learn-co-curriculum / dsc-oop-sklearn-lab Public

View license

0 stars 31 forks

Star Watch

Code Issues Pull requests Actions Projects Security Insights

solution ...

This branch is 2 commits ahead, 2 commits behind master.

hoffm386 oop with sklearn lab ... on Feb 4 3

View code

README.md

OOP with Scikit-Learn - Lab

Introduction

Now that you have learned some of the basics of object-oriented programming with scikit-learn, let's practice applying it!

Objectives:

In this lesson you will practice:

- Recall the distinction between mutable and immutable types
- Define the four main inherited object types in scikit-learn
- Instantiate scikit-learn transformers and models
- Invoke scikit-learn methods
- Access scikit-learn attributes

Mutable and Immutable Types

For each example below, think to yourself whether it is a mutable or immutable type. Then expand the details tag to reveal the answer.

1. ▶ Python dictionary (click to reveal)
2. ▶ Python tuple (click to reveal)
3. ▶ pandas `DataFrame` (click to reveal)
4. ▶ scikit-learn `OneHotEncoder` (click to reveal)

The Data

For this lab we'll use data from the built-in iris dataset:

```
from sklearn.datasets import load_iris

X, y = load_iris(return_X_y=True, as_frame=True)
```

```
X
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

```
y
```

```
0      0
1      0
2      0
3      0
4      0
      ..
145    2
146    2
147    2
148    2
149    2
Name: target, Length: 150, dtype: int64
```

Scikit-Learn Classes

For the following exercises, follow the documentation link to understand the class you are working with, but **do not** worry about understanding the underlying algorithm. The goal is just to get used to creating and using these types of objects.

Estimators

For all estimators, the steps are:

1. Import the class from the `sklearn` library
2. Instantiate an object from the class
3. Pass in the appropriate data to the `fit` method

`MinMaxScaler` (documentation here)

Import this scaler, instantiate an object called `scaler` with default parameters, and `fit` the scaler on `X`.

```
# Import
from sklearn.preprocessing import MinMaxScaler
# Instantiate
scaler = MinMaxScaler()
# Fit
scaler.fit(X)
```

```
MinMaxScaler()
```

`DecisionTreeClassifier` (documentation here)

Import the classifier, instantiate an object called `clf` (short for "classifier") with default parameters, and `fit` the classifier on `X` and `y`.

```
# Import
from sklearn.tree import DecisionTreeClassifier
# Instantiate
clf = DecisionTreeClassifier()
# Fit
clf.fit(X, y)
```

```
DecisionTreeClassifier()
```

Transformers

One of the two objects instantiated above (`scaler` or `clf`) is a transformer. Which one is it? Consult the documentation.

▶ Hint (click to reveal)

Using the transformer, print out two of the fitted attributes along with descriptions from the documentation.

▶ Hint (click to reveal)

```
# (Answers will vary)
print("Minimum feature seen in the data:", scaler.data_min_)
print("Maximum feature seen in the data:", scaler.data_max_)
```

```
Minimum feature seen in the data: [4.3 2.  1.  0.1]
Maximum feature seen in the data: [7.9 4.4 6.9 2.5]
```

Now, call the `transform` method on the transformer and pass in `X`. Assign the result to `X_scaled`

```
X_scaled = scaler.transform(X)
```

Predictors and Models

The other of the two scikit-learn objects instantiated above (`scaler` or `clf`) is a predictor and a model. Which one is it? Consult the documentation.

▶ Hint (click to reveal)

Using the predictor, print out two of the fitted attributes along with descriptions from the documentation.

```
# (Answers will vary)
print("Number of classes:", clf.n_classes_)
print("Number of features seen:", clf.n_features_in_)
```

```
Number of classes: 3
Number of features seen: 4
```

Now, call the `predict` method on the predictor, passing in `X`. Assign the result to `y_pred`

```
y_pred = clf.predict(X)
```

Now, call the `score` method on the predictor, passing in `X` and `y`

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

```
1.0
```

What does that score represent? Write your answer below

```
"""
According to the documentation, this score represents the mean accuracy
"""
```

Summary

In this lab, you practiced identifying mutable and immutable types as well as identifying transformers, predictors, and models using scikit-learn. You also instantiated scikit-learn objects, invoked the most common scikit-learn methods, and accessed some scikit-learn attributes.

Releases

No releases published

Packages

No packages published

Languages

- ● **Jupyter Notebook** 100.0%