University Institute of Engineering

Department of Computer Science

Experiment-1.3

: **Understand supervised learning to train and devolop classifier models.**

Student Name: UID:

Branch: Computer Science & Engineering Section/Group:

Semester: 1st Semester Date of Performance

Subject Name: Disruptive Technologies-1

Subject Code: 21ECP-102

1. **Aim of the practical:** Understand supervised learning to train and develop classifier models.

2. **Tool Used:** Google Colab

3. **Basic Concept/ Command Description:**

Python is a powerful general-purpose programming language. Python has simple easy-to-use syntax In the experiment performed, the basic concepts and command discussed are as follows:

- Getting Data: How to import data from PyCaret repository o Setting up Environment: How to setup an experiment in PyCaret and get started with building regression models

- Create Model: How to create a model, perform cross validation and evaluate regression metrics
- Tune Model: How to automatically tune the hyper parameters of a regression model
- Plot Model: How to analyze model performance using various plots o Finalize Model: How to finalize the best model at the end of the experiment
- Predict Model: How to make prediction on new / unseen data

  o Save / Load Model: How to save / load a model for future

  use

4. Code: Install Pycaret

```
   !pip install pycaret &> /dev/null

print ("Pycaret installed sucessfully!!")
```

Output:

```
Pycaret installed sucessfully!!
```

Code: Loading Dataset - Loading dataset from pycaret

```
from pycaret.datasets import get_data

# No output
```

Code: Get the list of datasets available in pycaret (55)

```
# Internet connection is required

dataSets = get_data('index')
```

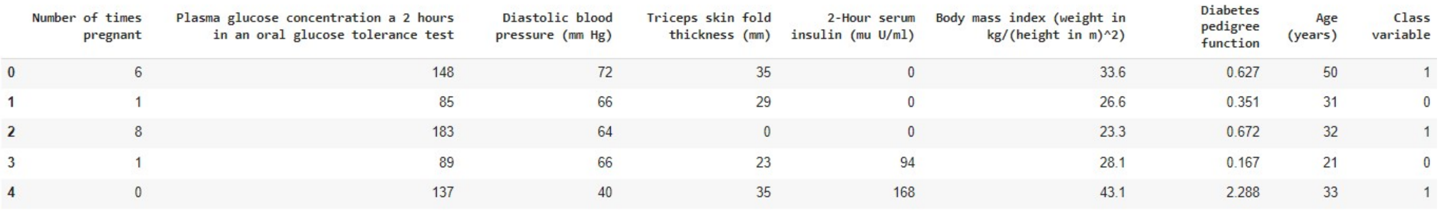
Output:

	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

Code: Get diabetes dataset

```
diabetesDataSet = get_data("diabetes")     # SN is 7# This is binary classification
dataset. The values in "Class variable" have two (binary) values.
```

Output:

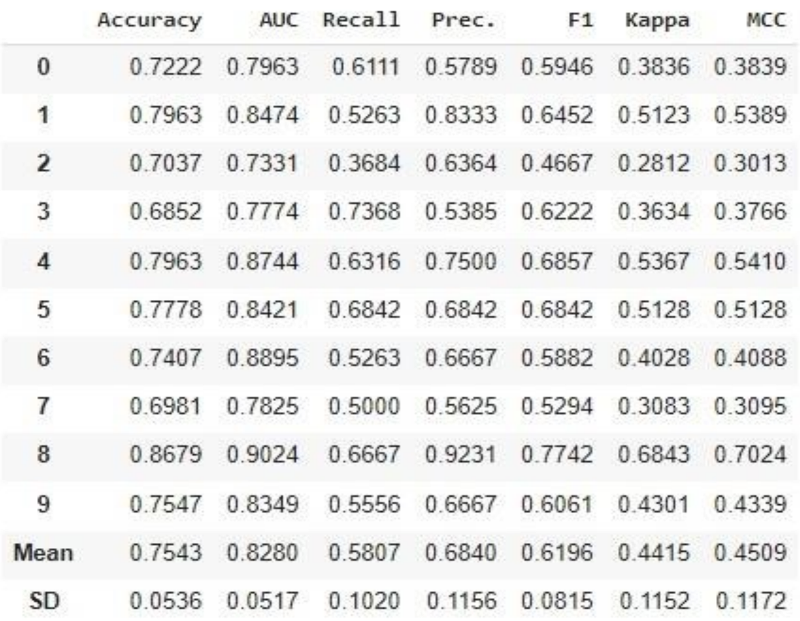


	Number of times pregnant	Plasma glucose concentration a 2 hours in an oral glucose tolerance test	Diastolic blood pressure (mm Hg)	Triceps skin fold thickness (mm)	2-Hour serum insulin (mu U/ml)	Body mass index (weight in kg/(height in m)^2)	Diabetes pedigree function	Age (years)	Class variable
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Code: Build a single model - "RandomForest"

```
from pycaret.datasets import get_data from
pycaret.classification import *
 diabetesDataSet = get_data("diabetes") s =
setup(data=diabetesDataSet, target='Class variable', silent=True)
rfModel = create_model('rf')
# Explore more parameters
```

Output:



	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7222	0.7963	0.6111	0.5789	0.5946	0.3836	0.3839
1	0.7963	0.8474	0.5263	0.8333	0.6452	0.5123	0.5389
2	0.7037	0.7331	0.3684	0.6364	0.4667	0.2812	0.3013
3	0.6852	0.7774	0.7368	0.5385	0.6222	0.3634	0.3766
4	0.7963	0.8744	0.6316	0.7500	0.6857	0.5367	0.5410
5	0.7778	0.8421	0.6842	0.6842	0.6842	0.5128	0.5128
6	0.7407	0.8895	0.5263	0.6667	0.5882	0.4028	0.4088
7	0.6981	0.7825	0.5000	0.5625	0.5294	0.3083	0.3095
8	0.8679	0.9024	0.6667	0.9231	0.7742	0.6843	0.7024
9	0.7547	0.8349	0.5556	0.6667	0.6061	0.4301	0.4339
Mean	0.7543	0.8280	0.5807	0.6840	0.6196	0.4415	0.4509
SD	0.0536	0.0517	0.1020	0.1156	0.0815	0.1152	0.1172

Code: Save the trained model

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

Output:

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

Code:

Load the model

```
rfModel = load_model('rfModelFile')
```

Output:

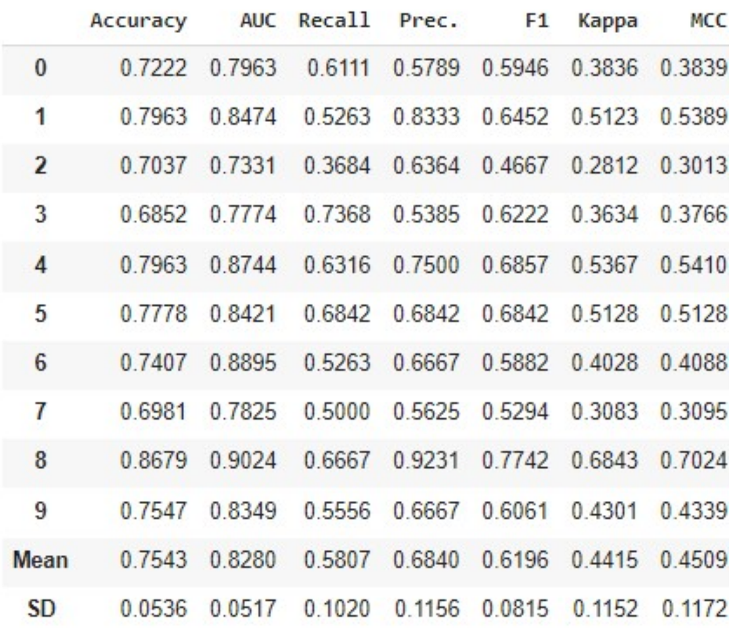
```
Transformation Pipeline and Model Successfully Loaded
```

Code: Save prediction results to csv

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

Code: Create RandomForest or any other model

```
rfModel = create_model('rf')
```

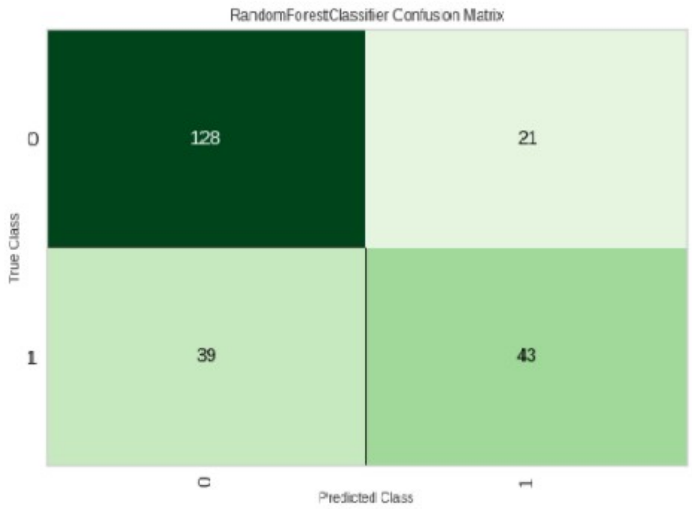


	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7222	0.7963	0.6111	0.5789	0.5946	0.3836	0.3839
1	0.7963	0.8474	0.5263	0.8333	0.6452	0.5123	0.5389
2	0.7037	0.7331	0.3684	0.6364	0.4667	0.2812	0.3013
3	0.6852	0.7774	0.7368	0.5385	0.6222	0.3634	0.3766
4	0.7963	0.8744	0.6316	0.7500	0.6857	0.5367	0.5410
5	0.7778	0.8421	0.6842	0.6842	0.6842	0.5128	0.5128
6	0.7407	0.8895	0.5263	0.6667	0.5882	0.4028	0.4088
7	0.6981	0.7825	0.5000	0.5625	0.5294	0.3083	0.3095
8	0.8679	0.9024	0.6667	0.9231	0.7742	0.6843	0.7024
9	0.7547	0.8349	0.5556	0.6667	0.6061	0.4301	0.4339
Mean	0.7543	0.8280	0.5807	0.6840	0.6196	0.4415	0.4509
SD	0.0536	0.0517	0.1020	0.1156	0.0815	0.1152	0.1172

Code: Confusion Matrix

```
plot_model(rfModel, plot='confusion_matrix')
```

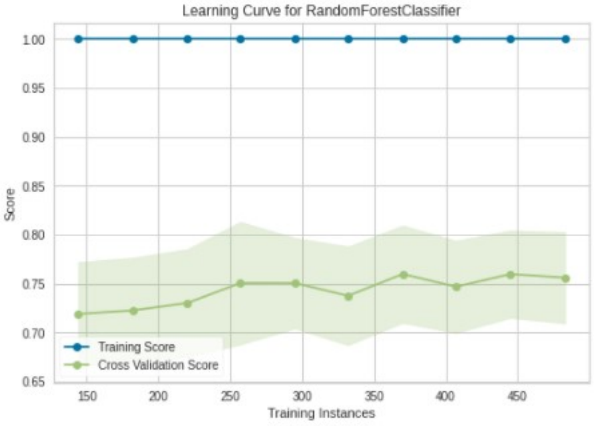
Output:



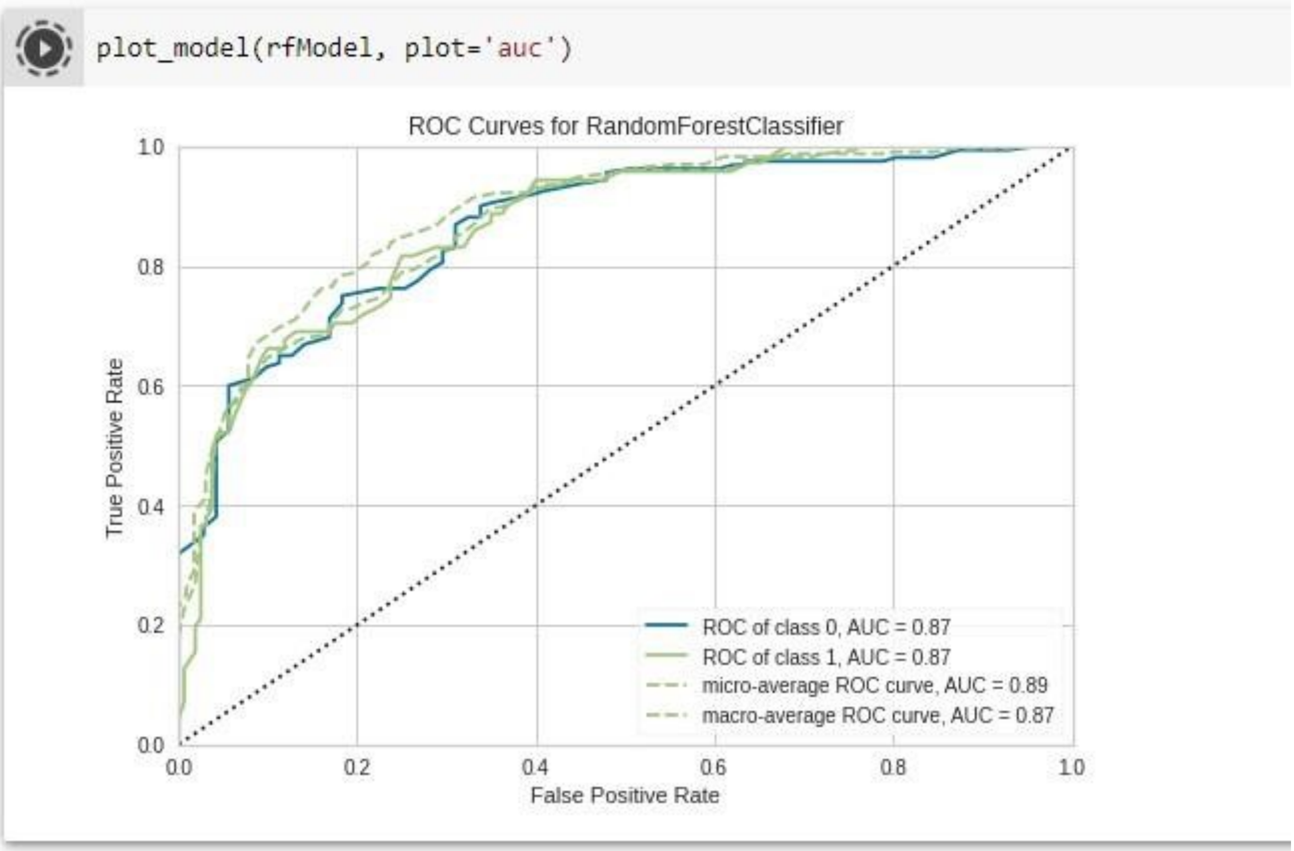
Code: Plot Learning Curve

```
plot_model(rfModel, plot='learning')
```

Output:



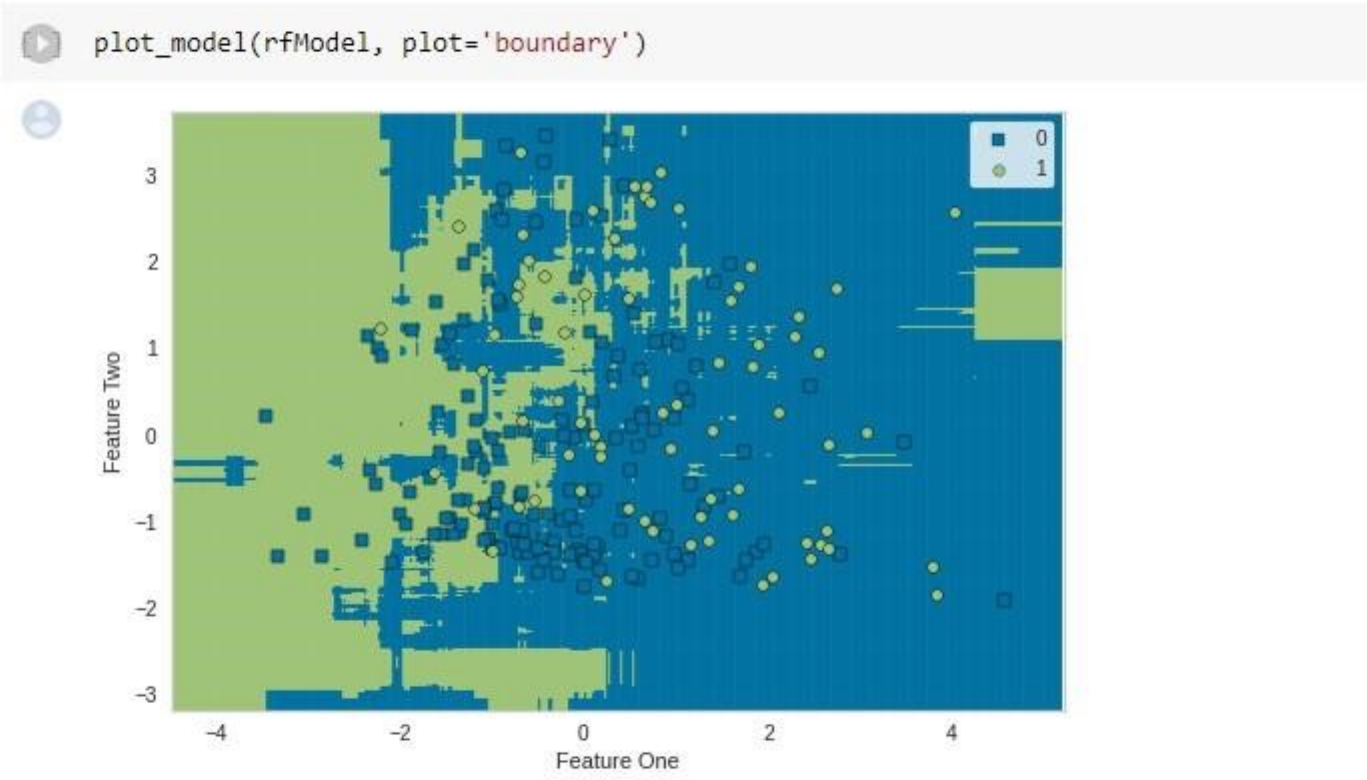
AUC Curve :



```
plot_model(rfModel, plot='auc')
```

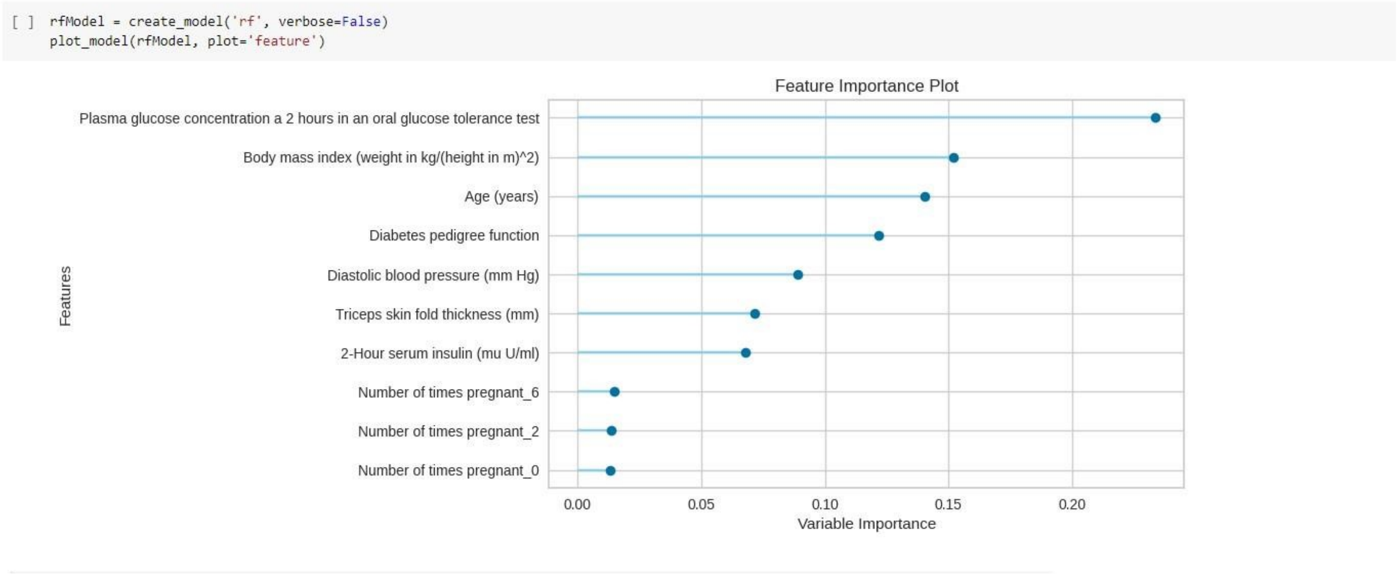
ROC Curves for RandomForestClassifier
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC of class 0, AUC = 0.87
ROC of class 1, AUC = 0.87
micro-average ROC curve, AUC = 0.89
macro-average ROC curve, AUC = 0.87

Decision Boundary



```
plot_model(rfModel, plot='boundary')
```

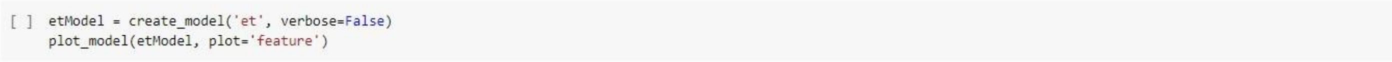
Feature importance



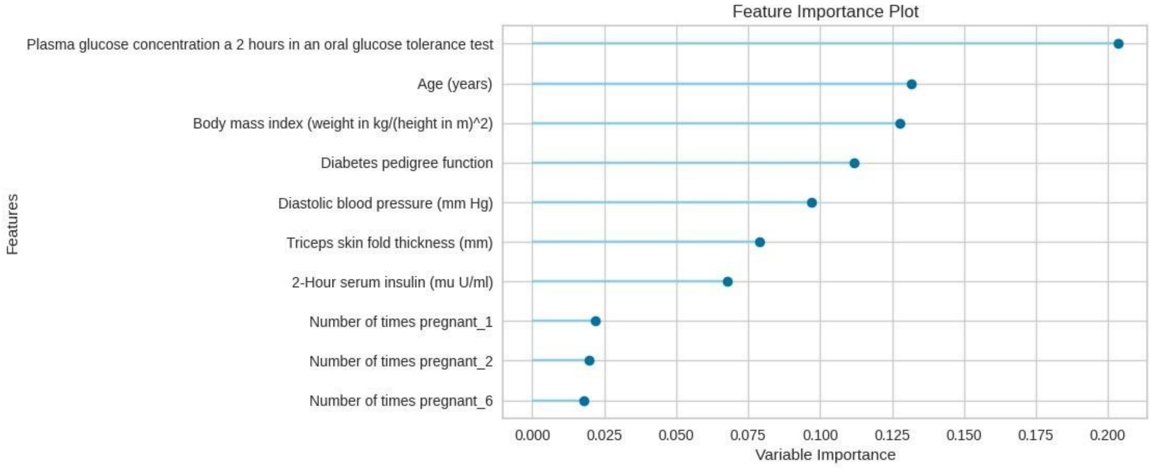
```
[ ] rfModel = create_model('rf', verbose=False)
    plot_model(rfModel, plot='feature')
```

