2023 Ver.

데이터 분석 언어 개론

생성형 AI - 랭체인 (LLM) 활용

백엔드(풀스택) 엔지니어(자바,파이썬) 양성과정

문유정

010-8766-5119

zeein119@gmail.com

CHAP 1

언어 1 : 변수/자료구조

Python

1. 문자, 숫자, 언더바(_)
2. Bool : True / False

   숫자 : int, float

   문자 : " " , ' '
3. 리스트 : [원소1, 원소2, ...]

   데이터프레임 : pandas.DataFrame
4. 인덱스 : 0부터 시작 / 종료 미포함 (종료-1)

R

1. 문자, 숫자, 언더바(_), 점(.)

   백틱(`) 사용시 숫자 시작 가능
2. Bool : TRUE (T) / FALSE (F)

   숫자 : numeric

   문자 : " " , ' '
3. 벡터 : c(원소1, 원소2, ...)

   리스트 : list(...) 파이썬 딕셔너리와 비교

   데이터프레임 : data.frame
4. 인덱스 : 1부터 시작 / 종료 포함

CHAP 1

언어 2 : 연산자

Python

1. 산술 연산자 : + , - , * , /

   ** (거듭제곱)

   // (몫)

   % (나머지)

2. 할당 연산자 : = , += , -= , *= , /= , **= , //= , %=

3. 논리 연산자 and , or , not

   * 비트 연산자 & , | , ~

   * 삼항 연산

     변수 = 참일때 값 if 조건 else 거짓일때 값

R

1. 산술 연산자 : + , - , * , /

   **, ^ (거듭제곱)

   %/% (몫)

   %% (나머지)

2. 할당 연산자 : <- , = , ->

3. 논리 연산자 & , | , !

   * 삼항 연산

     변수 <- ifelse(조건, 참인경우, 거짓인 경우)

CHAP 1

언어 3 : 조건문

Python

```
import datetime
now = datetime.datetime.now()
print(now)
print("현재는 ",now.year,"년",now.month,"월",now.day,"일 입니다.")
if 3 <= now.month <= 5:
    print("이번달은 ", now.month, "월로 봄입니다.")
elif 6 <= now.month <= 8:
    print("이번달은 ", now.month, "월로 여름입니다.")
elif 9 <= now.month <= 11:
    print("이번달은 ", now.month, "월로 가을입니다.")
else:
    print("이번달은 ", now.month, "월로 겨울입니다.")
```

R

```
student.score ← 100
student.grade ← 'F'
if(student.score>=90){
   student.grade ← 'A'
} else if(student.score >=80) {
  student.grade ← 'B'
} else {
  student.grade ← 'C'
}
print(student.grade)
```

CHAP 1

언어 4 : 반복문

Python

```
math_score = [88, 100, 70, 110, 77, 89]
for score in math_score:
    if(score < 0 or score > 100):
        continue
    print(score)

math_score = [88, 100, 70, 110, 77, 89]
for score in math_score:
    if(score < 0 or score > 100):
        print("잘못된 숫자가 입력되어 있습니다.")
        break
    print(score)
```

R

```
math_score ← c(88, 100, 70, 110, 77, 89)
for(score in math_score) {
    if( (score < 0) | (score > 100) ) next
    print(score)
}
math_score ← c(88, 100, 70, 110, 77, 89)
for(score in math_score) {
    if( (score < 0) | (score > 100)) {
        print("잘못된 숫자가 입력되어 있습니다.")
        break
    }
    print(score)
}
```

CHAP 1

언어 4 : 반복문

Python

```
sum = 0
i = 0
while i < 101:
    sum += i
    i += 1
print("1~100 까지의 합은 ", sum)
```

R

```
sum ← 0
i ← 0
while(i <= 100) {
  sum ← sum + i
  i ← i + 1
}
cat("1~100 까지의 합은 ", sum)
```

CHAP 1

언어 4 : 반복문

Python

```
sum = 0
i = 0
while True:
    if(i>100):
        break
    sum += i
    i += 1
print("1~100 까지의 합은 ", sum)
```

R

```
i ← 0
sum ← 0
repeat {
 if(i > 100){
   break
 }else{
   sum ← sum + i
 }
 i ← i + 1
}
cat("1~100 까지의 합은 ", sum)
```

CHAP 1

언어 5 : 함수

Python

```
def data_sum(x,y):
    idata_sum = x+y
    idata_mul = x*y
    return (idata_sum, idata_mul)

sum, mul = data_sum(3,5)
print(sum,mul)

# 가변인수      *매개변수 : 리스트 처리      **매개변수 : 딕셔너리 처리
```

R

```
data_sum ← function(x,y){
  idata.sum ← x+y
  idata.mul ← x*y
  return(list(sum=idata.sum, mul=idata.mul))
}

maindata ← data_sum(3,5)
maindata$sum
maindata$mul
```

CHAP 2

CSV 파일읽기

Python

```
import pandas as pd
# CSV 파일을 불러오기
data = pd.read_csv('data.csv')
# CSV 파일을 CP949 인코딩으로 불러오기
data = pd.read_csv('data.csv', encoding='CP949')
# 데이터 프레임 출력
data.head()
```

R

```
# CSV 파일을 불러오기
data <- read.csv('data.csv')
# CSV 파일을 CP949 인코딩으로 불러오기
data <- read.csv('data.csv', encoding='CP949')
# 데이터 프레임 출력
head(data)
```

CHAP 2

[데이터탐색[기본정보]

Python

```
# 데이터프레임의 구조 확인
data.shape
# 열의 이름 확인
data.columns
# 데이터의 요약 정보 확인
data.info()
# 데이터 기초 통계량 확인
data.describe()
# 컬럼 데이터 빈도확인
data['column1'].value_counts()
```

R

```
# 데이터프레임의 구조 확인
dim(data)
# 열의 이름 확인
colnames(data)
# 데이터의 요약 정보 확인
str(data)
# 데이터 기초 통계량 확인
summary(data)
# 컬럼 데이터 빈도확인
table(data$column1)
```

CHAP 2

[데이터탐색 [행추출]

Python

```
# 특정 조건을 만족하는 행 필터링
filtered_data = data[data['Age'] > 30]
# 필터링된 데이터 출력
print(filtered_data.head())
```

R

```
# 특정 조건을 만족하는 행 필터링
filtered_data <- data[data$Age > 30, ]
# 필터링된 데이터 출력
head(filtered_data)
```

CHAP 2

데이터탐색 [열추출]

Python

```
# 특정 열 선택
selected_columns = data[['Name', 'Age', 'Gender']]
# 선택된 열 출력
print(selected_columns.head())
```

R

```
# 특정 열 선택
selected_columns <- data[ ,c('Name', 'Age', 'Gender')]
# 선택된 열 출력
head(selected_columns)
```

CHAP 2

데이터전처리 [결측치]

Python

```
# 결측치 확인
missing_values = data.isnull().sum()
# 결측치가 있는 행 제거
data_cleaned = data.dropna()
# 결측치 대체
data_imputed = data.fillna(0) # 0으로 대체 (다른 값으로 대체 가능)


data_imputed.head()
```

R

```
# 결측치 확인
missing_values <- colSums(is.na(data))
# 결측치가 있는 행 제거
data_cleaned <- na.omit(data)
# 결측치 대체
data_imputed <- replace(data, is.na(data), 0) # 0으로 대체 (다른 값으로 대체 가능)


head(data_imputed)
```

CHAP 2

데이터전처리 [이상치]

Python

```
import numpy as np
# 이상치 확인 (예: 표준편차 기반)
mean = np.mean(data['Value'])
std_dev = np.std(data['Value'])
threshold = 2 # 예: 2 표준편차 범위 이상은 이상치로 간주
outliers = data[(data['Value'] < mean - threshold * std_dev) |
                (data['Value'] > mean + threshold * std_dev)]
# 이상치 제거
data_cleaned = data[~data.index.isin(outliers.index)]
```

R

```
# 이상치 확인 (예: 표준편차 기반)
mean_val <- mean(data$Value)
std_dev <- sd(data$Value)
threshold <- 2 # 예: 2 표준편차 범위 이상은 이상치로 간주
outliers <- data[data$Value < mean_val - threshold * std_dev |
                 data$Value > mean_val + threshold * std_dev, ]
# 이상치 제거
data_cleaned <- data[!rownames(data) %in% rownames(outliers),
]
```

CHAP 2

데이터전처리 [이상치]

Python

```
# 이상치 대체 (예: 중앙값으로 대체)
median = np.median(data['Value'])
data_imputed = data.copy()
data_imputed.loc[outliers.index, 'Value'] = median

data_imputed.head()
```

R

```
# 이상치 대체 (예: 중앙값으로 대체)
median_val <- median(data$Value)
data_imputed <- data
data_imputed[data$Value < mean_val - threshold * std_dev |
             data$Value > mean_val + threshold * std_dev, 'Value']
<- median_val

head(data_imputed)
```

CHAP 2

데이터전처리 [정규화]

Python

```
from sklearn.preprocessing import MinMaxScaler
# 정규화를 위한 Min-Max 스케일러 생성
scaler = MinMaxScaler()
# 특정 열을 선택 (예: 'Value' 열)
values = data[['Value']]
# 데이터 정규화
normalized_values = scaler.fit_transform(values)
# 정규화된 데이터를 데이터프레임으로 변환
normalized_data = pd.DataFrame(normalized_values, columns=
['Normalized_Value'])
normalized_data.head()
```

R

```
# 특정 열 선택 (예: 'Value' 열)
values <- data$Value
# 데이터 정규화 (0과 1 사이의 값으로 스케일링)
normalized_values <- (values - min(values)) / (max(values) -
min(values))
# 정규화된 데이터를 데이터프레임으로 변환
normalized_data <- data.frame(Normalized_Value =
normalized_values)


head(normalized_data)
```

CHAP 2

데이터전처리 [표준화]

