

Naive-Bayes

jerf

8/7/2021

Naive-Bayes

```
dataset = read.csv("Social_Network_Ads.csv")
dataset = dataset[,3:5]

dataset$Purchased = factor(dataset$Purchased, levels = c(0,1))
```

Dividir dataset en conjunto de entrenamiento y conjunto de test

```
library(caTools)

## Warning: package 'caTools' was built under R version 4.0.5
set.seed(123)
split = sample.split(dataset$Purchased, SplitRatio = 0.75)

training_set = subset(dataset, split == TRUE)
training_set
```

```
##      Age EstimatedSalary Purchased
## 1      19          19000        0
## 3      26          43000        0
## 6      27          58000        0
## 7      27          84000        0
## 8      32         150000        1
## 10     35          65000        0
## 11     26          80000        0
## 13     20          86000        0
## 14     32          18000        0
## 15     18          82000        0
## 16     29          80000        0
## 17     47          25000        1
## 21     45          22000        1
## 23     48          41000        1
## 24     45          22000        1
## 25     46          23000        1
## 26     47          20000        1
## 27     49          28000        1
## 28     47          30000        1
## 30     31          18000        0
## 31     31          74000        0
## 33     21          16000        0
```

## 36	35	27000	0
## 37	33	28000	0
## 39	26	72000	0
## 40	27	31000	0
## 41	27	17000	0
## 42	33	51000	0
## 43	35	108000	0
## 44	30	15000	0
## 47	25	79000	0
## 49	30	135000	1
## 50	31	89000	0
## 51	24	32000	0
## 53	29	83000	0
## 54	35	23000	0
## 55	27	58000	0
## 56	24	55000	0
## 57	23	48000	0
## 58	28	79000	0
## 59	22	18000	0
## 60	32	117000	0
## 61	27	20000	0
## 62	25	87000	0
## 63	23	66000	0
## 64	32	120000	1
## 65	59	83000	0
## 67	24	19000	0
## 68	23	82000	0
## 70	31	68000	0
## 71	25	80000	0
## 72	24	27000	0
## 73	20	23000	0
## 76	34	112000	1
## 77	18	52000	0
## 78	22	27000	0
## 79	28	87000	0
## 80	26	17000	0
## 81	30	80000	0
## 83	20	49000	0
## 88	28	85000	0
## 90	35	50000	0
## 91	22	81000	0
## 92	30	116000	0
## 93	26	15000	0
## 94	29	28000	0
## 95	29	83000	0
## 96	35	44000	0
## 97	35	25000	0
## 98	28	123000	1
## 99	35	73000	0
## 100	28	37000	0
## 101	27	88000	0
## 102	28	59000	0
## 105	19	21000	0
## 106	21	72000	0

## 110	38	80000	0
## 111	39	71000	0
## 112	37	71000	0
## 113	38	61000	0
## 114	37	55000	0
## 115	42	80000	0
## 116	40	57000	0
## 118	36	52000	0
## 119	40	59000	0
## 120	41	59000	0
## 121	36	75000	0
## 122	37	72000	0
## 123	40	75000	0
## 125	41	51000	0
## 128	26	32000	0
## 129	30	17000	0
## 130	26	84000	0
## 132	33	31000	0
## 133	30	87000	0
## 135	28	55000	0
## 136	23	63000	0
## 137	20	82000	0
## 138	30	107000	1
## 140	19	25000	0
## 141	19	85000	0
## 142	18	68000	0
## 143	35	59000	0
## 144	30	89000	0
## 145	34	25000	0
## 146	24	89000	0
## 147	27	96000	1
## 149	29	61000	0
## 150	20	74000	0
## 151	26	15000	0
## 152	41	45000	0
## 153	31	76000	0
## 155	40	47000	0
## 157	46	59000	0
## 158	29	75000	0
## 160	32	135000	1
## 161	32	100000	1
## 164	35	38000	0
## 165	33	69000	0
## 166	18	86000	0
## 167	22	55000	0
## 168	35	71000	0
## 169	29	148000	1
## 171	21	88000	0
## 172	34	115000	0
## 173	26	118000	0
## 174	34	43000	0
## 177	35	47000	0
## 178	25	22000	0
## 179	24	23000	0

