Национальный исследовательский университет

«Московский энергетический институт»

Кафедра Прикладной математики и искусственного интеллекта

Выпускная работа на тему: «Итеративный алгоритм классификации текстов»

Выполнил: студент гр. А-13-16 Вагнер А.С.

Научный руководитель: к.т.н. Кружилов И.С.

Москва, 2020 г.

Цель работы и задачи:

Цель: Разработка алгоритма для создания тематической модели. Использовать латентное размещение Дирихле для достижения цели.

Задачи:

1) Изучить представление текстовой информации в программах

2) Изучить вероятностные тематические модели

3) Разработать классификатор текстов

4) Проанализировать результаты.

Машинное обучение

Машинное обучение – это множество математических, статистических и вычислительных методов для разработки алгоритмов, способных решить задачу не прямым способом, а на основе поиска закономерностей в разнообразных входных данных.

Типы машинного обучения:

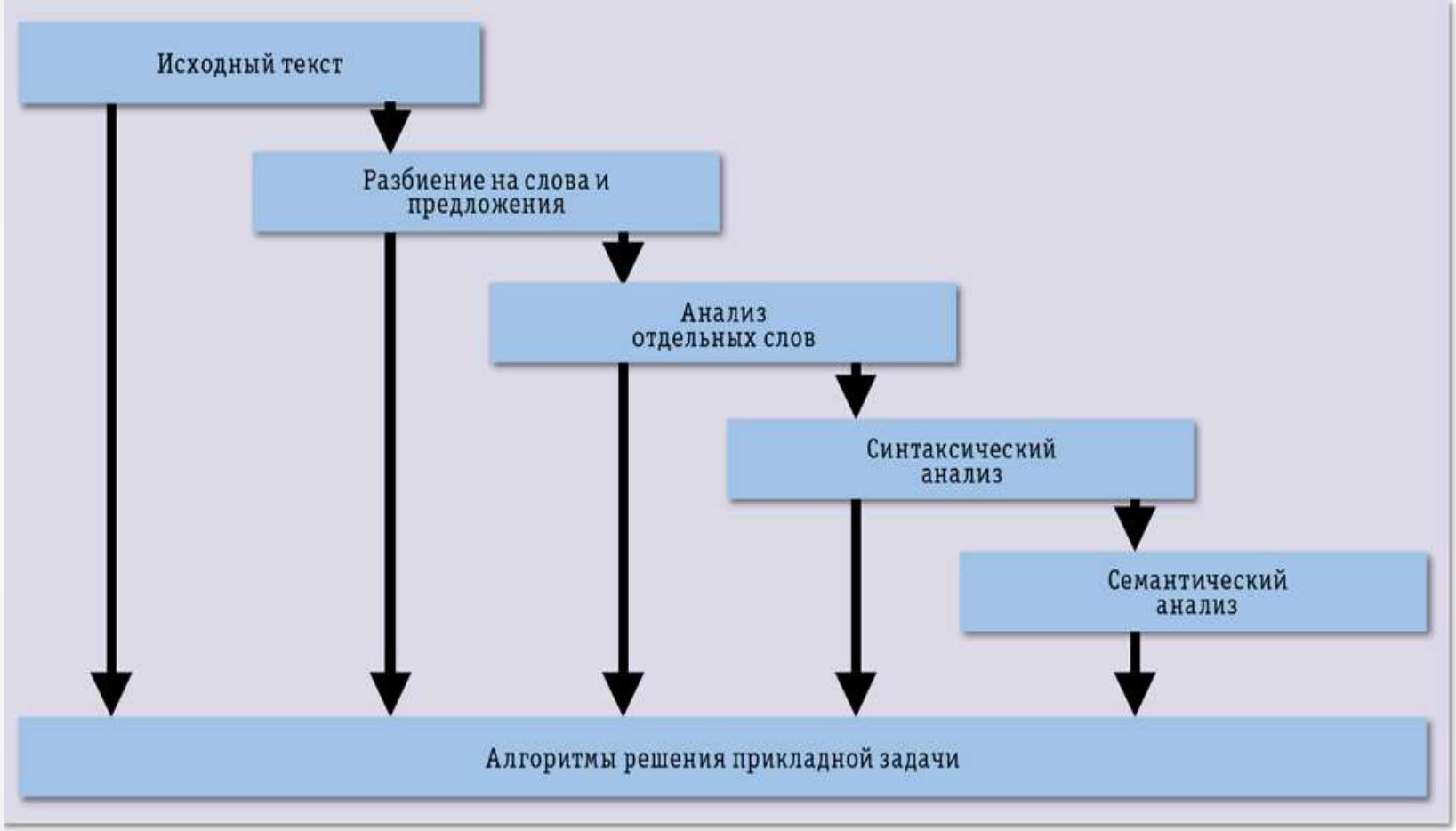
- Индуктивное (основано на выявлении эмпирических закономерностей во входных данных)
- Дедуктивное (предполагает формализацию знаний экспертов и их перенос в цифровую форму в виде базы знаний)

Обработка текстов

Задача обработки текстов на естественном языке касается задач информационного поиска, машинного перевода и т. п.

Анализ текста может проходить в классическая поэтапной обработке текстов (рассмотрим далее) или в две фазы:

Обработка текстов



Лемматизация -приведение к основной словоформе.

Стемминг - выделение некоторой неизменяемой части слова.

Вероятностные тематические модели

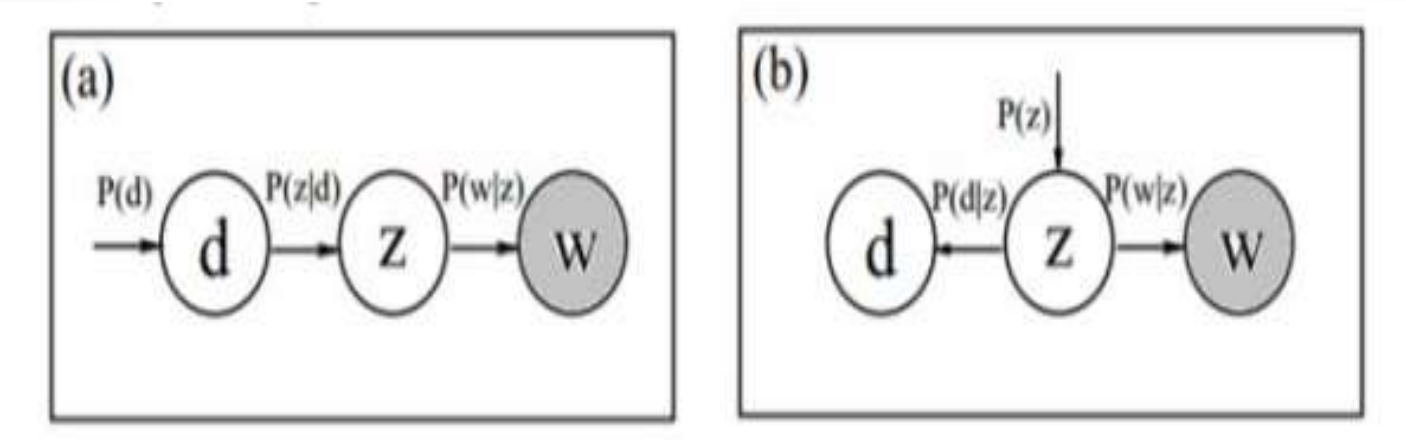
Вероятностное тематическое моделирование (ВТМ) – это способ построения модели коллекции текстовых документов, которая определяет, к каким темам относится каждый из документов.

Особенности:

1) Мягкая кластеризация (слово или документ могут быть отнесены сразу к нескольким темам с различными вероятностями).
2) Основаны на гипотезе «Мешка слов» (порядок слов в документе и порядок документов в коллекции не имеют значения).

Вероятностные тематические модели

Одной из первых ВТМ является вероятностное латентно-семантическое индексирование (Probabilistic Latent Semantic Indexing), PLSI.



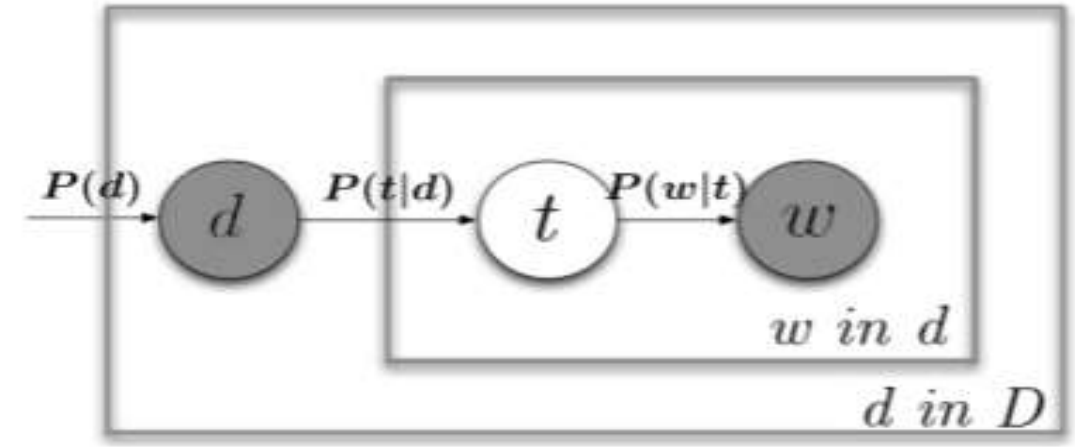
$P(d,w)= P(d) \sum P(w \mid z)P(z \mid d)$

$P(d,w)= P(z)P(w \mid z)P(d \mid z)$

Графическое представление модели вероятностного латентно-семантического индексирования с ассиметричной(a) и симметричной (b) параметризацией.

Вероятностные тематические модели

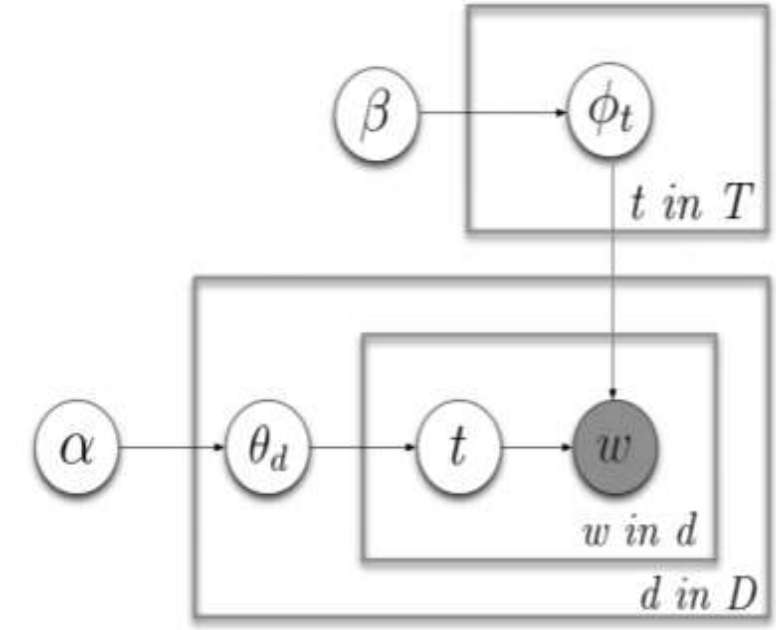
Вероятностный латентно-семантический анализ (индексирование) (PLSA, Probabilistic Latent Semantic Analysis) был предложен Томасом Хоффманом в работе “Probabilistic latent semantic indexing”.



$$P(d, w) = \sum_{t \in T} P(t)P(w|t)P(d|t) = \sum_{t \in T} P(d)P(t|d)P(w|t) = \sum_{t \in T} P(w)P(t|w)P(d|t)$$

Вероятностные тематические модели

Латентное размещение Дирихле (LDA, Latent Dirichlet Allocation) - это порождающая модель, объясняющая результаты наблюдений с помощью неявных групп.



Вероятностные тематические модели

Аддитивная регуляризация тематических моделей (ARTM) помогает разрешить проблемы, связанные с построением тематических моделей, такие как неустойчивость и плохая интерпретируемость тем.

Примеры регуляризаторов:

1) Сглаживающий регуляризатор
2) Разреживающий регуляризатор
3) Ковариационный регуляризатор для документов
4) Ковариационный регуляризатор для тем
5) Регуляризатор для частичного обучения и др.

Работа классификатора

Этап обработки текста:

```
['I like eat delicious food. Thats Im cooking food myself, case "10 Best '
 'Foods" helps lot, also "Best Before (Shelf Life)"']
```

```
['eat food s be cook food case food help lot shelf life', 'help eat exercise basis']
```

Работа классификатора

Этап обучения модели:

	Topic0	Topic1	Topic2	Topic3	Topic4	Topic5	Topic6	Topic7	Topic8	Topic9	dominant_topic
Doc0	0.01	0.01	0.01	0.01	0.01	**0.91**	0.01	0.01	0.01	0.01	**5**
Doc1	0.02	**0.47**	0.02	0.02	0.02	**0.37**	0.02	0.02	0.02	0.02	**1**
Doc2	0.03	0.03	0.03	0.03	0.03	0.03	0.03	**0.77**	0.03	0.03	**7**
Doc3	0.05	0.05	0.05	0.05	0.05	0.05	0.05	**0.55**	0.05	0.05	**7**
Doc4	0.05	**0.55**	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	**1**
Doc5	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0
Doc6	0.05	**0.55**	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	**1**
Doc7	**0.2**	0.02	0.02	0.02	0.02	**0.42**	0.02	0.02	**0.25**	0.02	**5**
Doc8	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0
Doc9	0.01	0.01	0.01	0.01	**0.52**	**0.27**	**0.13**	0.01	0.01	0.01	**4**
Doc10	0.02	0.02	0.02	0.02	0.02	**0.3**	0.02	0.02	0.02	**0.57**	**9**
Doc11	0.02	0.02	0.02	0.02	0.02	**0.18**	0.02	0.02	0.02	**0.66**	**9**
Doc12	0.03	0.03	0.03	0.03	0.03	**0.7**	0.03	0.03	0.03	0.03	**5**
Doc13	0.03	0.03	0.03	0.03	0.03	**0.77**	0.03	0.03	0.03	0.03	**5**
Doc14	0.02	0.02	0.02	0.02	**0.49**	**0.38**	0.02	0.02	0.02	0.02	**4**

Работа классификатора

	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9	Word 10	Word 11	Word 12
Topic 0	use	need	thank	recommend	app	year	news	date	download	want	story	enjoy	start
Topic 1	problem	look	way	picture	think	color	deal	work	order	understand	choice	episode	help
Topic 2	game	play	level	fun	make	money	player	spend	start	thing	character	buy	think
Topic 3	screen	video	hate	change	guy	access	datum	enter	application	second	file	button	article
Topic 4	star	thing	make	photo	way	month	fix	lose	item	miss	track	content	share
Topic 5	lot	help	book	price	check	food	option	learn	developer	choose	feature	information	stay
Topic 6	time	try	account	let	work	card	ask	need	log	device	waste	crash	watch
Topic 7	update	work	app	version	people	know	user	read	notification	option	review	page	want
Topic 8	phone	make	day	money	send	minute	email	try	life	save	app	instal	site
Topic 9	love	add	pay	say	fix	tell	want	thing	star	app	work	list	thank

Работа классификатора

Обработка моделью введенного текста:

```
# Predict the topic
mytext = ["Experience with content management systems a major plus (any blogging counts!)Familiar with
infer_topic, topic, prob_scores = predict_topic(text = mytext)
print(topic)
print(infer_topic)
```

```
['work', 'app', 'version', 'people', 'know', 'user', 'read', 'notification', 'option', 'review', 'pag
e', 'want', 'stop']
Topic 7
```

Результаты работы:

1)Изучены представление текстовой информации в программах и вероятностные тематические модели

2) Разработан классификатор текстов

3) В качестве тестирования я проанализировала 2 датасета.

В базе данных fake_job_postings.csv количество слов в одном документе в среднем составило 60 штук.

Была получена точность правильного предсказания класса текста 93.5%

В базе данных googlePlayStore_review_LDA.csv количество слов в одном документе в среднем составило 10 штук.

Была получена точность правильного предсказания класса текста 79.4%

.

Спасибо за внимание!