In [27]:
```
# install this libraries
library(tidyverse)
library(repr)
library(tidymodels)
library(readxl)
library(RCurl)
install.packages("kknn")
library(kknn)
install.packages("GGally")
library(GGally)
options(repr.matrix.max.rows = 6)
library(recipes)
install.packages("themis")
library(themis)
```

```
Updating HTML index of packages in '.Library'

Making 'packages.html' ...
 done

Updating HTML index of packages in '.Library'

Making 'packages.html' ...
 done

Updating HTML index of packages in '.Library'

Making 'packages.html' ...
 done
```

Classification Analysis: Predicting Hepatitis C And Its Progression

Introduction:

Hepatitis C virus (HCV/Hepatitis C) is frequently asymptotic at first, resulting in many patients being unaware of it. Despite the fact that doctors urge that adults over the age of 18 get tested at least once in their lives, many people still do not get a Hepatitis C test until or even after they have symptoms, and there are still many undiagnosed patients. Thus, predicting if a person has HCV using currently accessible laboratory data can help reduce the number of undiagnosed patients. Moreover, Hepatitis C can progress into Fibrosis or Cirrhosis, so being able to predict weather patient has regular Hepatitis C, Fibrosis, or Cirrhosis can add more benefits to our analysis.

Question: Based on data in HCV data set, does the patient have Hepatitis C, and if so, what is the progress of it?

For this study, we will use HCV data from the machine learning repository at UC Irvine. The data set contains laboratory results from patients with Hepatitis C and blood donors (healthy), together with their demographic data.

1. X (Patient ID/No.)
2. Category (diagnosis) (values: '0=Blood Donor', '0s=suspect Blood Donor', '1=Hepatitis', '2=Fibrosis', '3=Cirrhosis')
3. Age (in years)
4. Sex (f,m)
5. ALB - Albumin Blood Test
6. ALP - Alkaline phosphatase)
7. ALT - Alanine Transaminase
8. AST - Aspartate Transaminase
9. BIL - Bilirubin
10. CHE - Acetylcholinesterase
11. CHOL - Cholesterol
12. CREA - Creatinine
13. GGT - Gamma-Glutamyl Transferase
14. PROT - Proteins

In [29]:
```
#reading data
url <- "https://raw.githubusercontent.com/lilitvanyan/DS_Group_Project/a224c6f2fced1023c1047768fc54fe5aecd9c1bb/hcvdat0.csv"
data <- getURL(url)
hcv_data <- read_csv(data)

hcv_data <- hcv_data |>
mutate(Index = ...1) |>
select(-...1)
head(hcv_data)
```

```
New names:
• `` -> `...1`
Rows: 615 Columns: 14
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr  (2): Category, Sex
dbl (12): ...1, Age, ALB, ALP, ALT, AST, BIL, CHE, CHOL, CREA, GGT, PROT

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

A tibble: 6 × 14

Category	Age	Sex	ALB	ALP	ALT	AST	BIL	CHE	CHOL	CREA	GGT	PROT	Index
<chr>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
0=Blood Donor	32	m	38.5	52.5	7.7	22.1	7.5	6.93	3.23	106	12.1	69.0	1
0=Blood Donor	32	m	38.5	70.3	18.0	24.7	3.9	11.17	4.80	74	15.6	76.5	2
0=Blood Donor	32	m	46.9	74.7	36.2	52.6	6.1	8.84	5.20	86	33.2	79.3	3
0=Blood Donor	32	m	43.2	52.0	30.6	22.6	18.9	7.33	4.74	80	33.8	75.7	4
0=Blood Donor	32	m	39.2	74.1	32.6	24.8	9.6	9.15	4.32	76	29.9	68.7	5
0=Blood Donor	32	m	41.6	43.3	18.5	19.7	12.3	9.92	6.05	111	91.0	74.0	6

In [30]:
```
#wrangling data and splitting data set to have training and testing data
set.seed(10)
hcv_wrangled<- hcv_data  |>
    select(Category:PROT)|>
    filter(Category != "0s=suspect Blood Donor")|>#we don't need this category for this analysis
    na.omit()|>
    mutate(Category = as.factor(Category))|>
    mutate(Category = fct_recode(Category,
                            "Blood Donor" = "0=Blood Donor",
                            "Hepatitis"="1=Hepatitis",
                            "Fibrosis"="2=Fibrosis",
                            "Cirrhosis"="3=Cirrhosis"))

hcv_split <- initial_split(hcv_wrangled, prop = 0.75, strata = Category)

