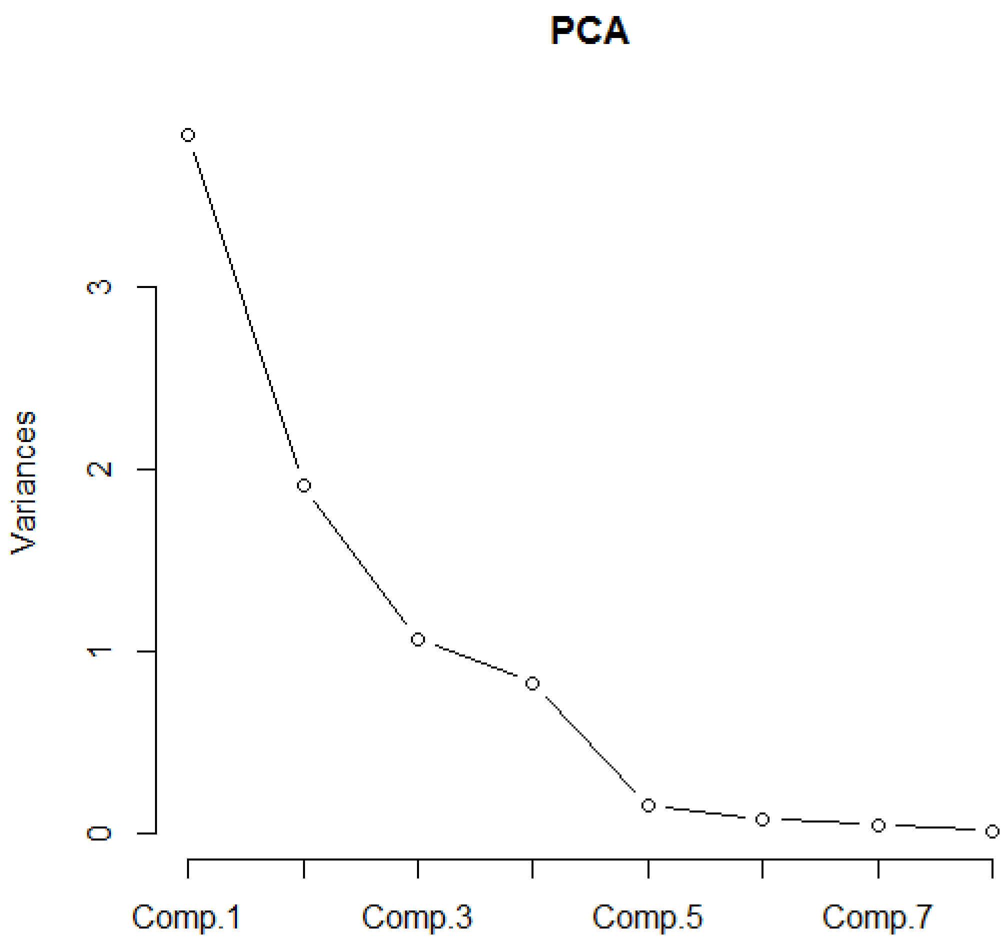
```
R version 3.2.2 (2015-08-14) -- "Fire Safety"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> setwd("C:/Users/TonyLaptop/Desktop/rowan/DM1/lecture11/lecture11")
> houses=read.csv("Houses.csv",header=TRUE,sep=",")
> minn = min(houses$MedianHomeVal)
> maxx = max(houses$MedianHomeVal)
> houses$MedianHomeVal2 <- sapply(houses$MedianHomeVal,function(b) {
+       return((b-minn)/(maxx-minn))
+   }
+  )
> m<-mean(houses$MedianHomeVal)
> sigma<-sd(houses$MedianHomeVal)
> houses$zMedHomeVal<-(houses$MedianHomeVal-m)/sigma
> m<-mean(houses$MedianIncome)
> sigma<-sd(houses$MedianIncome)
> houses$zMedInc<-(houses$MedianIncome-m)/sigma
> m<-mean(houses$MedianHomeAge)
> sigma<-sd(houses$MedianHomeAge)
> houses$zMedHomeAge<-(houses$MedianHomeAge-m)/sigma
> m<-mean(houses$TotalBedRooms)
> sigma<-sd(houses$TotalBedRooms)
> houses$zBedrooms<-(houses$TotalBedRooms-m)/sigma
> m<-mean(houses$TotalRooms)
> sigma<-sd(houses$TotalRooms)
> houses$zTotRooms<-(houses$TotalRooms-m)/sigma
> m<-mean(houses$Population)
> sigma<-sd(houses$Population)
> houses$zPop<-(houses$Population-m)/sigma
> m<-mean(houses$Households)
> sigma<-sd(houses$Households)
> houses$zHouseholds<-(houses$Households-m)/sigma
> m<-mean(houses$Latitude)
> sigma<-sd(houses$Latitude)
> houses$zLatitude<-(houses$Latitude-m)/sigma
> m<-mean(houses$Longitude)
> sigma<-sd(houses$Longitude)
> houses$zLongitude<-(houses$Longitude-m)/sigma
> vars<-
c("MedianHomeVal2","zMedInc","zMedHomeAge","zBedrooms","zTotRooms","zPop","zHouseholds","zLatitude","
zLongitude")
> houses2<-houses[vars]
> head(houses2)
  MedianHomeVal2     zMedInc zMedHomeAge  zBedrooms  zTotRooms       zPop
1      0.9022664  2.34470896   0.9821189 -0.9706826 -0.8047996 -0.9744050
2      0.7082466  2.33218146  -0.6070042  1.3486168  2.0458405  0.8614180
3      0.6950507  1.78265622   1.8561366 -0.8258748 -0.5357329 -0.8207575
4      0.6727828  0.93294491   1.8561366 -0.7190493 -0.6241995 -0.7660095
5      0.6746385 -0.01288068   1.8561366 -0.6122238 -0.4623928 -0.7598283
6      0.5251545  0.08744452   1.8561366 -0.7712751 -0.7869229 -0.8940491
  zHouseholds zLatitude zLongitude
1  -0.9770092  1.052523  -1.327803
2   1.6699206  1.043159  -1.322812
3  -0.8436165  1.038478  -1.332794
4  -0.7337637  1.038478  -1.337785
5  -0.6291419  1.038478  -1.337785
6  -0.8017678  1.038478  -1.337785
>
> dim(houses2)
[1] 20640     9
> library("neuralnet")
Loading required package: grid
Loading required package: MASS
> set.seed(2)
> train=sample(1:20640,16000)
> test=-train
> trainingData=houses2[train,]
> testingData=houses2[test,]
> dim(trainingData)
[1] 16000     9
> dim(testingData)
[1] 4640    9
> names(trainingData)
[1] "MedianHomeVal2" "zMedInc"        "zMedHomeAge"    "zBedrooms"
[5] "zTotRooms"      "zPop"           "zHouseholds"    "zLatitude"
[9] "zLongitude"
> formula=
MedianHomeVal2~zMedInc+zMedHomeAge+zBedrooms+zTotRooms+zPop+zHouseholds+zLatitude+zLongitude
> names(testingData)
[1] "MedianHomeVal2" "zMedInc"        "zMedHomeAge"    "zBedrooms"      "zTotRooms"      "zPop"
"zHouseholds"
[8] "zLatitude"      "zLongitude"
> dim(testingData)
[1] 4640    9
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=5, threshold=0.2)
> proc.time() - ptm
   user  system elapsed
 152.46    3.33  301.99
> results<-compute(nnet,testingData[,2:9])
> dim(results$net.result)
[1] 4640    1
>
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4640   11
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2007115494
```

```
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=8, threshold=0.15)
> proc.time() - ptm
   user  system elapsed 
 435.18   23.34  458.57 
> results<-compute(nnet,testingData[,2:9])
> dim(results$net.result)
[1] 4640    1
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.1898130528
> names(houses2)
[1] "MedianHomeVal2" "zMedInc"        "zMedHomeAge"    "zBedrooms"      "zTotRooms"      "zPop"           
"zHouseholds"   
[8] "zLatitude"      "zLongitude"    
> names(trainingData)
[1] "MedianHomeVal2" "zMedInc"        "zMedHomeAge"    "zBedrooms"      "zTotRooms"      "zPop"           
"zHouseholds"   
[8] "zLatitude"      "zLongitude"    
> dim(trainingData)
[1] 16000     9
> PCA<-princomp(trainingData[,2:9])
> screeplot(PCA,type="lines")
```



```
> PCA$loadings
```

```
Loadings:
            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
zMedInc                    0.897 -0.395                0.171
zMedHomeAge  0.219        -0.381 -0.891
zBedrooms   -0.490        -0.114         0.377 -0.216  0.230 -0.705
zTotRooms   -0.479               -0.111  0.307  0.546 -0.574  0.144
zPop        -0.475        -0.113        -0.850  0.117        -0.123
zHouseholds -0.493        -0.106         0.160 -0.388  0.308  0.679
zLatitude           0.701         0.109         0.482  0.505
zLongitude         -0.701                       0.497  0.486

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
> trainingData$C1<-0
> trainingData$C2<-0
> trainingData$C3<-0
> trainingData$C4<-0
> trainingData$C5<-0
> for (i in 1:16000){
+  trainingData$C1[i]<-trainingData$zMedHomeAge[i]*.219-trainingData$zBedrooms[i]*.49-
trainingData$zTotRooms[i]*.479-trainingData$zHouseholds[i]*.493-trainingData$zPop[i]*.475
+  trainingData$C2[i]<-trainingData$zLatitude[i]*.701-trainingData$zLongitude[i]*.701
+  trainingData$C3[i]<-trainingData$zMedInc[i]*.897-trainingData$zMedHomeAge[i]*.381-
trainingData$zBedrooms[i]*.114-trainingData$zPop[i]*.113-trainingData$zHouseholds[i]*.106
+  trainingData$C4[i]<-trainingData$zLatitude[i]*.109-trainingData$zMedInc[i]*.395-
trainingData$zMedHomeAge[i]*.891-trainingData$zTotRooms[i]*.111
+  trainingData$C5[i]<-trainingData$zBedrooms[i]*.377+trainingData$zTotRooms[i]*.307-
trainingData$zPop[i]*.85+trainingData$zHouseholds[i]*.16
+ }




> head(trainingData)
      MedianHomeVal2       zMedInc   zMedHomeAge      zBedrooms     zTotRooms          zPop
zHouseholds     zLatitude    zLongitude
```

```
3816    0.4329899671 -1.3233720718 -0.2891795935 -0.21103490916 -0.6897472321 -0.5999995330 -
0.2080394728 -0.6656803006  0.5389006047
14497   0.8298955468  0.6481287088 -1.0837411305 -0.09471385615  0.1188279732 -0.2308922649 -
0.1478819784 -1.3023986103  1.1877548179
11833   0.3298976087 -0.4582908989 -1.0042849768 -0.85436154931 -0.8103001114 -1.2198878158 -
1.2516412251  1.7173316083 -0.3794776662
3469    0.3245368060 -0.4582908989  0.1875573287 -0.66919742410 -0.7080822708 -0.4710768987 -
0.6840683423 -0.6188627779  0.5438917910
19478   0.2195887852 -0.7741100068  0.1875573287 -0.58136315958 -0.4527668573 -0.5593800729 -
0.5375979209  0.9542059871 -0.6989135865
19469   0.2016507148 -0.3610186137 -0.1302672861  0.53199549071  0.6280836722  0.3360141134
0.5713924123  0.9588877394 -0.7039047728
                  C1            C2            C3            C4             C5
3816   0.75802893700 -0.8444112147 -0.9629772123 0.78439377615  0.18540072636
14497 -0.06526847607 -1.7455975532  1.0468405178 0.55445115371  0.17337037261
11833  1.78333833910  1.4698633014  0.3394641495 1.35297527704  0.26578560911
3469   1.26916042014 -0.8150909528 -0.2805148384 0.02505241448 -0.17870425692
19478  1.07355963748  1.1588368212 -0.5793652902 0.29292544656  0.03128405822
19469 -1.03136156824  1.1656175511 -0.4333865370 0.29347198030  0.19919477694
> formula=MedianHomeVal2~C1+C2+C3+C4+C5
> dim(testingData)
[1] 4640   13
> testingData$C1<-0
> testingData$C2<-0
> testingData$C3<-0
> testingData$C4<-0
> testingData$C5<-0
> for (i in 1:4640){
+  testingData$C1[i]<-testingData$zMedHomeAge[i]*.219-testingData$zBedrooms[i]*.49-
testingData$zTotRooms[i]*.479-testingData$zHouseholds[i]*.493-testingData$zPop[i]*.475
+  testingData$C2[i]<-testingData$zLatitude[i]*.701-testingData$zLongitude[i]*.701
+  testingData$C3[i]<-testingData$zMedInc[i]*.897-testingData$zMedHomeAge[i]*.381-
testingData$zBedrooms[i]*.114-testingData$zPop[i]*.113-testingData$zHouseholds[i]*.106
+  testingData$C4[i]<-testingData$zLatitude[i]*.109-testingData$zMedInc[i]*.395-
testingData$zMedHomeAge[i]*.891-testingData$zTotRooms[i]*.111
+  testingData$C5[i]<-testingData$zBedrooms[i]*.377+testingData$zTotRooms[i]*.307-
testingData$zPop[i]*.85+testingData$zHouseholds[i]*.16
+ }
> names(testingData)
```

```
 [1] "MedianHomeVal2" "zMedInc"        "zMedHomeAge"    "zBedrooms"      "zTotRooms"      "zPop"
"zHouseholds"
 [8] "zLatitude"      "zLongitude"     "guess"          "Actual"         "absDiff"        "dummy"
"C1"
[15] "C2"             "C3"             "C4"             "C5"
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=8, threshold=0.15)
> proc.time() - ptm
   user  system elapsed
 282.94    2.06  285.01
> results<-compute(nnet,testingData[,14:18])
> dim(results$net.result)
[1] 4640    1
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4640   18
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2332437547
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=5, threshold=0.15)
> proc.time() - ptm
   user  system elapsed
 199.15    4.89  204.13
> results<-compute(nnet,testingData[,14:18])
> dim(results$net.result)
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4640   18
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2495409405
```

```
> #might be getting over trained; test on training data and see if results are better
> #if results are much better then the model is overtrained
> dim(trainingData)
[1] 16000    14
> names(trainingData)
 [1] "MedianHomeVal2" "zMedInc"        "zMedHomeAge"    "zBedrooms"      "zTotRooms"      "zPop"
"zHouseholds"
 [8] "zLatitude"      "zLongitude"     "C1"             "C2"             "C3"             "C4"
"C5"
> dim(testingData)
[1] 4640   18
> #save testing data
> t2<-testingData
> #use the first 4640 rows of training data as testing data
> testingData<-trainingData[1:4640,]
> dim(testingData)
[1] 4640   14
> results<-compute(nnet,testingData[,10:14])
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4640   18
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2474397734
> #now turn the last 4640 rows of training data into testing data
> dim(trainingData)
[1] 16000    14
> 16000-4640
[1] 11360
> testingData<-trainingData[11360:16000,]
> results<-compute(nnet,testingData[,10:14])
> dim(results$net.result)
[1] 4641    1
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4641   16
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2475267321
> #apparently not overtrained
> #go back to original training data
> testingData<-t2
> dim(testingData)
[1] 4640   18
> PCA$loadings

Loadings:
            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
zMedInc                    0.897 -0.395                0.171       
zMedHomeAge  0.219        -0.381 -0.891                            
zBedrooms   -0.490        -0.114         0.377 -0.216  0.230 -0.705
zTotRooms   -0.479               -0.111  0.307  0.546 -0.574  0.144
zPop        -0.475        -0.113        -0.850  0.117        -0.123
zHouseholds -0.493        -0.106         0.160 -0.388  0.308  0.679
zLatitude           0.701         0.109         0.482  0.505       
zLongitude         -0.701                       0.497  0.486       

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

```
> #perhaps the issue is that we only have 3 digits of accuracy from the loadings above
> #let's see if we can get a few more digits
> load <- with(PCA, unclass(loadings))
> load
                  Comp.1         Comp.2         Comp.3         Comp.4         Comp.5         Comp.6
Comp.7          Comp.8
zMedInc     -0.04256235977 -0.03721263091  0.89697310056 -0.39498145961 -0.05292607283 -0.05069985665
0.17132239177 -0.038853088520
zMedHomeAge  0.21904336969  0.02846998302 -0.38093625978 -0.89131163277  0.03594194492  0.09428586745
0.03827333138 -0.003755720344
zBedrooms   -0.48986944891  0.06379535786 -0.11350576403 -0.06462876723  0.37677755094 -0.21583563847
0.22999775707 -0.705303002500
zTotRooms   -0.47921520638  0.07697382345  0.09619023484 -0.11149814146  0.30680339698  0.54639593644
-0.57384888422  0.144097307274
zPop        -0.47516370568  0.03234963325 -0.11344625820 -0.08755608656 -0.85042459610  0.11714898519
0.02520026393 -0.122702487969
zHouseholds -0.49270345689  0.06718850658 -0.10571681754 -0.09795929987  0.15979221377 -0.38785615930
0.30838107295  0.678890344726
zLatitude    0.07662414023  0.70066739522  0.01583449593  0.10881449345  0.04618716232  0.48199697936
0.50506157980  0.039510450552
zLongitude  -0.07861713696 -0.70093813570 -0.06240090409  0.06198241721  0.09449106267  0.49691929364
0.48596413567  0.052448988161
>
> trainingData$C1<-0
> trainingData$C2<-0
> trainingData$C3<-0
> trainingData$C4<-0
> trainingData$C5<-0
> for (i in 1:16000){
+  trainingData$C1[i]<-trainingData$zMedHomeAge[i]*.21904336969-
trainingData$zBedrooms[i]*.48986944891-trainingData$zTotRooms[i]*.47921520638-
trainingData$zHouseholds[i]*.49270345689-trainingData$zPop[i]*.47516370568
+  trainingData$C2[i]<-trainingData$zLatitude[i]*.70066739522-trainingData$zLongitude[i]*.70093813570
+  trainingData$C3[i]<-trainingData$zMedInc[i]*.89697310056 -
trainingData$zMedHomeAge[i]*.38093625978-trainingData$zBedrooms[i]*.11350576403-
trainingData$zPop[i]*.11344625820-trainingData$zHouseholds[i]*.10571681754
+  trainingData$C4[i]<-trainingData$zLatitude[i]*.10881449345-trainingData$zMedInc[i]*.39498145961-
trainingData$zMedHomeAge[i]*.89131163277-trainingData$zTotRooms[i]*.11149814146
+  trainingData$C5[i]<-trainingData$zBedrooms[i]*.37677755094+trainingData$zTotRooms[i]*.30680339698-
trainingData$zPop[i]*.85042459610+trainingData$zHouseholds[i]*.15979221377
+ }
> testingData$C1<-0
> testingData$C2<-0
> testingData$C3<-0
> testingData$C4<-0
> testingData$C5<-0
> for (i in 1:4640){
+  testingData$C1[i]<-testingData$zMedHomeAge[i]*.21904336969-testingData$zBedrooms[i]*.48986944891-
testingData$zTotRooms[i]*.47921520638-testingData$zHouseholds[i]*.49270345689-
testingData$zPop[i]*.47516370568
+  testingData$C2[i]<-testingData$zLatitude[i]*.70066739522-testingData$zLongitude[i]*.70093813570
+  testingData$C3[i]<-testingData$zMedInc[i]*.89697310056 -testingData$zMedHomeAge[i]*.38093625978-
testingData$zBedrooms[i]*.11350576403-testingData$zPop[i]*.11344625820-
testingData$zHouseholds[i]*.10571681754
+  testingData$C4[i]<-testingData$zLatitude[i]*.10881449345-testingData$zMedInc[i]*.39498145961-
testingData$zMedHomeAge[i]*.89131163277-testingData$zTotRooms[i]*.11149814146
+  testingData$C5[i]<-testingData$zBedrooms[i]*.37677755094+testingData$zTotRooms[i]*.30680339698-
testingData$zPop[i]*.85042459610+testingData$zHouseholds[i]*.15979221377
+ }
>
> formula
MedianHomeVal2 ~ C1 + C2 + C3 + C4 + C5
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=8, threshold=0.15)
> proc.time() - ptm
   user  system elapsed
 287.31    8.14  295.53
> results<-compute(nnet,testingData[,14:18])
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4640   18
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2344961167
> #a little bit better, but not much
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=10, threshold=0.1)
> proc.time() - ptm
   user  system elapsed
 769.07   28.44  797.63
> results<-compute(nnet,testingData[,14:18])
> dim(results$net.result)
[1] 4640    1
> testingData$guess<-results$net.result
> testingData$Actual<-(testingData$MedianHomeVal2*(maxx-minn))+minn
> testingData$guess<-(testingData$guess*(maxx-minn))+minn
> dim(testingData)
[1] 4640   18
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2349090405
> #no improvement
> #A common trick is to change the target variable to log of that variable
> #this has the affect of making non-linear data mostly linear
> trainingData$LogHomeVal<-log(trainingData$Actual)
> head(trainingData)
      MedianHomeVal2       zMedInc   zMedHomeAge      zBedrooms     zTotRooms          zPop
zHouseholds     zLatitude    zLongitude
3816    0.4329899671 -1.3233720718 -0.2891795935 -0.21103490916 -0.6897472321 -0.5999995330 -
0.2080394728 -0.6656803006  0.5389006047
14497   0.8298955468  0.6481287088 -1.0837411305 -0.09471385615  0.1188279732 -0.2308922649 -
0.1478819784 -1.3023986103  1.1877548179
11833   0.3298976087 -0.4582908989 -1.0042849768 -0.85436154931 -0.8103001114 -1.2198878158 -
1.2516412251  1.7173316083 -0.3794776662
3469    0.3245368060 -0.4582908989  0.1875573287 -0.66919742410 -0.7080822708 -0.4710768987 -
0.6840683423 -0.6188627779  0.5438917910
19478   0.2195887852 -0.7741100068  0.1875573287 -0.58136315958 -0.4527668573 -0.5593800729 -
0.5375979209  0.9542059871 -0.6989135865
19469   0.2016507148 -0.3610186137 -0.1302672861  0.53199549071  0.6280836722  0.3360141134
0.5713924123  0.9588877394 -0.7039047728
                  C1            C2            C3            C4             C5 Actual  LogHomeVal
3816   0.75817381320 -0.8441564675 -0.9628555061 0.78492643791  0.18588126247 225000 12.32385568
14497 -0.06535947012 -1.7450908895  1.0467683546 0.55498330985  0.17349684347 417500 12.94203982
11833  1.78318616447  1.4692686326  0.3391801498 1.35336481608  0.26691300191 175000 12.07254125
3469   1.26910783603 -0.8148514686 -0.2808047913 0.02545299681 -0.17807402608 172400 12.05757264
19478  1.07352146524  1.1584762099 -0.5795224531 0.29290117573  0.03185161328 121500 11.70766954
19469 -1.03131849785  1.1652550738 -0.4331103385 0.29301512796  0.19869155414 112800 11.63337162
> max2<-max(trainingData$LogHomeVal)
> min2<-min(trainingData$LogHomeVal)
> max2
[1] 13.12236538
> min2
[1] 9.615738811
> trainingData$MedianHomeVal3<-(trainingData$LogHomeVal-min2)/(max2-min2)
> head(trainingData)
```

```
      MedianHomeVal2       zMedInc   zMedHomeAge      zBedrooms     zTotRooms          zPop
zHouseholds     zLatitude    zLongitude
3816    0.4329899671 -1.3233720718 -0.2891795935 -0.21103490916 -0.6897472321 -0.5999995330 -
0.2080394728 -0.6656803006  0.5389006047
14497   0.8298955468  0.6481287088 -1.0837411305 -0.09471385615  0.1188279732 -0.2308922649 -
0.1478819784 -1.3023986103  1.1877548179
11833   0.3298976087 -0.4582908989 -1.0042849768 -0.85436154931 -0.8103001114 -1.2198878158 -
1.2516412251  1.7173316083 -0.3794776662
3469    0.3245368060 -0.4582908989  0.1875573287 -0.66919742410 -0.7080822708 -0.4710768987 -
0.6840683423 -0.6188627779  0.5438917910
19478   0.2195887852 -0.7741100068  0.1875573287 -0.58136315958 -0.4527668573 -0.5593800729 -
0.5375979209  0.9542059871 -0.6989135865
19469   0.2016507148 -0.3610186137 -0.1302672861  0.53199549071  0.6280836722  0.3360141134
0.5713924123  0.9588877394 -0.7039047728
                  C1            C2            C3            C4             C5 Actual  LogHomeVal
MedianHomeVal3
3816   0.75817381320 -0.8441564675 -0.9628555061 0.78492643791  0.18588126247 225000 12.32385568
0.7722855054
14497 -0.06535947012 -1.7450908895  1.0467683546 0.55498330985  0.17349684347 417500 12.94203982
0.9485757748
11833  1.78318616447  1.4692686326  0.3391801498 1.35336481608  0.26691300191 175000 12.07254125
0.7006170732
3469   1.26910783603 -0.8148514686 -0.2808047913 0.02545299681 -0.17807402608 172400 12.05757264
0.6963484078
19478  1.07352146524  1.1584762099 -0.5795224531 0.29290117573  0.03185161328 121500 11.70766954
0.5965650151
19469 -1.03131849785  1.1652550738 -0.4331103385 0.29301512796  0.19869155414 112800 11.63337162
0.5753771520
> exp(12.32285568)
[1] 224775.1122
> #rounding costs us $25 which is about .01% error
> formula
MedianHomeVal2 ~ C1 + C2 + C3 + C4 + C5
> #rerun model with MedianHomeVal3 which is normalized Log of median home val
> formula<-MedianHomeVal3~C1+C2+C3+C4+C5
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=8, threshold=0.1)
> proc.time() - ptm
   user  system elapsed
 970.09   19.77  990.00
```

```
> results<-compute(nnet,testingData[,14:18])
> testingData$guess<-results$net.result
> testingData$guess<-testingData$guess*(max2-min2)
> testingData$guess<-testingData$guess+min2
> testingData$guess<-exp(testingData$guess)
> testingData$absDiff<-abs(testingData$guess-testingData$Actual)
> avgErr<-mean(testingData$absDiff)
> avgHomeVal<-mean(testingData$Actual)
> avgErr/avgHomeVal
[1] 0.2341468598
> #no improvement with log home values
> #Let's try a regression model to see if we can get a better result
> regmodel=lm(trainingData$MedianHomeVal3~trainingData$C1+trainingData$C2+trainingData$C3+
trainingData$C4+trainingData$C5)
> summary(regmodel)

Call:
lm(formula = trainingData$MedianHomeVal3 ~ trainingData$C1 + 
    trainingData$C2 + trainingData$C3 + trainingData$C4 + trainingData$C5)

Residuals:
        Min          1Q      Median          3Q         Max 
-0.77864590 -0.07110792  0.00272059  0.07356449  0.68574262 

Coefficients:
                     Estimate    Std. Error   t value               Pr(>|t|)    
(Intercept)      0.7045675482  0.0009144070 770.51854 < 0.000000000000000222 ***
trainingData$C1 -0.0195710714  0.0004945693 -39.57195 < 0.000000000000000222 ***
trainingData$C2 -0.0041434612  0.0006735235  -6.15192       0.00000000078374 ***
trainingData$C3  0.0808126417  0.0009068346  89.11508 < 0.000000000000000222 ***
trainingData$C4 -0.0779133549  0.0010346019 -75.30757 < 0.000000000000000222 ***
trainingData$C5  0.0517195795  0.0023857551  21.67850 < 0.000000000000000222 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1156565 on 15994 degrees of freedom
Multiple R-squared:  0.4933714, Adjusted R-squared:  0.493213 
F-statistic: 3115.095 on 5 and 15994 DF,  p-value: < 0.00000000000000022204
> #terrible R squared. This is not going to work too well
> dim(testingData)
```

```
[1] 4640   18
> for (i in 1:4640)
+ {
+   testingData$guess[i]<-testingData$C5[i]*.0517195795-
trainingData$C4[i]*.0779133549+trainingData$C3[i]*.0808126417-trainingData$C2[i]*
+ .0041434612-trainingData$C1[i]*.0195710714+.7045675482
+ }
> testingData$guess<-testingData$guess*(max2-min2)
> testingData$guess<-testingData$guess+min2
> testingData$guess<-exp(testingData$guess)
> mean(testingData$absDiff)
[1] 110002.2071
> mean(testingData$Actual)
[1] 204041.6976
> mean(testingData$absDiff)/mean(testingData$Actual)
[1] 0.53911631
> #awful
>
> #next thing to try is the use of user defined composite variable
> #go back to original houses data
> head(houses)
  MedianHomeVal MedianIncome MedianHomeAge TotalRooms TotalBedRooms Population Households Latitude
Longitude MedianHomeVal2
1        452600       8.3252            41        880           129        322        126    37.88
-122.23   0.9022663824
2        358500       8.3014            21       7099          1106       2401       1138    37.86
-122.22   0.7082465639
3        352100       7.2574            52       1467           190        496        177    37.85
-122.24   0.6950507421
4        341300       5.6431            52       1274           235        558        219    37.85
-122.25   0.6727827926
5        342200       3.8462            52       1627           280        565        259    37.85
-122.25   0.6746384551
6        269700       4.0368            52        919           213        413        193    37.85
-122.25   0.5251545354
   zMedHomeVal        zMedInc   zMedHomeAge     zBedrooms     zTotRooms          zPop   zHouseholds
zLatitude   zLongitude
1 2.1295798911  2.34470895612  0.9821188657 -0.9706826023 -0.8047995998 -0.9744049915 -0.9770091850
1.052522785 -1.327803055
```

```
2 1.3141243014  2.33218146484 -0.6070042083  1.3486167609  2.0458405374  0.8614179998  1.6699205726
1.043159280 -1.322811868
3 1.2586629222  1.78265621721  1.8561365564 -0.8258747608 -0.5357329073 -0.8207574685 -0.8436164799
1.038477528 -1.332794241
4 1.1650718450  0.93294490759  1.8561365564 -0.7190493040 -0.6241994689 -0.7660095005 -0.7337636638
1.038477528 -1.337785427
5 1.1728711014 -0.01288068384  1.8561365564 -0.6122238471 -0.4623927526 -0.7598282783 -0.6291419343
1.038477528 -1.337785427
6 0.5445976662  0.08744451941  1.8561365564 -0.7712750829 -0.7869229371 -0.8940491030 -0.8017677880
1.038477528 -1.337785427
> min3<-min(houses$MedianIncome)
> max3<-max(houses$MedianIncome)
> houses$medInc<-(houses$MedianIncome-min3)/(max3-min3)
> names(houses)
 [1] "MedianHomeVal"  "MedianIncome"   "MedianHomeAge"  "TotalRooms"     "TotalBedRooms"
"Population"     "Households"
 [8] "Latitude"       "Longitude"      "MedianHomeVal2" "zMedHomeVal"    "zMedInc"
"zMedHomeAge"    "zBedrooms"
[15] "zTotRooms"      "zPop"           "zHouseholds"    "zLatitude"      "zLongitude"     "medInc"
> min3<-min(houses$MedianHomeAge)
> max3<-max(houses$MedianHomeAge)
> houses$medAge<-(houses$MedianHomeAge-min3)/(max3-min3)
> min3<-min(houses$TotalRooms)
> max3<-max(houses$TotalRooms)
> houses$tRooms<-(houses$TotalRooms-min3)/(max3-min3)
> min3<-min(houses$Population)
> max3<-max(houses$Population)
> houses$Pop<-(houses$Population-min3)/(max3-min3)
> min3<-min(houses$Households)
> max3<-max(houses$Households)
> houses$Hholds<-(houses$Households-min3)/(max3-min3)
> min3<-min(houses$Latitude)
> max3<-max(houses$Latitude)
> houses$lat<-(houses$Latitude-min3)/(max3-min3)
> min3<-min(houses$Longitude)
> max3<-max(houses$Longitude)
> houses$long<-(houses$Longitude-min3)/(max3-min3)
> min3<-min(houses$TotalBedRooms)
> max3<-max(houses$TotalBedRooms)
> houses$tBedRooms<-(houses$TotalBedRooms-min3)/(max3-min3)
> names(houses)
 [1] "MedianHomeVal"  "MedianIncome"   "MedianHomeAge"  "TotalRooms"     "TotalBedRooms"
"Population"     "Households"
 [8] "Latitude"       "Longitude"      "MedianHomeVal2" "zMedHomeVal"    "zMedInc"
"zMedHomeAge"    "zBedrooms"
[15] "zTotRooms"      "zPop"           "zHouseholds"    "zLatitude"      "zLongitude"     "medInc"
"medAge"
[22] "tRooms"         "Pop"            "Hholds"         "lat"            "long"           "tBedRooms"
> vars<-c("MedianHomeVal2","medInc","medAge","tRooms","Pop","Hholds","lat","long","tBedRooms")
> houses3<-houses[vars]
> dim(houses3)
[1] 20640     9
> head(houses3)
  MedianHomeVal2       medInc       medAge        tRooms            Pop        Hholds          lat
long     tBedRooms
1   0.9022663824 0.5396684184 0.7843137255 0.02233073910 0.008940833544 0.02055582963 0.5674814028
0.2111553785 0.01986343886
2   0.7082465639 0.5380270617 0.3921568627 0.18050256880 0.067210403879 0.18697582634 0.5653560043
0.2121513944 0.17147734327
3   0.6950507421 0.4660280548 1.0000000000 0.03726028791 0.013817651840 0.02894260812 0.5642933050
0.2101593625 0.02932960894
4   0.6727827926 0.3546985559 1.0000000000 0.03235159469 0.015555368704 0.03584936688 0.5642933050
0.2091633466 0.03631284916
5   0.6746384551 0.2307761272 1.0000000000 0.04132967089 0.015751562544 0.04242723236 0.5642933050
0.2091633466 0.04329608939
6   0.5251545354 0.2439207735 1.0000000000 0.02332265120 0.011491353457 0.03157375432 0.5642933050
0.2091633466 0.03289882061
> max(houses3$long)
[1] 1
> #make a composite variable
> houses3$comp1<-(houses3$tRooms+houses3$Pop+houses3$Hholds+houses3$tBedRooms)/4
> head(houses3)
  MedianHomeVal2       medInc       medAge        tRooms            Pop        Hholds          lat
long     tBedRooms
1   0.9022663824 0.5396684184 0.7843137255 0.02233073910 0.008940833544 0.02055582963 0.5674814028
0.2111553785 0.01986343886
2   0.7082465639 0.5380270617 0.3921568627 0.18050256880 0.067210403879 0.18697582634 0.5653560043
0.2121513944 0.17147734327
3   0.6950507421 0.4660280548 1.0000000000 0.03726028791 0.013817651840 0.02894260812 0.5642933050
0.2101593625 0.02932960894
```

```
4   0.6727827926 0.3546985559 1.0000000000 0.03235159469 0.015555368704 0.03584936688 0.5642933050
0.2091633466 0.03631284916
5   0.6746384551 0.2307761272 1.0000000000 0.04132967089 0.015751562544 0.04242723236 0.5642933050
0.2091633466 0.04329608939
6   0.5251545354 0.2439207735 1.0000000000 0.02332265120 0.011491353457 0.03157375432 0.5642933050
