PyCaret_training_prj

August 4, 2020

[1]:
```
# Import module
from pycaret.classification import *

# Loading data from pycaret
from pycaret.datasets import get_data
data = get_data('iris')


# Initialize setup (when using Notebook environment)
clf1 = setup(data, target = 'species')

# Initialize setup (outside of Notebook environment)
clf1 = setup(data, target = 'species', html = False)

# Initialize setup (When using remote execution such as Kaggle / GitHub␣
 ↪actions / CI-CD pipelines)
clf1 = setup(data, target = 'species', html = False, silent = True)
```

```
Setup Succesfully Completed!
                     Description  \
0                     session_id
1                    Target Type
2                   Label Encoded
3                   Original Data
4                  Missing Values
5                Numeric Features
6            Categorical Features
7                Ordinal Features
8      High Cardinality Features
9        High Cardinality Method
10                   Sampled Data
11           Transformed Train Set
12            Transformed Test Set
13                Numeric Imputer
14            Categorical Imputer
15                     Normalize
```

16	Normalize Method	False
17	Transformation	False
18	Transformation Method	None
19	PCA	False
20	PCA Method	None
21	PCA Components	None
22	Ignore Low Variance	False
23	Combine Rare Levels	False
24	Rare Level Threshold	None
25	Numeric Binning	False
26	Remove Outliers	False
27	Outliers Threshold	None
28	Remove Multicollinearity	False
29	Multicollinearity Threshold	None
30	Clustering	False
31	Clustering Iteration	None
32	Polynomial Features	False
33	Polynomial Degree	None
34	Trignometry Features	False
35	Polynomial Threshold	None
36	Group Features	False
37	Feature Selection	False
38	Features Selection Threshold	None
39	Feature Interaction	False
40	Feature Ratio	False
41	Interaction Threshold	None
42	Fix Imbalance	False
43	Fix Imbalance Method	SMOTE

		Value
0		5102
1		Multiclass
2	Iris-setosa: 0, Iris-versicolor: 1, Iris-virgi...	
3		(150, 5)
4		False
5		4
6		0
7		False
8		False
9		None
10		(150, 5)
11		(104, 4)
12		(46, 4)
13		mean
14		constant
15		False
16		None
17		False

```
18                                                  None
19                                                 False
20                                                  None
21                                                  None
22                                                 False
23                                                 False
24                                                  None
25                                                 False
26                                                 False
27                                                  None
28                                                 False
29                                                  None
30                                                 False
31                                                  None
32                                                 False
33                                                  None
34                                                 False
35                                                  None
36                                                 False
37                                                 False
38                                                  None
39                                                 False
40                                                 False
41                                                  None
42                                                 False
43                                                 SMOTE
```

[97]:
```python
# import libraries
import pandas as pd
import sys

data = pd.read_csv('/scratch/PyCaret/Counts/LUAD_plus_LUSCNormal.
 ↪MirnaSeq_Count.txt', sep = '\t' )
# print( data.head() )
data.set_index('ID', inplace=True)

print(data.iloc[0:6, 0:3])

data = data.T

#print( data.head() )

print( data.shape )

print( data.iloc[0:5, 0:3] )
```

```
                    TCGA-05-4244-01A  TCGA-05-4249-01A  TCGA-05-4250-01A
```

```
ID
hsa-miR-6859-5p            0.0000            0.0000            0.5318
hsa-miR-6859-3p            0.3449            0.0000            0.0000
hsa-miR-1302               0.0000            0.0000            0.0000
hsa-miR-6723-5p            0.0000            0.0000            0.0000
hsa-miR-200b-5p            6.5533           10.3286           10.1049
hsa-miR-200b-3p          248.6807          928.1989          215.3940
(1191, 2772)
ID               hsa-miR-6859-5p hsa-miR-6859-3p hsa-miR-1302
TCGA-05-4244-01A               0          0.3449            0
TCGA-05-4249-01A               0               0            0
TCGA-05-4250-01A          0.5318               0            0
TCGA-05-4382-01A               0               0            0
TCGA-05-4384-01A               0               0            0
```

[ ]:

[ ]: # read list of selected mRNAs from DESeq2

[99]:
```
top_genes_file = open('/scratch/PyCaret/Counts/top.genes.21.txt', "r")

top_genes = top_genes_file.read().splitlines()
print(top_genes)

for n, word in enumerate(top_genes):
    word=word.replace('.', '-')
    top_genes[n] = word

print('print mRNA names replace dot with -')
print(top_genes)
```

```
['hsa.miR.139.3p', 'hsa.miR.139.5p', 'hsa.miR.30a.3p', 'hsa.miR.30c.2.3p',
'hsa.miR.133a.3p', 'hsa.miR.133a.3p_2', 'hsa.miR.1', 'hsa.miR.1_2',
'hsa.miR.145.3p', 'hsa.miR.133b', 'hsa.miR.30a.5p', 'hsa.miR.21.5p',
'hsa.miR.195.5p', 'hsa.miR.143.3p', 'hsa.miR.135b.5p', 'hsa.miR.598.3p',
'hsa.miR.141.3p', 'hsa.miR.140.3p', 'hsa.miR.1247.3p', 'hsa.miR.141.5p',
'hsa.miR.210.3p']
print mRNA names replace dot with -
['hsa-miR-139-3p', 'hsa-miR-139-5p', 'hsa-miR-30a-3p', 'hsa-miR-30c-2-3p', 'hsa-
miR-133a-3p', 'hsa-miR-133a-3p_2', 'hsa-miR-1', 'hsa-miR-1_2', 'hsa-miR-145-3p',
'hsa-miR-133b', 'hsa-miR-30a-5p', 'hsa-miR-21-5p', 'hsa-miR-195-5p', 'hsa-
miR-143-3p', 'hsa-miR-135b-5p', 'hsa-miR-598-3p', 'hsa-miR-141-3p', 'hsa-
miR-140-3p', 'hsa-miR-1247-3p', 'hsa-miR-141-5p', 'hsa-miR-210-3p']
```

[102]:
```
# load design table, with tumor vs normal classification
data_design = pd.read_csv('/scratch/PyCaret/Counts/LUAD_plus_LUSCNormal.
 ↪MirnaSeq_Count_Design.txt', sep = '\t')

# print( data.design.describe() )
data_design.set_index('ID', inplace=True)
print( data_design.shape )
print( data_design.iloc[0:3, 0:3] )
```

```
(1191, 17)
                 Tumor Type     SubjectID     SampleType
ID
TCGA-05-4244-01A       LUAD  TCGA-05-4244  Primary Tumor
TCGA-05-4249-01A       LUAD  TCGA-05-4249  Primary Tumor
TCGA-05-4250-01A       LUAD  TCGA-05-4250  Primary Tumor
```

[127]:
```
data = data[top_genes]

print( data.shape )

print( data.iloc[0:3, 0:3] )

print(data.dtypes )
```

```
(1191, 21)
ID               hsa-miR-139-3p hsa-miR-139-5p hsa-miR-30a-3p
TCGA-05-4244-01A         2.7593         27.248        4451.07
TCGA-05-4249-01A         1.0329         8.2629        3668.04
TCGA-05-4250-01A         1.5955        18.0825         5985.3
ID
hsa-miR-139-3p       object
hsa-miR-139-5p       object
hsa-miR-30a-3p       object
hsa-miR-30c-2-3p     object
hsa-miR-133a-3p      object
hsa-miR-133a-3p_2    object
hsa-miR-1            object
hsa-miR-1_2          object
hsa-miR-145-3p       object
hsa-miR-133b         object
hsa-miR-30a-5p       object
hsa-miR-21-5p        object
hsa-miR-195-5p       object
hsa-miR-143-3p       object
hsa-miR-135b-5p      object
hsa-miR-598-3p       object
```

```
hsa-miR-141-3p       object
hsa-miR-140-3p       object
hsa-miR-1247-3p      object
hsa-miR-141-5p       object
hsa-miR-210-3p       object
dtype: object
```

[138]:
```
# data_numeric = data.astype(str).astype(float)

data_numeric = data.apply( pd.to_numeric, errors='coerce' )

data = data_numeric

print( data.dtypes )
```

```
ID
hsa-miR-139-3p       float64
hsa-miR-139-5p       float64
hsa-miR-30a-3p       float64
hsa-miR-30c-2-3p     float64
hsa-miR-133a-3p      float64
hsa-miR-133a-3p_2    float64
hsa-miR-1            float64
hsa-miR-1_2          float64
hsa-miR-145-3p       float64
hsa-miR-133b         float64
hsa-miR-30a-5p       float64
hsa-miR-21-5p        float64
hsa-miR-195-5p       float64
hsa-miR-143-3p       float64
hsa-miR-135b-5p      float64
hsa-miR-598-3p       float64
hsa-miR-141-3p       float64
hsa-miR-140-3p       float64
hsa-miR-1247-3p      float64
hsa-miR-141-5p       float64
hsa-miR-210-3p       float64
dtype: object
```

[139]:
```
# only merge SampleType from data.design dataframe to data dataframe

# data2 = pd.merge(data, data.design[["SampleType"]], left_index=True )
data2 = data.join(data_design[["SampleType"]])

data2.head()
```

[139]:

	hsa-miR-139-3p	hsa-miR-139-5p	hsa-miR-30a-3p	\
TCGA-05-4244-01A	2.7593	27.2480	4451.074	
TCGA-05-4249-01A	1.0329	8.2629	3668.038	
TCGA-05-4250-01A	1.5955	18.0825	5985.295	
TCGA-05-4382-01A	3.2936	20.4206	2076.313	
TCGA-05-4384-01A	35.3987	38.6851	7721.054	

	hsa-miR-30c-2-3p	hsa-miR-133a-3p	hsa-miR-133a-3p_2	\
TCGA-05-4244-01A	6.8982	5.5186	5.5186	
TCGA-05-4249-01A	8.9515	2.4100	2.4100	
TCGA-05-4250-01A	14.8914	1.5955	1.5955	
TCGA-05-4382-01A	9.2222	4.6111	4.6111	
TCGA-05-4384-01A	54.9291	5.3521	5.3521	

	hsa-miR-1	hsa-miR-1_2	hsa-miR-145-3p	hsa-miR-133b	\
TCGA-05-4244-01A	8.9677	8.6228	18.9701	0.3449	
TCGA-05-4249-01A	11.3615	11.0172	14.8044	0.0000	
TCGA-05-4250-01A	11.7004	10.6367	38.2923	0.0000	
TCGA-05-4382-01A	3.2936	3.2936	32.2777	0.0000	
TCGA-05-4384-01A	20.1876	20.1876	41.3142	1.1268	

	hsa-miR-30a-5p	hsa-miR-21-5p	hsa-miR-195-5p	\
TCGA-05-4244-01A	16312.560	541087.9	32.4216	
TCGA-05-4249-01A	15395.840	414415.7	12.7386	
TCGA-05-4250-01A	19815.190	557526.9	22.3372	
TCGA-05-4382-01A	9240.645	393332.1	26.3491	
TCGA-05-4384-01A	18384.710	230144.2	37.1828	

	hsa-miR-143-3p	hsa-miR-135b-5p	hsa-miR-598-3p	\
TCGA-05-4244-01A	17632.19	334.5635	5.1737	
TCGA-05-4249-01A	23037.65	235.8369	16.1815	
TCGA-05-4250-01A	41931.63	121.7907	14.3596	
TCGA-05-4382-01A	38192.42	6.5873	7.2460	
TCGA-05-4384-01A	167363.60	36.2438	11.5492	

