Descubrimiento del Conocimiento usando herramientas de Big Data Módulo 2

Marco Andrés Vázquez Hernández

Práctica de Regresion Logística.

Septiembre de 2018

Instituto Politécnico Nacional

Descripción

1- Encontrar las variables que generen una mayor exactitud (accuracy) del conjunto de datos anterior. 2- Documentar el procedmiento. 3- Realizar el preprocesamiento de datos en caso de ser necesario.

Integración de los archivos

Se cargaron los datos y se separaron las variables: Obs.No se elimina y se separa "Buy" dado que es la variable que queremos explicar.

```
from sklearn import linear_model
from sklearn import preprocessing
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import scipy.stats as stats
import itertools
dataPeople = pd.read_csv("clientes.csv")
lst = dataPeople.columns
no_lst = ['Obs No.','Buy']
```

Se hicieron dos loops for: 1.- El primero corre en la variable "j" de 0 hasta el máximo de variables en lst -1 ya que hace uso de la función itertools.combinations que arroja una lista de todas las combinaciónes de tamaño j.

2.- Después se corre y compara la accuracy de cada una de las combinaciones de tamaño j y se guarda la combinación con mayor accuracy.

```
aux_acc=0
print(range(len(lst)))
for j in range(len(lst)-1):
    print (j)
    combinaciones = list(itertools.combinations([c for c in dataPeople.columns if c not in no_lst], j+1
    for i in combinaciones:
        train_features = dataPeople[[c for c in dataPeople.columns if c in i]]
        print(i)
        log_model = linear_model.LogisticRegression(solver='lbfgs')
        log_model.fit(X = train_features ,
                 y = dataPeople["Buy"])
        preds = log_model.predict(X= train_features)
        tablePred=pd.crosstab(preds,dataPeople["Buy"])
        accuracy  = log_model.score(X=train_features,
                                y=dataPeople["Buy"])
        print(accuracy)
        print(aux_acc)
        if accuracy>aux_acc:
            aux_acc=accuracy
            train_features_best = train_features
```

Finalmente se sacan los resultados de la combinación con mejor accuracy:

```
log_model = linear_model.LogisticRegression(solver='lbfgs')
log_model.fit(X = train_features_best, y = dataPeople["Buy"])
accuracy  = log_model.score(X=train_features_best, y=dataPeople["Buy"])
```

```
print("Accuracy")
print(accuracy)
print("Intercección ")
print(model.intercept)
print("Coeficientes")
print(model.coef)
print("Matriz de confusion")
aux_preds = log_model.predict(X= train_features)
tablePred=pd.crosstab(aux_preds,dataPeople["Buy"])
print (pd.crosstab(aux_preds,dataPeople["Buy"]))
```

Quedando los resultados:

```
[673 rows x 7 columns]
Accuracy
0.9479940564635958
Coeficientes
[[ 1.56115091e-04  9.70454559e-01 -8.62175603e-02  2.24908523e-01
  -8.33599946e-01  5.20588479e-01  1.30758428e+00]]
Matriz de confusion
Buy      0    1
row_0
0      529   16
1       19  109
PS C:\Users\marco\IPN_BigData\Modulo2\Practica2> python prac2-2.py
C:\Users\marco\AppData\Local\Programs\Python\Python37\lib\site-packages\sklearn\externals\joblib\externa
e\cloudpickle.py:47: DeprecationWarning: the imp module is deprecated in favour of importlib; see the mo
ntation for alternative uses
  import imp
Index(['Income', 'Is Married', 'Has College', 'Is Professional', 'Is Retired',
       'Own', 'White'],
      dtype='object')
Accuracy
0.9479940564635958
Coeficientes
[[ 1.56115091e-04  9.70454559e-01 -8.62175603e-02  2.24908523e-01
  -8.33599946e-01  5.20588479e-01  1.30758428e+00]]
Matriz de confusion
Buy      0    1
row_0
0      529   16
1       19  109
PS C:\Users\marco\IPN_BigData\Modulo2\Practica2>
```