STSCI-4740 PROJECT REPORT

Predicting Mapk1

By: Afolabi Ayoola: aaa347
Gunjan Sood: gs652
Salaheldin Atabani: sg2243

ABSTRACT

This project concerns the prediction of the Mapk1 gene from a mouse gene expression data set containing 24 genes and 40 samples. Throughout this project, various models are assessed, it was found that fitting a hybrid model using: natural cubic splines, polynomial and linear terms yielded the best prediction of the Mapk1 gene. The model has the following format:
Mapk1~poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+Rac1+ns(Nfat5,df=4)+Rik.
This report contains details of the statistical learning procedures used to predict the gene expression data and determine the best model that represents the Mapk1 gene. The methods compared include Best Subset Selection, shrinkage methods and Natural cubic splines. It was found that the hybrid model obtained the lowest cross validation error among the methods used.

(1) METHODS USED:

1.1 Best Subset Selection

This approach fits various models using all possible combination of the predictors and then select the best model based on the minimum AIC, BIC, Mallow's Cp or Cross Validation MSE.
Below Fig.1 shows the selected best models based on minimum values of AIC, BIC and Mallow's Cp obtained with list of significant genes. Fig 2 shows the plotted result.

Variable Selection Method	AIC			BIC			Cp		
	Min. value	No. of genes selected	Significant Genes	Min. value	No. of genes selected	Significant Genes	Min value	No. of genes selected	Significant Genes
Best Subset Selection	-198.0649	9	Cdc42, Akt2, Plcg2, Rac2, Sphk2 , Ptk2, Ppp3cb, Nfatc4 , Rac1	-19.2352041	4	Akt2 , Rik, Pik3r3, Rac1	-2.06948431	5	Akt2,Rik,Pik3r3,Pik3r1, Rac1

Fig.1

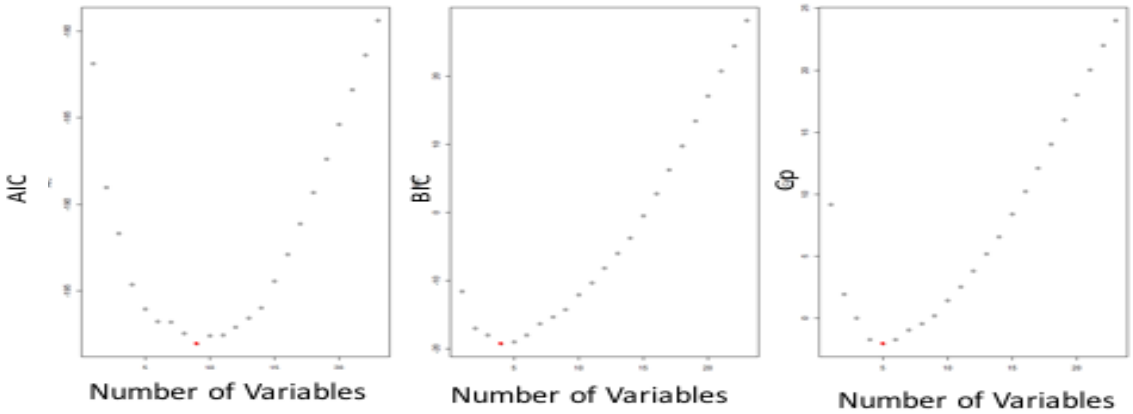


Fig.2

Best 3 models selected from the Best Subset Selection approach includes:

Model Selected using least AIC (Model1) has the form:
Mapk1~Cdc42+Akt2+Plcg2+Rac2+Sphk2+Ptk2+Ppp3cb+Nfatc4+Rac1

Model Selected using least BIC (Model2) has the form:
Mapk1~Akt2+Rik+Pik3r3+Rac1

Model Selected using least Mallow's Cp (Model3) has the form:
Mapk1~ Akt2+Rik+Pik3r3+Pik3r1+Rac1

1.2 Shrinkage Methods

The Ridge and Lasso regression models were fit in attempt to develop a model that makes the RSS as small as possible. They both penalized for the number of predictors. The Lasso model also performs variable selection.
Fig.3 and 5 Plot log(λ) against the coefficients showing how they are being shrunk towards 0 for the Ridge and Lasso Regression Respectively. Fig. 4 shows a plot of the cross validated mean square error of log(λ).

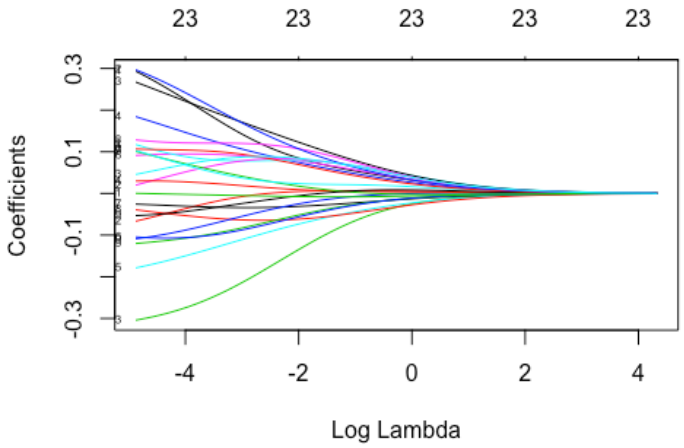


Fig.3: Coeffs against Ridge log(λ)

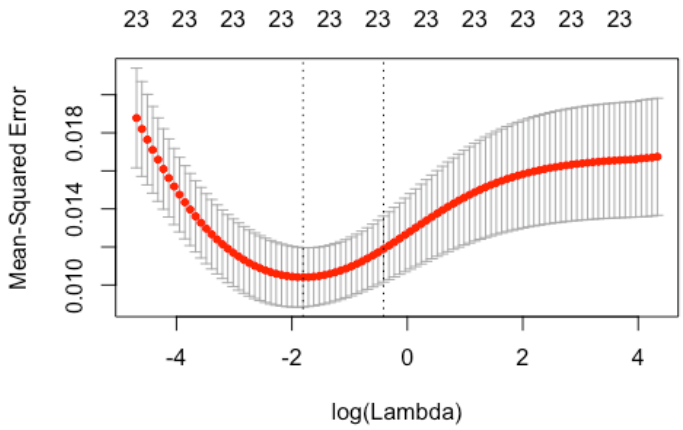


Fig.4: Ridge λ(min(MSE)) =1.009754

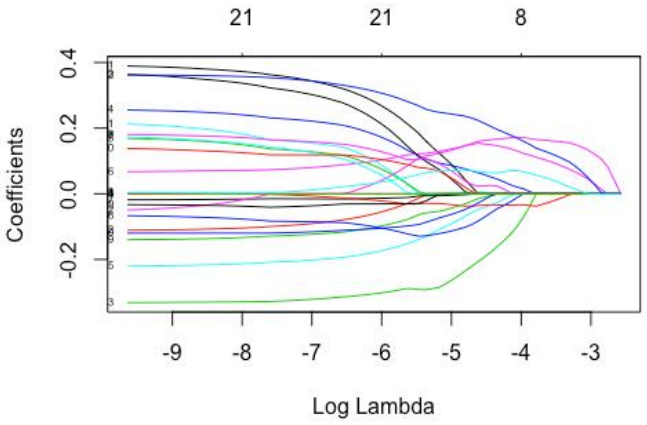


Fig.5: Coeffs against Lasso log(λ)

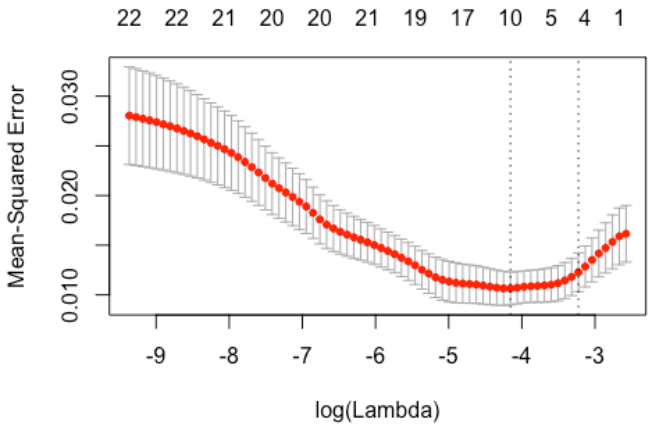


Fig.6:Lasso λ(min(MSE)) =1.011745

Model selected using Lasso has the form (Model 4):
Mapk1 ~ Rik + Pik3cd + Pik3r3 + Rac1 + Nfat5

1.3 Mixed Nonlinear Model

Through performing model diagnostics, the relationship between some of the predictors and the response gene Mapk1 is showing evident nonlinearity. This issue needed further investigation. In general, a linear model is preferred as it is simple and easy to interpret. However, if a nonlinear model shows significant improvement, or if a predictor is highly nonlinear, the nonlinear model may be preferred.
To compare different models, LOOCV error is calculated. LOOCV is universal so it can be used adequately for comparison. Also, cross-validation error is more reliable because it does not build on strict assumptions about the data.
Firstly, fit each predictor separately against Mapk1 and we use: (1) simple linear regression, (2) Natural Cubic Spline with three knots, and (3) three-degree polynomial. This is done for each predictor and the LOOCV error is computed each time.

Fig.7 below shows the LOOCV error resulting from each method. The column ALLmin shows the minimum generated error from the three methods. The column minCV shows the method that generates the lowest error among the 3 methods. The table is sorted using ALLmin from largest to smallest. The first six predictors (in red) were further excluded in the model building since they generated high LOOCV error.
It's worth mentioning that four of these predictors' coefficients have a p-value higher than 0.4 in a simple linear model. This gives more emphasis on the assumption that they are not significantly associated with the response. This step is of great computational benefit.

Based on that, 17 predictors were considered. For each predictor, the method (linear, spline, or polynomial) with the lowest CV error were used.
Best subset selection was used with minimum two predictors and maximum 10 predictors. As a result, 109,276 models were obtained. R^2 and Adjusted R^2 of these models were extracted and t he LOOCV error was computed for the top 200 models based on the Adjusted R2. Fig.7 shows t he results sorted by CV error from smallest to largest.

Predictor	Linear	Spline	Poly	minCv	ALLmin
Sphk2	0.0176	0.0257	0.0181	lm	0.0176
Pla2g6	0.0171	0.0198	0.0374	lm	0.0171
Pik3r1	0.0170	0.0201	0.0219	lm	0.0170
Pla2g5	0.0169	0.0217	0.0185	lm	0.0169
Akt2	0.0169	0.0176	0.0186	lm	0.0169
Nfatc4	0.0167	0.0198	0.0177	lm	0.0167
Map2k2	0.0166	0.0206	0.0185	lm	0.0166
Pik3ca	0.0164	0.0163	0.0172	ns	0.0163
Nras	0.0171	0.0172	0.0162	poly	0.0162
Mapkapk2	0.0158	0.0310	0.0406	lm	0.0158
Nos3	0.0170	0.0162	0.0150	poly	0.0150
Ptk2	0.0159	0.0183	0.0149	poly	0.0149
Mapk13	0.0160	0.0149	0.0163	ns	0.0149
Plcg2	0.0148	0.0159	0.0156	lm	0.0148
Rac2	0.0153	0.0144	0.0165	ns	0.0144
Cdc42	0.0154	0.0143	0.0145	ns	0.0143
Ppp3cb	0.0160	0.0137	0.0149	ns	0.0137
Map2k1	0.0128	0.0156	0.0136	lm	0.0128
Pik3cd	0.0127	0.0160	0.0133	lm	0.0127
Rac1	0.0126	0.0131	0.0128	lm	0.0126
Nfat5	0.0121	0.0119	0.0124	ns	0.0119
Rik	0.0113	0.0116	0.0119	lm	0.0113
Pik3r3	0.0107	0.0117	0.0112	lm	0.0107

Fig.7: Linear, Spline and Poly LOOCV Errors.

	Adj.r.squared	R.squared	CV.error
poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+Rac1+ns(Nfat5,df=4)+Rik	0.7703	0.8940	0.0069
poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+ns(Nfat5,df=4)+Rik	0.7581	0.8821	0.0071
poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+ns(Nfat5,df=4)	0.7497	0.8716	0.0074
Map2k2+poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+Rac1+ns(Nfat5,df=4)+Rik	0.7575	0.8943	0.0075
poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+Map2k1+ns(Nfat5,df=4)+Rik	0.7447	0.8822	0.0076
Map2k2+poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+ns(Nfat5,df=4)+Rik	0.7446	0.8821	0.0076
.....	.....	.....	.....
.....	.....	.....	.....
poly(Nras,3)+Mapkapk2+poly(Nos3,3)+poly(Ptk2,3)+ns(Mapk13,df=4)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+ns(Nfat5,df=4)+Pik3r3	0.7482	0.9483	0.0850
ns(Pik3ca,df=4)+poly(Nos3,3)+poly(Ptk2,3)+ns(Mapk13,df=4)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+Map2k1+Rac1+ns(Nfat5,df=4)	0.7677	0.9583	0.0990

Fig.8: Adj R^2 R^2 and CV Error of all possible mixed models

As seen, the best model result in 0.0069 LOOCV error. This is a great improvement compared to the linear models shown earlier. Below is a summary of the best resulting model (Model 5):
Mapk1~poly(Ptk2,3)+ns(Rac2,df=4)+ns(Cdc42,df=4)+ns(Ppp3cb,df=4)+Rac1+ns(Nfat5,df=4)+Rik

2. Selected Model

It was found that the Hybrid model outperformed the others as it has the lowest LOOCV error in comparison to the others. Examining Fig.9 shows the relative difference in the error. Fig 10 shows the adjusted R^2, which is highest for the mixed nonlinear model.

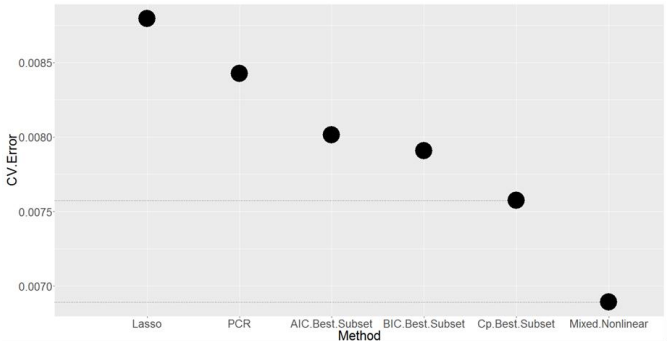


Fig.9: LOOCV Error for the different model

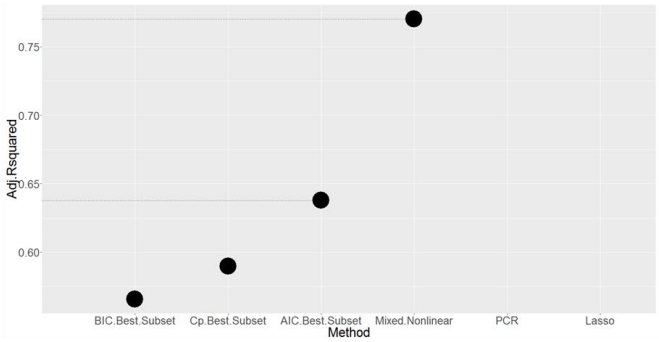
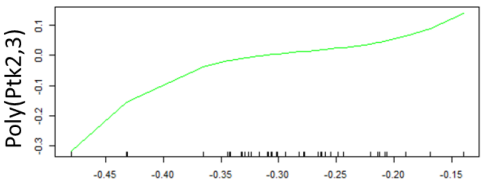


Fig.10: Adj. R^2 for the different models

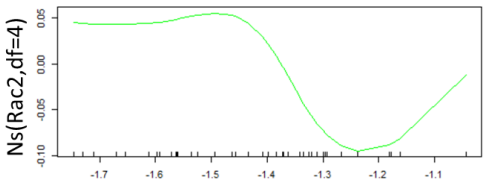
3. Model Description

Based on the LOOCV error, we find a significant improvement when we use nonlinear terms such as polynomial and splines. The nonlinear terms that we used make an improvement of 11% compared to the best possible linear model. This improvement is enough to justify moving to nonlinearity.
The below charts explain the effect of each term:



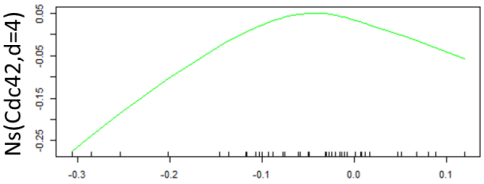
Ptk2

Monotonically increasing



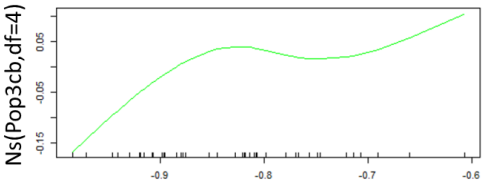
Rac2

Almost steady, decreasing after around Rac2≈-1.5 and then increasing after Rac2≈ -1.25



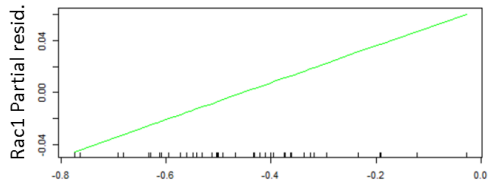
Cdc42

Increasing function until the point Cdc42≈-.05 and then increasing decreasing function



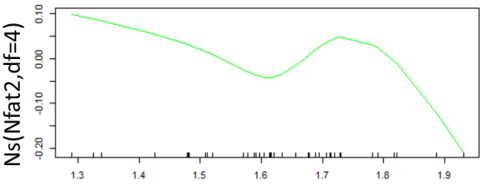
Pop3cb

Increasing function until the point Pop3cb≈-.85, then slightly decreasing and increasing again after Pop3cb≈-.73



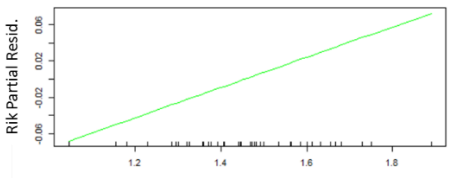
Rac1

Linear, monotonically increasing



Nfat2

Decreasing function until the point Nfat2≈1.6, then increasing and decreasing again after Nfat2≈1.73



Rik

Linear, monotonically increasing

Below is a summary of the model coefficients:

```
Anova for Parametric Effects
                  Df   Sum Sq  Mean Sq F value    Pr(>F)
poly(Ptk2, 3)      3 0.108746 0.036249  9.9975 0.0004211 ***
ns(Rac2, df = 4)   4 0.081854 0.020464  5.6439 0.0039971 **
ns(Cdc42, df = 4)  4 0.188562 0.047141 13.0016 3.775e-05 ***
ns(Ppp3cb, df = 4) 4 0.075545 0.018886  5.2089 0.0057538 **
Rac1               1 0.023799 0.023799  6.5638 0.0196022 *
ns(Nfat5, df = 4)  4 0.064182 0.016045  4.4254 0.0114988 *
Rik                1 0.007710 0.007710  2.1265 0.1619959
Residuals         18 0.065264 0.003626
---
```

4. CONCLUSION:

To fit a model that explains and predicts Mapk1, several linear and nonlinear models have been assessed. These models include but not limited to: best subset selection, Lasso, ridge regression, Principle Component Regression and a mixed model of linear, polynomial and splines terms. It was found that the data shows evident nonlinearity and so the nonlinear modeling of the predictors and the response outperformed any other type of modeling.

Appendix A: R CODE

```
if (!require("ggplot2")) install.packages("ggplot2");library(ggplot2)
if (!require("glmnet")) install.packages("glmnet");library (glmnet)
if (!require("pls")) install.packages("pls");library(pls)
if (!require("boot")) install.packages("boot");library(boot)
if (!require("leaps")) install.packages("leaps");library(leaps)
if (!require("MASS")) install.packages("MASS");library(MASS)
if (!require("pls")) install.packages("pls");library(pls)
if (!require("pls")) install.packages("pls");library(pls)
if (!require("splines")) install.packages("splines");library(splines)
if (!require("gam")) install.packages("gam");library(gam)

#read and transpose dataset. You have to put the data in the working directory
#or set working director in the folder where you have the data
kidney<-read.csv('C:/Users/Salah/Desktop/Cornell/4740 Data mining and machine learni
ng/project/Kidney_2.csv')
kidneyt = setNames(data.frame(t(kidney[,-1])), kidney[,1])
kidney<-kidneyt
Kidney<-kidney



# Variable selection
##############################################################################
##                              Part I: linear Model                        ##
##############################################################################
#########################                                  ###################
#########################    1.1 Best subset Selection    ###################
#########################                                  ###################
##############################################################################

# Using best subset selection
regfit_best = regsubsets(Mapk1~.,data=Kidney, nvmax=24)
rs_best<-summary(regfit_best)


# Obtaining number of significant predictors in best selection approach through AIC
AIC <- 40*log(rs_best$rss/40)+(2:24)*2
plot(AIC~I(1:23), ylab = "AIC", xlab = "Number of predictors")
#which.min(AIC)
points(which.min(AIC),AIC[which.min(AIC)],col="red",cex=2,pch=20)
```

```r
# Obtaining number of significant predictors in best selection approach through BIC
BIC<-rs_best$bic
plot(BIC~I(1:23),ylab = "BIC", xlab = "Number of predictors")
#which.min(BIC)
points(which.min(BIC),BIC[which.min(BIC)],col="red",cex=2,pch=20)


# Obtaining number of significant predictors in best selection approach through Mall
ow's cp
cp <- rs_best$cp
plot(cp~I(1:23),ylab = "Mallow's cp", xlab = "Number of predictors")
#which.min(cp)
points(which.min(cp),cp[which.min(cp)],col="red",cex=2,pch=20)

# # Obtaining number of significant predictors in best selection approach using LOOC
V error

predict_regsubsets = function(object,newdata,id){
  form =as.formula(object$call[[2]])
  mat = model.matrix(form,newdata)
  coefi =coef(object ,id=id)
  xvars = names(coefi)
  mat[,xvars]%*% coefi
}


k=40
set.seed(1)
#folds=sample(1:k,nrow(Kidney),replace = TRUE)
folds=1:k
cv_errors = matrix(NA,k,23,dimnames = list(NULL,paste(1:23)))


for (j in 1:k )
{ best_fit = regsubsets(Mapk1~.,data=Kidney[folds!=j,],nvmax=23)
for (i in 1:23){
  pred=predict_regsubsets(best_fit,Kidney[folds==j,],id=i)
  cv_errors[j,i]= mean((Kidney$Mapk1[folds==j]-pred)^2)
}
}


mean_cv_errors= apply(cv_errors, 2, mean)
```

```r
plot(mean_cv_errors, type ='b', ylab = "Mean CV Errors", xlab = " Number of Predicto
rs")


# Select the best model based AIC which chooses 9 predictors
coef(regfit_best,9)

AICbest.fit=glm(Mapk1~Cdc42+Akt2+Plcg2+Rac2+Sphk2+Ptk2+Ppp3cb+Nfatc4+Rac1, data = Ki
dney)
BICbest.fit=glm(Mapk1~Akt2+Rik+Pik3r3+Rac1 , data = Kidney)
Cpbest.fit=glm(Mapk1~ Akt2+Rik+Pik3r3+Pik3r1+Rac1 , data = Kidney)
AICbest.CV=cv.glm(kidneyt ,AICbest.fit)$delta[1]
BICbest.CV=cv.glm(kidneyt ,BICbest.fit)$delta[1]
Cpbest.CV=cv.glm(kidneyt ,Cpbest.fit)$delta[1]
AICbest.AdjR2=summary(lm(Mapk1~Cdc42+Akt2+Plcg2+Rac2+Sphk2+Ptk2+Ppp3cb+Nfatc4+Rac1,
data = Kidney))$adj.r.squared
BICbest.AdjR2=summary(lm(Mapk1~Akt2+Rik+Pik3r3+Rac1 , data = Kidney))$adj.r.squared
Cpbest.AdjR2=summary(lm(Mapk1~ Akt2+Rik+Pik3r3+Pik3r1+Rac1 , data = Kidney))$adj.r.s
quared

##############################################################################
##########################                                  ##########################
##########################      1.2 Ridge and Lasso         ##########################
##########################                                  ##########################
##############################################################################

#creating model matrix
x = model.matrix(Mapk1~.-1, data = Kidney)
#retrieving values of Mapk1
y = Kidney$Mapk1

#fitting ridge
set.seed(1)
fit.ridge = glmnet(x,y,alpha = 0)
#plotting log lambda against coefficients
plot(fit.ridge,xvar = "lambda", label = TRUE)
#cross validated ridge
set.seed(1)
cv.ridge = cv.glmnet(x,y,alpha = 0)
plot(cv.ridge)


#fiting lasso
set.seed(1)
fit.lasso = glmnet(x,y)
```

```r
plot(fit.lasso,xvar = "lambda",label = TRUE)
#fraction of deviance explained
plot(fit.lasso,xvar = "dev",label = TRUE)
#cross validated lasso
set.seed(1)
cv.lasso = cv.glmnet(x,y)
plot(cv.lasso)
#showing the coefficients that lasso selected
coef(cv.lasso)

#fitting final ridge and lasso models
x = model.matrix(Mapk1~., data = Kidney)
y = Kidney$Mapk1
Lasso.fit = lm(Mapk1 ~ Rik + Pik3cd + Pik3r3 + Rac1 + Nfat5, data = kidney)
Ridge.fit = glmnet(x,y,alpha = 0, lambda = 1.009754)

#obtain CV error for ridge and lasso (leave two out)
k=20
set.seed(1)
#folds=sample(1:k,nrow(Kidney),replace = TRUE)
folds=sample(rep(1:20,2))
cv_errorsR = vector()
cv_errorsL = vector()
for (j in 1:k)
{
modelR = glmnet(x[folds!=j,],y[folds!=j],alpha = 0, lambda = 1.009754)
modelL =lm(Mapk1~Rik+Pik3cd+Pik3r3+Rac1+Nfat5,data=kidney[folds!=j,])

predR=predict(modelR,s=1.011745,newx=x[folds==j,])
predL=predict(modelL,kidney[folds==j,])
cv_errorsR[j]= mean((Kidney$Mapk1[folds==j]-predR)^2)
cv_errorsL[j]= mean((Kidney$Mapk1[folds==j]-predL)^2)
}
Ridge.CV=mean(cv_errorsR)
Lasso.CV=mean(cv_errorsL)



##############################################################################
##########################                                  ##########################
##########################     1.3 Principle Component      ##########################
##########################            Regression            ##########################
##########################                                  ##########################
##############################################################################

#PCR:
set.seed(1)
```

```r
pcr.fit=pcr(Mapk1~., data=kidneyt ,scale =TRUE,validation ="LOO")
summary(pcr.fit)
validationplot(pcr.fit ,val.type="MSEP")
abline(v=4,col='blue')
pcr.CV=0.09179^2


##############################################################################
##                          Part II: Non-linear Model                       ##
##############################################################################
########################                                  ####################
########################      Non-Linear Model            ####################
########################                                  ####################
##############################################################################


#Create a data frame that has cross validation for simple models (each predictor in
a simple model with the response)
#And specify the model (linear, poly, or spline) that has the lowest CV error.
a<-vector()
b<-vector()
c<-vector()
e<-vector()
for (i in 1:24)
{
  fit1=glm(Mapk1~eval(parse(text = names(kidneyt)[i])),data=kidneyt)
  fit2<-glm(Mapk1~ns(eval(parse(text = names(kidneyt)[i])),df=4),data=kidneyt)
  fit4<-glm(Mapk1~poly(eval(parse(text = names(kidneyt)[i])),2),data=kidneyt)
  a[i]<-cv.glm(kidneyt ,fit1)$delta[1]
  b[i]<-cv.glm(kidneyt ,fit2)$delta[1]
  e[i]<-cv.glm(kidneyt ,fit4)$delta[1]
  c[i]<-names(kidneyt)[i]
}
d<-data.frame(predictor=c,lm=a,ns=b,poly=e)
d<-d[-5,]
d$minCV<-apply(d[,2:4],1,which.min)
d[d$minCV==1,'minCv']<-'lm'
d[d$minCV==2,'minCv']<-'ns'
d[d$minCV==3,'minCv']<-'poly'
d$ALLmin<-apply(d[,2:4],1,min)
d<-d[order(-d$ALLmin),]



#Mixed model:
```

```r
#best subset selection of 2 to 10 predictors out of the best 17 predictors using adj
usted r.squared
listcombo <- unlist(sapply(2:10,function(x) combn(17, x, simplify=FALSE)),recursive=
FALSE)
predterms <- lapply(listcombo, function(x) paste(c("Map2k2","ns(Pik3ca,df=4)","poly(
Nras,3)","Mapkapk2","poly(Nos3,3)","poly(Ptk2,3)","ns(Mapk13,df=4)","Plcg2","ns(Rac2
,df=4)","ns(Cdc42,df=4)","ns(Ppp3cb,df=4)","Map2k1","Pik3cd","Rac1","ns(Nfat5,df=4)"
,"Rik","Pik3r3")[x],collapse="+"))


coefm <- matrix(NA,length(listcombo),2)
system.time(
  for(i in 1:length(listcombo)){
    lmi <- lm(as.formula(paste("Mapk1 ~ ",predterms[[i]])), data= kidneyt)
    coefm[i,1] <- summary(lmi)$adj.r.squared
    coefm[i,2] <- summary(lmi)$r.squared
  })
rownames(coefm) <- predterms
colnames(coefm) <- c("adj.r.squared","r.squared")
coef<-data.frame(coefm)
coef<-coef[order(-coef$adj.r.squared),]


#save(coef, file = "coef.RData")

#Get the LOOCV for the top 200 models based on adjusted R2 and get CV error for them
CV.error<-vector()
for (i in 1:200)
{
  model<-glm(as.formula(paste("Mapk1 ~ ",rownames(coef)[i])),data=kidneyt)
  CV.error[i] <-cv.glm(kidneyt ,model)$delta[1]
}
coef$CV.error<-1
coef$CV.error[1:200]<-CV.error
coef<-coef[order(coef$CV.error),]
#Fit the model with the lowest CV error
nl.fit<-glm(as.formula(paste("Mapk1 ~ ",rownames(coef)[1])),data=kidneyt)
summary(nl.fit)

nl.CV=cv.glm(kidneyt ,nl.fit)$delta[1]
nl.AdjR2=summary(lm(as.formula(paste("Mapk1 ~ ",rownames(coef)[1])),data=kidneyt))$a
dj.r.squared

par(mfrow=c(3,3))
gam.fit=gam(as.formula(paste("Mapk1 ~ ",rownames(coef)[1])),data=kidneyt)
plot(gam.fit, se=F ,col ="green")
```

```
##############################################################################
##                              Part III: Summary                           ##
##############################################################################
########################                                ######################
########################      Summary and selected      ######################
########################             Model              ######################
##############################################################################



#Final model from part 1.1
#Final model from part 1.2
#Final model from part 1.3
#Final model from part 2

final_comparison<-data.frame(Method=c('AIC.Best.Subset','BIC.Best.Subset','Cp.Best.S
ubset','PCR','Lasso','Mixed.Nonlinear'), CV.Error=c(AICbest.CV,BICbest.CV,Cpbest.CV,
pcr.CV,Lasso.CV,nl.CV),Adj.Rsquared=c(AICbest.AdjR2,BICbest.AdjR2,Cpbest.AdjR2,NA,NA
,nl.AdjR2))



final_comparison$Method <- factor(final_comparison$Method , levels = final_compariso
n$Method [order(-final_comparison$CV.Error)])

ggplot(data=final_comparison,aes(x=Method,y=CV.Error))+geom_point(aes(x=Method,y=CV.
Error),shape=16, size=15)+geom_segment(aes(x=0,xend='Mixed.Nonlinear',y=CV.Error[6],
yend=nl.CV),linetype="dotted")+geom_segment(aes(x=0,xend='Cp.Best.Subset',y=CV.Error
[3],yend=Cpbest.CV),linetype="dotted")+ theme(text = element_text(size=25))



final_comparison$Method <- factor(final_comparison$Method , levels = final_compariso
n$Method [order(final_comparison$Adj.Rsquared)])

ggplot(data=final_comparison,aes(x=Method,y=Adj.Rsquared))+geom_point(aes(x=Method,y
=Adj.Rsquared),shape=16, size=15)+geom_segment(aes(x=0,xend='Mixed.Nonlinear',y=Adj.
Rsquared[6],yend=nl.AdjR2),linetype="dotted")+geom_segment(aes(x=0,xend='AIC.Best.Su
bset',y=0.6377131,yend=AICbest.AdjR2),linetype="dotted")+ theme(text = element_text(
size=25))
```