Copy_of_sklearn_decisiontree_baskin_robbins

June 22, 2023

1 Classification Tree

- Baskin Robbins nutritional information: http://www.baskinrobbins.ca/nutritional-information/
- ref: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[ ]:

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
# %config InlineBackend.figure_format = 'retina'
%config InlineBackend.figure_format = 'svg'
```

[ ]:

```
print(f"pandas  version: {pd.__version__}")
print(f"numpy   version: {np.__version__}")
print(f"seaborn version: {sns.__version__}")
```

[ ]:

```
url = "https://github.com/sophalITC/datasci/raw/master/baskin_robbins.csv"
df = pd.read_csv(url)
df.head()
```

[ ]:

```
df.info()
```

[ ]:

```
df.columns
```

[ ]:

```
cols = [
    "Calories",
    "Total Fat (g)",
    "Trans Fat (g)",
    "Carbohydrates (g)",
    "Sugars (g)",
    "Protein (g)",
]
```

[ ]:

```
fig, ax = plt.subplots(nrows=2, ncols=3, figsize=(20, 9))
ax = ax.ravel()
for i, col in enumerate(cols):
    sns.violinplot(x="Category", y=col, data=df, ax=ax[i])
```

1.1 Draw Decision Tree

[ ]:
```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import (
    confusion_matrix,
    ConfusionMatrixDisplay,
    classification_report,
    accuracy_score,
    precision_score,
    recall_score,
    precision_recall_fscore_support,
    f1_score,
)
```

[ ]:
```
df.columns
```

[ ]:
```
cols = [
    "Calories",
    "Total Fat (g)",
    "Trans Fat (g)",
    "Carbohydrates (g)",
    "Sugars (g)",
    "Protein (g)",
]
```

[ ]:
```
X = df[cols]  # features
y = df["Category"]  # label
```

[ ]:
```
test_size = 0.2
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=test_size, stratify=y, random_state=7
)
```

[ ]:
```
model = DecisionTreeClassifier(criterion="gini")  # gini is a default
# model=DecisionTreeClassifier(criterion='entropy')

model.fit(X_train, y_train)
```

[ ]:
```
X_train.columns
```

[ ]:
```
model.feature_importances_
```

[ ]:

```
fs = pd.Series(model.feature_importances_, index=X_train.columns).sort_values(
    ascending=False
)
fs
```

[ ]:

```
y_train.value_counts()
```

1.1.1 tree diagram (better way)

- **install graphviz on Windows**

1. install graphviz: `pip install -U graphviz`
2. download graphviz: https://www2.graphviz.org/Packages/stable/windows/10/msbuild/Release/Win32/
3. extract and copy to: `C:\Program Files (x86)\Graphviz\bin`
4. add `C:\Program Files (x86)\Graphviz\bin` to `PATH` enviroment variable

[ ]:

```
clf = DecisionTreeClassifier(criterion="gini")
clf.fit(X_train, y_train)
```

[ ]:

```
clf.__dict__
```

[ ]:

```
clf.__dict__["tree_"]
```

[ ]:

```
type(clf)
```

[ ]:

```
clf.__dict__["classes_"]
```

[ ]:

```
from sklearn.tree import export_graphviz
from graphviz import Digraph, Source


def view_tree(model, X, save_tree_img=False):
    estimators = clf
    dot_graph = export_graphviz(
        estimators,
        feature_names=X.columns,
        class_names=model.__dict__["classes_"],
        rounded=True,
        proportion=False,
        precision=2,
        filled=True,
    )
    #     with open(f'tree{tree_index}.dot') as f:
    #         dot_graph = f.read()
    g = Source(dot_graph)
    if save_tree_img:
        g.render(f"tree", format="png", view=False, cleanup=True)
    return g
```

[ ]:

```
view_tree(clf, X_test, save_tree_img=False)
```

Gini impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. … It reaches its minimum (zero) when all cases in the node fall into a single target category. (https://en.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity)

$$\mathrm{I}_G(p) = \sum_{i=1}^{J} \left(p_i \sum_{k \neq i} p_k \right) = \sum_{i=1}^{J} p_i (1 - p_i) = \sum_{i=1}^{J} (p_i - {p_i}^2) = \sum_{i=1}^{J} p_i - \sum_{i=1}^{J} {p_i}^2 = 1 - \sum_{i=1}^{J} {p_i}^2$$

gini value at the top level

$$gini = 1 - \left(\left(\frac{7}{56}\right)^2 + \left(\frac{23}{56}\right)^2 + \left(\frac{26}{56}\right)^2 \right) = .6$$

[ ]:

```
# from sklearn import tree
# tree.plot_tree(clf)
```

[ ]:

```
cm = confusion_matrix(y_test, model.predict(X_test))
cm
```

[ ]:

```
print(classification_report(y, model.predict(X)))
```

[ ]:

```
model.tree_.impurity  # gini
```

[ ]:

```
model.tree_.value
```

[ ]:

```
# This import registers the 3D projection, but is otherwise unused.
!pip install PyQt5
from mpl_toolkits.mplot3d import Axes3D
```

[ ]:

```
# switch to interactive matplotlib
# %matplotlib qt
# %pylab qt

# # switch back to inline mode
%matplotlib inline
```

1.2 3D scatter

[ ]:

```
fig = plt.figure(figsize=(10, 10))
ax = fig.add_subplot(111, projection="3d")
colors = y.map({"light": "green", "std": "blue", "rich": "purple"})
ax.scatter(X["Calories"], X["Total Fat (g)"], X["Sugars (g)"], alpha=0.5,␣
 ↪c=colors)
ax.set_xlabel("Calories")
```

```python
ax.set_ylabel("Total Fat (g)")
ax.set_zlabel("Sugars (g)")
```

[ ]:

```python
predicted = model.predict(X_test)
predicted
```

[ ]:

```python
pd.crosstab(y_test, predicted)
```
