Практическая работа 5.

Лазарев Александр. КМБО-03-22.

Вариант 16.

Установка данных.

```
import pandas as pd

# Загрузка данных
data = pd.read_csv('winequality-red.csv')
features = data.columns.tolist() # Список имен признаков
data.head() # Ознакомление с данными (первые 5 строк)
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar
chlorides  \
0            7.4              0.70         0.00             1.9
0.076
1            7.8              0.88         0.00             2.6
0.098
2            7.8              0.76         0.04             2.3
0.092
3           11.2              0.28         0.56             1.9
0.075
4            7.4              0.70         0.00             1.9
0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates
\
0                 11.0                  34.0   0.9978  3.51       0.56

1                 25.0                  67.0   0.9968  3.20       0.68

2                 15.0                  54.0   0.9970  3.26       0.65

3                 17.0                  60.0   0.9980  3.16       0.58

4                 11.0                  34.0   0.9978  3.51       0.56


   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
```

Данные установлены корректно.

```
print("Строки \ столбцы")
print(*data.shape, sep=' \ ')
```

```
Строки \ столбцы
1599 \ 12
```

```
print("Все признаки + их число уникальных значений:", *[{i :
data[i].nunique()} for i in data], sep='\n')
```

```
Все признаки + их число уникальных значений:
{'fixed acidity': 96}
{'volatile acidity': 143}
{'citric acid': 80}
{'residual sugar': 91}
{'chlorides': 153}
{'free sulfur dioxide': 60}
{'total sulfur dioxide': 144}
{'density': 436}
{'pH': 89}
{'sulphates': 96}
{'alcohol': 65}
{'quality': 6}
```

В наборе данных 1599 объектов и 12 признаков. Описание каждого признака:

1. Fixed acidity: Фиксированная кислотность
   - Большинство кислот, присутствующих в вине, являются фиксированными или неволатильными (не испаряются легко).

2. Volatile acidity: Летучая кислотность
   - Количество уксусной кислоты в вине, которая при слишком высоких уровнях может привести к неприятному вкусу уксуса.

3. Citric acid: Лимонная кислота
   - Находится в небольших количествах, может добавлять "свежесть" и аромат в вина.

4. Residual sugar: Остаточный сахар
   - Количество сахара, оставшегося после окончания ферментации; редко встречаются вина с менее чем 1 граммом/литром, а вина с более чем 45 граммами/литром считаются сладкими.

5. Chlorides: Хлориды
   - Количество соли в вине.

6. Free sulfur dioxide: Свободный диоксид серы

- Свободная форма SO2 находится в равновесии между молекулярным SO2 (в виде растворенного газа) и ионом бисульфита; предотвращает размножение микроорганизмов и окисление вина.

7. Total sulfur dioxide: Общий диоксид серы
   - Количество свободной и связанной формы SO2; при низких концентрациях SO2 в вине практически не обнаруживается, но при концентрациях свободного SO2 свыше 50 ppm, SO2 становится заметным по запаху и вкусу вина.

8. Density: Плотность
   - Плотность воды близка к плотности вина, в зависимости от содержания процента алкоголя и сахара.

9. pH: Уровень pH
   - Описывает кислотность или щелочность вина на шкале от 0 (очень кислотное) до 14 (очень щелочное); большинство вин находятся в диапазоне 3-4 на шкале pH.

10. Sulphates: Сульфаты
    - Добавка к вину, которая может способствовать уровню диоксида серы (SO2), который действует как антимикробное и антиоксидантное вещество.

11. Alcohol: Алкоголь
    - процентное содержание алкоголя в вине.

12. Quality: Качесто
    - выходная переменная (на основе сенсорных данных, оценка от 0 до 10).

```
data.info() # Проверим форматы признаков
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
```

```
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Заметим, что все признаки являются числовыми, следовательно категориальных и бинарных признаков в датафрейме нет.

```
data.isnull().sum() # Проверяем количество пропущенных элементов
```

```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

Пропущенные значения отсутствуют.

```
data.describe()
```

```
       fixed acidity  volatile acidity  citric acid  residual sugar  \
count    1599.000000       1599.000000  1599.000000     1599.000000
mean        8.319637          0.527821     0.270976        2.538806
std         1.741096          0.179060     0.194801        1.409928
min         4.600000          0.120000     0.000000        0.900000
25%         7.100000          0.390000     0.090000        1.900000
50%         7.900000          0.520000     0.260000        2.200000
75%         9.200000          0.640000     0.420000        2.600000
max        15.900000          1.580000     1.000000       15.500000

         chlorides  free sulfur dioxide  total sulfur dioxide
density  \
count  1599.000000          1599.000000           1599.000000
1599.000000
mean      0.087467            15.874922             46.467792
0.996747
std       0.047065            10.460157             32.895324
0.001887
min       0.012000             1.000000              6.000000
0.990070
```

```
25%       0.070000             7.000000             22.000000
0.995600
50%       0.079000            14.000000             38.000000
0.996750
75%       0.090000            21.000000             62.000000
0.997835
max       0.611000            72.000000            289.000000
1.003690
```

```
                pH    sulphates      alcohol      quality
count  1599.000000  1599.000000  1599.000000  1599.000000
mean      3.311113     0.658149    10.422983     5.636023
std       0.154386     0.169507     1.065668     0.807569
min       2.740000     0.330000     8.400000     3.000000
25%       3.210000     0.550000     9.500000     5.000000
50%       3.310000     0.620000    10.200000     6.000000
75%       3.400000     0.730000    11.100000     6.000000
max       4.010000     2.000000    14.900000     8.000000
```

Из данного ознакомления, а так же из описания, приведенного к датафрему, видно, что данные были специально отобранны для анализа и задач машинного обучения, следовательно в первом приближении факт наличия выбросов можно исключить.

Очевидно, что целевым признаком в этом набором служит quality (конечная оценка качества вина). Его сделаем бинарной переменной (больше 7 - вино выского качества, значение 1, в противном слече вино не высокого качетва, значение 0). Осталные признаки нормализуем через стандартное отклонение.

```
import numpy as np

# нормировка признаков через стандартное отклонение
data = (data - np.mean(data, axis=0)) / np.std(data, axis=0)

#отделение целевого признака
target = data['quality']

# результат
data.head()
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar
chlorides  \
0      -0.528360          0.961877    -1.391472       -0.453218  -
0.243707
1      -0.298547          1.967442    -1.391472        0.043416
0.223875
2      -0.298547          1.297065    -1.186070       -0.169427
0.096353
3       1.654856         -1.384443     1.484154       -0.453218  -
```

```
0.264960
4      -0.528360          0.961877    -1.391472       -0.453218  -
0.243707

   free sulfur dioxide  total sulfur dioxide   density        pH
sulphates  \
0            -0.466193             -0.379133  0.558274  1.288643  -
0.579207
1             0.872638              0.624363  0.028261 -0.719933
0.128950
2            -0.083669              0.229047  0.134264 -0.331177  -
0.048089
3             0.107592              0.411500  0.664277 -0.979104  -
0.461180
4            -0.466193             -0.379133  0.558274  1.288643  -
0.579207

    alcohol   quality
0 -0.960246 -0.787823
1 -0.584777 -0.787823
2 -0.584777 -0.787823
3 -0.584777  0.450848
4 -0.960246 -0.787823
```

```
# определение номера столбца, соответствующего максимальному среднему
значению
print("Столбец с максимальным средним значением после нормировки через
стандартное отклонение: ", np.argmax(np.mean(data, axis=0)))
```

```
Столбец с максимальным средним значением после нормировки через
стандартное отклонение:  7
```

Таким образом столбец density (плотность) имеет максимальное среднее значение.

Разделим набор данных на обучающую и тестовую выборки. Для этого используем функцию train_test_split() из библиотеки scikit-learn:

```
from sklearn.model_selection import train_test_split

# Удаление целевого признака из данных
data.drop(columns=['quality'], inplace=True)

# Разделение на обучающую и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split(data, target,
test_size=0.3, random_state=42)
```

Если при использовании train_test_split с параметрами test_size = 0.3, random_state = 42, в тренировочную выборку попадает 70% от всех объектов, то есть в неё попадёт 1119 объектов.

Между какими признаками наблюдается линейная зависимость (корреляция)?

Для вычисления коэффициента корреляции между каждой парой признаков можно воспользоваться функцией corr() в pandas. Результатом будет матрица корреляций, в которой каждый элемент показывает коэффициент корреляции между соответствующими признаками. Можно визуализировать матрицу корреляций в виде тепловой карты с помощью библиотеки seaborn.

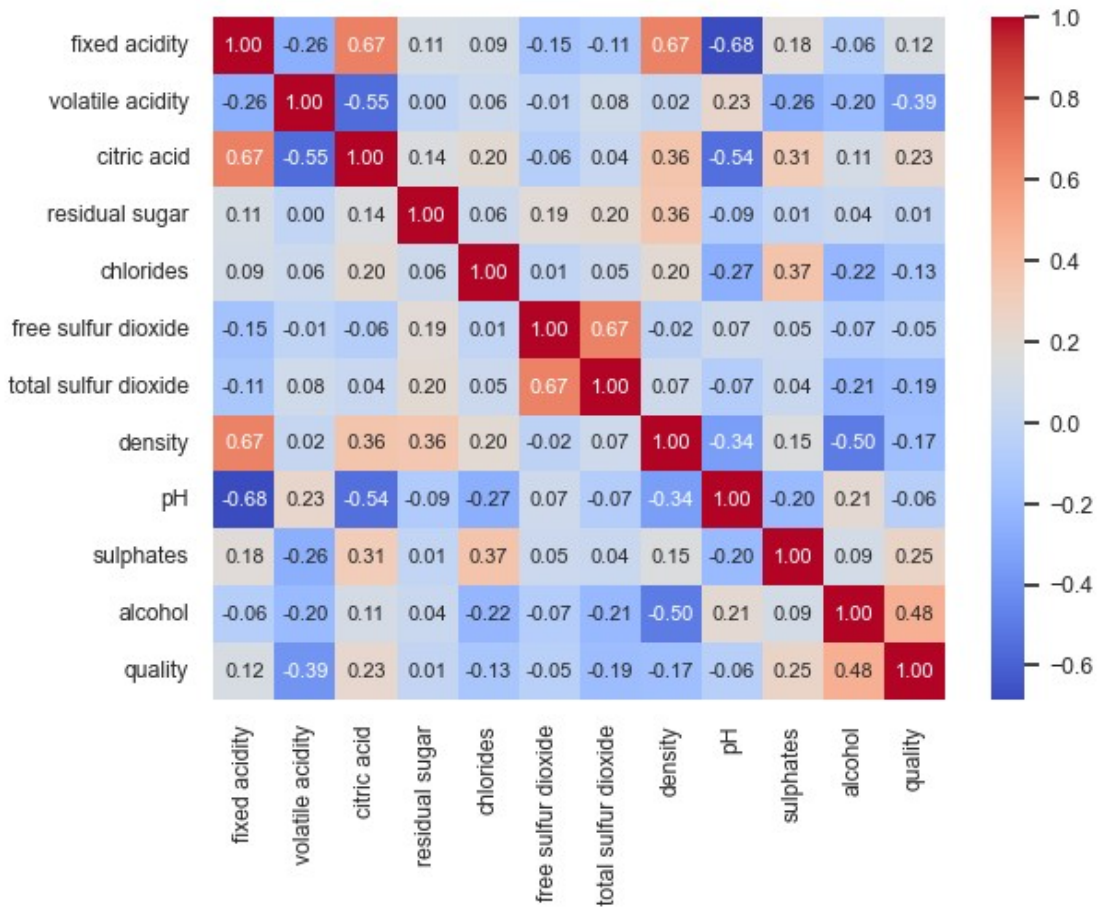
```
import seaborn as sns

# загружаем данные заново, чтобы проследить все изначальные корелляции
df = pd.read_csv('winequality-red.csv')

# находим матрицу корреляции
corr_matrix = df.corr()

# отображаем её визуально с помощью тепловой карты
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt='.2f',
annot_kws={"size": 8})

# настройка размера шрифта
sns.set(font_scale=0.8)
```




Чтобы определить, какая корреляция находится между парами признаков, можно использовать следующее правило:

Если значение корреляции находится между -1 и -0.7 или между 0.7 и 1, то это указывает на сильную отрицательную или положительную корреляцию соответственно.

Значения корреляции между -0.7 и -0.3 или между 0.3 и 0.7 указывают на умеренную отрицательную или положительную корреляцию соответственно.

Если значение корреляции находится между -0.3 и 0.3, то это указывает на отсутствие корреляции между признаками.

Таким образом, делаем следующие выводы:

1. Сильные линейные зависимости отсутствуют.
2. Между (fixed acidity и pH) (volatile acidity и citric acid) (volatile acidity и quality) (citric acid и pH) (density и pH) (density и alcohol) имеются слабые отрицательные линейные зависимости.

3. Между (fixed acidity и citric acid) (fixed acidity и density) (citric acid и density) (citric acid и sulphates) (residual sugar и density) (chlorides и sulphates) (free sulfur dioxide и total sulfur dioxide) (alcohol и quality) имеются слабые положительные линейные зависимости.

4. Между остальными парами признаков линейные зависимости отсутствуют.

Для определения количества признаков, достаточных для объяснения 90% дисперсии после применения метода главных компонент (PCA), можно использовать кумулятивную сумму долей объясненной дисперсии.

После применения PCA можно получить массив, содержащий долю объясненной дисперсии для каждой главной компоненты. Эти доли могут быть упорядочены по убыванию. Тогда можно вычислить кумулятивную сумму долей объясненной дисперсии по мере увеличения количества главных компонент и выбрать минимальное количество главных компонент, при котором кумулятивная сумма достигнет или превысит 90%.

```
from sklearn.decomposition import PCA

# Создание экземпляра PCA и подгонка модели
pca = PCA()
pca.fit(X_train, y_train)

# Получение массива долей объясненной дисперсии
variance_ratio = pca.explained_variance_ratio_

# Вычисление кумулятивной суммы долей объясненной дисперсии
cumulative_variance_ratio = np.cumsum(variance_ratio)
print(cumulative_variance_ratio)

# Находим индекс первого элемента в cumulative_variance_ratio, который
больше или равен 0.9 (90%)
n_components = np.argmax(cumulative_variance_ratio >= 0.9) + 1

# Вывод результата
print("Для объяснения 90% дисперсии необходимо использовать {}
компонент".format(n_components))
```

```
[0.27678308 0.44944412 0.58894397 0.70028474 0.79243323 0.85373637
 0.90702241 0.94651269 0.9779945  0.99438229 1.        ]
Для объяснения 90% дисперсии необходимо использовать 7 компонент
```

Чтобы понять, какой признак вносит наибольший вклад в первую компоненту, можно использовать атрибут components_ объекта PCA. Каждая строка этого массива соответствует главной компоненте, а каждый столбец соответствует признаку.

Таким образом, чтобы определить, какой признак вносит наибольший вклад в первую компоненту, нужно найти максимальное значение по модулю в первой строке массива components_

```
# Определяем, какой признак вносит наибольший вклад в первую
компоненту
max_feature_idx = np.argmax(np.abs(pca.components_[0]))

print(f"Первая компонента наиболее сильно зависит от признака
{features[max_feature_idx]}")
```

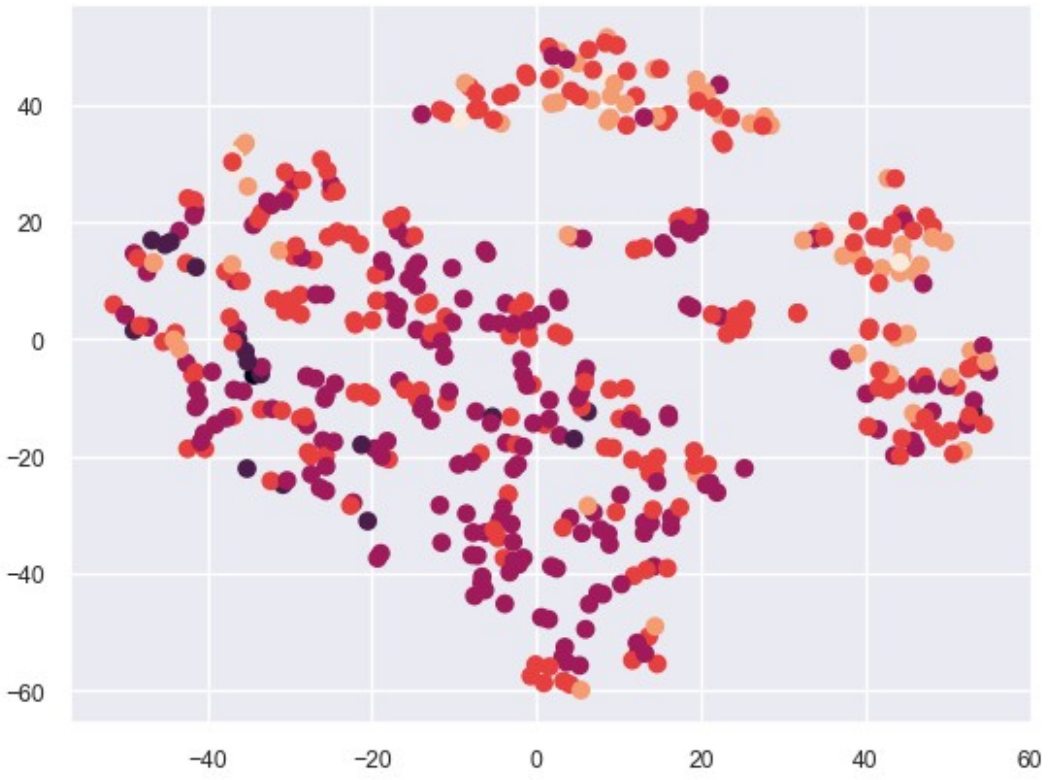
```
Первая компонента наиболее сильно зависит от признака fixed acidity
```

Для построения двухмерного представления данных с помощью алгоритма t-SNE можно воспользоваться библиотекой scikit-learn:

```
from sklearn.manifold import TSNE
import matplotlib.pyplot as plt
tsne = TSNE(perplexity=10, early_exaggeration=20, learning_rate=200,
init='pca', random_state=42)
X_tsne = tsne.fit_transform(X_test)
plt.scatter(X_tsne[:, 0], X_tsne[:, 1], c=y_test)
```

```
c:\Users\elaza\anaconda3\lib\site-packages\sklearn\manifold\
_t_sne.py:982: FutureWarning: The PCA initialization in TSNE will
change to have the standard deviation of PC1 equal to 1e-4 in 1.2.
This will ensure better convergence.
  warnings.warn(

<matplotlib.collections.PathCollection at 0x23979ab55e0>
```




Отчетливо видны, как минимум, 4 кластера. Два из них очевидны - черный (плохое вино, его больше) и стоящий от него на значительном расстоянии зеленый (хорошее вино). Другие два кластера, как по мне, демонстрируют неожиданно плохое и неожиданно хорошее вино (фиолетовый - неожиданно плохое, синий - неожиданно хорошее). То есть те вина, которые, несмотря на показатели, получили оценку, противоположную присущей данным показателям.

Alt text

Здесь кластеры – это группы объектов, которые имеют схожие признаковые описания. Кластеризация может помочь выявить скрытые закономерности в данных, облегчить понимание данных и дать возможность сделать выводы о взаимосвязях между признаками и объектами. Визуально выделяющиеся кластеры могут свидетельствовать о наличии существенных различий между группами объектов и помочь в дальнейшем анализе данных.