Time Series and Regression

Code ▾

Question 1

Describe a situation or problem from your job, everyday life, current events, etc., for which exponential smoothing would be appropriate. What data would you need? Would you expect the value of ?? (the first smoothing parameter) to be closer to 0 or 1, and why?

Response

Exponential smoothing can be a great optimization tool for a restaurant owner. They can use it to predict the amount of supplies they need and detect whether there are any trends in certain dishes. To determine the amount of supplies needed (ie pounds of flour or bottles of wine), they would need to gather historical consumption of the supply per day as the data needed and use a ?? value closer to 0. This is because restaurants can see large amounts of fluctuation due to local events. A football game at a nearby college may bring in a large number of hungry fans to celebrate a win.

Question 2

Using the 20 years of daily high temperature data for Atlanta (July through October) from Homework 2 Question 5, build and use an exponential smoothing model to help make a judgment of whether the unofficial end of summer has gotten later over the 20 years. (Part of the point of this assignment is for you to think about how you might use exponential smoothing to answer this question.) Note: in R, you can use either HoltWinters (simpler to use) or the smooth package's es function (harder to use, but more general). If you use es, the Holt-Winters model uses model="AAM" in the function call (the first and second constants are used "A"dditively, and the third (seasonality) is used "M"ultiplicatively; the documentation doesn't make that clear).

Response

loading data and libraries

Hide

```
library(ggplot2)
library(reshape2)
library(qcc)
library(forecast)

tempdf <- read.table('tempYearsCol.txt', header=TRUE)
```

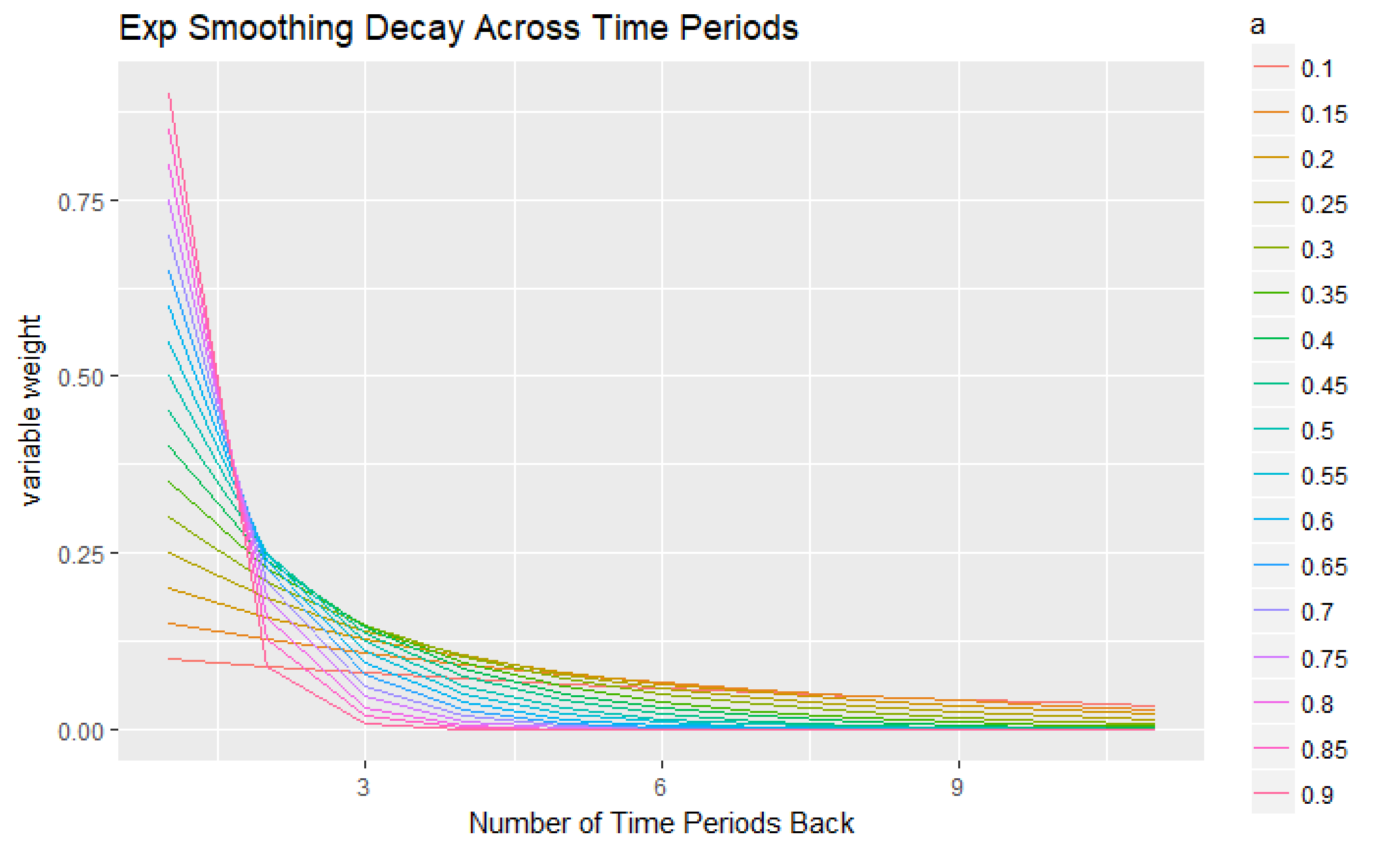
Exponential decay of alpha

Before diving into the homework assignment, I wanted to get a better intuition of how quickly the weight decays in our exponential smoothing function with respect to alpha. This can tell me how much of the "older" data do we really take into account when we calculate the newest S_t value in the below exponential smoothing equation.

