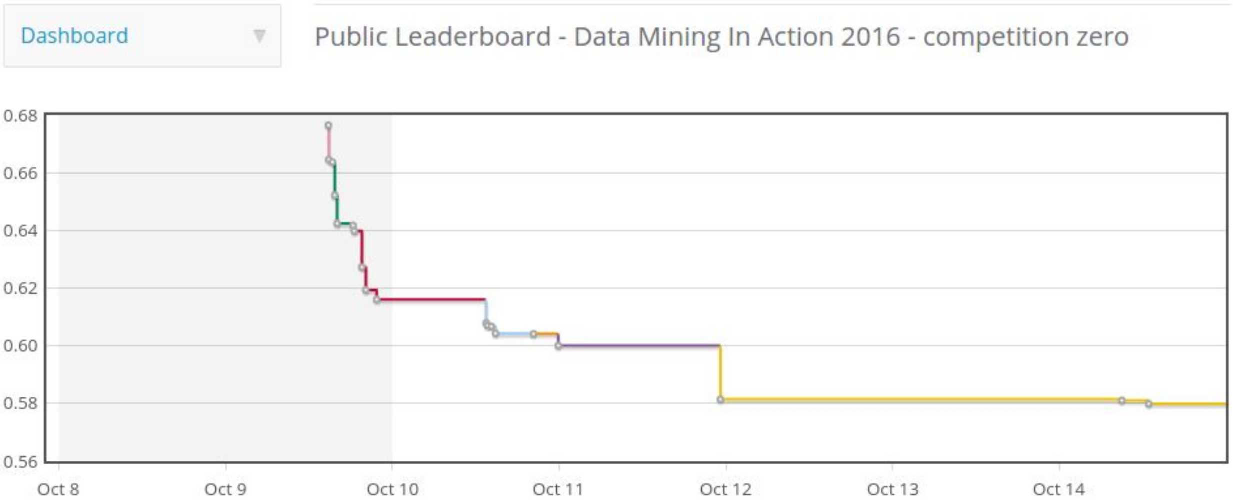
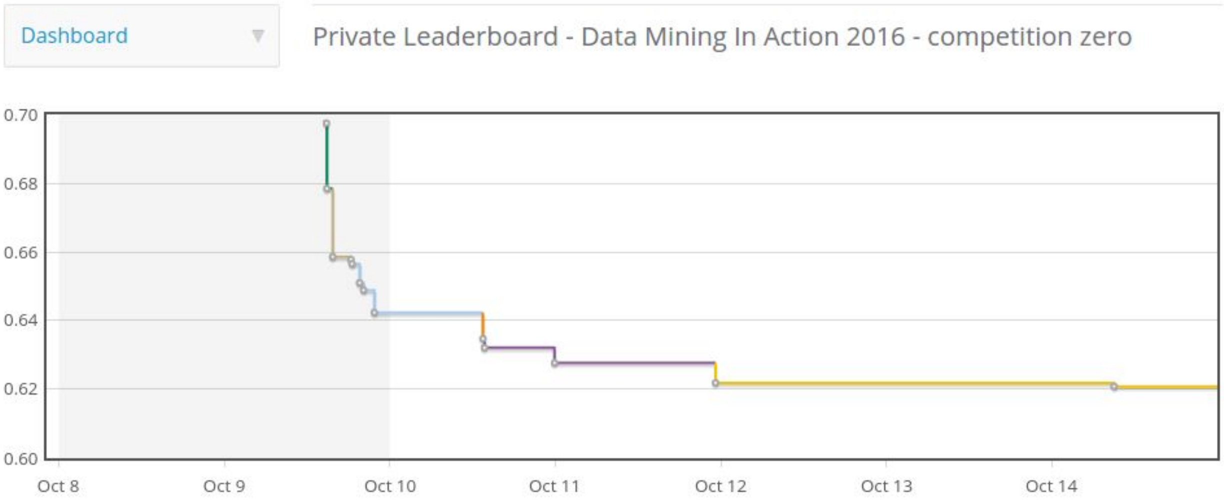
Competition zero

Гущин Александр,
15.10.2016



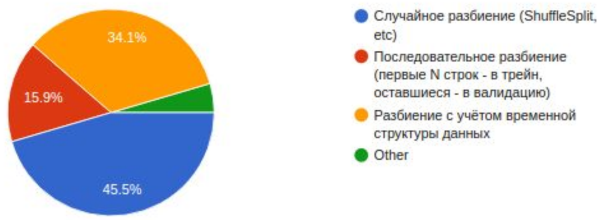
Public Leaderboard - Data Mining In Action 2016 - competition zero

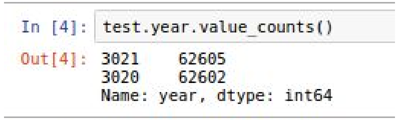


Private Leaderboard - Data Mining In Action 2016 - competition zero

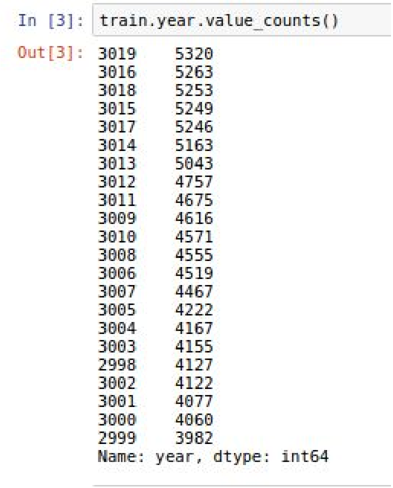
Cross-validation

Каким образом была устроена кросс-валидация (44 responses)

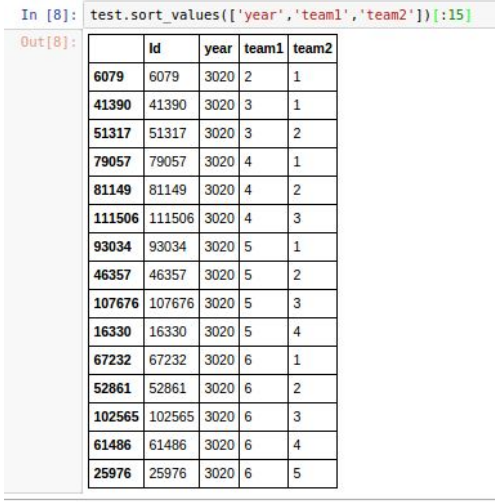




```
In [4]: test.year.value_counts()

Out[4]: 3021    62605
        3020    62602
        Name: year, dtype: int64
```



```
In [3]: train.year.value_counts()

Out[3]: 3019    5320
        3016    5263
        3018    5253
        3015    5249
        3017    5246
        3014    5163
        3013    5043
        3012    4757
        3011    4675
        3009    4616
        3010    4571
        3008    4555
        3006    4519
        3007    4467
        3005    4222
        3004    4167
        3003    4155
        2998    4127
        3002    4122
        3001    4077
        3000    4060
        2999    3982
        Name: year, dtype: int64
```



```
In [8]: test.sort_values(['year','team1','team2'])[:15]

Out[8]:
```

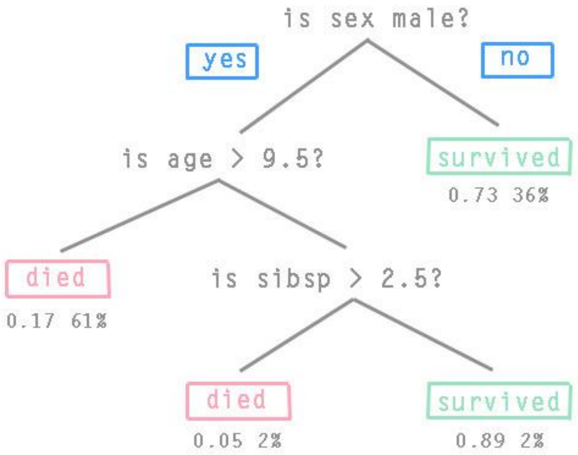
	Id	year	team1	team2
6079	6079	3020	2	1
41390	41390	3020	3	1
51317	51317	3020	3	2
79057	79057	3020	4	1
81149	81149	3020	4	2
111506	111506	3020	4	3
93034	93034	3020	5	1
46357	46357	3020	5	2
107676	107676	3020	5	3
16330	16330	3020	5	4
67232	67232	3020	6	1
52861	52861	3020	6	2
102565	102565	3020	6	3
61486	61486	3020	6	4
25976	25976	3020	6	5

Boosting

Features = [‘year’, ‘team1’, ‘team2’]

param['booster'] = 'gbtree'

param['max_depth'] = 8



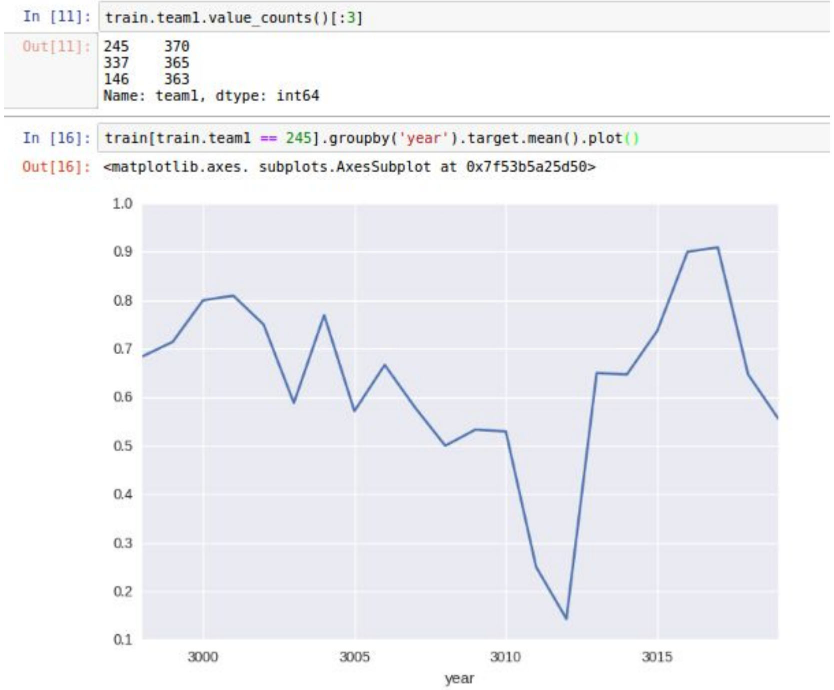
Linear model

Features = [‘year’, ‘team1_1’, …,

’team1_356’, ‘team2_1’, … , ‘team2_356’]

Ypred = np.sum(a * ‘year’

+ b * ‘team1_1’ + …)



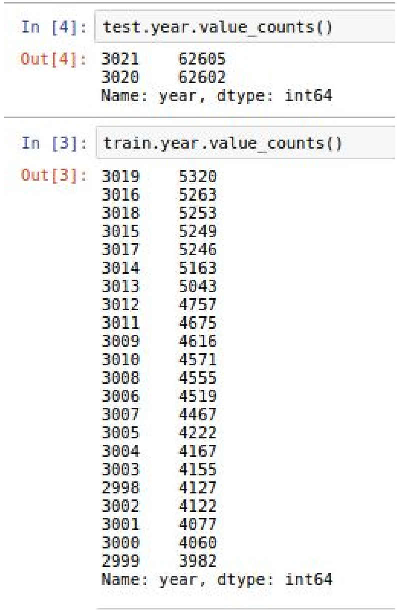
```
In [11]: train.team1.value_counts()[:3]

Out[11]: 245    370
         337    365
         146    363
         Name: team1, dtype: int64

In [16]: train[train.team1 == 245].groupby('year').target.mean().plot()

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x7f53b5a25d50>
```

Statistic features

1) Calculate for train => add to train and test

2) Year X: (X-1) + (X-2) + (X-3) + …

3) Year X: (X-1) + (X-2) * 0.9 + (X-3) * 0.9 ** 2 + ...



```
In [4]: test.year.value_counts()
Out[4]: 3021    62605
        3020    62602
        Name: year, dtype: int64

In [3]: train.year.value_counts()
Out[3]: 3019    5320
        3016    5263
        3018    5253
        3015    5249
        3017    5246
        3014    5163
        3013    5043
        3012    4757
        3011    4675
        3009    4616
        3010    4571
        3008    4555
        3006    4519
        3007    4467
        3005    4222
        3004    4167
        3003    4155
        2998    4127
        3002    4122
        3001    4077
        3000    4060
        2999    3982
        Name: year, dtype: int64
```

Data augmentation

train.target.mean() == 0.50096940231672393

Data:

Team1, team2, score1, score2, target

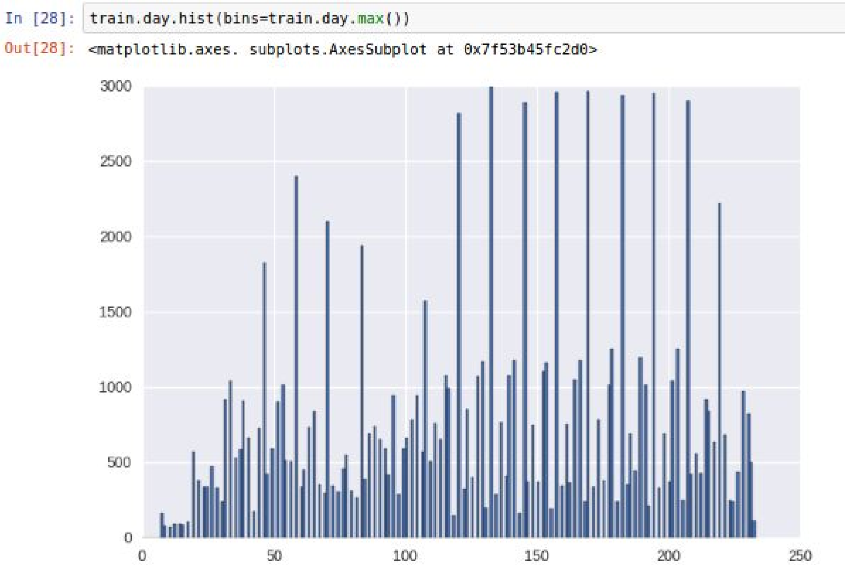
Additional data:

Team2, team1, score2, score1, 1 - target

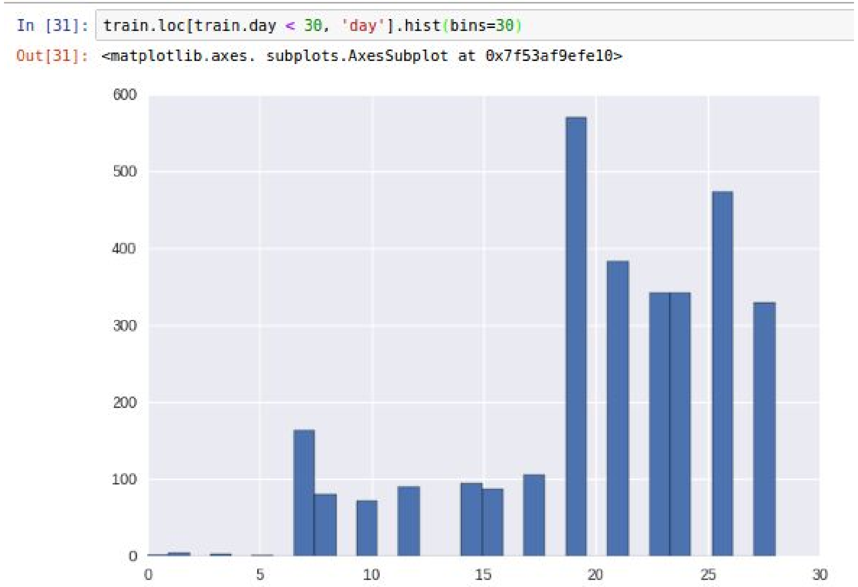
Pred = [f(team1, team2) + f(team2, team1)] / 2

Day

1.77



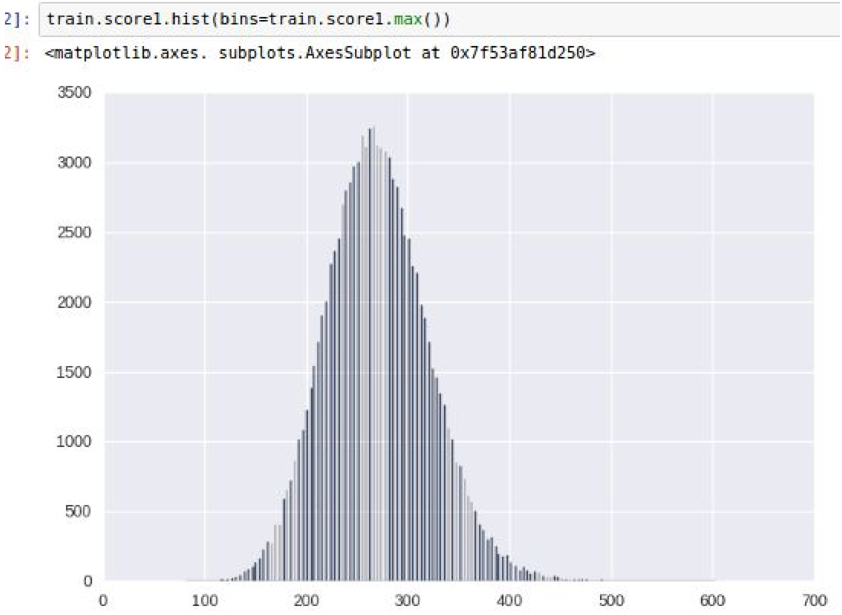
```
In [28]: train.day.hist(bins=train.day.max())
Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x7f53b45fc2d0>
```



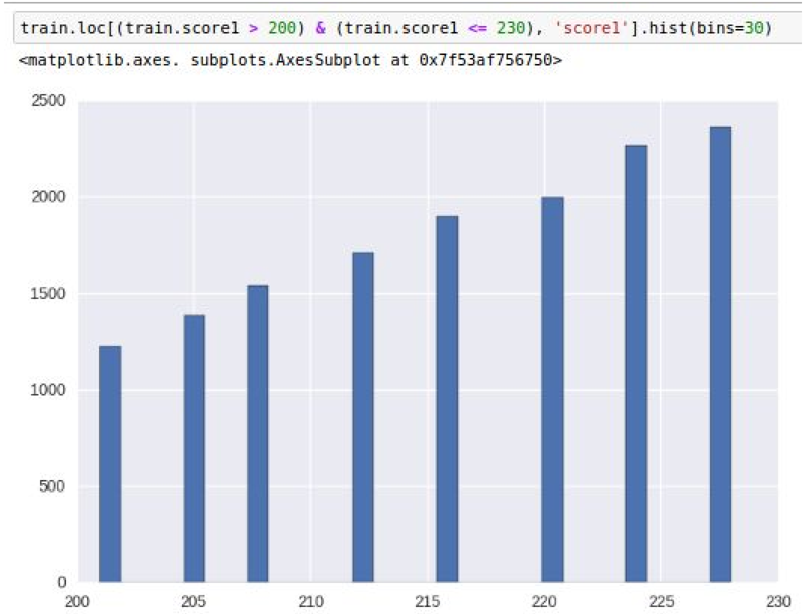
```
In [31]: train.loc[train.day < 30, 'day'].hist(bins=30)
Out[31]: <matplotlib.axes._subplots.AxesSubplot at 0x7f53af9efe10>
```

Score

3.87



```
2]: train.score1.hist(bins=train.score1.max())
2]: <matplotlib.axes. subplots.AxesSubplot at 0x7f53af81d250>
```



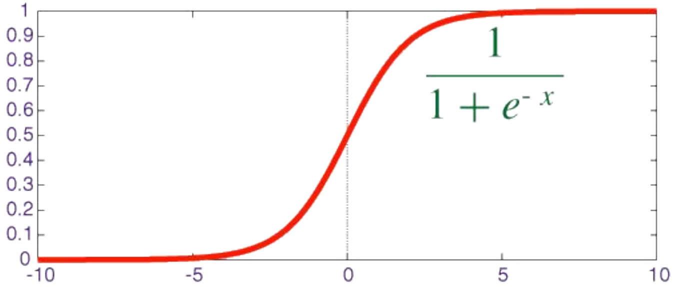
```
train.loc[(train.score1 > 200) & (train.score1 <= 230), 'score1'].hist(bins=30)
<matplotlib.axes. subplots.AxesSubplot at 0x7f53af756750>
```

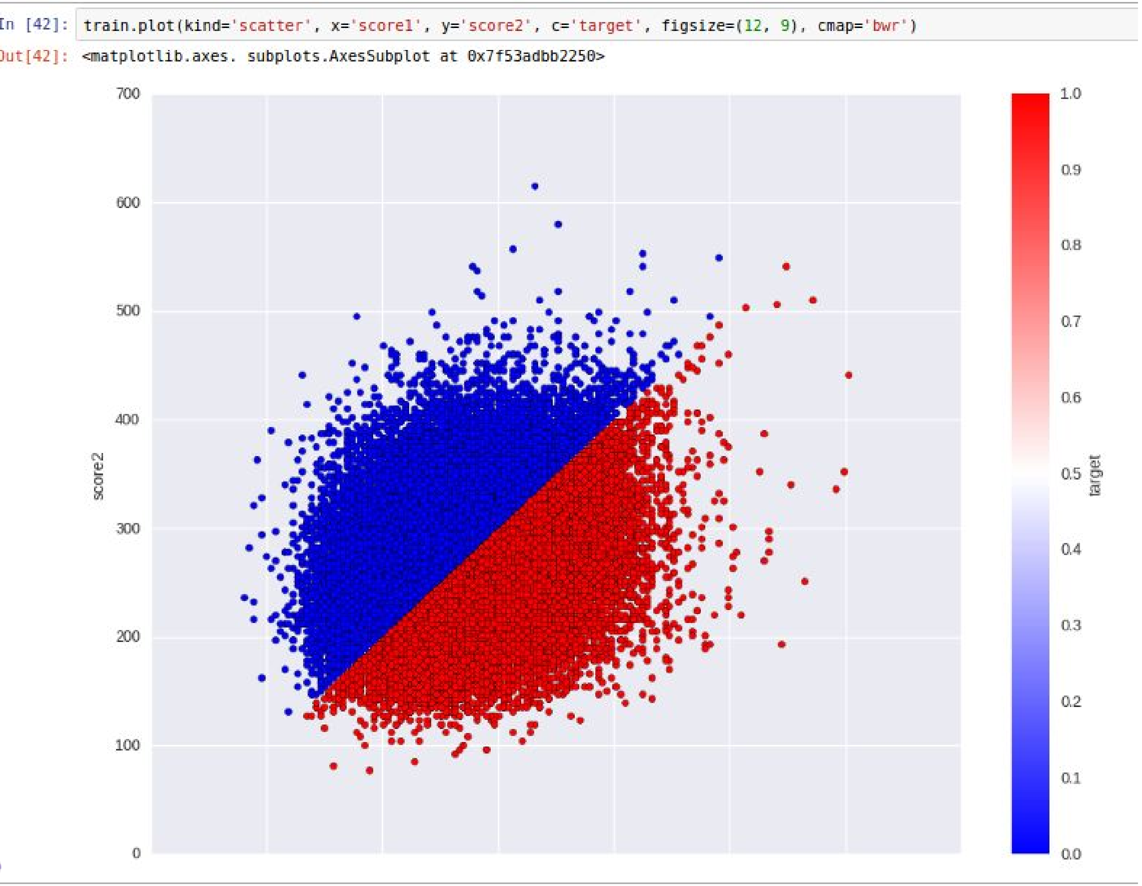
Score

sigmoid(ypred / 30)

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right)$$

np.clip(x, min, max)





```
In [42]: train.plot(kind='scatter', x='score1', y='score2', c='target', figsize=(12, 9), cmap='bwr')
Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x7f53adbb2250>
```

Transitivity

A = np.zeros((n_teams, n_teams))

A[**i**, **j**] - team_**i** wins team_**j** (count, probability, etc)

B = A + tau * A ** 2 + tau ** 2 * A ** 3 (+ …)

i wins **j**, if (**i** wins **k**) and (**k** wins **j**)

B[**i**, **j**] - new feature