Unemployment Lenght Prediction

Marcelo Bohrer

Abstract

We fit 5 different models to predict the length of unemployment in the US economy for the 1967-2015 period. The best fit is provided by the Cubist model.

The Problem

The idea of this project is to develop an algorithm to predict the length of unemployment in the US economy based on some economic indicators. The length of unemployment is closely related to government spending on social security, default rates on mortgages and credit cards, etc. Being able to predict the length on unemployment is of high value to make government spending more efficient and to reduce risk to financial institutions. Additionally it serves as a thermometer for the economy as a whole.

The problem we are trying to solve is to develop a simple method to predict the median weeks of unemployment based on monthly indicators that are easier to obtain than an accurate depiction of the length of unemployment.

The variables are [variable codes in brackets]:

- Median weeks of unemployment [uempmed];
- Unemployment rate [urate];
- Consumer price index [cpi];
- Producer price index [ppi];
- Fed interest rate [InterestRate];
- Oil price (month avg.) [OilPrice];
- Month [month];
- Stock Index (Dow Jones) [DowJones];
- Personal savings rate [psavert];
- Personal consumption spending [pce];
- Manufacturing capacity utlization [cap];
- Manufacturing Confidence report [mconf];
- Dummy variable for recession period [USREC].

The idea is to use the median weeks of uemployment as the dependent variable and the rest as explanatory variables.

Why were these variables chosen? It goes without much explanation that the unemployment rate and the lenght of unemployment are closely related. Cpi and ppi are two measures of inflation, it was believed in Economics untill the 1970's that there was a trade-off between inflation and unemployment, although the statistical relationship broke down in the later part of the 20th century, measures of inflation can still be relevant for our prediction, as they guide wage negociations. The interest rate measures the ease of monetary policy which may affect economic activity and in turn employment. Oil price is relevant because it affects income left after gas purchases and the rellocation cost for job searches. The month variable is included to capture some form of seasonality, for example to see if Christmas sales in November and December have any effect on the lenght of unemployment. Stock index captures part of the uncertainty in the economy and the activity of some of the largest firms. Savings and consumption rates reflect how long people can afford to remain off work. Capacity and Confidence reports guide future investment and hiring decisions. Finally, recession periods invariably generate unemployment.

The Data

The data for this project comes from a variety of sources. Median weeks of unemployment, unemployment rate (we use unemployed/total population), personal savings and consumption spending are available in the "economics" data set in the ggplot2 package. CPI and PPI are available in the BLS website[www.bls.gov]. Stock index and oil prices are available on the ipeadata website[www.ipeadata.gov.br]. Manufacturing capacity and confidence reports are published by the OECD[www.oecd.org]. Finally the dummy variable for recession periods is computed by the St. Louis FED and available using the Quantmod package.

Basic Data Wrangling

The data sets were downloaded then read into R. Some transformations were required to get the data into the "tidy" format. The code below reads the raw data and generates the final data frame.

```r
library(ggplot2)
library(tidyr)
library(dplyr)

data(economics)

economics <- economics %>% separate(date, into = c("Year", "month", "day"), sep = "\\-")
economics$day <- NULL

economics$month <- as.numeric(economics$month)

###

ppi <- read.csv("PPI.csv", skip = 10)

ppi <- ppi %>% gather(month, ppi, Jan:Dec )


ppi <- ppi[order(ppi$Year),]

ppi$month <-match(ppi$month, month.abb)

ppi$month <- as.numeric(ppi$month)
ppi$Year <- as.character(ppi$Year)
###
cpi <- read.csv("CPI.csv", skip = 10)
cpi$HALF1<-NULL
cpi$HALF2 <-NULL

cpi <- cpi %>% gather(month, cpi, Jan:Dec )
cpi <- cpi[order(cpi$Year),]

cpi$month <- match(cpi$month, month.abb)

cpi$Year <- as.character(cpi$Year)
###
ir <- read.csv("Interest Rate.csv", skip = 5)

ir <- ir %>% separate(Time.Period, into =  c("Year", "month"), sep = "\\-")
```

```r
colnames(ir) <- c("Year", "month", "InterestRate")


ir$month <- as.numeric(ir$month)
###
dj <- read.csv("DowJones.csv")

dj <- dj %>% separate(Data, into = c("Year", "month"), sep = "\\.")

colnames(dj ) <- c("Year", "month", "DowJones")

dj$month[dj$month=="1"] <- "10"
dj$month <- as.numeric(dj$month)
###
oil <- read.csv("Oil Price.csv")

oil <- oil %>% separate(Data, into = c("Year", "month"), sep = "\\."  )
oil$International.Financial.Statistics..FMI.IFS....IFS12_PETROLEUM12 <- NULL

colnames(oil) <- c("Year", "month", "OilPrice")

oil$month[oil$month=="1"]<-"10"
oil$month <- as.numeric(oil$month)
####
library(quantmod)
getSymbols("USREC",src="FRED")

usrec <- data.frame(date=index(USREC), coredata(USREC))
usrec <- usrec %>% separate(date, into=c("Year", "month", "day"), sep = "\\-" )
usrec$day <- NULL

newUSREC <- read.csv("newUSREC.csv")

usrec2 <- rbind(usrec, newUSREC)


usrec$month <- as.numeric(usrec$month)
###
#Capacity Utilization
mei <- read.csv("MEI.csv")

cap <- subset(mei, SUBJECT == "BSCURT", select = c(SUBJECT, TIME, Value ))

cap <- cap %>% separate(TIME, into = c("Year", "month"), sep = "\\-")
cap <- rename(cap, cap = Value)
cap$SUBJECT <- NULL

cap$month <- as.numeric(cap$month)
###
#Confidence Indicator
mconf <- subset(mei, SUBJECT == "BSCI", select = c(SUBJECT, TIME, Value) )
mconf <- mconf %>% separate(TIME, into = c("Year", "month"), sep = "\\-")
mconf <- rename(mconf, mconf = Value)
mconf$SUBJECT <- NULL

mconf$month <- as.numeric(mconf$month)

###############
#merging the dfs

mydata <- inner_join(economics, ppi, by = c("Year", "month"))
mydata <- inner_join(mydata, cpi,by = c("Year", "month"))
mydata <- inner_join(mydata, ir, by = c("Year", "month"))
mydata <- inner_join(mydata,dj, by = c("Year", "month"))
mydata <- inner_join(mydata, oil, by = c("Year", "month"))
mydata <- inner_join(mydata, usrec2, by = c("Year", "month"))
mydata <- inner_join(mydata, cap, by = c("Year", "month"))
mydata <- inner_join(mydata, mconf, by = c("Year", "month"))

mydata <- mydata %>% mutate(urate = unemploy/pop)
```

Exploratory Data Analysis

```r
library(dplyr)
library(quantmod)
library(zoo)
library(ggplot2)
options("getSymbols.warning4.0" = FALSE)
```

Now we can proceed to gain some insights into the data set. We begin by checking some summary statistics of the data.

```
##       pce             psavert          uempmed           ppi
##  Min.   :  507.4   Min.   : 1.900   Min.   : 4.00   Min.   : 35.7
##  1st Qu.: 1582.2   1st Qu.: 5.500   1st Qu.: 6.00   1st Qu.: 76.9
##  Median : 3953.6   Median : 7.700   Median : 7.50   Median :121.8
##  Mean   : 4843.5   Mean   : 7.937   Mean   : 8.61   Mean   :115.1
##  3rd Qu.: 7667.3   3rd Qu.:10.500   3rd Qu.: 9.10   3rd Qu.:142.9
##  Max.   :12161.5   Max.   :17.000   Max.   :25.20   Max.   :203.0
##       cpi          InterestRate       DowJones          OilPrice
##  Min.   : 34.70   Min.   : 0.070   Min.   :  607.9   Min.   :  1.79
##  1st Qu.: 71.55   1st Qu.: 3.062   1st Qu.:  942.2   1st Qu.: 12.73
##  Median :141.40   Median : 5.395   Median : 2910.3   Median : 20.18
##  Mean   :135.81   Mean   : 5.607   Mean   : 5378.0   Mean   : 32.13
##  3rd Qu.:193.07   3rd Qu.: 7.935   3rd Qu.:10165.5   3rd Qu.: 35.62
##  Max.   :241.80   Max.   :19.100   Max.   :18133.0   Max.   :132.55
##      USREC             cap            mconf             urate
##  Min.   :0.0000   Min.   :66.93   Min.   :-41.200   Min.   :0.01332
##  1st Qu.:0.0000   1st Qu.:77.90   1st Qu.: -0.950   1st Qu.:0.02404
##  Median :0.0000   Median :80.38   Median :  6.800   Median :0.02855
##  Mean   :0.1446   Mean   :80.47   Mean   :  5.125   Mean   :0.02989
##  3rd Qu.:0.0000   3rd Qu.:83.57   3rd Qu.: 12.200   3rd Qu.:0.03506
##  Max.   :1.0000   Max.   :88.85   Max.   : 44.200   Max.   :0.05169
```

We can see that the mean median weeks of unemployment for our data set is 8.61, meaning that on average of rour period half of the unemployed population were out of a job for 8 weeks or less.

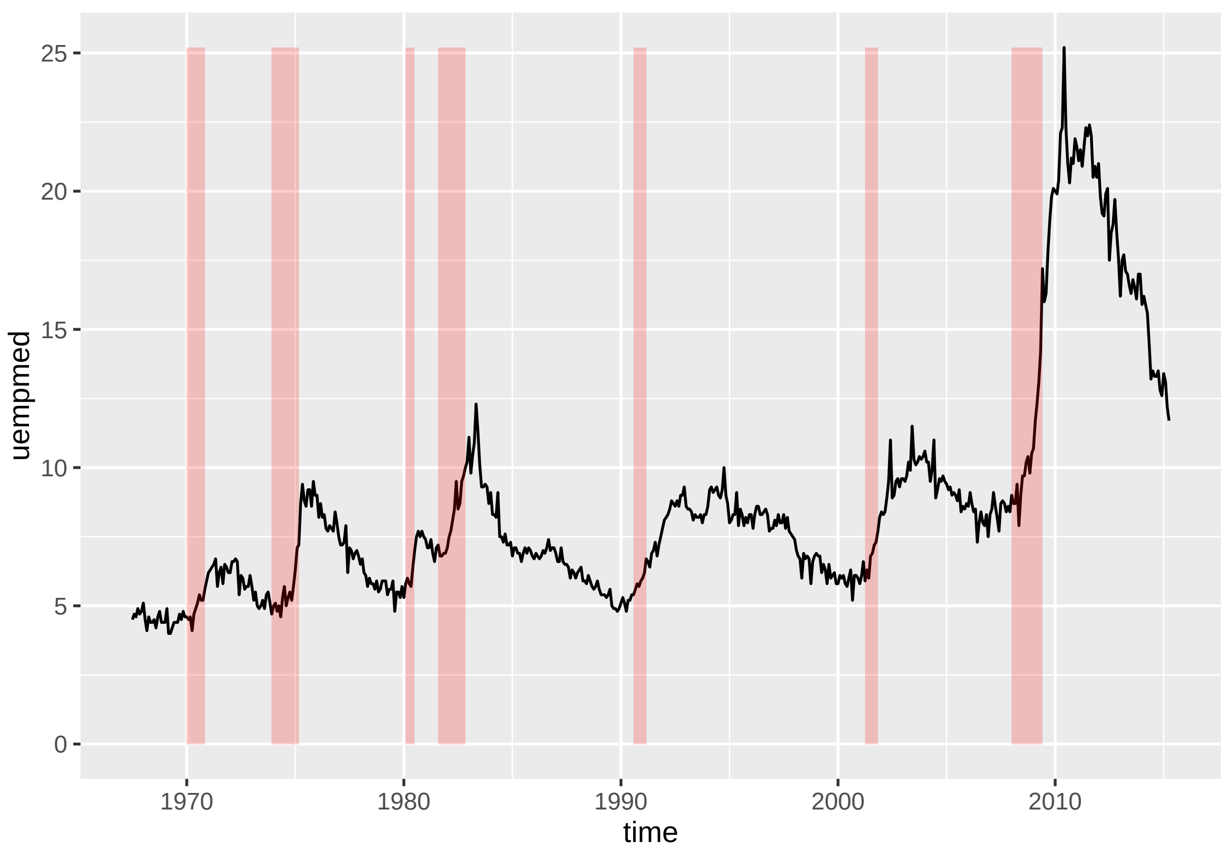
Next we construct a basic correlation matrix.

```
##                      pce      psavert     uempmed         ppi         cpi
## pce           1.00000000 -0.837069022  0.72734920  0.95938697  0.96995149
## psavert      -0.83706902  1.000000000 -0.38741589 -0.84223157 -0.89100204
## uempmed       0.72734920 -0.387415889  1.00000000  0.71358003  0.66110053
## ppi           0.95938697 -0.842231572  0.71358003  1.00000000  0.98452945
## cpi           0.96995149 -0.891002043  0.66110053  0.98452945  1.00000000
## InterestRate -0.70716734  0.542937908 -0.62034075 -0.58601958 -0.64704298
## DowJones      0.96281343 -0.820448264  0.62695303  0.88365255  0.91286187
## OilPrice      0.83202811 -0.540009082  0.79811885  0.81573291  0.74438182
## USREC        -0.09717555  0.209404807 -0.13702519 -0.09648551 -0.11735902
## cap          -0.50966422  0.306091504 -0.61499498 -0.53733908 -0.50629700
## mconf        -0.02891649  0.002591511  0.07239267 -0.09850582 -0.07532791
## urate         0.24839136 -0.022259939  0.66830258  0.39082727  0.28342342
##              InterestRate    DowJones    OilPrice       USREC        cap
## pce            -0.7071673  0.96281343  0.83202811 -0.09717555 -0.5096642
## psavert         0.5429379 -0.82044826 -0.54000908  0.20940481  0.3060915
## uempmed        -0.6203408  0.62695303  0.79811885 -0.13702519 -0.6149950
## ppi            -0.5860196  0.88365255  0.81573291 -0.09648551 -0.5373391
## cpi            -0.6470430  0.91286187  0.74438182 -0.11735902 -0.5062970
## InterestRate    1.0000000 -0.69369321 -0.47426930  0.22531671  0.3949031
## DowJones       -0.6936932  1.00000000  0.76739530 -0.12351670 -0.4097480
## OilPrice       -0.4742693  0.76739530  1.00000000  0.02326217 -0.4922828
## USREC           0.2253167 -0.12351670  0.02326217  1.00000000 -0.1957393
## cap             0.3949031 -0.40974801 -0.49228276 -0.19573929  1.0000000
## mconf          -0.1671471  0.02203815 -0.04720614 -0.57131565  0.3661819
## urate          -0.1294801  0.06695485  0.43696235  0.06275184 -0.7302579
##                     mconf       urate
## pce          -0.028916490  0.24839136
## psavert       0.002591511 -0.02225994
## uempmed       0.072392673  0.66830258
## ppi          -0.098505823  0.39082727
## cpi          -0.075327912  0.28342342
## InterestRate -0.167147129 -0.12948014
## DowJones      0.022038152  0.06695485
## OilPrice     -0.047206137  0.43696235
## USREC        -0.571315647  0.06275184
## cap           0.366181908 -0.73025790
## mconf         1.000000000 -0.18210397
## urate        -0.182103969  1.00000000
```

The highest correlation with the lenght of unemployment is the oil price, while the lowest is with the confidence measure.

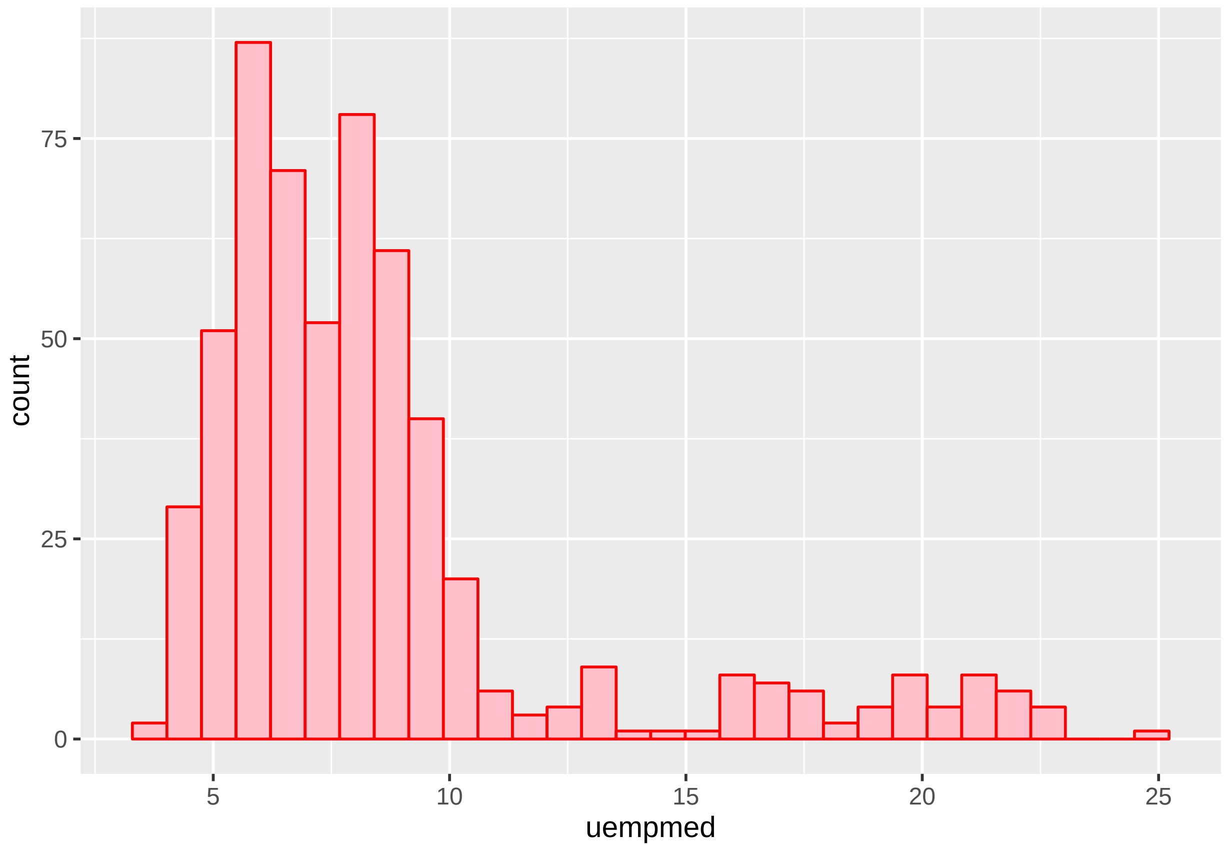
To check the evolution of the median weeks of unemployment we can make a simple plot. As one can see from the correlation table, the median weeks of unemployment and the recession dummy are negatively correlated, we can add the recession periods to the plot to gain some insight into this relation.

```
## [1] "USREC"
```

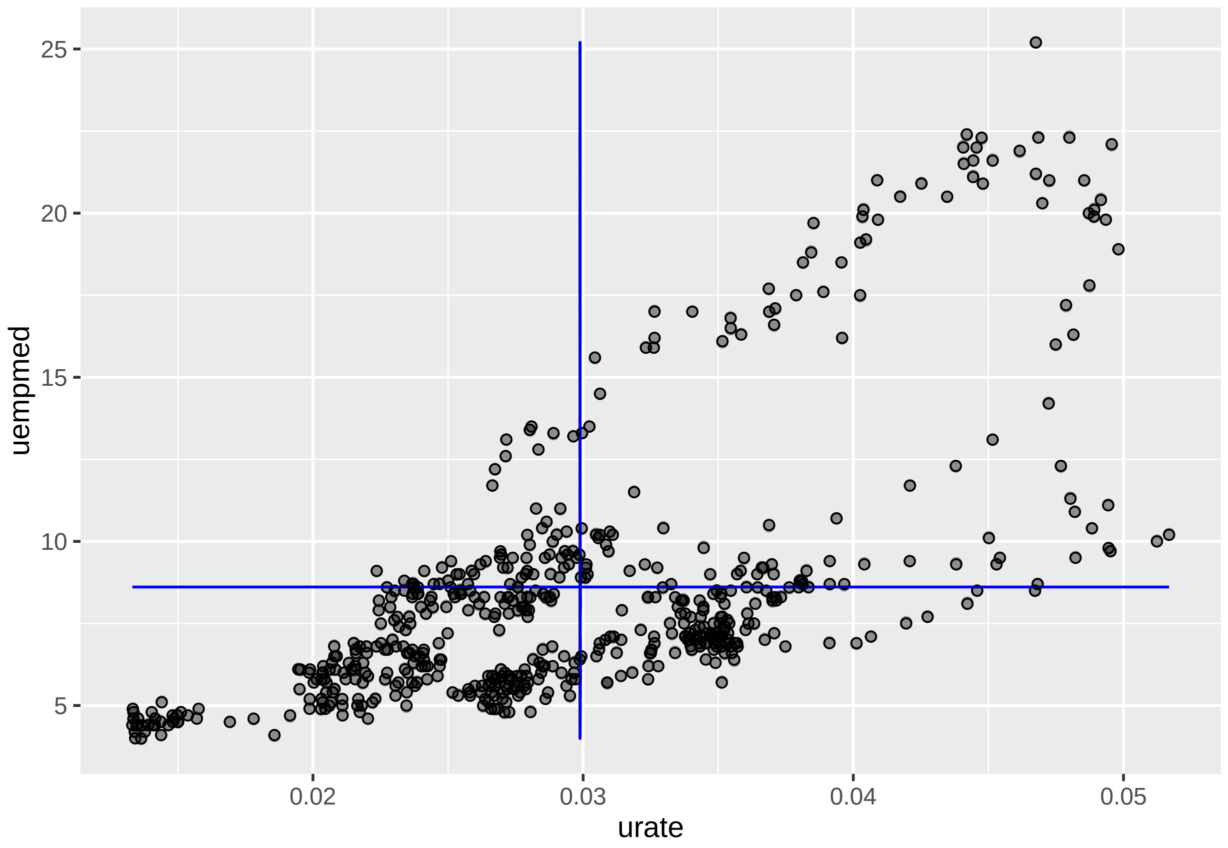


As we could expect, median weeks of unemployment remain high after the recession is over due to the fact that workers are laied off during the recession.

It's distribution is also of interest.



Despite the correlation not being extremely high, it is to expect that there is a close relation between the length and rate of unemployment. This relation is depicted in the following figure. Average values are added in blue.



It is interesting to note that when both variables are above their average values the relationship seems a bit erratic and there's no clear linear interaction.

Now that we have some preliminary ideas about how the data are structured, we can proceed to modelling.

Model Building

In this section different models are built to handle the prediction task. There is no way of telling beforehand which model is most suited for the data, different models need to be fit and some accuracy metric chosen to compare them.

The metric chosen here is the Root Mean Square Error (RMSE) which is defined as

$$RMSE = \sqrt(mean(predicted \quad values - observed \quad values))$$

. It's worth emphazising that the RMSE is measured in units of the dependent variable, in our case in weeks of uemployment.

The standard in machine learning algorithms is to split the data set into a training set and a test set, or to use some form of k-fold cross validation. In the problem at hand we are dealing with time series, in which case the split should be done chronologically. R handles this with the timeslice method, creating different partitions in the data, respecting their order, on which the model is tested. This is shown in the following code.

```
library(caret)
```

```
## Loading required package: lattice
```

```r
set.seed(314)
myTimeControl <- trainControl(method = "timeslice",
                              initialWindow = 36,
                              horizon = 12,
                              fixedWindow = TRUE)
```

Linear Regression

The obvious choice for a first model is linear regression, the basic model used throughout applied work. Given the distribution of points the model fits the line that minimizes the square distance between the line and the points. Although it is not possible to depict such line for a higher dimensional problem, the intuition behind it remains the same. since a line is fitted, it's obvious that it only captures a linear relationship between the variables.

Now we fit the model.

```r
set.seed(5551)
Mod_lm <- train(uempmed~., method = "lm", data = mydata, trControl = myTimeControl)

Mod_lm$results
```

```
##   intercept     RMSE  Rsquared   RMSESD RsquaredSD
## 1      TRUE 1.478704 0.2497201 1.189318  0.2268803
```

Tree

Decision trees are a standard model in classification problems. The idea is to divide the data at each node in order to reduce heterogeneity, that is, to create homogeneous groups. Variables are ranked according to their importance and threshold values determine the nodes, if above threshold the observation goes to one group, if below to the other. Such process is repeated many times to create multiple splits. The final groups are called the leaves of the trees, which entail the final classification. The leaves are reached by branches, that is, splits created at each node.

In the case of a continous outcome, like in the present problem, the algorithm uses the mean value of the dependent variable in each leaf as the predictor. The following code generates a tree.

```r
set.seed(5551)
Mod_tree <- train(uempmed~., method = "rpart", data = mydata, trControl = myTimeControl)
```

```
## Loading required package: rpart
```

```r
Mod_tree$results
```

```
##           cp     RMSE  Rsquared   RMSESD RsquaredSD
## 1 0.04226695 1.411583 0.2652692 1.395996  0.2332551
## 2 0.04667218 1.411931 0.2729980 1.402814  0.2331065
## 3 0.77710043 1.706400 0.2958815 1.616619  0.2408791
```

Random Forest

Random forest models are an extension of of decision trees in which multiple trees are fit and then aggregated to generate a model, the result is the mean prediction of the individual trees.

The following code fits a random forest to our problem.

```
library(caret)
```

```
## Loading required package: lattice
```

```
set.seed(5551)
Mod_rf <- train(uempmed~., method = "rf", data = mydata, trControl = myTimeControl, importance= TRUE)
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
Mod_rf$results
```

```
##   mtry     RMSE  Rsquared   RMSESD RsquaredSD
## 1    2 1.248232 0.2669520 1.340629  0.2375712
## 2    7 1.174131 0.2523244 1.219252  0.2330880
## 3   12 1.169347 0.2528327 1.200961  0.2325953
```

Cubist - Model Trees

The cubist algorithm is a variant of the decision tree in which the terminal leaves contain linear regression models. A prediction is made using the linear regression model at the terminal node of the tree, but is “smoothed” by taking into account the prediction from the linear model in the previous node of the tree (which also occurs recursively up the tree).

The following code implements the cubist algorithm.

```
library(caret)
set.seed(5551)
Mod_Cubist <- train(uempmed~., method = "cubist", data = mydata, trControl = myTimeControl)
```

```
## Loading required package: Cubist
```

```
Mod_Cubist$results
```

```
##   committees neighbors     RMSE  Rsquared    RMSESD RsquaredSD
## 1          1         0 1.298704 0.2961684 1.2446818  0.2609125
## 2          1         5 1.292768 0.2813131 1.2488372  0.2556903
## 3          1         9 1.297558 0.2916495 1.2518096  0.2564876
## 4         10         0 1.129970 0.2951655 0.9634677  0.2586703
## 5         10         5 1.127918 0.2859016 0.9722584  0.2562775
## 6         10         9 1.129245 0.2923003 0.9718373  0.2573033
## 7         20         0 1.139798 0.2915793 0.9554015  0.2576502
## 8         20         5 1.138018 0.2854753 0.9623714  0.2546947
## 9         20         9 1.138862 0.2893855 0.9625785  0.2560634
```

Support Vector Regression

The last model we fit is known as support vector regression and follows a different logic from the tree based algorithms. The idea is to construct a set of hyperplanes to separate the data. Separation is achieved by the hyperplane with the largest distance to the nearest training-data point of any class.

The following code implements a support vector regression.

```
set.seed(5551)
Mod_svm <- train(uempmed~., method = "svmRadial", data = mydata, trControl = myTimeControl)
```

```
## Loading required package: kernlab
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
Mod_svm$results
```

```
##        sigma    C     RMSE  Rsquared   RMSESD RsquaredSD
## 1 0.08893608 0.25 1.776487 0.3068927 1.801570  0.2550750
## 2 0.08893608 0.50 1.704528 0.3026566 1.688529  0.2553598
## 3 0.08893608 1.00 1.657157 0.3037787 1.604683  0.2525911
```

Model Comparison

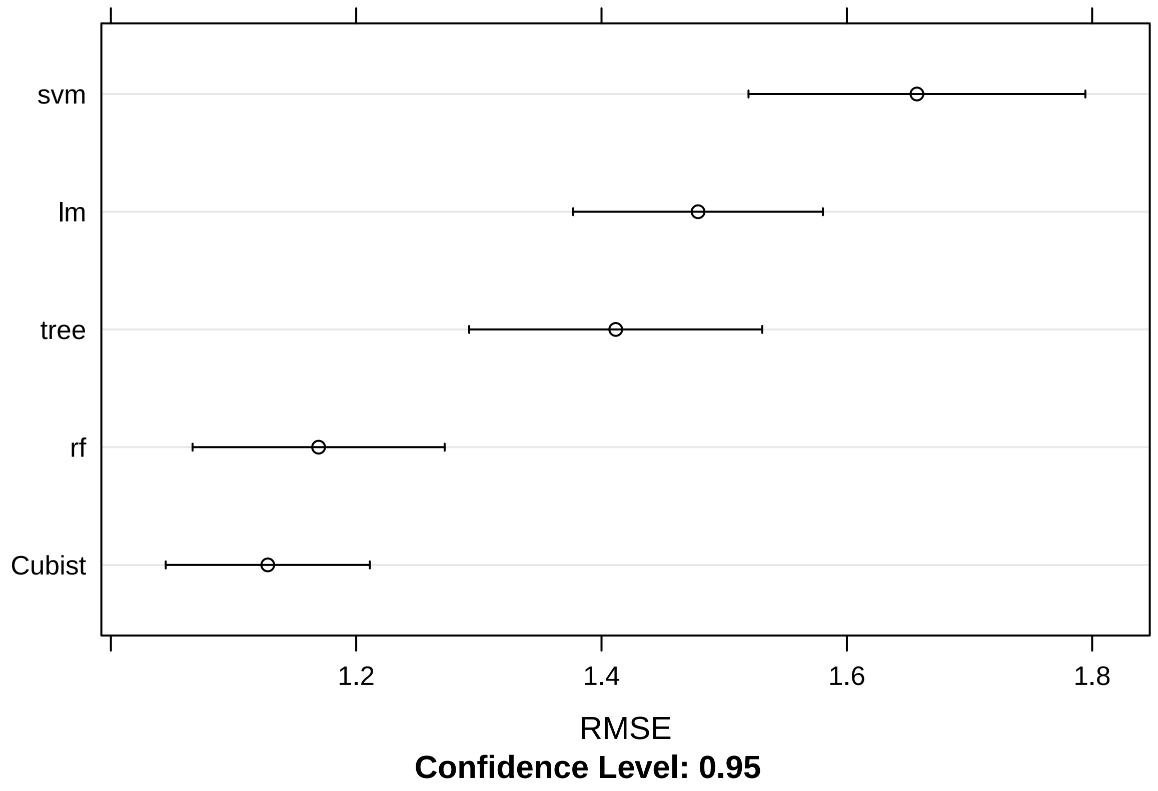
Now we can compare the accuracy of the models developed above, we do so by comparing their RMSE.

```
resamps <- resamples(list(lm = Mod_lm,
                          tree = Mod_tree,
                          rf=Mod_rf,
                          Cubist=Mod_Cubist,
                          svm = Mod_svm
))
summary(resamps)
```

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: lm, tree, rf, Cubist, svm
## Number of resamples: 527
##
## RMSE
##          Min. 1st Qu. Median  Mean 3rd Qu.   Max. NA's
## lm     0.2187  0.6858 1.1170 1.479   1.820  7.578    0
## tree   0.1785  0.5416 0.9898 1.412   1.784 10.580    0
## rf     0.1775  0.4786 0.7300 1.169   1.433  8.051    0
## Cubist 0.1619  0.5704 0.8613 1.128   1.294  6.517    0
## svm    0.2998  0.7178 1.1760 1.657   1.760  9.586    0
##
## Rsquared
##             Min. 1st Qu. Median   Mean 3rd Qu.   Max. NA's
## lm     2.886e-06 0.04350 0.1898 0.2497  0.4187 0.8971    0
## tree   3.622e-04 0.05395 0.2067 0.2653  0.4829 0.7800  406
## rf     3.882e-07 0.05344 0.1853 0.2528  0.3980 0.8879    0
## Cubist 2.058e-07 0.05656 0.2124 0.2859  0.4863 0.9544    0
## svm    1.916e-06 0.07655 0.2463 0.3038  0.4982 0.9139  217
```

```r
library(lattice)

trellis.par.set(caretTheme())
dotplot(resamps, metric = "RMSE")
```



We can see that the Cubist model offers the best fit followed closely by the Random Forest model, while a single tree, linear regression and Support Vector Machine fall behind by a larger RMSE difference. Despite being more computationally burdensome, the model with the best performance can still be implemented.

The result can be interpreted as saying that when the Cubist model faces newdata it is off, on average, by 1.3 weeks.

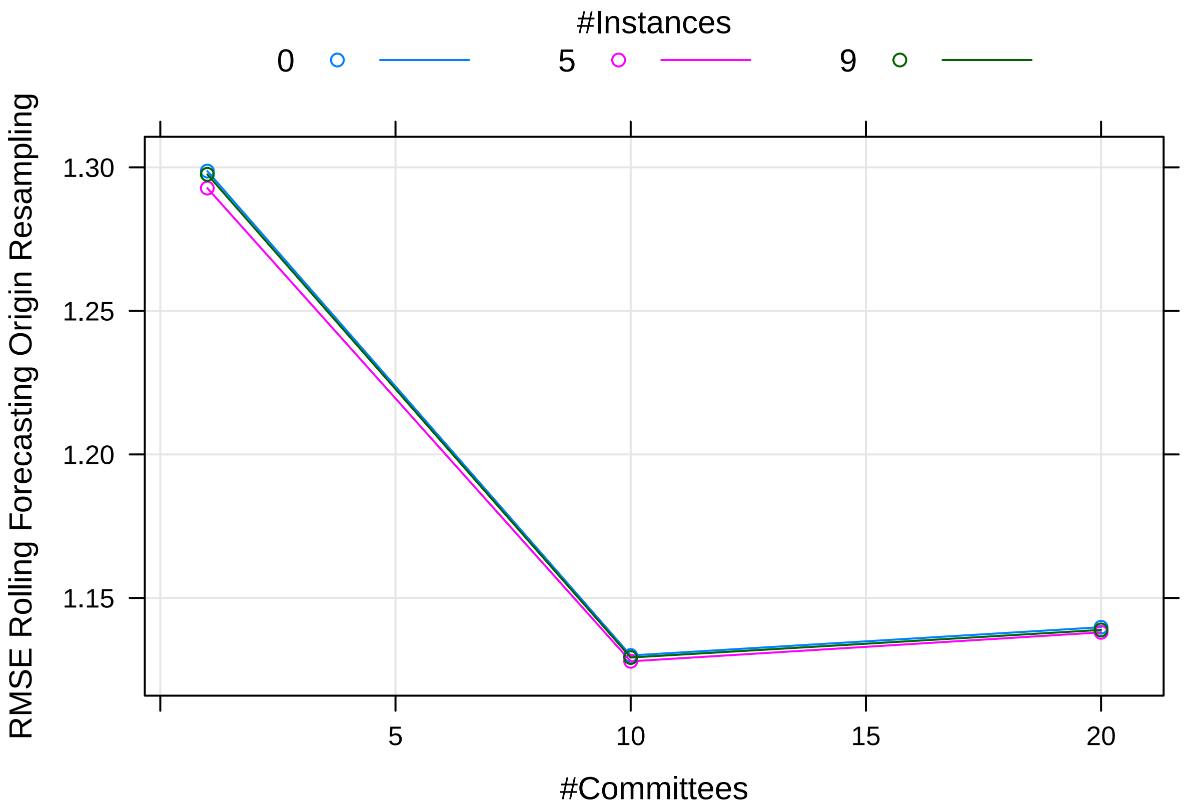
It is interesting to note that the "simple" linear regression performs better than support vector machine regression. This comes to show that the most complex algorithms are not always necessarily better.

We can further analyze some aspects of the best fitting model, namely the variable importance and RMSE reduction by the number of committees.

```
library(caret)
varImp(Mod_Cubist)
```

```
## cubist variable importance
##
##              Overall
## pce          100.000
## urate         98.780
## cpi           79.878
## ppi           51.829
## OilPrice      47.561
## cap           42.073
## mconf         40.244
## DowJones      32.317
## psavert        20.732
## InterestRate    9.756
## month           0.000
## USREC           0.000
```

```r
plot(Mod_Cubist)
```



The most important variable in constructing the random forest was the pce, while the month and recession were the least important.

The graph shows that 10 committees achieve the lowest error.

Limitations and Further Development

An obvious limitation of this approach is that we only have the median weeks of unemployment for the economy as a whole. There's no information on the other statistics pertaining the distribution of weeks of unemployment. Additional dispersion information could add to the analysis, not on the prediction problem but on understanding the phenomenon as a whole. The analysis can be expanded by adding more variables to the data set, especially to capture the effect of economic activity.