$$S_t = \alpha x_t + (1+\alpha)\alpha x_{t-1} + (1+\alpha)^2\alpha x_{t-2} + (1+\alpha)^3\alpha x_{t-3} + \ldots$$

Hide

```
loop <- seq(.1,0.9,0.05)
ary=matrix(NA,11,length(loop))
for (j in 1:length(loop))
{
  for (i in 1:11)
  {
    a=loop[j]
    ary[i,j] <- a*(1-a)^(i-1)
  }
}
colnames(ary) <- loop
meltdf <-melt(ary,id="loop")
meltdf$Var2=as.factor(meltdf$Var2)
colnames(meltdf) <- c("Var1",  "a",  "value")
ggplot(meltdf,aes(x=Var1,y=value,color=a,group=a)) + geom_line() + labs(title="Exp Smoothing Decay Across Time Periods", x="Number of Time Periods Back", y="variable weight")
```



From this graph, you can really see that with a low alpha value, the weight given to the latest data point is very low but it only gradually decays from there, taking into account all of the previous time periods. With a high alpha value, we give the latest data point a very high weight and then quickly decay it down to make previous data points almost negligable. It seems that 3 time periods back is the elbow joint for many of the high value alpha points.

HoltWinters Model

My method of applying Holtwinters model to determine whether the unofficial end of summer has gotten later over the 20 years is to first apply cusum to the dataset to determine the dates of the unofficial end of summer, then apply HoltWinters model to determine a trend value to see whether it is getting later or not.

Hide

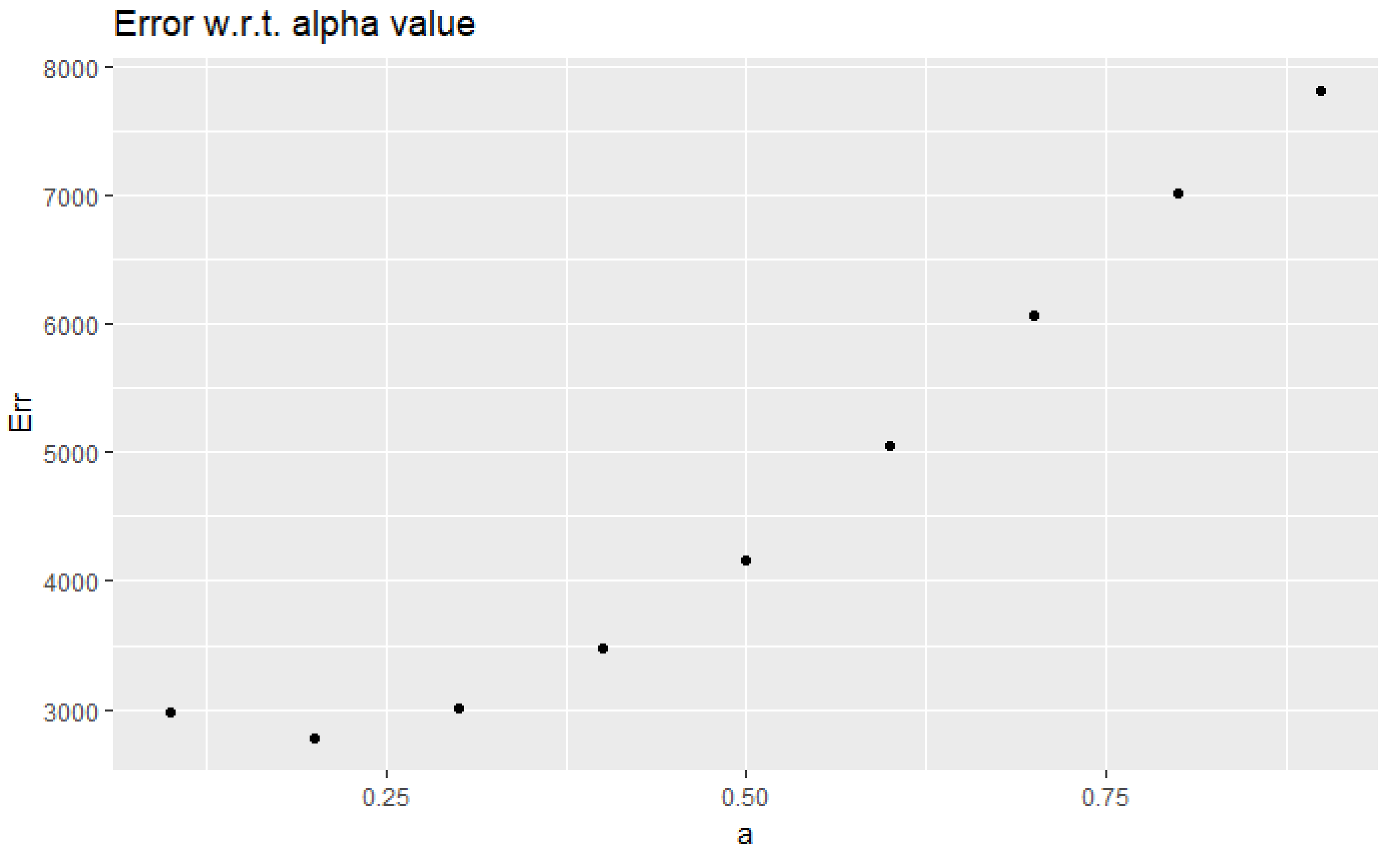
```
EosIndAry <- vector()
years <- colnames(tempdf)[2:length(colnames(tempdf))]
#loop through every year to generate an array of unofficial end of summer using CUSUM
for (i in 1:length(years))
{
  cusumAry <- cusum(tempdf[i+1],plot=FALSE)$violations$lower
  EosIndAry[i] <- min(cusumAry[cusumAry>63]) #any end of summer dates count as false positive be
fore August 31
}
#Run HoltWinters model and look at trend
EosTS <- as.ts(EosIndAry)
hwModel <- HoltWinters(EosTS,beta=TRUE,gamma=FALSE)
```

