Naive-Bayes dengan data Visit-Nominal.csv

1. Inisialisasikan library yang diperlukan untuk dataset ini.

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statistics

from sklearn.naive_bayes import MultinomialNB
from sklearn import svm

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from yellowbrick.classifier import ClassificationReport
```

2. Masukan dataset Visit-Nominal.csv kedalam dataframe

In [2]:
```
df = pd.read_csv("\\Users\\aftermath\\Documents\\Machine Learning\\Visit-Nominal.csv",header=None,
skipinitialspace=True)
attrs = []
for attr in range(7):
    attrs.append(df.at[0,attr])

dfnew = pd.read_csv("\\Users\\aftermath\\Documents\\Machine Learning\\Visit-Nominal.csv",header=No
ne, skipinitialspace=True, skiprows = 1)
dfnew.columns = attrs

dfnew.head()
```

Out[2]:

	Home	Browsed	Searched	Prod_A	Prod_B	Prod_C	Visit_Again
0	yes	no	no	no	no	no	no
1	yes	yes	yes	no	no	no	no
2	yes	no	no	no	no	no	no
3	yes	yes	yes	yes	no	no	yes
4	yes	no	yes	yes	yes	no	yes

In [3]:
```
print(dfnew.describe())
```

```
        Home Browsed Searched Prod_A Prod_B Prod_C Visit_Again
count    100     100      100    100    100    100         100
unique     2       2        2      2      2      2           2
top      yes     yes       no    yes    yes     no          no
freq      60      72       57     53     55     55          61
```

3. Representasikan data 'yes' dan 'no' kedalam bentuk biner '1' dan '0'

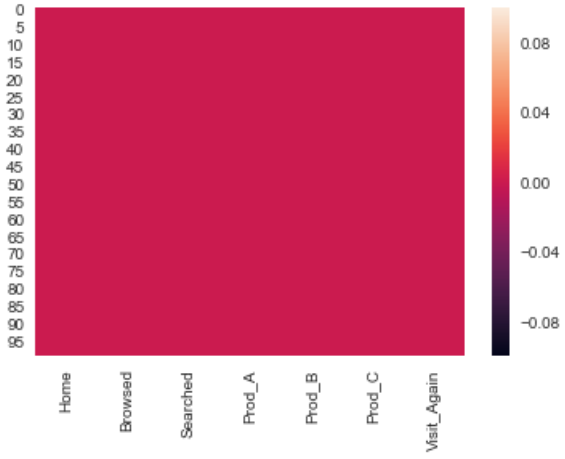
In [4]:
```
toBinary = lambda x: 1 if x=="yes" else 0
dfnew = dfnew.applymap(toBinary)
```

4. Cek apakah terdapat value fitur yang kosong

In [5]:
```
missing_values = dfnew.isnull()
missing_values

sns.heatmap(data = missing_values)
```

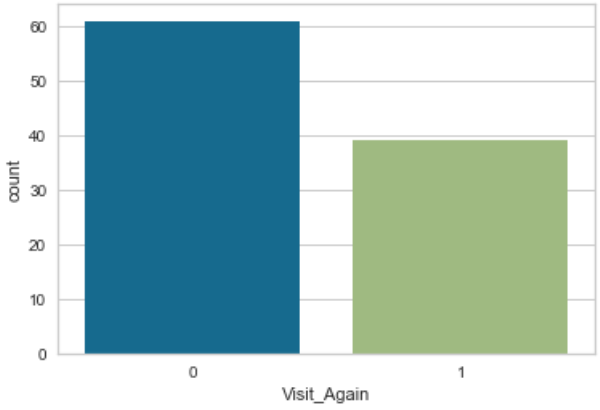
Out[5]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x23d6753e240>
```



5. Lihat perbandingan jumlah class 0 dan 1

In [6]:
```
sns.countplot(x='Visit_Again', data=dfnew)
dfnew.Visit_Again.value_counts()
```

Out[6]:
```
0    61
1    39
Name: Visit_Again, dtype: int64
```



6. Siapkan data untuk training secara biasa dan training dengan cross validation

In [7]:
```
feature = attrs
feature.pop()
feature
```

Out[7]:
```
['Home', 'Browsed', 'Searched', 'Prod_A', 'Prod_B', 'Prod_C']
```

In [8]:
```
features = dfnew[feature]
label = dfnew['Visit_Again']

X_train, X_test, y_train, y_test = train_test_split(features, label, test_size=0.25, random_state=
101)
feature
```

Out[8]:
```
['Home', 'Browsed', 'Searched', 'Prod_A', 'Prod_B', 'Prod_C']
```

7. Train data secara biasa, didapatkan skor sebesar 0.64

In [9]:
```
naivebayesmodel = MultinomialNB()

naivebayesmodel.fit(X_train, y_train)
print("SCORE : ",naivebayesmodel.score(X_test, y_test))
```

```
SCORE :  0.64
```

8. Train data dengan cross-validation, didapatkan skor sebesar 0.58

In [10]:
```
naivebayesmodelkfold = MultinomialNB()

score =  cross_val_score(naivebayesmodelkfold, features, label, cv=5)
print("CROSS VALIDATION SCORE : ",statistics.mean(score))
```

```
CROSS VALIDATION SCORE :  0.580125313283208
```

9. Melihat log probabilitas setiap fitur dari class 0 dan 1

In [11]:
```
class0_attrprob = naivebayesmodel.feature_log_prob_[0]
class1_attrprob = naivebayesmodel.feature_log_prob_[1]
print("probability of class 0 ")
for i in (range(len(naivebayesmodel.feature_log_prob_[0]))):
    print("feature ",i+1,": ",pow(2,class0_attrprob[i]))

print("\nprobability of class 1 ")
for i in (range(len(naivebayesmodel.feature_log_prob_[1]))):
    print("feature ",i+1,": ",pow(2,class1_attrprob[i]))
```

```
probability of class 0
feature  1 :  0.3161327376206385
feature  2 :  0.33772304414021753
feature  3 :  0.23034112959519082
feature  4 :  0.2708298753988346
feature  5 :  0.3087906120169887
feature  6 :  0.2629570684330914

probability of class 1
feature  1 :  0.2925860039176619
feature  2 :  0.34677099129351924
feature  3 :  0.19467587271644443
feature  4 :  0.3362493442478049
feature  5 :  0.2925860039176619
feature  6 :  0.2578512790415785
```

In [ ]: