Кластеризация точек на плоскости

Баталов Семен

25.02.2021

1. Постановка задачи

Рассматривается плоскость, на которую случайным образом наносятся точки. Нужно решить задачу кластеризации методом «**K-Means++**». Требуется показать конечный результат работы кластеризатора и оценить расположение классов на этапе инициализации, сравнить результаты с оригинальным разбиением на классы.

Для написания программы использовался язык «**Python**», подробнее о ней можно узнать в папке «**source**» проекта.

2. K-Means и K-Means++

2.1. K-Means

Алгоритм «**K-Means**» разбивает множество элементов векторного пространства на заранее известное число кластеров «k». В нашем случае размерность пространства равна 2, а элементами являются точки, которые описываются только двумя параметрами: абсциссой и ординатой.

Основная идея заключается в том, что на каждой итерации перевычисляется центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике.

Алгоритм завершается, когда на какой-то итерации не происходит изменения внутрикластерного расстояния. Это происходит за конечное число итераций, так как количество возможных разбиений конечного множества конечно, а на каждом шаге суммарное квадратичное отклонение уменьшается, поэтому возникновение мертвого цикла невозможно.

Важным этапом в алгоритме является первоначальная инициализация центров классов. В «**K-Means**» центры выбираются случайно.

2.2. K-Means++

Алгоритм «**K-Means++**» ничем, кроме способа начальной инициализации центров, не отличается от «**K-Means**». В «**K-Means++**» центры выбираются (как правило) удаленными друг от друга, что с большей вероятностью приводит к лучшим результатам, чем случайная инициализация.

3. Инструменты

Для работы была выбрана библиотека «**sklearn**», из нее были взяты алгоритмы «**KMeans**» и «**kmeans_plusplus**». Первый из них осуществляет метод «**K-Means**», второй генерирует центры кластеров в соответствии с алгоритмом «**K-Means++**».

Для генерации случайных точек на плоскости использовались алгоритмы библиотеки «**sklearn**», а именно «**make_blobs**». Этот алгоритм создает многоклассовые наборы данных, выделяя каждому классу один или несколько нормально распределенных кластеров точек. «**make_blobs**» обеспечивает контроль относительно центров и стандартных отклонений каждого кластера и используется для тестирования алгоритмов кластеризации.

4. Результаты

В экспериментах варьировалось количество точек на плоскости, количество кластеров, коэффициенты, отвечающие за распределение точек по плоскости.

Основной задачей было показать конечный результат работы кластеризатора и в случае «**K-Means++**» оценить расположение классов, используя начальную инициализацию центров, сравнить результаты с оригинальным разбиением на классы.

В целом, кластеризаторы «**K-Means**» и «**K-Means++**» работают корректно. Стоит отметить, что начальная инициализация (центров кластеров) во втором алгоритме позволяет с самого начала равномернее распределить центры кластеров по множеству точек, во всех примерах это отчетливо прослеживается. Также видно, что результаты работы обоих кластеризаторов (для оределенного множества) слабо отличаются друг от друга, то есть конечные положения центров кластеров во многих примерах совпадают, а в остальных различие незначительно. Можно сделать вывод, что начальные условия (на положения центров кластеров) при данной постановке задачи если и оказывают влияние на конечный результат, то незначительное.

Важно отметить, что в некоторых примерах (Рис. 2), (Рис. 5) результаты работы алгоритмов не совпадали с истинным разбиением множества на кластеры. Это объясняется возникновением плотных областей пересечения нескольких кластеров, в которых сконцентрировано столько точек, что программа считает их отдельными классами. В то же время примеры (Рис. 1), (Рис. 3), (Рис. 4) демонстрируют корректную работу классификаторов. Все тестовые множества и результаты работы алгоритмов представлены ниже.

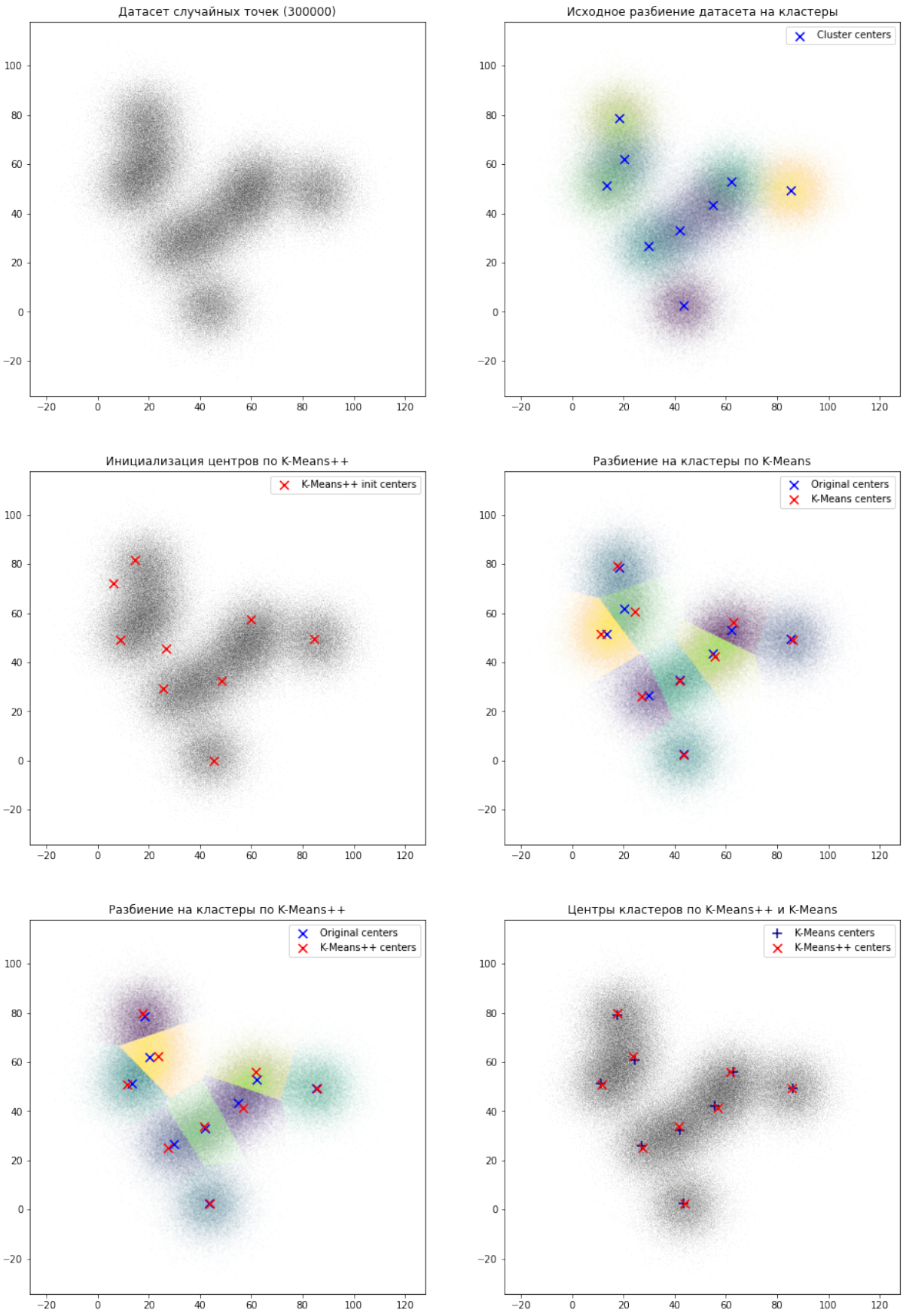

Рис. 1. 9 кластеров, 300000 точек

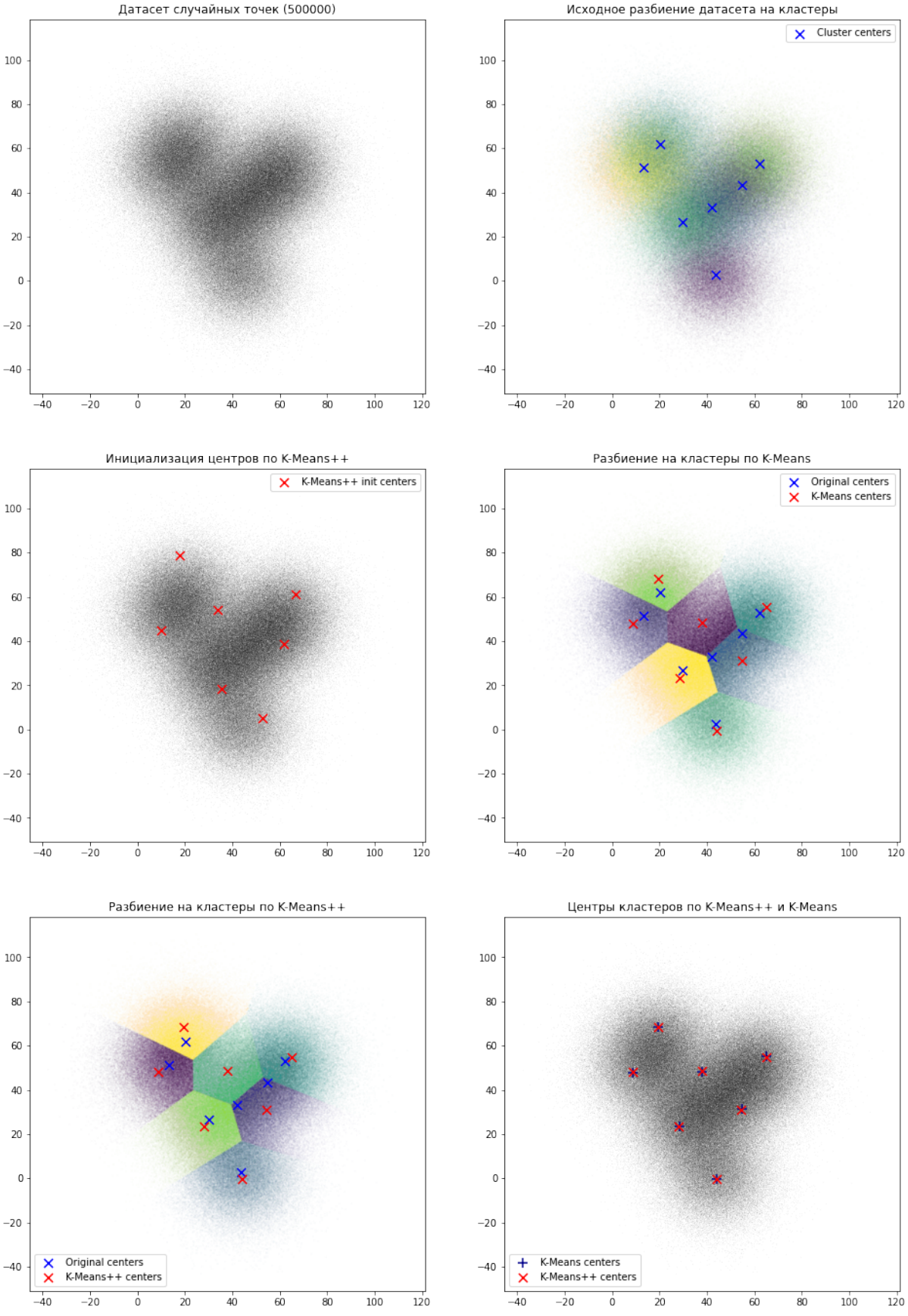

Рис. 2. 7 кластеров, 500000 точек

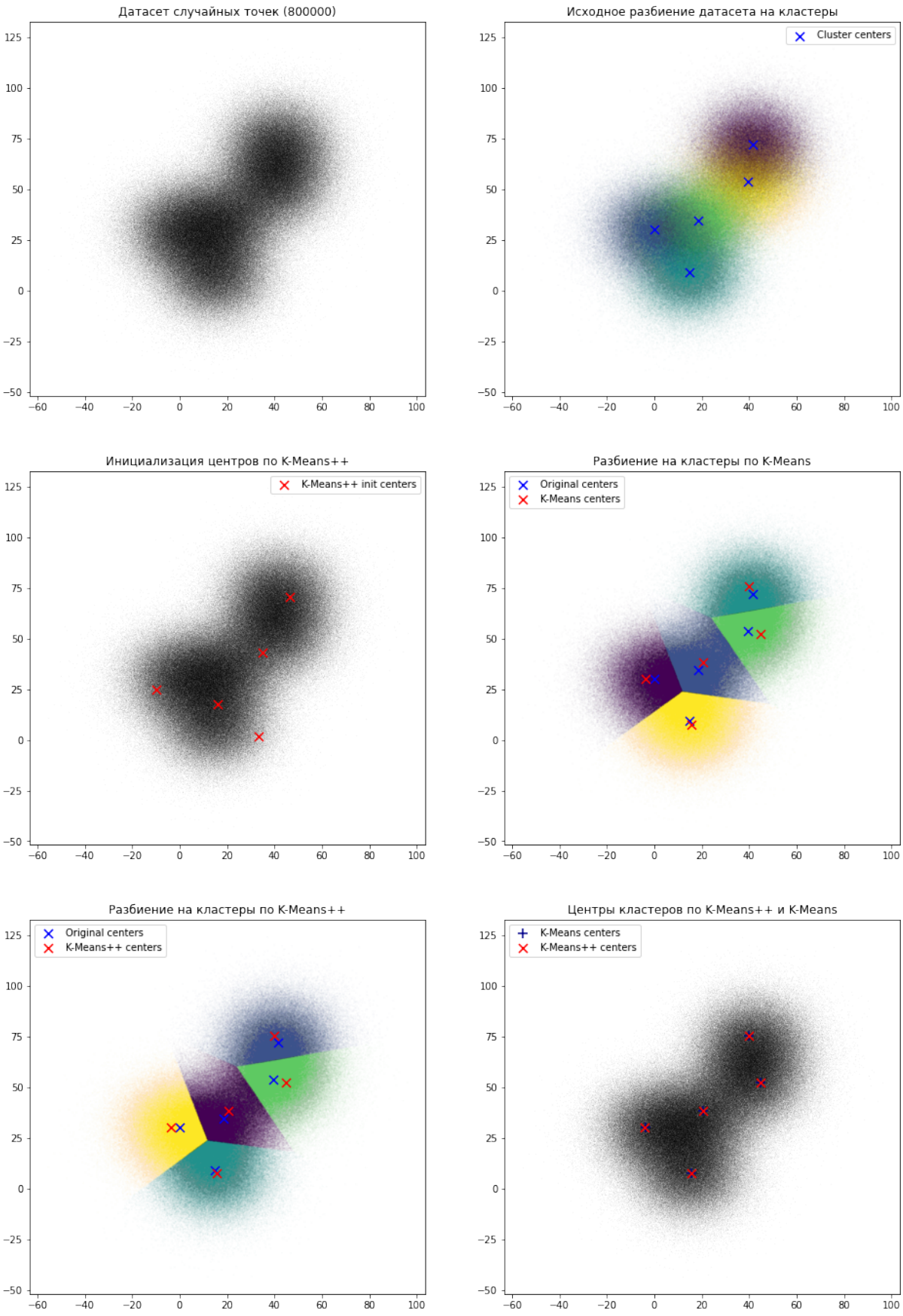

Рис. 3. 5 кластеров, 800000 точек

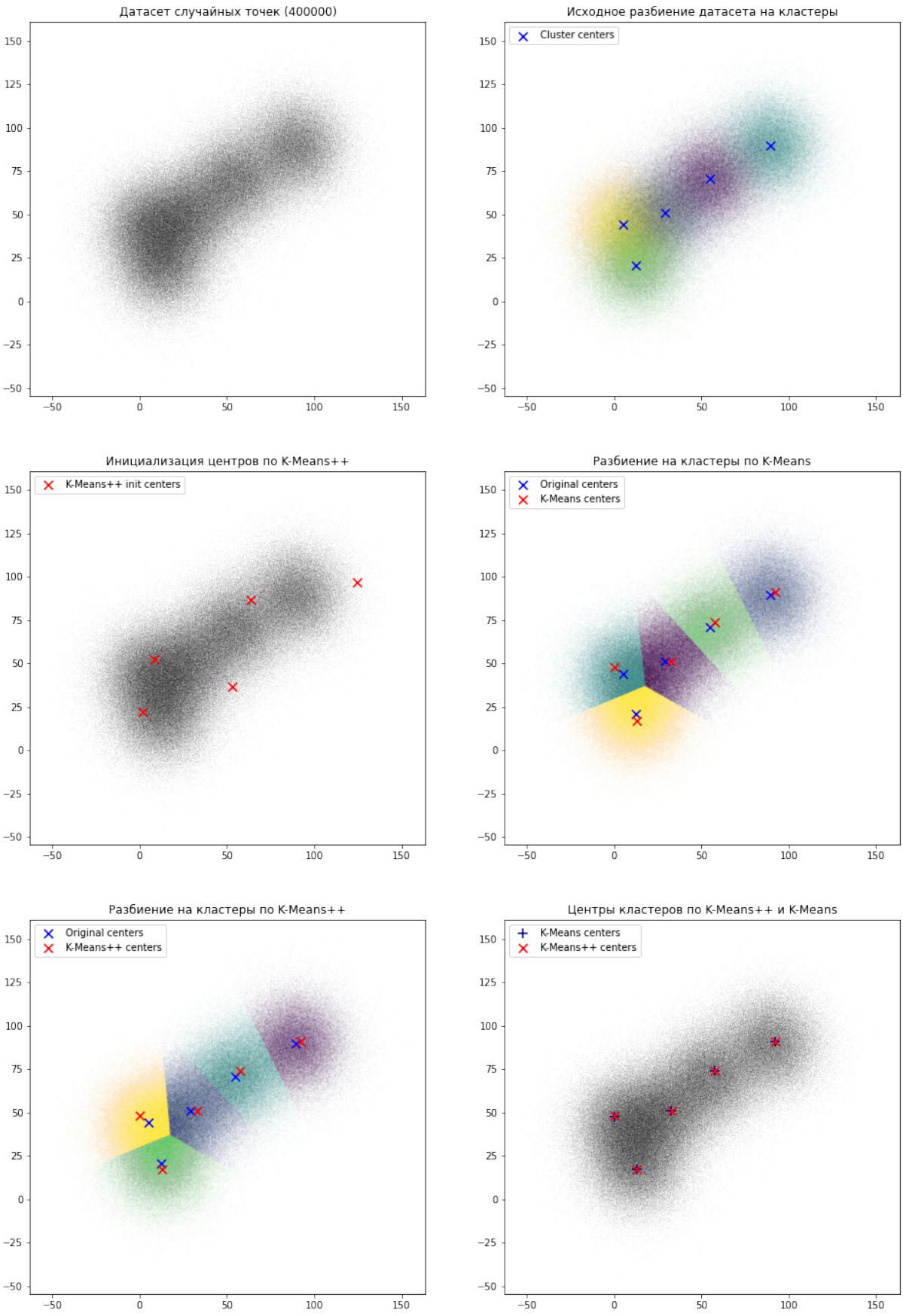

Рис. 4. 5 кластеров, 400000 точек

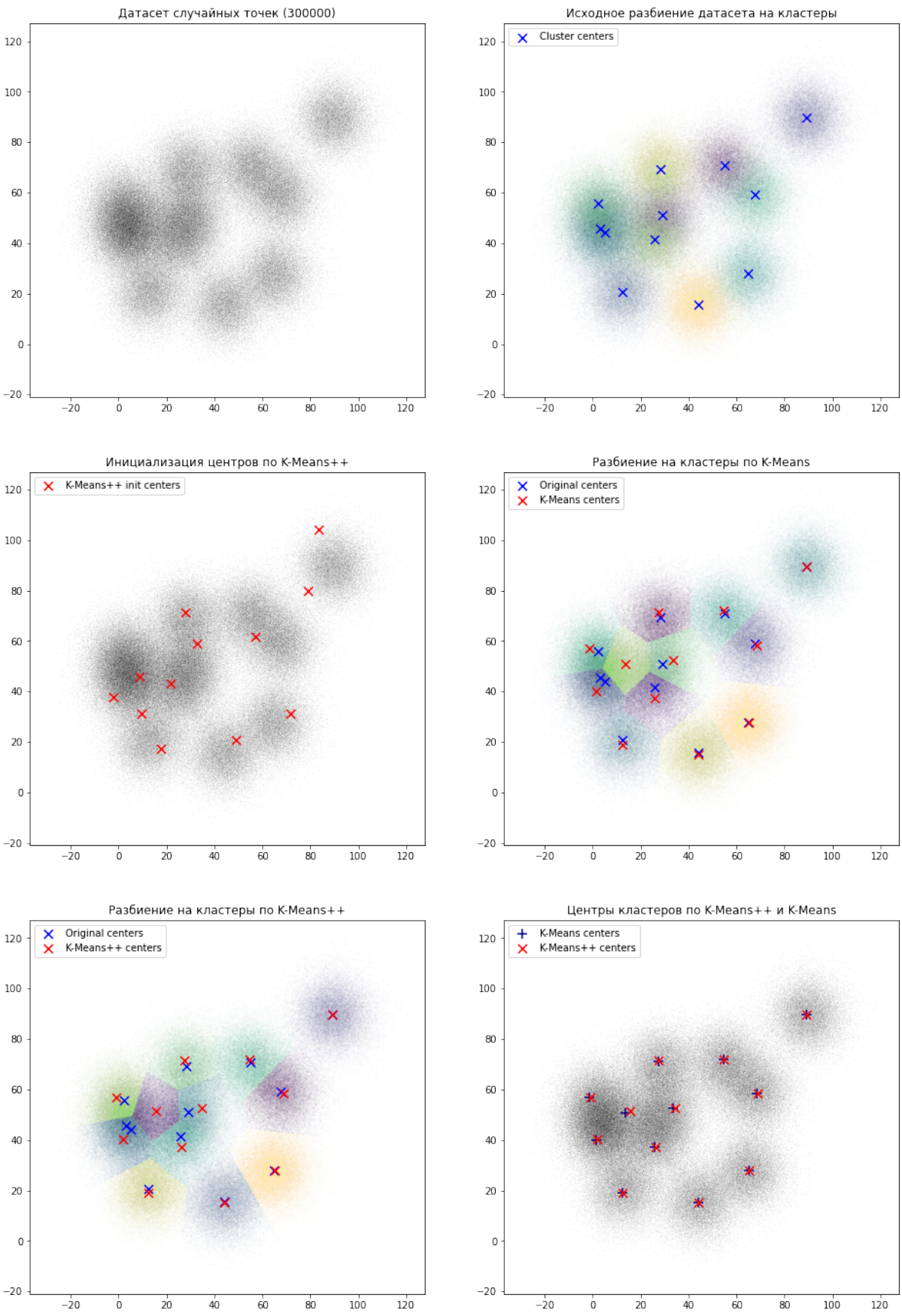

Рис. 5. 12 кластеров, 400000 точек