**02**

**데이터 핸들링 (1)**

1

**DataFrame 기본**

Pd.read\_csv(path, sep, sheet\_name, index\_col)

Pd.DataFrame(dict)

Pd.DataFrame(list or ndarray, columns=[ ])

Df.copy()

Df.index.values

Df.columns

Df.values

Df.reset\_index(inplace,drop)

2

**행, 열 선택/ 추가/ 삭제 (pandas)**

df[index:]

df[col]

df[‘new\_col’] = df[‘col1’] \* 100 + df[‘col2’]

df[‘new\_col’] = df[‘col1].apply(lambda x : ‘a’ if x > n else ‘b’)

df.drop(labels, inplace, axis)

df.dropna()

3

**조건에 따른 행, 열 선택 (loc,iloc)**

df.loc[:, :]

df.iloc[:, :]

df[df[‘col1’] >= n]

df.fillna(values, inplace=False)

4

**데이터 정렬 (sort\_value)**

df.sort\_values(by, ascending, inplace)

5

**데이터 결합 (concat, merge)**

Pd.concat([df1,df2], axis, join = ‘outer’ or ‘inner’, ignore\_index, keys)

Pd.merge(df1, df2, on=’col’, how=’outer’ or ‘inner’ or ‘right’, or ’left’)

Pd.merge(df1, df2, left\_on = ‘lkey’, right\_on = ‘rkey’)

Df1.join(df2, how)

**03**

**데이터 핸들링 (2)**

1

**데이터 요약 (groupby)**

Df.groupby(by=’col’).agg()

Df.groupby(by=’col1’)[‘col2’].agg([func1,func2])

Df.groupby([‘col1’,’col2’]).agg()

Df.groupby(by=’col1’).agg({‘Age’:’max’,’Fare’:’mean’})

2

**데이터 모양 변경 (pivot\_table)**

pd.pivot\_table(data, index, columns, values, aggfunc, fill\_value, margins, margins\_name)

df.pivot\_table(values, index, columns, aggfunc=’count’)

df.pivot(index, columns, values)

3

**문자열 데이터 변환하기 (str)**

str.split(sep, maxsplit)

‘-‘.join(list)

str.strip(value)

str.find(value)

str.count(value)

str.replace(old\_value, new\_value)

4

**데이터프레임에 함수 적용하기 (apply, map, lambda)**

Func = lambda x : x\*\*2

Map(lambda x : x\*\*2, list)

Df[‘new\_col’] = Df[‘col’].apply(lambda x : x\*\*2)

Df[‘new\_col’] = df[‘col’].apply(lambda x : ‘a’ if x > n else ‘b’)

5

날짜 데이터 변환하기 (datetime)

time= datetime.datetime(year,month,day,hour,minute,second)

time\_diff = day2 – day1

time\_diff.days

datetime.date.today()

datetime.datetime.now()

**04**

**데이터 전처리**

1

**데이터 분할 기법**

단순랜덤(shuffle=True) & 층화추출법(stratify)

From sklearn.model\_selection import train\_test\_split

X = df[df.columns.difference([‘target’])]

Y = df[‘target’]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size, test\_size, shuffle, stratify=df[‘col’])

Df.sample(n, replace)

From sklearn.model\_selection import Kfold

Kf = Kfold(n\_splits=4, shuffle=False)

Kf.get\_n\_splits(X)

For train\_index, test\_index in kf.split(X):

2

**데이터 표준화, 정규화**

From sklearn.preprocessing import MinMaxScaler, StandardScaler

Norm = MinMaxScaler()

Df\_\_norm = norm.fit\_transform(df)

Scaler = StandardScaler()

Df\_scaled = scaler.fit\_transform(df)

3

**결측치 처리하기**

Df.dropna(axis, subset=[‘col1’,’col2’], how = ‘any’ or ‘all’, inplace)

Df.fillna(values, method=[‘ffill’,’bfill’], inplace)

Df.fillna({‘A’:0, “B”:1, “C”:2})

Df.interpolate(method = ‘linear’)

imputer = sklearn.impute.SimpleImputer( np.nan ,'mean')

imputer.fit\_transform(df)

imputer = KNNImputer(n\_neighbots=5)

imputer.fit\_transform(df)

최빈값 : df[‘col’].mode()[0]

4

**범주형 변수 변환하기**

X1 = df[‘col1’].replace([1,2],[‘male’,’female’])

X1\_dum = pd.get\_dummies(x1)

Data = pd.concat([xy,x1\_dum],axis=1)

X\_ohe = pd.get\_dummies(df, columns=[ ])

5

**변수 축소 (PCA)**

From sklearn.decomposition import PCA

Pca = PCA(n\_components = 2)

Pca\_df = Pca.fit\_transform(X)

Pca.explained\_variance\_ration\_

6

**이상치 처리**

Sns.boxplot(df\_Data)

Q25, q75 = np.quantile(date[column], 0.25), np.quantile(data[column], 0.75)

Iqr = q75-q25

Cut\_off = 1.5 \* iqr

Lower\_line, upper\_line = q25 – cut\_off, q75 + cut\_off

Df[‘col’]\_clip = df[‘col’].clip(lower\_line, upper\_line)

7

**변수 선택, 특성 선택**

From sklearn.feature\_selection import f\_regression, SelectKBest

Selector = SelectKBest(score\_func = ‘f\_classif’ or ‘f\_regression’, k=20)

Selector.fit\_transform(X\_train, y\_train)

8

**데이터 불균형 해결**

from imblearn.over\_sampling import SMOTE

sm = SMOTE(sampling\_strategy = ‘auto’, random\_state = 0)

x\_smote, y\_smote = sm.fit\_resample(x\_train, y\_train)

from imblearn.over\_sampling import RandomOverSampler

ros = RandomOverSampler()

X\_ros, y\_ros = ros.fit\_resample(X,y)

**05**

**EDA (Exploratory Data Analysis) & 시각화**

1

**산점도**

Plt.scatter(x,y,s,c,marker,alpha, cmap) : 군집화 결과

2

**점, 선 그래프**

Plt.plot(x, y, fmt=’[color][linestyle][marker]’) : 시계열 변화

3

**막대그래프, 히스토그램**

Plt.bar(x,height,width,color,label, align={‘center’,’edge’}, tick\_label) : 비포 애프터 비교

Plt.barh(x,height,width,color,label)

Plt.hist(list, bins) : 어떤 항목의 개수를 시각화 할 때

Plt.pie(x, labels, autopct=’%.1f%%’, startangle, counterclock, shadow, explode) : 전체 분포 고려

4

**히트맵**

Import seaborn as 눈

Corr = data.corr()

sns.heatmap(data.corr(), cmap, vmin, vmax, annot, annot\_kws, linewidths, linecolor)

cmap = [‘PuBu’, ‘plasma’]

plt.show()

sns.pairplot(data, hue, diag\_kind, kind, markers, ,height, x\_vars, y\_vars, corner=True,

plot\_kws, diag\_kws)

5

**profiling**

Boxplot : sns.boxplot(x, y, hue, data, palette= ‘Set3’ or ‘vlag’)

Subplot : Fig, ax = plt.subplot(nrow, ncol, figsize=())

ax[0,0].plot()

**01**

**통계분석 - (1) 통계분석이란?**

1

**통계 분석의 개념**

데이터 분석을 수행할 때, 전체 데이터를 모두 활용하는 것은 수많은 시간과 비용이 들 수 있음으로, 모집단에서 일부 표본을 추출하여 분석에 활용한다.

통계란 특정집단을 대상으로 수행한 조사나 실험을 통해 나온 결과에 대한 요약된 형태의 표현이며, 통계분석은 이 통계를 가지고 분석을 수행하는 작업이다.

2

**모집단과 표본집단의 개념**

모집단 : 조사하고자 하는 대상 집단 전체

표본집단 : 조사하기 위해 추출한 모집단의 일부 원소

**02**

**통계분석 - (2) T-test**

1

**차이와 분산에 대한 이해**

Deviation = x – np.mean(x

Variation = sum(deviation\*\*2) # 편차 제곱의 평균

from scipy import stats

stats.mode(x) | np.var(x) | np.std(x) | np.quantitle(x,0.25) | stats.skew(x) | stats.kutosis()

2

**통계 검정의 개념 이해**

모집단의 특성에 대한 통계적 가설을 모집단으로부터 추출한 표본을 통해 검토하는 추론.모수적 방법은 모집단의 평균 혹은 분산을 추정하는 방법이고, 비모수적 방법은 정상성을 만족하지 않을 때 사용하는 방법이다.(중위값 추정.)

3

**t-test란**

두 집단의 평균을 통계적으로 비교하기 위해 사용되는 검정방법으로, 귀무가설은 ‘두 집단의 평균이 같다’이다.

4

**t-test 실습**

from scipy import stats

**정상성 & 등분산성 검정**

stats.shipiro(x)

stats.kstest(x, alternative = ‘two-sided’, cdf =’norm’)

stats.normaltest(x)

stats.levene(a, b)

**일표본 t-검정**

stats.ttest\_1samp(x, popmean, alternative=’two-sided’)

stats.wilcoxon(x = data – popmean, y=None, zero\_method = ‘wilcox’, correction=False)

**대응표본 t-검정**

stats.ttest\_rel(a, b, alternative=’two-sided’)

stats.wilcoxon(x, y, zero\_method = ‘wilcox’, correction=False )

**독립표본 t-검정**

stats.ttest\_ind(a, b, alternative=’two-sided’, equal\_var = True)

stats.ttest\_ind(a, b, alternative=’two-sided’, equal\_var = False)

stats.mannwhitneyu(a, b, alternative=’two-sided’)

**03**

**통계분석 - (3) 분산분석**

1

**ANOVA test란?**

두 개 이상의 다수 집단 간 평균을 비교하는 통계분석 방법.

2

**F-value에 대한 이해**

F검정통계량은 두 그룹의 분산에 대한 비율이다.

3

**일원 배치 분산 분석 개념**

분산분석은 두 개 이상의 집단에서 그룹 평균 간 차이를 그룹 내 변동에 비교하여 살펴보는 분석방법으로 두 개 이상 집단들의 평균 간 차이에 대한 통계적 유의성을 검증하는 분석이다.

4

**일원 배치 분산분석 실습**

Stats.shapiro(a)

Stats.levene(a,b,c)

Stats.chi2\_contingency(crosstab)

Stats.f\_oneway(a,b,c)

Stats.kruskal(a,b,c)

import statsmodels.formula.api as smf

import statsmodels.api as sm

reg = Smf.ols(formula = ‘x ~ y’, data=df\_1way)

reg\_fit = reg.fit()

anova\_table = sm.stats.anova\_lm(reg\_fit, typ=2)

print(anova\_table)

from statsmodel.stats.multicomp import pairwise\_tukeyhsd

tukey = pairwise\_tukeyhsd(endog = x, groups = y)

print(tukey)

print(tukey.plot\_simultaneous())

5

**이원 배치 분산 분석 개념**

분산분석에서 반응값에 대해 두 개의 범주형 변수 A, B의 영향을 알아보기 위해 사용되는 검증 방법이다.

6

**이원 배치 분산분석 실습**

Stats.shapiro(x)

Stats.levene(a,b,c,d)

import statsmodels.formula.api as smf

import statsmodels.api as sm

Reg = smf.ols(formula = ‘y ~ x1\*x2’, data=df\_2way)

Reg\_fit = reg.fit()

Anova\_table = sm.stats.anova\_lm(reg\_fit, typ=2)

Print(anova\_table)

**04**

**통계분석 - (4) 교차분석**

1

**교차분석 개념**

교차분석이란 명목척도 혹은 순서척도 등 한 두 개의 범주형 자료들 간의 상호 연관성을 알아볼 때 주로 사용되는 방법이다.

2

**교차분석 실습**

From scipy import stats

Pd.crosstab(index, columns, values, aggfunc)

Stats.chisquare(x, f\_exp=[0.8,0.2])

Stat, p\_value, \_, \_ = Stats.chi2\_contingency(crosstab)

**05**

**통계분석 - (5) 선형 회귀분석**

1

**선형 회귀 모형의 개념**

하나 혹은 그 이상의 독립변수가 종속변수에 미치는 영향을 추정하여 식으로 표현하는 통계기법

2

**단순선형 회귀분석**

하나의 독립변수가 종속변수에 미치는 영향을 추정하는 통계기법.

결측치, 이상치 처리 -> 상관성검증 -> 회귀분석시행 -> 회귀분석결과 -> 회귀진단

Import statsmodels.formula.api.smf

Lm = smf.ols(formula = ‘y ~ x’, data)

Result = lm.fit()

Print(Result.summary())

3

**다중 선형 회귀분석**

두 개 이상의 독립변수가 종속변수에 미치는 영향을 추정하는 통계기법.

결측치,이상치 처리 -> 상관성검증 -> 회귀분석 -> 회귀분석결과도출 -> 회귀진단

4

**다중 선형 회귀분석 실습**

Import statsmodels.formula.api.smf

Lm = smf.ols(formula = ‘y ~ ‘ + ‘+’.join(df.columns.difference([‘y’])), data=df)

Result = lm.fit()

Print(result.summary())

5

**다중 공선성 진단 방법**

# 분산팽창요인(VIF) 출력 함수 정의

def VIF(model):

for i,col in enumerate(model.exog\_names):

r = variance\_inflation\_factor(model.exog, i)

print(col,r)

VIF(lm) :

10이상일 경우, 다중공선성이 심한 특성.

6

**변수 선택법**

단계적 선택법 : 전진선택법, 후진제거법, 단계적선택법

06

통계분석 - (6) 로지스틱 회귀분석

1

로지스틱 회귀분석 개념

2

로지스틱 회귀분석 해석 방법

3

로지스틱 회귀분석 실습

4

다항 로지스틱 회귀분석 실습

07

통계분석 - (7) 군집분석

1

군집분석 개념

2

계층적 군집분석

3

비계층적 군집분석

4

혼합분포 군집분석

5

SOM

**08**

**통계분석 - (8) 연관분석**

1

**연관분석 Run-test**

모집단으로부터 추출된 표본집단이 무작위로 추출되었는지를 검정하는 작업

귀무가설 : 추출된 표본집단이 무작위로 추출되었다.

Import statsmodel.sandbox.stats.runs import Runs

Runs(x).runs\_test()

2

**연관분석 개념**

연관분석은 기업의 데이터베이스에서 상품의 구매, 서비스 등 일련의 거래 또는 사건들 간의 규칙을 발견하여 IF-THEN의 구조로 분석 결과의 연관성을 파악하는 데이터마이닝 방법론이다.

3

**연관분석 실습**

From mlxtend.preprocessing import TransactionEncoder

From mlxtend.frequent\_patterns import apriori, association\_rules

Te\_ary = Pd.get\_dummies(x)

Or

Te = TransactionEncoder()

Te\_ary = Te.fit(x).transform(x)

Runs(x).runs\_test()

Itemset = apriori(df, min\_support, max\_len, use\_colnames = True)

Association\_rules(itemset, metric = ‘confidence’ or ‘lift’, min\_threshold=0.5)

09

통계분석 - (9) 시계열분석

1

시계열분석 개념

2

시계열 분해

3

정상성 확인 및 검정

4

ARIMA

5

SARIMA 실습

6

Auto-ARIMA 실습

11

머신러닝 - (2) 선형회귀분석

2

LinearRegression 실습

3

Lidge, Rasso, ElasticNet 실습

12

머신러닝 - (3) 의사결정나무

1

의사결정나무 개념

2

의사결정나무 (분류)

3

의사결정나무 (회귀)

4

의사결정나무 실습(시각화)

13

머신러닝 - (4) 앙상블기법

1

앙상블 개념

2

배깅,부스팅 개념

3

배깅(회귀)

4

배깅(분류)

5

랜덤포래스트(회귀)

6

랜덤포래스트(분류)

7

부스팅(회귀)

8

부스팅(분류)

14

머신러닝 - (5) 나이브베이즈 & KNN

1

나이브 베이즈 개념

2

나이브 베이즈 분류기 실습

3

KNN 분류기 개념 & 실습

15

머신러닝 - (6) SVM & 인공신경망

1

Support Vector Machine 개념

2

SVC 실습

3

SVR 실습

4

인공신경망(분류)

5

인공신경망(회귀)

**16**

**머신러닝 - (7) 불균형 데이터 처리**

**1**

**언더 샘플링 기법**

From imblearn.under\_sampling import RandomUnderSampler, EditedNearestNeighbours

rus = RandomUnderSampler()

X\_rus, y\_rus = rus.fit\_resample(X,y)

**2**

**오버 샘플링 기법**

from imblearn.over\_sampling import SMOTE, RandomOverSampler

sm = SMOTE(sampling\_strategy = ‘auto’, random\_state = 0)

x\_smote, y\_smote = sm.fit\_resample(x\_train, y\_train)

ros = RandomOverSampler()

X\_ros, y\_ros = ros.fit\_resample(X,y)