УДК 004.8

ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА В ПРИКЛАДНОЙ ЗАДАЧЕ РАНЖИРОВАНИЯ СЛОЖНОСТИ ФИЛОСОФСКИХ ТРУДОВ ПО АВТОРАМ ПРОИЗВЕДЕНИЙ

Пылов П.А., студент гр. ИТм-201, 2 курс

Ивина О.А., к.т.н., доцент

Кузбасский государственный технический университет
имени Т.Ф. Горбачева

Аннотация: Данная статья посвящена решению одной из задач, которые были сформулированы в проекте Philosophy Data Project. Проект посвящен применению инструментов науки о данных к истории философии, главной целью которой является систематизация организации мыслей о мире. Однако, в рамках проекта внимание сфокусировано не на анализе сентиментов, а на концептуальном и идеологическом подходе. В статье представлено решение, позволяющее в автоматизированном формате ранжировать сложность чтения трудов отдельных философов по сравнению с их коллегами.

Основанием для написания статьи стала опубликованная задача проекта [2] с открытым исходным датасетом, представленным на международной соревновательной площадке специалистов по исследованию данных Kaagle [3].

Для начала определим, что языком программирования для решения поставленной задачи будет Python 3.9.2, поскольку он очень удобен и прост в применение, а также содержит в себе все необходимые библиотеки для выполнения математических и графических операций. Загрузка стандартного набора библиотек для обработки данных представлена на рисунке 1.

```
import numpy as np
import pandas as pd
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
from PIL import Image
import time
import math
from matplotlib.offsetbox import AnnotationBbox, OffsetImage
import ipywidgets as widgets
import plotly.graph_objects as go
from ipywidgets import Layout
```

Рисунок 1 – Загрузка вспомогательных библиотек

Следующим шагом загружаем данные авторов и принадлежащие им цитаты (рисунок 2).

```
DataFrame = pd.read_csv("/kaggle/input/history-of-philosophy/phil_nlp.csv")
DF["NumOfWords"] = DF["word_sentence"].apply(lambda x: len(x.split(" ")))
```

Рисунок 2 – Загрузка данных

Кроме этого, для сепарации сложности формулировки мыслей философов в реализуемую модель внесены наиболее часто встречающиеся слова английского алфавита под понятием «общих слов» (common words), частичный фрагмент представлен на рисунке 3. При помощи данного набора будет облегчена задача определения набора уникальных слов и понятий отдельных философов, а также открывается возможность ранжировать философов по «порогу входа» начинающего, неподготовленного читателя.

```
CommonWords="""#
the
of
and
to
a
in
for
is
on
that
by
this
with
i
you
it
not
or
be
are
from
at
as
your
all
have
```

Рисунок 3 – Внесение словаря «общих слов»

Теперь, когда присутствует словарь общеупотребительных слов, можно оценить сколько сложных терминологических понятий использует каждый философ. Для этого воспользуемся следующей формулой (1)

$$FrequenceUncommonWords = \frac{\sum_{x=FirstIndex}^{LastIndex} AmountTermins}{\sum_{x=FirstIndex}^{LastIndex} NumberOfWords} \quad (1)$$

Таким образом, частота сложных терминов определяется как отношение «сложных слов» к общему числу слов в предложение. Составим процедуру расчета сложности чтения трудов каждого философа в программной реализации (рисунок 4). Результаты определения сложности каждого философа из-за употребления сложных слов ранжированы по частоте употребления и наиболее удобны для представления в виде диаграммы (рисунок 5).

```
cmap = plt.get_cmap("viridis")
rescale = lambda y: (y - np.min(y)) / (np.max(y) - np.min(y))

Uncommonness = (DataFrame.groupby("author").sum()["AmountTermins"] / DataFrame.groupby("author").sum()["NumberOfWords"]).sort_values()
fig,ax = plt.subplots(figsize=(30,10))
plt.bar(Uncommonness.index,Uncommonness.values,color=cmap(rescale(Uncommonness.values)))
ax.tick_params(labelsize=15)
for item in ax.xaxis.get_ticklabels():
    item.set_rotation(40)
plt.xlabel('Philosopher', fontsize=21)
plt.ylabel('Uncommon word density', fontsize=21)
plt.title("Difficultness of reading according to use of uncommon words",fontsize=28)
```

Рисунок 4 – Определение частоты сложных слов в трудах философов

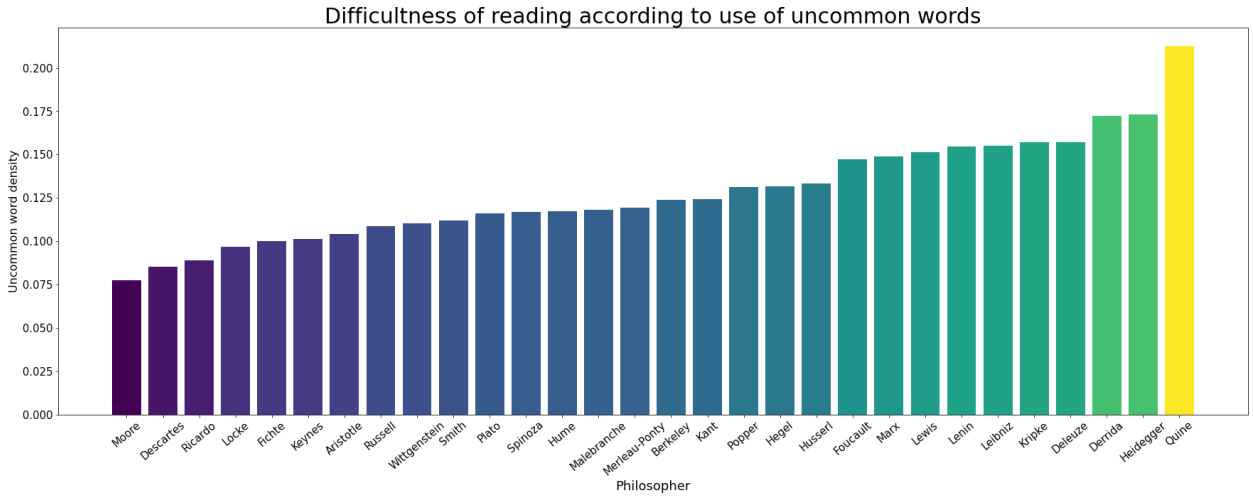

Рисунок 5 – Представление сложности чтения трудов философа на основании частоты употребления сложных слов

Аналогичным образом можно поступить для ранжирования сложности чтения философских школ, а не только авторов в целом. Для этого сконфигурируем соответствующий программный код (рисунок 6) и отобразим результат его деятельности на рисунке 7.

```
Uncommonness=(DataFrame.groupby("school").sum()["AmountTermins"]/DataFrame.groupby("school").sum()["NumberOfWords"]).sort_values()
fig,ax=plt.subplots(figsize=(30,10))
plt.bar(Uncommonness.index,Uncommonness.values,color=my_cmap(rescale(Uncommonness.values)))
ax.tick_params(labelsize=15)
for item in ax.xaxis.get_ticklabels():
    item.set_rotation(40)
plt.xlabel('School', fontsize=21)
plt.ylabel('Uncommon word density', fontsize=21)
plt.title("Difficultness of reading according to use of uncommon words",fontsize=28)
```

Рисунок 6 – Процедура ранжирования философских школ по сложности

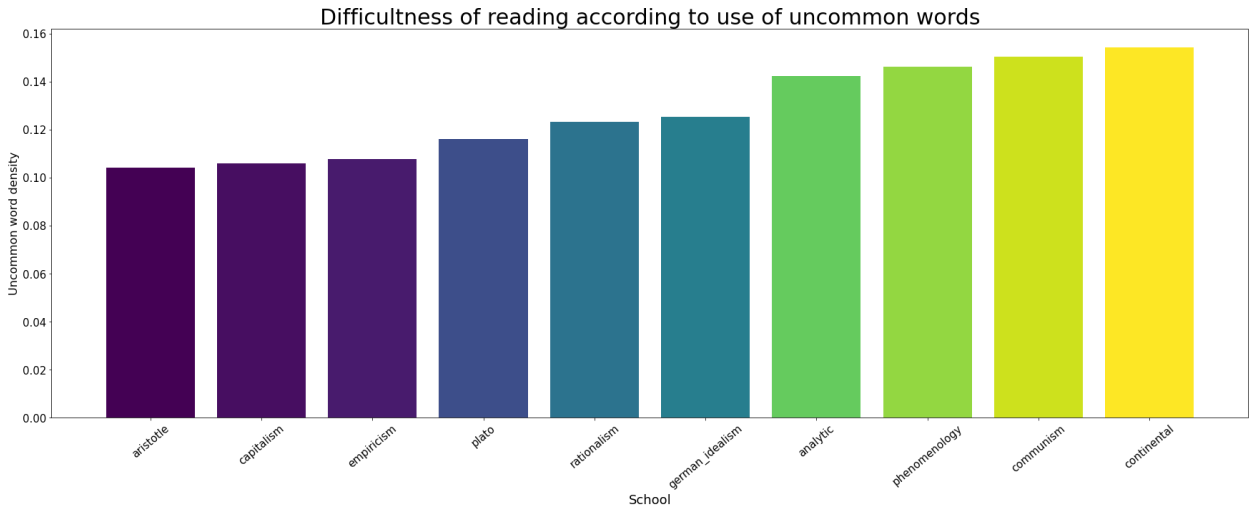

Рисунок 7 – Ранжирования сложности чтения философских школ на основании частоты употребления ими сложной лексики

Представленный в статье пример позволяет убедиться в том, что обработка естественного языка позволяет гибко адаптироваться к прикладным областям не только технического, но и гуманитарного направления, позволяя разрешать задачи до недавнего времени остававшиеся исключительно прерогативой человека.

Список литературы:

1. THE PHILOSOPHY DATA PROJECT [Электронный ресурс]. – Режим доступа: http://philosophydata.com/
2. History of Philosophy. Sentences taken from 51 texts spanning the history of philosophy [Электронный ресурс]. – Режим доступа: https://www.kaggle.com/kouroshalizadeh/history-of-philosophy/
3. Система организации конкурсов по исследованию данных, а также социальная сеть специалистов по обработке данных и машинному обучению [Электронный ресурс]. – Режим доступа: https://www.kaggle.com/