Alpha value

One way to determine the best alpha value without going through optimization is to plot the model’s sum of squared errors against varying alpha values. Here it looks like the most optimal alpha value would be 0.2.

Hide

```
errMat <- matrix(NA,9,2)
for (i in 1:9)
{
  #test alpha values of 0.1 to 0.9
  a=i/10
  errMat[i,2] <- HoltWinters(EosTS,alpha=a,beta=TRUE,gamma=FALSE)$SSE
  errMat[i,1] <- a
}
colnames(errMat) <- c("a","Err")
ggplot(data.frame(errMat),aes(x=a,y=Err))+geom_point()+labs(title="Error w.r.t. alpha value")
```



But being lazy, we can just have the algorithm calculate it for us.

Unofficial end of summer trend

By not placing an alpha value in the model, it will be optimally calculated and we can see it was pretty close to 0.2. We see the resulting trend

Hide

```
#Run HoltWinters model and look at trend
EosTS <- as.ts(EosIndAry)
hwModel <- HoltWinters(EosTS,beta=TRUE,gamma=FALSE)
print(paste0("alpha value is: ",hwModel$alpha))
```

```
[1] "alpha value is: 0.198381683678516"
```

Hide

```
print(hwModel$fitted)
```

```
Time Series:
Start = 3
End = 20
Frequency = 1
       xhat    level        trend
 3 91.00000 87.00000  4.000000000
 4 95.39676 91.19838  4.198381684
 5 94.27980 92.73909  1.540709211
 6 85.39365 89.06637 -3.672721781
 7 82.35827 85.71232 -3.354050834
 8 80.84589 83.27910 -2.433216115
 9 73.71266 78.49588 -4.783222934
10 70.63049 74.56319 -3.932692472
11 77.95378 76.25848  1.695297837
12 81.25447 78.75648  2.497993290
13 82.46121 80.60884  1.852365819
14 82.94029 81.77457  1.165725246
15 88.49411 85.13434  3.359768304
16 92.05459 88.59447  3.460128495
17 85.97074 87.28260 -1.311862411
18 87.44783 87.36522  0.082613838
19 87.35276 87.35899 -0.006227632
20 87.20657 87.28278 -0.076209135
```

At the current year, the appearance is that the unofficial end of summer is actually getting earlier by a very small amount. To answer the question, the end of summer has not gotten later over the 20 years.

Question 3

Describe a situation or problem from your job, everyday life, current events, etc., for which a linear regression model would be appropriate. List some (up to 5) predictors that you might use.

Response

A local ice cream shop may be able to predict the number of popsicle sales based on the following predictors: * temperature of the day * number of events happening in town * number of positive reviews the ice cream shop had in the past week on yelp * number of advertisements posted around town and online * % discount offered on coupons that week

Question 4

Using crime data from http://www.statsci.org/data/general/uscrime.txt (http://www.statsci.org/data/general/uscrime.txt) (description at http://www.statsci.org/data/general/uscrime.html (http://www.statsci.org/data/general/uscrime.html)), use regression (a useful R function is lm or glm) to predict the observed crime rate in a city with the following data: M = 14.0 So = 0 Ed = 10.0 Po1 = 12.0 Po2 = 15.5 LF = 0.640 M.F = 94.0 Pop = 150 NW = 1.1 U1 = 0.120 U2 = 3.6 Wealth = 3200 Ineq = 20.1 Prob = 0.04 Time = 39.0 Show your model (factors used and their coefficients), the software output, and the quality of fit.

Response

Load data and library

Hide

```
library(ggplot2)
library(graphics)
library(tsne)
library(caret)
```

```
Loading required package: lattice
```

Hide

```
#import crime data
crimeTbl <- read.csv('uscrime.csv', header=TRUE)
#crime data features
crimeFeat <- crimeTbl[,1:dim(crimeTbl)[2]-1]
#crime data labels
crimeLab <- crimeTbl[,dim(crimeTbl)[2]]
```

Linear Regression on all features

From here, we can run linear regression with Crime being the dependent variable and everything else as the independent variable. I decided to run it on the entire set of features for the first test. It returned the following coefficients for each variable.

	model.coefficients
(Intercept)	-5984.28760450
M	87.83017324
So	-3.80345030
Ed	188.32431475
Po1	192.80433828
Po2	-109.42192538
LF	-663.82614508
M.F	17.40685553
Pop	-0.73300815
NW	4.20446100
U1	-5827.10272440
U2	167.79967222
Wealth	0.09616624
Ineq	70.67209945

	model.coefficients
Prob	-4855.26581548
Time	-3.47901784

Linear Regression Performance on all features

To assess the performance of the model, I used the R^2 value along with a plot of the residuals to determine how well the model fits the data. First, we'll look at the r.squared value and also the adj.r.squared value. Searching online for the difference between the two, it looks like the r.squared value is the standard r.squared value we see describe by the formula below. This R^2 value will always increase with the number of variables you provide to the linear regression. adj.r.squared is actually a r.squared value that adjusts for that addition by taking into account the number of variables that gets added to the model.

$$R^2 = 1 - \frac{SS_{err}}{SS_{tot}}$$

SS_{err}: Sum of squared distances between the actual and predicted Y values SS_{tot}: Sum of squared distances between the actual Y values and their mean

Hide

```
modelSum <- summary(model)
print(paste0("r.squared value is: ",modelSum$r.squared))
```

```
[1] "r.squared value is: 0.803086758316909"
```

Hide

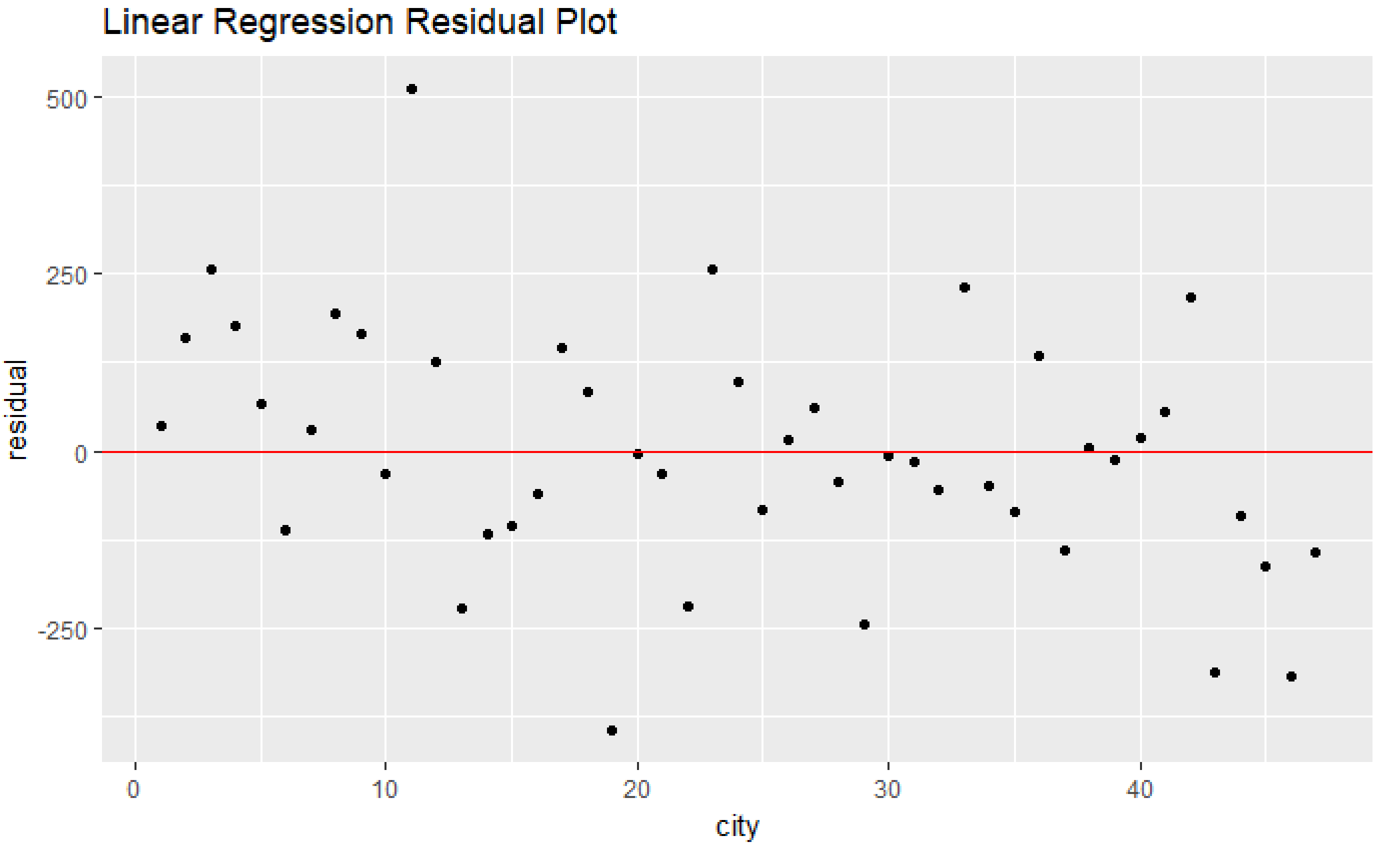
```
print(paste0("adj.r.squared value is: ",modelSum$adj.r.squared))
```

```
[1] "adj.r.squared value is: 0.70780615750251"
```

The difference between the two may explain overfitting if the difference is large. I think the difference is decently large so it is showing that we may be overfitting our data. Next, we'll take a look at the graph of the residuals

Hide

```
qplot(seq_along(model$residuals),model$residuals)+geom_hline(yintercept=0,colour='red')+labs(tit
le="Linear Regression Residual Plot",x="city",y="residual")
```



We can see a decent amount of variation in the datapoints some of which hitting up to 512 crimes when our max data value went up to 1993. There is a decent amount of variation so later on I'm going to see if I can narrow this down a bit more. But first, I used this model to predict the number of crimes based on the homework data provided to us.

Hide

```
print(paste0("Number of crimes in this city is: ",crime))
```

```
[1] "Number of crimes in this city is: 155.434896887448"
```

Picking Features

One awesome thing about lm in r is that you can pull the t value and the P value directly from the model. I'm going to use these values to pick out my features.

Hide

```
tpvalue <- data.frame(modelSum$coefficients[,3:4])
colnames(tpvalue) <- c('tValue','pValue')
kable(tpvalue[order(tpvalue[,2]),])
```

	tValue	pValue
(Intercept)	-3.6751336	0.0008930
Ineq	3.1110441	0.0039831

	tValue	pValue
Ed	3.0331654	0.0048614
Prob	-2.1366485	0.0406269
M	2.1055390	0.0434434
U2	2.0379878	0.0501613
Po1	1.8170288	0.0788920
U1	-1.3840149	0.1762380
Po2	-0.9314285	0.3588296
Wealth	0.9276542	0.3607538
M.F	0.8552122	0.3989953
NW	0.6487473	0.5212791
Pop	-0.5684193	0.5738452
Time	-0.4855386	0.6307084
LF	-0.4516657	0.6546541
So	-0.0255685	0.9797654

From the table above, you can see that the following values are good predictors because their p-value is 0.05 or smaller.

- Ineq: Income inequality: percentage of families earning below half the median income
- Ed: mean years of schooling of the population aged 25 years or over
- Prob: probability of imprisonment: ratio of number of commitments to number of offenses
- M: percentage of males aged 14-24 in total state population
- U2: unemployment rate of urban males 35-39
- Po1: per capita expenditure on police protection in 1960

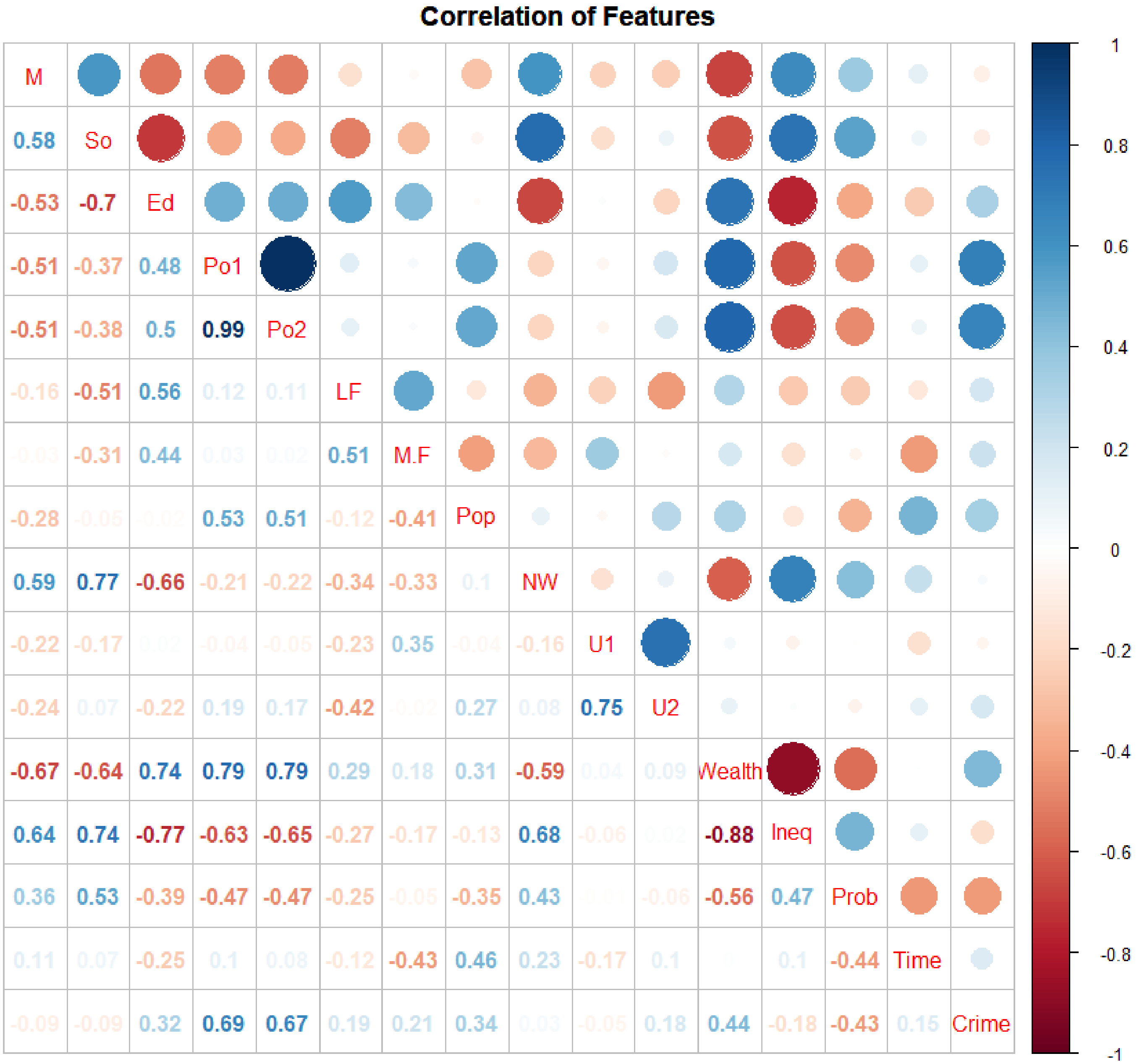
Below is a list of variables that are bad predictors.

- So: indicator variable for a southern state
- LF: labour force participation rate of civilian urban males in the age-group 14-24
- Time: average time in months served by offenders in state prisons before their first release
- Pop: state population in 1960 in hundred thousands
- NW: percentage of nonwhites in the population
- M.F: number of males per 100 females
- Wealth: wealth: median value of transferable assets or family income
- Po2: per capita expenditure on police protection in 1959
- U1:unemployment rate of urban males 14-24

Let’s also take a look at the correlation plot between all of the variable to find out which ones are highly correlated.

Hide

```
library(corrplot)
M <- cor(crimeTbl)
corrplot.mixed(M,title="Correlation of Features",mar=c(0,0,1,0))
```



From this figure, we can see the following features are highly correlated so we should probably only include one of them in the selected features. * Wealth :: Ineq * Ineq :: Ed * So :: Ed * Wealth :: Po1 :: Po2

Linear Regression on Selected Features

I actually tested this model by removing either Ineq and Ed and checked the adjusted r-squared value afound that removing either one significantly dropped the value. It seems that they may be helping the model in different ways.

Hide

```
featToUse <- c('Ineq','Ed','Prob','M','U2','Po1','Crime')
smallCrimeTbl <- crimeTbl[featToUse]
model2 <- lm(Crime~.,smallCrimeTbl)
model2Sum <- summary(model2)
print(paste0("r.squared value is: ",model2Sum$r.squared))
```

```
[1] "r.squared value is: 0.765866328518745"
```

Hide

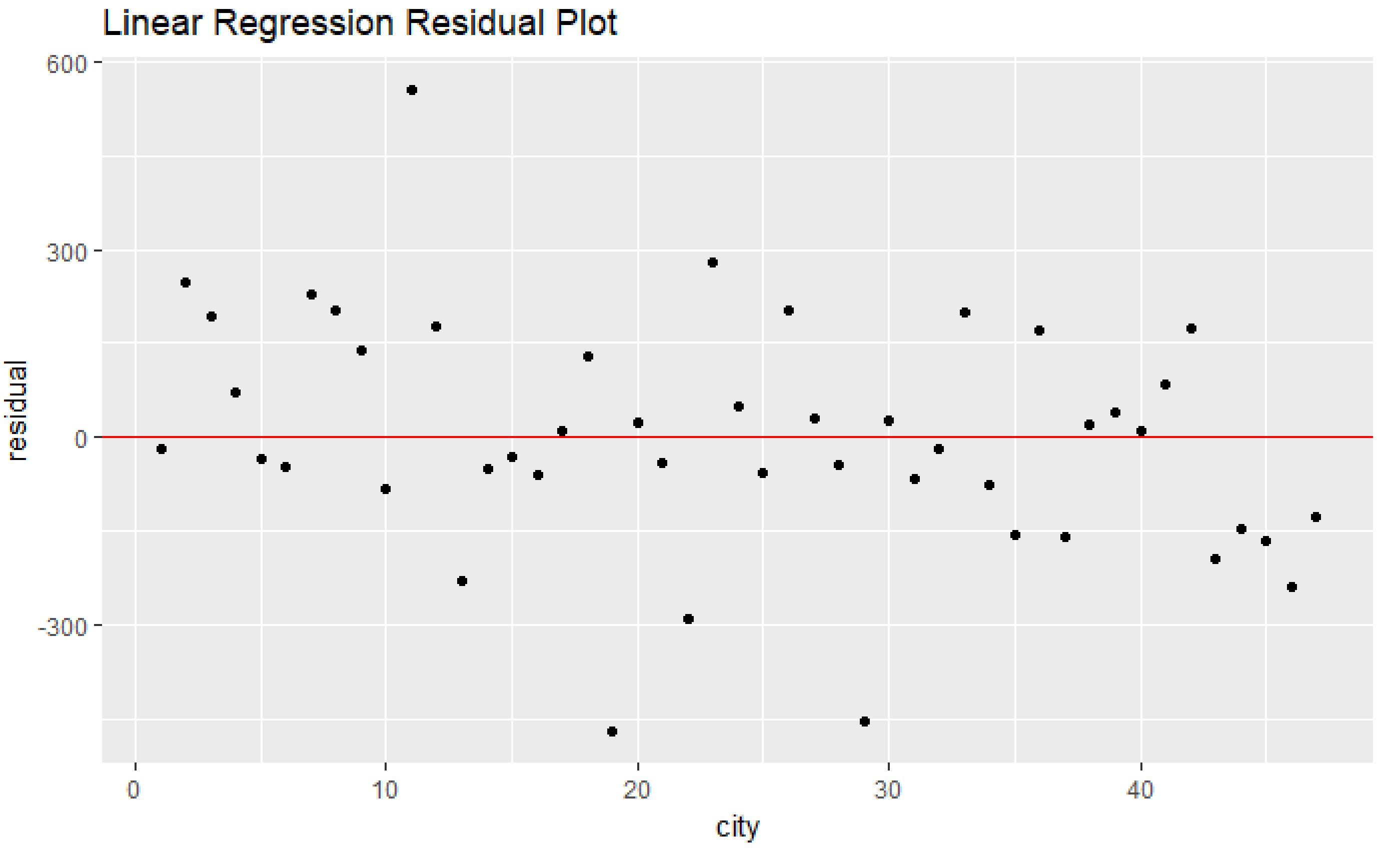
```
print(paste0("adj.r.squared value is: ",model2Sum$adj.r.squared))
```

```
[1] "adj.r.squared value is: 0.730746277796556"
```

We’re not seeing as large of a difference as we saw before.

