Joseph\_Davis\_Final

Joseph Davis

2023-06-18

**Introduction**

The United States has the world’s largest economy by nominal GDP and is a highly complex mixed market. Understanding the whole of the economy is both extremely useful for economists and wildly impractical. The economy of a country is also a reflection of the quality of life for its citizens.

A recession is characterized by an extended downturn in economic activity. A recession occurs over the course of a few months; however, the implications are far reaching and last many years, affecting the job market, stock prices, goods/services production and a whole host of other things vital to a healthy country. A general rule of thumb for determining if there is a recession is 2 quarters in a row with a negative change in GDP. Meaning the gross domestic product decreases 2 quarters in a row. Gross domestic product is the whole of final goods/services produced by a country over a given time period.

By investigating economic metrics, recessions may be able to be predicted and potentially prevented with the correct solutions in place. The goal is to understand what factors specifically can be used to help predict a recession. Additionally, creating models to predict when a recession is coming could be a useful tool for civilians to be better prepared for future economic turmoil.

**Analysis and Models**

The data used for this project comes from Kaggle user MIKOLAJFISH99 and access to the data will be included in the appendix. This dataset consisted of 26 different CSV files all containing the date and some corresponding economic metric. The final metrics used for this project were; gross domestic product, real gross domestic product, unemployment rate, personal savings rate, median consumer price index, delinquency rate on credit card loans, household owners’ equity, and commercial real estate prices. For clarification median consumer price index can be thought of as a measure of core inflation, and household owners’ equity is the amount of the real estate market that is owned by residents. All other metrics are straight forward.

This process began by reading in the Gross Domestic Product and real gross domestic product. The change in GDP was calculated by creating a lag column and subtracting this column from the current gdp, giving a change in gdp quarter over quarter.

Gross\_Domestic\_Product <- read\_csv("Definitely Useful/quarterly/Gross Domestic Product.csv")

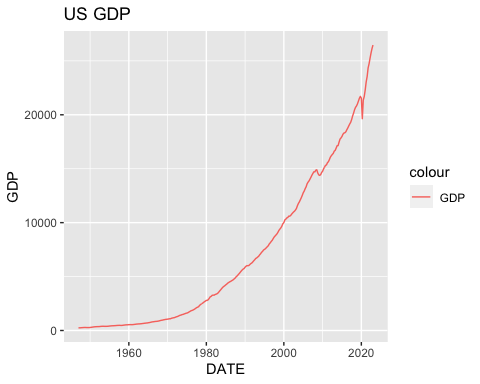
Gross\_Domestic\_Product$change <- c(diff(Gross\_Domestic\_Product$GDP, lag=1), 0)  
Gross\_Domestic\_Product$recession <- Gross\_Domestic\_Product$change  
Gross\_Domestic\_Product$recession <- replace(Gross\_Domestic\_Product$recession, Gross\_Domestic\_Product$recession>0, 0)  
Gross\_Domestic\_Product$recession <- replace(Gross\_Domestic\_Product$recession, Gross\_Domestic\_Product$recession<0, 1)  
RecessionYears <- Gross\_Domestic\_Product[Gross\_Domestic\_Product$recession==1,]  
RecessionYears <- RecessionYears[-3,]  
RecessionYears <- RecessionYears[-8:-12,]  
RecessionYears <- RecessionYears[-8,]  
Real\_Gross\_Domestic\_Product <- read\_csv("Definitely Useful/quarterly/Real Gross Domestic Product.csv")

## Rows: 305 Columns: 2

Real\_Gross\_Domestic\_Product$change <- c(diff(Real\_Gross\_Domestic\_Product$GDPC1, lag=1), 0)  
Real\_Gross\_Domestic\_Product$recession <- Real\_Gross\_Domestic\_Product$change  
Real\_Gross\_Domestic\_Product$recession <- replace(Real\_Gross\_Domestic\_Product$recession, Real\_Gross\_Domestic\_Product$recession>0, 0)  
Real\_Gross\_Domestic\_Product$recession <- replace(Real\_Gross\_Domestic\_Product$recession, Real\_Gross\_Domestic\_Product$recession<0, 1)

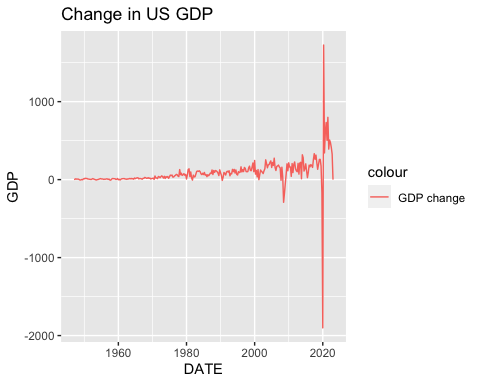
To better understand the trend of GDP and real GDP over time they were quickly plotted vs. the date. Additionally, the change in GDP was plotted to visualize the increase in volatility over time.

#Plotting for GDP visualization  
ggplot(Gross\_Domestic\_Product, aes(x=DATE)) +   
 geom\_line(aes(y=GDP, color="GDP")) +   
 ylab("GDP") + ggtitle("US GDP")



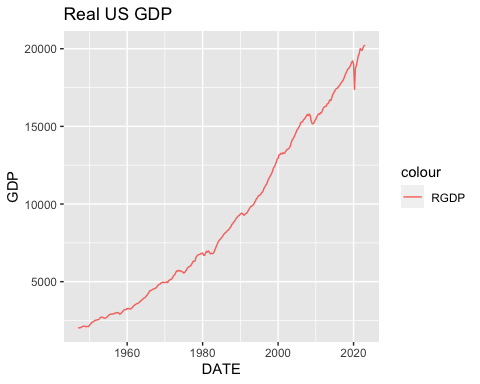
*Figure 1.* Gross Domestic Product Over Time

ggplot(Gross\_Domestic\_Product, aes(x=DATE)) +   
 geom\_line(aes(y=change, color="GDP change")) +   
 ylab("GDP") + ggtitle("Change in US GDP")



*Figure 2.* Change in Gross Domestic Product over Time

ggplot(Real\_Gross\_Domestic\_Product, aes(x=DATE)) +   
 geom\_line(aes(y=GDPC1, color="RGDP")) +   
 ylab("GDP") + ggtitle("Real US GDP")



*Figure 3*. Real Domestic Product over Time

Years of recession were calculated by seeing which columns had a negative change in GDP. When 2 of these rows were negative in a row it was considered a recession. Real\_RecessionYears is the culmination of all quarters where a recession was present.

Real\_RecessionYears <- Real\_Gross\_Domestic\_Product[Real\_Gross\_Domestic\_Product$recession==1,]  
Real\_RecessionYears <- Real\_RecessionYears[-5,]  
Real\_RecessionYears <- Real\_RecessionYears[-8:-10,]  
Real\_RecessionYears <- Real\_RecessionYears[-10:-11,]  
Real\_RecessionYears <- Real\_RecessionYears[-12:-14,]  
Real\_RecessionYears <- Real\_RecessionYears[-17,]  
Real\_RecessionYears <- Real\_RecessionYears[-19,]  
Real\_RecessionYears <- Real\_RecessionYears[-21:-23,]  
Real\_RecessionYears <- Real\_RecessionYears[-25:-27,]  
Real\_RecessionYears <- Real\_RecessionYears[-17,]  
Better\_Dates <- data.table(dates=c(Real\_Gross\_Domestic\_Product$DATE), recession=Real\_Gross\_Domestic\_Product$recession)

To better analyze the data, the recession years dates (quarterly) were changed to monthly.

DT\_month=data.table(Date=as.Date(c(Better\_Dates$dates)),Value=c(Better\_Dates$recession))  
DT\_month[,Month:=month(Date)]  
DT\_month[,Year:=year(Date)]  
start\_date=min(DT\_month$Date)  
end\_date=max(DT\_month$Date)  
DT\_month=data.table(Date=seq.Date(start\_date,end\_date,by="month"))  
DT\_month$recession <- 0  
DT\_month$recession[1:6] <- 1  
DT\_month$recession[22:27] <- 1  
DT\_month$recession[76:84] <- 1  
DT\_month$recession[127:132] <- 1  
DT\_month$recession[328:336] <- 1  
DT\_month$recession[397:402] <- 1  
DT\_month$recession[415:420] <- 1  
DT\_month$recession[523:528] <- 1  
DT\_month$recession[736:747] <- 1  
DT\_month$recession[874:879] <- 1  
DT\_month$recession[898:903] <- 1  
MonthREC <- data.table(dates=as.Date(c(DT\_month$Date)), recession=c(DT\_month$recession))  
MonthREC[,Month:=month(dates)]  
MonthREC[,Year:=year(dates)]  
RecessionMonthly <- MonthREC[,-3:-4]  
head(RecessionMonthly, 10)

## dates recession  
## 1: 1947-01-01 1  
## 2: 1947-02-01 1  
## 3: 1947-03-01 1  
## 4: 1947-04-01 1  
## 5: 1947-05-01 1  
## 6: 1947-06-01 1  
## 7: 1947-07-01 0  
## 8: 1947-08-01 0  
## 9: 1947-09-01 0  
## 10: 1947-10-01 0

Next, different monthly statistics were read in, including median consumer price index, personal savings rate, and unemployment rate. This was all compiled into a single dataframe with the dates/recession times.

MonthlyData <- RecessionMonthly  
Median\_Consumer\_Price\_Index <- read\_csv("Definitely Useful/Monthly/Median Consumer Price Index.csv")

## Rows: 483 Columns: 2

Personal\_Saving\_Rate <- read\_csv("Definitely Useful/Monthly/Personal Saving Rate.csv")

## Rows: 771 Columns: 2

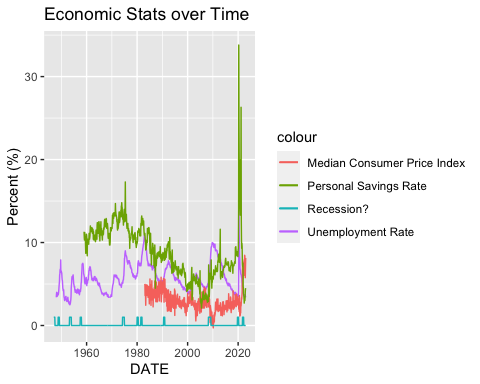
Unemployment\_Rate <- read\_csv("Definitely Useful/Monthly/Unemployment Rate.csv")

## Rows: 904 Columns: 2

Monthly1 <- merge(Unemployment\_Rate, Median\_Consumer\_Price\_Index, by = "DATE",all.x=TRUE)  
Monthly2 <- merge(Monthly1, Personal\_Saving\_Rate, by = "DATE",all.x=TRUE)  
colnames(RecessionMonthly)[1] <- 'DATE'  
MonthlyData <- merge(RecessionMonthly, Monthly2, by="DATE", all.x=TRUE)

To help visualize this data, it was plotted on a scatterplot. All the increases in the cyan line from 0 to 1 indicated a recession.

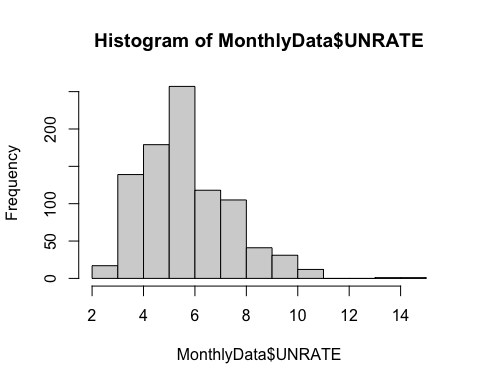
ggplot(MonthlyData, aes(x=DATE)) +   
 geom\_line(aes(y=UNRATE, color="Unemployment Rate")) +   
 geom\_line(aes(y=MEDCPIM158SFRBCLE, color="Median Consumer Price Index")) +   
 geom\_line(aes(y=PSAVERT, color="Personal Savings Rate")) +  
 geom\_line(aes(y=recession,color="Recession?")) +  
 ylab("Percent (%)") + ggtitle("Economic Stats over Time")



*Figure 4.* Recession vs. MCPI, PSR, and Unemployment Rate

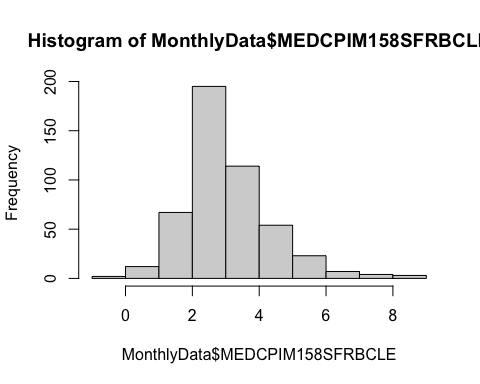
The spread of the different variables was understood by using histograms. The data was discretized with appropriate breaks.

hist(MonthlyData$UNRATE)



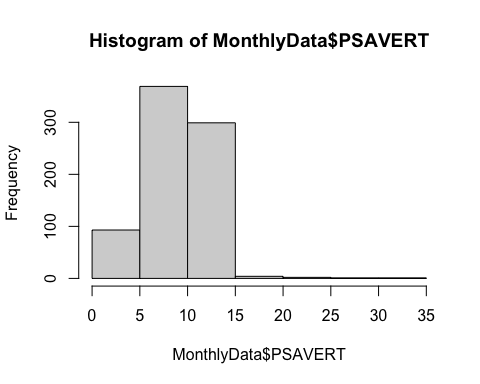
*Figure 5.* Unemployment Rate histogram

hist(MonthlyData$MEDCPIM158SFRBCLE)



*Figure 6.* Median Consumer Price Index histogram

hist(MonthlyData$PSAVERT)



*Figure 7.* Personal Savings Rate histogram

CleanMonthly=subset(MonthlyData, select=-c(DATE))  
CleanMonthly$UNRATE <- cut(CleanMonthly$UNRATE, breaks = c(0, 2, 4, 6, 8, 10,100) , labels=c("0-2", "2-4", "4-6", "6-8", "8-10", "10+"))  
CleanMonthly$MEDCPIM158SFRBCLE <- cut(CleanMonthly$MEDCPIM158SFRBCLE, breaks = c(-2, 0, 2, 4, 6, 8,100) , labels=c("under 0", "0-2", "2-4", "4-6", "6-8", "8+"))  
CleanMonthly$PSAVERT <- cut(CleanMonthly$PSAVERT, breaks = c( 0, 5, 8, 11, 14 , 100) , labels=c("0-5", "5-8", "8-11", "11-14", "14+"))  
CleanMonthly$recession <- factor(CleanMonthly$recession)  
CleanMonthly$UNRATE <- factor(CleanMonthly$UNRATE)  
CleanMonthly$MEDCPIM158SFRBCLE <- factor(CleanMonthly$MEDCPIM158SFRBCLE)  
CleanMonthly$PSAVERT <- factor(CleanMonthly$PSAVERT)

Finally; commercial real estate prices, delinquency rate on credit cards, and household owners equity were read in. This data was quarterly and was converted to monthly data and added to the rest of the monthly statistics. The quarterly data was converted to monthly by interpolating downwards making all 3 months that make up the quarter equal to the quarter statistic. This will be the cleaned data set that will be used as the base for all machine learning. It should be noted that each technique utilized will require unique cleaning of the data prior to use. Any sample/cleaning/removing/interpolating used in the future will be discussed when used. Additionally, a year from recession column was added. This was created by lagging the recession column by 12 months. Later, a 6 month from recession column was added. These were used for prediction models. The tail of this finalized data can be seen below, there is a date, recession, unemployment rate, median consumer price index, personal savings rate, commercial real estate prices, delinquency rate on credit card loans, household’s owners’ equity in real estate, and if there was a recession exactly 12 months in the future.

Commercial\_Real\_Estate\_Prices\_for\_United\_States <- read\_csv("Definitely Useful/quarterly/Commercial Real Estate Prices for United States.csv")

## Rows: 70 Columns: 2

Delinquency\_Rate\_on\_Credit\_Card\_Loans\_All\_Commercial\_Banks <- read\_csv("Definitely Useful/quarterly/Delinquency Rate on Credit Card Loans All Commercial Banks.csv")

## Rows: 128 Columns: 2

Households\_Owners\_Equity\_in\_Real\_Estate\_Level <- read\_csv("Definitely Useful/quarterly/Households Owners Equity in Real Estate Level.csv")

## Rows: 304 Columns: 2

Households\_Owners\_Equity\_in\_Real\_Estate\_Level$DATE <- as.Date(Households\_Owners\_Equity\_in\_Real\_Estate\_Level$DATE)  
MonthREC <- data.table(dates=as.Date(c(DT\_month$Date)), recession=c(DT\_month$recession))  
AllMonthlyData <- MonthlyData  
placeholderMonth1 <- merge(AllMonthlyData, Commercial\_Real\_Estate\_Prices\_for\_United\_States, by = "DATE",all.x=TRUE)  
placeholderMonth2 <- merge(placeholderMonth1, Delinquency\_Rate\_on\_Credit\_Card\_Loans\_All\_Commercial\_Banks, by = "DATE", all.x=TRUE)  
placeholderMonth3<- merge(placeholderMonth2, Households\_Owners\_Equity\_in\_Real\_Estate\_Level, by = "DATE", all.x=TRUE)  
interpolatedMonthly <- placeholderMonth3  
df1 <- interpolatedMonthly %>% fill(COMREPUSQ159N, .direction = 'down')  
df2 <- df1 %>% fill(DRCCLACBS, .direction = 'down')  
df3 <- df2 %>% fill(OEHRENWBSHNO, .direction = 'down')  
df3$yrfromrec <- lead(df3$recession, n=12)  
tail(df3, 15)

## DATE recession UNRATE MEDCPIM158SFRBCLE PSAVERT COMREPUSQ159N  
## 1: 2021-11-01 1 4.2 6.245070 7.1 13.628  
## 2: 2021-12-01 1 3.9 5.611797 7.5 13.628  
## 3: 2022-01-01 1 4.0 6.992826 4.7 13.993  
## 4: 2022-02-01 1 3.8 6.519555 4.5 13.993  
## 5: 2022-03-01 1 3.6 6.350249 3.8 13.993  
## 6: 2022-04-01 0 3.6 6.030320 3.6 8.504  
## 7: 2022-05-01 0 3.6 7.503944 3.4 8.504  
## 8: 2022-06-01 0 3.6 7.924919 2.7 8.504  
## 9: 2022-07-01 0 3.5 6.423611 3.5 8.504  
## 10: 2022-08-01 0 3.7 8.463917 3.2 8.504  
## 11: 2022-09-01 0 3.5 8.382964 3.0 8.504  
## 12: 2022-10-01 0 3.7 6.507748 3.4 8.504  
## 13: 2022-11-01 0 3.6 5.765770 4.1 8.504  
## 14: 2022-12-01 0 3.5 7.049161 4.4 8.504  
## 15: 2023-01-01 0 3.4 8.142495 4.5 8.504  
## DRCCLACBS OEHRENWBSHNO yrfromrec  
## 1: 1.57 27513.46 0  
## 2: 1.57 27513.46 0  
## 3: 1.65 29439.08 0  
## 4: 1.65 29439.08 NA  
## 5: 1.65 29439.08 NA  
## 6: 1.85 31144.04 NA  
## 7: 1.85 31144.04 NA  
## 8: 1.85 31144.04 NA  
## 9: 2.09 31207.41 NA  
## 10: 2.09 31207.41 NA  
## 11: 2.09 31207.41 NA  
## 12: 2.25 30981.35 NA  
## 13: 2.25 30981.35 NA  
## 14: 2.25 30981.35 NA  
## 15: 2.25 30981.35 NA

**Analysis and Models**

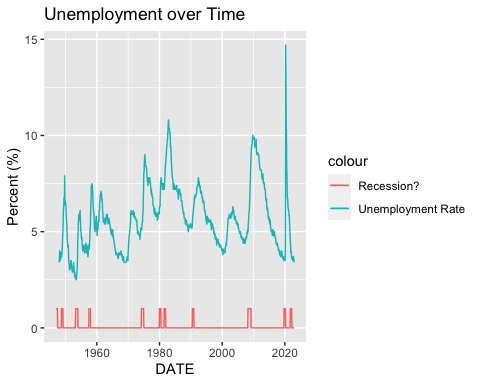
This data was used for apriori rule exploration, which can be seen below aside the corresponding visualization plots. Note the apriori rule algorithm and clustering only takes into consideration the unemployment rate, personal savings rate, and median consumer price index. The plots highlight the relationship discovered by apriori rule mining.

rules<-apriori(data=CleanMonthly, parameter=list(supp=0.01,conf = 0.01), appearance = list(rhs= c('recession=1'), default='lhs'))

rules<-sort(rules, decreasing=TRUE,by="confidence")  
rules<-sort(rules, decreasing=TRUE,by="confidence")  
inspect(rules)

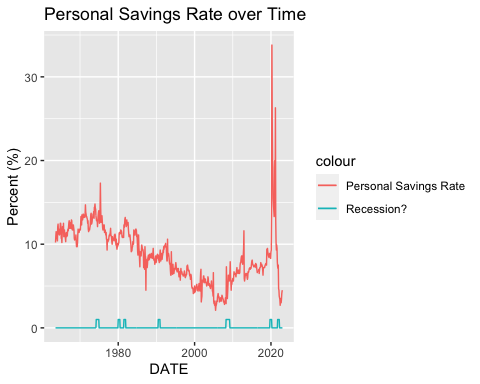
## lhs rhs support confidence coverage   
## [1] {UNRATE=2-4} => {recession=1} 0.02190581 0.12820513 0.1708653  
## [2] {PSAVERT=11-14} => {recession=1} 0.02081051 0.09547739 0.2179628  
## [3] {UNRATE=6-8} => {recession=1} 0.02190581 0.08968610 0.2442497  
## [4] {} => {recession=1} 0.08543264 0.08543264 1.0000000  
## [5] {UNRATE=4-6} => {recession=1} 0.03066813 0.06422018 0.4775465  
## [6] {PSAVERT=5-8} => {recession=1} 0.01423877 0.05508475 0.2584885  
## [7] {PSAVERT=8-11} => {recession=1} 0.01314348 0.05357143 0.2453450  
## [8] {MEDCPIM158SFRBCLE=2-4} => {recession=1} 0.01752464 0.05177994 0.3384447  
## lift count  
## [1] 1.5006575 20   
## [2] 1.1175751 19   
## [3] 1.0497873 20   
## [4] 1.0000000 78   
## [5] 0.7517055 28   
## [6] 0.6447740 13   
## [7] 0.6270604 12   
## [8] 0.6060908 16

ggplot(MonthlyData, aes(x=DATE)) +   
 geom\_line(aes(y=UNRATE, color="Unemployment Rate")) +  
 geom\_line(aes(y=recession,color="Recession?")) +  
 ylab("Percent (%)") + ggtitle("Unemployment over Time")



*Figure 8.* Unemployment Rate vs. Recession

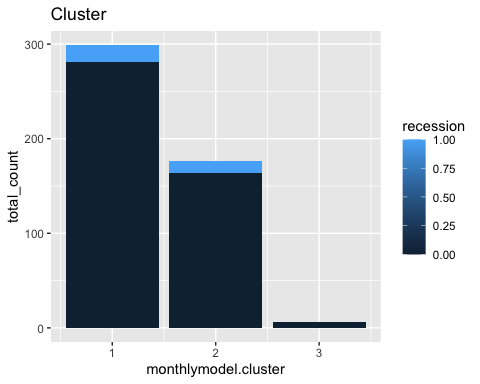
ggplot(MonthlyData[200:913], aes(x=DATE)) +   
 geom\_line(aes(y=PSAVERT, color="Personal Savings Rate")) +  
 geom\_line(aes(y=recession,color="Recession?")) +  
 ylab("Percent (%)") + ggtitle("Personal Savings Rate over Time")



*Figure 9.* Personal Savings Rate vs. Recession

Clustering using the same dataset, omitting NAs, was also completed.

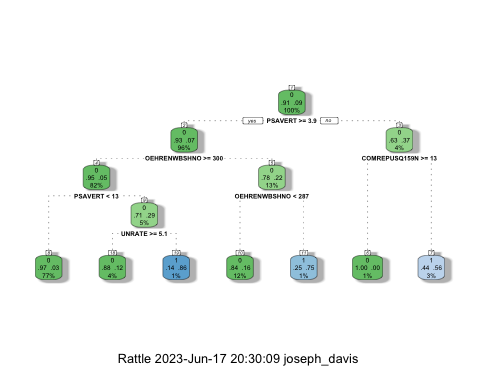
NAmonthnorec <- na.omit(MonthlyData)  
Monthnorec <- subset(NAmonthnorec, select = -c(recession, DATE) )  
monthlymodel <- kmeans(Monthnorec, 3)  
cluster\_assignment <- data.frame(NAmonthnorec, monthlymodel$cluster)  
clusters <- subset(cluster\_assignment, select = c(recession, monthlymodel.cluster))  
  
monthcluster <- clusters %>% group\_by(recession,monthlymodel.cluster) %>%   
 summarise(total\_count=n(),.groups = 'drop') %>%  
 as.data.frame()  
  
ggplot(monthcluster, aes(fill=recession, y=total\_count, x=monthlymodel.cluster)) +   
 geom\_bar(position="stack", stat="identity") + ggtitle('Cluster')



*Figure 10.* Clustering

A decision tree was created to classify 12 months from recession. The data was split 70/30 for all techniques. For the decision tree specifically the columns were factorized and a confusion matrix was created to test efficacy.

sample70 <- sample(c(TRUE, FALSE), nrow(df3), replace=TRUE, prob=c(0.3,0.7))  
test70 <- df3[sample70, ]  
train70 <- df3[!sample70, ]  
test70$recession <- factor(test70$recession)  
test70$yrfromrec<- factor(test70$yrfromrec)  
train70$recession <- factor(train70$recession)  
train70$yrfromrec <- factor(train70$yrfromrec)  
train70$OEHRENWBSHNO <- as.integer(train70$OEHRENWBSHNO)  
usefultrain70 <- train70[,c(-1,-2)]  
usefultest70 <- test70[,c(-1,-2)]  
#Decision Tree  
fit1 <- rpart(yrfromrec ~ ., data = usefultrain70, method="class")  
predicted <- predict(fit1,usefultest70, type="class")  
fancyRpartPlot(fit1)



*Figure 11.* Decision Tree 12 months from recession

confusion <- confusionMatrix(predicted, test70$yrfromrec)  
confusion

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 259 15  
## 1 10 4  
##   
## Accuracy : 0.9132   
## 95% CI : (0.8745, 0.943)  
## No Information Rate : 0.934   
## P-Value [Acc > NIR] : 0.9337   
##   
## Kappa : 0.1975   
##   
## Mcnemar's Test P-Value : 0.4237   
##   
## Sensitivity : 0.9628   
## Specificity : 0.2105   
## Pos Pred Value : 0.9453   
## Neg Pred Value : 0.2857   
## Prevalence : 0.9340   
## Detection Rate : 0.8993   
## Detection Prevalence : 0.9514   
## Balanced Accuracy : 0.5867   
##   
## 'Positive' Class : 0   
##

Naive Bayes was performed on the same dataset. No major changes were made to the testing and training data between Decision Tree and Naive Bayes.

#Naive Bayes  
nolabel70test <- usefultrain70  
nolabel70test <- nolabel70test[,c(-7)]  
Test\_justLabel<- train70$yrfromrec  
NB\_object<- naive\_bayes(yrfromrec~., data=train70, na.action = na.pass)

NB\_prediction <- predict(NB\_object, nolabel70test)

confusionNB <- confusionMatrix(NB\_prediction, Test\_justLabel)  
round(confusionNB$overall[1]\*100,2)

## Accuracy   
## 88.74

confusionNB$table

## Reference  
## Prediction 0 1  
## 0 541 50  
## 1 19 3

confusionNB

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 541 50  
## 1 19 3  
##   
## Accuracy : 0.8874   
## 95% CI : (0.8597, 0.9114)  
## No Information Rate : 0.9135   
## P-Value [Acc > NIR] : 0.9890287   
##   
## Kappa : 0.0308   
##   
## Mcnemar's Test P-Value : 0.0003043   
##   
## Sensitivity : 0.9661   
## Specificity : 0.0566   
## Pos Pred Value : 0.9154   
## Neg Pred Value : 0.1364   
## Prevalence : 0.9135   
## Detection Rate : 0.8825   
## Detection Prevalence : 0.9641   
## Balanced Accuracy : 0.5113   
##   
## 'Positive' Class : 0   
##

Major changes were performed on the data for the following 3 machine learning techniques. Different economic metrics began being measured/recorded at different dates. Because of this there are many dates near the beginning of this dataset with NA for most of the economic metrics. To account for this the data that was previously NA was filled upwards, so the first date that contained data became the value for all NA values above it for each metric. Normally, extrapolation to this degree isn’t ideal, however the models created had reasonable efficacy. The data with this level of filling will be referred to as approximated data.

proxdata <- df3  
adf1 <- proxdata %>% fill(COMREPUSQ159N, .direction = 'up')  
adf2 <- adf1 %>% fill(DRCCLACBS, .direction = 'up')  
adf3 <- adf2 %>% fill(OEHRENWBSHNO, .direction = 'up')  
adf4 <- adf3 %>% fill(UNRATE, .direction = 'up')  
adf5 <- adf4 %>% fill(MEDCPIM158SFRBCLE, .direction = 'up')  
adf6 <- adf5 %>% fill(PSAVERT, .direction = 'up')  
approxdata <- adf6  
approxdata <- approxdata[,c(-1,-2)]  
approxsample70 <- sample(c(TRUE, FALSE), nrow(approxdata), replace=TRUE, prob=c(0.3,0.7))  
approxtest70 <- approxdata[approxsample70, ]  
approxtrain70 <- approxdata[!approxsample70, ]  
approxtest70$yrfromrec<- factor(approxtest70$yrfromrec)  
approxtrain70$yrfromrec <- factor(approxtrain70$yrfromrec)  
approxtrain70$OEHRENWBSHNO <- as.integer(approxtrain70$OEHRENWBSHNO)  
approxtest70$OEHRENWBSHNO <- as.integer(approxtest70$OEHRENWBSHNO)

This approximated data was trained for KNN and can be seen below. again, a 70/30 split was used.

approxtrain70 <- na.omit(approxtrain70)  
approxtest70 <- na.omit(approxtest70)  
predKNN <- knn(train=approxtrain70, test=approxtest70, cl=approxtrain70$yrfromrec, k=1)  
mylabel\_col <- approxtest70$yrfromrec  
newpred1=cbind(mylabel\_col, predKNN)  
confusionMatrix(mylabel\_col, predKNN)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 242 0  
## 1 0 23  
##   
## Accuracy : 1   
## 95% CI : (0.9862, 1)  
## No Information Rate : 0.9132   
## P-Value [Acc > NIR] : 3.556e-11   
##   
## Kappa : 1   
##   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.9132   
## Detection Rate : 0.9132   
## Detection Prevalence : 0.9132   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##

SVM was performed with the approximated data and the results can be seen below in a confusion matrix.

#SVM with approximated data (92.04% accurate, the same as no information)  
SVMtrain70 <- approxtrain70  
  
SVMtest70 <- approxtest70  
  
svm<- svm(yrfromrec~., data = SVMtrain70)  
pred=predict(svm, newdata=SVMtest70, type=C)  
confusionNB <- confusionMatrix(pred, SVMtest70$yrfromrec)  
confusionNB

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 242 23  
## 1 0 0  
##   
## Accuracy : 0.9132   
## 95% CI : (0.8726, 0.9442)  
## No Information Rate : 0.9132   
## P-Value [Acc > NIR] : 0.5552   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 4.49e-06   
##   
## Sensitivity : 1.0000   
## Specificity : 0.0000   
## Pos Pred Value : 0.9132   
## Neg Pred Value : NaN   
## Prevalence : 0.9132   
## Detection Rate : 0.9132   
## Detection Prevalence : 1.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : 0   
##

To demonstrate the efficacy of the approximated data, an SVM without approximated data was used and showed an accuracy no greater than the no information rate. It should be noted now that SVM will not be used for modeling/prediction and approximating this data didn’t help the model. It is only included to show the extent of research done.

#SVM with no approximates gave an accuracy of 87.27%, again identical to the no information rate  
SVMtrain70 <- usefultrain70  
  
SVMtest70 <- usefultest70  
SVMtest70noNA <- na.omit(SVMtest70)  
svm<- svm(yrfromrec~., data = SVMtrain70)  
pred=predict(svm, newdata=SVMtest70, type=C)  
confusionNB <- confusionMatrix(pred, SVMtest70noNA$yrfromrec)  
confusionNB

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 59 6  
## 1 0 0  
##   
## Accuracy : 0.9077   
## 95% CI : (0.8098, 0.9654)  
## No Information Rate : 0.9077   
## P-Value [Acc > NIR] : 0.60643   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : 0.04123   
##   
## Sensitivity : 1.0000   
## Specificity : 0.0000   
## Pos Pred Value : 0.9077   
## Neg Pred Value : NaN   
## Prevalence : 0.9077   
## Detection Rate : 0.9077   
## Detection Prevalence : 1.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : 0   
##

The approximated data was next used in a random forest with 1000 trees and the resulting confusion matrix can be seen below.

#Random Forest  
RFSet <- df3[1:900]  
RFSet

## DATE recession UNRATE MEDCPIM158SFRBCLE PSAVERT COMREPUSQ159N  
## 1: 1947-01-01 1 NA NA NA NA  
## 2: 1947-02-01 1 NA NA NA NA  
## 3: 1947-03-01 1 NA NA NA NA  
## 4: 1947-04-01 1 NA NA NA NA  
## 5: 1947-05-01 1 NA NA NA NA  
## ---   
## 896: 2021-08-01 0 5.2 4.022038 9.5 13.240  
## 897: 2021-09-01 0 4.8 5.054275 7.9 13.240  
## 898: 2021-10-01 1 4.5 7.662853 7.3 13.628  
## 899: 2021-11-01 1 4.2 6.245070 7.1 13.628  
## 900: 2021-12-01 1 3.9 5.611797 7.5 13.628  
## DRCCLACBS OEHRENWBSHNO yrfromrec  
## 1: NA NA 0  
## 2: NA NA 0  
## 3: NA NA 0  
## 4: NA NA 0  
## 5: NA NA 0  
## ---   
## 896: 1.55 26364.27 0  
## 897: 1.55 26364.27 0  
## 898: 1.57 27513.46 0  
## 899: 1.57 27513.46 0  
## 900: 1.57 27513.46 0

RFtrain70 <- approxtrain70  
RFtest70 <- approxtest70  
rfm <- randomForest(yrfromrec~., data=RFtrain70, ntree=1000)

predRF <- predict(rfm, RFtest70, type=c("class"))  
confusionRF <- confusionMatrix(predRF, RFtest70$yrfromrec)  
confusionRF

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 242 15  
## 1 0 8  
##   
## Accuracy : 0.9434   
## 95% CI : (0.9084, 0.968)  
## No Information Rate : 0.9132   
## P-Value [Acc > NIR] : 0.0447372   
##   
## Kappa : 0.4934   
##   
## Mcnemar's Test P-Value : 0.0003006   
##   
## Sensitivity : 1.0000   
## Specificity : 0.3478   
## Pos Pred Value : 0.9416   
## Neg Pred Value : 1.0000   
## Prevalence : 0.9132   
## Detection Rate : 0.9132   
## Detection Prevalence : 0.9698   
## Balanced Accuracy : 0.6739   
##   
## 'Positive' Class : 0   
##

The exact same approach as before was done on data except with a 6 month lead on recession instead of a 12 month lead. As before, a 70/30 split was used for all the training/testing and the decision tree and naive bayes used the non-approximated data. The approximated data was used for KNN, SVM, and Random Forest. To avoid redundancy, these will be shown in quick succession.

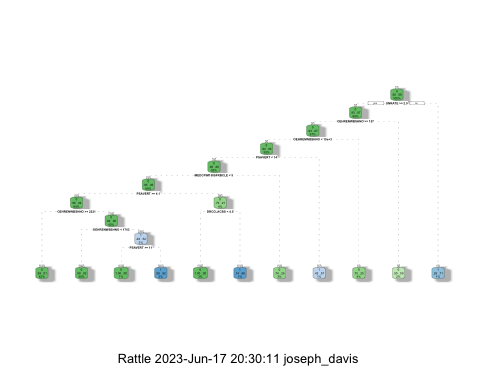
month6 <- df3  
month6$yrfromrec <- lead(df3$recession, n=6)  
#training and testing sets  
sample70 <- sample(c(TRUE, FALSE), nrow(df3), replace=TRUE, prob=c(0.3,0.7))  
test70 <- month6[sample70, ]  
train70 <- month6[!sample70, ]  
test70$recession <- factor(test70$recession)  
test70$yrfromrec<- factor(test70$yrfromrec)  
train70$recession <- factor(train70$recession)  
train70$yrfromrec <- factor(train70$yrfromrec)  
train70$OEHRENWBSHNO <- as.integer(train70$OEHRENWBSHNO)  
test70$OEHRENWBSHNO <- as.integer(test70$OEHRENWBSHNO)  
usefultrain70 <- train70[,c(-1,-2)]  
usefultest70 <- test70[,c(-1,-2)]  
summary(usefultest70)

## UNRATE MEDCPIM158SFRBCLE PSAVERT COMREPUSQ159N   
## Min. : 2.500 Min. :0.3635 Min. : 2.700 Min. :-28.142   
## 1st Qu.: 4.400 1st Qu.:2.2835 1st Qu.: 6.325 1st Qu.: 3.925   
## Median : 5.600 Median :2.8857 Median : 8.400 Median : 6.969   
## Mean : 5.711 Mean :2.9375 Mean : 8.576 Mean : 5.540   
## 3rd Qu.: 6.700 3rd Qu.:3.4156 3rd Qu.:11.275 3rd Qu.: 10.942   
## Max. :10.400 Max. :7.9249 Max. :17.300 Max. : 15.891   
## NA's :4 NA's :120 NA's :38 NA's :195   
## DRCCLACBS OEHRENWBSHNO yrfromrec   
## Min. :1.550 Min. : 149 0 :247   
## 1st Qu.:2.520 1st Qu.: 436 1 : 20   
## Median :4.080 Median : 4214 NA's: 1   
## Mean :3.793 Mean : 6110   
## 3rd Qu.:4.700 3rd Qu.:10012   
## Max. :6.770 Max. :31207   
## NA's :145 NA's :4

#Decision Tree  
fit1 <- rpart(yrfromrec ~ ., data = usefultrain70, method="class")  
summary(fit1)

## Call:  
## rpart(formula = yrfromrec ~ ., data = usefultrain70, method = "class")  
## n=635 (10 observations deleted due to missingness)

predicted <- predict(fit1,usefultest70, type="class")  
fancyRpartPlot(fit1)



*Figure 12.* Decision Tree 6 months from recession

confusion <- confusionMatrix(predicted, test70$yrfromrec)  
confusion

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 242 13  
## 1 5 7  
##   
## Accuracy : 0.9326   
## 95% CI : (0.8955, 0.9596)  
## No Information Rate : 0.9251   
## P-Value [Acc > NIR] : 0.37479   
##   
## Kappa : 0.404   
##   
## Mcnemar's Test P-Value : 0.09896   
##   
## Sensitivity : 0.9798   
## Specificity : 0.3500   
## Pos Pred Value : 0.9490   
## Neg Pred Value : 0.5833   
## Prevalence : 0.9251   
## Detection Rate : 0.9064   
## Detection Prevalence : 0.9551   
## Balanced Accuracy : 0.6649   
##   
## 'Positive' Class : 0   
##

#Naive Bayes  
nolabel70test <- usefultrain70  
nolabel70test <- nolabel70test[,c(-7)]  
Test\_justLabel<- train70$yrfromrec  
NB\_object<- naive\_bayes(yrfromrec~., data=train70, na.action = na.pass)

## Warning: naive\_bayes(): y contains NAs. They are excluded from the estimation  
## process.

## Warning: naive\_bayes(): Feature DATE - zero probabilities are present. Consider  
## Laplace smoothing.

NB\_prediction <- predict(NB\_object, nolabel70test)

## Warning: predict.naive\_bayes(): only 6 feature(s) out of 8 defined in the naive\_bayes object "NB\_object" are used for prediction.

confusionNB <- confusionMatrix(NB\_prediction, Test\_justLabel)  
confusionNB

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 581 48  
## 1 7 4  
##   
## Accuracy : 0.9141   
## 95% CI : (0.8896, 0.9346)  
## No Information Rate : 0.9188   
## P-Value [Acc > NIR] : 0.6989   
##   
## Kappa : 0.1015   
##   
## Mcnemar's Test P-Value : 6.906e-08   
##   
## Sensitivity : 0.98810   
## Specificity : 0.07692   
## Pos Pred Value : 0.92369   
## Neg Pred Value : 0.36364   
## Prevalence : 0.91875   
## Detection Rate : 0.90781   
## Detection Prevalence : 0.98281   
## Balanced Accuracy : 0.53251   
##   
## 'Positive' Class : 0   
##

round(confusionNB$overall[1]\*100,2)

## Accuracy   
## 91.41

confusionNB$table

## Reference  
## Prediction 0 1  
## 0 581 48  
## 1 7 4

bproxdata <- month6  
bdf1 <- bproxdata %>% fill(COMREPUSQ159N, .direction = 'up')  
bdf2 <- bdf1 %>% fill(DRCCLACBS, .direction = 'up')  
bdf3 <- bdf2 %>% fill(OEHRENWBSHNO, .direction = 'up')  
bdf4 <- bdf3 %>% fill(UNRATE, .direction = 'up')  
bdf5 <- bdf4 %>% fill(MEDCPIM158SFRBCLE, .direction = 'up')  
bdf6 <- bdf5 %>% fill(PSAVERT, .direction = 'up')  
bapproxdata <- bdf6  
bapproxdata <- bapproxdata[,c(-1,-2)]  
bapproxsample70 <- sample(c(TRUE, FALSE), nrow(bapproxdata), replace=TRUE, prob=c(0.3,0.7))  
bapproxtest70 <- bapproxdata[bapproxsample70, ]  
bapproxtrain70 <- bapproxdata[!bapproxsample70, ]  
bapproxtest70$yrfromrec<- factor(bapproxtest70$yrfromrec)  
bapproxtrain70$yrfromrec <- factor(bapproxtrain70$yrfromrec)  
bapproxtrain70$OEHRENWBSHNO <- as.integer(bapproxtrain70$OEHRENWBSHNO)  
bapproxtest70$OEHRENWBSHNO <- as.integer(bapproxtest70$OEHRENWBSHNO)

#remove the last few rows (the year from recession are NA)  
bapproxtrain70 <- na.omit(bapproxtrain70)  
bapproxtest70 <- na.omit(bapproxtest70)

predKNN <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=1)  
  
mylabel\_col <- bapproxtest70$yrfromrec  
newpred1 <- data.frame(mylabel\_col, predKNN)  
confusionMatrix(mylabel\_col, predKNN)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 251 0  
## 1 0 24  
##   
## Accuracy : 1   
## 95% CI : (0.9867, 1)  
## No Information Rate : 0.9127   
## P-Value [Acc > NIR] : 1.241e-11   
##   
## Kappa : 1   
##   
## Mcnemar's Test P-Value : NA   
##   
## Sensitivity : 1.0000   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 1.0000   
## Prevalence : 0.9127   
## Detection Rate : 0.9127   
## Detection Prevalence : 0.9127   
## Balanced Accuracy : 1.0000   
##   
## 'Positive' Class : 0   
##

common <- data.frame(mylabel\_col == predKNN)  
KNN1 <- sum(common)/length(common$mylabel\_col)  
KNN1

## [1] 1

#SVM with approximated data (92.04% accurate, the same as no information)  
SVMtrain70 <- bapproxtrain70  
SVMtest70 <- bapproxtest70  
svm<- svm(yrfromrec~., data = SVMtrain70)  
pred=predict(svm, newdata=SVMtest70, type=C)  
confusionSVM <- confusionMatrix(pred, SVMtest70$yrfromrec)  
confusionSVM

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 251 21  
## 1 0 3  
##   
## Accuracy : 0.9236   
## 95% CI : (0.8856, 0.9521)  
## No Information Rate : 0.9127   
## P-Value [Acc > NIR] : 0.3041   
##   
## Kappa : 0.2068   
##   
## Mcnemar's Test P-Value : 1.275e-05   
##   
## Sensitivity : 1.0000   
## Specificity : 0.1250   
## Pos Pred Value : 0.9228   
## Neg Pred Value : 1.0000   
## Prevalence : 0.9127   
## Detection Rate : 0.9127   
## Detection Prevalence : 0.9891   
## Balanced Accuracy : 0.5625   
##   
## 'Positive' Class : 0   
##

#SVM with no approximates gave an accuracy of 87.27%, again identical to the no information rate  
SVMtrain70 <- usefultrain70  
SVMtest70 <- usefultest70  
SVMtest70noNA <- na.omit(SVMtest70)  
svm<- svm(yrfromrec~., data = SVMtrain70)  
pred=predict(svm, newdata=SVMtest70, type=C)  
confusionNB <- confusionMatrix(pred, SVMtest70noNA$yrfromrec)  
confusionNB

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 63 6  
## 1 0 3  
##   
## Accuracy : 0.9167   
## 95% CI : (0.8274, 0.9688)  
## No Information Rate : 0.875   
## P-Value [Acc > NIR] : 0.18871   
##   
## Kappa : 0.4667   
##   
## Mcnemar's Test P-Value : 0.04123   
##   
## Sensitivity : 1.0000   
## Specificity : 0.3333   
## Pos Pred Value : 0.9130   
## Neg Pred Value : 1.0000   
## Prevalence : 0.8750   
## Detection Rate : 0.8750   
## Detection Prevalence : 0.9583   
## Balanced Accuracy : 0.6667   
##   
## 'Positive' Class : 0   
##

#Random Forest  
RFSet <- month6[1:900]  
RFtrain70 <- bapproxtrain70  
RFtest70 <- bapproxtest70  
rfm <- randomForest(yrfromrec~., data=RFtrain70, ntree=100)  
predRF <- predict(rfm, RFtest70, type=c("class"))  
confusionRF <- confusionMatrix(predRF, RFtest70$yrfromrec)  
confusionRF

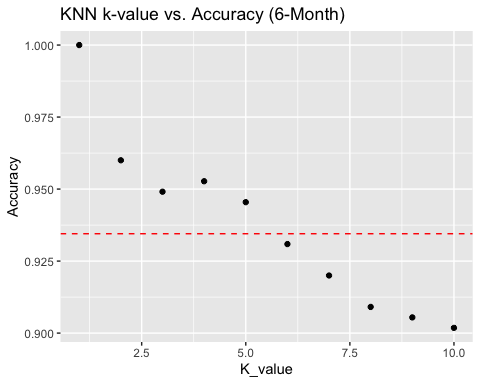
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 249 12  
## 1 2 12  
##   
## Accuracy : 0.9491   
## 95% CI : (0.9161, 0.9719)  
## No Information Rate : 0.9127   
## P-Value [Acc > NIR] : 0.01598   
##   
## Kappa : 0.6063   
##   
## Mcnemar's Test P-Value : 0.01616   
##   
## Sensitivity : 0.9920   
## Specificity : 0.5000   
## Pos Pred Value : 0.9540   
## Neg Pred Value : 0.8571   
## Prevalence : 0.9127   
## Detection Rate : 0.9055   
## Detection Prevalence : 0.9491   
## Balanced Accuracy : 0.7460   
##   
## 'Positive' Class : 0   
##

The 12-month KNN, 6-month KNN, and 6-month random forest models were the most efficacious and will therefore be explored further. The k-value was adjusted from 1-10 for both the 12 and 6 month predictions. Their accuracy is plotted below.

#KNN explored further (6-month)  
bapproxtrain70 <- na.omit(bapproxtrain70)  
bapproxtest70 <- na.omit(bapproxtest70)  
  
predKNN1 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=1)  
mylabel\_col1 <- bapproxtest70$yrfromrec  
common1 <- data.frame(mylabel\_col1 == predKNN1)  
KNN1 <- sum(common1)/length(common1$mylabel\_col1)  
  
predKNN2 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=2)  
mylabel\_col2 <- bapproxtest70$yrfromrec  
common2 <- data.frame(mylabel\_col2 == predKNN2)  
KNN2 <- sum(common2)/length(common2$mylabel\_col2)  
  
predKNN3 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=3)  
mylabel\_col3 <- bapproxtest70$yrfromrec  
common3 <- data.frame(mylabel\_col3 == predKNN3)  
KNN3 <- sum(common3)/length(common3$mylabel\_col3)  
  
predKNN4 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=4)  
mylabel\_col4 <- bapproxtest70$yrfromrec  
common4 <- data.frame(mylabel\_col4 == predKNN4)  
KNN4 <- sum(common4)/length(common4$mylabel\_col4)  
  
predKNN5 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=5)  
mylabel\_col5 <- bapproxtest70$yrfromrec  
common5 <- data.frame(mylabel\_col5 == predKNN5)  
KNN5 <- sum(common5)/length(common5$mylabel\_col5)  
  
predKNN6 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=6)  
mylabel\_col6 <- bapproxtest70$yrfromrec  
common6 <- data.frame(mylabel\_col6 == predKNN6)  
KNN6 <- sum(common6)/length(common6$mylabel\_col6)  
  
predKNN7 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=7)  
mylabel\_col7 <- bapproxtest70$yrfromrec  
common7 <- data.frame(mylabel\_col7 == predKNN7)  
KNN7 <- sum(common7)/length(common7$mylabel\_col7)  
  
predKNN8 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=8)  
mylabel\_col8 <- bapproxtest70$yrfromrec  
common8 <- data.frame(mylabel\_col8 == predKNN8)  
KNN8 <- sum(common8)/length(common8$mylabel\_col8)  
  
predKNN9 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=9)  
mylabel\_col9 <- bapproxtest70$yrfromrec  
common9 <- data.frame(mylabel\_col9 == predKNN9)  
KNN9 <- sum(common9)/length(common9$mylabel\_col9)  
  
predKNN10 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=10)  
mylabel\_col10 <- bapproxtest70$yrfromrec  
common10 <- data.frame(mylabel\_col10 == predKNN10)  
KNN10 <- sum(common10)/length(common10$mylabel\_col10)  
  
K\_value <- c(1:10)  
Accuracy <- c(KNN1, KNN2, KNN3, KNN4, KNN5, KNN6, KNN7, KNN8, KNN9, KNN10)  
KNNResults <- data.frame(K\_value, Accuracy)  
KNNResults

## K\_value Accuracy  
## 1 1 1.0000000  
## 2 2 0.9600000  
## 3 3 0.9490909  
## 4 4 0.9527273  
## 5 5 0.9454545  
## 6 6 0.9309091  
## 7 7 0.9200000  
## 8 8 0.9090909  
## 9 9 0.9054545  
## 10 10 0.9018182

ggplot(KNNResults, aes(K\_value, Accuracy))+ geom\_point() +ggtitle(label = "KNN k-value vs. Accuracy (6-Month)") + geom\_hline(yintercept=.9345,linetype='dashed', color="red")

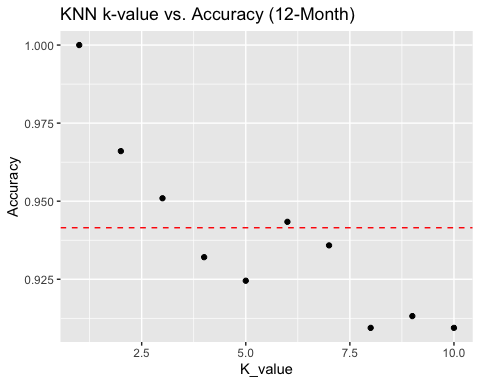


*Figure 13.* KNN 6-months with k-value 1-10

#KNN 12-month  
bapproxtrain70 <- na.omit(approxtrain70)  
bapproxtest70 <- na.omit(approxtest70)  
  
predKNN1 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=1)  
mylabel\_col1 <- bapproxtest70$yrfromrec  
common1 <- data.frame(mylabel\_col1 == predKNN1)  
KNN1 <- sum(common1)/length(common1$mylabel\_col1)  
  
predKNN2 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=2)  
mylabel\_col2 <- bapproxtest70$yrfromrec  
common2 <- data.frame(mylabel\_col2 == predKNN2)  
KNN2 <- sum(common2)/length(common2$mylabel\_col2)  
  
predKNN3 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=3)  
mylabel\_col3 <- bapproxtest70$yrfromrec  
common3 <- data.frame(mylabel\_col3 == predKNN3)  
KNN3 <- sum(common3)/length(common3$mylabel\_col3)  
  
predKNN4 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=4)  
mylabel\_col4 <- bapproxtest70$yrfromrec  
common4 <- data.frame(mylabel\_col4 == predKNN4)  
KNN4 <- sum(common4)/length(common4$mylabel\_col4)  
  
predKNN5 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=5)  
mylabel\_col5 <- bapproxtest70$yrfromrec  
common5 <- data.frame(mylabel\_col5 == predKNN5)  
KNN5 <- sum(common5)/length(common5$mylabel\_col5)  
  
predKNN6 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=6)  
mylabel\_col6 <- bapproxtest70$yrfromrec  
common6 <- data.frame(mylabel\_col6 == predKNN6)  
KNN6 <- sum(common6)/length(common6$mylabel\_col6)  
  
predKNN7 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=7)  
mylabel\_col7 <- bapproxtest70$yrfromrec  
common7 <- data.frame(mylabel\_col7 == predKNN7)  
KNN7 <- sum(common7)/length(common7$mylabel\_col7)  
  
predKNN8 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=8)  
mylabel\_col8 <- bapproxtest70$yrfromrec  
common8 <- data.frame(mylabel\_col8 == predKNN8)  
KNN8 <- sum(common8)/length(common8$mylabel\_col8)  
  
predKNN9 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=9)  
mylabel\_col9 <- bapproxtest70$yrfromrec  
common9 <- data.frame(mylabel\_col9 == predKNN9)  
KNN9 <- sum(common9)/length(common9$mylabel\_col9)  
  
predKNN10 <- knn(train=bapproxtrain70, test=bapproxtest70, cl=bapproxtrain70$yrfromrec, k=10)  
mylabel\_col10 <- bapproxtest70$yrfromrec  
common10 <- data.frame(mylabel\_col10 == predKNN10)  
KNN10 <- sum(common10)/length(common10$mylabel\_col10)  
  
K\_value <- c(1:10)  
Accuracy <- c(KNN1, KNN2, KNN3, KNN4, KNN5, KNN6, KNN7, KNN8, KNN9, KNN10)  
KNNResults <- data.frame(K\_value, Accuracy)  
KNNResults

## K\_value Accuracy  
## 1 1 1.0000000  
## 2 2 0.9660377  
## 3 3 0.9509434  
## 4 4 0.9320755  
## 5 5 0.9245283  
## 6 6 0.9433962  
## 7 7 0.9358491  
## 8 8 0.9094340  
## 9 9 0.9132075  
## 10 10 0.9094340

ggplot(KNNResults, aes(K\_value, Accuracy))+ geom\_point() +ggtitle(label = "KNN k-value vs. Accuracy (12-Month)") + geom\_hline(yintercept=.9415,linetype='dashed', color="red")



*Figure 14.* KNN 12-months with k-value 1-10

The 6-month random forest model was tested again, this time instead of approximate data na.roughfix was used and no extrapolating was necessary. 10,000 trees were included and the 70/30 split was again employed. A variable importance plot was also generated to help understand the value of each variable.

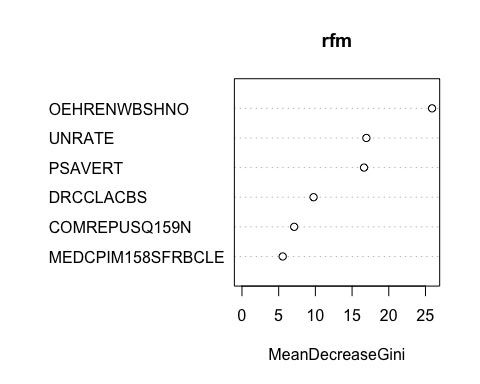
RFSet <- month6[1:907]  
RFSet <- RFSet[,c(-1,-2)]  
sample70 <- sample(c(TRUE, FALSE), nrow(RFSet), replace=TRUE, prob=c(0.3,0.7))  
RFsettest70 <- RFSet[sample70, ]  
RFsettrain70 <- RFSet[!sample70, ]  
RFsettest70$yrfromrec<- factor(RFsettest70$yrfromrec)  
RFsettrain70$yrfromrec <- factor(RFsettrain70$yrfromrec)  
RFsettrain70$OEHRENWBSHNO <- as.integer(RFsettrain70$OEHRENWBSHNO)  
RFsettest70$OEHRENWBSHNO <- as.integer(RFsettest70$OEHRENWBSHNO)  
rfm <- randomForest(yrfromrec~., data=RFsettrain70, ntree=10000, na.action = na.roughfix)  
print(rfm)

##   
## Call:  
## randomForest(formula = yrfromrec ~ ., data = RFsettrain70, ntree = 10000, na.action = na.roughfix)   
## Type of random forest: classification  
## Number of trees: 10000  
## No. of variables tried at each split: 2  
##   
## OOB estimate of error rate: 4.42%  
## Confusion matrix:  
## 0 1 class.error  
## 0 573 7 0.01206897  
## 1 21 32 0.39622642

predRF <- predict(rfm, RFsettest70, type=c("class"))  
confusionRF <- confusionMatrix(predRF, RFsettest70$yrfromrec)  
confusionRF

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 53 1  
## 1 0 2  
##   
## Accuracy : 0.9821   
## 95% CI : (0.9045, 0.9995)  
## No Information Rate : 0.9464   
## P-Value [Acc > NIR] : 0.191   
##   
## Kappa : 0.791   
##   
## Mcnemar's Test P-Value : 1.000   
##   
## Sensitivity : 1.0000   
## Specificity : 0.6667   
## Pos Pred Value : 0.9815   
## Neg Pred Value : 1.0000   
## Prevalence : 0.9464   
## Detection Rate : 0.9464   
## Detection Prevalence : 0.9643   
## Balanced Accuracy : 0.8333   
##   
## 'Positive' Class : 0   
##

varImpPlot(rfm)



*Figure 15.* Random Forest Mean Decrease Gini

**Results**

This analysis process began by performing 2 forms of unsupervised learning. The first of which was apriori rule mining and the second of which was clustering. The clustering was included and can be seen in figure 10. The clustering couldn’t separate recession vs. non-recession data and no insights could be made from it. Apriori rule mining was found to be successful. 2 main insights were gained when setting the left-hand side to recession. An unemployment rate from 2%-4% and a personal savings rate between 11%-14% were found to be associated with recession. Both phenomena were plotted and can be seen in figures 8 and 9 respectively. The unemployment pattern is easily seen, the unemployment falls until it reaches a low point (typically around 2-4%) a recession occurs and the unemployment spikes and the pattern repeats. The personal savings rate pattern is more difficult to discern however it appears visually that often when a recession happens, the personal savings rate is between 10% and 15%.

The next phase of analysis was a dive into recession prediction. The variable used were date, recession, unemployment rate, median consumer price index, personal savings rate, commercial real estate prices, delinquency rate on credit card loans, household owners’ equity in real estate, and if there was a recession exactly 12 months in the future. Eventually the same metrics were used in all the models substituting the recession in 12 months to a recession in 6 months. Because of the large number of models run, they are all summarized and shown below in figure 16.

A picture containing text, screenshot, font, number

Description automatically generated

*Figure 16.* Summary Statistics

The no information rate for each of these models caused great difficulty. Because most of the time there isn’t a recession, a model that always predicts no recession will be correct over 90% of the time. To then outperform that high standard required trying a multitude of models.

The decision tree models can be seen in figures 11 and 12. Neither of these models were able to achieve a 95% confidence interval that outperformed the no information rate. For this reason, both the 12 and 6 months from recession predictions using a decision tree aren’t useful models. Next a Naïve-Bayes model was created. Again, both the 12- and 6-month prediction didn’t have a 95% confidence interval that excluded the no information rate. Additionally, the 12-month prediction Naïve-Bayes model underperformed the no information rate. Meaning that using the model was worse than simply predicting no recession for all dates.

The remaining 3 models created were generated using the approximated data. It should be noted again, as above, that the approximated data had a fill up so any NA values at the start of the dataset would be filled with the first datapoint. This extensive extrapolation is far from ideal. The models generated in general performed better than the Naïve-Bayes and decision tree models. Because of the success of the KNN and random forest models, they will be discussed last and most thoroughly. The support vector machine model couldn’t outperform the no information rate. It should be noted that the SVM model was created with both the approximated and non-approximated data and in both the 6- and 12-month case the approximated data performed equal to the no information rate. However, as is the case with the Decision Trees and Naïve-Bayes, none of these models could reliably predict if there was going to be a future recession.

The k-nearest neighbor models performed the best of all models created. Models predicting recessions 12 and 6 months in the future were created with high accuracy and 95% confidence intervals not containing the no information rate. Both models were able to predict with 100% accuracy if there would be a recession in the future. A 100% accuracy model is often a sign of concern and overfitting, however after adjusting the k-values its believed that these models are not overfit. To explore the affect the k-value has on these models, KNNs with a k-value from 1-10 was used for both models and the accuracies were plotted. Figures 13 and 14 both display a very similar downward trend as k-value increases with a slight increase in accuracy for a k-value of 4 and 6 for the 6 month and 12 months respectively. Because k-value is the number of comparative points the test set will be compared with within the training set its possible this causes the decrease in accuracy. If most of the training points are “not recession” then adding more comparative points for the testing point would increase the likelihood that its classified as “not recession” even if the closest points to it are a recession. Essentially, because the dataset is saturated with non-recession datapoints, the higher the k-value the more likely something is to be predicted as non-recession. This could therefore cause errors and a decrease in accuracy with a higher k-value. Nevertheless, using KNN was by far the most accurate prediction model created. Again, it should be noted that both KNN models were created using the approximated data.

KNN however cannot explain which factors account for the likelihood of a recession. To help understand what factors do, a random forest model was created. This random forest initially used the approximated data and achieved an accuracy of 94.91% and the confidence interval did not contain the no information rate. It should be noted that this model would achieve varying accuracies every time it ran and sometimes the 95% confidence interval did contain the no information rate. Nevertheless, as this was the second-best model and could provide insight into the importance of the factors at play it was investigated further.

To begin, the initial random forest was created using the approximated data. However, using na.roughfix allowed a random forest to be created with an accuracy of 98.21% that didn’t require the approximated data. Because, na.roughfix was used the testing data was smaller and the confidence interval was wider, from 90.45% to 99.95% which does contain the no information rate. However, to understand the value of each variable it felt necessary to build a model without large amounts of extrapolation. The random forest model did achieve a high accuracy and with more data potentially could have a narrower confidence interval. The model used 10,000 trees and was trained for classification. To understand the dataset a MeanDecreaseGini plot is shown above in figure 15. This plot shows the value that each variable provided for the random forest. The variables in order of importance are as follows: household ownership equity in real estate level, unemployment rate, personal savings rate, delinquency rates on credit card loans, commercial real estate prices, and finally median consumer price index.

**Conclusion**

In conclusion many useful insights were gained from this investigation. It’s understood to a better level what specific metrics can help predict a recession. Specifically, household owners’ equity is very important for predicting if a recession will happen in the future. Other metrics such as commercial real estate prices have less of an impact. Additionally, a model was successfully created that can accurately predict a recession 6-months and 12-months in the future. These models had extremely high accuracy’s and given metrics today would be very likely to predict future recessions. To help alleviate future recessions the single most important variable is going to lie in residential ownership of real estate equity and as a nation it would be wise to prioritize this.

**Appendix**

Economics information:

<https://www.investopedia.com/terms/r/recession.asp#:~:text=The%20Bottom%20Line-,A%20recession%20is%20a%20significant%2C%20widespread%2C%20and%20prolonged%20downturn%20in,the%20economy%20is%20in%20recession>.

Dataset:

<https://www.kaggle.com/datasets/mikoajfish99/us-recession-and-financial-indicators?select=Unemployment+Level.csv>