

Data Analytics.
Lectures. Week 3.
Time Series Clustering.

Максимова Надежда

10 октября 2022 г.

Contents

1 Time Series Clustering	2
2 Ddinamic Time Warping	3
3 Демонстрация работы	4
3.1 Показания фитнес-кольца	4
3.2 Финансовые временные ряды	7

1 Time Series Clustering

Зачем?

- Найти похожие по динамике и паттернам группы временных рядов
- Финансовый анализ
- Поиск аномальных событий с помощью метрик расстояний
- ...

Проблемы:

- Похожие паттерны могут находиться в разных участках временных рядов (начало/конец)

Давайте представим такие временные ряды (рис.1)

Если бы мы использовали обычное евклидово расстояние, то эти ряды объединились бы в кластеры 1, 2 и 3, 4. Но если посмотреть на изуальные паттерны, то можно увидеть, что 1 очень похож на 3, а 2 на 4. Поэтому нужен метод, который умеет так кластеризовать.

- Временные ряды могут разной длины и их нельзя просто представить в матричном виде
- Сложность вычислений

Методы

- Классический k-means и другие методы кластеризации. Если ряды разной длины, их можно привести к одной длине с помощью паддинга или сэмплинга: обрезать длинный ряд или как-то продолжить короткий.



Рис. 1: Возможные паттерны временных рядов

Также можно извлечь признаки из временного ряда, которые уже не будут зависеть от временной составляющей.

Такой метод вычислительно дешевый, но он не может сравнивать паттерны, которые находятся в разных участках временных рядов.

- Dynamic Time Wrapping (Динамическое сжатие времени) - позволяет работать с рядами разной длины и находить паттерны в разных участках ряда, однако он достаточно вычислительно дорогой.
- Эмбеддинги временных рядов (про него не в этой лекции)

2 Ddinamic Time Warping

Очень хорошее описание находится здесь: https://tslearn.readthedocs.io/en/stable/user_guide/dtw.html#dtw-softdtw

Если у нас есть обычное евклидово расстояние, то у нас есть попарное сравнение координат объектов друг с другом. Допустим, у нас есть два временных ряда: красный и синий (рис. 3). Видно, что они очень похожи, но красный начал с падения и потом пошел в параболу, а синий сразу начался с параболы. Также посередине имеется пологий участок разной длины у разных рядов. Если мы будем использовать попарное сравнение, то эти паттерны никак не совместятся и у нас получится какая-то метрика расстояния, никак не отражающая действительность.

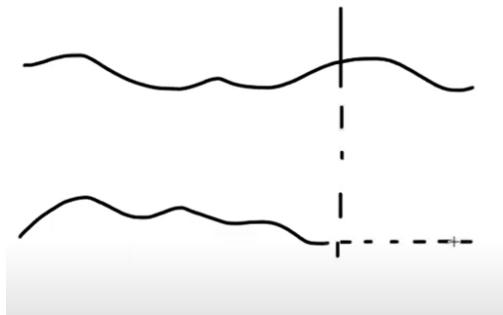


Рис. 2: Приведение временных рядов к одной длине

Если мы применим Dynamic Time Warping (DTW), он будет сравнивать эти ряды, последовательно растягивая их и сжимая, пока они друг на друга идеально не наложатся.

Мы хотим минимизировать среднее расстояние между всеми точками так, чтобы один ряд почти полностью ложился на другой.

Матрица попарных расстояний - матрица, где отображается расстояние между точками одного временного ряда и каждой точкой другого (рис. 4).

Тут представлена матрица двух очень похожих временных рядов, в которых паттерны находятся в разных местах. Чем дальше две точки друг от друга, тем светлее цвет ячейки. Наша задача - найти такой путь, который будет проходить от самого начала (точки $(0,0)$) до самого конца (точки (n,m)) таким образом, чтобы сумма значений расстояний на этом пути была минимальной. При этом наши ряды могут быть разной длины. Этот путь можно рассматривать как способ сопоставления участков временных рядов друг с другом безотносительно времени их появления. В результате мы находим похожие на паттерны участки и сопоставляем их друг с другом. Такая задача решается при помощи динамического программирования за $O(nm)$, где n, m - длины рядов.

Таким образом:

- Путь - это индексы временных рядов
- Путь может ходить по диагонали, горизонтали и вертикали, главное, чтобы был неубывающим по индексам

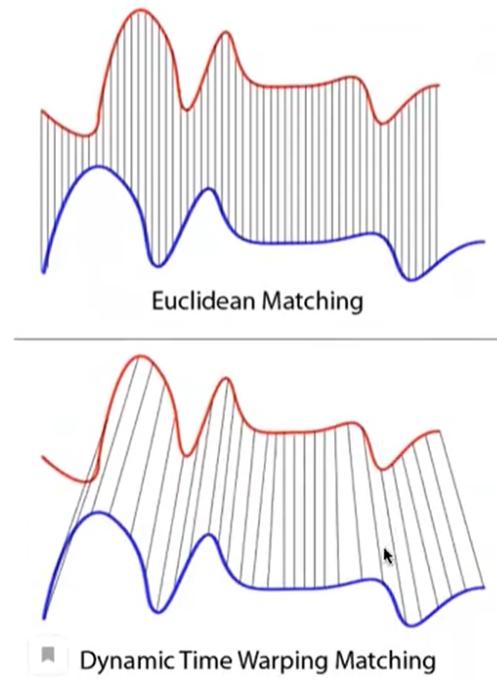


Рис. 3: Сравнение работы обычной метрики и DTW

- Стартуем всегда в начале временных рядов, заканчиваем всегда в конце

Или математически:

Введем расстояние между точками. Оно может быть любым, по умолчанию будет евклидово.

$$DTW(x, y) = \min_{\pi} \sqrt{\sum_{(i,j) \in \pi} d(x_i, y_j)^2}$$

где $\pi = [\pi_0, \dots, \pi_K]$ - это путь, удовлетворяющий набору условий:

- это лист парных индексов $\pi_k = (i_k, j_k)$, где $0 \leq i_k < n$ и $0 \leq j_k < m$
- $\pi_0 = (0, 0)$ и $\pi_K = (n - 1, m - 1)$
- для всех $k > 0$, $\pi_k = (i_k, j_k)$ и $\pi_{k-1} = (i_{k-1}, j_{k-1})$ удовлетворяет следующим неравенствам:
 - $i_{k-1} \leq i_k \leq i_{k-1} + 1$
 - $j_{k-1} \leq j_k \leq j_{k-1} + 1$

Такой алгоритм вычислительно крайне неэффективный, квадратично зависит от длины ряда. Но можно ввести некоторые ограничения. Например, самое распространенное - прокладывать путь вдоль главной диагонали и не дальше, чем несколько шагов вперед/вверх от нее (рис. 5)

При этом мы не будем искать похожие паттерны, которые находятся в разных концах временных рядов.

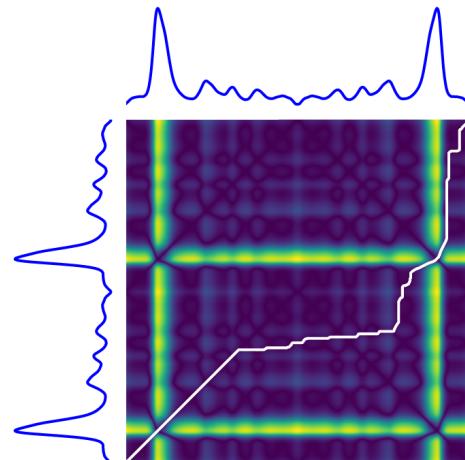


Рис. 4: Матрица попарных расстояний

3 Демонстрация работы

3.1 Показания фитнес-кольца

Рассмотрим задачу определения типа двигательной активности по данным акселерометра (подробнее в лекции Извлечение признаков из временных рядов (Python)). Попробуем ее решить с помощью кластеризации.

Первое, что попробуем сделать - это Baseline-кластеризация. У нас есть данные по 9 каналам: оси x, y и z для акселерометра, гироскопа и общего ускорения. Сконкатенируем их в общий временной ряд и затем приведем их к одной размерности.

Потом построим самый базовый k-means. Результаты моделирования нам говорят, что вероятнее всего у нас 2 кластера, что логично - лежание и нележание. Но мы знаем, что у нас должно быть 6 активностей, поэтому построим новый k-means с 6 кластерами.

Теперь визуализируем их. Для начала уменьшим раз мерность с помощью PCA (Метод главных компонент). Нашли данные не очень скроллизированы, поэтому 2 компоненты объясняют нам только 27,5% всей дисперсии данных. Чтобы объяснить 90% дисперсии данных, нам будет необходимо 200 компонент. Построим визуализацию (рис. 6)

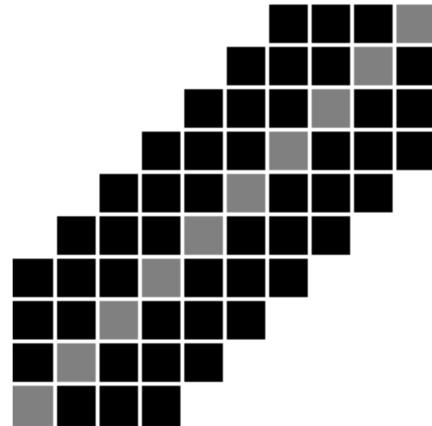


Рис. 5: Ограничения для матрицы попарных расстояний

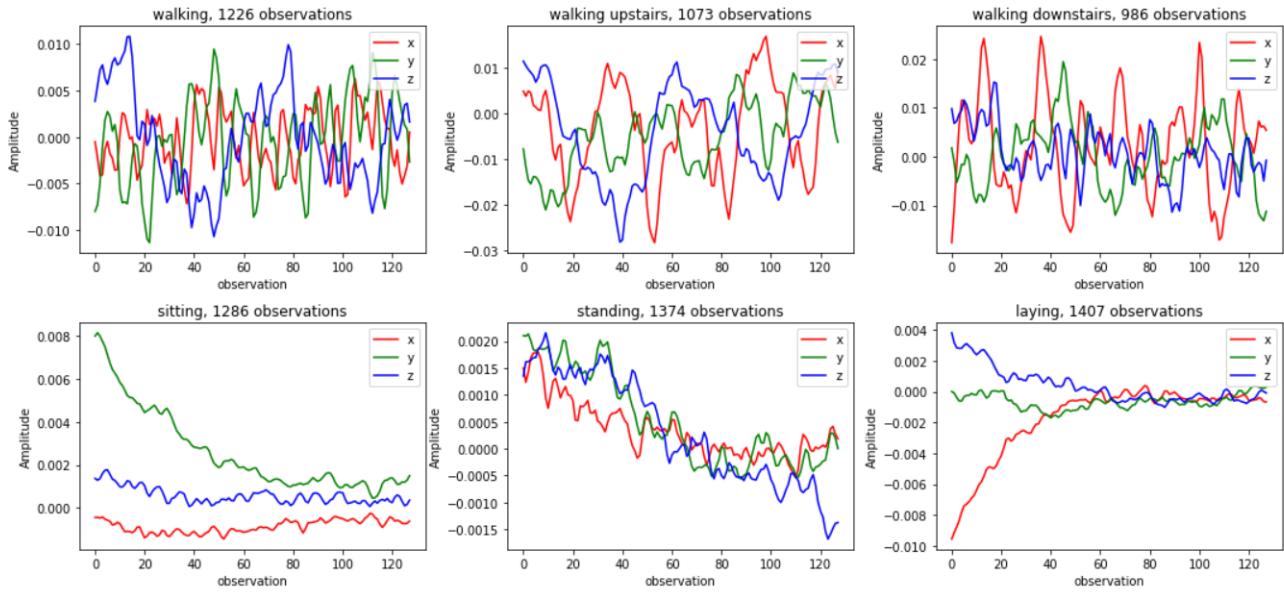


Рис. 8: Исходные центроиды для классов

Теперь используем для изуализации TSNE. Можем сравнить результаты (рис 7 справа) с истинными классами (рис 7 слева). Видим, что есть большой кластер, где скорее всего человек отдыхает. Среди других кластеров некоторые очертания похожи, однако есть большое перемешивание.

Более осмысленную кластеризацию можем сделать с помощью библиотеки tslearn. Здесь есть методы, позволяющие работать с временными рядами.

Построим исходные центроиды для истинных классов (рис. 8). Посмотрим, что получилось в кластеризации k-means (рис. 9). Неплохо определился кластер, где человек лежит, однако в остальных ничего не видно, какие-то рандомные средние.

Теперь посмотрим на k-means, выполненный с помощью tslearn (рис. 10). Он умеет работать с многомерными временными рядами. У нас есть 9 каналов измерений, и tslearn для каждого получит оценку расстояний и усреднит финальную оценку расстояний. Здесь уже стало чуть лучше, видны кластеры, где человек куда-то ходил.

Переходим к DTW (рис. 11). Кластеры выглядят очень похоже на исходные, однако кластер, где человек спит, получился разнесенным на два отдельных кластера. Если мы посмотрим на распределение кластеров, оно очень похоже на исходное (рис. 12). Значит, в наших данных было достаточно информации, чтобы восстановить исходные параметры.

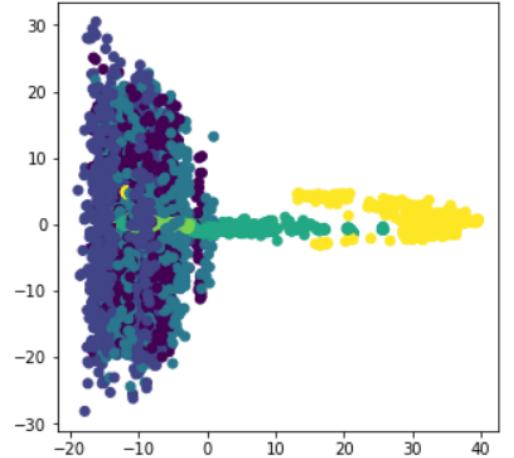


Рис. 6: Визуализация кластеров

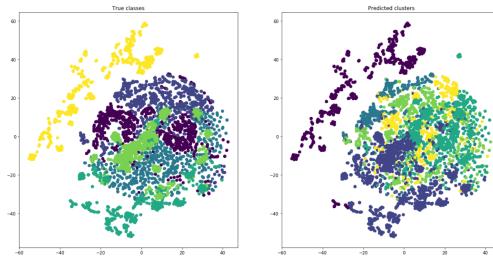


Рис. 7: Визуализация TSNE

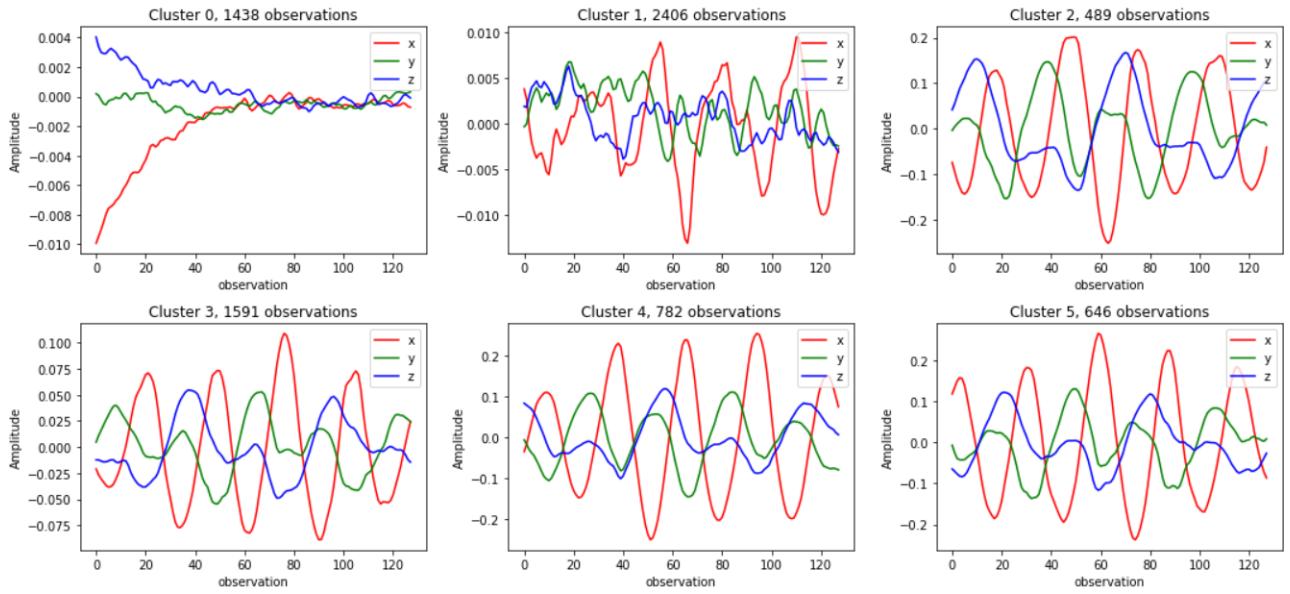


Рис. 9: Центроиды для кластеризации k-means

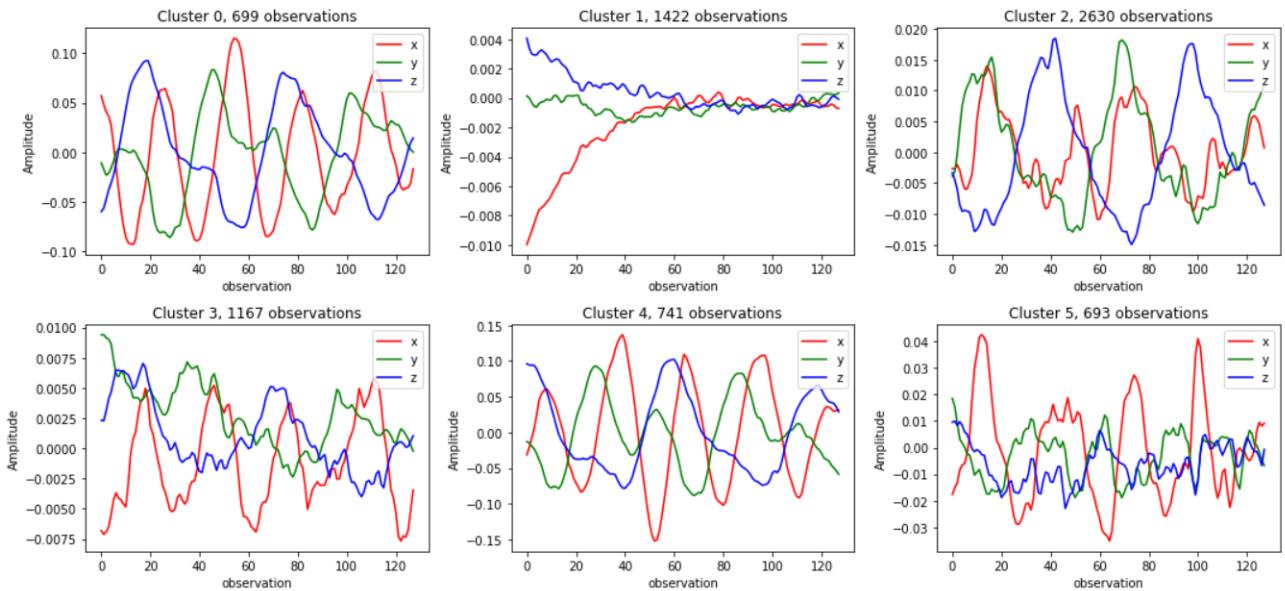


Рис. 10: Центроиды для кластеризации tslearn

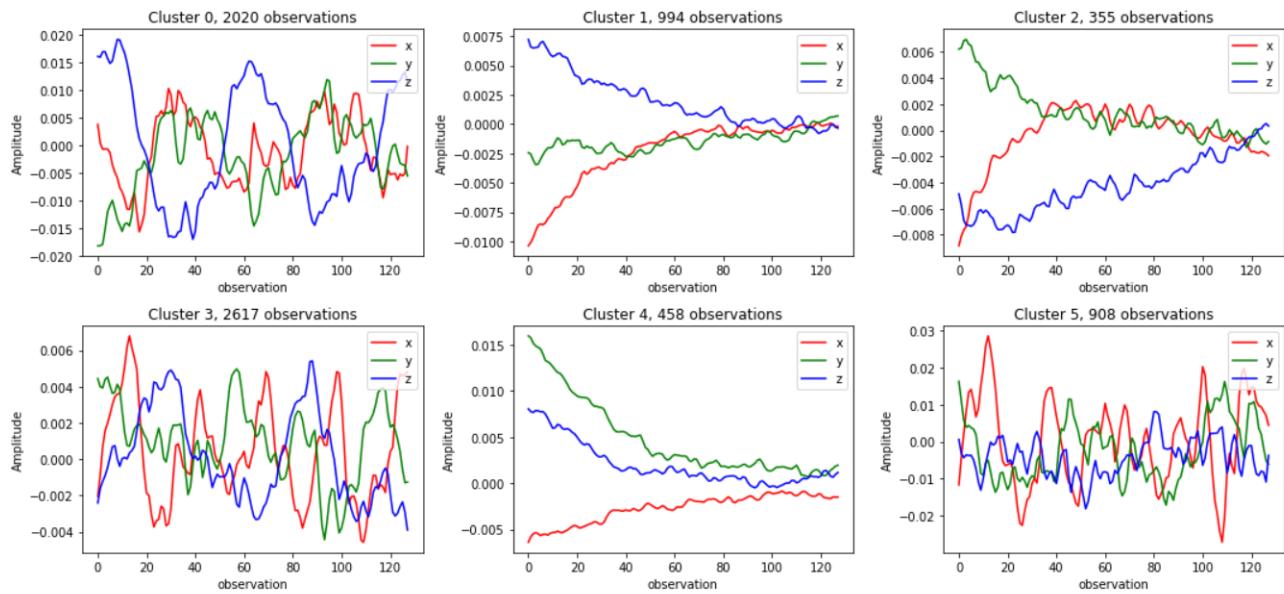


Рис. 11: Центроиды для кластеризации DTW

Далее перейдем к извлечению признаков с помощью Фурье и Вейвлет анализа и снова построим обычную k-means. Посмотрим на центроиды (рис. 13). Они получились очень правдоподобными, в отличие от исходного k-means Для временного ряда, значит, извлеченных данных было достаточно для определения кластеров. На распределении кластеров (рис. 14) кластеры немного перемешались, но тут может быть ошибка в сохранении данных, поэтому надо перепроверить код.

Мы посмотрели на 4 возможных варианта кластеризации: обычный k-means, k-means по каналам, DTW и извлечение признаков для кластеризации. Все они хорошо справляются, однако для разных задач подойдут разные методы.

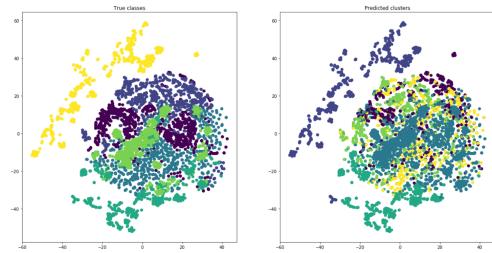


Рис. 12: Распределение кластеров DTW

3.2 Финансовые временные ряды

Будем пользоваться библиотекой Yahoo Finance с данными до июня 2022. в ней содержатся большое количество тикеров, но мы посмотрим на наиболее интересные.

Посмотрим на цены закрытия разных акций. Первое, что сделаем, это приведем все данные к одной размерности. Построим их все на одном графике (рис. 15). Видим, что кто-то движется вверх, кто-то вниз, попробуем это кластеризовать.

Применим обычный k-means, визуализируем получившиеся центроиды кластеров (рис.16). Кластер 0 - это те, кто все время терял стоимость, кластер 1 - это те, кто вырос, 2 - кто вырос, а потом резко упал, 3 - кто все время падал, а в конце еще более резко упал, 4 - те, кто к середине падал, а потом вырос.

Теперь посмотрим, какие тикеры попали к какой группе. Видим, что кластеры 0 - 4 (рис. 17 - 21) хорошо подходят под свое описание, однако есть и исключения. Тут стоит отметить, что в подобного рода анализе нам

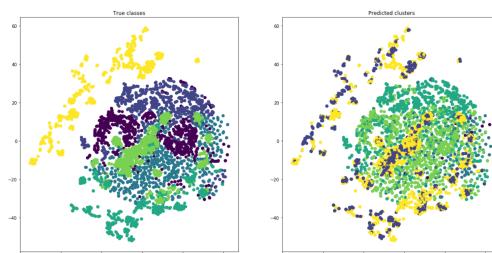


Рис. 14: Распределение кластеров для кластеризации с помощью извлечения признаков

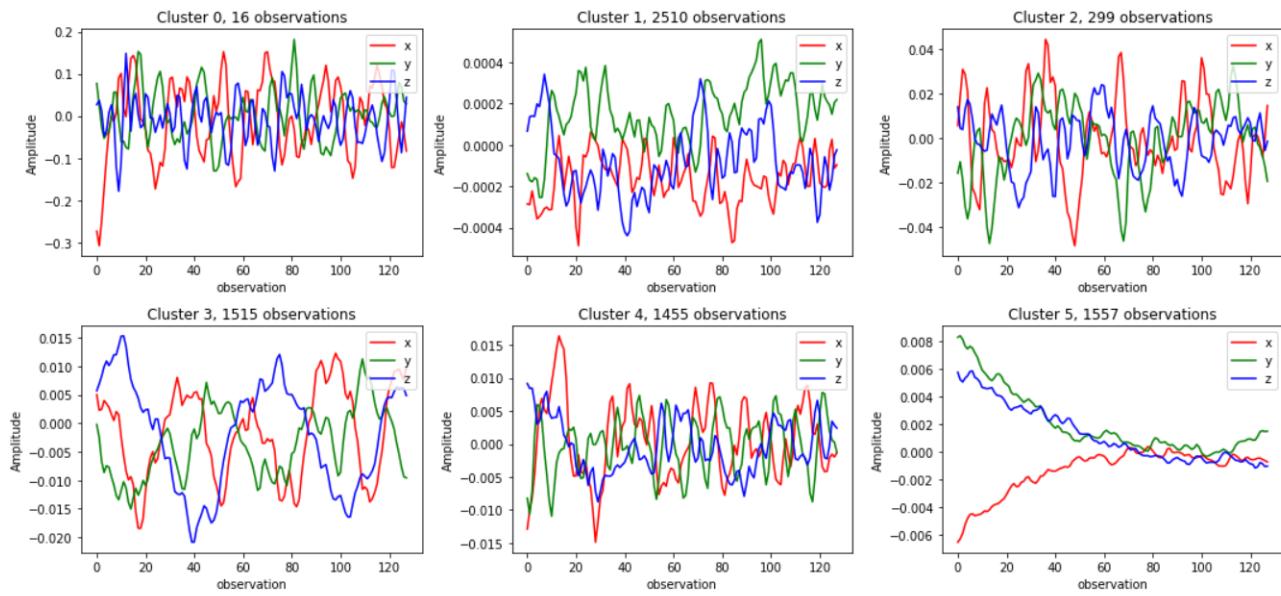


Рис. 13: Центроиды для кластеризации с помощью извлечения признаков

больше интересно сравнение значений по дням, а не поиск паттернов, поэтому обычное евклидово расстояние здесь хорошо работает.

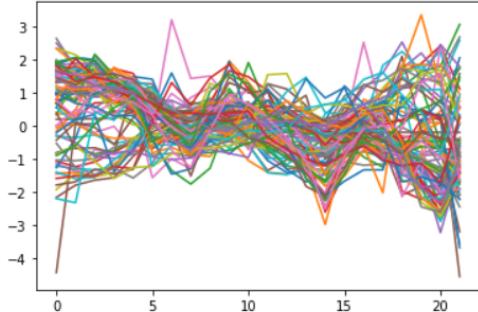


Рис. 15: Графики стоимостей акций

Теперь посмотрим на DTW. У нас получаются следующие паттерны (рис. 22) и предсказания принадлежности к кластерам (рис. 23 - 27). Видим, что здесь принадлежность к кластерам не сильно хорошо удалось предсказать. Это происходит по той же причине, по какой обычный k-means здесь работает лучше - в финансовых временных рядах нам важнее знать сравнение рядов с привязкой по дням, смотреть, как они реагируют на те или иные события, чем поиск определенных паттернов. DTW будет работать на дневных данных с минутной частотой, потому что здесь имеет смысл искать паттерны в пределах одного часа, а не общий тренд.

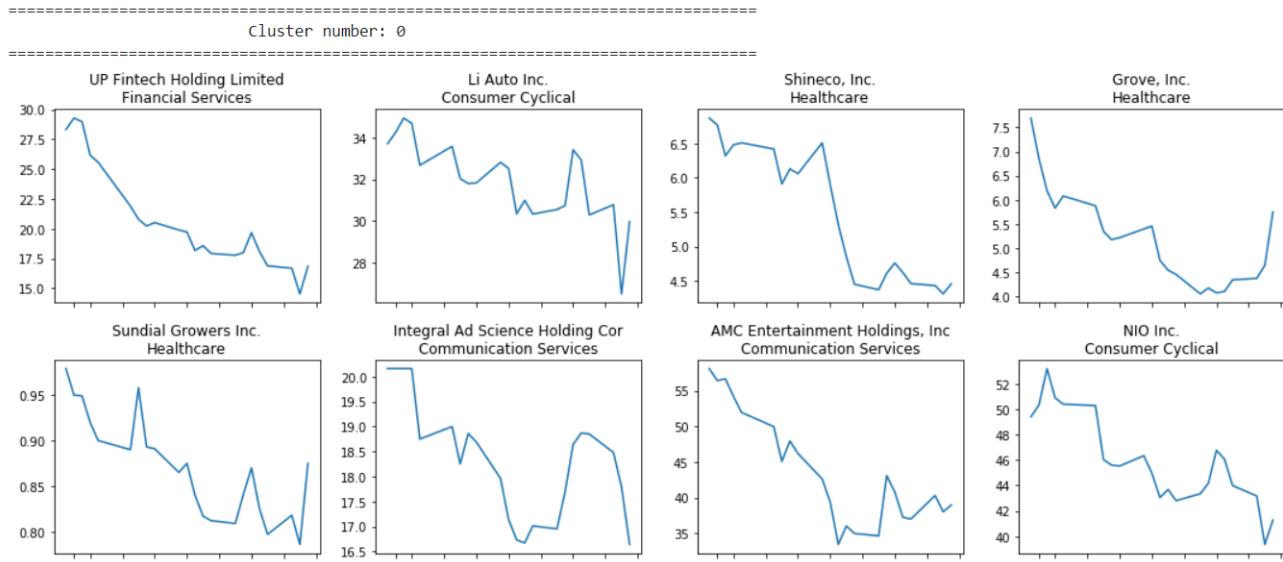


Рис. 17: Кластер 0

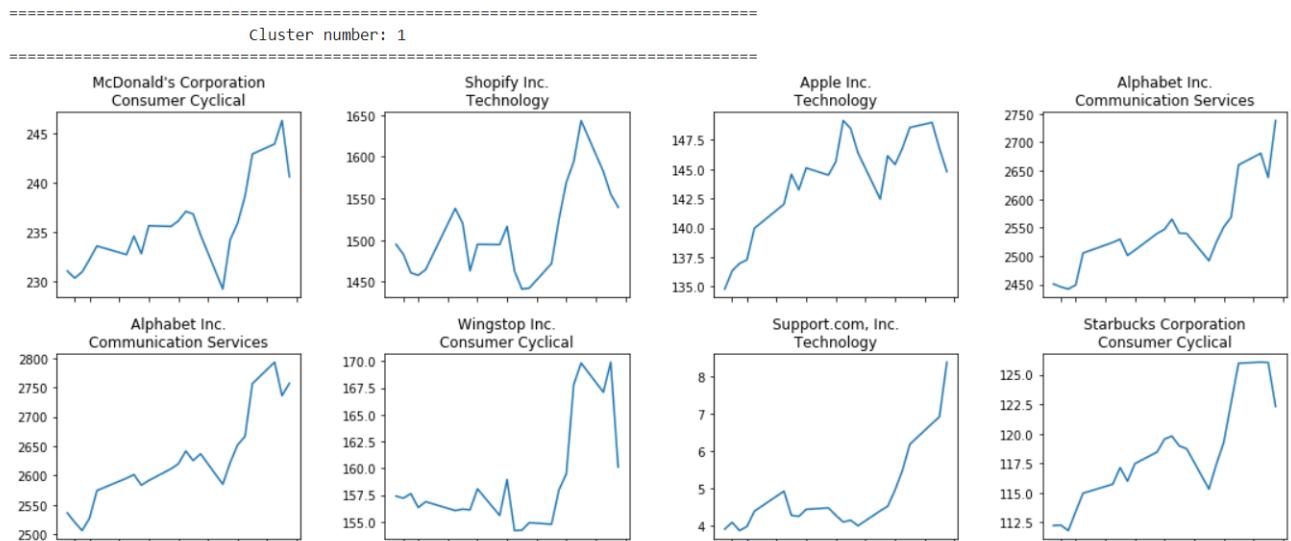


Рис. 18: Кластер 1

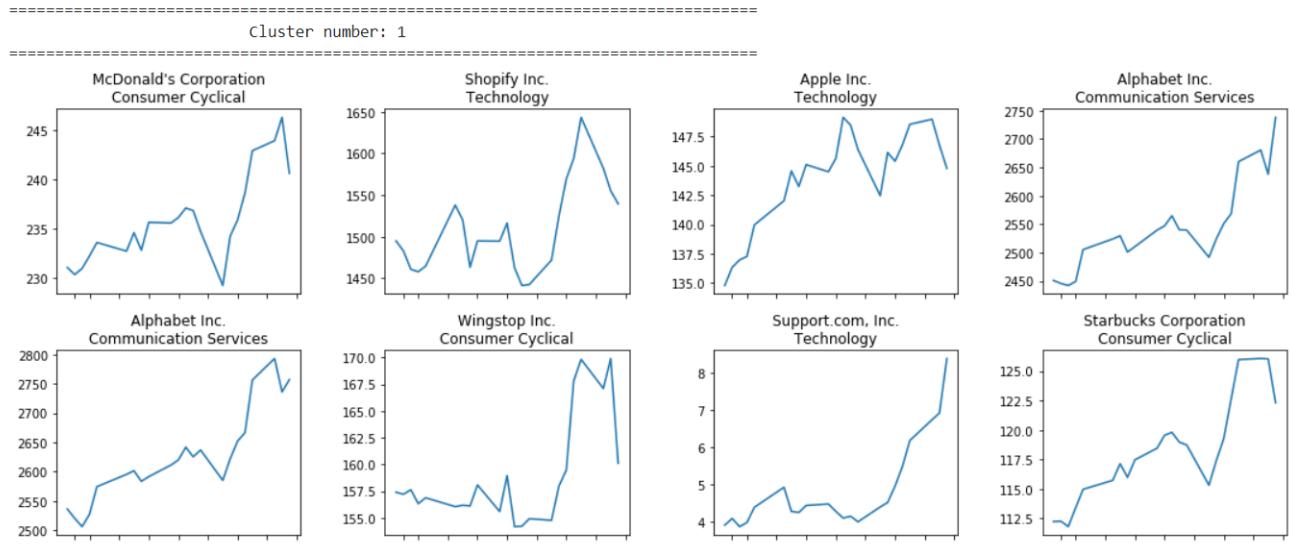


Рис. 19: Кластер 2

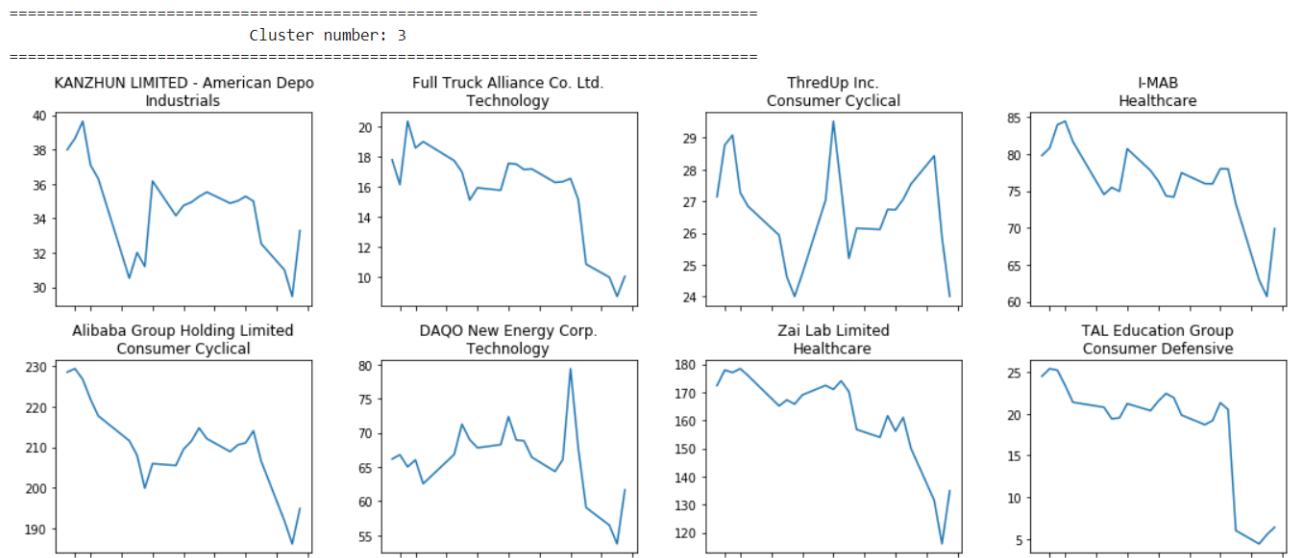


Рис. 20: Кластер 3

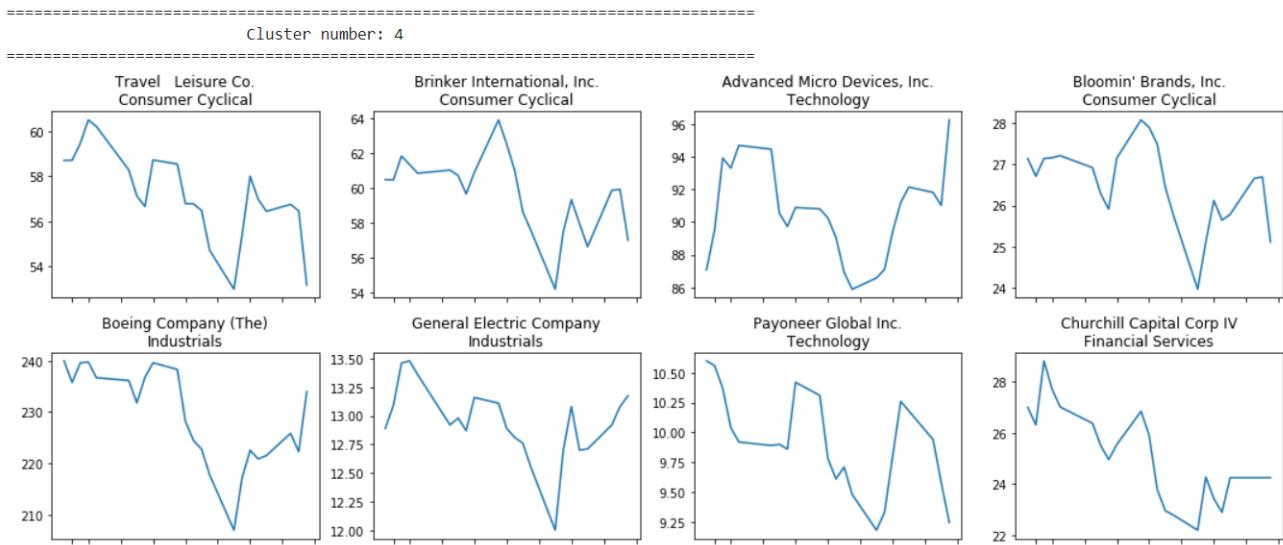


Рис. 21: Кластер 4

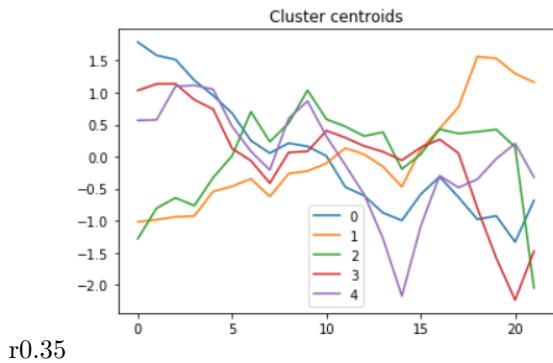


Рис. 16: Центроиды кластеров, полученных с помощью k-means

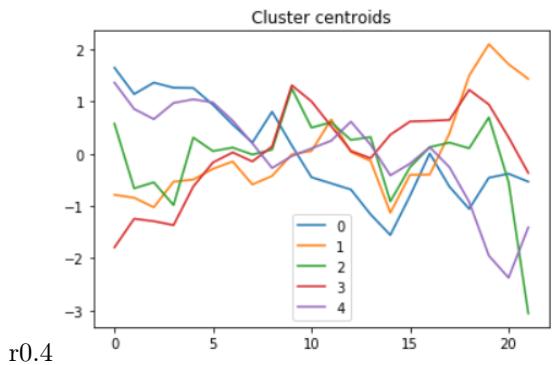


Рис. 22: Центроиды кластеров, полученных с помощью DTW

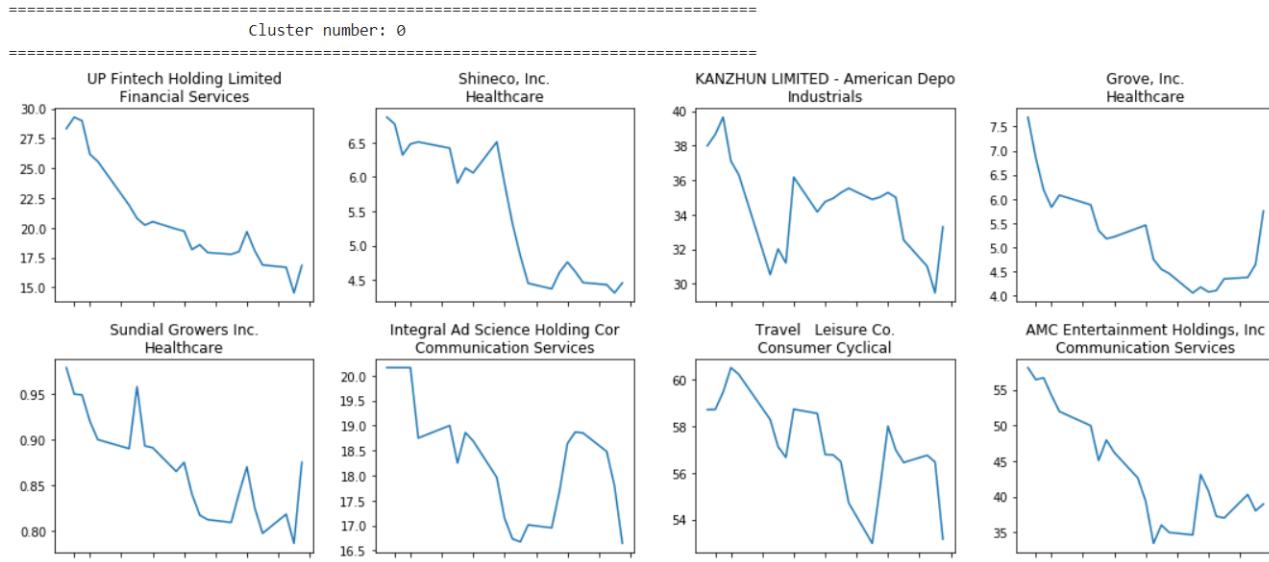


Рис. 23: Кластер 0

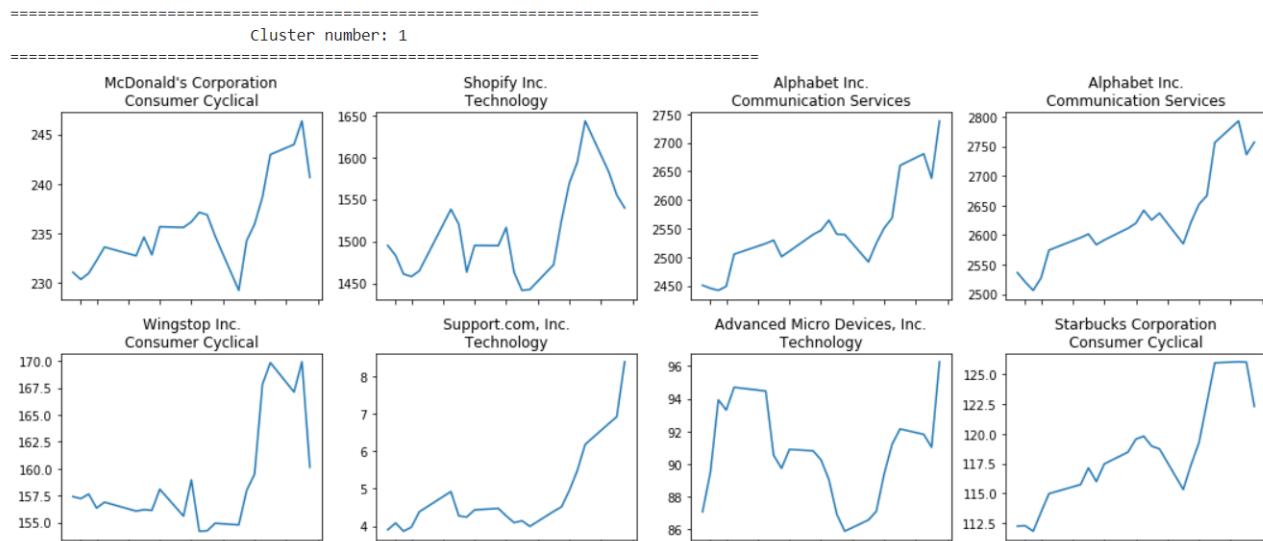


Рис. 24: Кластер 1

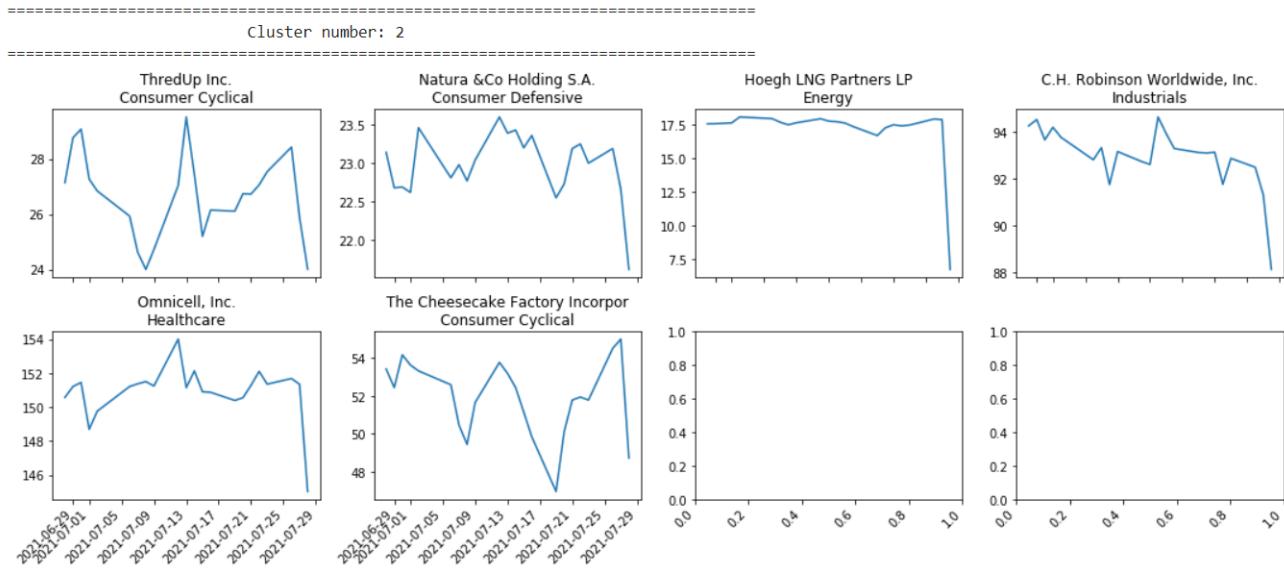


Рис. 25: Кластер 2

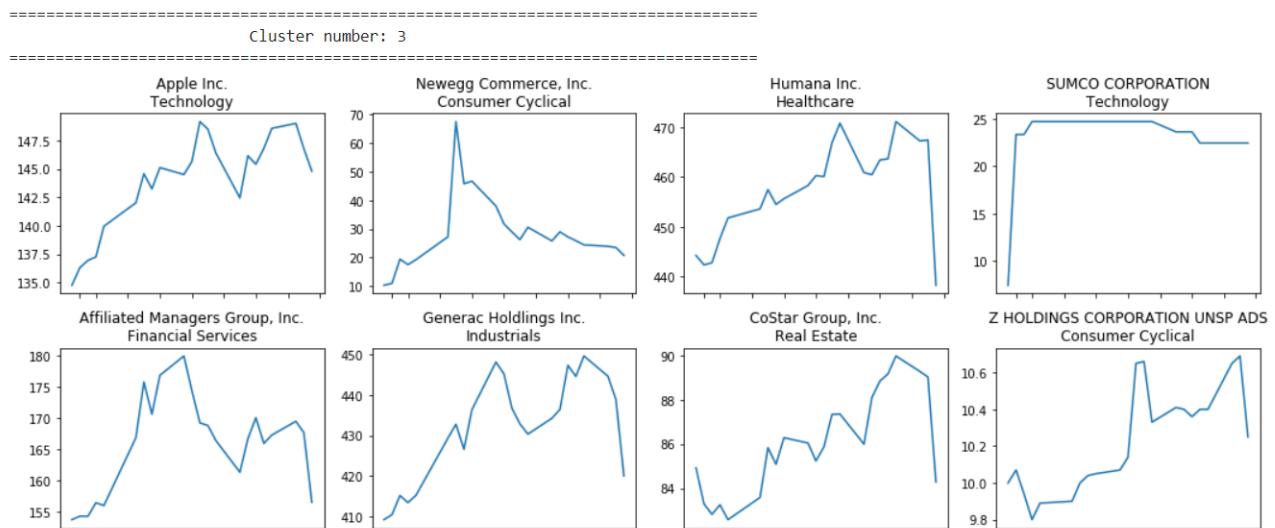


Рис. 26: Кластер 3

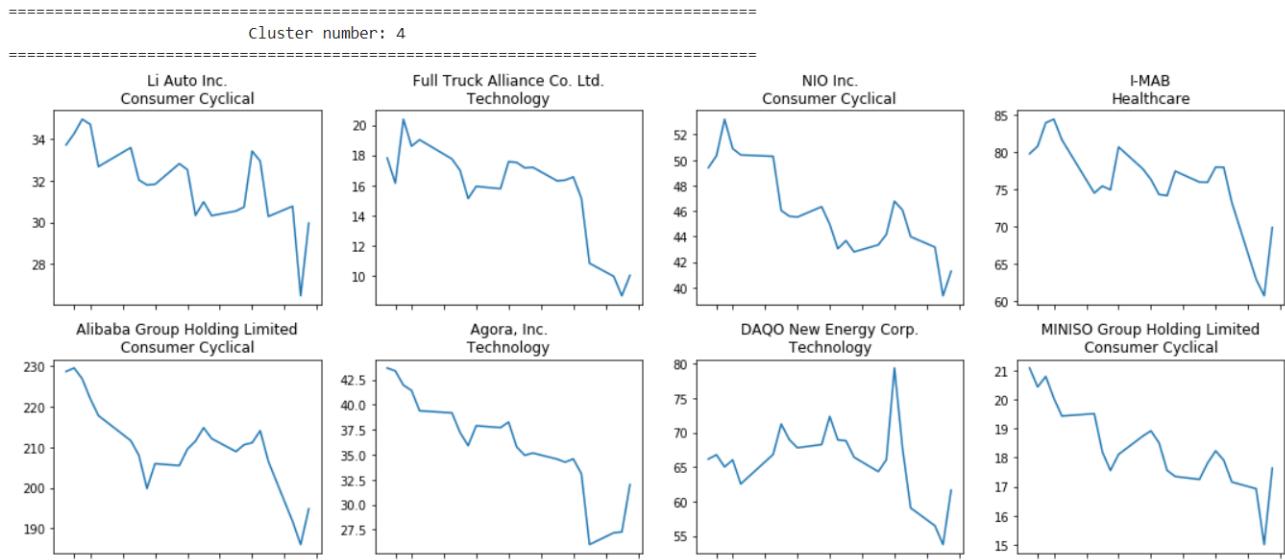


Рис. 27: Кластер 4