hcv_train <- training(hcv_split)
hcv_test <- testing(hcv_split)
head(hcv_train)
head(hcv_test)
```

A tibble: 6 × 13

Category	Age	Sex	ALB	ALP	ALT	AST	BIL	CHE	CHOL	CREA	GGT	PROT
<fct>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
Blood Donor	32	m	38.5	70.3	18.0	24.7	3.9	11.17	4.80	74	15.6	76.5
Blood Donor	32	m	46.9	74.7	36.2	52.6	6.1	8.84	5.20	86	33.2	79.3
Blood Donor	32	m	43.2	52.0	30.6	22.6	18.9	7.33	4.74	80	33.8	75.7
Blood Donor	32	m	41.6	43.3	18.5	19.7	12.3	9.92	6.05	111	91.0	74.0
Blood Donor	32	m	42.2	41.9	35.8	31.1	16.1	5.82	4.60	109	21.5	67.1
Blood Donor	32	m	50.9	65.5	23.2	21.2	6.9	8.69	4.10	83	13.7	71.3

A tibble: 6 × 13

Category	Age	Sex	ALB	ALP	ALT	AST	BIL	CHE	CHOL	CREA	GGT	PROT
<fct>	<dbl>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
Blood Donor	32	m	38.5	52.5	7.7	22.1	7.5	6.93	3.23	106	12.1	69.0
Blood Donor	32	m	39.2	74.1	32.6	24.8	9.6	9.15	4.32	76	29.9	68.7
Blood Donor	32	m	46.3	41.3	17.5	17.8	8.5	7.01	4.79	70	16.9	74.5
Blood Donor	32	m	44.3	52.3	21.7	22.4	17.2	4.15	3.57	78	24.1	75.4
Blood Donor	33	m	41.8	65.0	33.1	38.0	6.6	8.83	4.43	71	24.0	72.7
Blood Donor	34	m	29.0	41.6	29.1	16.1	4.8	6.82	4.03	62	14.5	53.2

In our analysis, it's crucial to check if our data is balanced. To address this, we will sum up the counts for each category in a table and create a visualization to address this question.

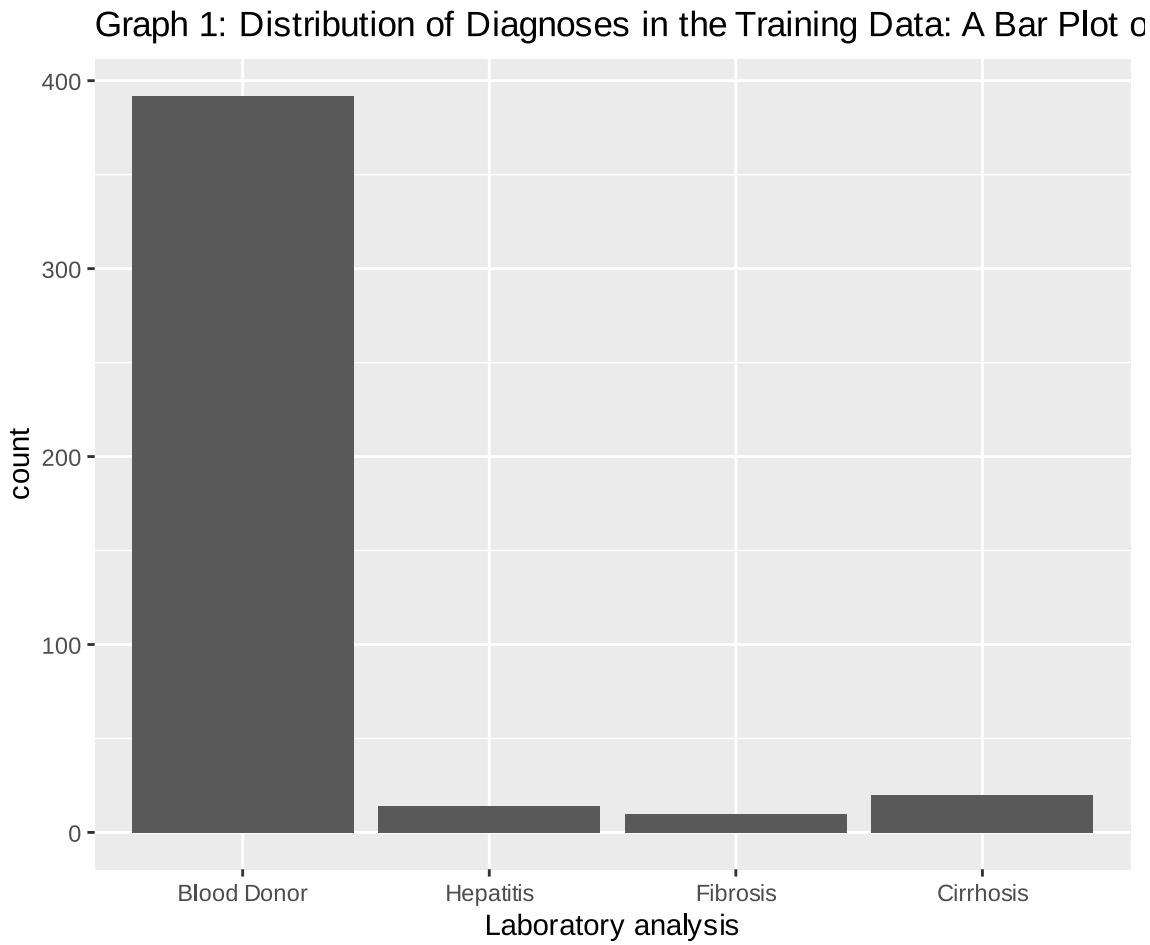
In [31]:
```
hcv_category_summary <- hcv_train |>
summarize(Blood_Donor = nrow((filter(hcv_train, Category == "Blood Donor"))),
          Hepatitis = nrow((filter(hcv_train, Category == "Hepatitis"))),
          Fibrosis = nrow((filter(hcv_train, Category == "Fibrosis"))),
          Cirrhosis = nrow((filter(hcv_train, Category == "Cirrhosis")))) |>
pivot_longer(Blood_Donor:Cirrhosis, names_to = "Diagnose", values_to = "Count")
hcv_category_summary


category_lab_result <- hcv_train |> ggplot(aes(x = Category)) +
                    geom_bar() +
```

```
                  xlab("Laboratory analysis") +
    ggtitle("Graph 1: Distribution of Diagnoses in the Training Data: A Bar Plot of Counts")
category_lab_result
```

A tibble: 4 × 2

Diagnose	Count
<chr>	<int>
Blood_Donor	392
Hepatitis	14
Fibrosis	10
Cirrhosis	20



The data we have is highly unbalanced, as shown in the graph and table above. This fact will be considered during our analysis to ensure that the training data is balanced.

Methods and results:

CHOOSING PREDICTORS

We may now begin our data analysis to answer the initial question. First, we must select appropriate predictors. Our data set includes 12 potential predictors, 10 distinct lab analyses, as well as the patient's gender and age. Let us first look at the lab analyses that are available. Hepatitis C primarily affects the patient's liver, causing it to enlarge. As a result, many patients use their lab results to monitor their liver functioning, thus monitoring their Hepatitis C. Doctors typically employ five types of blood tests to assess a patient's liver condition: Alanine aminotransferase (ALT), Aspartate aminotransferase (AST), Alkaline phosphatase (ALP), Albumin (ALB), and Bilirubin (BIL). In Hepatitis C patients, the first three of these lab tests often show abnormalities, but levels of ALB and BIL typically stay within the normal range. These levels become more relevant in signaling Cirrhosis, the progressive stage of Hepatitis C (Hep, 2023).

As ALT, AST, and ALP are the major indications of any type of Hepatitis C, we will first do cross-validation for multiple sets of ALT, AST, and ALP to select the optimal combination of these three lab analyses in our prediction. Then, we will examine how adding BIL or ALB (or both) to the chosen predictors affects our accuracy. Finally, we will see how adding Sex and Gender affects our accuracy. As a result, we will get the needed combination of predictors.

In [32]:
```
#Doing cross validation for different combinations of ALT, AST, ALP
#we will use step_upsample to balance our training set
options(repr.plot.width = 6, repr.plot.height = 5)
set.seed(10)

hcv_train_smaller <- hcv_train |> select(Category, ALT, AST, ALP)

accuracies <- data.frame( Combination=character(),
                          Accuracy=double(),
                         stringsAsFactors = FALSE)

# ALT + AST
number_recipe1 <- recipe(Category ~ ALT + AST, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()

knn_spec <- nearest_neighbor(weight_func = "rectangular", neighbors = tune()) |>
                        set_engine("kknn") |>
                        set_mode("classification")

number_vfold <- vfold_cv(hcv_train_smaller, v = 5, strata = Category)
```

```r
xvals <- tibble(neighbors = seq(from = 2, to = 20))

number_metrics <- workflow() |>
                  add_recipe(number_recipe1) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()

accuracy1 <- number_metrics|>
    filter(.metric == "accuracy")

acc1<-summarise(accuracy1, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT+AST", Accuracy = acc1)


#ALT + ALP

number_recipe2 <- recipe(Category ~ ALT + ALP, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()


number_metrics <- workflow() |>
                  add_recipe(number_recipe2) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()


acc2 <- number_metrics |>
    filter(.metric == "accuracy")

accuracy2<-summarise(acc2, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT+ALP", Accuracy = accuracy2)


#AST + ALP

number_recipe3 <- recipe(Category ~ AST + ALP, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
```

```r
  prep()


number_metrics <- workflow() |>
                  add_recipe(number_recipe3) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()

acc3 <- number_metrics |>
    filter(.metric == "accuracy")

acc3<-summarise(acc3, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "AST+ALP", Accuracy = acc3)

#AST + ALP + ALT

number_recipe4 <- recipe(Category ~ AST + ALP + ALT, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()


number_metrics <- workflow() |>
                  add_recipe(number_recipe4) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()

acc4 <- number_metrics |>
    filter(.metric == "accuracy")

acc4<-summarise(acc4, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT+AST+ALP", Accuracy = acc4)

accuracies<-arrange(accuracies, desc(Accuracy))
accuracies
```

```
→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1



→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1



→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1



→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1
```

A data.frame: 4 × 2

Combination	Accuracy
<chr>	<dbl>
ALT+AST	0.9496865
ALT+AST+ALP	0.9450627
AST+ALP	0.9106844
ALT+ALP	0.9014368

The maximum results for each combination of lab analyses for which cross-validation was performed are shown in the table above. As we can see, "ALT+AST" has the highest accuracy, so we will use ALT and AST as our initial chosen predictors. Let us now see if adding BIL or ALB to our chosen predictors improves our accuracy. Similarly, we will perform cross-validation.

In [33]:
```
#Doing cross validation by adding different combination of BIL ALB to chosen predictors ALT AST

options(repr.plot.width = 6, repr.plot.height = 5)
set.seed(10)

hcv_train_smaller <- hcv_train |> select(Category, ALT, AST, BIL, ALB)
accuracies <- filter(accuracies, Combination == "ALT+AST")

# ALT + AST + BIL
number_recipe5 <- recipe(Category ~ ALT + AST + BIL, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()

knn_spec <- nearest_neighbor(weight_func = "rectangular", neighbors = tune()) |>
                        set_engine("kknn") |>
                        set_mode("classification")

number_vfold <- vfold_cv(hcv_train_smaller, v = 5, strata = Category)


xvals <- tibble(neighbors = seq(from = 2, to = 20))

number_metrics <- workflow() |>
                  add_recipe(number_recipe5) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()
```

```r
acc5 <- number_metrics|>
    filter(.metric == "accuracy")

acc5<-summarise(acc5, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT+AST+BIL", Accuracy = acc5)


#ALT + AST + ALB

number_recipe6 <- recipe(Category ~ ALT + AST + ALB, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()


number_metrics <- workflow() |>
                  add_recipe(number_recipe6) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()


acc6 <- number_metrics |>
    filter(.metric == "accuracy")

acc6<-summarise(acc6, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT+AST+ALB", Accuracy = acc6)


#ALT + AST + ALB + BIL

number_recipe7 <- recipe(Category ~ AST + ALT + ALB + BIL, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()


number_metrics <- workflow() |>
                  add_recipe(number_recipe7) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
```

```
                  collect_metrics()

acc7 <- number_metrics |>
    filter(.metric == "accuracy")

acc7<-summarise(acc7, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT + AST + ALB + BIL", Accuracy = acc7)

accuracies<-arrange(accuracies, desc(Accuracy))
accuracies
```

```
→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1



→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1



→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1
```

A data.frame: 4 × 2

Combination	Accuracy
<chr>	<dbl>
ALT+AST	0.9496865
ALT+AST+BIL	0.9496604
ALT+AST+ALB	0.9450366
ALT + AST + ALB + BIL	0.9427638

As we can see, even after adding predictors, we get our highest accuracy when we use ALT and AST as predictors. And finally, let's see how adding sex and/or age will affect our accuracy.

In [34]:
```
set.seed(10)

# mutating df to represent sex as 0 or 1 (binarizing them)
hcv_train_smaller <- hcv_train |> select(Category, ALT, AST, ALP, ALB, BIL, Age, Sex)|>
                        mutate(Sex = ifelse(Sex == "m", "0", "1"))|>
                        mutate(Sex=as.double(Sex))

accuracies <- filter(accuracies, Combination == "ALT+AST")

#ALT+AST+Age+Sex
number_recipe8 <- recipe(Category ~ ALT+AST+Age+Sex, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE)|>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()

knn_spec <- nearest_neighbor(weight_func = "rectangular", neighbors = tune()) |>
                        set_engine("kknn") |>
                        set_mode("classification")
number_vfold <- vfold_cv(hcv_train_smaller, v = 5, strata = Category)


xvals <- tibble(neighbors = seq(from = 2, to = 6))

number_metrics <- workflow() |>
                  add_recipe(number_recipe8) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()

acc8 <- number_metrics |> filter(.metric == "accuracy")
```

```r
acc8<-summarise(acc8, max=max(mean))|>
    pull(max)

accuracies<-add_row(accuracies, Combination = "ALT+AST+Age+Sex", Accuracy = acc8)

#ALT+AST+Age

number_recipe9 <- recipe(Category ~ ALT+AST+Age, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE)|>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()

knn_spec <- nearest_neighbor(weight_func = "rectangular", neighbors = tune()) |>
                        set_engine("kknn") |>
                        set_mode("classification")
number_vfold <- vfold_cv(hcv_train_smaller, v = 5, strata = Category)


xvals <- tibble(neighbors = seq(from = 2, to = 6))

number_metrics <- workflow() |>
                  add_recipe(number_recipe9) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()

acc9 <- number_metrics |> filter(.metric == "accuracy")

acc9 <- summarise(acc9, max=max(mean))|>
    pull(max)

accuracies <- add_row(accuracies, Combination = "ALT+AST+Age", Accuracy = acc9)

#ALT+AST+Sex

number_recipe10 <- recipe(Category ~ ALT+AST+Sex, data = hcv_train_smaller) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE)|>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()

knn_spec <- nearest_neighbor(weight_func = "rectangular", neighbors = tune()) |>
                        set_engine("kknn") |>
                        set_mode("classification")
number_vfold <- vfold_cv(hcv_train_smaller, v = 5, strata = Category)
```

```r
xvals <- tibble(neighbors = seq(from = 2, to = 6))

number_metrics <- workflow() |>
                  add_recipe(number_recipe9) |>
                  add_model(knn_spec) |>
                  tune_grid(resamples = number_vfold, grid = xvals) |>
                  collect_metrics()

acc10 <- number_metrics |> filter(.metric == "accuracy")

acc10 <- summarise(acc10, max=max(mean))|>
    pull(max)

accuracies <- add_row(accuracies, Combination = "ALT+AST+Age", Accuracy = acc10)

accuracies
```

```
→ A | warning: No observations were detected in `truth` for level(s): 'Hepatitis'
               Computation will proceed by ignoring those levels.

→ A | warning: No observations were detected in `truth` for level(s): 'Fibrosis'
               Computation will proceed by ignoring those levels.

There were issues with some computations   A: x1

There were issues with some computations   A: x1



→ A | warning: No observations were detected in `truth` for level(s): 'Fibrosis'
               Computation will proceed by ignoring those levels.
```

A data.frame: 4 × 2

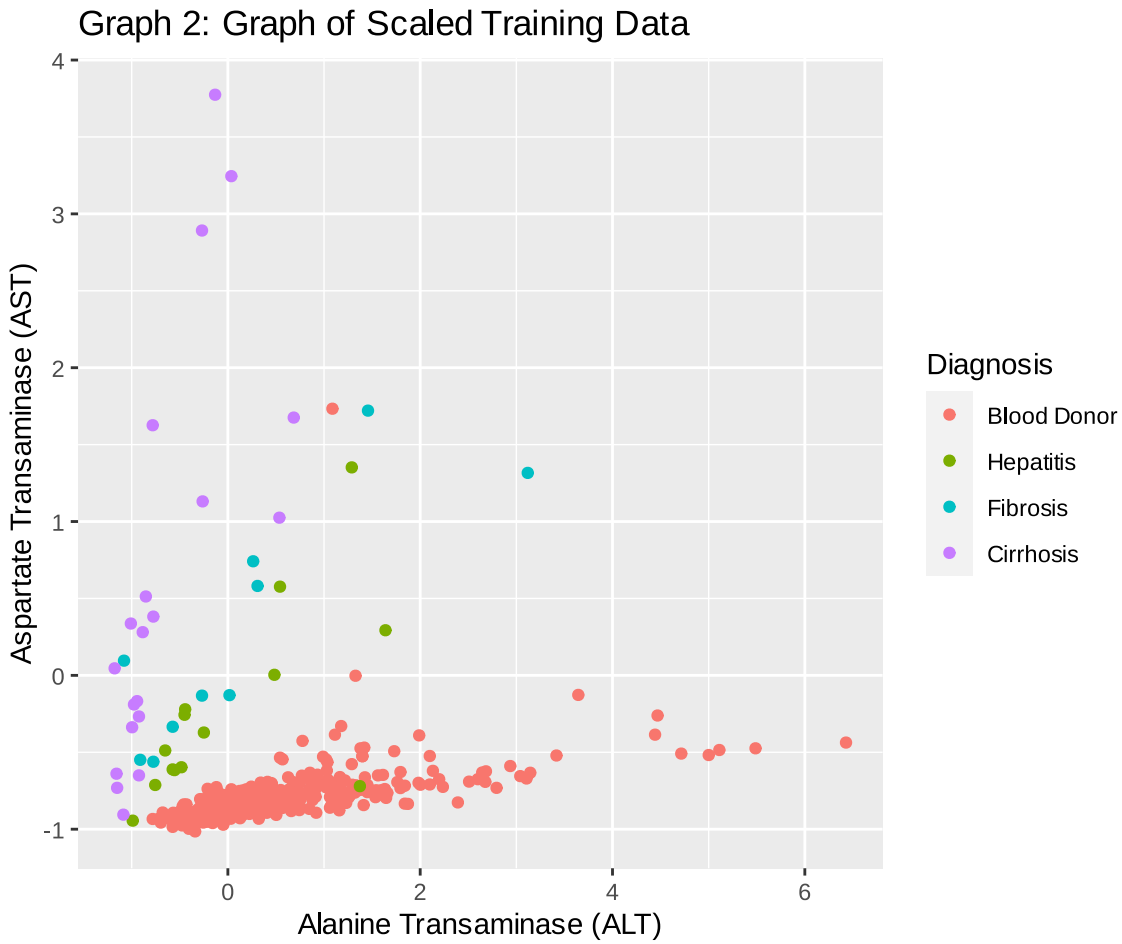
Combination	Accuracy
<chr>	<dbl>
ALT+AST	0.9496865
ALT+AST+Age+Sex	0.9405172
ALT+AST+Age	0.9449582
ALT+AST+Age	0.9472309

As we can see in the table above, adding the predictors Age or Sex did not improve our accuracy. As a result, we will be using two predictors in our analysis.

CHOSEN PREDICTORS: ALT, AST

In [45]:
```
# visualization of the predictors
# this graph will be also used later in our discussion part

hcv_data_scaled <- number_recipe1|>
                   prep() |>
                   bake(select(hcv_train_smaller, Category, ALT, AST))
data_plot <- hcv_data_scaled |>
ggplot(aes(x = ALT, y = AST, colour = Category)) +
geom_point() +
xlab("Alanine Transaminase (ALT)") +
ylab("Aspartate Transaminase (AST)")+
labs(colour = 'Diagnosis') +
ggtitle("Graph 2: Graph of Scaled Training Data")
data_plot
```



Our next step will be determining the number of neighbors that will be used in our classification. Again, cross-validation can be used to select the number of neighbors. We have previously done cross-validation for the predictor sets ALT and AST in the process of selecting predictors, so there is no need to repeat this step, and we can simply look at the findings.

In [36]:
```
accuracy1 #contains metrics from cross validation we need
```

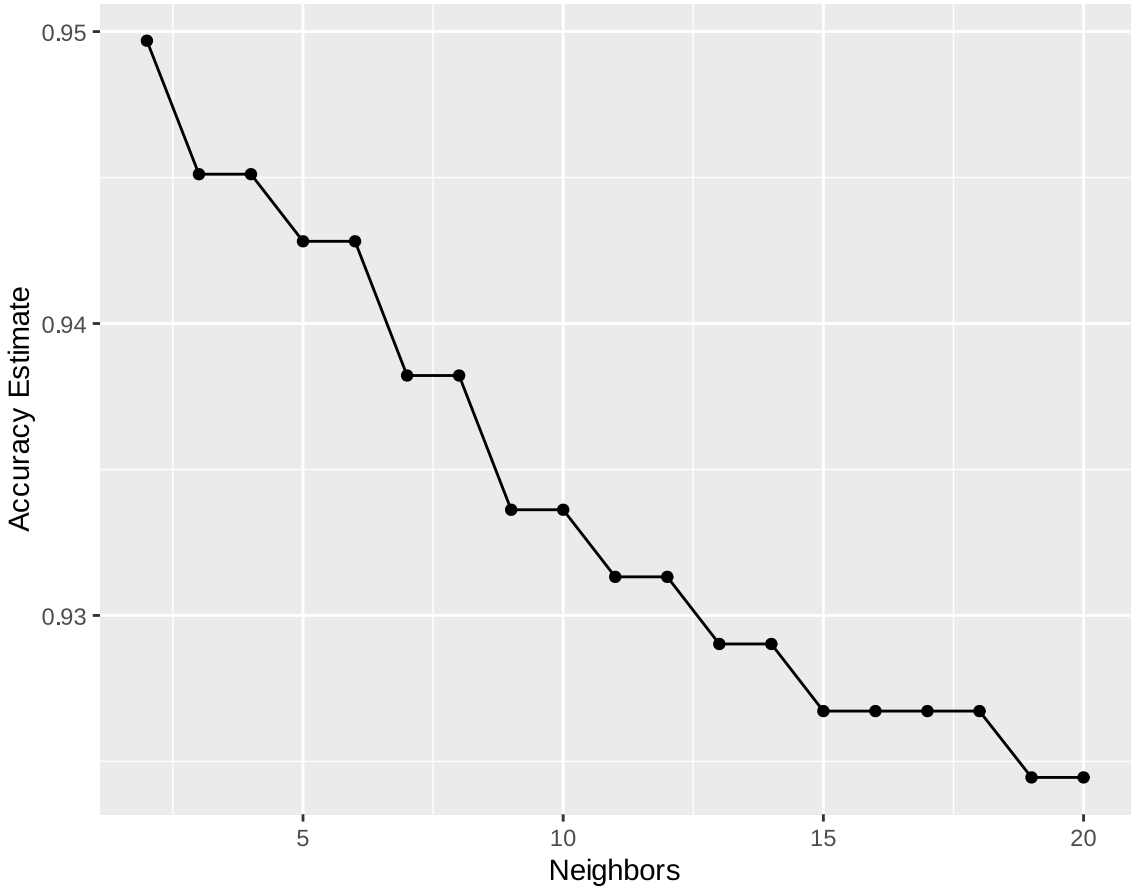
A tibble: 19 × 7

neighbors	.metric	.estimator	mean	n	std_err	.config
<int>	<chr>	<chr>	<dbl>	<int>	<dbl>	<chr>
2	accuracy	multiclass	0.9496865	5	0.01703664	Preprocessor1_Model01
3	accuracy	multiclass	0.9451149	5	0.01910649	Preprocessor1_Model02
4	accuracy	multiclass	0.9451149	5	0.01910649	Preprocessor1_Model03
⋮	⋮	⋮	⋮	⋮	⋮	⋮
18	accuracy	multiclass	0.9267241	5	0.01744374	Preprocessor1_Model17
19	accuracy	multiclass	0.9244514	5	0.01918923	Preprocessor1_Model18
20	accuracy	multiclass	0.9244514	5	0.01918923	Preprocessor1_Model19

In [37]:
```
#Using plot to visualise the data from the table above

cross_val_plot <- ggplot(accuracy1, aes(x = neighbors, y = mean)) +
      geom_point() +
      geom_line() +
      labs(x = "Neighbors", y = "Accuracy Estimate")+
      ggtitle("Graph 3: Estimated accuracy versus the number of neighbors")
cross_val_plot
```

Graph 3: Estimated accuracy versus the number of neighbors



As we can see from the table and graph above, our model reaches its maximum accuracy estimate when the number of neighbors is 2. So now let's make our model for N = 2.

In [38]:
```
number_recipe_0 <- recipe(Category ~ ALT + AST, data = hcv_train) |>
  step_upsample(Category, over_ratio = 1, skip = TRUE) |>
  step_scale(all_predictors())|>
  step_center(all_predictors())|>
  prep()

knn_spec_0 <- nearest_neighbor(weight_func = "rectangular", neighbors = 2) |>
  set_engine("kknn") |>
  set_mode("classification")

knn_fit <- workflow() |>
  add_recipe(number_recipe_0) |>
  add_model(knn_spec_0) |>
  fit(data = hcv_train)
```

```
knn_fit
```

```
══ Workflow [trained] ══════════════════════════════════════════════════════════
Preprocessor: Recipe
Model: nearest_neighbor()

── Preprocessor ────────────────────────────────────────────────────────────────
3 Recipe Steps

• step_upsample()
• step_scale()
• step_center()

── Model ───────────────────────────────────────────────────────────────────────

Call:
kknn::train.kknn(formula = ..y ~ ., data = data, ks = min_rows(2,     data, 5), kernel = ~"rectangular")

Type of response variable: nominal
Minimal misclassification: 0.00127551
Best kernel: rectangular
Best k: 2
```

Finally, let's use our model to predict the testing set. Let's also make a confusion matrix, the results of which will be discussed later.

In [39]:
```
prediction <- predict(knn_fit, hcv_test) |>
              bind_cols(hcv_test)

conf_mat <- prediction |>
  conf_mat(truth = Category, estimate = .pred_class)

conf_mat

acc <- prediction |>
  metrics(truth = Category, estimate = .pred_class) |>
  filter(.metric == "accuracy") |>
  select(.estimate) |>
  pull()
```

```
            Truth
Prediction    Blood Donor Hepatitis Fibrosis Cirrhosis
  Blood Donor         133         0        0         0
  Hepatitis             1         2        2         0
  Fibrosis              0         3        0         2
  Cirrhosis             0         1        0         2
```

In [40]:
```
#models accuracy
acc
```

0.938356164383562

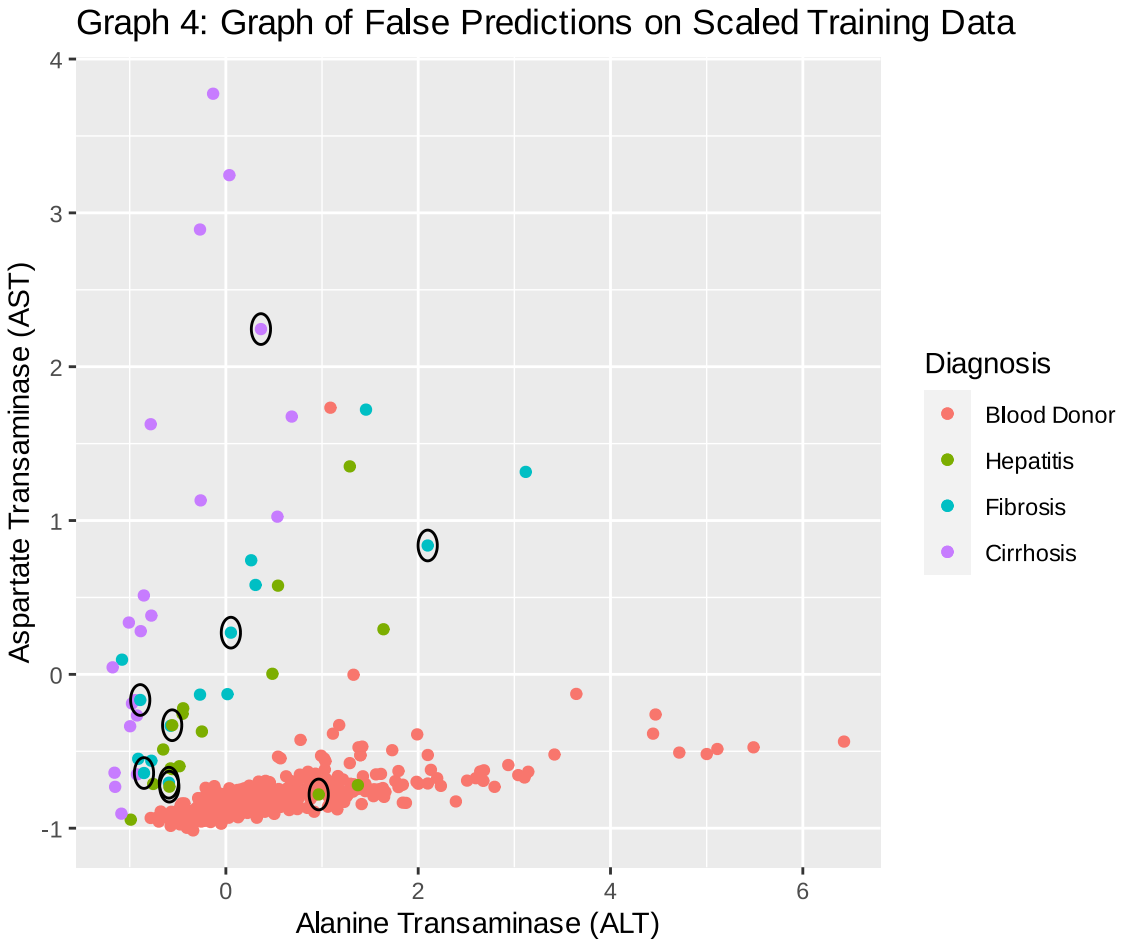
Let's use visualizations to better examine our findings.

In [41]:
```
false_predictions <- prediction |>
filter(.pred_class != Category) |>
select(.pred_class, Category, ALT, AST)


false_predictions_scaled <- recipe(.pred_class ~ ALT + AST, data = false_predictions) |>
                    step_scale(all_predictors()) |>
                    step_center(all_predictors()) |>
                    prep() |>
                    bake(false_predictions)
```

In [42]:
```
data_plot +
geom_point(data = false_predictions_scaled, aes(x = ALT, y = AST, color = .pred_class)) +
annotate("path",
                  x= 0.96554100 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y=-0.7806378 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= 2.09878981 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= 0.8377493 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= 0.05210339 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= 0.2711074 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= 0.36472375 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= 2.2445488 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= -0.59267610 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= -0.7042599 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= -0.55848325 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= -0.3306271 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= -0.58779140 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= -0.7300632 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= -0.89064238 + 0.1*cos(seq(0,2*pi,length.out=100)),
                  y= -0.1665177 + 0.1*sin(seq(0,2*pi,length.out=100))) +
annotate("path",
                  x= -0.85156483 + 0.1*cos(seq(0,2*pi,length.out=100)),
```

```
                y= -0.6412997 + 0.1*sin(seq(0,2*pi,length.out=100))) +
ggtitle("Graph 4: Graph of False Predictions on Scaled Training Data")
```



The graph above depicts false predictions from predicting diagnoses for our testing data, plotted over the training data's scatter plot. It is done to illustrate the possible reasons why the individuals from the testing data set were incorrectly predicted by the model created from our training data set. False predictions are indicated by black circles. As shown in the graph, the points that are incorrectly predicted can be found near multiple points of different diagnoses. Even though we saw that the highest accuracy is obtained when K is 2, for some points in our testing data, the nearest two neighbors have different diagnoses than the true diagnosis of our point, causing them to be incorrectly predicted. More specifically, we will discuss the reason behind the false predictions in the discussion part.

Discussion:

Now let's discuss our findings. The model we developed has 94% accuracy. In order to select the predictors, we performed cross validation on various sets of predictors, compared them, and selected our predictors based on the best accuracy obtained from cross validation, which was 95%. So, when we predicted the testing data, we expected to achieve close accuracy. But is it good accuracy? Because we did our best to select the best

predictors, the accuracy we obtained was really high for the data that we had. According to the table below, blood donors accounted for 91.7% of our available testing data. This means that our model is more accurate than simply guessing the Blood Donor each time.

In [43]:
```
count<-nrow(hcv_test)
result_dplyr <- group_by(hcv_test,Category)|>
  summarise(Percentage = length(Category)/count * 100)
result_dplyr

conf_mat
```

A tibble: 4 × 2

Category	Percentage
<fct>	<dbl>
Blood Donor	91.780822
Hepatitis	4.109589
Fibrosis	1.369863
Cirrhosis	2.739726

```
             Truth
Prediction    Blood Donor Hepatitis Fibrosis Cirrhosis
  Blood Donor         133         0        0         0
  Hepatitis             1         2        2         0
  Fibrosis              0         3        0         2
  Cirrhosis             0         1        0         2
```

Let's have a look at the confusion matrix. We can observe that there isn't a single patient who was diagnosed as a blood donor (healthy) while having Hepatitis C at any stage. Our model faced challenges in accurately identifying the Hepatitis C stage. Notably, three patients diagnosed with Hepatitis were mistakenly identified as having Fibrosis, two individuals with Fibrosis were misclassified as having Hepatitis, and two individuals with Cirrhosis were incorrectly labeled as having Fibrosis. The root of these issues becomes apparent when examining Figure 3. It is evident that the model excelled in predicting Blood donors (depicted by red points) due to their distinct clustering, facilitating precise predictions. Conversely, the predictions for other categories were more problematic, given the less distinct clustering of points in various colors, particularly complicating the accurate identification of Hepatitis C stages. So we can say that this model is good for forecasting Hepatitis C in general for patients, but it isn't reliable for determining the progression of it. Based on the findings, doctors may utilize the model as a helper in determining whether a patient has Hepatitis C in general, but if the model detects it, doctors need to perform further tests to accurately establish the stage of the disease.

Adding new variables to our model could improve its accuracy, especially when it comes to diagnosing the progression of the disease. For example, specialized blood tests such as ELF, FibroMeter, or Fibrotest are used to diagnose Fibrosis (Clémence M. Canivet, Jérôme Boursier, 2022). Having this data could potentially improve the accuracy of our model.

References

Hep (2023). Monitoring your Hepatitis C. https://www.hepmag.com/basics/liver-health/hepatitis-c-lab-tests.

Canivet CM, Boursier J. Screening for Liver Fibrosis in the General Population: Where Do We Stand in 2022? Diagnostics (Basel). 2022 Dec 28;13(1):91. doi: 10.3390/diagnostics13010091. PMID: 36611384; PMCID: PMC9818643.

Lichtinghagen,Ralf, Klawonn,Frank, and Hoffmann,Georg. (2020). HCV data. UCI Machine Learning Repository. https://doi.org/10.24432/C5D612.