



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ _____

КАФЕДРА ИУ5 _____

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

Решение задачи машинного обучения

Студент группы РТ5-61
(Группа)

_____ Калин В.Д.
(Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта

_____ Гапанюк Ю.Е.
(Подпись, дата) (И.О.Фамилия)

Консультант

_____ (Подпись, дата) (И.О.Фамилия)

2020 г.

Министерство науки и высшего образования Российской Федерации Федеральное
государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный
исследовательский университет)» (МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой _____
(Индекс)

(И.О.Фамилия)
« ____ » _____ 20 ____ г.

ЗАДАНИЕ
на выполнение курсового проекта

по дисциплине «Технологии машинного обучения» _____

Студент группы РТ5-61 _____

_____ Калинин Владимир Дмитриевич _____
(Фамилия, имя, отчество)

Тема курсового проекта _____

Направленность КП (учебный, исследовательский, практический, производственный, др.) _____

Источник тематики (кафедра, предприятие, НИР) _____

Задание решение задачи машинного обучения на основе материалов дисциплины. Выполняется студентом единолично.

Оформление курсового проекта:

Расчетно-пояснительная записка на __16__ листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

Дата выдачи задания « 12 » февраля 2020 г.

Руководитель курсового проекта _____ Гапанюк Ю.Е.
(Подпись, дата) (И.О.Фамилия)

Студент _____ Калинин В.Д.
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Оглавление

1. Задание установленного образца	4
2. Введение	5
3. Основная часть.	6
3.1 Описание набора данных	6
3.2 Ход работы	7
4. Выводы по проделанной работе	18
5. Список использованных источников	18

Задание установленного образца

Схема типового исследования, проводимого студентом в рамках курсовой работы, содержит выполнение следующих шагов:

- Поиск и выбор набора данных для построения моделей машинного обучения. На основе выбранного набора данных студент должен построить модели машинного обучения для решения или задачи классификации, или задачи регрессии.
- Проведение разведочного анализа данных. Построение графиков, необходимых для понимания структуры данных. Анализ и заполнение пропусков в данных.
- Выбор признаков, подходящих для построения моделей. Кодирование категориальных признаков Масштабирование данных. Формирование вспомогательных признаков, улучшающих качество моделей.
- Проведение корреляционного анализа данных. Формирование промежуточных выводов о возможности построения моделей машинного обучения. В зависимости от набора данных, порядок выполнения пунктов 2, 3, 4 может быть изменен.
- Выбор метрик для последующей оценки качества моделей. Необходимо выбрать не менее трех метрик и обосновать выбор.
- Выбор наиболее подходящих моделей для решения задачи классификации или регрессии. Необходимо использовать не менее пяти моделей, две из которых должны быть ансамблевыми.
- Формирование обучающей и тестовой выборки на основе исходного набора данных.
- Построение базового решения (baseline) для выбранных моделей без подбора гиперпараметров. Производятся

обучение моделей на основе обучающей выборки и оценка качества моделей на основе тестовой выборки.

- Подбор гиперпараметров для выбранных моделей.
Рекомендуется использовать методы кросс-валидации. В зависимости от используемой библиотеки можно применять функцию GridSearchCV, использовать перебор параметров в цикле, или использовать другие методы.
- Повторение пункта 8 для найденных оптимальных значений гиперпараметров. Сравнение качества полученных моделей с качеством baseline-моделей.
- Формирование выводов о качестве построенных моделей на основе выбранных метрик. Результаты сравнения качества рекомендуется отобразить в виде графиков и сделать выводы в форме текстового описания.
Рекомендуется построение графиков обучения и валидации, влияния значений гиперпараметров на качество моделей и т.д.

Введение

Курсовой проект – самостоятельная часть учебной дисциплины «Технологии машинного обучения» – учебная и практическая исследовательская студенческая работа, направленная на решение комплексной задачи машинного обучения. Результатом курсового проекта является отчет, содержащий описания моделей, тексты программ и результаты экспериментов.

Курсовой проект опирается на знания, умения и владения, полученные студентом в рамках лекций и лабораторных работ по дисциплине.

Основная часть. Описание постановки задачи и последовательности действий по решению поставленной задачи

Описание набора данных

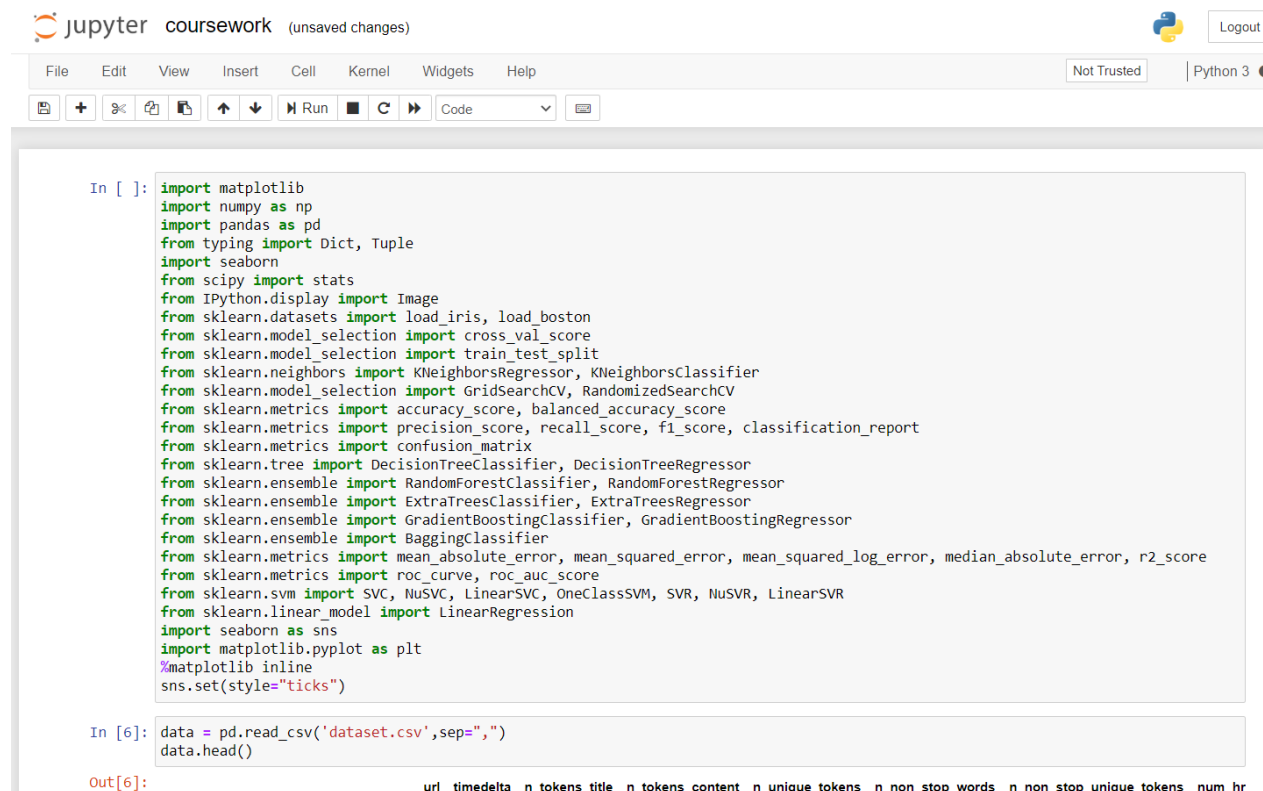
В данной работе для исследований был выбран следующий датасет: <https://archive.ics.uci.edu/ml/machine-learning-databases/00332/>

Задача: анализ популярности новостей на онлайн сервисах поисковых запросов и предсказание динамики процесса.

Ход работы

В ходе данной работы были разработаны алгоритмы анализа входного датасета для достижения поставленной задачи.

Реализация алгоритмов велась на языке Python в среде разработки PyCharm. Запуск и демонстрация алгоритмов производились на базе сервера Jupyter Notebook, который запускается из-под среды разработки PyCharm. Ниже приведены результаты работы алгоритмов на некоторых входных данных:



The screenshot shows a Jupyter Notebook interface with the title 'coursework (unsaved changes)'. The interface includes a top menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. Below the menu is a toolbar with icons for file operations, code execution, and output viewing. The main area contains two code cells. The first cell, labeled 'In []:', contains a series of import statements for various Python libraries including matplotlib, numpy, pandas, typing, seaborn, scipy, IPython, sklearn, and statsmodels. The second cell, labeled 'In [6]:', contains code to read a CSV file named 'dataset.csv' and display its first few rows. The output of the second cell is shown below the code, displaying a table with columns: url, timedelta, n_tokens_title, n_tokens_content, n_unique_tokens, n_non_stop_words, n_non_stop_unique_tokens, and num_hr.

```
In [ ]: import matplotlib
import numpy as np
import pandas as pd
from typing import Dict, Tuple
import seaborn
from scipy import stats
from IPython.display import Image
from sklearn.datasets import load_iris, load_boston
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score, classification_report
from sklearn.metrics import confusion_matrix
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from sklearn.ensemble import ExtraTreesClassifier, ExtraTreesRegressor
from sklearn.ensemble import GradientBoostingClassifier, GradientBoostingRegressor
from sklearn.ensemble import BaggingClassifier
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_squared_log_error, median_absolute_error, r2_score
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR, NuSVR, LinearSVR
from sklearn.linear_model import LinearRegression
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")

In [6]: data = pd.read_csv('dataset.csv', sep=',')
data.head()

Out[6]:
```

url	timedelta	n_tokens_title	n_tokens_content	n_unique_tokens	n_non_stop_words	n_non_stop_unique_tokens	num_hr
-----	-----------	----------------	------------------	-----------------	------------------	--------------------------	--------

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

Python 3

```
In [2]: data = pd.read_csv('dataset.csv', sep=",")
data.head()
```

```
Out[2]:
```

	url	timedelta	n_tokens_title	n_tokens_content	n_unique_tokens	n_non_stop_words	n_non_stop_unique_tokens	num_hr
0	http://mashable.com/2013/01/07/amazon-instant-...	731.0	12.0	219.0	0.663594	1.0	0.815385	
1	http://mashable.com/2013/01/07/ap-samsung-spon...	731.0	9.0	255.0	0.604743	1.0	0.791946	
2	http://mashable.com/2013/01/07/apple-40-billio...	731.0	9.0	211.0	0.575130	1.0	0.663866	
3	http://mashable.com/2013/01/07/astronaut-notre...	731.0	9.0	531.0	0.503788	1.0	0.665635	
4	http://mashable.com/2013/01/07/att-u-verse-apps/	731.0	13.0	1072.0	0.415646	1.0	0.540890	1

5 rows x 61 columns

```
In [3]: data.shape
```

```
Out[3]: (39644, 61)
```

```
In [4]: data.columns
```

```
Out[4]: Index(['url', 'timedelta', 'n_tokens_title', 'n_tokens_content',
'n_unique_tokens', 'n_non_stop_words', 'n_non_stop_unique_tokens',
'num_hrefs', 'num_self_hrefs', 'num_imgs', 'num_videos',
'average_token_length', 'num_keywords', 'data_channel_is_lifestyle',
'data_channel_is_entertainment', 'data_channel_is_bus',
'data_channel_is_socmed', 'data_channel_is_tech',
'data_channel_is_world', 'kw_min_min', 'kw_max_min', 'kw_avg_min',
'kw_min_max', 'kw_max_max', 'kw_avg_max', 'kw_min_avg',
'kw_max_avg', 'kw_avg_avg', 'self_reference_min_shares',
```

File Edit View Insert Cell Kernel Widgets Help

Not Trusted

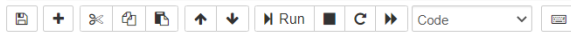
Python 3

```
In [5]: data.dtypes
```

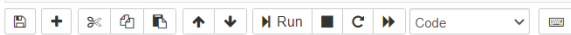
```
Out[5]: url                object
timedelta              float64
n_tokens_title         float64
n_tokens_content       float64
n_unique_tokens       float64
...
title_subjectivity    float64
title_sentiment_polarity float64
abs_title_subjectivity float64
abs_title_sentiment_polarity float64
shares                int64
Length: 61, dtype: object
```

```
In [6]: for col in data.columns:
# Количество пустых значений - все значения заполнены
temp_null_count = data[data[col].isnull()].shape[0]
print('{} - {}'.format(col, temp_null_count))
```

```
url - 0
timedelta - 0
n_tokens_title - 0
n_tokens_content - 0
n_unique_tokens - 0
n_non_stop_words - 0
n_non_stop_unique_tokens - 0
num_hrefs - 0
num_self_hrefs - 0
num_imgs - 0
num_videos - 0
average_token_length - 0
num_keywords - 0
data_channel_is_lifestyle - 0
data_channel_is_entertainment - 0
data_channel_is_bus - 0
```



```
kw_max_max - 0
kw_avg_max - 0
kw_min_avg - 0
kw_max_avg - 0
kw_avg_avg - 0
self_reference_min_shares - 0
self_reference_max_shares - 0
self_reference_avg_shares - 0
weekday_is_monday - 0
weekday_is_tuesday - 0
weekday_is_wednesday - 0
weekday_is_thursday - 0
weekday_is_friday - 0
weekday_is_saturday - 0
weekday_is_sunday - 0
is_weekend - 0
LDA_00 - 0
LDA_01 - 0
LDA_02 - 0
LDA_03 - 0
LDA_04 - 0
global_subjectivity - 0
global_sentiment_polarity - 0
global_rate_positive_words - 0
global_rate_negative_words - 0
rate_positive_words - 0
rate_negative_words - 0
avg_positive_polarity - 0
min_positive_polarity - 0
max_positive_polarity - 0
avg_negative_polarity - 0
min_negative_polarity - 0
max_negative_polarity - 0
title_subjectivity - 0
title_sentiment_polarity - 0
abs_title_subjectivity - 0
```

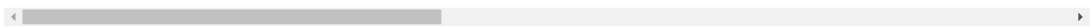


```
In [7]: # Основные статистические характеристики набора данных
data.describe()
```

```
Out[7]:
```

	timedelta	n_tokens_title	n_tokens_content	n_unique_tokens	n_non_stop_words	n_non_stop_unique_tokens	num_hrefs	num_self_hrefs	num_
count	39644.000000	39644.000000	39644.000000	39644.000000	39644.000000	39644.000000	39644.000000	39644.000000	39644.000000
mean	354.530471	10.398749	546.514731	0.548216	0.996469	0.689175	10.883690	3.293638	4.54
std	214.163767	2.114037	471.107508	3.520708	5.231231	3.264816	11.332017	3.855141	8.30
min	8.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
25%	164.000000	9.000000	246.000000	0.470870	1.000000	0.625739	4.000000	1.000000	1.00
50%	339.000000	10.000000	409.000000	0.539226	1.000000	0.690476	8.000000	3.000000	1.00
75%	542.000000	12.000000	716.000000	0.608696	1.000000	0.754630	14.000000	4.000000	4.00
max	731.000000	23.000000	8474.000000	701.000000	1042.000000	650.000000	304.000000	116.000000	128.00

