

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА - Российский технологический университет» РТУ МИРЭА**

**Институт** Информационных Технологий

**Кафедра** Вычислительной техники

**ПРАКТИЧЕСКАЯ РАБОТА №2**

**по дисциплине**

**«Проектирование интеллектуальных систем (Часть 1/2)»**

Студент группы ИКБО-04-22 Сариков Т.А.

*(Ф.И.О. студента)*

Руководитель работы Холмогоров В.В.

*(Ф.И.О. преподавателя)*

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc10014)

[1 ПОСТАНОВКА ЗАДАЧИ 4](#_Toc10015)

[2 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 5](#_Toc10016)

[2.1 Алгоритм Apriori 6](#_Toc10017)

[2.4 Генерация ассоциативных правил 8](#_Toc10020)

[3 ОПИСАНИЕ ДАННЫХ 9](#_Toc10021)

[3.1 Генерация данных 9](#_Toc10022)

[4 ПРАКТИЧЕСКАЯ ЧАСТЬ 11](#_Toc10023)

[4.1 Реализация алгоритма Apriori 11](#_Toc10024)

[ЗАКЛЮЧЕНИЕ 17](#_Toc10027)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 18](#_Toc10028)

[ПРИЛОЖЕНИЯ 19](#_Toc10029)

# 

# ВВЕДЕНИЕ

Кластерный анализ — это метод группировки объектов по схожим признакам, который широко применяется для исследовательского анализа данных. В данной работе используем кластеризацию для автоматического объединения компаний на основе их финансовых характеристик, таких как доходность, волатильность, рыночная капитализация, коэффициент P/E и дивиденды.

Кластеризация позволяет выявить скрытые типы компаний, упростить анализ больших массивов данных и поддержать принятие решений, например, при выборе инвестиционной стратегии. В рамках практики мы сгенерируем и проанализируем данные, чтобы наглядно увидеть, как алгоритмы способны выделять естественные группы компаний на рынке.

# 1 ПОСТАНОВКА ЗАДАЧИ

Выбранная предметная область: «Анализ компаний на основе денежных характеристик».

Цели:

* приобрести навыки кластеризации как инструмента исследовательского/разведочного анализа данных.

Поставлены следующие задачи:

* визуализировать подготовленные данные и провести предиктивную аналитику количества и качества кластеров;
* выполнить кластеризацию подготовленного набора данных хотя бы одним из методов любой категории алгоритмов
* выполнить кластеризацию не менее чем двумя методами разных категорий;

# 2 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

**Кластерный анализ** — совокупность методов многомерной статистики, направленных на разделение объектов на однородные группы (кластеры) на основе их признаков. Основная цель — минимизировать различия внутри кластера и максимизировать различия между кластерами):

Пусть имеется множество объектов  где каждый объект описывается вектором признаков. Требуется разбить множество  на K кластеров

Для оценки схожести объектов часто используется евклидово расстояние:

## 2.1 Алгоритмы кластеризации

Один из самых популярных алгоритмов кластеризации — метод K-средних. Его задача — минимизировать сумму квадратов расстояний объектов до центров своих кластеров (3.1).

где ​ — центр (среднее значение) кластера , определяемый как:

Метод DBSCAN (кластеризация на основе плотности) формирует кластеры как области пространства с высокой плотностью объектов, отделённые областями низкой плотности. В отличие от K-средних, не требует задания количества кластеров и хорошо работает с кластерами произвольной формы.

Основные параметры: 𝜀 — радиус окрестности (eps-neighborhood);

MinPts — минимальное число точек в 𝜀-окрестности, чтобы область считалась "плотной".

Основные определения: Точка ядра (core point): если в 𝜀 ε-окрестности точки содержится не менее MinPts объектов. Граничная точка (border point): находится в 𝜀 ε-окрестности точки ядра, но сама не является ядром. Шумовая точка (noise point): не относится ни к одному кластеру.

*Алгоритм:*

1. Для каждой точки определяются -соседи.
2. Если точка — ядро, начинается формирование нового кластера.
3. Кластер расширяется за счёт плотносвязанных точек.
4. Повторяется, пока все точки не будут посещены.

## 2.2 Оценка качества кластеризации

Коэффициент силуэта (Silhouette Score) (4.1).

где: a(i)— среднее расстояние от объекта i до всех других объектов в том же кластере, а b(i) — минимальное среднее расстояние от объекта i до объектов других кластеров.

## 2.3 Стандартизация данных

Перед кластеризацией признаки часто стандартизируют (5.1):

где:среднее j признака, а сред квад отклонение признака j

# 3 ОПИСАНИЕ ДАННЫХ

## 3.1 Анализ данных

Данные сгенерированы искусственно для 5 типов компаний (технологии, банки, энергетика, здравоохранение, коммунальные услуги).

Для каждого типа заданы свои параметры для доходности, волатильности, капитализации, P/E и дивидендов (среднее и разброс).

Для каждой компании эти показатели случайно выбираются из соответствующих распределений (нормальное или логнормальное).

В результате получается таблица, где каждая компания имеет уникальный набор финансовых характеристик, типичных для своей отрасли.

Это позволяет имитировать реальные рыночные данные для анализа и кластеризации.

Пример семплов на рисунке 3.1

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 3.1 — Количественные характеристики распределения корзины**

# 4 ПРАКТИЧЕСКАЯ ЧАСТЬ

Предварительно фичи компаний преобразованы в двухмерные компоненты, а также отнормированы для дальнейшей визуализации и кластеризации (рис 4.1).

Изображение выглядит как текст, снимок экрана, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 4.1 — Визуализация**

## 4.1 Кластеризация методом K-means

Для стандартизированных данных был применён алгоритм K-means. Оптимальное количество кластеров определялось с помощью метода локтя и коэффициента силуэта. В результате работы алгоритма получены кластеры, отражающие схожесть компаний по финансовым признакам (Рисунок 4.1).

Изображение выглядит как текст, снимок экрана, Красочность, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 4.2.1 — Результат работы k means**

## 4.2 Кластеризация методом DBSCAN

Также для тех же данных был применён алгоритм DBSCAN, позволяющий выявлять кластеры произвольной формы и выделять выбросы.

Изображение выглядит как снимок экрана, текст, Красочность, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

**Рисунок 4.2.1 — Результат работы dbscan**

# ЗАКЛЮЧЕНИЕ

В результате выполнения работы были успешно сгенерированы и кластеризованы данные о компаниях с различными финансовыми характеристиками. Применение методов кластерного анализа позволило выявить естественные группы компаний и оценить качество разбиения с помощью различных метрик.

Полученные результаты подтверждают, что кластеризация является эффективным инструментом для анализа и сегментации финансовых данных, а также может быть полезна для дальнейших исследований, принятия инвестиционных решений и построения интеллектуальных систем поддержки аналитики.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Azevedo, P. J., Jorge, A. M. Comparing Rule Measures for Predictive Association Rules 2007.

# ПРИЛОЖЕНИЯ

Приложение А — Реализация алгоритмов кластеризации.

**Приложение А**

Реализация алгоритмов кластеризации

*Листинг А.1 — Реализация кластеризации*

*import* pandas *as* pd

*import* numpy *as* np

*import* matplotlib.pyplot *as* plt

*import* seaborn *as* sns

*from* sklearn.cluster *import* KMeans, DBSCAN

*from* sklearn.preprocessing *import* StandardScaler

*from* sklearn.decomposition *import* PCA

*from* sklearn.metrics *import* silhouette\_score

*from* scipy.spatial.distance *import* pdist, squareform

np.random.seed(42)

n = 100

company\_types = {

    'Tech\_Growth': {

        'count': n // 5,

        'return\_params': (0.20, 0.08),

        'volatility\_params': (0.45, 0.08),

        'marketcap\_params': (11.5, 0.4),

        'pe\_params': (45, 8),

        'dividend\_params': (0.003, 0.002)

    },

    'Traditional\_Banks': {

        'count': n // 5,

        'return\_params': (0.06, 0.06),

        'volatility\_params': (0.15, 0.04),

        'marketcap\_params': (10.2, 0.3),

        'pe\_params': (8, 3),

        'dividend\_params': (0.055, 0.010)

    },

    'Oil\_Energy': {

        'count': n // 5,

        'return\_params': (0.02, 0.25),

        'volatility\_params': (0.55, 0.10),

        'marketcap\_params': (9.8, 0.8),

        'pe\_params': (12, 4),

        'dividend\_params': (0.065, 0.015)

    },

    'Healthcare': {

        'count': n // 5,

        'return\_params': (0.12, 0.05),

        'volatility\_params': (0.12, 0.03),

        'marketcap\_params': (11.0, 0.3),

        'pe\_params': (28, 5),

        'dividend\_params': (0.018, 0.005)

    },

    'Utilities': {

        'count': n - 4 \* (n // 5),

        'return\_params': (0.04, 0.03),

        'volatility\_params': (0.08, 0.02),

        'marketcap\_params': (9.2, 0.25),

        'pe\_params': (15, 3),

        'dividend\_params': (0.08, 0.012)

    }

}

data = []

company\_counter = 1

*for* group\_name, params *in* company\_types.items():

    print(f"Генерируем {params['count']} компаний типа {group\_name}")

*for* i *in* range(params['count']):

        data.append({

            'Company': f'{group\_name}\_{i+1}',

            'Group': group\_name,

            'Return': np.random.normal(\*params['return\_params']),

            'Volatility': np.random.normal(\*params['volatility\_params']),

            'MarketCap': np.random.lognormal(\*params['marketcap\_params']),

            'PE\_Ratio': np.random.normal(\*params['pe\_params']),

            'Dividend': np.random.normal(\*params['dividend\_params'])

        })

        company\_counter += 1

df = pd.DataFrame(data)

print("Исходные данные:")

print(df.head())

print(f"Пропуски: {df.isnull().sum().sum()}")

df['Volatility'] = df['Volatility'].abs()

df['MarketCap'] = df['MarketCap'].abs()

df['PE\_Ratio'] = df['PE\_Ratio'].abs()

df['Dividend'] = df['Dividend'].abs()

features = ['Return', 'Volatility', 'MarketCap', 'PE\_Ratio', 'Dividend']

X = df[features]

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

print(f"Данные подготовлены: {X.shape}")

print("\n=== АНАЛИЗ РАЗДЕЛИМОСТИ ГРУПП ===")

group\_means = df.groupby('Group')[features].mean()

print("Средние значения по группам:")

print(group\_means.round(3))

group\_distances = pdist(group\_means.values, *metric*='euclidean')

distance\_matrix = squareform(group\_distances)

print(f"\nМинимальное расстояние между группами: {group\_distances.min():.3f}")

print(f"Максимальное расстояние между группами: {group\_distances.max():.3f}")

print(f"Среднее расстояние между группами: {group\_distances.mean():.3f}")

plt.figure(*figsize*=(12, 8))

plt.subplot(2, 3, 1)

sns.heatmap(X.corr(), *annot*=True, *cmap*='coolwarm')

plt.title('Корреляция признаков')

print(X\_scaled.shape, ': \tX\_scaled.shape')

pca = PCA(*n\_components*=2)

X\_pca = pca.fit\_transform(X\_scaled)

plt.subplot(2, 3, 2)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], *alpha*=0.7)

plt.xlabel(f'PC1 ({pca.explained\_variance\_ratio\_[0]:.1%})')

plt.ylabel(f'PC2 ({pca.explained\_variance\_ratio\_[1]:.1%})')

plt.title('PCA компоненты')

inertias = []

K = range(1, 8)

*for* k *in* K:

    kmeans = KMeans(*n\_clusters*=k, *random\_state*=42)

    kmeans.fit(X\_scaled)

    inertias.append(kmeans.inertia\_)

plt.subplot(2, 3, 3)

plt.plot(K, inertias, 'bo-')

plt.xlabel('Количество кластеров')

plt.ylabel('Инерция')

plt.title('Метод локтя')

silhouette\_scores = []

*for* k *in* range(2, 8):

    kmeans = KMeans(*n\_clusters*=k, *random\_state*=42)

    labels = kmeans.fit\_predict(X\_scaled)

    score = silhouette\_score(X\_scaled, labels)

    silhouette\_scores.append(score)

plt.subplot(2, 3, 4)

plt.plot(range(2, 8), silhouette\_scores, 'ro-')

plt.xlabel('Количество кластеров')

plt.ylabel('Silhouette Score')

plt.title('Качество кластеризации')

optimal\_k = range(2, 8)[np.argmax(silhouette\_scores)]

print(f"Оптимальное количество кластеров: {optimal\_k}")

kmeans = KMeans(*n\_clusters*=optimal\_k, *random\_state*=42)

kmeans\_labels = kmeans.fit\_predict(X\_scaled)

dbscan = DBSCAN(*eps*=0.8, *min\_samples*=3)

dbscan\_labels = dbscan.fit\_predict(X\_scaled)

plt.subplot(2, 3, 5)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], *c*=kmeans\_labels, *cmap*='viridis', *alpha*=0.8)

plt.title(f'K-means ({optimal\_k} кластеров)')

plt.xlabel('PC1')

plt.subplot(2, 3, 6)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], *c*=dbscan\_labels, *cmap*='viridis', *alpha*=0.8)

plt.title(f'DBSCAN ({len(set(dbscan\_labels))} кластеров)')

plt.xlabel('PC1')

plt.tight\_layout()

plt.show()

plt.figure(*figsize*=(15, 5))

plt.subplot(1, 3, 1)

group\_colors = {'Tech\_Growth': 0, 'Traditional\_Banks': 1, 'Oil\_Energy': 2, 'Healthcare': 3, 'Utilities': 4}

real\_group\_labels = [group\_colors[group] *for* group *in* df['Group']]

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], *c*=real\_group\_labels, *cmap*='Set1', *alpha*=0.8)

plt.title('Реальные группы компаний')

plt.xlabel('PC1')

plt.ylabel('PC2')

plt.subplot(1, 3, 2)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], *c*=kmeans\_labels, *cmap*='viridis', *alpha*=0.8)

plt.title(f'K-means кластеры')

plt.xlabel('PC1')

plt.subplot(1, 3, 3)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], *c*=kmeans\_labels, *cmap*='viridis', *alpha*=0.3)

centroids\_pca = pca.transform(kmeans.cluster\_centers\_)

plt.scatter(centroids\_pca[:, 0], centroids\_pca[:, 1], *c*='red', *marker*='x', *s*=200, *linewidths*=3)

plt.title('K-means с центроидами')

plt.xlabel('PC1')

plt.tight\_layout()

plt.show()

df['Cluster'] = kmeans\_labels

print("\n=== АНАЛИЗ КЛАСТЕРОВ ===")

*for* cluster *in* range(optimal\_k):

    cluster\_data = df[df['Cluster'] == cluster]

    print(f"\nКластер {cluster} ({len(cluster\_data)} компаний):")

    print(f"Компании: {', '.join(cluster\_data['Company'].head(5).tolist())}{'...' *if* len(cluster\_data) > 5 *else* ''}")

    means = cluster\_data[features].mean()

    print("Характеристики:")

*for* feature *in* features:

        print(f"  {feature}: {means[feature]:.3f}")

    group\_distribution = cluster\_data['Group'].value\_counts()

    print("Состав по типам компаний:")

*for* group, count *in* group\_distribution.items():

        percentage = (count / len(cluster\_data)) \* 100

        print(f"  {group}: {count} ({percentage:.1f}%)")

print("\n=== СРАВНЕНИЕ С РЕАЛЬНЫМИ ГРУППАМИ ===")

*for* group *in* company\_types.keys():

    group\_data = df[df['Group'] == group]

    cluster\_distribution = group\_data['Cluster'].value\_counts().sort\_index()

    print(f"\n{group} ({len(group\_data)} компаний):")

*for* cluster, count *in* cluster\_distribution.items():

        percentage = (count / len(group\_data)) \* 100

        print(f"  Кластер {cluster}: {count} компаний ({percentage:.1f}%)")

print("\n=== ИНТЕРПРЕТАЦИЯ ===")

*for* cluster *in* range(optimal\_k):

    cluster\_data = df[df['Cluster'] == cluster]

    means = cluster\_data[features].mean()

*if* means['Return'] > 0.1 and means['Volatility'] > 0.3:

        cluster\_type = "Агрессивные акции роста"

*elif* means['Dividend'] > 0.025:

        cluster\_type = "Дивидендные акции"

*elif* means['Volatility'] < 0.2:

        cluster\_type = "Защитные акции"

*else*:

        cluster\_type = "Сбалансированные акции"

    print(f"Кластер {cluster}: {cluster\_type}")

plt.figure(*figsize*=(10, 6))

*for* cluster *in* range(optimal\_k):

    cluster\_means = df[df['Cluster'] == cluster][features].mean()

    plt.plot(features, cluster\_means, 'o-', *label*=f'Кластер {cluster}', *linewidth*=2)

plt.xlabel('Признаки')

plt.ylabel('Значения (нормализованные)')

plt.title('Профили кластеров')

plt.legend()

plt.xticks(*rotation*=45)

plt.grid(True, *alpha*=0.3)

plt.tight\_layout()

plt.show()

print(f"\nSilhouette Score: {silhouette\_score(X\_scaled, kmeans\_labels):.3f}")

*from* sklearn.metrics *import* adjusted\_rand\_score, normalized\_mutual\_info\_score

real\_labels = [group\_colors[group] *for* group *in* df['Group']]

ari\_score = adjusted\_rand\_score(real\_labels, kmeans\_labels)

nmi\_score = normalized\_mutual\_info\_score(real\_labels, kmeans\_labels)

print(f"Adjusted Rand Index: {ari\_score:.3f}")

print(f"Normalized Mutual Information: {nmi\_score:.3f}")

print("Кластеризация завершена!")

df.to\_csv('financial\_clusters.csv', *index*=False)

print("Результаты сохранены в 'financial\_clusters.csv'")

summary\_data = []

*for* group *in* company\_types.keys():

    group\_data = df[df['Group'] == group]

    summary\_data.append({

        'Group': group,

        'Count': len(group\_data),

        'Avg\_Return': group\_data['Return'].mean(),

        'Avg\_Volatility': group\_data['Volatility'].mean(),

        'Avg\_MarketCap': group\_data['MarketCap'].mean(),

        'Avg\_PE\_Ratio': group\_data['PE\_Ratio'].mean(),

        'Avg\_Dividend': group\_data['Dividend'].mean(),

        'Most\_Common\_Cluster': group\_data['Cluster'].mode().iloc[0] *if* len(group\_data) > 0 *else* -1

    })

summary\_df = pd.DataFrame(summary\_data)

summary\_df.to\_csv('group\_summary.csv', *index*=False)

print("Сводка по группам сохранена в 'group\_summary.csv'")