CIKM Cup 2016 Track 1: Cross-Device Entity Linking Challenge

Иван Бендына

https://competitions.codalab.org/competitions

CIKM Cup 2016 Track 1: Cross-Device Entity Linking Challenge

Organized by spirinus - Current server time: Oct. 14, 2016, 1:04 p.m. UTC

Previous	▶ Current	End
Phase 1: Validation Leaderboard	Phase 2: Testing Leaderboard	Competition Ends
Aug. 5, 2016, midnight UTC	Oct. 2, 2016, midnight UTC	Oct. 5, 2020, midnight UTC

Learn the Details | Phases | Participate | Results | Forums

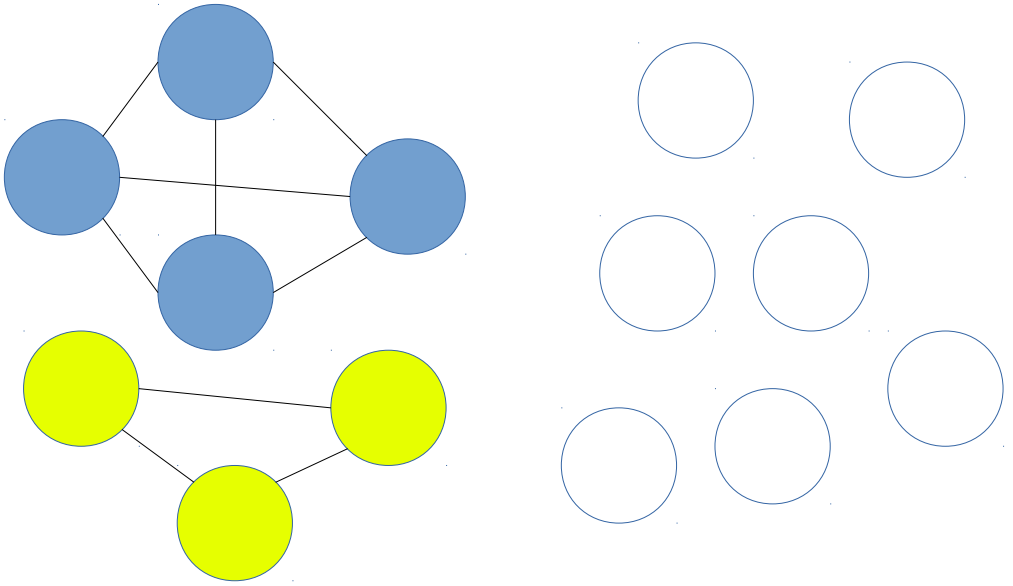
Phase 1: Validation Leaderboard | Phase 2: Testing Leaderboard

Phase description

Ongoing model development and evaluation with the results on the public leaderboard.

Max submissions per day: 10000

Max submissions total: 10000



Train Test

- Вершины графа – это девайсы.
- Кластеры – это пользователи.
- Нужно объединить вершины в кластеры в тестовой выборке (то есть предсказать, что у двух или более девайсов один владелец).

У каждого девайса известен список событий браузера:

{

Url: ed95a9a5be30e4c8/5162fc6a223f248d/31a42ef13edf7d8a/
c2ee3aa455c8b288/656e9a1b3fa15996

Title: b687bc71fc33713b 19475eb9e8506115 1c202a1bece8798
0e415cc3f3ea206 f27b5d716eb58ac 37a20a398fae482a

Timestamp: 1464769462076

},

...

В урлах хэшируются слова между символами / и ?.
В тайтлах хэшируются слова между пробелами.

Dataset Statistics:

The number of unique tokens in titles (dictionary size): 8,485,859
The number of unique tokens in URLs (dictionary size): 27,398,114
The average number of events/facts per userID: 197
The median number of events/facts per userID: 106
The number of unique domain names: 282,613
The number of all events for all users combined: 66,808,490
The number of unique userIDs: 339,405
The number of unique websites (domina + URL path): 14,148,535
The number of users in the train set: 240,732
The number of users in the test set (public and private leaderboard combined): 98,255
Known matching pairs for training: 506,136
Known matching pairs for testing (public and private leaderboard combined): 215,307

Dataset Statistics:

The number of unique tokens in titles (dictionary size): 8,485,859
The number of unique tokens in URLs (dictionary size): 27,398,114
The average number of events/facts per userID: 197
The median number of events/facts per userID: 106
The number of unique domain names: 282,613
The number of all events for all users combined: 66,808,490
The number of unique userIDs: 339,405
The number of unique websites (domain + URL path): 14,148,535
The number of users in the train set: 240,732
The number of users in the test set (public and private leaderboard combined): 98,255
Known matching pairs for training: 506,136
Known matching pairs for testing (public and private leaderboard combined): 215,307

Dataset Statistics:

The number of unique tokens in titles (dictionary size): 8,485,859
The number of unique tokens in URLs (dictionary size): 27,398,114
The average number of events/facts per userID: 197
The median number of events/facts per userID: 106
The number of unique domain names: 282,613
The number of all events for all users combined: 66,808,490
The number of unique userIDs: 339,405
The number of unique websites (domain + URL path): 14,148,535
The number of users in the train set: 240,732
The number of users in the test set (public and private leaderboard combined): 98,255
Known matching pairs for training: 506,136
Known matching pairs for testing (public and private leaderboard combined): 215,307

Dataset Statistics:

The number of unique tokens in titles (dictionary size): 8,485,859
The number of unique tokens in URLs (dictionary size): 27,398,114
The average number of events/facts per userID: 197
The median number of events/facts per userID: 106
The number of unique domain names: 282,613
The number of all events for all users combined: 66,808,490
The number of unique userIDs: 339,405
The number of unique websites (domain + URL path): 14,148,535
The number of users in the train set: 240,732
The number of users in the test set (public and private leaderboard combined): 98,255
Known matching pairs for training: 506,136
Known matching pairs for testing (public and private leaderboard combined): 215,307

