TDDE01 – Machine Learning
Group 9 Laboration Report 4

Martin Estgren <mares480>
Erik S. V. Jansson <erija578>
Sebastian Maghsoudi <sebma654>

Linköping University (LiU), Sweden

December 3, 2016

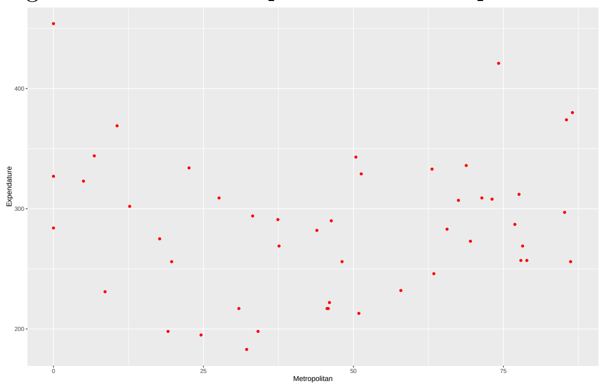
Assignment 1

This assignment involves examining a given data set called *State*, which contains the observations about the *population* & *economy* in different *states*. We are primarily interested in the relationship between the *metropolitan habitation rate (MET)* and the *public expenditure per capita (EX)* for a state.

Raw Data Analysis

We first analyze the data by plotting the *EX* target as a function of *MET*. These results can be observed in the plotted Figure 1, notice the spread of data.

Figure 1: Plot of Metropolitan Rate & Expenditure

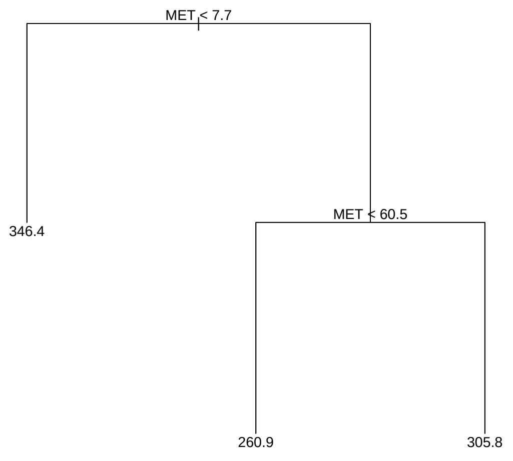


The figure indicates that a linear model isn't suitable for predicting the target in this data set, no visible pattern can be observed.

Regression Tree Analysis

We have examined how a *regression tree model* fits the data set using *cross-validation* for finding the optimal number of *leaves*. The plotted *tree* from the fitted model can be seen in Figure 2.

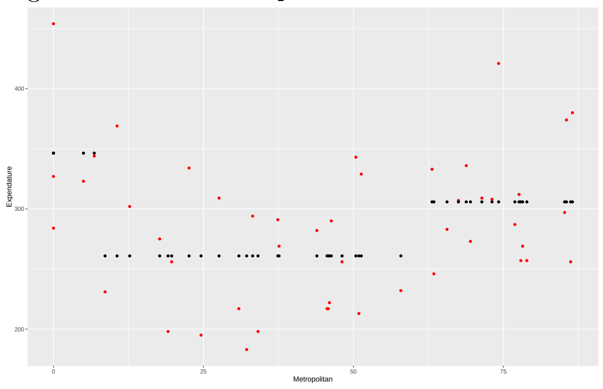
Figure 2: Plotted Decision Tree for Model



According to the cross-validation, the optimal number of leaves is 3, which we use to prune the full tree model and get the best optimal model. The predicted results can be seen in Figure 3.

As can be seen, the predictions have less labels

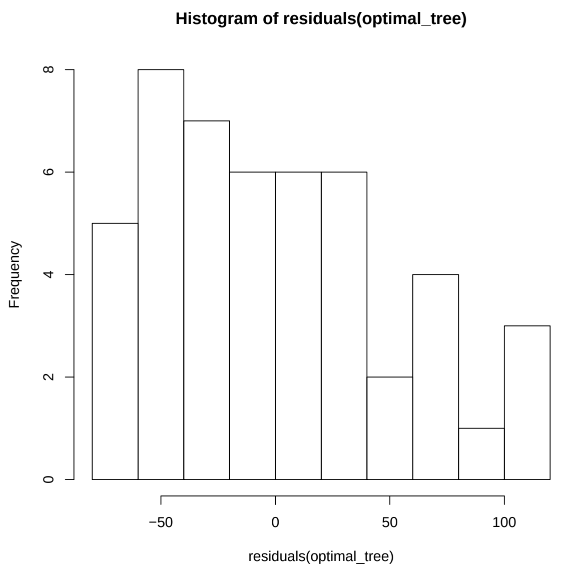
Figure 3: Plot of Metropolitan Rate & Predict Ex



compared to the original data.

The frequency of the models residuals are displayed in the histogram 4.

Figure 4: Histogram of the Residuals

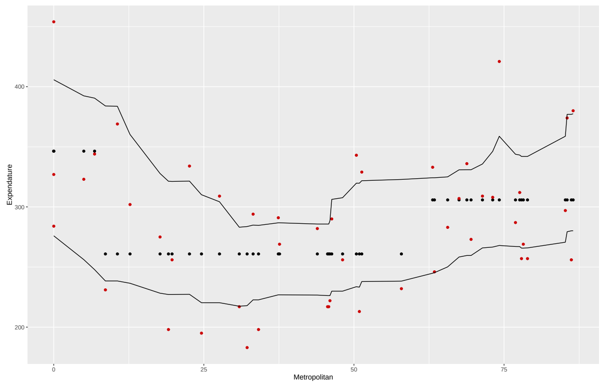


Non-Parametric Bootstrap

We examine the same data set with a *regression tree model* with *non-parametric bootstrap*. The model will follow the same structure as above but use *bootstrapping* instead of *cross-validation*.

The *non-parametric bootstrap* re-samples the given data set 1000 times and estimate the model based on the *regression tree model*. We examine the confidence band of the predicted model with a confidence level of 0.95 (fig 5).

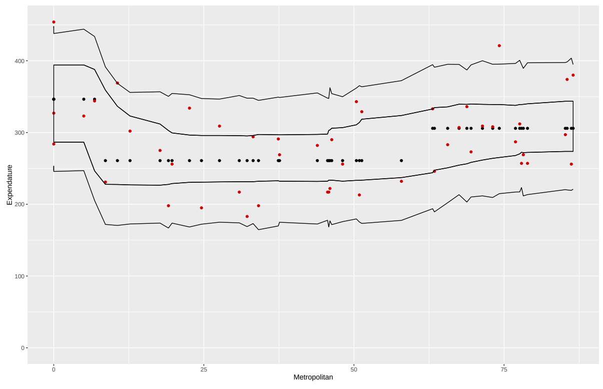
Figure 5: Confidence Bands of the Regression Tree Model



Parametric Bootstrap

The preconditions as opposed to parametric bootstrap is that we know the underlying given distribution. Assuming the expenditure label as a mean given a metropolitan rate it is possible to generate additional samples to be used in the bootstrapping process. This would also require a standard deviation, which in turn can be derived from the residual data. Plotting the confidence and prediction bands give Figure 6. In contrast to the confidence bands, prediction bands concerns the entire distribution of the data.

Figure 6: Confidence/Prediction Bands Regression Tree Model



As can be seen in the figure, there are a pair of observations outside the prediction band. This is of course the result of using 95% confidence.

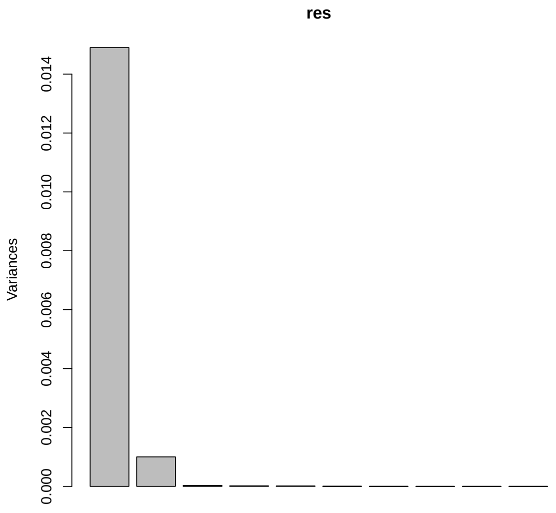
Assignment 2

In this assignment we are tasked with analyzing a data set containing observations regarding the *viscosity levels* in relation with several *near-infrared spectra* using *PCA* and *ICA* (*Component Analysis Functions*).

Principle Component Analysis

Principal Component Analysis (PCA) is used to reduce the number of dimensions in a data set by analyzing the variance that each feature contributes to the distribution. We conduct a PCA on the given data set, this gives us the histogram in Figure 7.

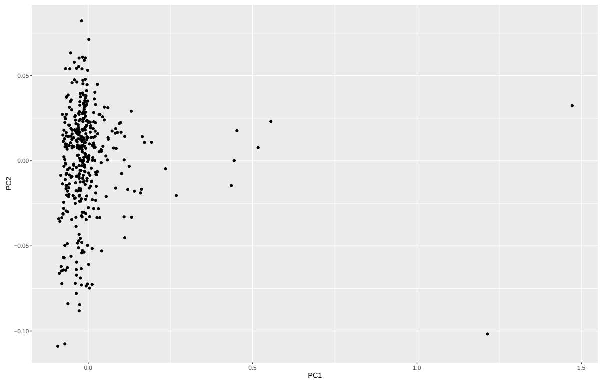
Figure 7: Confidence/Prediction Bands Regression Tree Model



We observe that only two components are required to reach 99 % cumulative variance, which are *X750 (PC1)* and *X752 (PC2)*. Afterwards, we plot the scores of the chosen principal components in Figure **??**.

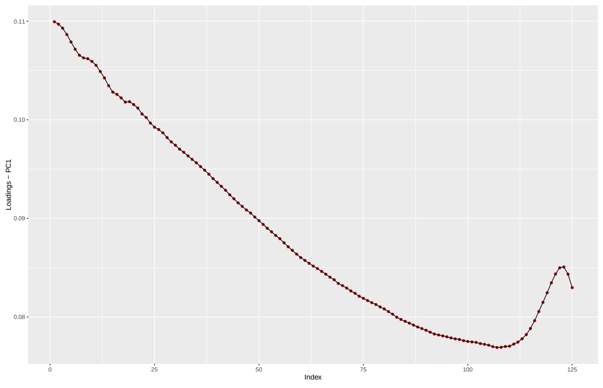
As can be seen, there is a large amount of independence between these two components. Now

Figure 8: Confidence/Prediction Bands Regression Tree Model



we plot the loadings, which indicates the correlation between components through proportional variance. This is done in Figures 9, 10. The outliers indicates unusual diesel fuels.

Figure 9: Confidence/Prediction Bands Regression Tree Model



Note the high correlation values in the same index on both plots.

Independent Component Analysis

We perform the same analysis again, this time using the *Independent component analysis* (ICA). In contrast to to PCA, where we assumed the features are correlated, we assume that they are independent.

The loadings are calculated with the function $\hat{W} = K\dot{W}$. We plot the traces for each column (the chosen components), the result can observed in Figures **??**, **??**.

Figure 10: Confidence/Prediction Bands Regression Tree Model

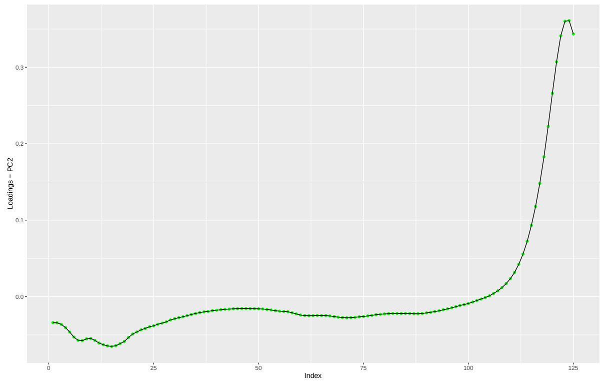


Figure 11: Confidence/Prediction Bands Regression Tree Model

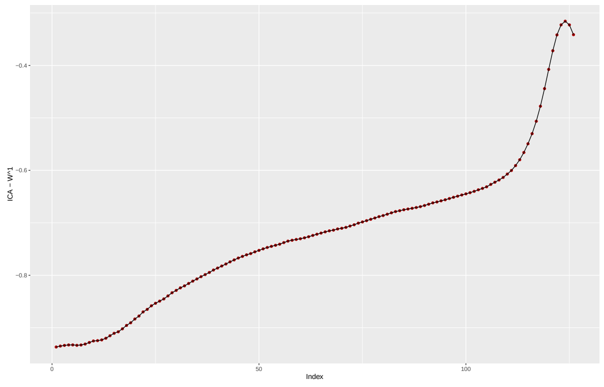
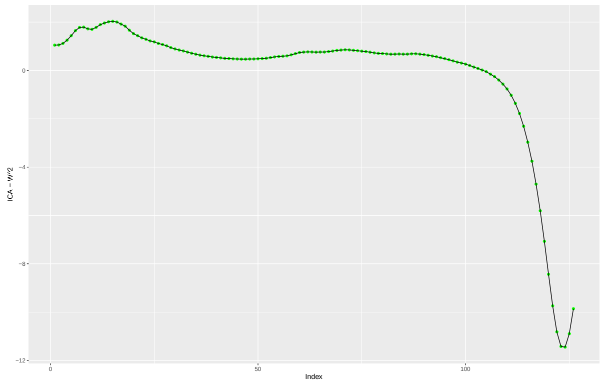


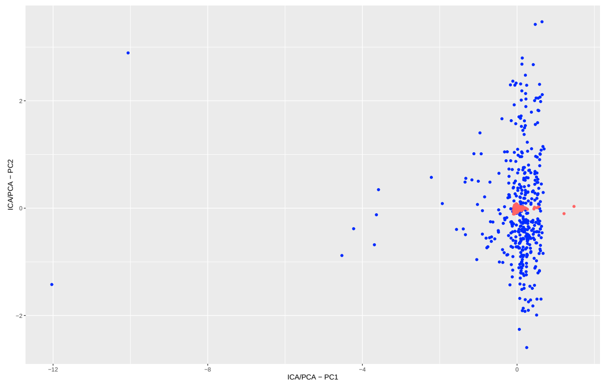
Figure 12: Confidence/Prediction Bands Regression Tree Model



The results are quite similar to those found in PCA, but inverted.

We now plot the scores found by doing ICA, which can be seen in Figure 14.

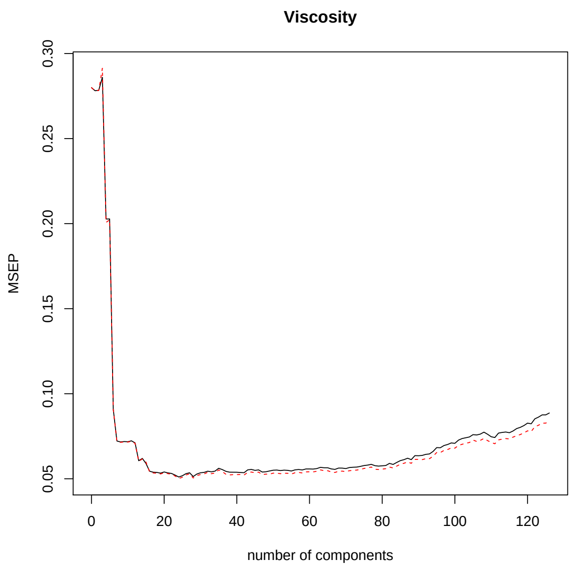
Figure 13: Confidence/Prediction Bands Regression Tree Model



PCA Cross-Validation

- Show the cross validation function
- Show the result?
- Questions

Figure 14: Confidence/Prediction Bands Regression Tree Model



Contributions

References

[FHT09] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*. Springer series in statistics, Berlin, second (11th) edition, 2009.

Appendix

Listing 1: Script for Assignment 1 on Bootstrapping

```r
library(tree)
library(boot)
library(ggplot2)
library(reshape2)
set.seed(12345)
data = read.csv2("State.csv", header = TRUE)
data = data[order(data$MET),]

controll = tree.control(nrow(data), minsize = 8)
fit = tree( EX ~ MET, data, control = controll)

fit.cv = cv.tree(fit)
best_k = fit.cv$size[which.min(fit.cv$dev)]
optimal_tree = prune.tree(fit, best=best_k)

#plot(optimal_tree)
#text(optimal_tree)

predictions = predict(optimal_tree, newdata=data)

fig_data = data.frame(x = data$MET, pred = predictions, orig = data$EX)
fig = ggplot(fig_data, aes(x, pred, orig) , xlab = "Metropolitan" , ylab = "Expendature")
fig = fig + geom_point(aes(x,orig), colour = "#FF1111") + geom_point(aes(x, pred))
print(fig)

hist(residuals(optimal_tree))

 nonparama = function(data,index){
      sample = data[index,]
      controll = tree.control(nrow(sample), minsize = 8)
      fit = tree( EX ~ MET, data=sample, control = controll)
      optimal_tree = prune.tree(fit, best=best_k)
      return(predict(optimal_tree, newdata=data))
 }

#
 set.seed(12345)
 nonparam_boot = boot(data, statistic = nonparama, R=1000)
 confidence_bound = envelope(nonparam_boot,level=0.95)
 predictions = predict(optimal_tree,data)


plot(nonparam_boot)

fig_data = data.frame(orig = data$EX, x=data$MET, pred=predictions, upper=confidence_bound$
    point[1,], lower=confidence_bound$point[2,])
fig = ggplot(fig_data, aes(x,predictions,upper,lower), xlab = "Metropolitan" , ylab = "
    Predicted Expendature")
fig = fig +
    geom_point(aes(x, pred)) +
    geom_point(aes(x, orig),colour="#CC1111") +
    geom_line(aes(x,upper)) +
    geom_line(aes(x,lower)) +
    geom_ribbon(aes(x = x, ymin=lower, ymax=upper), alpha=0.05)
print(fig)
# lines(data$MET,confidence_bound$point[1,], col="Red")
# lines(data$MET,confidence_bound$point[2,], col="Red")
```

```
 parama_conf = function(data){
   controll = tree.control(nrow(data), minsize = 8)
   fit = tree( EX ~ MET, data=data, control = controll)
   optimal_tree = prune.tree(fit, best=best_k)
   return(predict(optimal_tree, newdata=data))
 }

 parama_predic = function(data){
   controll = tree.control(nrow(data), minsize = 8)
   fit = tree( EX ~ MET, data=data, control = controll)
   optimal_tree = prune.tree(fit, best=best_k)
   predictions = predict(optimal_tree, newdata=data)
   return(rnorm(nrow(data),predictions,sd(resid(fit))))
 }

 random_predictions = function(data, model){
  sample = data.frame(MET=data$MET, EX=data$EX)
  sample$EX = rnorm(nrow(data), predict(model,newdata=data),sd(resid(model)))
  return(sample)
}

  set.seed(12345)
  param_boot_conf = boot(data, statistic = parama_conf, R=1000, mle = optimal_tree, ran.gen =
    random_predictions, sim = "parametric")
  confidence_bound_param = envelope(param_boot_conf, level=0.95)

plot(param_boot_conf)
 #
set.seed(12345)
param_boot_pred = boot(data, statistic = parama_predic, R=1000, mle = optimal_tree, ran.gen =
    random_predictions, sim = "parametric")
 prediction_bound_param = envelope(param_boot_pred, level=0.95)

plot(param_boot_pred)

predictions = predict(optimal_tree,data)
fig_data = data.frame(orig = data$EX, x=data$MET, pred=predictions, upper_c=confidence_bound_
    param$point[1,], lower_c=confidence_bound_param$point[2,], upper_p=prediction_bound_param
    $point[1,], lower_p=prediction_bound_param$point[2,])

 #
  fig = ggplot(fig_data, aes(orig,x,pred,upper_c,lower_c, upper_p, lower_p), xlab = "
    Metropolitan" , ylab = "Predicted Expendature")
  fig = fig +
    geom_point(aes(x, pred)) +
    geom_point(aes(x, orig),colour="#CC1111") +
    geom_line(aes(x,upper_c)) +
    geom_line(aes(x,lower_c)) +
    geom_ribbon(aes(x = x, ymin=lower_c, ymax=upper_c), alpha=0.05, colour = "#110011")+
    geom_line(aes(x,upper_p)) +
    geom_line(aes(x,lower_p))+
    geom_ribbon(aes(x = x, ymin=lower_p, ymax=upper_p), alpha=0.05)

  print(fig)
```

