project_methods.R

misaelmorales

2020-04-23

```
# This dataset has varriables from 1,000 unconventional wells including well average porosity, log tra
# of permeability (to linearize the relationships with other variables), accoustic impedance (kg/m2s*1
# total organic carbon (%), vitrinite reflectance (%), and production (MCFPD)

# Load  required libraries
library(plyr); library(scales)
library(ggplot2); library(lattice); library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(tree); library(gbm); library(glmnet); library(randomForest); library(boot)
```

```
## Registered S3 method overwritten by 'tree':
##   method     from
##   print.tree cli
```

```
## Loaded gbm 2.1.5
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:lattice':
##
##     melanoma
```

```
# Read the data from .CSV
DF0 <- read.table("mv_unconv.csv", as.is = TRUE)    #import csv (str character)
rock.data <- read.csv(text = DF0[[1]])              #fix data type to numeric
rock.data <- rock.data[,-1]                         #remove first column, WellIndex

# Attach the data
attach(rock.data)

# Visualize the data
head(rock.data); tail(rock.data)
```

```
##     Por LogPerm   AI Brittle  TOC   VR Production
## 1 15.91    1.67 3.06   14.05 1.36 1.85    177.382
```

```
## 2 15.34    1.65 2.60   31.88 1.37 1.79   1479.768
## 3 20.45    2.02 3.13   63.67 1.79 2.53   4421.222
## 4 11.95    1.14 3.90   58.81 0.40 2.03   1488.318
## 5 19.53    1.83 2.57   43.75 1.40 2.11   5261.095
## 6 19.47    2.04 2.73   54.37 1.42 2.12   5497.006
```

```
##        Por LogPerm   AI Brittle   TOC   VR Production
## 995  11.95    1.14 2.97   67.18  0.80 2.06   931.6109
## 996  17.99    2.29 3.38   44.32  0.98 2.08  4211.5278
## 997  12.12    0.82 3.52   57.07 -0.04 1.73  1560.3337
## 998  15.55    1.50 2.48   58.25  1.89 2.35  2858.1805
## 999  20.89    2.02 3.23   46.17  1.71 2.27  6934.5763
## 1000 15.74    1.31 2.37   73.08  1.24 2.06   963.6758
```

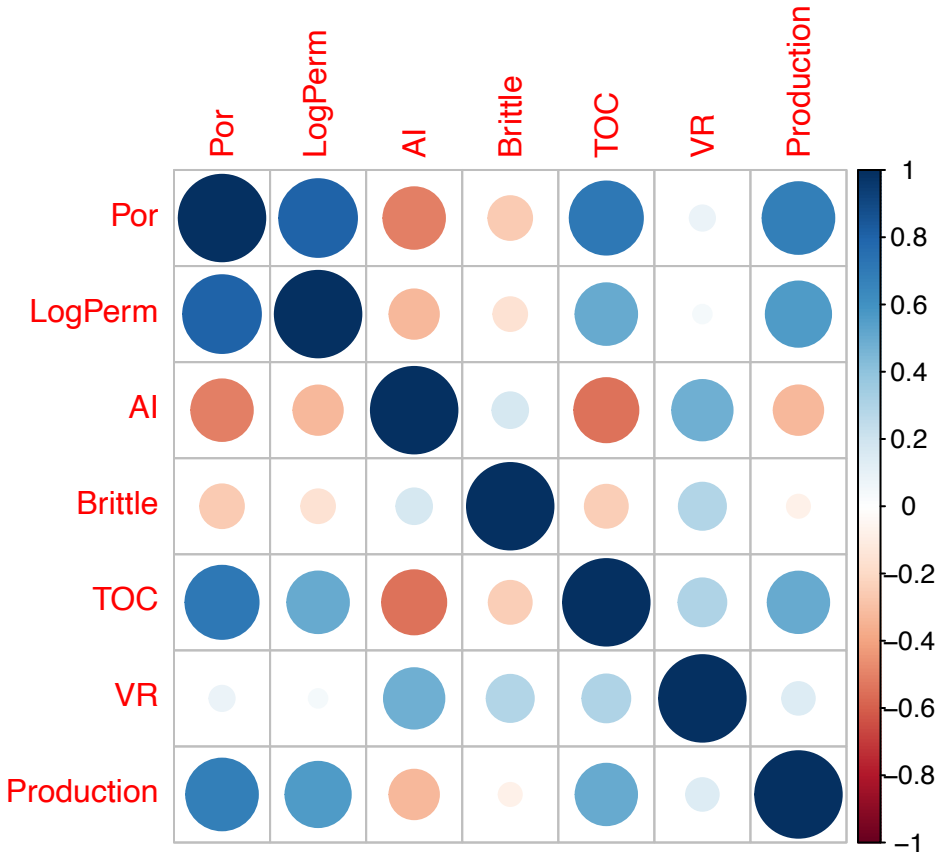
```
# Summary statistics for each column
summary(rock.data)
```

```
##       Por           LogPerm            AI           Brittle
##  Min.   : 5.40   Min.   :0.120   Min.   :0.960   Min.   :-10.50
##  1st Qu.:12.86   1st Qu.:1.130   1st Qu.:2.578   1st Qu.: 39.72
##  Median :14.98   Median :1.390   Median :3.010   Median : 49.68
##  Mean   :14.95   Mean   :1.399   Mean   :2.983   Mean   : 49.72
##  3rd Qu.:17.08   3rd Qu.:1.680   3rd Qu.:3.360   3rd Qu.: 59.17
##  Max.   :24.65   Max.   :2.580   Max.   :4.700   Max.   : 93.47
##       TOC               VR          Production
##  Min.   :-0.260   Min.   :0.900   Min.   :    2.714
##  1st Qu.: 0.640   1st Qu.:1.810   1st Qu.: 1191.370
##  Median : 0.995   Median :2.000   Median : 1976.488
##  Mean   : 1.004   Mean   :1.991   Mean   : 2247.296
##  3rd Qu.: 1.360   3rd Qu.:2.172   3rd Qu.: 3023.594
##  Max.   : 2.710   Max.   :2.900   Max.   :12568.644
```

```
# Correlation matrixand plot
cor_matrix <- round(cor(rock.data),3); print(cor_matrix)
```

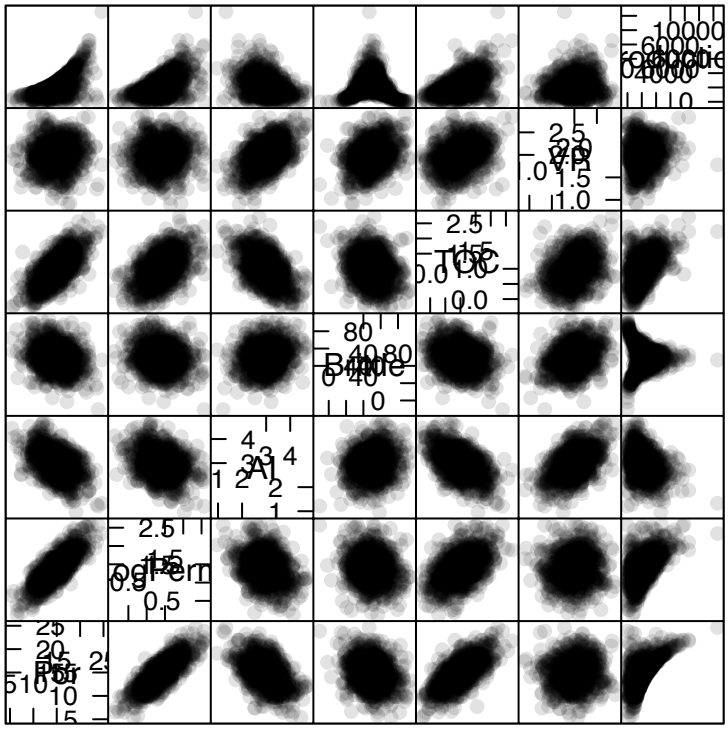
```
##                Por LogPerm     AI Brittle    TOC    VR Production
## Por          1.000   0.808 -0.506  -0.255  0.714 0.085      0.687
## LogPerm      0.808   1.000 -0.325  -0.152  0.507 0.046      0.569
## AI          -0.506  -0.325  1.000   0.170 -0.549 0.487     -0.327
## Brittle     -0.255  -0.152  0.170   1.000 -0.246 0.296     -0.071
## TOC          0.714   0.507 -0.549  -0.246  1.000 0.306      0.504
## VR           0.085   0.046  0.487   0.296  0.306 1.000      0.142
## Production   0.687   0.569 -0.327  -0.071  0.504 0.142      1.000
```

```
corrplot(cor_matrix, method = "circle")
```



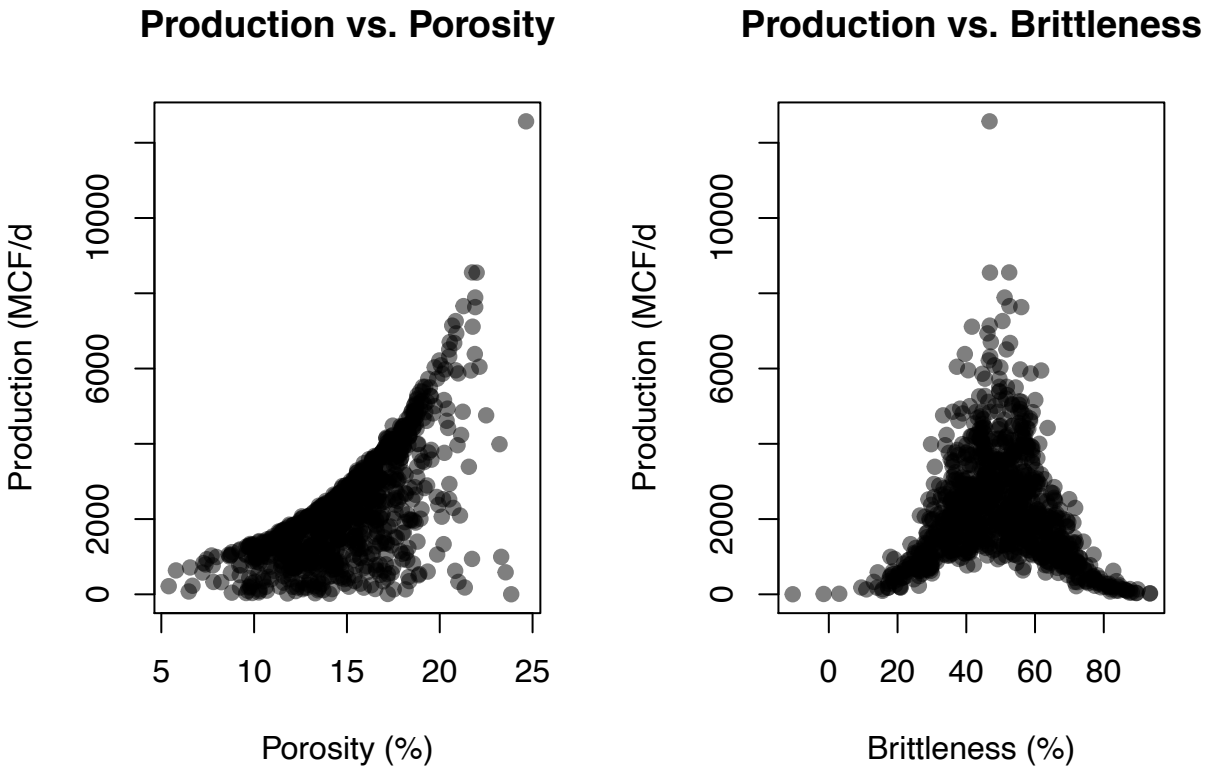

```
# Scatterplot from Lattice
splom(rock.data,col=rgb(0,0,0,50,maxColorValue=440), pch=19,main = "Unconventional Dataset")
```

Unconventional Dataset



Scatter Plot Matrix

```
# Comparison of Production against Porosity and Brittleness
par(mfrow=c(1,2))
plot(Por,Production, main="Production vs. Porosity", col=alpha("black",0.5),
     xlab="Porosity (%)", ylab="Production (MCF/d", pch=19)
plot(Brittle,Production, main="Production vs. Brittleness", col=alpha("black",0.5),
     xlab="Brittleness (%)", ylab="Production (MCF/d",pch=19)
```




```
par(mfrow=c(1,1))
#these two variables show highest and lowest correlation on the correlation matrix, respectively

# Comparison of Porosity and Brittleness
prod.deciles <- quantile(Production, 0:10/10); prod.deciles
```

```
##           0%          10%          20%          30%          40%          50%
##     2.713535   635.029558  1040.736163  1333.724569  1653.544353  1976.487820
##          60%          70%          80%          90%         100%
##  2370.192510  2807.014175  3355.907657  4144.058669 12568.644130
```

```
cut.prod <- cut(Production, prod.deciles, include.lowest=TRUE)
par(mfrow=c(1,2))
plot(Por, Brittle, col=grey(10:2/11)[cut.prod], pch=20,
     xlab="Porosity (%)", ylab="Brittleness (%)", main="Production (MCF/d)")

# Separate data into Training and Testing Sets
train <- rock.data[sample(nrow(rock.data),500,replace=FALSE),]
train_id <- sample(seq_len(nrow(rock.data)), size=500, replace=FALSE)
train <- rock.data[train_id,]
test <- rock.data[-train_id,]

# Check properties of training set to ensure correctness
head(train)
```

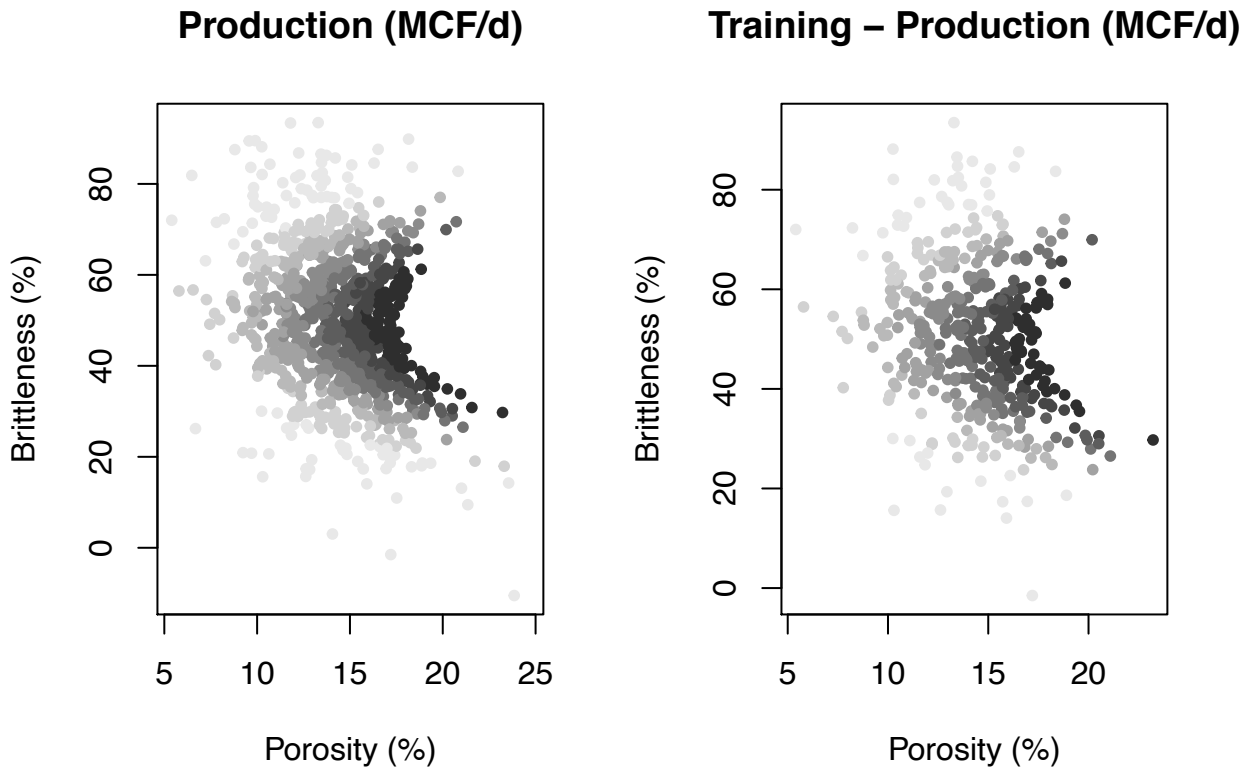
```
##       Por LogPerm   AI Brittle  TOC   VR Production
## 638 18.14    1.86 2.80   65.18 0.91 2.15   2689.943
## 252 13.84    1.49 3.08   52.98 1.07 2.12   2417.348
## 351  8.68    0.81 3.28   55.40 0.00 1.58   1002.715
## 600 16.48    1.84 2.47   56.75 1.12 1.85   3287.276
```

```
## 578 14.99    1.36 2.03   67.80 1.92 2.14   1491.311
## 900 14.44    1.07 2.34   68.76 0.98 1.97   1179.700
```

```
dim(train)
```

```
## [1] 500   7
```

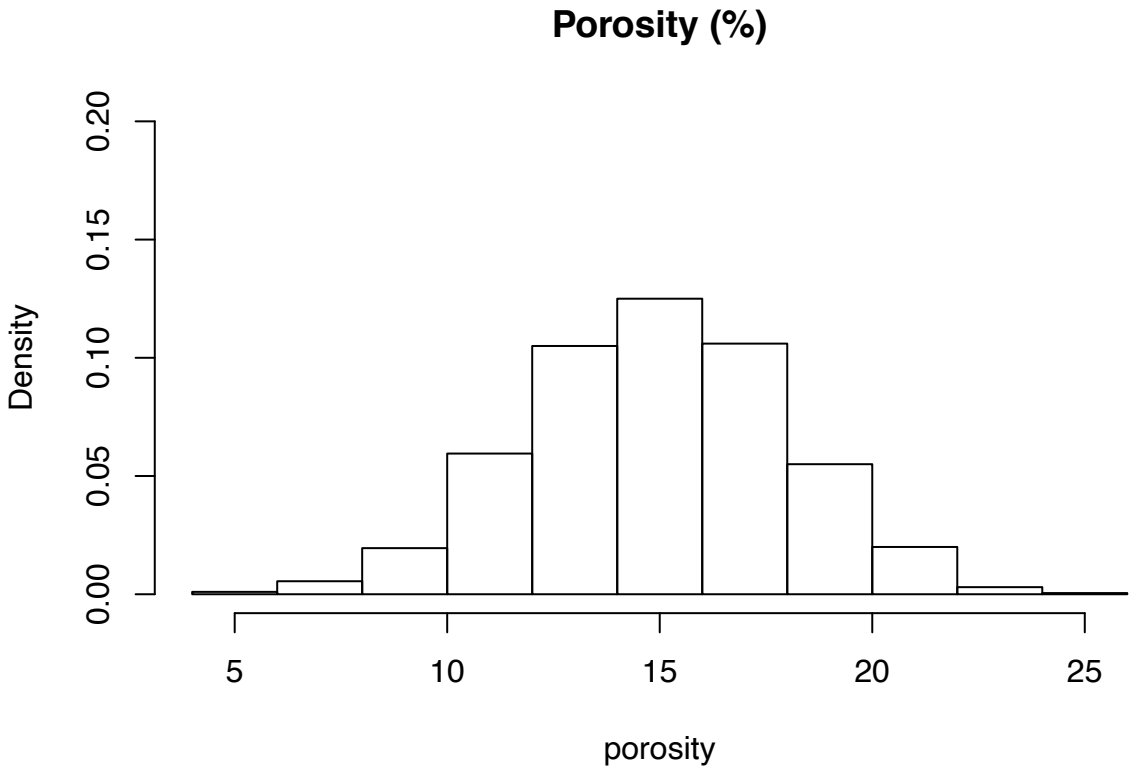
```
# Plot training set and compare to original/complete data set
cut.train.prod <- cut(train$Production, prod.deciles, include.lowest=TRUE)
plot(train$Por,train$Brittle, col=grey(10:2/11)[cut.train.prod], pch=20,
     xlab="Porosity (%)", ylab="Brittleness (%)", main="Training - Production (MCF/d)")
```



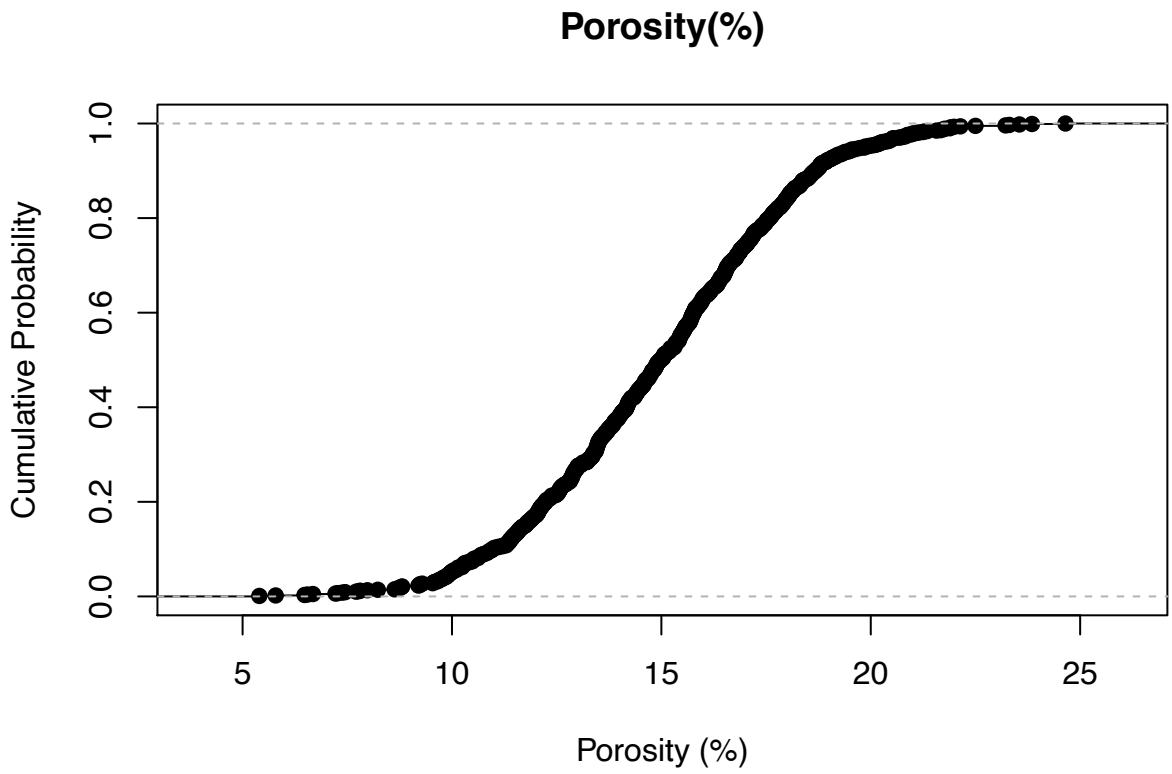

```
par(mfrow=c(1,1))

#####

# BOOTSTRAP
porosity <- rock.data$Por
summary(porosity)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5.40   12.86   14.98   14.95   17.08   24.65
```

```
hist(porosity,main="Porosity (%)", freq=FALSE, ylim=c(0,.2))
```

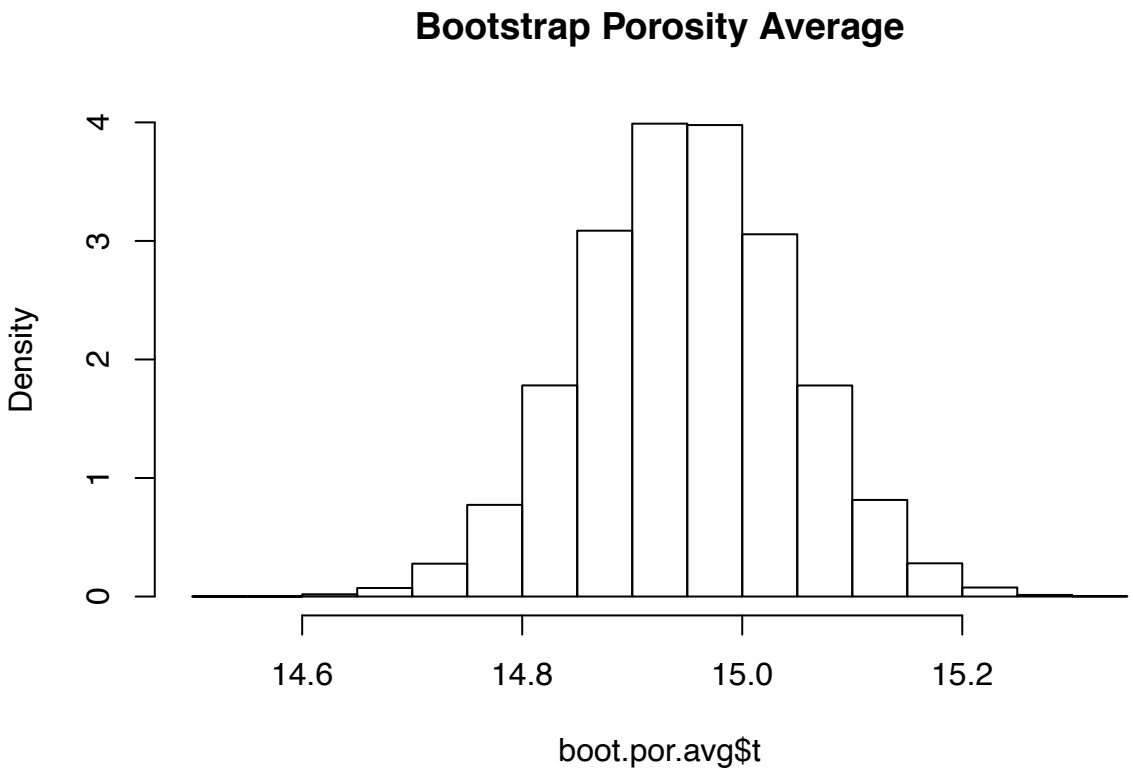



```
plot(ecdf(porosity), main="Porosity(%)", xlab="Porosity (%)", ylab="Cumulative Probability")
```

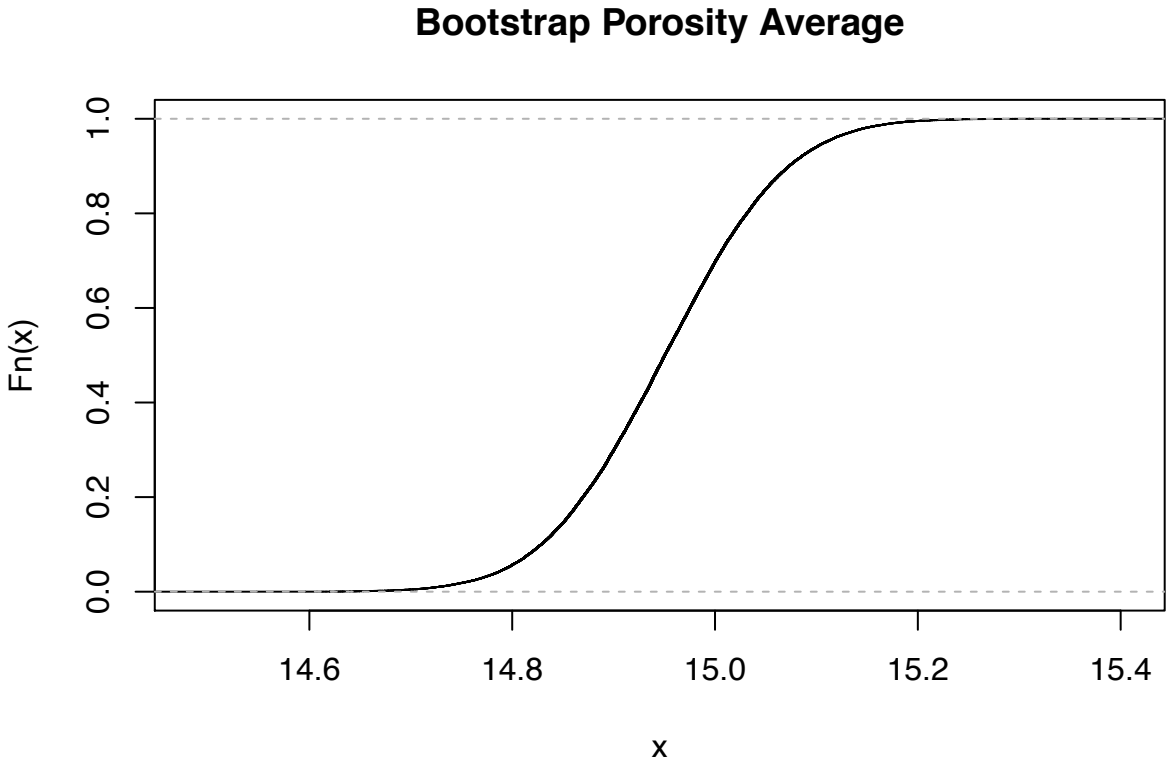



```
calc_average <- function(d, i=c(1:n)){
  d2 <- d[i]
  return(mean(d2))
```

```
}
boot.por.avg <- boot(data=porosity, statistic=calc_average, R=50000)
hist(boot.por.avg$t, main="Bootstrap Porosity Average", freq=FALSE)
```



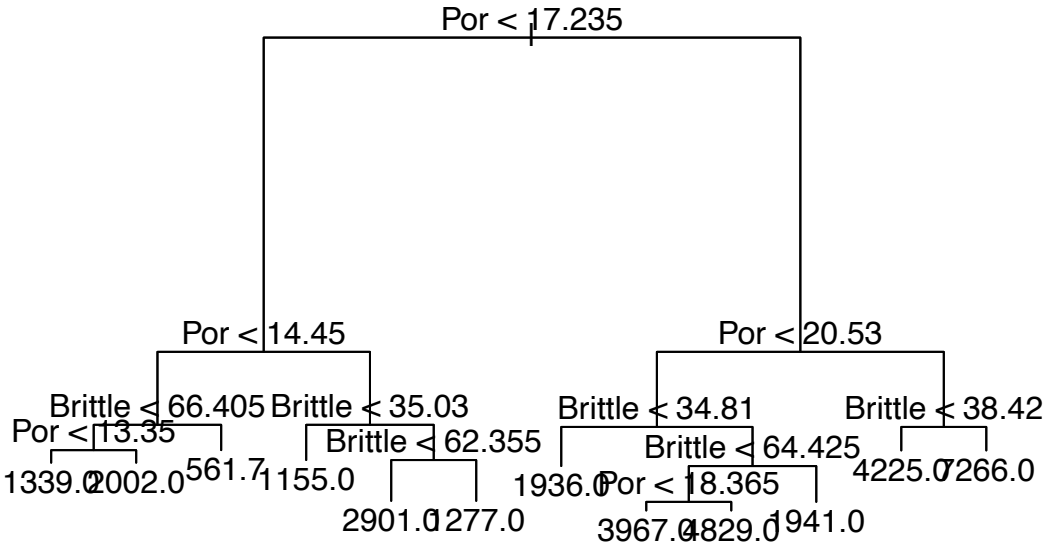

```
plot(ecdf(boot.por.avg$t), main="Bootstrap Porosity Average")
```



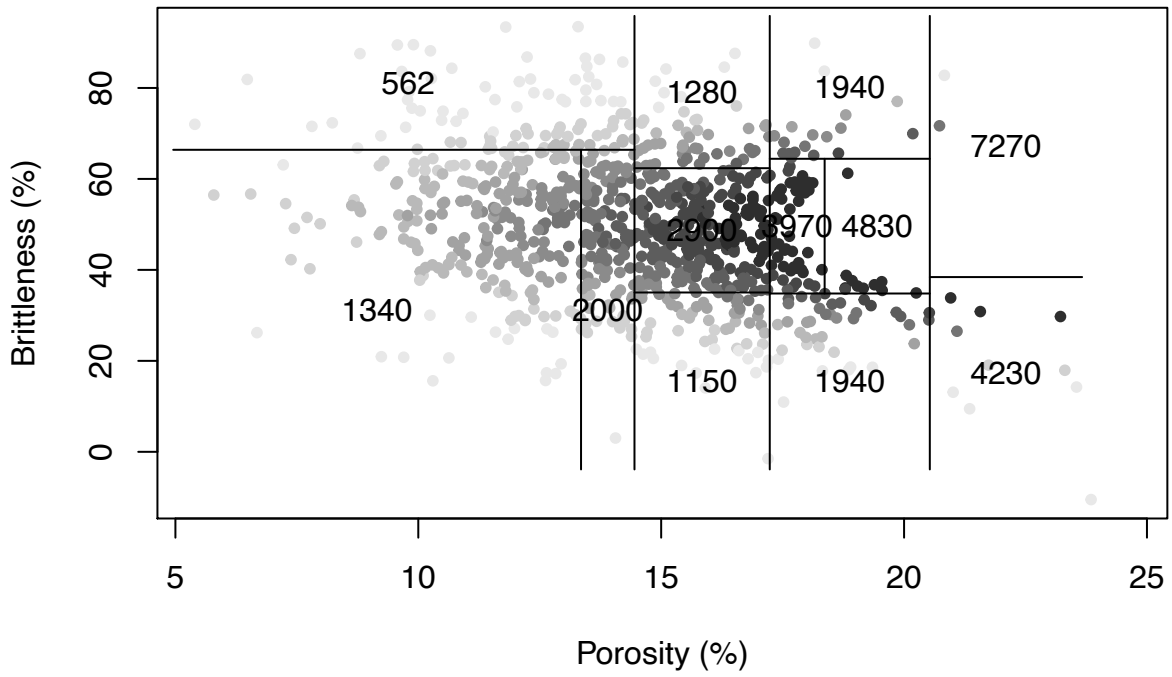
```r
# DECISION TREES
# Design Decision Tree control parameters
tree.control <- tree.control(nobs=500, mincut=5, minsize=10, mindev=0.01)

# Decision Tree on training data
tree.prod <- tree(Production~., data=train, control=tree.control)
summary(tree.prod)
```

```
##
## Regression tree:
## tree(formula = Production ~ ., data = train, control = tree.control)
## Variables actually used in tree construction:
## [1] "Por"     "Brittle"
## Number of terminal nodes:  12
## Residual mean deviance:  277900 = 135600000 / 488
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2130.00  -307.30    39.78     0.00   352.90  1822.00
```

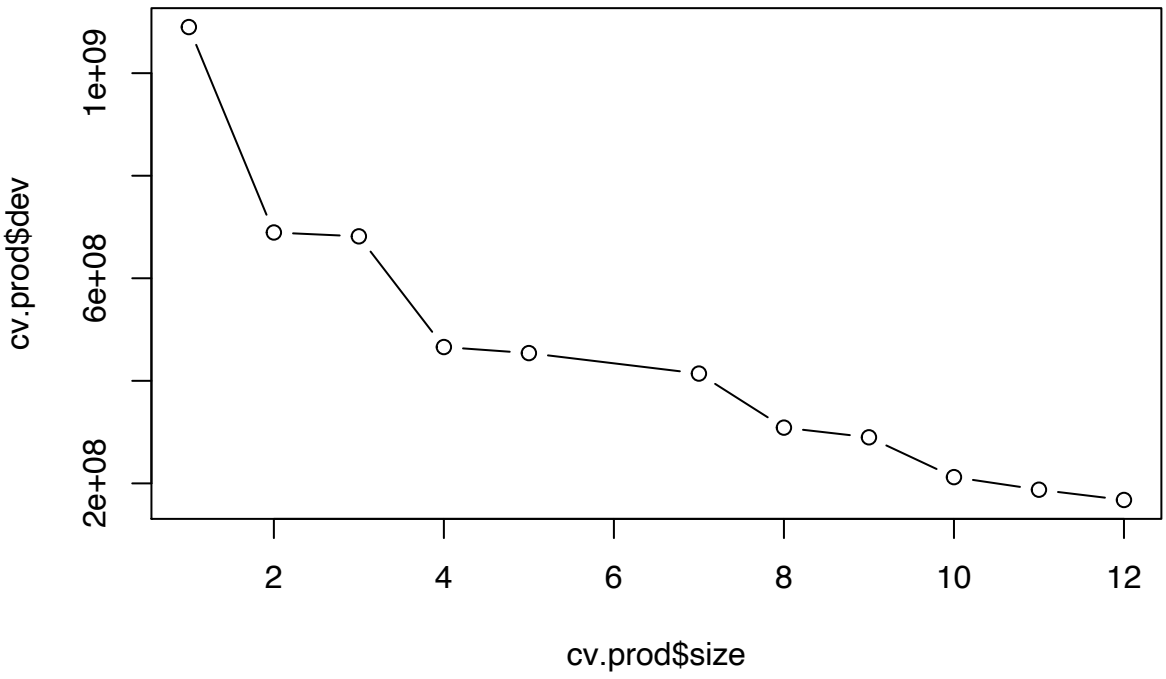
```r
plot(tree.prod); text(tree.prod, pretty=0)
```




```r
# Another way to visualize
plot(rock.data$Por,rock.data$Brittle, col=grey(10:2/11)[cut.prod], pch=20,
     xlab="Porosity (%)", ylab="Brittleness (%)")
partition.tree(tree.prod, ordvars=c("Por","Brittle"), add=TRUE)
```

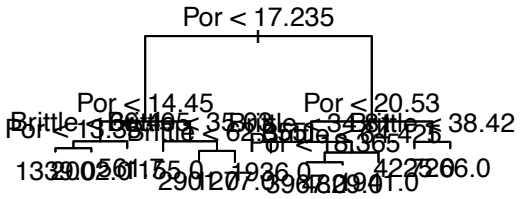


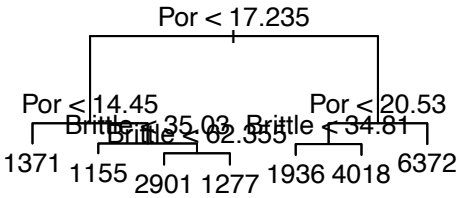

```
# Pruned Tree
cv.prod <- cv.tree(tree.prod, K=10)
plot(cv.prod$size, cv.prod$dev, type="b")      #looks like 6 nodes is good pruning
```

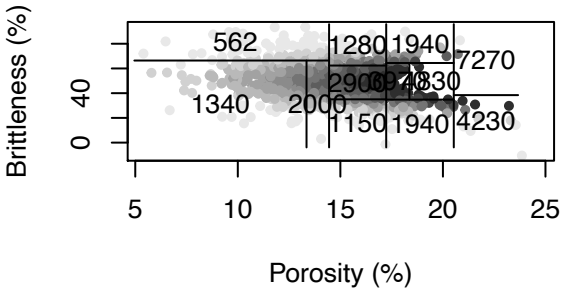


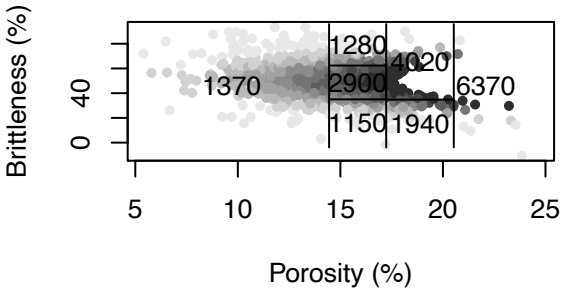

```
prune.prod <- prune.tree(tree.prod, best=6)

# Comparison of regular vs pruned trees
par(mfrow=c(2,2))
plot(tree.prod); text(tree.prod, pretty=0)
plot(prune.prod); text(prune.prod, pretty=0)
plot(rock.data$Por,rock.data$Brittle, col=grey(10:2/11)[cut.prod], pch=20, xlab="Porosity (%)", ylab="
```

```
    partition.tree(tree.prod, ordvars=c("Por","Brittle"), add=TRUE)
plot(rock.data$Por,rock.data$Brittle, col=grey(10:2/11)[cut.prod], pch=20, xlab="Porosity (%)", ylab="
    partition.tree(prune.prod, ordvars=c("Por","Brittle"), add=TRUE)
```

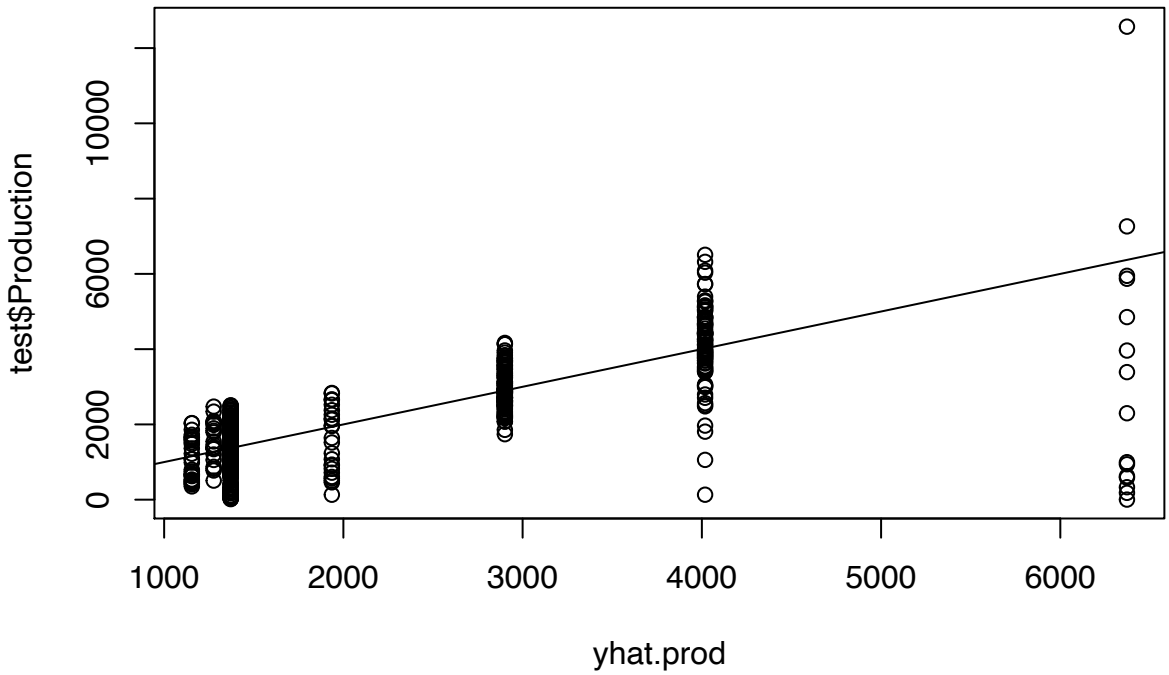









```
par(mfrow=c(1,1))

# Prediciton from pruned tree
yhat.prod <- predict(prune.prod, newdata=test)
plot(yhat.prod, test$Production)   #6 bins from the 6 terminal nodes
abline(0,1)
```




```
tree.error <- mean((yhat.prod-test$Production)^2); tree.error
```

```
## [1] 1127779
```

```
RMSE.tree <- sqrt(tree.error); RMSE.tree
```

```
## [1] 1061.97
```

```
var.prod <- var(test$Production); var.prod
```

```
## [1] 2109740
```

```
# LINEAR REGRESSION
lm.fit <- lm(Production ~., data=train); lm.fit
```

```
##
## Call:
## lm(formula = Production ~ ., data = train)
##
## Coefficients:
## (Intercept)          Por      LogPerm           AI      Brittle          TOC
##   -3447.941      327.196      174.896     -139.568       -4.026      -73.724
##          VR
##     663.055
```

```
lm.pred <- predict(lm.fit, test)
lm.error <- mean((test$Production - lm.pred)^2); lm.error
```

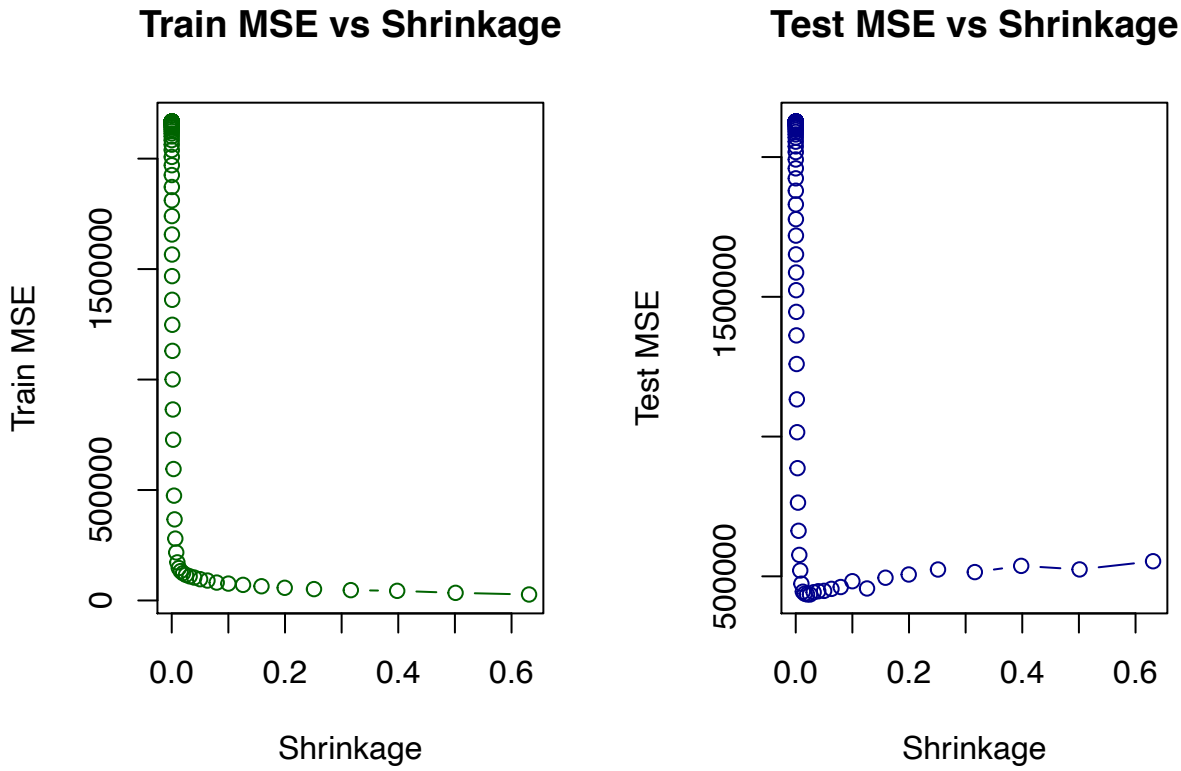
```
## [1] 1299266
```

```
# LASSO REGULARIZATION
x <- model.matrix(Production~., data=train)
y <- train$Production
lasso.fit <- glmnet(x,y,alpha=1)
x.test <- model.matrix(Production~., data=test)
lasso.pred <- predict(lasso.fit, s=0.01, newx=x.test)
```

```
lasso.error <- mean((test$Production - lasso.pred)^2); lasso.error
```

```
## [1] 1296476
```

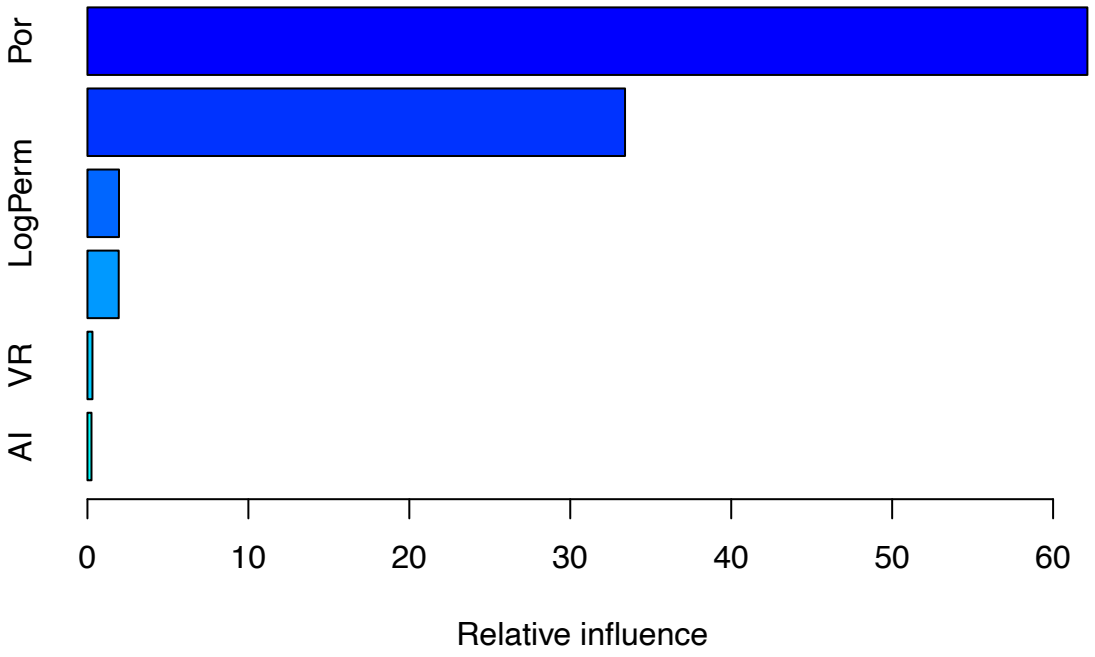
```
# BOOSTING
power <- seq(from=-10,to=-0.2,by=0.1)
lambdas <- 10^power
length.lambdas <- length(lambdas)
train.error <- rep(NA,length.lambdas)
test.error <- rep(NA,length.lambdas)
for (i in 1:length.lambdas) {
  boost.Prod <- gbm(Production~.,data=train,distribution="gaussian",n.trees=1000,shrinkage=lambdas[i])
  train.pred <- predict(boost.Prod, train, n.trees=1000)
  train.error[i] <- mean((train$Production-train.pred)^2)
  test.pred <- predict(boost.Prod, test, n.trees=1000)
  test.error[i] <- mean((test$Production-test.pred)^2)
}
par(mfrow=c(1,2))
plot(lambdas,train.error,type="b",xlab="Shrinkage",ylab="Train MSE",col="darkgreen",main="Train MSE vs
plot(lambdas,test.error,type="b",xlab="Shrinkage",ylab="Test MSE",col="darkblue",main="Test MSE vs Shr
```




```
par(mfrow=c(1,1))
# lambda-value for minimal test MSE
test.MSE.min <- min(test.error)
lambda.min.test.MSE <- lambdas[which.min(test.error)]
data.frame(lambda.min.test.MSE,test.MSE.min)
```

```
##   lambda.min.test.MSE test.MSE.min
## 1          0.02511886     435040.8
```

```
boost.error <- test.MSE.min
# Best variables for boosting
boost.best <- gbm(Production~., data=train, distribution="gaussian",
                  n.trees=1000, shrinkage=lambdas[which.min(test.error)])
summary(boost.best)
```

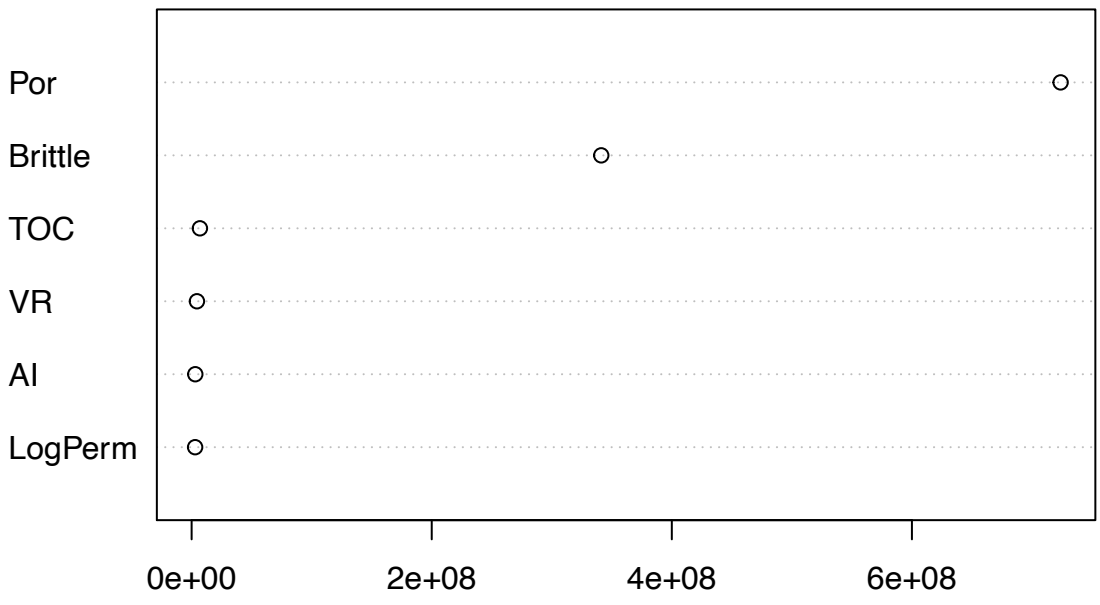



```
##             var    rel.inf
## Por         Por 62.1308635
## Brittle Brittle 33.4024443
## LogPerm LogPerm  1.9603981
## TOC         TOC  1.9433323
## VR           VR  0.3113005
## AI           AI  0.2516613
```

```
# RANDOM FOREST
rf.Prod <- randomForest(Production~., data=train, ntree=500, mtry=length(rock.data)-1)
importance(rf.Prod)
```

```
##         IncNodePurity
## Por         723846523
## LogPerm       2923200
## AI            3035284
## Brittle     341207514
## TOC           6901096
## VR            4426918
```

```
varImpPlot(rf.Prod)
```




```
rf.predict <- predict(rf.Prod, test)
rf.error <- mean((test$Production - rf.predict)^2); rf.error
```

```
## [1] 293684.8
```

```
# Comparing MSE for all methods
scientific(data.frame(tree.error,boost.error,lm.error,lasso.error,rf.error), digits=2)
```

```
##  tree.error boost.error    lm.error lasso.error    rf.error
##   "1.1e+06"   "4.4e+05"   "1.3e+06"   "1.3e+06"   "2.9e+05"
```

```
# PCA
PCA <- prcomp(rock.data[,-7], scale=TRUE)
PCA$center #the means substracted from variables
```

```
##      Por  LogPerm       AI  Brittle      TOC       VR
## 14.95046  1.39888  2.98261 49.71948  1.00381  1.99117
```

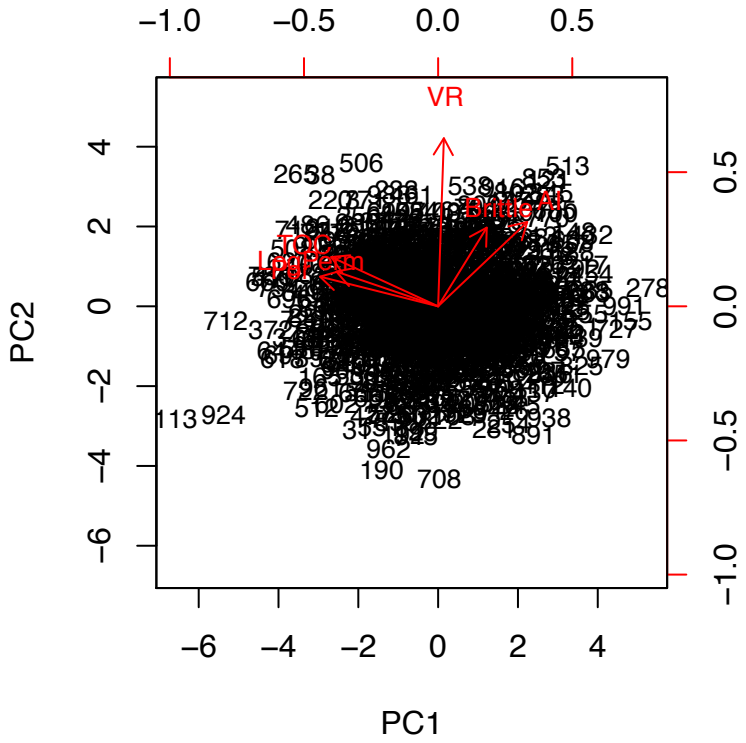
```
PCA$scale  #the standardization factor
```

```
##        Por    LogPerm         AI    Brittle        TOC         VR
##  3.0296343  0.4059657  0.5776287 15.0770064  0.5049778  0.3081940
```

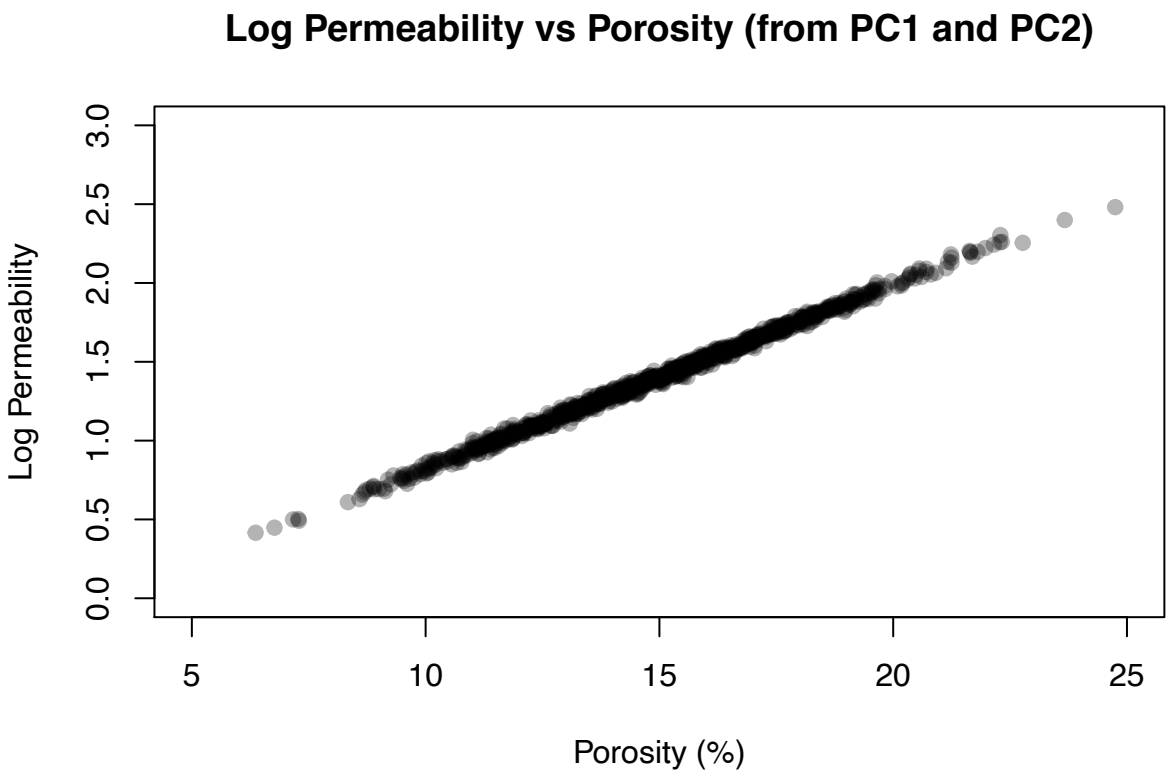
```
PCA$rotation  #principal component loadings
```

```
##                 PC1       PC2          PC3         PC4         PC5         PC6
## Por     -0.55024315 0.1381835 -0.005010946 -0.22272928  0.78608255  0.1028522
## LogPerm -0.47410001 0.1617101 -0.036607962 -0.66875903 -0.53354673 -0.1258099
## AI       0.41512557 0.3930196  0.457582045 -0.36799338  0.04539624  0.5712662
## Brittle  0.22649612 0.3665224 -0.870138284 -0.09235154  0.03386977  0.2180463
## TOC     -0.49811711 0.2295692  0.035471200  0.54699223 -0.30608030  0.5522984
## VR       0.02666979 0.7831028  0.175654719  0.24495835  0.02256057 -0.5428356
```

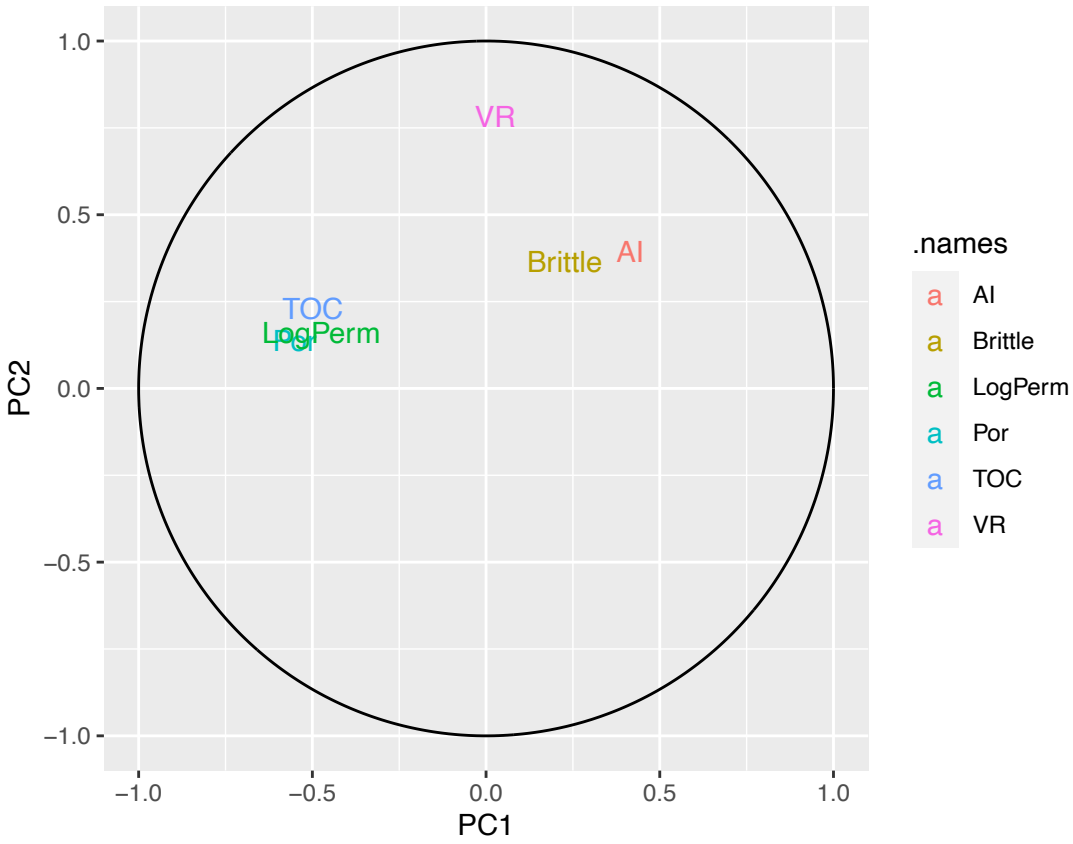
```r
# Plotting the PCA biplot
biplot(PCA, cex=0.8, scale=0, pch=21)
```




```r
# Analyze for two first principal components
nComp <- 2
Xhat2 <- t(t(PCA$x[,1:nComp] %*% t(PCA$rotation[,1:nComp])) *PCA$scale + PCA$center)
plot(LogPerm~Por, Xhat2, main="Log Permeability vs Porosity (from PC1 and PC2)", pch=19,
     xlim=c(5,25), ylim=c(0,3), xlab="Porosity (%)", ylab="Log Permeability", col=alpha("black",0.3))
```



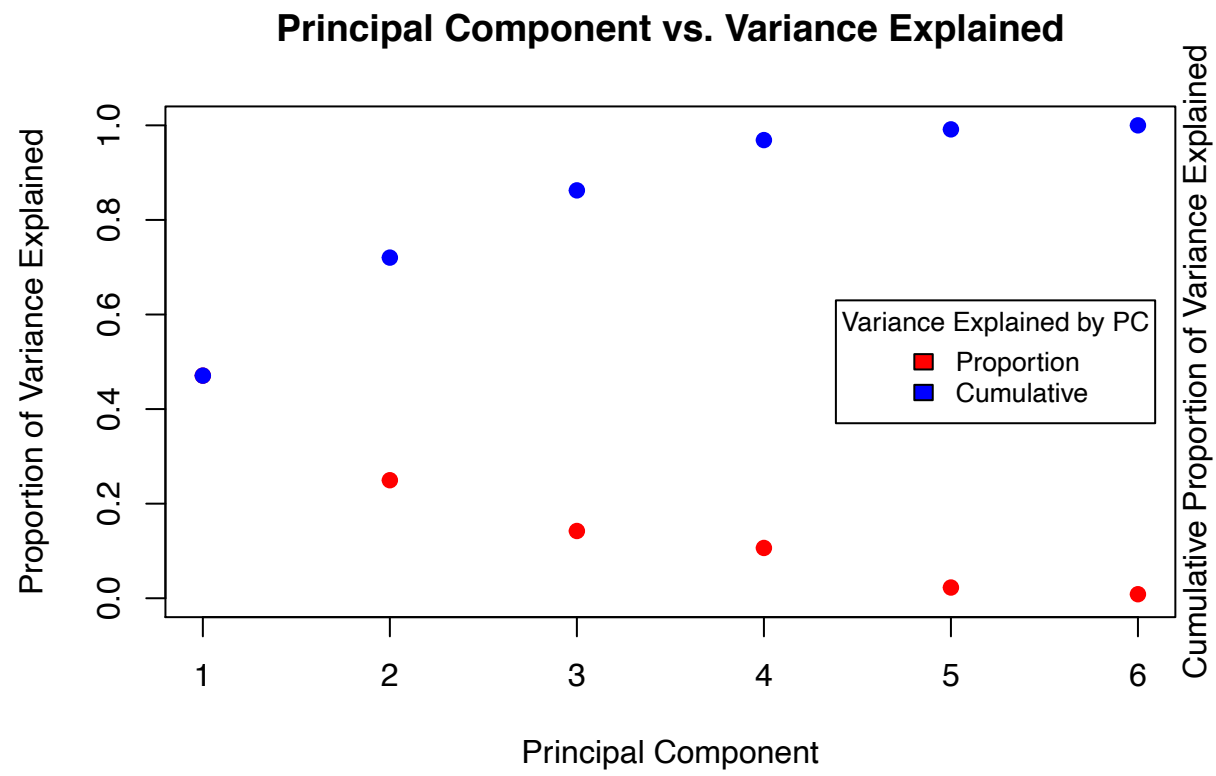

```
theta <- seq(0,2*pi,length.out=1000)
circle <- data.frame(x=cos(theta),y=sin(theta))
p <- ggplot(circle,aes(x,y)) + geom_path()
loadings <- data.frame(PCA$rotation, .names=row.names(PCA$rotation))
p + geom_text(data=loadings, mapping=aes(x=PC1, y=PC2, label=.names,colour=.names)) +
  coord_fixed(ratio=1) + labs(x="PC1", y="PC2")
```



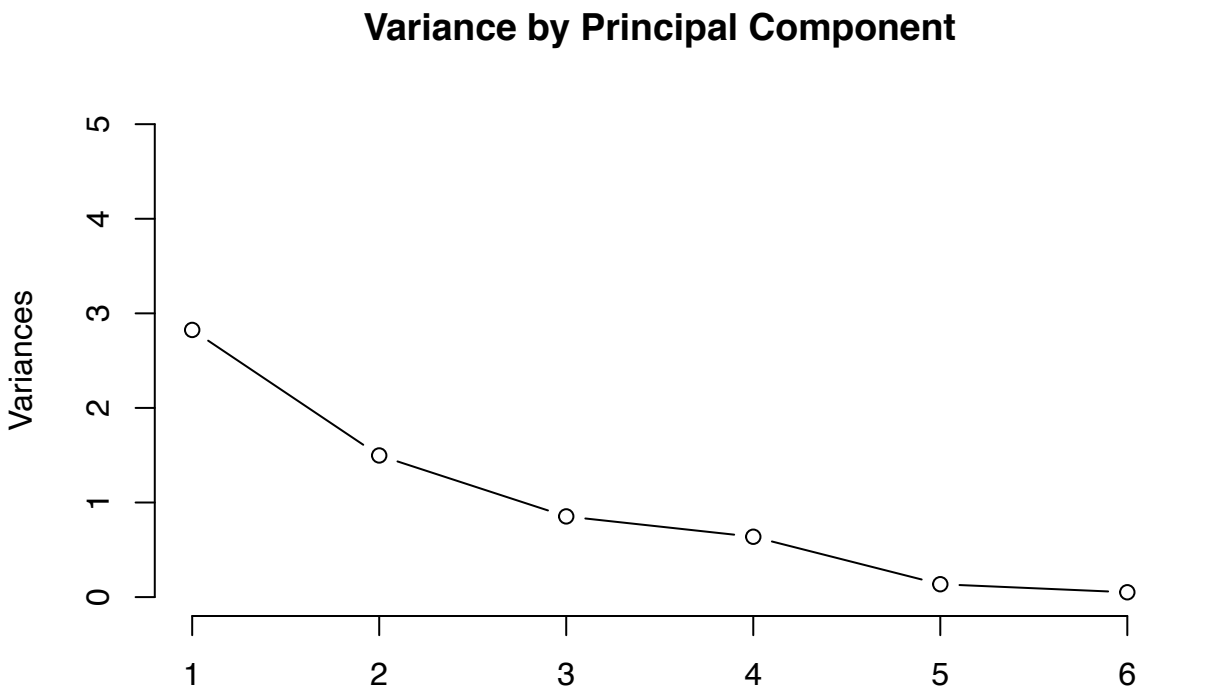

```
# Variance explained
pr.var <- PCA$sdev^2
pve <- pr.var / sum(pr.var); pve
```

```
## [1] 0.470751886 0.249563778 0.142183967 0.106323785 0.022664180 0.008512404
```

```
plot(pve, xlab="Principal Component", ylab="Proportion of Variance Explained",
     main="Principal Component vs. Variance Explained", pch=19, col="red", ylim=c(0,1))
points(cumsum(pve), xlab="Principal Component", ylim=c(0,1),
       pch=19, col="blue"); mtext("Cumulative Proportion of Variance Explained", side = 4)
legend("right",inset=.02,title="Variance Explained by PC", c("Proportion","Cumulative"),fill=c("red","
```




```
plot(PCA, type="l", main="Variance by Principal Component", ylim=c(0,5))
```




```
summary(PCA)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5     PC6
## Standard deviation     1.6806 1.2237 0.9236 0.7987 0.36876 0.22600
## Proportion of Variance 0.4708 0.2496 0.1422 0.1063 0.02266 0.00851
```

```
## Cumulative Proportion  0.4708 0.7203 0.8625 0.9688 0.99149 1.00000
```

```
#ggplot for Porosity and Brittleness against Production
ggplot(data=rock.data,aes(x=Por,y=Brittle)) + geom_point(aes(color=Production)) +
  ggtitle("Porosity vs. Brittleness \n for Unconventional Wells")
```