

In [48]:

```
#1. Импортируйте библиотеки pandas, numpy и matplotlib.  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt
```

In [49]:

```
#Загрузите "Boston House Prices dataset" из встроенных наборов данных библиотеки sklearn.  
from sklearn.datasets import load_boston  
boston = load_boston()  
boston.keys()
```

Out[49]:

```
dict_keys(['data', 'target', 'feature_names', 'DESCR', 'filename'])
```

In [50]:

```
#Создайте датафреймы X и y из этих данных.  
X = pd.DataFrame(boston.data, columns=boston.feature_names)  
y = boston.target  
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
```

In [52]:

```
#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.  
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)  
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[52]:

```
((404, 13), (102, 13), (404,), (102,))
```

In [53]:

```
#Масштабируйте данные с помощью StandardScaler.
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = pd.DataFrame(scaler.fit_transform(X_train), columns=X_train.columns)
X_test_scaled = pd.DataFrame(scaler.transform(X_test), columns=X_test.columns)
```

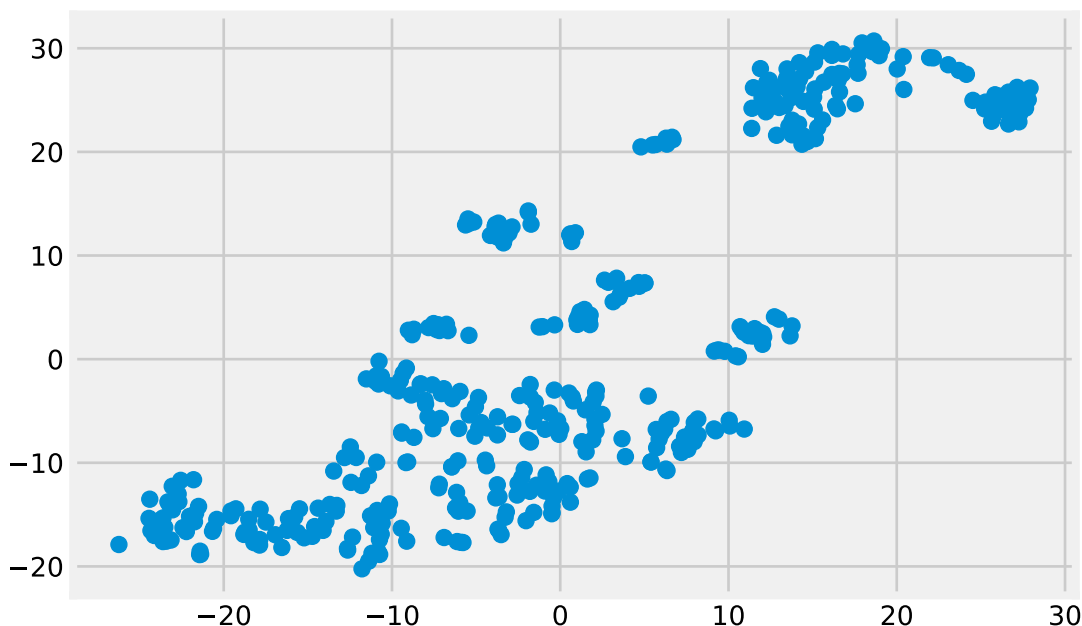
In [54]:

```
#Постройте модель TSNE на тренировочный данных с параметрами:
#n_components=2, learning_rate=250, random_state=42.
from sklearn.manifold import TSNE
tsne = TSNE(n_components=2, learning_rate=250, random_state=42)
X_train_tsne = tsne.fit_transform(X_train_scaled)
print('До:\t{}'.format(X_train_scaled.shape))
print('После:\t{}'.format(X_train_tsne.shape))
```

До: (404, 13)
После: (404, 2)

In [55]:

```
#Постройте диаграмму рассеяния на этих данных.
plt.style.use('fivethirtyeight')
%config InlineBackend.figure_format = 'svg'
%matplotlib inline
plt.scatter(X_train_tsne[:, 0], X_train_tsne[:, 1])
plt.show()
```

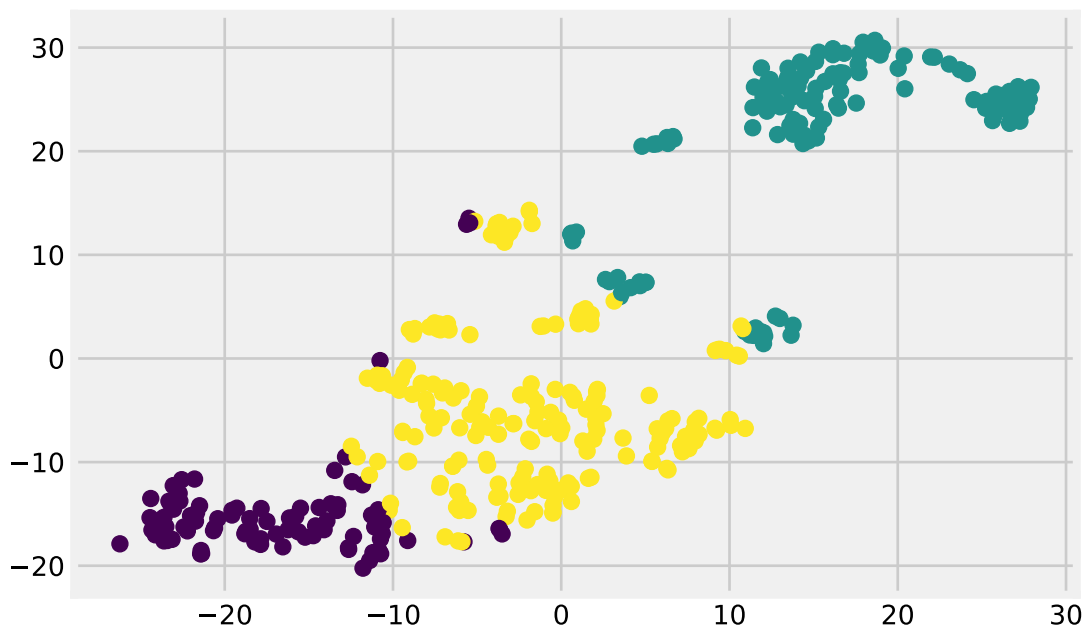


In [56]:

```
#2. 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.
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=3, max_iter=100, random_state=42)
labels_train = kmeans.fit_predict(X_train_scaled)
```

In [57]:

```
#Постройте еще раз диаграмму рассеяния на данных, полученных с помощью TSNE, и раскрасьте т
#из разных кластеров разными цветами.
plt.scatter(X_train_tsne[:, 0], X_train_tsne[:, 1], c=labels_train)
plt.show()
```



In [58]:

```
#Вычислите средние значения price и CRIM в разных кластерах.
print('Средние значения price:')
print('Кластер 0: {}'.format(y_train[labels_train == 0].mean()))
print('Кластер 1: {}'.format(y_train[labels_train == 1].mean()))
print('Кластер 2: {}'.format(y_train[labels_train == 2].mean()))
print('Средние значения CRIM:')
print('Кластер 0: {}'.format(X_train.loc[labels_train == 0, 'CRIM'].mean()))
print('Кластер 1: {}'.format(X_train.loc[labels_train == 1, 'CRIM'].mean()))
print('Кластер 2: {}'.format(X_train.loc[labels_train == 2, 'CRIM'].mean()))
```

Средние значения price:

Кластер 0: 27.78837209302326

Кластер 1: 16.165354330708663

Кластер 2: 24.958115183246072

Средние значения CRIM:

Кластер 0: 0.07356558139534886

Кластер 1: 10.797028425196853

Кластер 2: 0.42166020942408367

In [59]:

```
#3.Примените модель KMeans, построенную в предыдущем задании, к данным из тестового набора.
labels_test = kmeans.predict(X_test_scaled)
```

In [60]:

```
#Вычислите средние значения price и CRIM в разных кластерах на тестовых данных.
print('Средние значения price:')
print('Кластер 0: {}'.format(y_test[labels_test == 0].mean()))
print('Кластер 1: {}'.format(y_test[labels_test == 1].mean()))
print('Кластер 2: {}'.format(y_test[labels_test == 2].mean()))
print('Средние значения CRIM:')
print('Кластер 0: {}'.format(X_test.loc[labels_test == 0, 'CRIM'].mean()))
print('Кластер 1: {}'.format(X_test.loc[labels_test == 1, 'CRIM'].mean()))
print('Кластер 2: {}'.format(X_test.loc[labels_test == 2, 'CRIM'].mean()))
```

Средние значения price:

Кластер 0: 31.35

Кластер 1: 16.43714285714286

Кластер 2: 21.860784313725492

Средние значения CRIM:

Кластер 0: 0.062060000000000004

Кластер 1: 10.165531142857143

Кластер 2: 0.26607882352941176

In []: