

# [Machine Learning]

[2021-1]

---

## Homework 3

Lec 8, 9, 10

**[DATE]** 2021.05.14

---

**Student ID :** 2016112158

**Name :** KimHeeSuf

**Professor :** Juntae Kim



1. Explain what Feature Selection and Dimensionality Reduction are, and why they are needed in machine learning. You can use examples for explanation. (10pts)

Your Answer
<p>feature selection 은 데이터의 여러 feature 중에서 예측에 있어서 해당 feature 가 유용한지 유용하지 않은지 확인하는 과정을 뜻한다. 유용하다면 해당 feature 를 사용하고, 유용하지 않다면 버린다. 불필요한 특징을 제거하여 간결한 특징 집합을 만드는 것이다. 예를 들어, 사람의 나이를 예측할 때 feature 로 얼굴 주름의 개수, 흰머리의 개수, 피부색, 발의 크기 등이 있다고 가정하자. 이때 얼굴 주름의 개수나 흰머리의 개수는 나이를 예측할 때 유용하지만 발의 크기나 피부색 등은 유용하지 않다고 볼 수 있다. 따라서 나이 예측에 있어서 얼굴주름의 개수, 흰머리 개수 만으로 feature 를 설정하는 것을 feature selection 으로 볼 수 있다.</p> <p>Dimesionality Reduction 은 관찰 대상을 잘 설명할 수 있는 잠재 공간(latent space)은 실제 공간( observation space)보다 작을 수 있다. 이렇게 관찰 공간 위의 샘플을 기반으로 잠재공간을 파악하는 것을 차원축소라 한다. iris 데이터의 feature 들인 sepal_length, sepal_width, petal_length, petal_width 에 대해서 PCA 를 통해 2 차원 공간으로 만들 수 있다</p>

2. Answer following questions. (20pts)

2-1. Perform one-hot encoding on the class of Iris data.

Expected Output

	0	1	2	3	4
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

```
print(y[0])
print(y[50])
print(y[100])
```

```
[1.  0.  0.]
[0.  1.  0.]
[0.  0.  1.]
```

### Code

```
import pandas as pd
import numpy as np

iris = pd.read_csv("./iris.csv")
df = pd.DataFrame(iris)
df.columns=[0,1,2,3,4]

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df[4] = le.fit_transform(df[4])

onehot_df = pd.get_dummies(df,columns=[4])

y = np.array(onehot_df[["4_0","4_1","4_2"]], dtype=np.float32)
type(y)
print(y[0])
print(y[50])
print(y[100])
```

### Result(Captured images)

<pre>[15] ▶ ≡ ML  import pandas as pd import numpy as np  iris = pd.read_csv("./iris.csv") df = pd.DataFrame(iris) df.columns=[0,1,2,3,4]  from sklearn.preprocessing import LabelEncoder le = LabelEncoder() df[4] = le.fit_transform(df[4])  onehot_df = pd.get_dummies(df,columns=[4])  y = np.array(onehot_df[["4_0","4_1","4_2"]], dtype=np.float32) type(y) print(y[0]) print(y[50]) print(y[100])  [1. 0. 0.] [0. 1. 0.] [0. 0. 1.]</pre>	<table><tr><th>Description</th></tr><tr><td>iris.csv 를 읽어와서 column 을 설정해준 후, LabelEncoder 로 3 개의 label 을 0,1,2 로 바꾼다. 그 후 get_dummies 로 onehot encoding 해주면 "4_0", "4_1", "4_2"로 인코딩된다. 해당 부분만 np.array 로 가져오면 된다</td></tr></table>	Description	iris.csv 를 읽어와서 column 을 설정해준 후, LabelEncoder 로 3 개의 label 을 0,1,2 로 바꾼다. 그 후 get_dummies 로 onehot encoding 해주면 "4_0", "4_1", "4_2"로 인코딩된다. 해당 부분만 np.array 로 가져오면 된다
Description			
iris.csv 를 읽어와서 column 을 설정해준 후, LabelEncoder 로 3 개의 label 을 0,1,2 로 바꾼다. 그 후 get_dummies 로 onehot encoding 해주면 "4_0", "4_1", "4_2"로 인코딩된다. 해당 부분만 np.array 로 가져오면 된다			

2-2. Perform Standardization and Normalization on the Iris data(except class. split train and test set)

Expected Output

	0	1	2	3	4
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

X\_train\_std

```
array([[ -0.923122,  1.61656532, -1.05236498, -1.06144375],
       [ -0.923122,  0.96288358, -1.33253314, -1.19475906],
       [ -0.21128986,  2.9239288, -1.27649951, -1.06144375],
       [  2.16148395, -0.56237382,  1.63724937,  1.07160111],
```

X\_test\_std

```
array([[ -0.44856724,  0.96288358, -1.38856677, -1.32807436],
       [ -1.75359284,  0.30920184, -1.38856677, -1.32807436],
       [  1.21237443,  0.09130793,  0.90881215,  1.20491641],
       [  0.97509705,  0.52709575,  1.07691305,  1.73817763],
```

X\_train\_norm

```
array([[0.2, 0.75, 0.15254237, 0.13043478],
       [0.2, 0.625, 0.06779661, 0.08695652],
       [0.37142857, 1., 0.08474576, 0.13043478],
       [0.94285714, 0.33333333, 0.96610169, 0.82608696],
```

X\_test\_norm

```
array([[ 0.31428571, 0.625, 0.05084746, 0.04347826],
       [ 0., 0.5, 0.05084746, 0.04347826],
       [ 0.71428571, 0.45833333, 0.74576271, 0.86956522],
       [ 0.65714286, 0.54166667, 0.79661017, 1.04347826],
```

### Code

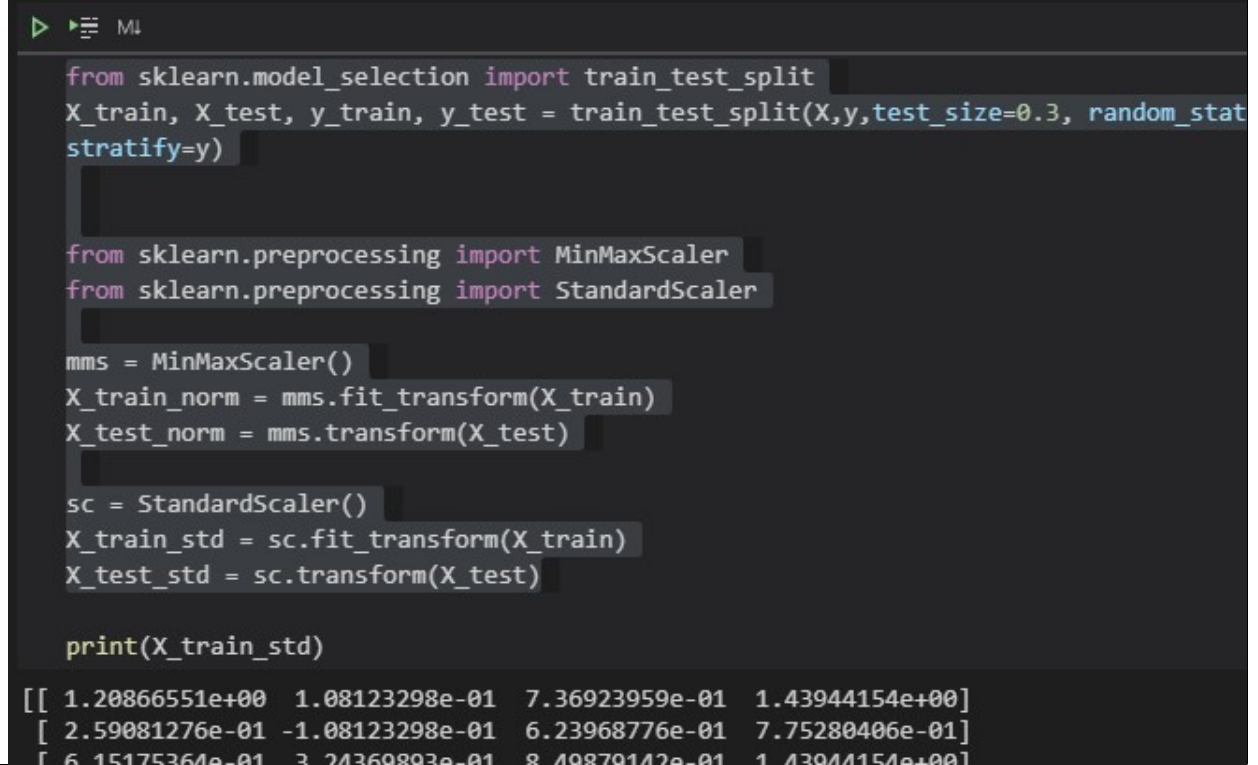
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3, random_state=0, stratify=y)

from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler

mms = MinMaxScaler()
X_train_norm = mms.fit_transform(X_train)
X_test_norm = mms.transform(X_test)

sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
X_test_std = sc.transform(X_test)
```

### Result(Captured images)



```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3, random_state=0, stratify=y)

from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler

mms = MinMaxScaler()
X_train_norm = mms.fit_transform(X_train)
X_test_norm = mms.transform(X_test)

sc = StandardScaler()
X_train_std = sc.fit_transform(X_train)
X_test_std = sc.transform(X_test)

print(X_train_std)
```

```
[[ 1.20866551e+00  1.08123298e-01  7.36923959e-01  1.43944154e+00]
 [ 2.59081276e-01 -1.08123298e-01  6.23968776e-01  7.75280406e-01]
 [ 6.15175364e-01  3.24369893e-01  8.40870142e-01  1.43944154e+00]]
```

### Description

MinMaxScaler 객체 mms 를 선언하여 X\_train\_norm, X\_test\_norm 으로 변환하였고, StandardScaler 객체 sc 를 선언하여 X\_train\_std, X\_test\_std 로 변환하였다



2-3. Perform PCA on the iris data, check the explained variance ratio and choose 2 components.

Expected Output

```
pca.explained_variance_ratio_
```

```
array([0.73388509, 0.22547013, 0.03516739, 0.00547739])
```

```
X_train_pca
```

```
array([[ -2.14486538e+00,  1.00530441e+00],  
       [ -2.19536586e+00,  3.91637317e-01],  
       [ -2.28031379e+00,  2.47772726e+00],  
       [  2.82749300e+00,  4.78884590e-01],
```

```
X_test_pca
```

```
array([[ 1.90725282, -0.90395608],  
       [ 2.48651617,  0.10200288],  
       [-1.96227467, -0.48731458],  
       [-2.20910465, -0.84096144],
```

Code

```
from sklearn.decomposition import PCA  
  
pca = PCA()  
X_train_pca = pca.fit_transform(X_train_std)  
X_test_pca = pca.transform(X_test_std)  
pca.explained_variance_ratio_
```

Result(Captured images)




```

from sklearn.decomposition import PCA

pca = PCA()
X_train_pca = pca.fit_transform(X_train_std)
X_test_pca = pca.transform(X_test_std)
pca.explained_variance_ratio_

array([0.72641723, 0.2327667 , 0.03557513, 0.00524093])

```

**X\_train\_pca**

```

array([[ 1.84485110e+00,  6.56588154e-01,  1.94530294e-01,
         4.63881713e-01],
       [ 9.65665624e-01,  5.31080178e-02,  3.65236270e-01,
         1.11111111e-01]])

```

Description
<p>PCA 객체 <code>pca</code> 를 선언하여 차원축소 하여 <code>X_train_pca</code>, <code>X_test_pca</code> 로 변환하였다. 예시에서, <code>explained_variance_ratio_</code> 는 4 개인데 <code>X_train_pca[0]</code> 의 차원은 2D 이다. <code>explained_variance_ratio_</code> 가 4 개라면 <code>X_train_pca[0]</code> 의 차원은 4D 여야 하는거 아닌가..</p>

3. Explain what PCA(Principle Component Analysis) is and why you need it in machine learning. (10pts)

Your Answer

PCA 는 데이터의 **variance** 를 최대한 보존하면서 서로 직교하는 축을 찾아 고차원 공간의 표본을 저차원으로 투영하는 차원축소 기법이다. 데이터의 **feature** 가 많아 차원이 늘어나면 데이터 공간의 부피가 기하급수적으로 증가하여 데이터의 밀도가 차원이 증가할수록 희소(**sparse**)해진다. 밀도가 희소해진다는건 데이터포인트 간의 거리가 증가한다는 것이고, 이를 이용해 머신러닝 모델을 학습시키면 모델이 오버피팅될 위험이 커진다. 이를 해결하기 위해 분산을 최대한 보존하는 **PCA** 를 통해 차원의 수를 줄일 필요가 있다.

4. Explain what Bias and Variances are and how they affect the performance of machine learning models. (10pts)

Your Answer



---

**Bias** 는 지나치게 단순한 모델로 인한 **error** 이다. 예측값과 실제값의 차이로 볼 수 있다. **Bias** 가 높으면 **underfitting** 이 발생한다.

**Variance** 는 지나치게 복잡한 모델로 인한 **error** 이다. **Variance** 는 주어진 데이터로 학습한 모델이 예측한 값의 변동성을 의미한다. **Variance** 가 높으면 **overfitting** 이 일어난다.

---

5. Apply Bagging and Adaboost algorithms on Iris dataset. Set the base\_estimator to “DecisionTreeClassifier”. We recommend you use “BaggingClassifier” and “AdaBoostClassifier” in Scikit-learn. (20pts)

### Iris data Setting(Use this Code)

```
import numpy as np
import pandas as pd
df = pd.read_csv('https://archive.ics.uci.edu/ml/'
                 'machine-learning-databases/iris/iris.data', header=None)
X = df.iloc[50:150, [2, 3]].values
y = df.iloc[50:150, 4].values
y = np.where(y == 'Iris-virginica', -1, 1)
```

### Expected Output

```
print("Decision Tree train/test accuracies %.3f/%.3f"
      % (tree_clf.score(X_train, y_train), tree_clf.score(X_test, y_test)))
print("Bagging train/test accuracies %.3f/%.3f"
      % (bag_clf.score(X_train, y_train), bag_clf.score(X_test, y_test)))
print("Adaboost train/test accuracies %.3f/%.3f"
      % (boost_clf.score(X_train, y_train), boost_clf.score(X_test, y_test)))
```

```
Decision Tree train/test accuracies 0.929/0.933
Bagging train/test accuracies 0.929/0.967
Adaboost train/test accuracies 0.986/0.967
```

Code
hw3-5.ipynb 참고
Result(Captured images)

```
[9] ▶ M4

print("Decision Tree train/test accuracies %.3f/%.3f"
      % (tree.score(X_train_std, y_train), tree.score(X_test_std, y_test)))
print("Bagging train/test accuracies %.3f/%.3f"
      % (bag.score(X_train_std, y_train), bag.score(X_test_std, y_test)))
print("Adaboost train/test accuracies %.3f/%.3f"
      % (ada.score(X_train_std, y_train), ada.score(X_test_std, y_test)))

Decision Tree train/test accuracies 0.988/0.950
Bagging train/test accuracies 0.975/0.950
Adaboost train/test accuracies 0.988/0.950
```

Description
X_train, X_test, y_train, y_test 를 생성하고 DecisionTreeClassifier 객체 tree, BaggingClassifier 객체 bag, AdaBoostClassifier 객체 ada 를 선언하고 학습한 후, score 를 측정하였다.

6. Apply Logistic Regression and Decision Tree Classifier on 20newsgroups Dataset(Document Classification). (30pts)

20newsgroup dataset is a news dataset consisting of 20 categories.

In this question, we only use samples from 4 categories.

Follow this process:

- 1) Load the data.

```

from sklearn.datasets import fetch_20newsgroups

categories = ['alt.atheism', 'soc.religion.christian',
              'comp.graphics', 'sci.med']

news = fetch_20newsgroups(categories=categories,
                          shuffle=False,
                          random_state=42)

X = news.data
y = news.target

X[0]

"From: keith@cco.caltech.edu (Keith Allan Schneider)\nSubject: Re: <
d.caltech.edu\n\nbobbe@vice.ICO.TEK.COM (Robert Beauchaine) writes:\n
ral hypothesis, you have to successfully argue that\n>domestication
animals exhibit behaviors not found in the wild. I\ndon't think tha
years\nto produce certain behaviors, etc.\n\nkeith\n"

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3,
                                                    stratify=y,
                                                    random_state=1)

```

- 2) Make preprocessor function & porter stemmer tokenizer function.
- 3) Apply the TF-IDF(TFidfVectorizer) on the data.
  - Use the preprocessor function & porter stemmer tokenizer
  - Use stop-words
  - Drop terms occurred in more than 10% of docs
  - Drop terms occurred in less than 10 docs
- 4) Train machine learning models(Logistic Regression, Decision Tree) using TF-IDF vectors.
- 5) Predict the categories of following 4 sentences

'The outbreak was declared a global pandemic by the World Health Organization (WHO) on 11 March.'

'Today, computer graphics is a core technology in digital photography, film, video games, cell phone and computer displays, and many specialized applications.'

'Arguments for atheism range from philosophical to social and historical approaches.'

'The Bible is a compilation of many shorter books written at different times by a variety of authors, and later assembled into the biblical canon.'

## Expected Output

### Training a Logistic Regression Model for Document Classification

```
# train using Logistic Regression
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(penalty="l2", verbose=1)
lr.fit(train_vector, y_train)

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers
[Parallel(n_jobs=1)]: Done 1 out of 1 | elapsed: 0.0s finished
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                    intercept_scaling=1, l1_ratio=None, max_iter=100,
                    multi_class='auto', n_jobs=None, penalty='l2',
                    random_state=None, solver='lbfgs', tol=0.0001, verbose=1,
                    warm_start=False)

# train score
lr.score(train_vector, y_train)

0.9905003166561115

# test score
lr.score(test_vector, y_test)

0.9542772861356932
```

## Training a Decision Tree Model for Document Classification

```
from sklearn.tree import DecisionTreeClassifier
```

```
tree = DecisionTreeClassifier()  
tree.fit(train_vector, y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',  
                       max_depth=None, max_features=None, max_leaf_nodes=None,  
                       min_impurity_decrease=0.0, min_impurity_split=None,  
                       min_samples_leaf=1, min_samples_split=2,  
                       min_weight_fraction_leaf=0.0, presort='deprecated',  
                       random_state=None, splitter='best')
```

```
tree.score(train_vector, y_train) # Train Accuracy
```

```
1.0
```

```
tree.score(test_vector, y_test) # Test Accuracy
```

```
0.831858407079646
```



### finding 20 most important terms

```
importances = tree.feature_importances_  
indices = np.argsort(importances)[::-1]  
  
for f in range(20):  
    print("%2d. %-30s %f" % (f+1,  
                             [w for w, n in tfidf.vocabulary_.items()  
                              if n == indices[f]],  
                             importances[indices[f]]))
```

1.	['graphic']	0.099578
2.	['christ']	0.067288
3.	['keith']	0.057261
4.	['islam']	0.046768
5.	['church']	0.045765
6.	['file']	0.039110
7.	['pitt']	0.036919
8.	['doctor']	0.031218
9.	['atheism']	0.028081
10.	['faith']	0.025263
11.	['food']	0.021838
12.	['medic']	0.020223
13.	['imag']	0.017457
14.	['moral']	0.014317
15.	['3d']	0.011887
16.	['benedikt']	0.011624
17.	['window']	0.011060
18.	['methodolog']	0.010056
19.	['algorithm']	0.010012
20.	['wvc']	0.009134

### Test Sentences

```
docs = ['The outbreak was declared a global pandemic by the World Hea  
        'Today, computer graphics is a core technology in digitalphot  
        'Arguments for atheism range from philosophical to social and  
        'The Bible is a compilation of many shorter books written at  
        ]
```

```
test = tfidf.transform(docs)
```

```
lr.predict(test)
```

```
array([2, 1, 0, 3], dtype=int64)
```

```
news.target_names
```

```
['alt.atheism', 'comp.graphics', 'sci.med', 'soc.religion.christian']
```

Code
hw3-6.ipynb 참고
Result(Captured images)
<pre> &gt; ▶ M4 docs = ['The outbreak was declared a global pandemic by the World Health Organization (WHO) on 11 March.',  'Today, computer graphics is a core technology in digitalphotography, film, video games, cell phone and computer displays,and many specialized applications.',  'Arguments for atheism range from philosophical to social and historical approaches.',  'The Bible is a compilation of many shorter books written at different times by a variety of authors, and later assembled into the biblical canon.' ]  test = tfidf.transform(docs) lr.predict(test)  array([2, 1, 0, 3], dtype=int64) </pre>

<pre>[18]</pre>	<pre>from sklearn.tree import DecisionTreeClassifier tree = DecisionTreeClassifier() tree.fit(X_train_vector, y_train)  DecisionTreeClassifier()  [19] tree.score(X_train_vector, y_train)  1.0  [22] tree.score(X_test_vector, y_test)  0.8362831858407079  [26] tree.predict(test)  array([3, 1, 0, 2], dtype=int64)</pre>
	Description
	<p>X_train, X_test, y_train, y_test 를 생성하고 re 모듈을 사용해 preprocessor 함수를 정의하였다. 그리고 nltk.stem.porter 에서 PorterStemmer 를 이용해 tokenizer_porter 함수도 정의하였고 nltk 모듈을 활용해 stopword 또한 생성하였다</p> <p>그 후, TfidfVectorizer 객체를 생성하여 X_train, X_test 를 transform 하였다. transform 된 X_train_vector 를 기반으로 로지스틱회귀모델과 결정트리 모델을 학습하고 이 모델로 docs 에 대한 score 를 측정하였다</p>

## Note

1. Summit the file to e-class as pdf.
2. Specify your file name as “hw3\_<StudentID>\_<Name>.pdf”