Contents

1. Written Homework1
2. Written Homework2
3. Programming Homework1

Written Homework1  
(reference : <https://www.nesdc.go.kr/portal/bbs/B0000005/view.do?nttId=10176&menuNo=200467&searchTime=::%20%EB%82%A0%EC%A7%9C%EA%B5%AC%EB%B6%84%20::&sdate=&edate=&pdate=&pollGubuncd=VT019&searchCnd=::%20%EA%B2%80%EC%83%89%EC%96%B4%EA%B5%AC%EB%B6%84%20::&searchWrd=&pageIndex=2>

)

The number of people selected

According to KBS, MBC and SBS, their sample size of 20th president election is 2,003 If Korean that enable to vote, population, are 50,000,000, It is 0.004percent.

Demographics

|  |  |  |
| --- | --- | --- |
| Sex / Gender | MALE | FEMALE |
| The number of sample | 997 | 1006 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Age | [18, 29] | [30, 39] | [40, 49] | [50, 59] | [60, 69] | [70, inf) |
| The number of sample | 340 | 297 | 372 | 394 | 334 | 266 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| location | Seoul | Gyeonggi / Incheon | Daejeon / Chungcheong / Sejong | Gwangju / Jeolla | Daegu / Gyeongsangbuk | Busan / Ulsan / Gyeongsangnam | Gangwon / Jeju |
| The number of sample | 383 | 636 | 211 | 197 | 192 | 300 | 84 |

Questions asked

1. What is Location that you live?
2. Sex / gender(not question, but prediction through voice)
3. What is your age?
4. Will you vote 20th president election on March 9th?
5. Are you vote when pre-election?
6. Who do you want to vote for?
7. Are you convinced about you chose?
8. What is the biggest reason that you chose?
9. Regardless of your choice, Who do you think will be selected?
10. Yoon seokyeoul and An chulsu is not merged. Who do you think was more at fault?
11. What do you think yoon seokyeoul and An chulsu become unification?
12. Choice one  
    1. To change power(정권)  
    2. To extend power
13. Do you think Moon jaein is good about government management?
14. Which party do you support?

Written homework2

According to digital marketing lecture, order of market research is setting search purpose, design research, collection data, design sampling, doing research, analysis data and reporting.

1. Set search purpose  
   My purpose is to start a food-delivery service
2. Design research  
   I should do not only descriptive research but also casual research. If I select my main target group and understand their characteristics well, my chances of having a successful business increase.
3. Collection data  
   I already have database about people who live in pangyo and I have a small budget, I will collect data through survey research.
4. Design sampling  
   I need to design sampling because I have a small budget. I suppose database columns are age and job.  
   For stratified sampling, I will bunble age data like 20’s, 30’s. and I concatenate age and job like 20’s-students, 20’s-time worker, 50’s-CEO. And I will do stratified sampling. The number of sample is 0.1% of length of data. I will consider only man(male) data because I think I can’t do business targeting women.
5. Doing research  
   For survey research, I have to make survey.  
   1. How many Frequency do you food delivery order?  
   2. What do you prefer?(like Korean food, salad, low calorie, Italian food and Chinese food)  
   3. How much is your average budget for food delivery per meal?  
   4. What do you prefer time of day for delivery?  
   5. What do you consider Importance of factors like delivery speed, food quality, price and variety  
     
   and do survey research. I utilize online survey and call survey.
6. Analysis data  
   I have to decide food and location based on survey research.  
   I suppose the most received answer is every day, Korean food, 10,000, launch and delivery speed.  
   Considering all this, I will sell stir-fried pork(제육볶음). Because It is Korean food, turnover rate is fast, price is less than 10,000₩ and you can eat this every day and never get tired of it.

Analysis result

A blue dots in a triangle

Description automatically generated with medium confidence

A graph of a blue dot

Description automatically generated with medium confidence

A graph of blue dots

Description automatically generated

A diagram of a number of blue dots

Description automatically generated

A blue graph with numbers and a black grid

Description automatically generated with medium confidence

A graph with many colored dots

Description automatically generated

A line graph with numbers

Description automatically generated

Code

*import* pandas *as* pd

*import* numpy *as* anp

*import* sklearn

*from* sklearn *import* datasets

*import* matplotlib.pyplot *as* plt

*import* seaborn *as* sns

*from* sklearn.model\_selection *import* train\_test\_split

*from* sklearn.decomposition *import* PCA

*from* sklearn.preprocessing *import* StandardScaler

*from* mpl\_toolkits.mplot3d *import* Axes3D

*import* numpy *as* np

df = pd.read\_csv('housing.csv')

df.head()

|  | **longitude** | **latitude** | **housing\_median\_age** | **total\_rooms** | **total\_bedrooms** | **population** | **households** | **median\_income** | **median\_house\_value** | **ocean\_proximity** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | 322.0 | 126.0 | 8.3252 | 452600.0 | NEAR BAY |
| 1 | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | 2401.0 | 1138.0 | 8.3014 | 358500.0 | NEAR BAY |
| 2 | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | 496.0 | 177.0 | 7.2574 | 352100.0 | NEAR BAY |
| 3 | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | 558.0 | 219.0 | 5.6431 | 341300.0 | NEAR BAY |
| 4 | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | 565.0 | 259.0 | 3.8462 | 342200.0 | NEAR BAY |

X, y = df.drop(labels='median\_house\_value', axis=1), df['median\_house\_value']

X.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 20640 entries, 0 to 20639

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 longitude 20640 non-null float64

1 latitude 20640 non-null float64

2 housing\_median\_age 20640 non-null float64

3 total\_rooms 20640 non-null float64

4 total\_bedrooms 20433 non-null float64

5 population 20640 non-null float64

6 households 20640 non-null float64

7 median\_income 20640 non-null float64

8 ocean\_proximity 20640 non-null object

dtypes: float64(8), object(1)

memory usage: 1.4+ MB

min(df['total\_bedrooms'])

1.0

X = X.fillna(0)

X.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 20640 entries, 0 to 20639

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 longitude 20640 non-null float64

1 latitude 20640 non-null float64

2 housing\_median\_age 20640 non-null float64

3 total\_rooms 20640 non-null float64

4 total\_bedrooms 20640 non-null float64

5 population 20640 non-null float64

6 households 20640 non-null float64

7 median\_income 20640 non-null float64

8 ocean\_proximity 20640 non-null object

dtypes: float64(8), object(1)

memory usage: 1.4+ MB

class LabelEncoder():

def \_\_init\_\_(self, series:pd.Series):

*self*.series = series

*self*.label = {}

def fit(self):

*self*.label = {}

change = 0

*for* i *in* *self*.series:

*if*(i not in *self*.label.keys()):

*self*.label[i] = change

change+=1

def transforms(self):

tmp = *self*.series.copy()

*for* i, j *in* enumerate(*self*.series):

tmp[i] = *self*.label[j]

*return* tmp

encoder = LabelEncoder(X['ocean\_proximity'])

encoder.fit()

X['ocean\_proximity'] = encoder.transforms().astype('float64')

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_scaled

array([[-1.32783522, 1.05254828, 0.98214266, ..., -0.97703285,

2.34476576, -1.71472665],

[-1.32284391, 1.04318455, -0.60701891, ..., 1.66996103,

2.33223796, -1.71472665],

[-1.33282653, 1.03850269, 1.85618152, ..., -0.84363692,

1.7826994 , -1.71472665],

...,

[-0.8237132 , 1.77823747, -0.92485123, ..., -0.17404163,

-1.14259331, 0.62663072],

[-0.87362627, 1.77823747, -0.84539315, ..., -0.39375258,

-1.05458292, 0.62663072],

[-0.83369581, 1.75014627, -1.00430931, ..., 0.07967221,

-0.78012947, 0.62663072]])

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X\_scaled)

X\_pca

array([[-2.12161627, 1.76578647],

[ 2.90641102, 2.29772754],

[-1.99987349, 1.84985111],

...,

[-0.43120358, 1.58513546],

[-0.84282735, 1.57410014],

[ 0.15621492, 1.63343721]])

X\_df = pd.DataFrame(X\_pca,columns=['feature1','feature2'])

X\_df.head()

|  | **feature1** | **feature2** |
| --- | --- | --- |
| 0 | -2.121616 | 1.765786 |
| 1 | 2.906411 | 2.297728 |
| 2 | -1.999873 | 1.849851 |
| 3 | -1.949403 | 1.870331 |
| 4 | -1.807118 | 1.905899 |
|  |  |  |

sns.scatterplot(X\_df, x='feature1',y='feature2')

A blue dots in a triangle

Description automatically generated with medium confidence

fig = plt.figure(figsize=(5, 5))

ax = fig.add\_subplot(111,projection='3d')

ax.scatter(X\_df['feature1'],X\_df['feature2'],y, s=10)

plt.show()

A graph of a blue dot

Description automatically generated with medium confidence

sns.scatterplot(x=X\_df['feature1'],y=y)

A graph of blue dots

Description automatically generated

sns.scatterplot(x=X\_df['feature2'],y=y)

A diagram of a number of blue dots

Description automatically generated

sns.histplot(y, bins=5)

A blue graph with numbers and a black grid

Description automatically generated with medium confidence

categrical\_target = np.digitize(y, bins = [-np.inf, 100000, 200000, 300000,400000, np.inf])

fig = plt.figure(figsize=(8,8))

ax = fig.add\_subplot(1,1,1)

ax.set\_xlabel('feature1')

ax.set\_ylabel('feature2')

ax.set\_title('2 component PCA', fontsize=20)

targets = [1,2,3,4,5]

colors = ['r','g','b','y','darkgreen']

*for* target, color *in* zip(targets,colors):

idx = categrical\_target == target

ax.scatter(X\_df.loc[idx,'feature1'], X\_df.loc[idx,'feature2'],c=color,s=50)

ax.legend(targets)

ax.grid()

A graph with many colored dots

Description automatically generated

result = []

*for* i *in* range(1,len(X.columns)):

pca = PCA(n\_components=i)

X\_pca = pca.fit(X\_scaled)

result.append(pca.explained\_variance\_ratio\_)

result

array([[-2.12161627, 1.76578647],

[ 2.90641102, 2.29772754],

[-1.99987349, 1.84985111],

...,

[-0.43120358, 1.58513546],

[-0.84282735, 1.57410014],

[ 0.15621492, 1.63343721]])

array([[-1.32783522, 1.05254828, 0.98214266, ..., -0.97703285,

2.34476576, -1.71472665],

[-1.32284391, 1.04318455, -0.60701891, ..., 1.66996103,

2.33223796, -1.71472665],

[-1.33282653, 1.03850269, 1.85618152, ..., -0.84363692,

1.7826994 , -1.71472665],

...,

[-0.8237132 , 1.77823747, -0.92485123, ..., -0.17404163,

-1.14259331, 0.62663072],

[-0.87362627, 1.77823747, -0.84539315, ..., -0.39375258,

-1.05458292, 0.62663072],

[-0.83369581, 1.75014627, -1.00430931, ..., 0.07967221,

-0.78012947, 0.62663072]])

|  | **feature1** | **feature2** |
| --- | --- | --- |
| 0 | -2.121616 | 1.765786 |
| 1 | 2.906411 | 2.297728 |
| 2 | -1.999873 | 1.849851 |
| 3 | -1.949403 | 1.870331 |
| 4 | -1.807118 | 1.905899 |

<Axes: xlabel='feature1', ylabel='feature2'>

<Axes: xlabel='feature1', ylabel='median\_house\_value'>

<Axes: xlabel='feature2', ylabel='median\_house\_value'>

<Axes: xlabel='median\_house\_value', ylabel='Count'>

[array([0.43227689]),

array([0.43227689, 0.21571067]),

array([0.43227689, 0.21571067, 0.12849346]),

array([0.43227689, 0.21571067, 0.12849346, 0.11779363]),

array([0.43227689, 0.21571067, 0.12849346, 0.11779363, 0.07193313]),

array([0.43227689, 0.21571067, 0.12849346, 0.11779363, 0.07193313,

0.01684726]),

array([0.43227689, 0.21571067, 0.12849346, 0.11779363, 0.07193313,

0.01684726, 0.00869974]),

array([0.43227689, 0.21571067, 0.12849346, 0.11779363, 0.07193313,

0.01684726, 0.00869974, 0.0053102 ])]

result\_sum = []

*for* i *in* result:

result\_sum.append(1-sum(i))

result\_sum

[0.5677231051685814,

0.35201243656954395,

0.2235189731456777,

0.10572534663780764,

0.03379221243315411,

0.016944951464945124,

0.008245211301703748,

0.002935015570931787]

sns.lineplot(x=range(1,len(X.columns)),y=result\_sum)

A line graph with numbers

Description automatically generated