НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ВЫСШАЯ ШКОЛА ЭКОНОМИКИ

Факультет математики

Ковалев Евгений, Новиков Лев, Сухарев Иван

Отчет по проекту

Quora Question Pairs

3 курс, майнор «Интеллектуальный анализ данных», группа ИАД-4

Москва, 2017 г.

1. Описание задачи

Для получения практических навыков применения методов машинного обучения в работе с текстами было принято участие в соревновании «Quora Question Pairs» по поиску семантических дубликатов, которое проводится на Kaggle — онлайн-платформе для проведения конкурсов по машинному обучению.

Quora — ресурс, устроенный по типу вопрос/ответ, на котором можно узнать интересующую информацию самой разной тематики. Задача заключается в том, чтобы в предоставленных парах вопросов выявлять содержащие в себе одинаковые по смыслу.

Обучающая выборка состоит из 404290 объектов (пар вопросов), 5 признаков (id — идентификатор пары, qid1 и qid2 — идентификаторы вопросов, question1 и question2 — тексты вопросов) и целевой переменной is_duplicate, принимающей значение 1, если вопросы в паре одинаковы по смыслу, и 0 — иначе. Тестовая выборка состоит из 2345796 объектов и 3 признаков (test_id — идентификатор пары, question1 и question2 — тексты вопросов). Метрика качества — Log Loss:

$$\text{Log Loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log\left(p_{ij}\right),$$

где N — количество объектов, M — количество возможных классов, y_{ij} — коэффициент, который обращается в 1, если объект i действительно находится в классе j, и в 0 — иначе, p_{ij} — спрогнозированная алгоритмом вероятность отношения объекта i к классу j.

Понятно, что признаки-идентификаторы почти не несут в себе никакой полезной в решении задачи информации, однако на основе текстов вопросов можно извлечь относительно много признаков, которые будут определять их близость по смыслу.

2. Извлечение признаков

Из данных были извлечены следующие признаки:

- len_q1, len_q2 — длины первого и второго вопросов соответственно в символах
- diff_len — разница между длинами первого и второго вопросов в символах
- len_char_q1, len_char_q2 — длины первого и второго вопросов соответственно в символах без учета пробелов
- diff_len_char — разница между длинами первого и второго вопросов в символах без учета пробелов
- len_word_q1, len_word_q2 — длины первого и второго вопросов соответственно в словах
- diff_len_word — разница между длинами первого и второго вопросов в словах

- common_words — количество общих слов в вопросах

Следующие признаки были сгенерированы с помощью библиотеки, которая позволяет оценивать близость строк друг с другом по смыслу — Fuzzy Wuzzy [1]:

- fuzz_qratio:

$$\text{fuzz\_qratio}(q_1, q_2) = \left\lfloor \frac{2m}{s} \times 100 + \frac{1}{2} \right\rfloor,$$

  где q_1, q_2 — вопросы, m — максимальная длина общей подстроки в вопросах в символах, s — общее количество символов в вопросах

- fuzz_token_sort_ratio — то же, что и fuzz_qratio, только с предварительной токенизацией вопросов и сортировкой по словам

- fuzz_token_set_ratio — максимальное значение fuzz_qratio для всех пар из множества $\{t_0, t_1, t_2\}$, где t_0 — отсортированное в лексикографическом порядке пересечение по словам вопросов после токенизации, t_1 и t_2 — строки, образованные добавлением к t_0 отсортированного в лексикографическом порядке остатка вопроса после токенизации и удаления t_0

- fuzz_partial_ratio:

$$\text{fuzz\_partial\_ratio}(q_1, q_2) = \max_{y \in q_{\max}} \text{fuzz\_qratio}(q_{\min}, y),$$

  где $q_{\min}$ и $q_{\max}$ — наименьший и наибольший по длине в символах вопросы соответственно, y — подстрока из последовательных символов в вопросе $q_{\max}$, имеющую ту же длину в символах, что и $q_{\min}$

- fuzz_partial_token_sort_ratio — аналогично fuzz_token_sort_ratio для fuzz_partial_ratio

- fuzz_partial_token_set_ratio — аналогично fuzz_token_set_ratio для fuzz_partial_ratio

- fuzz_wratio — максимальное значение fuzz_qratio, 0.9×fuzz_token_sort_ratio и 0.9×fuzz_token_set_ratio в случае, если длины вопросов в символах различаются не более, чем в 1.5 раза, и максимальное значение fuzz_partial_ratio, 0.9×fuzz_partial_token_sort_ratio и 0.9×fuzz_partial_token_set_ratio, домноженное на 0.6, если различаются более чем в 8 раз, и на 0.9 — иначе

При извлечении некоторых признаков вопросы были представлены в векторном виде: произведена токенизация, выброшены символы, не являющиеся буквами алфавита, а также стоп-слова. Затем с помощью модели word2vec, обученной на корпусе новостей Google, вмещающем в себя около 3 миллиардов уникальных слов, каждому слову был сопоставлен вектор. Вопросу сопоставлялся нормализованный вектор, полученный покоординатной суммой векторов-слов, входящих в него.

- wmd — расстояние Word Mover's Distance между вопросами в векторном представлении [2]:

$$\text{wmd}(\vec{v}, \vec{w}) = \min_{T \geq 0} \sum_{i=1}^{n} \sum_{j=1}^{n} T_{ij} c(x_i, x_j),$$

где $c(x_i, x_j)$ — евклидово расстояние между словами x_i и x_j, $\vec{v} = (v_1, \ldots, v_n)$, $\vec{w} = (w_1, \ldots, w_n)$, а $T \in \mathbb{R}^{n \times n}$ — матрица «потока», где $T_{ij} \geq 0$ отражает, «сколько» слова x_i перешло в x_j при переводе $\vec{v}$ в $\vec{w}$, и

$$\begin{cases} \sum_{j=1}^{n} T_{ij} = v_i \\ \sum_{i=1}^{n} T_{ij} = w_j \end{cases}$$

- norm_wmd — то же самое, что и wmd, только с предварительной нормализацией всех векторов во избежание экстремальных значений при использовании евклидовой метрики

- cosine_distance — косинусная близость:

$$\text{cosine\_distance}\,(\vec{v}, \vec{w}) = \frac{\vec{v} \cdot \vec{w}}{\|\vec{v}\|_2 \|\vec{w}\|_2} = \frac{\sum_{k=1}^{n} v_k \times w_k}{\sqrt{\sum_{k=1}^{n} v_k^2} \sqrt{\sum_{k=1}^{n} w_k^2}}$$

- cityblock_distance — манхэттенское расстояние:

$$\text{cityblock\_distance}\,(\vec{v}, \vec{w}) = \|\vec{v} - \vec{w}\|_1 = \sum_{k=1}^{n} |v_k - w_k|$$

- jaccard_distance — расстояние Жаккара:

$$\text{jaccard\_distance}\,(\vec{v}, \vec{w}) = \frac{\sum_{k=1}^{n} \min(v_k, w_k)}{\sum_{k=1}^{n} \max(v_k, w_k)}$$

- canberra_distance — расстояние Канберра:

$$\text{canberra\_distance}\,(\vec{v}, \vec{w}) = \sum_{k=1}^{n} \frac{|v_k - w_k|}{|v_k| + |w_k|}$$

- euclidean_distance — расстояние Евклида:

$$\text{euclidean\_distance}\,(\vec{v},\vec{w}) = \|\vec{v}-\vec{w}\|_2 = \sqrt{\sum_{k=1}^{n}(v_k-w_k)^2}$$

- braycurtis_distance — расстояние Брея-Кертиса:

$$\text{braycurtis\_distance}\,(\vec{v},\vec{w}) = \frac{\|\vec{v}-\vec{w}\|_1}{\|\vec{v}+\vec{w}\|_1} = \frac{\sum\limits_{k=1}^{n}|v_k-w_k|}{\sum\limits_{k=1}^{n}|v_k+w_k|}$$

- dice_distance — расстояние Дайса:

$$\text{dice\_distance}\,(\vec{v},\vec{w}) = \frac{2\times\sum\limits_{k=1}^{n}\min(v_k,w_k)}{\sum\limits_{k=1}^{n}(v_k+w_k)}$$

- skew_q1vec, skew_q2vec — коэффициенты асимметрии первого и второго вопросов соответственно в векторном представлении как выборок
- kur_q1vec, kur_q2vec — коэффициенты эксцесса первого и второго вопросов соответственно в векторном представлении как выборок

Вышеупомянутые расстояния были рассчитаны также для модели с предварительной токенизацией и стеммингом, обученной на всем датасете и сопоставляющей каждому вопросу вектор из TF-IDF весов.

- cosine_distance_tfidf — косинусная близость между векторами-вопросами для TF-IDF модели
- cityblock_distance_tfidf — манхэттенское расстояние между векторами-вопросами для TF-IDF модели
- jaccard_distance_tfidf — расстояние Жаккара между векторами-вопросами для TF-IDF модели
- canberra_distance_tfidf — расстояние Канберра между векторами-вопросами для TF-IDF модели
- euclidean_distance_tfidf — расстояние Евклида между векторами-вопросами для TF-IDF модели
- braycurtis_distance_tfidf — расстояние Брея-Кертиса между векторами-вопросами для TF-IDF модели

- dice_distance_tfidf — расстояние Дайса между векторами-вопросами для TF-IDF модели

Были также построены N-граммные модели с предварительными токенизацией и стеммингом, натренированные примерно на 10 миллионах вопросов из корпуса вопросов-ответов с коллаборативной базы знаний Freebase.

- bigram_perplexity_q1, bigram_perplexity_q2 — perplexity первого и второго вопросов соответственно, основанные на биграммной (2-граммной) модели со сглаживанием Лапласа

- trigram_perplexity_q1, trigram_perplexity_q2 — perplexity первого и второго вопросов соответственно, основанные на триграммной (3-граммной) модели со сглаживанием Лапласа

- fourgram_perplexity_q1, fourgram_perplexity_q2 — perplexity первого и второго вопросов соответственно, основанные на 4-граммной модели со сглаживанием Лапласа

3. Пропуски

В исходном датасете пропусков почти не было — в обучающей и тестовой выборках было обнаружено 2 и 4 вопроса без единого символа, соответственно. В процессе извлечения признаков ввиду очищения вопросов от небуквенных символов и стоп-слов образовались пропуски при расчете косинусной близости, расстояний Жаккара, Брея-Кертиса и Дайса. Это связано с тем, что некоторые вопросы после предобработки сокращались до пустой строки, и, соответственно, им присваивался нулевой вектор, что вызывало нули в знаменателе. Также, если после предобработки вопрос не имел слов, знакомых модели word2vec, то расстояние Word Mover's Distance было равно бесконечности.

Количество данных случаев показано в следующей таблице (train — обучающая выборка, test — тестовая).

	train	test
wmd	10894 (2.69%)	126822 (5.41%)
norm_wmd	10894 (2.69%)	126822 (5.41%)
cosine_distance	1799 (0.44%)	37333 (1.59%)
cosine_distance_tfidf	259 (0.06%)	24249 (1.03%)
jaccard_distance	525 (0.13%)	1454 (0.06%)
jaccard_distance_tfidf	78 (0.02%)	429 (0.02%)
braycurtis_distance	525 (0.13%)	1454 (0.06%)
braycurtis_distance_tfidf	78 (0.02%)	429 (0.02%)
dice_distance	522 (0.13%)	1445 (0.06%)
dice_distance_tfidf	78 (0.02%)	429 (0.02%)

Все пропуски (бесконечные значения в случае wmd и norm_wmd) были заполнены средним по соответствующим столбцам.

4. Применение алгоритмов и результаты

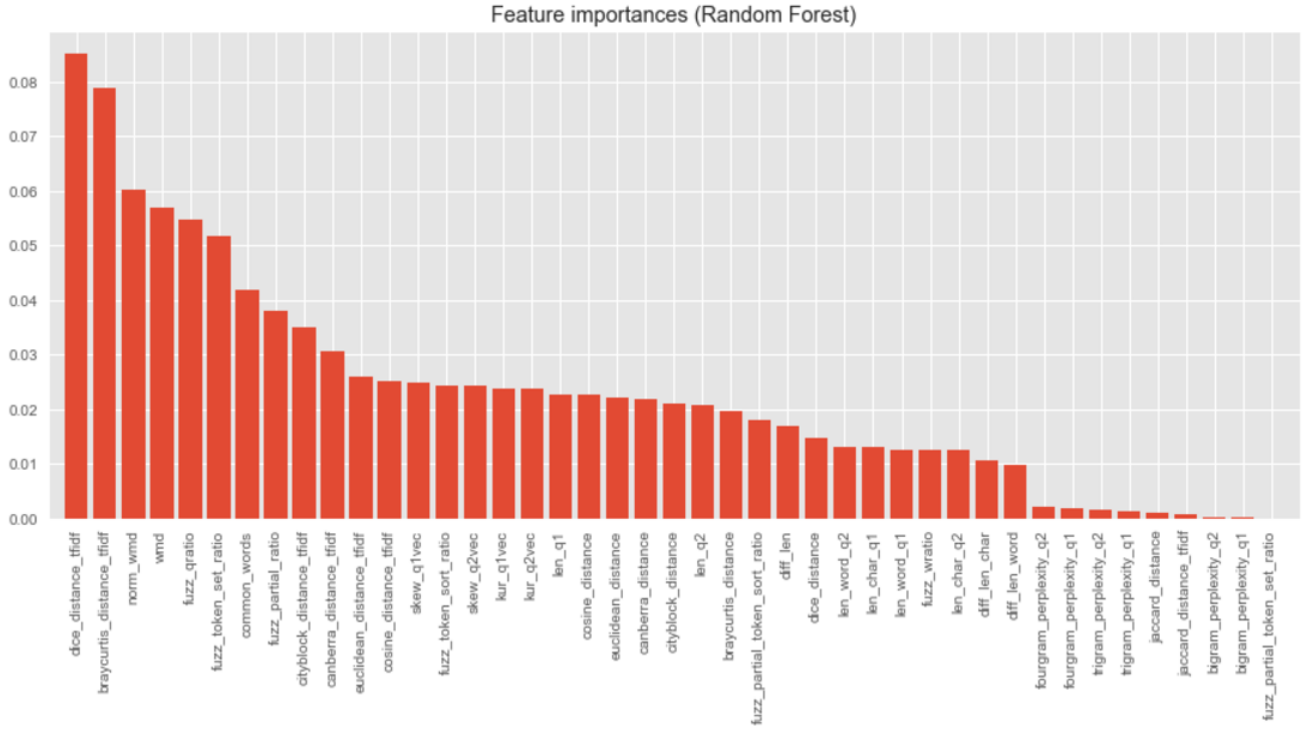
После извлечения признаков и заполнения пропусков вопросы вместе с идентификаторами были исключены из рассмотрения, а оставшиеся данные были отмасштабированы.

Обучающая выборка была случайным образом поделена на две части в отношении 75/25 (обозначим их для удобства как X_{train} и X_{test} соответственно). На X_{train} проводилось обучение, а на X_{test} — тестирование моделей. С помощью кросс-валидации с 3 фолдами подбирались оптимальные гиперпараметры алгоритмов.

Результаты (по метрике Log Loss):

	Cross-validation (best)	X_{test}	test
Random Forest	0.42747	0.42212	0.41258
xgboost	0.15804	0.33744	0.42285
Gradient Boosting	0.46696	0.46674	0.43775
Decision Tree	0.48454	0.48281	0.45307
Logistic Regression	0.52415	0.52287	0.46387
k Nearest Neighbors	0.53039	0.52663	0.53021
Naive Bayes	0.59639	0.69315	6.01888

Итак, случайный лес имеет наилучшее качество на данном датасете из примененных алгоритмов. С помощью встроенных в него функций можно оценить вклад сгенерированных признаков в работу модели:



Как видно из построенной диаграммы, наиболее важные признаки были сгенерированы с помощью методов TF-IDF (расстояние Дайса, расстояние Брея-Кертиса), word2vec и Word Mover's Distance, а также библиотеки FuzzyWuzzy, основанной на расстоянии Левенштейна. Более «незамысловатые» признаки (к примеру, длины вопросов в символах и разницы между ними) в целом были менее информативны и усту-

пали большинству признаков, извлеченных с помощью методов машинного обучения. Однако еще бесполезней оказались признаки, построенные по N-граммным моделям. Скорее всего, это следствие чувствительности качества N-граммных моделей к обучающему корпусу [3, ч. 4.3.1] — вероятно, несмотря на то, что он тоже состоял из вопросов, он сильно отличался по стилю и содержанию от исходного датасета.

5. Заключение

В рамках эксперимента для данного датасета с объектами в текстовом виде (парами вопросов с сервиса Quora) были сгенерированы простейшие признаки, не использующие методы машинного обучения, а также признаки, основанные на векторных представлениях слов с помощью инструмента word2vec и меры TF-IDF и на N-граммных моделях. При предобработке текста использовались токенизация, стемминг и удаление стоп-слов. Образованные в ходе этого процесса пропуски и бесконечные значения были заполнены средним по соответствующим столбцам. После подбора необходимых гиперпараметров с помощью кросс-валидации на полученном датасете было проведено обучение и тестирование нескольких алгоритмов, выявлен лучший (случайный лес). В конечном итоге были определены наиболее и наименее информативные признаки. Простейшие признаки, не использующие методы машинного обучения, а также признаки, построенные по N-граммным моделям, в целом показали себя хуже, чем остальные.

Список литературы

[1] Cohen Adam. FuzzyWuzzy: Fuzzy String Matching in Python. // *July 8th, 2011.*

[2] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, Kilian Q. Weinberger. From Word Embeddings To Document Distances. // *Washington University in St. Louis, 1 Brookings Dr., St. Louis, MO 63130.*

[3] Daniel Jurafsky, James H. Martin. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 2nd Edition. // *Prentice-Hall. 2009.*