26 задача: Кластеризация предобученных векторов методом K-means

Михаил Алексеевич Сысак

Московский физико-технический институт

29 апреля 2019 г.

- Выборка состоит из предобученных векторов fasttext.
- Количество объектов — 19924, количество признаков — 300.

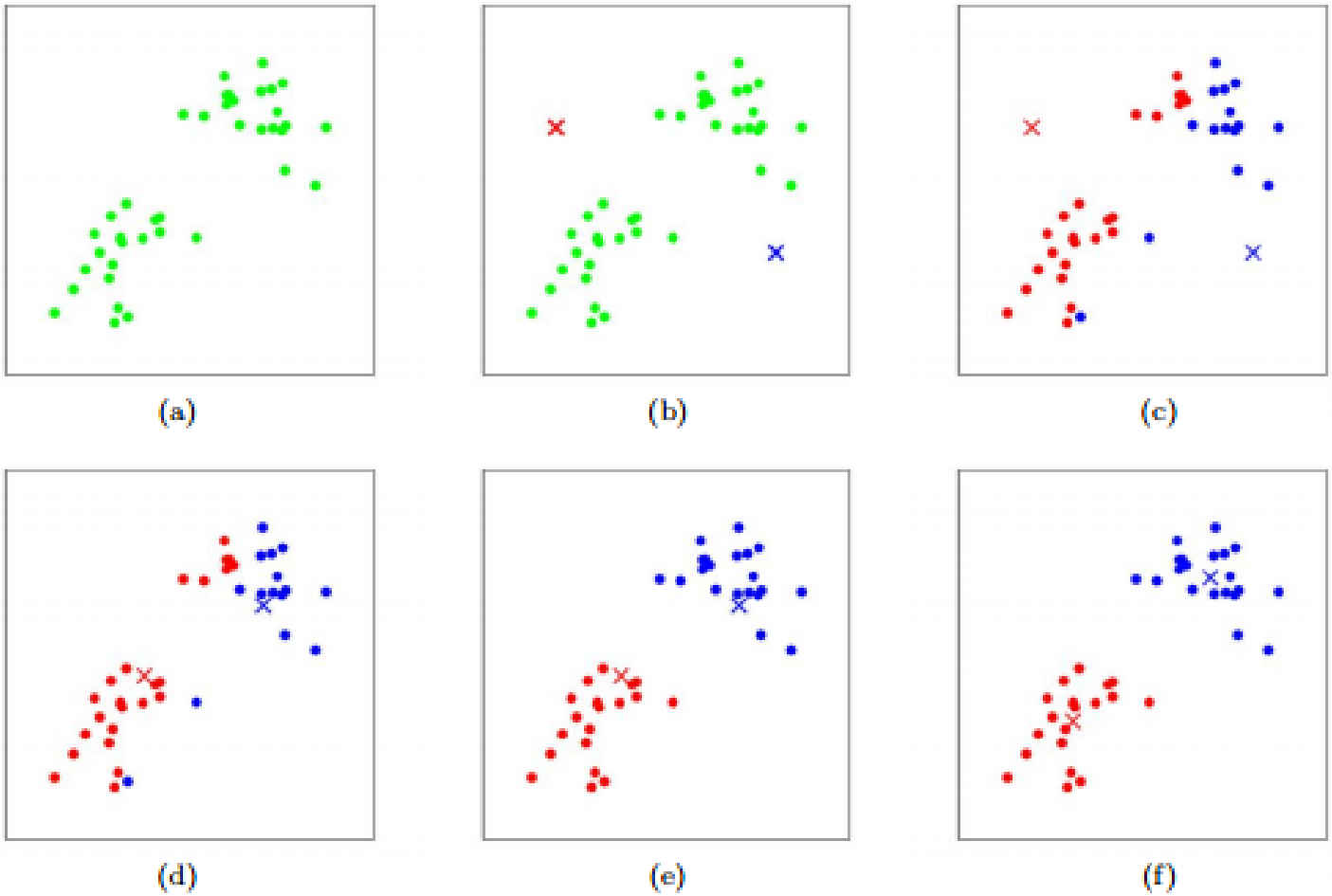
Метод k-средних:

1. Случайно выбирается k центроид.
2. Для каждого объекта кластер определяется ближайшей к нему центроидой.
3. В качестве новых центроид берутся центры масс полученных кластеров.
4. Шаги 2-3 повторяются фиксированное количество раз, или пока функционал качества не сойдется.

Используемые модификации:

- k-means++
- MiniBatchKMeans

Метод k-средних



Структурный параметр — k (количество кластеров) в четырех диапазонах:

- $2, 3, \ldots, 10$
- $10, 20, \ldots, 100$
- $100, 200, \ldots, 1000$
- $1000, 2000, \ldots, 10000$

Критерии качества:

- Сумма средних внутрикластерных расстояний Φ_0 и C_0
- Сумма межкластерных расстояний Φ_1 и C_1
- Силуэт s_e и s_c

Каждый критерий вычисляется с использованием евклидовой метрики и косинусного расстояния.

$$\Phi_0 = \sum_{y \in Y} \frac{1}{|K_y|} \sum_{i:y_i=y} \rho^2(x_i, \mu_y) \qquad \Phi_1 = \sum_{y \in Y} \rho^2(\mu_y, \mu)$$

```
def Euclidean_Internal_Dist(data, pred, model, k):
    ret = 0
    for y in range(k):
        summ = 0
        cnt = 0
        for i in range(19924):
            if y == pred[i]:
                cnt += 1
                summ += np.linalg.norm(data.values[i] - model.cluster_centers_[y]) ** 2
        if cnt > 0:
            ret += summ / cnt
    return ret
```

```
def Euclidean_External_Dist(data, pred, model, k):
    ret = 0
    mu = np.sum(data.values, axis = 0)/19924
    for y in range(k):
        ret += np.linalg.norm(model.cluster_centers_[y] - mu) ** 2
    return ret
```

Постановка задачи

Список структурных параметров и критериев качества

$$C_0 = \sum_{y \in Y} \frac{1}{|K_y|} \sum_{i:y_i=y} \left(1 - \frac{(x_i, \mu_y)}{||x_i|| \cdot ||\mu_y||}\right) \quad C_1 = \sum_{y \in Y} \left(1 - \frac{(\mu_y, \mu)}{||\mu_y|| \cdot ||\mu||}\right)$$

```
def Cosine_Internal_Dist(data, pred, model, k):
    ret = 0
    for y in range(k):
        summ = 0
        cnt = 0
        for i in range(19924):
            if y == pred[i]:
                cnt += 1
                summ += sc.spatial.distance.cosine(data.values[i], model.cluster_centers_[y])
        if cnt > 0:
            ret += summ / cnt
    return ret
```

```
def Cosine_External_Dist(data, pred, model, k):
    ret = 0
    mu = np.sum(data.values, axis = 0)/19924
    for y in range(k):
        ret += sc.spatial.distance.cosine(model.cluster_centers_[y], mu)
    return ret
```

$$s = \frac{\sum\limits_{x_i \in X} \frac{b_i - a_i}{\max(a_i, b_i)}}{\sum\limits_{y \in Y} |K_y|}$$

Величины a и b для элемента x кластера y:

$$a = \frac{1}{|K_y| - 1} \sum_{x' \in K_y, x' \neq x} d(x, x') \quad b = \min_{y' \neq y} \frac{1}{|K_{y'}|} \sum_{x' \in K_{y'}} d(x, x')$$

$d(i, j)$ определяется метрикой или функцией расстояния.

Критерий качества реализован в библиотеке scikit-learn.

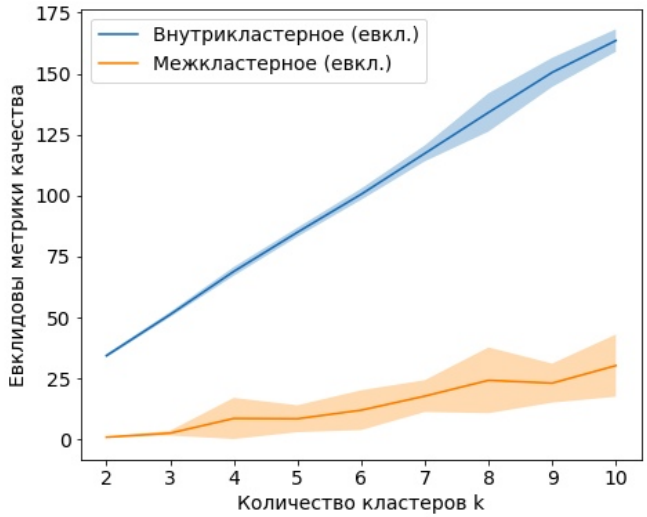
Вычислительный эксперимент
Код

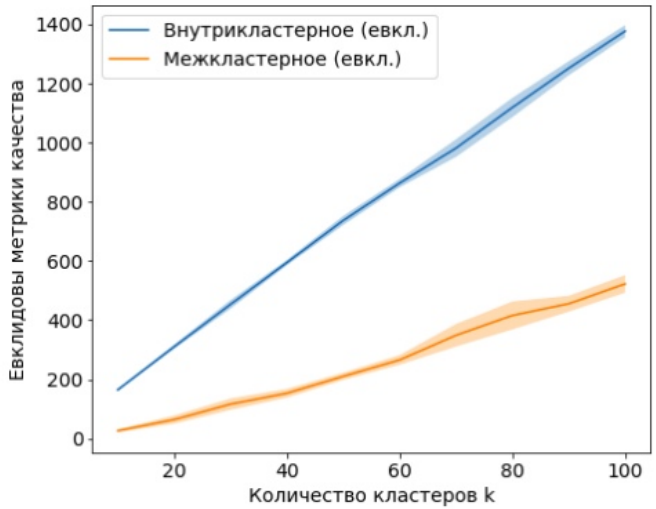
```
euc_int = np.array([[0. for i in range(10)] for i in range(9)])
euc_ext = np.array([[0. for i in range(10)] for i in range(9)])
cos_int = np.array([[0. for i in range(10)] for i in range(9)])
cos_ext = np.array([[0. for i in range(10)] for i in range(9)])
euc_sil = np.array([[0. for i in range(10)] for i in range(9)])
cos_sil = np.array([[0. for i in range(10)] for i in range(9)])

j = 0
for k in range(2, 11):
    print("k =", k)
    for i in range(10):
        print("i =", i)
        model = MiniBatchKMeans(k, batch_size = max(20 * k, 142))
        pred = model.fit_predict(data)
        cos_int[j][i] = Cosine_Internal_Dist(data, pred, model, k)
        cos_ext[j][i] = Cosine_External_Dist(data, pred, model, k)
        euc_ext[j][i] = Euclidian_External_Dist(data, pred, model, k)
        euc_int[j][i] = Euclidian_Internal_Dist(data, pred, model, k)
        euc_sil[j][i] = silhouette_score(data, pred)
        cos_sil[j][i] = silhouette_score(data, pred, metric='cosine')
    j += 1
```

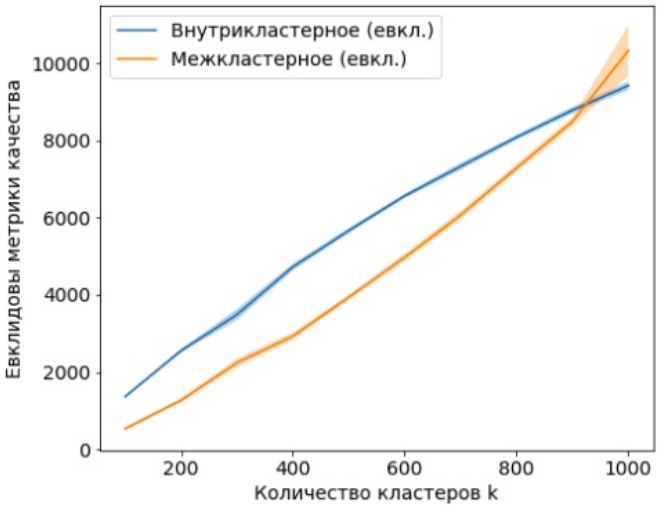
```
plt.figure(figsize = (16, 6))
plt.subplot(1, 2, 1)
plt.plot(range(2, 11), [np.mean(i) for i in euc_int])
plt.fill_between(range(2,11), [np.mean(i) - np.std(i) for i in euc_int],
                              [np.mean(i) + np.std(i) for i in euc_int], alpha=0.3)
plt.plot(range(2, 11), [np.mean(i) for i in euc_ext])
plt.fill_between(range(2,11), [np.mean(i) - np.std(i) for i in euc_ext],
                              [np.mean(i) + np.std(i) for i in euc_ext], alpha=0.3)
plt.xlabel("Количество кластеров k")
plt.ylabel("Евклидовы метрики качества")
plt.legend(["Внутрикластерное (евкл.)", "Межкластерное (евкл.)"])

plt.subplot(1, 2, 2)
plt.plot(range(2, 11), [np.mean(i) for i in cos_int], c = "g")
plt.fill_between(range(2,11), [np.mean(i) - np.std(i) for i in cos_int],
                              [np.mean(i) + np.std(i) for i in cos_int], alpha=0.3, color = "green")
plt.plot(range(2, 11), [np.mean(i) for i in cos_ext], c = "r")
plt.fill_between(range(2,11), [np.mean(i) - np.std(i) for i in cos_ext],
                              [np.mean(i) + np.std(i) for i in cos_ext], alpha=0.3, color = "red")
plt.xlabel("Количество кластеров k")
plt.ylabel("Косинусные метрики качества")
plt.legend(["Внутрикластерное (кос.)", "Межкластерное (кос.)"])
plt.show()
```

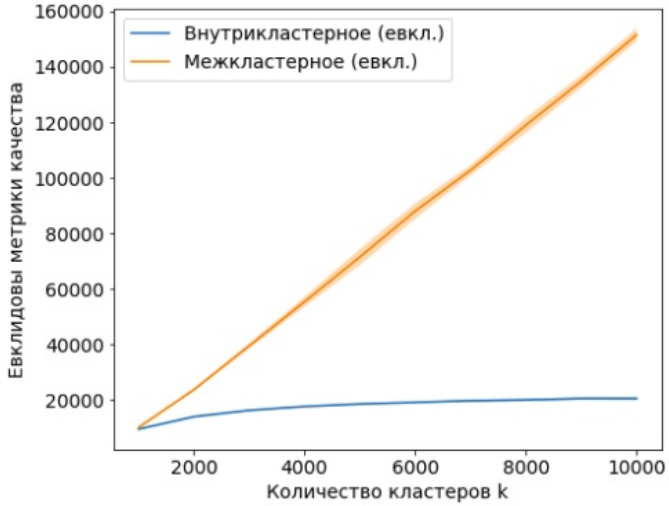
Вычислительный эксперимент

Графики критериев качества от структурного параметра





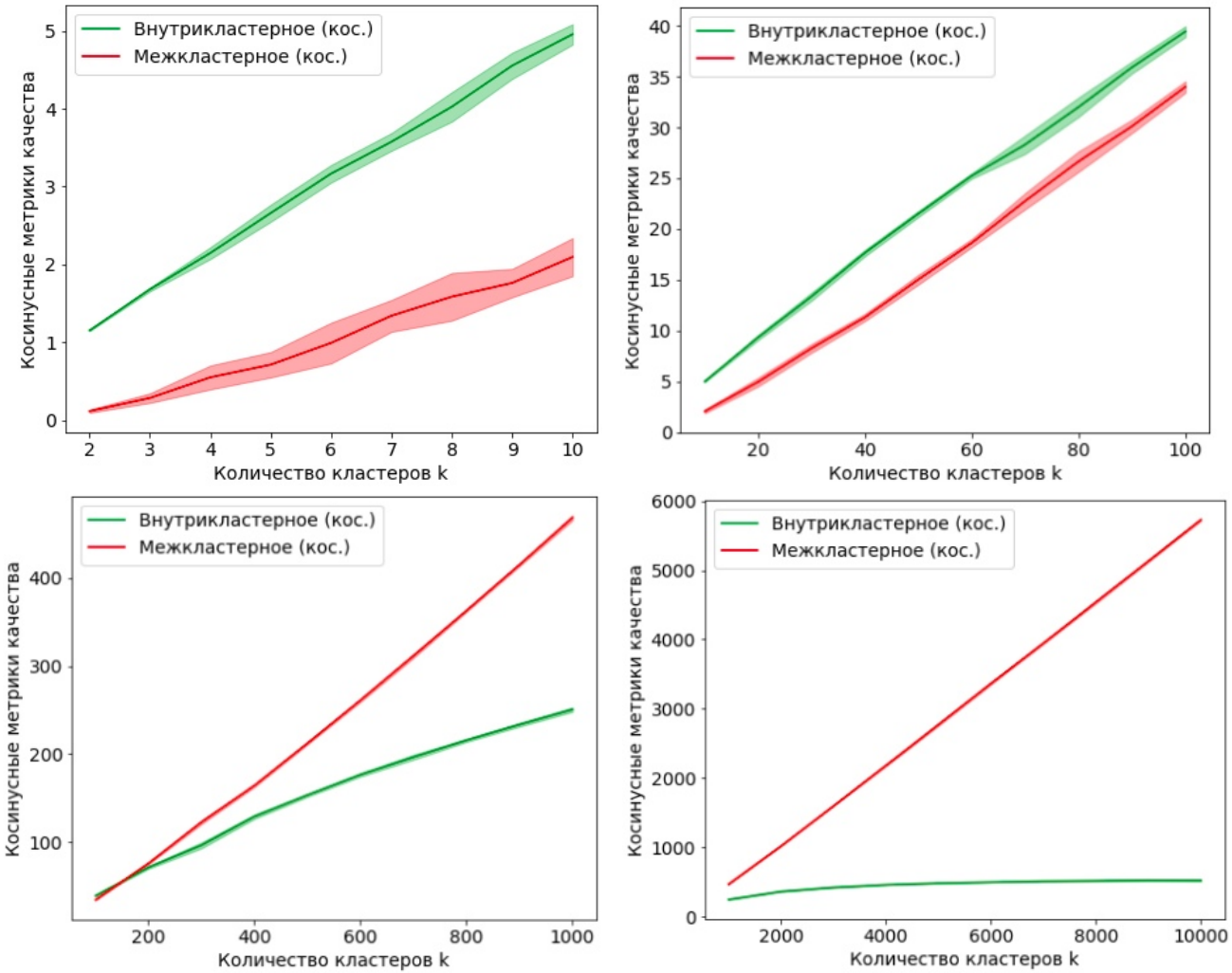




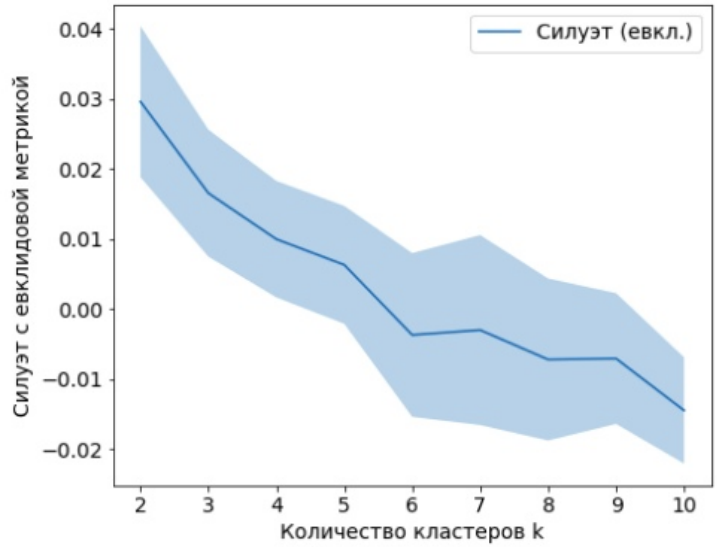
По отдельности критерии качества недостаточно информативны.

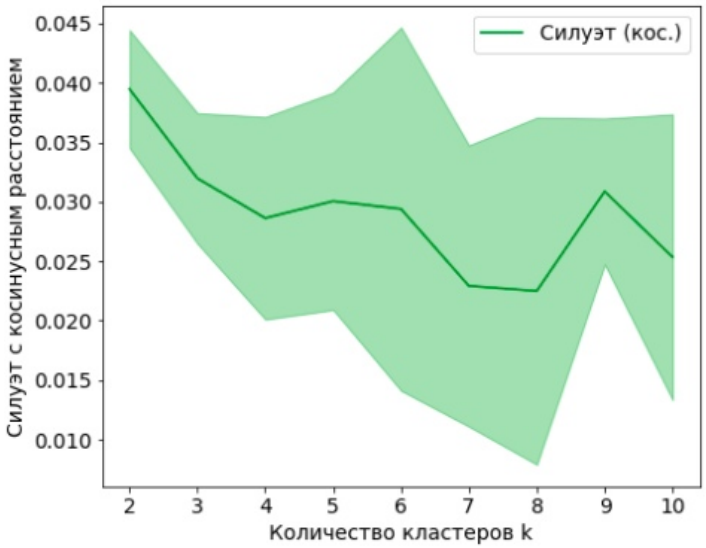
Вычислительный эксперимент

Графики критериев качества от структурного параметра



По отдельности критерии качества недостаточно информативны.



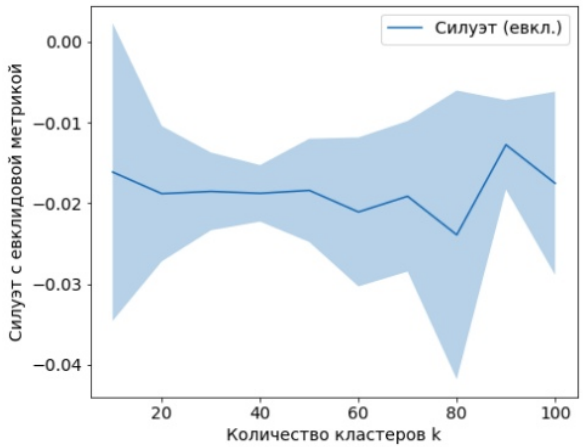


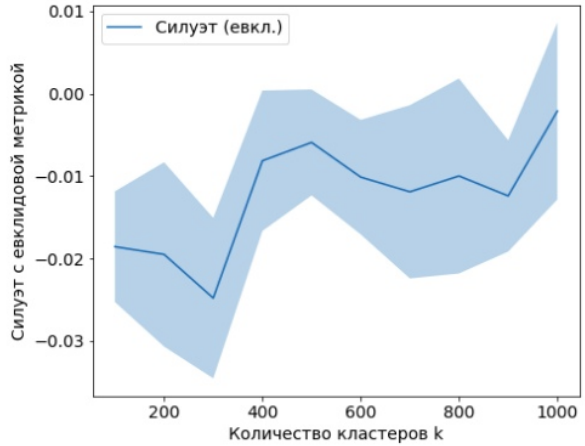
Начиная с $k = 6$, кластеризация перестает терять смысл с точки зрения «евклидова» силуэта, поскольку критерий становится отрицательным.

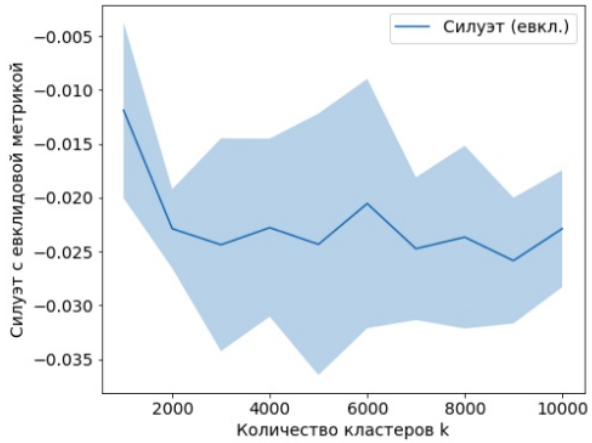
Близкие к нулю значения при $k < 6$ показывают, что в этом промежутке получаемые кластеры расположены на малых расстояниях друг от друга.

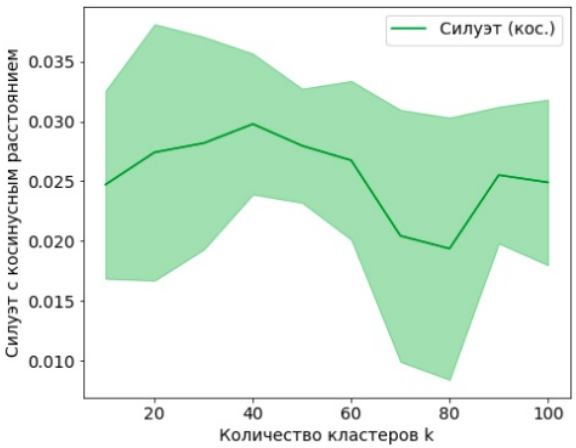
Вычислительный эксперимент

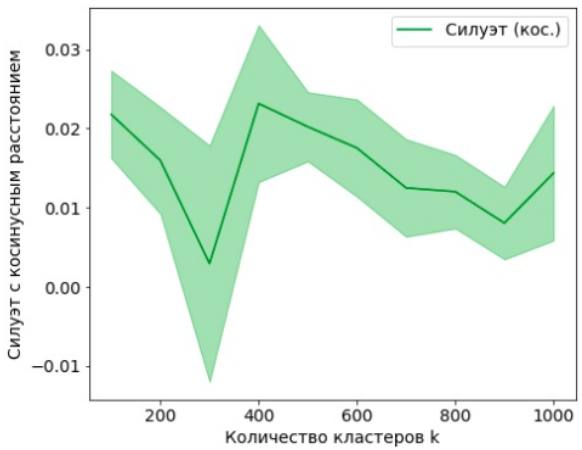
Графики критериев качества от структурного параметра

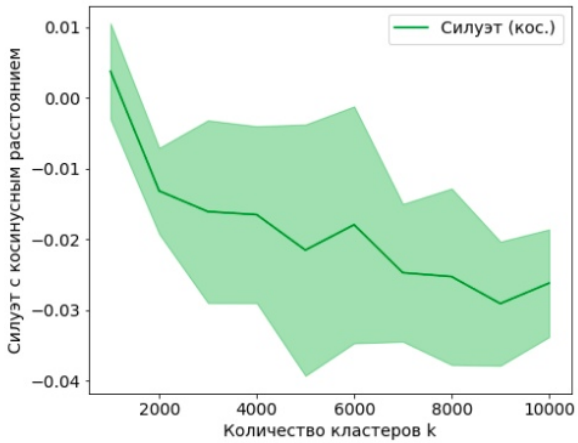












Кластеризация остается неэффективной для больших значений k.

Заключение

- При $k > 6$ кластеризация становится неэффективной.
- Для $k \leq 6$ кластеры имеют смысл.
- Разделение на $k = 2$ кластера дает лучший результат.
- Несмотря на это, малое значение силуэта показывает, что такое сочетание алгоритма и критериев качества плохо подходит для кластеризации данной выборки.