Предсказание появления рёбер в темпоральных (временных) графах

Командный проект по курсу «Теория графов и ее приложения»

Цель проекта:

Предсказать, появится ли ребро между парой вершин (u, v) к моменту времени t″, если на момент времени t* ребро между этими вершинами отсутствовало.

Наша команда

Тарелкина Анастасия

Task 1.1
Task 2.Б

Киселёв Владимир

Task 1.3 , 1.4
Task 2.А(III)

Федорова Анастасия

Task 1.2АБ
Task 2.А(I)

Данные

01 **opsahl-ucsocial**

отправленные сообщения между пользователями онлайн-сообщества студентов

02 **radoslaw_email**

сеть электронной связи между сотрудниками производственной компании

03 **soc-sign-bitcoinotc**

сеть доверия/недоверия между пользователями от внебиржевой платформы Bitcoin

04 **dnc-corecipient**

сеть людей, получивших одно и то же электронное письмо во время утечки электронной почты Нац. комитета Демократической партии в 2016 году

05 **email-Eu-core-temporal**

сеть была создана с использованием данных электронной почты крупного европейского исследовательского учреждения

Структура данных для хранения

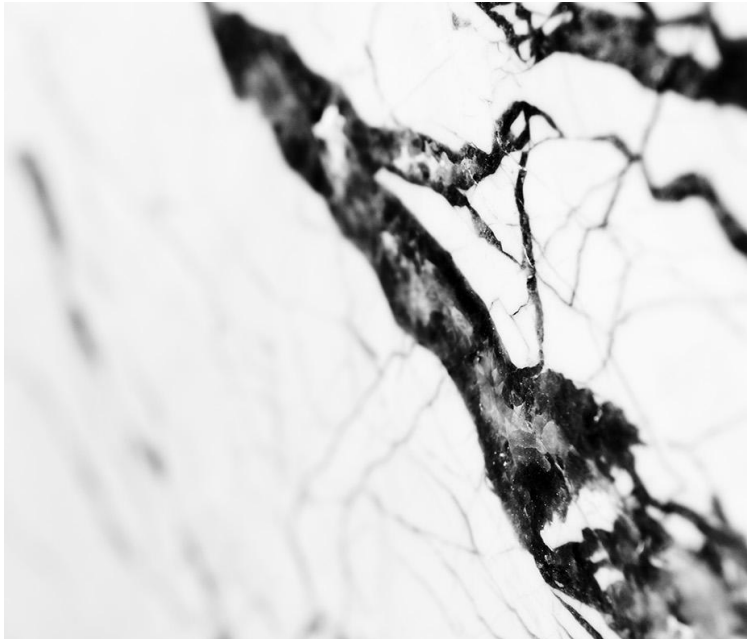
	v1	v2	timestamp
0	582	364	0
1	168	472	2797
2	168	912	3304
3	2	790	4523
4	2	322	7926

Структура для работы с файлами - DataFrame

- структура данных табличного типа.
- манипулирование и управление столбцами и строками.
- позволяет читать и записывать данные разных форматов
- быстрое слияние и объединение наборов больших данных, например, два и более объектов DataFrame.

Свойства сетей (для статических графов)

Для расчета статических свойств данных мы считаем ребра в графе неориентированными (без учета кратности рёбер)



Task 1.1

Далее рассмотрим такие характеристики графа:

- Число вершин
- Число направленных рёбер
- Число ненаправленных рёбер (некратные ненаправленные ребра)
- Плотность графа - отношение числа рёбер к максимально возможному числу рёбер
- Число компонент слабой связности - поиск компонент происходил с помощью DFS
- Доля максимальной компоненты связности

	opsahl-ucsocial	radoslaw_email	soc-sign-bitcoinotc	dnc-corecipient	email-Eu-core-temporal
Число вершин	1899	167	5881	906	986
Число направленных рёбер	59835	82927	35592	12085	332334
Число ненаправленных рёбер	13838	3251	21492	10429	16064
Плотность графа	0.007678601848568738	0.23454296226823462	0.001243020588612932	0.025438756967058163	0.033080384262929745
Число компонент слабой связности	4	1	4	25	1
Доля максимальной компоненты связности	0.9968404423380727	1	0.9989797653460296	0.9370860927152318	1

Task 1.2

Для наибольшей компоненты слабой связности оценить значения диаметра, радиуса сети и 90 процентиля

“Случайные вершины”

Выбираются 500 случайных вершин из наибольшей компоненты.

“Снежный ком”

Выбираются 2 случайных вершины, затем, пока кол-во <500 добавляем соседей этих вершин, соседей соседей и так далее .

Для наибольшей компоненты слабой связности		opsahl-ucsocial	radoslaw_email	soc-sign-bitcoinotc	dnc-corecipient	email-Eu-core-temporal
Подграф - 500 случайно выбранных вершин	Радиус	5	3	5	4	4
	Диаметр	8	5	9	8	7
	90 процентиль	4	3	4	4	3
Подграф - “снежный ком” (500 вершин)	Радиус	5	3	6	4	4
	Диаметр	6	5	7	6	6
	90 процентиль	4	3	4	4	3

Для наибольшей компоненты слабой связности		**opsahl-ucsocial**	**radoslaw_email**	**soc-sign-bitcoinotc**	**dnc-corecipient**	**email-Eu-core-temporal**
Подграф - 1000 случайно выбранных вершин	Радиус	4	3	5	4	4
	Диаметр	8	5	9	8	7
	90 процентиль	4	3	5	4	3
Подграф - “снежный ком” (1000 вершин)	Радиус	4	3	5	4	4
	Диаметр	6	5	7	8	7
	90 процентиль	4	3	4	4	3

Средний кластерный коэффициент

Чтобы получить множество Г требуется найти всех соседей для каждой из вершины данной компоненты.

L_u - пересечение двух множеств всевозможных пар между соседями узла u. Далее зная, как найти Г(u) и L_u подставляем в формулу и считаем ACC.

$$\bar{C}l = \frac{1}{|V|}\sum_{u \in G} Cl_u$$

$$Cl_u = \begin{cases} \frac{2L_u}{|\Gamma(u)| \cdot |\Gamma(u)-1|}, & |\Gamma(u)| \geqslant 2 \\ 0, & \text{иначе.} \end{cases}$$

Коэффициент корреляции Пирсона
(Коэффициент ассортативности по степени вершин)

Для расчета нам также потребуется список всех соседей для каждой из вершин. Как результат, k_i это модуль от множества всех соседей i-й вершины. Подставляем известные нам данные в нашу формулу и считаем DA(PCC)

$$r = \frac{\sum_{i,j}\left(A_{ij} - \frac{k_i k_j}{2m}\right) k_i k_j}{\sum_{i,j}\left(k_i \delta_{ij} - \frac{k_i k_j}{2m}\right) k_i k_j} = \frac{R_e R_1 - R_2^2}{R_3 R_1 - R_2^2},$$

где $R_1 = \sum_i k_i = 2m,\ R_2 = \sum_i k_i^2,\ R_3 = \sum_i k_i^3, R_e = \sum_{i,j} A_{ij} k_i k_j.$

	opsahl-ucsocial	radoslaw_email	soc-sign-bitcoinotc	dnc-corecipient	email-Eu-core-temporal
Средний кластерный коэффициент	0.10974567163130962	0.5918632085486949	0.17768577523831092	0.5071859111900863	0.40705044751955388
Коэффициент корреляции Пирсона (Коэффициент ассортативности по степени вершин)	-0.1877757871466803	-0.2944241603890704	-0.1648335945131314522	-0.12489244283998932	-0.025743368083089496

Предсказания появления ребер в графе. Признаки.

Static topological features

01 Common Neighbours (CN)

Функция CN равна числу общих соседей двух вершин. Для рассматриваемой пары вершин находим общих соседей - их число CN.

02 Adamic-Adar (AA)

Функция AA учитывает всех общих соседей. Она определяется как сумма обратных логарифмических степеней центральности соседей.

Static topological features

03 Jaccard Coefficient (JC)

Функция JC определяется как отношение числа общих соседей к числу соседей обеих вершин вместе (объединение).

04 Preferential Attachment (PA)

Функция PA определяется как произведение количества соседей вершин.

Node activity features

01 Temporal weighting - топологическое взвешивание

Необходимо посчитать топологический вес в зависимости от времени появления ребра.

l = 0.2 - нижняя граница

t_min = df.timestamp.min()

t_max = df.timestamp.max()

delta_t = t_max - t_min

Node activity features

02 Aggregation of node activity - агрегирование активности

Чтобы получить фиксированный вектор признаков для каждого узла, набор весов агрегируется с использованием 7-ми функций:
(1) нулевой, (2) первый,
(3) второй, (4) третий, (5) четвертый квартиль, (6) сумма и (7) среднее значение

```
[ ]  def get_zeroth(w):
       return np.min(w)

     def get_first(w):
       return np.quantile(w, .25)

     def get_second(w):
       return np.median(w)

     def get_third(w):
       return np.quantile(w, .75)

     def get_fourth(w):
       return w[-1]

     def get_sum(w):
       return np.sum(w)

     def get_mean(w):
       return np.mean(w)
```

Node activity features

03 Combining node activity - Объединение активности узла

Используем четыре различные комбинированные функции:
(1) сумма, (2)абсолютная разница, (3) минимум и (4) максимум.
Делая это, мы получаем вектор признаков активности узла.

```python
def get_sum_c(a, b):
  return a + b

def get_abs_diff(a, b):
  return abs(a - b)

def get_min(a, b):
  return min(a, b)

def get_max(a, b):
  return max(a, b)
```

Первая попытка обучить

- Рассматриваем весь датасет
- Выберем пары узлов, которые находятся на расстоянии 2.
- Для исходных пар вершин и расположенных на расстоянии 2 посчитаем вектора признаков (функции, о которых говорили ранее).
- С помощью выборки с повторениями выберем 10000 пар вершин между которыми существует ребро и 10000 пар вершин между которыми ребра нет - далее будем работать с этими данными.

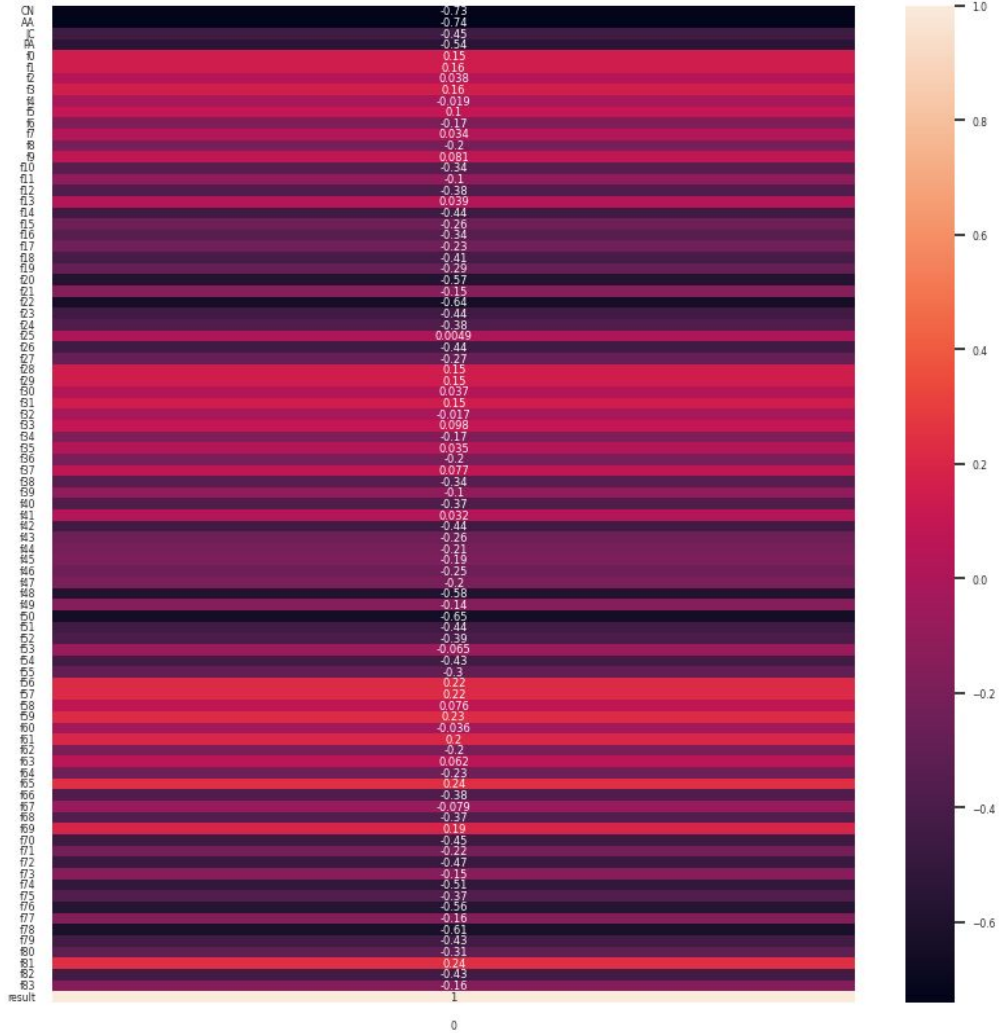
Первая попытка обучить

- Мы разделили нашу выборку на 2 части - train и test
- Из выборки train составляет 70% пар вершин (14000), test - 30% (7000)

- Получив результаты обучения, подозрительно высокие значения метрики ROCAUC
- И решили проверить, как полученные фичи коррелируют с результатом

```
opsahl-ucsocial
0.8887081111111111
radoslaw email email
0.9540153425809162
soc-sign-bitcoinotc
0.9161591111111111
dnc-corecipient
0.9853147777777778
email-Eu-core-temporal.txt
0.9670284444444444
```

Корреляция



Корреляция для
`dnc-corecipient`
`(смешанные фичи)`

Вывод

- Такой результат получился из-за того, что мы “заглядывали в будущее”: например, у нас в признаках фигурирует количество соседей. И, соответственно, у пар вершин положительного класса метрики, зависящие от количества соседей во много раз больше

- Далее мы решили попробовать другим способом

Вторая попытка обучить

Во второй раз мы изначально разделили наш датасет на dataset_before и dataset_after.
dataset_before - датасет, в котором время появления ребер меньше указанного timestampa.

Соответственно, в dataset_after - оставшиеся ребра.

Далее по датасету dataset_before мы находили пары вершин, где кратчайшее расстояние - 2(считая dataset_before исходным графом).

