**Методические указания к лабораторной работе № 3**

**Автор: Якунин Е.М.**

**Кластеризация**

Данная лабораторная работа посвящена обучению кластеризации текста.

**Цель работы:** кластеризировать спарсенные данные.

**Задачи:**

1. Получить теоретические знания по теме.
2. Загрузить данные из csv-файлов.
3. Обработать данные (очистить).
4. Кластеризировать данные.

**Кластеризация**

**Кластеризация (или кластерный анализ)** — это задача разбиения множества объектов на группы, называемые кластерами. Внутри каждой группы должны оказаться «похожие» объекты, а объекты разных группы должны быть как можно более отличны.

Список прикладных областей, где она применяется, широк: сегментация изображений, маркетинг, [борьба с мошенничеством](https://wiki.loginom.ru/articles/antifraud.html), [прогнозирование](https://wiki.loginom.ru/articles/forecasting.html), [анализ текстов](https://wiki.loginom.ru/articles/text-mining.html) и многие другие.

Существует много способов кластеризации данных, мы рассмотрим метод «K-means» (к-средних).

**Метод *k*-средних** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *k-means*) — наиболее популярный метод [кластеризации](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F). Был изобретён в 1950-х годах математиком [Гуго Штейнгаузом](https://ru.wikipedia.org/wiki/%D0%A8%D1%82%D0%B5%D0%B9%D0%BD%D0%B3%D0%B0%D1%83%D0%B7,_%D0%93%D1%83%D0%B3%D0%BE) и почти одновременно Стюартом Ллойдом. Особую популярность приобрёл после работы Маккуина.

**Алгоритм**

Алгоритм разбивает [множество](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) элементов [векторного пространства](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D1%81%D1%82%D0%B2%D0%BE) на заранее известное число кластеров *k*.

Основная идея заключается в том, что на каждой итерации перевычисляется [центр масс](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BD%D1%82%D1%80_%D0%BC%D0%B0%D1%81%D1%81) 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](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) конечно, а на каждом шаге суммарное квадратичное отклонение  уменьшается, поэтому зацикливание невозможно.

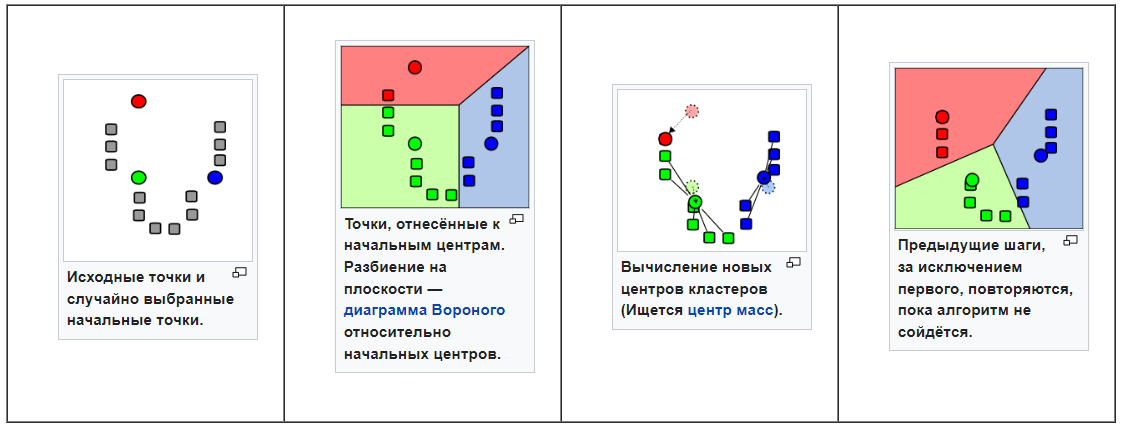


Рисунок 1 − Демонстрация алгоритма

**Проблемы K-means**

* Не гарантируется достижение глобального минимума суммарного квадратичного отклонения, а только одного из локальных минимумов.
* Результат зависит от выбора исходных центров кластеров, их оптимальный выбор неизвестен.
* Число кластеров надо знать заранее.

**Пример применения**

Мы уже имеем данные из прошлой лабораторной работы.

Данная работа будет выполнять в google.colab, т.к. там можно запускать код отдельно по блокам, что делает работу нагляднее.

Для начала импортируем необходимые библиотеки:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import string

import re

import nltk

from sklearn.preprocessing import LabelBinarizer

from nltk.corpus import stopwords

from nltk.stem import WordNetLemmatizer

from nltk.tokenize import word\_tokenize,sent\_tokenize

from google.colab import files

from sklearn.utils import shuffle

from nltk.corpus import stopwords

from string import punctuation

nltk.download('stopwords')

nltk.download('punkt')

nltk.download('wordnet')

Загрузим csv-файлы:

# загрузим спарсенные данные

uploaded = files.upload()

 **News.csv**(text/csv) - 596462 bytes, last modified: 03.05.2022 - 100% done

 **NewsBisnes.csv**(text/csv) - 169830 bytes, last modified: 03.05.2022 - 100% done

 **NewsCulture.csv**(text/csv) - 1245419 bytes, last modified: 03.05.2022 - 100% done

 **NewsPolitics.csv**(text/csv) - 2207207 bytes, last modified: 03.05.2022 - 100% done

Saving News.csv to News.csv

Saving NewsBisnes.csv to NewsBisnes.csv

Saving NewsCulture.csv to NewsCulture.csv

Saving NewsPolitics.csv to NewsPolitics.csv

Прочитаем данные и объединим их (рис.1):

sport = pd.read\_csv('/content/News.csv', delimiter = ';')

bisnes = pd.read\_csv('/content/NewsBisnes.csv', delimiter = ';')

culture = pd.read\_csv('/content/NewsCulture.csv', delimiter = ';')

politics = pd.read\_csv('/content/NewsPolitics.csv', delimiter = ';')

# Создадим одну таблицу со всеми данными

news = pd.concat([sport, bisnes, culture, politics], axis=0)

Посмотрим информацию о нашем датасете:

news.info()

<class 'pandas.core.frame.DataFrame'>

Int64Index: 1494 entries, 0 to 757

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Title 1494 non-null object

1 Дата 1494 non-null object

2 Ссылка 1494 non-null object

3 Text 1494 non-null object

4 Topic 1494 non-null object

dtypes: object(5)

memory usage: 70.0+ KB

Добавим в текст заголовки и удалим лишние столбцы из датасета (рис.1):

news['Text']=news['Text']+" "+news['Title']

news.drop(['Дата','Ссылка','Title'],axis=1,inplace=True)

news.head()

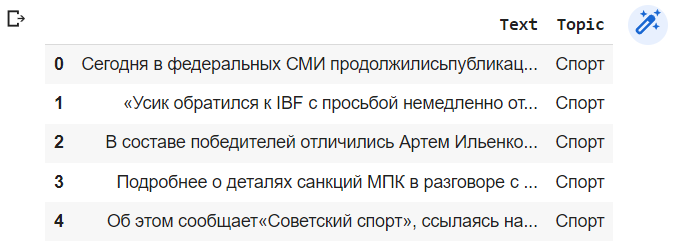


Рисунок 1 − Данные news

Перемешаем данные и выведем первые 5 строк датасета (рис.2):

# Перемешаем данные

news = shuffle(news).reset\_index(drop=True)

news.head()

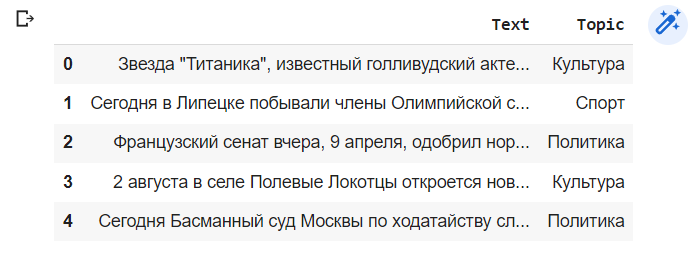


Рисунок 2 − Обновленные данные news

Выведем наши категории:

topics = news['Topic'].unique()

print(topics, len(topics))

['Культура' 'Спорт' 'Политика' 'Бизнес'] 4

На этом этап импорта данных закончен. Дальше нужно очистить данные.

Создадим функцию для очистки пунктуации, символов, стоп-слов, приведения текста к нижнему регистру, токенизации и лематизации текста.

Лематизация ставит формы каждого слова в общую основу или корень с учетом морфологического анализа слов.

stopwords\_russian = stopwords.words('russian')

def clean\_text(text):

  lemmatizer = WordNetLemmatizer()

  stopwords\_russian = stopwords.words('russian')

  text= re.sub('\[[^]]\*\]', '', text)

  # remove stock market tickers like $GE

  text = re.sub(r'\$\w\*', '', text)

  #removal of html tags

  review =re.sub(r'<.\*?>',' ',text)

  # remove old style retweet text "RT"

  text = re.sub(r'^RT[\s]+', '', text)

  # remove hyperlinks

  text = re.sub(r'https?:\/\/.\*[\r\n]\*', '', text)

  # remove hashtags

  # only removing the hash # sign from the word

  text = re.sub(r'#', '', text)

  text = re.sub("["

                 u"\U0001F600-\U0001F64F" # removal of emoticons

                 u"\U0001F300-\U0001F5FF" # symbols & pictographs

                 u"\U0001F680-\U0001F6FF" # transport & map symbols

                 u"\U0001F1E0-\U0001F1FF" # flags (iOS)

                 u"\U00002702-\U000027B0"

                 u"\U000024C2-\U0001F251"

                 "]+",' ',text)

  text = re.sub('[^a-zA-Z]',' ',text)

  text = text.lower()

  text\_tokens =word\_tokenize(text)

  text\_clean = []

  for word in text\_tokens:

    if (word not in stopwords\_russian and # remove stopwords

          word not in string.punctuation): # remove punctuation

        lem\_word =lemmatizer.lemmatize(word) # lemmitiging word

        text\_clean.append(lem\_word)

  text\_mod=[i for i in text\_clean if len(i)>2]

  text\_clean=' '.join(text\_mod)

  return text\_clean

Создадим переменную с текстом из таблицы:

texts = news['Text']

Используем класс TfidfVectorizer. Он дает меру, учитывающую важность слова в зависимости от того, как часто оно встречается в документе и корпусе.

from sklearn.feature\_extraction.text import TfidfVectorizer

n\_featur=200000

tfidf\_vectorizer = TfidfVectorizer(max\_df=0.8, max\_features=10000,

                                 min\_df=0.01, stop\_words=stopwords\_russian,

                                 use\_idf=True, tokenizer=clean\_text, ngram\_range=(1,3))

Будем использовать максимальное количество слов 10000, выбирать слова с частотой от 0,01 до 0,8, стоп слова – русские, в качестве токенизатора используем нашу функцию. Ngram\_range – будет рассматривать не только единичные слова, но и пары слов.

Применим tfidf на наших текстах:

tfidf\_matrix = tfidf\_vectorizer.fit\_transform(texts)

Импортируем и создадим модель K-means:

num\_clusters = 4

# Метод к-средних - KMeans

from sklearn.cluster import KMeans

km = KMeans(n\_clusters=num\_clusters, init = 'k-means++')

Обучим нашу модель:

idx = km.fit(tfidf\_matrix)

clusters = km.labels\_.tolist()

Посмотрим полученные данные:

print(len(km.labels\_))

clusters[:10]

1494

[0, 1, 3, 0, 1, 1, 1, 1, 0, 1]

Создадим таблицу с нашими данными и добавим принадлежность к кластеру каждой новости (рис. 3):

clusterkm = km.labels\_.tolist()

#k-means

out = { 'text': texts, 'cluster': clusterkm, 'topic': news['Topic'] }

frame1 = pd.DataFrame(out, columns = ['text', 'cluster', 'topic'])

frame1.head(10)

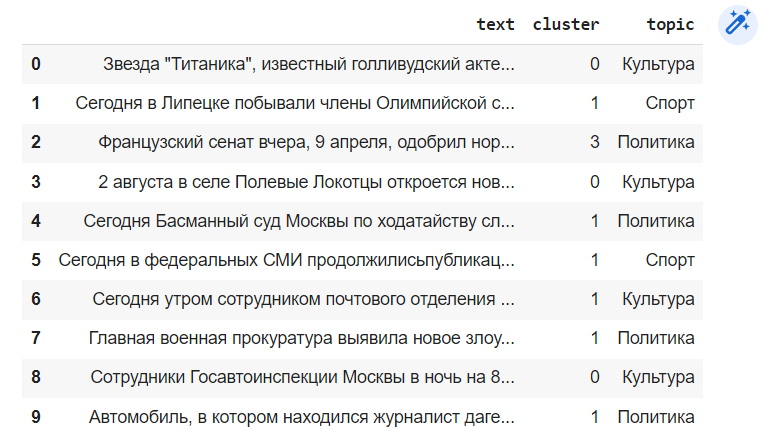


Рисунок 3 − frame1

И создадим еще одну таблицу, где представим распределение данных по кластерам и типам новостей (рис. 4):

clust\_res = pd.DataFrame(columns=['topic', 'cluster\_0', 'cluster\_1', 'cluster\_2', 'cluster\_3'])

i = 0

for topic in topics:

    cnt = []

    for cluster in range(num\_clusters):

        cnt.append(len(frame1[ frame1.topic.eq(topic) &  frame1.cluster.eq(cluster) ]))

    clust\_res.loc[i] = [topic] + cnt

    i += 1

clust\_res

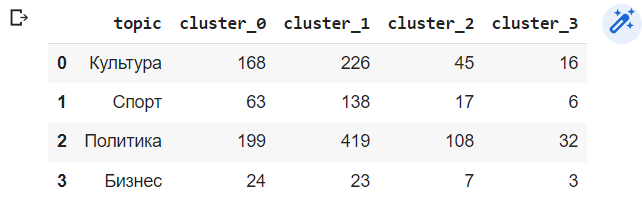


Рисунок 4 − clust\_res

Можно предположить, что культура принадлежит 0 кластеру, а политика - 1. На счет спорта и бизнеса все не так однозначно. Возможно, на такое распределение повлияло малое количество новостей данных групп.

Проведем тестирование нашей модели, передадим текста, которые мы брали для проверки классификации:

# Спорт

econ\_text = '''

Российский боксер Арест Саакян скончался через несколько дней после нокаута в бое на турнире «Короли нокаутов», сообщила сестра спортсмена Светлана Петросян на своей странице в Instagram.

Турнир состоялся 26 декабря в Тольятти. От полученных травм после нокаута Саакян впал в кому. У него был диагностирован отек головного мозга. Врачи несколько дней боролись за жизнь 26-летнего боксера, но он скончался, так и не придя в сознание.

'''

text\_vec = tfidf\_vectorizer.transform([econ\_text])

km.predict(text\_vec)

array([0], dtype=int32)

# Политика

econ\_text = '''

США рассчитывают на сотрудничество с Россией по Международной космической станции по крайней мере до 2030 года, заявил глава NASA Билл Нельсон, его цитирует РИА «Новости».

«У нас есть все основания полагать, что русские будут [продолжать работу по] космической станции в ближайшем будущем», — сказал он на слушаниях в конгрессе.

Нельсон также отметил, что организация надеется на продолжение сотрудничества по программе до 2030 года и что его слова основываются на существующих между NASA и «Роскосмосом» «профессиональных отношений». Он также указал, что информация по поводу выхода России из программы МКС неверна.

Ранее глава «Роскосмоса» Дмитрий Рогозин сообщил, что Россия определилась со сроками завершения работы на МКС. Он не назвал сроки, но уточнил, что РФ продолжает работу на орбитальной станции до 2024 года. По мнению эксперта Института космических исследований РАН Натан Эйсмонт заявил, что эксплуатация МКС станет практически невозможной, если Россия выйдет из программы.

'''

text\_vec = tfidf\_vectorizer.transform([econ\_text])

km.predict(text\_vec)

array([0], dtype=int32)

Как видим, модель неверно предсказала категорию новостей. Причиной может быть разность данных в датасете, обработка текста, может нужно было использовать стемминг или др.

Можете поэкспериментировать, спарсить больше данных, сделать равную по количеству выборку новостей и провести кластеризацию заново. Можете уменьшить количество типов новостей.

**Вывод**

В результате выполнения данной лабораторной работы познакомились с кластеризацией текста. Выполнили загрузку данных. Очистили данные от стоп-слов, пунктуации и ненужных символов. Провели лематизацию и токенизацию. Создали, обучили и протестировали модель для кластеризации текста.

**Критерии оценки**

…