Decision Tree for Bangle.isWorn()

In [1]:
```
import pandas as pd

import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report

from dtreeviz.trees import dtreeviz

import matplotlib.pyplot as plt
%config InlineBackend.figure_format = 'svg'

import warnings
warnings.filterwarnings('ignore', category=FutureWarning)
```

In [2]:
```
df = pd.read_csv('worn data.csv')

df.describe()
```

Out[2]:

	Time (s)	Acceleration	Temperature (C)
count	1413.000000	1413.000000	1413.000000
mean	699311.129512	216.694268	25.904105
std	24526.732674	281.251722	1.991462
min	656706.000000	59.000000	22.250000
25%	678153.000000	96.000000	24.500000
50%	699333.000000	101.000000	24.750000
75%	720513.000000	254.000000	27.750000
max	741808.000000	5444.000000	35.500000

In [3]:
```
X = df[['Charging', 'Acceleration']]
y = df[['Worn']]
```

In [4]:
```
%%time

params = {
    'criterion': ['gini', 'entropy', 'log_loss'],
    'splitter': ['best', 'random'],
    'max_depth': [None, 1, 2, 3, 4],
    'max_features': ['auto', 'sqrt', 'log2'],
    'class_weight': [None, 'balanced'],
}

dtc = DecisionTreeClassifier()
clf = GridSearchCV(
    dtc,
    params,
    cv=5,
    scoring='f1',
)
clf.fit(X, y)
dtc = clf.best_estimator_

# print(clf.score(X_test, y_test))
```

```
print(clf.score(X, y))

dtc
```

```
0.9873949579831933
CPU times: user 1.73 s, sys: 0 ns, total: 1.73 s
Wall time: 1.73 s
```

Out[4]:

DecisionTreeClassifier

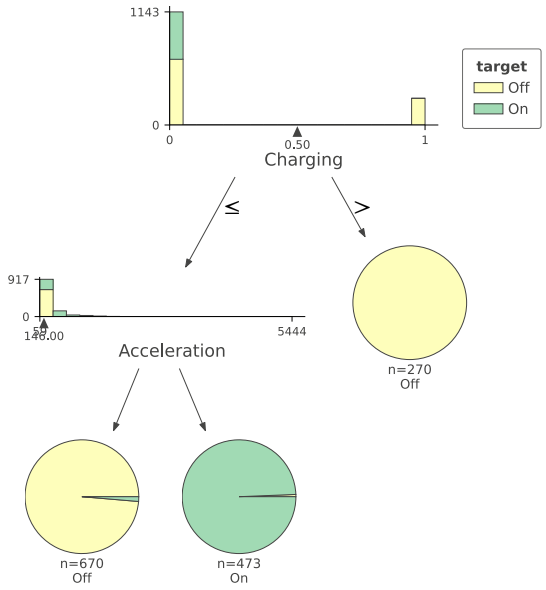
```
DecisionTreeClassifier(class_weight='balanced', max_depth=2,
                       max_features='sqrt')
```

In [5]:

```
viz = dtreeviz(
    dtc,
    X.to_numpy(),
    y.to_numpy().reshape(1,-1)[0],
    target_name='target',
    feature_names=X.columns,
    class_names=['Off', 'On']
)

viz
```

Out[5]:



In [6]:

```
def isWorn(Charging, Acceleration):
    if Charging:
        return 0
    if Acceleration > 146:
        return 1
    return 0
```

In [7]:

```
df['isWorn'] = df.apply(lambda row: isWorn(
    row['Charging'], row['Acceleration']
), axis=1)
```

In [8]:

```
print(classification_report(df[['Worn']], df[['isWorn']]))
```

```
              precision    recall  f1-score   support

       False       0.99      1.00      0.99       934
        True       0.99      0.98      0.99       479

    accuracy                           0.99      1413
   macro avg       0.99      0.99      0.99      1413
weighted avg       0.99      0.99      0.99      1413
```