



데이터 분석

전남대학교 소프트웨어중심대학사업단

유재명

데이터 분석

1. numpy, pandas, matplotlib 다루기
2. 데이터 수집(빅데이터)
3. 데이터 탐색 및 분석 모델 구축해 보기

Numpy

NUMPY

■ numpy

- Numpy에서는 array라는 배열로 계산 수행
- 모듈 추가 : `import numpy as np`

```
import numpy as np  
arr = np.array([1,2,3,4,5])
```

크기는 `shape`로 알 수 있습니다

```
arr.shape
```

```
(5,)
```

array에 대한 덧셈 연산을 해 봅시다

```
arr + 2
```

```
array([3, 4, 5, 6, 7])
```

제곱근을 취해 봅시다

```
np.sqrt(arr)
```

```
array([1.          , 1.41421356, 1.73205081, 2.          , 2.23606798])
```

NUMPY

■ 슬라이스로 요소 추출

```
print(arr)
print('-----')
arr[:2]
```

```
[1 2 3 4 5]
```

```
-----
```

```
array([1, 2])
```

■ Arrange 함수 사용

```
np.arange(5)
```

```
array([0, 1, 2, 3, 4])
```

■ Linspace(start, end, 분할) 함수 사용

```
np.linspace(0, 3, 11)
```

```
array([0. , 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.4, 2.7, 3. ])
```

NUMPY

■ 2차원 생성

2차원 array도 간단히 생성합니다

```
arr2 = np.array([[1, 3],  
                 [5, 7]])
```

```
arr2
```

```
array([[1, 3],  
       [5, 7]])
```

■ 저장 save

save 함수로 array를 저장할 수 있습니다

첫 번째 인수는 저장되는 이름이고, 두 번째 인수가 저장되는 array입니다

```
np.save('test.npy', arr2)
```

■ 불러오기 load

읽어들이는 것은 load 함수로 가능합니다

```
np.load('test.npy')
```

```
array([[1, 3],  
       [5, 7]])
```


NUMPY

■ 배열 재구성 numpy.array().reshape(행, 열)

```
import numpy as np
ar = np.array([1,2,3,4,5,6,7,8,9])
arr2 = ar.reshape(3,3)
arr2
```

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

```
arr3 = ar.reshape(9,1)
arr3
```

```
array([[1],
       [2],
       [3],
       [4],
       [5],
       [6],
       [7],
       [8],
       [9]])
```

NUMPY

- 초기값 지정 `numpy.zeros(튜플(행, 열))`, `numpy.ones(튜플(행, 열))`

```
ar4 = np.zeros((2,3))  
ar4
```

```
array([[0., 0., 0.],  
       [0., 0., 0.]])
```

```
ar4 = np.ones((2,3))  
ar4
```

```
array([[1., 1., 1.],  
       [1., 1., 1.]])
```

- 슬라이싱

```
arr2
```

```
array([[1, 2, 3],  
       [4, 5, 6],  
       [7, 8, 9]])
```

```
print('ar2 배열\n',arr2)  
ar5 = arr2[1:2,1:2]  
print('ar5 배열\n',ar5)  
ar6 = arr2[1,:]  
print('ar6 배열\n',ar6)
```

```
ar2 배열  
[[1 2 3]  
 [4 5 6]  
 [7 8 9]]  
ar5 배열  
[[5]]  
ar6 배열  
[4 5 6]
```

- 행렬곱(dot)

```
print(arr2)  
ar7 = arr2[0:2,0:2]  
print('ar7\n',ar7)  
ar8 = arr2[1:3,1:3]  
print('ar8\n',ar8)  
ar9 = np.dot(ar7,ar8)  
print('ar7 dot ar8\n',ar9)
```

```
[[1 2 3]  
 [4 5 6]  
 [7 8 9]]  
ar7  
[[1 2]  
 [4 5]]  
ar8  
[[5 6]  
 [8 9]]  
ar7 dot ar8  
[[21 24]  
 [60 69]]
```


Pandas

PANDAS

- pandas 임포트: `import pandas as pd`
- pandas 1차원 자료구조: `Series`
- pandas 2차원 자료구조: `DataFrame`
- pandas 3차원 자료구조: `Panel`
- pandas 버전: `pd.__version__`

■ pandas 1차원 자료구조: Series

```
import pandas as pd
data1 = [10,20,30,40,50]
data2 = ['1반', '2반', '3반', '4반', '5반']
sr1 = pd.Series(data1)
print('sr1\n',sr1)
sr2 = pd.Series(data2)
print('sr2\n',sr2)
sr3 = pd.Series([101,102,103,104,105])
print('sr3\n',sr3)
sr4 = pd.Series(['월', '화', '수', '목', '금'])
print('sr4\n',sr4)
```

```
sr1
0    10
1    20
2    30
3    40
4    50
dtype: int64
sr2
0    1반
1    2반
2    3반
3    4반
4    5반
dtype: object
sr3
0    101
1    102
2    103
3    104
4    105
dtype: int64
sr4
0    월
1    화
2    수
3    목
4    금
dtype: object
```

■ pandas 1차원 자료구조: Series

```
import pandas as pd
data1 = [10,20,30,40,50]
data2 = ['1반', '2반', '3반', '4반', '5반']
```

```
sr5 = pd.Series(data1, index=[1000,1001,1002,1003,1004])
print('sr5\n', sr5)
```

```
sr5
1000    10
1001    20
1002    30
1003    40
1004    50
dtype: int64
```

```
sr6 = pd.Series(data1, index=data2)
print('sr6\n', sr6)
```

```
sr6
1반     10
2반     20
3반     30
4반     40
5반     50
dtype: int64
```

■ pandas 1차원 자료구조: Series

```
import pandas as pd
data1 = [10,20,30,40,50]
data2 = ['1반', '2반', '3반', '4반', '5반']
```

```
sr6 = pd.Series(data1,index=data2)
print('sr6\n',sr6)
```

```
sr6
1반    10
2반    20
3반    30
4반    40
5반    50
dtype: int64
```

인덱싱, 슬라이싱

```
# 인덱싱
print('sr6.iloc[2]\n',sr6.iloc[2])
print('sr6["3반"]\n',sr6['3반'])
print('sr6.iloc[-1]\n',sr6.iloc[-1])
```

```
# 슬라이싱
print('sr6.iloc[3:5]\n',sr6.iloc[3:5])
```

```
sr6.iloc[2]
30
sr6["3반"]
30
sr6.iloc[-1]
50
sr6.iloc[3:5]
4반    40
5반    50
dtype: int64
```

- pandas 1차원 자료구조: Series
- 인덱싱 및 값 구하기

```
# 인덱스 구하기  
sr6.index
```

```
Index(['1반', '2반', '3반', '4반', '5반'], dtype='object')
```

```
# 값 구하기  
sr6.values
```

```
array([10, 20, 30, 40, 50], dtype=int64)
```


- pandas 1차원 자료구조: Series
- 연산

```
sr1
0    10
1    20
2    30
3    40
4    50
dtype: int64
sr2
0    1반
1    2반
2    3반
3    4반
4    5반
dtype: object
sr3
0    101
1    102
2    103
3    104
4    105
dtype: int64
sr4
0    월
1    화
2    수
3    목
4    금
dtype: object
```

```
# 동일한 숫자 형식
sr1 + sr3
```

```
0    111
1    122
2    133
3    144
4    155
dtype: int64
```

```
# 동일한 문자열 형식
sr2 + sr4
```

```
0    1반월
1    2반화
2    3반수
3    4반목
4    5반금
dtype: object
```

■ pandas 2차원 자료구조: DataFrame

```
import pandas as pd
```

```
# DataFrame 생성
```

```
테스트 = pd.DataFrame({'온도': [18, 19, 20, 21, 22, 22.3, 25]})
```

```
테스트
```

	온도
0	18.0
1	19.0
2	20.0
3	21.0
4	22.0
5	22.3
6	25.0

```
data = {
    'year' : [2019, 2020, 2021, 2022, 2023, 2024],
    'sales' : [ 342, 333, 222, 564, 444, 567]
}
df1 = pd.DataFrame(data)
df1
```

	year	sales
0	2019	342
1	2020	333
2	2021	222
3	2022	564
4	2023	444
5	2024	567

■ pandas 2차원 자료구조: DataFrame

```
df2 = pd.DataFrame([[67,85,99],[78,90,95]],index=['중간고사','기말고사'], columns=['1반','2반','3반'])  
df2
```

	1반	2반	3반
중간고사	67	85	99
기말고사	78	90	95

■ pandas 2차원 자료구조: DataFrame

```
data3 = [['20201101', 'Yoo', '90', '95'], ['20201102', 'kim', '93', '87'], ['20201103', 'lee', '87', '98']]
df3 = pd.DataFrame(data3)
df3
```

	0	1	2	3
0	20201101	Yoo	90	95
1	20201102	kim	93	87
2	20201103	lee	87	98

```
df3.columns = ['학번', '이름', '중간고사', '기말고사']
df3
```

	학번	이름	중간고사	기말고사
0	20201101	Yoo	90	95
1	20201102	kim	93	87
2	20201103	lee	87	98

■ pandas 2차원 자료구조: DataFrame

```
df3.columns = ['학번', '이름', '중간고사', '기말고사']
df3
```

	학번	이름	중간고사	기말고사
0	20201101	Yoo	90	95
1	20201102	kim	93	87
2	20201103	lee	87	98

```
df3.head(2)
```

	학번	이름	중간고사	기말고사
0	20201101	Yoo	90	95
1	20201102	kim	93	87

```
df3.tail(2)
```

	학번	이름	중간고사	기말고사
1	20201102	kim	93	87
2	20201103	lee	87	98

```
df3['이름']
```

```
0    Yoo
1    kim
2    lee
Name: 이름, dtype: object
```

- pandas 2차원 자료구조: DataFrame
- 저장하기 : `pd.to_csv('파일명', header=False 또는 True)`

```
df3.to_csv('학생정보1.csv', header=False)
```

	학번	이름	중간고사	기말고사
0	20201101	Yoo	90	95
1	20201102	kim	93	87
2	20201103	lee	87	98

```
df4 = pd.read_csv('학생정보1.csv', encoding='utf-8', index_col=0, engine='python')
df4
```

	학번	이름	중간고사	기말고사
0	20201101	Yoo	90	95
1	20201102	kim	93	87
2	20201103	lee	87	98

```
df3.to_csv('학생정보1.csv', header=True)
```

```
df4 = pd.read_csv('학생정보1.csv', encoding='utf-8', index_col=0, engine='python')
df4
```

	학번	이름	중간고사	기말고사
0	20201101	Yoo	90	95
1	20201102	kim	93	87
2	20201103	lee	87	98

- pandas 2차원 자료구조: DataFrame
- 저장하기 : `pd.to_csv('파일명', header=True)`

```
df3.to_csv('학생정보1.csv', header=True)
```

```
usecols=['학번', '이름', '중간고사', '기말고사']
df4 = pd.read_csv('학생정보1.csv', encoding='utf-8', index_col=0, usecols=usecols, skiprows=[1], engine='python')
df4
```

	이름	중간고사	기말고사
학번			
20201102	kim	93	87
20201103	lee	87	98

python

`df.loc[행_레이블, 열_레이블]`

```
df4.loc[20201103, ['이름', '기말고사']]
```

```
이름      lee
기말고사   98
Name: 20201103, dtype: object
```

python

`df.iloc[행_번호, 열_번호]`

```
df4.iloc[1]
```

```
이름      lee
중간고사   87
기말고사   98
Name: 20201103, dtype: object
```

- pandas 2차원 자료구조: DataFrame
- 저장하기 : `pd.to_csv('파일명', header=True)`

```
usecols=['학번','이름','중간고사','기말고사']
df4 = pd.read_csv('학생정보1.csv',encoding='utf-8',index_col=0,usecols=usecols,skiprows=[1], engine='python')
df4
```

	이름	중간고사	기말고사
학번			
20201102	kim	93	87
20201103	lee	87	98

`df4.index`

`Index([20201102, 20201103], dtype='int64', name='학번')`

`df4.columns`

`Index(['이름', '중간고사', '기말고사'], dtype='object')`

`df4.values`

`array([['kim', 93, 87],
 ['lee', 87, 98]], dtype=object)`

데이터 수집(빅데이터)

데이터 수집(빅데이터)

1. 빅데이터 정보를 활용하는 서비스

- A. 자율주행차(0 ~ 5단계) : 인지 - 판단 - 제어
- B. 커넥티드 카 : 정보통신기술 + 자동차(양방향 인터넷, 모바일 서비스가 가능한 차량)
- C. 스마트 시티 : IoT와 AI, 빅데이터 분석, AR/VR/MR, 건강/교통/교육/기기제어 등의 요소 기술
- D. 스마트 헬스케어 : 종합 건강 정보 빅데이터 구축, 분야별 지식베이스 구축, 진단 및 처방용 AI

2. 빅데이터

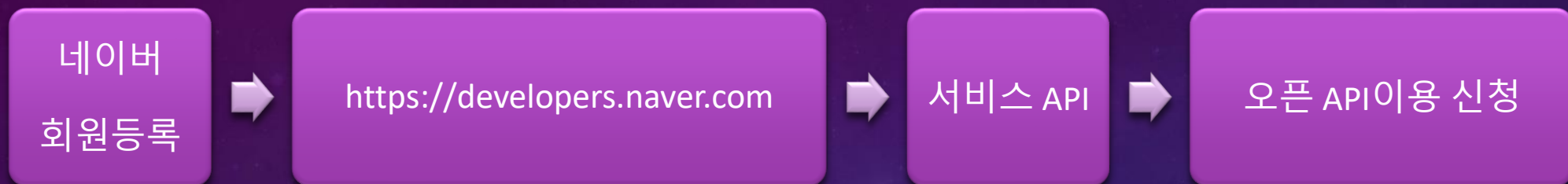
(위키피디아) 기존 데이터베이스 관리 도구의 수집, 저장, 관리, 분석 역량을 넘어서는 대량의 정형 또는 비정형 데이터셋 및 이러한 데이터로부터 가치 추출, 결과 분석 기술
(국가전략위원회) 대용량 데이터를 활용 및 분석하여 가치 있는 정보를 추출하고 생성된 지식을 바탕으로 능동적으로 대응하거나 변화를 예측하기 위한 정보화 기술
(삼성경제연구소) 기존의 관리 및 분석 체계로는 감당할 수 없을 정도의 거대한 데이터 집합으로 대규모 데이터와 관계된 기술 및 도구(수집, 저장, 검색, 공유, 분석, 시각화 등)을 모두 포함
(한국정보화진흥원) 저장, 관리, 분석할 수 있는 범위를 초과하는 규모의 데이터와 이것을 저장, 관리, 분석할 수 있는 하드웨어 및 소프트웨어 기술, 데이터 유통 및 활용하는 전 과정 거대 플랫폼

3. 빅데이터 분류

정형 데이터 : 관계형 데이터베이스처럼 스키마 형식에 맞게 저장(RDB, 스프레드시트 등)
반정형 데이터 : 메타데이터나 스키마 등을 포함(XML, HTML, JSON, 웹문서, 웹로그 등)
비정형 데이터 : 데이터 구조가 일정하지 않음, 예로, SNS, Text문서, 이미지/동영상/음성 데이터, PDF문서 등)

데이터 수집(빅데이터)

■ 웹 크롤링(Naver 오픈API를 활용한 검색)



애플리케이션 이름	<div>nvBig</div> <ul style="list-style-type: none">네이버 로그인할 때 사용자에게 표시되는 이름이므로 서비스 브랜드를 대표할 수 있는 이름으로 가급적 10자 이내로 간결하게 설정해주세요.40자 이내의 영문, 한글, 숫자, 공백문자, 쉼표(,), "/", "-", ".", "_" 만 입력 가능합니다.
사용 API	<div>선택하세요. ▼ ✓</div> <div>검색 ✕</div>
비로그인 오픈 API 서비스 환경	<div>환경 추가 ▼</div> <div>WEB 설정 ✕ ^</div> <div>웹 서비스 URL (최대 10개)</div> <div><div>http://localhost + ✓</div><ul style="list-style-type: none">텍스트 폼 우측 끝의 '+' 버튼을 누르면 행이 추가되며, '-' 버튼을 누르면 행이 삭제됩니다.http와 https는 구분하지 않습니다.www는 빼고 입력해 주세요. 예) http://naver.com서브 도메인이 있으면 대표 도메인명만 입력해 주세요. (예: http://naver.com)하이브리드 앱은 location.href 객체 출력 값을 입력하면 됩니다. (예: file://로컬 URI)</div>

등록

nvBig					
개요	API 설정	멤버관리	로그인 통계	API 통계	Playground(Beta)
애플리케이션 정보					
Client ID	gT18U_ii2Vv0plqYI2jD				
Client Secret	<div>.....</div> <div>보기</div>				

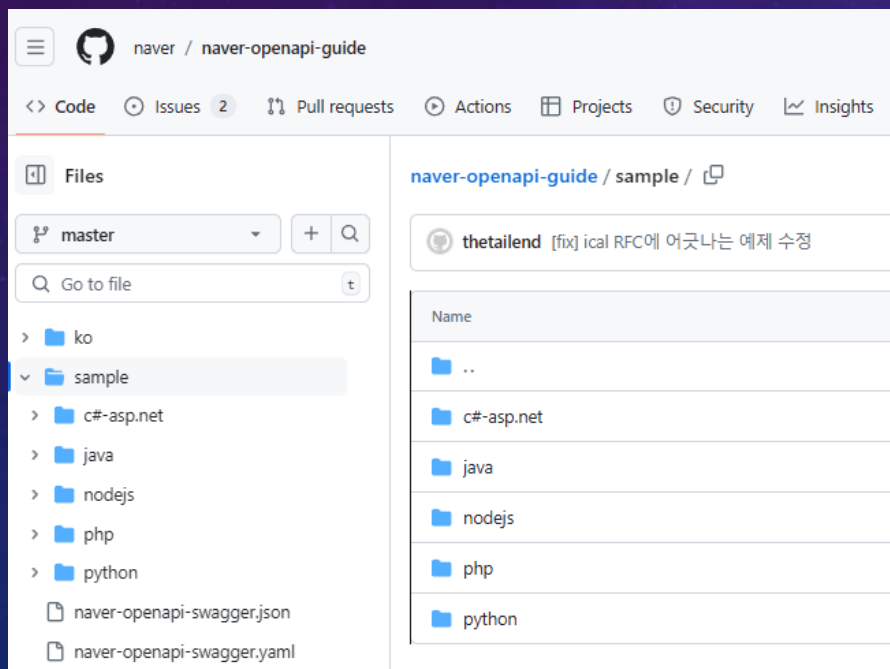
데이터 수집(빅데이터)

■ 웹 크롤링(Naver 오픈API를 활용한 검색) – 비로그인 방식 오픈 API 호출

■ 네이버 검색 구분

- 뉴스(news) : <https://openapi.naver.com/v1/search/news.json>
- 블로그(blog) : <https://openapi.naver.com/v1/search/blog.json>
- 카페(cafearticle) : <https://openapi.naver.com/v1/search/cafearticle.json>
- 영화(movie) : <https://openapi.naver.com/v1/search/movie.json>
- 쇼핑(shop) : <https://openapi.naver.com/v1/search/shop.json>

■ 코드 가이드: <https://github.com/naver/naver-openapi-guide/tree/master/sample>



데이터 수집(빅데이터)

- 웹 크롤링(Naver 오픈API를 활용한 검색)
- 코드 가이드: <https://github.com/naver/naver-openapi-guide/tree/master/sample>

naver-openapi-guide / sample / python / APIExamSearchBlog.py

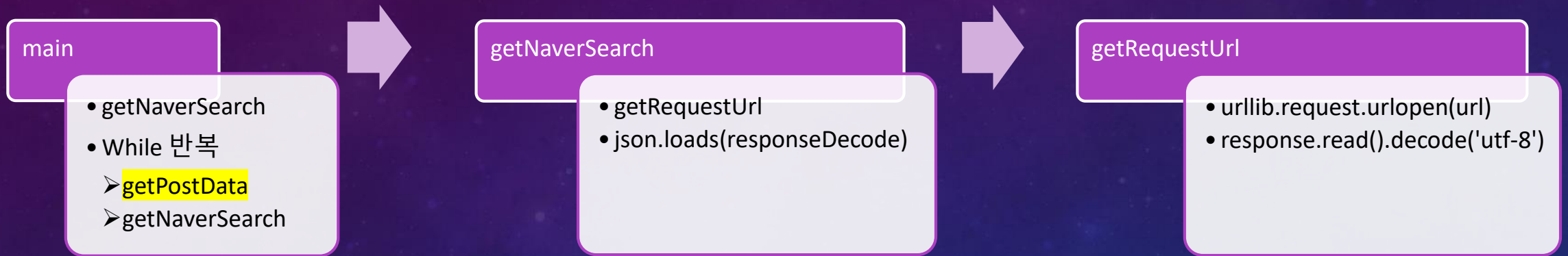
okgosu Open API 예제 코드들

Code Blame 17 lines (17 loc) · 681 Bytes

```
1  import os
2  import sys
3  import urllib.request
4  client_id = "YOUR_CLIENT_ID" # 개발자센터에서 발급받은 Client ID 값
5  client_secret = "YOUR_CLIENT_SECRET" # 개발자센터에서 발급받은 Client Secret 값
6  encText = urllib.parse.quote("출입동")
7  url = "https://openapi.naver.com/v1/search/blog.xml?query=" + encText
8  request = urllib.request.Request(url)
9  request.add_header("X-Naver-Client-Id",client_id)
10 request.add_header("X-Naver-Client-Secret",client_secret)
11 response = urllib.request.urlopen(request)
12 rescode = response.getcode()
13 if(rescode==200):
14     response_body = response.read()
15     print(response_body.decode('utf-8'))
16 else:
17     print("Error Code:" + rescode)
```

데이터 수집(빅데이터)

■ 함수 흐름



데이터 수집(빅데이터)(1)

- 코드 가이드: <https://github.com/naver/naver-openapi-guide/tree/master/sample>

```
import urllib.request
import datetime
import json
client_id = '본인의 client ID' # Naver에서 본인의 ID, SECRET 생성하기
client_secret = '본인 Secret'

def getRequestUrl(url):
    req = urllib.request.Request(url)
    req.add_header('X-Naver-Client-Id', client_id) # 서버에 보내는 요청 객체에 헤더 정보 추가
    req.add_header('X-Naver-Client-Secret', client_secret)
    try:
        response = urllib.request.urlopen(req)
        if response.getcode() == 200:
            print('[%s] Url(%s) Request Success' % (datetime.datetime.now(), url))
            return response.read().decode('utf-8')
    except Exception as e:
        print(e)
        print('[%s] Error for Url : %s' % (datetime.datetime.now(), url))
        return None
```

데이터 수집(빅데이터)(2)

- 코드 가이드: <https://github.com/naver/naver-openapi-guide/tree/master/sample>

```
def getNaverSearch(node,srcText,start,display):
```

```
    base = 'https://openapi.naver.com/v1/search'
```

```
    node = '/%s.json' % node
```

```
    parameters = '?query=%s&start=%s&display=%s' % (urllib.parse.quote(srcText), start, display)
```

```
    url= base + node + parameters
```

```
    responseDecode=getRequestUrl(url)
```

```
    if (responseDecode == None):
```

```
        return None
```

```
    else:
```

```
        return json.loads(responseDecode)
```

```
def getPostData(post, jsonResult, cnt):
```

```
    title = post['title']
```

```
    description = post['description']
```

```
    org_link = post['originallink']
```

```
    link = post['link']
```

```
    pDate = datetime.datetime.strptime(post['pubDate'],'%a, %d %b %Y %H:%M:%S +0900') # 날짜,시간의 문자열을 datetime으로 변환  
                                                # %a(요일), %d(월중 일(01,02,...)), %b(월) -> Mon 19 Aug 2024, 11:34:22.99999 datetime 포맷
```

```
    pDate = pDate.strftime('%Y-%m-%d %H:%M:%S') # datetime의 날짜, 시간 형식을 문자열로 변환
```

```
    jsonResult.append({'cnt':cnt,'title':title,'description':description,'org_link':org_link,'link':link,'pDate':pDate})
```

```
    return
```

■ 네이버 검색 구분

- 뉴스(news) : <https://openapi.naver.com/v1/search/news.json>
- 블로그(blog) : <https://openapi.naver.com/v1/search/blog.json>
- 카페(cafearticle) : <https://openapi.naver.com/v1/search/cafearticle.json>
- 영화(movie) : <https://openapi.naver.com/v1/search/movie.json>
- 쇼핑(shop) : <https://openapi.naver.com/v1/search/shop.json>

데이터 수집(빅데이터)(3)

- 코드 가이드: <https://github.com/naver/naver-openapi-guide/tree/master/sample>

```
def main():
    node = 'news' #크롤링할 대상 : news, blog, cafearticle, movie, shop
    srcText = input('검색어를 입력하시오')
    cnt = 0
    jsonResult = []
    jsonResponse = getNaverSearch(node,srcText,1,100) # start = 1(검색 시작 위치로 1(기본값) ~ 1000(최대값)), display =
100(출력건수)
    print(jsonResponse.keys())
    total = jsonResponse['total']
    while((jsonResponse != None) and (jsonResponse['display'] != 0)):
        for post in jsonResponse['items']:
            cnt += 1
            getPostData(post, jsonResult, cnt)
        start = jsonResponse['start'] + jsonResponse['display'] #
        jsonResponse = getNaverSearch(node,srcText,start,100)

    print('전체 검색: %d 건' % total)
    with open('%s_naver_%s.json' % (srcText,node), 'w',encoding='utf8') as outfile:
        jsonFile = json.dumps(jsonResult, indent=4, sort_keys=True, ensure_ascii = False)
        outfile.write(jsonFile)
    print('가져온 데이터 : %d 건' % cnt)
    print('%s_naver_%s.json Saved' % (srcText,node))
```

데이터 수집(빅데이터)(4)

- 코드 가이드: <https://github.com/naver/naver-openapi-guide/tree/master/sample>

```
if __name__ == '__main__':  
    main()
```


데이터 수집: 공공데이터

■ 공공데이터 활용

<https://www.data.go.kr>
공공데이터 포털 회원가입

오픈 API
(5,029건)

한국문화관광연구원_
출입국관광통계서비스

오픈 API (5,029건)

종류별 검색 등록기관 검색

XML 한국문화관광연구원_출입국관광통계서비스

[출입국관광통계서비스]

==>


신청후 아래 오픈API 활용: <https://data.seoul.go.kr/together/guide/useGuide.do>

Open API 이용방법




열린데이터광장 접속

01




Open API 인증키 신청

02



Open API 검색 / 확인

03



Open API 활용 / 애플리케이션 제작

04



애플리케이션 등록

05

◆ 인증키 발급

- 열린데이터광장에서 제공하는 오픈API를 사용하기 위해서는 먼저 인증키를 발급 받으셔야 합니다.
- 오픈API는 다양한 서비스와 데이터를 좀 더 쉽게 이용할 수 있도록 공개한 개발자를 위한 인터페이스입니다.

◆ 인증키 사용

- 오픈API를 통해서 1회 호출시 최대 1,000건만 요청이 가능하며, 1,000건이 넘는 경우 호출을 나누어 요청하셔야 합니다.(오류 발생 시 제한됩니다.)
- 실시간 지하철 오픈API는 1일 1,000회만 호출이 가능합니다.(인증키 1개당)
- 실시간 지하철 오픈API는 알림사서(웹리미)에 인증키와 함께 서비스를 등록하고 승인 후 호출 제한없이 사용이 가능합니다.

일반 인증키 신청 실시간 지하철 인증키 신청

정보확인: 홈 > 마이페이지 > 데이터 활용 > Open API > 활용신청 현황

이 누리집은 대한민국 공식 전자정부 누리집입니다.

로그아웃 마이페이지

DATA 공공데이터포털 .GO .KR 데이터찾기 국가데이터맵 데이터요청 데이터활용 정보공유

홈 > 마이페이지 > 데이터 활용 > Open API > 활용신청 현황

마이페이지 개발계정 상세보기

데이터 활용 >

데이터 요청 >

나의 문의 >

회원정보 수정 >

기본정보

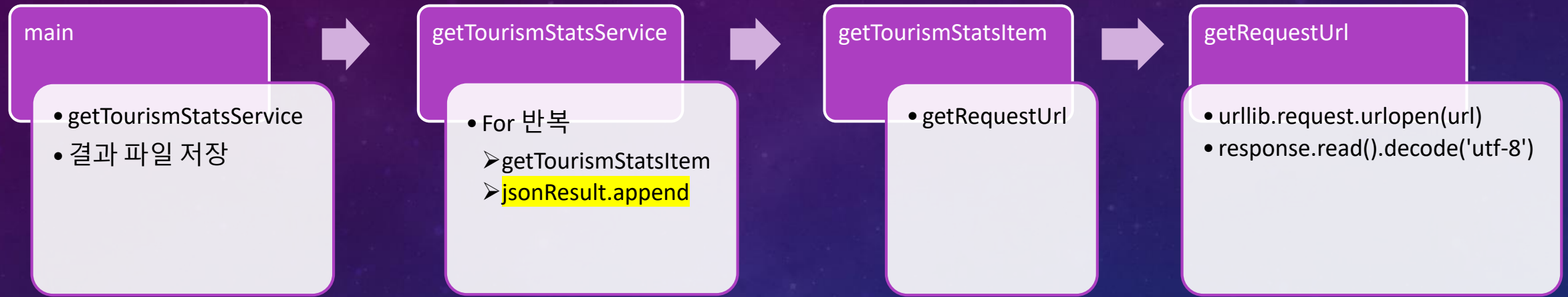
데이터명	한국문화관광연구원_출입국관광통계서비스		
서비스유형	REST	서비스부	자동승인
신청유형	개발계정 활용신청	처리상태	승인
활용기간	2024-04-12 ~ 2026-04-12		

서비스정보

참고문서	한국문화관광연구원_출입국관광통계서비스_활용가이드_v1.2.docx
데이터포맷	XML
End Point	http://openapi.tour.go.kr/openapi/service

데이터 수집: 공공데이터

■ 함수 흐름도



데이터 수집: 공공데이터(1)

```
import urllib.request
import datetime
import json
import pandas as pd
ServiceKey = '공공데이터서비스키'

def getRequestUrl(url):
    req = urllib.request.Request(url)
    try:
        response = urllib.request.urlopen(req)
        if response.getcode() == 200:
            print('[%s] Url(%s) Request Success' % (datetime.datetime.now(),url))
            return response.read().decode('utf-8')
    except Exception as e:
        print(e)
        print('[%s] Error for Url : %s' % (datetime.datetime.now(), url))
        return None
```

데이터 수집: 공공데이터(2)

```
def getTourismStatsItem(yyyymm, nat_cd, ed_cd):  
    base = 'http://openapi.tour.go.kr/openapi/service/EdrcntTourismStatsService/getEdrcntTourismStatsList'  
    parameters = '?_type=json&serviceKey='+ServiceKey #인증키  
    parameters += '&YM='+yyyymm  
    parameters += '&NAT_CD='+nat_cd  
    parameters += '&ED_CD='+ed_cd  
    url= base + parameters  
    responseDecode=getRequestUrl(url)  
    if (responseDecode == None):  
        return None  
    else:  
        return json.loads(responseDecode)
```

데이터 수집: 공공데이터(3)

```
def getTourismStatsService(nat_cd, ed_cd, nStartYear, nEndYear):
    jsonResult = []
    result = []
    dataEND = ""
    for year in range(nStartYear, nEndYear+1):
        for month in range(1, 13):
            yyyyymm = "{0}{1:0>2}".format(str(year), str(month))
            jsonData = getTourismStatsItem(yyyyymm, nat_cd, ed_cd)
            if (jsonData['response']['header']['resultMsg'] == 'OK'):
                if jsonData['response']['body']['items'] == '':
                    dataEND = "{0}{1:0>2}".format(str(year), str(month-1)) # {1:0>글자수}.format(숫자): 글자수보다 적은 부분을 0으로 채움, > 오른쪽정렬, < 왼쪽정렬
                    print('데이터 없음...\n 제공되는 데이터는 %s년 %s월까지입니다.' % (str(year), str(month-1)))
                    break
                print(json.dumps(jsonData, indent=4, sort_keys = True, ensure_ascii = False))
                natName = jsonData['response']['body']['items']['item']['natKorNm']
                natName = natName.replace(' ', '')
                num = jsonData['response']['body']['items']['item']['num']
                ed = jsonData['response']['body']['items']['item']['ed']
                print('[%s_%s: %s]' % (natName, yyyyymm, num))
                print('-----')
                jsonResult.append({'nat_name': natName, 'nat_cd': nat_cd, 'yyyyymm': yyyyymm, 'visit_cnt': num})
                result.append([natName, nat_cd, yyyyymm, num])
    return (jsonResult, result, natName, ed, dataEND)
```

데이터 수집: 공공데이터(4)

```
def main():
    jsonResult = []
    result = []
    print('입국 외국인 통계 데이터')
    nat_cd = input('국가코드(중국:112, 일본:130, 미국: 275): ')
    nStartYear = int(input('입국년도:'))
    nEndYear = int(input('출국년도:'))
    ed_cd = 'E' # E: 방한외래관광객, D: 해외 출국
    jsonResult, result, natName, ed, dataEND = getTourismStatsService(nat_cd, ed_cd, nStartYear, nEndYear)
    #json파일 저장
    with open('%s_%s_%d_%s.json' % (natName,ed,nStartYear,dataEND), 'w',encoding='utf8') as outfile:
        jsonFile = json.dumps(jsonResult, indent=4, sort_keys=True, ensure_ascii = False)
        outfile.write(jsonFile)

    #csv파일 저장
    columns = ["입국자국가", "국가코드", "입국년월", "입국자수"]
    result_df = pd.DataFrame(result,columns = columns)
    result_df.to_csv('%s_%s_%d_%s.csv' % (natName,ed,nStartYear,dataEND), index=False, encoding='utf8')

if __name__ == '__main__':
    main()
```

데이터 탐색, 모델 구축

머신러닝 흐름

1. 과거의 데이터 준비

2. 모델 구조 생성

3. 데이터로 모델 학습(FIT)

4. 결과

상관관계	
온도	판매량
20	40
21	42
22	44

독립변수 종속변수



판매량
예측

필요한 실습 자료

레모네이드(다운로드)(URL) <https://raw.githubusercontent.com/blackdew/tensorflow1/master/csv/lemonade.csv>

■ 사용방법

```
레모네이드 = pd.read_csv(URL)
```

```
독립 = 레모네이드[['온도']]
```

```
종속 = 레모네이드[['판매량']]
```

데이터 탐색, 모델 구축

관련 라이브러리 추가

```
import tensorflow as tf      # 머신러닝: tensorflow
import pandas as pd         # 데이터: Pandas
```

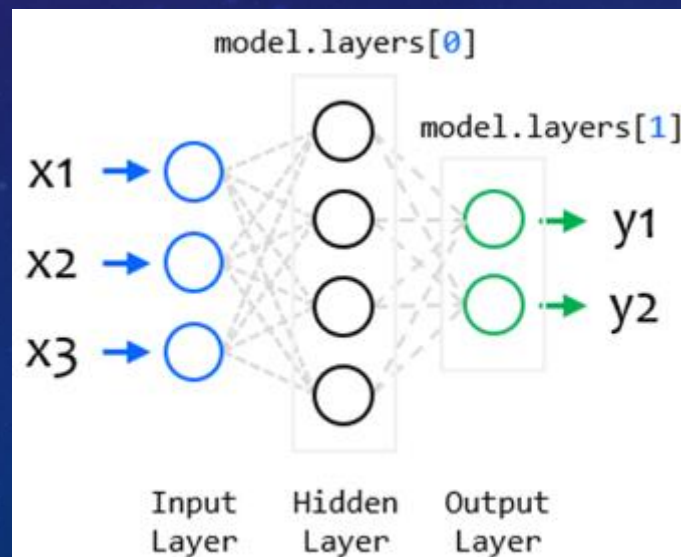
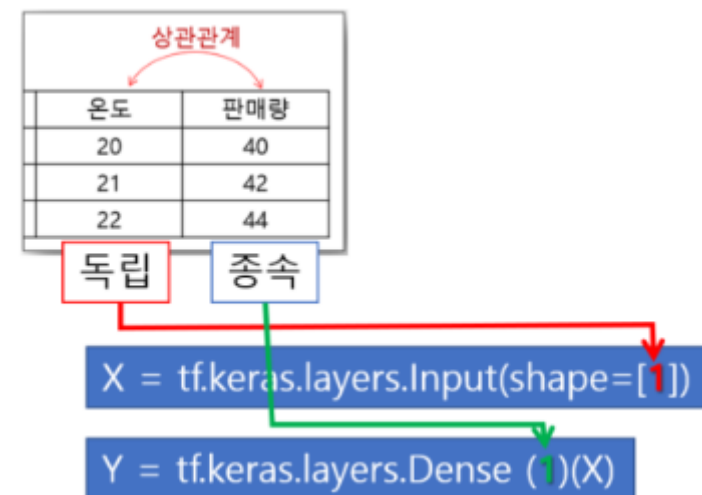
##1. 과거의 데이터 준비

```
레모네이드 = pd.read_csv("https://raw.githubusercontent.com/blackdew/tensorflow1/master/csv/lemonade.csv")
독립 = 레모네이드[['온도']]
종속 = 레모네이드[['판매량']]
print('독립변수:', 독립.shape, '종속변수:', 종속.shape)
```

독립변수: (6, 1) 종속변수: (6, 1)

##2. 모델 구조 생성

```
X = tf.keras.layers.Input(shape=[1])
Y = tf.keras.layers.Dense(1)(X)
model = tf.keras.models.Model(X,Y)
model.compile(loss='mse')          # mse, mean square error
```



데이터 탐색, 모델 구축

##3. 데이터로 모델 학습(fit)

```
model.fit(독립, 종속, epochs=2000, verbose=False) # verbose=False 옵션은 내용보기 안함  
model.fit(독립, 종속, epochs=10)                # default는 내용보기
```

```
Epoch 1/10  
1/1 ————— 0s 14ms/step - loss: 0.0065  
Epoch 2/10  
1/1 ————— 0s 14ms/step - loss: 0.0065  
Epoch 3/10  
1/1 ————— 0s 14ms/step - loss: 0.0064  
Epoch 4/10  
1/1 ————— 0s 13ms/step - loss: 0.0064  
Epoch 5/10  
1/1 ————— 0s 13ms/step - loss: 0.0064  
Epoch 6/10  
1/1 ————— 0s 13ms/step - loss: 0.0064  
Epoch 7/10  
1/1 ————— 0s 13ms/step - loss: 0.0064  
Epoch 8/10  
1/1 ————— 0s 13ms/step - loss: 0.0064  
Epoch 9/10  
1/1 ————— 0s 14ms/step - loss: 0.0064  
Epoch 10/10  
1/1 ————— 0s 14ms/step - loss: 0.0064  
<keras.src.callbacks.history.History at 0x2947ffac8e0>
```

데이터 탐색, 모델 구축

```
##4. 모델을 이용하여 예측
```

```
##DataFrame 생성하여 테스트
```

```
테스트 = pd.DataFrame({'온도':[18,18.2,19.3,19.6]})
```

```
print("온도에 따른 판매량 예측:\n",model.predict(테스트))
```

```
1/1  0s 21ms/step
```

```
온도에 따른 판매량 예측:
```

```
[[36.22327 ]
```

```
[36.614124]
```

```
[38.76381 ]
```

```
[39.35009 ]]
```

```
## 정보확인 : 가중치
```

```
ww =model.get_weights()
```

```
print('기울기',ww[0][0][0],'절편',ww[1][0])
```

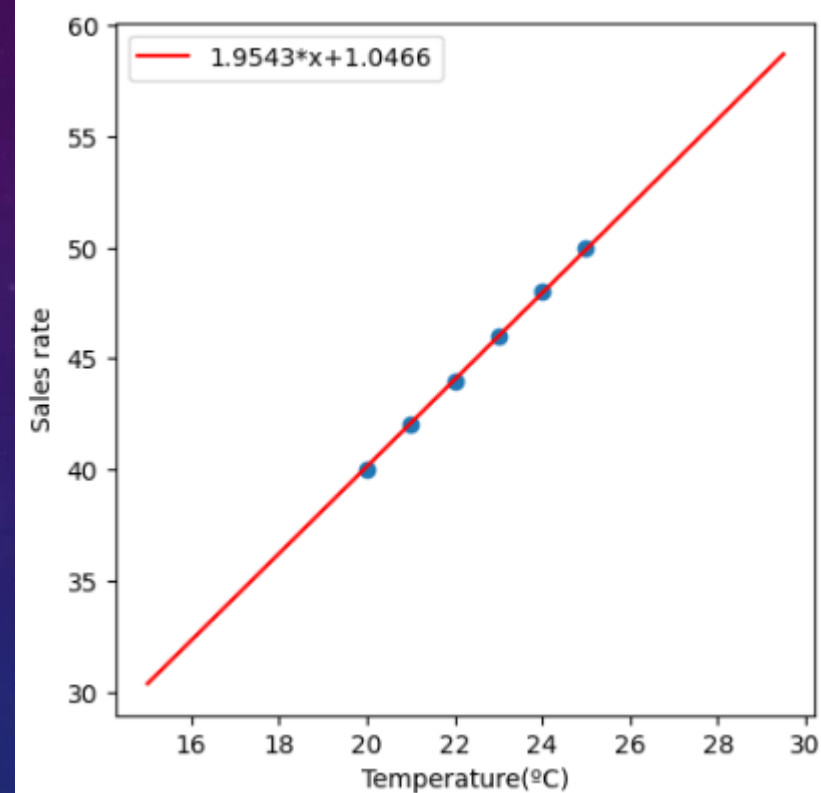
```
기울기 1.9542618 절편 1.0465583
```

데이터 탐색, 모델 구축

```
# 5. 시각화
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
# 기울기, 절편
ww = model.get_weights()

# 데이터
temp=np.arange(15.0, 30.0, 0.5, dtype=np.float64)
sales=ww[0][0][0]*temp+ww[1][0]

# 그래프
fig = plt.figure(figsize=(5,5))
ax = fig.add_subplot(111)
ax.set_xlabel('Temperature(°C)')
ax.set_ylabel('Sales rate')
ax.scatter(독립, 종속)
ax.plot(temp,sales, color='red', label=f'1.9543*x+1.0466')
ax.legend()
plt.show()
```

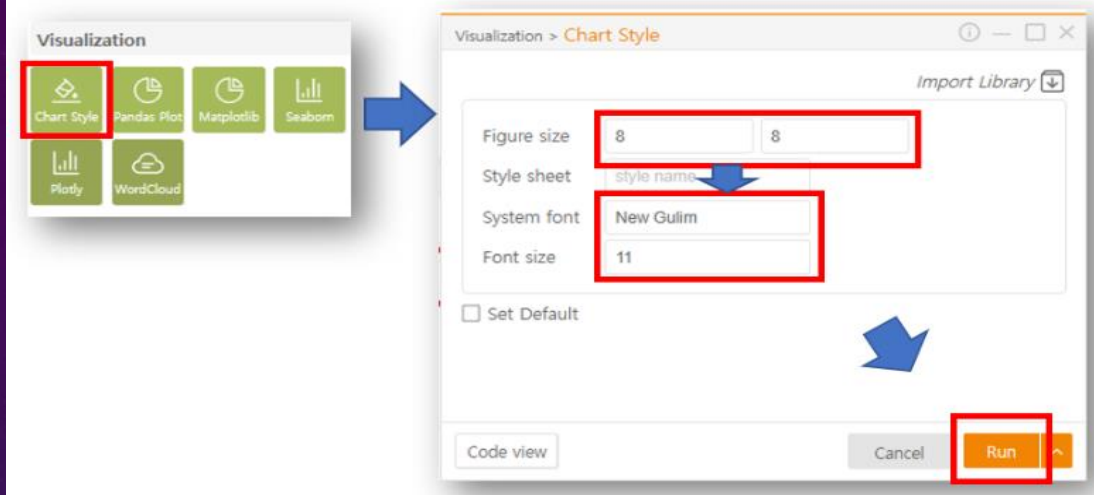


Visual Python을 활용한 머신러닝 실습

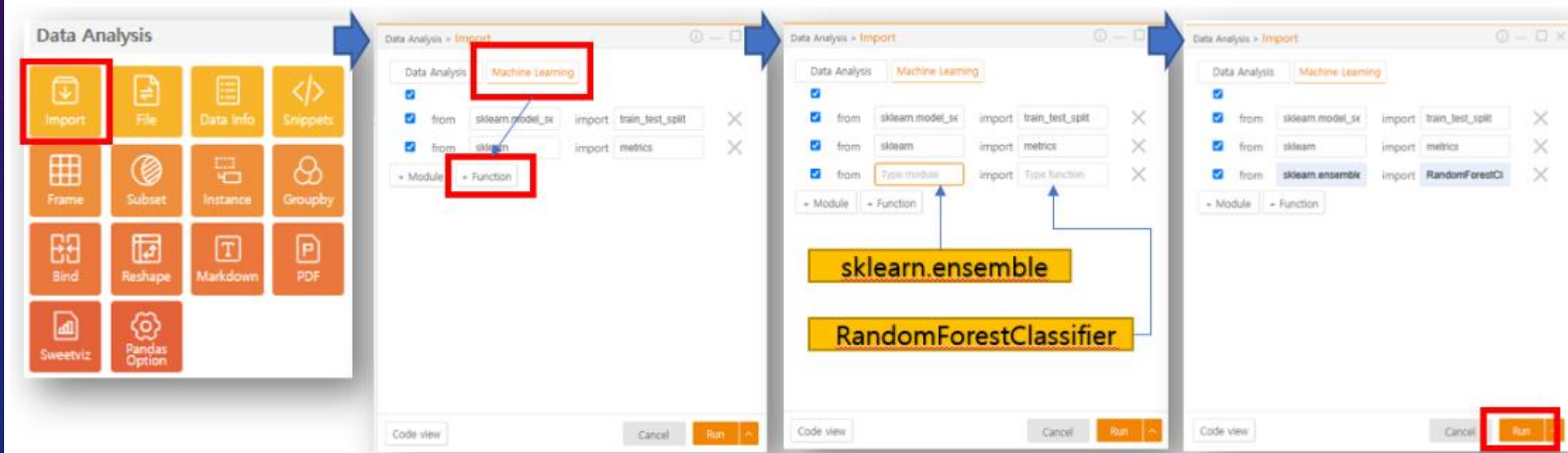
- 순서
 1. 데이터 수집
 2. 데이터 전처리 및 `train_test_split`
 3. 모델 생성
 4. 훈련 `fit`
 5. 예측 `predict`
 6. 평가 `evaluation`
 7. 시각화

데이터 탐색, 모델 구축(지도학습)

환경구성 : Chart Style



환경구성: Package 추가 : Import => Machine Learning



데이터 탐색, 모델 구축(지도학습)

1. 데이터 수집 : Data Sets

The screenshot displays the Machine Learning interface. On the left, a grid of icons represents various machine learning tasks. The 'Data Sets' icon, located in the top-left corner of the grid, is highlighted with a red box. A blue arrow points from this icon to the 'Data Sets' panel on the right. The 'Data Sets' panel is titled 'Machine Learning > Data Sets' and contains two dropdown menus: 'Load type' and 'Allocate to'. Both dropdown menus are set to 'load_breast_cancer' and are highlighted with red boxes. A blue arrow points from the 'load_breast_cancer' option in the 'Allocate to' dropdown to the 'load_breast_cancer' option in the 'Load type' dropdown. At the bottom right of the panel, there are three buttons: 'Code view', 'Cancel', and 'Run'. The 'Run' button is highlighted with a red box.

Machine Learning

Data Sets Data Split Data Prep AutoML

Regressor Classifier Clustering Dimension

GridSearch Fit/Predict Model Info Evaluation

Pipeline Save/Load

Machine Learning > Data Sets

Load type load_breast_cancer

Allocate to load_breast_cancer

load_breast_cancer

Code view Cancel Run

2. 데이터 전처리 및 train_test_split



1. Frame 선택

2. 독립변수 선택: df

df_data DataFrame에서 target 속성을 제거하고 df로 복사

The screenshot shows the 'Data Analysis' window with a 'Frame' icon selected in the left sidebar. The main window displays a DataFrame named 'df_data' with columns: mean radius, mean texture, mean perimeter, mean area, mean smoothness, mean compactness, dimension, and target. The 'target' column is highlighted in blue. A red box highlights the 'target' column header, and a red arrow points to the 'Delete' option in the context menu. Another red box highlights the 'Run' button at the bottom right. Red text annotations are present: '데이터' (Data) next to the DataFrame name, '독립변수 df로 변경 및 target 속성을 찾아 삭제' (Change to independent variable df and delete target attribute), and '데이터' (Data) next to the 'target' column header.

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	dimension	target
0	17.99	10.38	122.8	1001	0.1184	0.2776		
1	20.57	17.77	132.9	1326	0.08474	0.07864		
2	19.69	21.25	130	1203	0.1096	0.1592		
3	11.42	20.38	77.58	386.1	0.1425	0.2839		
4	20.29	14.34	135.1	1297	0.1003	0.1328		
5	12.45	15.7	82.57	477.1	0.1278	0.17		
6	18.25	19.98	119.6	1040	0.09463	0.109		
7	13.71	20.83	90.2	577.9	0.1189	0.1645		
8	13	21.82	87.5	519.8	0.1273	0.1932		

데이터 탐색, 모델 구축(지도학습)

2. 데이터 전처리 및 train_test_split



3. Subset 선택 :

4. 종속변수 선택: y

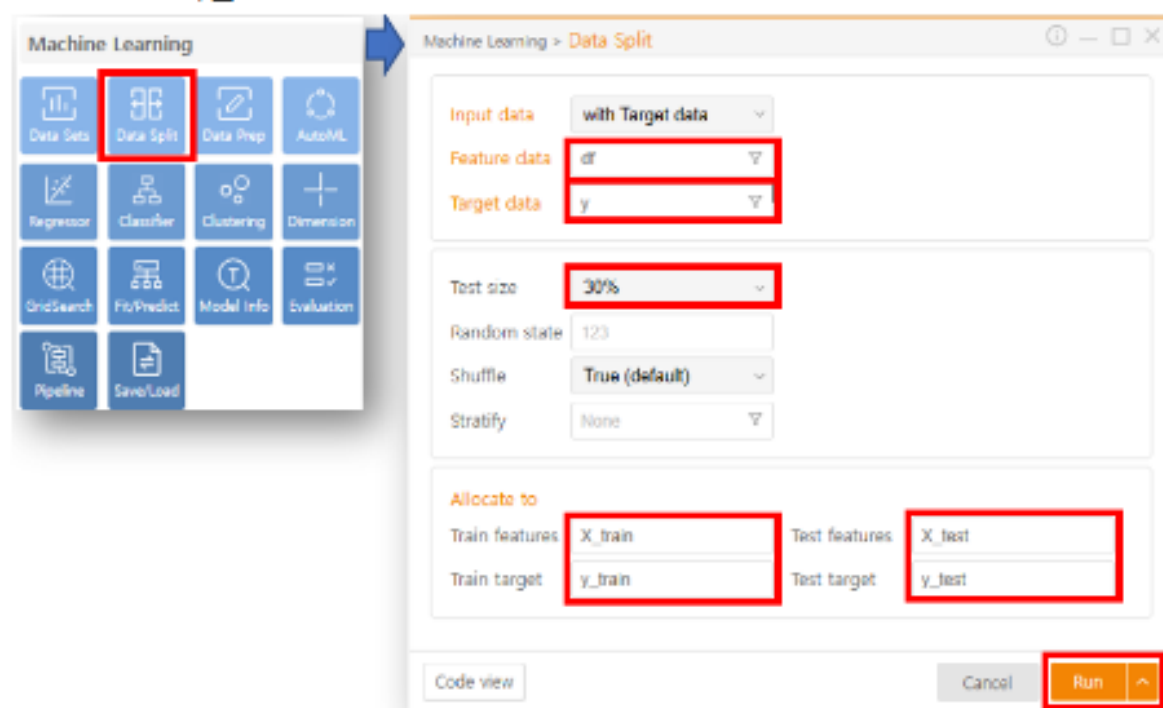
수집된 데이터(dfldata)에서 종속변수(y) 추출

The screenshot shows the 'Data Analysis > Subset' window. On the left, a 'Data Analysis' sidebar contains icons for Import, File, Data Info, Snippets, Frame, Subset (highlighted with a red box), Instance, Groupby, Bind, Reshape, Markdown, PDF, Sweetviz, and Pandas Option. The main window displays the code `y = dfldata.loc[:, 'target']`. Below the code, the 'DataFrame' field is set to 'dfldata' (labeled '데이터' with a red box), the 'Method' is 'loc (location)', and 'Allocate to' is set to 'y' (labeled '종속변수 y 입력' with a red box). The 'Column Subset' section shows a list of columns with 'target' selected and moved to the 'Indexing' field (both highlighted with red boxes). The 'Run' button at the bottom right is also highlighted with a red box.

2. 데이터 전처리 및 train_test_split

5. train_test_split :  선택

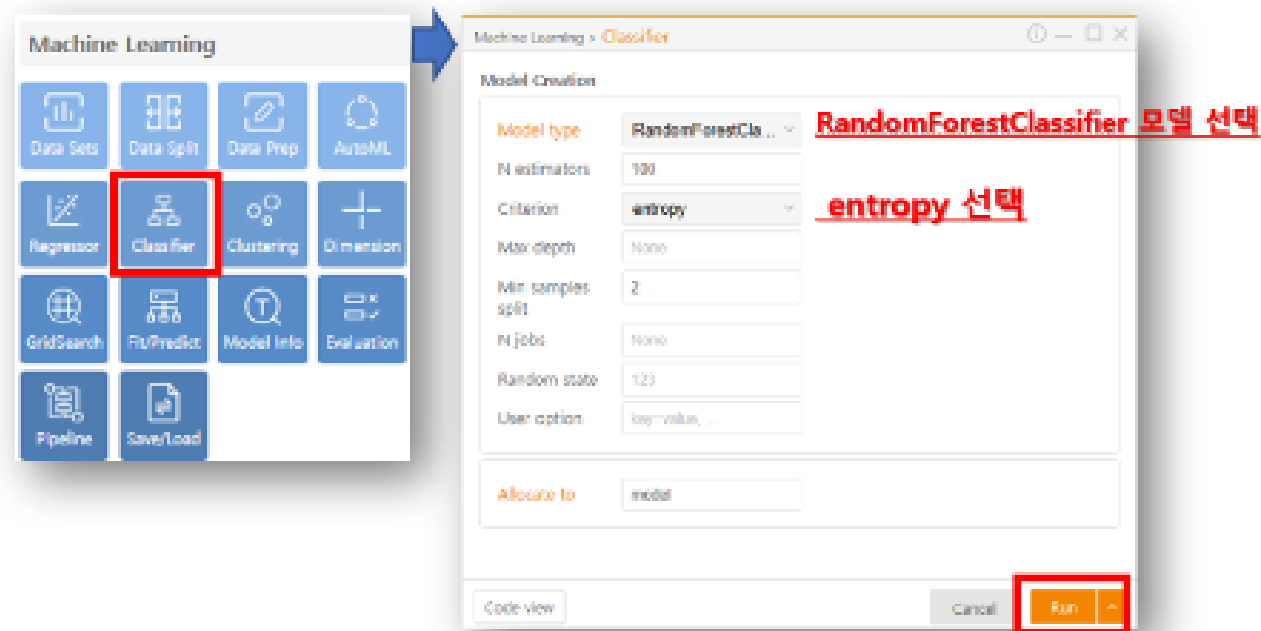
train : test 비율 => 7 : 3



The screenshot shows the 'Data Split' configuration window. On the left, a 'Machine Learning' sidebar contains icons for Data Sets, Data Split (highlighted with a red box), Data Prep, AutoML, Regressor, Classifier, Clustering, Dimension, GridSearch, Fit/Predict, Model Info, Evaluation, Pipeline, and Save/Load. An arrow points from the 'Data Split' icon to the main window. The main window has a title bar 'Machine Learning > Data Split'. It contains several sections: 'Input data' with a dropdown set to 'with Target data'; 'Feature data' with a dropdown set to 'x' (highlighted with a red box); 'Target data' with a dropdown set to 'y' (highlighted with a red box); 'Test size' with a dropdown set to '30%' (highlighted with a red box); 'Random state' with a text input '123'; 'Shuffle' with a dropdown set to 'True (default)'; 'Stratify' with a dropdown set to 'None'; and an 'Allocate to' section with four text inputs: 'Train features' set to 'X_train', 'Test features' set to 'X_test', 'Train target' set to 'y_train', and 'Test target' set to 'y_test' (all four inputs are highlighted with red boxes). At the bottom, there are buttons for 'Code view', 'Cancel', and 'Run' (highlighted with a red box).

3. 머신러닝 모델 생성 ¶

1. 모델 선택 :



데이터 탐색, 모델 구축(지도학습)

4. 학습 fit

- 모델 훈련

The screenshot illustrates the 'Fit/Predict' workflow in a machine learning interface. On the left, a 'Machine Learning' sidebar contains various tool icons. The 'Fit/Predict' icon is highlighted with a red box. A blue arrow points from this icon to the main workspace. The workspace is divided into two panels, both titled 'Machine Learning > Fit/Predict'. The top panel, labeled 'Model', shows a list of models. The 'Classification' category is highlighted with a red box, and a red arrow points to the 'model (RandomForestClassifier)' entry, which is also highlighted with a red box. The bottom panel, labeled 'Action', shows a list of actions. The 'Fit' action is highlighted with a red box. Below this, the 'Options' section shows 'Feature Data' set to 'X_train' and 'Target Data' set to 'y_train', both highlighted with red boxes. At the bottom right, the 'Run' button is highlighted with a red box.

Machine Learning

Data Sets Data Split Data Prep AutoML

Regressor Classifier Clustering Dimension

GridSearch **Fit/Predict** Model Info Evaluation

Pipeline Save/Load

Machine Learning > Fit/Predict

Model

All Data Preparation Regression **Classification**

model (RandomForestClassifier)

Machine Learning > Fit/Predict

Action

Search Action

Fit Predict Predict probability

Options

Feature Data X_train

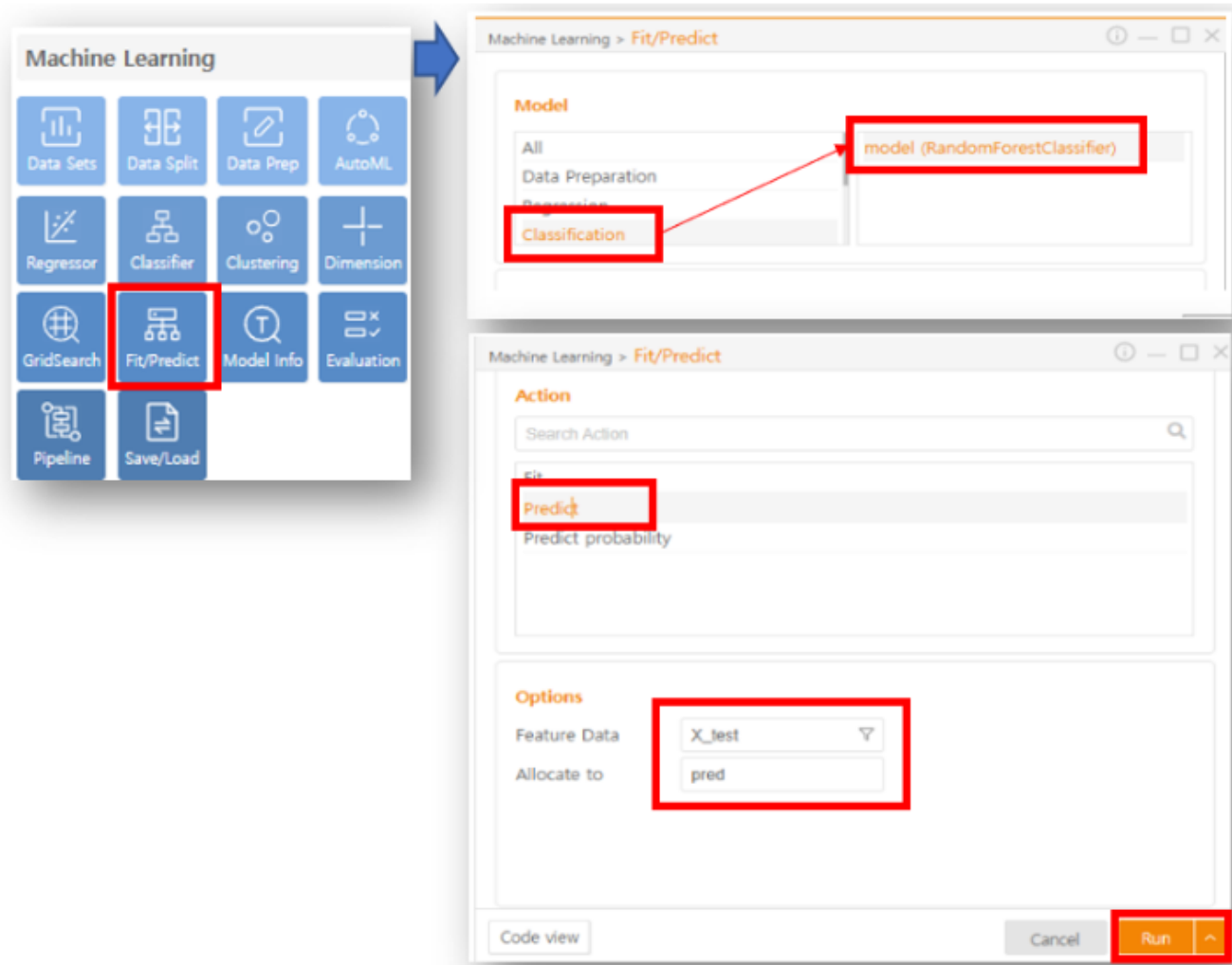
Target Data y_train

Code view Cancel **Run**

데이터 탐색, 모델 구축(지도학습)

