Experiment: 3

Student Name: Animesh Kumar **UID:22BCS13257**

Branch: Computer Science & Engineering **Section/Group:22CSE-212/C**

Semester: 1st **Date of Performance:09/11/2022**

Subject Name: Disruptive Technology-1

Subject Code: 22ECH-102

1. **Aim of the practical:** Explore,visualize,transform and summarize input datasets for building Classificatio/regression/prediction models.
2. **Tool Used:** Google Colaboratory
3. **Basic Concept/ Command Description:**

- It is a bundle of many Machine Learning algorithms.
- Only three lines of code is required to compare 20 ML models.
- Pycaret is available for:
  - Classification
  - Regression
  - Clustering

4. **Code:**

(a) Install Pycaret

Exclamation sign: It means run it as a shell command rather than a notebook command. This is actually not specific to pip, but really any shell command from the iPython notebook. In computing, a shell is a computer program which exposes an operating system's services to a human user or other program.

&> /dev/null: /dev/null is the null file. Anything written to it is discarded.

Together they mean "throw away any error messages".

```
!pip install pycaret &> /dev/null
print ("Pycaret installed sucessfully!!")
```

```
Pycaret installed sucessfully!!
```

(b) Get the version of the pycaret

```
[3] ## Utils is a collection of small Python functions and classes which make common patterns shorter and easier.
from pycaret.utils import version
version()
```

```
'2.3.10'
```

1. Classification: Basics

1.1 Loading Dataset - Loading dataset from pycaret

```
[4] from pycaret.datasets import get_data

# No output
```

1.2 Get the list of datasets available in pycaret (55)

1.2 Get the list of datasets available in pycaret (55)

```python
# Internet connection is required
dataSets = get_data('index')
```

	Dataset	Data Types	Default Task	Target Variable 1	Target Variable 2	# Instances	# Attributes	Missing Values
0	anomaly	Multivariate	Anomaly Detection	None	None	1000	10	N
1	france	Multivariate	Association Rule Mining	InvoiceNo	Description	8557	8	N
2	germany	Multivariate	Association Rule Mining	InvoiceNo	Description	9495	8	N
3	bank	Multivariate	Classification (Binary)	deposit	None	45211	17	N
4	blood	Multivariate	Classification (Binary)	Class	None	748	5	N
5	cancer	Multivariate	Classification (Binary)	Class	None	683	10	N
6	credit	Multivariate	Classification (Binary)	default	None	24000	24	N
7	diabetes	Multivariate	Classification (Binary)	Class variable	None	768	9	N
8	electrical_grid	Multivariate	Classification (Binary)	stabf	None	10000	14	N
9	employee	Multivariate	Classification (Binary)	left	None	14999	10	N
10	heart	Multivariate	Classification (Binary)	DEATH	None	200	16	N
11	heart_disease	Multivariate	Classification (Binary)	Disease	None	270	14	N
12	hepatitis	Multivariate	Classification (Binary)	Class	None	154	32	Y
13	income	Multivariate	Classification (Binary)	income >50K	None	32561	14	Y
14	juice	Multivariate	Classification (Binary)	Purchase	None	1070	15	N

1.3 Get juice dataset

```python
juiceDataSet = get_data("juice")    # SN is 14
# This is binary classification dataset. The values in "Purchase" have two (binary) values.
print(type(juiceDataSet))
```

	Id	Purchase	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	SpecialMM	LoyalCH	SalePriceMM	SalePriceCH	PriceDiff	Store7	PctDiscMM	PctDiscCH	ListPriceDiff	STORE
0	1	CH	237	1	1.75	1.99	0.00	0.0	0	0	0.500000	1.99	1.75	0.24	No	0.000000	0.000000	0.24	1
1	2	CH	239	1	1.75	1.99	0.00	0.3	0	1	0.600000	1.69	1.75	-0.06	No	0.150754	0.000000	0.24	1
2	3	CH	245	1	1.86	2.09	0.17	0.0	0	0	0.680000	2.09	1.69	0.40	No	0.000000	0.091398	0.23	1
3	4	MM	227	1	1.69	1.69	0.00	0.0	0	0	0.400000	1.69	1.69	0.00	No	0.000000	0.000000	0.00	1
4	5	CH	228	7	1.69	1.69	0.00	0.0	0	0	0.956535	1.69	1.69	0.00	Yes	0.000000	0.000000	0.00	0

```
<class 'pandas.core.frame.DataFrame'>
```

```python
juiceDataSet.columns
```

```
Index(['Id', 'Purchase', 'WeekofPurchase', 'StoreID', 'PriceCH', 'PriceMM',
       'DiscCH', 'DiscMM', 'SpecialCH', 'SpecialMM', 'LoyalCH', 'SalePriceMM',
       'SalePriceCH', 'PriceDiff', 'Store7', 'PctDiscMM', 'PctDiscCH',
       'ListPriceDiff', 'STORE'],
      dtype='object')
```

```python
[17] #Get the statistical summary of the dataset
juiceDataSet.describe()
```

	Id	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	SpecialMM	LoyalCH	SalePriceMM	SalePriceCH	PriceDiff	PctDiscMM	PctDiscCH	ListPriceDiff	STORE
count	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000	1070.000000
mean	535.500000	254.381308	3.959813	1.867421	2.085411	0.051860	0.123364	0.147664	0.161682	0.565782	1.962047	1.815561	0.146486	0.059298	0.027314	0.217991	1.630841
std	309.026698	15.558286	2.308984	0.101970	0.134386	0.117474	0.213834	0.354932	0.368331	0.307843	0.252697	0.143384	0.271563	0.101760	0.062232	0.107535	1.430387
min	1.000000	227.000000	1.000000	1.690000	1.690000	0.000000	0.000000	0.000000	0.000000	0.000011	1.190000	1.390000	-0.670000	0.000000	0.000000	0.000000	0.000000
25%	268.250000	240.000000	2.000000	1.790000	1.990000	0.000000	0.000000	0.000000	0.000000	0.325257	1.690000	1.750000	0.000000	0.000000	0.000000	0.140000	0.000000
50%	535.500000	257.000000	3.000000	1.860000	2.090000	0.000000	0.000000	0.000000	0.000000	0.600000	2.090000	1.860000	0.230000	0.000000	0.000000	0.240000	2.000000
75%	802.750000	268.000000	7.000000	1.990000	2.180000	0.000000	0.230000	0.000000	0.000000	0.850873	2.130000	1.890000	0.320000	0.112676	0.000000	0.300000	3.000000
max	1070.000000	278.000000	7.000000	2.090000	2.290000	0.500000	0.800000	1.000000	1.000000	0.999947	2.290000	2.090000	0.640000	0.402010	0.252688	0.440000	4.000000

```python
[13] print("type(juiceDataSet)-->",type(juiceDataSet))
```

```
type(juiceDataSet)--> <class 'pandas.core.frame.DataFrame'>
```

```python
##Get the dimention of the dataset
```

```python
[14] print("juiceDataSet.shape -->", juiceDataSet.shape)
print("Rows     -->", juiceDataSet.shape[0])  ##axis 0---row
print("Columns  -->", juiceDataSet.shape[1])   ###column
```

```
juiceDataSet.shape --> (1070, 19)
Rows     --> 1070
Columns  --> 19
```

```python
[ ] ### Show top 5 rows of the dataset
```

```python
[16] juiceDataSet.head()
```

	Id	Purchase	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	SpecialMM	LoyalCH	SalePriceMM	SalePriceCH	PriceDiff	Store7	PctDiscMM	PctDiscCH	ListPriceDiff	STORE
0	1	CH	237	1	1.75	1.99	0.00	0.0	0	0	0.500000	1.99	1.75	0.24	No	0.000000	0.000000	0.24	1
1	2	CH	239	1	1.75	1.99	0.00	0.3	0	1	0.600000	1.69	1.75	-0.06	No	0.150754	0.000000	0.24	1
2	3	CH	245	1	1.86	2.09	0.17	0.0	0	0	0.680000	2.09	1.69	0.40	No	0.000000	0.091398	0.23	1
3	4	MM	227	1	1.69	1.69	0.00	0.0	0	0	0.400000	1.69	1.69	0.00	No	0.000000	0.000000	0.00	1
4	5	CH	228	7	1.69	1.69	0.00	0.0	0	0	0.956535	1.69	1.69	0.00	Yes	0.000000	0.000000	0.00	0

```python
[24] ## Get the maximum of each column in the dataset
juiceDataSet.max()
```

```
Id                    1070
Purchase                MM
WeekofPurchase         278
StoreID                  7
PriceCH               2.09
PriceMM               2.29
DiscCH                 0.5
DiscMM                 0.8
SpecialCH                1
SpecialMM                1
LoyalCH           0.999947
SalePriceMM           2.29
SalePriceCH           2.09
PriceDiff             0.64
Store7                 Yes
```

```python
juiceDataSet.isnull().sum()
#diabetesDataSet.dropna()
```

```
Id                0
Purchase          0
WeekofPurchase    0
StoreID           0
PriceCH           0
PriceMM           0
DiscCH            0
DiscMM            0
SpecialCH         0
SpecialMM         0
LoyalCH           0
SalePriceMM       0
SalePriceCH       0
PriceDiff         0
Store7            0
PctDiscMM         0
PctDiscCH         0
ListPriceDiff     0
STORE             0
dtype: int64
```

3. Classification

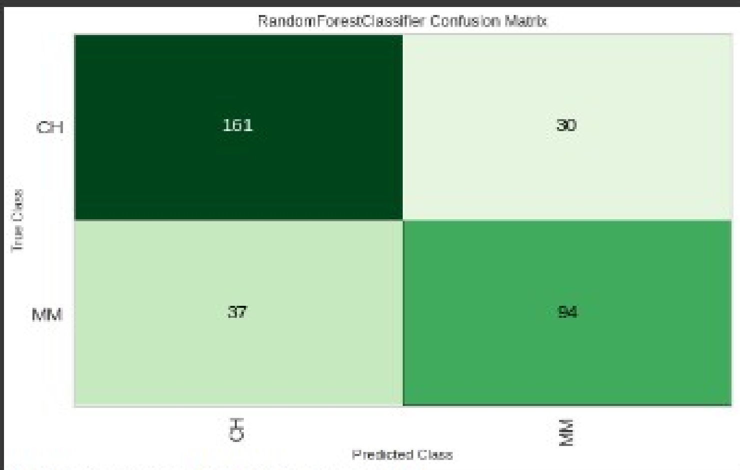
If you don't want PyCaret to display the dialogue for confirmation of data types you may pass silent as True within setup to perform a unattended run of experiment.

3.1 Build a single model - "RandomForest"

```
#from pycaret.datasets import get_data
from pycaret.classification import *

#diabetesDataSet = get_data("diabetes")
s = setup(data=juiceDataSet, target='Purchase', silent=True)

rfModel = create_model('rf')
plot_model(rfModel, plot='confusion_matrix')
#Explore more parameters
```




```
INFO:logs:Visual Rendered Successfully
INFO:logs:plot_model() succesfully completed......................................
```

3.3 Save the trained model

```
[ ] sm = save_model(rfModel, 'rfModelFile')
```

```
Transformation Pipeline and Model Successfully Saved
```

3.5 Make prediction on the new dataset

Get new dataset

```
[10] # Select top 10 rows from juice dataset
     newDataSet = get_data("juice").iloc[:10]
```

	Id	Purchase	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	SpecialMM	LoyalCH	SalePriceMM	SalePriceCH	PriceDiff	Store7	PctDiscMM	PctDiscCH	ListPriceDiff	STORE
0	1	CH	237	1	1.75	1.99	0.00	0.0	0	0	0.500000	1.99	1.75	0.24	No	0.000000	0.000000	0.24	1
1	2	CH	239	1	1.75	1.99	0.00	0.3	0	1	0.600000	1.69	1.75	-0.06	No	0.150754	0.000000	0.24	1
2	3	CH	245	1	1.86	2.09	0.17	0.0	0	0	0.680000	2.09	1.69	0.40	No	0.000000	0.091398	0.23	1
3	4	MM	227	1	1.69	1.69	0.00	0.0	0	0	0.400000	1.69	1.69	0.00	No	0.000000	0.000000	0.00	1
4	5	CH	228	7	1.69	1.69	0.00	0.0	0	0	0.956535	1.69	1.69	0.00	Yes	0.000000	0.000000	0.00	0

Make prediction on new dataset

```
[13] newPredictions = predict_model(rfModel, data = newDataSet)
     newPredictions
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Random Forest Classifier	0	1.0	0	0	0	0	0

	Id	Purchase	WeekofPurchase	StoreID	PriceCH	PriceMM	DiscCH	DiscMM	SpecialCH	SpecialMM	...	SalePriceMM	SalePriceCH	PriceDiff	Store7	PctDiscMM	PctDiscCH	ListPriceDiff	STORE	Label	Score
0	1	CH	237	1	1.75	1.99	0.00	0.0	0	0	...	1.99	1.75	0.24	No	0.000000	0.000000	0.24	1	MM	0.63
1	2	CH	239	1	1.75	1.99	0.00	0.3	0	1	...	1.69	1.75	-0.06	No	0.150754	0.000000	0.24	1	CH	0.90
2	3	CH	245	1	1.86	2.09	0.17	0.0	0	0	...	2.09	1.69	0.40	No	0.000000	0.091398	0.23	1	CH	0.90
3	4	MM	227	1	1.69	1.69	0.00	0.0	0	0	...	1.69	1.69	0.00	No	0.000000	0.000000	0.00	1	MM	0.99
4	5	CH	228	7	1.69	1.69	0.00	0.0	0	0	...	1.69	1.69	0.00	Yes	0.000000	0.000000	0.00	0	CH	0.93
5	6	CH	230	7	1.69	1.99	0.00	0.0	0	1	...	1.99	1.69	0.30	Yes	0.000000	0.000000	0.30	0	CH	0.91
6	7	CH	232	7	1.69	1.99	0.00	0.4	1	1	...	1.59	1.69	-0.10	Yes	0.201005	0.000000	0.30	0	CH	0.87
7	8	CH	234	7	1.75	1.99	0.00	0.4	1	0	...	1.59	1.75	-0.16	Yes	0.201005	0.000000	0.24	0	CH	0.73
8	9	CH	235	7	1.75	1.99	0.00	0.4	0	0	...	1.59	1.75	-0.16	Yes	0.201005	0.000000	0.24	0	CH	1.00
9	10	CH	238	7	1.75	1.99	0.00	0.4	0	0	...	1.59	1.75	-0.16	Yes	0.201005	0.000000	0.24	0	CH	0.99

10 rows × 21 columns

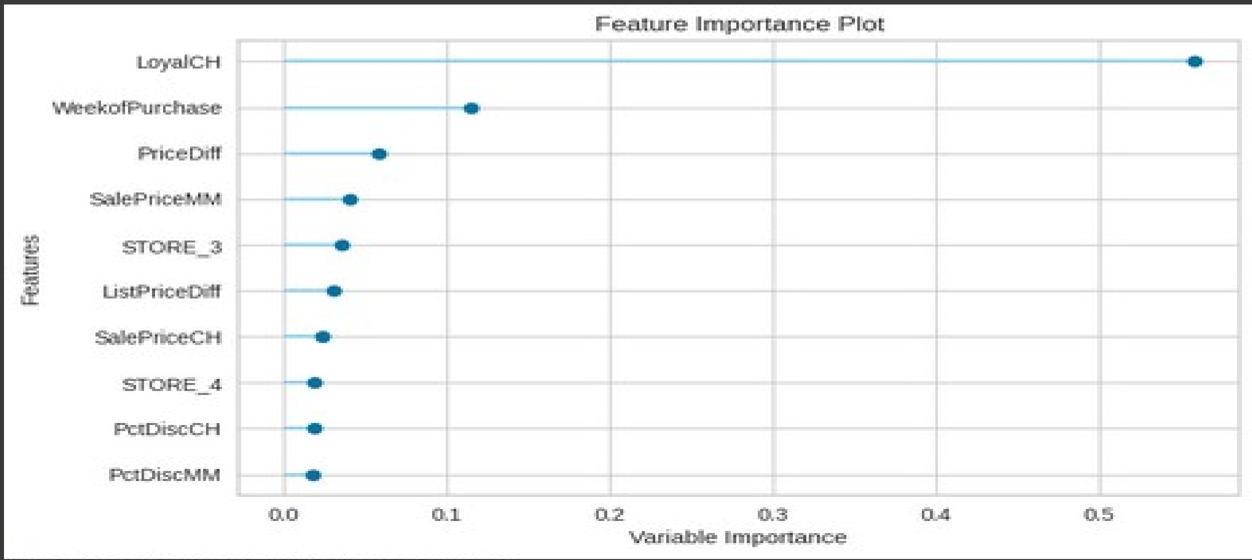
3.6 Save prediction results to csv

```python
newPredictions.to_csv("NewPredictions.csv")
# No output
```

3.8 Feature Importance

3.8.1 Feature Importance using Random Forest

```python
[15] rfModel = create_model('rf', verbose=True)
     plot_model(rfModel, plot='feature')
```




```
INFO:logs:Visual Rendered Successfully
```

1.4 Parameter setting for all classification models

- Train/Test division
- Sampling
- Normalization
- Transformation
- PCA (Dimention Reduction)
- Handaling of Outliers
- Feature Selection

```python
from pycaret.classification import *
s = setup(data=juiceDataSet, target='Purchase', silent=True)
```

	Description	Value
0	session_id	2083
1	Target	Purchase
2	Target Type	Binary
3	Label Encoded	CH: 0, MM: 1
4	Original Data	(1070, 19)
5	Missing Values	False
6	Numeric Features	13
7	Categorical Features	5
8	Ordinal Features	False
9	High Cardinality Features	False

1.5 Run and compare the Model Performance

```
cm = compare_models()
# Explore more parameters
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ridge	Ridge Classifier	0.8383	0.0000	0.7908	0.8013	0.7931	0.6608	0.6641	0.026
lda	Linear Discriminant Analysis	0.8343	0.8986	0.7874	0.7949	0.7881	0.6525	0.6560	0.034
lr	Logistic Regression	0.8302	0.8982	0.7638	0.8005	0.7788	0.6417	0.6452	0.918
gbc	Gradient Boosting Classifier	0.8142	0.8943	0.7570	0.7725	0.7613	0.6097	0.6135	0.255
ada	Ada Boost Classifier	0.8061	0.8802	0.7329	0.7736	0.7483	0.5914	0.5965	0.229
lightgbm	Light Gradient Boosting Machine	0.7981	0.8766	0.7434	0.7503	0.7443	0.5779	0.5807	0.256
rf	Random Forest Classifier	0.7889	0.8701	0.7332	0.7339	0.7316	0.5579	0.5601	0.678
dt	Decision Tree Classifier	0.7781	0.7707	0.7089	0.7335	0.7157	0.5343	0.5395	0.034
nb	Naive Bayes	0.7647	0.8334	0.7634	0.6799	0.7169	0.5172	0.5227	0.027
et	Extra Trees Classifier	0.7608	0.8410	0.7098	0.6929	0.6977	0.5004	0.5048	0.502
knn	K Neighbors Classifier	0.7126	0.7448	0.5933	0.6572	0.6192	0.3899	0.3948	0.157
svm	SVM - Linear Kernel	0.6373	0.0000	0.3336	0.4274	0.3259	0.1787	0.1985	0.033
dummy	Dummy Classifier	0.6056	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.039
qda	Quadratic Discriminant Analysis	0.4854	0.5133	0.6420	0.3666	0.4577	0.0246	0.0220	0.054

2. Classification: Advance - 1

2.1 Model Performance using data "Normalization"

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

```
## Commonly used techniques: clipping, log scaling, z-score, minmax, maxabs, robust
s = setup(data=juiceDataSet, target='Purchase', normalize = True, normalize_method = 'zscore', silent=True)
cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ridge	Ridge Classifier	0.8261	0.0000	0.7557	0.8060	0.7787	0.6358	0.6380	0.018
lda	Linear Discriminant Analysis	0.8221	0.8893	0.7559	0.7967	0.7749	0.6281	0.6296	0.020
lr	Logistic Regression	0.8208	0.8907	0.7491	0.7990	0.7721	0.6248	0.6268	0.054
gbc	Gradient Boosting Classifier	0.8140	0.8902	0.7523	0.7819	0.7648	0.6114	0.6137	0.255
ada	Ada Boost Classifier	0.8020	0.8732	0.7225	0.7772	0.7457	0.5842	0.5883	0.172
rf	Random Forest Classifier	0.7915	0.8707	0.7495	0.7413	0.7436	0.5681	0.5700	0.671
lightgbm	Light Gradient Boosting Machine	0.7913	0.8727	0.7391	0.7520	0.7417	0.5671	0.5712	0.089
knn	K Neighbors Classifier	0.7901	0.8451	0.7066	0.7613	0.7316	0.5599	0.5621	0.147
dt	Decision Tree Classifier	0.7739	0.7711	0.7259	0.7214	0.7226	0.5319	0.5331	0.018
et	Extra Trees Classifier	0.7674	0.8346	0.7133	0.7184	0.7134	0.5179	0.5205	0.599
nb	Naive Bayes	0.7566	0.8290	0.7625	0.6784	0.7168	0.5049	0.5092	0.018
svm	SVM - Linear Kernel	0.7566	0.0000	0.6918	0.7110	0.6860	0.4899	0.5032	0.020
qda	Quadratic Discriminant Analysis	0.6244	0.6128	0.4863	0.4435	0.4405	0.1990	0.2079	0.027
dummy	Dummy Classifier	0.5949	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.023

2.2 Model Performance using "Feature Selection"

Feature Selection is one of the core concepts in machine learning which hugely impacts the performance of your model. The data features that you use to train your machine learning models have a huge influence on the performance you can achieve. The goal of feature selection in machine learning is to find the best set of features that allows one to build useful models of studied phenomena. Threshold used for feature selection (including newly created polynomial features). A higher value will result in a higher feature space. It is recommended to do multiple trials with different values of feature_selection_threshold.

```
s = setup(data=juiceDataSet, target='Purchase', feature_selection = True, feature_selection_threshold = 0.6, silent=True)
cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.8274	0.8985	0.7268	0.7955	0.7582	0.6248	0.6274	0.106
ridge	Ridge Classifier	0.8234	0.0000	0.7304	0.7863	0.7554	0.6178	0.6205	0.024
lda	Linear Discriminant Analysis	0.8207	0.8990	0.7339	0.7775	0.7525	0.6126	0.6154	0.022
ada	Ada Boost Classifier	0.8088	0.8878	0.6990	0.7764	0.7325	0.5847	0.5896	0.133
gbc	Gradient Boosting Classifier	0.8047	0.8913	0.7339	0.7454	0.7357	0.5814	0.5851	0.142
rf	Random Forest Classifier	0.7900	0.8641	0.7342	0.7155	0.7224	0.5539	0.5563	0.556
lightgbm	Light Gradient Boosting Machine	0.7821	0.8724	0.7057	0.7200	0.7087	0.5351	0.5390	0.068
qda	Quadratic Discriminant Analysis	0.7807	0.8492	0.8264	0.6734	0.7411	0.5550	0.5647	0.037
nb	Naive Bayes	0.7754	0.8439	0.7555	0.6850	0.7164	0.5316	0.5358	0.016
dt	Decision Tree Classifier	0.7742	0.7650	0.6952	0.7134	0.6994	0.5191	0.5236	0.023
et	Extra Trees Classifier	0.7701	0.8324	0.6956	0.6970	0.6948	0.5105	0.5119	0.476
knn	K Neighbors Classifier	0.7380	0.7768	0.5691	0.6896	0.6208	0.4239	0.4304	0.136
dummy	Dummy Classifier	0.6217	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.014
svm	SVM - Linear Kernel	0.5737	0.0000	0.2000	0.0760	0.1101	0.0000	0.0000	0.026

2.3 Model Performance using "Outlier Removal"

Sometimes a dataset can contain extreme values that are outside the range of what is expected and unlike the other data. These are called outliers and often machine learning modeling and model skill in general can be improved by understanding and even removing these outlier values. outliers_threshold = 0.05 is the default value.

```
[13] s = setup(data=juiceDataSet, target='Purchase', remove_outliers = True, outliers_threshold = 0.05, silent=True)
     cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
ridge	Ridge Classifier	0.8352	0.0000	0.7757	0.8039	0.7875	0.6530	0.6556	0.016
lda	Linear Discriminant Analysis	0.8352	0.9014	0.7793	0.8010	0.7882	0.6534	0.6556	0.022
gbc	Gradient Boosting Classifier	0.8324	0.9044	0.7757	0.7971	0.7846	0.6475	0.6496	0.146
lr	Logistic Regression	0.8268	0.9012	0.7509	0.8014	0.7728	0.6332	0.6367	0.140
lightgbm	Light Gradient Boosting Machine	0.8268	0.8872	0.7687	0.7910	0.7782	0.6361	0.6380	0.098
ada	Ada Boost Classifier	0.8225	0.8881	0.7546	0.7903	0.7697	0.6258	0.6284	0.127
rf	Random Forest Classifier	0.8070	0.8834	0.7405	0.7675	0.7520	0.5942	0.5964	0.528
et	Extra Trees Classifier	0.7803	0.8453	0.7085	0.7318	0.7162	0.5375	0.5411	0.475
dt	Decision Tree Classifier	0.7676	0.7634	0.6911	0.7165	0.7005	0.5112	0.5144	0.047
nb	Naive Bayes	0.7592	0.8388	0.7613	0.6778	0.7132	0.5071	0.5147	0.018
knn	K Neighbors Classifier	0.7183	0.7914	0.5956	0.6611	0.6241	0.4004	0.4034	0.121
dummy	Dummy Classifier	0.6028	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.018
svm	SVM - Linear Kernel	0.5239	0.0000	0.4000	0.1606	0.2291	0.0000	0.0000	0.023
qda	Quadratic Discriminant Analysis	0.4437	0.5222	0.9071	0.4177	0.5657	0.0422	0.0427	0.020

2.4 Model Performance using "Transformation"

Data transformation is the process in which you take data from its raw, siloed and normalized source state and transform it into data that's joined together, dimensionally modeled, de-normalized, and ready for analysis

[14]
```
s = setup(data=juiceDataSet, target='Purchase', transformation = True, transformation_method = 'yeo-johnson', silent=True)
cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.8223	0.8933	0.7659	0.7839	0.7732	0.6273	0.6292	0.034
ridge	Ridge Classifier	0.8089	0.0000	0.7623	0.7606	0.7592	0.6012	0.6036	0.017
lda	Linear Discriminant Analysis	0.8075	0.8922	0.7623	0.7580	0.7579	0.5986	0.6011	0.021
gbc	Gradient Boosting Classifier	0.8022	0.8950	0.7383	0.7615	0.7479	0.5852	0.5873	0.151
ada	Ada Boost Classifier	0.7995	0.8783	0.7141	0.7638	0.7357	0.5747	0.5778	0.131
svm	SVM - Linear Kernel	0.7914	0.0000	0.7420	0.7430	0.7359	0.5647	0.5720	0.022
lightgbm	Light Gradient Boosting Machine	0.7834	0.8756	0.7177	0.7324	0.7234	0.5456	0.5472	0.072
rf	Random Forest Classifier	0.7808	0.8560	0.7216	0.7242	0.7220	0.5413	0.5422	0.516
knn	K Neighbors Classifier	0.7755	0.8399	0.6976	0.7287	0.7102	0.5274	0.5304	0.122
dt	Decision Tree Classifier	0.7688	0.7678	0.7110	0.7066	0.7066	0.5162	0.5182	0.019
et	Extra Trees Classifier	0.7595	0.8171	0.6870	0.6975	0.6906	0.4941	0.4953	0.468
nb	Naive Bayes	0.7527	0.8317	0.7621	0.6691	0.7096	0.4969	0.5031	0.020
qda	Quadratic Discriminant Analysis	0.6954	0.7612	0.7255	0.6275	0.6518	0.3940	0.4027	0.017
dummy	Dummy Classifier	0.6070	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.016

2.5 Model Performance using "PCA"

An important machine learning method for dimensionality reduction is called Principal Component Analysis. It is a method that uses simple matrix operations from linear algebra and statistics to calculate a projection of the original data into the same number or fewer dimensions.

```
s = setup(data=juiceDataSet, target='Purchase', pca = True, pca_method = 'linear', silent=True)
cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
knn	K Neighbors Classifier	0.6657	0.6808	0.4690	0.5898	0.5183	0.2686	0.2749	0.122
dt	Decision Tree Classifier	0.6629	0.6371	0.5345	0.5702	0.5495	0.2813	0.2832	0.018
rf	Random Forest Classifier	0.6616	0.6922	0.5345	0.5680	0.5483	0.2788	0.2808	0.528
et	Extra Trees Classifier	0.6616	0.6484	0.5345	0.5695	0.5490	0.2792	0.2813	0.470
lightgbm	Light Gradient Boosting Machine	0.6457	0.6455	0.4276	0.5622	0.4807	0.2208	0.2280	0.050
gbc	Gradient Boosting Classifier	0.6430	0.6502	0.3621	0.5676	0.4341	0.1947	0.2083	0.089
ada	Ada Boost Classifier	0.6390	0.6324	0.3379	0.5665	0.4159	0.1800	0.1954	0.109
ridge	Ridge Classifier	0.6136	0.0000	0.0138	0.1400	0.0247	0.0088	0.0216	0.017
dummy	Dummy Classifier	0.6123	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.016
lr	Logistic Regression	0.6056	0.5686	0.0138	0.0900	0.0239	-0.0069	-0.0126	0.038
lda	Linear Discriminant Analysis	0.6043	0.5686	0.0138	0.0833	0.0235	-0.0094	-0.0166	0.016
nb	Naive Bayes	0.6030	0.5672	0.0931	0.4411	0.1499	0.0228	0.0434	0.017
qda	Quadratic Discriminant Analysis	0.6030	0.5672	0.0931	0.4411	0.1499	0.0228	0.0434	0.016
svm	SVM - Linear Kernel	0.5627	0.0000	0.3276	0.3501	0.3291	0.0426	0.0435	0.021

2.6 Model Performance using "Outlier Removal" + "Normalization"

```
[17] s = setup(data=juiceDataSet, target='Purchase', remove_outliers = True, outliers_threshold = 0.05, normalize = True, normalize_method = 'zscore', silent=True)
     cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.8169	0.8916	0.7521	0.7757	0.7625	0.6137	0.6152	0.034
ridge	Ridge Classifier	0.8141	0.0000	0.7591	0.7667	0.7616	0.6094	0.6107	0.019
lda	Linear Discriminant Analysis	0.8141	0.8927	0.7591	0.7676	0.7616	0.6094	0.6113	0.022
gbc	Gradient Boosting Classifier	0.8127	0.8801	0.7488	0.7705	0.7561	0.6047	0.6082	0.151
ada	Ada Boost Classifier	0.8014	0.8697	0.7234	0.7604	0.7389	0.5793	0.5819	0.111
rf	Random Forest Classifier	0.7887	0.8491	0.7017	0.7494	0.7185	0.5510	0.5564	0.516
knn	K Neighbors Classifier	0.7817	0.8349	0.6803	0.7464	0.7067	0.5345	0.5398	0.117
lightgbm	Light Gradient Boosting Machine	0.7789	0.8697	0.7093	0.7250	0.7129	0.5339	0.5376	0.066
nb	Naive Bayes	0.7535	0.8243	0.7735	0.6595	0.7106	0.4986	0.5051	0.018
et	Extra Trees Classifier	0.7521	0.8175	0.6660	0.6896	0.6714	0.4745	0.4782	0.523
svm	SVM - Linear Kernel	0.7507	0.0000	0.6827	0.6934	0.6712	0.4737	0.4848	0.017
dt	Decision Tree Classifier	0.7423	0.7324	0.6733	0.6740	0.6700	0.4594	0.4623	0.019
qda	Quadratic Discriminant Analysis	0.6451	0.6828	0.5193	0.5144	0.4745	0.2408	0.2645	0.023
dummy	Dummy Classifier	0.6085	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.019

2.7 Model Performance using "Outlier Removal" + "Normalization" + "Transformation"

```
[18] s = setup(data=juiceDataSet, target='Purchase', remove_outliers = True, outliers_threshold = 0.05, normalize = True, normalize_method = 'zscore', transformation = True, transformation_method = 'yeo-johnson', silent=True)
     cm = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lda	Linear Discriminant Analysis	0.8296	0.8940	0.7821	0.7771	0.7773	0.6396	0.6420	0.018
ridge	Ridge Classifier	0.8282	0.0000	0.7784	0.7764	0.7753	0.6364	0.6386	0.041
lr	Logistic Regression	0.8268	0.8958	0.7525	0.7883	0.7679	0.6300	0.6325	0.034
ada	Ada Boost Classifier	0.8141	0.8852	0.7378	0.7675	0.7504	0.6026	0.6047	0.125
gbc	Gradient Boosting Classifier	0.8085	0.8801	0.7343	0.7586	0.7439	0.5912	0.5936	0.150
knn	K Neighbors Classifier	0.7972	0.8390	0.6971	0.7556	0.7222	0.5633	0.5671	0.120
lightgbm	Light Gradient Boosting Machine	0.7930	0.8575	0.7452	0.7229	0.7324	0.5638	0.5656	0.062
rf	Random Forest Classifier	0.7845	0.8578	0.7011	0.7262	0.7099	0.5392	0.5424	0.518
qda	Quadratic Discriminant Analysis	0.7718	0.8376	0.7602	0.6812	0.7178	0.5275	0.5304	0.016
et	Extra Trees Classifier	0.7690	0.8190	0.6754	0.7095	0.6873	0.5053	0.5099	0.483
nb	Naive Bayes	0.7606	0.8379	0.7452	0.6710	0.7045	0.5048	0.5084	0.017
dt	Decision Tree Classifier	0.7592	0.7454	0.6602	0.6972	0.6743	0.4841	0.4881	0.021
svm	SVM - Linear Kernel	0.7592	0.0000	0.7597	0.6650	0.6959	0.4998	0.5145	0.022
dummy	Dummy Classifier	0.6183	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.014

University Institute of Engineering

Department of Computer Science & Engineering

Evaluation Grid (To be filled by Faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	Worksheet completion including writinglearning objectives/Outcomes. (To be submitted at the end of the day)		10
2.	Post Lab Quiz Result.		5
3.	Student Engagement in Simulation/Demonstration/Performance and Controls/Pre-Lab Questions.		5
	Signature of Faculty (with Date):	**Total Marks Obtained:**	**20**