“Classification problem using Logistic Regression on Python and Exploratory Data Analysis for predicting health Insurance top up propensity post Covid pandemic”

Hardware Requirements

Software Requirements

TASKS PERFORMED

Projects

Tools used

HARDWARE REQUIREMENTS-

- Memory - 1.5 GB or 4 GB with Oracle XE
- Processor Type - 64 bit
- Processor Speed - 1.83 GHz*1
- Swap Space - 2.1 GB
- Hard Disk Space - 5 GB in /u01 or 2 GB in /tmp

SOFTWARE REQUIREMENTS

- Windows 8
- Jupyter Notebook
- Python Libraries(Pandas,Scikit Learn,Matplotlib)
- Anaconda

Project Abstract:

There is surge in buying and topping up insurance cover by customers post Covid Wave II in India. It is imperative for insurance providers to understand the segment of customers who are preferring this top-up and understanding what factors are driving this decision. This will help insurance provider to create more focused products.

The problem the team agreed to research is "*To understand the Factors (post Covid wave) that influences the intent to purchase or top up the insurance product in Indian market*". The drive for doing this study was to get an idea of how a cross section of the Indian adult population were reacting to the pandemic in terms of their buying pattern for Health & Life Insurance policies. The idea of the direct survey was to get feedback as to how people are looking at safeguarding their Life & Health for themselves & their loved ones considering the conditions prevailing due to the pandemic.
The study tried to focus on different socioeconomic factors like income, age, gender, residential areas around Metros or non-Metros, hinterland & so on to find patterns in their propensity for Health & Life risk coverages through Insurance.

Exploratory Data Analysis

Create independent and dependent variable as input for model.

```
x = df_input.drop('plan_lcover_inc', axis=1)
y = df_input['plan_lcover_inc']
X_train, X_test, Y_train, Y_test = train_test_split(x, y, test_size=0.3, random_state=1)
X_train.info()
```

In [24]:
```
import os
os.getcwd()
import pandas as pd
dataset=pd.read_csv("C:\\Users\\lenovo\\Desktop\\Nook Project\\Input.csv")
dataset.head()
```

Out[24]:

	ID	gender	age	education	marital_status	residence	dep_0_12	dep_13_17	dep_18_60	dep_60_A	...	motivation	plan_lcover_inc	plan_hcover_inc	aware_
0	1	1	51	3	2	1	0	1	1	0	...	0	0	1	
1	2	2	26	3	1	1	0	0	0	0	...	0	1	1	
2	3	1	43	4	1	1	0	0	0	2	...	0	0	1	
3	4	2	26	3	1	1	0	0	0	0	...	0	0	1	
4	5	1	51	4	2	1	0	0	2	0	...	0	1	1	

5 rows × 25 columns

In [25]:
```
dataset.head()
```

Out[25]:

	ID	gender	age	education	marital_status	residence	dep_0_12	dep_13_17	dep_18_60	dep_60_A	...	motivation	plan_lcover_inc	plan_hcover_inc	aware_
0	1	1	51	3	2	1	0	1	1	0	...	0	0	1	
1	2	2	26	3	1	1	0	0	0	0	...	0	1	1	
2	3	1	43	4	1	1	0	0	0	2	...	0	0	1	
3	4	2	26	3	1	1	0	0	0	0	...	0	0	1	
4	5	1	51	4	2	1	0	0	2	0	...	0	1	1	

5 rows × 25 columns

In [8]:
```
dataset.shape
```

Out[8]:
```
(79, 25)
```

In [26]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 79 entries, 0 to 78
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   ID                    79 non-null     int64
 1   gender                79 non-null     int64
 2   age                   79 non-null     int64
 3   education             79 non-null     int64
 4   marital_status        79 non-null     int64
 5   residence             79 non-null     int64
 6   dep_0_12              79 non-null     int64
 7   dep_13_17             79 non-null     int64
 8   dep_18_60             79 non-null     int64
 9   dep_60_A              79 non-null     int64
 10  income                79 non-null     object
 11  job_type              79 non-null     int64
 12  bfsi                  79 non-null     int64
 13  existing_health_prob  79 non-null     int64
 14  family_health_history 79 non-null     int64
```

In [10]:
```
dataset.describe()
```

Out[10]:

	ID	gender	age	education	marital_status	residence	dep_0_12	dep_13_17	dep_18_60	dep_60_A	...	motivation	plan_Icover_inc	plar
count	79.000000	79.000000	79.000000	79.000000	79.000000	79.000000	79.000000	79.000000	79.000000	79.000000	...	79.000000	79.000000	
mean	40.000000	1.291139	35.481013	3.379747	1.594937	1.202532	0.468354	0.189873	1.075949	0.721519	...	0.088608	0.303797	
std	22.949219	0.457190	10.330265	0.561678	0.494041	0.585722	0.694953	0.455056	1.152168	0.918984	...	0.285992	0.462835	
min	1.000000	1.000000	21.000000	2.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	
25%	20.500000	1.000000	27.500000	3.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	
50%	40.000000	1.000000	32.000000	3.000000	2.000000	1.000000	0.000000	0.000000	1.000000	0.000000	...	0.000000	0.000000	
75%	59.500000	2.000000	45.000000	4.000000	2.000000	1.000000	1.000000	0.000000	2.000000	1.000000	...	0.000000	1.000000	
max	79.000000	2.000000	60.000000	5.000000	2.000000	3.000000	2.000000	2.000000	4.000000	4.000000	...	1.000000	1.000000	

8 rows × 24 columns

Logistic Regression

```
lr = LogisticRegression(max_iter=10000)
lr.fit(X_train, Y_train)
lr_pred = lr.predict(X_test)
```

In [4]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, plot_confusion_matrix

df_input = pd.read_csv("C:\\Users\\lenovo\\Desktop\\Nook Project\\Input.csv")
df_input.head()
```

Out[4]:

	ID	gender	age	education	marital_status	residence	dep_0_12	dep_13_17	dep_18_60	dep_60_A	...	motivation	plan_lcover_inc	plan_hcover_inc	aware_
0	1	1	51	3	2	1	0	1	1	0	...	0	0	1	
1	2	2	26	3	1	1	0	0	0	0	...	0	1	1	
2	3	1	43	4	1	1	0	0	0	2	...	0	0	1	
3	4	2	26	3	1	1	0	0	0	0	...	0	0	1	
4	5	1	51	4	2	1	0	0	2	0	...	0	1	1	

5 rows × 25 columns

Transform income into three band (1) Low Income i.e. 0 - 200000 (2) Mid Income i.e. 200001 - 1000000 and (3) High Income i.e. more than 1000000

In [6]:
```
# Drop the factors which are not used in the model.

del df_input['ID']
del df_input['education']
del df_input['dep_0_12']
del df_input['dep_13_17']
del df_input['dep_18_60']
del df_input['dep_60_A']
del df_input['existing_health_prob']
del df_input['family_health_history']

del df_input['plan_hcover_inc']
del df_input['aware_covid_ins']

del df_input['KIDS']
del df_input['SENIORS']
del df_input['HEALTH_INS']
del df_input['COVID_INS']

#del df_input['gender']
#del df_input['age']
#del df_input['marital_status']
#del df_input['residence']
#del df_input['income']
#del df_input['job_type']
#del df_input['bfsi']
#del df_input['motivation']
#del df_input['TOT_DEP']
#del df_input['LIFE_INSURANCE']

#del df_input['plan_lcover_inc']
```

Create independent and dependent variable as input for model.

In [7]:
```
x = df_input.drop('plan_lcover_inc', axis=1)
y = df_input['plan_lcover_inc']
X_train, X_test, Y_train, Y_test = train_test_split(x, y, test_size=0.3, random_state=1)
X_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 55 entries, 19 to 37
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   gender          55 non-null     int64
 1   age             55 non-null     int64
 2   marital_status  55 non-null     int64
 3   residence       55 non-null     int64
 4   income          55 non-null     int64
 5   job_type        55 non-null     int64
 6   bfsi            55 non-null     int64
 7   motivation      55 non-null     int64
 8   TOT_DEP         55 non-null     int64
 9   LIFE_INSURANCE  55 non-null     int64
dtypes: int64(10)
memory usage: 4.7 KB
```

Accuracy of the model

```
In [8]: lr = LogisticRegression(max_iter=10000)
        lr.fit(X_train, Y_train)
        lr_pred = lr.predict(X_test)
```

```
In [9]: print('Classification Report of Logistic Regression Model \n\n', classification_report(Y_test, lr_pred))
```

```
Classification Report of Logistic Regression Model

              precision    recall  f1-score   support

           0       0.70      0.93      0.80        15
           1       0.75      0.33      0.46         9

    accuracy                           0.71        24
   macro avg       0.72      0.63      0.63        24
weighted avg       0.72      0.71      0.67        24
```

```
In [11]: print('Accuracy Score of Logistic Regression Model \n\n',accuracy_score(Y_test, lr_pred))
```

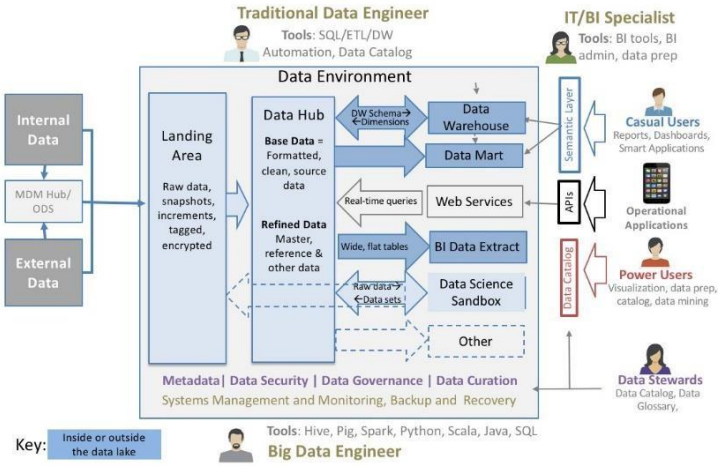
```
Accuracy Score of Logistic Regression Model

 0.7083333333333334
```

Input.csv

Predict.csv

Above dataset input.csv used through direct survey of 80 respondents post EDA where 25 attributes or columns were treated for the model to be trained ,Testing set Predict.csv used to test model for accuracy with a split of 70% for training & 30% for testing

Architecture:



TOOLS USED

Python

Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected..

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.

Python consistently ranks as one of the most popular programming languages

REFLECTION NOTES

.I was thoroughly involved in the entirety of the project duration. I worked on projects which had different limitations in terms of time based on the real world requirements. All these projects were implemented in real world scenarios hence this gave me invaluable experience and knowledge that I would not have gotten by building projects on my own.

Along with these, I gained knowledge about working with which were used to implement these different projects.

Apart from the technical skills, I greatly improved my verbal and non-technical skills through the various interactions that I had during my internship period. Along with that, as the projects were to be implemented in real world scenarios, time was very

important. By working on the different problems encountered while building these projects my problem solving skills have greatly improved.