

Яндекс Алгоритм 2018

ML track

Побывать в топе
пока не подтянулись те,
у кого модель училась две недели :)

🕒 1 апр 2018, 06:58:22

Объявления жюри

старт: 30 мрт 2018, 10:00:00

финиш: 23 апр 2018, 10:00:00

до финиша: 22д. 3ч.

• • •

[Положение участников](#) [Задачи](#) [Посылки](#) [Сообщения](#) [Участники](#)

Последний правильный ответ: 1, yabberd, 1д. 19ч.

Последнее отправленное решение: 1, yabberd, 1д. 194.

№	Участник	1	Очки
		109/109	
1	hyperslonic	85251.... 1д 12ч	85251.00
2	olegcheremisin	84961.... 1д 15ч	84961.00
3	achikin82	84814.... 1д 11ч	84814.00
4	Александр Свиридов	84194.... 1д 8ч	84194.00
5	fursov.i.a	84153.... 1д 13ч	84153.00
6	lo-r-d4	84053.... 1д 1ч	84053.00
7	ilya.belkin-trade	83671.... 1д 18ч	83671.00
8	manish kumar	82022.... 10:26	82022.00
9	000 Anatoly Tolstobrov	81933.... 09:29	81933.00
10	YegorYu	81810.... 1д 3ч	81810.00
11	Антѐм	81630....	81630.00

Ooops

```
In [1]: import pandas as pd  
import numpy as np
```

```
In [3]: train = pd.read_csv('datasets/yandex/train.tsv', sep='\t', header=None)
```

```
1746     def read(self, nrows=None):  
1747         try:  
-> 1748             data = self._reader.read(nrows)  
1749         except StopIteration:  
1750             if self._first_chunk:  
  
pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader.read (pandas/_libs/parsers.c:10862)()  
pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._read_low_memory (pandas/_libs/parsers.c:11138)()  
pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._read_rows (pandas/_libs/parsers.c:11884)()  
pandas/_libs/parsers.pyx in pandas._libs.parsers.TextReader._tokenize_rows (pandas/_libs/parsers.c:11755)()  
pandas/_libs/parsers.pyx in pandas._libs.parsers.raise_parser_error (pandas/_libs/parsers.c:28765)()  
  
ParserError: Error tokenizing data. C error: Expected 8 fields in line 134, saw 10
```

Данные

```
In [12]: train.head(7)
```

```
Out[12]:
```

	0	1	2	3	4	5	6	7
0	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	0	не могу .	good	0.875352
1	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	1	нет , звонить буду я .	neutral	0.900968
2	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	2	слушай , я не мог уйти .	bad	0.884320
3	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	3	я не прекращу звонить .	good	0.982530
4	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	4	я звоню им .	good	0.838054
5	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	5	просто повесь трубку .	bad	0.955718
6	50117448291	бывало и получше .	слушайте , мы с женой . . . совсем заблудились .	я пытаюсь добраться до юджина , но , кажется ,...	0	едете ?	bad	0.909115

```
In [13]: test.head(7)
```

```
Out[13]:
```

	0	1	2	3	4	5
0	138920940977	знаешь , я иногда подумываю , что тебе надо пр...	не - а .	нет ?	0	неа .
1	138920940977	знаешь , я иногда подумываю , что тебе надо пр...	не - а .	нет ?	1	нет , не хочу .
2	138920940977	знаешь , я иногда подумываю , что тебе надо пр...	не - а .	нет ?	2	нет .
3	138920940977	знаешь , я иногда подумываю , что тебе надо				

Метрика

Задача участников – вернуть ранжирование реплик-кандидатов представленных в порядке убывания скоров, выданных моделями участников. Метрика для оценивания этих ранжирований – NDCG.

$$DCG_p = \sum_{i=1}^p \frac{rel_i}{\log_2(i + 1)} \quad nDCG_p = \frac{DCG_p}{IDCG_p}$$

rel_i принимает три возможных значения - 2, 1 и 0 - для меток **good**, **neutral** и **bad** соответственно.

Особо отмечаем, что информация об уверенности меток (доступная только для тренировочных данных) никак не учитывается в метрике.

Скор участников отображаемый в контесте - это среднее NDCG для всех `context_id` тестовых данных, умноженное на 100 000.

```
In [142]: # расставляем значения по формуле хорошесть * confidence
# Good (2)
# Neutral (1)
# Bad (0)
```

```
In [143]: def rank2num(st):
#         if st == 'good':
#             return 2
#         else:
#             if st == 'neutral':
#                 return 1
#             else:
#                 return 0
```

```
In [144]: train['rank'] = train[6].apply(rank2num)
train['target'] = train['rank'] * train[7]
train.head()
```

Out[144]:

	0	1	2	3	4	5	6	7	rank	target
0	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	0	не могу .	good	0.875352	2	1.750703
1	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	1	нет , звонить буду я .	neutral	0.900968	1	0.900968
2	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	2	слушай , я не мог уйти .	bad	0.884320	0	0.000000
3	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	3	я не прекращу звонить .	good	0.982530	2	1.965061
4	22579918886	кликни на меня а потом на надпись " видео - зв...	о , я тебя вижу .	ладно , повесь трубку .	4	я звоню им .	good	0.838054	2	1.676107

Quick'n'dirty валидация

```
In [18]: X_train.shape, X_test.shape
Out[18]: ((97533, 505874), (9968, 505874))

In [19]: y_train = train['target']

In [20]: train_part_size = int(0.75 * train['target'].shape[0])
X_train_part = X_train[:train_part_size, :]
y_train_part = y_train[:train_part_size]
X_valid = X_train[train_part_size:, :]
y_valid = y_train[train_part_size:]

In [21]: X_train_part.shape
Out[21]: (73149, 505874)

In [22]: from sklearn.linear_model import LinearRegression, Ridge, Lasso
reg = Ridge()
reg.fit(X_train_part, y_train_part)

Out[22]: Ridge(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=None,
normalize=False, random_state=None, solver='auto', tol=0.001)

In [23]: reg_pred = reg.predict(X_valid)

In [24]: %%time
from sklearn.metrics import mean_absolute_error

valid_mae = mean_absolute_error(y_valid, reg_pred)
print(valid_mae)

0.72641238803
CPU times: user 214 µs, sys: 3.92 ms, total: 4.13 ms
Wall time: 2.85 ms
```

Submission

```
In [202]: sub['context_id'] = test[0]
sub['reply_id'] = test[4]
sub['rank'] = - y_test
```

```
In [203]: sub.head()
```

```
Out[203]:
```

	context_id	reply_id	rank
0	138920940977	0	-0.659569
1	138920940977	1	-0.922145
2	138920940977	2	-0.356104
3	138920940977	3	-0.730093
4	138920940977	4	-0.852768

```
In [205]: submission = sub.sort_values(by=['context_id', 'rank'])
```

```
In [206]: del submission['rank']
```

```
In [207]: submission.head()
```

```
Out[207]:
```

	context_id	reply_id
1	138920940977	1
4	138920940977	4
3	138920940977	3
0	138920940977	0
5	138920940977	5

```
In [208]: test.shape, sub.shape
```

```
Out[208]: ((9968, 6), (9968, 3))
```

```
In [209]: submission.to_csv('yandex-ml-ridge.tsv', header=None, index=False, sep=' ')
```


Дальнейшие планы

- Реализовать метрику из постановки задачи для валидации
- Попробовать стемминг или работу с морфологией (к сожалению, `rumystem3` оказался слишком медленным)
- Попробовать другое ранжирование (делать классификации Good / Normal / Bad)
- Попробовать что-нибудь новое модное ранее не пробованное из упомянутого на следующем слайде

Материалы по теме

- **Запись трансляции тренировки ML 31.03.2018** | Kaggle Mercari, Recruit, Whatever Hack - в начале обсуждается конкурс, Алиса и подходы к чатикам на темы “ни о чем” <https://youtu.be/N2WMVUKUt8I>
- **Sberbank Data Science Contest:** определение релевантности вопроса параграфу https://youtu.be/eN_jxfofBOk
- **Sberbank Data Science Journey: базовые подходы и идеи решения** <https://youtu.be/c7EdABKz2SE>
- **DrQA**, a PyTorch implementation of the DrQA system described in the ACL 2017 paper [Reading Wikipedia to Answer Open-Domain Questions](#)
<https://github.com/facebookresearch/DrQA>
- **Another Twitter sentiment analysis with Python – Part 6 (Doc2Vec)**
<https://towardsdatascience.com/another-twitter-sentiment-analysis-with-python-part-6-doc2vec-603f11832504>

Предложенное на семинаре

- SyntaxNet
- SVD/LSA
- Сбалансировать пропорции good/normal/bad для каждого контекста (например, если мало bad - добавить строку из случайного места)
- Добавить CountVectorizer к TFIDF