VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

DIGITAL ASSESSMENT

Machine Learning Techniques
(CSE6024)
Faculty Name:- Prof. SASIKALA .R
Slot:- L33+L34

Student Names –

1) ANJALI CHAKRABORTY (21MAI0012)
2) MANISHA DAS (21MAI0029)

Dept – CSE with Sp in AI & ML

AIM: TO CONSIDER ANY DATASET AND IMPLEMENT DIFFERENT ALGORITHMS TO FIND THE FITTEST ALGORITHM FOR ANALYSIS OF THE DATASET, WHICH GIVES THE MOST ACCURATE SCORE.

MODELS USED:

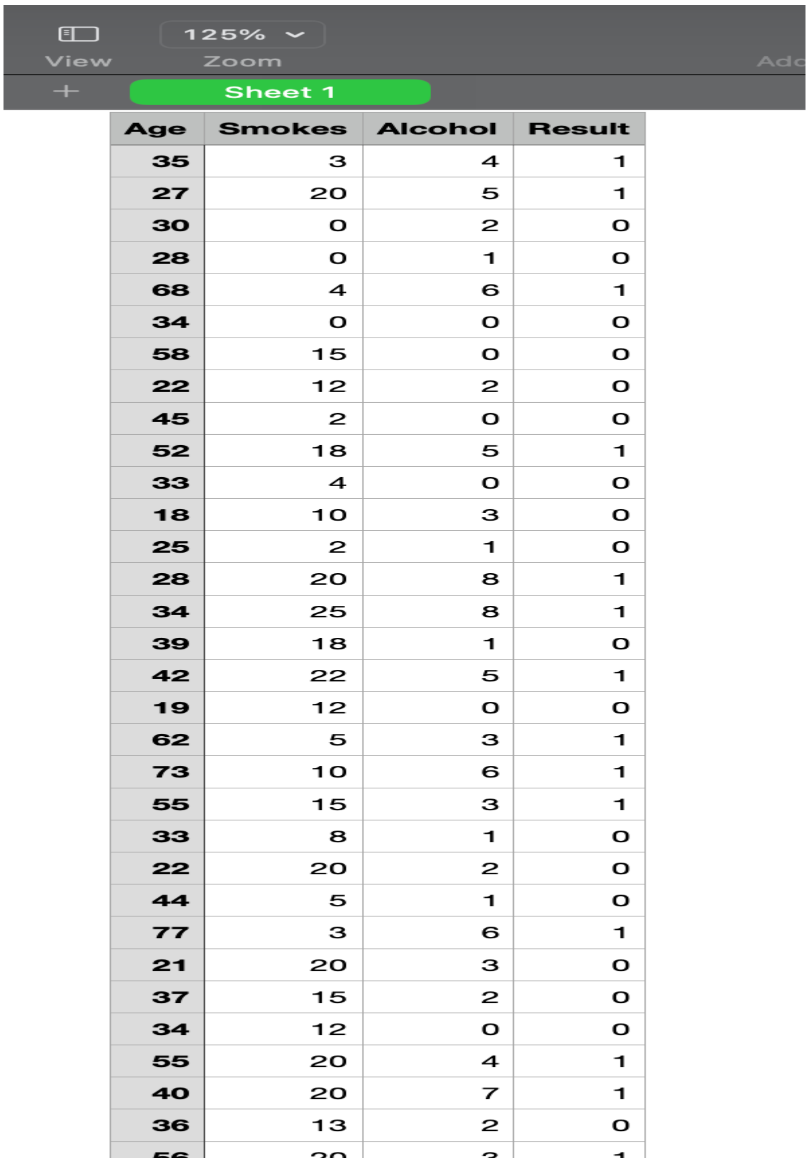
1)LOGISTIC REGRESSION

2)NAÏVE BAYES

3)KMEANS CLUSTERING

4)DECISION TREE

DATASET USED: LUNG CANCER DATASET



Age	Smokes	Alcohol	Result
35	3	4	1
27	20	5	1
30	0	2	0
28	0	1	0
68	4	6	1
34	0	0	0
58	15	0	0
22	12	2	0
45	2	0	0
52	18	5	1
33	4	0	0
18	10	3	0
25	2	1	0
28	20	8	1
34	25	8	1
39	18	1	0
42	22	5	1
19	12	0	0
62	5	3	1
73	10	6	1
55	15	3	1
33	8	1	0
22	20	2	0
44	5	1	0
77	3	6	1
21	20	3	0
37	15	2	0
34	12	0	0
55	20	4	1
40	20	7	1
36	13	2	0

INTRODUCTION TO THE METHODS USED

1) LOGISTIC REGRESION:

Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set. A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables.

2) NAÏVE BAYES CLASSIFIER:

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. A naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature, given the class variable.

3) KMEANS CLUSTERING:

The K-means clustering algorithm computes centroids and repeats until the optimal centroid is found. It is presumptively known how many clusters there are. It is also known as the flat clustering algorithm. The number of clusters found from data by the method is denoted by the letter ‘K’ in K-means. In this method, data points are assigned to clusters in such a way that the sum of the squared distances between the data points and the centroid is as small as possible. It is essential to note that reduced diversity within clusters leads to more identical data points within the same cluster.

4) DECISION TREE:

Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. Decision trees can be used for classification as well as regression problems.

OBSERVATION:

After implementing the four different methods on our dataset, the observed accuracy scores are –

- Logistic regression - 97.67 %
- Naïve bayes - 95.35%
- Kmeans clustering – 90.69%
- Decision tree – 97.67%

RESULT:

Both logistic regression and decision tree show the same accuracy level , but yes, some data sets do better with one and some with the other, so you always have the option of comparing the two models.

Provided below are the codes for implementing the above mentioned models.

1) LOGISTIC REGRESSION MODEL ¶

In [27]:

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score
```

In [28]:

```
import warnings
warnings.filterwarnings('ignore')
```

DATASET USED IS LUNG CANCER

In [29]:

```
df= pd.read_csv('/Users/anjali98/lc.csv')
df.head()
df.describe()
```

Out[29]:

	Age	Smokes	Alcohol	Result
count	107.000000	107.000000	107.000000	107.000000
mean	43.635514	16.046729	3.280374	0.485981
std	15.229931	6.724193	2.188281	0.502155
min	18.000000	0.000000	0.000000	0.000000
25%	32.000000	12.000000	2.000000	0.000000
50%	42.000000	20.000000	3.000000	0.000000
75%	56.000000	20.000000	4.500000	1.000000
max	77.000000	34.000000	8.000000	1.000000

In [30]:

```
print("Number of null values in the data set are - ",df.isnull().values.any().sum())
```

```
Number of null values in the data set are -  0
```

In [31]:

```
from sklearn.model_selection import train_test_split
y= df['Result']
x= df.drop(['Result'], axis=1)
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size= 0.4,random_state
```

In [32]:

```
modelLogistic = LogisticRegression()
modelLogistic.fit(x_train,y_train)
```

Out[32]:

```
LogisticRegression()
```

In [33]:

```
print("The intercept b0= ", modelLogistic.intercept_)

print("The coefficient b1= ", modelLogistic.coef_)
```

```
The intercept b0=  [-20.40018975]
The coefficient b1=  [[0.24838255 0.24177754 1.70960268]]
```

In [34]:

```
ypred = modelLogistic.predict(x_test)
```

In [35]:

```
log_score = modelLogistic.score(x_test, y_test)

from sklearn import metrics
cm = metrics.confusion_matrix(y_test, ypred)
print(cm)
```

```
[[23  0]
 [ 1 19]]
```

FINAL ACCURACY SCORE

In [36]:

```
all_sample_title = 'Accuracy Score: {0}'.format(log_score)
print(all_sample_title)
```

```
Accuracy Score: 0.9767441860465116
```

In [ ]:

```

```

NAIVE BAYES CLASSIFIER MODEL

In [55]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score
```

In [56]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [57]:

```
df= pd.read_csv('/Users/anjali98/lc.csv')
df.head()
df.describe()
```

Out[57]:

	Age	Smokes	Alcohol	Result
count	107.000000	107.000000	107.000000	107.000000
mean	43.635514	16.046729	3.280374	0.485981
std	15.229931	6.724193	2.188281	0.502155
min	18.000000	0.000000	0.000000	0.000000
25%	32.000000	12.000000	2.000000	0.000000
50%	42.000000	20.000000	3.000000	0.000000
75%	56.000000	20.000000	4.500000	1.000000
max	77.000000	34.000000	8.000000	1.000000

In [58]:

```
y= df['Result']
X= df.drop(['Result'], axis=1)
```

In [59]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.4, random_st
```

In [60]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [61]:

```
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)
```

Out[61]:

```
GaussianNB()
```

In [62]:

```
y_pred  =  classifier.predict(X_test)
```

In [63]:

```
from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[22  1]
 [ 1 19]]
```

FINAL ACCURACY SCORE

In [64]:

```
ac = accuracy_score(y_test,y_pred)
print('Accuarcy score : ', ac)
```

```
Accuarcy score :  0.9534883720930233
```

In [ ]:

```

```

KMEANS CLUSTERING MODEL

In [1]:

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import scipy.cluster.hierarchy as sch
```

In [2]:

```
df= pd.read_csv('/Users/anjali98/lc.csv')
df.head()
df.describe()
```

Out[2]:

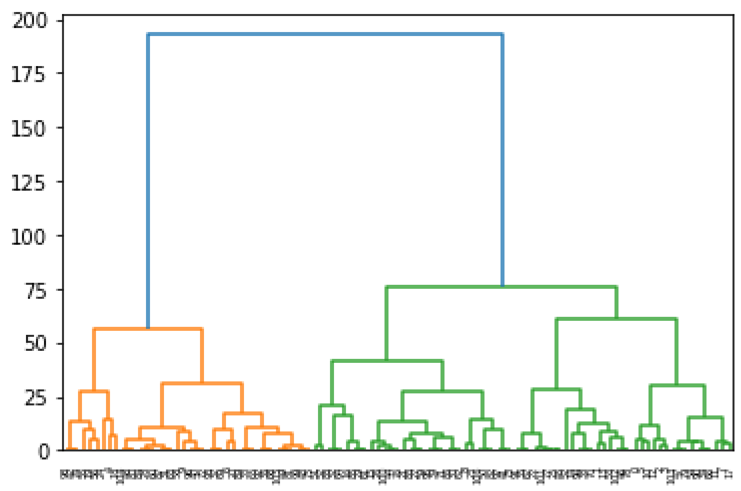
	Age	Smokes	Alcohol	Result
count	107.000000	107.000000	107.000000	107.000000
mean	43.635514	16.046729	3.280374	0.485981
std	15.229931	6.724193	2.188281	0.502155
min	18.000000	0.000000	0.000000	0.000000
25%	32.000000	12.000000	2.000000	0.000000
50%	42.000000	20.000000	3.000000	0.000000
75%	56.000000	20.000000	4.500000	1.000000
max	77.000000	34.000000	8.000000	1.000000

In [3]:

```
y= df['Result']
X= df.drop(['Result'], axis=1)
```

In [4]:

```
dendrogram = sch.dendrogram(sch.linkage(X, method='ward'))
```



In [5]:

```

y= df['Result']
X= df.drop(['Result'], axis=1)
```

In [6]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y, test_size=0.4, random_state=42
```

In [7]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=56, random_state=0)
kmeans.fit(X)
kmeans.cluster_centers_

print (kmeans.inertia_)
```

```
4.1
```

In [8]:

```
identified_clusters = kmeans.fit_predict(X)
identified_clusters
```

Out[8]:

```
array([33, 32, 38, 38,  7,  0, 26, 47, 10, 51, 41, 40, 22, 52, 30, 49,
48,
       18, 29, 36, 45, 35, 28, 46, 27, 28,  4, 25, 34, 14, 44, 54, 12,
23,
        5,  8, 11,  3, 21, 19,  1, 15,  6,  9, 39, 28,  2, 42, 24, 53,
20,
       16, 13, 43, 50, 37, 17, 55, 31, 25, 34, 14, 44, 54, 12, 23,  5,
8,
       11,  3, 21, 19,  1, 15,  6,  9, 39, 28,  2, 42, 24, 53, 20, 25,
34,
       14, 44, 54, 12, 23,  5,  8, 11,  3, 21, 19,  1, 15,  6,  9, 39,
28,
        2, 42, 24, 53, 20], dtype=int32)
```

In [12]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X_train, X_test,y_train,y_test = train_test_split(X,y,test_size=0.40,random_state=70
k_means = KMeans(n_clusters=2)
k_means.fit(X_train,y_train)
```

Out[12]:

```
KMeans(n_clusters=2)
```

FINAL ACCURACY SCORE

In [15]:

```
from sklearn.metrics import accuracy_score
score = accuracy_score(y_test,k_means.predict(X_test))
print('Accuracy: ',score)
```

```
Accuracy:  0.9069767441860465
```

In [ ]:

```
```

DECISION TREE CLASSIFIER MODEL ¶

In [32]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
import matplotlib.pyplot as plt
%matplotlib inline
```

In [33]:

```
df= pd.read_csv('/Users/anjali98/lc.csv')
df.head()
df.describe()
```

Out[33]:

	Age	Smokes	Alcohol	Result
count	107.000000	107.000000	107.000000	107.000000
mean	43.635514	16.046729	3.280374	0.485981
std	15.229931	6.724193	2.188281	0.502155
min	18.000000	0.000000	0.000000	0.000000
25%	32.000000	12.000000	2.000000	0.000000
50%	42.000000	20.000000	3.000000	0.000000
75%	56.000000	20.000000	4.500000	1.000000
max	77.000000	34.000000	8.000000	1.000000

In [34]:

```
y= df['Result']
X= df.drop(['Result'], axis=1)
```

In [35]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.4, random_st
```

In [36]:

```
clf_entropy=DecisionTreeClassifier(criterion='entropy',random_state=100,max_depth=3,
clf_entropy.fit(X_train,y_train)
```

Out[36]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=3, min_samples_l
eaf=5,
                       random_state=100)
```

In [37]:

```
y_pred=clf_entropy.predict(X_test)
```

FINAL ACCURACY SCORE

In [38]:

```
ac = (accuracy_score(y_test,y_pred)*100)
print('Accuarcy score : ', ac)
```

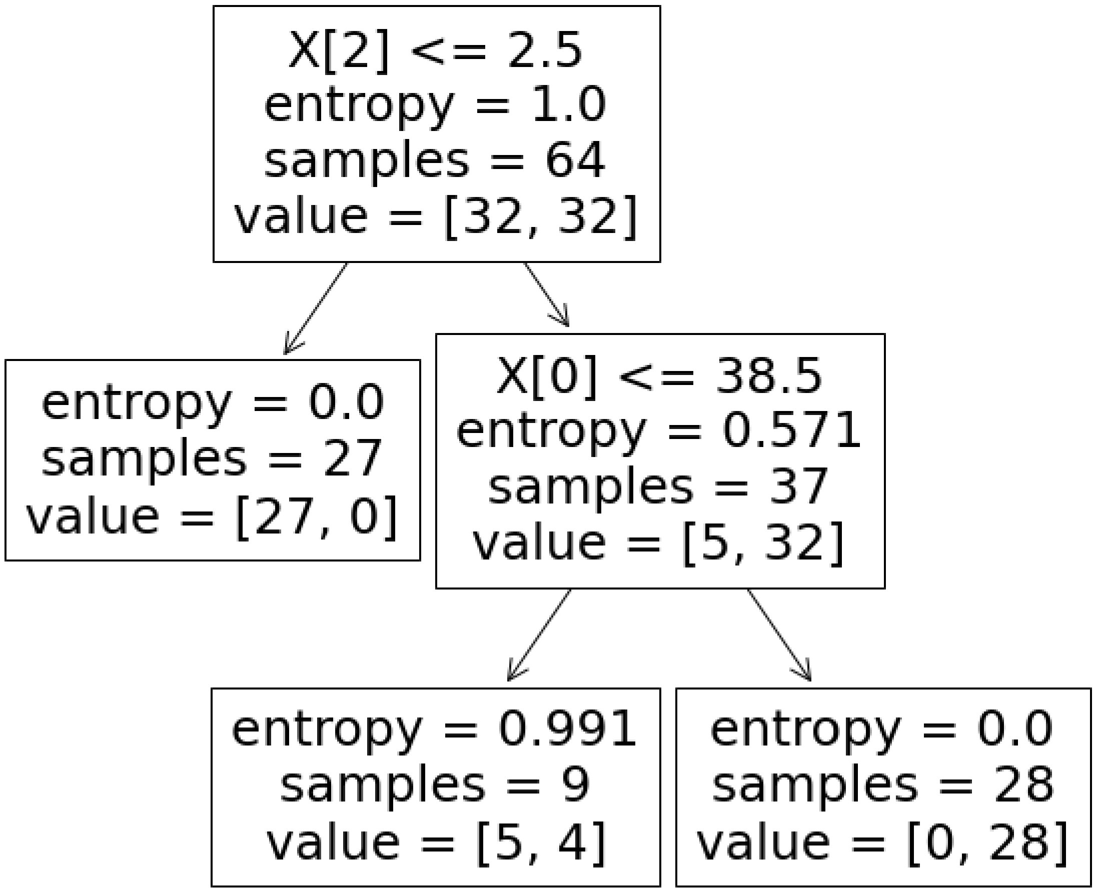
```
Accuarcy score :  97.67441860465115
```

In [39]:

```
plt.figure(figsize=(10,9))
from sklearn import tree
tree.plot_tree(clf_entropy.fit(X_train, y_train))
```

Out[39]:

```
[Text(223.2, 407.7, 'X[2] <= 2.5\nentropy = 1.0\nsamples = 64\nvalue =
[32, 32]'),
 Text(111.6, 244.62, 'entropy = 0.0\nsamples = 27\nvalue = [27, 0]'),
 Text(334.79999999999995, 244.62, 'X[0] <= 38.5\nentropy = 0.571\nsamp
les = 37\nvalue = [5, 32]'),
 Text(223.2, 81.53999999999996, 'entropy = 0.991\nsamples = 9\nvalue =
[5, 4]'),
 Text(446.4, 81.53999999999996, 'entropy = 0.0\nsamples = 28\nvalue =
[0, 28]')]
```



In [ ]: