CLASSIFICATION

In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
import sklearn

from pandas import Series, DataFrame
from pylab import rcParams
from sklearn import preprocessing
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import mean_squared_error
from sklearn.metrics import explained_variance_score
```

PERFORMING DECISION TREE CLASSIFIER ON CONGRESS-TERMS DATA SET

In [2]:

```
data1=pd.read_csv('congress-terms.csv')
data1.head()
```

Out[2]:

	congress	chamber	bioguide	firstname	middlename	lastname	suffix	birthday	state	party	incumbent	termstart	age
0	80	house	M000112	Joseph	Jefferson	Mansfield	NaN	1861-02-09	TX	D	Yes	1947-01-03	85.9
1	80	house	D000448	Robert	Lee	Doughton	NaN	1863-11-07	NC	D	Yes	1947-01-03	83.2
2	80	house	S000001	Adolph	Joachim	Sabath	NaN	1866-04-04	IL	D	Yes	1947-01-03	80.7
3	80	house	E000023	Charles	Aubrey	Eaton	NaN	1868-03-29	NJ	R	Yes	1947-01-03	78.8
4	80	house	L000296	William	NaN	Lewis	NaN	1868-09-22	KY	R	No	1947-01-03	78.3

EXPLORING THE DATA SET

In [3]:

```
data1.describe()
```

Out[3]:

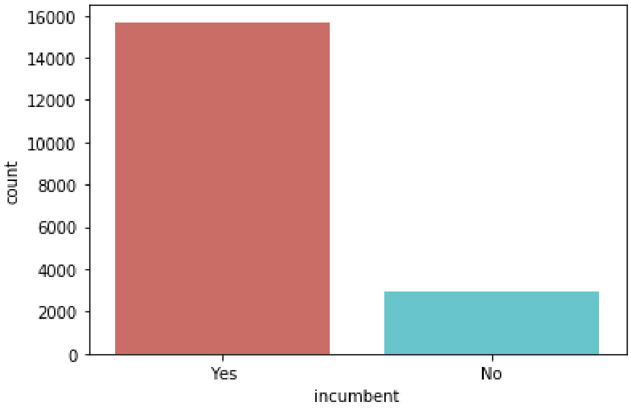
	congress	age
count	18635.000000	18635.000000
mean	96.445989	53.313732
std	9.823429	10.678469
min	80.000000	25.000000
25%	88.000000	45.400000
50%	96.000000	53.000000
75%	105.000000	60.550000
max	113.000000	98.100000

In [4]:

```
sb.countplot(x='incumbent',data=data1, palette='hls')
```

Out[4]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a181f9450>
```



In [5]:

```
data1.isnull().sum()
```

Out[5]:

```
congress          0
chamber           0
bioguide          0
firstname         0
middlename     3536
lastname          0
suffix        16937
birthday          0
state             0
party             0
incumbent         0
termstart         0
age               0
dtype: int64
```

In [6]:

```
data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18635 entries, 0 to 18634
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   congress    18635 non-null  int64  
 1   chamber     18635 non-null  object 
 2   bioguide    18635 non-null  object 
 3   firstname   18635 non-null  object 
 4   middlename  15099 non-null  object 
 5   lastname    18635 non-null  object 
 6   suffix      1698 non-null   object 
 7   birthday    18635 non-null  object 
 8   state       18635 non-null  object 
 9   party       18635 non-null  object 
 10  incumbent   18635 non-null  object 
 11  termstart   18635 non-null  object 
 12  age         18635 non-null  float64
dtypes: float64(1), int64(1), object(11)
memory usage: 1.8+ MB
```

In [7]:

```
data1 = data1.drop(['congress','bioguide','firstname','middlename','lastname','suffix','birthday',
'state','party','termstart'], 1)
data1.head()
```

Out[7]:

	chamber	incumbent	age
0	house	Yes	85.9
1	house	Yes	83.2
2	house	Yes	80.7
3	house	Yes	78.8
4	house	No	78.3

In [8]:

```
data1 = data1.fillna(method='ffill')
```

In [9]:

```
data1.isnull().sum()
```

Out[9]:

```
chamber      0
incumbent    0
age          0
dtype: int64
```

In [10]:

```
data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18635 entries, 0 to 18634
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   chamber    18635 non-null  object
 1   incumbent  18635 non-null  object
 2   age        18635 non-null  float64
dtypes: float64(1), object(2)
memory usage: 436.9+ KB
```

In [11]:

```
incumbent_b = pd.get_dummies(data1['incumbent'],drop_first=True)
incumbent_b.head()
```

Out[11]:

	Yes
0	1
1	1
2	1
3	1
4	0

In [12]:

```
chamber_b=pd.get_dummies(data1['chamber'])
chamber_b.head()
```

Out[12]:

Out[12]:

	house	senate
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0

In [13]:

```
data1 = data1.drop(['chamber', 'incumbent'],1)
data1.head()
```

Out[13]:

	age
0	85.9
1	83.2
2	80.7
3	78.8
4	78.3

In [14]:

```
data2 = pd.concat([data1,chamber_b,incumbent_b],axis=1)
data2.head()
```

Out[14]:

	age	house	senate	Yes
0	85.9	1	0	1
1	83.2	1	0	1
2	80.7	1	0	1
3	78.8	1	0	1
4	78.3	1	0	0

In [15]:

```
data_f = data2.rename(columns={'Yes': 'incumbent_n'}, index={'ONE': 'one'})
data_f.head()
```

Out[15]:

	age	house	senate	incumbent_n
0	85.9	1	0	1
1	83.2	1	0	1
2	80.7	1	0	1
3	78.8	1	0	1
4	78.3	1	0	0

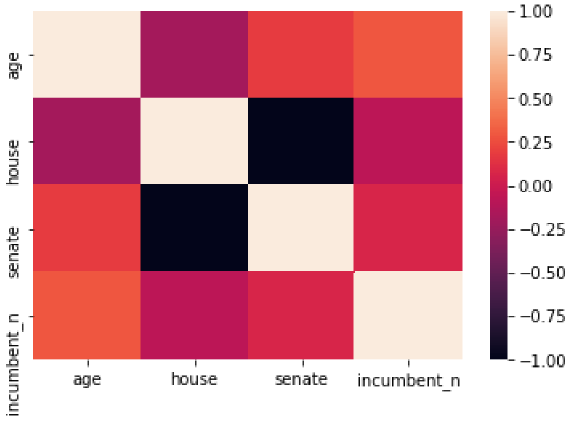
In [16]:

```
sb.heatmap(data_f.corr())
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a18a26710>
```

```
<matplotlib.axes._subplots.AxesSubplot at [illegible]
```



splitting of data

In [25]:

```
M_train = data_f[['age','house','senate']].values
N_train = data_f['incumbent_n'].values
```

In [26]:

```
M_train, M_test, N_train, N_test = train_test_split(M_train, N_train, test_size = .3, random_state=
25)
MR_train, MR_test, NR_train, NR_test = train_test_split(M_train, N_train, test_size = .3, random_st
ate=25)
```

In [27]:

```
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()
clf2 = DecisionTreeClassifier()
# Train Decision Tree Classifer
clf = clf.fit(M_train,N_train)
clf2 = clf.fit(MR_train,NR_train)
```

In [28]:

```
#Predict the response for test dataset
N_pred = clf.predict(M_test)
```

In [20]:

```
print("Accuracy:",metrics.accuracy_score(N_test, N_pred))
```

```
Accuracy: 0.8399213020926489
```

In [31]:

```
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(N_test, N_pred)
confusion_matrix
```

Out[31]:

```
array([[  83,   784],
       [ 125,  4599]])
```

THE CONFUSION MATRIX TELLS THAT 83 AND 4599 ARE CORRECTLY PREDICTED, 125 AND 784 are wrongly predicted

In [34]:

In [34]:

```
import os

os.system('jupyter nbconvert --to html CLASSIFICATION.ipynb')
```

Out[34]:

```
0
```

In [ ]:

```

```