과제 보고서

(R-Py Computing Homework 2)

|  |  |
| --- | --- |
| **과목명** | AI+X R-PY 컴퓨팅(AIX0004) |
| **담당 교수님** | 이정환 교수님 |
| **제출일** | 2019년 11월 23일(토요일) |
| **소속** | 한양대학교 공과대학 |
| 컴퓨터소프트웨어학부 |
| **학번** | **이름** |
| 2019009261 | 최가온(CHOI GA ON) |

-목차-

**Ⅰ. Part 1: 와인 클래스에 대한 kNN 알고리즘 적용**

1. **문제 1** - csv 파일 열기

2. **문제 2** - describe() 메소드를 통한 요약 통계량 구하기

3. **문제 3** - Data Split(Train / Test)

4. **문제 4** - KNeighborsClassifier과 모형 트레이닝

5. **문제 5** - 트레이닝된 모형을 바탕으로 Training Set 예측

6. **문제 6** - 트레이닝된 모형을 바탕으로 Test Set 예측

7. **문제 7** - k값을 바꾸어 재예측

8. **문제 8** - 데이터를 바꾸어 모델 재설계 및 재예측

9. 전체 코드

**Ⅱ. Part 2: 새 알고리즘을 적용해야 할 상황**

1. **문제 1** - importing SVC from sklearn.svm

2. **문제 2** - SVC 인자

3. **문제 3** - Data Split(Train / Test)

4. **문제 4** - 트레이닝된 모형을 바탕으로 Training Set 예측

5. **문제 5** - 트레이닝된 모형을 바탕으로 Test Set 예측

6. 전체 코드

**Ⅲ. Part 3: K-means Clustering 알고리즘**

1. **문제 1** - 데이터 프레임 Slicing

2. **문제 2** - K-means Clustering 알고리즘 적용

3. **문제 3** - 라벨 예측 및 Crosstab 명령어를 이용한 라벨값 비교

4. **문제 4** - k=1부터 9까지 변형시키면서 각 모형의 inertia 구하기

5. 전체 코드

Ⅰ. Part 1: 와인 클래스에 대한 kNN 알고리즘 적용

다음의 라이브러리를 사용하여 과제를 수행하였다.

**import** pandas **as** pd  
**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn.neighbors **import** KNeighborsClassifier

1. **문제 1** – csv 파일 열기

path = **"C:\\Users\\choig\\Downloads\\wine\_data.csv"**wine = pd.read\_csv(path)

2. **문제 2** – describe() 메소드를 통한 요약 통계량 구하기

print(wine.describe())

3. **문제 3** – Data Split (Train / Test)

X = wine.iloc[:, 1:]  
y = wine[[**"Class"**]]  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=1 - 0.7, random\_state=0)

X는 Class를 제외한 모든 변수이고, y는 Class 변수이다. Training Set의 비율이 전체의 70%가 되도록 Data Split을 하였다.

4. **문제 4** – KNeighborsClassifier과 모형 트레이닝

knn = KNeighborsClassifier(n\_neighbors=5)  
knn.fit(X\_train, y\_train.values.ravel())

k=5인 KNeighborsClassifier 객체인 knn을 만든다. 이후 Training Set을 학습시키기 위해 knn을 fitting하였다.

5. **문제 5** – 트레이닝된 모형을 바탕으로 Training Set 예측

prediction = knn.predict(X\_train)  
y\_train\_ = y\_train.values.ravel()

show\_accuracy(X\_train, y\_train\_, prediction)

6. **문제 6** – 트레이닝된 모형을 바탕으로 Test Set 예측

prediction2 = knn.predict(X\_test)  
y\_test\_ = y\_test.values.ravel()

show\_accuracy(X\_test, y\_test\_, prediction2)

7. **문제 7** – k값을 바꾸어 재예측

knn2 = KNeighborsClassifier(n\_neighbors=3)  
knn2.fit(X\_train, y\_train.values.ravel())

prediction3 = knn2.predict(X\_train)

show\_accuracy(X\_train, y\_train\_, prediction3)

prediction4 = knn2.predict(X\_test)

show\_accuracy(X\_test, y\_test\_, prediction4)

k값을 3으로 하여 새로운 KNeighborsClassifier 객체 knn2을 만들었다. knn과 마찬가지로 Training Set으로 학습을 시켰으며, X\_train을 예측한 결과를 prediction3에 저장하였다.