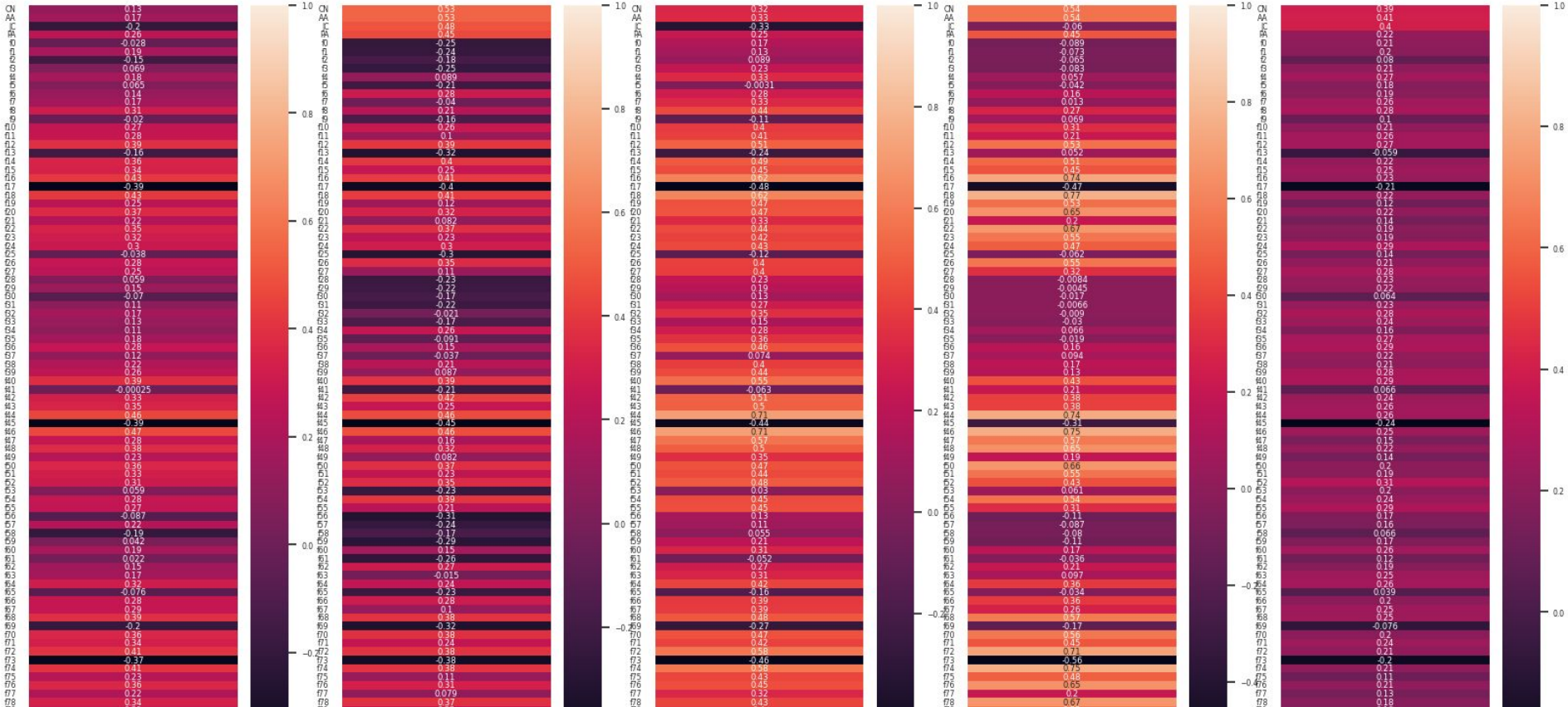
Вторая попытка обучить

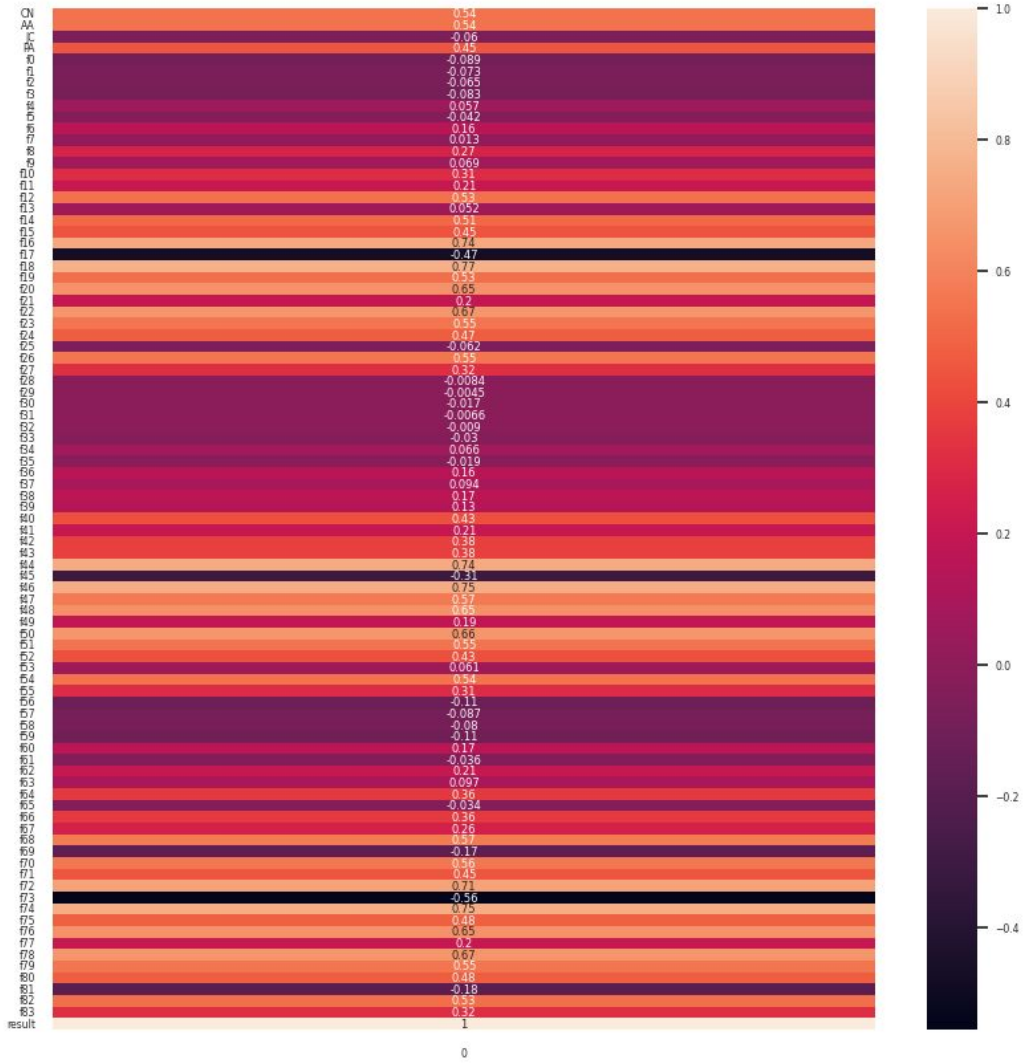
Далее находили пары вершин, которые появятся после нашего timestamp (list_of_good[i] - будет хранить пары вершин)

Соответственно, list_of_bad[i] - датасет с ребрами которые не появятся после нашего timestamp.

Результаты!

Проверим корреляцию





Корреляция для `dnc-corecipient` (смешанные фичи)

ROCAUC

Смешанные

opsahl-ucsocial
0.8626110555555555

radoslaw_email_email
0.8810643888888889

soc-sign-bitcoinotc
0.9560036666666666

dnc-corecipient
0.9551271666666667

email-Eu-core-temporal
0.8664972222222221

Статические

opsahl-ucsocial
0.7672692222222222

radoslaw_email_email
0.8351891666666666

soc-sign-bitcoinotc
0.8794977777777778

dnc-corecipient
0.9461202777777779

email-Eu-core-temporal
0.8085819444444444

Временные

opsahl-ucsocial
0.8426486111111111

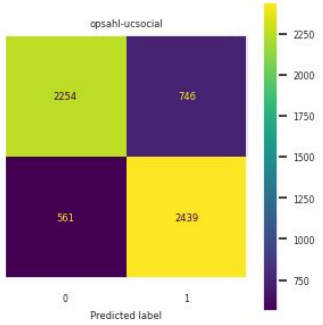
radoslaw_email_email
0.8440225

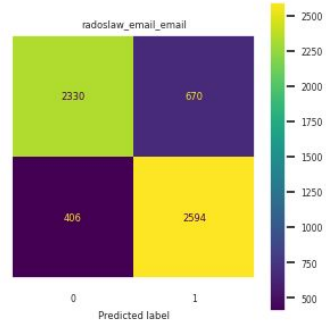
soc-sign-bitcoinotc
0.9488447777777779

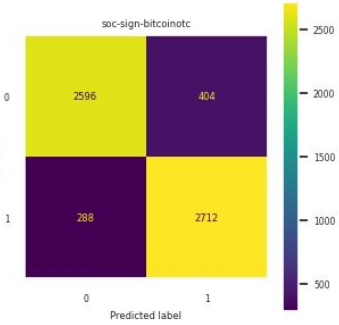
dnc-corecipient
0.9481207222222222

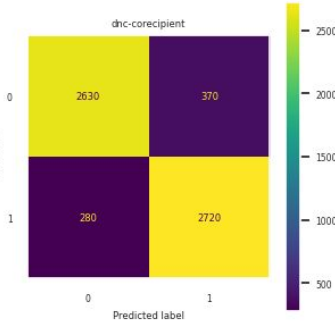
email-Eu-core-temporal
0.7590081111111111

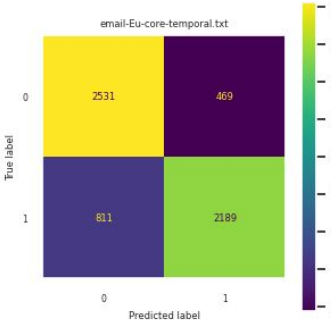
Смешанные



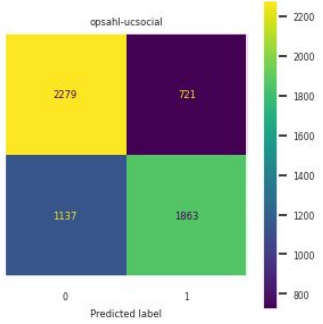


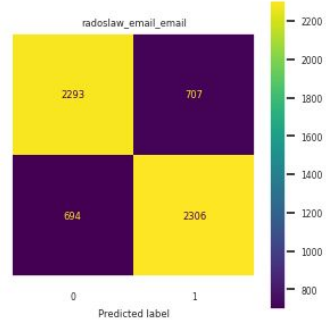


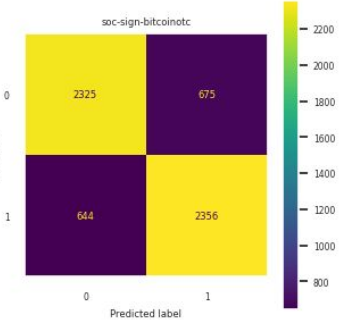


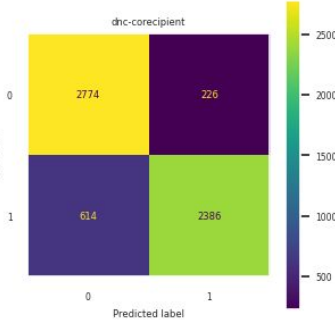


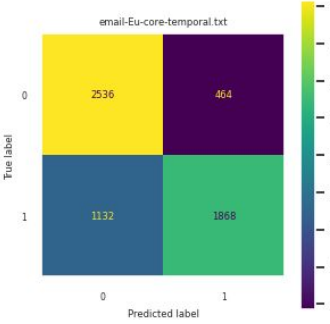
Статические



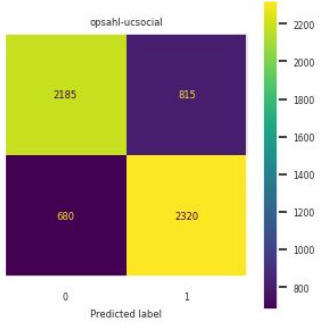


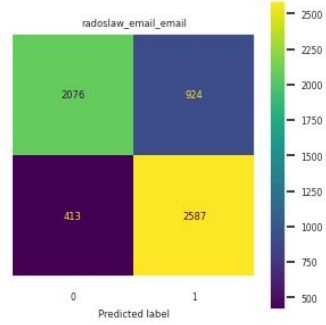


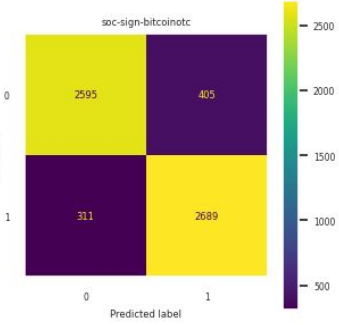


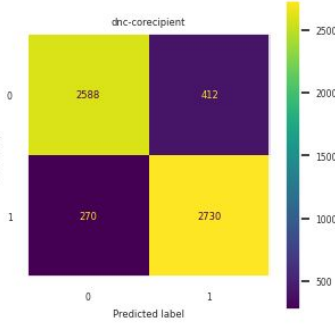


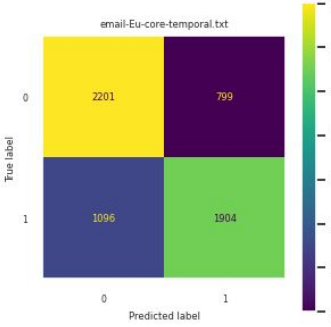
Временные



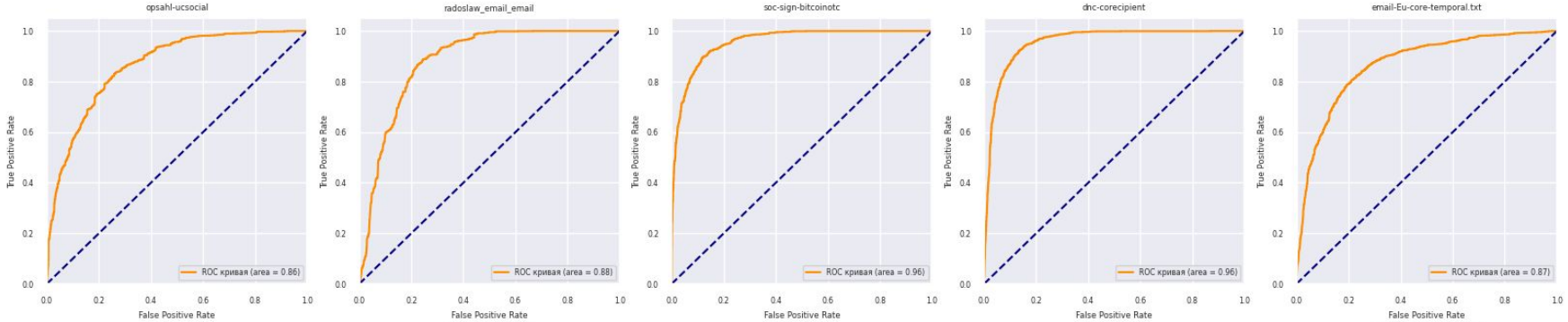




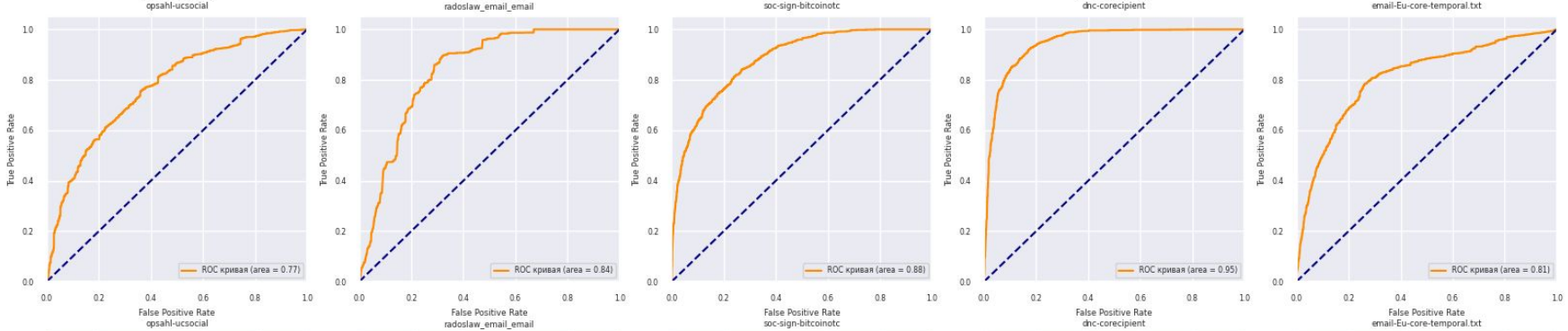




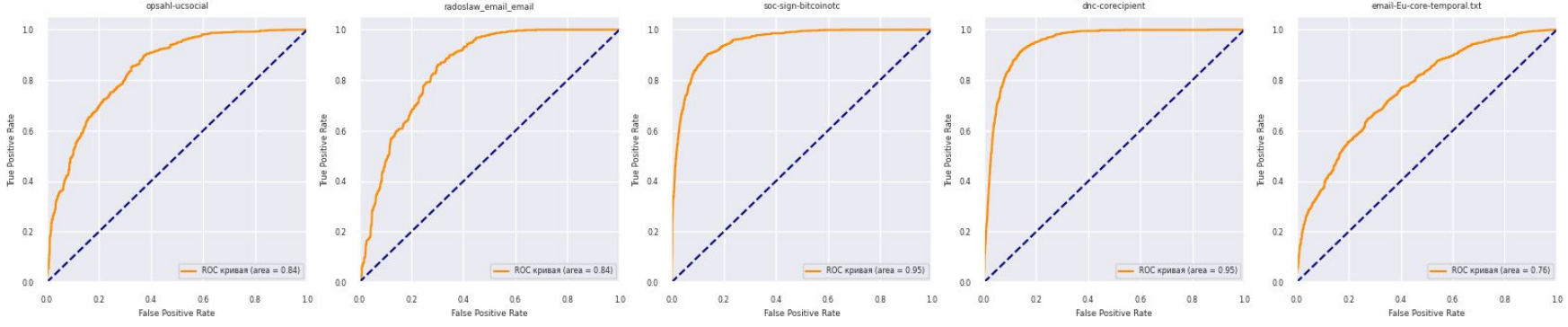
Смешанные



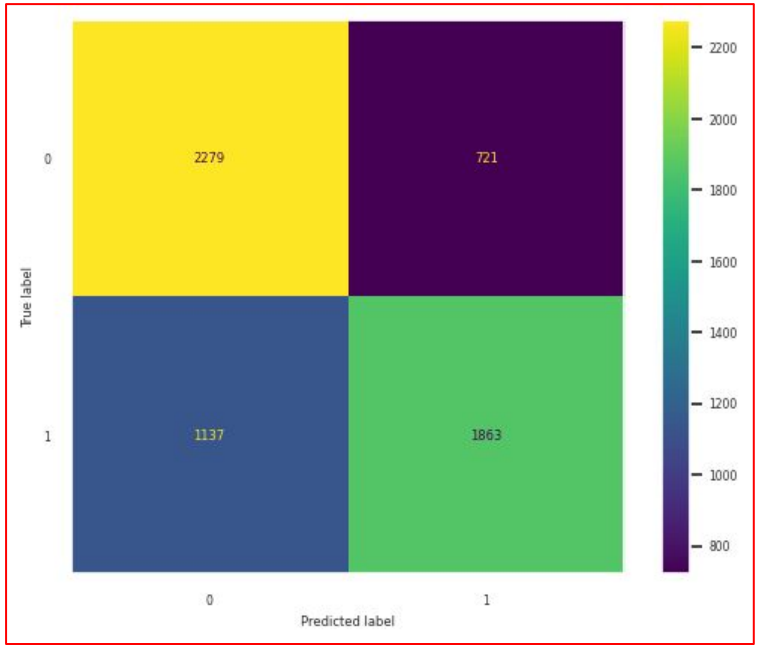
Статические

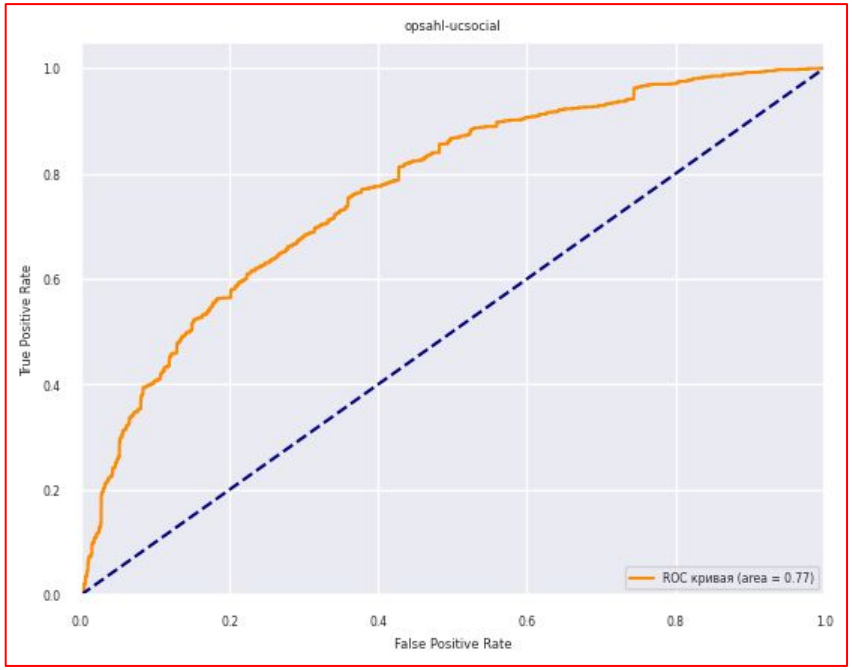


Временные



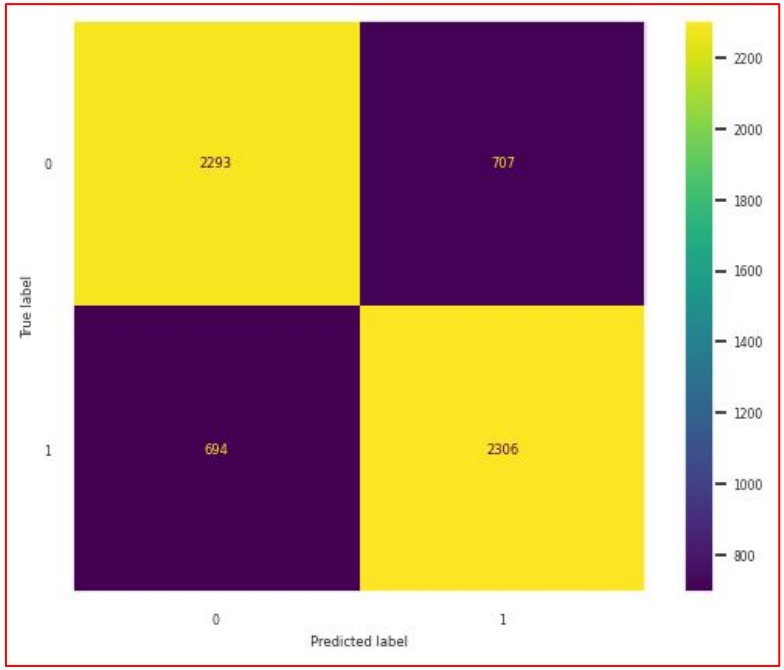
Результаты (статические)

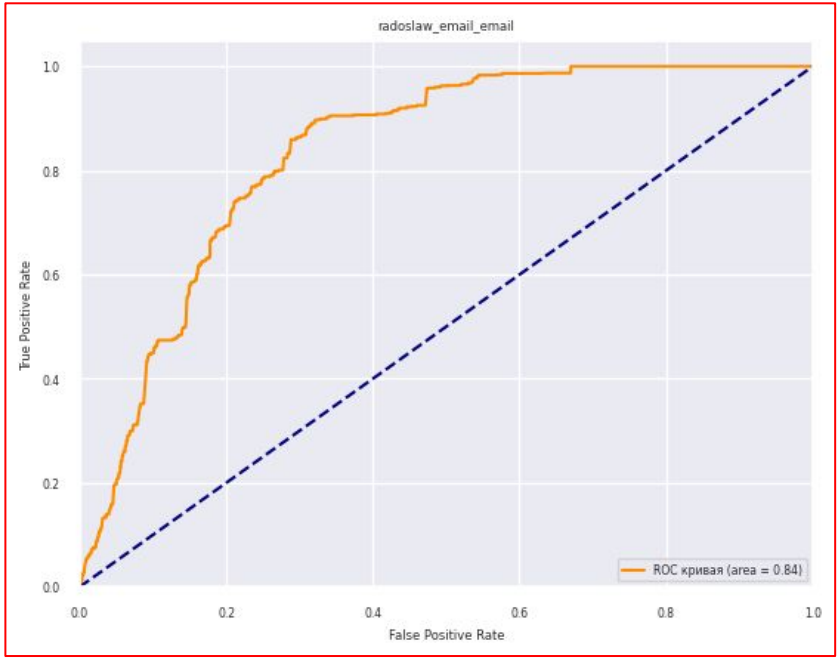




opsahl-ucsocial
0.7672692222222222

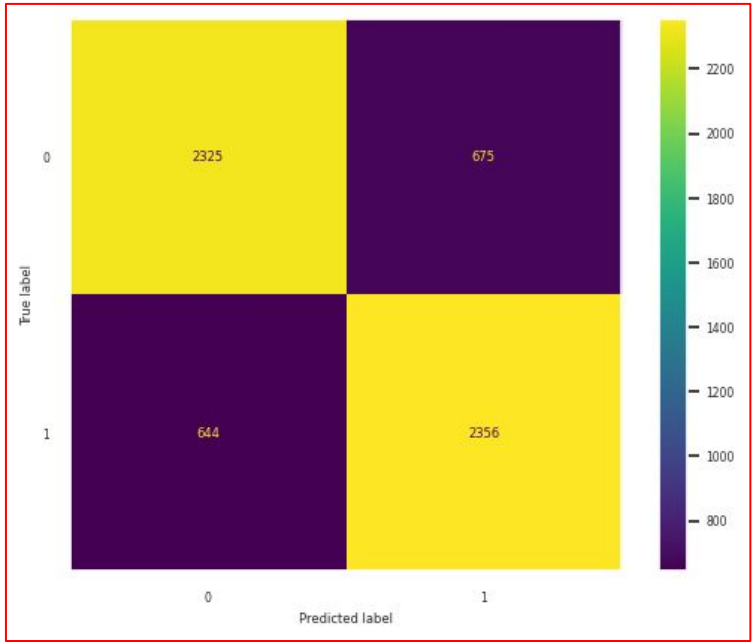
Результаты (статические)

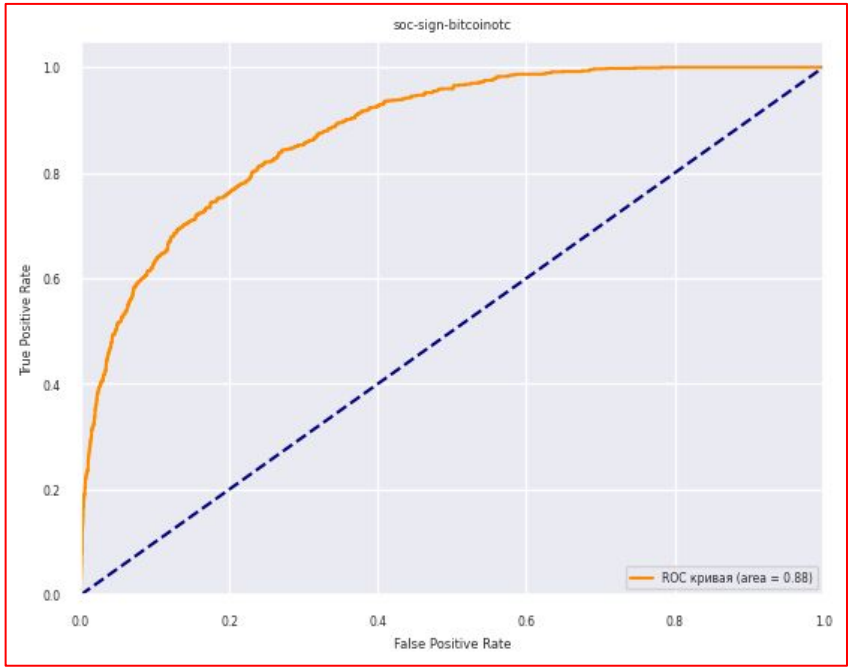





```
radoslaw_email_email
0.8351891666666666
```

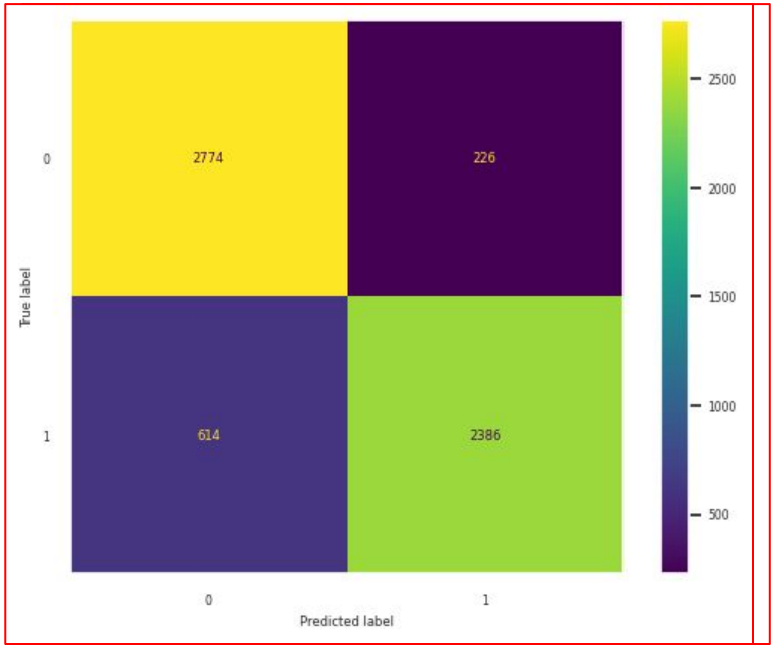
Результаты (статические)

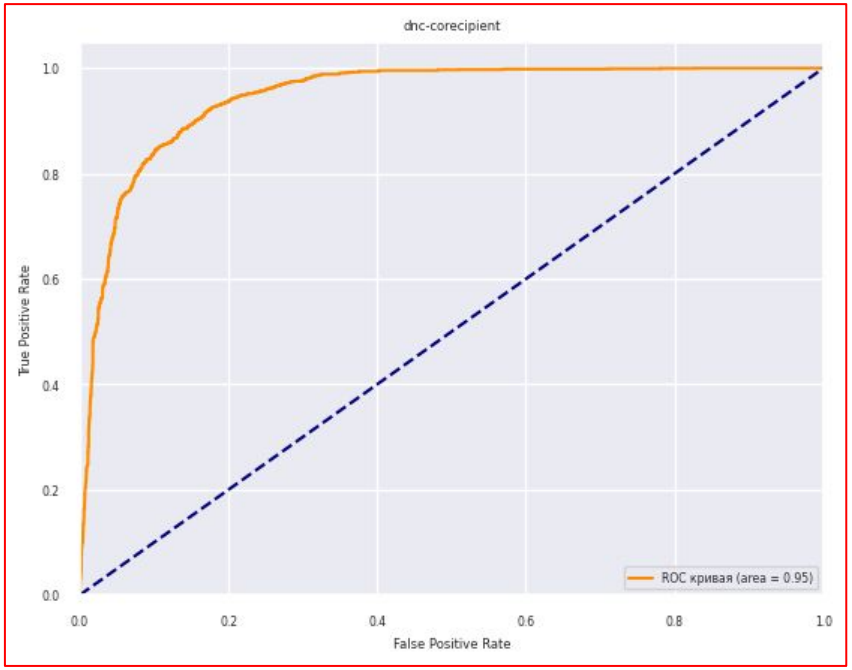




soc-sign-bitcoinotc
0.8794977777777778

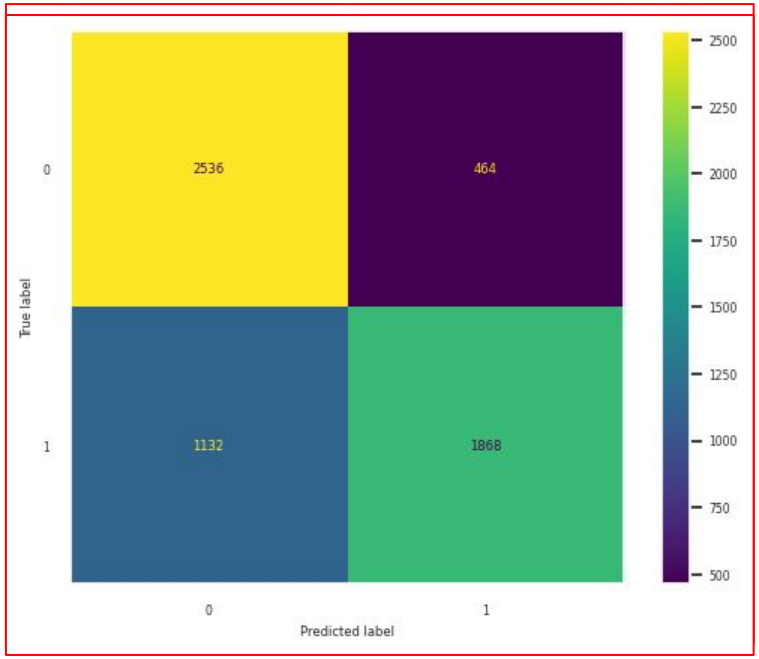
Результаты (статические)

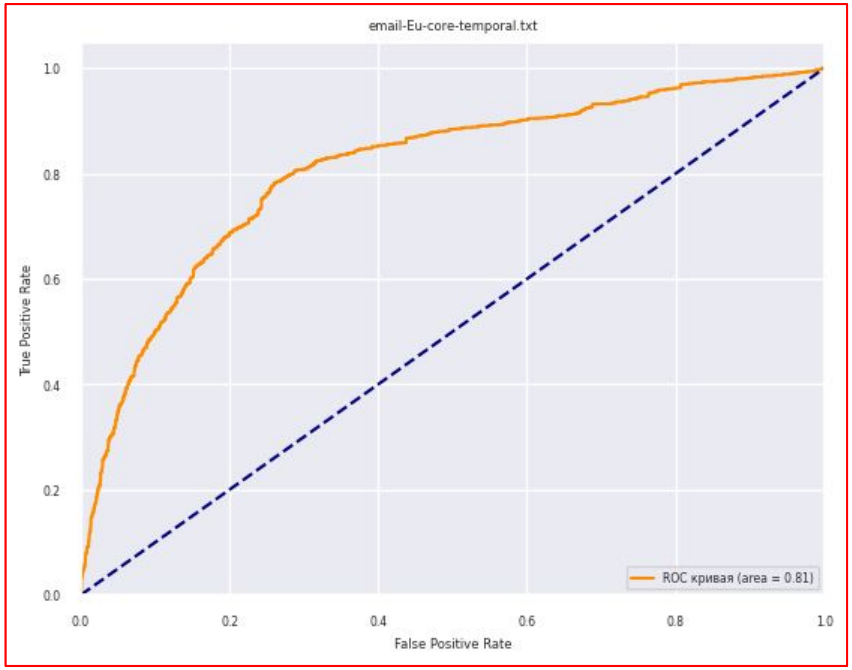




dnc-corecipient
0.9461202777777779

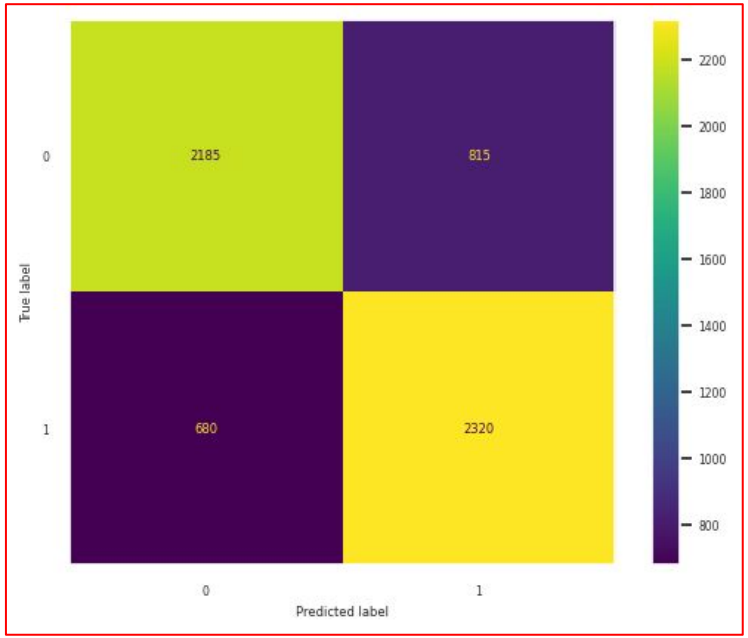
Результаты (статические)

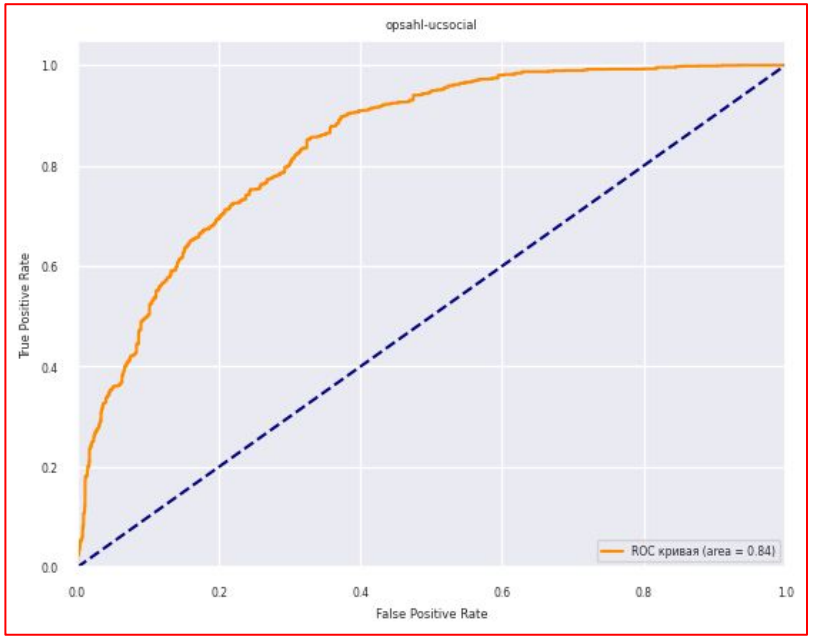




email-Eu-core-temporal.txt
0.8085819444444444

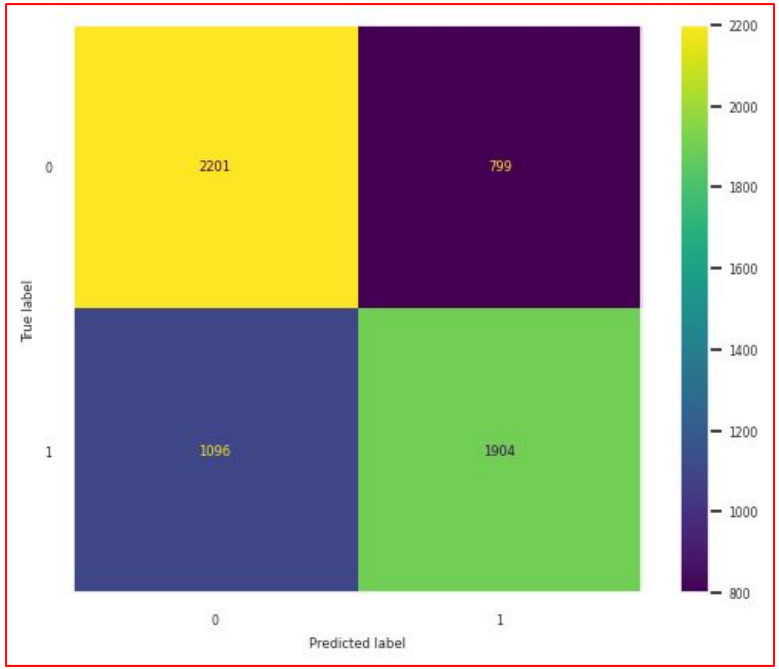
Результаты (динамические)

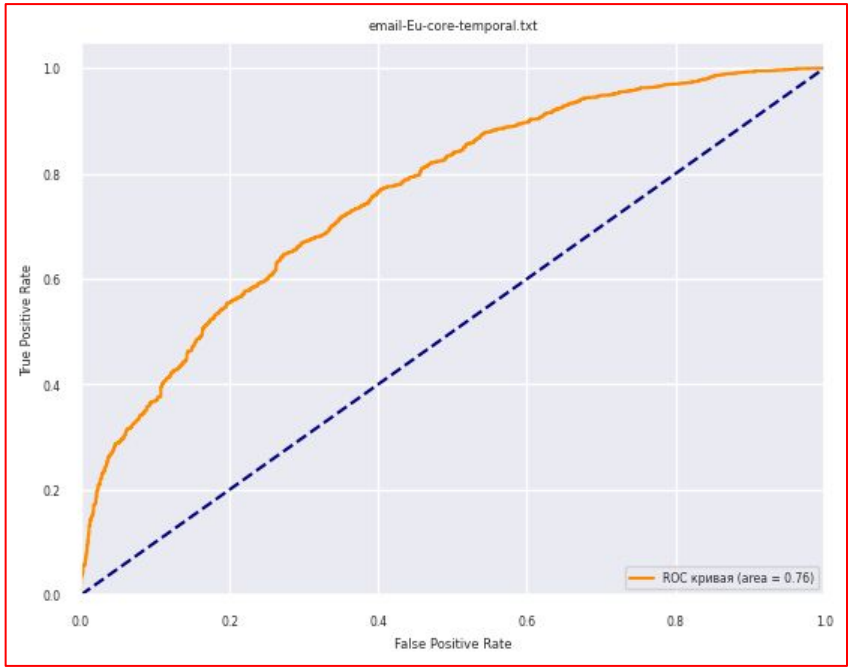




opsahl-ucsocial
0.8426486111111111

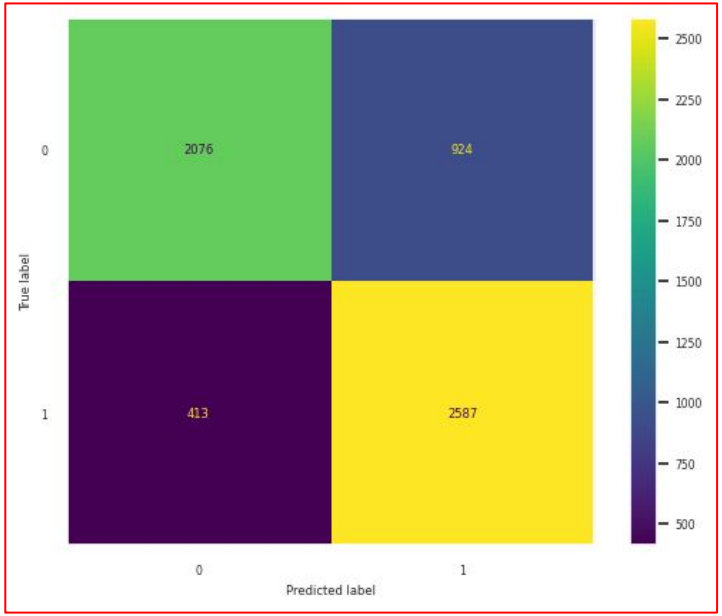
Результаты (динамические)

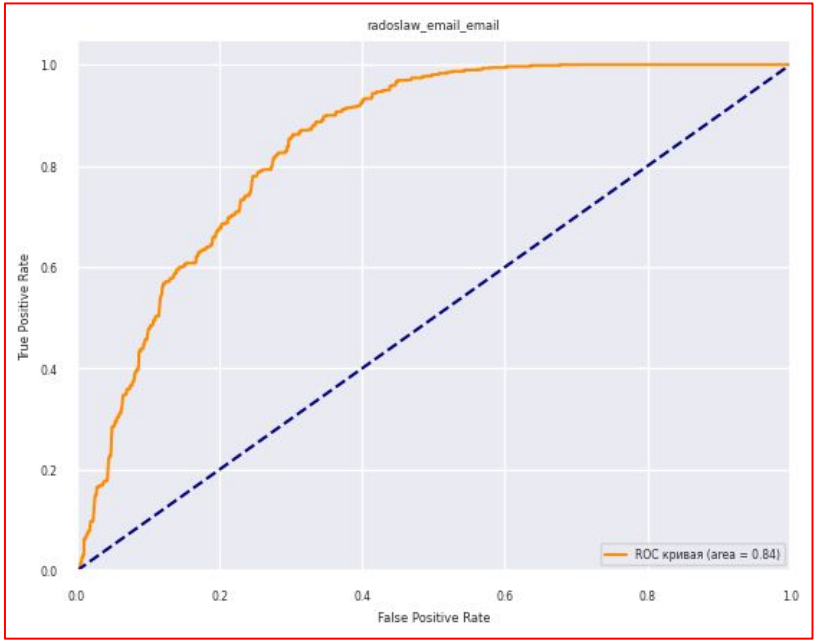




email-Eu-core-temporal.txt
0.7590081111111111

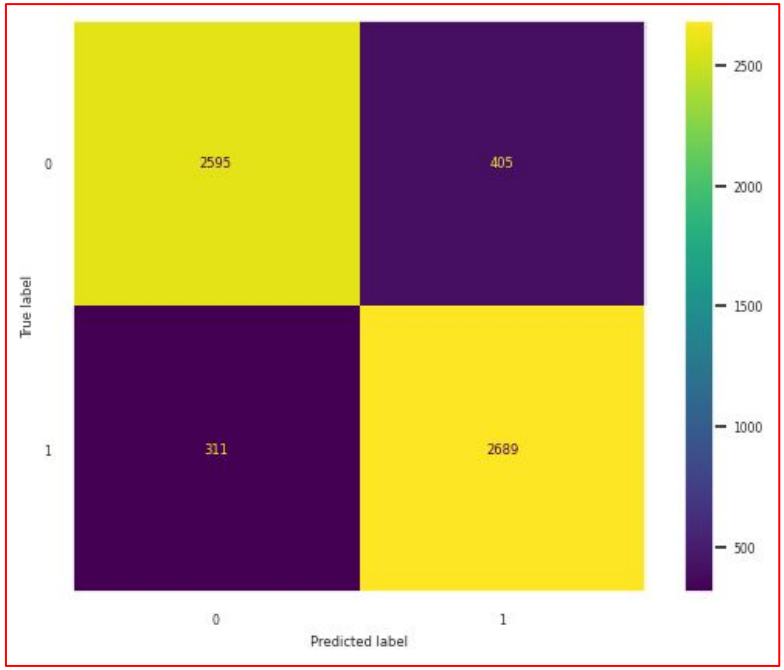
Результаты (динамические)

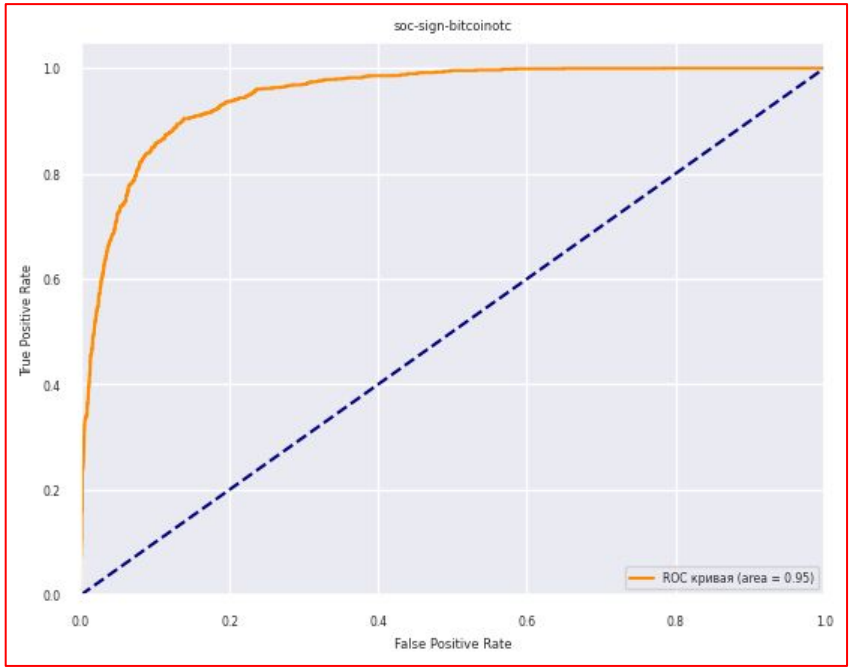





```
radoslaw_email_email
0.8440225
```

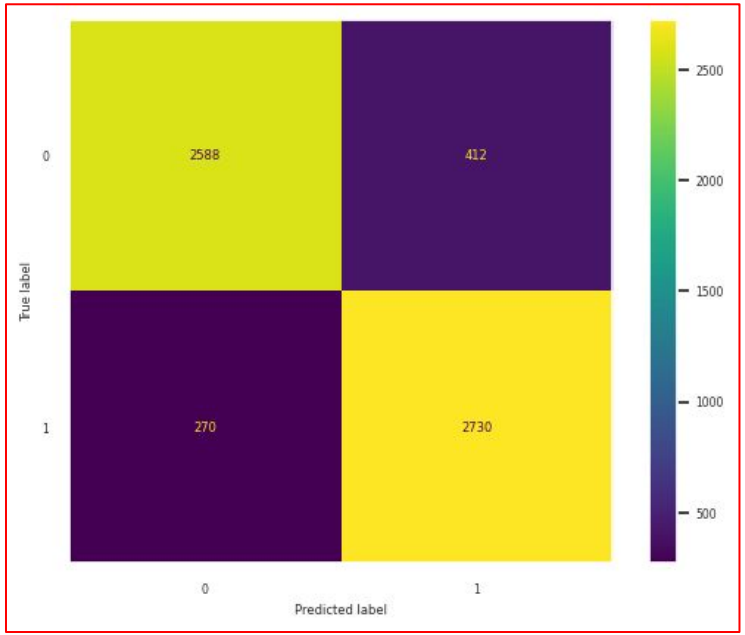
Результаты (динамические)

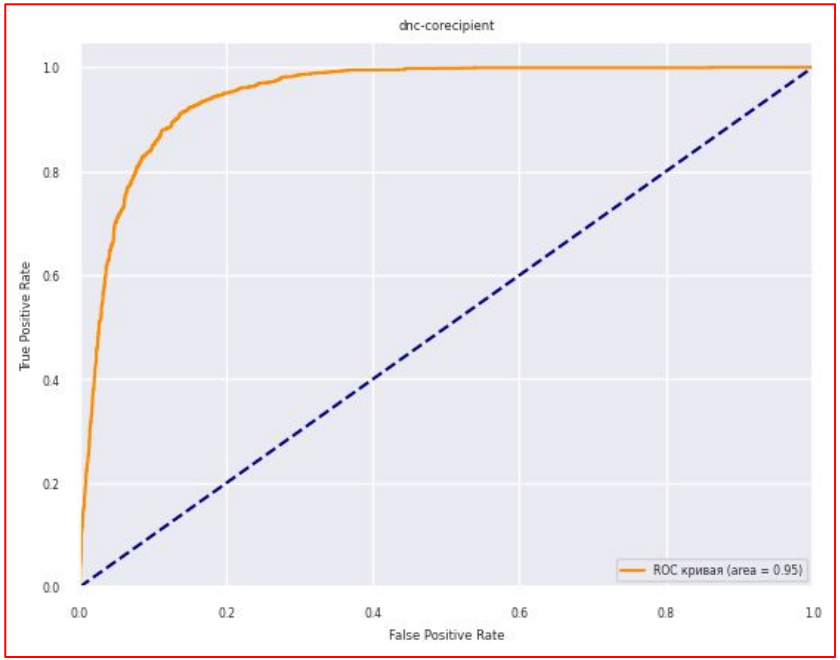




soc-sign-bitcoinotc
0.9488447777777779

Результаты (динамические)





dnc-corecipient
0.9481207222222222

Спасибо за внимание :)