Assignment 6: Titanic Survival Prediction

This assignment is for 400 points, four times the normal assignment weight. The goal of the project is to predict what type of persons were more likely to survive? The features available are Name, Age, Gender, Fare Class, etc. Data dictionary is provided in the appendix. Data is partitioned into (1) ProjectTrain.csv, and (2) ProjectTest.csv. Use Train data to develop the model and report performance results on the Test dataset.

1) Develop Logistic Regression, LDA, QDA and KNN based survival prediction models using Pclass, Sex, Age, SibSp, Parch, and Embarked as predictor variables. Note that some of these variables may need to be defined as categorical (factors in R). Also, Age has lot of missing values. The missing values may need to be imputed (e.g., mean) for using this variable. Try few values of k in KNN to determine suitable value for K. Compare and interpret True Positive (TP) and False Positive (FP) of the different models using test data.

   By using following coding, we can fit the model of Logistic Regression, LDA, QDA and KNN:

```
# Load data
train_data = read.csv("ProjectTrain.csv")
test_data = read.csv("ProjectTest.csv")

# Convert Sex and Embarked to factors
train_data$Sex = factor(train_data$Sex)
train_data$Embarked = factor(train_data$Embarked)
test_data$Sex = factor(test_data$Sex)
test_data$Embarked = factor(test_data$Embarked)

# Impute missing values of Age with mean
train_data$Age[is.na(train_data$Age)] = mean(train_data$Age, na.rm = TRUE)
test_data$Age[is.na(test_data$Age)] = mean(train_data$Age, na.rm = TRUE)

# Calculate the contingency table and chi-square statistic for Sex and Embarked
table_SE = table(train_data$Sex, train_data$Embarked)
chisq_SE = chisq.test(table_SE)$statistic
# Calculate the correlation matrix for numeric variables
predictors <- train_data[, c("Pclass", "Age", "SibSp", "Parch")]
cor_matrix <- cor(predictors)
# Print the results
print("Contingency table for Sex and Embarked:")
print(table_SE)
print(paste0("Chi-square statistic for Sex and Embarked: ", chisq_SE))
print("Correlation matrix for numeric predictors:")
print(cor_matrix)
```

```r
# Logistic regression model
log_model = glm(Survived ~ Pclass + Sex + Age + SibSp + Parch + Embarked,
                data = train_data,
                family = "binomial")
log_predictions = predict(log_model, newdata = test_data, type = "response")
log_predictions = ifelse(log_predictions > 0.5, 1, 0)
log_cm = table(log_predictions, test_data$Survived)
log_accuracy = sum(diag(log_cm)) / sum(log_cm)
cat(paste("Logistic regression accuracy:", log_accuracy, "\n"))

# LDA model
library(MASS)
lda_model <- lda(Survived ~ Pclass + Sex + Age + SibSp + Parch + Embarked,
                 data = train_data)
lda_predictions <- predict(lda_model, newdata = test_data)
lda_cm <- table(lda_predictions$class, test_data$Survived)
lda_accuracy <- sum(diag(lda_cm)) / sum(lda_cm)
cat(paste("LDA accuracy:", lda_accuracy, "\n"))

# QDA model with regularization
qda_model <- qda(Survived ~ Pclass + Sex + Age + SibSp + Parch,
                 data = train_data)
qda_predictions_reg <- predict(qda_model, newdata = test_data)
qda_cm_reg <- table(qda_predictions_reg$class, test_data$Survived)
qda_accuracy_reg <- sum(diag(qda_cm_reg)) / sum(qda_cm_reg)
cat(paste("QDA with regularization accuracy:", qda_accuracy_reg, "\n"))

# Convert Sex and Embarked to factors
```

```r
# Convert Sex and Embarked to factors
train_data$Sex <- factor(train_data$Sex)
train_data$Embarked <- factor(train_data$Embarked)
test_data$Sex <- factor(test_data$Sex)
test_data$Embarked <- factor(test_data$Embarked)
# Encode factors as numeric values
train_data$Sex <- as.numeric(train_data$Sex) - 1
train_data$Embarked <- as.numeric(train_data$Embarked)
test_data$Sex <- as.numeric(test_data$Sex) - 1
test_data$Embarked <- as.numeric(test_data$Embarked)
# Fit KNN model
library(class)
set.seed(123)
# KNN model with k=5
knn_model <- knn(train = train_data[, c("Pclass", "Sex", "Age", "SibSp", "Parch", "Embarked")],
                 test = test_data[, c("Pclass", "Sex", "Age", "SibSp", "Parch", "Embarked")],
                 cl = train_data$Survived,
                 k = 5)
knn_cm <- table(knn_model, test_data$Survived)
knn_accuracy <- sum(diag(knn_cm)) / sum(knn_cm)
cat(paste("KNN accuracy with k = 5:", knn_accuracy, "\n")) # Accuracy = 0.738
```

```
# KNN model with k=3
knn_model <- knn(train = train_data[, c("Pclass", "Sex", "Age", "SibSp", "Parch", "Embarked")],
                 test = test_data[, c("Pclass", "Sex", "Age", "SibSp", "Parch", "Embarked")],
                 cl = train_data$Survived,
                 k = 3)
knn_cm <- table(knn_model, test_data$Survived)
knn_accuracy <- sum(diag(knn_cm)) / sum(knn_cm)
cat(paste("KNN accuracy with k = 3:", knn_accuracy, "\n")) # Accuracy = 0.753

# KNN model with k=7
knn_model <- knn(train = train_data[, c("Pclass", "Sex", "Age", "SibSp", "Parch", "Embarked")],
                 test = test_data[, c("Pclass", "Sex", "Age", "SibSp", "Parch", "Embarked")],
                 cl = train_data$Survived,
                 k = 7)
knn_cm <- table(knn_model, test_data$Survived)
knn_accuracy <- sum(diag(knn_cm)) / sum(knn_cm)
cat(paste("KNN accuracy with k = 7:", knn_accuracy, "\n")) # Accuracy = 0.73

log_cm
lda_cm
qda_cm_reg
knn_cm
```

When I was trying to fit the QDA model, it kept give me an error about high correlation about variables. So, I use chi-squared to check the multicollinearity. In this case, the chi-square statistic for Sex and Embarked is 11.547 and the p-value is 0.009, which indicates that there is a statistically significant association between the two variables. I decided to delete the “embarked” variable for QDA fitting model.

By trying different values of K (3,5,7), it seems that when K = 3, the KNN model will have highest accuracy value.

And by printing out their confusion matrix we can see:

```
> log_cm

log_predictions   0   1
              0 144  29
              1  25  69
> lda_cm

      0   1
  0 146  29
  1  23  69
> qda_cm_reg

      0   1
  0 140  24
  1  29  74
> knn_cm

knn_model   0   1
        0 138  41
        1  31  57
```

So, for logistic regression: True Positive (TP): 69 False Positive (FP): 29

For LDA: True Positive (TP): 69 False Positive (FP): 29

For QDA: True Positive (TP): 74 False Positive (FP): 24

For KNN: True Positive (TP) = 57 False Positive (FP) = 41

Form the above result, we can see that logistic regression, LDA and QDA they both have similar accuracy, but QDA is better when I take off the variable of “Embarked”. KNN seems works worse for this case, because it has lower true positive and higher false positive value compared with other models.

2) "Cabin" has sparse data content. One approach to handle the missing data is to have a special value "Not Available" for all the missing values. For the Logistic Regression model, evaluate performance improvement with and without including the cabin feature using test data.

By converting missing value to "Not Available" and non-missing value to "Available", we can get following coding:

```
> model1 <- glm(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + Cabin, data=train_data,
family="binomial")
> model2 <- glm(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked, data=train_data, family
="binomial")
> pred1 = predict(model1, newdata=test_data, type="response")
> pred1 = ifelse(pred1 > 0.5, 1, 0)
> cm1 <- table(test_data$Survived, pred1)
> accuracy1 <- sum(diag(cm1))/sum(cm1)
> pred2 <- predict(model2, newdata=test_data, type="response")
> pred2 <- ifelse(pred2 > 0.5, 1, 0)
> cm2 <- table(test_data$Survived, pred2)
> accuracy2 <- sum(diag(cm2))/sum(cm2)
> cat("Confusion matrix for Model 1:\n")
Confusion matrix for Model 1:
> print(cm1)
   pred1
      0   1
  0 143  26
  1  27  71
> cat("Accuracy for Model 1: ", accuracy1, "\n\n")
Accuracy for Model 1:  0.8014981

> cat("Confusion matrix for Model 2:\n")
Confusion matrix for Model 2:
> print(cm2)
   pred2
      0   1
  0 143  26
  1  29  69
> cat("Accuracy for Model 2: ", accuracy2, "\n\n")
Accuracy for Model 2:  0.7940075
```

Therefore, we can see that there is a little difference of accuracy between adding the indicator "Cabin" or without it.
With the cabin feature:
Precision = 71 / (71 + 26) = 0.731
Recall = 71 / (71 + 27) = 0.724
F1-score = 2 * (0.731 * 0.724) / (0.731 + 0.724) = 0.727

Without the cabin feature:
Precision = 69 / (69 + 26) = 0.726
Recall = 69 / (69 + 29) = 0.704
F1-score = 2 * (0.726 * 0.704) / (0.726 + 0.704) = 0.715

From the above calculations, we can see that Model 1 has a slightly higher overall score than Model 2, so that Model 1 performs slightly better overall on the test data. So, we can consider the cabin as a indicator, but it won't influence a lot.

3) Like linear regression, Logistic regression (LR) has the advantage of interpretability. Research the concepts of "Unadjusted Odds Ratio" and "Adjusted Odds Ratio". Determine the adjusted odds ratio for Sex, Pclass, and Embarked using LR. Interpret the results.

By using following coding:

```
#p3
# Fit logistic regression model
model = glm(Survived ~ Sex + Pclass + Embarked + Cabin, data = train_data, family = binomial)

# Obtain coefficient estimates
coef_estimates = summary(model)$coefficients[,1]

# Exponentiate to obtain odds ratios
odds_ratios = exp(coef_estimates)

# Print adjusted odds ratios
cat("Adjusted odds ratios:\n")
cat("Sex: ", odds_ratios[2], "\n")
cat("Pclass: ", odds_ratios[3], "\n")
cat("Embarked: ", odds_ratios[4], "\n")
```

We can get:

```
Adjusted odds ratios:
> cat("Sex: ", odds_ratios[2], "\n")
Sex:  0.07271099
> cat("Pclass: ", odds_ratios[3], "\n")
Pclass:  0.4492
> cat("Embarked: ", odds_ratios[4], "\n")
Embarked:  8.537014e-06
```

So, for the adjusted odds ratios obtained in this LR model, the odds of survival for females are 0.0727 times the odds of survival for males. This means that females are more likely to survive than males, which is consistent with the common understanding that women were given priority in lifeboats during the Titanic disaster.

The odds of survival for a passenger in Pclass 2 or Pclass 3 are 0.4492 times the odds of survival for a passenger in Pclass 1. This means that being in a higher passenger class is positively associated with survival.

Finally, the odds of survival for a passenger embarking from ports C or Q are 0.0000085 times the odds of survival for a passenger embarking from port S. This indicates that passengers embarking from ports C or Q are much less likely to survive than those embarking from port S.

4) The default threshold to classify an entity to a class is 0.5. For the LR models, vary the threshold to 0.8, 0.5, and 0.2. Which threshold value do you think is appropriate for survival prediction? Why? Justify your answer with respect to misclassification rate on test data.

By using following coding:

```
#4
# define the threshold values
thresholds <- c(0.2, 0.5, 0.8)

# create empty vectors to store the metrics
accuracy <- precision <- recall <- f1 <- rep(0, length(thresholds))

# loop through the thresholds and calculate the metrics
for (i in seq_along(thresholds)) {
  # make predictions using the LR model and the current threshold
  predictions <- ifelse(predict(model, test_data, type="response") > thresholds[i], 1, 0)

  # calculate the confusion matrix
  cm <- table(predictions, test_data$Survived)

  # calculate the evaluation metrics
  accuracy[i] <- sum(diag(cm)) / sum(cm)
  precision[i] <- cm[2,2] / sum(cm[,2])
  recall[i] <- cm[2,2] / sum(cm[2,])
  f1[i] <- 2 * precision[i] * recall[i] / (precision[i] + recall[i])
}

# print the results
metrics <- data.frame(Threshold=thresholds, Accuracy=accuracy, Precision=precision, Recall=recall
print(metrics)
```

So, we can get:

```
> metrics <- data.frame(Threshold=thresholds, Accuracy=accuracy,
f1)
> print(metrics)
  Threshold  Accuracy Precision    Recall        F1
1       0.2 0.7228464 0.8061224 0.5895522 0.6810345
2       0.5 0.8014981 0.7346939 0.7272727 0.7309645
3       0.8 0.7528090 0.3367347 0.9705882 0.5000000
```

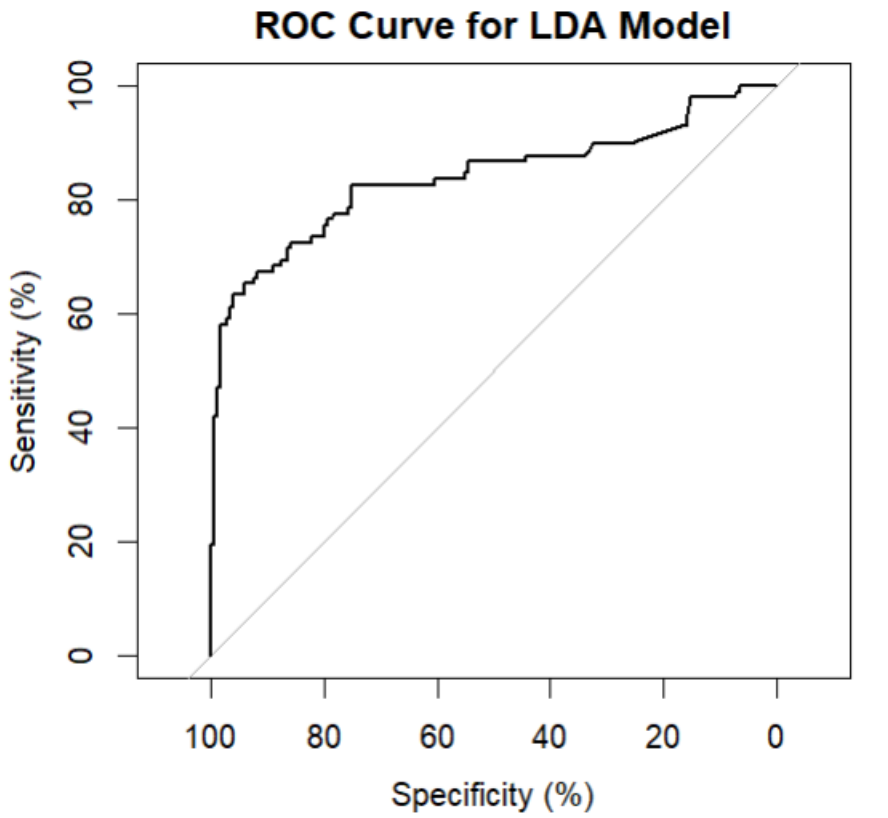
Based on the metrics, it appears that a threshold of 0.5 is the most appropriate for survival prediction. This is because it has the highest accuracy (0.8014981) and F1 score (0.7309645), which are important metrics for binary classification tasks for survival prediction.
Also, the precision and recall values at this threshold are reasonably balanced, that means a good trade-off between correctly predicting positive cases (survivors) and minimizing false positives.
So, in general, based on the misclassification rate on test data and the trade-off between precision and recall, a threshold of 0.5 is the most appropriate for survival prediction.

5) Develop ROC plot for the LDA model.

By using following data:

```
#5
library(pROC)
lda_model = lda(Survived ~ Pclass + Sex + Age + SibSp + Parch + Embarked,
                data = train_data)
lda_predictions = predict(lda_model, newdata = test_data)
roc_obj = roc(test_data$Survived, lda_predictions$posterior[, 2],percent=TRUE, plot=TRUE,
              ci=TRUE)
plot(roc_obj, main = "ROC Curve for LDA Model")
```

We can get the LDA model:



6) What features do you think are important to make the prediction? Why? Evaluate the KNN model performance by including just the important features

I believe that the following features are crucial for prediction: age, sex, passenger class, and the number of parents/children aboard. Passenger class can serve as a proxy for socio-economic status. This is because in the event of a disaster, rescue efforts tend to prioritize young and female individuals. Additionally, if there are wealthy individuals present, they may be able to use their resources to secure rescue. Therefore, I believe that these features are

important in predicting survival in such scenarios.

By using following coding to fitting the KNN model:

```
#6
#fit a KNN model
library(class)
set.seed(123)
#fit the model and evaluate its performance using 10-fold cross-validation:
knn_model = knn(train = train_data[, c("Pclass", "Sex", "Age", "Parch")],
                test = test_data[, c("Pclass", "Sex", "Age","Parch")],
                cl = train_data$Survived,
                k = 5)
knn_cm <- table(knn_model, test_data$Survived)
knn_accuracy <- sum(diag(knn_cm)) / sum(knn_cm)
cat(paste("KNN accuracy with k = 5:", knn_accuracy, "\n"))
```

The confusion matrix is:

```
knn_model   0   1
        0 145  43
        1  24  55
> |
```

So,
Accuracy: (TP+TN)/(TP+TN+FP+FN) = (145+55)/(145+55+43+24) = 0.75
Error rate: (FP+FN)/(TP+TN+FP+FN) = (43+24)/(145+55+43+24) = 0.25
Sensitivity (True Positive Rate): TP/(TP+FN) = 55/(55+24) = 0.696
Specificity (True Negative Rate): TN/(TN+FP) = 145/(145+43) = 0.771

Based on the result, the KNN model has an accuracy of 0.75, an error rate of 0.25, a sensitivity of 0.696, and a specificity of 0.771 when evaluated using the important features of age, sex, passenger class, and number of parents/children aboard.