Feature Importance Plot
Features
Plasma glucose concentration a 2 hours in an oral glucose tolerance test
Body mass index (weight in kg/(height in m)^2)
Age (years)
Diabetes pedigree function
Diastolic blood pressure (mm Hg)
Triceps skin fold thickness (mm)
2-Hour serum insulin (mu U/ml)
Number of times pregnant_6
Number of times pregnant_2
Number of times pregnant_0
0.00
0.05
0.10
0.15
0.20
Variable Importance

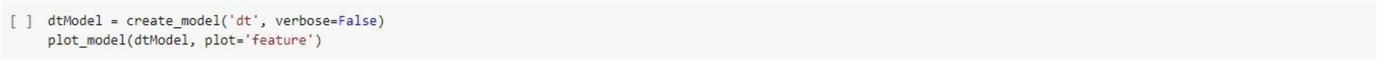
Feature importance using extra tree regression



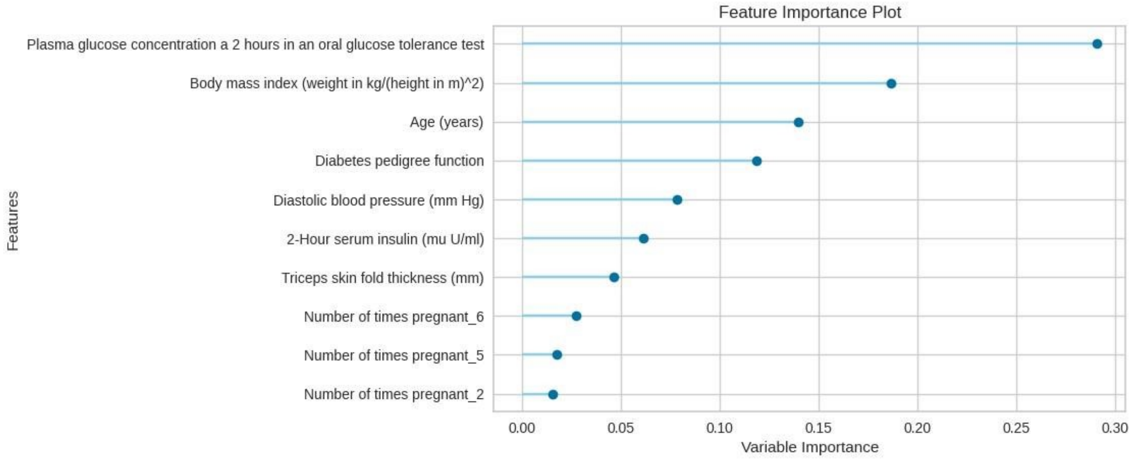
```
[ ] etModel = create_model('et', verbose=False)
    plot_model(etModel, plot='feature')
```



Feature importance using decision Tree :



```
[ ] dtModel = create_model('dt', verbose=False)
    plot_model(dtModel, plot='feature')
```



italicized text---

3.9 Deploy the model on AWS Click Here

6. Additional Creative Inputs (If Any):

Learning outcomes (What I have learnt):

- **Getting Data: How to import data from PyCaret repository**
- **Setting up Environment: How to setup an experiment in PyCaret and get started with building regression models**
- **Create Model: How to create a model, perform cross validation and evaluate regression metrics**
- **Tune Model: How to automatically tune the hyperparameters of a regression model**
- **Plot Model: How to analyze model performance using various plots**
- **Finalize Model: How to finalize the best model at the end of the experiment**
- **Predict Model: How to make prediction on new / unseen data**

- **Save / Load Model: How to save / load a model for future use**

Evaluation Grid (To be filled by Faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	Worksheet completion including writinglearning objectives/Outcomes.(To besubmitted 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**