МГТУ им. Н. Э. Баумана

Факультет: Информатика, искусственный интеллект и системы управления

Кафедра: Системы обработки информации и управления

Дисциплина: Методы машинного обучения

Лабораторная работа №5 "Предобработка и классификация текстовых данных"

Выполнил: Солохов И. Р. ИУ5-23М

1. Для произвольного предложения или текста решите следующие задачи:
   - Токенизация.
   - Частеречная разметка.
   - Лемматизация.
   - Выделение (распознавание) именованных сущностей.
   - Разбор предложения.
2. Для произвольного набора данных, предназначенного для классификации текстов, решите задачу классификации текста двумя способами:
   - Способ 1. На основе CountVectorizer или TfidfVectorizer.
   - Способ 2. На основе моделей word2vec или Glove или fastText.
   - Сравните качество полученных моделей.

In [90]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import datetime
import matplotlib.pyplot as plt
from sklearn.svm import SVR
from sklearn.impute import SimpleImputer
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.impute import MissingIndicator
from sklearn.impute import KNNImputer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neighbors import KNeighborsRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import LogisticRegression
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
from sklearn.datasets import load_wine
from sklearn.datasets import load_linnerud
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import RobustScaler
from sklearn.feature_selection import mutual_info_classif, mutual_info_regression
from sklearn.feature_selection import SelectKBest, SelectPercentile
from sklearn.metrics.pairwise import cosine_similarity, euclidean_distances, manhattan_dis
from sklearn.model_selection import cross_val_score
from IPython.display import Image
import matplotlib.pyplot as plt
import scipy.stats as stats
%matplotlib inline
sns.set(style="ticks")
```

In [37]:
```
data = pd.read_csv('titles.csv')
data.head()
```

Out[37]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Docu
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	In TV D
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	In TV
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	D
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	In Rc Sh

In [31]:
```
data = data[data['description'] == 'english']
```

In [32]:
```
data['description'].unique()
```

Out[32]:
```
array([], dtype=object)
```

In [33]:
```
data.keys()
```

Out[33]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
```

```
      dtype='object')
```

In [38]:
```
sentence = data.iloc[0]['description']
```

In [35]:
```
sentence
```

Out[35]:
```
'As her father nears the end of his life, filmmaker Kirsten Johnson stages his death in in
ventive and comical ways to help them both face the inevitable.'
```

Токенизация

In [39]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/solokhovir/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[39]:
```
True
```

In [41]:
```
from nltk import tokenize
nltk_tk_1 = nltk.WordPunctTokenizer()
nltk_tk_1.tokenize(sentence)
```

Out[41]:
```
['As',
 'her',
 'father',
 'nears',
 'the',
 'end',
 'of',
 'his',
 'life',
 ',',
 'filmmaker',
 'Kirsten',
 'Johnson',
 'stages',
 'his',
 'death',
 'in',
 'inventive',
 'and',
 'comical',
 'ways',
 'to',
 'help',
 'them',
 'both',
 'face',
 'the',
 'inevitable',
 '.']
```

In [42]:
```
# Токенизация по предложениям
nltk_tk_sents = nltk.tokenize.sent_tokenize(sentence)
print(len(nltk_tk_sents))
nltk_tk_sents
```

```
1
```

Out[42]:
```
['As her father nears the end of his life, filmmaker Kirsten Johnson stages his death in i
nventive and comical ways to help them both face the inevitable.']
```

Частеречная разметка

In [58]:
```
from spacy.lang.en import English
import spacy
nlp = spacy.load('en_core_web_sm')
```

In [60]:
```
spacy_text1 = nlp(sentence)
spacy_text1
```

Out[60]:
```
As her father nears the end of his life, filmmaker Kirsten Johnson stages his death in inv
entive and comical ways to help them both face the inevitable.
```

In [61]:
```
for t in spacy_text1:
    print(t)
```

```
As
her
father
nears
the
end
of
his
life
,
filmmaker
Kirsten
Johnson
stages
his
death
in
inventive
and
comical
ways
to
help
them
both
face
the
inevitable
.
```

Лемматизация

In [63]:
```
for token in spacy_text1:
      print(token, token.lemma, token.lemma_)
```

```
As 7437575085468336610 as
her 4115755726172261197 her
father 17071697760115891398 father
nears 4578410152339589874 near
```

```
the 7425985699627899538 the
end 18250316222013540736 end
of 886050111519832510 of
his 2661093235354845946 his
life 16477424308483498569 life
, 2593208677638477497 ,
filmmaker 4153866014530574051 filmmaker
Kirsten 5568109463636304900 Kirsten
Johnson 140201702615140384O6 Johnson
stages 87645220396502300 71 stage
his 2661093235354845946 his
death 178358667354806821 25 death
in 3002984154512732771 in
inventive 16220318996301312843 inventive
and 2283656566040971221 and
comical 127990007559324253 71 comical
ways 6878210874361030284 way
to 3791531372978436496 to
help 17461235395181654430 help
them 16875582379069451158 they
both 7111508780595485950 both
face 173953973342144755 56 face
the 7425985699627899538 the
inevitable 142088459945000197 98 inevitable
. 12646065887601541794 .
```

Выделение (распознавание) именованных сущностей

In [65]:
```
for ent in spacy_text1.ents:
    print(ent.text, ent.label_)
```

```
Kirsten Johnson PERSON
```

In [66]:
```
from spacy import displacy
displacy.render(spacy_text1, style='ent', jupyter=True)
```

As her father nears the end of his life, filmmaker Kirsten Johnson **PERSON** stages his death in inventive and comical ways to help them both face the inevitable.

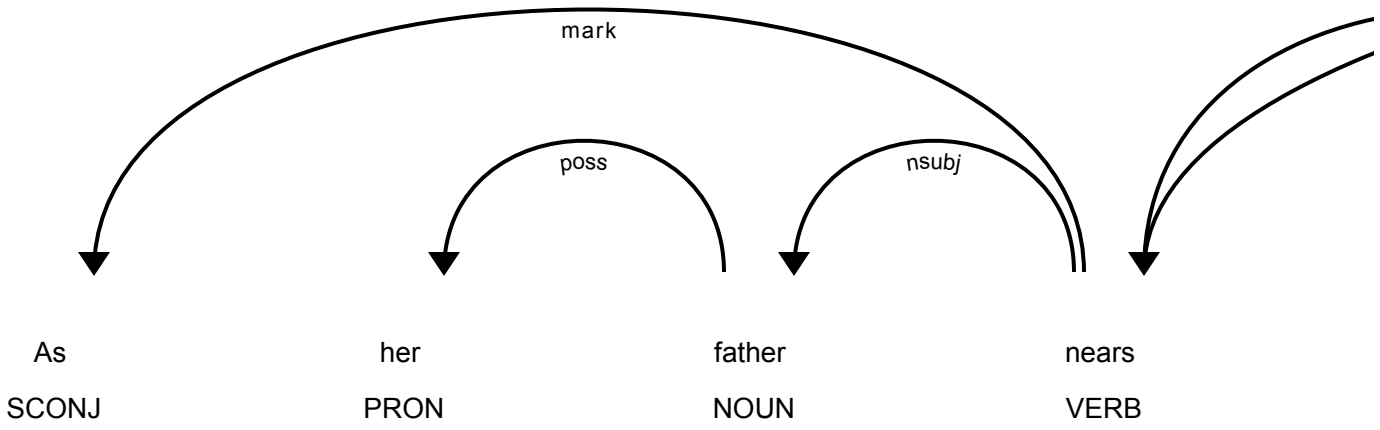
In [67]:
```
print(spacy.explain("PERSON"))
```

```
People, including fictional
```

Разбор предложения

In [68]:
```
from spacy import displacy
```

In [69]:
```
displacy.render(spacy_text1, style='dep', jupyter=True)
```



Классификация текста

TfidfVectorizer

In [70]:
```
data = data.dropna()
```

In [71]:
```
tfidfv = TfidfVectorizer(ngram_range=(1,3))
tfidf_ngram_features = tfidfv.fit_transform(data['description'])
tfidf_ngram_features
```

Out[71]:
```
<5332x187893 sparse matrix of type '<class 'numpy.float64'>'
        with 344111 stored elements in Compressed Sparse Row format>
```

In [72]:
```
y = data['title'].values
```

In [74]:
```
tfidfv = TfidfVectorizer(ngram_range=(1,3))
tfidf_ngram_features = tfidfv.fit_transform(y)
tfidf_ngram_features
```

Out[74]:
```
<5332x20321 sparse matrix of type '<class 'numpy.float64'>'
        with 31444 stored elements in Compressed Sparse Row format>
```

In [75]:
```
tfidf_ngram_features.todense()
```

Out[75]:
```
matrix([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
```

```
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]])
```

In [76]:
```
len(tfidf_ngram_features.todense()[0].getA1())len(tfidf_ngram_features.todense()[0].getA1
```

Out[76]:
```
20321
```

In [77]:
```
# Непустые значения нулевой строки
[i for i in tfidf_ngram_features.todense()[0].getA1() if i>0]
```

Out[77]:
```
[1.0]
```

fastText

In [79]:
```
!pip install fasttext
```

```
Collecting fasttext
  Downloading fasttext-0.9.2.tar.gz (68 kB)
     |████████████████████████████████| 68 kB 1.1 MB/s eta 0:00:01
Collecting pybind11>=2.2
  Using cached pybind11-2.9.2-py2.py3-none-any.whl (213 kB)
Requirement already satisfied: setuptools>=0.7.0 in /Users/solokhovir/opt/anaconda3/lib/py
thon3.9/site-packages (from fasttext) (58.0.4)
Requirement already satisfied: numpy in /Users/solokhovir/opt/anaconda3/lib/python3.9/site
-packages (from fasttext) (1.22.3)
Building wheels for collected packages: fasttext
  Building wheel for fasttext (setup.py) ... done
  Created wheel for fasttext: filename=fasttext-0.9.2-cp39-cp39-macosx_10_16_x86_64.whl si
ze=302720 sha256=012893e5656a362cf6343d18adc74e30e204401b88683ca377625c27f591a4ee
  Stored in directory: /Users/solokhovir/Library/Caches/pip/wheels/64/57/bc/1741406019061d
5664914b070bd3e71f6244648732bc96109e
Successfully built fasttext
Installing collected packages: pybind11, fasttext
Successfully installed fasttext-0.9.2 pybind11-2.9.2
```

In [82]:
```
import fasttext
```

In [95]:
```
model_path_2 = 'cc.en.300.bin'
```

In [96]:
```
ft = fasttext.load_model(model_path_2)
```

```
Warning : `load_model` does not return WordVectorModel or SupervisedModel any more, but a
`FastText` object which is very similar.
```

In [104…
```
matrix_ft = []
for text in data['description'].values:

    matrix_ft = np.array(matrix_ft)
```

In [105…
```
matrix_ft
```

```
array([], dtype=float64)
```

Out[105…

In [ ]: