В домашнем задании №4 (предсказание вероятности того, что комментарий “токсичный”) я попробовал 3 архитектуры:

- NBSVM – Naïve Bayes (NB) and Support Vector Machine (SVM)
- LSTM
- RoBERTa

Рассмотрим каждую из них.

1. NBSVM (основная статья – https://nlp.stanford.edu/pubs/sidaw12_simple_sentiment.pdf)

NBSVM – это подход в задаче классификации текстов, который использует линейную модель (SVM или логистическую регрессию) с Байесовскими вероятностями (признаки, полученные на основе текстов (в моем случае TFIDF), умножаются на ‘log-ratio’ (далее показано, что это). NBSVM является очень хорошим бейзлайном.

Теория:

Для начала мы хотим знать вероятность классификации данного предложения: P(Class|Sentence). Так как у нас есть только два класса 0,1 мы можем определить класс, разделив их:

$$result = \frac{P(C=1|S)}{P(C=0|S)}$$

Проблема заключается в том, как получить P(C|S). Согласно теореме Байеса:

$$P(C=1|S) = \frac{P(S|C=1)P(C=1)}{P(S)}, P(C=0|S) = \frac{P(S|C=0)P(C=0)}{P(S)}$$

тогда, $$result = \frac{P(S|C=1)}{P(S|C=0)}\frac{P(C=1)}{P(C=0)}$$, где P(C=1)/P(C=0) – константа.

Для P(S|C=1) мы допускаем предположение, что каждое слово появляется независимо, тогда:

$$P(S|C=1) = P(w_1|C)P(w_2|C)P(w_3|C)\dots P(w_n|C)$$

Таким образом, все, что нам здесь нужно – это P(w|C=1) и P(w|C=0) для всех слов. Для каждого P (S|C) мы можем просто умножить вероятности слов вместе.

Следовательно,

$$result = \frac{\prod_{i=0}^{n} P(w_i|C=1)}{\prod_{i=0}^{n} P(w_i|C=0)}\frac{P(C=1)}{P(C=0)}$$

Определим log-ratio, взяв логарифм из результата:

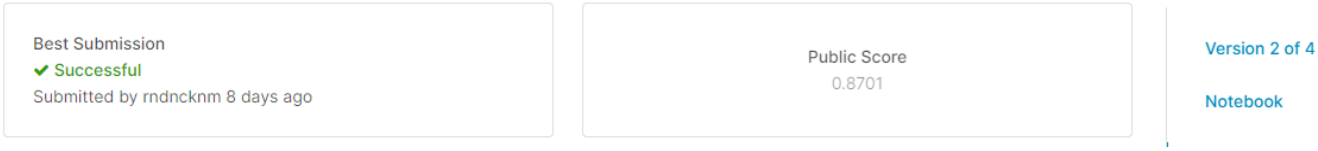
$$r = \log\frac{ratio\ of\ word\ w\ in\ class\ 1}{ratio\ of\ word\ w\ in\ class\ 0} = \log\frac{\frac{\boldsymbol{p}}{||\boldsymbol{p}||}}{\frac{\boldsymbol{q}}{||\boldsymbol{q}||}}$$

Практика:

В python, где x – матрица признаков (TFIDF), а y-таргет, p равно alpha+x[y==1].sum(0) и ||p|| равно alpha+(y==1).sum(). Соответственно, q = alpha+x[y==0].sum(0), а ||q|| = alpha+(y==0).sum(). В нашем случае 0 – не токсичный комментарий, 1 – токсичный. В решении alpha=1.

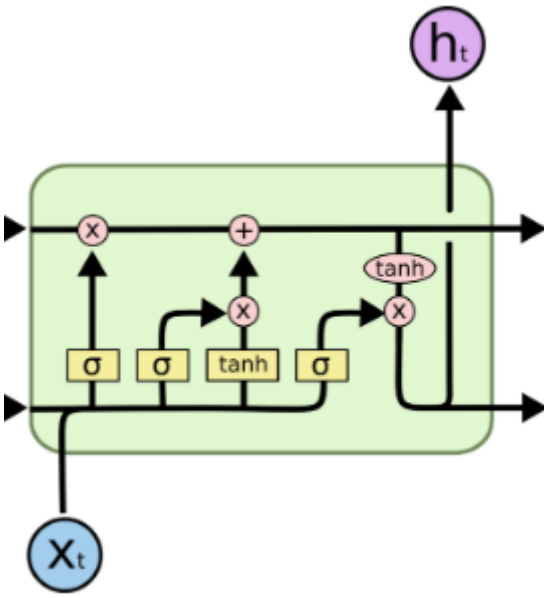
In [13]:
```
def pr(y_i, y):
    p = x[y==y_i].sum(0)
    return (p+1) / ((y==y_i).sum()+1)
```

Результат NBSVM:



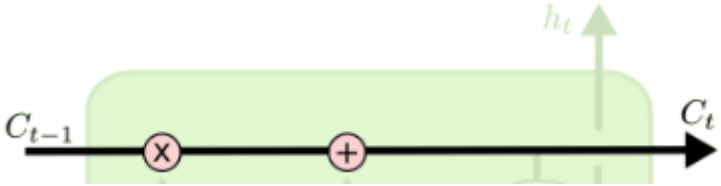
2. LSTM

LSTM – особая разновидность архитектуры рекуррентных нейронных сетей, способная к обучению долговременным зависимостям.



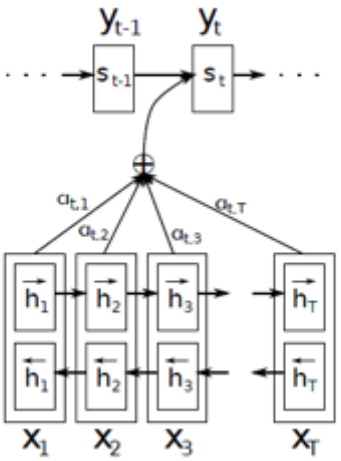
Ключевой компонент LSTM – это состояние ячейки (cell state) – горизонтальная линия, проходящая по верхней части схемы.

Состояние ячейки напоминает конвейерную ленту. Она проходит напрямую через всю цепочку, участвуя лишь в нескольких линейных преобразованиях. Информация может легко течь по ней, не подвергаясь изменениям.



Тем не менее, LSTM может удалять информацию из состояния ячейки; этот процесс регулируется структурами, называемыми фильтрами (gates). Фильтры позволяют пропускать информацию на основании некоторых условий. Сигмоидальный слой возвращает числа от нуля до единицы, которые обозначают, какую долю каждого блока информации следует пропустить дальше по сети. Ноль в данном случае означает “не пропускать ничего”, единица – “пропустить все”.

Я использовал архитектуру Bidirectional LSTM with Attention. Attention используется для того, чтобы обратить внимание на определенные слова во входной последовательности для каждого слова в выходной последовательности.



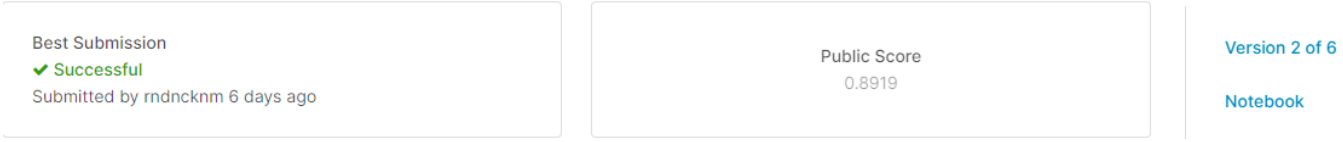
Это схема модели Attention, показанная в статье https://arxiv.org/pdf/1409.0473.pdf. Основная идея: использовать все скрытые состояния. Векторы h1, h2, h3… – это представление количества слов (T) во входном предложении.

$$h_j = \left[\overrightarrow{h}_j^\top ; \overleftarrow{h}_j^\top \right]^\top$$

В простой модели кодера и декодера в качестве вектора контекста использовалось только последнее состояние кодера LSTM (в данном случае hT). В нашем случае для каждого элемента вектора контекста считается свой вес.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j .$$

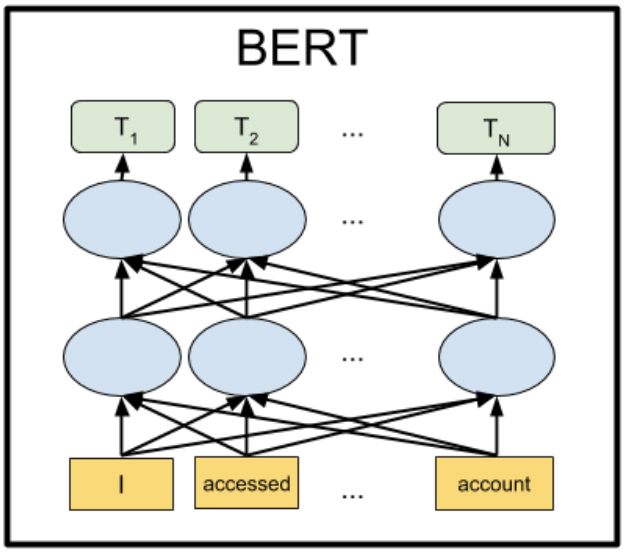
Результат LSTM:



Заметим, что результат лучше, чем у NBSVM, но все еще не тот, который мы хотели.

3. RoBERTa (основная статья - https://arxiv.org/pdf/1907.11692.pdf)

Для начала вспомним, что такое BERT. BERT – это двунаправленная мультиязычная модель с transformer-архитектурой.



Традиционные контекстно-свободные модели (такие как word2vec или GloVe) генерируют одно вложение представления слова для каждого слова в словаре, что означает слово “правильно” будет иметь такое же представление без контекста в “*Я уверен, что я прав*” а также “*Поверните направо”*. В BERT используют две стратегии:

1) **Модель языка маски (MLM)**- маскируя некоторые слова во входных данных, а затем обуславливая каждое слово двунаправленно, чтобы предсказать замаскированные слова. Перед вводом последовательностей слов в BERT 15% слов в каждой последовательности заменяются токеном [MASK]. Затем модель пытается предсказать исходное значение маскированных слов на основе контекста, предоставленного другими немаскированными словами в последовательности.

Input: The $[MASK]_1$ is not working. It’s unable to $[MASK]_2$

Labels: $[MASK]_1$ = computer; $[MASK]_2$ = start.

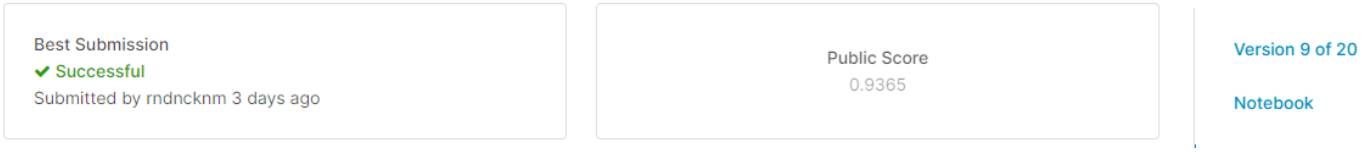
2) Предсказание следующего предложения (NSP), где BERT учится моделировать отношения между предложениями.

RoBERTa - является модификацией BERT с улучшенной методологией обучения, на 1000% больше данных и вычислительной мощности.

Чтобы улучшить процедуру обучения, RoBERTa удаляет задачу предсказания следующего предложения (NSP) из предварительной подготовки Берта и вводит динамическую маскировку, чтобы замаскированный токен менялся в течение периодов обучения. Кроме того, использует батчи большего размера.

В ноутбуке была использована xlm-roberta-large, обученная на 2.5 ТБ данных на 100 языках.

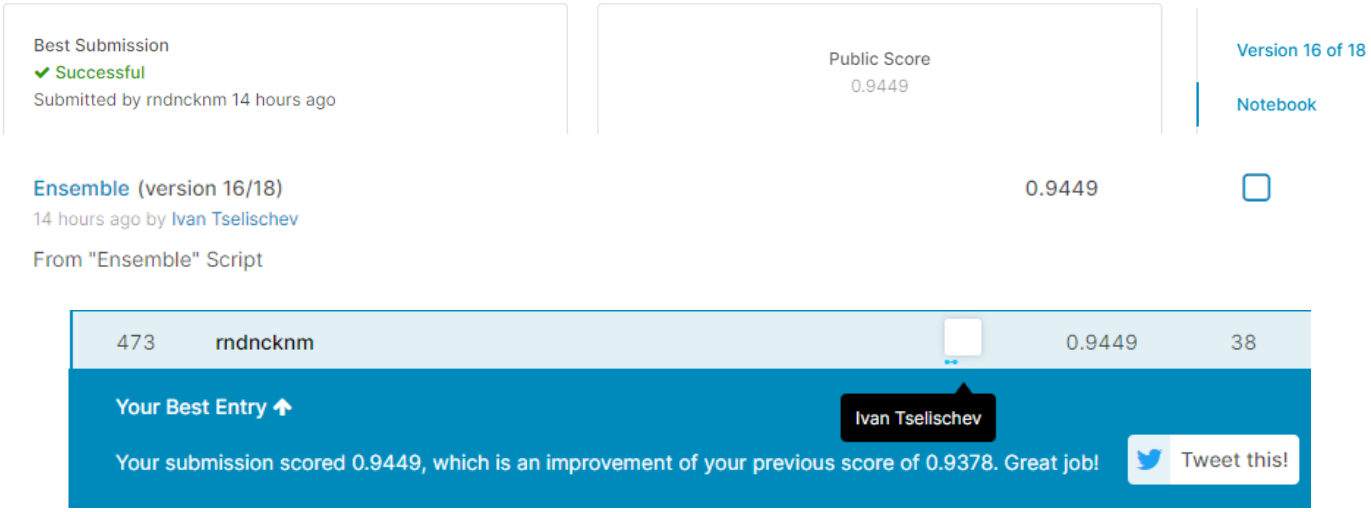
Такая мощная модель, естественно, перебила результаты предыдущих, ее результат:



Но это все еще не 0.94.

4. Ensemble

Для финального сабмита заэнсемблим решения, полученные в предыдущих пунктах (при этом возьмем две версии RoBERTa с разными гиперпараметрами), простой взвешенной суммой с коэффициентами 0.05 для NBSVM и LSTM и 0.6 0.3 для двух версий RoBERTa. По количеству версий на картинке можно понять, что подобрать веса для энсембла случайным образом не так просто (особенно с ограничением в 5 сабмитов в день).



5. Об обучении

Также стоит отметить, что LSTM и RoBERTa используют одинаковый подход к обучению, состоящий в разделении обучения на два этапа: на первом учиться на большом трейне с английскими комментариями, на втором - на маленькой валидации с разными языками. Этот метод (именно для этого соревнования) был предложен (вроде) пользователем “xhulu”: « Моя гипотеза заключается в том, что этот двухэтапный процесс улучшает способность модели сначала изучить основную структуру токсичных комментариев (путем обучения на 400 тыс. английских комментариев), а затем адаптировать эти изученные структуры для выделения других языков из набора проверки (турецкий, итальянский, португальский и т. д.).» (переведено с английского).