Last updated: 16 Feb 2023

PyCaret Binary Classification

PyCaret is an open-source, low-code machine learning library in Python that automates machine learning workflows. It is an end-to-end machine learning and model management tool that exponentially speeds up the experiment cycle and makes you more productive.

```
pip install pycaret
```

```
Downloading schemdraw-0.15-py3-none-any.whl (106 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 106.8/106.8 kB 11.4 MB/s eta 0:00:00
Downloading sktime-0.26.0-py3-none-any.whl (21.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.8/21.8 MB 77.4 MB/s eta 0:00:00
Downloading category_encoders-2.6.3-py2.py3-none-any.whl (81 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.9/81.9 kB 9.4 MB/s eta 0:00:00
Downloading deprecation-2.1.0-py2.py3-none-any.whl (11 kB)
Downloading joblib-1.3.2-py3-none-any.whl (302 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 302.2/302.2 kB 25.1 MB/s eta 0:00:00
Downloading kaleido-0.2.1-py2.py3-none-manylinux1_x86_64.whl (79.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.9/79.9 MB 8.7 MB/s eta 0:00:00
Downloading plotly_resampler-0.10.0-py3-none-any.whl (80 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 80.7/80.7 kB 8.8 MB/s eta 0:00:00
Downloading pmdarima-2.0.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl (2.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.1/2.1 MB 80.0 MB/s eta 0:00:00
Downloading scikit_learn-1.4.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.1/12.1 MB 96.8 MB/s eta 0:00:00
Downloading scikit_plot-0.3.7-py3-none-any.whl (33 kB)
Downloading scipy-1.11.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (36.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 36.4/36.4 MB 20.3 MB/s eta 0:00:00
Downloading tbats-1.1.3-py3-none-any.whl (44 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.0/44.0 kB 4.7 MB/s eta 0:00:00
Downloading wurlitzer-3.1.1-py3-none-any.whl (8.6 kB)
Downloading xxhash-3.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.1/194.1 kB 20.7 MB/s eta 0:00:00
Downloading dash-2.18.1-py3-none-any.whl (7.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.5/7.5 MB 98.3 MB/s eta 0:00:00
Downloading dash_core_components-2.0.0-py3-none-any.whl (3.8 kB)
Downloading dash_html_components-2.0.0-py3-none-any.whl (4.1 kB)
Downloading dash_table-5.0.0-py3-none-any.whl (3.9 kB)
Using cached jedi-0.19.1-py2.py3-none-any.whl (1.6 MB)
Downloading orjson-3.10.7-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (141 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 141.9/141.9 kB 15.2 MB/s eta 0:00:00
Downloading scikit_base-0.7.8-py3-none-any.whl (130 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 130.1/130.1 kB 14.4 MB/s eta 0:00:00
Downloading tsdownsample-0.1.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.1/2.1 MB 78.2 MB/s eta 0:00:00
Downloading retrying-1.3.4-py3-none-any.whl (11 kB)
Building wheels for collected packages: pyod
  Building wheel for pyod (setup.py) ... done
  Created wheel for pyod: filename=pyod-2.0.2-py3-none-any.whl size=198469 sha256=be7df6b231b2fcbfb40fead4a1486ffb1ab5fbe8fe6a5d1
  Stored in directory: /root/.cache/pip/wheels/77/c2/20/34d1f15b41b701ba69f42a32304825810d680754d509f91391
Successfully built pyod
Installing collected packages: kaleido, dash-table, dash-html-components, dash-core-components, xxhash, wurlitzer, tsdownsample,
  Attempting uninstall: scipy
    Found existing installation: scipy 1.13.1
    Uninstalling scipy-1.13.1:
      Successfully uninstalled scipy-1.13.1
  Attempting uninstall: joblib
    Found existing installation: joblib 1.4.2
    Uninstalling joblib-1.4.2:
      Successfully uninstalled joblib-1.4.2
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 1.5.2
    Uninstalling scikit-learn-1.5.2:
      Successfully uninstalled scikit-learn-1.5.2
Successfully installed category-encoders-2.6.3 dash-2.18.1 dash-core-components-2.0.0 dash-html-components-2.0.0 dash-table-5.0.0
```

```
pip install pycaret[full]
```

```
Downloading websockets-12.0-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 130.2/130.2 kB 12.6 MB/s eta 0:00:00
Downloading wtforms-3.1.2-py3-none-any.whl (145 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 146.0/146.0 kB 15.2 MB/s eta 0:00:00
Downloading ansi2html-1.9.2-py3-none-any.whl (17 kB)
Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Downloading fs-2.4.16-py2.py3-none-any.whl (135 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 135.3/135.3 kB 14.1 MB/s eta 0:00:00
Downloading Mako-1.3.5-py3-none-any.whl (78 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 78.6/78.6 kB 8.2 MB/s eta 0:00:00
Downloading pywavelets-1.7.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.5/4.5 MB 84.9 MB/s eta 0:00:00
Downloading rich_click-1.8.3-py3-none-any.whl (35 kB)
Downloading Deprecated-1.2.14-py2.py3-none-any.whl (9.6 kB)
Downloading gitdb-4.0.11-py3-none-any.whl (62 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.7/62.7 kB 6.6 MB/s eta 0:00:00
Downloading Faker-29.0.0-py3-none-any.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 83.3 MB/s eta 0:00:00
Downloading zope.event-5.0-py3-none-any.whl (6.8 kB)
Downloading zope.interface-7.0.3-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.wh
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 254.1/254.1 kB 24.8 MB/s eta 0:00:00
Downloading smmap-5.0.1-py3-none-any.whl (24 kB)
Building wheels for collected packages: htmlmin, fugue-sql-antlr, dash-cytoscape
  Building wheel for htmlmin (setup.py) ... done
  Created wheel for htmlmin: filename=htmlmin-0.1.12-py3-none-any.whl size=27081 sha256=0aa300c879abac1fb99b0bbf83babeceb85e43195
  Stored in directory: /root/.cache/pip/wheels/dd/91/29/a79cecb328d01739e64017b6fb9a1ab9d8cb1853098ec5966d
  Building wheel for fugue-sql-antlr (setup.py) ... done
  Created wheel for fugue-sql-antlr: filename=fugue_sql_antlr-0.2.2-py3-none-any.whl size=158202 sha256=06344488f42ad0c305cf75c82
  Stored in directory: /root/.cache/pip/wheels/2b/be/4b/27ebc4ae02e605628d7bbe2a49ab875eb84275a1ddb46cbe2c
  Building wheel for dash-cytoscape (setup.py) ... done
  Created wheel for dash-cytoscape: filename=dash_cytoscape-1.0.2-py3-none-any.whl size=4010717 sha256=efe8d0431337bdcabf9bc9cfd0
  Stored in directory: /root/.cache/pip/wheels/91/23/5e/56fa701c668444b121ad2353a96478179dc49086a9c44ee930
Successfully built htmlmin fugue-sql-antlr dash-cytoscape
Installing collected packages: pydub, htmlmin, dash-testing-stub, appdirs, antlr4-python3-runtime, aniso8601, zope.interface, zop
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 3.0.4
    Uninstalling Werkzeug-3.0.4:
      Successfully uninstalled Werkzeug-3.0.4
  Attempting uninstall: importlib-metadata
    Found existing installation: importlib_metadata 8.5.0
    Uninstalling importlib_metadata-8.5.0:
      Successfully uninstalled importlib_metadata-8.5.0
Successfully installed Flask-WTF-1.2.1 Mako-1.3.5 PyWavelets-1.7.0 SALib-1.5.1 Werkzeug-2.3.8 adagio-0.2.6 aiofiles-23.2.1 alembi
```

```
# check installed version
import pycaret
pycaret.__version__
```

```
'3.3.2'
```