8 rows x 60 columns

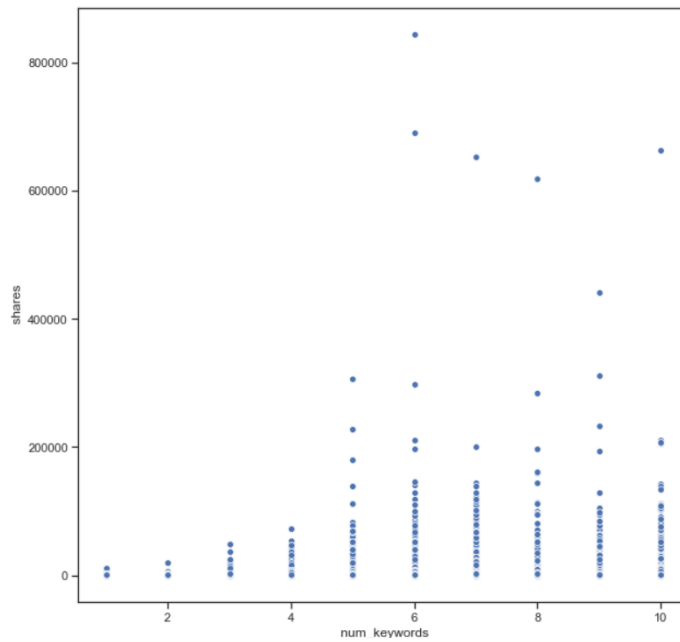


```
In [8]: fig, ax = matplotlib.pyplot.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x=' num_keywords', y=' shares', data=data)
```

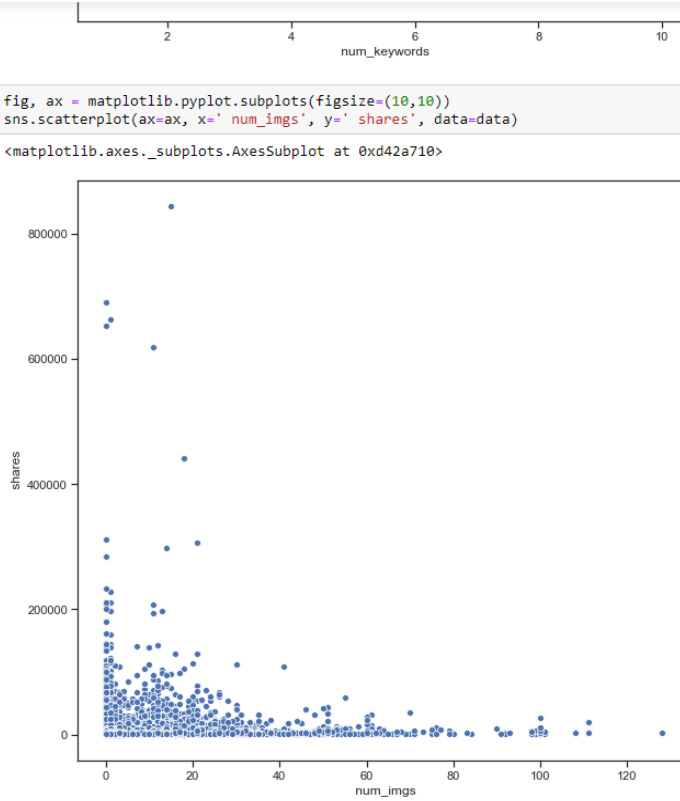
```
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0xf19abf0>
```



Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0x119a0107>



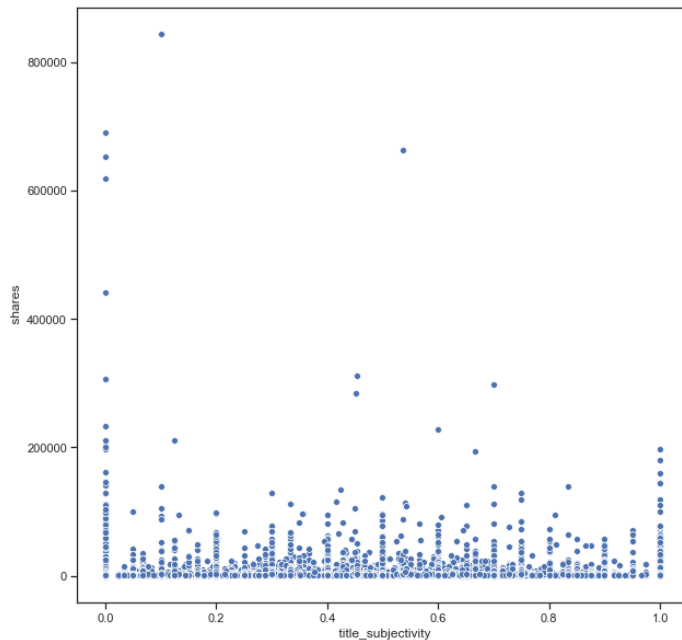
Out[9]: <matplotlib.axes._subplots.AxesSubplot at 0xd42a710>



In [10]: fig, ax = plt.subplots(figsize=(10,10))

```
In [10]: fig, ax = plt.subplots(figsize=(10,10))
sns.scatterplot(ax=ax, x='title_subjectivity', y='shares', data=data)
```

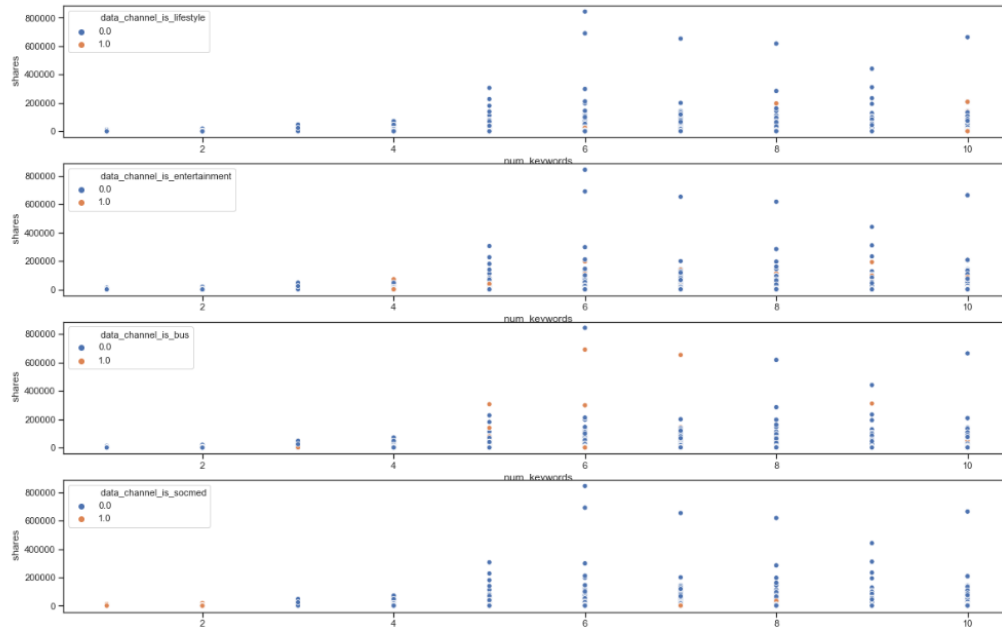
```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0xd2ddb50>
```



```
In [11]: fig, (ax1, ax2, ax3, ax4, ax5, ax6) = plt.subplots(nrows=6, ncols=1, figsize=(20,20))
sns.scatterplot(ax=ax1, x='num_keywords', y='shares', data=data, hue='data_channel_is_lifestyle')
sns.scatterplot(ax=ax2, x='num_keywords', y='shares', data=data, hue='data_channel_is_entertainment')
```

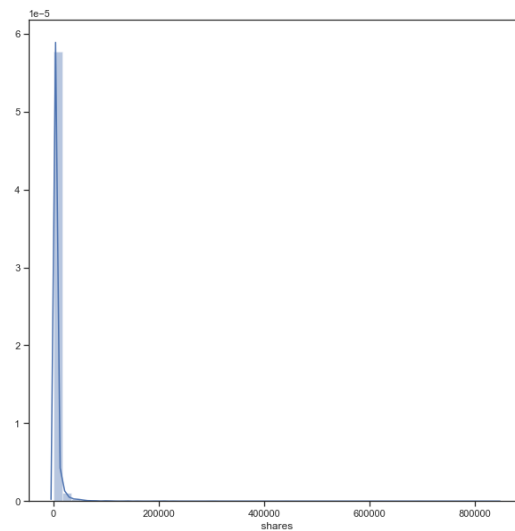
```
In [11]: fig, (ax1, ax2, ax3, ax4, ax5, ax6) = plt.subplots(nrows=6, ncols=1, figsize=(20,20))
sns.scatterplot(ax=ax1, x='num_keywords', y='shares', data=data, hue='data_channel_is_lifestyle')
sns.scatterplot(ax=ax2, x='num_keywords', y='shares', data=data, hue='data_channel_is_entertainment')
sns.scatterplot(ax=ax3, x='num_keywords', y='shares', data=data, hue='data_channel_is_bus')
sns.scatterplot(ax=ax4, x='num_keywords', y='shares', data=data, hue='data_channel_is_socmed')
sns.scatterplot(ax=ax5, x='num_keywords', y='shares', data=data, hue='data_channel_is_tech')
sns.scatterplot(ax=ax6, x='num_keywords', y='shares', data=data, hue='data_channel_is_world')
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0xd7cc310>



```
In [12]: fig, ax = plt.subplots(figsize=(10,10))
sns.distplot(data[['shares']])

Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0xdab2630>
```

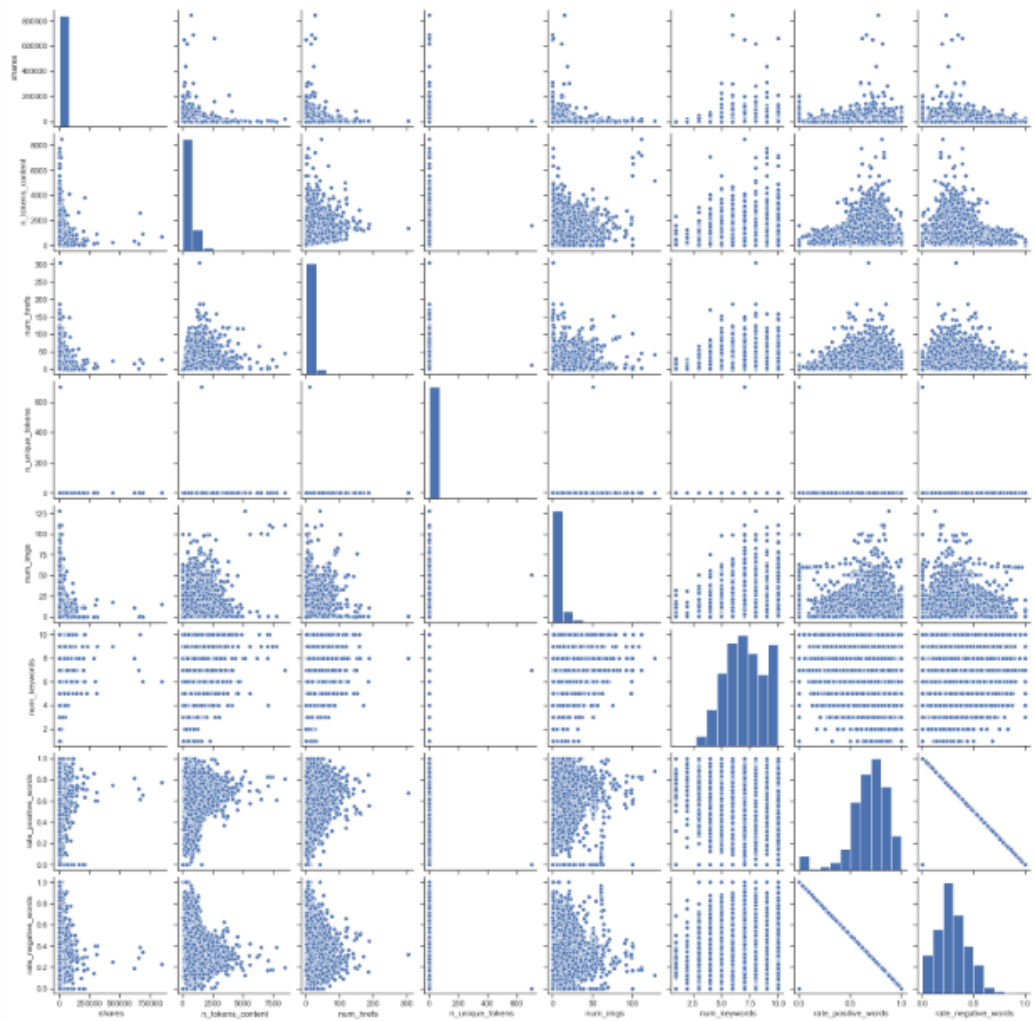


```
In [13]: sns.pairplot(data, vars=['shares', 'n_tokens_content', 'num_hrefs', 'n_unique_tokens',
                                'num_imgs', 'num_keywords', 'rate_positive_words', 'rate_negative_words'])
```

Out[13]: <matplotlib.figure.Figure at 0xd904010>

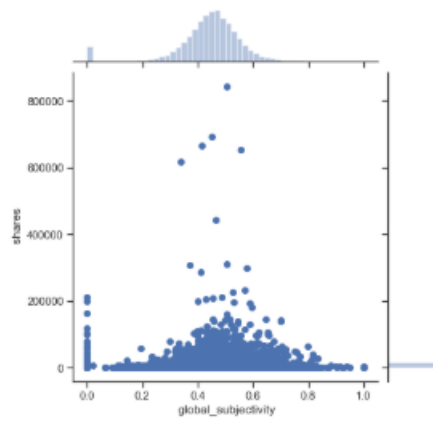
```
In [13]: sns.pairplot(data, vars=['shares', 'n_tokens_content', 'num_hrefs', 'n_unique_tokens',
    , 'num_imgs', 'num_keywords', 'rate_positive_words', 'rate_negative_words'])
```

```
Out[13]: <seaborn.axisgrid.PairGrid at 0xe09ba10>
```



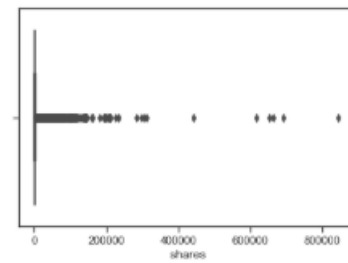
```
In [14]: sns.jointplot(x=' global_subjectivity', y=' shares', data=data)
```

```
Out[14]: <seaborn.axisgrid.JointGrid at 0x16fe47b0>
```



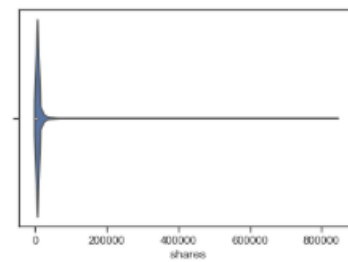
```
In [15]: sns.boxplot(x=data[' shares'])
```

```
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x3c6c90>
```



```
In [16]: sns.violinplot(x=data[' shares'])
```

```
Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0xbd0b10>
```



In [17]: data.corr()

Out[17]:		timedelta	n_tokens_title	n_tokens_content	n_unique_tokens	n_non_stop_words	n_non_stop_unique_tokens	num_hrefs	num_imgs
	timedelta	1.000000	-0.240320	-0.062867	0.002868	0.000089	0.003805	-0.000832	
	n_tokens_title	-0.240320	1.000000	0.018160	-0.005318	-0.004754	-0.005420	-0.053496	
	n_tokens_content	-0.062867	0.018160	1.000000	-0.004737	0.017512	0.000373	0.423065	
	n_unique_tokens	0.002868	-0.005318	-0.004737	1.000000	0.999572	0.999572	-0.004352	
	n_non_stop_words	0.000089	-0.004754	0.017512	0.999572	1.000000	0.999532	0.005521	
	n_non_stop_unique_tokens	0.003805	-0.005420	0.000373	0.999572	0.999532	1.000000	-0.004983	
	num_hrefs	-0.000832	-0.053496	0.423065	-0.004352	0.005521	-0.004983	1.000000	
	num_imgs	0.064530	-0.014856	0.304682	0.006620	0.013598	0.007584	0.396452	
	num_videos	-0.027636	-0.008858	0.342800	0.018802	0.028486	0.014230	0.342633	
	average_token_length	0.000936	0.051460	0.103699	-0.000597	-0.000899	-0.000963	0.114518	
	num_keywords	0.130465	-0.071403	0.167789	0.026407	0.031554	0.034185	0.222588	
	data_channel_is_lifestyle	0.046884	-0.006077	0.072845	-0.003679	-0.001439	-0.004440	0.125890	
	data_channel_is_entertainment	0.054492	-0.070815	0.037548	-0.001653	-0.000314	-0.000417	0.052906	
	data_channel_is_bus	-0.049109	0.132791	0.060200	0.011016	0.010903	0.010554	-0.007968	
	data_channel_is_socmed	0.055788	-0.023902	-0.006105	-0.000264	-0.000012	0.001840	-0.058360	
	data_channel_is_tech	0.076287	-0.090394	0.033424	-0.000945	-0.000078	-0.000526	0.050470	
	data_channel_is_world	0.083277	-0.046716	0.025408	-0.002328	0.000061	-0.000921	-0.061734	
	kw_min_min	-0.170250	0.049223	0.055989	-0.005535	-0.002702	-0.003801	-0.031567	
	kw_max_min	0.591199	-0.110672	-0.054345	0.001601	-0.000352	0.002001	-0.043263	
	kw_avg_min	0.029503	-0.005890	0.000068	-0.000552	-0.000595	-0.000615	0.012844	
	kw_min_max	0.133225	-0.031400	-0.003545	-0.000826	-0.000892	-0.000644	0.008307	
	kw_max_max	-0.076590	0.012926	-0.022786	0.000577	-0.000553	-0.000089	-0.020150	
	kw_avg_max	-0.637824	0.120841	0.058860	-0.001624	0.000394	-0.002032	0.051265	
	kw_min_avg	-0.493093	0.115746	-0.096460	0.000805	-0.002939	-0.002006	-0.019269	
	kw_max_avg	-0.157204	-0.002370	-0.022286	0.004563	0.003284	0.002510	0.058820	
	kw_avg_avg	-0.051820	0.006918	-0.030496	-0.002120	-0.003408	-0.003694	0.089692	
	self_reference_min_shares	-0.163164	0.004296	-0.079624	-0.002083	-0.005415	-0.005944	0.121419	
	self_reference_max_shares	-0.011438	-0.004563	-0.030686	0.001038	0.000339	0.000989	-0.004804	
	self_reference_avg_shares	-0.014501	0.000128	0.025857	-0.000222	0.000170	-0.000077	0.080394	
	weekday_is_monday	-0.015655	0.000661	-0.013809	0.001992	0.001614	0.002009	0.025239	
	weekday_is_tuesday	-0.006129	0.004274	-0.002484	-0.002142	-0.002147	-0.002042	-0.005759	
	weekday_is_wednesday	-0.005781	0.009322	-0.004027	0.010538	0.010501	0.010510	-0.010691	
	weekday_is_thursday	0.009961	0.008935	-0.016891	-0.002224	-0.002517	-0.002070	-0.032437	
	weekday_is_friday	0.004042	-0.015472	-0.007395	-0.002248	-0.002360	-0.002283	-0.012776	
	weekday_is_saturday	-0.002853	-0.002015	-0.015949	-0.001398	-0.001963	-0.001301	-0.001306	
	is_weekend	-0.004067	-0.015013	0.034538	-0.002563	-0.001508	-0.002623	0.054661	
	LDA_00	0.004226	0.006289	0.036394	-0.001803	-0.001338	-0.002167	0.044220	
	LDA_01	0.000272	-0.005996	0.052024	-0.003186	-0.002082	-0.003502	0.072279	
	LDA_02	0.080894	-0.070038	0.026218	-0.002213	-0.001031	0.000342	-0.020100	
	LDA_03	0.004423	0.063568	-0.009724	-0.000827	-0.002869	-0.000965	-0.053803	
	LDA_04	-0.141713	0.038365	0.087266	-0.008855	-0.003286	-0.004965	-0.012531	
	global_subjectivity	-0.030838	0.042208	-0.140141	-0.003689	-0.009761	-0.009781	0.123786	
	global_sentiment_polarity	0.092908	-0.065063	0.041265	-0.004260	-0.001072	-0.002097	-0.054977	
	global_rate_positive_words	0.133837	-0.056804	0.127879	-0.000180	0.002565	0.005498	0.203464	
	rate_positive_words	0.158646	-0.072226	0.021937	0.000523	0.000831	0.002043	0.088659	
	rate_negative_words	0.207604	-0.064951	0.133979	0.000014	0.001535	0.005002	0.056428	
	avg_positive_polarity	0.010266	0.015530	0.125013	-0.000877	0.001036	0.002590	0.032515	
	min_positive_polarity	0.198654	-0.065589	0.098960	-0.000667	0.002421	0.005258	0.101663	
	max_positive_polarity	-0.071968	0.034186	0.101053	-0.001657	0.001245	0.001709	0.059617	
		0.126344	-0.049619	0.135123	-0.000487	0.002285	0.004671	0.188236	
		0.054772	-0.025089	-0.261493	0.009193	0.000904	0.008380	-0.082168	
		0.098288	-0.021662	0.415706	-0.009054	0.002062	-0.002245	0.286733	

80 rows x 80 columns

```
In [18]: sharescorr=data.corr()  
sharescorr[' shares']
```

```
Out[18]: timedelta          0.008662  
n_tokens_title             0.008783  
n_tokens_content           0.002459  
n_unique_tokens            0.008086  
n_non_stop_words           0.008443  
n_non_stop_unique_tokens   0.008114  
num_hrefs                  0.045484  
num_self_hrefs             -0.001908  
num_ings                   0.039388  
num_videos                 0.023936  
average_token_length       -0.022087  
num_keywords               0.021818  
data_channel_is_lifestyle   0.005831  
data_channel_is_entertainment -0.017086  
data_channel_is_bus        -0.012376  
data_channel_is_socmed     0.005021  
data_channel_is_tech       -0.013253  
data_channel_is_world      -0.049497  
kw_min_min                 -0.001051  
kw_max_min                 0.038114  
kw_avg_min                 0.038486  
kw_min_max                 0.003981  
kw_max_max                 0.007863  
kw_avg_max                 0.044686  
kw_min_avg                 0.039551  
kw_max_avg                 0.064386  
kw_avg_avg                 0.110413  
self_reference_min_shares  0.055958  
self_reference_max_shares  0.047115  
self_reference_avg_shares  0.057789  
weekday_is_monday          0.009726  
weekday_is_tuesday         -0.007941  
weekday_is_wednesday       -0.003881  
weekday_is_thursday        -0.008833  
weekday_is_friday          -0.003884  
weekday_is_saturday        0.015082  
weekday_is_sunday          0.008238  
is_weekend                 0.016958  
LDA_00                     -0.003793  
LDA_01                     -0.010183  
LDA_02                     -0.059163  
LDA_03                     0.003771  
LDA_04                     -0.016622  
global_subjectivity         0.031684  
global_sentiment_polarity   0.004163  
global_rate_positive_words  0.008543  
global_rate_negative_words  0.006615  
rate_positive_words         -0.013241  
rate_negative_words         -0.005183  
avg_positive_polarity       0.012142  
min_positive_polarity       -0.008048  
max_positive_polarity       0.010068  
avg_negative_polarity       -0.032029  
min_negative_polarity       -0.019297  
max_negative_polarity       -0.019388  
title_subjectivity          0.021967  
title_sentiment_polarity    0.012772  
abs_title_subjectivity      0.001481  
abs_title_sentiment_polarity 0.027135  
shares                     1.000000  
Name: shares, dtype: float64
```

```
In [ ]: for col in pd.read_csv('dataset.csv', sep=',').columns:  
        if col != 'url':  
            print("Колонка {}".format(col))  
            print(pd.read_csv('dataset.csv', sep=',').corr()[col])
```

```
Колонка timedelta  
timedelta          1.000000  
n_tokens_title     -0.240320  
n_tokens_content   -0.062867  
n_unique_tokens    0.002866  
n_non_stop_words   0.008089  
n non stop unique tokens 0.003885
```

Выводы по проделанной работе

В ходе курсовой работы были закреплены полученные в течение курса знания и навыки. Для исследования использовались различные модели в

Список использованных источников

1. Конспект лекций по дисциплине “Технологии машинного обучения”. 2020:
https://github.com/ugapanyuk/ml_course_2020/wiki/COURSE_TMO
2. Документация scikit-learn:
<https://scikit-learn.org/stable/index.html>
3. Метрики в задачах машинного обучения:
<https://habr.com/ru/company/ods/blog/328372/>