Vivian Do

ADS502 Assignment 4

Data Science Using Python and R: Chapter 13 - Page 195: Questions #13, 14, 15, 16, & 17

For the following exercises, work with the clothing_sales_training and clothing_sales_test data sets.

13) Create a logistic regression model to predict whether or not a customer has a store credit card, based on whether they have a web account and the days between purchases. Obtain the summary of the model.

In [37]:
```
#suppress all warnings
import warnings
warnings.filterwarnings("ignore")

#import necessary libraries
import numpy as np
import pandas as pd
import statsmodels.api as sm
from scipy import stats
```

In [38]:
```
#import data sets
sales_train=pd.read_csv("clothing_sales_training.csv")
sales_test=pd.read_csv("clothing_sales_test.csv")
sales_train.head(5)
```

Out[38]:

	CC	Days	Web	Sales per Visit
0	0	333.0	0	184.230000
1	0	171.5	0	38.500000
2	0	213.0	0	150.326667
3	1	71.4	1	104.240000
4	1	145.0	0	782.080000

In [39]:
```
#separate predictor and target variables
X = pd.DataFrame(sales_train[['Web', 'Days']])
y = pd.DataFrame(sales_train[['CC']])

#add constant term for regression model
X = sm.add_constant(X)

#perform logistic regression
logreg01 = sm.Logit(y, X).fit()

#obtain model results
logreg01.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.655955
         Iterations 5
```

Out[39]:

Model:	Logit	Pseudo R-squared:	0.053
Dependent Variable:	CC	AIC:	1909.5825
Date:	2022-11-21 16:06	BIC:	1925.4226
No. Observations:	1451	Log-Likelihood:	-951.79
Df Model:	2	LL-Null:	-1004.9
Df Residuals:	1448	LLR p-value:	8.3668e-24
Converged:	1.0000	Scale:	1.0000
No. Iterations:	5.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
const	0.4962	0.0887	5.5968	0.0000	0.3224	0.6699
Web	1.2537	0.3307	3.7914	0.0001	0.6056	1.9018
Days	-0.0037	0.0004	-8.4491	0.0000	-0.0046	-0.0028

14) Are there any variables that should be removed from the model? If so, remove them and rerun the model.

Both predictors have a p-value of <=0.0001, so we will keep them in the model.

15) Write the descriptive form of the logistic regression model using the coefficients obtained from Question 1.

p(credit card) = [exp(0.4962 - 0.0037(Days) + 1.2537(Web)]/[1+exp(0.4962 - 0.0037(Days) + 1.2537(Web)]

16) Validate the model using the test data set.

In [40]:

```
#perform the same steps as above to the test set
#separate predictor and target variables
X_test = pd.DataFrame(sales_test[['Web', 'Days']])
y_test = pd.DataFrame(sales_test[['CC']])

#add constant term
X_test = sm.add_constant(X_test)

#perform logistic regression
logreg01_test = sm.Logit(y_test, X_test).fit()

#obtain model results
logreg01_test.summary2()
```

```
Optimization terminated successfully.
         Current function value: 0.656885
         Iterations 5
```

Out[40]:

Model:	Logit	Pseudo R-squared:	0.052
Dependent Variable:	CC	AIC:	1838.7104
Date:	2022-11-21 16:06	BIC:	1854.4324
No. Observations:	1395	Log-Likelihood:	-916.36
Df Model:	2	LL-Null:	-966.40
Df Residuals:	1392	LLR p-value:	1.8534e-22
Converged:	1.0000	Scale:	1.0000
No. Iterations:	5.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
const	0.4634	0.0873	5.3105	0.0000	0.2924	0.6345
Web	1.0973	0.2830	3.8780	0.0001	0.5427	1.6519
Days	-0.0035	0.0004	-8.2261	0.0000	-0.0043	-0.0026

17) Obtain the predicted values of the response variable for each record in the data set.

Predicted values from logistic regression models are probabilities between 0 and 1. In this case, it is the probability that a customer has a store credit card.

In [41]:
```
#save predictions
logreg01.predict(X)
```

Out[41]:
```
0       0.323777
1       0.465391
2       0.427447
3       0.815413
4       0.489860
          ...
1446    0.809628
1447    0.578054
1448    0.480615
1449    0.482463
1450    0.539094
Length: 1451, dtype: float64
```

Data Science Using Python and R: Chapter 6 - Page 93: Questions #19 & 20

Work with the adult_ch6_training and adult_ch6_test data sets.

19) Use random forests on the training set to predict income using marital status and capital gains and losses.

In [53]:
```
#import necessary libraries
import statsmodels.tools.tools as stattools
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.ensemble import RandomForestClassifier

#import data sets
```

```
adult_train = pd.read_csv("adult_ch6_training")
adult_test = pd.read_csv("adult_ch6_test")
adult_train.head()
```

Out[53]:

	Marital status	Income	Cap_Gains_Losses
0	Never-married	<=50K	0.02174
1	Divorced	<=50K	0.00000
2	Married	<=50K	0.00000
3	Married	<=50K	0.00000
4	Married	<=50K	0.00000

In [54]:
```
#prepare dataset for modeling

#save target variable as y
y = adult_train[['Income']]

#convert marital status to dummy variables and add them back into the dataframe
mar_np = np.array(adult_train['Marital status'])
(mar_cat, mar_cat_dict) = stattools.categorical(mar_np, drop=True, dictnames=Tr
mar_cat_pd = pd.DataFrame(mar_cat)
X = pd.concat((adult_train[['Cap_Gains_Losses']], mar_cat_pd), axis=1)

#show dummy variable dictionary
mar_cat_dict
```

Out[54]:
```
{0: 'Divorced', 1: 'Married', 2: 'Never-married', 3: 'Separated', 4: 'Widowe
d'}
```

In [64]:
```
#change dummy variable column names in new dataframe
X_names = ['Cap_Gains_Losses',
           'Divorced', 'Never-Married',
           'Separated', 'Widowed']

#specify levels of target variable
y_names = ['<=50K', '>50K']

#format response variable into a one-dimensional array
rfy = np.ravel(y)

#create random forest
rf01 = RandomForestClassifier(n_estimators=100,
                              criterion='gini').fit(X,rfy)

#show predictions counts
rf01.predict(X) #predictions are stored here
pred_counts= np.unique(rf01.predict(X), return_counts=True)
pred_counts
```

Out[64]:
```
(array(['<=50K', '>50K'], dtype=object), array([17375,  1386]))
```

The random forest made the following predictions on the training set:

<=50K 17375

>50K 1386

20) Use random forests using the test data set that utilizes the same target and predictor variables. Does the test data result match the training test result?

In [65]:
```
#prepare test set for modeling as before

#save target variable as y
y_test = adult_test[['Income']]

#convert marital status to dummy variables and add them back into the dataframe
mar_np_test = np.array(adult_test['Marital status'])
(mar_cat_test, mar_cat_dict) = stattools.categorical(mar_np_test, drop=True,
                                                     dictnames=True)
mar_cat_pd_test = pd.DataFrame(mar_cat_test)
X_test = pd.concat((adult_test[['Cap_Gains_Losses']],
                    mar_cat_pd_test), axis=1)

#format response variable into a one-dimensional array
rfy_test = np.ravel(y_test)

#create random forest
rf02 = RandomForestClassifier(n_estimators=100,
                              criterion='gini').fit(X_test,rfy_test)

#show predictions counts
rf02.predict(X_test) #predictions are stored here
pred_counts= np.unique(rf01.predict(X_test), return_counts=True)
pred_counts
```

Out[65]:
```
(array(['<=50K', '>50K'], dtype=object), array([5708,  447]))
```

The random forest made the following predictions on the test set:

<=50K 5708

>50K 447