데이터에듀 모의고사(정형데이터, 통계분석)

모의고사 1

정형데이터

In [1]:

```
import pandas as pd
```

In [10]:

```
data = pd.read_csv('data/lotto.csv')
```

In [11]:

```
data.head()
```

Out[11]:

	time_id	num1	num2	num3	num4	num5	num6
0	859	8	22	35	38	39	41
1	858	9	13	32	38	39	43
2	857	6	10	16	28	34	38
3	856	10	24	40	41	43	44
4	855	8	15	17	19	43	44

In [6]:

```
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
```

In [9]:

```
dataset = df.iloc[:, 1:].values
```

In [14]:

```
dataset
```

Out[14]:

```
array([[ 8, 22, 35, 38, 39, 41],
       [ 9, 13, 32, 38, 39, 43],
       [ 6, 10, 16, 28, 34, 38],
       ...,
       [11, 16, 19, 21, 27, 31],
       [ 9, 13, 21, 25, 32, 42],
       [10, 23, 29, 33, 37, 40]], dtype=int64)
```

In [18]:

```
te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
```

In [19]:

```
df = pd.DataFrame(te_ary, columns = te.columns_)
```

In [23]:

```
df.head()
```

Out[23]:

	1	2	3	4	5	6	7	8	9	10	...	36	37	38
0	False	False	False	False	False	False	False	True	False	False	...	False	False	True
1	False	False	False	False	False	False	False	False	True	False	...	False	False	True
2	False	False	False	False	False	True	False	False	False	True	...	False	False	True
3	False	False	False	False	False	False	False	False	False	True	...	False	False	False
4	False	False	False	False	False	False	False	True	False	False	...	False	False	False

5 rows × 45 columns

In [49]:

```
df.shape
```

Out[49]:

```
(859, 45)
```

In [31]:

```
df_top10 = df.sum(axis=0).sort_values(ascending=False).head(10)
```

In [32]:

```
df_top10
```

Out[32]:

```
34    134
27    126
40    126
43    125
20    124
12    123
17    123
1     122
10    122
13    122
dtype: int64
```

In [33]:

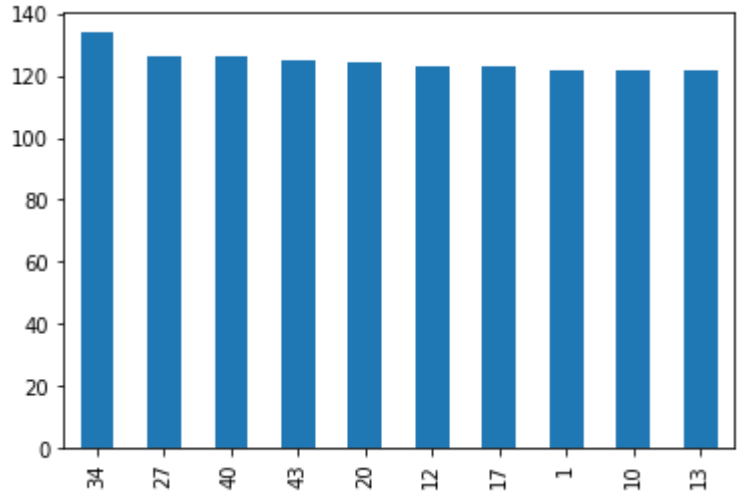
```
import matplotlib.pyplot as plt
```

In [34]:

```
df_top10.plot(kind='bar')
```

Out[34]:

<AxesSubplot:>



In [35]:

```
frequent_itemsets = apriori(df, min_support=0.002, max_len=6)
```

In [44]:

```
rules_1 = association_rules(frequent_itemsets, metric='lift')
```

In [45]:

```
rules_1
```

Out[45]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
0	(0)	(1)	0.142026	0.130384	0.022119	0.155738	1.194453
1	(1)	(0)	0.130384	0.142026	0.022119	0.169643	1.194453
2	(0)	(2)	0.142026	0.129220	0.023283	0.163934	1.268646
3	(2)	(0)	0.129220	0.142026	0.023283	0.180180	1.268646
4	(0)	(4)	0.142026	0.138533	0.017462	0.122951	0.887519
...	...	...	...	...	...	...	...
35741	(35)	(25, 42, 30, 15)	0.133877	0.002328	0.002328	0.017391	7.469565
35742	(42)	(25, 35, 30, 15)	0.145518	0.002328	0.002328	0.016000	6.872000
35743	(15)	(25, 42, 35, 30)	0.128056	0.002328	0.002328	0.018182	7.809091
35744	(25)	(42, 35, 30, 15)	0.135041	0.002328	0.002328	0.017241	7.405172
35745	(30)	(25, 42, 35, 15)	0.137369	0.002328	0.002328	0.016949	7.279661

35746 rows × 9 columns

In [46]:

```
# 남은 규칙 적용
rules_1 = rules_1[rules_1['confidence']>=0.8]
rules_1 = rules_1[rules_1['antecedents'].map(lambda x: len(x) >=2)]
```

In [57]:

```
# 704개의 규칙이 생성됨
rules_1.shape
```

Out[57]:

```
(704, 9)
```

In [52]:

```
# Lift 기준 정렬
rules_1.sort_values('lift', ascending=False).head(10)
```

Out[52]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
35730	(25, 30, 15)	(42, 35)	0.002328	0.012806	0.002328	1.0	78.090909
35484	(33, 21, 23)	(6, 30)	0.002328	0.012806	0.002328	1.0	78.090909
35483	(33, 6, 30)	(21, 23)	0.002328	0.012806	0.002328	1.0	78.090909
35636	(25, 20, 13)	(17, 14)	0.002328	0.013970	0.002328	1.0	71.583333
35511	(33, 35, 9)	(43, 21)	0.002328	0.013970	0.002328	1.0	71.583333
35490	(21, 30, 23)	(33, 6)	0.002328	0.013970	0.002328	1.0	71.583333
35577	(19, 14, 23)	(11, 29)	0.002328	0.013970	0.002328	1.0	71.583333
35727	(25, 42, 15)	(35, 30)	0.002328	0.013970	0.002328	1.0	71.583333
35721	(42, 35, 15)	(25, 30)	0.002328	0.015134	0.002328	1.0	66.076923
35515	(33, 21, 9)	(43, 35)	0.002328	0.016298	0.002328	1.0	61.357143

In [65]:

```
# 데이터 타입이 frozenset이라고 한다
rules_1['consequents'][35511]
```

Out[65]:

```
frozenset({21, 43})
```

In [69]:

```
# 위에서 확인한 가장 많이 나온 번호 34를 조회함
rules_1[rules_1['consequents']==frozenset({34})].sort_values('lift', ascending=False)
```

Out[69]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
30017	(3, 4, 44)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
30267	(3, 25, 19)	(34)	0.003492	0.123399	0.003492	1.0	8.103774
30773	(42, 4, 30)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
31864	(16, 35, 7)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
32073	(8, 17, 22)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
32186	(9, 15, 23)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
32298	(9, 21, 39)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
32660	(25, 10, 19)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
33293	(42, 12, 22)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
33405	(42, 12, 44)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
33642	(13, 21, 38)	(34)	0.003492	0.123399	0.003492	1.0	8.103774
34173	(26, 15, 23)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
34369	(16, 19, 35)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
34858	(32, 26, 19)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
35015	(21, 30, 39)	(34)	0.002328	0.123399	0.002328	1.0	8.103774
35392	(32, 42, 44)	(34)	0.002328	0.123399	0.002328	1.0	8.103774

통계분석

In [107]:

```
import pandas as pd

data = pd.read_csv('D:/github/FIFA.csv', encoding='cp949')
```

In [108]:

```
data
```

Out[108]:

	ID	Name	Age	Nationality	Overall	Club	Preferred_Foot	Work_Ra
0	158023	L. Messi	31	Argentina	94	FC Barcelona	Left	Mediu Medi
1	20801	Cristiano Ronaldo	33	Portugal	94	Juventus	Right	High/ L
2	190871	Neymar Jr	26	Brazil	92	Paris Saint-Germain	Right	Hi Medi
3	193080	De Gea	27	Spain	91	Manchester United	Right	Mediu Medi
4	192985	K. De Bruyne	27	Belgium	91	Manchester City	Right	High/ H
...	...	...	...	...	...	...	...	
16637	238813	J. Lundstram	19	England	47	Crewe Alexandra	Right	Mediu Medi
16638	243165	N. Christoffersson	19	Sweden	47	Trelleborgs FF	Right	Mediu Medi
16639	241638	B. Worman	16	England	47	Cambridge United	Right	Mediu Medi
16640	246268	D. Walker-Rice	17	England	47	Tranmere Rovers	Right	Mediu Medi
16641	246269	G. Nugent	16	England	46	Tranmere Rovers	Right	Mediu Medi

16642 rows × 16 columns

In [109]:

```
data['Height'].isna().sum()
```

Out[109]:

```
0
```

In [78]:

```
temp1 = data['Height'].str.split("'", expand=True)
temp1[0] = temp1[0].astype('int')
temp1[1] = temp1[1].astype('int')

temp1['Height_cm']=temp1[0]*30 + temp1[1]*2.5
temp1
```

Out[78]:

	0	1	Height_cm
0	5	7	167.5
1	6	2	185.0
2	5	9	172.5
3	6	4	190.0
4	5	11	177.5
...	...	...	...
16637	5	9	172.5
16638	6	3	187.5
16639	5	8	170.0
16640	5	10	175.0
16641	5	10	175.0

16642 rows × 3 columns

In [83]:

```
data2 = pd.concat([data, temp1['Height_cm']], axis=1)
```

In [88]:

```
forward = ['LS','ST','RS','LW','LF','CF','RF','RW']
midfielder = ['LAM','CAM','RAM','LM','LCM','CM','RCM','RM']
defender = ['LWB','LDM','CDM','RDM','RWB','LB','LCB','CB','RCB','RB']
goalkeeper = ['GK']

data2.loc[data2['Position'].map(lambda x : x in forward),'Position_Class'] = 'Forward'
data2.loc[data2['Position'].map(lambda x : x in midfielder),'Position_Class'] = 'Midfie
lder'
data2.loc[data2['Position'].map(lambda x : x in defender),'Position_Class'] = 'Defende
r'
data2.loc[data2['Position'].map(lambda x : x in goalkeeper),'Position_Class'] = 'Goalke
eper'
```

In [111]:

```
data2['Position_Class'].isna().sum()
```

Out[111]:

```
0
```

In [90]:

```
data2['Position_Class'].unique()
```

Out[90]:

```
array(['Forward', 'Goalkeeper', 'Midfielder', 'Defender'], dtype=object)
```

일원배치분산분석, 이원배치분산분석

In [112]:

```
from statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
```

In [116]:

```
import warnings
warnings.filterwarnings('ignore')

df = data2[['Position_Class', 'Value']]

# the "C" indicates categorical data
model = ols('Value ~ C(Position_Class)', df).fit()
# model = ols('Value ~ C(col1)+C(col2)+C(col1):C(col2)', df).fit() # 이원배치분산분석

print(anova_lm(model))
```

```
                     df        sum_sq       mean_sq           F  \
C(Position_Class)     3.0  4.081181e+09  1.360394e+09  41.873906
Residual          16638.0  5.405330e+11  3.248786e+07        NaN

                       PR(>F)
C(Position_Class)  5.988667e-27
Residual                   NaN
```

해석

- SSA의 자유도는 3(집단의 수 -1), SST의 자유도는 16638(관측값의 수 - 집단의 수)임을 확인할 수 있다. p-value-값(5.98e-27)이 매우 낮게 나와 유의수준 0.05하에서 귀무가설을 기각한다. 따라서 네 가지 포지션(Position Class)에 따른 선수의 시장가치(Value)가 모두 동일하지는 않다고 결론내릴 수 있다. 즉, 포지션(Position_Class)별 선수의 시장가치(Vlaue)의 평균값들 중에서 적어도 어느 하나의 포지션은 통계적으로 유의한 차이가 있는 값을 가진다고 말 할 수 있다.

사후검정 - Tukey'HSD Test

In [117]:

```
from statsmodels.stats.multicomp import pairwise_tukeyhsd
```

In [120]:

```
posthoc = pairwise_tukeyhsd(df['Value'], df['Position_Class'], alpha=0.05)
print(posthoc)
```

```
        Multiple Comparison of Means - Tukey HSD, FWER=0.05
===================================================================
  group1     group2    meandiff  p-adj    lower      upper    reject
-------------------------------------------------------------------
  Defender    Forward    930.373  0.001   610.7429  1250.0031   True
  Defender Goalkeeper  -507.3848 0.0034  -887.6261  -127.1436   True
  Defender Midfielder   760.8787  0.001   486.7123   1035.045   True
   Forward Goalkeeper -1437.7579  0.001 -1865.9234 -1009.5923   True
   Forward Midfielder  -169.4944 0.5609  -506.9991   168.0104  False
Goalkeeper Midfielder  1268.2635  0.001   872.8782  1663.6488   True
-------------------------------------------------------------------
```

해석

- 사후분석에서는 귀무가설을 '집단들 사이의 평균은 같다', 대립가설을 '집단들 사이의 평균은 같지 않다'로 두고, 모든 집단 수준에 대해서 두 집단씩 짝을 지어 각각 다중 비교를 수행한다.
- 사후 분석 결과를 보면 Forward-Midfielder 포지션 간의 비교에 대해서 p-value가 0.05보다 크므로 귀무가설을 기각하지 않는다. 즉, 포지션이 Forward인 선수와 Midfielder인 선수들 간의 시장가치는 통계적으로 유의한 차이가 있다고 볼 수 없다. 반면 나머지 비교들에 대해서는 모두 수정된 p-value값이 0.05보다 작으므로 각각의 비교에 대한 귀무가설을 모두 기각한다. 즉 해당 포지션들에 대한 선수들의 시장가치 평균값은 각각 통계적으로 유의한 차이가 있다는 것을 알 수 있다. 또한 meandiff는 group1과 group2의 반응값 차이를 나타내는데(group2-group1), 음수인 경우 group1의 포지션보다 group2의 포지션의 Value 평균이 반응값만큼 높다는 것을 뜻한다. 예를들어 Forward일 때 시장가치가 통계적으로 유의하게 큰 값을 가진다고 해석할 수 있다.

이원배치분산분석

In [126]:

```
df2 = data2[['Value','Preferred_Foot','Position_Class']]
```

In [128]:

```
import warnings
warnings.filterwarnings('ignore')

# the "C" indicates categorical data
model = ols('Value ~ C(Preferred_Foot)+C(Position_Class)+C(Preferred_Foot):C(Position_Class)', df2).fit() # 이원배치분산분석

print(anova_lm(model))
```

```
                                    df         sum_sq       mean_sq
\
C(Preferred_Foot)                  1.0   1.460850e+08  1.460850e+08
C(Position_Class)                  3.0   4.087367e+09  1.362456e+09
C(Preferred_Foot):C(Position_Class)  3.0   4.736156e+08  1.578719e+08
Residual                       16634.0   5.399071e+11  3.245805e+07

                                     F        PR(>F)
C(Preferred_Foot)             4.500734  3.389505e-02
C(Position_Class)            41.975902  5.151392e-27
C(Preferred_Foot):C(Position_Class)   4.863874  2.207249e-03
Residual                           NaN           NaN
```

해석

- 주발(Preferred_Foot) 변수에 대한 p-value는 0.0039로 유의수준 0.05보다 작으므로 귀무가설을 기각하고 대립가설을 채택하여 선수의 주발에 따른 선수의 가치는 차이가 있다고 판단할 수 있다.
- 그리고 포지션(Position_Class) 변수에 대한 p-value는 5.12e-27보다 작아 유의수준 0.05하에서 매우 유의하므로 귀무가설을 기각하고 대립가설을 채택하여 선수의 포지션에 따른 선수의 가치에는 차이가 있다고 판단할 수 있다.
- 마지막으로 주발(Preferred Foot)과 포지션(Position_Class) 변수의 교호작용에 대한 p-value는 0.00221로 유의수준 0.05보다 작으므로 귀무가설을 기각하고 대립가설을 채택하여 선수의 주발과 포지션의 상호작용에 의한 효과가 있다고 판단할 수 있다.

모의고사 2

통계분석

In [130]:

```
data = pd.read_csv('data/Admission.csv')
```

In [131]:

```
data.head()
```

Out[131]:

	GRE	TOEFL	Univ_Rating	SOP	LOR	CGPA	Research	Chance_of_Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

In [132]:

```
data.shape
```

Out[132]:

```
(400, 8)
```

상관분석 - 상관계수, p-value

In [133]:

```
import scipy.stats

a = scipy.stats.pearsonr(data['GRE'], data['Chance_of_Admit'])
print('피어슨 상관계수 : ', a[0])
print('p-value : ', a[1])
```

해석

- p-value가 0.05보다 작으므로, 두 변수간의 상관관계는 통계적으로 유의하다.
- 상관계수는 약 0.80으로 GRE와 Chance_of_Admit은 양의 상관관계를 가지고 있음을 알 수 있다.

정형데이터

모의고사 3

통계분석

In [165]:

```
df = pd.read_csv('data/Carseats.csv')
```

In [167]:

```
df.head()
```

Out[167]:

	Sales	CompPrice	Income	Advertising	Population	Price	ShelveLoc	Age	Education	U
0	9.50	138	73	11	276	120	Bad	42	17	
1	11.22	111	48	16	260	83	Good	65	10	
2	10.06	113	35	10	269	80	Medium	59	12	
3	7.40	117	100	4	466	97	Medium	55	14	
4	4.15	141	64	3	340	128	Bad	38	13	

In [168]:

```
df['Urban'].unique()
```

Out[168]:

```
array(['Yes', 'No'], dtype=object)
```

In [177]:

```
uy = df[df['Urban']=='Yes']['Sales']
un = df[df['Urban']=='No']['Sales']
```

등분산성 먼저 검정

In [186]:

```
import scipy.stats as spst
print('바틀렛 검정 : ', spst.bartlett(uy, un))
print('플리그너 검정 : ', spst.fligner(uy, un))
print('레빈 검정 : ', spst.levene(uy, un))
```

```
바틀렛 검정 :  BartlettResult(statistic=0.019034162105407128, pvalue=0.89026
8591965826)
플리그너 검정 :  FlignerResult(statistic=6.416940137288126e-06, pvalue=0.9979
788265211879)
레빈 검정 :  LeveneResult(statistic=2.9559854965252977e-06, pvalue=0.9986290
607894824)
```

해석

- 등분산 검정 결과 유의확률(p-value)이 0.89, 1.0, 1.0으로 유의수준 0.05보다 크기 때문에 귀무가설을 기각하지 않는다. 따라서 도시와 도시가 아닌 집단의 데이터는 등분산 가정을 만족한다고 볼 수 있다.

독립표본 t-검정

In [188]:

```
import scipy.stats as spst
spst.ttest_ind(uy, un, equal_var=True) # 등분산이니까 equal_var=True
```

Out[188]:

```
Ttest_indResult(statistic=-0.30765346670661126, pvalue=0.7585069603942775)
```

해석

- 독립표본 t-검정 수행결과 검정통계량(t값)은 0.30765, 유의확률(p-value)는 0.7585다.
- p-value가 유의수준 0.05보다 크기 때문에 귀무가설을 기각하지 않는다. 따라서 도시지역과 도시가 아닌 지역간의 차 판매량에는 통계적으로 유의한 차이가 있다고 볼 수 없다는 결론을 내릴 수 있다.

잔차분석

- 잔차는 실제 값 y와 예측된 값 ŷ의 차 e를 말한다.
- sklearn의 linear_model.LinearRegression를 사용해서 모델을 생성하면 .score()라는 메서드를 사용할 수 있는데 R^2라고 하는 결정계수(coefficient of determination)를 돌려준다. 표현은 1-(u/v)
- 분자에 있는 u는 잔차의 제곱의 평균 RSS(residual sum of square)는 직선이 미처 Y에 대해 설명하지 못한 변화량을 의미한다.
- 분모에 있는 v는 TSS(total sum of squares), y값의 총 변화량으로 이해하면 된다.
- 결국 결정계수 R^2는 전체에서 직선이 미처 설명하지 못한 부분의 비율을 뺀 거다. 결정계수 R^2가 클수록 실제값과 예측값이 유사함을 의미하며, 데이터를 잘 설명한다고 이해하자.그러니 모델이 얼마나 정확한지 평가할 때도 이 결정계수 R^2가 기준이 될 수 있는 거다.
- 예를 들어, 주택 사이즈(“size_sqft”)와 침실 개수(“bedrooms”)를 기준으로 임대료를 예측하는 모델의 R^2이 0.72인 경우 그 2개의 변수들이 함께 임대료 변동의 72%를 설명한다는 뜻이다.
- 당연히 최선의 R^2는 1이겠지만 그건 말이 안 되고, 일반적으로 0.7 정도면 양호한 것으로 간주한다.

단계 선택법, 전진선택법, 후진제거법

- 별도 첨부