Задание 1.
Исходные данные:
Импортируйте библиотеки pandas, numpy и matplotlib.
Загрузите "Boston House Prices dataset" из встроенных наборов
данных библиотеки sklearn.
Создайте датафреймы X и y из этих данных.
Разбейте эти датафреймы на тренировочные (X_train, y_train) и тестовые (X_test, y_test)
с помощью функции train_test_split так, чтобы размер тестовой выборки
составлял 20% от всех данных, при этом аргумент random_state должен быть равен 42.
Масштабируйте данные с помощью StandardScaler.
Постройте модель TSNE на тренировочный данных с параметрами:
n_components=2, learning_rate=250, random_state=42.
Постройте диаграмму рассеяния на этих данных.
Решение:

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> import pandas as pd
>>> import matplotlib.pyplot as plt
>>> import sklearn
>>> from sklearn.datasets import load_boston
>>> city_boston = load_boston()
>>> city_boston.keys()
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
>>> data = city_boston["data"]
>>> data.shape
(506, 13)
>>> feature_names = city_boston["feature_names"]
>>> feature_names
array(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD',
       'TAX', 'PTRATIO', 'B', 'LSTAT'], dtype='<U7')
>>> target = city_boston["target"]
>>> target[:10]
array([24. , 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 27.1, 16.5, 18.9])
>>> X = pd.DataFrame(data, columns=feature_names)
>>> X.head()
      CRIM    ZN  INDUS  CHAS    NOX     RM   AGE     DIS  RAD    TAX  PTRATIO       B
LSTAT
0  0.00632  18.0   2.31   0.0  0.538  6.575  65.2  4.0900  1.0  296.0     15.3  396.90   4.98
1  0.02731   0.0   7.07   0.0  0.469  6.421  78.9  4.9671  2.0  242.0     17.8  396.90   9.14
2  0.02729   0.0   7.07   0.0  0.469  7.185  61.1  4.9671  2.0  242.0     17.8  392.83   4.03
3  0.03237   0.0   2.18   0.0  0.458  6.998  45.8  6.0622  3.0  222.0     18.7  394.63   2.94
4  0.06905   0.0   2.18   0.0  0.458  7.147  54.2  6.0622  3.0  222.0     18.7  396.90   5.33
>>> X.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 13 columns):
CRIM       506 non-null float64
ZN         506 non-null float64
INDUS      506 non-null float64
CHAS       506 non-null float64
```

```
NOX        506 non-null float64
RM         506 non-null float64
AGE        506 non-null float64
DIS        506 non-null float64
RAD        506 non-null float64
TAX        506 non-null float64
PTRATIO    506 non-null float64
B          506 non-null float64
LSTAT      506 non-null float64
dtypes: float64(13)
memory usage: 51.5 KB
>>> y = pd.DataFrame(target, columns=["price"])
>>> y.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 1 columns):
price    506 non-null float64
dtypes: float64(1)
memory usage: 4.1 KB
>>> from sklearn.model_selection import train_test_split
>>> X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
>>> from sklearn.preprocessing import StandardScaler
>>> scaler = StandardScaler()
>>> scaler.fit(X_train, y_train)
StandardScaler(copy=True, with_mean=True, with_std=True)
>>> scaler.transform(X_train)
array([[ 1.28770177, -0.50032012,  1.03323679, ...,  0.84534281,
        -0.07433689,  1.75350503],
       [-0.33638447, -0.50032012, -0.41315956, ...,  1.20474139,
         0.4301838 , -0.5614742 ],
       [-0.40325332,  1.01327135, -0.71521823, ..., -0.63717631,
         0.06529747, -0.65159505],
       ...,
       [-0.40547014,  2.95931752, -1.30336132, ..., -0.59225149,
         0.37901005, -0.91069248],
       [ 0.85189733, -0.50032012,  1.03323679, ...,  0.84534281,
        -2.69458597,  1.52257036],
       [-0.38135592, -0.50032012, -0.35216694, ...,  1.15981657,
        -3.12158061, -0.25731635]])
>>> from sklearn.manifold import TSNE
>>> model = TSNE(n_components=2, learning_rate=250, random_state=42)
>>> transformed = model.fit_transform(X_train)
>>> X_axis = transformed[:, 0]
>>> y_axis = transformed[:, 1]
>>> plt.scatter(X_axis, y_axis)
<matplotlib.collections.PathCollection object at 0x7ff9c1ee94f0>
>>> plt.show()
>>>
```

Задание 2.

Исходные данные:

С помощью KMeans разбейте данные из тренировочного набора на 3 кластера,

используйте все признаки из датафрейма X_train.
Параметр max_iter должен быть равен 100, random_state сделайте равным 42.
Постройте еще раз диаграмму рассеяния на данных, полученных с помощью TSNE,
и раскрасьте точки из разных кластеров разными цветами.
Вычислите средние значения price и CRIM в разных кластерах.
Решение:

```
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> import pandas as pd
>>> import matplotlib.pyplot as plt
>>> import sklearn
>>> from sklearn.datasets import load_boston
>>> city_boston = load_boston()
>>> city_boston.keys()
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
>>> data = city_boston["data"]
>>> data.shape
(506, 13)
>>> feature_names = city_boston["feature_names"]
>>> feature_names
array(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NOX', 'RM', 'AGE', 'DIS', 'RAD',
       'TAX', 'PTRATIO', 'B', 'LSTAT'], dtype='<U7')
>>> target = city_boston["target"]
>>> target[:10]
array([24. , 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 27.1, 16.5, 18.9])
>>> X = pd.DataFrame(data, columns=feature_names)
>>> X.head()
      CRIM    ZN  INDUS  CHAS    NOX     RM   AGE     DIS  RAD    TAX  PTRATIO       B
LSTAT
0  0.00632  18.0   2.31   0.0  0.538  6.575  65.2  4.0900  1.0  296.0     15.3  396.90   4.98
1  0.02731   0.0   7.07   0.0  0.469  6.421  78.9  4.9671  2.0  242.0     17.8  396.90   9.14
2  0.02729   0.0   7.07   0.0  0.469  7.185  61.1  4.9671  2.0  242.0     17.8  392.83   4.03
3  0.03237   0.0   2.18   0.0  0.458  6.998  45.8  6.0622  3.0  222.0     18.7  394.63   2.94
4  0.06905   0.0   2.18   0.0  0.458  7.147  54.2  6.0622  3.0  222.0     18.7  396.90   5.33
>>> X.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 13 columns):
CRIM       506 non-null float64
ZN         506 non-null float64
INDUS      506 non-null float64
CHAS       506 non-null float64
NOX        506 non-null float64
RM         506 non-null float64
AGE        506 non-null float64
DIS        506 non-null float64
RAD        506 non-null float64
TAX        506 non-null float64
PTRATIO    506 non-null float64
B          506 non-null float64
```

```
LSTAT      506 non-null float64
dtypes: float64(13)
memory usage: 51.5 KB
>>> y = pd.DataFrame(target, columns=["price"])
>>> y.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 1 columns):
price    506 non-null float64
dtypes: float64(1)
memory usage: 4.1 KB
>>> from sklearn.model_selection import train_test_split
>>> X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42)
>>> from sklearn.cluster import KMeans
>>> model = KMeans(n_clusters=3)
>>> model.fit(X_train)
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
>>> predicted_label = model.predict(X_train)
>>> all_predictions = model.predict(X_train)
>>> print(predicted_label)
[0 2 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 1 0 0 1 0 0 2 1 0 0 0 0 1 1 0 1 0 0 0 0
 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 1 1 1 0 0 2 0 1 0 1 0 0 1 1 0 0 0 1 0 0
 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1
 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 2 0 0 1 0 0 0 0 0 0 0 0 0 0
 1 0 0 2 1 0 0 0 0 1 0 0 2 0 0 1 1 1 0 0 1 2 0 0 0 0 0 1 2 0 0 0 0 0 0 0 0
 0 0 0 1 0 0 0 2 0 1 1 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0
 1 1 0 0 0 2 0 0 1 0 0 2 0 2 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 2 0 0 0 0 1
 0 0 1 0 0 0 2 0 1 1 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0
 1 0 1 1 0 2 0 0 0 2 0 2 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1
 0 2 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 2 0 0 0 0 0 0 1 0 1 0 1 0 0 2 0
 0 0 0 0 0 0 0 2 2]
>>> print(all_predictions)
[0 2 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 1 0 0 1 0 0 2 1 0 0 0 0 1 1 0 1 0 0 0 0
 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 1 1 1 0 0 2 0 1 0 1 0 0 1 1 0 0 0 1 0 0
 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1
 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 2 0 0 1 0 0 0 0 0 0 0 0 0 0
 1 0 0 2 1 0 0 0 0 1 0 0 2 0 0 1 1 1 0 0 1 2 0 0 0 0 0 1 2 0 0 0 0 0 0 0 0
 0 0 0 1 0 0 0 2 0 1 1 0 2 0 0 0 0 0 0 0 2 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0
 1 1 0 0 0 2 0 0 1 0 0 2 0 2 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 2 0 0 0 0 1
 0 0 1 0 0 0 2 0 1 1 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0
 1 0 1 1 0 2 0 0 0 2 0 2 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1
 0 2 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 2 0 0 0 0 0 0 1 0 1 0 1 0 0 2 0
 0 0 0 0 0 0 0 2 2]
>>> from sklearn.manifold import TSNE
>>> model = TSNE(n_components=2, learning_rate=250, random_state=42)
>>> transformed = model.fit_transform(X_train)
>>> X_axis = transformed[:, 0]
>>> y_axis = transformed[:, 1]
>>> plt.scatter(X_axis, y_axis)
<matplotlib.collections.PathCollection object at 0x7f8c38ccbe50>
```

```
>>> plt.show()
>>>
```