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) Saeri Datta (21MAI0038)
2) Reola (21MAI0054)

Dept – CSE with Sp in AI & ML

To apply different learning models on a chosen dataset and compare them to get the fittest model which gives the highest accuracy score.

I. Models Used:

- SVM model
- Naïve Bayes model
- KNN
- Decision Tree

II. Introduction:

1) SVM - Support Vector Machine(SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points.
2) Naïve Bayes - Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset.
3) KNN - The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It's easy to implement and understand, but has a major drawback of becoming significantly slows as the size of that data in use grows.
4) Decision Tree - Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable.

III. Result : Accuracy score of each model is

- SVM model – 84 %
- Naïve Bayes model – 60 %
- KNN – 40 %
- Decision Tree – 62.06%

DECISION TREE

In [1]:

```
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 [57]:

```
df=pd.read_csv('C:/Users/USER/Desktop/petrol_consumption.csv')

df.head()
df.describe()
```

Out[57]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consu
count	48.000000	48.000000	48.000000	48.000000	48
mean	7.668333	4241.833333	5565.416667	0.570333	576
std	0.950770	573.623768	3491.507166	0.055470	111
min	5.000000	3063.000000	431.000000	0.451000	344
25%	7.000000	3739.000000	3110.250000	0.529750	509
50%	7.500000	4298.000000	4735.500000	0.564500	568
75%	8.125000	4578.750000	7156.000000	0.595250	632
max	10.000000	5342.000000	17782.000000	0.724000	968

In [58]:

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

In [59]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.6, random_state =1)
```

In [60]:

```
clf_entropy=DecisionTreeClassifier(criterion='entropy',random_state=100,max_depth=2,min_sam
clf_entropy.fit(X_train,y_train)
```

Out[60]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=2, min_samples_leaf=5,
                       random_state=100)
```

In [61]:

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

In [62]:

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

```
Accuarcy score :  62.06896551724138
```

KNN

In [136]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [137]:

```
df=pd.read_csv('C:/Users/USER/Desktop/petrol_consumption.csv')
```

In [106]:

```
df.head()
```

Out[106]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumpt
0	9.0	3571	1976	0.525	
1	9.0	4092	1250	0.572	
2	9.0	3865	1586	0.580	
3	7.5	4870	2351	0.529	
4	8.0	4399	431	0.544	

In [105]:

```
df.shape
```

Out[105]:

```
(48, 6)
```

In [107]:

```
col_names = ['Petrol_tax','Average_income','Paved_Highways','Population_Driver_licence(%)',

df.columns = col_names

df.columns
```

Out[107]:

```
Index(['Petrol_tax', 'Average_income', 'Paved_Highways',
       'Population_Driver_licence(%)', 'Petrol_Consumption', 'target'],
      dtype='object')
```

In [108]:

```
for var in df.columns:

    print(df[var].value_counts())
```

```
7.00     19
8.00     10
9.00      8
7.50      4
8.50      3
10.00     1
6.58      1
5.00      1
6.00      1
Name: Petrol_tax, dtype: int64
5126    2
3571    1
4045    1
3846    1
4188    1
3601    1
3640    1
3333    1
3063    1
```

In [109]:

```
df.isnull().sum()
```

Out[109]:

```
Petrol_tax                    0
Average_income                0
Paved_Highways                0
Population_Driver_licence(%)  0
Petrol_Consumption            0
target                        0
dtype: int64
```

In [110]:

```
df.isna().sum()
```

Out[110]:

```
Petrol_tax                    0
Average_income                0
Paved_Highways                0
Population_Driver_licence(%)  0
Petrol_Consumption            0
target                        0
dtype: int64
```

In [114]:

```
df['target'].value_counts()
```

Out[114]:

```
1    28
0    20
Name: target, dtype: int64
```

In [115]:

```
df['target'].unique()
```

Out[115]:

```
array([1, 0], dtype=int64)
```

In [116]:

```
df['target'].isna().sum()
```

Out[116]:

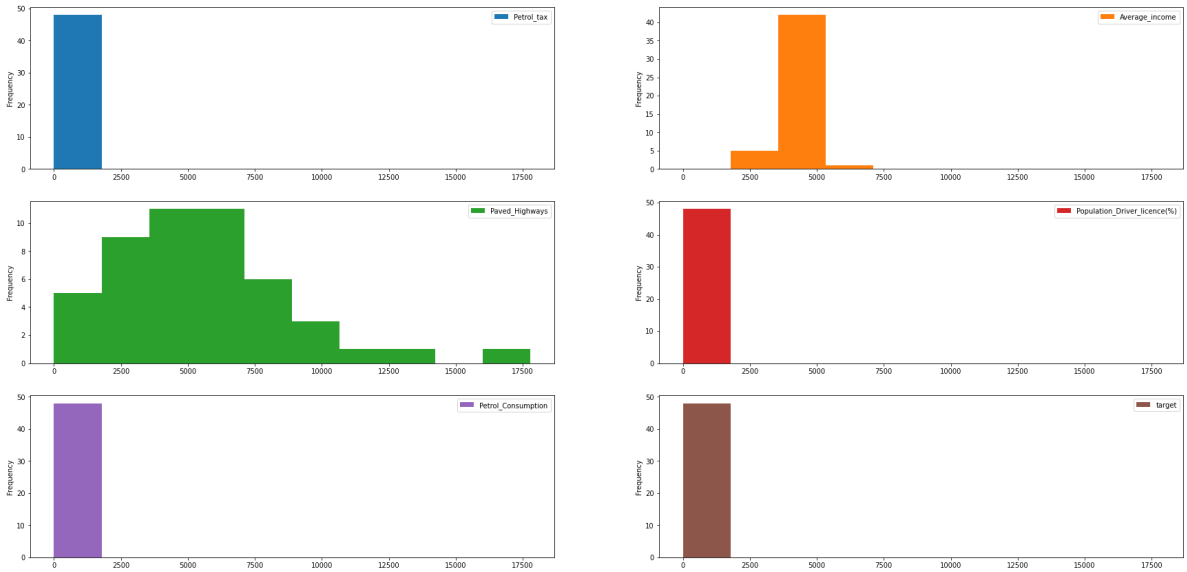
```
0
```

In [117]:

```
plt.rcParams['figure.figsize']=(30,25)

df.plot(kind='hist', bins=10, subplots=True, layout=(5,2), sharex=False, sharey=False)

plt.show()
```



In [118]:

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

y = df['target']
```

In [119]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0
```

In [120]:

```
X_train.shape, X_test.shape
```

Out[120]:

```
((38, 5), (10, 5))
```

In [121]:

```
cols = X_train.columns
```

In [122]:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)
```

In [123]:

```
X_train = pd.DataFrame(X_train, columns=[cols])
```

In [124]:

```
X_test = pd.DataFrame(X_test, columns=[cols])
```

In [125]:

```
X_train.head()
```

Out[125]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumpt
0	-0.132526	1.505774	2.258546	-0.915791	-0.965
1	-0.628193	-1.833670	0.212334	-1.128527	-0.256
2	-0.132526	-1.788970	-0.454195	-0.525775	0.376
3	-0.132526	-0.878213	0.877245	0.041521	0.401
4	-0.628193	1.274825	1.074805	0.289713	-0.512

In [126]:

```
from sklearn.neighbors import KNeighborsClassifier
```

In [127]:

```
knn = KNeighborsClassifier(n_neighbors=3)
```

In [128]:

```
knn.fit(X_train, y_train)
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                     weights='uniform')
```

Out[128]:

```
KNeighborsClassifier(n_neighbors=3)
```

In [131]:

```
y_pred = knn.predict(X_test)

y_pred
```

Out[131]:

```
array([1, 0, 0, 0, 0, 1, 1, 0, 0, 0], dtype=int64)
```

In [132]:

```
knn.predict_proba(X_test)[:,0]
```

Out[132]:

```
array([0.33333333, 1.        , 0.66666667, 0.66666667, 0.66666667,
       0.        , 0.33333333, 1.        , 0.66666667, 0.66666667])
```

In [134]:

```
from sklearn.metrics import accuracy_score

print('Model accuracy score: {0:0.2f}'. format(accuracy_score(y_test, y_pred)))
```

```
Model accuracy score: 0.40
```

In [ ]:

```

```

NAIVE BAYESIAN ¶

In [19]:

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

In [33]:

```
df=pd.read_csv('C:/Users/USER/Desktop/petrol_consumption.csv')
df.head()
```

Out[33]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumpt
0	9.0	3571	1976	0.525	
1	9.0	4092	1250	0.572	
2	9.0	3865	1586	0.580	
3	7.5	4870	2351	0.529	
4	8.0	4399	431	0.544	

In [10]:

```
print(df.shape)
```

```
(48, 6)
```

In [22]:

```
X = df.iloc[:, [1,2,3,4]].values
y = df.iloc[:, -1].values
```

In [23]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
X[:,0] = le.fit_transform(X[:,0])
```

In [24]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state =
```

In [25]:

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

In [26]:

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

Out[26]:

```
GaussianNB()
```

In [27]:

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

In [28]:

```
 y_pred
```

Out[28]:

```
array([1, 1, 1, 1, 1, 0, 1, 1, 1, 1], dtype=int64)
```

In [29]:

```
y_test
```

Out[29]:

```
array([0, 1, 0, 1, 1, 1, 1, 1, 0, 1], dtype=int64)
```

In [30]:

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

In [32]:

```
print("Accuracy=",ac*100)
```

```
Accuracy= 60.0
```

In [ ]:

```

```

SUPPORT VECTOR MACHINE(SVM)

In [27]:

```
import numpy as np
import pandas as pd
from sklearn.datasets import make_classification
from sklearn import svm
from sklearn.model_selection import train_test_split
```

In [28]:

```
df=pd.read_csv('C:/Users/USER/Desktop/petrol_consumption.csv')
df
```

Out[28]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consump
0	9.00	3571	1976	0.525	
1	9.00	4092	1250	0.572	
2	9.00	3865	1586	0.580	
3	7.50	4870	2351	0.529	
4	8.00	4399	431	0.544	
5	10.00	5342	1333	0.571	
6	8.00	5319	11868	0.451	
7	8.00	5126	2138	0.553	
8	8.00	4447	8577	0.529	
9	7.00	4512	8507	0.552	
10	8.00	4391	5939	0.530	
11	7.50	5126	14186	0.525	
12	7.00	4817	6930	0.574	
13	7.00	4207	6580	0.545	
14	7.00	4332	8159	0.608	
15	7.00	4318	10340	0.586	
16	7.00	4206	8508	0.572	
17	7.00	3718	4725	0.540	
18	7.00	4716	5915	0.724	
19	8.50	4341	6010	0.677	
20	7.00	4593	7834	0.663	
21	8.00	4983	602	0.602	
22	9.00	4897	2449	0.511	
23	9.00	4258	4686	0.517	
24	8.50	4574	2619	0.551	
25	9.00	3721	4746	0.544	
26	8.00	3448	5399	0.548	
27	7.50	3846	9061	0.579	
28	8.00	4188	5975	0.563	
29	9.00	3601	4650	0.493	
30	7.00	3640	6905	0.518	
31	7.00	3333	6594	0.513	
32	8.00	3063	6524	0.578	
33	7.50	3357	4121	0.547	

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consump
34	8.00	3528	3495	0.487	
35	6.58	3802	7834	0.629	
36	5.00	4045	17782	0.566	
37	7.00	3897	6385	0.586	
38	8.50	3635	3274	0.663	
39	7.00	4345	3905	0.672	
40	7.00	4449	4639	0.626	
41	7.00	3656	3985	0.563	
42	7.00	4300	3635	0.603	
43	7.00	3745	2611	0.508	
44	6.00	5215	2302	0.672	
45	9.00	4476	3942	0.571	
46	7.00	4296	4083	0.623	
47	7.00	5002	9794	0.593	

In [29]:

```
classes = 4
X,t= make_classification(100, 5, n_classes = classes, random_state= 40, n_informative = 2,
```

In [30]:

```
X_train, X_test, y_train, y_test=  train_test_split(X, t , test_size=0.50)
```

In [31]:

```
model = svm.SVC(kernel = 'linear', random_state = 0, C=1.0)
```

In [32]:

```
model.fit(X_train, y_train)
```

Out[32]:

```
SVC(kernel='linear', random_state=0)
```

In [33]:

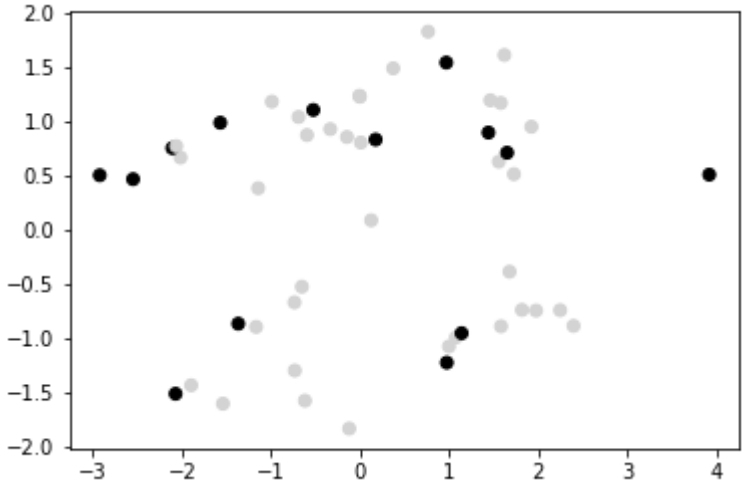
```
y=model.predict(X_test)
y2=model.predict(X_train)
```

In [34]:

```
import matplotlib.pyplot as plt
color = ['black' if c == 0 else 'lightgrey' for c in y]
plt.scatter(X_train[:,0], X_train[:,1], c=color)
```

Out[34]:

```
<matplotlib.collections.PathCollection at 0x17a00341490>
```



In [11]:

```
from sklearn.metrics import accuracy_score
score =accuracy_score(y, y_test)
print(score*100)
score2 =accuracy_score(y2, y_train)
print(score2*100)
```

```
84.0
96.0
```