WEEK 10

: Random Forest / Scikit-learn

발표자 : 성신여자대학교 김지한

2020. 07. 03 (금)

공유드라이브 강의요약 폴더에 있는

"challenge2" ,"homework2" 미리 열어놔 주세요!

목차

01. Scikit-learn

- data set 구성계획짜기
- scikit-learn으로 코드구현
- 성능평가

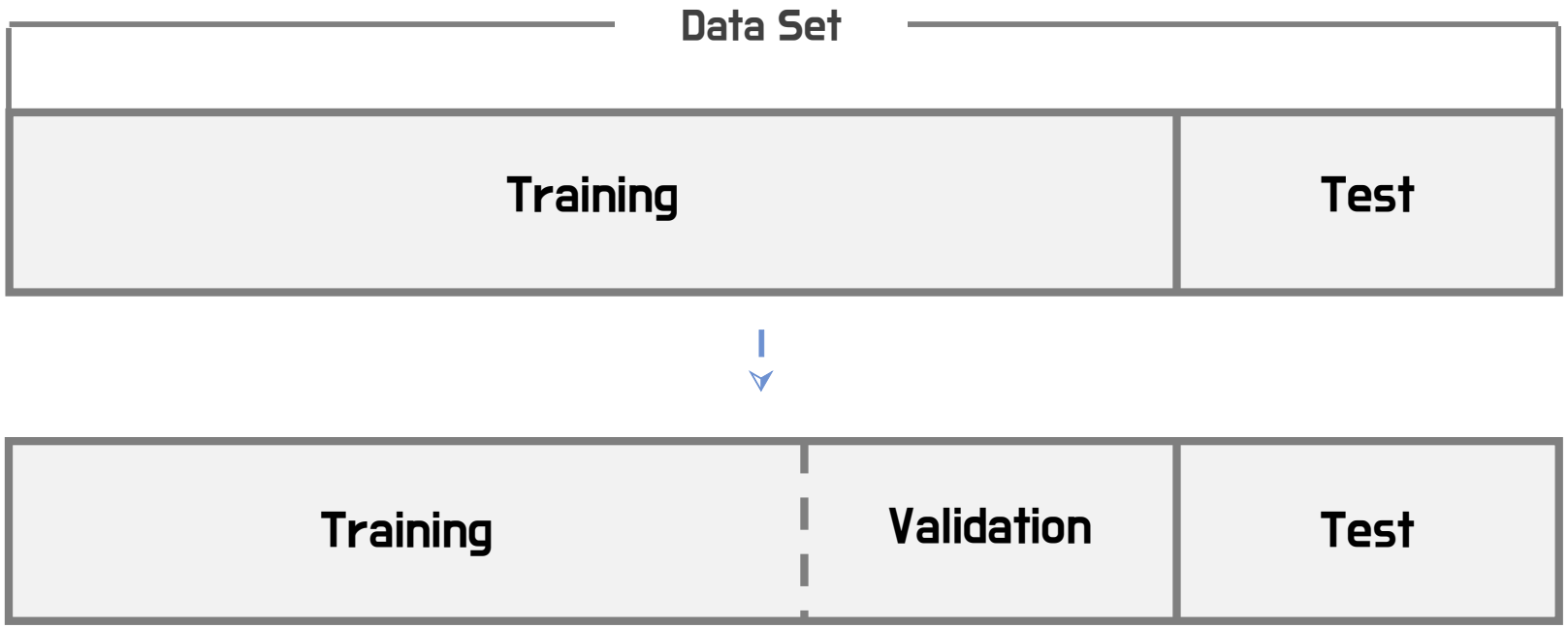
02. 고객분석

- 계획 수립
- 분류 모델 코딩

'싸이킷-런'이란?

데이터를 분석하기 위해 간단하고 효율적인 툴은 제공해주는 라이브러리

Scikit-learn



- Training set : 모델의 학습에 사용되는 데이터
- Validation set : 모델성능을 중간점검하기 위해 임의로 만드는 데이터
- Test set : 모델의 최종성능을 평가하기 위한 데이터

Scikit-learn

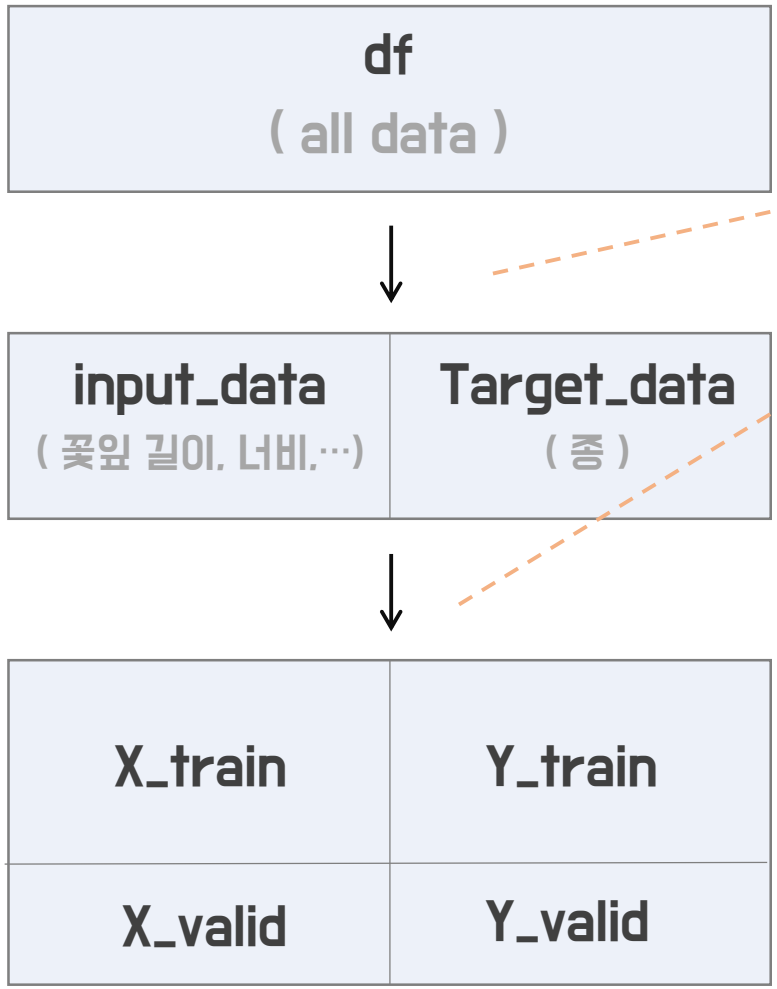
무작위 (randomizing)
75 : 25 (자동비율)

```
from sklearn.model_selection import train_test_split

input_data = df [["SepalLengthCm","SepalWidthCm","PetalLengthCm","PetalWidthCm"]
target_data = df["Species"]

x_train, x_valid, y_train, y_valid = train_test_split(input_data, target_data)

x_train.head()
```

randomizing에 의해
Run을 실행할 때 마다
추출되는 데이터가 다름.

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
88	5.6	3.0	4.1	1.3
38	4.4	3.0	1.3	0.2
85	6.0	3.4	4.5	1.6
108	6.7	2.5	5.8	1.8
119	6.0	2.2	5.0	1.5

Scikit-learn




```python
from sklearn.model_selection import train_test_split

input_data = df [["SepalLengthCm","SepalWidthCm","PetalLengthCm","PetalWidthCm"]
target_data = df["Species"]

x_train, x_valid, y_train, y_valid = train_test_split(input_data, target_data)

x_train.head()
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
88	5.6	3.0	4.1	1.3
38	4.4	3.0	1.3	0.2
85	6.0	3.4	4.5	1.6
108	6.7	2.5	5.8	1.8
119	6.0	2.2	5.0	1.5

Scikit-learn

```
from sklearn.tree import DecisionTreeClassifier
tree = DecisionTreeClassifier()
tree.fit(x_train, y_train)

print("training set accuracy:", tree.score(x_train, y_train))
print("valid set accuracy:", tree.score(x_valid, y_valid))
```

의사결정모델 !

모델 학습!

학습된 모델로 x_train의 결과를 얻은 다음,
그 예측과, y_train을 비교해 정확도를 보여줘!

Valid set 확인!

```
training set accuracy: 1.0
valid set accuracy: 0.9210526315789473
```

→ 왜 Training set의 정확도는 항상 1.00이 나올까?

" 너무 간단한 데이터라
한번만 학습시켜도 정확도를 100% 찍음."

Scikit-learn

* RandomForestClassfier를 이용해 예측해보기

```
from sklearn.ensemble import RandomForestClassifier

forest = RandomForestClassifier(n_estimators=10, max_depth=10)
forest.fit(x_train, y_train)

print("training set accuracy:", forest.score(x_train, y_train))
print("valid set accuracy:", forest.score(x_valid, y_valid))

prediction_by_forest = forest.predict(x_test)
prediction_by_forest
```

```
training set accuracy: 1.0
valid set accuracy: 0.8947368421052632

array(['Iris-setosa', 'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor', 'Iris-virginica', 'Iris-versicolor',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-setosa',
       'Iris-virginica', 'Iris-virginica', 'Iris-setosa', 'Iris-setosa',
       'Iris-virginica', 'Iris-setosa', 'Iris-setosa', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-setosa', 'Iris-setosa', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-virginica', 'Iris-virginica',
       'Iris-virginica', 'Iris-versicolor', 'Iris-setosa',
       'Iris-versicolor', 'Iris-setosa', 'Iris-setosa'], dtype=object)
```

- **n_estimators**

: 트리 개수 (Decision Tree와의 차이점)

: 꼭 n_estimator 숫자가 크다고 좋은 건 아님.

- **Max_depth**

: 최대 깊이

: 깊을 수록 정확도는 좋아지지만.

내가 만든 모델에만 최적화되기 때문에

유연함 X (그 데이터에만 적용되는 모델)

고객분석

* 불러오기

```
import pandas as pd

df = pd.read_csv('data/my_customers.csv')
df
```

	ID	우수고객분류	이름	나이	성별	지역	관심사
0	cdg	1	최도근	10	male	서울	축구
1	leekun	0	이헌영	42	male	서울	축구
2	son7	0	손흥민	38	female	경기	esports
3	parklangers	1	박지성	12	male	서울	축구
4	bluedragon	0	이청용	22	female	경기	esports
5	tuna	1	지동원	11	male	서울	야구
6	key	1	기성용	19	female	경기	축구
7	koojc	0	구자철	28	male	제주	esports

* 문자데이터 → 간단한 숫자데이터로 변환

```
# 성별
df.loc[df["성별"]=="male", "성별"] = 0
df.loc[df["성별"]=="female", "성별"] = 1

# 관심사
df.loc[df["관심사"]=="축구", "관심사"] = 0
df.loc[df["관심사"]=="esports", "관심사"] = 1
df.loc[df["관심사"]=="야구", "관심사"] = 2

# 지역
df.loc[df["지역"]=="서울", "지역"] = 0
df.loc[df["지역"]=="경기", "지역"] = 1
df.loc[df["지역"]=="제주", "지역"] = 2

df
```

	ID	우수고객분류	이름	나이	성별	지역	관심사
0	cdg	1	최도근	10	0	0	0
1	leekun	0	이헌영	42	0	0	0
2	son7	0	손흥민	38	1	1	1
3	parklangers	1	박지성	12	0	0	0
4	bluedragon	0	이청용	22	1	1	1
5	tuna	1	지동원	11	0	0	2
6	key	1	기성용	19	1	1	0
7	koojc	0	구자철	28	0	2	1

고객분석

* 모델 수립 및 학습

In [31]:
```
x_train = df[["나이", "성별", "지역","관심사"]]
y_train = df["우수고객분류"]
```

In [32]:
```
from sklearn.ensemble import RandomForestClassifier

forest = RandomForestClassifier(n_estimators=10, max_depth=10)
forest.fit(x_train, y_train)

print("training set accuracy:", forest.score(x_train, y_train))
```

```
training set accuracy: 1.0
```

고객분석

* 모델 적용

In [33]:
```
df_new = pd.read_csv("data/new_customers.csv")
df_new
```

Out[33]:

	ID	이름	나이	성별	지역	관심사
0	tseter1	이강인	10	male	서울	축구
1	tester2	이승우	25	female	경기	야구

In [34]:
```
# 성별
df_new.loc[df_new["성별"]=="male", "성별"] = 0
df_new.loc[df_new["성별"]=="female", "성별"] = 1

# 관심사
df_new.loc[df_new["관심사"]=="축구", "관심사"] = 0
df_new.loc[df_new["관심사"]=="esports", "관심사"] = 1
df_new.loc[df_new["관심사"]=="야구", "관심사"] = 2

# 지역
df_new.loc[df_new["지역"]=="서울", "지역"] = 0
df_new.loc[df_new["지역"]=="경기", "지역"] = 1
df_new.loc[df_new["지역"]=="제주", "지역"] = 2

df_new
```

Out[34]:

	ID	이름	나이	성별	지역	관심사
0	tseter1	이강인	10	0	0	0
1	tester2	이승우	25	1	1	2

In [35]:
```
test_data = df_new[["나이", "성별", "지역","관심사"]]

prediction = forest.predict(test_data)
prediction
```

Out[35]:
```
array([1, 0], dtype=int64)
```

* 기타

In [36]:
```
# prediction을 표에 추가하고 싶을 땐, 이렇게 적어라!
df_new["prediction"] = prediction
df_new
```

Out[36]:

	ID	이름	나이	성별	지역	관심사	prediction
0	tseter1	이강인	10	0	0	0	1
1	tester2	이승우	25	1	1	2	0

In [37]:
```
target_index = 0
name = df_new.loc[target_index,"이름"]

if prediction[target_index] ==1:
    predict = "우수고객입니다."
else :
    predict = "일반고객입니다."
print(name+"회원님은", predict)
```

이강인회원님은 우수고객입니다.