Latent Structures: An Exploration of Partial Least Squares and the Hidden Variables that Affect Reindeer Herd Size

Anna Leoni
March 22, 2024

Table of Contents

SUMMARY

This study explored the relationships between several human and human-related variables and Sami reindeer herd size in Finnmark, Norway. It employed Partial Least Squares Regression in RStudio to build a predictive model, of which the second rendition was more successful but still not reliable. By examining variables and their relationship importance with herd size, more research into how latent structures containing education and technology behaviors might play a role in determining herd size is suggested.

INTRODUCTION

Reindeer have been an integral part of Sami way of life for centuries [6]. With changing climate, technology use, and human behaviors changing around the world and locally, it is important to examine if there are any relationships that may impact the size of reindeer herds, as this may affect Sami cultural traditions. Although it is common to explore habitat, diet, and climate when studying animal populations, the aim of this report is to explore the human-related effects to determine if a model can be built that can predict reindeer herd size using Partial Least Squares (Projection to Latent Structures or PLS).

DATA SET

To explore this research question, data about education levels, construction in the area, technology use, price of reindeer meat, climate and predators in the area were considered. Finding data that described all of the variables for overlapping years proved to be a limiting factor, and as a result 15 samples were collected for the timeframe of 2000–2014. This made PLS an attractive method due to its ability to handle more variables than samples. In addition, there were a variety of units for the samples: degrees in Celsius, price in Krona, percent of population, number of people, number of permits, etc. On top of this, there is likely multicollinearity across several of the variables. This was another reason to use PLS as it can handle multicollinearity and different units. The dataset was analyzed using a Partial Least Squares Regression in RStudio 4.3.2

To start, the data was sorted into predictor variables: Sami population [9] , higher education [12], basic schooling [12], permit to build [15], under construction [15], radio [11], computer [11], internet [11], greenhouse gases [14], and agricultural land in Finnmark in decares [13], surface temperature maximum [1], minimum [1], and precipitation [1], lynx [6], wolverine [6], average price of reindeer meat [8], and carcass weight [4]; with the outcome variable being the total number of reindeer (which was counted on March 31 given that April 1 is the last day to report) [10]. Before running a Partial Least Squares (PLS) Regression, it is common practice to mean-center and standardize the predictor variables, especially if they are of different units as this prevents outsized consideration or weight in certain variables on the model [16]. If the model is used for prediction, which it was in this case, the data is divided into a training set and a test set.

METHOD

After the predictor variables were mean-centered and standardized into a matrix [X], PLS computes the covariance between [X] and the outcome variable matrix [Y]. The resulting covariance matrix represents the relationship between each of the predictor variables and also includes their relationship to the outcome variable. This is notable in that it accomplishes the three goals of PLS: explain X-space, explain Y-space, and find the greatest relationship between X and Y space [2].

The algorithm then performs eigenvalue decomposition to determine the directional vectors and scalar values [3] which are then formed into linear combinations of the predictor variables that maximize covariance between X and Y [2]. These latent factors are called components and they explain the maximum variance in X and Y [2].

After each component is determined, the algorithm deflates the covariance matrix of predictor variables to account for the variance in the previously extracted component so that it does not impact the detection of new patterns. This process was complete once it was determined that all variance is accounted for in the components [2].

In RStudio, the process of running a PLS Regression consists of running the plsr code on the training set, determining the number of components to use, and testing predictions against the test set. Determining the number of components is somewhat subjective and a common technique is to find the number of components at the global minimum on the RMSE Validation plot and subtract components as long as model performance is not compromised. Another method is the "one-sigma heuristic" that involves "choosing the model with fewest components that is still less than one standard error away from the overall best model" [5]. Once the number of components was determined, the model can generate predicted values to test against the test values.

RESULTS

Running this model the first time reduced 17 variables into 9 and returned the RMSE Cross-Validations and % Variance in a summary table. The calculated number of components using the ncomp.onesigma is 2 components (see Figure 1). Strictly visually, 3 components made the most sense as RMSE was reduced without any further benefit of adding another component, and % variance explained was at 77.93% for predictors variables and 99.21% for the outcome variable. The cross-validated predictions for data with 3 components (Figure 2) are a great way to determine

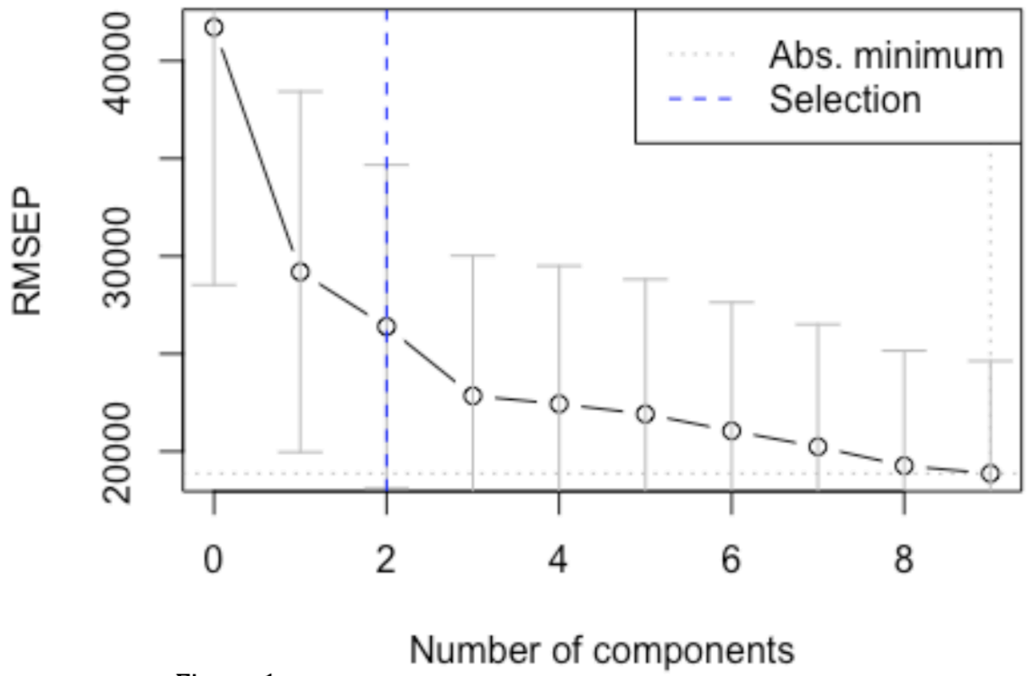


Figure 1.
RMSEP plot from PLS model 1 indicating the "one-sigma heuristic" number of components

how well the model fits the data. In this case, the line does not fit the data well, there are points quite far away from the line, one far enough perhaps to be considered outlier. In more data

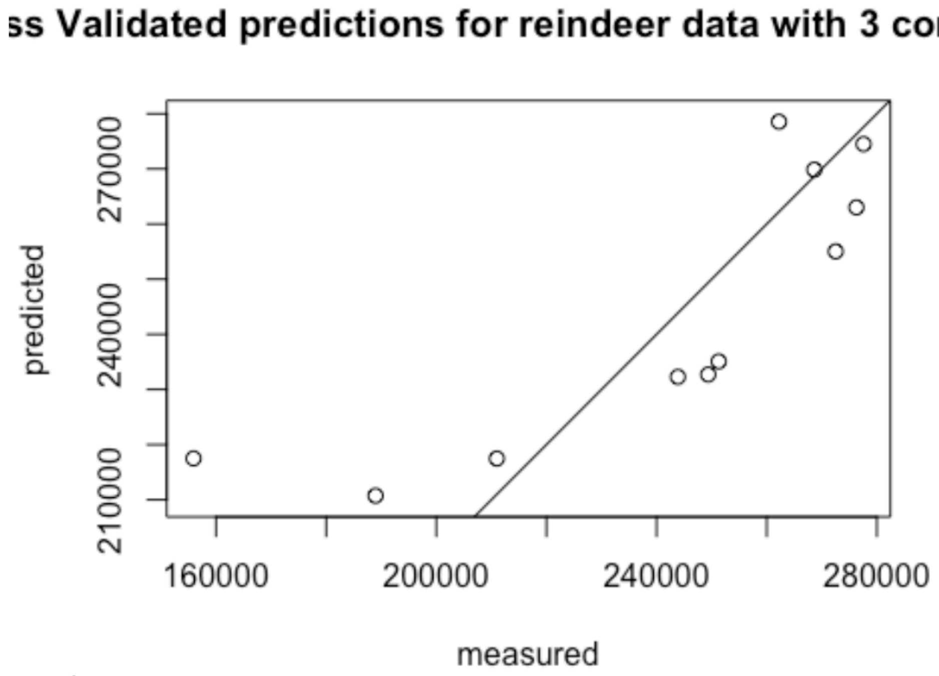


Figure 2.
Cross Validated prediction for reindeer data with 3 components for model 1. This plot is a good way to determine how well the model fits the data. Given that the line doesn't describe the points well, it can be seen that this model may not be very accurate.

visualizations like then loading plot and a correlation loading plot it becomes clear that there are several variables with loadings close to zero, meaning that they don't contribute much to the model. The prediction with 3 components returns: 236388.4, 244008.4, 240146.4, 236093.9 compared to the actual: 274969, 256095, 234044, 211227. These results return a Mean Absolute Error of: 20409.1208267018 and an R-squared of: -0.0050982776057007. Since R-squared values range from zero to one, this is further proof that this model is not well fitted.

This was an unsatisfying result, so a second model was initiated. Since the data is limited to the 15 year time frame that is available, this model looked at the VIPs (Variable Importance in Projection) from the first model (Figure 3).

Since the dimensions were reduced to 9 in the first model, the top 9 most important variables were considered in the second model. These were: percent of population completed

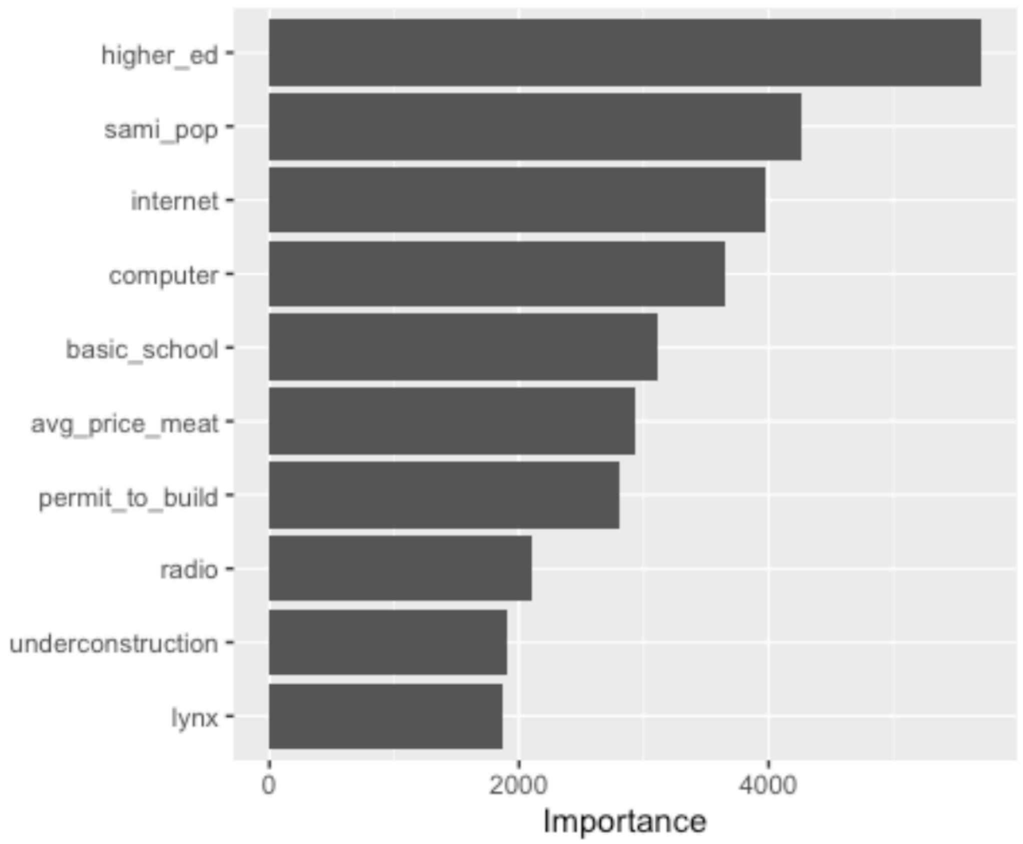


Figure 3.
Variable Importance in Projection from model 1 plotted as a bar plot.

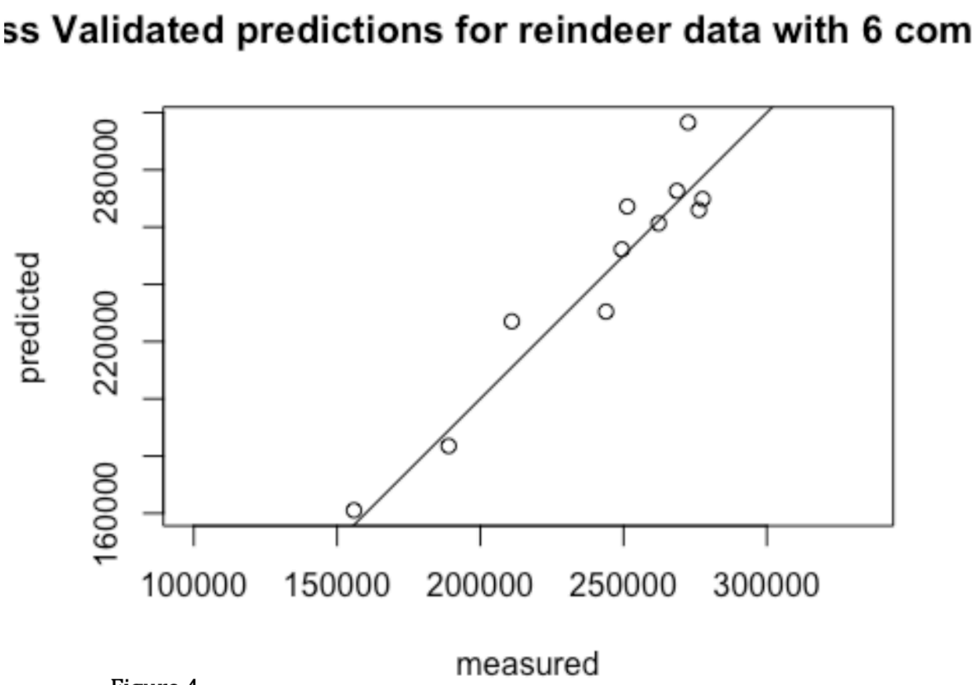
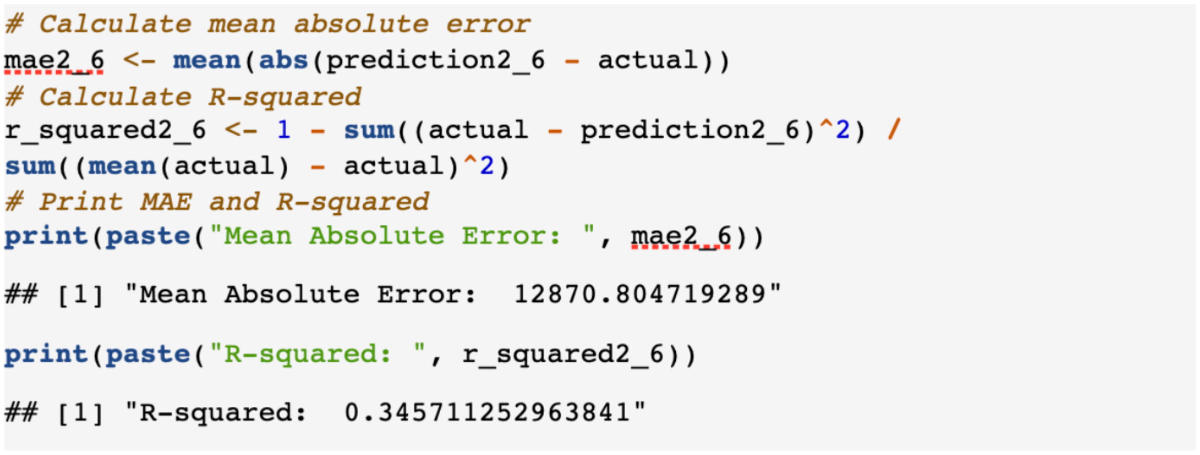


Figure 4.
Cross Validated Predictions for reindeer data in model 2. This shows a better fit than in Figure 2.

higher education, Sami population, percent of homes with internet, percent of people who own personal computers, percent of population that completed basic school as their highest education, the average price of meat, number of permits to build in Finnmark, percent of people who use radios, and the number of buildings under construction. The process was repeated as above, but yielded a much better fitting model (see Figure 4). The resulting prediction was: 282177.7, 258534.9, 271662.1, 207010.4 compared to the actual: 274969, 256095, 234044, 211227. These results return a Mean Absolute Error of: 12870.804719289 and an R-squared of: 0.345711252963841 (see Figure 5).



```
# Calculate mean absolute error
mae2_6 <- mean(abs(prediction2_6 - actual))
# Calculate R-squared
r_squared2_6 <- 1 - sum((actual - prediction2_6)^2) /
sum((mean(actual) - actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae2_6))

## [1] "Mean Absolute Error:  12870.804719289"

print(paste("R-squared: ", r_squared2_6))

## [1] "R-squared:  0.345711252963841"
```

Figure 5.
Code and results for predictive model in PLS model 2 using 6 components.

DISCUSSION

The results from the two PLS models do not have acceptable R-squared values. The results of the second model reduced the Mean Absolute Error by almost half and explained close to 35% of the variance compared to a negative value close to 0 in the first model. This could be due to selecting the variables that had the most importance in the model and contributed most to shaping the model. By eliminating the other variables, it is possible that the noise was reduced and left a clearer more focused model. It is interesting that the two suggested methods for determining the number of components led to worse results compared to a visual method partnered with examining the

RMSEP values and % variance explained values. In the first case, the suggested number of components(ncomp) was 1 or 2, when in fact 3 was selected and returned the best R-squared. In the second model, the suggested ncomp was 1 or 8, and 6 was selected. Partial Least Squares was an appropriate choice given that there were more variables than samples, several variables had a high chance of being correlated (education variables with each other, construction variables with each other, and technology variables like computers and internet), variables of different units, and the desire to make predictions. An alternative option might have been a Ridge Regression or Lasso Regression.

Overall this research was not a success in that it did not accomplish the goal of revealing an effective predictor model for reindeer herd size from the variables. It is, however, one of the few projects that focuses on human behaviors and trends in education and technology as they relate to herd size. Most of the reports found while researching for this project focus on the physical surroundings that impact the landscape or health of the reindeer, but it would be interesting to continue exploring some of the underlying or invisible behaviors and mindsets that can shift motivations for tending these semi-domesticated animals. This is a complex, multifaceted question and a way of life may depend on it.

CONCLUSION

This project did not successfully create an accurate predictive model for reindeer herd size based on the various human-related predictors. The second model was an improvement on the first, but still had a large mean absolute error of 12870.8, and an R-squared of 0.345711, which is not strong enough to be considered successful. It was curious that the most important variables were related to education level and technology use among the population. It raises interest in the possibility that these variables describe a certain latent variable of human behavior. This could be explored in more detail in future projects.

REFERENCE LIST

1. Climate Change Knowledge Portal. (2024). What is Climate Change? Understanding the Big Picture. Retrieved from https://climateknowledgeportal.worldbank.org/overview
2. Dunn, K. (2010-2023). Process Improvement Using Data. Retrieved from https://learnche.org/pid/contents#
3. Geladi, P., & Kowalski, B. R. (1986). Partial least-squares regression: A tutorial. Analytica Chimica Acta, 186, 1-17. Elsevier Science Publishers B.V., Amsterdam. Retrieved from https://home.mit.bme.hu/~horvath/IDA/1-s2.0-0003267086800289-main.pdf
4. Landbruksdirektoratet. (2024). Nøkkeltall for reinslakt. Retrieved from https://www.landbruksdirektoratet.no/nb/statistikk-og-utviklingstrekk/reindrift/nokkeltall-for-reinslakt
5. Mevik, B.-H., & Wehrens, R. (2023). Introduction to the pls Package. University Center for Information Technology, University of Oslo, Norway & Biometris, Wageningen University & Research, The Netherlands. Retrieved from https://cran.r-project.org/web/packages/pls/vignettes/pls-manual.pdf
6. ReinBase. (2024). Lynx and wolverine densities. Retrieved from https://www.reinbase.no/en-us/Estimation-of-losses/Lynx-and-wolverine-densities
7. Reindeer Herding. (2024). Sámi - Norway. Retrieved from https://reindeerherding.org/sami-norway
8. Sametinget. Renslakten i kronor. Retrieved from https://www.sametinget.se/statistik/ekonomi
9. Statistics Norway. (2022). 07520: Population at 1 January, births, deaths and migration. STN area - total by contents and year. Retrieved from https://www.ssb.no/en/statbank/table/07520/tableViewLayout1/
10. Statistics Norway. (2022). 07669: Number of reindeer in spring herd at 31 March , by contents, area and interval (year). Retrieved from https://www.ssb.no/en/statbank/table/07669/tableViewLayout1/

11. Statistics Norway. (2023). 04487: Percentage who has used different type of medie an average day, by contents, mass media and year. Retrieved from https://www.ssb.no/en/statbank/table/04487/tableViewLayout1/
12. Statistics Norway. (2023). 08921: Persons 16 years and above, by contents, region, sex, age, level of education and year. Retrieved from https://www.ssb.no/en/statbank/table/08921/tableViewLayout1/
13. Statistics Norway. (2024). 04496: Agricultural area (decares), by contents, region, size of agricultural area in use and year. Retrieved from https://www.ssb.no/en/statbank/table/04496/tableViewLayout1/
14. Statistics Norway. (2023). 13931: Greenhouse gases AR5, by contents, source (activity), energy product, pollutant and year. Retrieved from https://www.ssb.no/en/statbank/table/13931/tableViewLayout1/
15. Statistics Norway. (2024). 03723: Dwellings and utility floor space in dwellings, by region, month and contents Retrieved from https://www.ssb.no/en/statbank/table/03723/tableViewLayout1/
16. TIBCO. (1995-2020). Partial Least Squares (PLS) Overview - Computational Approach. Data Science Textbook. Retrieved from https://docs.tibco.com/data-science/GUID-B03FA85B-0773-425F-889A-F7433115208B.html

APPENDIX

The following is the code written in RStudio to run two PLS Regressions modeling the variables described above as they relate to reindeer herd size in Finnmark, Norway.

```
knitr::opts_chunk$set(fig.align ="center",
                      fig.height=6,
                      fig.width=12,
                      warning= FALSE,
                      message= FALSE,
                      comment=NA,
                      echo = FALSE)
#install.packages("caret")
```

```r
#install.packages("vip")
#install.packages("pls")
#install.packages("ggpubr")
library(pls)

library(vip)

library(caret)

library(ggpubr)
library(knitr)

#load data
reindeer_data<- read.csv("reindeer_dataframe.csv")
(reindeer_data)

# I found the plots easier to read when the variable names were clear:
sami_pop<- reindeer_data$sami_pop
sa_temp_min<- reindeer_data$sa_temp_min
sa_temp_max<- reindeer_data$sa_temp_max
prec <- reindeer_data$prec
lynx<-reindeer_data$lynx
wolverine <- reindeer_data$wolverine
carcass_weight<-reindeer_data$carcass_weight
avg_price_meat<-reindeer_data$avg_price
radio <- reindeer_data$radio
higher_ed <-reindeer_data$higher_ed
basic_school <-reindeer_data$basic_school
permit_to_build <- reindeer_data$permit_to_build
underconstruction <- reindeer_data$underconstruction
computer <- reindeer_data$computer
internet <- reindeer_data$internet
greenhouse_gases_emitted <- reindeer_data$greenhouse_gases_emitted
ag_area_decares_finnmark <- reindeer_data$ag_area_decares_finnmark


# Split the data into predictor variables [X] and response variable
[Y]:
total_reindeer <- (reindeer_data$Reindeer_Total)
predictors <- data.frame(sami_pop,
                         sa_temp_min,
                         sa_temp_max,
                         prec,
                         lynx,
                         wolverine,
                         carcass_weight,
                         avg_price_meat,
                         radio,
                         higher_ed,
                         basic_school,
                         permit_to_build,
                         underconstruction,
                         computer,
                         internet,
                         greenhouse_gases_emitted,
                         ag_area_decares_finnmark)

#Mean-center and scale the predictors dataframe:
predictors_scaled <- as.data.frame(scale(predictors))
reindeer<-cbind(total_reindeer,predictors_scaled)
#summary(reindeer)



################## MODEL 1 ########################


# Split the data into a training set and test set:
#training data: the rows 1-11:
training_data <- reindeer[1:11,
c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17)]
(training_data)

# testing data: the rows 12-15:
y_test <- reindeer[12:nrow(reindeer), c("total_rein")]
test <- reindeer[12:nrow(reindeer),
c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17)]

# This code runs the partial least squsare regression:
pls_model<- plsr(training_data$total_reindeer ~ ., data=
training_data,validation = "LOO")
# this is to veiw the summary, which is important in determining what
to do next:
summary(pls_model)

# This is the mathematical way to determine the number of components
to use:
ncomp.onesigma <- selectNcomp(pls_model, method = "onesigma", plot =
TRUE)

ncomp.permut <- selectNcomp(pls_model, method = "randomization", plot
= TRUE)

(ncomp.onesigma)

(ncomp.permut)
```

```r
#RMSEP over Components: this is the visual way to determine the number
of components: 3
rme<- plot(RMSEP(pls_model), legendpos = "topright")

# these plots let us examine how well our model fits the data based on
the number of
# components we selected:
plot(pls_model, ncomp = 1, asp = 1, line = TRUE, main="Cross Validated
predictions for reindeer data with 1 component")

plot(pls_model, ncomp = 2, asp = 1, line = TRUE, main="Cross Validated
predictions for reindeer data with 2 components")

plot(pls_model, ncomp = 3, asp = 1, line = TRUE, main="Cross Validated
predictions for reindeer data with 3 components")

# this is a way to visually inspect the data for any outliers or
oddities:
plot(pls_model, plottype = "scores", comps = 1:3)

explvar(pls_model)

# these plots show variable loading value on component: in simple
terms, how much the
# variable has an effect on the component, with points near zero
having little effect
# plot 1
plot(pls_model, "loadings", comps = 1:3,main="Loading plot")
legend("bottomright", legend = paste("Component", 1:3), col = 1:3, lty
= 1, cex = 0.8)
abline(h = 0)
axis(side = 1, at = 1:17, labels = colnames(pls_model$Xvar), cex.axis
= 0.7)

# plot 2
plot(pls_model, plottype = "correlation", ncomp=1:3, legendpos =
"bottomleft",main="Correlations loading plot for reindeer data")

# the vi and vip function shows the "variable importance in
projection" and tells us
# which variables had the strongest effect on the model
vip(pls_model)

vi(pls_model)

# the regression coefficient plot looks at how strongly each variable
related to the
# outcome variable. This plot shows 3 components:
plot(pls_model, plottype = "coef", ncomp=1:3, legendpos =
"bottomright",main="Regression coefficients")
```

```r
axis(side = 1, at = 1:17, labels = colnames(pls_model$Xvar), cex.axis
= 0.7)

# This is the predict feature of the model for each of the number of
the components
# suggested above.
prediction_1 = predict(pls_model, comps=1, newdata=test)
prediction_2 = predict(pls_model, comps=2, newdata=test)
prediction_3 = predict(pls_model, comps=3, newdata=test)
#this is the actual number of reindeer
actual = (reindeer$total_reindeer[12:15])
# print to console for a visual comparison
(prediction_1)

(prediction_2)

(prediction_3)

(actual)

# Outcomes for the 3 component choices:

# From the permutation method: 1 component:
print("RMSE for prediction::actual with ncomp = 1")

sqrt(mean((prediction_1 - actual)^2))

# Calculate mean absolute error
mae1 <- mean(abs(prediction_1 - actual))
# Calculate R-squared
r_squared1 <- 1 - sum((actual - prediction_1)^2) / sum((mean(actual) -
actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae1))

print(paste("R-squared: ", r_squared1))

# From the one-sigma method: 2 components:
print("RMSE for prediction::actual with ncomp = 2")

sqrt(mean((prediction_2 - actual)^2))

# Calculate mean absolute error
mae2 <- mean(abs(prediction_2 - actual))
# Calculate R-squared
r_squared2 <- 1 - sum((actual - prediction_2)^2) / sum((mean(actual) -
actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae2))
print(paste("R-squared: ", r_squared2))

# From the visual method: 3 components:
print("RMSE for prediction::actual with ncomp = 3")

sqrt(mean((prediction_3 - actual)^2))

# Calculate mean absolute error
mae3 <- mean(abs(prediction_3 - actual))
# Calculate R-squared
r_squared3 <- 1 - sum((actual - prediction_3)^2) / sum((mean(actual) -
actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae3))

print(paste("R-squared: ", r_squared3))






####################### MODEL 2 ########################


# Build a new data frame based on the top 9 variables:
predictors2 <- data.frame(sami_pop,
                         avg_price_meat,
                         computer,
                         higher_ed,
                         underconstruction,
                         radio,
                         basic_school,
                         permit_to_build,
                         internet)

predictors_scaled2 <- as.data.frame(scale(predictors2))
reindeer2<-cbind(total_reindeer,predictors_scaled2)
summary(reindeer2)

#split data:
#training data:
training_data2 <- reindeer2[1:11, c(1,2,3,4,5,6,7,8,9)]
(training_data2)

# testing data:
y_test2 <- reindeer2[12:nrow(reindeer2), c("total_rein")]
test2 <- reindeer2[12:nrow(reindeer2), c(1,2,3,4,5,6,7,8,9)]
```

```r
# This code runs the partial least squares regression
pls_model2<- plsr(total_reindeer ~ ., data= training_data2,validation
= "LOO")
# this is to veiw the summary, which is important in determining what
to do next:
summary(pls_model2)

# This is the mathematical way to determine the number of components
to use:
ncomp.onesigma <- selectNcomp(pls_model2, method = "onesigma", plot =
TRUE)

ncomp.permut <- selectNcomp(pls_model2, method = "randomization", plot
= TRUE)

(ncomp.onesigma)

(ncomp.permut)

#RMSEP over Components: this is the visual way to determine the number
of components: 6
rme<- plot(RMSEP(pls_model2), legendpos = "topright")

# these plots let us examine how well our model fits the data based on
the number of
# components we selected:
plot(pls_model2, ncomp = 1, asp = 1, line = TRUE, main="Cross
Validated predictions for reindeer data with 1 component")

plot(pls_model2, ncomp = 8, asp = 1, line = TRUE, main="Cross
Validated predictions for reindeer data with 8 components")

plot(pls_model2, ncomp = 6, asp = 1, line = TRUE, main="Cross
Validated predictions for reindeer data with 6 components")

# this is a way to visually inspect the data for any outliers or
oddities:
plot(pls_model2, plottype = "scores", comps = 1:6)

explvar(pls_model2)

# these plots show variable loading value on component: in simple
terms, how much the
# variable has an effect on the component, with points near zero
having little effect
# plot 1#
plot(pls_model2, "loadings", comps = 1:6,main="Loading plot")
legend("bottomright", legend = paste("Component", 1:6), col = 1:9, lty
= 1, cex = 0.8)
abline(h = 0)
```

```r
axis(side = 1, at = 1:9, labels = colnames(pls_model2$Xvar), cex.axis
= 0.7)

# plot 2
plot(pls_model2, plottype = "correlation", ncomp=1:6, legendpos =
"bottomleft",main="Correlations loading plot for reindeer data")

# the vi and vip function shows the "variable importance in
projection" and tells us
# which variables had the strongest effect on the model
vip(pls_model2)

vi(pls_model2)

# the regression coefficient plot looks at how strongly each variable
related to the
# outcome variable. This plot shows 3 components:
plot(pls_model2, plottype = "coef", ncomp=1:6, legendpos =
"topright",main="Regression coefficients")
axis(side = 1, at = 1:9, labels = colnames(pls_model2$Xvar), cex.axis
= 0.7)

# This is the predict feature of the model for each of the number of
the components
# suggested above.
prediction2_1 = predict(pls_model, ncomp = 1, newdata = test)
prediction2_8 = predict(pls_model2, ncomp = 8, newdata = test)
prediction2_6 = predict(pls_model2, ncomp = 6, newdata=test)
# actual outcome:
actual = (reindeer$total_reindeer[12:15])
# print to console for a visual comparison
(prediction2_1)

(prediction2_8)

(prediction2_6)

(actual)

# Outcomes for the 3 component choices:

# From the permutation method: 1 component:
print("RMSE for prediction::actual with ncomp = 1")

sqrt(mean((prediction2_1 - actual)^2))

# Calculate mean absolute error
mae2_1 <- mean(abs(prediction2_1 - actual))
# Calculate R-squared
r_squared2_1 <- 1 - sum((actual - prediction2_1)^2) /
```

```r
sum((mean(actual) - actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae2_1))

print(paste("R-squared: ", r_squared2_1))

# From the one-sigma method: 8 components:
print("RMSE for prediction::actual with ncomp = 8")

sqrt(mean((prediction2_8 - actual)^2))

# Calculate mean absolute error
mae2_8 <- mean(abs(prediction2_8 - actual))
# Calculate R-squared
r_squared2_8 <- 1 - sum((actual - prediction2_8)^2) /
sum((mean(actual) - actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae2_8))

print(paste("R-squared: ", r_squared2_8))

# From the visual method: 6 components:
print("RMSE for prediction::actual with ncomp = 6")

sqrt(mean((prediction2_6 - actual)^2))

# Calculate mean absolute error
mae2_6 <- mean(abs(prediction2_6 - actual))
# Calculate R-squared
r_squared2_6 <- 1 - sum((actual - prediction2_6)^2) /
sum((mean(actual) - actual)^2)
# Print MAE and R-squared
print(paste("Mean Absolute Error: ", mae2_6))

print(paste("R-squared: ", r_squared2_6))
```