Detecting Cancer with a Urine Test

Dallin Munger, Isabella T Oakes, and Jack McCullers

ADS-503 Applied Predictive Modeling

Shiley-Marcos School of Engineering

University of San Diego

Abstract

Pancreatic cancer has the highest mortality rate when comparing major cancers in the United States and is responsible for close to 50,0000 deaths in 2021. It is extremely difficult to find pancreatic cancer early, as it has very few, if any, symptoms in the early stages and is difficult to detect. Finding early detection strategies is key to giving people the highest chance of survival. Urine tests could provide a potential solution by examining specific biomarkers in the urine to help detect pancreatic cancer when it is at its most treatable.

Contents

Introduction……..4

Exploratory Data Analysis……4

Data Pre-Processing and Splitting……9

Model Strategies……10

Performance Metrics……10

Data Mining Results……..11

R Shiny Web Application……13

Conclusion……14

References……15

Appendix A……16

Appendix B……42

Detecting Cancer with a Urine Test

Pancreatic cancer is very difficult to detect, leading it to be the deadliest cancer (Hirshberg Foundation for Pancreatic Cancer Research, 2021). With mortality rates that are already high, not being able to detect it early means that most of the time it is diagnosed when it is inoperable and terminal. While maintaining a healthy lifestyle can decrease one's likelihood of having pancreatic cancer, finding a way to detect it early on could save the lives of some of the roughly 50,000 people who die from pancreatic cancer each year (Hirshberg Foundation for Pancreatic Cancer Research , 2021). In order to help lower the mortality rate, Silvana Debernardi and colleagues are working on a test that would detect pancreatic cancer earlier using urine biomarkers (2020). If it is possible to detect cancer by testing urine biomarkers, it could be very easy to screen patients regularly and give patients who are diagnosed with pancreatic cancer the highest survival rates. Timely diagnosis is one of the most important factors in successfully curing cancer and creating an accurate test for detecting it is vitally important. Using the data gathered by Debernardi and colleagues, models will be generated and assessed to analyze the biomarkers and look at how accurately one can predict whether someone is cancer free, have non-cancerous pancreatic conditions, or has pancreatic cancer.

Exploratory Data Analysis

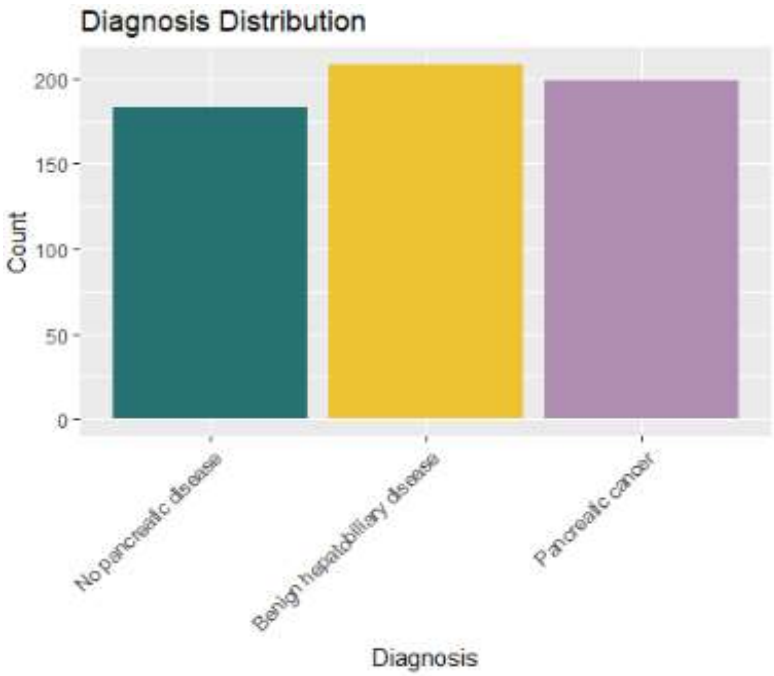
The data consists of 14 columns with four urinary biomarkers which are the main predictors used by Debernardi to test for pancreatic cancer. Each observation includes the Sample ID to identify each subject. Subjects are divided into two cohorts, Cohort 1 is the previously used samples, and Cohort 2 are newly added samples. Each sample has an origin description, with the locations: BPTB (Barts Pancreas Tissue Bank in London), ESP (Spanish National Cancer Research Centre in Madrid), LIV (Liverpool University in the UK), UCL (University College London, UK). Age is the subject's age in years. Sex is their sex, male or female (M or F). Diagnosis includes three categories: 1 (control, no pancreatic disease), 2 (benign hepatobiliary disease), and 3 (pancreatic ductal adenocarcinoma, which is pancreatic

cancer). Stage is missing if the diagnosis was not cancer, and if it is listed is IA, IB, IIA, IIIB, III, or IV. Benign samples have their diagnosis listed. Plasma CA19-9 lists the antibody amount in U/ml, which is a plasma that is often elevated in patients with pancreatic cancer and was not evaluated in all patients. Creatinine is measured in mg/ml, which is a urinary biomarker for kidney function. The four urinary biomarkers are listed in the last four columns. They are LYVE1, measured in ng/ml and measuring a protein called Lymphatic vessel endothelial hyaluronan receptor 1, which is potentially related to tumor metastasis. REG1B is measured in ng/ml and is a protein that may be involved in pancreas regeneration. The third biomarker may be related to urinary tract regeneration and is labeled TFF1 and measured in ng/ml. The final biomarker is REG1A and is measured in ng/ml and only assessed in 306 of the subjects. It may also play a role in pancreas regeneration.

The data consists of 590 samples with only diagnosis equaling pancreatic cancer responding to the 'stage' variable and diagnosis equaling other hepatobiliary disease responding to benign sample diagnosis. REG1A only has responses for 206 samples, and data was not collected for the other samples. Patient cohort has 332 patients in the first cohort and 258 in the second cohort. Most pancreatic cancer patients belong to cohort 1 with four times as many pancreatic cancer subjects than cohort 2. The sample origin of the samples primarily comes from BPTB at 409 samples, with 29 from ESP, 132 from LIV, and 20 from UCL. The only samples without any pancreatic disease or cancer originate from BPTB and UCL only has samples with benign hepatobiliary disease. Age is normally distributed with the distributions for samples with pancreatic cancer moved slightly toward the older subjects (right-skewed). Sex is almost evenly distributed with 299 female and 291 male samples. Males have more pancreatic cancer diagnosis and females have a higher number of no pancreatic disease subjects. The variable diagnosis is also almost equally distributed with 183 subjects classified as no pancreatic disease, 208 classified as benign hepatobiliary disease, and 199 classified as pancreatic cancer, as shown in Figure 1.

Figure 1

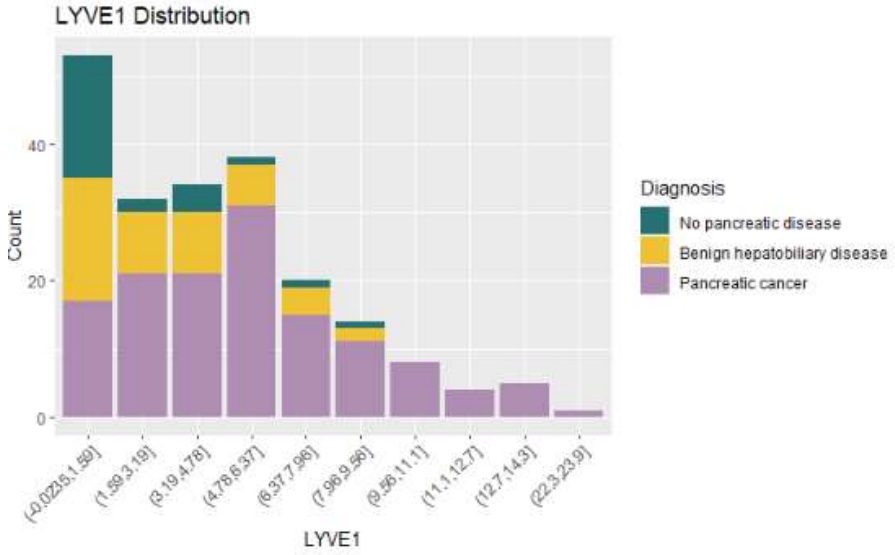
Cancer Diagnosis Distribution



Most of the stage distribution subjects had stage IIIB or III pancreatic cancer. There are several responses for which benign hepatobiliary disease the subject was diagnosed with in the benign sample diagnosis variable, the largest categories of which are cholecystitis, gallstones, types of pancreatitis, and premalignant lesions. Plasma CA19-9 levels had 196 values between −30.4U/ml to 31,000U/ml with 13 subjects higher than 31,000U/ml. Creatinine distribution has left-skewed distribution. LYVE1 also has left-skewed distribution, with most no pancreatic disease and benign hepatobiliary disease focused between −0.0235ng/ml to 6.37ng/ml as shown in Figure 2.

Figure 2

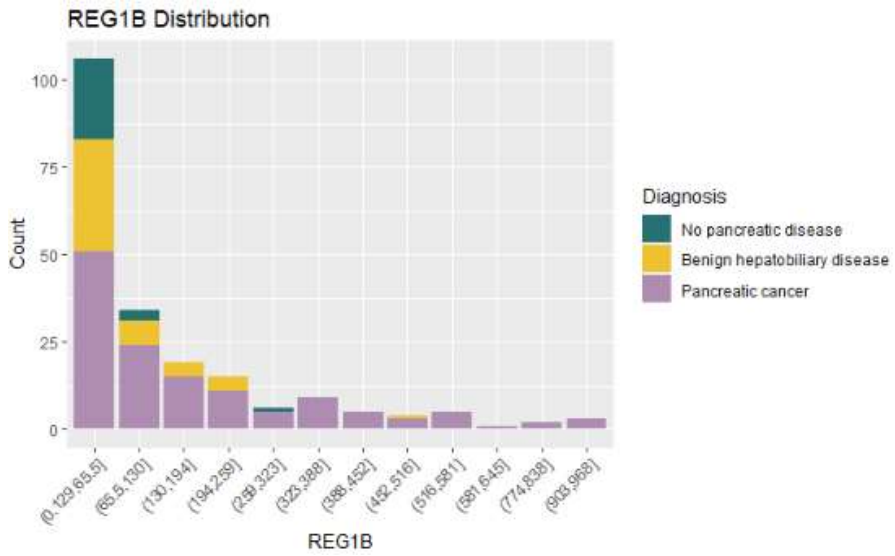
Urinary Biomarker LYVE1 Distribution



REG1B is similarly distributed to LYVE1 with most of the no pancreatic disease and benign hepatobiliary disease between 0.129ng/ml to 65.5ng/ml and an equal amount of pancreatic cancer subjects in the same bin, shown in Figure 3.

Figure 3

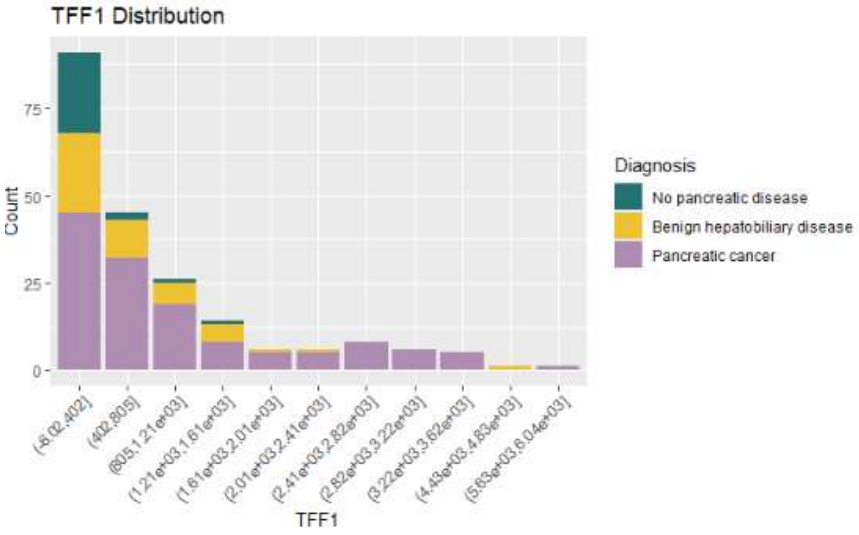
Urinary Biomarker REG1B Distribution



Similar to the other biomarkers, TFF1 is right-skewed with most non-pancreatic cancer responses in the first bin at -6.02ng/ml to 402ng/ml with an equal number of pancreatic cancer responses in the bin as well, shown in Figure 4.

Figure 4

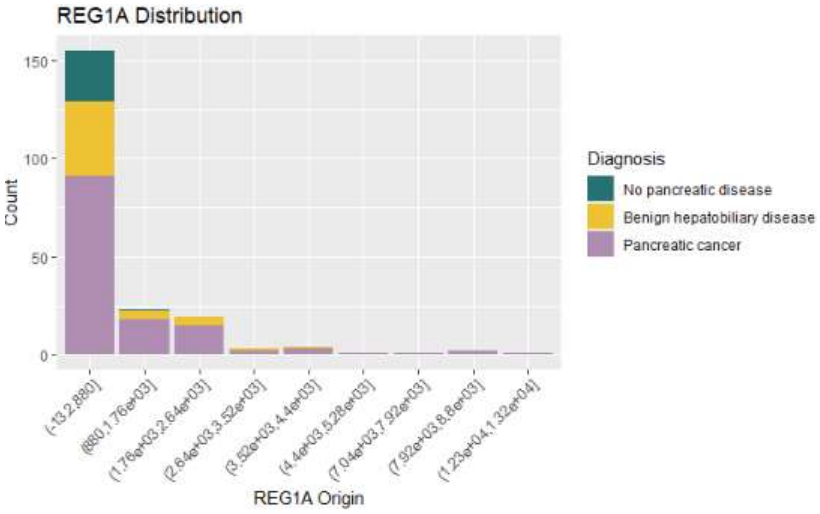
Urinary Biomarker TFF1 Distribution



REG1A has the most skewed distribution out of the biomarkers and it follows a similar distribution to the other biomarkers. Most subjects have values under 880ng/ml. The distribution is shown in Figure 5.

Figure 5

Urinary Biomarker REG1A Distribution



The correlational statistics for the variables are in Figure 6. The top variables correlated to diagnosis are LYVE1 (0.54), TFF1 (0.39), REG1B (0.38), and age (0.31). Some of the biomarkers are correlated, with TFF1 and REG1B having the highest correlation out of any predictors (0.69). TFF1 also has a correlation of 0.58 with LYVE1. Creatinine, plasma levels, sex, and age have the lowest correlation statistics to the other variables.

Figure 6

Correlational Statistics for Variables

	age	sex	diagnosis	plasma_CA19_9	creatinine	LYVE1	REG1B	TFF1	REG1A
age	1.00000000	0.02402341	0.30825060	0.11156715	-0.06939318	0.3298052	0.2043165	0.2078679	0.05724181
sex	0.02402341	1.00000000	0.16908114	0.05381047	0.18005891	0.1770081	0.1575791	0.1146233	0.13853866
diagnosis	0.30825060	0.16908114	1.00000000	0.23826704	0.07488787	0.5403837	0.3835162	0.3926134	0.25365730
plasma_CA19_9	0.11156715	0.05381047	0.23826704	1.00000000	-0.02969385	0.2091402	0.1383811	0.1458377	0.16331530
creatinine	-0.06939318	0.18005891	0.07488787	-0.02969385	1.00000000	0.3385325	0.2619057	0.3977954	0.10775096
LYVE1	0.32980522	0.17700814	0.54038373	0.20914023	0.33853255	1.0000000	0.5430746	0.5773628	0.28216552
REG1B	0.20431655	0.15757906	0.38351617	0.13838106	0.26190569	0.5430746	1.0000000	0.6902278	0.33601737
TFF1	0.20786791	0.11462335	0.39261345	0.14583767	0.39779545	0.5773628	0.6902278	1.0000000	0.29734268
REG1A	0.05724181	0.13853866	0.25365730	0.16331530	0.10775096	0.2821655	0.3360174	0.2973427	1.00000000

Data Pre-Processing and Splitting

To prepare the data for model fitting, missing data was dealt with, predictors were checked for degenerate distributions, and the predictors were checked to determine if any highly correlated variables existed. It was discovered that there was no missing data in any of the predictors or the response save two only: REG1A and Plasma CA19-9. These two variables were missing a considerable amount of data. This was due to the fact that REG1A was only assessed in 306 patients and Plasma CA19-9 was only assessed in 350 patients out of 590 total patients.

To deal with the missing data, predictive mean matching was used to impute missing values. After imputation, the means of the two variables with missing values were compared to the means without missing data and after predictive mean matching imputation. No statistical difference was found in the means before and after imputation, demonstrating the effectiveness of predictive mean matching in maintaining the original distribution of the variables. The variances were also compared

before and after imputation. No major differences existed. Therefore, the predictive mean matching imputation method was effective in preserving the original center and spread of the variables REG1A and Plasma CA19-9.

After imputation was performed for missing values, each variable was examined to determine if any sparse predictors existed. The predictors were also compared to determine if any were highly correlated. Based on that analysis, no predictors were found to be sparse or highly correlated. No predictors were then removed from the predictor set.

Once the pre-processing was completed, the dataset was split into a test and a training set, with 70% of the observations being sampled for the training set. A 10-fold cross validation method was used to train the respective models.

Model Strategies

Six separate models were trained, validated, and tested to predict the diagnosis of a given individual. The diagnosis variable was transformed into a binary response, no pancreatic cancer or pancreatic cancer. The models selected for this binary classification problem included a random forest model, a support vector machine, a penalized logistic regression model, an extreme gradient boost tree model, a naïve Bayes model, and a neural network. The models were selected to provide a variety of approaches, including linear classification models, nonlinear classification models, and tree-based models.

Performance Metrics

The team identified the following model performance metrics to highlight within the data mining results section, the model's Receiver Operating Characteristic curve, sensitivity, specificity, accuracy, positive prediction value, the negative prediction value, and the most likely class returned.

Here is a summary of the performance metrics to help with understanding the team's individual models. Receiver operating characteristic curve (ROC) is a graph showing the performance of the model's classification thresholds. The curve plots the true positive rate and false positive rate parameters. In association with ROC was the area under the curve of the ROC curve. This provides an aggregate measure of performance across all possible classification thresholds. Model sensitivity is the measure of the proportion of actual positive cases predicted as positive, while the model's specificity is the metric that evaluates a model's ability on predicting true negatives. Each of these are presented as percentages in the following section. The team tried to tune appropriate models to avoid too high or low of sensitivity and specificity. The next metric was accuracy. The model accuracy is defined as the number of classifications a model correctly predicts from the overall total predictions. The team used this metric to asses performances of each model created. Finally, the team will be reporting out what the likeliest class returned was. This would help with the models overfitting or underfitting of the urinary biomarkers data. The team created the following types of predictive models: random forest, support vector machines, Naïve Bayes, artificial neural networks, XGBoost tree, and penalized logistic regression.

Data Mining Results

The team began with constructing a random forest model and training it against the train data set. The model utilized diagnosis as the response variable. The performance metric for the model would be the ROC value. The model resulted in a high ROC of roughly 94.94%, a sensitivity value of 92.37%, and a specificity of 81.43%. After documenting the performance of the initial model training, the team used the prediction function to measure the test set performance. The model performed well with a returned accuracy of 85.23%. The sensitivity of the random forest model was 91.45% and the specificity was lower with a value of 72.88%. This accuracy was one the highest amongst the models the team created. The model returned that the most likely class to be predicted was the "No" pancreatic cancer class.

The team then executed a Naïve Bayes predictive model utilizing diagnosis as the response variable. The first train and tune of the model returned a ROC of roughly 93%. The team then analyzed the test set performance on the Naïve Bayes model. The Naïve Bayes model returned an accuracy of roughly 76.14%. The downside to this model is that it had a high number of false positives after reviewing the confusion matrix statistics. The class with the highest probability and determined to be the most likely was the "No" pancreatic cancer class. This model performed well, but did not improve on the standard set by the random forest model. The team moved onto developing an artificial neural network model next.

The team created an artificial neural network model utilizing diagnosis again as the response variable. The first steps the team did was to create the neural network grid and setting the maximum grid size the model. By utilizing the nnet package, the team trained a neural network model from the train split. The model yielded an ROC similar to the team's initial Naïve Bayes model, roughly 93%. The model's true positive rate was in line with our Naïve Bayes. The team then executed the prediction function of the model against the test data set. The team noticed a slight five percent increase in overall model accuracy against the Naïve Bayes model, 81.25%. The model's p-value retuned at .2963. The positive prediction rate was around 83% while the negative prediction rate was lower with 75%. The model's sensitivity was high with roughly 88% while the specificity was about 20% lower at 66.10%. The most likely class returned from this model was "No" pancreatic cancer.

The team moved onto creating a third model utilizing support vector machines. The team trained an initial model with diagnosis as the response variable and a radial method. The results from training the model, with the train split set, returned an ROC of roughly 90.71%. This was the lowest of the initial ROCs of the first three models. After predicting, utilizing the test split, the team concluded that the final accuracy of the model was 75.57%. The model accuracy was lower than the Naïve Bayes and the neural network models. The model's sensitivity was 79.49% and the specificity was about

67.80%. The model's positive prediction rate was on par with the neural network model, but the model's negative prediction rate was about 5% lower at 62.50%. The most likely class returned by the model was again the "No" pancreatic cancer.

The team then moved onto creating an XGBoost Tree model. The team trained the initial model by utilizing a tune length on 5, establishing diagnosis as the response variable, and with the performance metric of ROC. The model retuned an ROC of 94.61%, a sensitivity of 91.26%, and a specificity of 81.43%. The team then moved onto collecting the prediction performance on the test set. The model accuracy was closer to the team's initial random forest model with 84.09%. The model had a decently high sensitivity rate of 89.74% and a specificity rate of 72.88%. For the XGBoost tree model, the most likely class returned was the "No" pancreatic cancer classification.

The final model the team created was a penalized logistic regression model. The model was trained utilizing diagnosis as the response variable, the performance metric as ROC, and a tune length of 5. The model retuned an ROC of 91.15%, a sensitivity of 93.06%, and a specificity of 70.71%. The team then moved onto collecting the prediction performance on the test set. The model accuracy was lower than the previous XGBoost model with a value of 79.55%. The model returned a sensitivity value of 86.32% and a specificity of 66.10%. The penalized logistic regression model returned the "No' pancreatic cancer as the most likely classification.

R Shiny Web Application

The team was able to develop an R Shiny Web application to determine pancreatic cancer presence though manual entering of predictor values. The web application is attached with these submittals. The team's application determines the pancreatic cancer presence utilizing the random forest model the team initially created. Code for the R Shiny Web application is listed in Appendix B.

Conclusion

The team utilized the urinary biomarkers data set to build multiple predictive models utilizing diagnosis as the response variable. The team noticed the top performing models to be the Random Forest and XGBoost models. From the overall performances, the team concluded that the models created could accurately predict diagnosis of pancreatic cancer. The team was able to determine these models as the top performers based on their accurate predictions from the confusion matrixes, and their ease of interpretability. The team went on to create a protype R Shiney web application to enter specific urinary biomarker values, patient age, and patient sex to display the distribution of predicted pancreatic cancer results. The team strongly recommends that further data collection should be accomplished in order to create a more robust model with more training and testing items.

References

Debernardi, S., O'Brien, H., Algahmdi, A. S., Malats, N., Stewart, G. D., Pliješa-Ercegovac, M., Costello, E., Greenhalf, W., Saad, A., Roberts, R., Ney, A., Pereira, S. P., Kocher, H. M., Duffy, S., Blyuss, O., Crnogorac-Jurcevic, T. (2020, December 10). A combination of urinary biomarker panel and PancRISK score for earlier detection of pancreatic cancer: A case-control study. *PLOS Medicine*. *17(12)*. https://journals.plos.org/plosmedicine/article?id=10.1371/journal.pmed.1003489

Hirshberg Foundation for Pancreatic Cancer Research. (2021) *Pancreatic Cancer Facts.* https://pancreatic.org/pancreatic-cancer/pancreatic-cancer-facts/

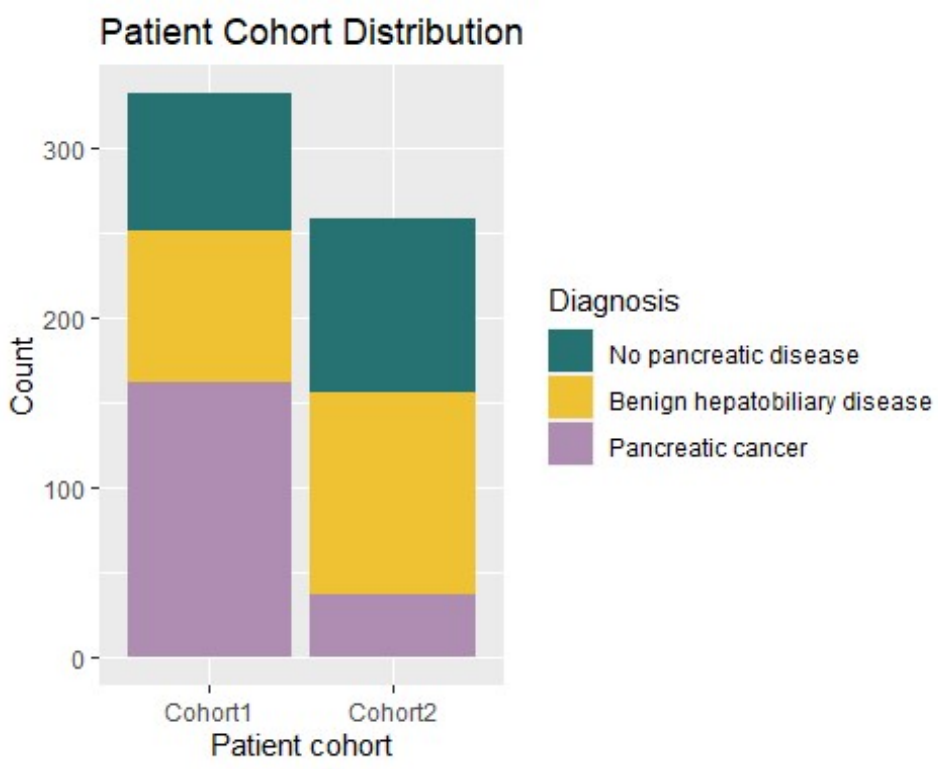
Appendix A

```
library(dplyr)
library(ggplot2)
library(OneR)
library(corrplot)
library(DataExplorer)
library(caret)
library(klaR)
library(MASS)

#Read in the data
pancreatic <- read.csv('C:/Users/Ladybug/Desktop/pancreatic.csv', header = TR
UE)

#Summarise the variables in the dataset
summary(pancreatic)

##  ï..sample_id       patient_cohort     sample_origin           age
##  Length:590         Length:590         Length:590         Min.   :26.00
##  Class :character   Class :character   Class :character   1st Qu.:50.00
##  Mode  :character   Mode  :character   Mode  :character   Median :60.00
##                                                           Mean   :59.08
##                                                           3rd Qu.:69.00
##                                                           Max.   :89.00
##
##      sex              diagnosis        stage           benign_sample_diagn
osis
##  Length:590         Min.   :1.000   Length:590         Length:590
##  Class :character   1st Qu.:1.000   Class :character   Class :character
##  Mode  :character   Median :2.000   Mode  :character   Mode  :character
##                     Mean   :2.027
##                     3rd Qu.:3.000
##                     Max.   :3.000
##
##  plasma_CA19_9       creatinine          LYVE1              REG1B
##  Min.   :    0.0   Min.   :0.05655   Min.   : 0.000129   Min.   :   0.0011
##  1st Qu.:    8.0   1st Qu.:0.37323   1st Qu.: 0.167179   1st Qu.:  10.7572
##  Median :   26.5   Median :0.72384   Median : 1.649862   Median :  34.3034
##  Mean   :  654.0   Mean   :0.85538   Mean   : 3.063530   Mean   : 111.7741
##  3rd Qu.:  294.0   3rd Qu.:1.13948   3rd Qu.: 5.205037   3rd Qu.: 122.7410
##  Max.   :31000.0   Max.   :4.11684   Max.   :23.890323   Max.   :1403.8976
##  NA's   :240
##       TFF1              REG1A
##  Min.   :    0.005   Min.   :    0.00
##  1st Qu.:   43.961   1st Qu.:   80.69
##  Median :  259.874   Median :  208.54
##  Mean   :  597.869   Mean   :  735.28
##  3rd Qu.:  742.736   3rd Qu.:  649.00
```

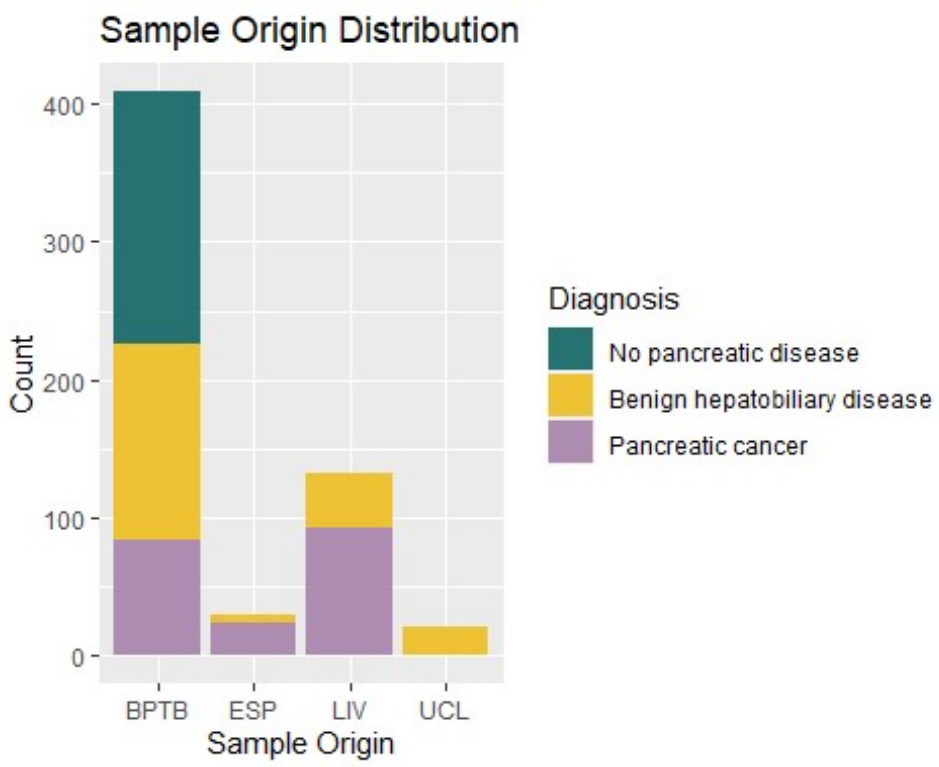
```
##  Max.   :13344.300   Max.   :13200.00
##                      NA's   :284

#View the first few observations of the dataset
head(pancreatic)

##   ï..sample_id patient_cohort sample_origin age sex diagnosis stage
## 1           S1        Cohort1          BPTB  33   F         1
## 2          S10        Cohort1          BPTB  81   F         1
## 3         S100        Cohort2          BPTB  51   M         1
## 4         S101        Cohort2          BPTB  61   M         1
## 5         S102        Cohort2          BPTB  62   M         1
## 6         S103        Cohort2          BPTB  53   M         1
##   benign_sample_diagnosis plasma_CA19_9 creatinine      LYVE1     REG1B
## 1                                 11.7    1.83222 0.89321920  52.94884
## 2                                   NA    0.97266 2.03758500  94.46703
## 3                                  7.0    0.78039 0.14558890 102.36600
## 4                                  8.0    0.70122 0.00280488  60.57900
## 5                                  9.0    0.21489 0.00085956  65.54000
## 6                                   NA    0.84825 0.00339300  62.12600
##       TFF1    REG1A
## 1 654.2822 1262.000
## 2 209.4882  228.407
## 3 461.1410       NA
## 4 142.9500       NA
## 5  41.0880       NA
## 6  59.7930       NA
```

Exploratory Data Analysis

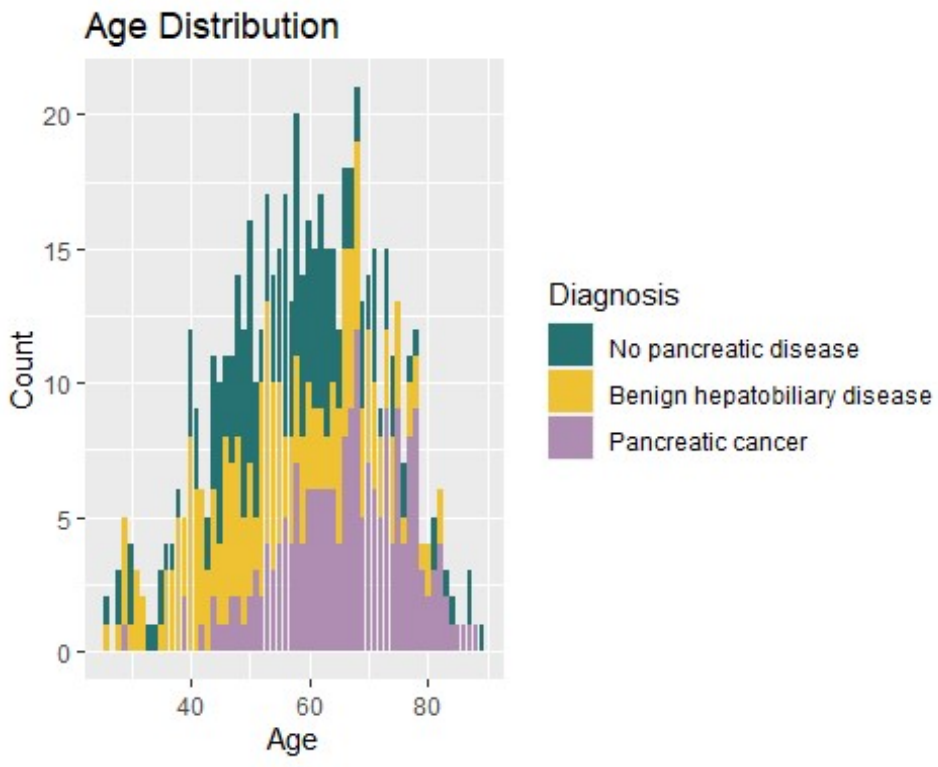
```
#Patient cohort distribution
pancreaticdata <- as.data.frame(pancreatic)
pancreaticdata$diagnosis <- as.factor(pancreaticdata$diagnosis)
levels(pancreaticdata$diagnosis) <- c("No pancreatic disease", "Benign hepato
biliary disease", "Pancreatic cancer")
qplot(patient_cohort, data = pancreaticdata, geom = "bar", fill = diagnosis)
+
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Patient Cohort Distribution", x = "Patient cohort", y = "Coun
t")
```



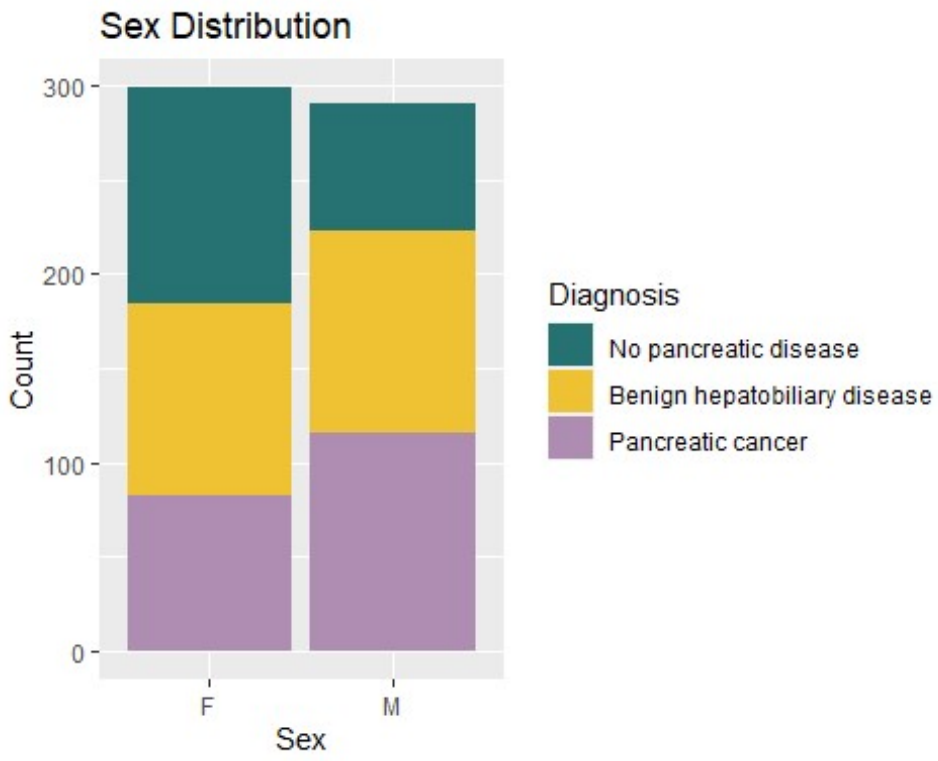

```
pancreaticdata %>% count(patient_cohort)

##   patient_cohort   n
## 1        Cohort1 332
## 2        Cohort2 258

#Sample origin distribution
qplot(sample_origin, data = pancreaticdata, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Sample Origin Distribution", x = "Sample Origin", y = "Count"
)
```



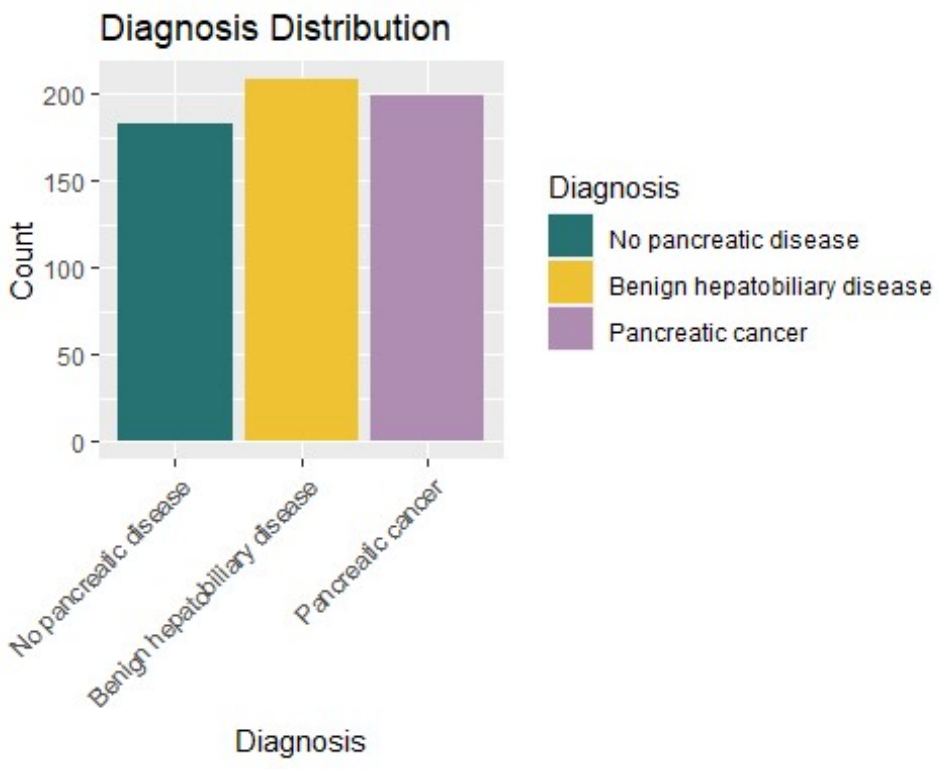

```
pancreaticdata %>% count(sample_origin)

##   sample_origin   n
## 1          BPTB 409
## 2           ESP  29
## 3           LIV 132
## 4           UCL  20

#Age distribution
qplot(age, data = pancreaticdata, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Age Distribution", x = "Age", y = "Count")
```



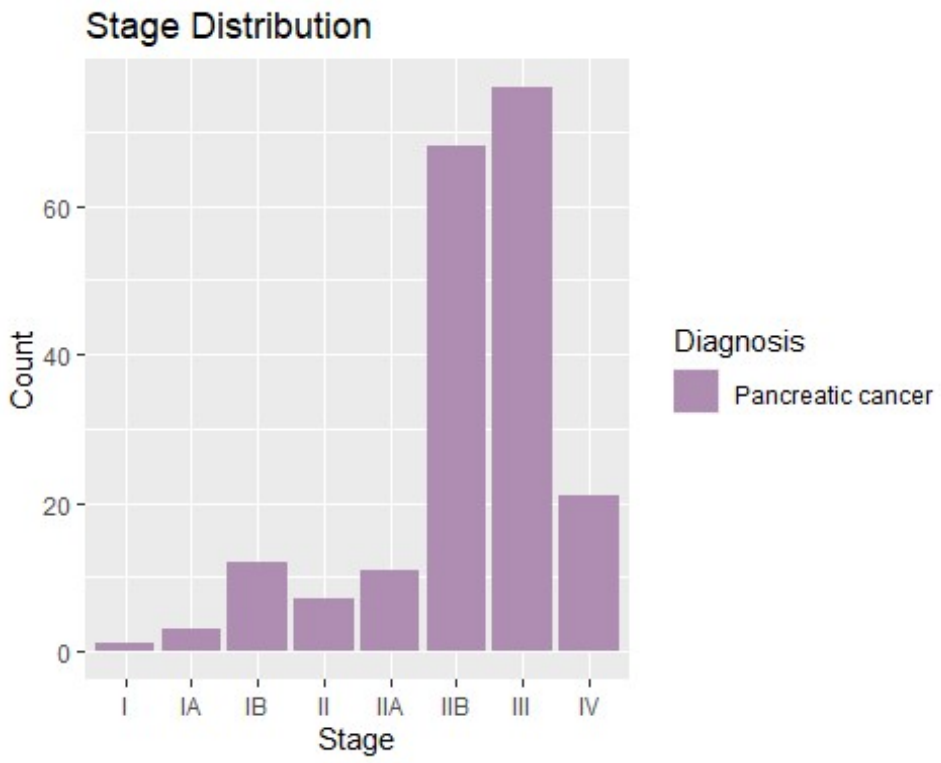

```
#Sex distribution
qplot(sex, data = pancreaticdata, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Sex Distribution", x = "Sex", y = "Count")
```



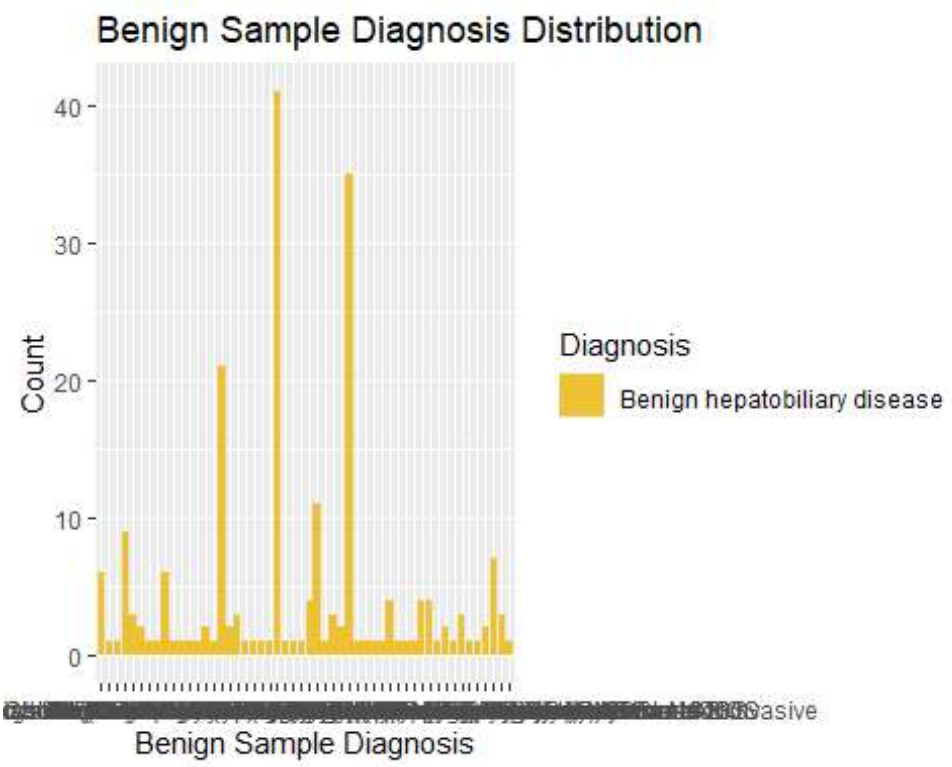

```
pancreaticdata %>% count(sex)

##   sex   n
## 1   F 299
## 2   M 291

#Diagnosis distribution
qplot(diagnosis, data = pancreaticdata, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Diagnosis Distribution", x = "Diagnosis", y = "Count") +
  theme(axis.text.x = element_text(angle=45, vjust=1, hjust=1))
```




```
pancreaticdata %>% count(diagnosis)

##                       diagnosis   n
## 1        No pancreatic disease 183
## 2 Benign hepatobiliary disease 208
## 3            Pancreatic cancer 199

#Stage distribution
panstage <- pancreaticdata[pancreaticdata$diagnosis == "Pancreatic cancer", ]
qplot(stage, data = panstage, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#af8db2"), name = "Diagnosis") +
  labs(title = "Stage Distribution", x = "Stage", y = "Count")
```



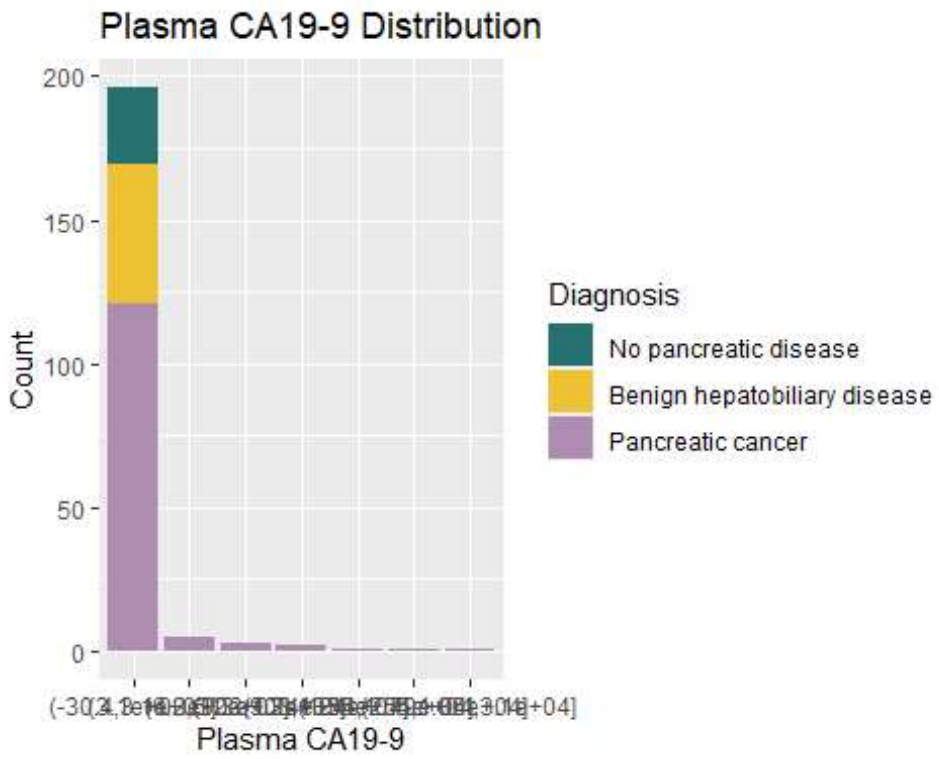

```
pancreaticdata %>% count(diagnosis)

##                       diagnosis   n
## 1        No pancreatic disease 183
## 2 Benign hepatobiliary disease 208
## 3            Pancreatic cancer 199

#Benign sample diagnosis distribution
benign <- pancreaticdata[pancreaticdata$diagnosis == "Benign hepatobiliary di
sease", ]
qplot(benign_sample_diagnosis, data = benign, geom = "bar", fill = diagnosis)
+
  scale_fill_manual(values = c("#efc234"), name = "Diagnosis") +
  labs(title = "Benign Sample Diagnosis Distribution", x = "Benign Sample Dia
gnosis", y = "Count")
```



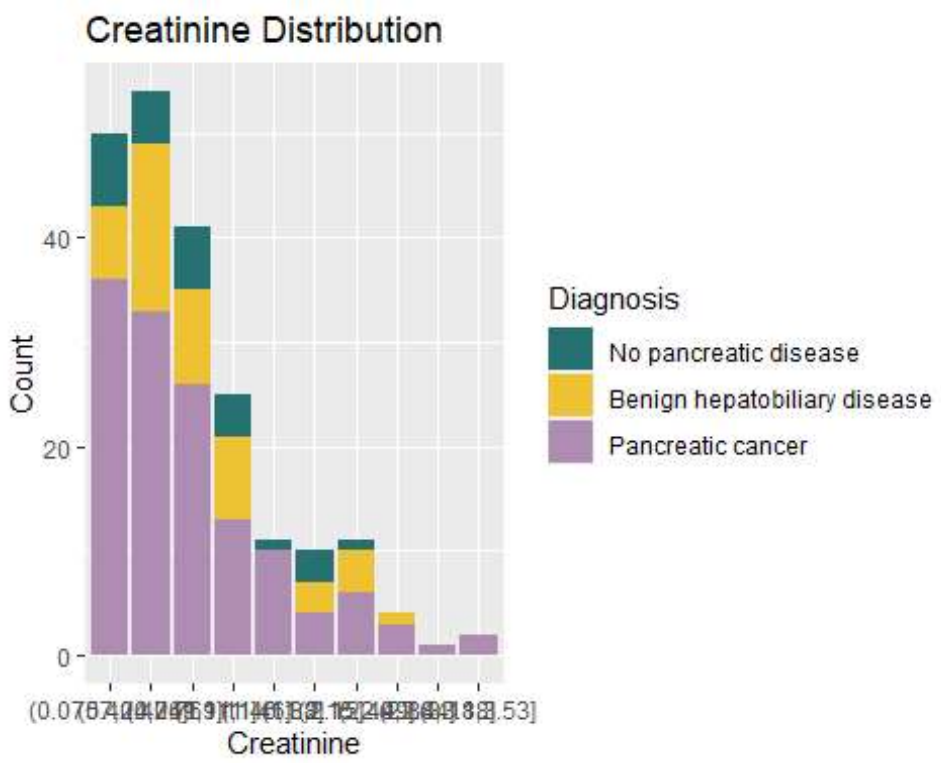

```
benign %>% count(benign_sample_diagnosis)
```

```
##                                          benign_sample_diagnosis  n
## 1                                                  Abdominal Pain   6
## 2                        Biliary Stricture (Secondary to Stent)   1
## 3                                                  Cholecystitis  1
## 4                                                 Cholecystitis   9
## 5                                      Cholecystitis (Chronic)   3
## 6                        Cholecystitis (Chronic) Cholelithiasis  2
## 7                        Cholecystitis (Chronic) Cholesterolsis  1
## 8                                             Choledochal Cyst   1
## 9                                         Choledocholiathiasis  6
## 10                                        Choledocholiathiasis   1
## 11              Cholelithiasis with adenomyomatous hyperplasia   1
## 12                                         Duodenal Stricture  1
## 13                                                 Duodenitis  1
## 14                                       Gallbladder polyps   2
## 15                                       Gallbladder Porcelain  1
## 16                                                 Gallstones 21
## 17                                                Gallstones   2
## 18                                     Gallstones - Incidental   3
## 19                                                 Gastritis   1
## 20                                       Gastritis and Reflux   1
## 21                       Ill defined lesion in uncinate process  1
## 22                         Ischaemic Common Bile Duct Stricture  1
## 23                                                Pancreatitis 41
## 24                                               Pancreatitis   1
```

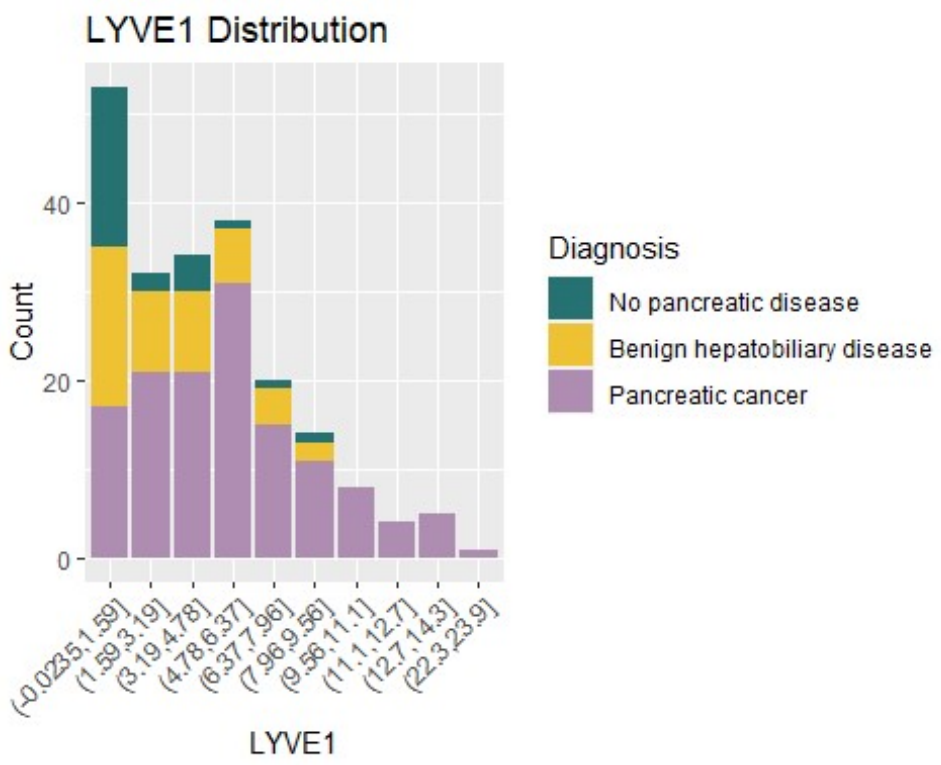
```
## 25                                        Pancreatitis (Abscess)  1
## 26                                          Pancreatitis (Acute)  1
## 27                       Pancreatitis (Alcohol-Chronic-Pseuodcyst)  4
## 28                                  Pancreatitis (Alcohol-Chronic) 11
## 29                                          Pancreatitis (Alcohol)  1
## 30                                     Pancreatitis (Autoimmune)   3
## 31                               Pancreatitis (Chronic-Pseudocyst)   2
## 32                                       Pancreatitis (Chronic)  35
## 33                      Pancreatitis (Chronic) (Later became PDAC)  1
## 34                      Pancreatitis (Chronic) Choledocholithiasis  1
## 35                    Pancreatitis (Gallstone-Alcohol-Pseudocyst)  1
## 36                            Pancreatitis (Gallstone-Pseudocyst)  1
## 37                                       Pancreatitis (Gallstone)   4
## 38                             Pancreatitis (Hereditary-Chronic)   1
## 39                            Pancreatitis (Hypertriglyceridemia)  1
## 40                                       Pancreatitis (Idiopathic)  1
## 41                                     Pancreatitis (Idiopathic)   4
## 42                                       Pancreatitis (Pseudocyst)  4
## 43                    Pancreato-jejunostomy Anastomoses Stricture  1
## 44                                 Premalignant lesions-Adenoma-NOS  2
## 45 Premalignant lesions-Mucinous cystadenocarcinoma-noninvasive  1
## 46                   Premalignant lesions-Mucinous cystadenoma-NOS  3
## 47                        Premalignant lesions-Tubular adenoma-NOS  1
## 48                  Premalignant lesions-Tubulovillous adenoma-NOS  1
## 49                        Premalignant lesions-Villous adenoma-NOS  2
## 50                                        Serous cystadenoma - NOS  7
## 51                                     Serous microcystic adenoma  3
## 52                                       Simple benign liver cyst  1

#Plasma distribution
plasma <- bin(pancreaticdata, nbins = 10, labels = NULL, method = c("length")
, na.omit = TRUE)
qplot(plasma_CA19_9, data = plasma, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Plasma CA19-9 Distribution", x = "Plasma CA19-9", y = "Count"
)
```



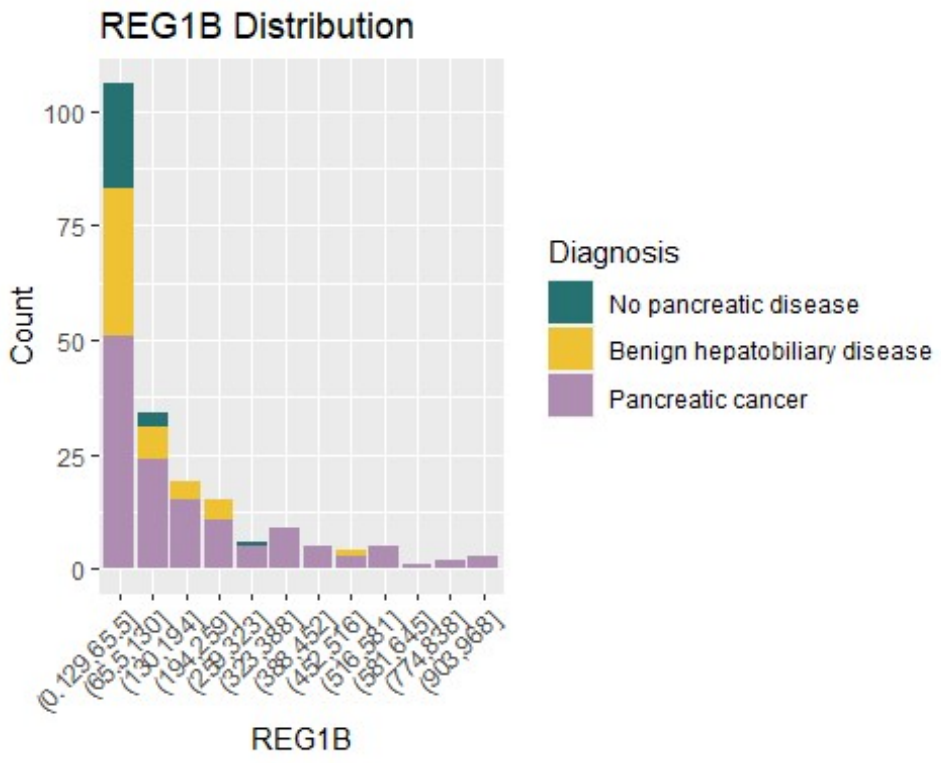

```
plasma %>% count(plasma_CA19_9)

##         plasma_CA19_9   n
## 1     (-30.4,3.1e+03] 196
## 2   (3.1e+03,6.2e+03]   5
## 3   (6.2e+03,9.3e+03]   3
## 4  (9.3e+03,1.24e+04]   2
## 5 (1.24e+04,1.55e+04]   1
## 6 (1.55e+04,1.86e+04]   1
## 7  (2.79e+04,3.1e+04]   1

#Creatinine Distribution
creatinine <- bin(pancreaticdata, nbins = 10, labels = NULL, method = c("leng
th"), na.omit = TRUE)
qplot(creatinine, data = creatinine, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "Creatinine Distribution", x = "Creatinine", y = "Count")
```



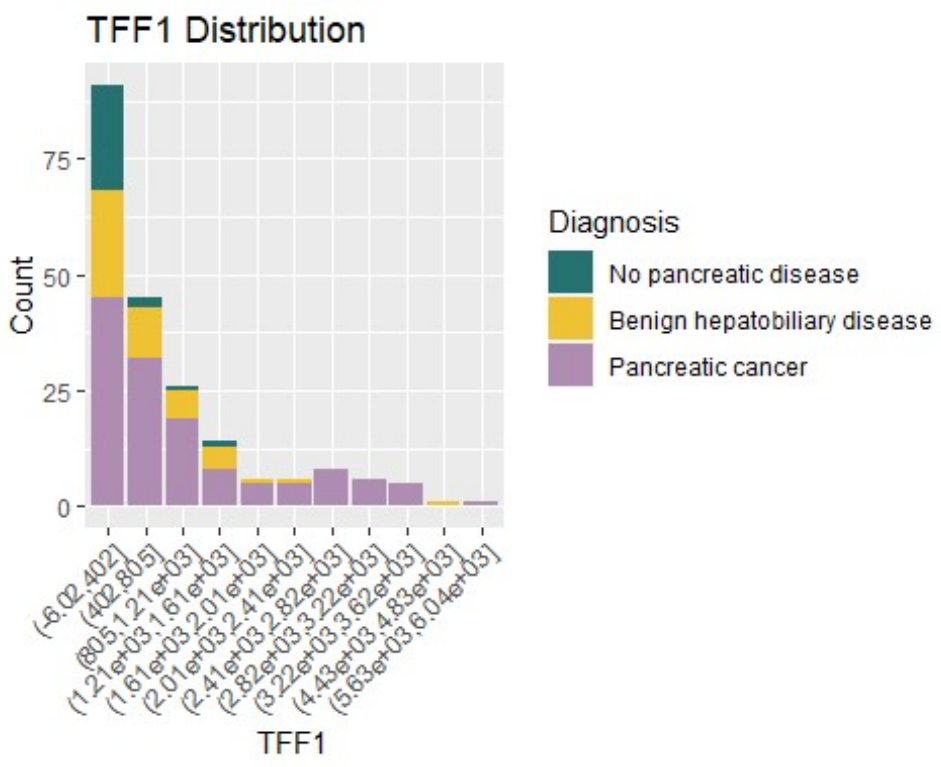

```
#LYVE1 Distribution
LYVE1 <- bin(pancreaticdata, nbins = 15, labels = NULL, method = c("length"), na.omit = TRUE)
qplot(LYVE1, data = LYVE1, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diagnosis") +
  labs(title = "LYVE1 Distribution", x = "LYVE1", y = "Count") +
  theme(axis.text.x = element_text(angle=45, vjust=1, hjust=1))
```



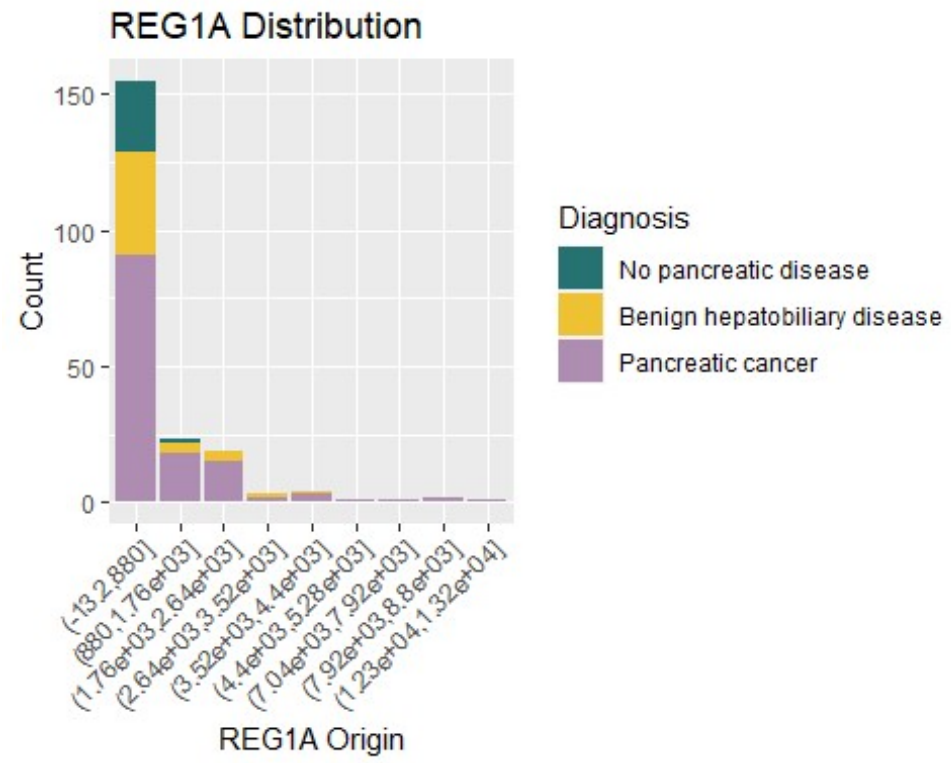

```
#REG1B Distribution
REG1B <- bin(pancreaticdata, nbins = 15, labels = NULL, method = c("length"), na.omit = TRUE)
qplot(REG1B, data = REG1B, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diagnosis") +
  labs(title = "REG1B Distribution", x = "REG1B", y = "Count") +
  theme(axis.text.x = element_text(angle=45, vjust=1, hjust=1))
```




```
#TFF1 Distribution
TFF1 <- bin(pancreaticdata, nbins = 15, labels = NULL, method = c("length"),
na.omit = TRUE)
qplot(TFF1, data = TFF1, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "TFF1 Distribution", x = "TFF1", y = "Count") +
  theme(axis.text.x = element_text(angle=45, vjust=1, hjust=1))
```




```
#REG1A Distribution
REG1A <- bin(pancreaticdata, nbins = 15, labels = NULL, method = c("length"), 
na.omit = TRUE)
qplot(REG1A, data = REG1A, geom = "bar", fill = diagnosis) +
  scale_fill_manual(values = c("#257272", "#efc234", "#af8db2"), name = "Diag
nosis") +
  labs(title = "REG1A Distribution", x = "REG1A Origin", y = "Count") +
  theme(axis.text.x = element_text(angle=45, vjust=1, hjust=1))
```



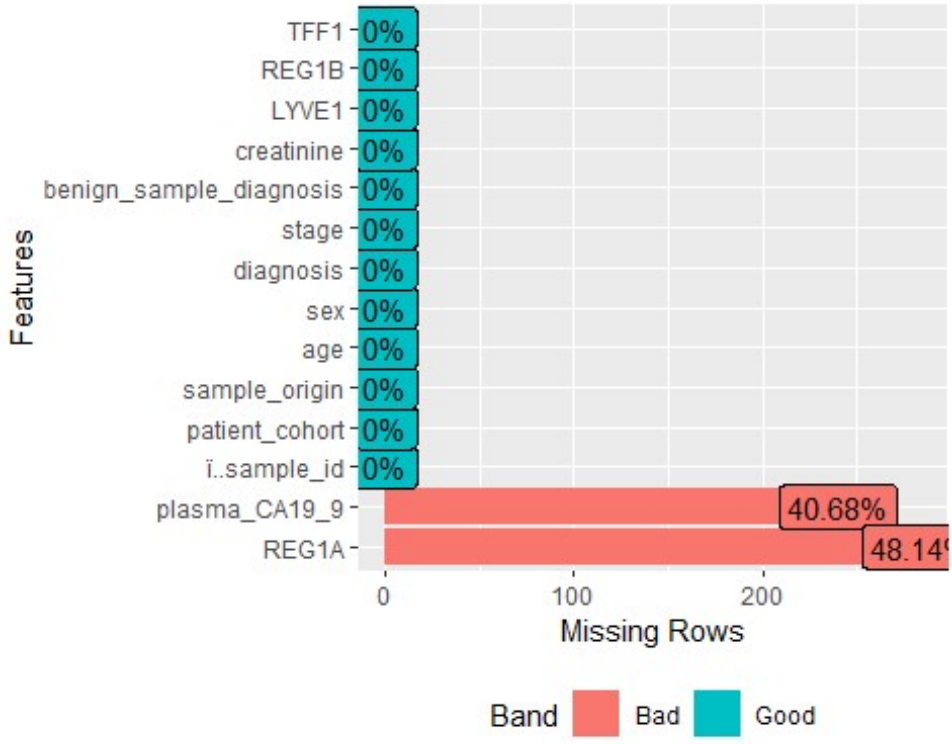

```
#Correlation values
pancreatic2 <- pancreatic
pancreatic2$sex <- as.factor(pancreatic2$sex)
pancreatic2$sex <- as.numeric(pancreatic2$sex)
corrpan <- pancreatic2[c(4:6, 9:14)]
corrpan[is.na(corrpan)] <- 0
cor(corrpan)
```

```
##                       age        sex  diagnosis plasma_CA19_9  creatinine
## age            1.00000000 0.02402341 0.30825060    0.11156715 -0.06939318
## sex            0.02402341 1.00000000 0.16908114    0.05381047  0.18005891
## diagnosis      0.30825060 0.16908114 1.00000000    0.23826704  0.07488787
## plasma_CA19_9  0.11156715 0.05381047 0.23826704    1.00000000 -0.02969385
## creatinine    -0.06939318 0.18005891 0.07488787   -0.02969385  1.00000000
## LYVE1          0.32980522 0.17700814 0.54038373    0.20914023  0.33853255
## REG1B          0.20431655 0.15757906 0.38351617    0.13838106  0.26190569
## TFF1           0.20786791 0.11462335 0.39261345    0.14583767  0.39779545
## REG1A          0.05724181 0.13853866 0.25365730    0.16331530  0.10775096
##                    LYVE1     REG1B      TFF1      REG1A
## age            0.3298052 0.2043165 0.2078679 0.05724181
## sex            0.1770081 0.1575791 0.1146233 0.13853866
## diagnosis      0.5403837 0.3835162 0.3926134 0.25365730
## plasma_CA19_9  0.2091402 0.1383811 0.1458377 0.16331530
## creatinine     0.3385325 0.2619057 0.3977954 0.10775096
## LYVE1          1.0000000 0.5430746 0.5773628 0.28216552
## REG1B          0.5430746 1.0000000 0.6902278 0.33601737
```

```
## TFF1           0.5773628 0.6902278 1.0000000 0.29734268
## REG1A          0.2821655 0.3360174 0.2973427 1.00000000
```

Data cleaning - missing values, degenerate distributions, highly correlated features

```
#First handle missing values - first view the distribution of missing values
plot_missing(pancreatic)
```




```
library(mice)
#Use MICE to impute missing values for both variables
tempDat <- mice(pancreatic, m = 5, maxit = 50, method = "pmm", seed = 27, pri
ntFlag = FALSE)
pancreatic_complete <- complete(tempDat, 2)
#Compare means of original variables and imputed variables using t-test
t.test(pancreatic$REG1A, pancreatic_complete$REG1A)

##
##  Welch Two Sample t-test
##
## data:  pancreatic$REG1A and pancreatic_complete$REG1A
## t = 0.33237, df = 595.19, p-value = 0.7397
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -167.4481  235.6697
## sample estimates:
## mean of x mean of y
##  735.2812  701.1704
```

```
t.test(pancreatic$plasma_CA19_9, pancreatic_complete$plasma_CA19_9)

##
##  Welch Two Sample t-test
##
## data:  pancreatic$plasma_CA19_9 and pancreatic_complete$plasma_CA19_9
## t = -0.075565, df = 821.63, p-value = 0.9398
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -355.7754  329.3980
## sample estimates:
## mean of x mean of y
##  654.0029  667.1916

#Compare spread using standard deviation
c("Missing REG1A" = sd(pancreatic$REG1A, na.rm = TRUE), "No Missing REG1A" =
sd(pancreatic_complete$REG1A))

##    Missing REG1A No Missing REG1A
##         1477.248         1416.569

c("Missing Plasma" = sd(pancreatic$plasma_CA19_9, na.rm = TRUE), "No Missing
Plasma" = sd(pancreatic_complete$plasma_CA19_9))

##    Missing Plasma No Missing Plasma
##          2430.318          2831.297

#Check for missing values
sum(is.na(pancreatic_complete))

## [1] 0

library(caret)
#Check for near zero variance features
nearZeroVar(pancreatic_complete)

## integer(0)

#Check for highly correlated features
findLinearCombos(pancreatic_complete[,c(4,9:14)])

## $linearCombos
## list()
##
## $remove
## NULL
```

Data splitting

```
#Change the response variable diagnosis from three classes to two - pancreati
c cancer (Yes) or no pancreatic cancer (No)
pancreatic_complete <- pancreatic_complete %>%
  mutate(diagnosis = as.factor(ifelse(diagnosis == 1 | diagnosis == 2, "No",
```

```
"Yes")))
pancreatic_complete <- subset( pancreatic_complete, select = -c(stage, benign
_sample_diagnosis, ï..sample_id, patient_cohort, sample_origin))


#Split the data into a train and a test set
samples <- createDataPartition(pancreatic_complete$diagnosis, p = 0.7, list =
FALSE)[,1]
train <- pancreatic_complete[samples,]
test <- pancreatic_complete[-samples,]

#Ensure that the train and test have the same distribution for the response v
ariable
sum(train$diagnosis == "Yes")/nrow(train)

## [1] 0.3381643

sum(test$diagnosis == "Yes")/nrow(test)

## [1] 0.3352273

#Set control parameters
ctrl <- trainControl(summaryFunction = twoClassSummary,
                     method = "cv",
                     classProbs = TRUE,
                     savePredictions = TRUE)
```

Random Forest Model

```
#Random Forest model
set.seed(100)
RFmodel <- train(diagnosis~., data = train,
                 method = "rf",
                 metric = "ROC",
                 trControl = ctrl)

RFmodel$results[RFmodel$results$mtry == RFmodel$bestTune[1,], c(2:4)]

##         ROC      Sens      Spec
## 1 0.9493764 0.9236772 0.8142857
```

Random Forest Statistics

```
#Test set performance
confusionMatrix(predict(RFmodel, test), test$diagnosis)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  107  16
##        Yes  10  43
```

```
##
##                Accuracy : 0.8523
##                  95% CI : (0.7911, 0.9012)
##     No Information Rate : 0.6648
##     P-Value [Acc > NIR] : 1.48e-08
##
##                   Kappa : 0.66
##
##  Mcnemar's Test P-Value : 0.3268
##
##             Sensitivity : 0.9145
##             Specificity : 0.7288
##          Pos Pred Value : 0.8699
##          Neg Pred Value : 0.8113
##              Prevalence : 0.6648
##          Detection Rate : 0.6080
##    Detection Prevalence : 0.6989
##       Balanced Accuracy : 0.8217
##
##        'Positive' Class : No
##
```

```
#Variable importance
varImp(RFmodel, scale = FALSE)
```

```
## rf variable importance
##
##               Overall
## plasma_CA19_9  64.402
## LYVE1          31.477
## TFF1           24.120
## REG1B          19.465
## REG1A          15.861
## age            15.216
## creatinine     12.641
## sexM            1.721
```

Naive Bayes model

```
#Naive Bayes
set.seed(100)
NBmodel <- train(diagnosis~., data = train,
                 method = "nb",
                 metric = "ROC",
                 trControl = ctrl)

NBmodel$results[NBmodel$results$usekernel == NBmodel$bestTune[1,2] &
                NBmodel$results$fL == NBmodel$bestTune[1,1] &
                NBmodel$results$adjust == NBmodel$bestTune[1,3], c(4:6)]
```

```
##         ROC      Sens      Spec
## 2 0.9289494 0.9232804 0.7142857

#Test set performance
suppressWarnings({NBpred <- predict(NBmodel, newdata = test)})
confusionMatrix(NBpred, test$diagnosis)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  100  25
##        Yes  17  34
##
##                Accuracy : 0.7614
##                  95% CI : (0.6914, 0.8223)
##     No Information Rate : 0.6648
##     P-Value [Acc > NIR] : 0.003476
##
##                   Kappa : 0.446
##
##  Mcnemar's Test P-Value : 0.280087
##
##             Sensitivity : 0.8547
##             Specificity : 0.5763
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.6667
##              Prevalence : 0.6648
##          Detection Rate : 0.5682
##    Detection Prevalence : 0.7102
##       Balanced Accuracy : 0.7155
##
##        'Positive' Class : No
##
```

NNet Model

```
#Neural Network Model
library(nnet)

set.seed(800)


nnetgrid <- expand.grid(.size = 1:10,
                        .decay = c(0, .1, 1, 2))
maxsize <- max(nnetgrid$.size)
numwts <- 1*(maxsize * (length(train) +1) + maxsize + 1)
nnetmodel <- train(diagnosis~., data = train,
                   method = "nnet",
                   metric = "ROC",
                   tuneGrid = nnetgrid,
```

```
                  trace = FALSE,
                  maxit = 200,
                  MaxNWts = numwts,
                  trControl = ctrl)

nnetmodel$results[nnetmodel$results$size == nnetmodel$bestTune[1,1] &
                  nnetmodel$results$decay == nnetmodel$bestTune[1,2], c(3:5)]

##          ROC      Sens      Spec
## 19 0.9301115 0.8800265 0.8071429

#Test set performance
confusionMatrix(predict(nnetmodel, test), test$diagnosis)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  104  20
##        Yes  13  39
##
##                Accuracy : 0.8125
##                  95% CI : (0.7469, 0.8673)
##     No Information Rate : 0.6648
##     P-Value [Acc > NIR] : 1.008e-05
##
##                   Kappa : 0.5666
##
##  Mcnemar's Test P-Value : 0.2963
##
##             Sensitivity : 0.8889
##             Specificity : 0.6610
##          Pos Pred Value : 0.8387
##          Neg Pred Value : 0.7500
##              Prevalence : 0.6648
##          Detection Rate : 0.5909
##    Detection Prevalence : 0.7045
##       Balanced Accuracy : 0.7750
##
##        'Positive' Class : No
##
```

SVM model

```
#SVM model
set.seed(800)
svm_fit <- train(diagnosis~., data = train,
                 method = "svmRadial",
                 tuneLength = 10,
                 preProcess = c("center", "scale"),
                 trControl = ctrl)
```

```
svm_fit$results[svm_fit$results$sigma == svm_fit$bestTune[1,1] &
                svm_fit$results$C == svm_fit$bestTune[1,2], c(3:5)]

##         ROC     Sens      Spec
## 3 0.9071429 0.901455 0.7214286

#Test set performance
confusionMatrix(predict(svm_fit, test), test$diagnosis)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  93  19
##        Yes 24  40
##
##                Accuracy : 0.7557
##                  95% CI : (0.6853, 0.8172)
##     No Information Rate : 0.6648
##     P-Value [Acc > NIR] : 0.005704
##
##                   Kappa : 0.4631
##
##  Mcnemar's Test P-Value : 0.541866
##
##             Sensitivity : 0.7949
##             Specificity : 0.6780
##          Pos Pred Value : 0.8304
##          Neg Pred Value : 0.6250
##              Prevalence : 0.6648
##          Detection Rate : 0.5284
##    Detection Prevalence : 0.6364
##       Balanced Accuracy : 0.7364
##
##        'Positive' Class : No
##
```

XGBoost Tree

```
set.seed(77)
xgboost_fit <- train(diagnosis ~., data = train,
                     method = "xgbTree",
                     tuneLength = 5,
                     metric = "ROC",
                     trControl = ctrl)

xgboost_fit$results[xgboost_fit$results$nrounds == xgboost_fit$bestTune[1,1]
&
                    xgboost_fit$results$max_depth == xgboost_fit$bestTune[1,2
] &
```

```
                    xgboost_fit$results$eta == xgboost_fit$bestTune[1,3] &
                    xgboost_fit$results$gamma == xgboost_fit$bestTune[1,4] &
                    xgboost_fit$results$colsample_bytree == xgboost_fit$bestT
une[1,5] &
                    xgboost_fit$results$min_child_weight == xgboost_fit$bestT
une[1,6] &
                    xgboost_fit$results$subsample == xgboost_fit$bestTune[1,7
], c(8:10)]
```

```
##          ROC      Sens      Spec
## 11 0.9460506 0.9125661 0.8142857
```

```
#Performance on test set
confusionMatrix(predict(xgboost_fit, test), test$diagnosis)
```

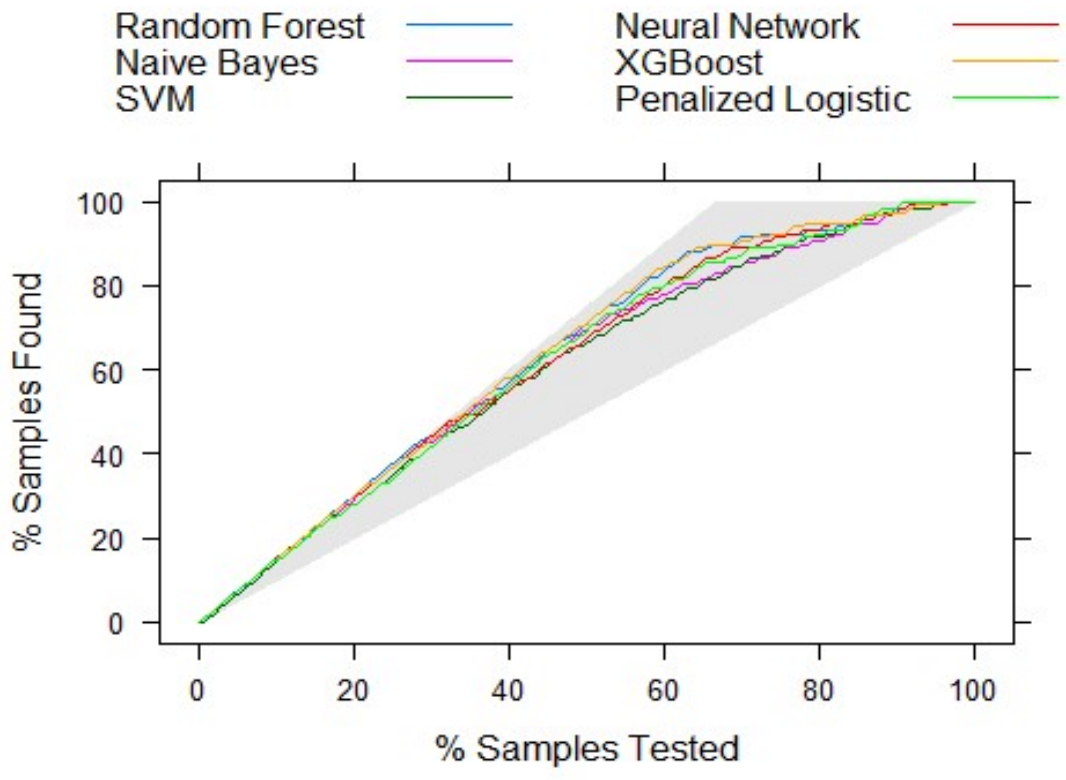
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  105  16
##        Yes  12  43
##
##                Accuracy : 0.8409
##                  95% CI : (0.7783, 0.8916)
##     No Information Rate : 0.6648
##     P-Value [Acc > NIR] : 1.165e-07
##
##                   Kappa : 0.637
##
##  Mcnemar's Test P-Value : 0.5708
##
##             Sensitivity : 0.8974
##             Specificity : 0.7288
##          Pos Pred Value : 0.8678
##          Neg Pred Value : 0.7818
##              Prevalence : 0.6648
##          Detection Rate : 0.5966
##    Detection Prevalence : 0.6875
##       Balanced Accuracy : 0.8131
##
##        'Positive' Class : No
##
```

Penalized Logistic Regression

```
set.seed(77)
pen_fit <- train(diagnosis ~., data = train,
                 method = "glmnet",
                 metric = "ROC",
                 tuneLength = 5,
                 preProcess = c("center", "scale"),
```

```
                trControl = ctrl)

pen_fit$results[pen_fit$results$alpha == pen_fit$bestTune[1,1] &
                pen_fit$results$lambda == pen_fit$bestTune[1,2], c(3:5)]

##          ROC      Sens      Spec
## 21 0.9115457 0.9305556 0.7071429

#Performance on test set
confusionMatrix(predict(pen_fit, test), test$diagnosis)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  101  20
##        Yes  16  39
##
##                Accuracy : 0.7955
##                  95% CI : (0.7282, 0.8524)
##     No Information Rate : 0.6648
##     P-Value [Acc > NIR] : 9.466e-05
##
##                   Kappa : 0.5332
##
##  Mcnemar's Test P-Value : 0.6171
##
##             Sensitivity : 0.8632
##             Specificity : 0.6610
##          Pos Pred Value : 0.8347
##          Neg Pred Value : 0.7091
##              Prevalence : 0.6648
##          Detection Rate : 0.5739
##    Detection Prevalence : 0.6875
##       Balanced Accuracy : 0.7621
##
##        'Positive' Class : No
##

predictions <- subset(test, select = c("diagnosis"))
RFpreds <- predict(RFmodel, test, type = "prob")
predictions$RFprob <- RFpreds[,"No"]
predictions$RFclass <- predict(RFmodel, test)

NBpreds <- predict(NBmodel, test, type = "prob")
predictions$NBprob <- NBpreds[,"No"]
predictions$NBclass <- predict(NBmodel, test)

nnetpreds <- predict(nnetmodel, test, type = "prob")
predictions$nnetprob <- nnetpreds[,"No"]
predictions$nnetclass <- predict(nnetmodel, test)
```

```
svmpreds <- predict(svm_fit, test, type = "prob")
predictions$svmprob <- svmpreds[,"No"]
predictions$svmclass <- predict(svm_fit, test)

xgbpreds <- predict(xgboost_fit, test, type = "prob")
predictions$xgbprob <- xgbpreds[,"No"]
predictions$xgbclass <- predict(xgboost_fit, test)

penpreds <- predict(pen_fit, test, type = "prob")
predictions$penprob <- penpreds[,"No"]
predictions$penclass <- predict(pen_fit, test)

predictions$diagnosis <- as.factor(predictions$diagnosis)
labs <- c(RFprob = "Random Forest",
          NBprob = "Naive Bayes",
          svmprob = "SVM",
          nnetprob = "Neural Network",
          xgbprob = "XGBoost",
          penprob = "Penalized Logistic")
liftCurve <- lift(diagnosis ~ RFprob + NBprob + svmprob + nnetprob + xgbprob
+ penprob, data = predictions, labels = labs)
xyplot(liftCurve, auto.key = list(columns = 2, lines = TRUE, points = FALSE))
```



Appendix B

```
library(shiny)
library(shinythemes)
library(dplyr)
library(readr)
library(randomForest)
library(data.table)
library(caret)
library(shinyWidgets)


pancreatic <- fread("C:/Users/Ladybug/Desktop/pancreatic.csv")[,2:10]

#Split the data into a train and a test set
samples <- createDataPartition(pancreatic$diagnosis, p = 0.7, list = FALSE)[,1]
train <- pancreatic[samples,]
test <- pancreatic[-samples,]

ctrl <- trainControl(summaryFunction = twoClassSummary,
                     method = "cv",
                     classProbs = TRUE)

set.seed(100)
rf_fit <- train(diagnosis~., data = train,
                method = "rf",
                metric = "ROC",
                trControl = ctrl)

# Define UI for application that draws a histogram
ui <- fluidPage(theme = shinytheme("flatly"),
                setBackgroundColor(color = c("lightblue", "ghostwhite"), gradient = "linear", direction = "bottom"),

    # Application title
    titlePanel("Pancreatic Cancer"),

    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            numericInput(inputId = "age", label = "Patient Age", value = 18, min = NA, max = NA, step = NA),
            selectInput(inputId = "sex", label = "Sex", choices = c("M", "F")),
            numericInput(inputId = "plasma_CA19_9", label = "Plasma CA19-9", value = 15000, min = 0, max = 31000, step = NA),
            numericInput(inputId = "creatinine", label = "Creatinine Level:", value = 2.5, min = 0, max = 5, step = NA),
            numericInput(inputId = "LYVE1", label = "LYVE1 Level:", value = 12, min = 0, max = 25, step = NA),
            numericInput(inputId = "REG1B", "REG1B Level:", value = 750, min = 0, max = 1500, step = NA),
            numericInput(inputId = "TFF1", "TFF1 Level:", value = 7500, min = 0, max = 15000, step = NA),
            numericInput(inputId = "REG1A", "REG1A Level:", value = 7500, min = 0, max = 15000, step = NA),
            p("Data utilized: Urinary biomarkers for pancreatic cancer."),
        ),
         # Show a plot of the generated distribution
         mainPanel(
             textOutput("diagnosis")
         )
    )
)

# Define server logic required to draw a histogram
server <- function(input, output) {
    data <- reactive({
             req(input$sex)
             data.frame(
                 age = input$age,
                 sex = input$sex,
                 plasma_CA19_9 = input$plasma_CA19_9,
                 creatinine = input$creatinine,
                 LYVE1 = input$LYVE1,
                 REG1B = input$REG1B,
                 TFF1 = input$TFF1,
                 REG1A = input$REG1A
             )}
    )

    pred <- reactive({
        predict(rf_fit, data())
    })

    output$diagnosis <- renderText({
        ifelse(as.numeric(pred()) == 2, "Pancreatic Cancer", "No Cancer")
    })
}

# Run the application
shinyApp(ui = ui, server = server)
```