산업 인공지능 - 실습 6

Python 프로그래밍

2021 Spring

1. 내장 함수

❖ 내장 함수

- 파이썬에서 기본적으로 제공하는 함수
- import문으로 포함시킬 필요 없음
- 대부분의 객체에 대해서 적용 가능

abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

내장 함수

❖ 절대값 함수 abs(x)

```
>>> abs(-3)
3
>>> abs(3+4j)
5.0
```

❖ 모든 요소가 참인지 확인 all(iterable)

```
def all(iterable):
    for element in iterable:
        if not element:
            return False
    return True
```

❖ 어느 하나라도 참인지 확인 any(iterable)

```
def any(iterable):
    for element in iterable:
        if element:
            return True
    return False
```

내장 함수

❖ 참/거짓으로 변환하여 반환 **bool(x)**
- 정수 0이면 False, 정수 1이면 True 반환

❖ ASCII 코드의 문자 반환 **chr(i)**

❖ 문자를 받아 ASCII 코드 반환 **ord(c)**

```
>>> chr(65)
'A'
>>> ord('A')
65
```

❖ 문자열 읽어 컴파일후 저장 **compile(문자열, 파일이름, 모드)**

```
>>> with open('mymodule.py') as f:
     lines = f.read()

>>> code_obj = compile(lines, 'mymodule.py', 'exec')
>>> exec(code_obj)
...
```

내장 함수

❖ 복소수 객체 생성 complex(실수, 허수)

```
>>> x = complex(4, 2)
>>> x
(4+2j)
```

❖ 객체의 변수와 함수 출력 dir(객체)

```
>>> dir([1, 2, 3])
['__add__', '__class__', '__contains__', '__delattr__',
'__delitem__', '__dir__', '__doc__', '__eq__', '__format__',
'__ge__', '__getattribute__', '__getitem__', '__gt__',
'__hash__', '__iadd__', '__imul__', '__init__', '__iter__',
'__le__', '__len__', '__lt__', '__mul__', '__ne__', '__new__',
'__reduce__', '__reduce_ex__', '__repr__', '__reversed__',
'__rmul__', '__setattr__', '__setitem__', '__sizeof__',
'__str__', '__subclasshook__', 'append', 'clear', 'copy',
'count', 'extend', 'index', 'insert', 'pop', 'remove',
'reverse', 'sort']
```

내장 함수

❖ 몫과 나머지 계산 **divmod(a,b)** ⇒ **(a//b, a %b)**

```
>>> divmod(10,3)
(3,1)
```

❖ **enumerate 객체 반환** **enumerate(iterable, start=0)**

- **enumerate 객체**
  - 첫 번째 인자는 **번호**, 두 번째 인자에 해당하는 **값**을 갖는 객체

```
seasons = ['Spring', 'Summer', 'Fall', 'Winter']
print(list(enumerate(seasons)))
for i, str in enumerate(["aaaa", "bbbb", "cccc"]):
    print(i, str)
```

```
[(0, 'Spring'), (1, 'Summer'), (2, 'Fall'), (3, 'Winter')]
0 aaaa
1 bbbb
2 cccc
```

내장 함수

❖ **문자열을 실행 eval()**

- 사용자가 입력하는 파이썬 문장 실행 가능

```
>>> eval("1+2");
3

>>> x = 1
>>> y = 1
>>> eval('x+y')
2

>>> eval("print('Hi!')")
Hi!

>>> expression = input("Enter an expression")
>>> eval(expression)
```

내장 함수

❖ **수식과 타 문장 실행 exec( )**

- **인터프리터**에서 사용

```
>>> exec("y=2+3")
>>> y
5

>>> statements = '''
import math
def area_of_circle(radius):
    return math.pi * radius * radius

'''
>>> exec(statements)
>>> area_of_circle(5)
78.53981633974483
```

내장 함수

❖ 문자열을 실수로 변환 float( )

```
>>> str = input("실수를 입력하시오: ")
실수를 입력하시오: 12.345
>>> str
'12.345'
>>> value = float(str)
>>> value
12.345
```

❖ 문자열을 정수로 변환 int( )

❖ 객체이 길이 len( )

- 객체의 길이 측정

내장 함수

❖ 시퀀스 객체를 받아 리스트로 변환하는 list()
- 튜플, 문자열

```
>>> list("python")
['p', 'y', 't', 'h', 'o', 'n']
```

❖ 가장 큰 항목 반환 max( )
❖ 가장 작은 항목 반환 min( )

```
>>> values = [ 1, 2, 3, 4, 5]
>>> max(values)
5
>>> min(values)
1
```

2. 정렬

❖ **정렬(sort)**

- 항목을 크기 순으로 재배열
- 내장 함수 **sorted**( )
  - iterable 객체로 부터 정렬된 리스트 생성

```
>>> sorted([4, 2, 3, 5, 1])
[1, 2, 3, 4, 5]

>>> myList = [4, 2, 3, 5, 1]
>>> myList.sort()
>>> myList
[1, 2, 3, 4, 5]
```

정렬

❖ 매개변수 key

- 정렬에 사용되는 **키 변경**

```
>>> sorted("The health know not of their health, but only the
sick".split(), key=str.lower)
['but', 'health', 'health,', 'know', 'not', 'of', 'only', 'sick',
'The', 'the', 'their']
```

```
>>> students = [
        ('홍길동', 3.9, 20160303),
        ('김철수', 3.0, 20160302),
        ('최무선', 4.3, 20160301),
]
>>> sorted(students, key = lambda student: student[2]) # 학번기준 정렬
[('최무선', 4.3, 20160301), ('김철수', 3.0, 20160302), ('홍길동', 3.9,
20160303)]

>>> sorted(students, key = lambda student: student[0]) # 학번기준 정렬
[('최무선', 4.3, 20160301), ('김철수', 3.0, 20160302), ('홍길동', 3.9,
20160303)]
```

정렬

```
>>> class Student:
    def __init__(self, name, grade, number):
        self.name = name
        self.grade = grade
        self.number = number
    def __repr__(self):
        return repr((self.name, self.grade, self.number))

>>> students = [
        Student('홍길동', 3.9, 20160303),
        Student('김철수', 3.0, 20160302),
        Student('최무선', 4.3, 20160301),
]
>>> sorted(students, key=lambda student: student.number)
[('최무선', 4.3, 20160301), ('김철수', 3.0, 20160302), ('홍길동
', 3.9, 20160303)]
```

정렬

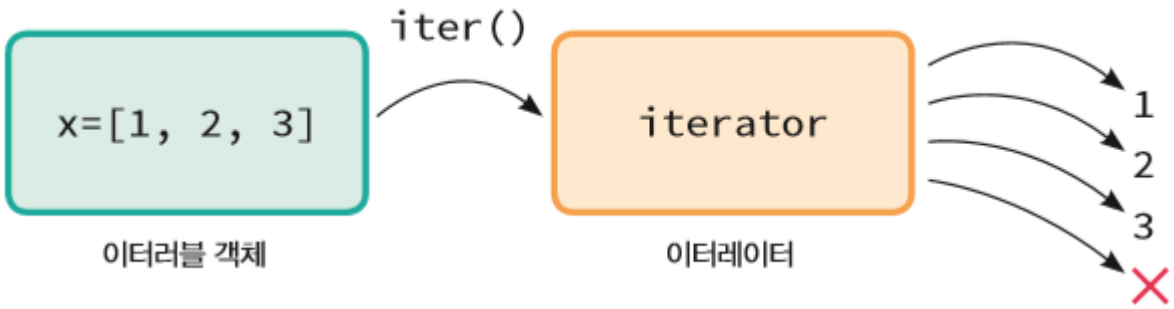
❖ 오름차순 정렬과 내림차순 정렬

```
>>> sorted(students, key=lambda student: student.number, reverse=True)
[('홍길동', 3.9, 20160303), ('김철수', 3.0, 20160302), ('최무선', 4.3,
20160301)]
```

3. 이터레이터와 제너레이터

❖ **이터러블 객체 (iterable object)**

- for 문과 함께 사용할 수 있는 객체



❖ **이터러블 객체 만들기 위한 메소드 정의**

- **__iter__ ( )**
  - 이터러블 **객체 자신** 반환
- **__next__ ( )**
  - **다음** 반복을 위한 **값** 반환
  - 만약 더 이상의 값이 없으면 **StopIteration 예외** 발생

```
class MyCounter(object):
    def __init__(self, low, high):
        self.current = low
        self.high = high

    # 이터레이터 객체로서 자신 반환
    def __iter__(self):
        return self

    def __next__(self):
        # current가 high 보다 크면 StopIteration 예외 발생
        # current가 high 보다 작으면 다음 값을 반환
        if self.current > self.high:
            raise StopIteration
        else:
            self.current += 1
            return self.current - 1

c = MyCounter(1, 10)
for i in c:
    print(i, end=' ')
```

```
1 2 3 4 5 6 7 8 9 10
```

이터레이터와 제너레이터

❖ **제너레이터(generators)**

- 키워드 yield를 사용하여서 함수로부터 이터레이터 생성

```
def myGenerator():
     yield 'first'
     yield 'second'
     yield 'third'

for word in myGenerator():
     print(word)
```

```
first
second
third
```

이터레이터와 제너레이터

❖ **클로저(closure)**

- **함수**에 의하여 **반환**되는 **함수**

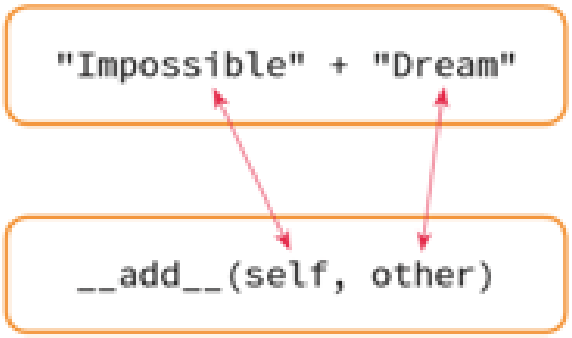
```
def addNumber(fixedNum):
    def add(number):
        return fixedNum + number
    return add

func = addNumber(10)
print(func(20))
```

```
30
```

4. 연산자 오버로딩

❖ **연산자 오버로딩(operator overloading)**

- 연산자의 **의미**를 메소드로 **추가**로 **정의**하는 것



연산자 오버로딩

❖ 오버로딩 가능 연산자

연산자	수식예	내부적인 함수 호출
덧셈	x + y	x.__add__(y)
뺄셈	x - y	x.__sub__(y)
곱셈	x * y	x.__mul__(y)
지수	x ** y	x.__pow__(y)
나눗셈(실수)	x / y	x.__truediv__(y)
나눗셈(정수)	x // y	x.__floordiv__(y)
나머지	x % y	x.__mod__(y)
비트 왼쪽 이동	x << y	x.__lshift__(y)
비트 오른쪽 이동	x >> y	x.__rshift__(y)
비트 AND	x & y	x.__and__(y)
비트 OR	x | y	x.__or__(y)
비트 XOR	x ^ y	x.__xor__(y)
비트 NOT	~x	x.__invert__()

연산자 오버로딩

```
>>> s1="Impossible "
>>> s2="Dream"
>>> s3 = s1.__add__(s2)
>>> s3
'Impossible Dream'
>>>
```

```
>>> class Point:
    def __init__(self, x = 0,y = 0):
        self.x = x
        self.y = y

>>> p1 = Point(1, 2)
>>> p2 = Point(3, 4)
>>> p1 + p2
...
TypeError: unsupported operand type(s) for +: 'Point' and
'Point'
```

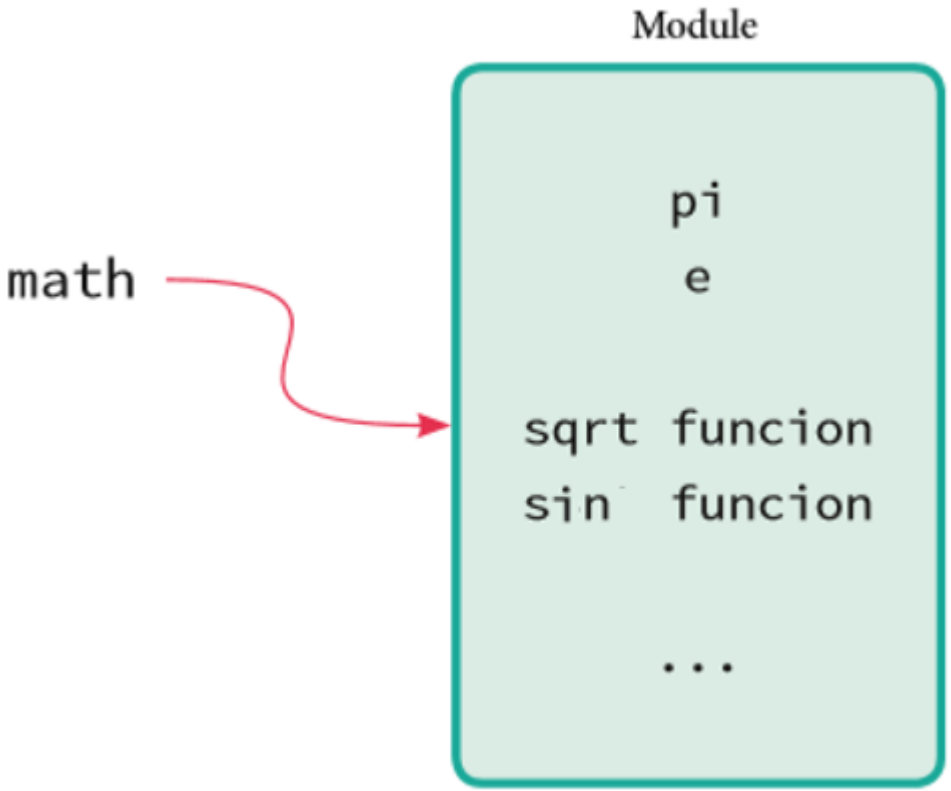
연산자 오버로딩

```
>>> class Point:
    def __init__(self,x = 0,y = 0):
        self.x = x
        self.y = y

    def __add__(self, other):
        x = self.x + other.x
        y = self.y + other.y
        return Point(x, y)

    def __str__(self):
        return 'Point(' + str(self.x) + ', ' +str(self.y) + ')'

>>> p1 = Point(1, 2)
>>> p2 = Point(3, 4)
>>> print(p1+p2)
(4,6)
```

5. 모듈

❖ **모듈(module)**

- **함수**나 **변수** 또는 **클래스** 등을 **모아 놓은 파일**



모듈

❖ 모듈 작성하기

fibo.py

```
# 피보나치 수열 모듈

def fib(n):           # 피보나치 수열 출력
    a, b = 0, 1
    while b < n:
        print(b, end=' ')
        a, b = b, a+b
    print()

def fib2(n):   # 피보나치 수열을 리스트로 반환
    result = []
    a, b = 0, 1
    while b < n:
        result.append(b)
        a, b = b, a+b
    return result
```

모듈

❖ 모듈 사용하기 – cont.

```
>>> import fibo

>>> fibo.fib(1000)
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987

>>> fibo.fib2(100)
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]

>>> fibo.__name__
'fibo'
```

모듈

❖ 모듈 이름을 매번 사용하지 않도록 모듈 import 하는 방법

- from 모듈 import 함수
  - 함수 1개 import

```
>>> from fibo  import fib
>>> fib(1000)
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

  - 함수 여러 개 import

```
>>> from fibo  import fib, fib2
>>> fib2(100)
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

  - 함수 전체 import

```
>>> from fibo  import *
```

모듈

❖ 모듈 실행하기

```
C> python fibo.py <arguments>
```

- DOS 창에서 실행시 __name__ 변수가 "__main__"으로 자동으로 설정

```
...
if __name__ == "__main__":
    import sys
    fib(int(sys.argv[1]))
```

- sys.argv[1] : DOS창에서 실행할 때 인자로 전달되는 첫번째 값

```
C> python fibo.py 50
1 1 2 3 5 8 13 21 34
```

모듈

❖ **모듈 탐색 경로**

① **입력 스크립트**가 있는 **디렉토리**
(파일이 지정되지 않으면 현재 디렉토리)

① **PYTHONPATH 환경 변수**

② 설치에 의존하는 **디폴트값**

6. 유용한 모듈

❖ **이미 개발된 유용한 모듈**

- copy 모듈
- keyword 모듈
- random 모듈
- os 모듈
- sys 모듈
- time 모듈
- calendar 모듈

유용한 모듈

❖ copy 모듈

```
import copy
colors = ["red", "blue", "green"]
clone = copy.deepcopy(colors)

clone[0] = "white"
print(colors)
print(clone)
```

```
['red', 'blue', 'green']
['white', 'blue', 'green']
```

유용한 모듈

❖ keyword 모듈

```
import keyword

name = input("변수 이름을 입력하시오: ")

if keyword.iskeyword(name):
    print (name, "은 예약어")
    print ("다음은 키워드의 전체 리스트: ")
    print (keyword.kwlist)
else:
    print (name, "은 사용할 수 있는 변수이름")
```

```
변수 이름을 입력하시오: for
for 은 예약어
다음은 키워드의 전체 리스트:
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class',
'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for',
'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal',
'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with',
'yield']
```

유용한 모듈

❖ random 모듈

```
>>> import random
>>> print(random.randint(1, 6))
6
>>> print(random.randint(1, 6))
3

>>> print(random.random()*100)
81.1618515880431

>>> myList = [ "red", "green", "blue" ]
>>> random.choice(myList)
'blue'
```

유용한 모듈

❖ **random 모듈 - cont.**

```
>>> import random
>>> myList = [ [x] for x in range(10) ]
>>> random.shuffle(myList)
>>> myList
[[3], [2], [7], [9], [8], [1], [4], [6], [0], [5]]

>>> for i in range(3):
        print(random.randrange(0, 101, 3))

81
21
57
```

유용한 모듈

❖ os 모듈

- 운영체제(operating system)의 기본적인 기능 사용

```
>>> import os
>>> os.system("calc")  # 계산기 실행
```

```
>>> os.getcwd()      # 현재 작업 디렉토리
'D:\\'
>>> os.chdir("D:\\tmp")  # 디렉토리 변경
>>> os.getcwd()
'D:\\tmp’

>>> os.listdir(".")   # 디렉토리 내용 출력
[ 'chap01', 'chap02', 'chap03', 'chap04', 'chap05', 'chap06',
'chap07', 'chap08', 'chap09', 'chap10', 'chap11', 'chap12',
'chap13', 'chap14', 'chap15', 'chap16', 'chap17', 'chap18',
'chap19', 'chap20' ]
```

유용한 모듈

❖ **sys 모듈**

- 파이썬 인터프리터에 대한 다양한 정보 제공

```
>>> import sys
>>> sys.prefix # 파이썬이 설치된 경로
'C:\\Users\\chun\\AppData\\Local\\Programs\\Python\\Python35-32’

>>> sys.executable  # 인터프리터 파일
'C:\\Users\\chun\\AppData\\Local\\Programs\\Python\\Python35-
32\\python.exe’

>>> sys.modules    # 현재 설치되어 있는 모듈

>>> sys.path       # 모듈 참조할 때 사용하는 경로

>>> sys.version    # 설치된 파이썬 버전
```

유용한 모듈

❖ **time 모듈**

- 시간 정보를 가져와서 다양한 형식으로 출력

```
>>> import time
>>> time.time()
1461203464.6591916
```

유용한 모듈

❖ **time 모듈 – cont.**

```
>>> import time
>>> time.asctime()   # 문자 형태로 시각 출력
Thu Apr 15 09:48:52 2021

>>> time.localtime()  # 현지 날짜 시각
time.struct_time(tm_year=2021, tm_mon=4, tm_mday=15,
tm_hour=9, tm_min=49, tm_sec=25, tm_wday=3,
tm_yday=105, tm_isdst=0)
```

유용한 모듈

❖ **calendar 모듈**

- 달력 출력

```
import calendar

cal = calendar.month(2021, 4)
print(cal)
```

```
 April 2021
Mo Tu We Th Fr Sa Su
           1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30
```

기계학습 – 분류, 회귀

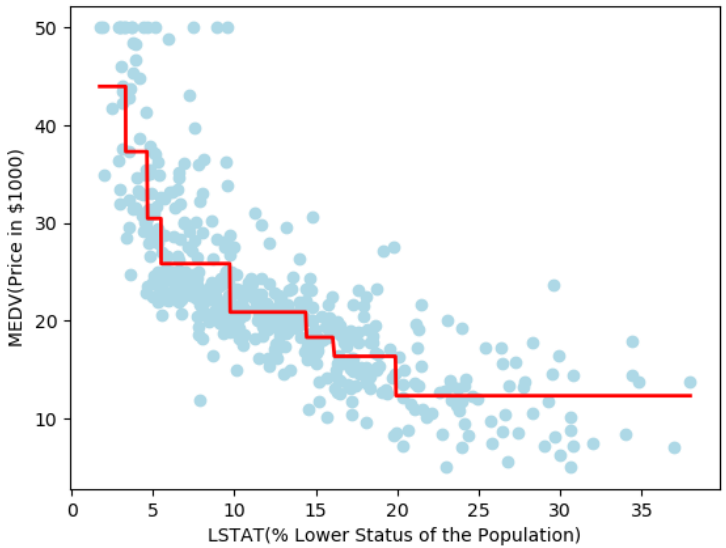
❖ [실습] 결정트리 회귀

```
from sklearn.tree import DecisionTreeRegressor
import pandas as pd
import matplotlib.pyplot as plt
```

https://archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.data

```
df = pd.read_csv('housing.data', header=None, sep='₩s+')
df.columns = ['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD',
              'TAX', 'PTRATIO', 'B', 'LSTAT', 'MEDV']
print(df.head())
X = df[['LSTAT']].values
y = df['MEDV'].values

tree = DecisionTreeRegressor(max_depth=3)
tree.fit(X,y)

sort_idx = X.flatten().argsort()
plt.scatter(X[sort_idx], y[sort_idx], c='lightblue')
plt.plot(X[sort_idx], tree.predict(X[sort_idx]), color='red', linewidth=2)
plt.xlabel('LSTAT(% Lower Status of the Population)')
plt.ylabel('MEDV(Price in $1000)')
plt.show( )
```

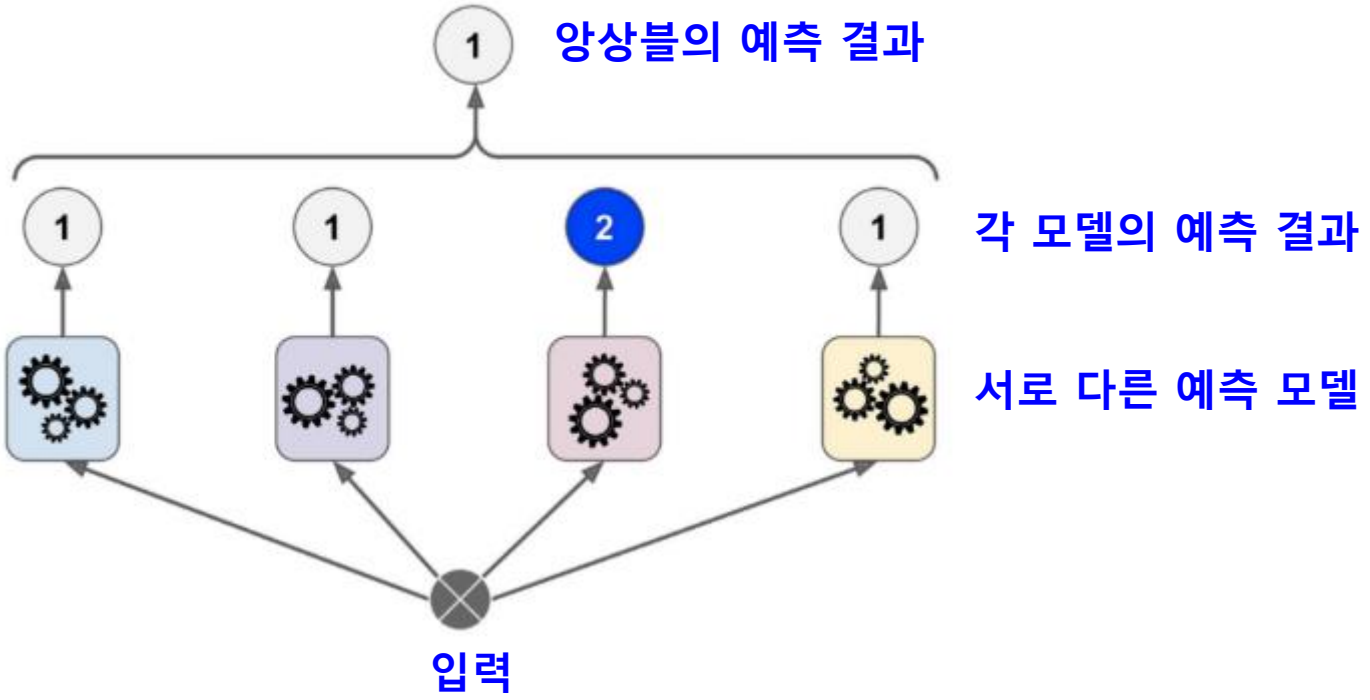
```
      CRIM    ZN  INDUS  CHAS    NOX  ...    TAX  PTRATIO       B  LSTAT  MEDV
0  0.00632  18.0   2.31     0  0.538  ...  296.0     15.3  396.90   4.98  24.0
1  0.02731   0.0   7.07     0  0.469  ...  242.0     17.8  396.90   9.14  21.6
2  0.02729   0.0   7.07     0  0.469  ...  242.0     17.8  392.83   4.03  34.7
3  0.03237   0.0   2.18     0  0.458  ...  222.0     18.7  394.63   2.94  33.4
4  0.06905   0.0   2.18     0  0.458  ...  222.0     18.7  396.90   5.33  36.2
```



1. 앙상블 학습

❖ **대중의 지혜**(wisdom of crowd)
- 무작위로 선택된 많은 사람의 답변을 모은 것이 전문가의 답보다 낫다

❖ **앙상블 학습 (ensemble learning)**
- 일련의 예측 모델(분류 또는 회귀 모델)을 사용한 모델의 학습



Source: Hands-on Machine Learning with Scikit-Learn & TensorFlow, ch7

앙상블 학습

❖ **붓스트랩**(bootstrap)

- 주어진 학습 데이터 집합에서 **복원추출**(resampling with replacement) 하여 **다수**의 **학습 데이터 집합**을 만들어내는 기법

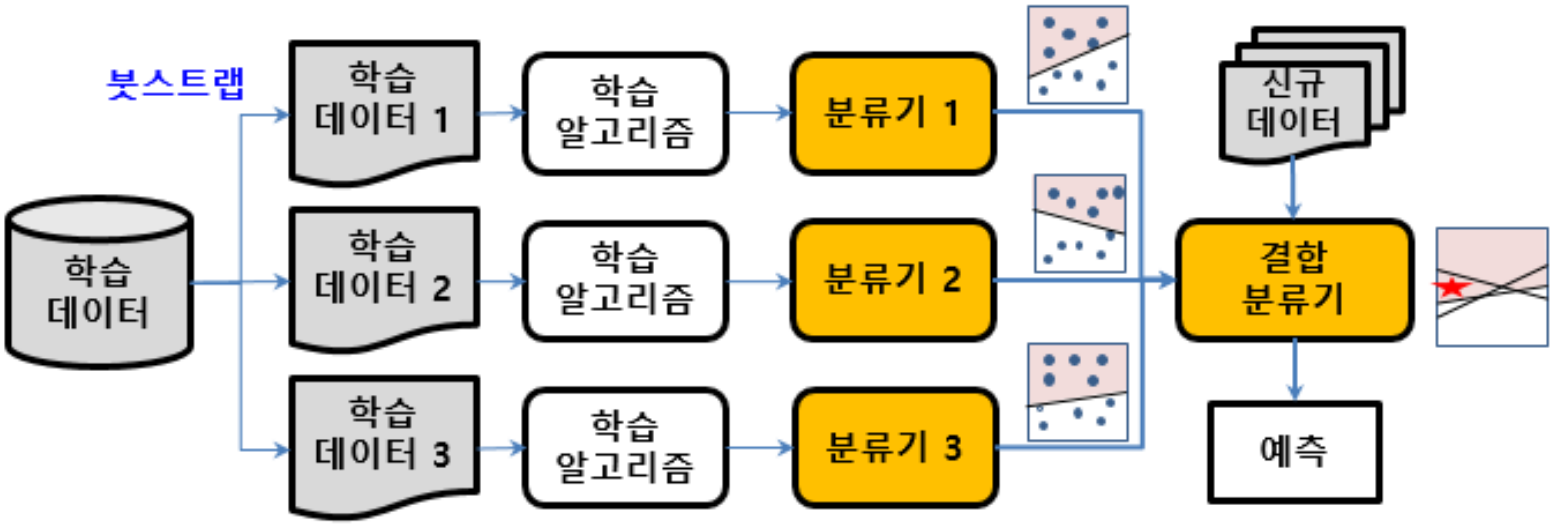
❖ **배깅**(bagging, **b**ootstrap **agg**regat**ing**)

❖ **부스팅**(boosting)

2. 배깅 알고리즘

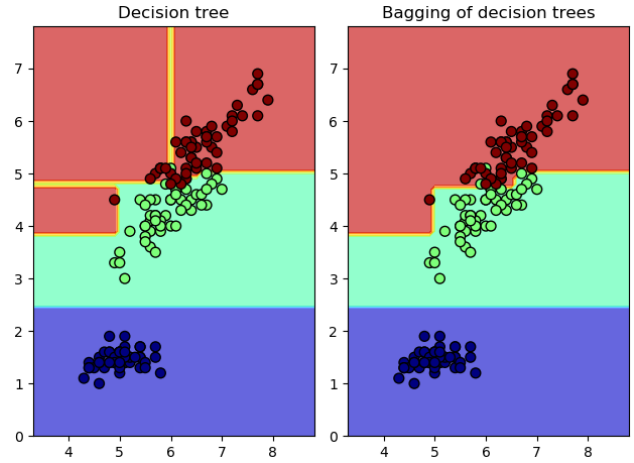
❖ **배깅**(bagging, **b**ootstrap **agg**regat**ing**)

- **붓스트랩**을 통해 **여러 개**의 **학습 데이터 집합** 생성
- 각 학습 데이터 집합별로 **분류기** 또는 **회귀모델 생성**
- 최종판정
  - 분류기들의 **투표**나 **가중치 투표**
  - 회귀 모델들의 평균



❖ [실습] 배깅

```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier
iris = load_iris( )
X, y = iris.data[:, [0, 2]], iris.target

model1 = DecisionTreeClassifier(max_depth=10, random_state=0).fit(X, y)
model2 = BaggingClassifier(DecisionTreeClassifier(max_depth=4),
                           n_estimators=50, random_state=0).fit(X, y)

x_min, x_max = X[:, 0].min( ) - 1, X[:, 0].max( ) + 1
y_min, y_max = X[:, 1].min( ) - 1, X[:, 1].max( ) + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.1), np.arange(y_min, y_max, 0.1))

plt.subplot(121)
Z1 = model1.predict(np.c_[xx.ravel( ), yy.ravel( )]).reshape(xx.shape)
plt.contourf(xx, yy, Z1, alpha=0.6, cmap=mpl.cm.jet)
plt.scatter(X[:, 0], X[:, 1], c=y, alpha=1, s=50, cmap=mpl.cm.jet, edgecolors="k")
plt.title("Decision tree")
plt.subplot(122)

Z2 = model2.predict(np.c_[xx.ravel( ), yy.ravel( )]).reshape(xx.shape)
plt.contourf(xx, yy, Z2, alpha=0.6, cmap=mpl.cm.jet)
plt.scatter(X[:, 0], X[:, 1], c=y, alpha=1, s=50, cmap=mpl.cm.jet,
            edgecolors="k")
plt.title("Bagging of decision trees")
plt.tight_layout()
plt.show( )
```



https://scikit-learn.org/

배깅 알고리즘 : 랜덤 포리스트

❖ **랜덤 포리스트(random forest)** 알고리즘
- 분류기로 **결정트리**를 사용하는 배깅 기법
- **Random (무작위) + Forest (숲)**
  - Random : 무작위로 선택한 속성중에서 분할 속성을 선택
  - Forest : 여러 결정트리로 구성

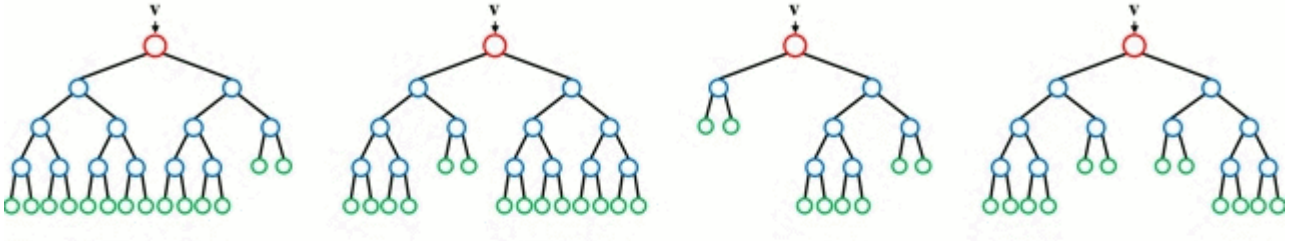


image source: Wikipedia.org

❖ [실습] Random Forest

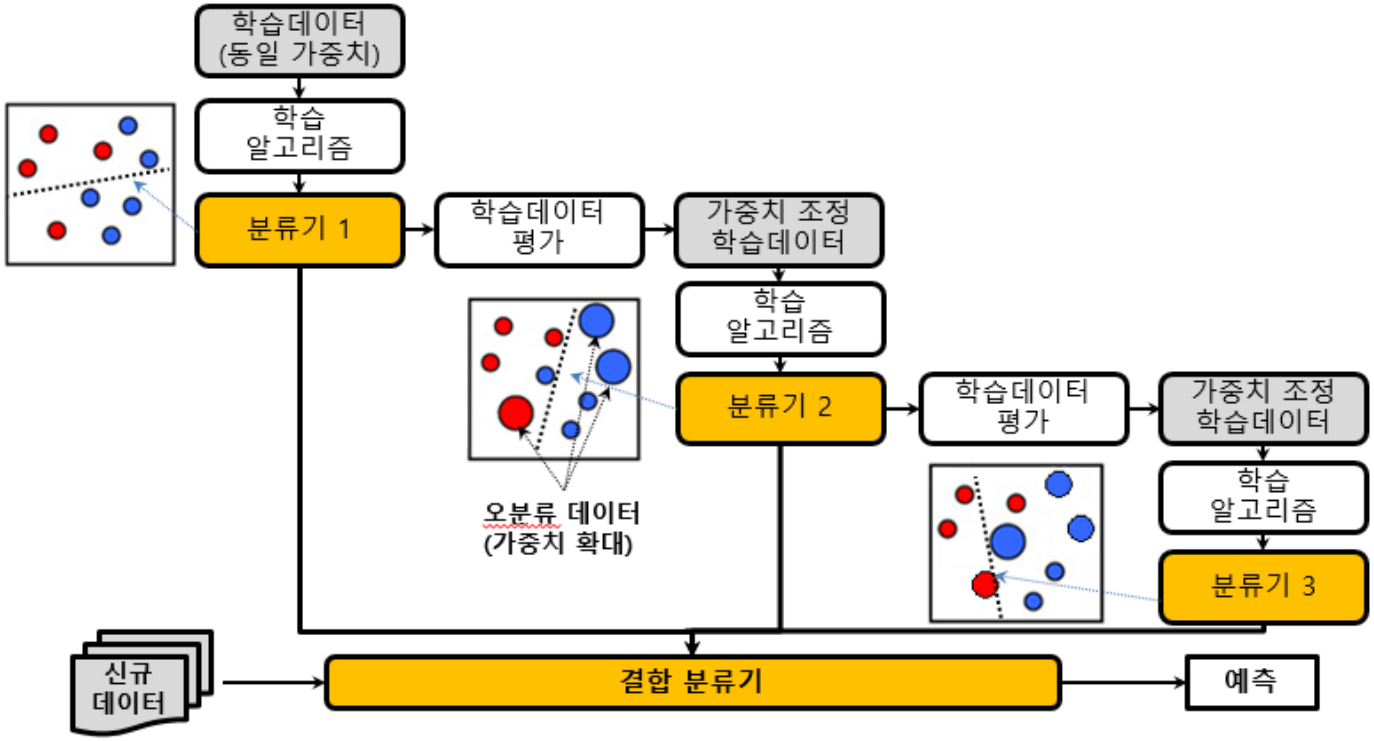
```
import pandas as pd
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics

iris = datasets.load_iris()
print('Class names :', iris.target_names)
print('target : [0:setosa, 1:versicolor, 2:virginica]')
print('No. of Data :', len(iris.data))
print('Feature names :', iris.feature_names)

data = pd.DataFrame( {
    'sepal length': iris.data[:, 0], 'sepal width': iris.data[:, 1], 'petal length': iris.data[:, 2],
    'petal width': iris.data[:, 3], 'species': iris.target }
)
print(data.head())  # 일부 데이터 출력

x = data[['sepal length', 'sepal width', 'petal length', 'petal width']]  # 입력
y = data[ ' species ' ]  # 출력
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)  # 데스트 데이터 30%
print("No. of training data: ", len(x_train))
print("No. of test data:", len(y_test))

forest = RandomForestClassifier(n_estimators=100)  # 모델 생성
forest.fit(x_train, y_train)

y_pred = forest.predict(x_test)  # 추론(예측)
print('Accuracy :', metrics.accuracy_score(y_test, y_pred))
```

```
Class names : ['setosa' 'versicolor' 'virginica']
target : [0:setosa, 1:versicolor, 2:virginica]
No. of Data : 150
Feature names : ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
   sepal length  sepal width  petal length  petal width  species
0           5.1          3.5           1.4          0.2        0
1           4.9          3.0           1.4          0.2        0
2           4.7          3.2           1.3          0.2        0
3           4.6          3.1           1.5          0.2        0
4           5.0          3.6           1.4          0.2        0
No. of training data:  105
No. of test data: 45
Accuracy : 1.0
```

3. 부스팅 알고리즘

❖ **부스팅**(boosting)

- k개의 예측 모델을 **순차적**으로 만들어 가는 **앙상블 모델** 생성
- **오차**에 따라 **학습 데이터**에 **가중치 또는 값**을 **변경**해가면서 예측 모델 생성

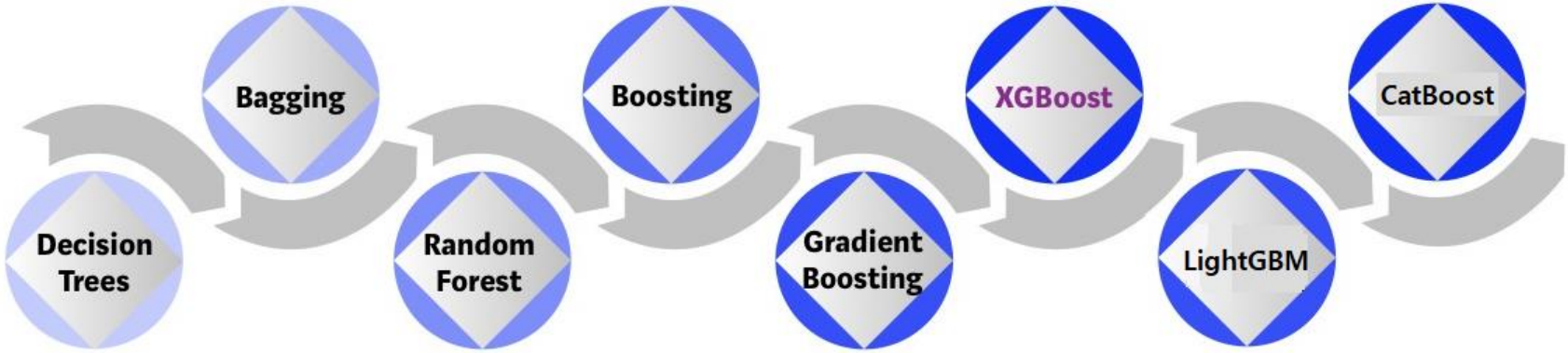


- **AdaBoost**
- **Gradient Boosting**
- **XGB**

부스팅 알고리즘

❖ **부스팅 알고리즘**

- 다양한 부스팅 알고리즘 개발
- Gradient Boosting
- XGBoost
- LightBoost
- CatBoost



[실습] LightBoost

```
!pip install lightgbm
from lightgbm import LGBMClassifier, LGBMRegressor
from lightgbm import plot_importance, plot_metric, plot_tree
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_validate

iris = load_iris()
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.2, random_state=123)
lgbmc = LGBMClassifier(n_estimators=400)
evals = [(X_test, y_test)]
lgbmc.fit(X_train, y_train, early_stopping_rounds=100, eval_metric='logloss', eval_set=evals, verbose=True)
preds = lgbmc.predict(X_test)

cross_val = cross_validate(
    estimator=lgbmc,
    X=iris.data, y=iris.target,
    cv=5
)
print('avg fit time: {} (+/- {})'.format(cross_val['fit_time'].mean(), cross_val['fit_time'].std()))
print('avg score time: {} (+/- {})'.format(cross_val['score_time'].mean(), cross_val['score_time'].std()))
print('avg test score: {} (+/- {})'.format(cross_val['test_score'].mean(), cross_val['test_score'].std()))

plot_metric(lgbmc)
plot_importance(lgbmc, figsize=(10,12))
```

```
[58]    valid_0's multi_logloss: 0.128187       valid_0's multi_logloss: 0.128187
avg fit time: 0.07379713058471679 (+/- 0.002962591142713687)
avg score time: 0.002210187911987305 (+/- 0.0017766080812927243)
avg test score: 0.9533333333333335 (+/- 0.06182412330330468)
```

[실습] CatBoost

❖ Rotton_tomatoes 데이터

- 영화평점 DB

	id	synopsis	rating_MPAA	genre	director	writer	theater_date	dvd_date	box_office	runtime	studio	dvd_date_int	theater_date_int	review	rating	fresh	critic	top_critic	publisher	date	date_int	rating_10
0	830.0	A gay New Yorker stages a marriage of convenie...	R	Art House and International | Comedy | Drama |...	Ang Lee	Ang Lee | James Schamus | Neil Peng	1993-08-04	2004-06-15	NaN	111.0	NaN	20040615	19930804	NaN	0.800000	fresh	Carol Cling	0	Las Vegas Review-Journal	2004-04-16	20040416.0	8.0
1	1161.0	Screenwriter Nimrod Antal makes an impressive ...	R	Action and Adventure | Art House and Internati...	NaN	NaN	2005-04-01	2005-08-30	116783.0	105.0	ThinkFilm Inc.	20050830	20050401	One very long, dark ride.	0.647059	rotten	NaN	0	E! Online	2005-04-22	20050422.0	6.0
2	596.0	"Arctic Tale" is an epic adventure that explor...	G	Documentary | Special Interest	Adam Ravetch | Sarah Robertson	Linda Woolverton | Mose Richards | Kristin Gore	2007-08-17	2017-08-01	598103.0	86.0	Paramount Vantage	20170801	20070817	I'm no holdout about the reality of global war...	0.625000	rotten	Jack Mathews	1	New York Daily News	2007-07-27	20070727.0	6.0
3	1585.0	A dating doctor claims that with his services ...	PG-13	Comedy | Romance	Andrew Tennant | Andy Tennant	Kevin Bisch	2005-02-11	2005-06-14	177575142.0	120.0	Sony Pictures	20050614	20050211	... Adds up to far more than the formula tvpic...	0.875000	fresh	Greg Maki	0	Star-Democrat (Easton, MD)	2005-02-11	20050211.0	9.0

```
!pip install catboost
```

```
import pandas as pd
import numpy as np
from catboost import Pool, CatBoostClassifier
from catboost.datasets import rotten_tomatoes
```

```
learn, _ = rotten_tomatoes()
print('Feature names:₩n' + ', '.join(list(learn)))
```

```
Feature names: id, synopsis, rating_MPAA, genre, director, writer, theater_date, dvd_date,
box_office, runtime, studio, dvd_date_int, theater_date_int, review, rating, fresh, critic,
top_critic, publisher, date, date_int, rating_10
```

```
auxiliary_columns = ['id', 'theater_date', 'dvd_date', 'rating', 'date']
cat_features = ['rating_MPAA', 'studio', 'fresh', 'critic', 'top_critic', 'publisher']
text_features = ['synopsis', 'genre', 'director', 'writer', 'review']
```

```python
def get_processed_rotten_tomatoes():
    learn, test = rotten_tomatoes()

    def fill_na(df, features):
        for feature in features:
            df[feature].fillna('', inplace=True)

    def preprocess_data_part(data_part):
        data_part = data_part.drop(auxiliary_columns, axis=1)

        fill_na(data_part, cat_features)
        fill_na(data_part, text_features)

        X = data_part.drop(['rating_10'], axis=1)
        y = data_part['rating_10']
        return X, y

    X_learn, y_learn = preprocess_data_part(learn)
    X_test, y_test = preprocess_data_part(test)

    return X_learn, X_test, y_learn, y_test

X_train, X_test, y_train, y_test = get_processed_rotten_tomatoes()
```

	id	synopsis	rating_MPAA	genre	director	writer	theater_date	dvd_date
0	830.0	A gay New Yorker stages a marriage of convenie...	R	Art House and International | Comedy | Drama |...	Ang Lee	Ang Lee | James Schamus | Neil Peng	1993-08-04	2004-06-15
1	1161.0	Screenwriter Nimrod Antal makes an impressive ...	R	Action and Adventure | Art House and Internati...	NaN	NaN	2005-04-01	2005-08-30
2	596.0	"Arctic Tale" is an epic adventure that explor...	G	Documentary | Special Interest	Adam Ravetch | Sarah Robertson	Linda Woolverton | Mose Richards | Kristin Gore	2007-08-17	2017-08-01

	synopsis	rating_MPAA	genre	director	writer	box_office
0	A gay New Yorker stages a marriage of convenie...	R	Art House and International | Comedy | Drama |...	Ang Lee	Ang Lee | James Schamus | Neil Peng	NaN
1	Screenwriter Nimrod Antal makes an impressive ...	R	Action and Adventure | Art House and Internati...			116783.0
2	"Arctic Tale" is an epic adventure that explor...	G	Documentary | Special Interest	Adam Ravetch | Sarah Robertson	Linda Woolverton | Mose Richards | Kristin Gore	598103.0
3	A dating doctor claims that with his services ...	PG-13	Comedy | Romance	Andrew Tennant | Andy Tennant	Kevin Bisch	177575142.0

```python
def fit_catboost_on_rotten_tomatoes(X_train, X_test, y_train, y_test, catboost_params={}, verbose=100):
    learn_pool = Pool(
        X_train,
        y_train,
        cat_features=cat_features,
        text_features=text_features,
        feature_names=list(X_train)
    )
    test_pool = Pool(
        X_test,
        y_test,
        cat_features=cat_features,
        text_features=text_features,
        feature_names=list(X_train)
    )

    catboost_default_params = {
        'iterations': 1000,
        'learning_rate': 0.03,
        'eval_metric': 'Accuracy',
        'task_type': 'GPU'
    }

    catboost_default_params.update(catboost_params)

    model = CatBoostClassifier(**catboost_default_params)
    model.fit(learn_pool, eval_set=test_pool, verbose=verbose)

    return model
```

```
0:      learn: 0.3858217        test: 0.3980927 best: 0.3980927 (0)
100:    learn: 0.4458303        test: 0.4559237 best: 0.4561682 (99)
200:    learn: 0.4560100        test: 0.4603252 best: 0.4608143 (198)
300:    learn: 0.4637442        test: 0.4652158 best: 0.4652158 (299)
400:    learn: 0.4716312        test: 0.4683947 best: 0.4683947 (395)
500:    learn: 0.4780509        test: 0.4660716 best: 0.4686392 (402)
600:    learn: 0.4850208        test: 0.4679056 best: 0.4686392 (402)
700:    learn: 0.4905539        test: 0.4682724 best: 0.4692505 (619)
800:    learn: 0.4954145        test: 0.4698618 best: 0.4698618 (799)
900:    learn: 0.4997860        test: 0.4705954 best: 0.4716958 (843)
999:    learn: 0.5036989        test: 0.4699841 best: 0.4716958 (843)
bestTest = 0.4716958063
bestIteration = 843
Shrink model to first 844 iterations.
```

```python
fit_catboost_on_rotten_tomatoes(X_train, X_test, y_train, y_test)
```