## 180	31	34000	0
## 181	26	16000	0
## 182	31	71000	0
## 183	32	117000	1
## 184	33	43000	0
## 185	33	60000	0
## 186	31	66000	0
## 187	20	82000	0
## 188	33	41000	0
## 189	35	72000	0
## 190	28	32000	0
## 191	24	84000	0
## 192	19	26000	0
## 194	19	70000	0
## 195	28	89000	0
## 196	34	43000	0
## 197	30	79000	0
## 198	20	36000	0
## 201	35	39000	0
## 202	49	74000	0
## 203	39	134000	1
## 204	41	71000	0
## 205	58	101000	1
## 206	47	47000	0
## 207	55	130000	1
## 209	40	142000	1
## 210	46	22000	0
## 211	48	96000	1
## 212	52	150000	1
## 214	35	58000	0
## 215	47	43000	0
## 216	60	108000	1
## 217	49	65000	0
## 218	40	78000	0
## 219	46	96000	0
## 220	59	143000	1
## 221	41	80000	0
## 222	35	91000	1
## 223	37	144000	1
## 225	35	60000	0
## 227	36	126000	1
## 231	35	147000	1
## 232	39	42000	0
## 233	40	107000	1
## 235	38	112000	0
## 238	37	80000	0
## 240	53	143000	1
## 242	38	59000	0
## 243	50	88000	1
## 244	56	104000	1
## 245	41	72000	0
## 246	51	146000	1
## 247	35	50000	0
## 248	57	122000	1

## 249	41	52000	0
## 250	35	97000	1
## 251	44	39000	0
## 252	37	52000	0
## 253	48	134000	1
## 254	37	146000	1
## 256	52	90000	1
## 257	41	72000	0
## 258	40	57000	0
## 259	58	95000	1
## 260	45	131000	1
## 261	35	77000	0
## 262	36	144000	1
## 263	55	125000	1
## 267	40	75000	0
## 268	37	74000	0
## 269	47	144000	1
## 270	40	61000	0
## 271	43	133000	0
## 272	59	76000	1
## 275	57	26000	1
## 276	57	74000	1
## 277	38	71000	0
## 278	49	88000	1
## 279	52	38000	1
## 280	50	36000	1
## 282	35	61000	0
## 283	37	70000	1
## 284	52	21000	1
## 285	48	141000	0
## 287	37	62000	0
## 288	48	138000	1
## 289	41	79000	0
## 290	37	78000	1
## 291	39	134000	1
## 293	55	39000	1
## 294	37	77000	0
## 295	35	57000	0
## 296	36	63000	0
## 297	42	73000	1
## 298	43	112000	1
## 300	46	117000	1
## 301	58	38000	1
## 303	37	137000	1
## 304	37	79000	1
## 306	42	54000	0
## 308	47	113000	1
## 309	36	125000	1
## 311	42	70000	0
## 312	39	96000	1
## 313	38	50000	0
## 314	49	141000	1
## 315	39	79000	0
## 317	54	104000	1

## 318	35	55000	0
## 319	45	32000	1
## 320	36	60000	0
## 321	52	138000	1
## 322	53	82000	1
## 323	41	52000	0
## 325	48	131000	1
## 327	41	72000	0
## 328	42	75000	0
## 329	36	118000	1
## 330	47	107000	1
## 331	38	51000	0
## 333	42	65000	0
## 334	40	65000	0
## 335	57	60000	1
## 336	36	54000	0
## 337	58	144000	1
## 338	35	79000	0
## 340	39	122000	1
## 342	35	75000	0
## 344	47	51000	1
## 345	47	105000	1
## 346	41	63000	0
## 348	54	108000	1
## 349	39	77000	0
## 350	38	61000	0
## 351	38	113000	1
## 352	37	75000	0
## 354	37	57000	0
## 355	36	99000	1
## 356	60	34000	1
## 357	54	70000	1
## 358	41	72000	0
## 359	40	71000	1
## 360	42	54000	0
## 361	43	129000	1
## 362	53	34000	1
## 365	42	104000	1
## 366	59	29000	1
## 370	54	26000	1
## 371	60	46000	1
## 374	59	130000	1
## 375	37	80000	0
## 376	46	32000	1
## 377	46	74000	0
## 378	42	53000	0
## 379	41	87000	1
## 381	42	64000	0
## 382	48	33000	1
## 384	49	28000	1
## 385	57	33000	1
## 386	56	60000	1
## 387	49	39000	1
## 388	39	71000	0

```

## 390 48          35000      1
## 391 48          33000      1
## 393 45          45000      1
## 394 60          42000      1
## 396 46          41000      1
## 397 51          23000      1
## 398 50          20000      1
## 399 36          33000      0

testing_set = subset(dataset, split == FALSE)
testing_set

```

	Age	EstimatedSalary	Purchased
## 2	35	20000	0
## 4	27	57000	0
## 5	19	76000	0
## 9	25	33000	0
## 12	26	52000	0
## 18	45	26000	1
## 19	46	28000	1
## 20	48	29000	1
## 22	47	49000	1
## 29	29	43000	0
## 32	27	137000	1
## 34	28	44000	0
## 35	27	90000	0
## 38	30	49000	0
## 45	28	84000	0
## 46	23	20000	0
## 48	27	54000	0
## 52	18	44000	0
## 66	24	58000	0
## 69	22	63000	0
## 74	33	113000	0
## 75	32	18000	0
## 82	39	42000	0
## 84	35	88000	0
## 85	30	62000	0
## 86	31	118000	1
## 87	24	55000	0
## 89	26	81000	0
## 103	32	86000	0
## 104	33	149000	1
## 107	26	35000	0
## 108	27	89000	0
## 109	26	86000	0
## 117	35	75000	0
## 124	35	53000	0
## 126	39	61000	0
## 127	42	65000	0
## 131	31	58000	0
## 134	21	68000	0
## 139	28	59000	0
## 148	41	30000	0
## 154	36	50000	0

## 156	31	15000	0
## 159	26	30000	0
## 162	25	90000	0
## 163	37	33000	0
## 170	29	47000	0
## 175	34	72000	0
## 176	23	28000	0
## 193	29	43000	0
## 199	26	80000	0
## 200	35	22000	0
## 208	52	114000	0
## 213	59	42000	0
## 224	60	102000	1
## 226	37	53000	0
## 228	56	133000	1
## 229	40	72000	0
## 230	42	80000	1
## 234	49	86000	1
## 236	46	79000	1
## 237	40	57000	0
## 239	46	82000	0
## 241	42	149000	1
## 255	50	44000	0
## 264	35	72000	0
## 265	48	90000	1
## 266	42	108000	1
## 273	60	42000	1
## 274	39	106000	1
## 281	59	88000	1
## 286	37	93000	1
## 292	49	89000	1
## 299	45	79000	0
## 302	48	74000	1
## 305	40	60000	0
## 307	51	134000	0
## 310	38	50000	0
## 316	39	75000	1
## 324	48	30000	1
## 326	41	60000	0
## 332	48	119000	1
## 339	38	55000	0
## 341	53	104000	1
## 343	38	65000	0
## 347	53	72000	1
## 353	42	90000	1
## 363	47	50000	1
## 364	42	79000	0
## 367	58	47000	1
## 368	46	88000	1
## 369	38	71000	0
## 372	60	83000	1
## 373	39	73000	0
## 380	58	23000	1
## 383	44	139000	1

```

## 389 47          34000      1
## 392 47          23000      1
## 395 39          59000      0
## 400 49          36000      1

```

Escalado de datos

Standardisation

$$x_{stand} = \frac{x - mean(x)}{sd(x)}$$

NO es necesario, el escalado es necesario cuando el algoritmo requiere distancias

Ajustar el modelo de Regresión Logística con el conjunto de entrenamiento y hacer las predicciones con el conjunto testing

```

library(rpart)

## Warning: package 'rpart' was built under R version 4.0.5
classifier = rpart(formula = Purchased ~.,
                     data = training_set)

y_pred = predict(classifier, newdata = testing_set[,-3], type = "class")
y_pred

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

```

Comparar uno a uno los resultados predichos con los esperados no es una buena técnica por lo que se construye la matriz de confusión

```

cm = table(testing_set[, 3], y_pred)
cm

##     y_pred
##     0 1
## 0 56 8
## 1 6 30

```

La diagonal principal es la cantidad de datos que son predichos correctamente. La diagonal secundaria son los fallos.

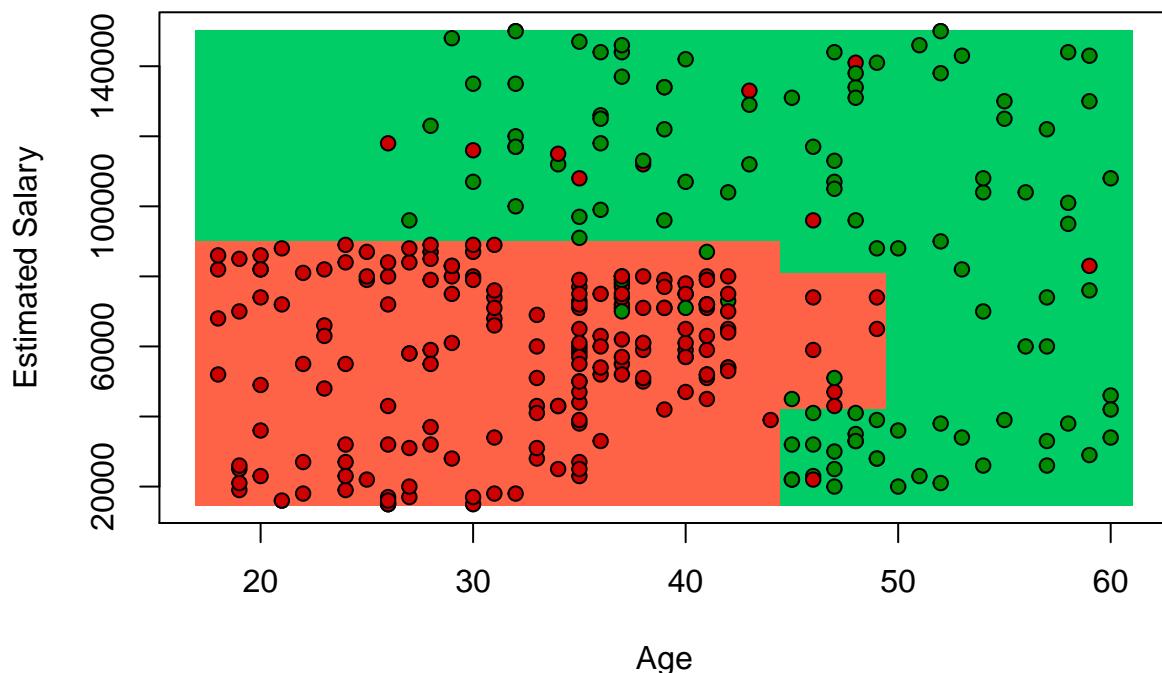
Visualización del conjunto de entrenamiento

```

library(ElemStatLearn)
set = training_set
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 500)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
y_grid = predict(classifier,
                 newdata = grid_set, type = "class")
plot(set[, -3],
     main = 'Arboles de decision (Training set)',
     xlab = 'Age', ylab = 'Estimated Salary',
     xlim = range(X1), ylim = range(X2))
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add = TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'springgreen3', 'tomato'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

```

Arboles de decision (Training set)

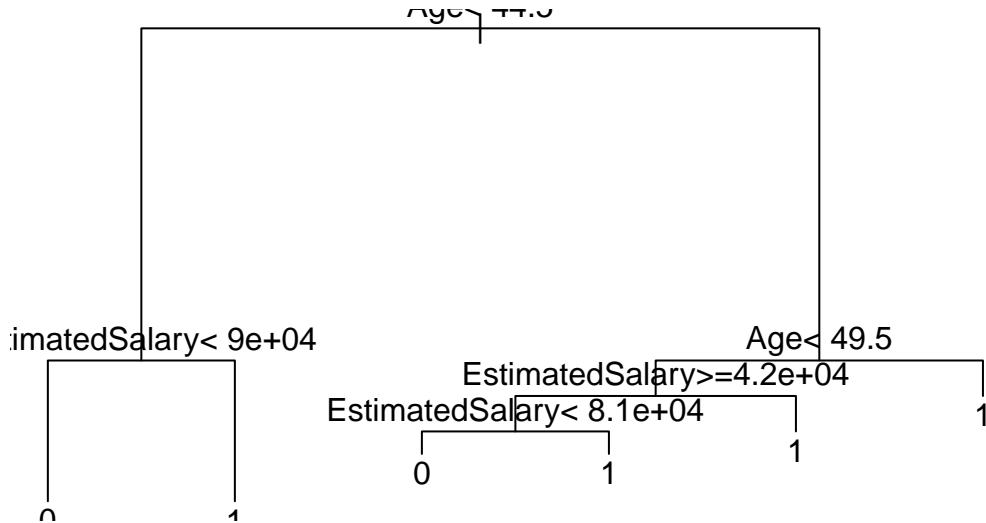


representacion del arbol de clasificacion

```

plot(classifier)
text(classifier)

```



Visualising the Test set results

```

library(ElemStatLearn)
set = testing_set
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 500)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
y_grid = predict(classifier,
                 newdata = grid_set, type="class")
plot(set[, -3],
     main = 'Arboles de decision (Test set)',
     xlab = 'Age', ylab = 'Estimated Salary',
     xlim = range(X1), ylim = range(X2))
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add = TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'springgreen3', 'tomato'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))
  
```

Arboles de decisión (Test set)