8. **문제 8** – 데이터를 바꾸어 모델 재설계 및 재예측

X2 = wine.iloc[:, 1:5]  
 *# y는 위에서 만들어진 y를 그대로 사용함.*X2\_train, X2\_test = train\_test\_split(X2, test\_size=1 - 0.7, random\_state=0)

knn3 = KNeighborsClassifier(n\_neighbors=5)  
knn3.fit(X2\_train, y\_train.values.ravel())

prediction5 = knn3.predict(X2\_train)  
show\_accuracy(X2\_train, y\_train\_, prediction5)  
  
prediction6 = knn3.predict(X2\_test)  
show\_accuracy(X2\_test, y\_test\_, prediction6)

X2는 wine 데이터프레임에서 "Alcohol", "Malic acid", "Ash", "Alcalinity of ash"이다. 마찬가지로 Training Set의 비율이 전체의 70%가 되도록 Data Split을 하였으며, k=5인 KNeighborsClassifier 객체 knn3을 만들었다.

prediction5, prediction6은 각각 Training Set과 Test Set을 예측한 것의 결과값이며 각각에 대해 정확도를 나타내었다. 정확도를 나타내는 것은 이번 과제에서 반복되는 행위이므로 이에 대한 함수를 따로 정의해두었다. 그 내용은 다음과 같다.

**def** show\_accuracy(X, Y, prediction):  
 num = 0  
 **for** x **in** range(len(X)):  
 **if** (prediction[x] == Y[x]):  
 num += 1  
 print(**"\tAccuracy: "**, (num / len(X)) \* 100, **"%"**)

9. 전체 코드

|  |
| --- |
| **import** pandas **as** pd **from** sklearn.model\_selection **import** train\_test\_split **from** sklearn.neighbors **import** KNeighborsClassifier   **def** show\_accuracy(X, Y, prediction):  num = 0  **for** x **in** range(len(X)):  **if** (prediction[x] == Y[x]):  num += 1  print(**"\tAccuracy: "**, (num / len(X)) \* 100, **"%"**)   *# [Q1-1]* print(**"(Q1-1)"**) path = **"C:\\Users\\choig\\Downloads\\wine\_data.csv"** wine = pd.read\_csv(path) print(**"opened CSV file successfully.\n"**)  *# [Q1-2]* print(**"(Q1-2)"**) print(wine.describe()) print()  *# [Q1-3]* print(**"(Q1-3)"**) print(**"data split into train / test"**) X = wine.iloc[:, 1:] y = wine[[**"Class"**]] X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=1 - 0.7, random\_state=0) print()  *# [Q1-4]* print(**"(Q1-4)"**) print(**"KNeighborsClassifier object knn(k = 5) and fitting"**) knn = KNeighborsClassifier(n\_neighbors=5) knn.fit(X\_train, y\_train.values.ravel()) print()  *# [Q1-5]* print(**"(Q1-5)"**) print(**"prediction of X\_train"**) prediction = knn.predict(X\_train) y\_train\_ = y\_train.values.ravel() print(**"training set 예측에 대한 정확도"**) show\_accuracy(X\_train, y\_train\_, prediction) print()  *# [Q1-6]* print(**"(Q1-6)"**) print(**"prediction of X\_test"**) prediction2 = knn.predict(X\_test) y\_test\_ = y\_test.values.ravel() print(**"test set 예측에 대한 정확도"**) show\_accuracy(X\_test, y\_test\_, prediction2) print()  *# [Q1-7]* print(**"(Q1-7)"**) print(**"KNeighborsClassifier object knn2(k = 3) and fitting"**) knn2 = KNeighborsClassifier(n\_neighbors=3) knn2.fit(X\_train, y\_train.values.ravel()) prediction3 = knn2.predict(X\_train) print(**"training set 예측에 대한 정확도"**) show\_accuracy(X\_train, y\_train\_, prediction3)  prediction4 = knn2.predict(X\_test) print(**"test set 예측에 대한 정확도"**) show\_accuracy(X\_test, y\_test\_, prediction4) print()  *# [Q1-8]* print(**"(Q1-8)"**) print(**"X = Alcohol, Malic acid, Ash, Alcalinity of ash"**) X2 = wine.iloc[:, 1:5]  *# y는 위에서 만들어진 y를 그대로 사용함.* X2\_train, X2\_test = train\_test\_split(X2, test\_size=1 - 0.7, random\_state=0) print(**"KNeighborsClassifier object knn3(k = 5) and fitting"**) knn3 = KNeighborsClassifier(n\_neighbors=5) knn3.fit(X2\_train, y\_train.values.ravel())  prediction5 = knn3.predict(X2\_train) print(**"train 예측에 대한 정확도"**) show\_accuracy(X2\_train, y\_train\_, prediction5)  prediction6 = knn3.predict(X2\_test) print(**"test set 예측에 대한 정확도"**) show\_accuracy(X2\_test, y\_test\_, prediction6) |