Quick start

PyCaret's Classification Module is a supervised machine learning module that is used for classifying elements into groups. The goal is to predict the categorical class labels which are discrete and unordered.

Some common use cases include predicting customer default (Yes or No), predicting customer churn (customer will leave or stay), the disease found (positive or negative).

This module can be used for binary or multiclass problems. It provides several pre-processing features that prepare the data for modeling through the setup function. It has over 18 ready-to-use algorithms and several plots to analyze the performance of trained models.

A typical workflow in PyCaret consist of following 5 steps in this order:

Setup ➡️ Compare Models ➡️ Analyze Model ➡️ Prediction ➡️ Save Model

```
# loading sample dataset from pycaret dataset module
from pycaret.datasets import get_data
data = get_data('bank')
print("Data before sampling:", data.shape)
data = data.sample(frac=0.02, random_state=123)

print("Data after sampling:", data.shape)
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	p
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	u
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	u
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	u
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	u
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	u

```
Data before sampling: (45211, 17)
Data after sampling: (904, 17)
```

Setup

This function initializes the training environment and creates the transformation pipeline. Setup function must be called before executing any other function in PyCaret. It only has two required parameters i.e. `data` and `target`. All the other parameters are optional.

```
# import pycaret classification and init setup
from pycaret.classification import *
s = setup(data, target = 'loan', session_id = 123)
```

	Description	Value
0	Session id	123
1	Target	loan
2	Target type	Binary
3	Target mapping	no: 0, yes: 1
4	Original data shape	(904, 17)
5	Transformed data shape	(904, 49)
6	Transformed train set shape	(632, 49)
7	Transformed test set shape	(272, 49)
8	Numeric features	7
9	Categorical features	9
10	Preprocess	True
11	Imputation type	simple
12	Numeric imputation	mean
13	Categorical imputation	mode
14	Maximum one-hot encoding	25
15	Encoding method	None
16	Fold Generator	StratifiedKFold
17	Fold Number	10
18	CPU Jobs	-1
19	Use GPU	False
20	Log Experiment	False
21	Experiment Name	clf-default-name
22	USI	36aa

Once the setup has been successfully executed it shows the information grid containing experiment level information.

- **Session id:** A pseudo-random number distributed as a seed in all functions for later reproducibility. If no `session_id` is passed, a random number is automatically generated that is distributed to all functions.

- **Target type:** Binary, Multiclass, or Regression. The Target type is automatically detected.

- **Label Encoding:** When the Target variable is of type string (i.e. 'Yes' or 'No') instead of 1 or 0, it automatically encodes the label into 1 and 0 and displays the mapping (0 : No, 1 : Yes) for reference. In this tutorial, no label encoding is required since the target variable is of numeric type.

- **Original data shape:** Shape of the original data prior to any transformations.
- **Transformed train set shape :** Shape of transformed train set
- **Transformed test set shape :** Shape of transformed test set
- **Numeric features :** The number of features considered as numerical.
- **Categorical features :** The number of features considered as categorical.

PyCaret has two set of API's that you can work with. (1) Functional (as seen above) and (2) Object Oriented API.

With Object Oriented API instead of executing functions directly you will import a class and execute methods of class.

```
# import ClassificationExperiment and init the class
from pycaret.classification import ClassificationExperiment
exp = ClassificationExperiment()
```

```
# check the type of exp
type(exp)
```

```
pycaret.classification.oop.ClassificationExperiment
def __init__() -> None

Class for all standard transformation steps.
```

```
# init setup on exp
exp.setup(data, target = 'loan', session_id = 123)
```

	Description	Value
0	Session id	123
1	Target	loan
2	Target type	Binary
3	Target mapping	no: 0, yes: 1
4	Original data shape	(904, 17)
5	Transformed data shape	(904, 49)
6	Transformed train set shape	(632, 49)
7	Transformed test set shape	(272, 49)
8	Numeric features	7
9	Categorical features	9
10	Preprocess	True
11	Imputation type	simple
12	Numeric imputation	mean
13	Categorical imputation	mode
14	Maximum one-hot encoding	25
15	Encoding method	None
16	Fold Generator	StratifiedKFold
17	Fold Number	10
18	CPU Jobs	-1
19	Use GPU	False
20	Log Experiment	False
21	Experiment Name	clf-default-name
22	USI	2e29

```
<pycaret.classification.oop.ClassificationExperiment at 0x7c9331bbe7d0>
```

You can use any of the two method i.e. Functional or OOP and even switch back and forth between two set of API's. The choice of method will not impact the results and has been tested for consistency.

Compare Models

This function trains and evaluates the performance of all the estimators available in the model library using cross-validation. The output of this function is a scoring grid with average cross-validated scores. Metrics evaluated during CV can be accessed using the `get_metrics` function. Custom metrics can be added or removed using `add_metric` and `remove_metric` function.

```
# compare baseline models
best = compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
dummy	Dummy Classifier	0.8402	0.5000	0.8402	0.7060	0.7672	0.0000	0.0000	0.1510
xgboost	Extreme Gradient Boosting	0.8387	0.6398	0.8387	0.8011	0.8036	0.1850	0.2172	0.4330
rf	Random Forest Classifier	0.8371	0.6173	0.8371	0.7223	0.7684	0.0073	0.0158	0.4020
gbc	Gradient Boosting Classifier	0.8355	0.6433	0.8355	0.7799	0.7879	0.1062	0.1407	0.3200
ridge	Ridge Classifier	0.8339	0.5896	0.8339	0.7051	0.7641	-0.0114	-0.0190	0.1570
lr	Logistic Regression	0.8324	0.6020	0.8324	0.7473	0.7713	0.0239	0.0461	0.9450
knn	K Neighbors Classifier	0.8275	0.5639	0.8275	0.7722	0.7808	0.0803	0.1107	0.3040
lda	Linear Discriminant Analysis	0.8229	0.5879	0.8229	0.7453	0.7703	0.0284	0.0406	0.1610
ada	Ada Boost Classifier	0.8197	0.6098	0.8197	0.7749	0.7863	0.1201	0.1381	0.2940
lightgbm	Light Gradient Boosting Machine	0.8197	0.6450	0.8197	0.7818	0.7830	0.1042	0.1349	0.6110
et	Extra Trees Classifier	0.8118	0.5463	0.8118	0.7448	0.7653	0.0246	0.0382	0.3680
dt	Decision Tree Classifier	0.7579	0.5754	0.7579	0.7708	0.7629	0.1435	0.1449	0.1580
svm	SVM - Linear Kernel	0.7467	0.6099	0.7467	0.7686	0.7376	0.1106	0.1219	0.1610
nb	Naive Bayes	0.4572	0.6124	0.4572	0.7720	0.5133	0.0629	0.0966	0.1560
qda	Quadratic Discriminant Analysis	0.2753	0.5584	0.2753	0.7832	0.2677	0.0205	0.0623	0.2580

```
# compare models using OOP
exp.compare_models()
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
dummy	Dummy Classifier	0.8402	0.5000	0.8402	0.7060	0.7672	0.0000	0.0000	0.1540
xgboost	Extreme Gradient Boosting	0.8387	0.6398	0.8387	0.8011	0.8036	0.1850	0.2172	0.4110
rf	Random Forest Classifier	0.8371	0.6173	0.8371	0.7223	0.7684	0.0073	0.0158	0.3530
gbc	Gradient Boosting Classifier	0.8355	0.6433	0.8355	0.7799	0.7879	0.1062	0.1407	0.3230
ridge	Ridge Classifier	0.8339	0.5896	0.8339	0.7051	0.7641	-0.0114	-0.0190	0.1580
lr	Logistic Regression	0.8324	0.6020	0.8324	0.7473	0.7713	0.0239	0.0461	0.3190
knn	K Neighbors Classifier	0.8275	0.5639	0.8275	0.7722	0.7808	0.0803	0.1107	0.2390
lda	Linear Discriminant Analysis	0.8229	0.5879	0.8229	0.7453	0.7703	0.0284	0.0406	0.1590
ada	Ada Boost Classifier	0.8197	0.6098	0.8197	0.7749	0.7863	0.1201	0.1381	0.5750
lightgbm	Light Gradient Boosting Machine	0.8197	0.6450	0.8197	0.7818	0.7830	0.1042	0.1349	0.6470
et	Extra Trees Classifier	0.8118	0.5463	0.8118	0.7448	0.7653	0.0246	0.0382	0.3310
dt	Decision Tree Classifier	0.7579	0.5754	0.7579	0.7708	0.7629	0.1435	0.1449	0.1940
svm	SVM - Linear Kernel	0.7467	0.6099	0.7467	0.7686	0.7376	0.1106	0.1219	0.1560
nb	Naive Bayes	0.4572	0.6124	0.4572	0.7720	0.5133	0.0629	0.0966	0.2670
qda	Quadratic Discriminant Analysis	0.2753	0.5584	0.2753	0.7832	0.2677	0.0205	0.0623	0.2110

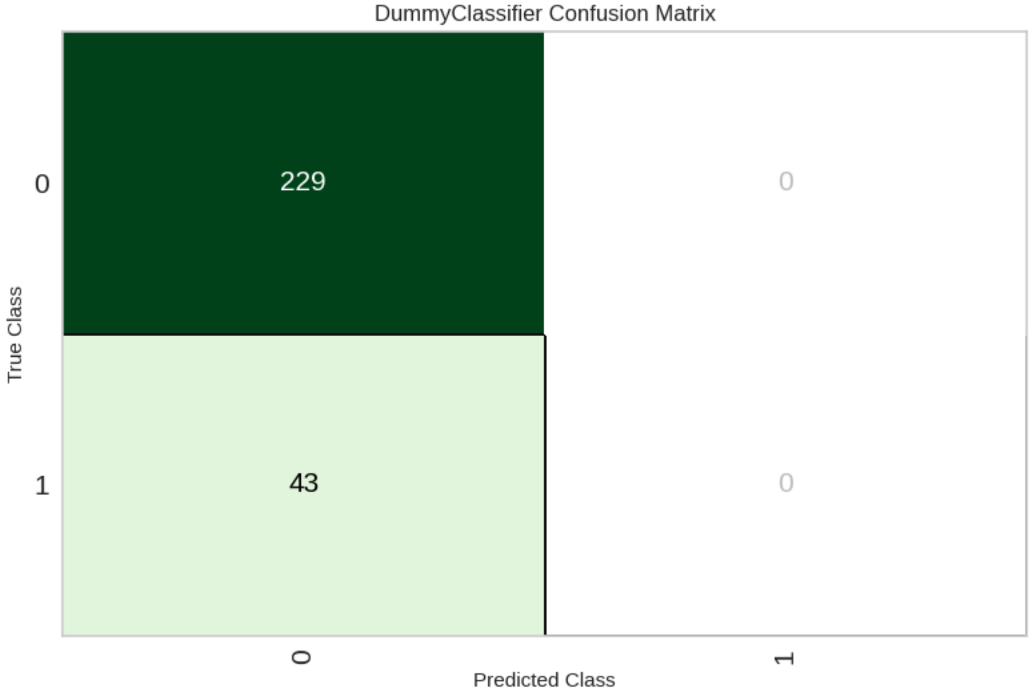
```
DummyClassifier
DummyClassifier(constant=None, random_state=123, strategy='prior')
```

Notice that the output between functional and OOP API is consistent. Rest of the functions in this notebook will only be shown using functional API only.

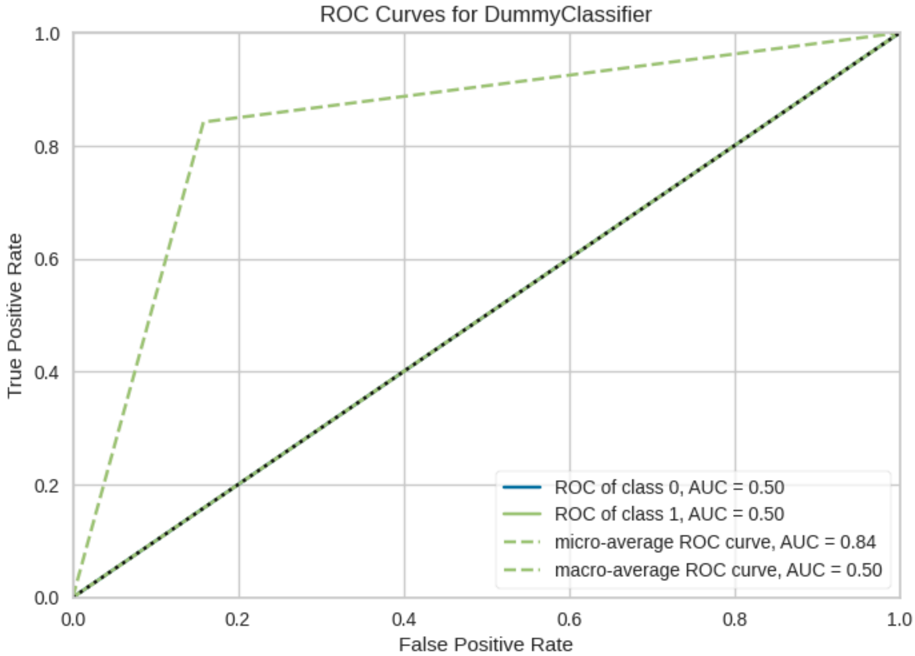
Analyze Model

You can use the `plot_model` function to analyzes the performance of a trained model on the test set. It may require re-training the model in certain cases.

```
# plot confusion matrix
plot_model(best, plot = 'confusion_matrix')
```




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



An alternate to `plot_model` function is `evaluate_model`. It can only be used in Notebook since it uses ipywidget.

```
evaluate_model(best)
```

Plot Type:

Pipeline Plot	Hyperparameters	AUC	Confusion Matrix	Threshold	Precision Recall
Prediction Error	Class Report	Feature Selection	Learning Curve	Manifold Learning	Calibration Curve
Validation Curve	Dimensions	Feature Importance	Feature Importance…	Decision Boundary	Lift Chart
Gain Chart	Decision Tree	KS Statistic Plot			

Prediction

The `predict_model` function returns `prediction_label` and `prediction_score` (probability of the predicted class) as new columns in dataframe. When data is `None` (default), it uses the test set (created during the setup function) for scoring.

```
# predict on test set
holdout_pred = predict_model(best)
```

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Dummy Classifier	0.8419	0.5000	0.8419	0.7088	0.7697	0.0000	0.0000

```
# show predictions df
holdout_pred.head()
```

	age	job	marital	education	default	balance	housing	contact	day	month	duration	campaign	pdays	previous	pou
20525	34	management	single	tertiary	no	2835	yes	cellular	12	aug	107	2	-1	0	unk
33174	30	management	married	tertiary	no	860	yes	cellular	20	apr	185	1	-1	0	unk
25469	50	housemaid	divorced	secondary	no	0	no	cellular	19	nov	126	1	-1	0	unk
20867	31	technician	married	secondary	no	-302	no	cellular	13	aug	195	2	-1	0	unk
25416	31	admin.	married	secondary	no	3584	yes	cellular	18	nov	467	1	-1	0	unk

The same function works for predicting the labels on unseen dataset. Let's create a copy of original data and drop the `Class variable`. We can then use the new data frame without labels for scoring.

```
# copy data and drop Class variable

new_data = data.copy()
new_data.drop('loan', axis=1, inplace=True)
new_data.head()
```

	age	job	marital	education	default	balance	housing	contact	day	month	duration	campaign	pdays	previous	pou
7281	56	technician	married	secondary	no	589	yes	unknown	29	may	535	2	-1	0	unl
19469	37	management	married	tertiary	no	649	no	cellular	7	aug	64	2	-1	0	unl
31637	27	unemployed	single	secondary	no	1972	no	cellular	6	apr	97	1	-1	0	unl
22484	43	management	married	tertiary	no	1	no	cellular	22	aug	239	4	-1	0	unl
35919	58	retired	divorced	secondary	no	-808	yes	cellular	8	may	75	4	-1	0	unl

```
# predict model on new_data
predictions = predict_model(best, data = new_data)
predictions.head()
```

	age	job	marital	education	default	balance	housing	contact	day	month	duration	campaign	pdays	previous	pou
7281	56	technician	married	secondary	no	589	yes	unknown	29	may	535	2	-1	0	unl
19469	37	management	married	tertiary	no	649	no	cellular	7	aug	64	2	-1	0	unl
31637	27	unemployed	single	secondary	no	1972	no	cellular	6	apr	97	1	-1	0	unl
22484	43	management	married	tertiary	no	1	no	cellular	22	aug	239	4	-1	0	unl
35919	58	retired	divorced	secondary	no	-808	yes	cellular	8	may	75	4	-1	0	unl

Save Model

Finally, you can save the entire pipeline on disk for later use, using pycaret's `save_model` function.

```
# save pipeline
save_model(best, 'my_first_pipeline')
```

```
Transformation Pipeline and Model Successfully Saved
(Pipeline(memory=Memory(location=None),
          steps=[('label_encoding',
                  TransformerWrapperWithInverse(exclude=None, include=None,
                                                transformer=LabelEncoder())),
                 ('numerical_imputer',
                  TransformerWrapper(exclude=None,
                                     include=['age', 'balance', 'day',
                                              'duration', 'campaign', 'pdays',
                                              'previous'],
                                     transformer=SimpleImputer(add_indicator=False,
                                                               copy=True,
                                                               fill_value=None,
                                                               keep...
                                     include=['job', 'marital', 'education',
                                              'contact', 'month', 'poutcome'],
                                     transformer=OneHotEncoder(cols=['job',
                                                                     'marital',
                                                                     'education',
                                                                     'contact',
                                                                     'month',
                                                                     'poutcome'],
                                                               drop_invariant=False,
                                                               handle_missing='return_nan',
                                                               handle_unknown='value',
                                                               return_df=True,
                                                               use_cat_names=True,
                                                               verbose=0))),
                 ('trained_model',
                  DummyClassifier(constant=None, random_state=123,
                                  strategy='prior'))],
          verbose=False),
 'my_first_pipeline.pkl')
```

```
# load pipeline
loaded_best_pipeline = load_model('my_first_pipeline')
loaded_best_pipeline
```

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

Pipeline

label_encoding: TransformerWrapperWithInverse