DSC-530: Predictive Modeling
Week 6 Assignment: Predictive Modeling Using Generalized Linear Models
Author: Jonathan Ibifubara Pollyn
College of Science, Engineering and Technology, Grand Canyon University

In [44]:
```
#Importing the required packages
import pandas as pd
import numpy as np
import statsmodels.api as sm
from scipy import stats
import statsmodels.tools.tools as stattools
import matplotlib.pyplot as plt
import math
```

In [4]:
```
#Importing the Adult Data set
adult = pd.read_csv('C:/School/DSC-530/DataSets/Adult')
```

In [6]:
```
#View first 5 of the data
adult.head(5)
```

Out[6]:

	age	workclass	demogweight	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba

1. Build a logistic regression model to predict the income of a person based on their age, education (as a number, with variable num), and the hours worked per week. Obtain the summary of the model.

In [9]:
```
#Isolating the predictor and response variables
X = pd.DataFrame(adult[['age', 'education-num', 'hours-per-week']])
X = sm.add_constant(X)
```

In [ ]:

In [11]:
```
X.head(5)
```

Out[11]:

	const	age	education-num	hours-per-week
0	1.0	39	13	40
1	1.0	50	13	13
2	1.0	38	9	40
3	1.0	53	7	40
4	1.0	28	13	40

In [22]:
```
income_var = np.array(adult['income'])
(income_cat, income_cat_dict) = stattools.categorical(income_var, drop=True, dictnames=True)
```

In [23]:
```
income_cat
```

Out[23]:
```
array([[1., 0.],
       [1., 0.],
       [1., 0.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]])
```

In [24]:
```
income_cat_dict
```

Out[24]:
```
{0: '<=50K.', 1: '>50K.'}
```

In [25]:
```
y = pd.DataFrame(income_cat)
```

In [29]:
```
income_ind = pd.get_dummies(adult['income'], drop_first = True)
```

In [31]:
```
y = pd.DataFrame(income_ind)
```

In [32]:
```
y
```

Out[32]:

	>50K.
0	0
1	0
2	0
3	0
4	0
...	...
24995	0
24996	0
24997	0
24998	0
24999	0

25000 rows × 1 columns

In [34]:
```
logreg01 = sm.Logit(y, X).fit()
```

```
Optimization terminated successfully.
         Current function value: inf
         Iterations 7
C:\Anaconda\lib\site-packages\statsmodels\discrete\discrete_model.py:1810: RuntimeWarning: overflow encountered
in exp
  return 1/(1+np.exp(-X))
C:\Anaconda\lib\site-packages\statsmodels\discrete\discrete_model.py:1863: RuntimeWarning: divide by zero encou
ntered in log
  return np.sum(np.log(self.cdf(q*np.dot(X,params))))
```

In [35]:
```
logreg01.summary2()
```

```
C:\Anaconda\lib\site-packages\statsmodels\discrete\discrete_model.py:1810: RuntimeWarning: overflow encountered
in exp
  return 1/(1+np.exp(-X))
C:\Anaconda\lib\site-packages\statsmodels\discrete\discrete_model.py:1863: RuntimeWarning: divide by zero encou
ntered in log
  return np.sum(np.log(self.cdf(q*np.dot(X,params))))
C:\Anaconda\lib\site-packages\statsmodels\base\model.py:547: HessianInversionWarning: Inverting hessian failed,
no bse or cov_params available
  warnings.warn('Inverting hessian failed, no bse or cov_params '
C:\Anaconda\lib\site-packages\statsmodels\base\model.py:547: HessianInversionWarning: Inverting hessian failed,
no bse or cov_params available
  warnings.warn('Inverting hessian failed, no bse or cov_params '
```

Out[35]:

Model:	Logit	Pseudo R-squared:	inf
Dependent Variable:	>50K.	AIC:	inf
Date:	2021-10-11 12:14	BIC:	inf
No. Observations:	25000	Log-Likelihood:	-inf
Df Model:	3	LL-Null:	0.0000
Df Residuals:	24996	LLR p-value:	1.0000
Converged:	1.0000	Scale:	1.0000
No. Iterations:	7.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
const	-8.4611	0.1252	-67.5880	0.0000	-8.7064	-8.2157
age	0.0459	0.0013	34.9566	0.0000	0.0434	0.0485
education-num	0.3449	0.0074	46.5794	0.0000	0.3304	0.3595
hours-per-week	0.0423	0.0014	29.1656	0.0000	0.0394	0.0451

1. Are there any variables that should be removed from the model from the previous exercise? If so, remove the variables and rerun the model.

In [ ]:
```
#No
```

1. Write the descriptive form of the final logistic regression model from the previous exercise.

In [37]:
```
#phat_income = (exp(-8.461+0.046(age)+0.345(education-num)+0.042(hours-per-week)/1+exp(-8.461+0.046(age)+0.345
```

1. Interpret the coefficient of the age variable.

In [46]:
```
math.exp(0.049)
```

Out[46]: 1.050220350740028

1. Find the impact on the probability of having high income for every 10 years a person is older.

In [48]:
```
math.exp(0.049*10)
```

Out[48]: 1.632316219955379

1. Interpret the coefficient of the num variable.

In [49]:
```
math.exp(0.3449)
```

Out[49]: 1.4118487277354066

1. Find the impact on the probability of having high income for every four or more years of education a person has.

In [50]:
```
math.exp(0.3449*4)
```

Out[50]: 3.9733119847934852

1. Interpret the coefficient of the per.week variable.

In [51]:
```
math.exp(0.0423)
```

Out[51]: 1.0432073940293334

1. Find the impact on the probability of having high income for every five or more hours per week a person works.

In [52]:
```
math.exp(0.0423*5)
```

Out[52]: 1.2355299656287

1. Obtain the predicted values using the model from the previous exercise. Compare the predicted values to the actual values.

In [38]:
```
ypred = logreg01.predict(X)
```

In [39]:
```
ypred
```

Out[39]:
```
0        0.378923
1        0.244160
2        0.127881
3        0.127805
4        0.269032
           ...
24995    0.122238
24996    0.057714
24997    0.112543
24998    0.086591
24999    0.186102
Length: 25000, dtype: float64
```

In [40]:
```
#Actual values
yture = adult['income']
```

In [41]:
```
y
```

Out[41]:

	>50K.
0	0
1	0
2	0
3	0
4	0
...	...
24995	0
24996	0
24997	0
24998	0
24999	0

25000 rows × 1 columns

1. Build a Poisson regression model to predict the years of education a person has (using the variable num) based on a person's age and the hours they work per week. Obtain the summary of the model.

In [53]:
```
poisreg01 = sm.GLM(y, X, family = sm.families.Poisson()).fit()
```

In [54]:
```
poisreg01.summary()
```

Out[54]:

Generalized Linear Model Regression Results

Dep. Variable:	>50K.	**No. Observations:**	25000
Model:	GLM	**Df Residuals:**	24996
Model Family:	Poisson	**Df Model:**	3
Link Function:	log	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-12770.
Date:	Mon, 11 Oct 2021	**Deviance:**	13571.
Time:	13:05:20	**Pearson chi2:**	1.66e+04
No. Iterations:	6		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	-5.8308	0.080	-73.288	0.000	-5.987	-5.675
age	0.0275	0.001	28.924	0.000	0.026	0.029
education-num	0.2100	0.005	39.866	0.000	0.200	0.220
hours-per-week	0.0230	0.001	24.289	0.000	0.021	0.025

1. Are there any variables that should be removed from the model from the previous exercise? If so, remove the variables and rerun the model.

In [ ]:
```
#No
```

1. Write the descriptive form of the final Poisson regression model from the previous exercise.

In [55]:
```
#income = exp(-5.831+0.028(age)+0.210(enducation-num)+0.023(hours-per-week))
```

1. Obtain the predicted values using the model from the previous exercise. Compare the predicted values to the actual values.

In [56]:
```
ypred_poisson = poisreg01.predict(X)
```

In [57]:
```
ypred_poisson
```

Out[57]:
```
0        0.330078
1        0.239652
2        0.138660
3        0.137560
4        0.244014
           ...
24995    0.127161
24996    0.082287
24997    0.126570
24998    0.107243
24999    0.186880
Length: 25000, dtype: float64
```

In [58]:
```
#Actual value
ytrue = adult['income']
```

In [ ]: