2024 год

Построение модели для классификации входящих звонков в голосовом помощнике

Цель проекта

- Увеличение автоматизации, которая позволит компании снизить трудоемкость колл-центра и перераспределить ресурсы на более важные задачи.

Задача проекта

- Построить модель, которая будет классифицировать тематику звонка, и на основе данной тематики решать запросы клиентов.

Что было сделано командой

- Размечены данные (использовали несколько подходов). Построена модель, которая классифицирует тематику звонка.

Base line

1. Тематическое моделирование. Нужно выделить темы и разметить данные.

Было решено использовать один из подходов LDA, CTM, GSDMM.
В результате тематического моделирования планировалось выделить темы, о которых говорят клиенты в своих вопросах. Условный пример:

- Получить код товара
- Узнать срок хранение заказ

2. Построить модель классификации на размеченных данных

В качестве модели классификатора попробовать fasttext.

Base line

Как решали задачу на этом этапе:

1. **Сделали развед анализ.**

   Посмотрели какие данные, сколько дублей, частота повторения идентичных вопросов и т д.

2. **Предобработали текстовые данные**

a. Удалили стоп-слова (из библиотеки nltk.corpus)
b. Нормализовали текстовые данные
c. Удалили знаки препинания, символы
d. Удалили выбросы

3. **Разметили данные при помощи алгоритма LDA**

   Выделили 4 и 8 классов (два датасета)

4. **Построили модель классификации fasttext (на 4-ех классах)**

   Получили accuracy 0.97

Выводы по итогам base line

Мы поняли, что классификатор на данном датасете работает достаточно хорошо, и задача скорее сводится именно к кластеризации (максимально корректной разметке данных).

Дальнейшие шаги по разметке данных

Решили дополнительно попробовать два варианта разметки данных кроме LDA:

1. K-means. Данный метод показал более адекватные результаты по сравнению с LDA.
2. Логика + K-means. Как размечали логикой:

а) Нормализовали текстовые данные

б) При помощи value_counts выделяли самые большие классы по вхождению слов и словосочетаний из метода value_counts. Таким образом удалось выделить 11 основных классов которые составили около 65% от всего датасета. Остальные 9 менее однозначных класса выделили при помощи метода kmeans.

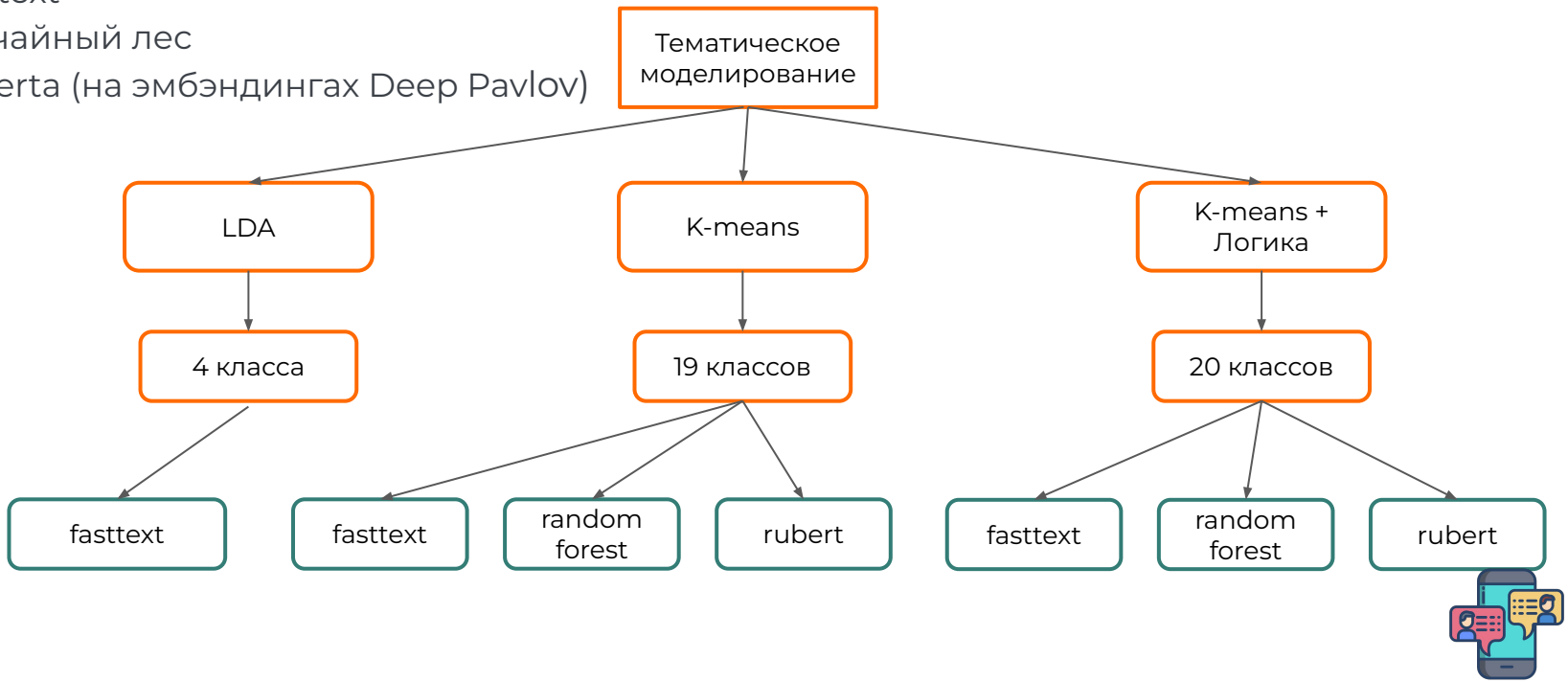
После разметки у нас получилось два размеченных датасета:

1. По методу kmeans
2. Логическим методом + kmeans

Дальнейшие шаги по модели классификации

На основе размеченных данных, для каждого из датасетов обучили три классификатора

1. Fasttext
2. Случайный лес
3. Roberta (на эмбэндингах Deep Pavlov)



Итоги

Тип размеченных данных	Модель классификации	Кол-во классов	Accuracy
kmeans	fasttext	19	0.93
kmeans	random_forest	19	0.95
kmeans	rubert	19	0.92 (на двух эпохах)
логика+kmeans	fasttext	20	0.96
логика+kmeans	random_forest	20	0.96
логика+kmeans	rubert	20	0.95 (на первой эпохе)

По итогам нашей работы лучшее решение было получено в результате:

1. Разметка - гибридный подход (логика + kmeans)
2. Модель классификатора - Хорошие результаты показали random forest и fasttext. Fastext очень легковесный и считается в десятки, а то и в сотни раз быстрее всех остальных, а качество по метрикам не хуже. Также стоит отметить rubert transformer. Нам хватило ресурсов на одну эпоху, но даже на ней accuracy 95%. Возможно у rubert самый большой потенциал по качеству классификации. Но его минус, что он требует очень много ресурсов

Технические детали работы

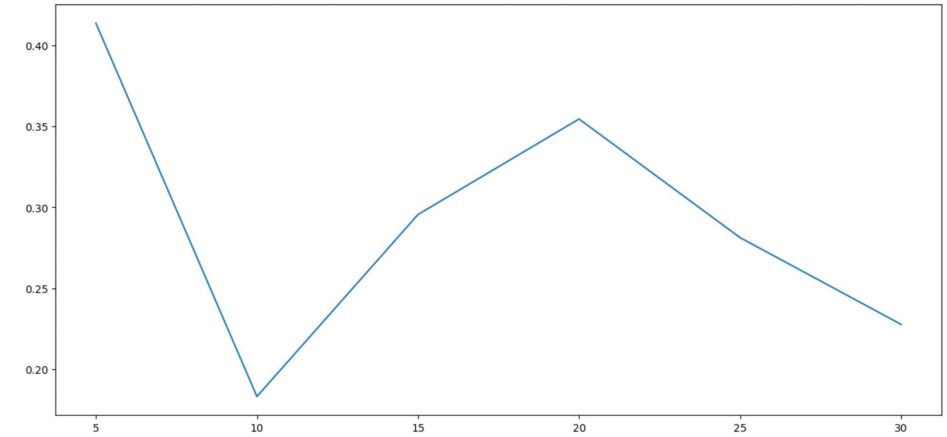
EDA

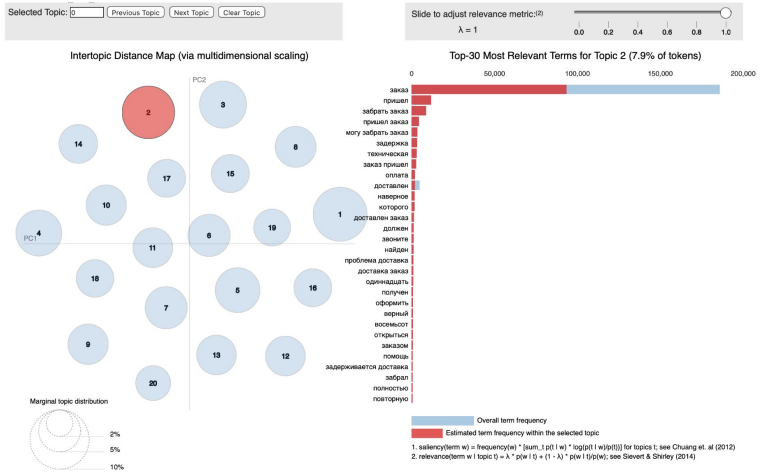
- Посмотрели как распределенные вопросы по частоте упоминания (уже на данном этапе стало понятно, что большинство вопросов связано с небольшим кол-вом классов, а вся сложность выделить оставшиеся классы)
- Поискали и удали выбросы (по минимальному кол-ву символов и методом боксплота)

Технические детали работы

Тематическое моделирование. Как выделяли темы:

1. LDA. Использовали решение из библиотеки gensim. В качестве метрики мы использовалась когерентность. Кол-во тем выбирали по методу локтя. Тематику классов определяли исходя из визуализации, которая показывает вхождение наиболее вероятных токенов в топике

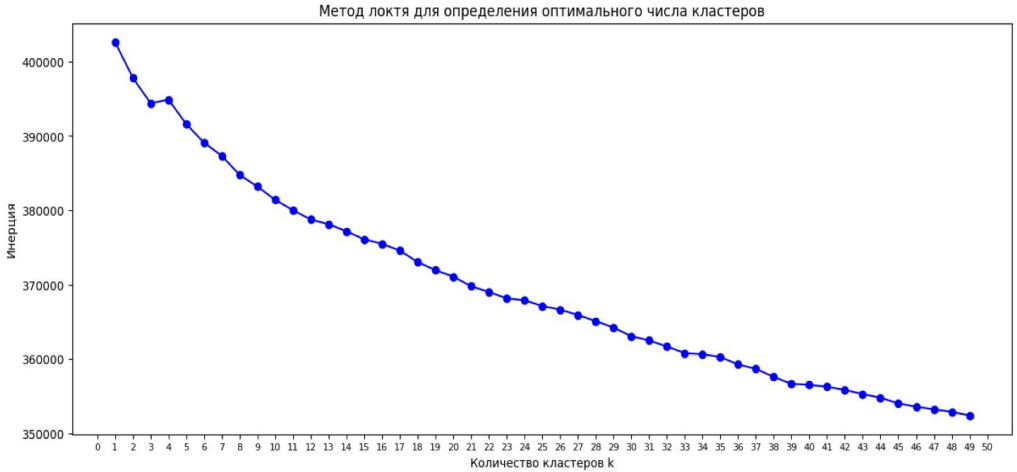


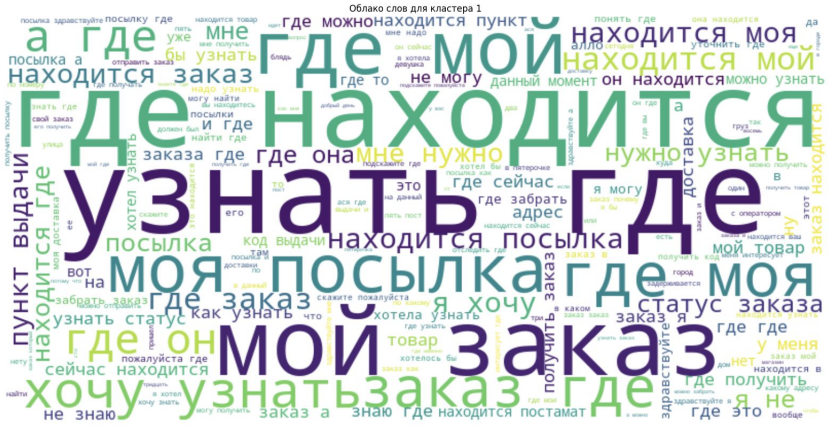


Технические детали работы

Тематическое моделирование.

2. K-means. Преобразовываем наши текстовые строки в вектора с помощью TFIDF метода. Далее при помощи метода локтя ищем излом для определения оптимального количества кластеров, которое указываем для K-Means модели. После строим облака слов для визуализации текстов из кластеров. В Облаках, превалирующие тексты выделены более крупным шрифтом для удобства и визуализации.





Технические детали работы

Модели классификации

Random forest:

1. Для преобразования текста в вектора используем TfidfVectorizer()
2. Обучаем модель

Fasttext:

1. Подготовка данных (очистка, нормализация, удаление стоп слов)
2. Преобразуем размеченные классы в нужный формат
3. Обучаем модель используя библиотек fasttext

Технические детали работы

Модели классификации

Rubert transformers:

1. Подготовка данных (очистка, нормализация, удаление стоп слов)
2. Токенезируем текстовые данные используя AutoTokenize, используя библиотеку transformers
3. Эмбендинги - DeepPavlov
4. Размер батча - 16
5. learning_rate - 5e-5,
6. Кол-во эпох - 3
7. Оптимайзер - Адам