	9.00	
Fare	0	1	33.22	51.77	0.00	7.90	14.45	31.27	512.33	

Since we will be using logistic regression, it is not required to standardize our data, hence, our work at data cleaning is done.

Now, we are ready to split the data set again:

```
titanic_train <- titanic_data_final %>% dplyr::filter(!is.na(Survived))
titanic_test <- titanic_data_final %>% dplyr::filter(is.na(Survived)) %>% select(-c("Survived"))
```

but since only the training data has observations for survived, let's split it further

```
train_index <- createDataPartition(titanic_train$Survived, p = 0.8, list = FALSE)
train_data <- titanic_train[train_index, ]
test_data <- titanic_train[-train_index, ]
```

More Exploratory Data Analysis

```
titanic_train %>% skim()
```

Table 11: Data summary

Name	Piped data
Number of rows	889
Number of columns	10
Column type frequency:	
factor	6
numeric	4

Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Survived	0	1	FALSE	2	No: 549, Yes: 340
Sex	0	1	FALSE	2	mal: 577, fem: 312
Pclass.2	0	1	FALSE	2	No: 705, Yes: 184
Pclass.3	0	1	FALSE	2	Yes: 491, No: 398
EmbarkedQ	0	1	FALSE	2	No: 812, Yes: 77
EmbarkedS	0	1	FALSE	2	Yes: 644, No: 245

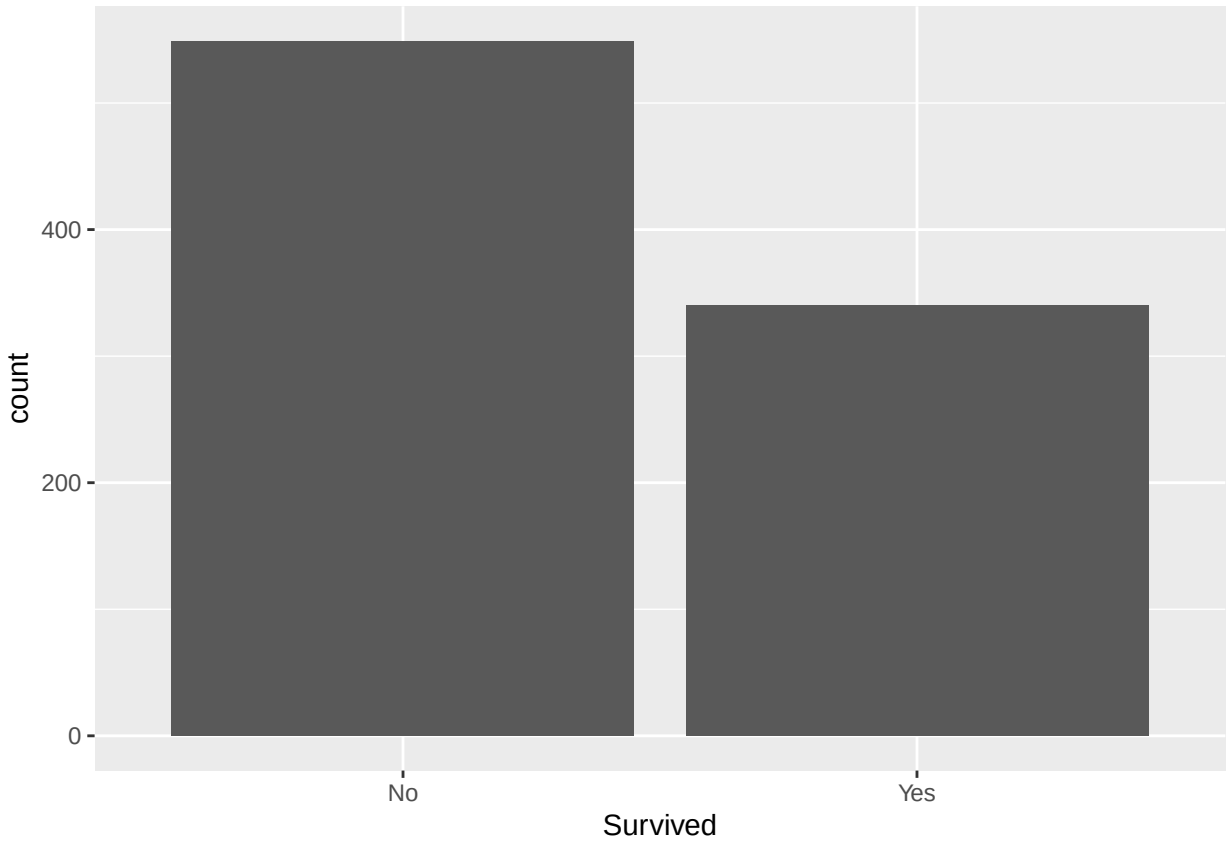
Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Age	0	1	29.17	14.24	0.33	20.0	28.00	38	80.00	
SibSp	0	1	0.52	1.10	0.00	0.0	0.00	1	8.00	
Parch	0	1	0.38	0.81	0.00	0.0	0.00	0	6.00	
Fare	0	1	32.10	49.70	0.00	7.9	14.45	31	512.33	

Now that we have cleaned our data, we can do more exploratory data analysis.

Let's get the distribution of the surviving passengers:

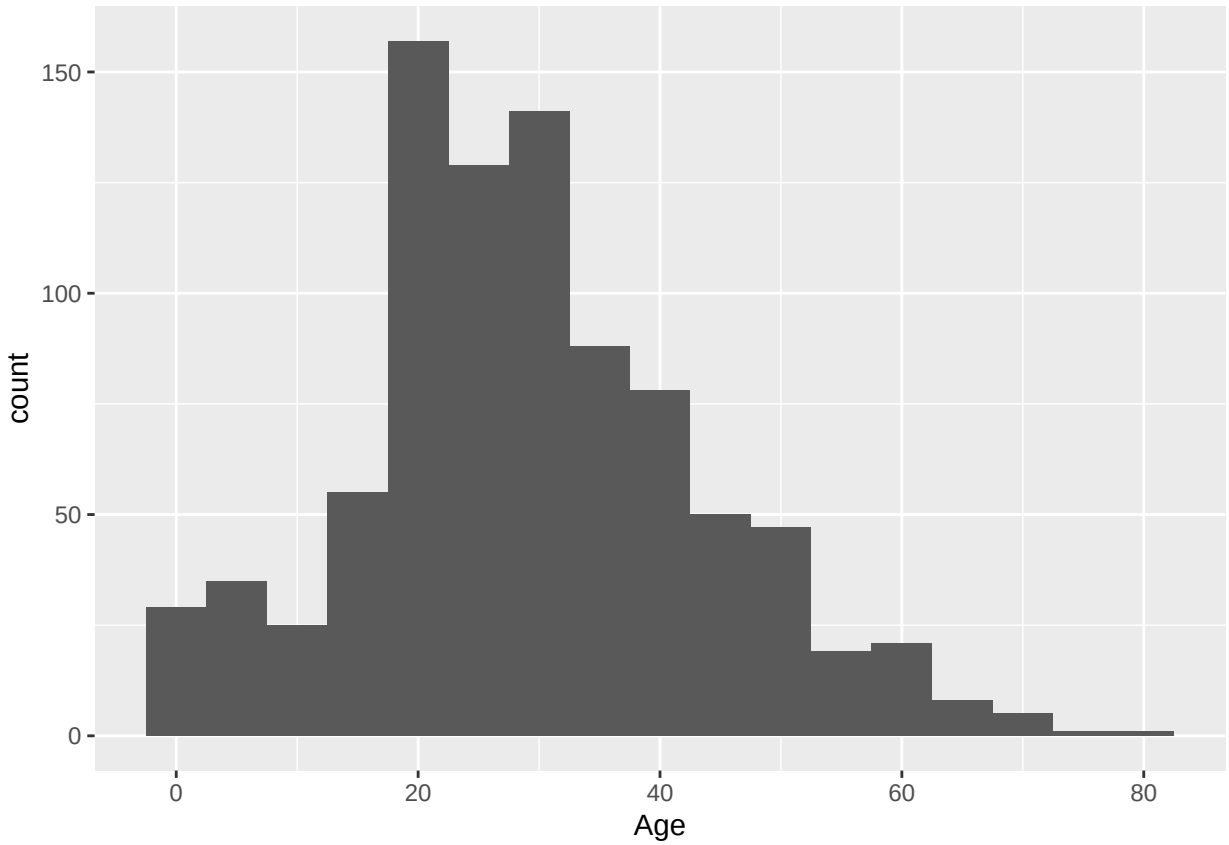
```
titanic_train %>% ggplot(aes(x=Survived)) +
  geom_bar()
```



As we can see, the number of passengers surviving the tragedy is less than those that perished because of it, with the number of surviving passengers being around 60% to 70% of the perished passengers.

Now, let's check the age distribution of the passengers:

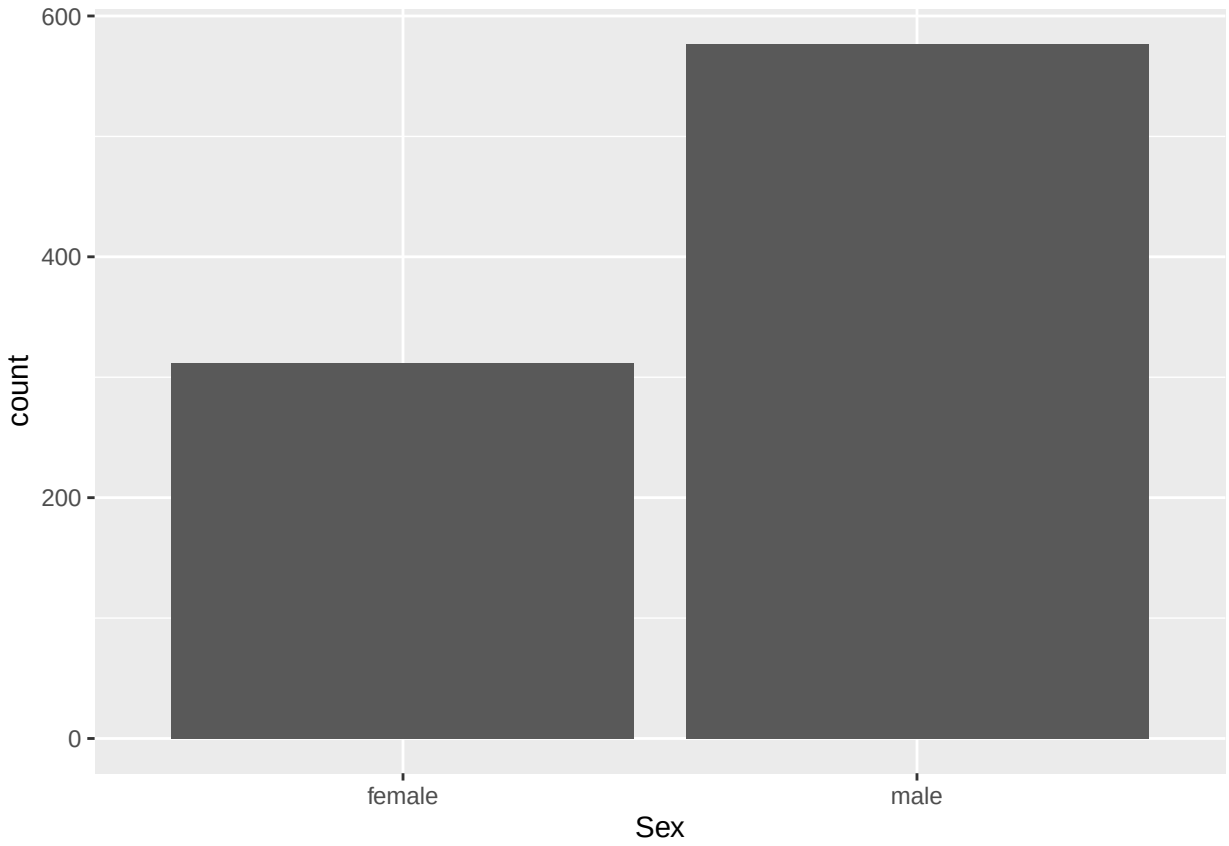
```
titanic_train %>% ggplot(aes(x=Age)) +
  geom_histogram(binwidth = 5)
```



We can see that the distribution is skewed to the left, with most passengers being around 20 to 35.

Now, let's check the sex.

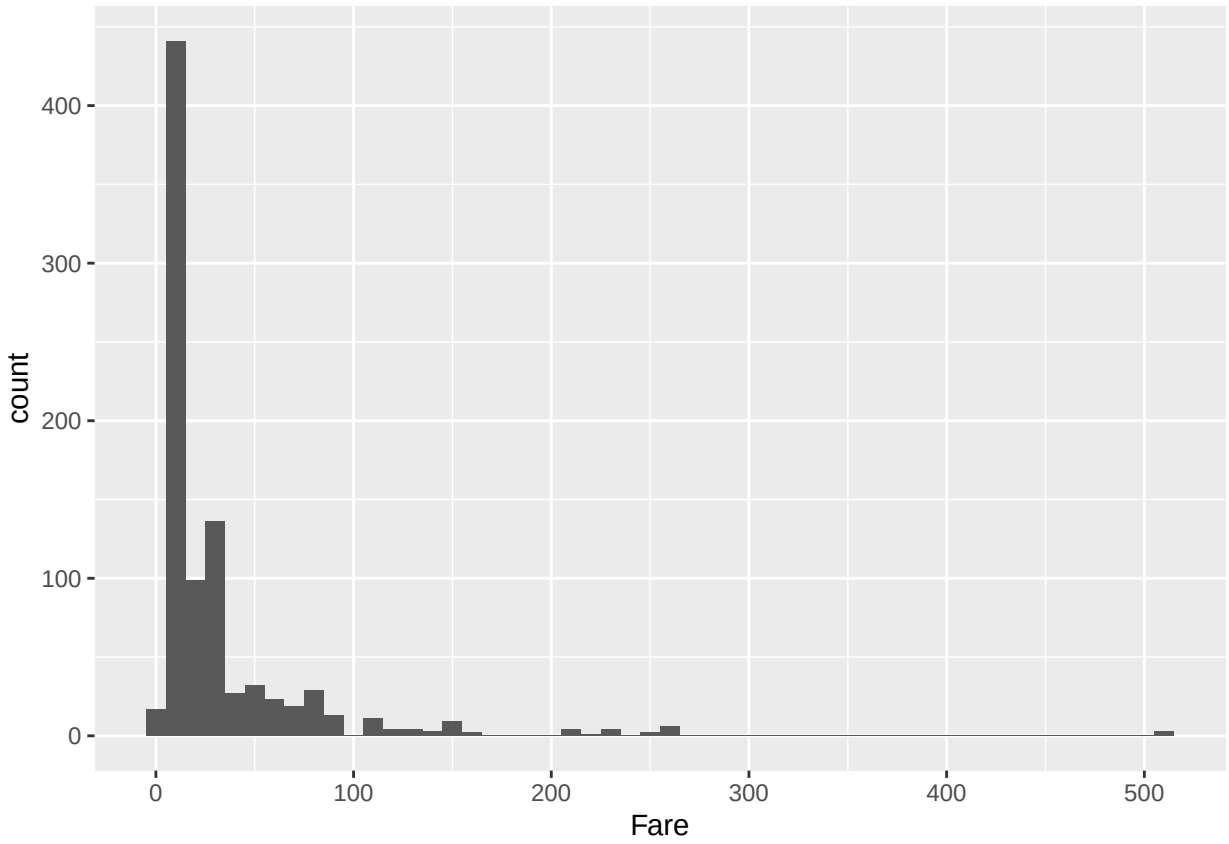
```
titanic_train %>% ggplot(aes(x=Sex)) +
  geom_bar()
```



Most of the passengers of the titanic are male, double than the number of females in the ship.

Let's then check the fare:

```
titanic_train %>% ggplot(aes(x=Fare)) +
  geom_histogram(binwidth = 10)
```



Most of the passengers paid somewhere between 0 to 40 USD for their ticket.

Model Implementation and Evaluation

Now, let's train a logistic regression model using caret:

```
model <- train(
  Survived ~ .,
  data = train_data,
  method = "glm",
  family = binomial,
  trControl = trainControl(
    method = "cv",
    number = 10,
    classProbs = TRUE,
    summaryFunction = twoClassSummary,
    savePredictions = "final"
  ),
  metric = "ROC"
)
```

Now, let's test the new model:

Creating predictions

```
pred_class <- predict(model, newdata = test_data)
true_class <- test_data$Survived
```

```r
# Probabilities for ROC
pred_probs <- predict(model, newdata = test_data, type = "prob")[, "Yes"]
```

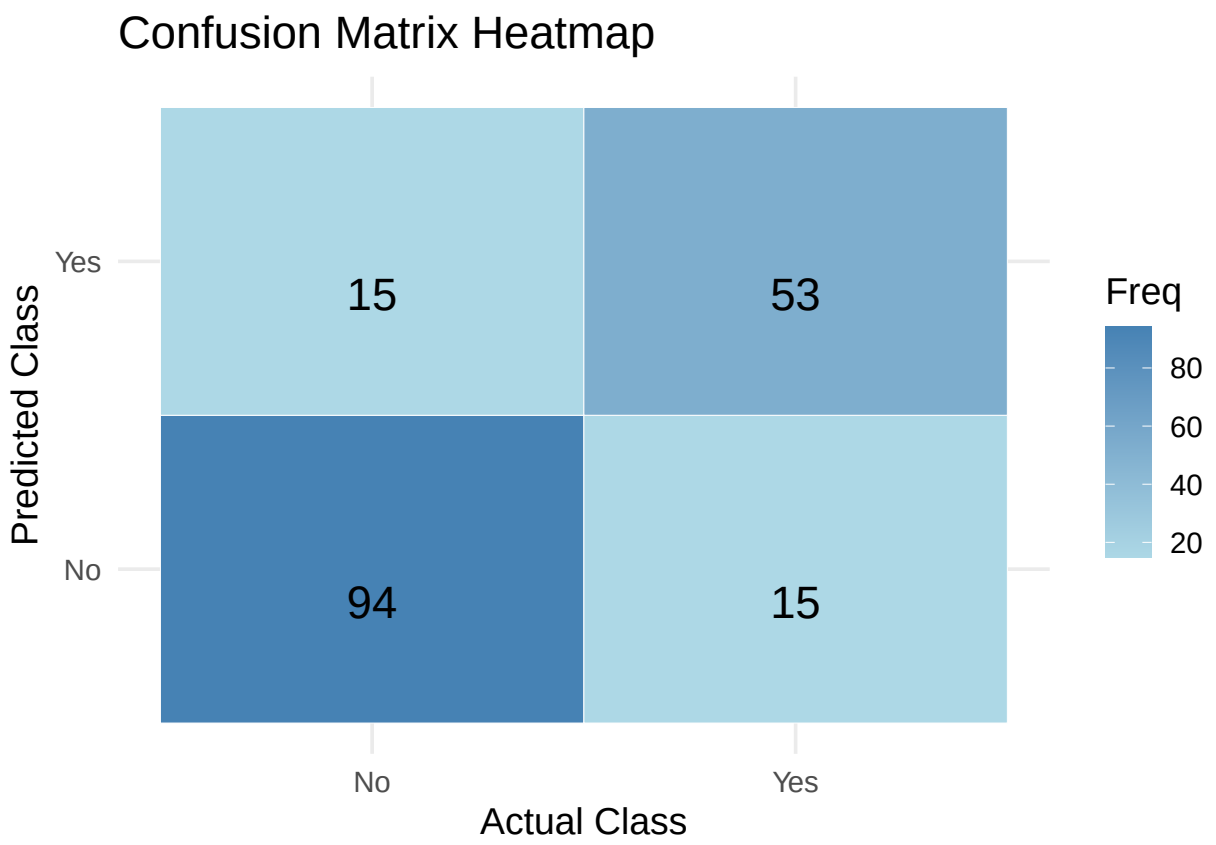
Confusion Matrix, Accuracy, Precicion, Recall, and F1-Score

```r
cm <- confusionMatrix(pred_class, true_class, positive = "Yes", mode = "prec_recall")
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  94  15
##        Yes 15  53
##
##                Accuracy : 0.8305
##                  95% CI : (0.767, 0.8826)
##     No Information Rate : 0.6158
##     P-Value [Acc > NIR] : 4.325e-10
##
##                   Kappa : 0.6418
##
##  Mcnemar's Test P-Value : 1
##
##               Precision : 0.7794
##                  Recall : 0.7794
##                      F1 : 0.7794
##              Prevalence : 0.3842
##          Detection Rate : 0.2994
##    Detection Prevalence : 0.3842
##       Balanced Accuracy : 0.8209
##
##        'Positive' Class : Yes
##
```

```r
cm_df <- as.data.frame(cm$table)
names(cm_df) <- c("Predicted", "Actual", "Freq")

ggplot(data = cm_df, aes(x = Actual, y = Predicted, fill = Freq)) +
  geom_tile(color = "white") +
  geom_text(aes(label = Freq), vjust = 1.5, size = 6) +
  scale_fill_gradient(low = "lightblue", high = "steelblue") +
  labs(
    title = "Confusion Matrix Heatmap",
    x = "Actual Class",
    y = "Predicted Class"
  ) +
  theme_minimal(base_size = 14)
```



We can see from the given statistics that the accuracy of our model is at 0.8305, While its precision, recall, and F1 score are all at 0.7794

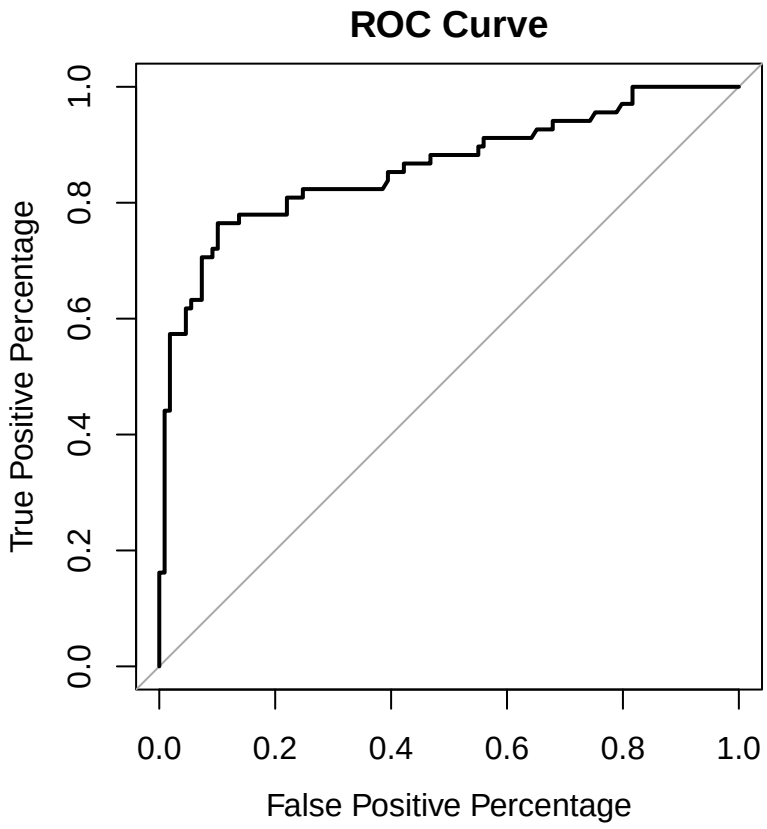
ROC Curve and AUC

```
roc_obj <- roc(test_data$Survived, pred_probs )
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```
par(pty = "s")
plot(roc_obj, main = "ROC Curve", legacy.axes = TRUE, percent=TRUE, xlab = "False Positive Percentage",
```




```
auc_val <- pROC::auc(roc_obj)
cat(sprintf("AUC       : %.3f\n", auc_val))
```

```
## AUC       : 0.862
```

The statistics show an area under the curve of 0.862, signifying that the model is good for predicting the survival of a passenger.

The ROC Curve also shows that a threshold close to 60% to 80% would provide the best tradeoffs between false positives and true positives, since we don't want a lot of false positives, but we also want to be able to identify if a passenger did survive, so that we can support them.

Results Interpretation and discussion

Our logistic regression model was trained to predict the survival of the passengers based on the following predictors: Sex (male or female), Passenger Class (1, 2, or 3), Port of Embarkment (C, S, or Q), Number of siblings/spouses aboar, Number of parents/children aboard, and their Ticket fare.

891 rows of data were used, 80% of which were used to train the model using caret glm, and 20% were used to test the model.

The test showed that the model has an accuracy of 0.8305, meaning that the model largely predicts around the correct value. The model also showed a precision of 0.7794, showing that, 78% of the positive predictions of the model is actually a positive. A recall of 0.7794 also shows that, for 78% of the time, the model is able to predict correctly that a positive event happened, given that the positive event (the passenger surviving), did happen.

The F1 score simply takes this two into account, and thus an F1 score of 0.7794 indicates that the model, 78% of the time, can be trusted to be able to properly predict if a passenger survived.

The AUC score also showed that the model performs well in predicting the survival of the passenger. And,

depending on our risk appetite between true positives and false positives, a threshold of aroun 0.6 to 0.8 is best for the model.