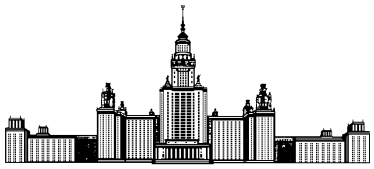
Московский государственный университет имени М. В. Ломоносова



Факультет Вычислительной математики и Кибернетики
Кафедра Математических Методов Прогнозирования

Математические методы анализа текстов

Отчет по практическому заданию №4.

«Векторные представления слов, тематическое моделирование. Анализ тональности.»

Выполнил:
магистр 1 курса 517 группы
Таскынов Ануар

Москва, 2018

Содержание.

1 Первая часть **2**
1.1 Используемые модели 2
1.2 Эксперименты 4

2 Вторая часть **5**

3 Третья часть **7**

4 Выводы **7**

1 Первая часть

1.1 Используемые модели

В качестве моделей для векторных представлений были использованы:

1. TF-IDF;
2. word2vec (CBOW, skip-gram);
3. GloVe;
4. doc2vec;
5. LDA;
6. FastText классификация.

Самое простое представление для документов может быть получено с помощью TF-IDF. TF-IDF — это статистическая мера, используемая для оценки важности слова в контексте документа, являющегося частью коллекции документов или корпуса. Формула выглядит следующим образом:

$$\mathrm{tf}(t,d) = \frac{n_t}{\sum_k n_k},$$

где n_t — число вхождений слова t в документ, суммирование ведётся по словам документа.

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d_i \in D | t \in d_i\}|},$$

где $|D|$ — число документов в коллекции, $|\{d_i \in D | t \in d_i\}|$ — число документов из коллекции D, в которых встречается t.

$$\text{tf-idf}(t,d,D) = \mathrm{tf}(t,d) \times \mathrm{idf}(t,D).$$

Однако данное представление крайне разреженно и требует хранения матрицы документы $\times$ размер словаря.

Следующие класс моделей это word2vec. Идея word2vec состоит в том, чтобы научиться предсказывать слово из его контекста и наоборот. Будет приведено описание модели CBOW (continuous bag of words):

1. вход сети — это one-hot представление слова (размерность V).
2. скрытый слой — это фактически и есть матрица W векторных представлений слов, $W \in \mathbb{R}^{V \times d}$, где d — это размерность векторного представления.
3. при вычислении выхода скрытого слоя мы берем просто среднее всех входных векторов;
4. на выходе получается некоторая оценка u для каждого слова в словаре; затем апостериорное распределение модели вычисляется с помощью обычного softmax:

$$\hat{p}(i|c_1,\ldots,c_n) = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})};$$

5. функция потерь:

$$L = -\log p(i|c_1, \ldots, c_n) = -u_j + \log \sum_{j'=1}^{V} \exp(u_{j'}),$$

то есть апостериорное распределение должно быть как можно ближе к распределению данных.

Схема CBOW приведена на Рис. 1

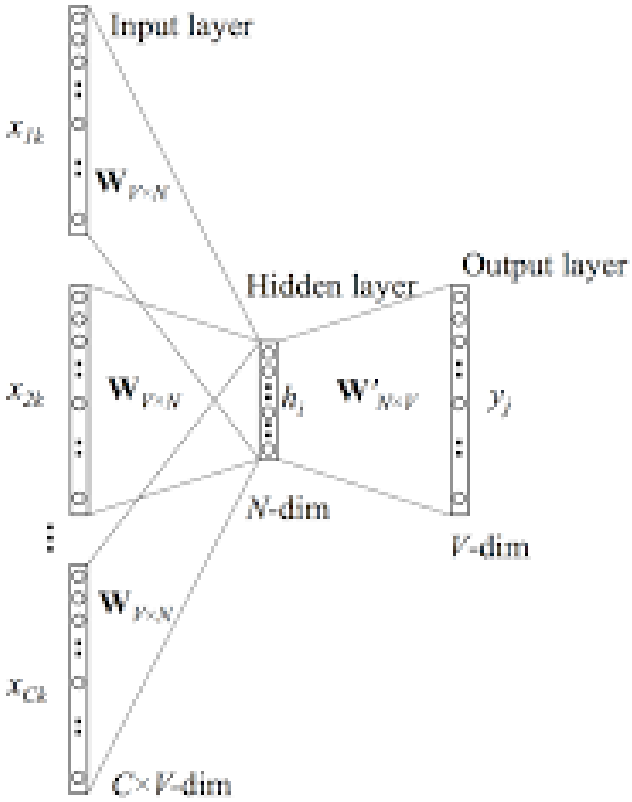


Схема CBOW.

Модель GloVe чем-то схожа на LDA. Введём обозначения: $X \in \mathbb{R}^{V\times V}$ — матрица совместной встречаемости слов (X_{ij} показывает то, сколько раз в нашем корпусе слово i встретилось вместе со словом j), $X_i = \sum_j X_{ij}$ — общее число раз, которое какое-то другое слово встречается в контексте слова i.

$$p_{ij} = p(j|i) = \frac{X_{ij}}{X_i} = \frac{X_{ij}}{\sum_k X_{ik}},$$

p_{ij} — это вероятность того, что слово j встретится в контексте слова i. Модель GloVe моделирует на сами вероятности, а их отношения:

$$F(w_i, w_j|\hat{w}_k) = \frac{p_{ik}}{p_{jk}},$$

где w_i, w_j — вектора слов i, j в пространстве представлений, а $\hat{w}_k$ — это вектор контекста. Мы хотим отразить тот факт, что соотношение между словами i и j мы сейчас приближаем не вообще, а в контексте слова k. Также наша функция должна как-то выражать простые отношения между словами, поэтому в GloVe ограничили набор функций:

$$F\big((w_i - w_j)^T \hat{w}_k\big) = \frac{p_{ik}}{p_{jk}}.$$

Данное ограничение позволяет GloVe обучить именно линейные соотношения между векторами. Далее, создатели ввели еще одно ограничение в силу симметрии матрицы X:

$$F\big((w_i - w_j)^T \hat{w}_k\big) = \frac{F\big(w_i^T \hat{w}_k\big)}{F\big(w_j^T \hat{w}_k\big)} = \frac{p_{ik}}{p_{jk}}.$$

Таким образом функция F — это экспонента:

$$w_i^T \hat{w}_k = \log(p_{ik}) = \log\big(X_{ik}\big) - \log\big(X_i\big).$$

Чтобы векторы слов и контекстов стали симметричными нужно добавить еще по слагаемым b_i, $\hat{b}_k$:

$$w_i^T \hat{w}_k + b_i + \hat{b}_k = \log\big(X_{ik}\big).$$

Функция потерь выглядит следующим образом:

$$L = \sum_{i,j=1}^{V} f(X_{ij})\Big(w_i^T \hat{w}_j + b_i + \hat{b}_j - \log X_{ij}\Big),$$

где f — это функция, которая не присваивает частым совместным встречаемостям слишком больших весов, а также $f(0) = 0$.

Следующая модель doc2vec — это расширение моделей word2vec. Здесь также имеются две разновидности: PV-DM, PV-DBOW. В модели PV-DM вектор, представляющий документ (например, официальный документ, письмо) определяется как еще один дополнительный вектор весов, который подаётся вместе со словами контекста. А в модели PV-DBOW слова контекста полностью игнорируются, и модель использует только вектор документа, чтобы предсказаь слова из окна.

Следующая модель — LDA. Данная модель позволяет строить для документа тематический профиль, который логично описывает структуру документа.

Идея FastText довольна проста: используем вместо one-hot представлений слов one-hot представления триграмм символов. Это значит, что мы вводим векторные представления для триграмм символов, затем раскладываем каждое слово как сумму векторов его триграмм, а на этих суммах строим обычные конструкции CBOW, skip-gram. Полученные модели обучить проще и быстрее, они не такие большие, и также происходит неявный учёт морфологии. Также вместо линейной модели используется линейная модель с ограничением на ранг для классификации.

1.2 Эксперименты

Был дан датасет EUR-lex, содержащий документы разных европейских языков. Также был дан текстовый файл, содержащий классы этих документов, причем у каждого документа может быть несколько классов.

Так как датасет изначально лемматизирован, то в качестве предобработки были удалены стоп-слова, а также документы, не имеющие классов.

Так как почти все модели обучаются без учителя, то они были обучены на всём датасете. Далее датасет был разделен случайным образом на обучающую и тестовую выборки в соотношении 9:1.

В качестве алгоритма классификации взят метод ближайших соседей с косинусной метрикой. Все векторные представления обучались с размерностью 100 на выходе. Так как такие модели как word2vec, GloVe не выучивают векторное представление для всего документа было сделано следующее: получены векторные представления для слов из документа, которые были затем усреднены.

Модели сравнивались следующим образом:

1. для каждого документа будем рассматривать вектор, длиной в число меток классов, где 1 означает верный класс, а 0 — нет;

2. метод классификации предсказывает вероятность того, что документ принадлежит данному классу;

3. для данных двух векторов высчитывается AUC-ROC и AUC-PR;

4. итоговый AUC-ROC и AUC-PR это усреднение по всем документам данной метрики.

Результаты экспериментов приведены в Табл. 1, 2. В Табл. 1 указаны наилучшие модели по метрике AUC ROC, в Табл. 2 по метрике AUC PR.

Models	k	AUC ROC	AUC PR
TF-IDF	100	0.956	0.435
CBOW	100	0.938	0.346
Skip-gram	10	0.942	0.361
DBOW	100	0.955	0.419
LDA	100	0.855	0.192
Glove	100	0.934	0.339
FastText	–	**0.977**	**0.563**

Таблица 1: Сравнение методов для векторных представлений слов/документов, k — число соседей.

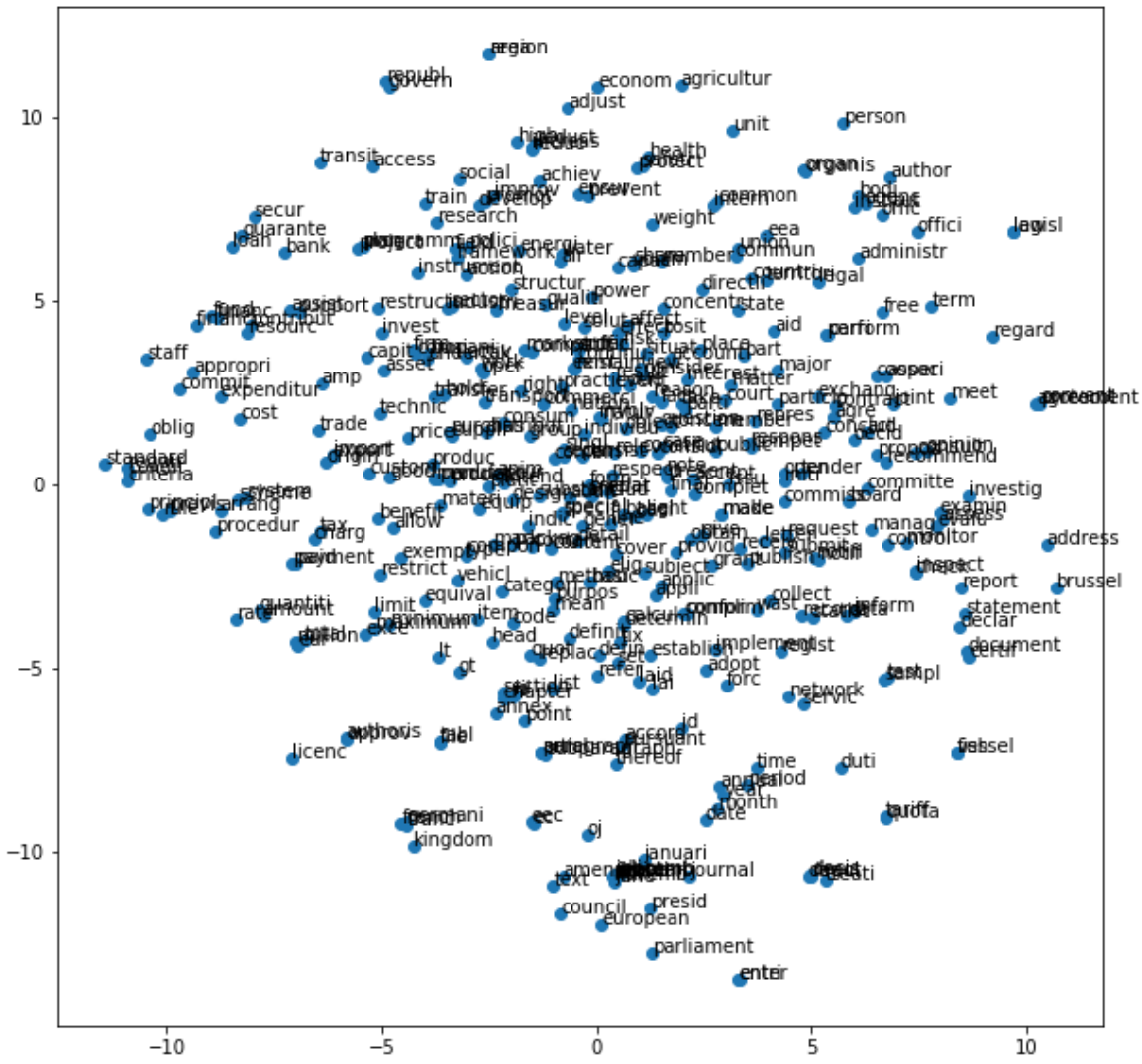
Models	k	AUC ROC	AUC PR
TF-IDF	10	0.894	0.497
CBOW	10	0.86	0.4096
Skip-gram	100	0.867	0.425
DBOW	10	0.887	0.479
LDA	50	0.819	0.199
Glove	10	0.857	0.3996
FastText	–	**0.977**	**0.563**

Таблица 2: Сравнение методов для векторных представлений слов/документов, k — число соседей.

По результатам экспериментов видно, что FastText из-за учёта морфологии лучше срабатывает на данном датасете. Можно обратить внимание на то, что при большем количестве соседей увеличивается метрика AUC ROC, а для хорошего качества по метрике AUC PR достаточно 10 соседей. LDA хуже всего отработал на данных датасетах (возможно, потому что в качестве метрики в методе ближайших соседей нужно было использовать что-то похожее на kl-дивергенцию). Также можно заметить, что несмотря на простоту модели TF-IDF довольно неплохо справляется с задачей.

2 Вторая часть

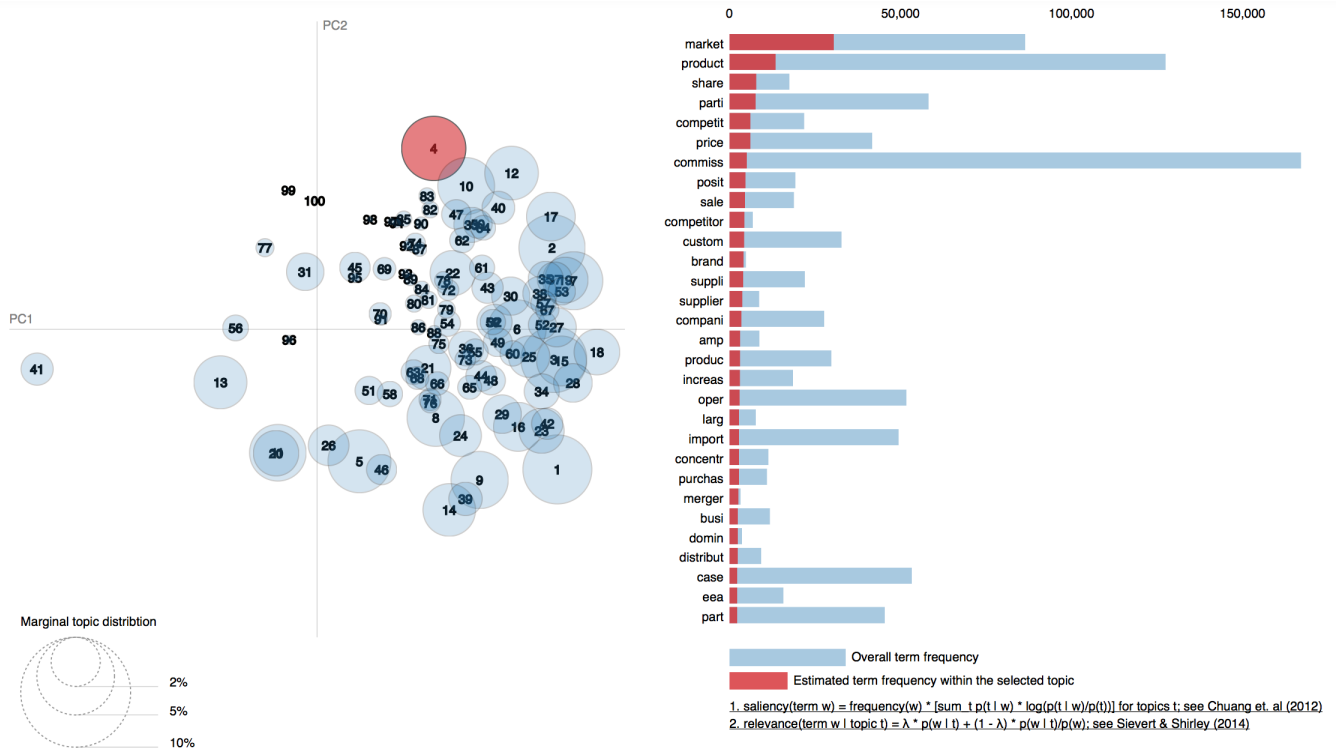
В данной части необходимо было визуализировать вектора, полученные с предыдущего пункта. В качестве модели для векторных представлений был выбран word2vec. Визуализация происходила с помощью алгоритма t-SNE (реализация Ульянова Дмитрия). Результаты на Рис. 2.



Визуализация векторных представлений.

Получилась довольно забавная картинка. Скорее всего из-за нехватки данных и разных языков не прослеживается близость между одинаковыми словами.

Нужно было визуализировать работу LDA. Для этого была выбрана библиотека LDAvis. Результаты на Рис. 3. Видно, что многие тематики пересекаются друг с другом, некоторые тематики строго входят в другие.



Визуализация работы LDA на 100 тематиках.

3 Третья часть

В данной части необходимо было решить задачу анализа тональности, то есть решить бинарную классификацию. Датасет — твиты из twitter. В качестве предобработки данных были удалены повторяющиеся символы. Все знаки припинания остались, так как сильно характеризуют тональность. Стоп-слова также не удалены по тем же причинам.

В качестве классификатора использовалась линейная модель логистической регрессии. В качестве векторных представлений TF-IDF на символах с биграммами и триграммами, а также skip-gram модель word2vec. Результаты приведены на Табл. 3

Models	AUC ROC	AUC PR	accuracy
TF-IDF	**0.863**	**0.864**	**0.782**
Skip-gram	0.847	0.846	0.768

Таблица 3: Сравнение методов на задаче анализа тональности.

Здесь также видно, что TF-IDF лучше всего описывает твит. Однако не был использован FastText (не хватило памяти на ноутбуке).

4 Выводы

В ходе выполнения задания я был ознакомлен с тем, как обучать различные векторные представления слов/документов. Также было выяснено, что FastText довольно мощная модель (хотя была проверена только на одной задаче). Выяснено, что tf-idf очень хорошо описывает документ, но для представления приходится пожертвовать оперативной памятью.