Project 2 - Naive Baye's

Ola Tranum Arnegaard - 10/08/2019

Naive Baye's Implementation: *Please implement the Naive Baye's algorithm and submit the deliverables as specified in the project rubric document under Modules.*

Intro

Use Naive Baye's to predict if `isLegendary` is **True** for a Pokemon given its other features.

Preprocessing

- All Pokemon are randomized (rows)
- Only included categorical features
- Catch_Rate_Cat and Total_Cat are created from numerical to categorical variables arranged with respect to their given max and min values.
- Removed categorical features that are clearly dependent on eachother

This is what the data looks like:

```
head(Pokemon,3)
```

```
##     isLegendary      Name   Type_1 Color hasGender Egg_Group_1
## 290       False   Nincada      Bug  Grey      True         Bug
## 701       False  Hawlucha Fighting Green      True  Human-Like
## 148       False Dragonair   Dragon  Blue      True     Water_1
##     hasMegaEvolution       Body_Style Catch_Rate_Cat Total_Cat
## 290            False        insectoid      (213,255] (180,270]
## 701            False bipedal_tailless       (87,129] (450,540]
## 148            False  serpentine_body         (3,45] (360,450]
```

Implementation

Create Model

```
train_data <- Pokemon[1:675,-c(1,2)] #Train data (no names or labels)
train_labels <- Pokemon[1:675,1] #Corresponding prediction labels
Pokemon_NB_model <- naiveBayes(train_data, train_labels) #Create model
```

Prediction

```r
test_data <- Pokemon[676:721,-c(1,2)] #Corresponding to the training data
test_labels <- Pokemon[676:721,1] #Actual labels
Pokemon_NB_prediction <- predict(Pokemon_NB_model, test_data) #Predicted labels

CrossTable(Pokemon_NB_prediction, test_labels,
           prop.chisq = FALSE, prop.t = FALSE, dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  46
##
##
##              | actual
##    predicted |     False |      True | Row Total |
## -------------|-----------|-----------|-----------|
##        False |        40 |         0 |        40 |
##              |     1.000 |     0.000 |     0.870 |
##              |     1.000 |     0.000 |           |
## -------------|-----------|-----------|-----------|
##         True |         0 |         6 |         6 |
##              |     0.000 |     1.000 |     0.130 |
##              |     0.000 |     1.000 |           |
## -------------|-----------|-----------|-----------|
## Column Total |        40 |         6 |        46 |
##              |     0.870 |     0.130 |           |
## -------------|-----------|-----------|-----------|
##
##
```

Results

Because the dataset is randomized, the results are different from every compilation. However, in most examples, the predictions are close to 100% correct. In one compilation, we predicted 5 legendary Pokemon correctly; however, we also predicted one to incorrectly be legendary (total of 6). In conclusion, we predicted 83.3% correctly for `isLegendary` **True** and correspondingly 97.6% `isLegendary` correctly to be **False**.

Prediction model

- Our prediction picks **True** or **False** for `isLegendary` based on which has highest `posterior probability`.
- This probability is calculated as follows

$$P(\texttt{isLegendary}|feat1, feat2, ...featN) = \frac{P(feat1, feat2, ...featN|\texttt{isLegendary})P(\texttt{isLegendary})}{P(feat1, feat2, ...featN)}$$

- Our *prior and conditional probabilities* as well as our *likelihood* of `isLegendary` are calculated in the model which we use in our prediction. This can be seen below:

```
Pokemon_NB_model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = train_data, y = train_labels)
##
## A-priori probabilities:
## train_labels
##      False       True
## 0.94074074 0.05925926
##
## Conditional probabilities:
##             Type_1
## train_labels         Bug        Dark      Dragon    Electric       Fairy
##        False 0.091338583 0.036220472 0.026771654 0.045669291 0.025196850
##        True  0.000000000 0.050000000 0.125000000 0.050000000 0.025000000
##             Type_1
## train_labels    Fighting        Fire      Flying       Ghost       Grass
##        False 0.039370079 0.064566929 0.003149606 0.033070866 0.096062992
##        True  0.000000000 0.125000000 0.025000000 0.025000000 0.050000000
##             Type_1
## train_labels      Ground         Ice      Normal      Poison     Psychic
##        False 0.039370079 0.029921260 0.137007874 0.044094488 0.058267717
##        True  0.050000000 0.025000000 0.050000000 0.000000000 0.200000000
##             Type_1
## train_labels        Rock       Steel       Water
##        False 0.058267717 0.023622047 0.148031496
##        True  0.025000000 0.100000000 0.075000000
##
##             Color
## train_labels      Black       Blue      Brown      Green       Grey
##        False 0.04251969 0.18425197 0.15118110 0.10866142 0.09921260
##        True  0.05000000 0.17500000 0.12500000 0.12500000 0.07500000
##             Color
## train_labels       Pink     Purple        Red      White     Yellow
##        False 0.05984252 0.08976378 0.10551181 0.07244094 0.08661417
##        True  0.02500000 0.07500000 0.12500000 0.07500000 0.15000000
##
##             hasGender
## train_labels      False       True
##        False 0.05354331 0.94645669
##        True  0.87500000 0.12500000
##
##             Egg_Group_1
## train_labels   Amorphous         Bug       Ditto      Dragon       Fairy
##        False 0.062992126 0.094488189 0.001574803 0.012598425 0.045669291
##        True  0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##             Egg_Group_1
## train_labels       Field      Flying       Grass  Human-Like     Mineral
```

```
##        False 0.251968504 0.067716535 0.039370079 0.056692913 0.064566929
##        True  0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##             Egg_Group_1
## train_labels     Monster Undiscovered     Water_1     Water_2     Water_3
##        False 0.110236220  0.042519685 0.108661417 0.018897638 0.022047244
##        True  0.000000000  1.000000000 0.000000000 0.000000000 0.000000000
##
##             hasMegaEvolution
## train_labels      False       True
##        False 0.93543307 0.06456693
##        True  0.90000000 0.10000000
##
##             Body_Style
## train_labels bipedal_tailed bipedal_tailless four_wings  head_arms
##        False     0.22519685       0.15433071 0.02834646 0.05039370
##        True      0.17500000       0.17500000 0.00000000 0.05000000
##             Body_Style
## train_labels  head_base  head_legs  head_only  insectoid multiple_bodies
##        False 0.04566929 0.02519685 0.04881890 0.04251969      0.02204724
##        True  0.00000000 0.00000000 0.02500000 0.00000000      0.00000000
##             Body_Style
## train_labels  quadruped serpentine_body several_limbs  two_wings
##        False 0.17952756      0.03779528    0.01889764 0.08188976
##        True  0.27500000      0.07500000    0.00000000 0.20000000
##             Body_Style
## train_labels  with_fins
##        False 0.03937008
##        True  0.02500000
##
##             Catch_Rate_Cat
## train_labels     (3,45]    (45,87]   (87,129]  (129,171]  (171,213]
##        False 0.38118022 0.17224880 0.14513557 0.02711324 0.14992026
##        True  1.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##             Catch_Rate_Cat
## train_labels  (213,255]
##        False 0.12440191
##        True  0.00000000
##
##             Total_Cat
## train_labels   (180,270]   (270,360]   (360,450]   (450,540]   (540,630]
##        False 0.083596215 0.299684543 0.205047319 0.380126183 0.029968454
##        True  0.000000000 0.000000000 0.000000000 0.000000000 0.650000000
##             Total_Cat
## train_labels   (630,720]
##        False 0.001577287
##        True  0.350000000
```

Extras

For those interested, it is also interesting to see which Pokemon that were actually and mistakenly predicted to `isLegendary` from the test data.

```r
Names = as.data.frame(Pokemon[676:721,2]) #Select the test data names
Names$actually = test_labels #Actual boolean isLegendary values
Names$predicted = as.data.frame(Pokemon_NB_prediction) #Predicted isLegendary
```

- Actually isLegendary:

```r
Names[Names$actually=='True',1] #Names that isLegendary
```

```
## [1] Reshiram  Regice    Zekrom    Thundurus Terrakion Diancie
## 721 Levels: Abomasnow Abra Absol Accelgor Aegislash Aerodactyl ... Zygarde
```

- Predicted isLegendary:

```r
Names[Names$predicted=='True',1] #Names predicted isLegendary
```

```
## [1] Reshiram  Regice    Zekrom    Thundurus Terrakion Diancie
## 721 Levels: Abomasnow Abra Absol Accelgor Aegislash Aerodactyl ... Zygarde
```