Data Mining ECS766P Data Mining Assignment 1

1. Consider the following sales data: [3, 16, 20, 4, 2, 5, 10, 9, 13, 7, 14, 8]. Apply the following binning techniques on the data, assuming 3 bins in each case:

1. Equal-frequency binning
2. Smoothing by bin boundaries

2. Use the below methods to normalize the following data: [10, 5, 25, 50, 35]:

1. min-max normalization with min=0 and max=1.
2. z-score normalization

3. Students at two universities, University A and University B, have been provided with feedback forms on student satisfaction, with the below responses recorded. Is student satisfaction correlated with a specific university? Use a chi-square test to find out, assuming a significance level of 0.001 and a corresponding chi-square significance value of 10.828. [1 mark out of 5]

|Rating/University |University A |University B |

|--- |--- |--- |

|Satisfied |71 |129 |

|Dissatisfied |37 |73 |

4. Load the CSV file country-income.csv which includes both numerical and categorical attributes. Perform data cleaning in order to replace any NaN values with the mean of the value for a given field. Then replace any categorical labels with numerical labels. Display the resulting dataset. You can use the sklearn.impute and sklearn.preprocessing packages to assist you.

5. Load the CSV file shoesize.csv, which includes measurements of shoe size and height (in inches) for 408 subjects, both female and male. Plot the scatterplots of shoe size versus height for female and male subjects separately. Compute the Pearson's correlation coefficient of shoe size versus height for female and male subjects separately. What can be inferred by the scatterplots and computed correlation coefficients? You can implement your own formulation of the correlation coefficient or use the scipy.stats package to assist you.

6. Using the pre-processed breast cancer dataset from subsection 1.1 of this notebook (which replaced any missing values with their median), perform Principal Component Analysis with 2 components. Compute the explained variance ratio for each component, and plot the scatterplot of all samples along the two principal components, color-coded according to the "Class" column (this column should not be used in the PCA analysis). Ensure that your data is normalized prior to performing PCA. What insights can you obtain by the explained variance ratio of each component, and by viewing the scatterplot of the principal components?

Part 1 of 2

Question 1:

Data: 3, 16, 20, 4, 2, 5, 10, 9, 13, 7, 14, 8

Sorted Data: 2, 3, 4, 5, 7, 8, 9, 10, 13, 14, 16, 20

A:

List count/bin count = count in each bin

12/3 = 4

Use the sorted list and bin each number consecutively, until it fills up to the bin count. Then move to the next bin.

Bin 1: 2, 3, 4, 5

Bin 2: 7, 8, 9, 10

Bin 3: 13, 14, 16, 20

B:

Use the result from equal-frequency binning, change the value of each value to the minimum or maximum of their respective bin, whichever is closer.

Bin 1: 2, 2, 5, 5

Bin 2: 7, 7, 10, 10

Bin 3: 13, 13, 13, 20

Question 2:

Data: 10, 5, 25, 50, 35

Sorted Data: 5, 10, 25, 35, 50

Min: 5 Max: 50

Range = 50-5 = 45

A:

For each value, subtract the minimum and divide by the range.

Original	5	10	25	35	50
Calculation	5-5/45	10-5/45	25-5/45	35-5/45	50-5/45
Normalised	0	0.11	0.44	0.67	1

B:

For each value, subtract the mean and divide by the standard deviation.

Mean: 5 +10 +25 +35 +50 /5 = 25

Standard deviation:

The square root of ((the sum of the values subtract the mean)/number of values)

Values	5	10	25	35	50
Sub Mean	-20	-15	0	10	25
Squared	400	225	0	100	625

Sum squared = 1350 Number of values = 5

3350/5 = 270

Square root of 270 = 16.43

Original	5	10	25	35	50
Calculation	5-25/16.43	10-25/16.43	25-25/16.43	35-25/16.43	50-25/16.43
Z Normalised	-1.22	-0.91	0	0.61	1.52

Question 3:

RAW VALUES	A	B	TOTAL
SATISFIED	71	129	200
DISSATISFIED	37	73	110
TOTAL	108	202	310

EXPECTED	A	B
SATISFIED	69.67742	130.3226
DISSATISFIED	38.32258	71.67742

CHI SQUARED	A	B
SATISFIED	0.025105	0.013422
DISSATISFIED	0.045645	0.024404

The expected value is the total of one group multiplied by the total of an outcome divided by the total of both groups. For example: satisfied and A is expected to be 200 x 108 / 310 = 69.8

The chi squared test: for each value you subtract the expected value, then square this. Then divide that by the expected value. For example for satisfied and A, the calculation is ((71 – 69.7)^2)/69.7 = 0.025

Sum up the results of all values, and then evaluate if this is statistically significant (higher is a more significant difference from the expected).

Chi square = 0.025 + 0.013 + 0.04 + 0.02 = 0.108575 (not rounded in calculation)

0.108 is less than 10.828, therefore we can accept the null hypothesis, which means that the two universities and satisfactions are independent and not correlated.

Question 4:

Replacing NaN values with mean:

```
import pandas as pd
income = pd.read_csv('country-income.csv')
income2 = income.fillna(income.mean())
income2
```

Output:

	Region	Age	Income	Online Shopper
0	India	49.000000	86400.000000	No
1	Brazil	32.000000	57600.000000	Yes
2	USA	35.000000	64800.000000	No
3	Brazil	43.000000	73200.000000	No
4	USA	45.000000	76533.333333	Yes
5	India	40.000000	69600.000000	Yes
6	Brazil	43.777778	62400.000000	No
7	India	53.000000	94800.000000	Yes
8	USA	55.000000	99600.000000	No
9	India	42.000000	80400.000000	Yes

Replacing categorical variables with numerical:

```
from sklearn.preprocessing import LabelEncoder
labenc = LabelEncoder()
income2['Online Shopper'] = labenc.fit_transform(income2['Online Shopper'])
income2['Region'] = labenc.fit_transform(income2['Region'])
income2
```

Output:

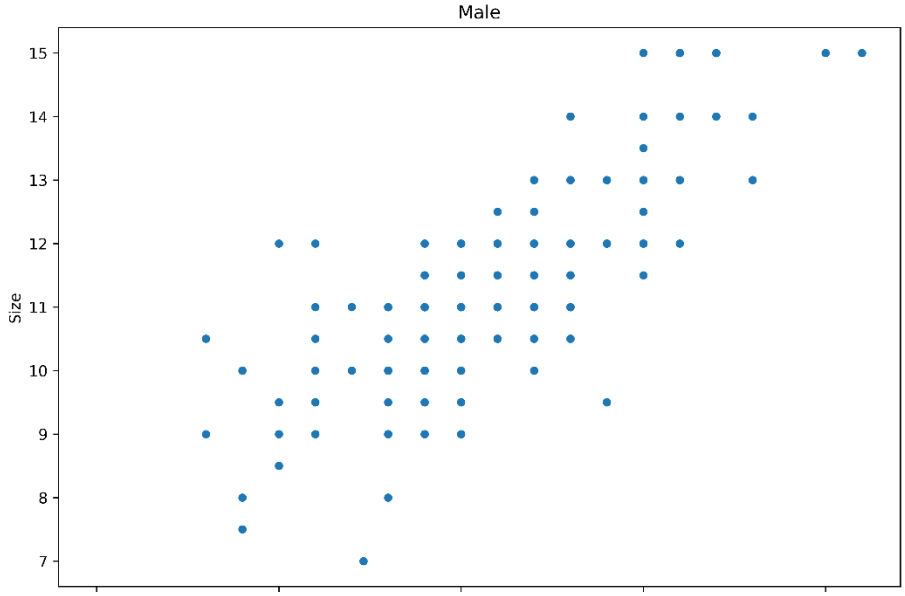
	Region	Age	Income	Online Shopper
0	1	49.000000	86400.000000	0
1	0	32.000000	57600.000000	1
2	2	35.000000	64800.000000	0
3	0	43.000000	73200.000000	0
4	2	45.000000	76533.333333	1
5	1	40.000000	69600.000000	1
6	0	43.777778	62400.000000	0
7	1	53.000000	94800.000000	1
8	2	55.000000	99600.000000	0
9	1	42.000000	80400.000000	1

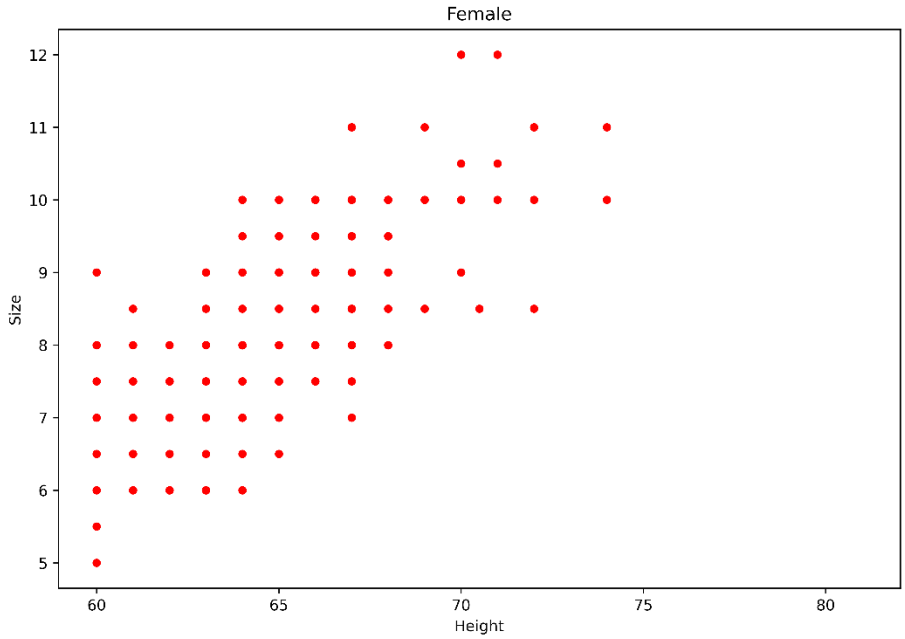
Question 5

Plotting genders separately:

```
shoe = pd.read_csv('shoesize.csv')
shoeM = shoe.loc[shoe.Gender == 'M']
shoeF = shoe.loc[shoe.Gender == 'F']

fig, (a1,a2) = plt.subplots(2, 1, figsize=(10,15), sharex=True)

shoeM.plot(ax=a1, kind='scatter', x='Height', y='Size')
shoeF.plot(ax=a2, kind='scatter', x='Height', y='Size', color='red')
a1.set_title("Male")
a2.set_title("Female")
```

Output:





Calculating Pearson's correlation coefficient

```
from scipy import stats
m_coef = stats.pearsonr(shoeM.loc[:, 'Height'], shoeM.loc[:, 'Size'])
f_coef = stats.pearsonr(shoeF.loc[:, 'Height'], shoeF.loc[:, 'Size'])
print('Coefficient for male is: ' +
str(m_coef[0]))
print('Coefficient for female is: ' +
str(f_coef[0]))
```

Output:

Coefficient for male is: 0.7677093547300977

Coefficient for female is: 0.707811941714397

Pearson's correlation coefficient gives out a value from -1 to +1. As the value computed here is close to 1, it shows a strong correlation between height and shoe size. The male coefficient is higher which suggests that the correlation is truer for males rather than females.

Question 6:

PCA:

```
cancer = data2
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

scaled_data = scaler.fit_transform(cancer)

pca = PCA(n_components = 2)
dim_reduced = pca.fit_transform(scaled_data)
dim_reduced
```

Output:

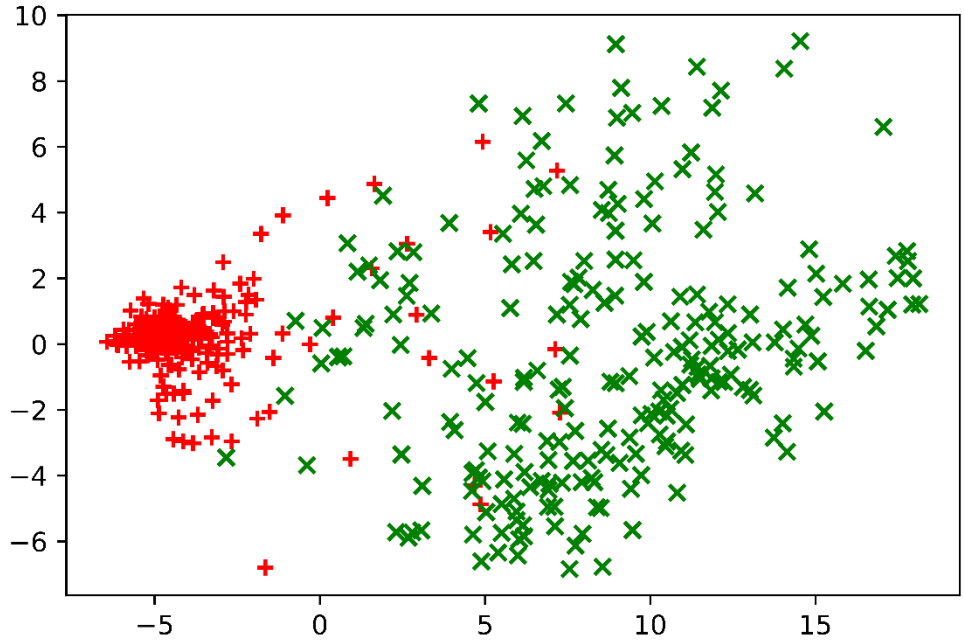
```
array([[-1.45622036, -0.11021043],
       [ 1.46627924, -0.54489351],
       [-1.5793114 , -0.07485359],
       ...,
       [ 3.8253587 , -0.18046559],
       [ 2.26948193, -1.11343514],
       [ 2.66445312, -1.19724198]])
```

Explained Variance Ratio

```
pca.explained_variance_ratio_ = 0.65445704, 0.0860859
```

Plotting:

```
reduced = pd.DataFrame(dim_reduced, columns=['dim1','dim2'])
reduced2 = reduced.join(breast.Class)

colors = {2:'r', 4:'g'}
markerTypes = {2:'+', 4:'x'}

for classedtype in markerTypes:
    r = reduced2[reduced2['Class'] == classedtype]
    plt.scatter(r['dim1'],r['dim2'], c=colors[classedtype], marker=markerTypes[classedtype])
```



As PCA has reduced the attribute dimensions, we can see all of the attributes plotted in 2D. Samples classed as ‘2’ (red) or ‘4’ (green) are loosely grouped together, which is what we expect given that a human has decided the class of each sample non-computationally. The red values are more similar as they are grouped closer together, however, the green values are more sparsely spread which suggests that samples in this class do not have as similar attributes. It also shows that there are many values which overlap each class, this could suggest that the sample has been classed wrongly, or there is a ‘grey area’.

However, the explained variance ratio shows us that the first principal component explains 65% of the data, and the second only 8%. So, with these two components, 74% of the data is explained.

Part 2 of 2

1. In Section 1, what kind of relationship can be inferred from summary statistics regarding ``ACT composite score`` and ``SAT total score``? Which visualisations make this relationship apparent?

2. Based on the box plots presented in Section 1, what is the relationship between ``parental level of education`` and ``parental income``? Using table visualisation, find and show the entire rows that correspond to the outliers regarding ``parental income`` whose parents have a master's degree.

3. Using an example, explain the importance of scaling features so that their magnitudes are comparable when computing distances.

4. In Section 1, the distance matrix visualisation is not very informative. However, it is still possible to infer that the average distance between students whose parents only have some high school education and students whose parents have a master's degree is larger

than the average distance between students whose parents only have some high school education. Explain how this inference is possible from the visualisation.

5. In Section 2, increase the number of evenly spaced numbers from 10 to 100 for both axes and observe the corresponding heat map created through nearest neighbour interpolation. Read about this interpolation method and explain what you observed.

6. The function ``load_wine`` from ``sklearn.datasets`` can be used to load the *wine dataset* into a ``DataFrame`` by using the commands ``data = load_wine()``, ``df = pd.DataFrame(data.data, columns=data.feature_names)``, and ``df['target'] = pd.Series(data.target)``.

6.1. Load the wine dataset. Compute the frequency of each value of the 'target' feature.

6.2. Compute univariate and multivariate summaries for all numerical features (except from the target feature). Group observations by the target feature and compute the corresponding **median** for each numerical feature.

6.3. Group observations by the target feature and create one box plot of ``alcohol`` for each group.

6.4. Create a scatter plot for the pair of **distinct** numerical features with the highest correlation.

6.5. Exclude the target feature, standardize the remaining numerical features, and display a projection obtained by multidimensional scaling. Color the points by the target feature.

Question 1:

Both SAT and ACT score are strongly correlated, from the value of 0.89 which is close to the value of 1 which would imply a perfect correlation. You can see this correlation in the pairplot. The ACT and SAT scores have different ranges, with the SAT score ranging from 1498 - 2397 and the ACT score ranging from 19 – 36, thus, we cannot make comparisons on the summary statistics in this current form.

Question 2:

The categories with higher education, have spreads higher than categories lower than it, generally, the higher the parental education level, the higher the full range of parental income.

Outliers:

Code:

```
masters = df[df['parental level of education'] == "master's degree"]
Q3 = masters.loc[:, "parental income"].quantile(0.75)
Q1 = masters.loc[:, "parental income"].quantile(0.25)
IQR = Q3 - Q1
upper = Q3 + 1.5 * IQR
lower = Q1 - 1.5 * IQR

def outlier(row):
    if row.loc["parental income"] > upper:
        return "Outlier"
    elif row.loc["parental income"] < lower:
        return "Outlier"
    else:
        return "No"

outlier_pre = masters.assign(Outlier = masters.apply(outlier, axis=1))

outliers = outlier_pre.loc[(outlier_pre.Outlier == 'Outlier')]
outliers
```

Output:

	ACT composite score	SAT total score	parental level of education	parental income	high school gpa	college gpa	years to graduate
411	31	2108	master's degree	120391	4.0	3.6	4
420	28	2097	master's degree	59724	3.9	3.2	4

Question 3:

Attributes are scaled to make sure they are equally weighted amongst each other. This prevents features that have large ranges (such as the parental income 1882 - 120391), ‘looking’ as if it has a bigger difference than smaller-ranged features (like ACT score 19-36). One method is to make all the values scaled from 0-1 (0 being the lowest value and 1 being the highest).

For income, the value 54000 would be converted like so:

54000-1882/120391-1882 = 0.44

For ACT score, the value 28 would be converted like so:

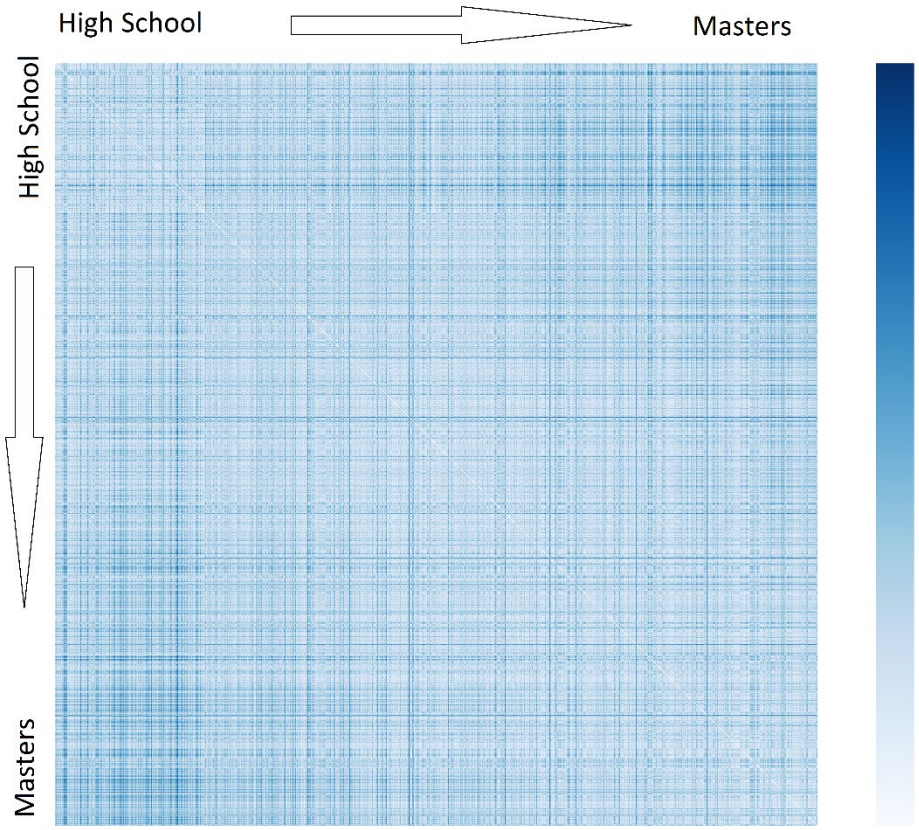
28-19/36-19 = 0.52

Hence, you can now compare these two values: the ACT score is higher up in its range than the income.

Question 4:

The heatmap is arranged in 6x6 blocks dependent on the number of education groups, with a mirror image on either side of the downward diagonal. From left to right and top to

down it begins with high school descending to masters. It is hard to see where the boundary between the groups is, but if you look at the bottom left corner, this area shows the distances of high school compared to masters, which is very dark compared to the rest of the heatmap. This darkness shows that the distances are larger between high school and masters.



Question 5:

Looks smoother because nearest neighbour relies on the closest single point. When we increase the amount of data points (like increasing the linspace), there are more points to be nearest neighbours, so the resulting image looks like it has a smooth gradient.

Question 6:

1:

```
from sklearn.datasets import load_wine
data = load_wine()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['target'] = pd.Series(data.target)
freq_target = df['target'].value_counts()/len(df)
freq_target
```

Output:

1 0.398876

0 0.331461

2 0.269663

2:

Univariate:

	ALCOHOL	MALIC_ACID	ASH	ALCALINITY_OF_ASH	MAGNESIUM	TOTAL_PHENOLS	FLAVANOIDS	NONFLAVANOID_PHENOLS	PROANTHOCYANINS	COLOR_INTENSITY	HUE	OD280/OD315_OF_DILUTED_WINES	PROLINE
COUNT	178	178	178	178	178	178	178	178	178	178	178	178	178
MEAN	13.000062	2.336348	2.366517	19.49494	99.74157	2.295112	2.02927	0.361854	1.590899	5.05809	0.957449	2.611685	746.8933
STD	0.811827	1.117146	0.274344	3.339564	14.28248	0.625851	0.998859	0.124453	0.572359	2.318286	0.228572	0.70999	314.9075
MIN	11.03	0.74	1.36	10.6	70	0.98	0.34	0.13	0.41	1.28	0.48	1.27	278
25%	12.3625	1.6025	2.21	17.2	88	1.7425	1.205	0.27	1.25	3.22	0.7825	1.9375	500.5
50%	13.05	1.865	2.36	19.5	98	2.355	2.135	0.34	1.555	4.69	0.965	2.78	673.5
75%	13.6775	3.0825	2.5575	21.5	107	2.8	2.875	0.4375	1.95	6.2	1.12	3.17	985
MAX	14.83	5.8	3.23	30	162	3.88	5.08	0.66	3.58	13	1.71	4	1680

```
no_target = df.drop(columns="target")
no_target.describe()
```

Medians

```
median = df.groupby('target').median()
median
```

TARGET	ALCOHOL	MALIC_ACID	ASH	ALCALINITY_OF_ASH	MAGNESIUM	TOTAL_PHENOLS	FLAVANOIDS	NONFLAVANOID_PHENOLS	PROANTHOCYANINS	COLOR_INTENSITY	HUE	OD280/OD315_OF_DILUTED_WINES	PROLINE
0	13.75	1.77	2.44	16.8	104	2.8	2.98	0.29	1.87	5.4	1.07	3.17	1095
1	12.29	1.61	2.24	20	88	2.2	2.03	0.37	1.61	2.9	1.04	2.83	495
2	13.165	3.265	2.38	21	97	1.635	0.685	0.47	1.105	7.55	0.665	1.66	627.5

Multivariate

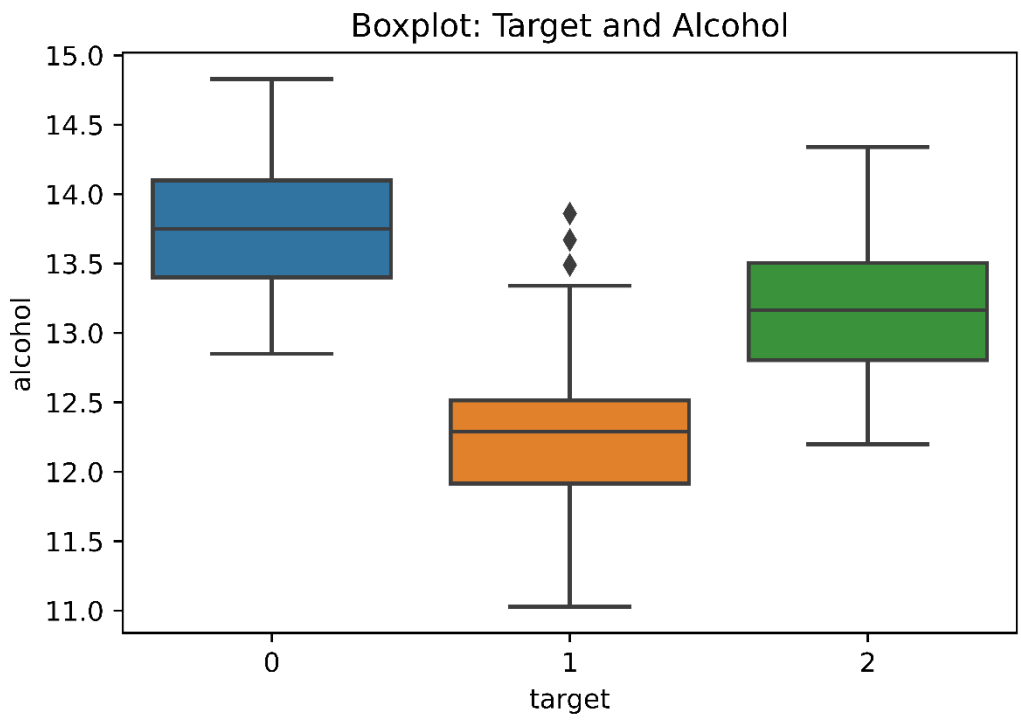
```
no_target.corr()
```

	ALCOHOL	MALIC_ACID	ASH	ALCALINITY_OF_ASH	MAGNESIUM	TOTAL_PHENOLS	FLAVANOIDS	NONFLAVANOID_PHENOLS	PROANTHOCYANINS	COLOR_INTENSITY	HUE	OD280/OD315_OF_DILUTED_WINES	PROLINE
ALCOHOL	1	0.094397	0.211545	-0.310235	0.270798	0.289101	0.236815	-0.155929	0.136698	0.546364	-0.071747	0.072343	0.643772
MALIC_ACID	0.094397	1	0.164045	0.2885	-0.054575	-0.335167	-0.411007	0.292977	-0.220746	0.248985	-0.561296	-0.36871	-0.192011
ASH	0.211545	0.164045	1	0.443367	0.28658	0.12898	0.115077	0.18623	0.009652	0.258887	-0.074667	0.003911	0.223626

ALCALINITY_OF_ASH	-0.310235	0.2885	0.443367	1	-0.0833333	-0.321113	-0.35137	0.361922	-0.197327	0.018732	-0.273955	-0.276769	-0.440597
MAGNESIUM	0.270798	-0.054575	0.286587	-0.083333	1	0.214401	0.195784	-0.256294	0.236441	0.19995	0.055398	0.066004	0.393351
TOTAL_PHENOLS	0.289101	-0.335167	0.128988	-0.321113	0.214401	1	0.864564	-0.449935	0.612413	-0.055136	0.433681	0.699949	0.498115
FLAVANOIDS	0.236815	-0.411007	0.115077	-0.35137	0.195784	0.864564	1	-0.5379	0.652692	-0.172379	0.543479	0.787194	0.494193
NONFLAVANOID_PHENOLS	-0.155929	0.29297	0.18623	0.361922	-0.256294	-0.449935	-0.5379	1	-0.365845	0.139057	-0.26264	-0.503270	-0.311385
PROANTHOCYANINS	0.136698	-0.220746	0.009652	-0.197327	0.236441	0.612413	0.652692	-0.365845	1	-0.025255	0.295544	0.519067	0.330417
COLOR_INTENSITY	0.546364	0.248985	0.258887	0.018732	0.19995	-0.055136	-0.172379	0.139057	-0.025255	1	-0.521813	-0.428815	0.316100
HUE	-0.071747	-0.561296	-0.074667	-0.273955	0.055398	0.433681	0.543479	-0.26264	0.295544	-0.521813	1	0.565468	0.236183
OD280/OD315_OF_DILUTED_WINES	0.072343	-0.36871	0.003911	-0.276769	0.066004	0.699949	0.787194	-0.503270	0.519067	-0.428815	0.565468	1	0.312761
PROLINE	0.643772	-0.192011	0.223626	-0.440597	0.393351	0.498115	0.494193	-0.311385	0.330417	0.316100	0.236183	0.312761	1

3:

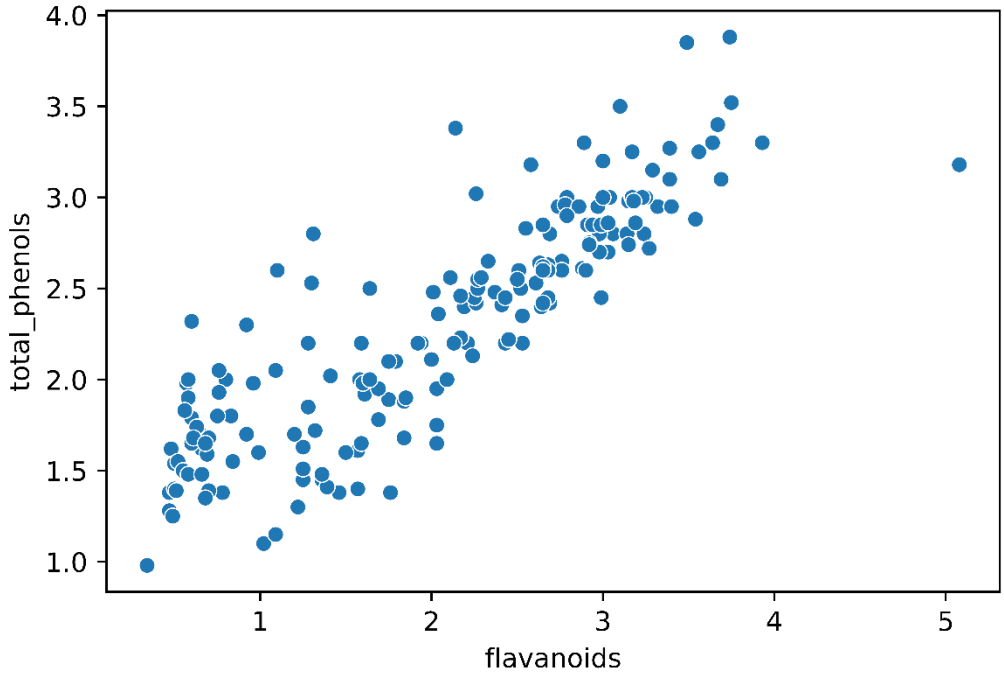
```
ax = sns.boxplot(x='target', y='alcohol', data=df)
plt.title('Boxplot: Target and Alcohol')

import textwrap
ax.set_xticklabels([textwrap.fill(t.get_text(), 10)  for t in ax.get_xticklabels()])

plt.show()
```



4:

Total phenols and flavonoids

```
sns.scatterplot(x='flavanoids', y='total_phenols', data=df)
plt.show()
```



5:

```
from sklearn.preprocessing import StandardScaler
from sklearn.manifold import MDS
sns.color_palette("rocket", as_cmap=True)
df_sorted = df.sort_values(by='target', ascending=True)
target_sorted = df_sorted['target']
```

```
X = df_sorted.drop(columns='target').to_numpy()
scaler = StandardScaler()
X = scaler.fit_transform(X)

embedding = MDS(n_components=2)

Xp = embedding.fit_transform(X)
df_projection = pd.DataFrame({'x': Xp[:, 0], 'y': Xp[:, 1],
                              'target': target_sorted})

sns.scatterplot(x='x', y='y', hue='target', data=df_projection, palette="viridis")

plt.legend(title='Target', bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)

plt.show()
```