ML Day7 (용어 정리 및 Python기초 알고리즘)

▼ **MSE (Mean Squared Error) - ^(hat); 예측값**

```
pred1, pred2, pred3 = 10, 20, 30
y1, y2, y3 = 10, 25, 40
n_data = 3
s_error1 = (pred1 - y1)**2
s_error2 = (pred2 - y2)**2
s_error3 = (pred3 - y3)**2
mse = (s_error1 + s_error2 + s_error3)/n_data
print(mse)
```

▼ [1]: 예측값

▼ [2]: 결괏값

```
C:\Users\SBAUser\anaconda3\python.exe C:\Users\SBAUser\anaconda3\envs\SBA\Jong\test.py
41.666666666666664
```

▼ **Mean subtraction(2)**

```python
scores = [10, 20, 30]
n_student = len(scores)
mean = (scores[0] + scores[1] + scores[2])/n_student
print("score mean:", mean)
scores[0] -= mean
scores[1] -= mean
scores[2] -= mean
mean = (scores[0] + scores[1] + scores[2])/n_student
print("score mean:", mean)
```

▼ [3]: scores의 각 index에 해당하는 값들을 합산하여 n_student(len(scores))로 나눠 평균을 구함

▼ [5]: scores[index]에 해당하는 각 원소에 평균을 차감하여 다시 scores[index]에 치환

▼ [8]: 각 원소에서 평균을 차감한 값을 다시 합산하여 n_student로 나누어 평균을 구함(Mean subtraction)

```
C:\Users\SBAUser\anaconda3\python.exe C:\Users\SBAUser\anaconda3\envs\SBA\Jong\test.py
score mean: 20.0
score mean: 0.0
```

▼ **List를 이용한 variance와 표준편차(standard deviation)**

```
scores = [10, 20, 30]
n_student = len(scores)
mean = (scores[0] + scores[1] + scores[2])/n_student
square_of_mean = mean**2
mean_of_square = (scores[0]**2 + scores[1]**2 + scores[2]**2)/n_student
variance = mean_of_square - square_of_mean
std = variance**0.5
print("score mean:", mean)
print("score standard deviation:", std)
```

▼ [4]: square_of_mean = 평균의 제곱

▼ [5]: mean_of_square = 제곱의 평균(각 원소의 제곱들의 합을 n_student로 나눔)

▼ [6]: variance = 제곱의 평균 - 평균의 제곱

▼ [7]: std(표준편차) = variance**0.5

```
score mean: 20.0
score standard deviation: 8.16496580927726
```

▼ Standardization(2)

```
scores = [10, 20, 30]
n_student = len(scores)
mean = (scores[0] + scores[1] + scores[2])/n_student
square_of_mean = mean**2
mean_of_square = (scores[0]**2 + scores[1]**2 + scores[2]**2)/n_student
variance = mean_of_square - square_of_mean
std = variance**0.5
print("score mean:", mean)
print("score standard deviation:", std)
scores[0] = (scores[0] - mean)/std
scores[1] = (scores[1] - mean)/std
scores[2] = (scores[2] - mean)/std
mean = (scores[0] + scores[1] + scores[2])/n_student
square_of_mean = mean**2
mean_of_square = (scores[0]**2 + scores[1]**2 + scores[2]**2)/n_student
variance = mean_of_square - square_of_mean
std = variance**0.5
print("score mean:", mean)
print("score standard deviation:", std)
```

▼ [10~12] : 각각의 score값에서 위에서 구한 mean(평균)을 빼준 후 std(표준편차)로 나눠준다.

▼ [13:] : 바뀐 score값으로 다시 평균과 분산, 표준편차를 구해준다.

```
score mean: 20.0
score standard deviation: 8.16496580927726
score mean: 0.0
score standard deviation: 1.0
```

▼ Hadamard Product(2)

```python
# method.1
v1, v2 = [1, 2, 3], [3, 4, 5]
v3 = [v1[0] * v2[0], v1[1] * v2[1], v1[2] * v2[2]]
print(v3)
# method.2
v1, v2 = [1, 2, 3], [3, 4, 5]
v3 = [0, 0, 0]
v3[0] = v1[0] * v2[0]
v3[1] = v1[1] * v2[1]
v3[2] = v1[2] * v2[2]
print(v3)
```

▼ [3] : v1, v2 list의 같은 index값끼리 곱해주고 그 결과들로 v3 list로 생성

```
[3, 8, 15]
[3, 8, 15]
```

▼ Hadamard Product(3)

```python
v1, v2 = [1, 2, 3], [3, 4, 5]
v3 = list()
v3.append(v1[0] * v2[0])
v3.append(v1[1] * v2[1])
v3.append(v1[2] * v2[2])
print(v3)
```

▼ v1, v2 list의 같은 index값끼리 곱해준 후 비어있는 v3 list에 같은 index자리에 append

```
[3, 8, 15]
```

▼ Vector Norm(2)

```python
v1 = [1, 2, 3]
# method.1
norm = (v1[0]**2 + v1[1]**2 + v1[2]**2)**0.5
print(norm)
# method.2
norm = 0
norm += v1[0]**2
norm += v1[1]**2
norm += v1[2]**2
norm **= 0.5
print(norm)
```

▼ [3]: v1 list의 각 원소의 제곱을 해준후 합산하여 루트를 씌운 값(norm)

```
3.7416573867739413
3.7416573867739413
```

▼ Making Unit Vectors(2)

```
v1 = [1, 2, 3]
norm = (v1[0]**2 + v1[1]**2 + v1[2]**2)**0.5
print(norm)
v1 = [v1[0]/norm, v1[1]/norm, v1[2]/norm]
norm = (v1[0]**2 + v1[1]**2 + v1[2]**2)**0.5
print(norm)
```

▼ [4]: 위에서 구한 norm값으로 v1 list의 각 원소들을 norm으로 나눠준 후 다시 v1 list에 대입

▼ [5] : [4]에서 바뀐 v1 list 원소들의 값으로 다시 norm을 구함

```
3.7416573867739413
1.0
```

▼ Dot Product(2)

```
v1, v2 = [1, 2, 3], [3, 4, 5]
# method.1
dot_prod = v1[0]*v2[0] + v1[1]*v2[1] + v1[2]*v2[2]
print(dot_prod)
# method.2
dot_prod = 0
dot_prod += v1[0]*v2[0]
dot_prod += v1[1]*v2[1]
dot_prod += v1[2]*v2[2]
print(dot_prod)
```

▼ [3]: v1, v2 list의 같은 index위치의 원소들끼리 곱한 각각의 값을 합산(dot product)

```
26
26
```

▼ Euclidean Distance(2)

```python
v1, v2 = [1, 2, 3], [3, 4, 5]
e_distance = 0
e_distance += (v1[0] - v2[0])**2
e_distance += (v1[1] - v2[1])**2
e_distance += (v1[2] - v2[2])**2
e_distance **= 0.5
print(e_distance)
```

▼ [3] : v1과 v2 list의 같은 index위치의 원소들의 차에 제곱을 한 후 e_distance(시작값=0)에 더해준 결과를 다시 변수에 대입

▼ [6] : 마지막 e_distance의 제곱근(euclidean distance)

```
3.4641016151377544
```

▼ **Mean Squared Error(2)**

```python
predictions = [10, 20, 30]
labels = [10, 25, 40]
n_data = len(predictions)
mse = 0
mse += (predictions[0] - labels[0])**2
mse += (predictions[1] - labels[1])**2
mse += (predictions[2] - labels[2])**2
mse /= n_data
print(mse)
```

▼ [1] : 예측값

▼ [2] : 결과값

▼ [5~8] : 시작값을 0으로 가진 mse 변수에 예측값에서 결과값을 뺀 후 제곱을 해준 값을 계속 합산하여 준다. 마지막의 mse값을 n_data(predictions의 길이)로 나눈다. (mean squared error)

```
41.666666666666664
```

<Python 연습>

▼ Iteration 횟수 구하기

```python
numbers = [1, 4, 5, 6, 4, 2, 1]
iter_cnt = 0
for _ in numbers:
    iter_cnt += 1
print(iter_cnt)
```

```
7
```

▼ 1~100까지의 합 구하기

```python
num_sum = 0
for i in range(101):
    num_sum += i
print(num_sum)
```

```
5050
```

▼ 100개의 0을 가진 list 만들기

```python
numbers = []
for _ in range(1, 101):
    numbers.append(0)
print(numbers)
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

▼ for loop로 list의 원소 접근하기(2)

```python
scores = [10, 20, 30]
# method.1
for score in scores: print(score)
# method.2 (인덱스 번호로 접근하기)
for score_idx in range(len(scores)): print(scores[score_idx])
```

```
10
20
30
10
20
30
```

▼ 수학 점수들의 평균 구하기(3)

```python
scores = [10, 20, 30]
# method.1
score_sum = 0
n_student = 0
for score in scores:
    score_sum += score
    n_student += 1
score_mean = score_sum / n_student
print("score mean:", score_mean)
# method.2
score_sum = 0
for score_idx in range(len(scores)):
    score_sum += scores[score_idx]
score_mean = score_sum / len(scores)
print("score mean:", score_mean)
```

```
score mean: 20.0
score mean: 20.0
```

▼ **두 과목의 평균 구하기**

```
1  math_scores = [40, 60, 80]
2  english_scores = [30, 40, 50]
3  n_class = 2
4  n_student = len(math_scores)
5
6  score_sums = []
7  score_means = []
8
9  for _ in range(n_class):
10     score_sums.append(0)
11
12 for student_idx in range(n_student):
13     score_sums[0] += math_scores[student_idx]
14     score_sums[1] += english_scores[student_idx]
15 print("sums of scores:", score_sums)
16
17 for class_idx in range(n_class):
18     class_mean = score_sums[class_idx] / n_student
19     score_means.append(class_mean)
20 print("mean of scores:", score_means)
```

▼ [9] : n_class의 길이(0, 1)만큼 for문으로 score_sum에 0을 append해준다.

▼ [12] : for문으로 n_student길이만큼 student_idx에 해당하는 값들을 합산하여 score_sum[0](수학점수)에 대입하고 또한 score_sums[1](영어점수)에 같은 방식으로 대입하여 준다.

▼ [17] : 위의 결과인 score_sums를 for문을 이용하여 수학과 영어 점수 합을 n_student로 나눠 score_means list에 append

```
sums of scores: [180, 120]
mean of scores: [60.0, 40.0]
```