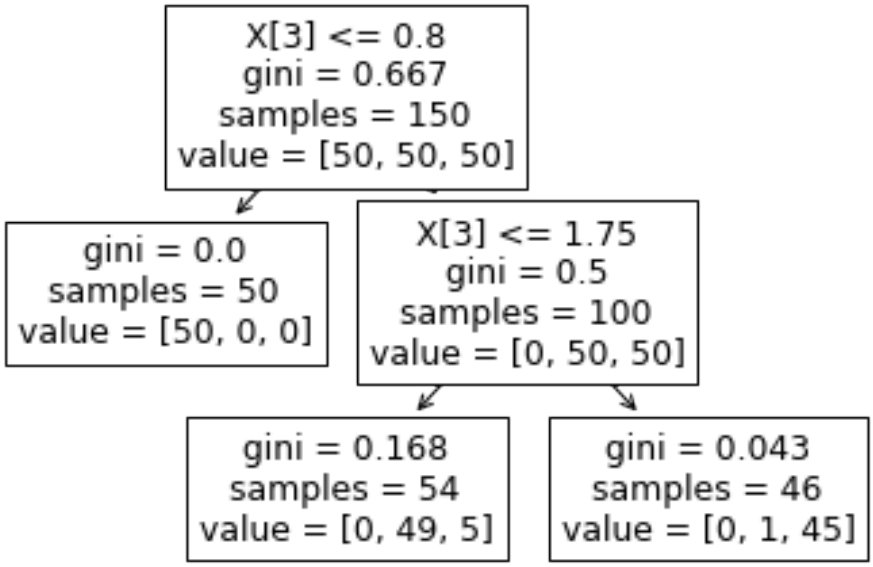
```
!pip install dtreeviz

import graphviz.backend as be

from sklearn.datasets import *
from dtreeviz.trees import *
from IPython.display import Image, display_svg, SVG

clas = tree.DecisionTreeClassifier()
iris = load_iris()

X_train = iris.data
y_train = iris.target
clas.fit(X_train, y_train)

DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
                       max_depth=None, max_features=None,
max_leaf_nodes=None,
                       min_impurity_decrease=0.0,
min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0,
presort='deprecated',
                       random_state=None, splitter='best')

from sklearn.tree import plot_tree
plot_tree(clas)

[Text(133.92000000000002, 181.2, 'X[3] <= 0.8\ngini = 0.667\nsamples =
150\nvalue = [50, 50, 50]'),
 Text(66.96000000000001, 108.72, 'gini = 0.0\nsamples = 50\nvalue =
[50, 0, 0]'),
 Text(200.88000000000002, 108.72, 'X[3] <= 1.75\ngini = 0.5\nsamples =
100\nvalue = [0, 50, 50]'),
 Text(133.92000000000002, 36.23999999999998, 'gini = 0.168\nsamples =
54\nvalue = [0, 49, 5]'),
 Text(267.84000000000003, 36.23999999999998, 'gini = 0.043\nsamples =
46\nvalue = [0, 1, 45]')]
```



1. Classification

```
viz = dtreeviz(clas,
               X_train,
               y_train,
               feature_names=iris.feature_names,
               class_names=["setosa", "versicolor", "virginica"])
viz
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83:
VisibleDeprecationWarning: Creating an ndarray from ragged nested
sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays
with different lengths or shapes) is deprecated. If you meant to do
this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order)
```

2. Regression

```
regr = tree.DecisionTreeRegressor(max_depth=1)
boston = load_boston()

X_train = boston.data
y_train = boston.target
regr.fit(X_train, y_train)

viz = dtreeviz(regr,
               X_train,
               y_train,
               target_name='price',
               feature_names=boston.feature_names,
               scale=2
```

```
                )
viz
```

3. Horizontal Decision Tree

```
viz = dtreeviz(clas,
               X_train,
               y_train,
               target_name='price',
               feature_names=iris.feature_names,
               class_names=["setosa", "versicolor", "virginica"],
               scale = 1.5,
               orientation='LR')
viz
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83:
VisibleDeprecationWarning: Creating an ndarray from ragged nested
sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays
with different lengths or shapes) is deprecated. If you meant to do
this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order)
```

4. Show prediction path

```
clas = tree.DecisionTreeClassifier()
iris = load_iris()

X_train = iris.data
y_train = iris.target
clas.fit(X_train, y_train)

X = iris.data[np.random.randint(0, len(iris.data)),:]

viz = dtreeviz(clas,
               X_train,
               y_train,
               feature_names=iris.feature_names,
               class_names=["setosa", "versicolor", "virginica"],
               X=X)
viz
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83:
VisibleDeprecationWarning: Creating an ndarray from ragged nested
sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays
with different lengths or shapes) is deprecated. If you meant to do
this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order)
```

```
X
```

```
array([5.5, 2.3, 4. , 1.3])
```

5. Show node number

```
viz = dtreeviz(clas,
               X_train,
               y_train,
               target_name='price',
               feature_names=iris.feature_names,
               class_names=["setosa", "versicolor", "virginica"],
               histtype= 'barstacked',
               scale = 1.5,
```

```
                orientation='LR',
                show_node_labels=True)
viz
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83:
VisibleDeprecationWarning: Creating an ndarray from ragged nested
sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays
with different lengths or shapes) is deprecated. If you meant to do
this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order)
```

6. Without Any graphs

```
viz = dtreeviz(clas,
                X_train,
                y_train,
                target_name='price',
                feature_names=iris.feature_names,
                class_names=["setosa", "versicolor", "virginica"],
                histtype= 'barstacked',
                scale = 1.5,
                orientation='LR',
                fancy=False)
viz
```

7. Show just prediction path, nothing else

```
clas = tree.DecisionTreeClassifier()
iris = load_iris()

X_train = iris.data
y_train = iris.target
clas.fit(X_train, y_train)

X = iris.data[np.random.randint(0, len(iris.data)),:]

viz = dtreeviz(clas,
               X_train,
               y_train,
               target_name='price',
               feature_names=iris.feature_names,
               class_names=["setosa", "versicolor", "virginica"],
               X=X,
               show_just_path=True)
viz
```

```
/usr/local/lib/python3.7/dist-packages/numpy/core/_asarray.py:83:
VisibleDeprecationWarning: Creating an ndarray from ragged nested
sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays
with different lengths or shapes) is deprecated. If you meant to do
this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order)
```

8. Prediction Path in Plain english

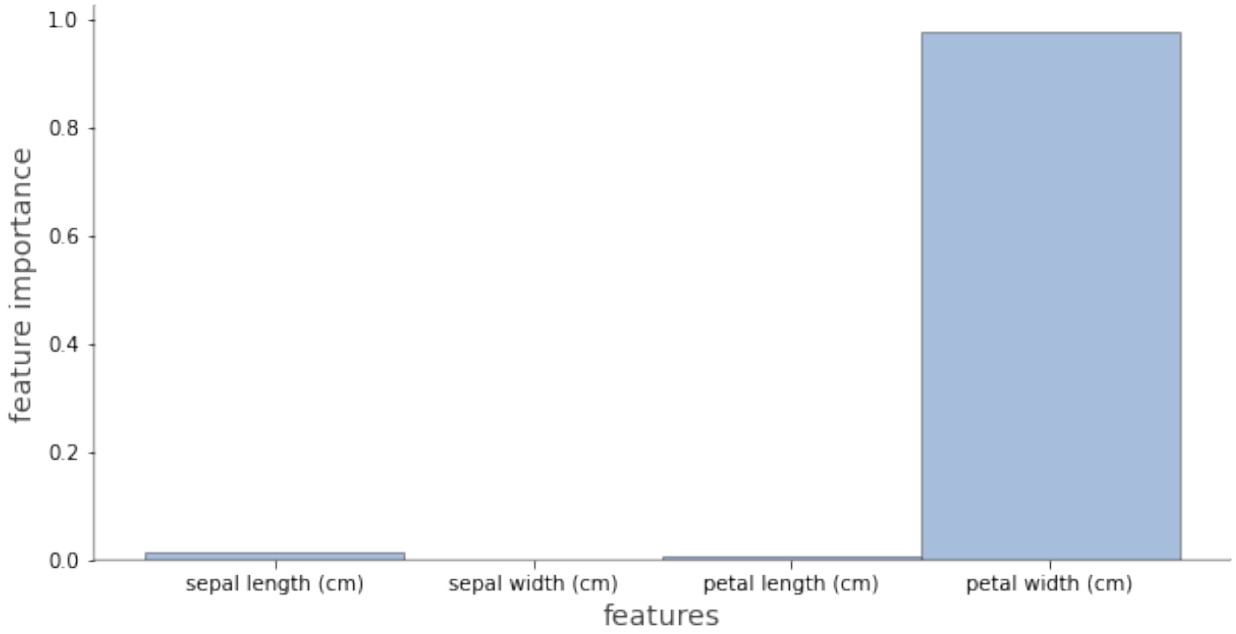
```
print(explain_prediction_path(clas, X,
feature_names=iris.feature_names, explanation_type="plain_english"))
```

```
sepal length (cm) < 5.95
petal length (cm) < 4.85
1.75 <= petal width (cm)
```

9. Feature Importance

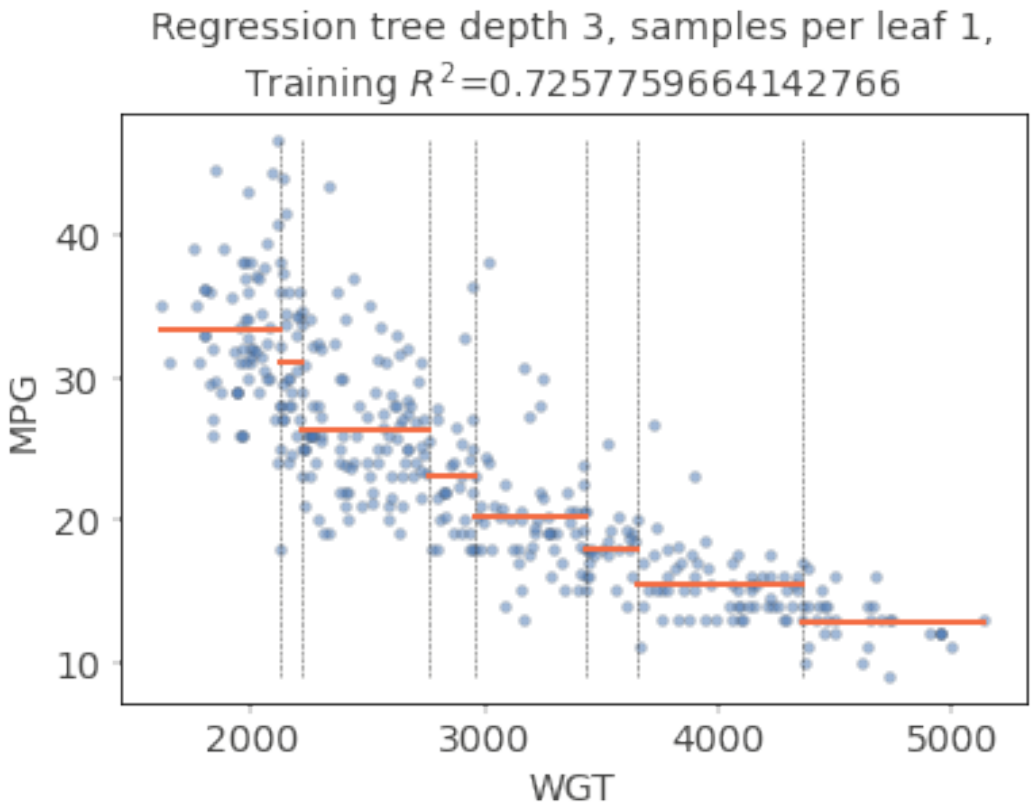
```
print(explain_prediction_path(clas, X,
feature_names=iris.feature_names, explanation_type="sklearn_default"))
```

```
AxesSubplot(0.125,0.125;0.775x0.755)
```



10. Univariate Regression

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor
from dtreeviz.trees import *

df_cars = pd.read_csv("cars.csv")
X, y = df_cars[['WGT']], df_cars['MPG']

dt = DecisionTreeRegressor(max_depth=3, criterion="mae")
dt.fit(X, y)

fig = plt.figure()
ax = fig.gca()
rtreeviz_univar(dt, X, y, 'WGT', 'MPG', ax=ax)
plt.show()
```



11. 3-D Regression

```
df_cars = pd.read_csv("cars.csv")

df_cars
```

```
      MPG  CYL    ENG   WGT
0    18.0    8  307.0  3504
1    15.0    8  350.0  3693
2    18.0    8  318.0  3436
3    16.0    8  304.0  3433
4    17.0    8  302.0  3449
..    ...  ...    ...   ...
387  27.0    4  140.0  2790
388  44.0    4   97.0  2130
389  32.0    4  135.0  2295
390  28.0    4  120.0  2625
391  31.0    4  119.0  2720

[392 rows x 4 columns]
```

```
from mpl_toolkits.mplot3d import Axes3D
from sklearn.tree import DecisionTreeRegressor
from dtreeviz.trees import *
```

```
X = df_cars[['WGT','ENG']]
y = df_cars['MPG']

dt = DecisionTreeRegressor(max_depth=3, criterion="mae")
dt.fit(X, y)

figsize = (6,5)
fig = plt.figure(figsize=figsize)
ax = fig.add_subplot(111, projection='3d')

t = rtreeviz_bivar_3D(dt,
                      X, y,
                      feature_names=['Vehicle Weight', 'Horse Power'],
                      target_name='MPG',
                      fontsize=14,
                      elev=20,
                      azim=25,
                      dist=8.2,
                      show={'splits','title'},
                      ax=ax)
plt.show()
```