	hsa-miR-141-3p	hsa-miR-140-3p	hsa-miR-1247-3p	\
TCGA-05-4244-01A	1253.4060	238.6783	6.2084	
TCGA-05-4249-01A	528.4812	388.3562	5.8529	
TCGA-05-4250-01A	1146.6410	303.6790	15.4233	
TCGA-05-4382-01A	1324.0440	490.0941	3.2936	
TCGA-05-4384-01A	573.0462	647.2239	4.9765	

	hsa-miR-141-5p	hsa-miR-210-3p	SampleType
TCGA-05-4244-01A	280.4125	1853.2060	Primary Tumor
TCGA-05-4249-01A	267.1670	222.0654	Primary Tumor

```
TCGA-05-4250-01A        345.1623       2019.3850  Primary Tumor
TCGA-05-4382-01A        533.5701       2778.5170  Primary Tumor
TCGA-05-4384-01A        593.3277        499.0563  Primary Tumor
```

[140]:
```
print( data2.dtypes)
```

```
hsa-miR-139-3p       float64
hsa-miR-139-5p       float64
hsa-miR-30a-3p       float64
hsa-miR-30c-2-3p     float64
hsa-miR-133a-3p      float64
hsa-miR-133a-3p_2    float64
hsa-miR-1            float64
hsa-miR-1_2          float64
hsa-miR-145-3p       float64
hsa-miR-133b         float64
hsa-miR-30a-5p       float64
hsa-miR-21-5p        float64
hsa-miR-195-5p       float64
hsa-miR-143-3p       float64
hsa-miR-135b-5p      float64
hsa-miR-598-3p       float64
hsa-miR-141-3p       float64
hsa-miR-140-3p       float64
hsa-miR-1247-3p      float64
hsa-miR-141-5p       float64
hsa-miR-210-3p       float64
SampleType            object
dtype: object
```

[ ]:

[ ]:

[3]:
```
from platform import python_version

print(python_version())

import sys
sys.executable
```

```
3.6.10
```

[3]:
```
'/home/jdu/anaconda3/envs/pyCaret/bin/python'
```

[141]:
```
# import classification module
from pycaret.classification import *

# init setup
clf1 = setup(data2, target = 'SampleType')

# return best model
best = compare_models()

# return best model based on Recall
best = compare_models(sort = 'Recall') #default is 'Accuracy'

# compare specific models
best_specific = compare_models(whitelist = ['dt','rf','xgboost'])

# blacklist certain models
best_specific = compare_models(blacklist = ['catboost','svm'])

# return top 3 models based on Accuracy
top3 = compare_models(n_select = 3)
```

```
<pandas.io.formats.style.Styler at 0x7fb28ee78f98>
```

[7]:
```
# load data (replace this part with your own script)
# Create Model
# train custom model
import numpy as np
# pip install gplearn

from gplearn.genetic import SymbolicClassifier
```

[142]:
```
# import classification module
from pycaret.classification import *
import numpy as np

# init setup
clf1 = setup(data2, target = 'SampleType')

# train logistic regression model
lr = create_model('lr') #lr is the id of the model
```

```
# check the model library to see all models
models()

# train rf model using 5 fold CV
rf = create_model('rf', fold = 5)

# train svm model without CV
svm = create_model('svm', cross_validation = False)

# train xgboost model with max_depth = 10
xgboost = create_model('xgboost', max_depth = 10)

# train xgboost model on gpu
# xgboost_gpu = create_model('xgboost', tree_method = 'gpu_hist', gpu_id =␣
 ↪0) #0 is gpu-id

# train multiple lightgbm models with n learning_rate
lgbms = [create_model('lightgbm', learning_rate = i) for i in np.arange(0.
 ↪1,1,0.1)]
```

```
<pandas.io.formats.style.Styler at 0x7fb298882710>
```

[ ]:

[143]:
```
# train custom model
from gplearn.genetic import SymbolicClassifier
symclf = SymbolicClassifier(generations=20)
sc = create_model(symclf)
```

```
<pandas.io.formats.style.Styler at 0x7fb28cd7f1d0>
```

[ ]:

[16]:
```
## tune the model
data2
```

[16]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa

```
3            4.6          3.1           1.5          0.2     Iris-setosa
4            5.0          3.6           1.4          0.2     Iris-setosa
..           ...          ...           ...          ...             ...
145          6.7          3.0           5.2          2.3  Iris-virginica
146          6.3          2.5           5.0          1.9  Iris-virginica
147          6.5          3.0           5.2          2.0  Iris-virginica
148          6.2          3.4           5.4          2.3  Iris-virginica
149          5.9          3.0           5.1          1.8  Iris-virginica

[150 rows x 5 columns]
```

[144]:
```
# import classification module
from pycaret.classification import *

# init setup
clf1 = setup(data2, target = 'SampleType')

# train a decision tree model
dt = create_model('dt')

# tune hyperparameters of decision tree
tuned_dt = tune_model(dt)

# tune hyperparameters with increased n_iter
tuned_dt = tune_model(dt, n_iter = 50)

# tune hyperparameters to optimize AUC
tuned_dt = tune_model(dt, optimize = 'AUC') #default is 'Accuracy'

# tune hyperparameters with custom_grid
params = {"max_depth": np.random.randint(1, (len(data.columns)*.85),20),
          "max_features": np.random.randint(1, len(data.columns),20),
          "min_samples_leaf": [2,3,4,5,6],
          "criterion": ["gini", "entropy"]
          }

tuned_dt_custom = tune_model(dt, custom_grid = params)

# tune multiple models dynamically
top3 = compare_models(n_select = 3)
tuned_top3 = [tune_model(i) for i in top3]
```

```
<pandas.io.formats.style.Styler at 0x7fb28cdb67f0>
```

[ ]:

[ ]:
```
## Ensemble Model
```

[ ]:
```
# import classification module
from pycaret.classification import *

# init setup
clf1 = setup(data2, target = 'SampleType')

# train a decision tree model
dt = create_model('dt')

# train a bagging classifier on dt
bagged_dt = ensemble_model(dt, method = 'Bagging')

# train a adaboost classifier on dt with 100 estimators
boosted_dt = ensemble_model(dt, method = 'Boosting', n_estimators = 100)

# train a votingclassifier on all models in library
blender = blend_models()

# train a voting classifier on specific models
dt = create_model('dt')
rf = create_model('rf')
adaboost = create_model('ada')
blender_specific = blend_models(estimator_list = [dt,rf,adaboost], method =␣
 ↪'soft')

# train a voting classifier dynamically
blender_top5 = blend_models(compare_models(n_select = 5))

# train a stacking classifier
stacker = stack_models(estimator_list = [dt,rf], meta_model = adaboost)

# stack multiple models dynamically
top7 = compare_models(n_select = 7)
stacker = stack_models(estimator_list = top7[1:], meta_model = top7[0])
```

[ ]:

[ ]:

[ ]:

