Author Name : SRAVAN KUMAR VASAM
R & D Project : Machine Learning Classification models including linear and non linear models
Technologies : R version 4.0.2, Rstudio, Linux
Year of submission : November, 2020

Data source : social networks advertising

important points to be noted

1) classification is used to predict a category.
2) Regression is used to predict continuous number.
3) There are variety of classification applications like medicine, marketing etc.
4) there are linear and non-linear models in classification.

classification models including Linear & non linear models :

1. Logistic Regression (Linear model), library(class), function : glm
2. K-Nearest Neighbors (K-NN) (non-linear model), library(class), function : knn
3. Support Vector Machine (SVM) (Linear model), library(e1071), function : svm
4. Kernel SVM (non-linear model), library(e1071), function : svm
5. Naive Bayes, library(e1071), function : naiveBayes
6. Decision Tree Classification, library(rpart), function : rpart
7. Random Forest Classification (non-linear model), library(randomForest), function : randomForest

---

1. Logistic Regression :
Pros:Probabilistic approach, gives informations about statistical significance of features
cons:The Logistic Regression Assumptions
formula :

```
classifier = glm(formula = Purchased ~ .,
                 family = binomial,
                 data = training_set)
```

logistic regression classification model output :

Data	
classifier	List of 30
dataset	400 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 57 10 7 26
prob_pred	Named num [1:100] 0.01624 0.01171 0.00378 0.00245 0.00733 ...
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
y_pred	Named num [1:100] 0 0 0 0 0 0 0 0 1 0 ...

logistic regression classification model predict variables:

```
> y_pred = ifelse(prob_pred > 0.5, 1, 0)
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46
  0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0
 48  52  66  69  74  75  82  84  85  86  87  89 103 104 107 108
  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 193 199 200 208 213 224 226 228 229 230 234 236 237 239 241
  0   0   0   0   1   1   1   0   1   0   1   1   1   0   1   1
255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324
  1   0   1   1   1   1   1   0   1   1   1   0   1   0   0   0
326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383
  0   1   0   1   0   1   1   1   1   1   1   0   1   0   1   1
389 392 395 400
  0   0   0   1
> cm = table(test_set[, 3], y_pred > 0.5)
> cm

    FALSE TRUE
  0    57    7
  1    10   26
>
```

logistic regression classification model training set Rplot

logistic regression classification model test set Rplot

2. K-Nearest Neighbors (K-NN) :

formula
library(class)
y_pred = knn(train = training_set[, -3],
test = test_set[, -3],
cl = training_set[, 3],
k = 5,
prob = TRUE)

pros:Simple to understand, fast and efficient
cons:Need to choose the number of neighbours k

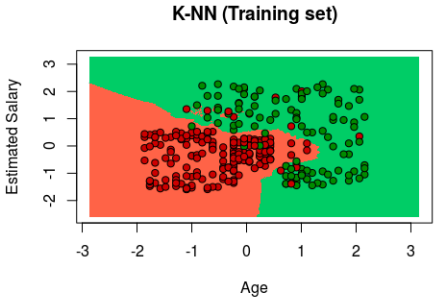
K-Nearest Neighbors (K-NN) classification model variables :

Data	
dataset	400 obs. of 3 variables
grid_set	376707 obs. of 2 variables
set	100 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 59 6 5 30
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
X1	num [1:597] -2.91 -2.9 -2.89 -2.88 -2.87 ...
X2	num [1:631] -2.67 -2.66 -2.65 -2.64 -2.63 ...
y_grid	Large factor (376707 elements, 1.5 MB)
y_pred	Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 2 1 1 ...

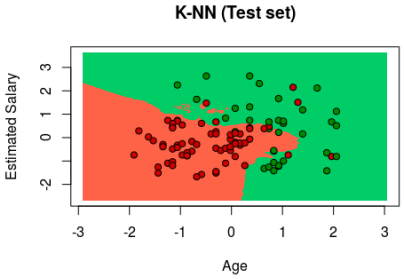
K-Nearest Neighbors (K-NN) classification model predict ouput

```
> points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red
3'))
> y_pred
  [1] 0 0 0 0 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1
 [31] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 1
 [61] 0 0 0 1 1 0 1 1 1 1 1 1 1 0 0 0 1 0 0 1 0 1 0 1 0 1 1 0 0 1
 [91] 1 0 1 0 1 1 1 1 0 1
attr(,"prob")
  [1] 1.0 1.0 1.0 1.0 1.0 0.8 0.8 1.0 0.6 1.0 1.0 1.0 0.8 1.0 1.0
 [16] 1.0 1.0 1.0 1.0 1.0 0.6 1.0 1.0 0.8 1.0 0.8 1.0 1.0 0.8 1.0
 [31] 1.0 0.8 0.8 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.8 1.0 1.0 1.0 0.8
 [46] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.8 0.8
 [61] 0.8 1.0 0.6 1.0 0.6 1.0 0.8 0.8 1.0 0.8 0.8 0.8 0.8 0.8 0.6
 [76] 1.0 0.8 1.0 1.0 1.0 1.0 0.8 1.0 1.0 0.8 0.8 0.8 0.6 0.8 1.0
 [91] 0.8 0.8 0.8 0.8 1.0 0.6 1.0 0.8 1.0 1.0
Levels: 0 1
> cm
   y_pred
     0  1
  0 59  5
  1  6 30
>
```

K-Nearest Neighbors (K-NN) classification model training set Rplot



K-Nearest Neighbors (K-NN) classification model test set Rplot



3. Support Vector Machine (SVM):

formula :

```
library(e1071)
classifier = svm(formula = Purchased ~ .,
                 data = training_set,
                 type = 'C-classification',
                 kernel = 'linear')
```

pros:Performant, not biased by outliers, not sensitive to overfitting

cons:Not appropriate for non linear problems, not the best choice for large number of features

Support Vector Machine (SVM) classification model variables output:

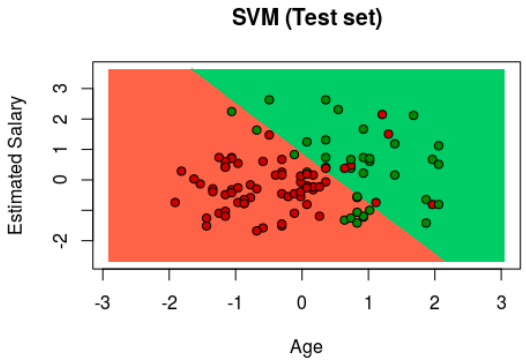
Data	
classifier	List of 30
dataset	400 obs. of 3 variables
grid_set	376707 obs. of 2 variables
set	100 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 57 13 7 23
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
X1	num [1:597] -2.91 -2.9 -2.89 -2.88 -2.87 ...
X2	num [1:631] -2.67 -2.66 -2.65 -2.64 -2.63 ...
y_grid	Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
y_pred	Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...

Support Vector Machine (SVM) classification model variables output:

```
> y_pred = predict(classifier, newdata = test_set[-3])
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 48  52  66  69  74  75  82  84  85  86  87  89 103 104 107 108
  0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 193 199 200 208 213 224 226 228 229 230 234 236 237 239 241
  0   0   0   0   1   1   1   0   1   0   1   1   1   0   1   1
255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324
  1   0   1   1   1   1   1   0   1   1   1   0   1   0   0   0
326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383
  0   1   0   1   0   1   1   0   1   1   1   0   1   0   1   1
389 392 395 400
  0   0   0   0
Levels: 0 1
> cm = table(test_set[, 3], y_pred)
> cm
   y_pred
     0  1
  0 57  7
  1 13 23
```

Support Vector Machine (SVM) classification model training set Rplot

Support Vector Machine (SVM) classification model test set Rplot



4. Kernel SVM:

formula :

```
library(e1071)
classifier = svm(formula = Purchased ~ .,
                 data = training_set,
                 type = 'C-classification',
                 kernel = 'radial')
```

pros:High performance on nonlinear problems, not biased by outliers, not sensitive to overfitting

cons:Not the best choice for large number of features, more complex

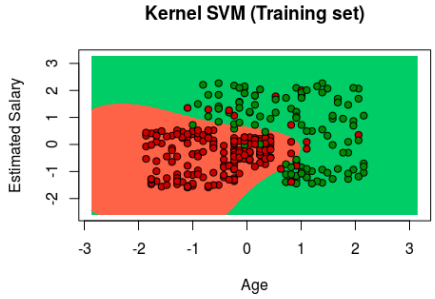
Kernel SVM classification model variable output

Data	
classifier	List of 30
dataset	400 obs. of 3 variables
grid_set	376707 obs. of 2 variables
set	100 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 58 4 6 32
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
X1	num [1:597] -2.91 -2.9 -2.89 -2.88 -2.87 ...
X2	num [1:631] -2.67 -2.66 -2.65 -2.64 -2.63 ...
y_grid	Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
y_pred	Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 2 1 1 ...

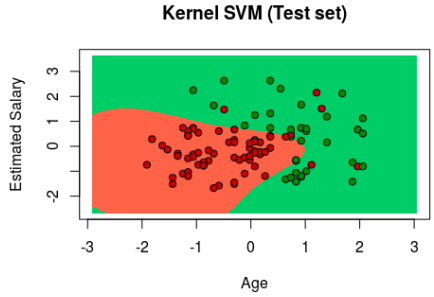
Kernel SVM classification model predict ouput

```
> y_pred = predict(classifier, newdata = test_set[-3])
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46
  0   0   0   0   0   1   1   1   0   0   1   0   0   0   0   0
 48  52  66  69  74  75  82  84  85  86  87  89 103 104 107 108
  0   0   0   0   1   0   0   0   0   1   0   0   0   1   0   0
109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
176 193 199 200 208 213 224 226 228 229 230 234 236 237 239 241
  0   0   0   0   1   1   1   0   1   0   0   1   1   0   1   1
255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324
  1   0   1   1   1   1   1   1   1   0   1   0   1   0   0   1
326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383
  0   1   0   1   0   1   1   0   0   1   1   0   1   0   1   1
389 392 395 400
  1   1   0   1
Levels: 0 1
> cm = table(test_set[, 3], y_pred)
> cm
   y_pred
     0  1
  0 58  6
  1  4 32
> library(ElemStatLearn)
```

Kernel SVM classification model training set Rplot



Kernel SVM classification model test set Rplot



5. Naive Bayes:

formula:

```
library(e1071)
classifier = naiveBayes(x = training_set[-3],
                        y = training_set$Purchased)
```

pros:Efficient, not biased by outliers, works on nonlinear problems, probabilistic approach

cons:Based on the assumption that features have same statistical relevance

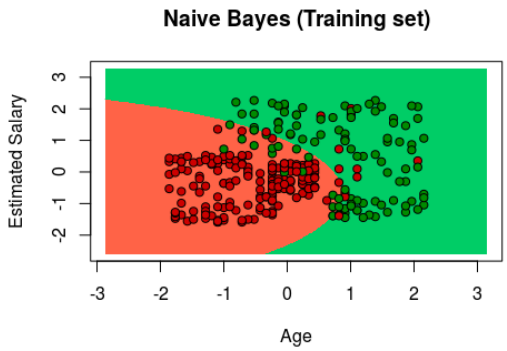
Naive Bayes classification model variable output

Data	
classifier	List of 5
dataset	400 obs. of 3 variables
grid_set	376707 obs. of 2 variables
set	100 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 57 7 7 29
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
X1	num [1:597] -2.91 -2.9 -2.89 -2.88 -2.87 ...
X2	num [1:631] -2.67 -2.66 -2.65 -2.64 -2.63 ...
y_grid	Large factor (376707 elements, 1.5 MB)
y_pred	Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...

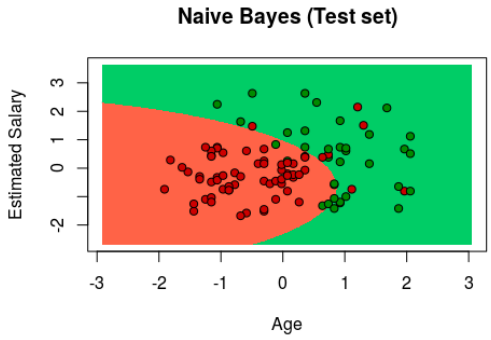
Naive Bayes classification model predict output

```
> y_pred = predict(classifier, newdata = test_set[-3])
> y_pred
  [1] 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1
 [31] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 0 1
 [61] 1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 0 0 1 0 1 0 1 0 1 1 0 0 1
 [91] 1 0 1 0 1 1 1 1 0 1
Levels: 0 1
> cm = table(test_set[, 3], y_pred)
> cm
   y_pred
     0  1
  0 57  7
  1  7 29
```

Naive Bayes classification model training set Rplot



Naive Bayes classification model test set Rplot



6. Decision Tree Classification:

formula

library(rpart)

classifier = rpart(formula = Purchased ~ .,

data = training_set)

pros:Interpretability, no need for feature scaling, works on both linear / nonlinear problems

cons:Poor results on too small datasets, overfitting can easily occur.

Decision Tree Classification model variable output

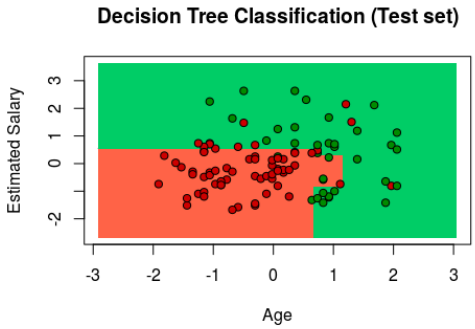
Data	
classifier	List of 14
dataset	400 obs. of 3 variables
grid_set	376707 obs. of 2 variables
set	100 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 53 6 11 30
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
X1	num [1:597] -2.91 -2.9 -2.89 -2.88 -2.87 ...
X2	num [1:631] -2.67 -2.66 -2.65 -2.64 -2.63 ...
y_grid	Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
y_pred	Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 2 1 1 ...

Decision Tree Classification model predict output

```
> y_pred = predict(classifier, newdata = test_set[-3], type = 'clas
s')
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46
  0   0   0   0   0   0   1   1   0   0   1   0   1   0   0   0
 48  52  66  69  74  75  82  84  85  86  87  89 103 104 107 108
  0   0   0   0   1   0   0   1   0   1   0   0   1   1   0   1
109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175
  1   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0
176 193 199 200 208 213 224 226 228 229 230 234 236 237 239 241
  0   0   0   0   1   1   1   0   1   0   0   1   1   0   1   1
255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324
  0   0   1   1   1   1   1   1   1   0   0   0   1   0   0   1
326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383
  0   1   0   1   0   1   1   0   0   1   1   0   1   0   1   1
389 392 395 400
  1   1   0   1
```

Decision Tree Classification model training set Rplot

Decision Tree Classification model test set Rplot



7. Random Forest Classification:

formula :

```
library(randomForest)
set.seed(123)
classifier = randomForest(x = training_set[-3],
                          y = training_set$Purchased,
                          ntree = 500)
```

pros:Powerful and accurate, good performance on many problems including non linear.
cons:No interpretability, overfitting can easily occur, need to choose the number of trees.

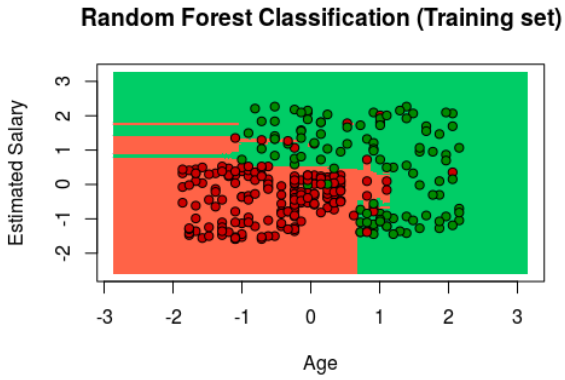
Random Forest Classification model variable output

Data	
classifier	List of 18
dataset	400 obs. of 3 variables
grid_set	376707 obs. of 2 variables
set	100 obs. of 3 variables
test_set	100 obs. of 3 variables
training_set	300 obs. of 3 variables
Values	
cm	'table' int [1:2, 1:2] 56 7 8 29
split	logi [1:400] TRUE FALSE TRUE FALSE FALSE TRUE ...
X1	num [1:597] -2.91 -2.9 -2.89 -2.88 -2.87 ...
X2	num [1:631] -2.67 -2.66 -2.65 -2.64 -2.63 ...
y_grid	Large factor (376707 elements, 25.6 MB)
y_pred	Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 2 1 1 ...

Random Forest Classification model predict output

```
> y_pred = predict(classifier, newdata = test_set[-3], type = 'clas
s')
> y_pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46
  0   0   0   0   0   0   1   1   0   0   1   0   1   0   0   0
 48  52  66  69  74  75  82  84  85  86  87  89 103 104 107 108
  0   0   0   0   1   0   0   1   0   1   0   0   1   1   0   1
109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175
  1   0   0   0   0   0   0   0   0   0   0   0   1   0   0   0
176 193 199 200 208 213 224 226 228 229 230 234 236 237 239 241
  0   0   0   0   1   1   1   0   1   0   0   1   1   0   1   1
255 264 265 266 273 274 281 286 292 299 302 305 307 310 316 324
  0   0   1   1   1   1   1   1   1   0   0   0   1   0   0   1
326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383
  0   1   0   1   0   1   1   0   0   1   1   0   1   0   1   1
389 392 395 400
  1   1   0   1
```

Random Forest Classification model training set Rplot



Random Forest Classification model test set Rplot