0.2091633466 0.03289882061
          comp1
1 0.01792271028
2 0.15154153557
3 0.02733753920
4 0.03001729486
5 0.03570113880
6 0.02482164490
> max(houses3$comp1)
[1] 0.9075818929
> min(houses3$comp1)
[1] 0.0001009282068
> houses3$comp2<-(houses3$lat-houses3$long)/2
> max(houses3$comp2)
[1] 0.4916651143
> min(houses3$comp2)
[1] -0.4742327184
> #add .5 so min and max go back to 0 to 1
> houses3$comp2<-houses3$comp2+.5
> formula<-MedianHomeVal2~comp1+comp2+medInc+medAge
> trainingData<-houses3[train,]
> testingData<-houses3[test,]
> dim(trainingData)
[1] 16000    11
> dim(testingData)
[1] 4640   11
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=5, threshold=0.1)
> proc.time() - ptm
   user  system elapsed
 399.67   23.70  423.45
> formula
MedianHomeVal2 ~ comp1 + comp2 + medInc + medAge
> vars<-c("comp1","comp2","medInc","medAge")
> test2<-testingData[vars]
> results<-compute(nnet,test2)
> testingData$guess<-results$net.result
> min3<-min(houses$MedianHomeVal)
> max3<-max(houses$MedianHomeVal)
> testingData$Actual<-(testingData$MedianHomeVal2*(max3-min3))+min3
> head(testingData)
   MedianHomeVal2        medInc       medAge        tRooms           Pop        Hholds          lat
long     tBedRooms
8    0.4668042606 0.18069405939 1.0000000000 0.07889516252 0.03234395583 0.10623252754 0.5632306057
0.2091633466 0.10645561763
10   0.5074226498 0.22008661949 1.0000000000 0.09021313393 0.04338686622 0.11725045223 0.5632306057
0.2091633466 0.10955927995
19   0.2962894998 0.10283996083 0.9607843137 0.05689506079 0.02766333137 0.06873869429 0.5632306057
0.2081673307 0.07045313470
21   0.2731968116 0.05916470118 0.7647058824 0.01904979907 0.01137924269 0.02713369512 0.5642933050
0.2071713147 0.02839851024
24   0.1746405169 0.11590874609 1.0000000000 0.04288112315 0.02382353766 0.05328071041 0.5632306057
0.2071713147 0.05214152700
33   0.1967022816 0.08944704211 0.9215686275 0.04883259576 0.02867232826 0.05492517678 0.5632306057
0.2071713147 0.06331471136
           comp1        comp2        guess       absDiff Actual
8  0.08098181588 0.6770336296 0.5086637652 0.04185950463 241400
10 0.09010243308 0.6770336296 0.6182620975 0.11083944768 261100
19 0.05593755529 0.6775316375 0.2656692483 0.03062025157 158700
21 0.02149031178 0.6785609951 0.1293202291 0.14387658241 147500
24 0.04303172455 0.6780296455 0.2925568040 0.11791628710  99700
33 0.04893620304 0.6780296455 0.2177137461 0.02101146446 110400
> testingData$guess<-(testingData$guess*(max3-min3))+min3
> testingData$absDiff<-abs(testingData$Actual-testingData$guess)
> mean(testingData$absDiff)
[1] 55019.87943
> mean(testingData$absDiff)/mean(testingData$Actual)
[1] 0.2696501748
> #worse than PCA
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=8, threshold=0.05)
> proc.time() - ptm
   user  system elapsed
 729.37   45.05  774.56
> vars<-c("comp1","comp2","medInc","medAge")
> test2<-testingData[vars]
> results<-compute(nnet,test2)
> testingData$guess<-results$net.result
> testingData$guess<-(testingData$guess*(max3-min3))+min3
> testingData$absDiff<-abs(testingData$Actual-testingData$guess)
> testingData$percErr<-testingData$absDiff/testingData$Actual
> mean(testingData$absDiff)/mean(testingData$Actual)
[1] 0.2687390437
> #let’s see if we improve if we undo the lat and longitude composite
> formula<-MedianHomeVal2~comp1+medInc+medAge+lat+long
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=8, threshold=0.05)
Warning message:
algorithm did not converge in 1 of 1 repetition(s) within the stepmax
> proc.time() - ptm
   user  system elapsed
2394.73  143.58 2541.89
> #took forever and didn’t work
> #try again with more layers and higher threshold
> ptm <- proc.time()
> nnet<-neuralnet(formula,trainingData, hidden=10, threshold=0.09)
Warning message:
algorithm did not converge in 1 of 1 repetition(s) within the stepmax
> proc.time() - ptm
   user  system elapsed
2835.84  152.79 2992.83
> #still doesn’t work.  Give up
```