5. 예측 predict

- 예측



테스트데이터: X_test

예측결과: pred

6. 평가 : evaluation

- 평가

Machine Learning

Data Sets, Data Split, Data Prep, AutoML, Regressor, Classifier, Clustering, Dimension, GridSearch, Fit/Predict, Model Info, **Evaluation**, Pipeline, Save/Load

Machine Learning > Evaluation

Import Library

Model Type: Classification

Target Data: y_test

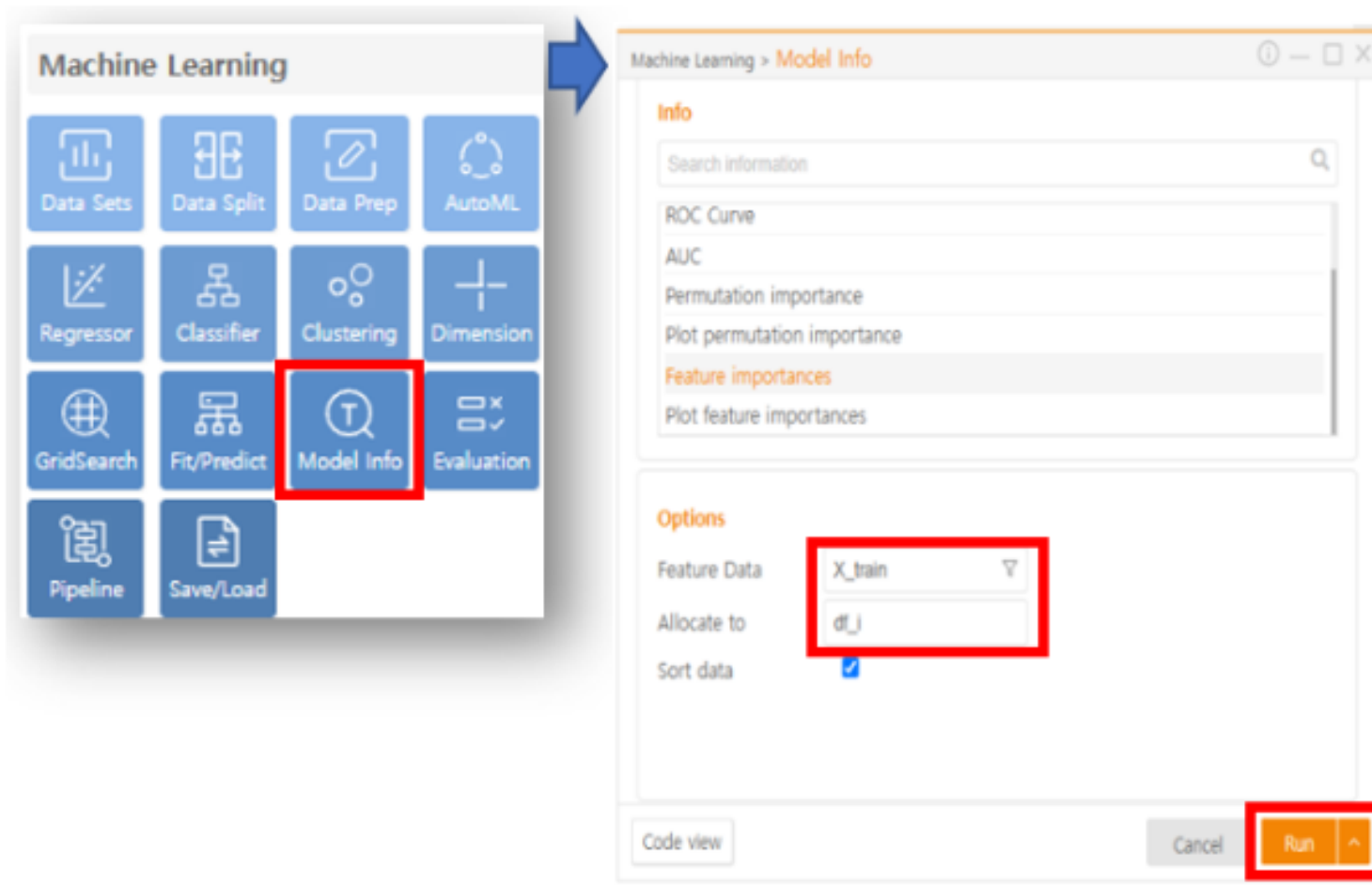
Predict Data: pred

Evaluation Metrics

- ☒ Confusion Matrix
- ☒ Report (Accuracy/Precision/Recall/F1-score)
** You can select individually*
- ☒ Accuracy
- ☐ Precision
- ☐ Recall
- ☐ F1-score

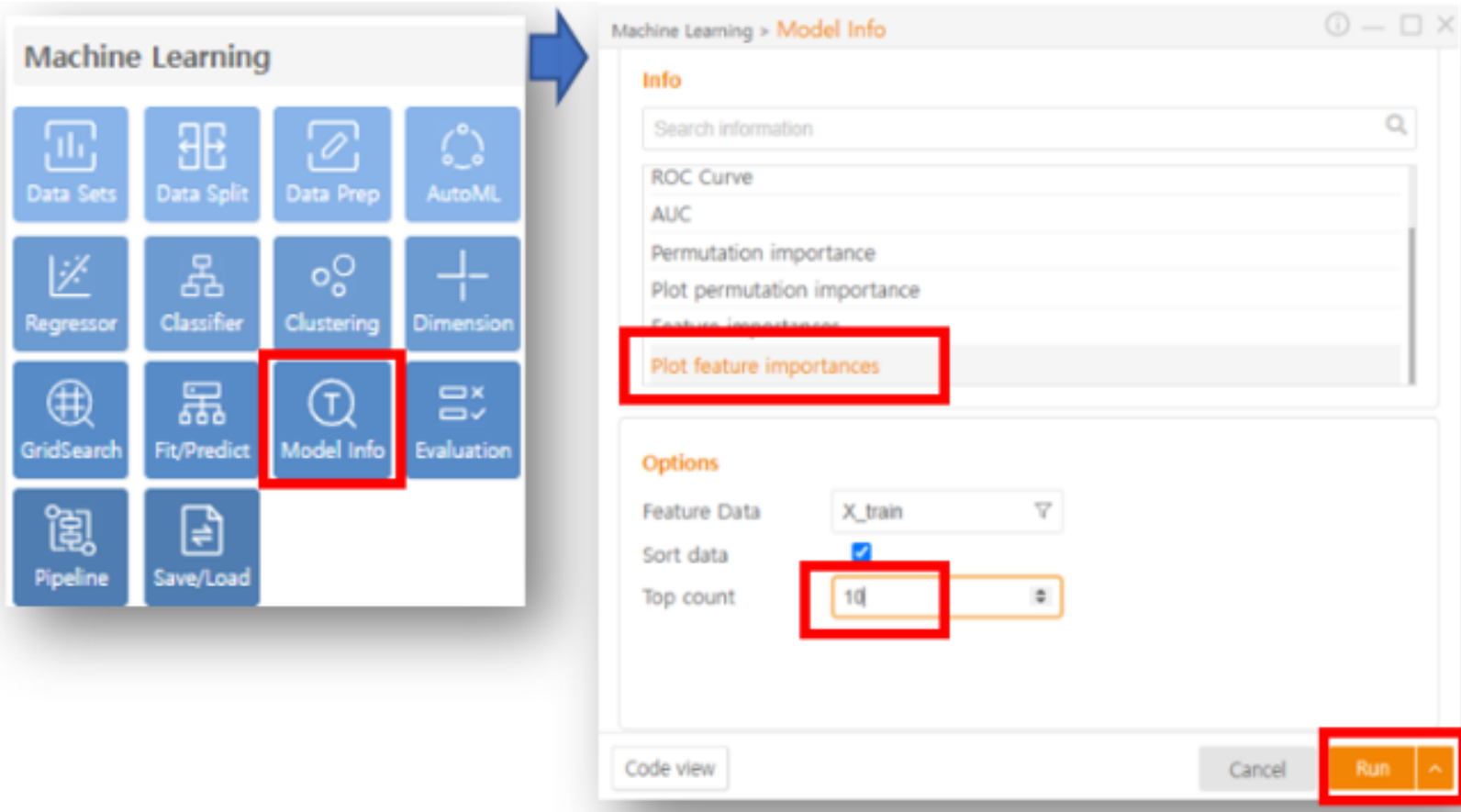
Code view, Cancel, **Run**

7. 시각화(표, features importance)



8. 시각화(차트) - plot

-  선택



감사합니다.