Hide

```
qplot(seq_along(model2Sum$residuals),model2Sum$residuals)+geom_hline(yintercept=0,colour='red')+l
abs(title="Linear Regression Residual Plot",x="city",y="residual")
```



Because the scales are slightly adjusted, I can’t say for sure that the spread has changed, so it would be inconclusive from this chart. The best way to determine the difference is to run cross validation on both models.

Cross Validation

We can perform cross validation on both models. First one where we used all the features and the second one where we only used a series of features we hand picked.

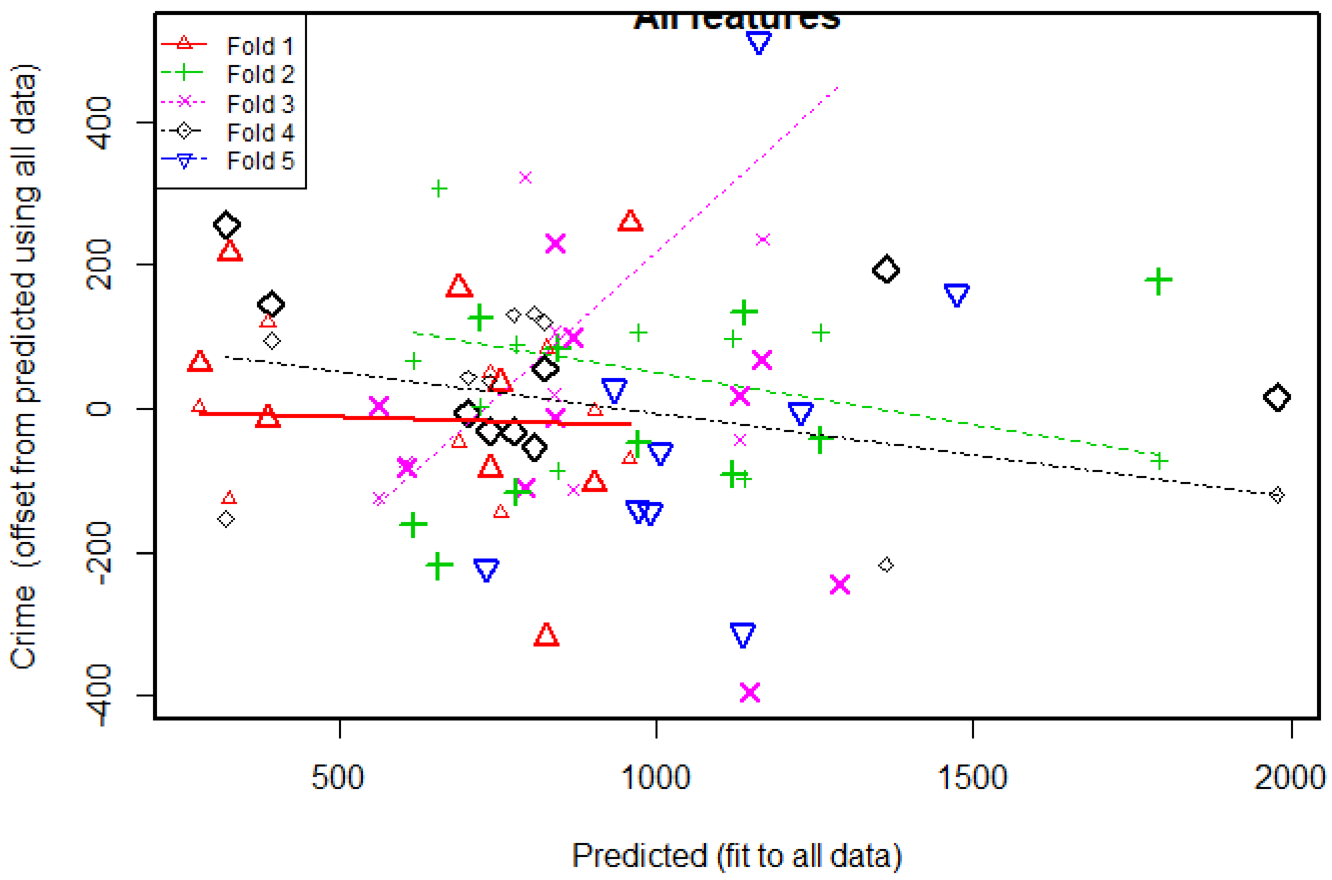
Hide

```
library(DAAG)
crimedf <- data.frame(crimeTbl)
modAll <-
CVlm(crimedf,form.lm=formula(Crime~.),m=5,seed=44,printit=FALSE,plotit="Residual",main="All feat
ures")
```

```
prediction from a rank-deficient fit may be misleadingprediction from a rank-deficient fit may b
e misleadingprediction from a rank-deficient fit may be misleadingprediction from a rank-deficie
nt fit may be misleadingprediction from a rank-deficient fit may be misleading
```

```


 As there is >1 explanatory variable, cross-validation
 predicted values for a fold are not a linear function
 of corresponding overall predicted values.  Lines that
 are shown for the different folds are approximate
```

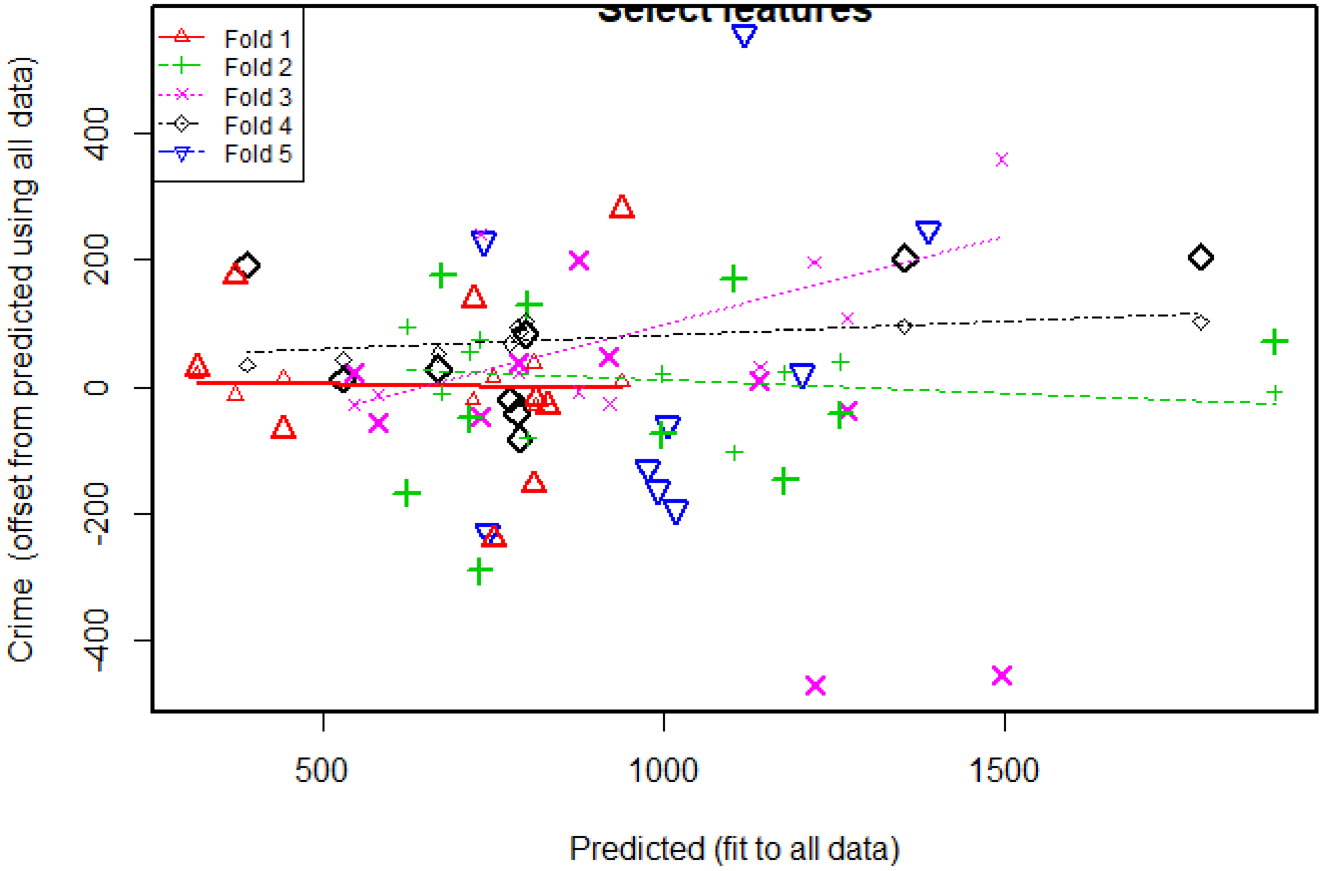


Hide

```
modSel <- CVlm(smallCrimeTbl,form.lm=formula(Crime~.),m=5,seed=44,printit=FALSE,plotit="Residua
l",main="Select features")
```

```
prediction from a rank-deficient fit may be misleadingprediction from a rank-deficient fit may b
e misleadingprediction from a rank-deficient fit may be misleadingprediction from a rank-deficie
nt fit may be misleadingprediction from a rank-deficient fit may be misleading
```

```
 As there is >1 explanatory variable, cross-validation
 predicted values for a fold are not a linear function
 of corresponding overall predicted values.  Lines that
 are shown for the different folds are approximate
```



From plotting out the residuals it looks like model2(select features) is more consistent and form a tighter grouping than model1(all features). So we go ahead and take a look at the R^2 value and also the mean squared error of both plots.

Hide

```
SStot <- sum((modAll$Crime-mean(modAll$Crime))^2)
SSres_mod1 <- sum((modAll$Predicted-modAll$Crime)^2)
SSres_mod2 <- sum((modSel$Predicted-modSel$Crime)^2)
#Calculate R^2
mod1R2 <- 1-SSres_mod1/SStot
mod2R2 <- 1-SSres_mod2/SStot
print(paste0('all features R^2: ',mod1R2))
```

```
[1] "all features R^2: 0.803086758316909"
```

Hide

```
print(paste0('select features R^2: ',mod2R2))
```

```
[1] "select features R^2: 0.765866328518745"
```

Hide

```
print(paste0('all features mean square: ',attr(modAll,"ms")))
```

```
[1] "all features mean square: 234975.270850653"
```

Hide

```
print(paste0('select features mean square: ',attr(modSel,"ms")))
```

```
[1] "select features mean square: 367815.594652495"
```

When we look at the cross validation data, we can see that the model with all features actually does better than the selected features model. This actually comes as a surprise to me since the adjusted r squared value displayed better performance for the select features. But this is the reason why we perform cross validation to confirm.