Machine Learning algorithms and deploying models

Afreen Fatima

2023-04-28

```
# Loading
library(lattice)
library(caret)
```

```
## Loading required package: ggplot2
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.1.3
```

```
## corrplot 0.92 loaded
```

```
library(ggplot2)
library(mlbench)
library(MASS)
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.1.3
```

```
# Loading dataset
dataset <- read.csv("escapesClean.csv", header = T, stringsAsFactors = T)
```

```
# Exploring dataset
summary(dataset)
```

```
##     Season            Species          Age         Average.Weight     Number
##   Autumn:57   Other        : 55   Min.   : 2.00   Min.   :  15   Min.   :     1
##   Spring:58   Salmon       :151   1st Qu.:10.00   1st Qu.: 600   1st Qu.:   216
##   Summer:51   Salmon.Brood:  2   Median :15.00   Median :2000   Median :  3000
##   Winter:55   Salmon.Fresh: 13   Mean   :15.31   Mean   :2191   Mean   : 13536
##                                   3rd Qu.:19.00   3rd Qu.:3400   3rd Qu.: 10775
##                                   Max.   :48.00   Max.   :9250   Max.   :336470
##      Cause     Producing      SLR               Cu                Zn
##   Human  :138   No : 21   Min.   :-0.7633   Min.   :-3.9100   Min.   :-0.4252
##   Natural: 83   Yes:200   1st Qu.: 0.6599   1st Qu.: 0.5429   1st Qu.: 6.8072
##                           Median : 2.2654   Median : 1.7553   Median : 9.5058
##                           Mean   : 3.1627   Mean   : 2.1218   Mean   :10.1518
##                           3rd Qu.: 3.1242   3rd Qu.: 3.6470   3rd Qu.:13.8066
##                          Max.   :35.2477   Max.   : 8.4502   Max.   :24.7346
##        N                 P                Org
##  Min.   :-105.5   Min.   :-13.97   Min.   :  65.13
##  1st Qu.: 240.9   1st Qu.: 80.49   1st Qu.: 356.62
##  Median : 358.3   Median :121.98   Median : 564.55
##  Mean   : 340.0   Mean   :122.59   Mean   : 553.28
##  3rd Qu.: 437.0   3rd Qu.:162.52   3rd Qu.: 726.65
##  Max.   : 696.5   Max.   :244.29   Max.   :1092.56
```

```
# Making another copy of dataset
results1 <- dataset
```

There are no missing values in the dataset Next we analyse the correlation between pairs of variables. Our target cause is category so won't work without some preprocessing

```
# Train test split for linear regression
split = createDataPartition(dataset$Cause, p = 0.8, list = F)
trainData = dataset[split, ]
testData = dataset[-split, ]
```

```
# Creating training control for use by all models
control = trainControl(method = "repeatedcv", number = 10, repeats = 3)
```

```
# Pre-processing
prep = c('range') # to normalise to a scale [0, 1]
prep = c('center', 'scale') # to standardise to zero mean and stdev 1
pca = c('range', 'pca')
pca = c('center', 'scale', 'pca')
```

```
# Creating grid of values for tuneGrid parameter of the train function
grid <- expand.grid(size=c(0,5,10,20,50), k=c(0,1,2,3,4,5))
```

MODEL 1: LOGISTIC REGRESSION

Designing and implementing a Logistic Regression model to predict Cause. We are using caret train for predicting cause where method is logistic regression, trControl and preProcess is defined in initial stages of loading and summarising data. trControl is using method "repeatedcv" for 10 fold cross validation and repeats 3. For preprocessing we are using principal component analysis which might improve a model's ability since it creates a new set of variables, that are independent, from the existing ones. Running the model with logistic regression confusion matrix predicts the cause of escapes that can be seen as Human predicted correctly as 49.9 and wrongly predicted as 23.4. Natural cause of escapes is lower than human where 12.5 is wrongly predicted and 14.2 is correctly predicted. Overall accuracy of model1 is 64% and Kappa is 18.5%

```
set.seed(123)
model1 = train(Cause ~ ., data = dataset, method = "glm", family = "binomial", trControl = control,  pr

model1$results
```

```
##   parameter  Accuracy     Kappa AccuracySD   KappaSD
## 1      none 0.6510132 0.2202608 0.09665065 0.1983665
```

```
confusionMatrix(model1)
```

```
## Cross-Validated (10 fold, repeated 3 times) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##            Reference
## Prediction Human Natural
##    Human    49.6    22.0
##    Natural  12.8    15.5
##
##  Accuracy (average) : 0.6516
```

```
#saveRDS(model1, "model1cause.rds")
varImp(model1)
```

```
## glm variable importance
##
##                       Overall
## SeasonWinter        100.0000
## Number               85.3766
## SpeciesSalmon        76.6953
## Cu                   74.4614
## N                    74.4256
## SeasonSpring         60.6912
## SpeciesSalmon.Fresh  56.5321
## P                    47.5565
## Zn                   36.3722
## Org                  35.4569
## ProducingYes         24.7147
## SpeciesSalmon.Brood  23.7744
## Average.Weight       22.9422
## SeasonSummer         17.6708
## SLR                   0.1444
## Age                   0.0000
```

MODEL 2: LINEAR DISCRIMINANT ANALYSIS TO PREDICT THE CAUSE:

We are using caret train for predicting cause where method is linear discriminant analysis, trControl and preProcess is defined in initial stages of loading and summarising data. trControl is using method "repeatedcv" for 10 fold cross validation and repeats 3. For preprocessing we are using principal component analysis which might improve a model's ability since it creates a new set of variables, that are independent, from the existing ones. Running the model with LDA, confusion matrix predicts the cause of escapes that can be seen as Human predicted correctly as 52.2% and wrongly predicted as 24%. Natural cause of escapes is lower than human where 10.3% is wrongly predicted and 13.6% is correctly predicted as natural cause. Overall accuracy of model1 is 64% and Kappa is 18.5%

```
set.seed(123)
model2 = train(Cause ~ ., data = dataset, method = "lda", trControl = control, preProcess = pca)
```

```
model2$results
```

```
##   parameter  Accuracy     Kappa AccuracySD   KappaSD
## 1      none 0.6496832 0.1989226 0.09645714 0.2006416
```

```
confusionMatrix(model2)
```

```
## Cross-Validated (10 fold, repeated 3 times) Confusion Matrix
## 
## (entries are percentual average cell counts across resamples)
## 
##           Reference
## Prediction Human Natural
##    Human    51.7    24.3
##    Natural  10.7    13.3
##                             
##  Accuracy (average) : 0.6501
```

```
#saveRDS(model2, "model2cause.rds")
```

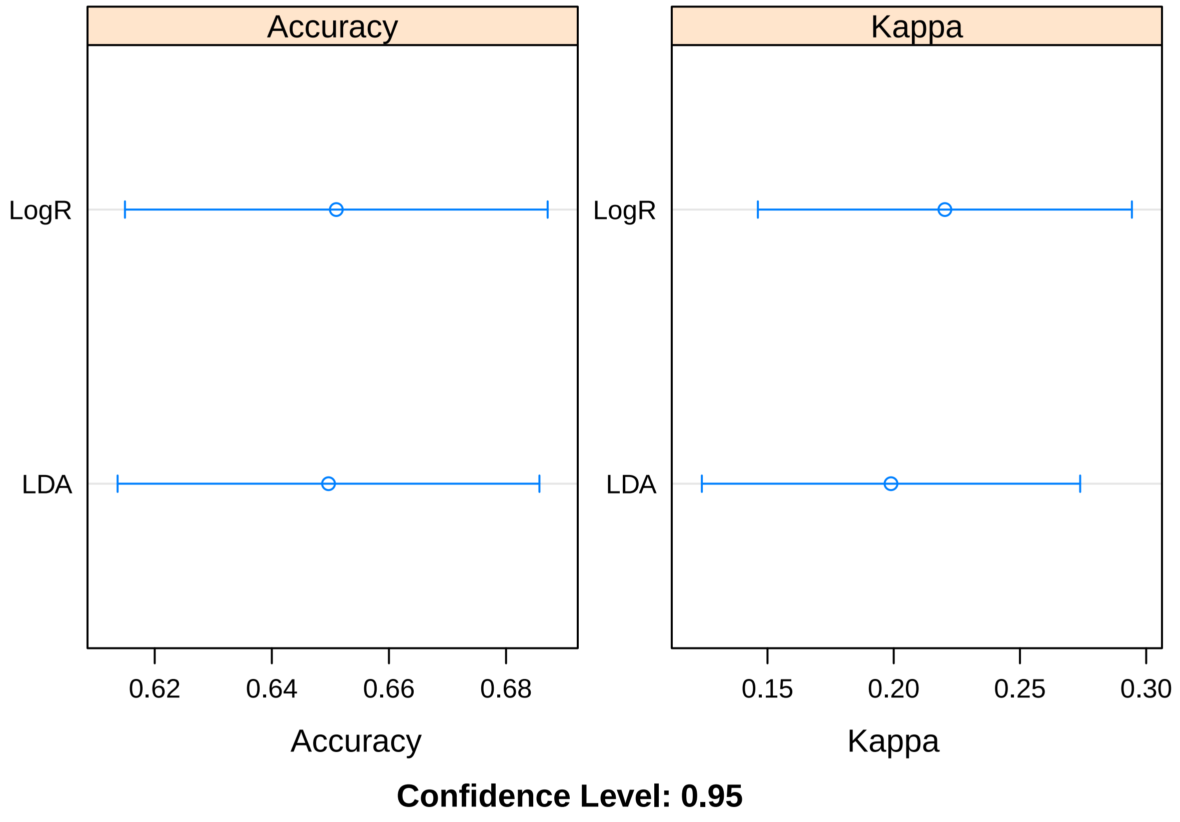
Comparing and contrasting the effectiveness of both models:

An overview is seen of relative effectiveness of both model1 and model2 by plotting the estimates of accuracy and kappa, and their confidence intervals as 0.96. We are seeing both the estimated values and range in which we can be confident to 95% that the metric lies. We can see Accuracy is little high ranged with Linear discriminant analysis than Logistic regression. Accuracy mean of Logistic regression is 64% and LDA is 66%. Kappa logistic regression mean is 18.5% and LDA is 21%. We can critically compare the results and conclude that model2 with LDA is more effective with high accuracy and kappa than model1 with logistic regression

```
results = resamples(list(LogR = model1, LDA = model2))
summary(results)
```

```
## 
## Call:
## summary.resamples(object = results)
## 
## Models: LogR, LDA 
## Number of resamples: 30 
## 
## Accuracy 
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LogR  0.5 0.5909091 0.6363636 0.6510132 0.6921937 0.8260870    0
## LDA   0.5 0.5762987 0.6363636 0.6496832 0.7272727 0.8181818    0
## 
## Kappa 
##            Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LogR -0.1100917 0.07831988 0.1698113 0.2202608 0.3304348 0.6034483    0
## LDA  -0.1100917 0.03883495 0.1698113 0.1989226 0.3400000 0.5849057    0
```

```r
dotplot(results, conf.level = 0.95, scales = "free")
```



MODEL3 : LINEAR REGRESSION:

```r
#Feature selection using correlation matrix and linear regression using caret train function
# calculate correlation matrix
options(scipen=999)
correlationMatrix <- cor(dataset[-c(1,2,6,7)])
print(correlationMatrix)
```

```
##                        Age Average.Weight     Number        SLR          Cu
## Age             1.00000000    0.046600937 0.05110458 0.06759792 -0.03990393
## Average.Weight  0.04660094    1.000000000 0.01339962 0.03486187  0.01416259
## Number          0.05110458    0.013399622 1.00000000 0.65854059  0.05943139
## SLR             0.06759792    0.034861872 0.65854059 1.00000000  0.05595894
## Cu             -0.03990393    0.014162589 0.05943139 0.05595894  1.00000000
## Zn             -0.13490187    0.002102437 0.09730514 0.08493140  0.37362291
## N              -0.15960826   -0.025161055 0.03499374 0.13739761  0.34079432
## P              -0.03305502    0.039115292 0.08462716 0.25889292  0.40279344
## Org            -0.11414952    0.017950181 0.07226190 0.21231897  0.32604087
##                          Zn           N           P         Org
## Age            -0.134901869 -0.15960826 -0.03305502 -0.11414952
## Average.Weight  0.002102437 -0.02516106  0.03911529  0.01795018
## Number          0.097305140  0.03499374  0.08462716  0.07226190
## SLR             0.084931404  0.13739761  0.25889292  0.21231897
## Cu              0.373622908  0.34079432  0.40279344  0.32604087
## Zn              1.000000000  0.54090244  0.49378165  0.60873910
## N               0.540902442  1.00000000  0.57542769  0.63641894
## P               0.493781655  0.57542769  1.00000000  0.57071861
## Org             0.608739103  0.63641894  0.57071861  1.00000000
```

```
# finding attributes that are highly correlated that is greater than 0.75
imp <- findCorrelation(correlationMatrix, cutoff=0.5)
print(imp)
```

```
## [1] 9 8 7 4
```

```
# # trying spearman in case any relationships are non-linear
#corrplot(cor(dataset[-c(1,2,6,7)], method = "spearman"))
```

```
modelAll = lm(Cause ~ . , results1)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored

## Warning in Ops.factor(y, z$residuals): ’-’ not meaningful for factors
```

```
#summary(modelAll)
varImp(modelAll)
```

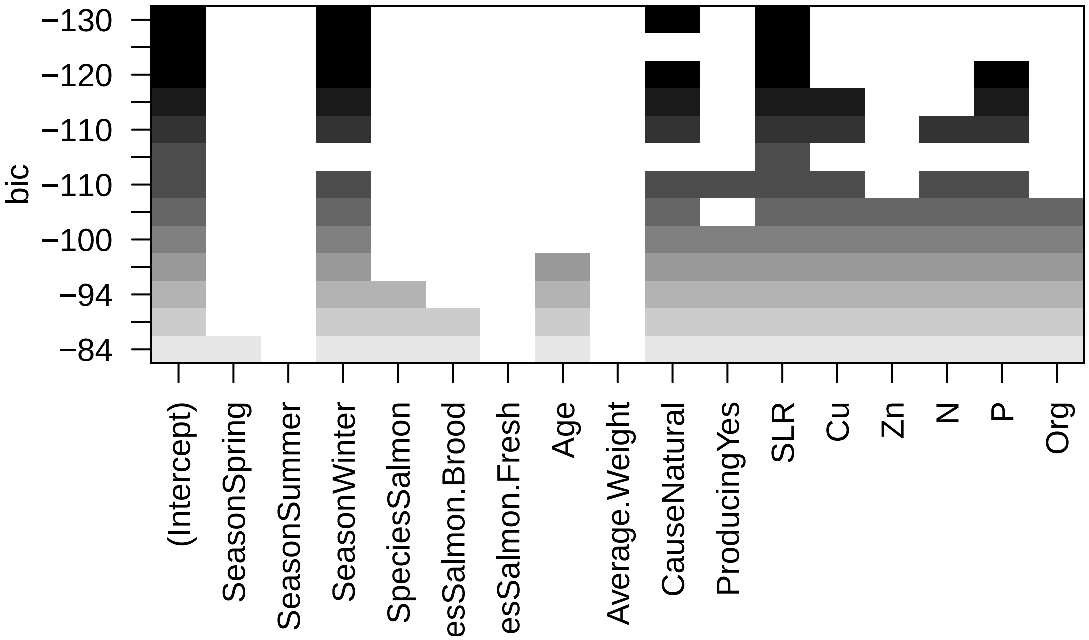
```
## Warning in Ops.factor(r, 2): ’^’ not meaningful for factors

##                       Overall
## SeasonSpring               NA
## SeasonSummer               NA
## SeasonWinter               NA
## SpeciesSalmon              NA
## SpeciesSalmon.Brood        NA
## SpeciesSalmon.Fresh        NA
## Age                        NA
## Average.Weight             NA
## Number                     NA
## ProducingYes               NA
## SLR                        NA
## Cu                         NA
## Zn                         NA
## N                          NA
## P                          NA
## Org                        NA
```

```
fullSearch = regsubsets(Number ~ .,data = trainData,
method = "exhaustive", nvmax = 13)
full = summary(fullSearch)
```

```
full$outmat
```

```
##           SeasonSpring SeasonSummer SeasonWinter SpeciesSalmon
## 1  ( 1 )  " "          " "          " "          " "
## 2  ( 1 )  " "          " "          "*"          " "
## 3  ( 1 )  " "          " "          "*"          " "
## 4  ( 1 )  " "          " "          "*"          " "
## 5  ( 1 )  " "          " "          "*"          " "
## 6  ( 1 )  " "          " "          "*"          " "
## 7  ( 1 )  " "          " "          "*"          " "
## 8  ( 1 )  " "          " "          "*"          " "
## 9  ( 1 )  " "          " "          "*"          " "
## 10  ( 1 ) " "          " "          "*"          " "
## 11  ( 1 ) " "          " "          "*"          "*"
## 12  ( 1 ) " "          " "          "*"          "*"
## 13  ( 1 ) "*"          " "          "*"          "*"
##           SpeciesSalmon.Brood SpeciesSalmon.Fresh Age Average.Weight
## 1  ( 1 )  " "                 " "                 " " " "
## 2  ( 1 )  " "                 " "                 " " " "
## 3  ( 1 )  " "                 " "                 " " " "
## 4  ( 1 )  " "                 " "                 " " " "
## 5  ( 1 )  " "                 " "                 " " " "
## 6  ( 1 )  " "                 " "                 " " " "
## 7  ( 1 )  " "                 " "                 " " " "
## 8  ( 1 )  " "                 " "                 " " " "
## 9  ( 1 )  " "                 " "                 " " " "
## 10  ( 1 ) " "                 " "                 "*" " "
## 11  ( 1 ) " "                 " "                 "*" " "
## 12  ( 1 ) "*"                 " "                 "*" " "
## 13  ( 1 ) "*"                 " "                 "*" " "
##           CauseNatural ProducingYes SLR Cu  Zn  N   P   Org
## 1  ( 1 )  " "          " "          "*" " " " " " " " " " "
## 2  ( 1 )  " "          " "          "*" " " " " " " " " " "
## 3  ( 1 )  "*"          " "          "*" " " " " " " " " " "
## 4  ( 1 )  "*"          " "          "*" " " " " " " "*" " "
## 5  ( 1 )  "*"          " "          "*" "*" " " " " "*" " "
## 6  ( 1 )  "*"          " "          "*" "*" " " "*" "*" " "
## 7  ( 1 )  "*"          "*"          "*" "*" " " "*" "*" " "
## 8  ( 1 )  "*"          " "          "*" "*" "*" "*" "*" "*"
## 9  ( 1 )  "*"          "*"          "*" "*" "*" "*" "*" "*"
## 10  ( 1 ) "*"          "*"          "*" "*" "*" "*" "*" "*"
## 11  ( 1 ) "*"          "*"          "*" "*" "*" "*" "*" "*"
## 12  ( 1 ) "*"          "*"          "*" "*" "*" "*" "*" "*"
## 13  ( 1 ) "*"          "*"          "*" "*" "*" "*" "*" "*"
```

```
plot(fullSearch) # shows similar info but in a blocked tabular form
```



```
full$rsq # shows R^2 increasing as number of variables increases
```

```
##  [1] 0.4997516 0.5375525 0.5600957 0.5626304 0.5678973 0.5718459 0.5742350
##  [8] 0.5774157 0.5808475 0.5825390 0.5835482 0.5840800 0.5845250
```

```
full$adjr2 # shows adjusted R^2 varies as number of variables # increases (this can go down as the vari
```

```
##  [1] 0.4969093 0.5322674 0.5525111 0.5525178 0.5553362 0.5568230 0.5567035
##  [8] 0.5574117 0.5583929 0.5575413 0.5559520 0.5538312 0.5515910
```

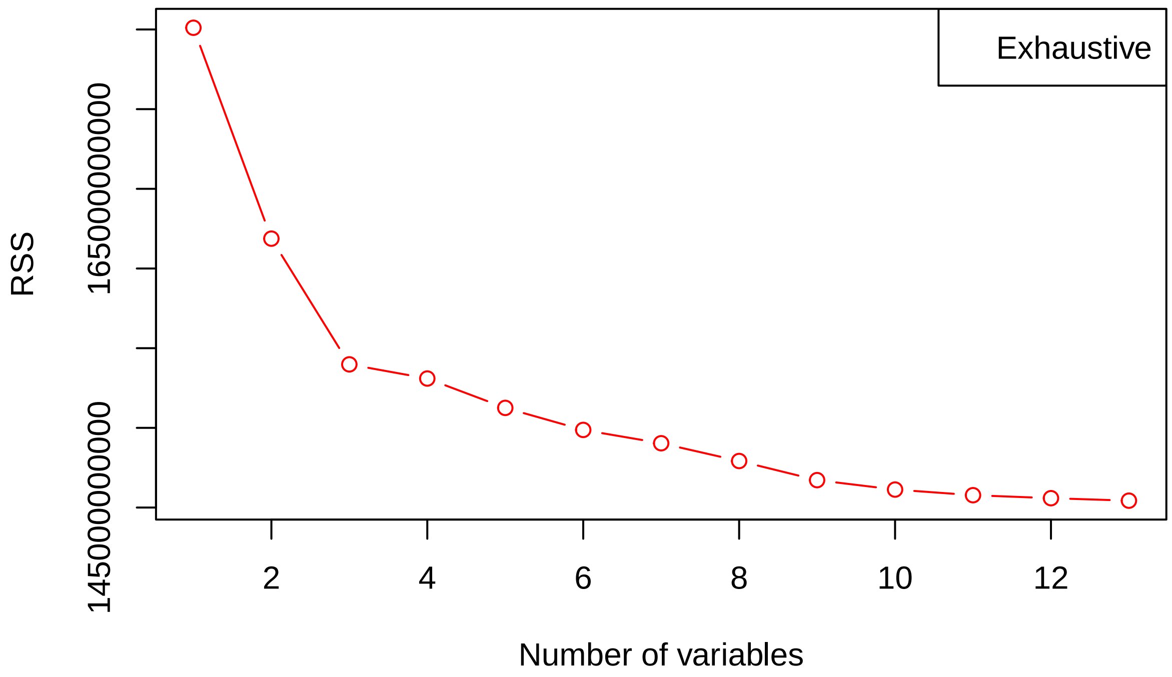
```
full$rss # shows RSS decreasing as number of variables increases
```

```
##  [1] 175106579122 161874791987 153983794573 153096550012 151252923242
##  [6] 149870752845 149034480626 147921105596 146719847692 146127759363
## [11] 145774489748 145588353783 145432584992
```

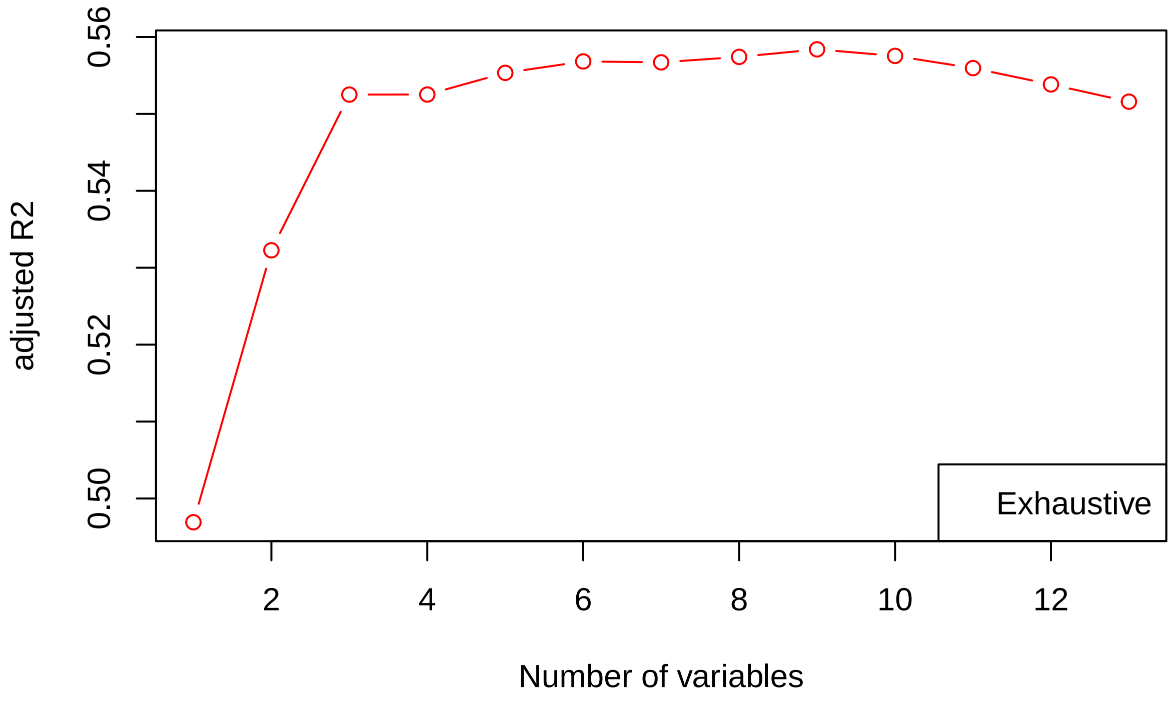
```
full$cp #
```

```
##  [1] 19.9783648  7.3205464  0.5791114  1.5962457  1.5539255  2.0227943
##  [7]  3.0963944  3.8630274  4.5323062  5.8764066  7.4850641  9.2788676
## [13] 11.1063112
```

```
plot(full$rss, type = "b", col = "red",
ylab = "RSS", xlab = "Number of variables")
legend("topright", col = c("red", "blue", "green", "purple"),
legend = "Exhaustive")
```



```
plot(full$adjr2, type = "b", col = "red",
ylab = "adjusted R2", xlab = "Number of variables")
legend("bottomright", col = c("red", "blue", "green", "purple"),
legend ="Exhaustive")
```



```
q = full$which[3,-c(1)]
vars = paste(names(q[q == TRUE]), collapse = "+")
form = as.formula(paste("Number ~ ", vars))
```

```
# Model fitting
model3 = train(Number~., data = trainData, method = "lm", preProcess = prep,
trControl = control)
model3$results
```

```
##   intercept    RMSE  Rsquared      MAE   RMSESD RsquaredSD    MAESD
## 1      TRUE 29959.2 0.4270696 17225.57 14125.96  0.3062819 5577.438
```

```
summary(model3)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -142833   -8454     -19    9063  171452
##
## Coefficients:
##                      Estimate Std. Error t value             Pr(>|t|)
## (Intercept)          13893.2     2252.0   6.169        0.00000000534 ***
## SeasonSpring          1531.0     3087.5   0.496              0.62067    
## SeasonSummer           802.7     2983.9   0.269              0.78827    
## SeasonWinter          9049.5     3027.1   2.989              0.00323 ** 
## SpeciesSalmon        -1693.5     2794.7  -0.606              0.54538    
## SpeciesSalmon.Brood  -1152.0     2439.3  -0.472              0.63737    
## SpeciesSalmon.Fresh   -312.8     2546.4  -0.123              0.90238    
## Age                   2063.4     2440.8   0.845              0.39915    
## Average.Weight        -281.8     2309.7  -0.122              0.90306    
## CauseNatural          6451.5     2512.3   2.568              0.01114 *  
## ProducingYes          3082.7     2330.7   1.323              0.18783    
## SLR                  31293.5     2466.4  12.688 < 0.0000000000000002 ***
## Cu                    3344.3     2603.7   1.284              0.20083    
## Zn                    3332.3     3089.7   1.079              0.28241    
## N                     4069.7     3281.8   1.240              0.21674    
## P                    -5067.3     3113.0  -1.628              0.10553    
## Org                  -4559.9     3273.5  -1.393              0.16554    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 30050 on 161 degrees of freedom
## Multiple R-squared:  0.5848, Adjusted R-squared:  0.5435 
## F-statistic: 14.17 on 16 and 161 DF,  p-value: < 0.00000000000000022
```

```
#saveRDS(model1, "modelLR.rds")
```

```
# Evaluating the model on the test data:
pred = predict(model3, testData)
postResample(pred, testData$Number)
```

```
##           RMSE       Rsquared            MAE 
## 30711.50473812     0.04553883 19796.55378771
```

MODEL 4: REGRESSION MODEL OF RANDOM FOREST - tuning parameter mtry.

```
model4 = train(Number ~ ., data = trainData, method = "rf",
trControl = control, preProcess = prep, tune = "mtry")
```

```
model4$results
```

```
##   mtry     RMSE  Rsquared      MAE   RMSESD RsquaredSD    MAESD
## 1    2 26869.66 0.4018926 13874.03 18693.95  0.3642574 5857.243
## 2    9 27064.48 0.4007699 12960.12 16535.88  0.3620164 5295.244
## 3   16 30495.30 0.3986796 13278.76 20910.19  0.3491661 6217.723
```

```
summary(model4)
```

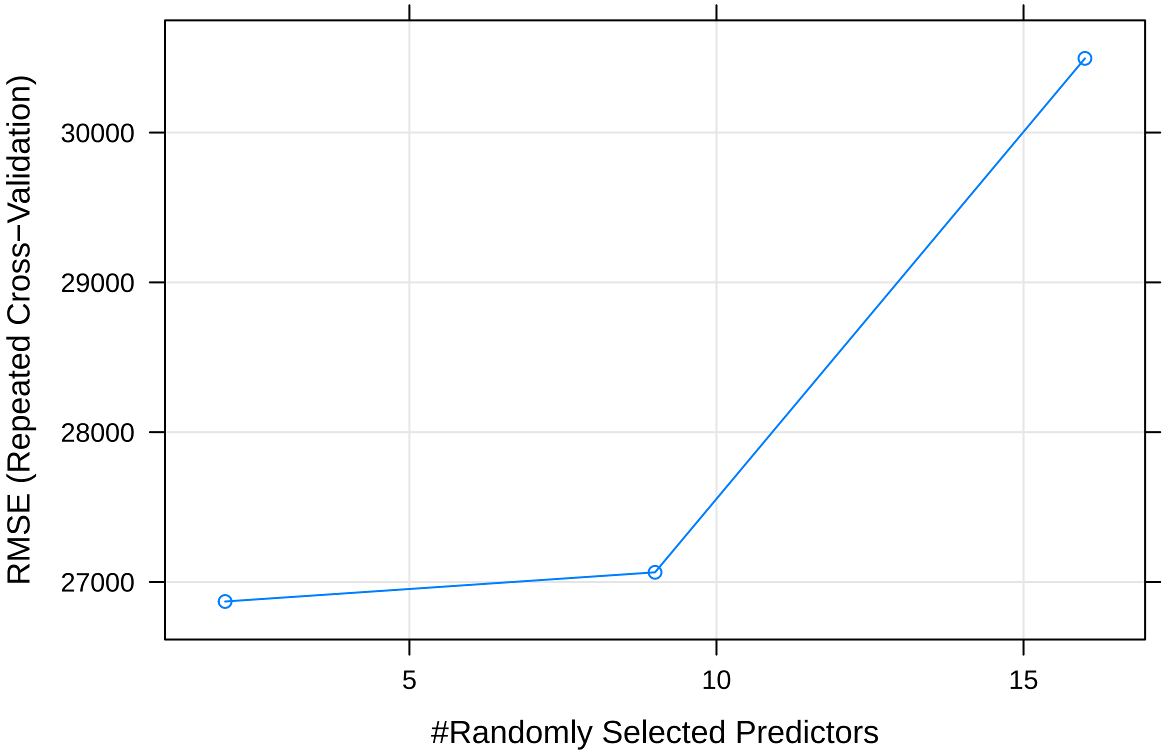
```
##                 Length Class      Mode
## call              5    -none-     call
## type              1    -none-     character
## predicted       178    -none-     numeric
## mse             500    -none-     numeric
## rsq             500    -none-     numeric
## oob.times       178    -none-     numeric
## importance       16    -none-     numeric
## importanceSD      0    -none-     NULL
## localImportance   0    -none-     NULL
## proximity         0    -none-     NULL
## ntree             1    -none-     numeric
## mtry              1    -none-     numeric
## forest           11    -none-     list
## coefs             0    -none-     NULL
## y               178    -none-     numeric
## test              0    -none-     NULL
## inbag             0    -none-     NULL
## xNames           16    -none-     character
## problemType       1    -none-     character
## tuneValue         1    data.frame list
## obsLevels         1    -none-     logical
## param             1    -none-     list
```

```
#saveRDS(model4, "modelRF.rds")
```

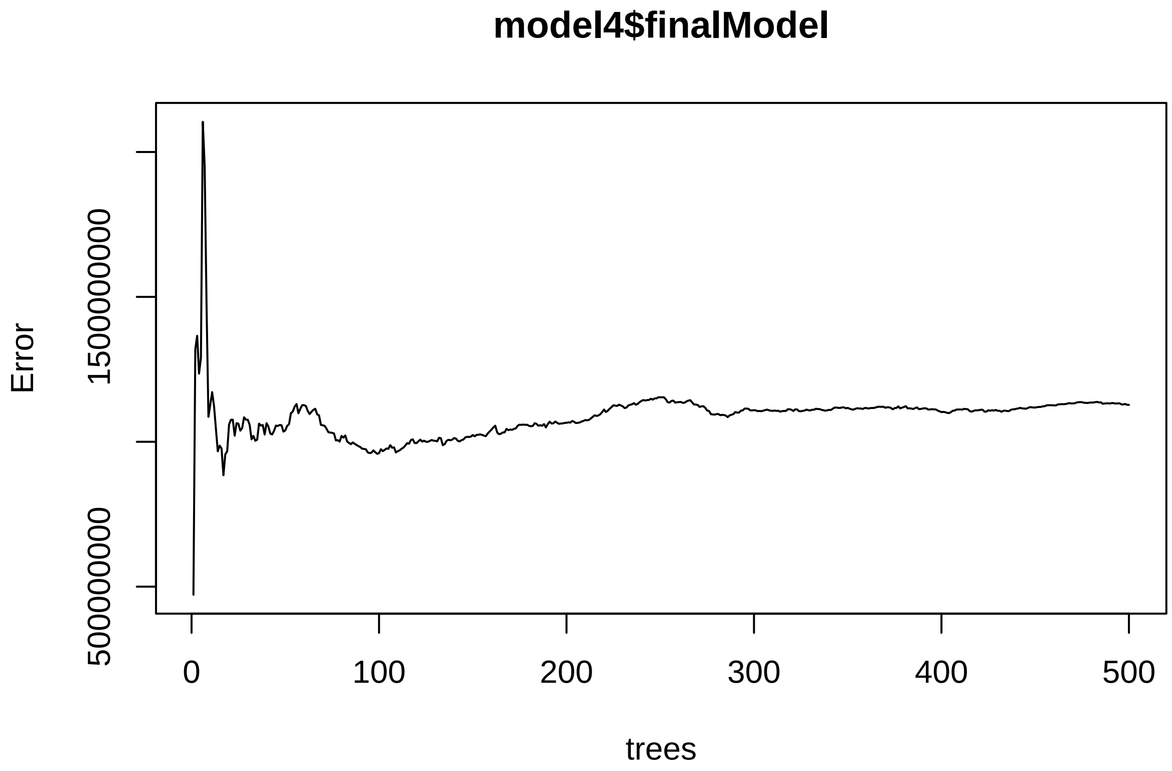
```
model4$bestTune
```

```
##   mtry
## 1    2
```

```
plot(model4)
```



```
plot(model4$finalModel)
```



```r
# Evaluation on test data
pred = predict(model4, testData)
postResample(pred, testData$Number)
```

```
##             RMSE        Rsquared             MAE
## 25604.451998608     0.008806967 12796.873975242
```

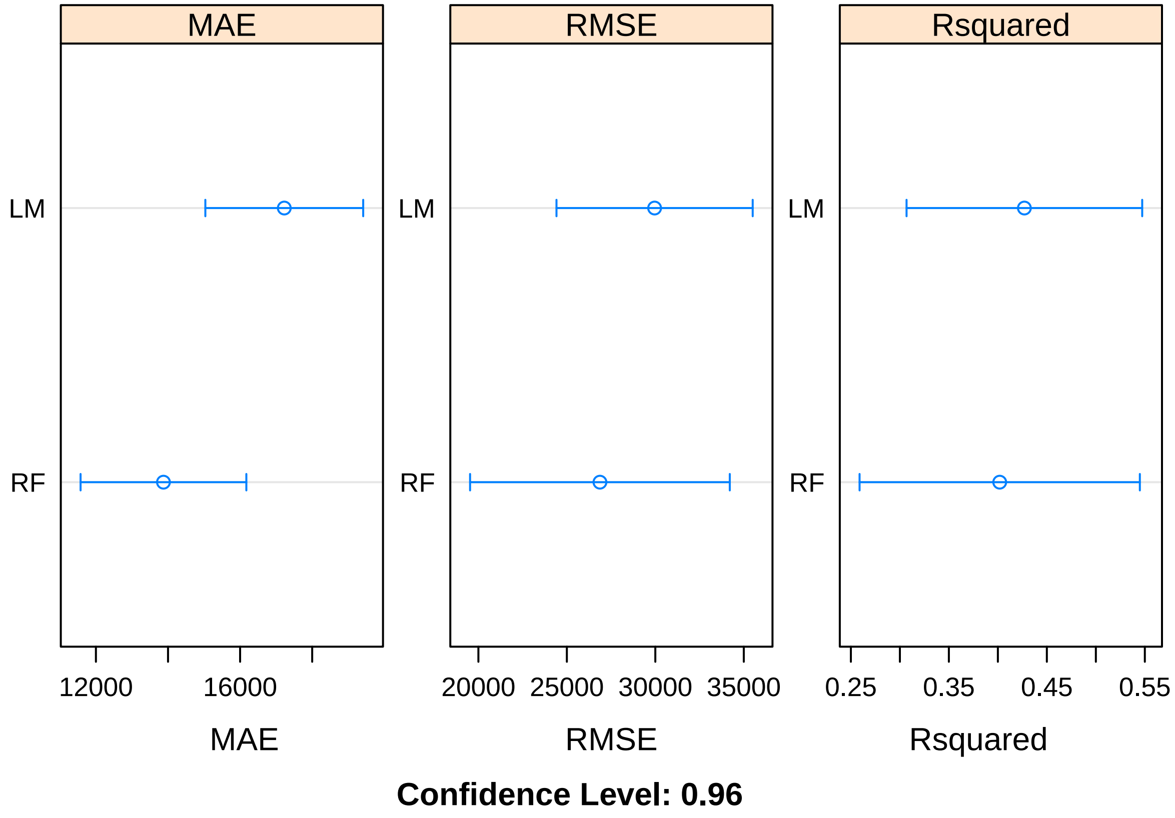
Comparison of effectiveness of model 3 and model 4

```r
results = resamples(list(LM = model3, RF = model4))
summary(results)
```

```
## 
## Call:
## summary.resamples(object = results)
## 
## Models: LM, RF 
## Number of resamples: 30 
## 
## MAE 
##        Min.   1st Qu.   Median     Mean  3rd Qu.     Max. NA’s
## LM 6846.665 13629.482 17423.77 17225.57 21525.02 28383.40    0
## RF 6675.086  9154.239 12096.10 13874.03 18474.80 26795.95    0
```

```
##
## RMSE
##        Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA's
## LM 9034.729 17629.28 24535.24 29959.20 42361.86 53182.01    0
## RF 7758.768 12097.29 17369.65 26869.66 45707.54 64225.46    0
##
## Rsquared
##               Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LM 0.0009842226401 0.15714609 0.4157685 0.4270696 0.6190297 0.9459993    0
## RF 0.0000003820678 0.07445763 0.2713043 0.4018926 0.6762085 0.9797170    0
```

```
dotplot(results, conf.level = 0.96, scales = "free")
```



The above plot shows three comparison boxes for results obtained for model 3 and model 4 interpreting, MAE, RMSE and RSquared. MAE: Linear regression model for MAE is ranging between 14000 to 18500, For random forest MAE ranging between 12000 to 17500. More effective here is Linear regression model with highest range RMSE: Linear model is ranging between 20500 to 35000. RF ranging between 15000 to 40000. Random forest is more effective with highest range. RSquared: Linear regression model is ranging between 0.16 to 0.40 and RF is ranging between 0.15 to 0.36. Hence we can conclude that Linear regression model is more effective than random forest as it has highest range for MAE and RSquared.

MODEL 5:

```r
dataset5 <- read.csv("escapesClean.csv", header = T, stringsAsFactors = T)
model5 = train(Cause ~ Season+Number+Cu+N+Org , data = dataset5, method = "glm", family = "binomial")
summary(model5)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2010  -0.9063  -0.6638   1.1352   1.9371
##
## Coefficients:
##                  Estimate   Std. Error z value Pr(>|z|)
## (Intercept)   0.319506372  0.479762113   0.666   0.5054
## SeasonSpring  0.713955475  0.438446082   1.628   0.1034
## SeasonSummer  0.419128161  0.447162999   0.937   0.3486
## SeasonWinter  0.908114649  0.444994979   2.041   0.0413 *
## Number        0.000017893  0.000009521   1.879   0.0602 .
## Cu           -0.130925781  0.071034738  -1.843   0.0653 .
## N            -0.002682070  0.001524212  -1.760   0.0785 .
## Org          -0.000754395  0.000865305  -0.872   0.3833
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 292.54  on 220  degrees of freedom
## Residual deviance: 259.80  on 213  degrees of freedom
## AIC: 275.8
##
## Number of Fisher Scoring iterations: 5
```

```r
saveRDS(model5, "model5.rds")
summary(dataset5)
```

```
##     Season           Species          Age        Average.Weight     Number
##  Autumn:57   Other        : 55   Min.   : 2.00   Min.   :  15   Min.   :     1
##  Spring:58   Salmon       :151   1st Qu.:10.00   1st Qu.: 600   1st Qu.:   216
##  Summer:51   Salmon.Brood:  2   Median :15.00   Median :2000   Median :  3000
##  Winter:55   Salmon.Fresh: 13   Mean   :15.31   Mean   :2191   Mean   : 13536
##                                 3rd Qu.:19.00   3rd Qu.:3400   3rd Qu.: 10775
##                                 Max.   :48.00   Max.   :9250   Max.   :336470
##      Cause     Producing      SLR                Cu                 Zn
##  Human  :138   No : 21   Min.   :-0.7633   Min.   :-3.9100   Min.   :-0.4252
##  Natural: 83   Yes:200   1st Qu.: 0.6599   1st Qu.: 0.5429   1st Qu.: 6.8072
##                          Median : 2.2654   Median : 1.7553   Median : 9.5058
##                          Mean   : 3.1627   Mean   : 2.1218   Mean   :10.1518
##                          3rd Qu.: 3.1242   3rd Qu.: 3.6470   3rd Qu.:13.8066
##                          Max.   :35.2477   Max.   : 8.4502   Max.   :24.7346
##        N                 P                Org
##  Min.   :-105.5   Min.   :-13.97   Min.   :  65.13
##  1st Qu.: 240.9   1st Qu.: 80.49   1st Qu.: 356.62
##  Median : 358.3   Median :121.98   Median : 564.55
##  Mean   : 340.0   Mean   :122.59   Mean   : 553.28
##  3rd Qu.: 437.0   3rd Qu.:162.52   3rd Qu.: 726.65
##  Max.   : 696.5   Max.   :244.29   Max.   :1092.56
```

The model 5 is trained with Liner regression and 5 inputs has been choosen as per the most important variables to the model as per Cause. Variable Importance shows the variables Season, Number, Cu, N and Org are important. As per coursework specification only 5 variables has been choosen to deploy the model5. In Shiny app "app.R" 5 inputs has been choosen as per 5 variables. Model 5 is choosen in the app.R file. Title panel updated. And input layer defines 5 different input with slider inputs and one selection input. Each of these given it's variable name so the server can work for it. Each has min and max value as per each variable's summary. Layout of server is equal to function that relates to input and output. We have 5 different variables reading 5 different values that is stored in output modelCalcLR variable that is assigned to renderText. This function read content from input object and it's going to set things up as a single rule data frame as the same structure as the model that we trained with linear regression to feed the new data in to our linear regression model in predict function that will enable to make a single prediction to a one row of data. Finally shiny app have the input for ui and server. Running the app will deploy our model that predict the cause. Human cause is 1 and Natural cause is 2. Our model has one select input and 4 slider input. That when changed as per values will refer and give the cause as 1(Human) and 2(Natural)

Bibliography

Brownlee, J., 2019. Feature selection using Caret R. [Online] Available at: https://machinelearningmastery.com/feature-selection-with-the-caret-r-package/ [Accessed 01 05 2022]. Lab Notes - David Lonie, 2022. Lab 6 - 10. Aberdeen: RGU. Lecture notes - David Lonie, 2022. Lecture 6 - 10. Aberdeen: RGU.