**(실행결과)**

|  |
| --- |
| C:\Users\choig\Anaconda3\envs\pytorch\python.exe C:/Users/choig/PycharmProjects/끄쩎끄쩎/HAI.py  (Q1-1)  opened CSV file successfully.  (Q1-2)  Class Alcohol ... OD280/OD315 of diluted wines Proline  count 178.000000 178.000000 ... 178.000000 178.000000  mean 1.938202 13.000618 ... 2.611685 746.893258  std 0.775035 0.811827 ... 0.709990 314.907474  min 1.000000 11.030000 ... 1.270000 278.000000  25% 1.000000 12.362500 ... 1.937500 500.500000  50% 2.000000 13.050000 ... 2.780000 673.500000  75% 3.000000 13.677500 ... 3.170000 985.000000  max 3.000000 14.830000 ... 4.000000 1680.000000  [8 rows x 14 columns]  (Q1-3)  data split into train / test  (Q1-4)  KNeighborsClassifier object knn(k = 5) and fitting  (Q1-5)  prediction of X\_train  training set 예측에 대한 정확도  Accuracy: 79.03225806451613 %  (Q1-6)  prediction of X\_test  test set 예측에 대한 정확도  Accuracy: 72.22222222222221 %  (Q1-7)  KNeighborsClassifier object knn2(k = 3) and fitting  training set 예측에 대한 정확도  Accuracy: 90.32258064516128 %  test set 예측에 대한 정확도  Accuracy: 70.37037037037037 %  (Q1-8)  X = Alcohol, Malic acid, Ash, Alcalinity of ash  KNeighborsClassifier object knn3(k = 5) and fitting  train 예측에 대한 정확도  Accuracy: 81.45161290322581 %  test set 예측에 대한 정확도  Accuracy: 83.33333333333334 %  Process finished with exit code 0 |

Ⅱ. Part 2: 새 알고리즘을 적용해야 할 상황

다음의 라이브러리를 사용하여 과제를 수행하였다.

**import** pandas **as** pd  
**from** sklearn.model\_selection **import** train\_test\_split

1. **문제 1** – importing SVC from sklearn.svm

**from** sklearn.svm **import** SVC

2. **문제 2** – SVC 인자

svc = SVC(kernel=**'linear'**, C=1.0, gamma=**'auto'**)

3. **문제 3** – Data Split (Train / Test)

path = **"C:\\Users\\choig\\Downloads\\wine\_data.csv"**wine = pd.read\_csv(path)  
*# Data Slicing*X = wine.iloc[:, 1:]  
y = wine[[**"Class"**]]

*# Data Split*  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size=1 - 0.3, random\_state=0)  
y\_train\_=y\_train.values.ravel()  
y\_test\_=y\_test.values.ravel()  
  
svc.fit(X\_train, y\_train\_)

4. **문제 4** – 트레이닝된 모형을 바탕으로 Training Set 예측

prediction1 = svc.predict(X\_train)  
show\_accuracy(X\_train, y\_train\_, prediction1)

5. **문제 5** – 트레이닝된 모형을 바탕으로 Test Set 예측

prediction2 = svc.predict(X\_test)  
show\_accuracy(X\_test, y\_test\_, prediction2)

6. 전체 코드

여기에 「Part 2: 새 알고리즘을 적용해야 할 상황」에 대해 작성된 코드의 전체 부분을 첨부한다. 또한 그 코드를 실행했을 때의 결과를 함께 보인다.

|  |
| --- |
| **import** pandas **as** pd **from** sklearn.model\_selection **import** train\_test\_split   **def** show\_accuracy(X, Y, prediction):  num = 0  **for** x **in** range(len(X)):  **if** (prediction[x] == Y[x]):  num += 1  print(**"\tAccuracy: "**, (num / len(X)) \* 100, **"%"**)   *# [Q2-1]* print(**"(Q2-1)"**) **from** sklearn.svm **import** SVC  print(**"Import SVC(Support Vector Machine)\n"**)  *# [Q2-2]* print(**"(Q2-2)"**) print(**"SVC object"**) svc = SVC(kernel=**'linear'**, C=1.0, gamma=**'auto'**) print()  *# [Q2-3]* print(**"(Q2-3)"**) print(**"Reading csv file..."**) path = **"C:\\Users\\choig\\Downloads\\wine\_data.csv"** wine = pd.read\_csv(path) *# Data Slicing* X = wine.iloc[:, 1:] y = wine[[**"Class"**]] *# Data Split* print(**"Data Split into train / test"**) X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size=1 - 0.3, random\_state=0) y\_train\_=y\_train.values.ravel() y\_test\_=y\_test.values.ravel() print(**"Fitting"**) svc.fit(X\_train, y\_train\_) print()  *# [Q2-4]* print(**"(Q2-4)"**) print(**"prediction1 : X\_train"**) prediction1=svc.predict(X\_train) show\_accuracy(X\_train, y\_train\_, prediction1) print()  *# [Q2-5]* print(**"(Q2-5)"**) print(**"prediction2 : X\_test"**) prediction2=svc.predict(X\_test) show\_accuracy(X\_test, y\_test\_, prediction2) print() |

**(실행결과)**

|  |
| --- |
| C:\Users\choig\Anaconda3\envs\pytorch\python.exe C:/Users/choig/PycharmProjects/끄쩎끄쩎/HAI.py  (Q2-1)  Import SVC(Support Vector Machine)  (Q2-2)  SVC object  (Q2-3)  Reading csv file...  Data Split into train / test  Fitting  (Q2-4)  prediction1 : X\_train  Accuracy: 99.19354838709677 %  (Q2-5)  prediction2 : X\_test  Accuracy: 98.14814814814815 %  Process finished with exit code 0 |

Ⅲ. Part 3: K-means Clustering 알고리즘

다음의 라이브러리를 사용하여 과제를 수행하였다.

**import** pandas **as** pd  
**from** sklearn.cluster **import** KMeans

1. **문제 1** – 데이터 프레임 Slicing

path = **"C:\\Users\\choig\\Downloads\\wine\_data.csv"**wine = pd.read\_csv(path)

X = wine.iloc[:, 1:]  
y = wine[[**'Class'**]]

2. **문제 2** – K-means Clustering 알고리즘 적용

model = KMeans(n\_clusters=3)

model.fit(X)

3. **문제 3** – 라벨 예측 및 Crosstab 명령어를 이용한 라벨값 비교

prediction = model.predict(X)  
print(**"CrossTab : (prediction) \* (real label)"**)  
ct = pd.crosstab(prediction, y.values.ravel())  
print(ct)

4. **문제 4** – k=1부터 9까지 변형시키면서 각 모형의 intertia 구하기

ks = range(1, 10)  
inertias = []  
  
**for** k **in** ks:  
 model2 = KMeans(n\_clusters=k)  
 model2.fit(X)  
 inertias.append(model2.inertia\_)  
print(**"k = i --> (inertia when k = i)"**)  
**for** k **in** ks:  
 print(**"k ="**, k, **"-->"**, inertias[k - 1])

5. 전체 코드

|  |
| --- |
| **import** pandas **as** pd **from** sklearn.cluster **import** KMeans  *# [Q3-1]* print(**"(Q3-1)"**) print(**"opened CSV file successfully."**) path = **"C:\\Users\\choig\\Downloads\\wine\_data.csv"** wine = pd.read\_csv(path)  print(**"Data Setting"**) X = wine.iloc[:, 1:] y = wine[[**'Class'**]] print()  *# [Q3-2]* print(**"(Q3-2)"**) print(**"KMeans Clustering Model (n\_clusters=3)"**) model = KMeans(n\_clusters=3) print(**"Model Fitting"**) model.fit(X) print()  *# [Q3-3]* print(**"(Q3-3)"**) print(**"Prediction : X"**) prediction = model.predict(X) print(**"CrossTab : (prediction) \* (real label)"**) ct = pd.crosstab(prediction, y.values.ravel()) print(ct) print()  *# [Q3-4]* print(**"(Q3-4)"**) ks = range(1, 10) inertias = []  **for** k **in** ks:  model2 = KMeans(n\_clusters=k)  model2.fit(X)  inertias.append(model2.inertia\_) print(**"k = i --> (inertia when k = i)"**) **for** k **in** ks:  print(**"k ="**, k, **"-->"**, inertias[k - 1]) |

(실행결과)

|  |
| --- |
| C:\Users\choig\Anaconda3\envs\pytorch\python.exe C:/Users/choig/PycharmProjects/끄쩎끄쩎/HAI.py |
| (Q3-1) |
| opened CSV file successfully. |
| Data Setting |
|  |
| (Q3-2) |
| KMeans Clustering Model (n\_clusters=3) |
| Model Fitting |
|  |
| (Q3-3) |
| Prediction : X |
| CrossTab : (prediction) \* (real label) |
| col\_0 1 2 3 |
| row\_0 |
| 0 46 1 0 |
| 1 0 50 19 |
| 2 13 20 29 |
|  |
| (Q3-4) |
| k = i --> (inertia when k = i) |
| k = 1 --> 17592296.383508474 |
| k = 2 --> 4543749.614531862 |
| k = 3 --> 2370689.686782968 |
| k = 4 --> 1333139.2086165315 |
| k = 5 --> 916379.187153917 |
| k = 6 --> 647326.0020260847 |
| k = 7 --> 412137.50910045847 |
| k = 8 --> 326221.8829475143 |
| k = 9 --> 270317.4625062386 |
|  |
| Process finished with exit code 0 |