Final Report - Group 20

Hoon Kim, Ricky Cai, Ricky Yavits, Sherry Fan

Aim and Background

In Australia, the current allocation of kidneys is coordinated through a computer software known as Organ Match which allocates kidneys based on rules defined by the National Renal Transplant Advisory Committee. Organ Match provides clear user friendly reports of donors and recipients with additional information on Donor Specific Antibodies and HLA matches for clinicians and transplant coordinators.(TSANZ, 2019) However the transplant recipient is also an important party in the transplantation process. To best share this technology with the patients, so that they can make the best decisions for their treatment and kidney transplantation, the KTRA aims at aiding in the management of their illness with innovations on the HLA matching selection criteria.

Methods - Data Collection and Developed Models

The main additional data source that was used in the project was the inclusion of the ElliPro score. The following section details the approach we took to integrate the ElliPro score into our project.

The ElliPro Score (Data Collection)

The Organ Match software utilises HLA matching for donor recipient kidney pairing. The issue is however there are numerous examples in which the donor and recipient differ only by a single amino-acid difference and yet the recipient developed de-novo DSA. (Tambur, 2018). Hence eplet matching has become a trending topic in Organ transplantation. But even with eplet matching, different eplets induce different levels of immune reactions. This is observed experimentally where two recipient pairs with the same number of eplet matches exhibit different immune responses.

Due to this phenomenon we incorporate the ElliPro score, a score describing the immunogenicity of the Eplet, into our models.(Duquesnoy & Marrari, 2017) The ElliPro score was taken from the HLA epitope registry from the Federal University of Paiui in Brazil. It must be noted that the HLA epitope registry is an intellectual community resource thus information on each eplet is consistently being updated.

The Ellipro Scores were scraped from the HLA epitope registry which organsed the eplets in terms of HLA loci. These HLA loci included ABC, DRB1/3/4/5, DQB + DQA, DPB + DPA, Interlocus Cl. II and MICA. These raw scores which were categorical were then quantified as 1,2,3 and 4 in ascending order of immunogenicity levels. The code below shows the processes of constructing data frames for each HLA loci and row binding to create a final dictionary dataframe. The code describing the data collection is included in the appendix (Appendix - Extracting ElliPro).

Now Dr Wong's dataset contains eplet mismatch information in which mismatches between donor and recipient in each sample is detonated as a 1 and a match as 0. Hence this dataset was modified by first identifying class 1 and class 2 eplets. Afterwards, the total Ellipro score for class 1 and class 2 eplets were calculated by multiplying each mismatched eplet by it's corresponding Ellipro score. Two columns were then incorporated into the dataset denoting the total Class 1 and 2 Ellipro score for each sample. The code to merge the ElliPro scores with the original data provided by Dr Wong is included in the appendix (Appendix - Merging ElliPro)

Introducing the KTRA

The KTRA has three main features. In the pre and post transplantation stage, patients can use the calendar feature to check their appointments as well as their medicine and dialysis schedule.In the decision-making stage, patients can compare between themselves and their donor. KTRA also has a risk factors report, which shows the prediction results of transplantation risk.

This risk report gives 6 seperate scores. The first set of scores inform the recipient of their risk level of developing Denovo DSA as well as the current presence of Preformed DSA, which is included in Dr Wong's dataset. The second set of scores show the risk of developing Antibody Mediated Rejection as well as the joint impact of Gender Mismatch and Kidney Size as well as Cold Ischemia time on graft survivability.

Cox Proportional Hazards Model

Within the KTRA, the Cox Proportional Hazards model was used in three of the risk scores (DSA Class 1, DSA Class 2 and AMR Class 2). The Hazard function measures how specific covariates influence the risk of events happening at time t. In the Cox model, we are interested in the relative risk of observed covariates to the mean of the training covariates, which is the Hazard ratio.

Methods - Evaluation Strategies

The 3 Cox models that were generated were evaluated on the significance of the variables and its adherence to the proportional hazards and homoscedasticity linearity assumptions. Lastly the Out of bag C-index performance of the models were evaluated.

Firstly variable significance was tested using Wald's test, Likelihood ratio test and Logrank test, in which all models pointed to significant variables. The linear assumption is the requirement that the continuous variable need to be in a linear correlation with the observed times of events. Here all 3 models passed this assumption but there was some non linearity showing.

The proportional hazards assumption is the most important assumption for Cox regression. This assumption requires the model to have the ratio of the hazards between the training data and the mean of the training data to be constant over time.All models adhered to the proportional hazard assumption The homoscedasticity assumption was checked based on residual deviance. For the model to be considered a good fit these residuals should be roughly symmetrically distributed about zero with a standard deviation of 1. However for all three models this assumption was not satisfied. This means that the models are not reliable in the prediction of large hazard ratio values.

On top of trying the basic logging, square rooting of the covariate to no avail, the application of robust regression methods were considered, however the literature on robust Cox proportional hazard methods are still a big area of research and thus a conservative approach was taken. The index of concordance is a "global" index for validating the predictive ability of a survival model. More specifically it gives the probability a randomly selected recipient who experienced an event (such as developing Denovo DSAs) had a higher risk score than a patient who had not experienced the event. A model is considered "good" with a C-index of 0.7. Despite our DSA models only slightly missing out on this threshold, our AMR model breezed past this threshold with ease.

Results - Final Model

Donor Specific Antibodies Risk Scores (DSA Risk Scores)

The presence of DSA or Donor Specific Antibodies are a major risk in kidney rejection.(Lionaki et al., 2013) While there is an association between epitope mismatching and DSA (Wong et al, 2020), our cox proportional hazards models use the ElliPro scores as the only variable. The two Class 1 and Class2 Donor Specific Antibodies models use Class 1 and Class 2 ElliPro scores respectively. The figure below shows how the level of risk for DSA Class 1 and Class 2 is communicated to a patient in the KTRA app.

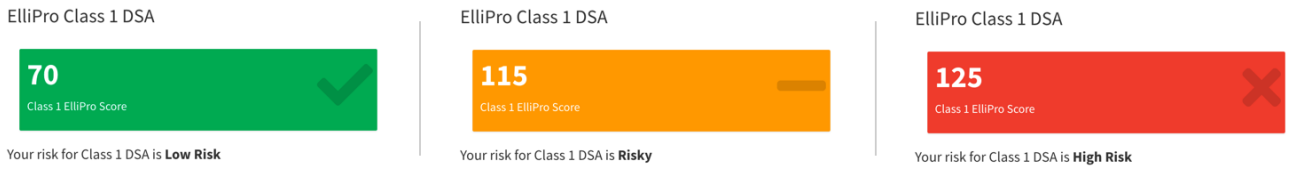

Figure 1: Example of Low, Risky and High Risk for DSA (Class 1 and 2) in KTRA App

As discussed earlier in evaluation strategies, a C index of 0.7 is considered good which helped guide which variables we used in the final product. To evaluate these cox proportional hazards models, we used 5 cross validation with 25 repetitions. The following figure shows the C index for the DSA models over 25 repetitions which shows the median C index for both models above 0.6.

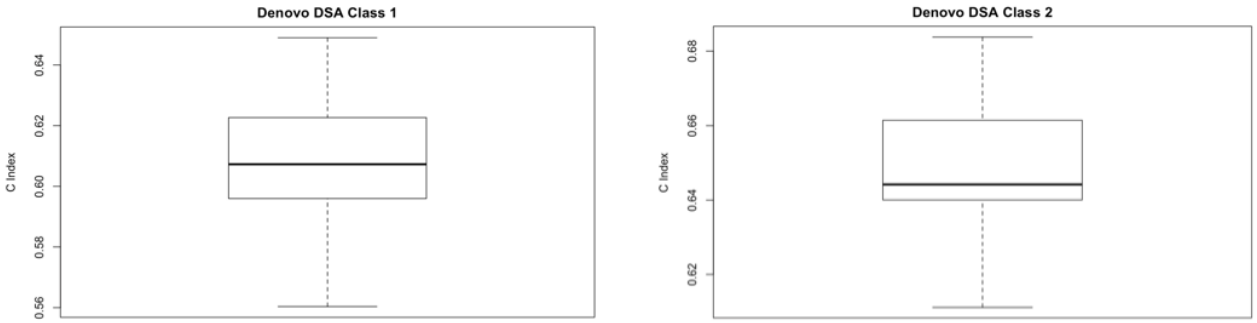

Figure 2: DSA Class 1 and 2 evaluations with median C index around 0.61 and 0.64 respectively

For details on how these models were created, see appendix (Appendix - DSA Models)

Donor Specific Antibodies Preformed Risk Score

The app also features a preformed DSA score where the risk score is classified as high risk if present and low risk if not present as seen in the figure below.

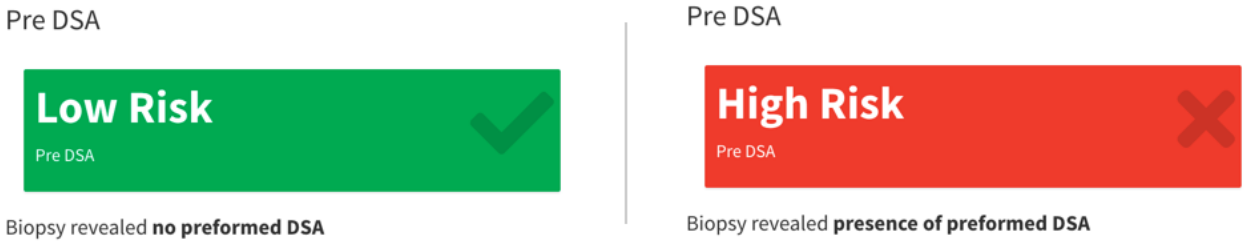

Figure 3: Example of Low risk of pre DSA (Left) and High risk of pre DSA (Right) in KTRA App

Antibody Mediated Rejection Risk Score (AMR Risk Score)

Antibody Mediated Rejection or AMR covers all allograft rejection caused by antibodies and is recognised as the major cause of allograft failure. (Singh et al., 2009) Research shows that de novo DSA developing class 2 eplet mismatches is more prominent so only the Class 2 ElliPro score is used to model the AMR risk score using a coxph model. (Stegal, et al, 2012). For details on how the AMR model was created, see appendix (Appendix - AMR Model)

The following figure shows the C index for the AMR model over 25 repetitions which shows the median C index above 0.7.

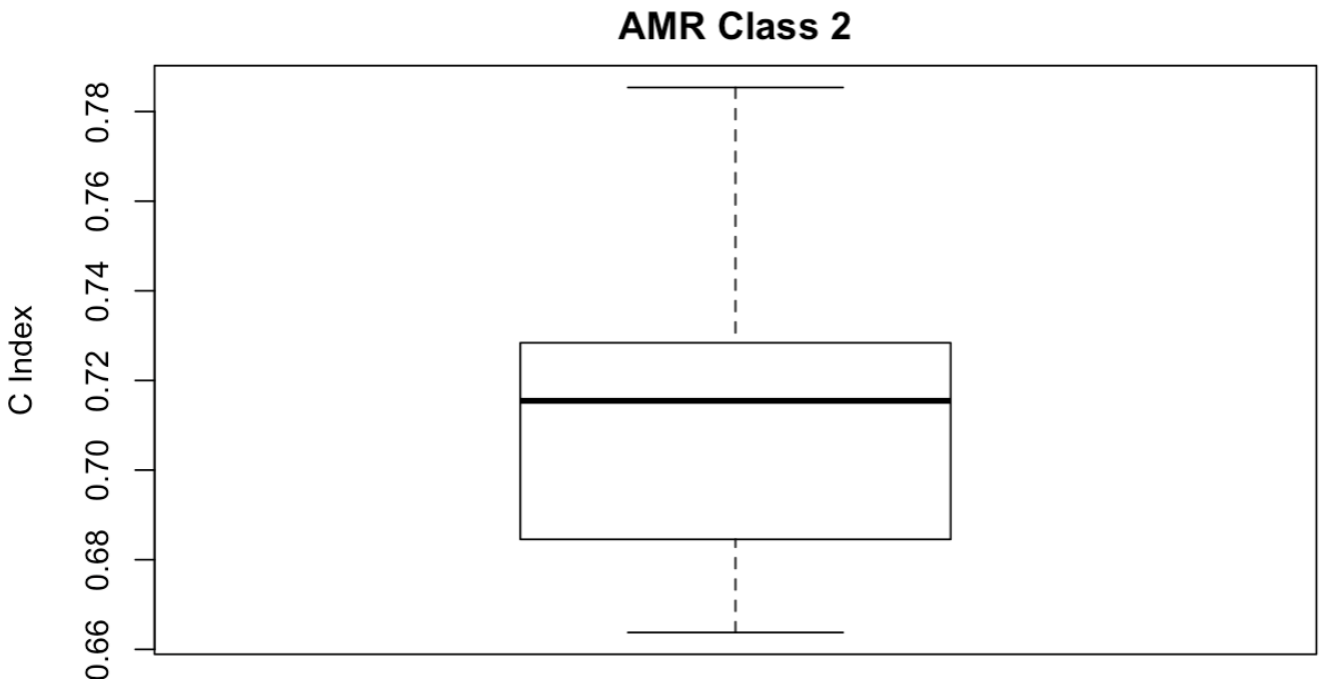

Figure 4: AMR model using total ElliPro class 2 score with median C index around 0.61 and 0.64 respectively

Gender + BMI (Kidney Size) Risk Score

There is research that proposes that although graft failure risk is higher when donor and recipient genders are different, this risk is mitigated when the donor has a larger kidney. (McGee et al, 2020) Bmi was used to represent kidney size and this was justified as in the research there was a significant positive correlation between BMI and kidney size with a p value threshold of 0.05. (McGee et al, 2020) The combined score is calculated by the following equation.

$BMI \times GenderMM$

Where BMI is scored as 2 when the recipients BMI is greater, 1 otherwise and GenderMM is scored as 2 when there is a mismatch of gender and 1 for a match. If the resulting output is 4, the risk level is classified as high risk, risky if 2 and low risk if 1. A major limitation of this scoring method is that the research only provided evidence for the mitigation of higher risk when it is a male donor and female recipient.

Cold Ischemia Time Risk Score

The final risk score is concerned with the relationship of cold ischemia time and graft failure. Cold ischemia time is defined as the time in between transplants. A Hazard Ratio model deduced from the data provided by an article from the International society of Nephrology, is used to calculate the proportional association of higher cold ischemia time and risk of graft failure. (Debout et al., 2015)The equation is given by

$HR = \frac{1}{70} * Cold\ Ischemia\ Time\ (hours) + \frac{13}{14}$

The limitation of the model, according to the paper, is that it underestimates the hazard ratio. This limitation stems from graft failure being defined as patient death. This means that patients who die after returning to dialysis should be counted. However as the French Organ Matching Network does not follow up on kidney patients who return to dialysis, there is undercounting in the data, thus underestimating the hazard ratio.

Results - Deployment

For the final deployment of the project, we produced a shiny app named "KTRA" or Kidney Transplantation Recipient Aid. The code is hosted on Github and can be run with the following lines of code in R:

```
library(shiny)

# Run KTRA App
runGitHub("KTRA", "iRick92")
```

As previously mentioned, the KTRA incorporates three main features. The first feature compares the compatibility between a kidney recipient and a kidney donor. The second feature shows a risk report outlining 6 risk factors that may influence the success of a kidney transplant. The final feature is a calendar that provides the patient with a daily schedule outlining the clinical appointments, dialysis sessions and medication schedule for the day. As a patient moves from along the stages of kidney transplantation, more features of the KTRA are made available. When a patient is on the waiting list for a new kidney, they are issued login details to allow them to log into the KTRA.

For a patient in the post transplantation stage, the KTRA app provides a summary about themselves including age, location, blood type, BMI, gender and their kidney match status. The patient is also able to check the calendar on upcoming appointments, dialysis sessions and medication schedules. The figure below shows this important information on one screen.

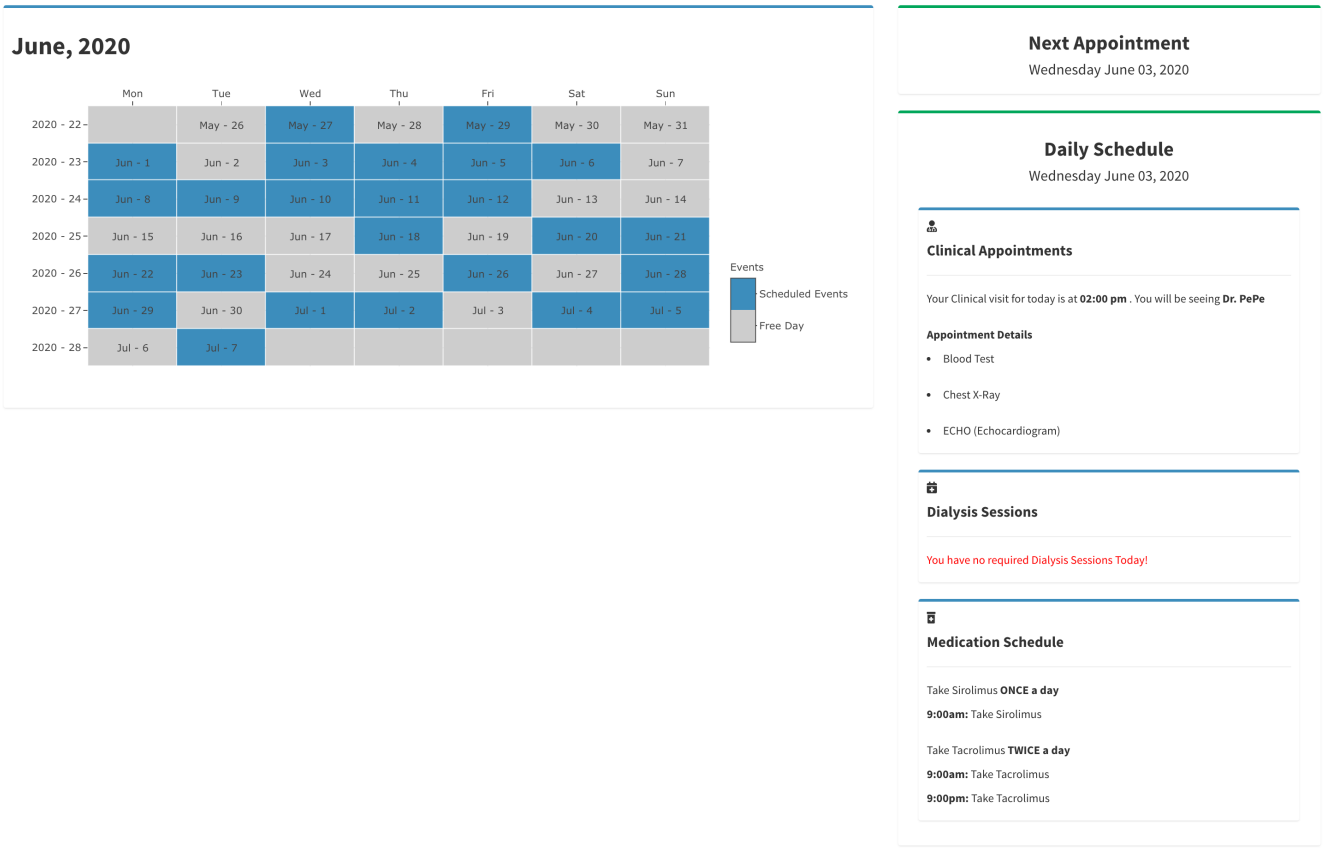

Figure 5: The calendar tab for a patient in the pre-transplantation stage. This includes their pre transplant test and medicine schedule

Once a kidney is matched to a recipient, the KTRA app is automatically updated. The patient is then able to see a comparison between themselves and the kidney donor through the compatibility tab. The patient is also able to view a comprehensive risk report which provides the risks that contribute to a successful transplant. The following figures below show a patient with a matched kidney what risks are associated with kidney transplantation to help them make an important decision - to accept or reject the kidney.

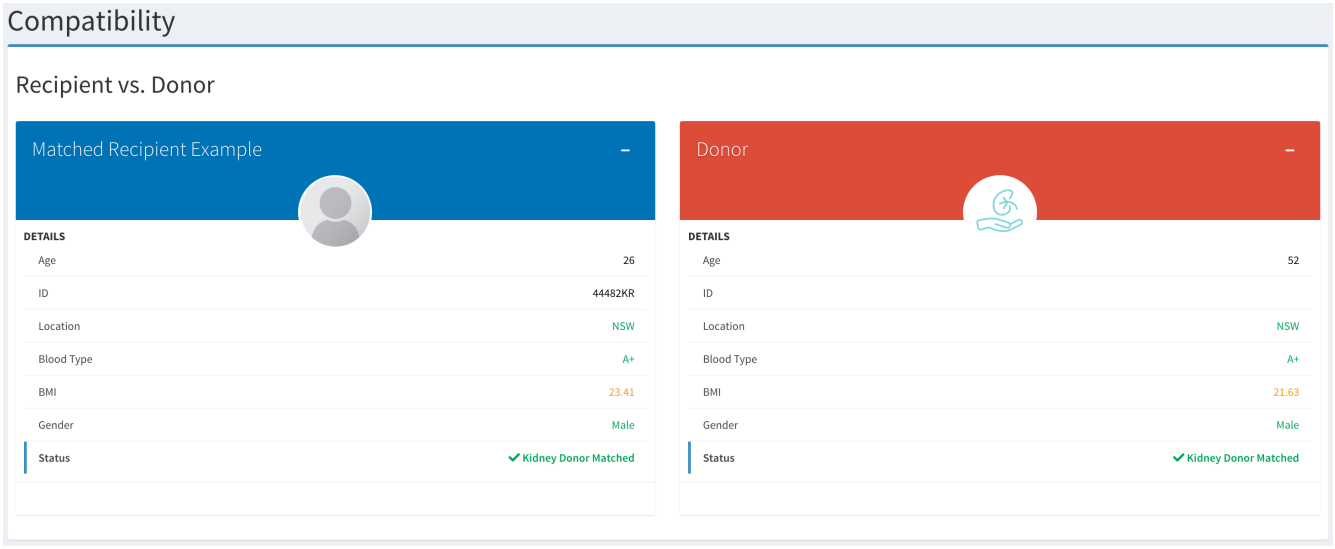

Figure 6: The compatibility tab for a patient with a matched kidney comparing kidney donor and recipient

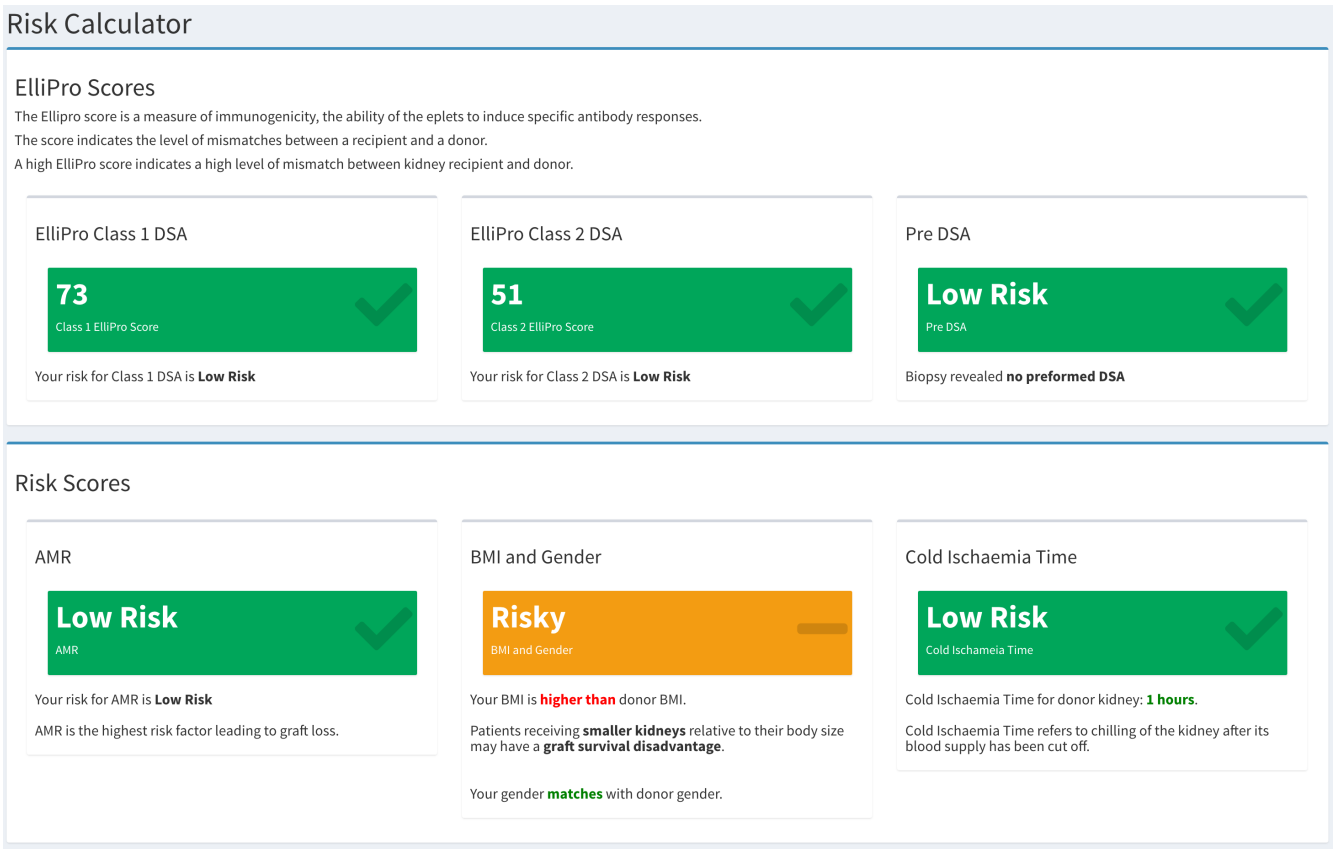

Figure 7: The risk calculator tab for a patient with a matched kidney showing the severity of 6 risks and an explanation

When a patient finally receives a kidney, the main feature of the KTRA they will use is the calendar. Kidney transplant recipients are required to keep up with clinical visits involving physical therapy and frequent blood tests. This is what the calendar feature of the KTRA aims to assist the patient with. The figure below shows what upcoming scheduled appointments look like.

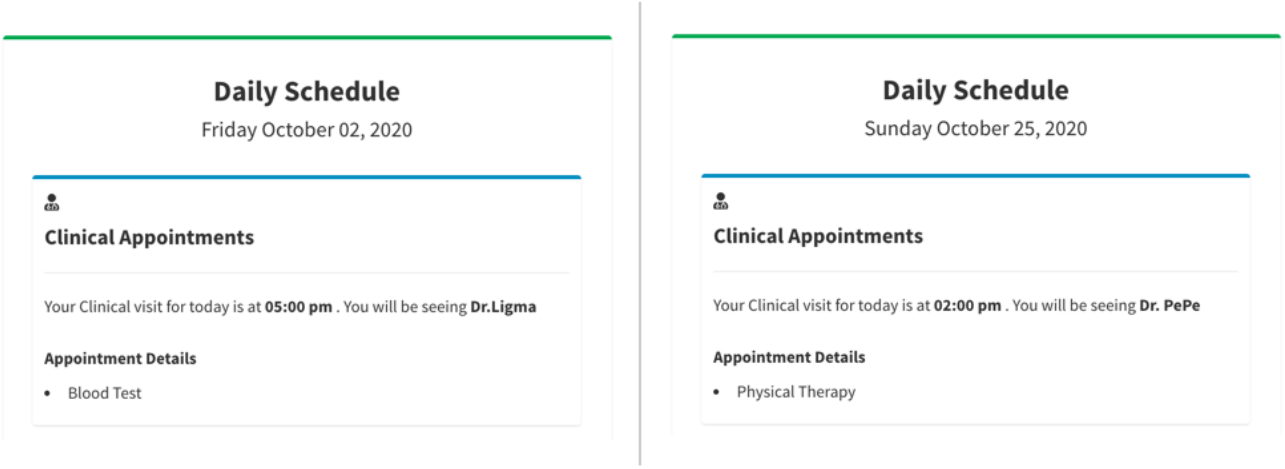

Figure 8: The calendar tab for a post transplant patient showing the scheduled appointments

Discussion and Conclusion

In conclusion The KTRA helps patients in the management of their illness and their kidney transplant by firstly aiding in the management of Chronic Kidney Disease pre and post transplant and secondly by providing a comprehensive risk report which the recipient can incorporate in his or her decision making process. Some future development for the KTRA include a comprehensive input system for the Nephrologist to upload the data and manage their assigned patients, multi discipline integration since CKD is usually associated with another illness, integration of pre transplant biopsy results, so patients can identify the risk of developing cellular rejection and further research on eplet matching.

Additionally, as there is no study shows a clear causation between ElliPro score and kidney rejection, using ElliPro score instead of counting the number of eplets mismatch is an innovation in our project, as it improves the understanding of immunological risk from different eplets. However, as our ElliPro score dictionary does not cover all mismatched eplets of the patients, there could be some information missing in our modelling. Additionally, although the result of the models with ElliPro score suggested significant coefficients and acceptable C-index, the deviance residuals of these models are not symmetric around 0. Future work on robust Cox regression models would be recommended to solve the outliers problem.

Student Contributions

Hoon Kim

For this final project I was able to procure the Ellipro Scores by web scraping. Afterwards I assisted each one in the processing of the data and developing of the models. After review of these observations made by these models, I assisted with the calendar feature in the final application. Finally I gave my insight in fine tuning the final product.

Ricky Cai

My contributions for this project include assisting in processing and merging ElliPro scores with the original data, developing random forest survival models (which were not used for the final product), developing the coxph models, user interface design for the KTRA shiny app, development of the KTRA app, keeping meeting minutes and report writing.

Ricky Yavits

In this project, I firstly assisted in model selection by creating a model for testing. Further, I helped with the design of the shiny app by providing a draft and critique of other designs. I also reviewed the models' performance together with my group members. Finally, I made the script for the video, recorded the video with the help of another group member.

Sherry Fan

In this project, I firstly assisted in processing ElliPro scores and converting eplets mismatch to ElliPro score. And then I tested coxph models with different covariates and reviewed the models' performance together with my group members. And finally, I constructed presentation, made presentation slides with my group members' help, and assisted in report writing.

Appendix

Libraries

```
library(tidyverse)
library(class)
library(cvTools)
library(e1071)
library(caret)
library(survival)
library(survminer)
library(Hmisc)
library(knitr)
library(htmltab)
library(textreadr)
library(rvest)
```

Extracting ElliPro

The following code describes the process in which the ElliPro scores were extracted from the HLA epitope registry.

```
## Get HLA Class 1 Eplets

url <- "https://www.epregistry.com.br/index/databases/database/ABC/"
# first half of the table i just imported the table utilising the htmltab library and function

eplet_abc_pt1 <- htmltab(doc = url)[,1:2]
colnames(eplet_abc_pt1)[2] <- "ElliPro.Score" # adjust column name for future concatenation

# second half - had to go through and download the html file and extract the table manually

test <- read_html("second_hla_abc_table.html") %>%
  html_nodes("tr")

eplet_name = NA
ellipro_score = NA
for (i in 2:length(test)){
  if (length(str_split(test[i], "td")[[1]]) > 12){
```

```
    pattern = "[:space:][:alnum:]+[:space:]"
    ept <-  str_split(test[i], "td")[[1]][2]
    eplet_name[i] <- (str_extract(ept,pattern))

    pattern=">(.*?)<"
    elipro <- str_split(test[i], "td")[[1]][4]
    ellipro_score[i] <- (regmatches(elipro,regexec(pattern,elipro))[[1]][2])
  }else{
    eplet_name[i] <- NA
    ellipro_score[i] <- NA
  }
}
eplet_name[50] <- "163LS/G"# regex didnt catch the name of the eplet because of the forward slash
eplet_abc_pt2 <- na.omit(data.frame('Eplet' = trimws(eplet_name),
           'ElliPro.Score' = ellipro_score))
```

```
eplet_abc_pt1 <- eplet_abc_pt1 %>% #convert to numeric score
  mutate(ElliPro.Score =
           case_when(
             ElliPro.Score == 'High' ~ 4,
             ElliPro.Score == 'Intermediate' ~ 3,
             ElliPro.Score == 'Low' ~ 2,
             ElliPro.Score == 'Very Low' ~ 1,
         ))
eplet_abc_pt1$Eplet = paste('X', eplet_abc_pt1$Eplet, sep='') #convert name starting with X


eplet_abc_pt2 <- eplet_abc_pt2 %>% #convert to numeric score
  mutate(ElliPro.Score =
           case_when(
             ElliPro.Score == 'High' ~ 4,
             ElliPro.Score == 'Intermediate' ~ 3,
             ElliPro.Score == 'Low' ~ 2,
             ElliPro.Score == 'Very Low' ~ 1,
         ))
eplet_abc_pt2$Eplet = paste('X', eplet_abc_pt2$Eplet, sep='') #convert name starting with X
```

```
## Get HLA Class 2 Eplets

url <- "https://www.epregistry.com.br/index/databases/database/DRB/"
# first half of the table i just imported the table utilising the htmltab library and function

eplet_drb <- htmltab(doc = url)[,1:2]
colnames(eplet_drb)[2] <- "ElliPro.Score"
eplet_drb <- eplet_drb %>% #convert to numeric score
  mutate(ElliPro.Score =
           case_when(
             ElliPro.Score == 'High' ~ 4,
             ElliPro.Score == 'Intermediate' ~ 3,
             ElliPro.Score == 'Low' ~ 2,
             ElliPro.Score == 'Very Low' ~ 1,
         ))
eplet_drb$Eplet = paste('X', eplet_drb$Eplet, sep='') #convert name starting with X
```

```
# DQB + DQA

test <- read_html("eplet_dq.html") %>%
  html_nodes("tr")

eplet_name = NA
ellipro_score = NA
for (i in 2:length(test)){
  if (length(str_split(test[i], "td")[[1]]) > 12){
    pattern = "[:space:][:alnum:]+[:space:]"
    ept <-  str_split(test[i], "td")[[1]][2]
    eplet_name[i] <- (str_extract(ept,pattern))

    pattern=">(.*?)<"
    elipro <- str_split(test[i], "td")[[1]][4]
    ellipro_score[i] <- (regmatches(elipro,regexec(pattern,elipro))[[1]][2])
  }else{
```

```
    eplet_name[i] <- NA
    ellipro_score[i] <- NA
  }
}

eplet_dq <- na.omit(data.frame('Eplet' = trimws(eplet_name),
           'ElliPro.Score' = ellipro_score))[-22,]

eplet_dq <- eplet_dq %>% #convert to numeric score
  mutate(ElliPro.Score =
           case_when(
             ElliPro.Score == 'High' ~ 4,
             ElliPro.Score == 'Intermediate' ~ 3,
             ElliPro.Score == 'Low' ~ 2,
             ElliPro.Score == 'Very Low' ~ 1,
         ))
eplet_dq$Eplet = paste('X', eplet_dq$Eplet, sep='') #convert name starting with X
```

```
# DQB + DPA
url <- "https://www.epregistry.com.br/index/databases/database/DP/"
# first half of the table i just imported the table utilising the htmltab library and function

eplet_dpb_dpa <- htmltab(doc = url)[,1:2]
colnames(eplet_dpb_dpa)[2] <- "ElliPro.Score"

eplet_dpb_dpa <- eplet_dpb_dpa %>% #convert to numeric score
  mutate('ElliPro.Score' =
           case_when(
             ElliPro.Score == 'High' ~ 4,
             ElliPro.Score == 'Intermediate' ~ 3,
             ElliPro.Score == 'Low' ~ 2,
             ElliPro.Score == 'Very Low' ~ 1,
         ))
eplet_dpb_dpa$Eplet = paste('X', eplet_dpb_dpa$Eplet, sep='') #convert name starting with X
```

```
# MICA

url <- "https://www.epregistry.com.br/index/databases/database/MICA/"
# first half of the table i just imported the table utilising the htmltab library and function

eplet_MICA <- htmltab(doc = url)[,1:2]

rid_rows <- which(eplet_MICA$`ElliPro Score` == 'Â')

eplet_MICA <- eplet_MICA[-rid_rows,][,1:2]

colnames(eplet_MICA)[2] <- "ElliPro.Score"

eplet_MICA <- eplet_MICA %>% #convert to numeric score
  mutate('ElliPro.Score' =
           case_when(
             ElliPro.Score == 'High' ~ 4,
             ElliPro.Score == 'Intermediate' ~ 3,
             ElliPro.Score == 'Low' ~ 2,
             ElliPro.Score == 'Very Low' ~ 1,
         ))
eplet_MICA$Eplet = paste('X', eplet_MICA$Eplet, sep='') #convert name starting with X
```

```
# Concatentate the two tables together and export as CSV

eplet_final <- rbind(eplet_abc_pt1,eplet_abc_pt2,eplet_dpb_dpa,eplet_dq,eplet_drb,eplet_MICA)
write.csv(eplet_final,"eplet_final.csv", row.names = FALSE)
```

Merging ElliPro

The following code merges the ElliPro scores with the original data set. A class 1 and class 2 total ElliPro score is merged as a result.

```
# Load Data

# Dr Wong original data
```

```
eplets_data <- read.csv("Merged_data_250719_final.csv")

# ElliPro Scores extracted
eplet_to_score_all <- read.csv("eplet_final.csv")
# Convert Eplet to string
eplet_to_score_all$Eplet = as.character(eplet_to_score_all$Eplet)
```

```
# Extract Class 1 and 2 Eplets


# Get index of Class 1 Eplets
X17RS <- which(colnames(eplets_data) == "X17RS")  # the column number corresponding to X17RS
X275G <- which(colnames(eplets_data) == "X275G")  # the column number corresponding to X275G

# Get index of Class 2 Eplets
X4Q <- which(colnames(eplets_data) == "X4Q")  # the column number corresponding to X4Q
X199T <- which(colnames(eplets_data) == "X199T")  # the column number corresponding to X199T

# Select Class 1 and Class 2 Eplets
class1 = subset(eplets_data, select=X17RS:X275G)
class2 = subset(eplets_data, select=X4Q:X199T)

# Select all eplet data  (Class 1 and 2) in one data frame
all_eplets = subset(eplets_data, select=X17RS:X199T)
```

```
# Convert Eplet Score to Dictionary

eplet_to_score_dict = vector(mode="list", length=nrow(eplet_to_score_all))
names(eplet_to_score_dict) = c(eplet_to_score_all$Eplet)

# Assign cores for each eplet in dictionary
for (i in 1:nrow(eplet_to_score_all)) {
 eplet_to_score_dict[[i]] = eplet_to_score_all$ElliPro.Score[i]
}

# Usage:
# eg. eplet_to_score_dict$X4Q or eplet_to_score_dict[["X4Q"]] returns 4 (Eplet mismatch for X4Q equals 4)
```

```
# Check if eplets match

count = 0
# Prints eplets that are not in the epitope registry
for (eplet in colnames(all_eplets)) {
  if (is.null(eplet_to_score_dict[[eplet]])) {
    count = count + 1
  }
}
count/ncol(all_eplets)
colnames(all_eplets)[2]
all_eplets[,2]
```

```
# Calculate Class 1 ElliPro totals

for (i in 1:ncol(class1)) { # for each column
  eplet = colnames(class1)[i] # grab the column name
  new_scores = NA # initialize a column replacement carrier
  eplet_column = class1[,i]   # set the eplet_column variable as all the values in the column
  if (!is.null(eplet_to_score_dict[[eplet]])) { # if the eplet from the lab dataset is in the dictionary
    for (k in 1:length(eplet_column)){  # for each row in that column
      if (eplet_column[k] > 0){  # if the entry in that row is greater than 0
        new_scores[k] <- eplet_to_score_dict[[eplet]] # replace it with the dictionary valu
      } else { # else
        new_scores[k] <- 0 #set it as 0
      }
    }
  } else { # if the eplet is not in the dictionary
    for (k in 1:length(eplet_column)){ # for each value in the column
      new_scores[k] <- 0 # set the value as zero
    }
  }
  class1[,i] <- new_scores # transfer the replacement column to the main column
}
```

```r
class1 <- class1 %>% # create column and fill with rowsum
  mutate(total_score = rowSums(.))

class1$total_score
```

```r
# Calculate Class 2 ElliPro totals

for (i in 1:ncol(class2)) { # for each column
  eplet = colnames(class2)[i] # grab the column name
  new_scores = NA # initialize a column replacement carrier
  eplet_column = class2[,i]   # set the eplet_column variable as all the values in the column
  if (!is.null(eplet_to_score_dict[[eplet]])) { # if the eplet from the lab dataset is in the dictionary
    for (k in 1:length(eplet_column)){  # for each row in that column
      if (eplet_column[k] > 0){  # if the entry in that row is greater than 0
        new_scores[k] <- eplet_to_score_dict[[eplet]] # replace it with the dictionary valu
      } else { # else
        new_scores[k] <- 0 #set it as 0
      }
    }
  } else { # if the eplet is not in the dictionary
    for (k in 1:length(eplet_column)){ # for each value in the column
      new_scores[k] <- 0 # set the value as zero
    }
  }
  class2[,i] <- new_scores # transfer the replacement column to the main column
}
class2 <- class2 %>% # create column and fill with rowsum
  mutate(total_score = rowSums(.))

class2$total_score
```

```r
# Merge and save to CSV

eplets_data <- cbind(subset(eplets_data, select=-c(X17RS:X199T)),class1$total_score)
eplets_data <- cbind(eplets_data,class2$total_score)
colnames(eplets_data)[87] <- 'class1.total_score'
colnames(eplets_data)[88] <- 'class2.total_score'
write.csv(eplets_data,"final_epletdata_sepclass.csv", row.names = FALSE)

read.csv("final_epletdata_sepclass.csv")
```

DSA Models

The following code creates DSA Class 1 and Class 2 models using coxph and the Class 1 and Class 2 total ElliPro Scores

```r
# Data preparation

# Load merged data
df <-read.csv("final_epletdata_sepclass.csv")
df <- na.omit(df)

df_dsa1 <- df %>%
  select(class1.total_score)
df_dsa1 <- data.frame(df_dsa1)
colzero <- which(colSums(df_dsa1)==0)
if (length(colzero)> 0){
  df_dsa1 <-df_dsa1[,-which(colSums(df_dsa1)==0)] #get rid of those always have 0
}

df_dsa1 <- cbind(df$C1daystodnDSA,df$C1dnDSA,df_dsa1)
df_dsa1 <- rename(df_dsa1, C1daystodnDSA = 'df$C1daystodnDSA',C1dnDSA = 'df$C1dnDSA')

df_dsa2 <- df %>%
  select(class2.total_score)
df_dsa2 <- data.frame(df_dsa2)
colzero <- which(colSums(df_dsa2)==0)
if (length(colzero)> 0){
  df_dsa2 <-df_dsa2[,-which(colSums(df_dsa2)==0)] #get rid of those always have 0
}
```

```r
df_dsa2 <- cbind(df$C2daystodnDSA,df$C2dnDSA,df_dsa2)
df_dsa2 <- rename(df_dsa2, C2daystodnDSA = 'df$C2daystodnDSA',C2dnDSA = 'df$C2dnDSA')
```

```r
# C1 and C2 DSA Model training
## Both models were trained utilising 5 fold cross validation with 25 repetitions.

X <- df_dsa1
cvK = 5  # number of CV folds
n_sim = 25
cv.means_C1 = cv_eval = NA
for(i in 1:n_sim){
  cvSets = cvTools::cvFolds(nrow(X), cvK)  # permute all the data, into 5 folds
  cv_eval = NA  # initialise results vector
  for (j in 1:cvK) {
    test_id = cvSets$subsets[cvSets$which == j]
    X_test = X[test_id, ]
    X_train = X[-test_id, ]

    mod1 <- coxph(Surv(C1daystodnDSA/365, C1dnDSA ) ~  class1.total_score , data = X_train)
    cv_eval[j] <- concordance(mod1, newdata = X_test)$concordance
  }
  cv.means_C1[i] <- mean(cv_eval)
}

#PH assumption test
mod1zph <- cox.zph(mod1)$table


X <- df_dsa2
cvK = 5  # number of CV folds
n_sim = 25
cv.means_C2 = cv_eval = NA
for(i in 1:n_sim){
  cvSets = cvTools::cvFolds(nrow(X), cvK)  # permute all the data, into 5 folds
  cv_eval = NA  # initialise results vector
  for (j in 1:cvK) {
    test_id = cvSets$subsets[cvSets$which == j]
    X_test = X[test_id, ]
    X_train = X[-test_id, ]

    mod2 <- coxph(Surv(C2daystodnDSA/365, C2dnDSA ) ~  class2.total_score , data = X_train)
    cv_eval[j] <- concordance(mod2, newdata = X_test)$concordance
  }
  cv.means_C2[i] <- mean(cv_eval)
}

#PH assumption test
mod2zph <- cox.zph(mod2)$table
```

```r
## Summary of the DSA C1 model

#The following output shows the model summary for DSA and AMR models.
#It can be seen from the summaries that the Ellipro score is a significant variable in the model. This is further justified by the li

summary(mod1)
```

```r
## Summary of the DSA C2 model

#The following output shows the model summary for DSA and AMR models.
#It can be seen from the summaries that the Ellipro score is a significant variable in the model. This is further justified by the li

summary(mod2)
```

```r
# Proportional Hazard Assumption of DSA C1 model

# The proportional hazards assumption is the most important assumption for Cox regression. The tests below for each model checks the

mod1zph
```

```r
# Proportional Hazard Assumption of DSA C2 model

# The proportional hazards assumption is the most important assumption for Cox regression. The tests below for each model checks the
```

```
mod2zph
```

```
# Checking for DSA C1 model homosckedasticity

# To ensure the coxph model assumptions are met, we need to check for homoscedasticity.
# The following plots show the deviance residuals for each model. For the assumption to be met, the residuals should be symmetric aro

ggcoxdiagnostics(mod1, type = "deviance",
                 linear.predictions = FALSE, ggtheme = theme_bw(), title = "Denovo DSA Class 1")
```

```
# Checking for DSA C2 model homosckedasticity

# To ensure the coxph model assumptions are met, we need to check for homoscedasticity.
# The following plots show the deviance residuals for each model. For the assumption to be met, the residuals should be symmetric aro

ggcoxdiagnostics(mod2, type = "deviance",
                 linear.predictions = FALSE, ggtheme = theme_bw(), title = "Denovo DSA Class 2")
```

```
# OOB C index Boxplot for DSA C1 model

# Boxplot of C-indexes
boxplot(cv.means_C1, main = "Denovo DSA Class 1", ylab = "C Index")
```

```
# OOB C index Boxplot for DSA C2 model

# Boxplot of C-indexes
boxplot(cv.means_C2, main = "Denovo DSA Class 2",  ylab = "C Index")
```

AMR Model

The following code creates the AMR model using coxph and the Class 2 total ElliPro Score

```
# Data preparation

# Load merged data
df <-read.csv("final_epletdata_sepclass.csv")
df <- na.omit(df)

df_amr <- df %>%  select(class2.total_score)
df_amr <- data.frame(df_amr)
colzero <- which(colSums(df_amr)==0)
if (length(colzero)> 0){
  df_amr <-df_amr[,-which(colSums(df_amr)==0)] #get rid of those always have 0
}

df_amr <- cbind(df$DaystoAMRrej,df$AMR_0619,df_amr)
df_amr <- rename(df_amr, DaystoAMRrej = 'df$DaystoAMRrej', AMR_0619 = 'df$AMR_0619')
```

```
# C2 DSA Model training

X <- df_amr
cvK = 5  # number of CV folds
n_sim = 25
cv.means = cv_eval = NA
for(i in 1:n_sim){
  cvSets = cvTools::cvFolds(nrow(X), cvK)  # permute all the data, into 5 folds
  cv_eval = NA  # initialise results vector
  for (j in 1:cvK) {
    test_id = cvSets$subsets[cvSets$which == j]
    X_test = X[test_id, ]
    X_train = X[-test_id, ]

    mod <- coxph(Surv(DaystoAMRrej/365, AMR_0619 ) ~  . , data = X_train)
    cv_eval[j] <- concordance(mod, newdata = X_test)$concordance
  }
  cv.means[i] <- mean(cv_eval)
}

#PH assumption test
amr_zph <- cox.zph(mod)$table
```

```r
amr_fnc <- ggcoxfunctional(mod)
```

```r
# Summary of AMR C2 model

summary(mod)
```

```r
#  Linearity Assumption of AMR C2 model

amr_fnc
```

```r
# Proportional Hazard Assumption of AMR C2 model

amr_zph
```

```r
# HeteroSkedacity of DSA C2 model

# Checking for model homosckedasticity
ggcoxdiagnostics(mod, type = "deviance",
                 linear.predictions = FALSE, ggtheme = theme_bw(), title = "AMR ~ class 2 Ellipro Score")
```

```r
# OOB C index Boxplot for AMR C2 model

# The AMR model has an Out of Bag C Index of around 0.7 using 25 repetitions.

# Boxplot of C-indexes
boxplot(cv.means, main = "AMR Class 2",  ylab = "C Index")
```

References

- Cooper, J. E., Gralla, J., Chan, L., & Wiseman, A. C. (2011). Clinical significance of post kidney transplant de novo DSA in otherwise stable grafts. Clinical transplants, 359–364.
- Debout A, Foucher Y, Trébern-Launay K, et al (2015). Each additional hour of cold ischemia time significantly increases the risk of graft failure and mortality following renal transplantation. Kidney Int. 2015;87(2):343-349. doi:10.1038/ki.2014.304
- Do Nguyen HT, Wong G, Chapman JR, et al. The Association Between Broad Antigen HLA Mismatches, Eplet HLA Mismatches and Acute Rejection After Kidney Transplantation. Transplant Direct. 2016;2(12):e120. Published 2016 Nov 23. doi:10.1097/TXD.0000000000000632 Duquesnoy, R. and Marrari, M., (2017). Usefulness of the ElliPro epitope predictor program in defining the repertoire of HLA-ABC eplets. Human Immunology, 78(7-8), pp.481-488.
- Lionaki S, Panagiotellis K, Iniotaki A, Boletis JN. (2013). Incidence and clinical significance of de novo donor specific antibodies after kidney transplantation. Clin Dev Immunol. 2013; 2013:849835.
- McGee, J. et al. (2010). Donor-recipient gender and size mismatch affects graft success after kidney transplantation. Journal of the American College of Surgeons 210, 718-725.e711, 725-716, 10.1016/j.jamcollsurg.2009.12.032.
- Singh, N., Pirsch, J., & Samaniego, M. (2009). Antibody-mediated rejection: treatment alternatives and outcomes. Transplantation reviews (Orlando, Fla.), 23(1), 34–46. https://doi.org/10.1016/j.trre.2008.08.004
- T- SANZ (2019). OrganMatch A life-changing Link. Retrieved 23 May 2020. https://www.tsanz.com.au/TSANZ%20OrganMatch%20Communique_.pdf