Listing 2: Script for Assignment 2 on Component Analysis

```
library("pls")
library("ggplot2")
library("fastICA")
library("reshape2")

setEPS() # Enables saving EPS format.
spectra <- read.csv2("NIRSpectra.csv")
xspectra <- spectra[,-ncol(spectra)]
yspectra <- spectra[,ncol(spectra)]
principal_comp <- prcomp(xspectra)
lambda <- principal_comp$sdev^2

# Notice both X750, X752.
cairo_ps("screeplot.eps")
screeplot(principal_comp,
          ncol(xspectra))
dev.off()
cairo_ps("biplot.eps")
biplot(principal_comp)
dev.off()

cairo_ps("score.eps")
print(qplot(principal_comp$x[,1],
            principal_comp$x[,2],
            xlab = "X750",
            ylab = "X752"))
dev.off()

x750loadings <- principal_comp$rotation[,1]
x752loadings <- principal_comp$rotation[,2]

cairo_ps("x750loadings.eps")
print(qplot(1:length(x750loadings),
            x750loadings, xlab="i",
            ylab="X750 Loadings"))
dev.off()

cairo_ps("x752loadings.eps")
print(qplot(1:length(x752loadings),
            x752loadings, xlab="i",
            ylab="X752 Loadings"))
dev.off()

set.seed(12345) # But WHY?!?!?!??!?!?!?!
independent_comp <- fastICA(xspectra, 2)

W <- independent_comp$K %*% independent_comp$W
x750whitening <- W[,1] # Un-mixed and whitened
x752whitening <- W[,2] # Un-mixed and whitened

cairo_ps("x750traceplot.eps")
print(qplot(1:length(x750whitening),
            x750whitening, xlab="i",
            ylab="X750 Inverse Loadings"))
dev.off()

cairo_ps("x752traceplot.eps")
print(qplot(1:length(x752whitening),
            x752whitening, xlab="i",
            ylab="X752 Inverse Loadings"))
dev.off()

cairo_ps("icascore.eps")
print(qplot(independent_comp$S[,1],
            independent_comp$S[,2],
            xlab = "X750",
            ylab = "X752"))
dev.off()

set.seed(12345)
principal_compcv <- pcr(Viscosity ~ ., data = spectra,
                        validation = "CV")
# cairo_ps("pcacv.eps")
validationplot(principal_compcv,
               val.type = "MS")
# dev.off()
```