Метрика

$$F1 = \frac{2pr}{p + r}$$

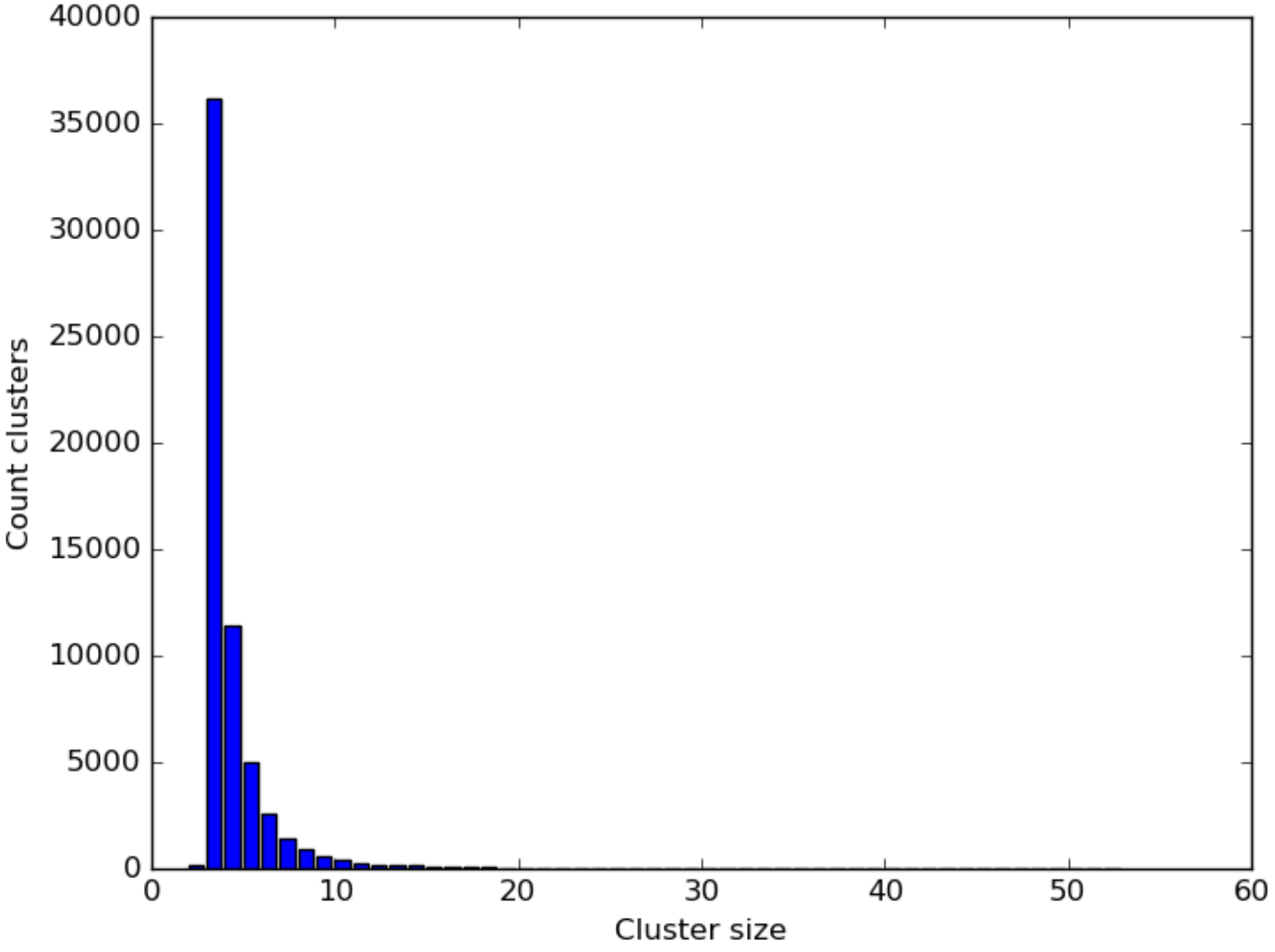
Но проверка решения производится только на половине правильных ребер, поэтому точность уменьшается в два раза.

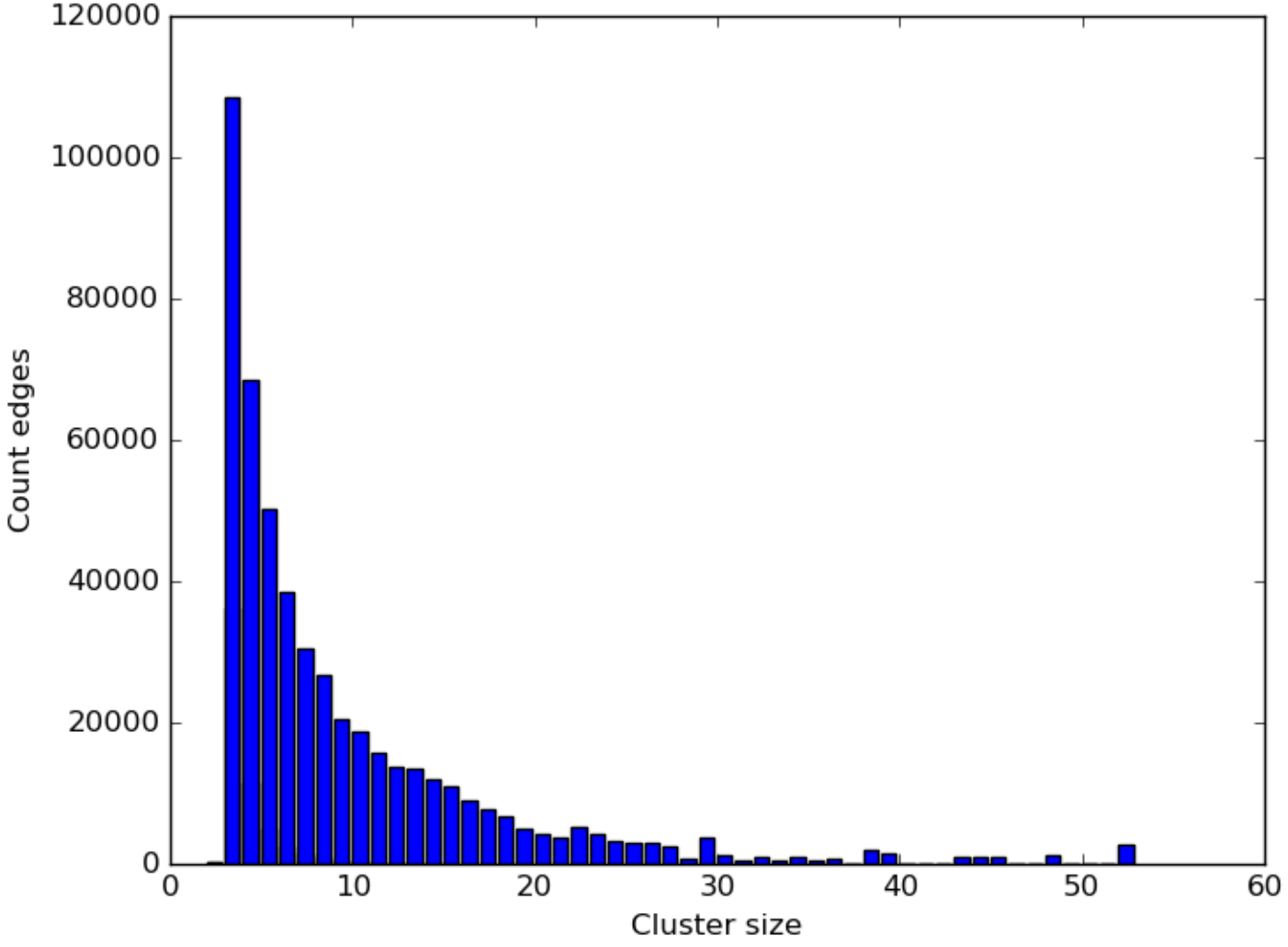
$$F1 = \frac{pr}{0.5p + r}$$

Базовое решение

- Разбить урлы на слова
- Составить матрицу девайсы-слова (TF-IDF)
- Для каждого девайса, найти 15 ближайших соседей
- Отсортировать полученные пары по расстоянию
- Взять топ-215307 пар

Результат на LB: **0.056**


40000
35000
30000
25000
20000
15000
10000
5000
0
Count clusters
0
10
20
30
40
50
60
Cluster size


120000
100000
80000
60000
40000
20000
0
Count edges
0
10
20
30
40
50
60
Cluster size

Валидационная выборка

- Делим трейн на кластеры.
- Для валидационной выборки выбираем несколько случайных кластеров так, чтобы количество ребер в тесте и валидации было примерно одинаково

План

1)Собрать небольшую (<20.000.000) выборку ребер с наибольшей полнотой для трейна, валидации, теста.
2)Сгенерировать признаки для каждого ребра (построить матрицу объекты-признаки).
3)Обучиться по трейн выборке и предсказать вероятности для валидационной и тестовой выборки.
4)Взять топ N наиболее вероятных ребер, дополнить до полных подграфов (N и параметры дополнения подобрать на валидационной выборке)

План

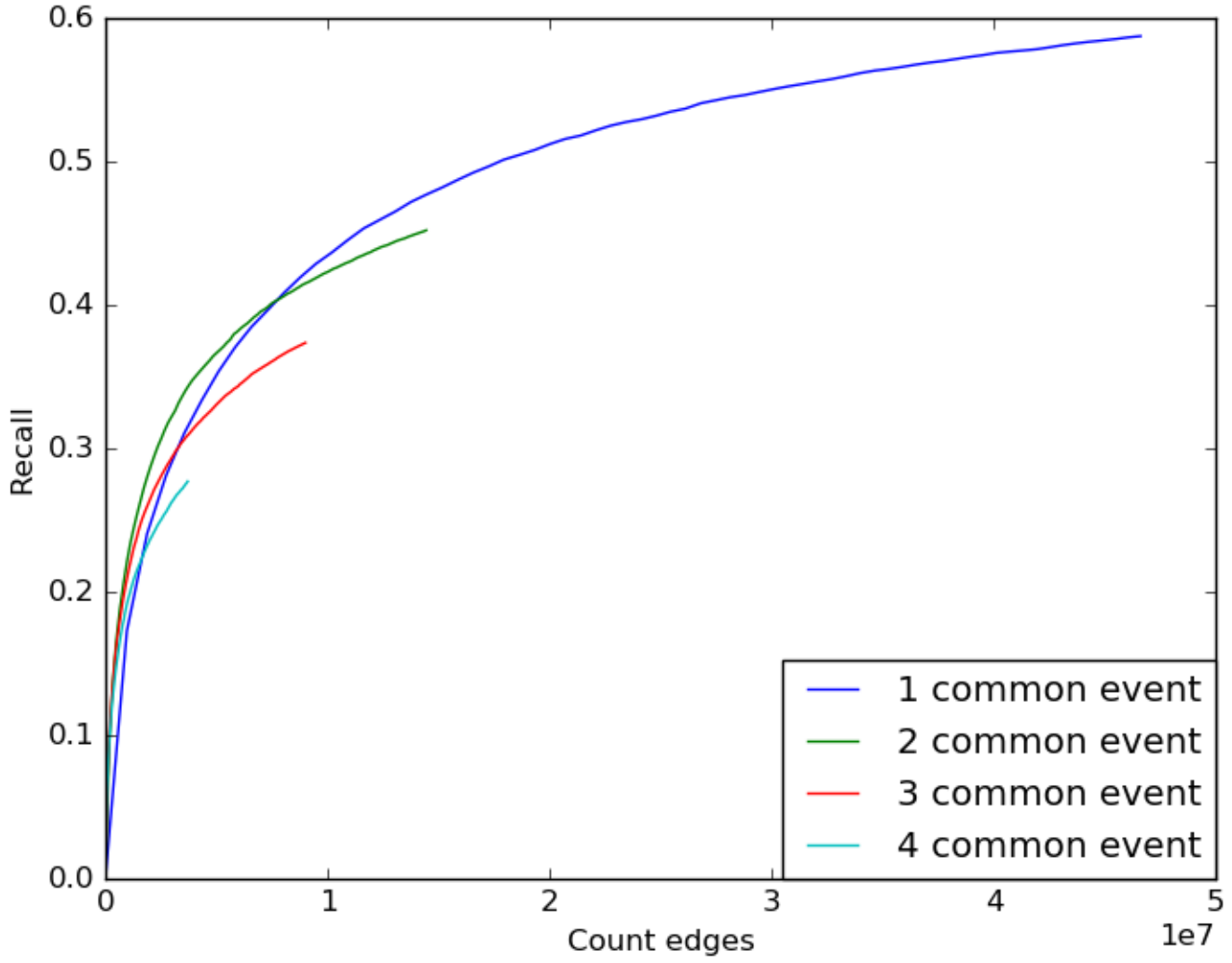
1)Собрать небольшую (<20.000.000) выборку ребер с наибольшей полнотой для трейна, валидации, теста.
2)Сгенерировать признаки для каждого ребра (построить матрицу объекты-признаки).
3)Обучиться по трейн выборке и предсказать вероятности для валидационной и тестовой выборки.
4)Взять топ N наиболее вероятных ребер, дополнить до полных подграфов (N и параметры дополнения подобрать на валидационной выборке)

Девайсы с общими событиями:

Найдем все такие пары девайсов, у которых есть как минимум K общих событий, причем каждое из них есть не более, чем у M девайсов.

Девайсы с общими событиями:

M	1 общее событие	2 общих события	3 общих события	4 общих события
2	0.1727 (0.9m)	0.0878 (0.15m)	0.0537 (0.07m)	0.0368 (0.04m)
3	0.2405 (1.8m)	0.1367 (0.32m)	0.0904 (0.14m)	0.0647 (0.08m)
4	0.2811 (2.7m)	0.1665 (0.49m)	0.1126 (0.22m)	0.0819 (0.13m)
5	0.3105 (3.5m)	0.1884 (0.65m)	0.1291 (0.29m)	0.0947 (0.17m)
7	0.3536 (5m)	0.2210 (0.9m)	0.1538 (0.43m)	0.1138 (0.26m)
10	0.3964 (7.3m)	0.2543 (1.4m)	0.1795 (0.64m)	0.1342 (0.38m)
20	0.4769 (14m)	0.3221 (2.9m)	0.2341 (1.3m)	0.1785 (0.79m)
30	0.5181 (21m)	0.3587 (4.5m)	0.2658 (2m)	0.2053 (1.2m)
50	0.5634 (34m)	0.4035 (7.7m)	0.3042 (3.4m)	0.2375 (2m)
70	0.5873 (46m)	0.4260 (10m)	0.3249 (4.6m)	0.2561 (2.7m)
100		0.4520 (14m)	0.3487 (6.3m)	0.2769 (3.7m)
150			0.3763 (9m)	


0.6
0.5
0.4
0.3
0.2
0.1
0.0
Recall
0
1
2
3
4
5
1e7
Count edges
1 common event
2 common event
3 common event
4 common event

Девайсы с общими событиями:

M	1 общее событие	2 общих события	3 общих события	4 общих события
2	0.1727 (0.9m)	0.0878 (0.15m)	0.0537 (0.07m)	0.0368 (0.04m)
3	0.2405 (1.8m)	0.1367 (0.32m)	0.0904 (0.14m)	0.0647 (0.08m)
4	0.2811 (2.7m)	0.1665 (0.49m)	0.1126 (0.22m)	0.0819 (0.13m)
5	0.3105 (3.5m)	0.1884 (0.65m)	0.1291 (0.29m)	0.0947 (0.17m)
7	0.3536 (5m)	**0.2210 (0.9m)**	0.1538 (0.43m)	0.1138 (0.26m)
10	0.3964 (7.3m)	0.2543 (1.4m)	0.1795 (0.64m)	0.1342 (0.38m)

- Выделить список событий, которые есть не более, чем у 7ми девайсов
- Найти пары девайсов, у которых есть минимум 2 общих события из списка
- Результат на LB: **0.2092**

Девайсы с общими событиями:

M	1 общее событие	2 общих события	3 общих события	4 общих события
2	0.1727 (0.9m)	0.0878 (0.15m)	0.0537 (0.07m)	0.0368 (0.04m)
3	0.2405 (1.8m)	0.1367 (0.32m)	0.0904 (0.14m)	0.0647 (0.08m)
4	0.2811 (2.7m)	0.1665 (0.49m)	0.1126 (0.22m)	0.0819 (0.13m)
5	0.3105 (3.5m)	0.1884 (0.65m)	0.1291 (0.29m)	0.0947 (0.17m)
7	0.3536 (5m)	**0.2210 (0.9m)**	0.1538 (0.43m)	0.1138 (0.26m)
10	0.3964 (7.3m)	0.2543 (1.4m)	0.1795 (0.64m)	0.1342 (0.38m)

- Выделить список событий, которые есть не более, чем у 7ми девайсов
- Найти пары девайсов, у которых есть минимум 2 общих события из списка
- Удалить девайсы, у которых более 15ти ребер (вместе с ребрами)
- Результат на LB: **0.2206**

Выборка для обучения:

M	1 общее событие	2 общих события	3 общих события	4 общих события
20	0.4769 (14m)	0.3221 (2.9m)	0.2341 (1.3m)	0.1785 (0.79m)
30	**0.5181 (21m)**	0.3587 (4.5m)	0.2658 (2m)	0.2053 (1.2m)
50	0.5634 (34m)	**0.4035 (7.7m)**	**0.3042 (3.4m)**	0.2375 (2m)
70	0.5873 (46m)	0.4260 (10m)	0.3249 (4.6m)	0.2561 (2.7m)

Объединяем выделенные три выборки + все ребра из базового KNN (для 30 соседей)

Полнота на трейне **0.62**, на тесте **0.59**

План

1)Собрать небольшую (<20.000.000) выборку ребер с наибольшей полнотой для трейна, валидации, теста.

2)Сгенерировать признаки для каждого ребра (построить матрицу объекты-признаки).

3)Обучиться по трейн выборке и предсказать вероятности для валидационной и тестовой выборки.

4)Взять топ N наиболее вероятных ребер, дополнить до полных подграфов (N и параметры дополнения подобрать на валидационной выборке)

Признаки для каждого ребра (55 признаков)

- Расстояние в KNN.
- Номер соседа в KNN.
- Номер соседа с другой стороны в KNN.

- Минимальное M из таблицы, при котором ребро появляется (для каждого столбца).

- Количество общих урлов (уникальных, с учетов повторений), количество общих урлов, которые есть только у 2, 3, 4, 5, 6-10 девайсов.
- То же для доменов, слов урлов, домен + первое слово.
- То же для тайтлов, слов тайтлов, 2грамм тайтлов.

Признаки для каждого ребра (55 признаков)

- Расстояние в KNN.
- Номер соседа в KNN.
- Номер соседа с другой стороны в KNN.

- Минимальное M из таблицы, при котором ребро появляется (для каждого столбца).

- Количество общих урлов (уникальных, с учетов повторений), количество общих урлов, которые есть только у 2, 3, 4, 5, 6-10 девайсов.
- То же для доменов, слов урлов, домен + первое слово.
- То же для тайтлов, слов тайтлов, 2грамм тайтлов.

Признаки для каждого ребра (55 признаков)

- Расстояние в KNN.
- Номер соседа в KNN.
- Номер соседа с другой стороны в KNN.

- Минимальное M из таблицы, при котором ребро появляется (для каждого столбца).

- Количество общих урлов (уникальных, с учетов повторений), количество общих урлов, которые есть только у 2, 3, 4, 5, 6-10 девайсов.
- То же для доменов, слов урлов, домен + первое слово.
- То же для тайтлов, слов тайтлов, 2грамм тайтлов.

Признаки для каждого ребра (55 признаков)

- Расстояние в KNN.
- Номер соседа в KNN.
- Номер соседа с другой стороны в KNN.

- Минимальное M из таблицы, при котором ребро появляется (для каждого столбца).

- Количество общих урлов (уникальных, с учетов повторений), количество общих урлов, которые есть только у 2, 3, 4, 5, 6-10 девайсов.
- То же для доменов, слов урлов, домен + первое слово.
- То же для тайтлов, слов тайтлов, 2грамм тайтлов.

План

1)Собрать небольшую (<20.000.000) выборку ребер с наибольшей полнотой для трейна, валидации, теста.
2)Сгенерировать признаки для каждого ребра (построить матрицу объекты-признаки).
3)Обучиться по трейн выборке и предсказать вероятности для валидационной и тестовой выборки.
4)Взять топ N наиболее вероятных ребер, дополнить до полных подграфов (N и параметры дополнения подобрать на валидационной выборке)

Обучение модели

Random Forest, с подбиранием лучшего max_depth по валидации
ROC AUC = 0.957
Результат на LB: **0.3584**

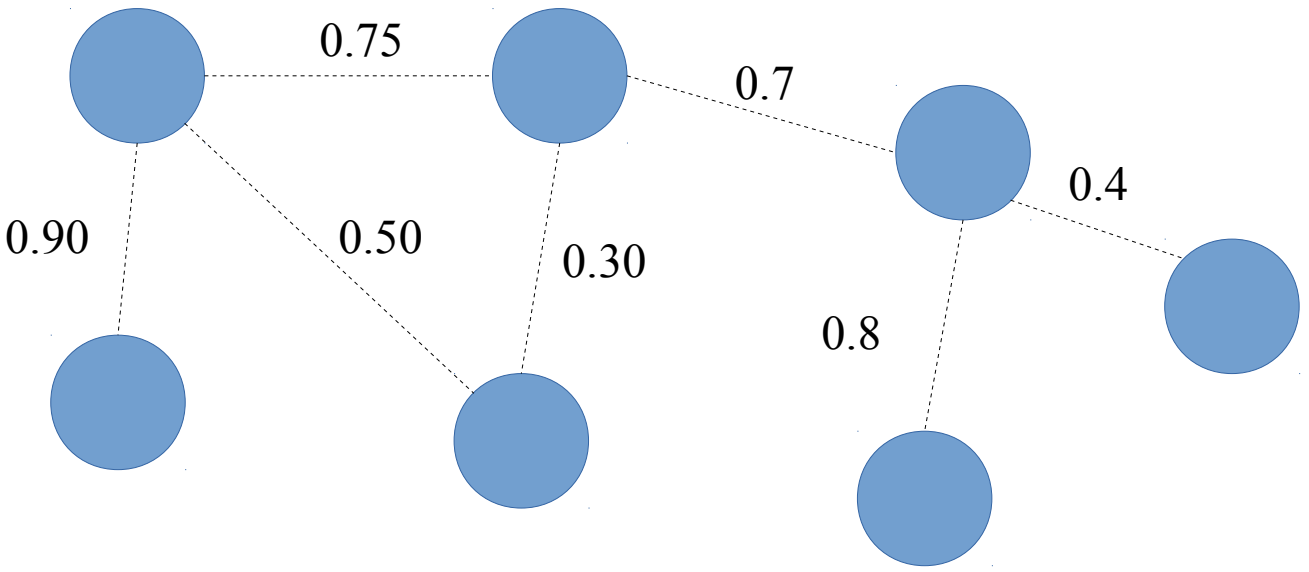
План

1)Собрать небольшую (<20.000.000) выборку ребер с наибольшей полнотой для трейна, валидации, теста.
2)Сгенерировать признаки для каждого ребра (построить матрицу объекты-признаки).
3)Обучиться по трейн выборке и предсказать вероятности для валидационной и тестовой выборки.
4)Взять топ N наиболее вероятных ребер, дополнить до полных подграфов (N и параметры дополнения подобрать на валидационной выборке)

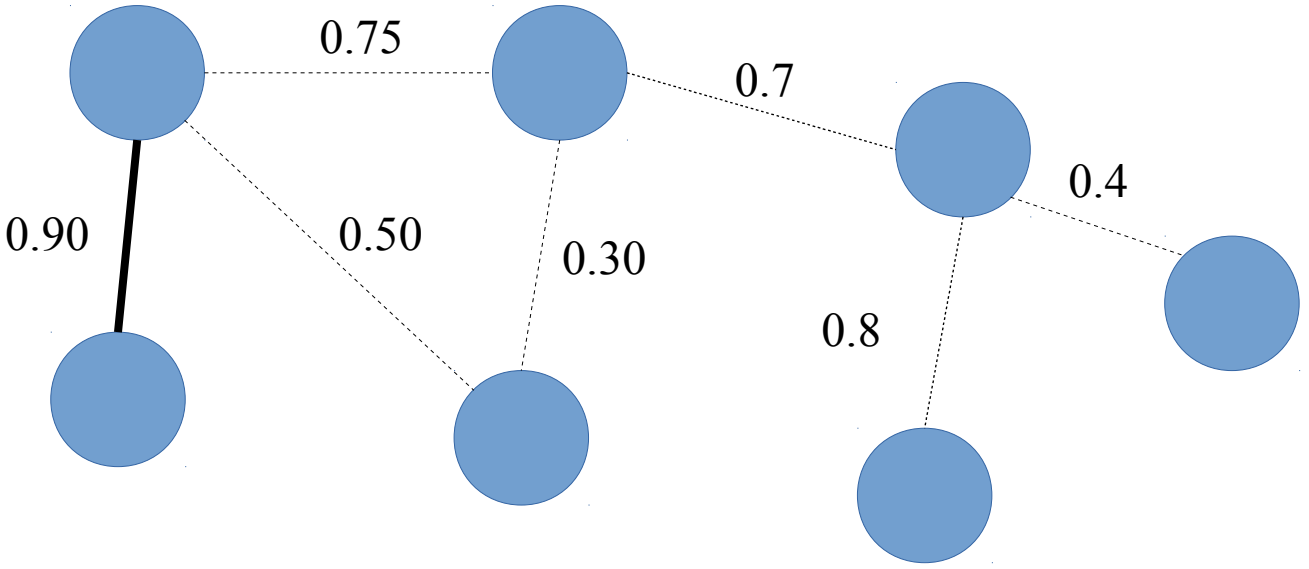
А что если взять топ-N ребер (где N очень большое), дополнить их до полных кластеров и получившуюся выборку снова отправить на шаг 2? Насколько вырастет полнота выборки с шага 1?

Алгоритм:
1)Отсортировать ребра по предсказанной вероятности по убыванию.
2)По одному добавлять ребра в граф.
3)Не добавлять ребро, если вероятность < p.
4)Не добавлять ребро, если оно объединяет компоненты связности, произведение размеров которых > T.

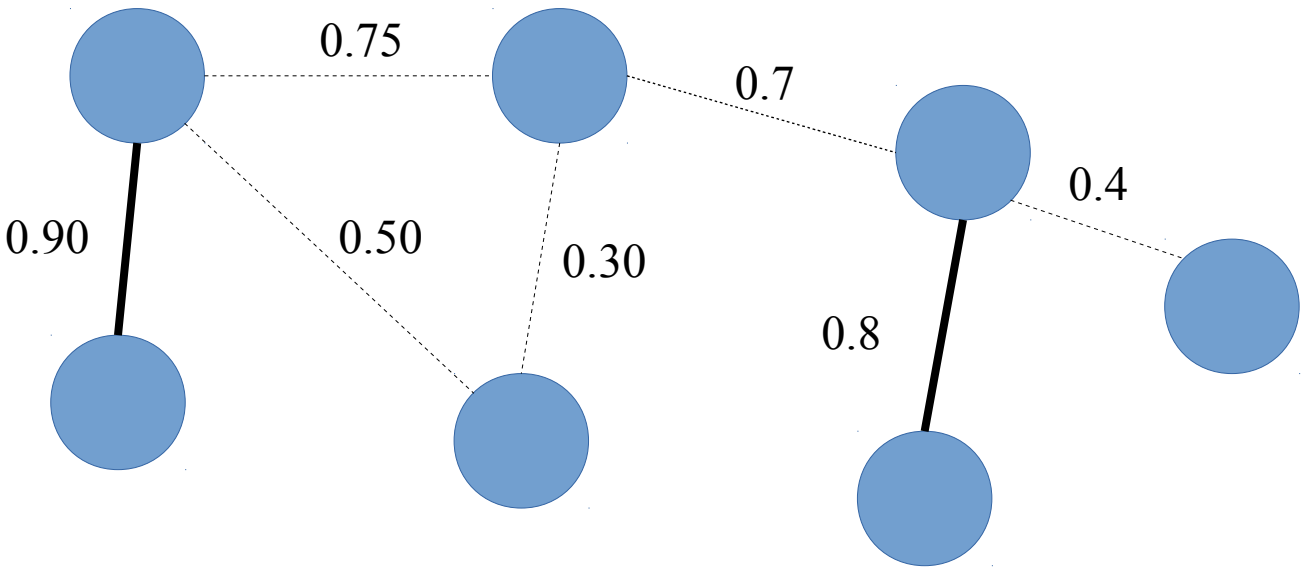
Пример (T=5, p=0.35):



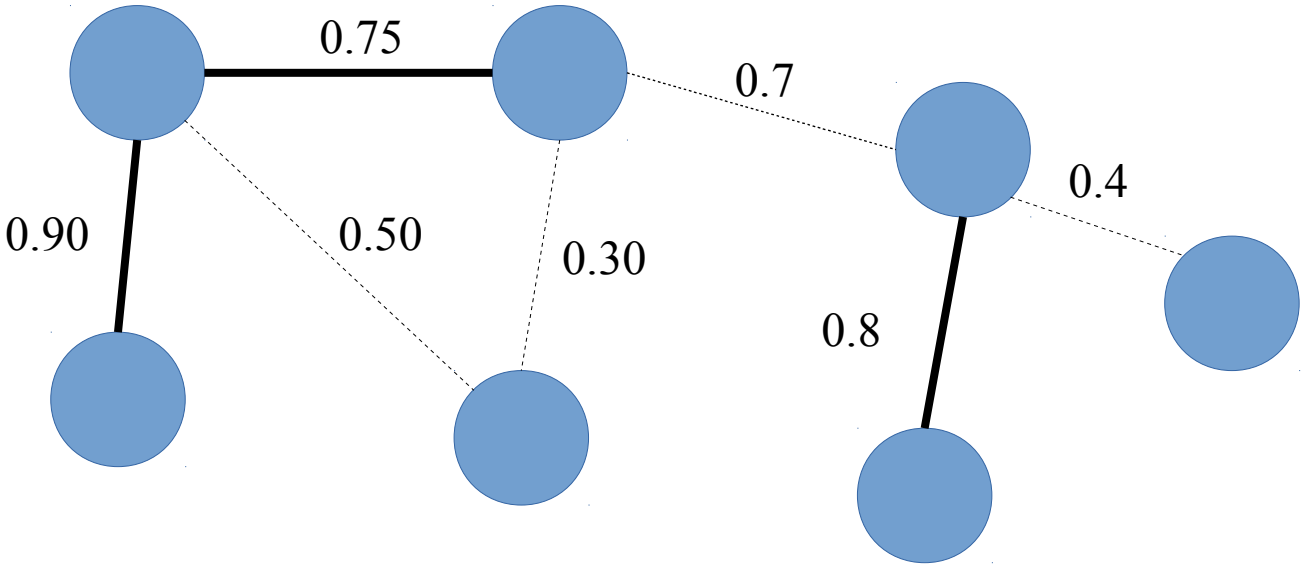
Пример (T=5, p=0.35):



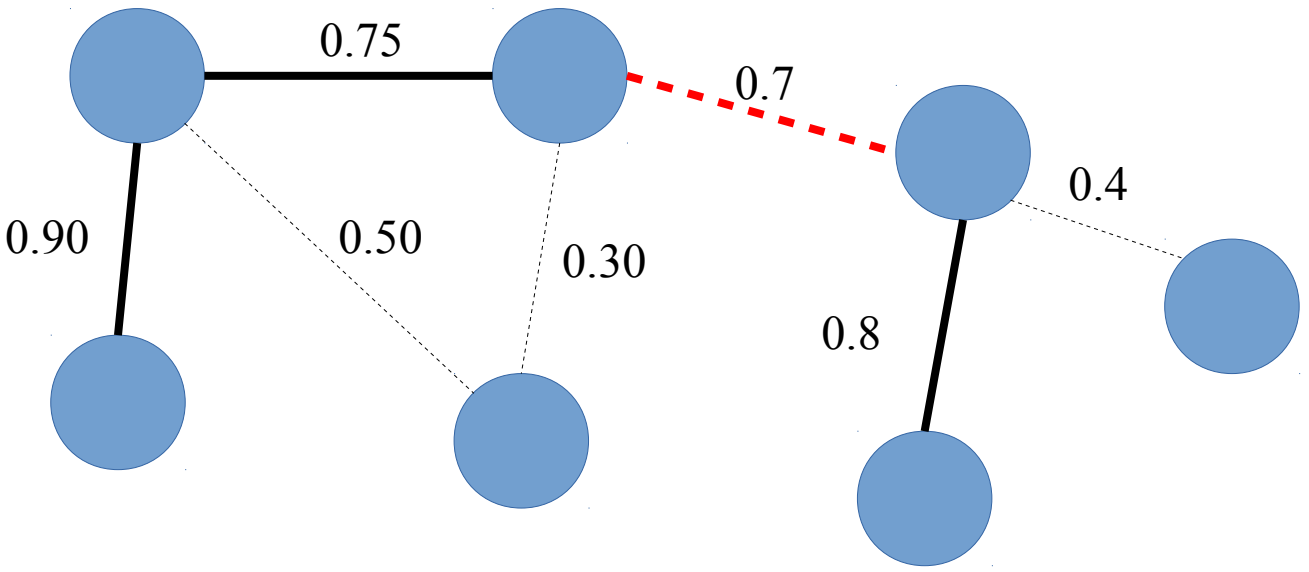
Пример (T=5, p=0.35):



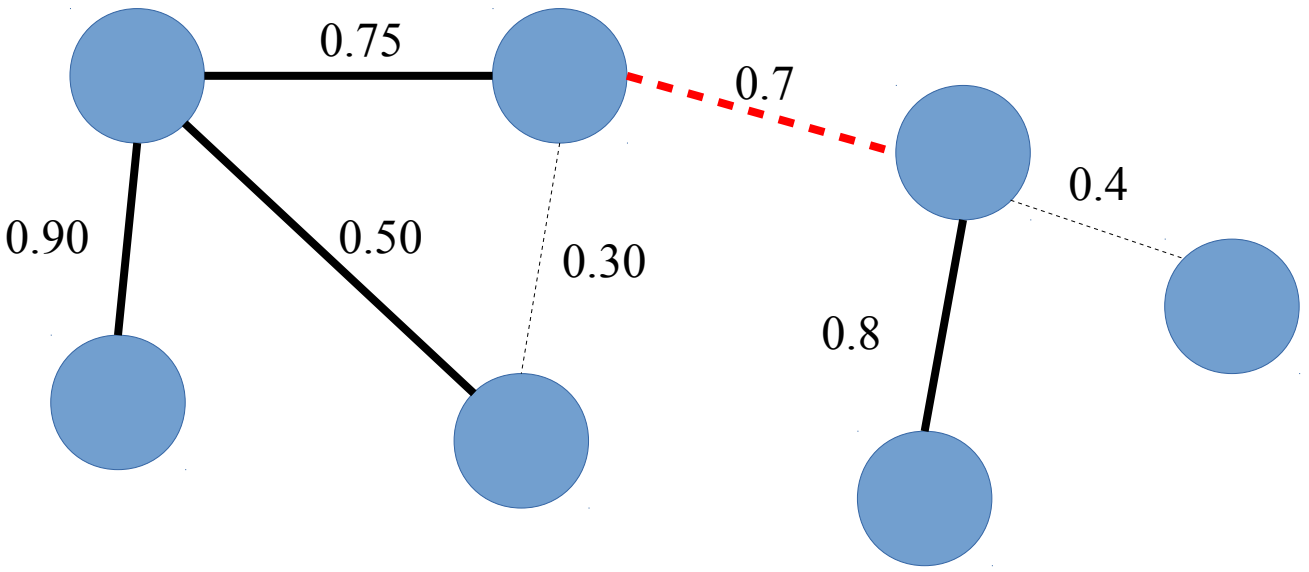
Пример (T=5, p=0.35):



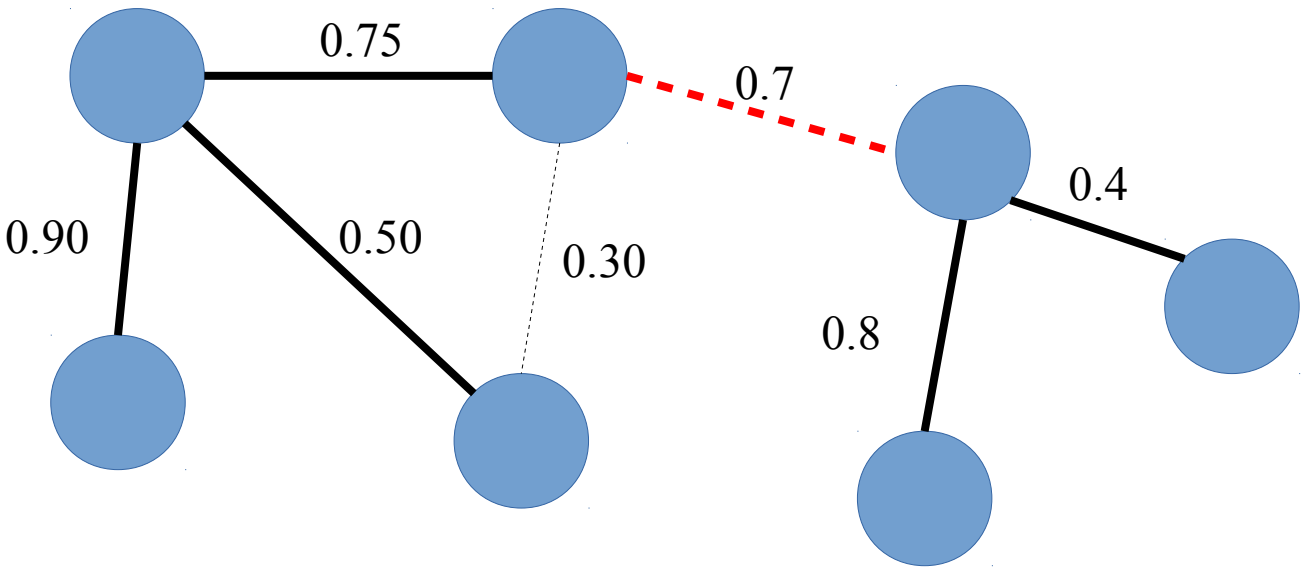
Пример (T=5, p=0.35):



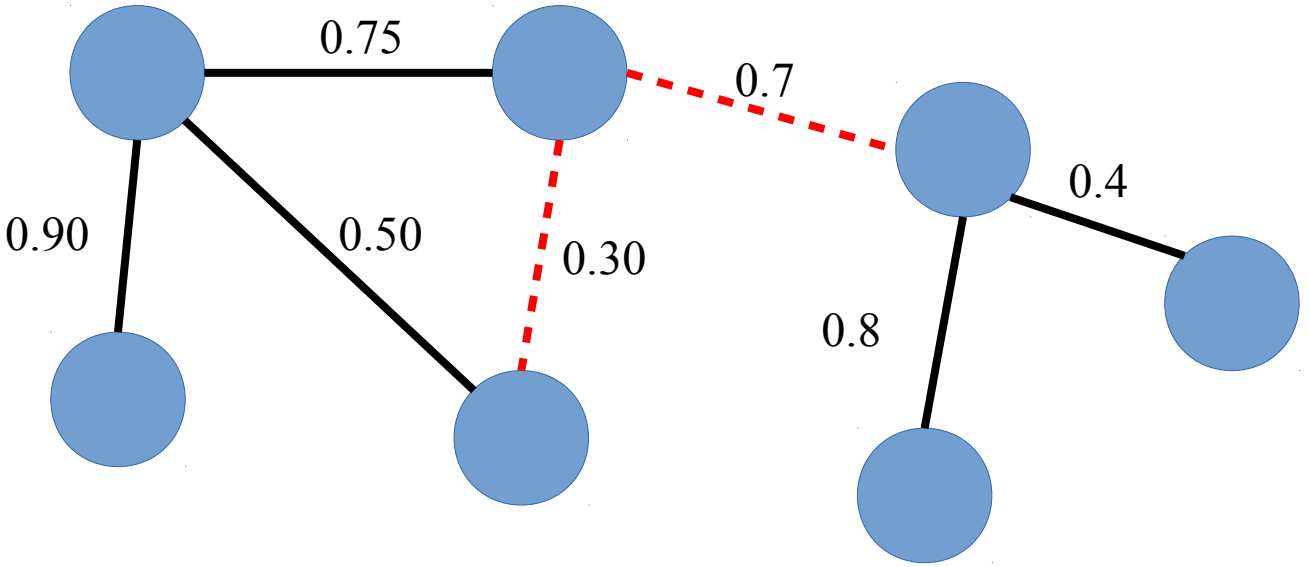
Пример (T=5, p=0.35):



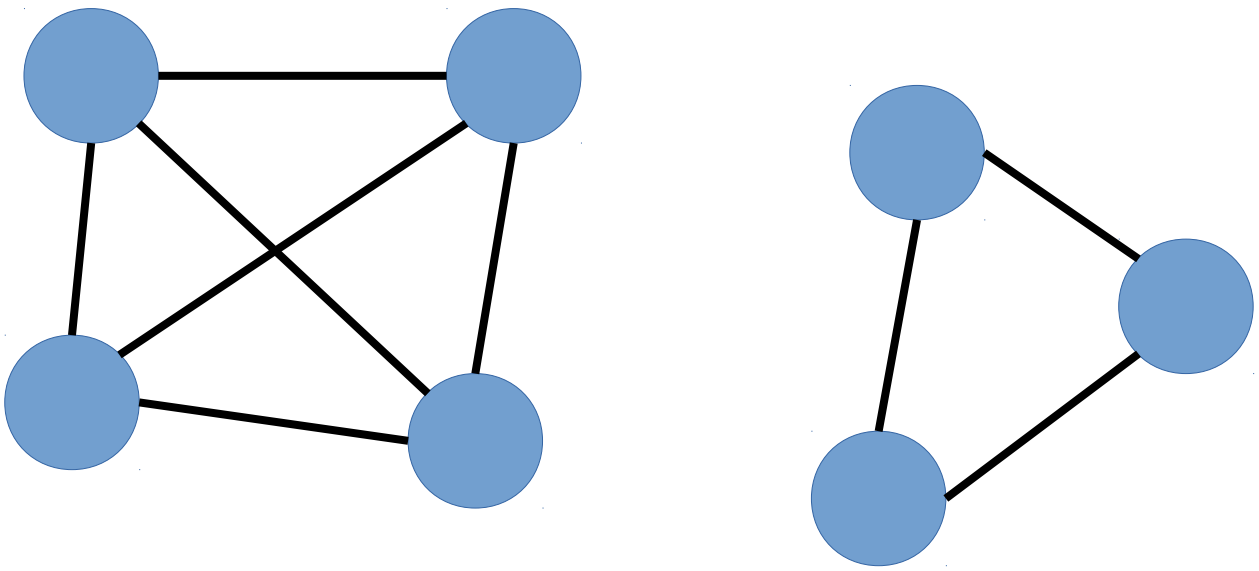
Пример (T=5, p=0.35):



Пример (T=5, p=0.35):



Пример:



Дополняем до полных кластеров и добавляем все эти ребра в выборку

Параметры алгоритма p и T подбираем на валидации так, чтобы полнота была наибольшей и количество ребер было не больше 10M.

p=0, T=400 – recall = 0.77

Выборка для обучения:

M	1 общее событие	2 общих события	3 общих события	4 общих события
50	0.5634 (34m)	0.4035 (7.7m)	0.3042 (3.4m)	0.2375 (2m)
70	**0.5873 (46m)**	0.4260 (10m)	0.3249 (4.6m)	0.2561 (2.7m)
100		**0.4520 (14m)**	0.3487 (6.3m)	**0.2769 (3.7m)**
150			**0.3763 (9m)**	

Объединяем выделенные четыре выборки + все ребра из базового KNN (для 200 соседей) + полные подграфы из предыдущего пункта.

Полнота на трейне **0.837**, на тесте **0.778**

Количество ребер в трейне **21M**

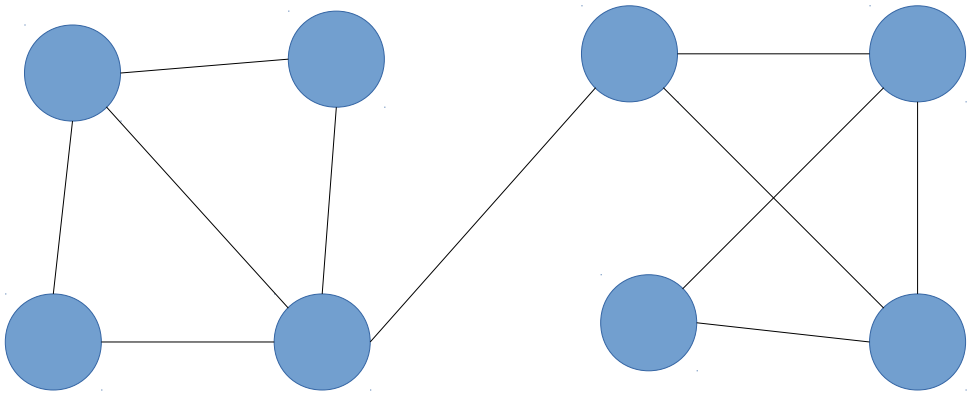
Обучение модели

Те же признаки
Та же модель (Random Forest)
Такое же качество при обучении: ROC AUC ≈ 0.955
Результат на LB: **0.372**

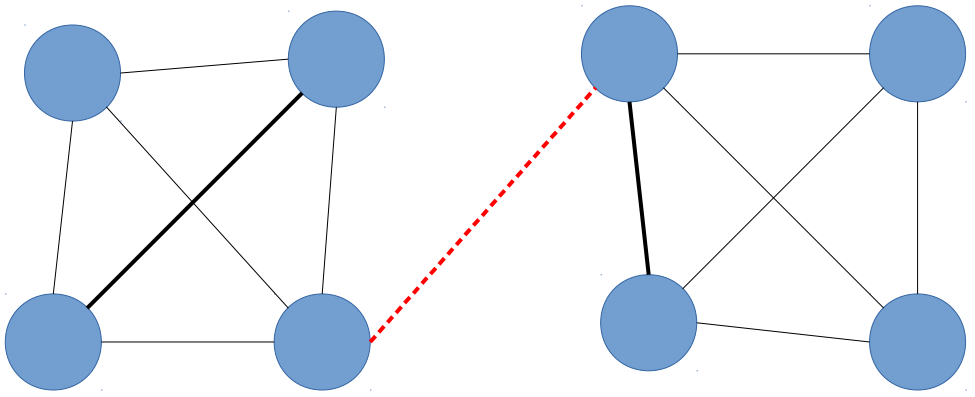
План

1)Собрать небольшую (<20.000.000) выборку ребер с наибольшей полнотой для трейна, валидации, теста.
2)Сгенерировать признаки для каждого ребра (построить матрицу объекты-признаки).
3)Обучиться по трейн выборке и предсказать вероятности для валидационной и тестовой выборки.
4)Взять топ N наиболее вероятных ребер, дополнить до полных подграфов (N и параметры дополнения подобрать на валидационной выборке)

Кластеризация



Кластеризация



Кластеризация

Алгоритм:

1)Отсортировать ребра по предсказанной вероятности по убыванию.

2)По одному добавлять ребра в граф.

3)Не добавлять ребро, если вероятность < p.

4)Не добавлять ребро, если оно объединяет компоненты связности, произведение размеров которых > T.

Получаем много компонент связности размера не больше T.

5)Применяем Markov Cluster Algoritm – разбиваем каждую компоненту на кластеры.

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

0	0.9	0.1	0.1
0.9	0	0.05	0.2
0.1	0.05	0	0.95
0.1	0.2	0.95	0

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

1	0.9	0.1	0.1
0.9	1	0.05	0.2
0.1	0.05	1	0.95
0.1	0.2	0.95	1

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

0.47	0.41	0.04	0.04
0.42	0.46	0.02	0.08
0.04	0.02	0.47	0.42
0.04	0.09	0.45	0.44

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

0.41	0.39	0.07	0.09
0.40	0.40	0.08	0.10
0.07	0.08	0.42	0.39
0.10	0.11	0.42	0.39

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

0.16	0.15	0	0
0.16	0.16	0	0.01
0	0	0.17	0.15
0.01	0.01	0.17	0.15

MCL

1)Добавить петли в граф.
2)Нормализовать, чтобы веса соответстовали вероятностям.
3)Возвести матрицу в степень n, что соответствует случайному блужданию с n шагами.
4)Возвести каждый элемент матрицы в степень p, чтобы усилить большие вероятности и ослабить маленькие.
5)Повторять шаги 2-5, пока матрица не сойдется.

0	0	0	0
1	1	0	0
0	0	0	0
0	0	1	1

MCL

Анимация: http://www.micans.org/mcl/ani/mcl-animation.html

Финальное решение

По валидационной выборке подбираем параметры разбиения на компоненты и параметры MCL.
Результат на public лидерборде: **0.418** (1е место)
Результат на private лидерборде: **0.401** (4е место)

ЭВМ

Генерация выборки и признаков – Amazon C3.4 (16 cpu, 30Gb), 100 Hrs
Обучение модели - Amazon M4.10 (40 cpu, 160Gb), 20 Hrs
Кластеризация - ноутбук

Ссылки

1)Страница соревнования
https://competitions.codalab.org/competitions/11171
2)MCL от создателя http://www.micans.org/mcl/
3)Объяснение MCL на пальцах
http://dogdogfish.com/mathematics/markov-clustering-what-is-it-and-why-use-it/
4)Подробная презентация про MCL
http://www.cs.ucsb.edu/~xyan/classes/CS595D-2009winter/MCL_Presentation2.pdf

Вопросы