Python

```
from sklearn.preprocessing import StandardScaler
# 표준화를 위한 스케일러 생성
scaler = StandardScaler()
# 특정 열을 선택 (예: 'Value' 열)
values = data[['Value']]
# 데이터 표준화
standardized_values = scaler.fit_transform(values)
# 표준화된 데이터를 데이터프레임으로 변환
standardized_data = pd.DataFrame(standardized_values,
columns=['Standardized_Value'])
standardized_data.head()
```

R

```
# 특정 열 선택 (예: 'Value' 열)
values <- data$Value
# 데이터 표준화 (평균이 0, 표준편차가 1이 되도록 스케일링)
standardized_values <- scale(values)
# 표준화된 데이터를 데이터프레임으로 변환
standardized_data <- data.frame(Standardized_Value =
standardized_values)


head(standardized_data)
```

CHAP 2

[데이터전처리 [상관분석]

Python

```
import pandas as pd
# 예시 데이터 생성
data = pd.DataFrame({'X': [1, 2, 3, 4, 5], 'Y': [2, 3, 5, 4, 6]})
# 공분산 계산
covariance = data['X'].cov(data['Y'])
# 상관계수 계산
correlation = data['X'].corr(data['Y'])
# 결과 출력
print("공분산:", covariance)
print("상관계수:", correlation)
```

R

```
# 예시 데이터 생성
data <- data.frame(X = c(1, 2, 3, 4, 5), Y = c(2, 3, 5, 4, 6))
# 공분산 계산
covariance <- cov(data$X, data$Y)
# 상관계수 계산
correlation <- cor(data$X, data$Y)
# 결과 출력
cat("공분산:", covariance, "\n")
cat("상관계수:", correlation, "\n")
```

CHAP 2

시각화[히스토그램/박스]

Python

```
import pandas as pd
import matplotlib.pyplot as plt

# 한글 깨지는 것 방지
plt.rcParams["font.family"] = 'Malgun Gothic'
iris = pd.read_csv("iris.csv")
iris['Sepal.Length'].plot(kind='hist')
plt.show()
iris['Sepal.Length'].plot(kind='box')
plt.show()
iris.plot(kind='hist')
iris.plot(kind='box')
```

R

```
hist(iris$Sepal.Length)
hist(iris$Sepal.Width)
hist(iris$Petal.Length)
hist(iris$Petal.Width)
boxplot(iris)
boxplot(Sepal.Length~Sepal.Width, data = iris)
```

R 시각화 참조 : https://r-graphics.org https://r-graph-gallery.com/

CHAP 2

시각화[Plot/Line]

Python

```
import pandas as pd
import matplotlib.pyplot as plt
iris = pd.read_csv("iris.csv")
plt.plot(iris.index,'Sepal.Length',data=iris,linestyle='none',marker='o',markersize=5,color='blue',alpha=0.5)
plt.plot(iris.index,'Sepal.Width',data=iris,linestyle='none',marker='o',markersize=5,color='blue',alpha=0.5)
plt.plot(iris_df.index,'Petal.Length',data=iris_df,linestyle='solid',lw=3,color='green',alpha=0.5)
plt.plot(iris.index,'Petal.Width',data=iris,linestyle='none',marker='o',markersize=5,color='blue',alpha=0.5)
plt.plot(iris.index,'Species',data=iris,linestyle='none',marker='p',markersize=5,color='blue',alpha=0.5)
```

R

```
head(iris,3)


plot(x = iris$Sepal.Length, type = "p", ylim=c(0, 10), col=1)
lines(iris$Sepal.Width, type = "p", ylim=c(0, 10), col=2)
lines(iris$Petal.Length, type = "l", ylim=c(0, 10), col=3)
lines(iris$Petal.Width, type = "o", ylim=c(0, 10), col=4)
lines(iris$Species, type = "b", ylim=c(0, 10), col=5)
legend("topleft", colnames(iris), pch=1, col=1:5)
```

CHAP 2

시각화[Bar]

Python

```
import pandas as pd
import matplotlib.pyplot as plt
iris = pd.read_csv("iris.csv")
iris['Sepal.Length'].plot(kind='bar')
iris.plot.bar(x='Sepal.Length',y='Sepal.Width')
```

R

```
install.packages("ggplot2")
library(ggplot2)

ggplot(iris)+geom_bar(aes(x=Sepal.Length))
ggplot(iris)+geom_bar(aes(x=Sepal.Length, y=Sepal.Width))
ggplot(iris)+geom_col(aes(x=Sepal.Length, y=Sepal.Width))
```

CHAP 2

시각화[산점도행렬]

Python

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

iris = pd.read_csv("iris.csv")
sns.pairplot(iris)

sns.scatterplot(x='Sepal.Length', y='Sepal.Width', data=iris,
hue='Species', style='Species')
```

R

```
pairs(iris[ ,1:4])

ggplot(iris)+geom_point(aes(x=Sepal.Length,y=Sepal.Width))
ggplot(iris)+geom_point(aes(x=Sepal.Length,y=Sepal.Width,
col=Species))
ggplot(iris)+geom_point(aes(x=Sepal.Length,y=Sepal.Width,
alpha=Species))
ggplot(iris)+geom_point(aes(x=Sepal.Length,y=Sepal.Width,
shape=Species))
```