[ ]:
```
# Predict Model
```

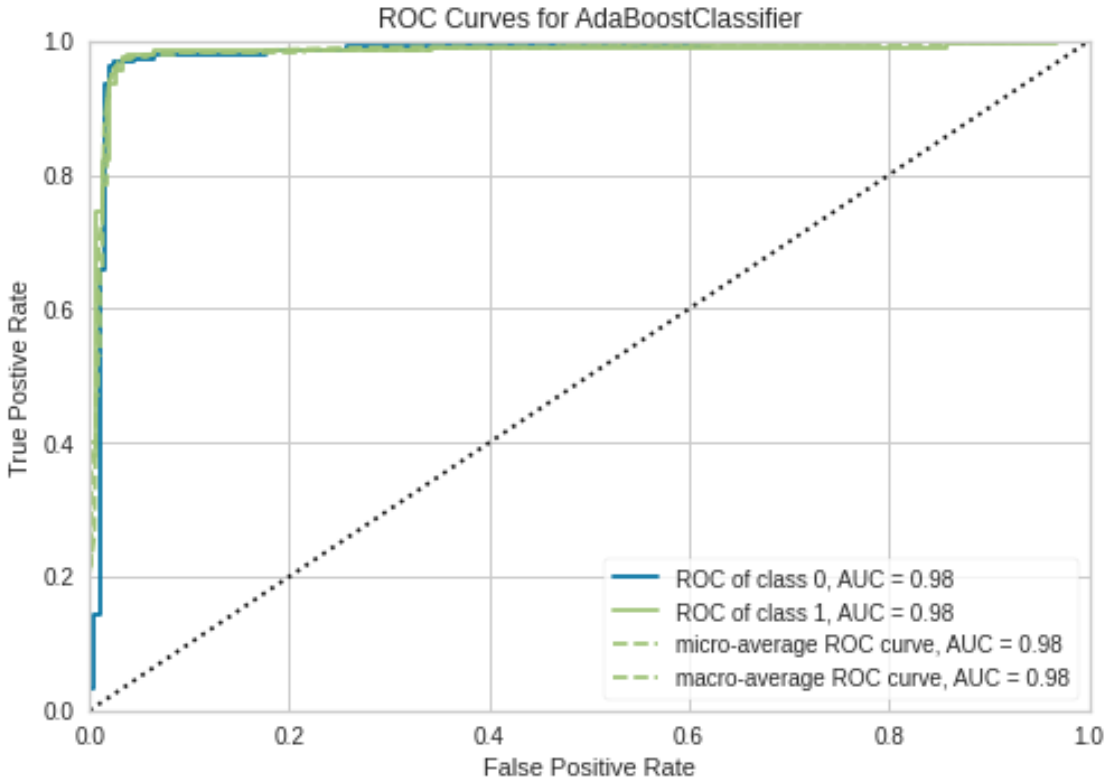
[ ]:
```
# train a catboost model
catboost = create_model('catboost')

# predict on holdout set (when no data is passed)
pred_holdout = predict_model(catboost)

# predict on new dataset
new_data = pd.read_csv('new-data.csv')
pred_new = predict_model(catboost, data = new_data)
```

[ ]:

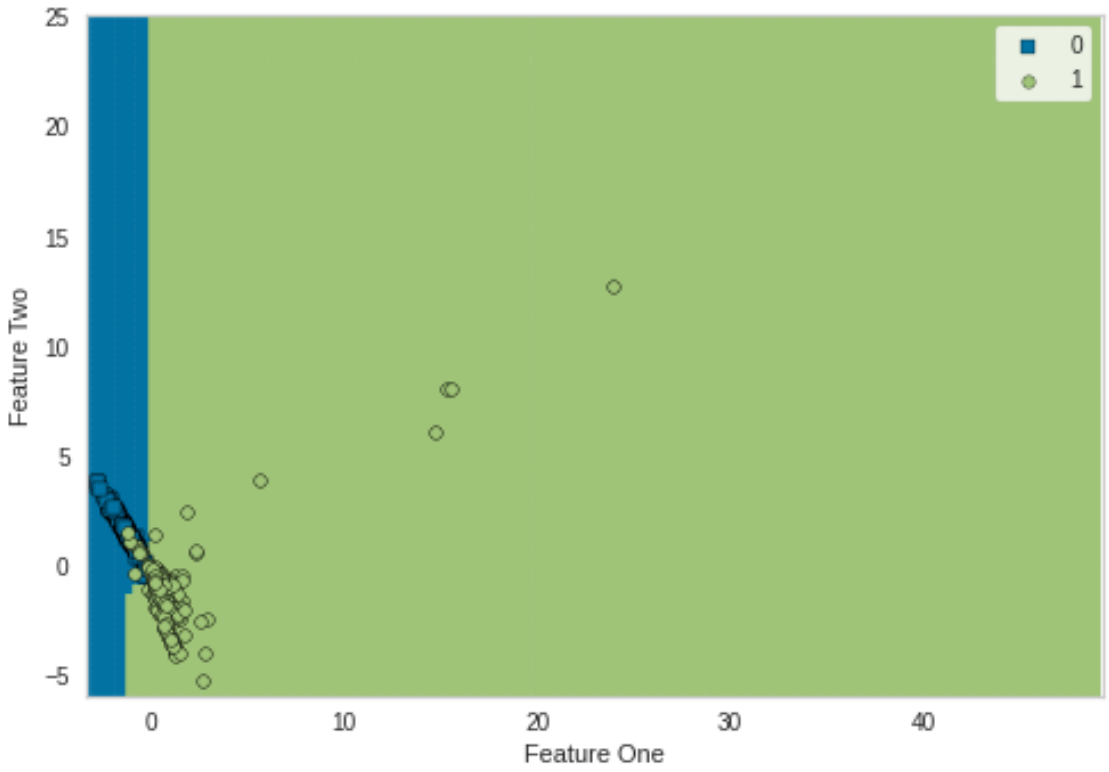
[ ]:

[ ]:
```
# Plot Model
```

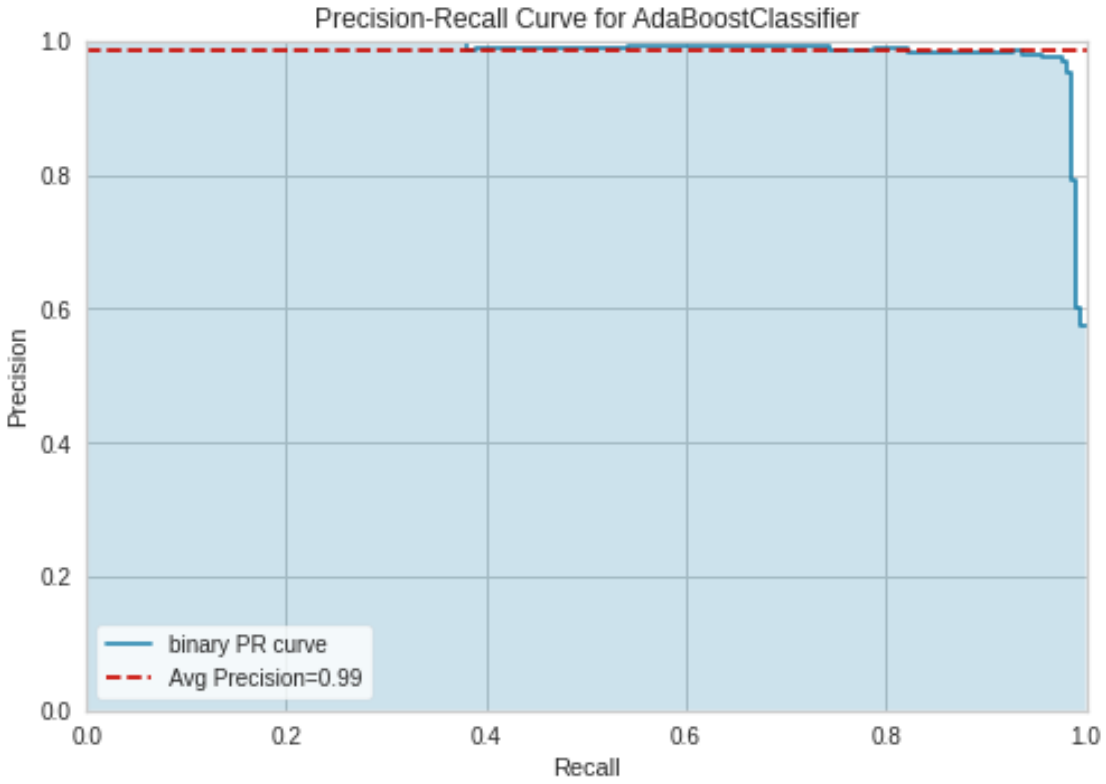
[146]:
```
# import classification module
from pycaret.classification import *

# init setup
clf1 = setup(data2, target = 'SampleType')

# train adaboost model
adaboost = create_model('ada')

# AUC plot
plot_model(adaboost, plot = 'auc')
```



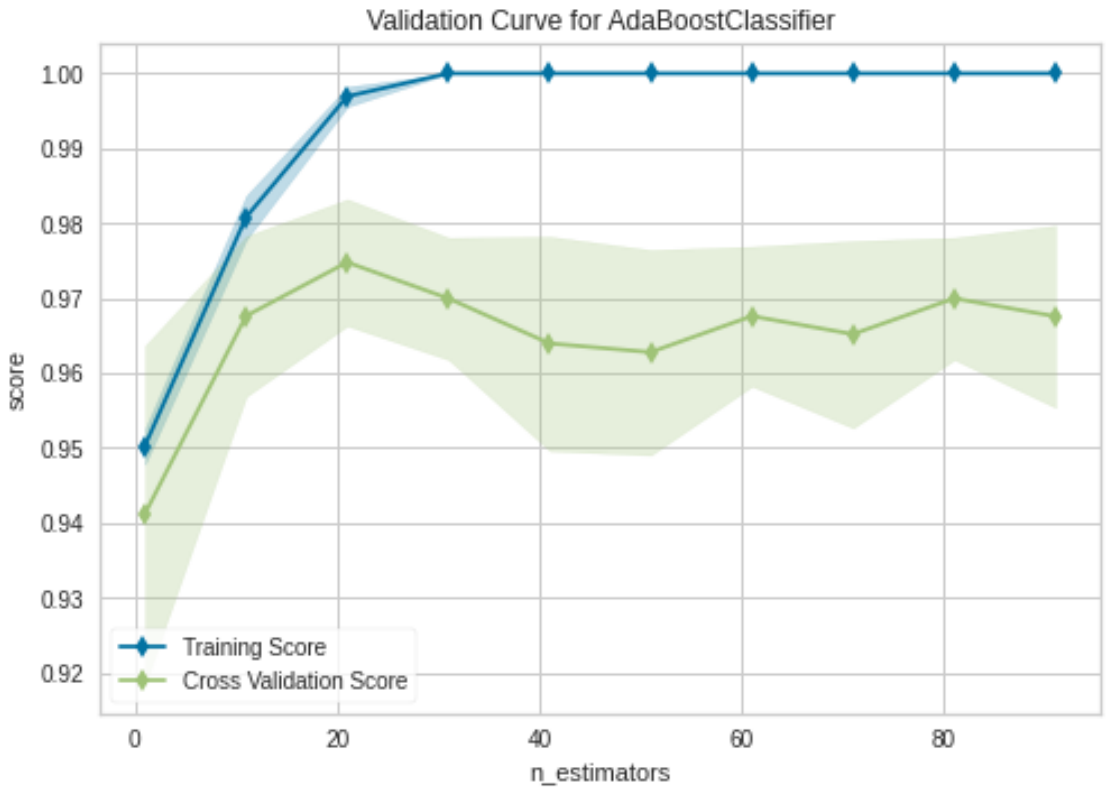
[147]:
```
# Decision Boundary
plot_model(adaboost, plot = 'boundary')
```



[148]:
```
# Precision Recall Curve
plot_model(adaboost, plot = 'pr')
```



[149]:
```
# Validation Curve
plot_model(adaboost, plot = 'vc')
```



[150]:
```
evaluate_model(adaboost)
```

```
interactive(children=(ToggleButtons(description='Plot Type:', icons=('',), options=(('Hyperparam
```

[ ]:

[ ]: