**Национальный исследовательский университет**

**«Высшая школа экономики».**

Дисциплина «Интеллектуальные информационные системы».

Направление«Бизнес-Информатика» подготовки бакалавра.

Исследовательская работы на тему:

Churn Prediction for KKBOX service

Выполнили студенты:

Согоян Гурген Арамович, ББИ156

Римский Никита Валерьевич, ББИ153

Нестерова Милана Вадимовна, ББИ156

Преподаватель, проверяющий работу:

Шевгунов Тимофей Яковлевич

Москва, 2017

Оглавление

[Введение 3](#_Toc500873525)

[Основная часть 4](#_Toc500873526)

[Формирование выборки 4](#_Toc500873527)

[Описание данных 6](#_Toc500873528)

[Таблица Is\_churn 6](#_Toc500873529)

[Таблица Members 6](#_Toc500873530)

[Таблица User\_logs 9](#_Toc500873531)

[Таблица Transactions 11](#_Toc500873532)

[Дополнительный переменные 11](#_Toc500873533)

[Формирование итоговых выборок 13](#_Toc500873534)

[Создание моделей 13](#_Toc500873535)

[Простые классические модели 14](#_Toc500873536)

[Логистическая регрессия 14](#_Toc500873537)

[Дерево решений 15](#_Toc500873538)

[Байес 15](#_Toc500873539)

[SVM 15](#_Toc500873540)

[Ансамблевые методы 15](#_Toc500873541)

[Случайный лес 15](#_Toc500873542)

[Градиентный бустинг 16](#_Toc500873543)

[Adaboost 17](#_Toc500873544)

[Другие 17](#_Toc500873545)

[Нейросеть 17](#_Toc500873546)

[XGboost 17](#_Toc500873547)

[Вывод 18](#_Toc500873548)

# Введение

Задача классификации это одна из классических задач, которую решают исследователи в области анализа данных. Понять к какому классу следует отнести тот или иной объект в зависимости от значений его параметров и быть максимально точным в этом предсказании это цель которую ставит для себя каждый data scientist, решая подобную задачу.

Стоит обратить внимание, что предсказывать класс объекта можно абсолютно в разных областях, начиная с одной из наиболее известной задачи классификации ирисов, заканчивая сложными моделями кредитного скоринга, дающую нам информацию о том, сможет ли клиент вернуть кредит или нет. В последние годы широкое распространение получил такой вид задач, как churn prediction, цель которого предсказать отток клиентов или иначе определить будет ли пользователь продолжать пользоваться сервисом компании или нет.

Очевидно, что любая компания заинтересована в том, чтобы человек продолжал пользоваться предоставляемыми ею услугами, а не отказывался, считая это бесполезным или считая, что уровень обслуживания достаточно низок, чтобы уйти к конкурентам. Если компании удастся вовремя определить подобную угрозу, то у нее еще есть шанс и время до конца подписки доказать клиенту, насколько важно и полезно то, что для него делает компания.

Одной из таких компаний является KKBOX, крупнейший азиатский музыкальный сервис, предоставляющий безграничный доступ к библиотеке, содержащей более 30 миллионов композиций, в том числе малоизвестных азиатских артистов. Соответственно, в зависимости от срока подписки музыке клиент должен будет заплатить определенную сумму. Через небезызвестный сайт kaggle.com в наши руки попала база данных подписчиков данного сервиса, информации о них, об их действиях об их транзакциях и т.д., а также информация о том, остались они клиентами или нет.

**Объектом исследования** работы является база данных клиентов KKBOX

**Предметом исследования** является модель, предсказывающая продолжение подписки пользователя.

Таким образом, целью нашего исследование является построение модели, предсказывающей продолжен ли клиент пользоваться услугами KKBOX, или перестанет быть ее клиентом, определение факторов, влияющих на значение целевой переменной, а также поиск модели, дающей наименьшую ошибку при тестировании данных. Для достижения цели нам будет необходимо решить следующие задачи:

# Основная часть

## Формирование выборки

Изначально, выборка состояла из четырех таблиц

1. Members – содержит информацию обо всех клиентах данного сервиса каждая строка содержит значения для следующих признаков:

* ID пользователя
* Город
* Возраст
* Пол
* Способ регистрации
* Дата регистрации

1. User\_logs – содержит информацию о том, когда пользователь заходил на сайт и какое количество музыки прослушивал, каждая строка в этом столбце является фактов входа в аккаунт на сайте, соответственно действия одного пользователя описываются не одной строкой, как это было в предыдущей таблице, а несколькими, рассмотрим признаки:

* ID пользователя
* Дата вхождения
* Количество треков, прослушанных менее, чем на 25%
* Количество треков, прослушанных менее, чем на 50%, но более чем на 25%
* Количество треков, прослушанных менее, чем на 75%, но более чем на 50%
* Количество треков, прослушанных менее, чем на 75%, но более чем на 98.5%
* Количество треков, прослушанных более чем на 98.5%
* Количество уникальных композиций, включаемых пользователям
* Общее количество секунд затраченных на прослушивание композиций

\*\*\* не дописал все признаки у трансакшн

1. Transactions – состоит из информации о проведенных, непроведенных, отмененных платежах, которые были инициированы пользователями, чтобы продлить подписку на сервис. Каждая строка данной таблицы — это информация об одной транзации пользователя, в этом Transactions схоже с User\_logs, то есть здесь действия одного пользователя описываются несколькими строками, которые содержит данные о следующих свойствах:

* ID пользователя
* Способ оплаты
* На сколько дней была куплена подписка
* Необходимая сумма оплаты
* Заплаченная сумма
* Дата окончания подписки
* Дата транзакции
* Автоматическое обновление подписки
* Отменена ли оплата

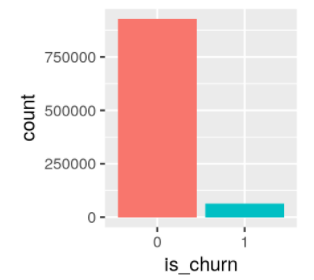
1. Ischurn – финальная таблица, в которой содержится значение нашей целевой переменной, продолжил ли пользователь пользоваться сервисом или нет:

* ID пользователя
* Целевая переменная

Важным фактом является то, что таблица Members состоит из 970960 строк, что уже говорит о потенциально больших подсчетах, кроме того в данных большое количество пропусков и из-за способа организации таблиц Transactions и User\_logs, для которых отсутствует агрегированная информация о пользователях и их действиях внутри сервиса, можно говорить о некоторой неструктурированности в данных. Обычно эти две проблемы в вопросах построения различных моделей решаются с использованием методов из Big Data. \*\*\* следующий абзац не оч нрав начало

Мы же стараемся строить модель, используя «классическое» машинное обучение. Поэтому из нашей большой выборки, мы будем случайным образом формировать подвыборку.

Кроме того, в базе, предоставленной данным сервисом есть определенный дисбаланс классов, так как количество людей, которые продлили подписку всего 10% от общего числа пользователей. *(Изображение 1)*



*Изображение 1*

\*\*\* нуэно списздить картинку у чувака где про баланс классов написано

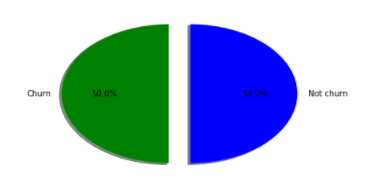
Поэтому, при отборе финальной выборки, будем пытаться сделать классы сбалансированными.

Таким образом, из таблицы Members были случайным образом взяты 3000 пользователей, которые продлили подписку и 3000, которые не продлили, а далее по их ID из остальных таблиц была взята вся остальная информация о них. Приступая к рассмотрению данных необходимо понимать, что помимо выявления закономерностей в признаковом описании этих данных, нам необходимо будет придумать способ агрегированиях дял тех пользователей информация о которых в таблицах Transactions и User\_logs содержится в нескольких строках.

## Описание данных

### Таблица Is\_churn

Опишем значения целевой переменной. Так как мы искусственно создали выборку, в которой классы сбалансированы, круговая диаграмма выглядит весьма предсказуемо. *(Изображение 2)*. При построении модели будем считать класс Churn положительным, то есть значения переменной is\_churn для объектов, принадлежащих ему будет равно 1, а второму классу 0.

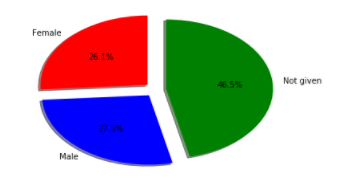


*Изображение 2*

### Таблица Members

**Пол (Gender)**

Обратим внимание на то, что по данному признаку, мы имеем большое количество пропусков: 2787 из 6000. Скорее всего, при пользовании сервисом пользователям необязательно указывать свой пол, поэтому в нашей выборке, пол известен только для чуть более чем половины клиентов. Как мы можем заметить, мужчин и женщин примерно одинаковое количество: мужчин 27.5%, женщин 26.1%. *(Изображение 3)*



*Изображение 3*

Поскольку пол не указан у большого числа пользователей, будем считать этот недостаток информации за отдельный класс, таким образом по параметру пол пользователи будут делиться на три группы: мужчины, женщины и те кто не указали данную информацию.

**Возраст (Age)**

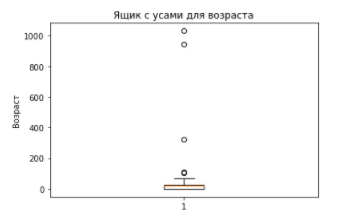
Переменная, измеряемая по относительной шкале. Не содержит пропусков, обратим внимание на то, что минимальное значение — это 0, а максимальное 1032, очевидно, что такие значения в реальности быть не могут, скорее всего при создании профиля пользователь указал неверный год рождения исходя из чего, мы имеем такие выбросы. (Таблица 1)

*Таблица 1*

*Статистика по возрасту*



Диаграмма «ящик с усами» показывает, что количество выбросов не очень большое, их всего 5, но можно заметить, что я нижняя граница ящика находится на 0, значит у нас большое количество значений равных 0, скорее всего, как и в случае с возрастом, клиенты могли не указывать свой возраст. *(Изображение 4)*



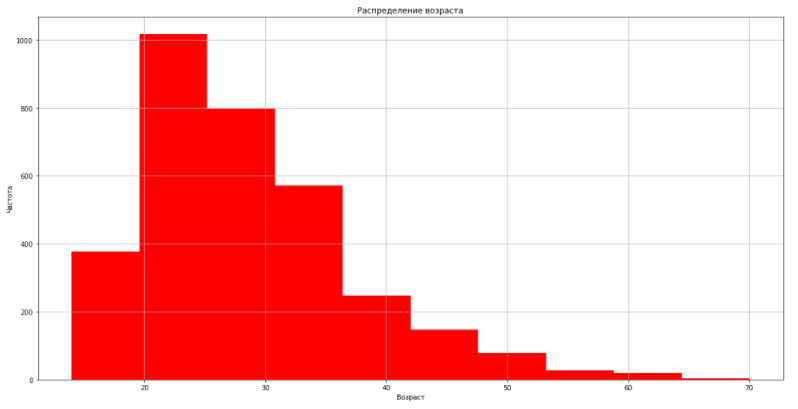
*Изображение 4*

Рассмотрим значения по данной переменной, когда она более 6 и не превышает 100, потому что это более близко к реальному возрасту аудитории сервиса KKBox. Как можно заметить число таких наблюдений уменьшилось вдвое (столбец count) и равно 3288 и по сравнению с предыдущими описательными статистиками мы видим, что среднее сдвинулось с 16 до 28.5, а медиана с 19 до 27 (*Таблица 2*), то есть аудитория в своем большинстве молодая от 20 до 30 лет, что также подтверждается гистограммой распределения возраста. *(Изображение 5)*

*Таблица 2*

*Статистика по возрасту без выбросов*



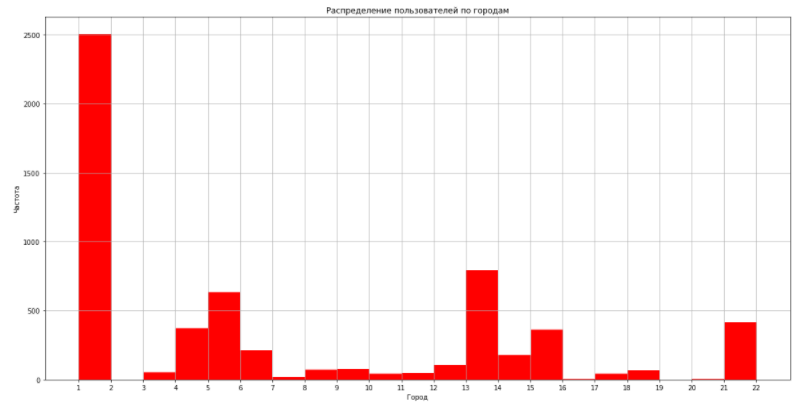


*Изображение 5*

Соответственно необходимо будет скрыть значения выбросов, чтобы они не вносили помехи при построении моделей.

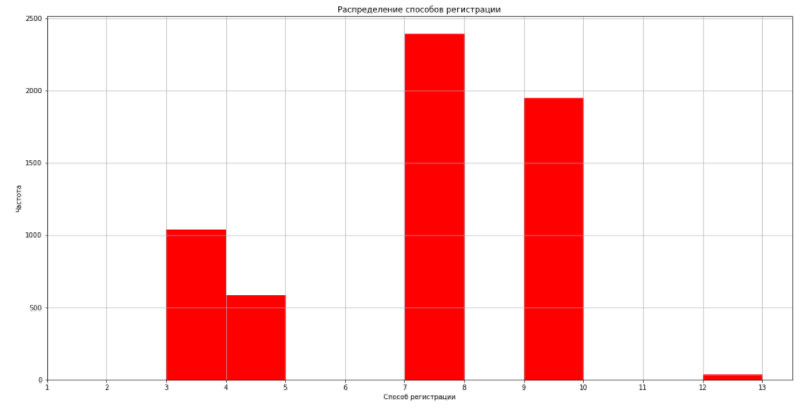
**City**

Номинальная шкала. В которой закодированы 22 города, в которых проживают пользователи, как мы видим и почти половина пользователей проживают в городе 1, и в принципе пользователи по городам распределены неравномерно, пропущенных данных нет.



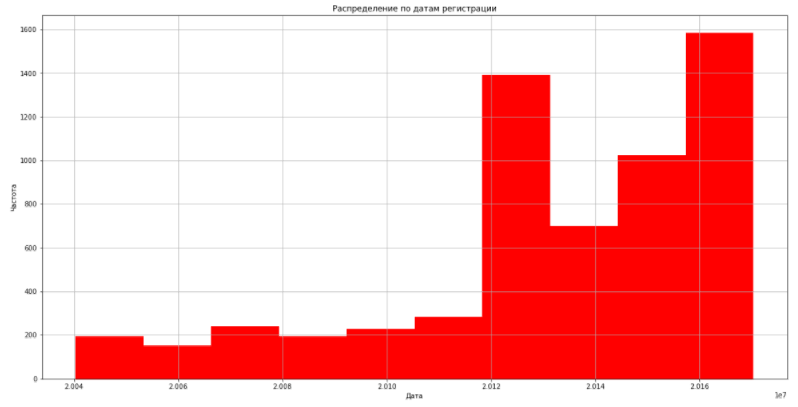
**Registered\_via**

Номинальная шкала, в которой закодированы 13 способов регистрации (Изображение 6). Как можно заметить, люди из нашей выборки пользуются далеко не всеми способами, а только 3, 4, 7 ,9 и совсем маленькое количество 12, причем более трети регистраций приходится на 7 способ.



**Registration\_init\_time**

Рассмотри график для последней переменной в этой таблице, которая имеет тип времени, соответственно из гистограммы мы можем посмотреть в какое время регистрировалось больше всего пользователей. Можно сделать вывод что в целом популярность сервиса растет, что количество регистраций с годами увеличивается, несмотря на то что после пика в 2012 скорость роста регистраций уменьшилась в 2016 она снова доросла и переросла этот уровень. (Изображение 7). Примерно четверть людей стала пользоваться сайтом KKBOX недавно, что может влиять на их желание продолжить подписку, так как если пользователь пользуется сервисом



### Таблица User\_logs

Нами уже было замечено, что данные в этой таблице имеют структуру, в которой каждому клиенту соответствуют несколько строк данных, которые нам предстоит агрегировать, чтобы объект передаваемые в модель для обучения были представимы в виде массива данных одинаковой структуры. Посмотрим, какие поля мы можем сгенерировать.

* Num\_of\_logs: количество входов на сайт

А для каждого из num\_25, num\_50, num\_75, num\_985, num\_100, num\_unq, total\_secs будем считать:

* **Avg:** сколько в среднем пользователь прослушивает аудиозаписей или секунд он слушает музыку за одно захождение на сайт
* **Sum:** сколько всего песен было прослушано пользователем и сколько секунд было потрачено на прослушивание аудиозаписей за все время пользования сервисом

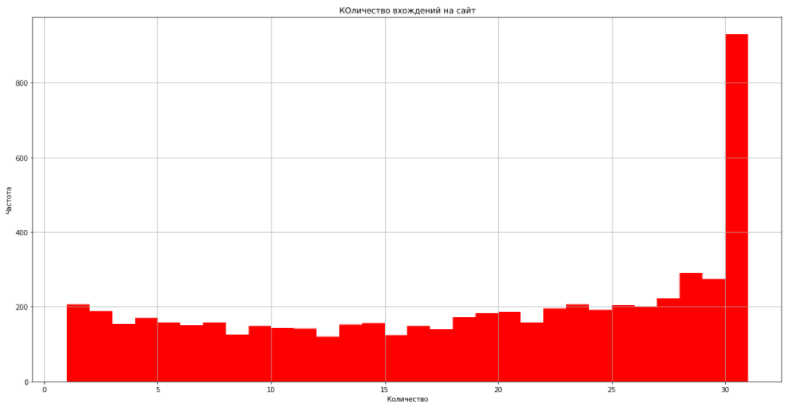
После преобразования данных нужным нам образом, получаем таблицу, состоящую из 6000 строк, в каждой из которых содержится сводная информация по действиям одного пользователя. Рассмотрим данные из этой таблицы

**Num\_of\_logs**

Как можно заметить каждый пользователь хотя бы один раз заходил на сайт, среднее находится в окрестности 18, а медиана 20, что значит, что пользователи достаточно активно пользуются KKBOX. (Таблица)



Гистограмма распределения говорит о том, что относительно остальных значений, значения свыше 30, повторяются значительно чаще, а в промежутке от 1 до 30 примерно с одинаковой частотой

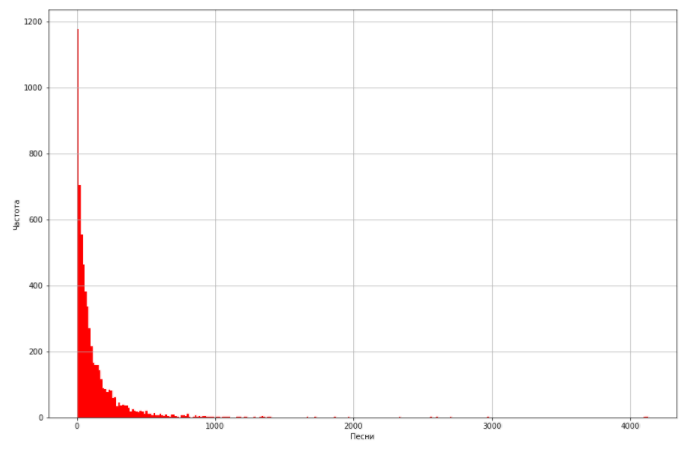
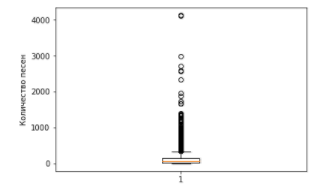


**Sum\_Num\_25**

Как можно заметить по указанной статистике (Таблица) разброс данных достаточно большой, что видно по стандартному о тклонению, равному 200, также количество таких песен у одного из пользователей достигло 4128, при медиане равной 60, значит большое количество значений приближено к 0. Рассмотрим гистограмму распределения этих данных (Изображение).



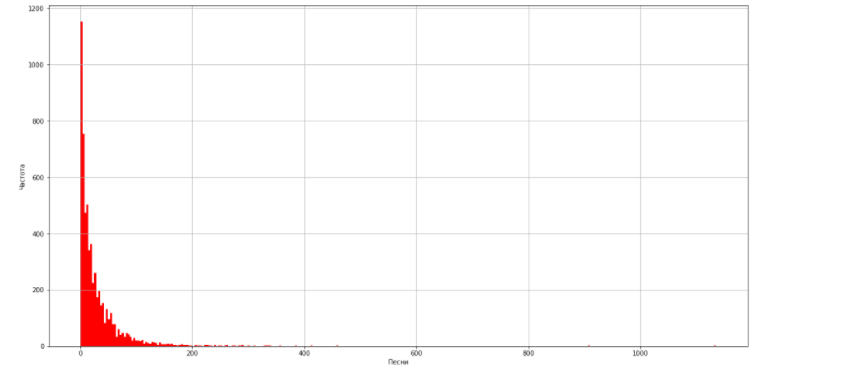
Как мы видим из диаграммы бокса-блабла количество потенциально аномальных очень большое и если на диаграмме частот они едва заметны, так как большое количество значений находится у нуля, то тут явно заметно их число.

**Sum\_Num\_50**

Примерно такой же характер как и у предыдущий переменной имеется у Sum\_num\_50. Единтсевнное что максимальное количество песен здесь 1135, что в 4 раза меньше, чем у предыдущей переменной.

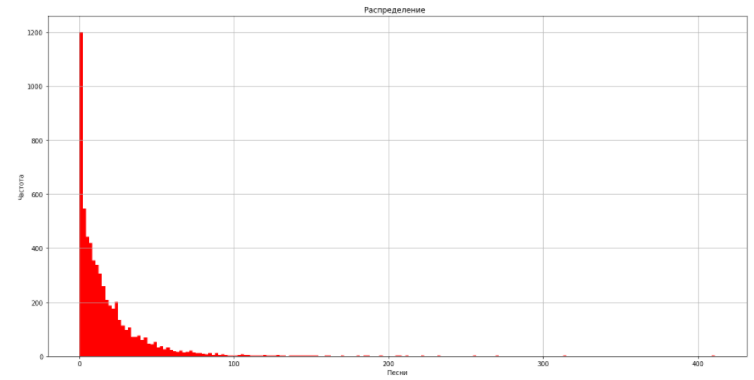




**Sum\_Num\_75**

Примерно такой же характер как и у предыдущий переменной имеется у Sum\_num\_50. Единтсевнное что максимальное количество песен здесь 1135, что в 4 раза меньше, чем у предыдущей переменной.





**Sum\_Num\_75**

Примерно такой же характер как и у предыдущий переменной имеется у Sum\_num\_50. Единтсевнное что максимальное количество песен здесь 1135, что в 4 раза меньше, чем у предыдущей переменной.

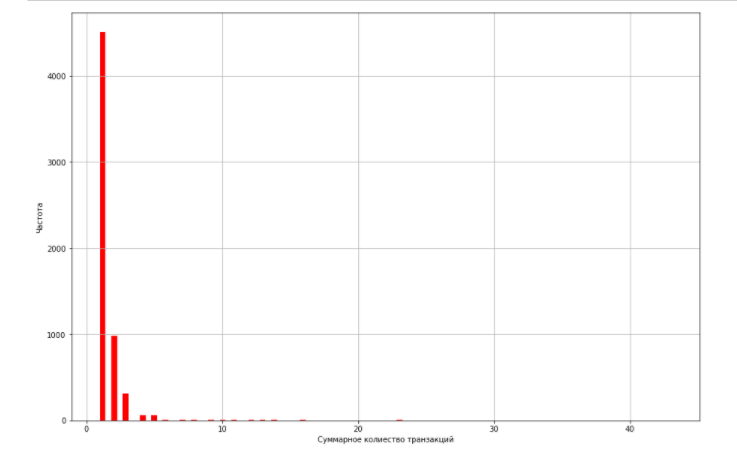
### Таблица Transactions

Заметим, что для таблицы Transactions мы будем так же приводить к таблице вида 6000 строк, где параметры образованы за свет сведения информации по строкам для каждого пользователя.

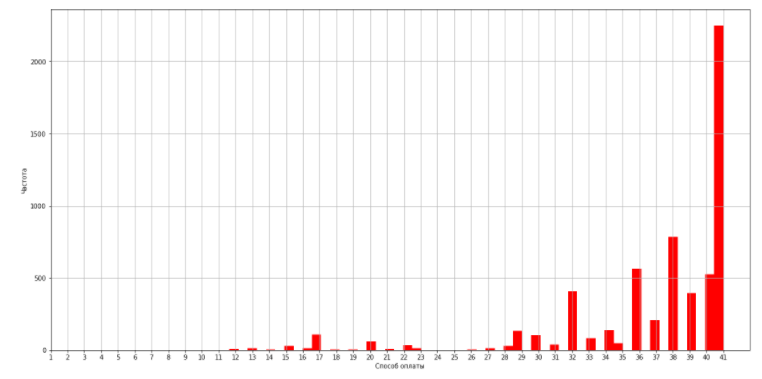
* Num\_of\_transactions количество сделанных транзакций
* Most\_popular\_payment\_id наиболее популярный способ оплаты, который используется клиентом
* Avg\_plan\_list\_pri

'num\_of\_transactions'



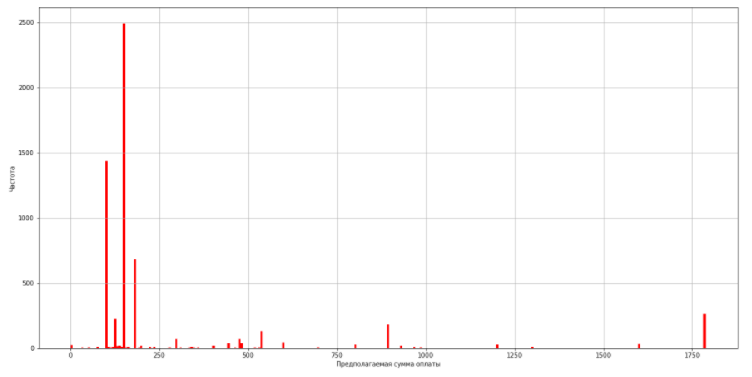


'most\_popular\_payment\_id',



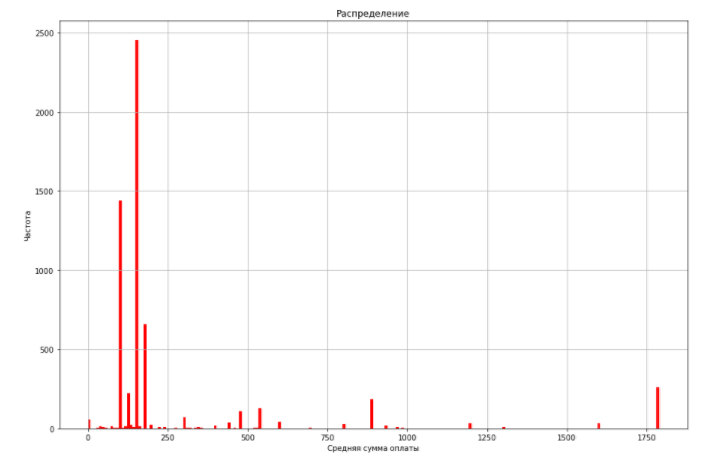
avg\_plan\_list\_price





'avg\_actual\_amount\_paid'





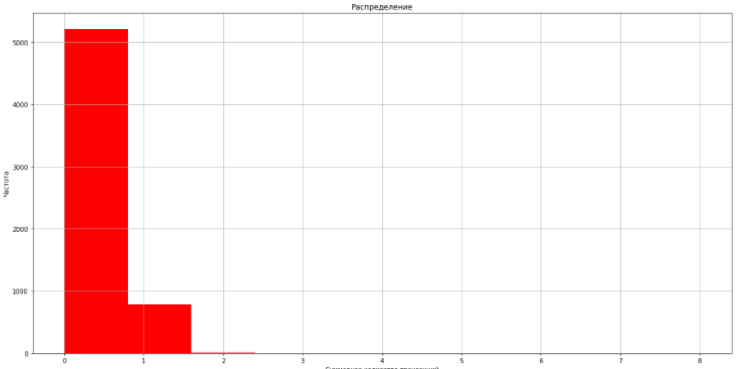
'is\_auto\_renew\_total',



'is\_cancel\_total',

\*\*\*ТУТ Я ПЕРЕДЕЛАЛ ГОВНО ЖОПА ХУЙ

### 



### Дополнительный переменные

Добавим переменные, которые будут образованы их изначальных данных, которые могут так же иметь влияние на нашу переменную.

* Days – день регистрации (1, 2, 3 - начало недели, середина, конец недели), которая образована из переменной дата регистрации
* Season – время года регистрации (1, 2, 3, 4 – зима, весна, лето, очень), которая образована из переменной дата регистрации
* perc\_of\_accidentally\_songs – процент случайно включенных песен, предположим, что песни, которые были прослушаны менее чем на 25% были включены случайно соответственно посмотрим, какой процент от всех песен был включен случайно. Образован путем деления количества песен прослушанных на 25% на общее количество включенных песен
* perc\_of\_favourite\_songs – процент любимых песен, предположим, что песни, которые прослушиваются более чем на 98.5% любимые у пользователя посмотрим, какой процент от всех песен любимые. Образован путем деления количества песен прослушанных более чем на 98.5% и на 100% на общее количество включенных песен.
* avg\_songs\_per\_month – среднее количество прослушиваемых в месяц песен, образовано за счет деления переменных, общее количество прослушанных песен, деленное на количество месяцев, сколько пользователь подписан на сервис
* avg\_days\_diff- среднее количество дней, проходящих между входами на сайт
* max\_days\_diff – максимальное количество дней, проходящих между входами на сайт
* min\_days\_diff – минимальное количество дней, проходящих между входами на сайт
* most\_popular\_days – самый популярный момент недели для прослушивания музыки (1, 2, 3 - начало недели, середина, конец недели)
* least\_popular\_days – самый непопулярный момент недели для прослушивания музыки (1, 2, 3 - начало недели, середина, конец недели)
* most\_popular\_season - самый популярное время года для прослушивания музыки (1, 2, 3, 4 – зима, весна, лето, очень),
* least\_popular\_season – самый непопулярное время года для прослушивания музыки (1, 2, 3, 4 – зима, весна, лето, очень),
* avg\_transaction\_days\_diff – среднее количество дней, проходящих между транзакциями
* max\_transaction\_days\_diff – максимальное количество дней, проходящих между транзакциями
* min\_transaction\_days\_diff – минимальное количество дней, проходящих между транзакциями
* avg\_expire\_days\_diff – среднее количество дней, проходящих между транзакциями
* max\_expire\_days\_diff
* min\_expire\_days\_diff
* most\_popular\_transaction\_days - самый популярный момент недели для оплаты (1, 2, 3 - начало недели, середина, конец недели)
* least\_popular\_transaction\_days - самый непопулярный момент недели для оплаты (1, 2, 3 - начало недели, середина, конец недели)
* most\_popular\_transaction\_seasons - самый популярное время года для прослушивания музыки (1, 2, 3, 4 – зима, весна, лето, очень),
* least\_popular\_transaction\_seasons - самый непопулярное время года для прослушивания музыки (1, 2, 3, 4 – зима, весна, лето, очень),
* most\_popular\_expire\_days
* least\_popular\_expire\_days
* most\_popular\_expire\_seasons
* least\_popular\_expire\_seasons

### Формирование итоговых выборок

Модели, которые мы в последствии будем обучать имеют различные требования к выборкам. Давайте разделим их на три части и опишем какими свойствами должны обладать выборки и исходя из этого подготовим три разные выборки для максимально эффективного обучения. \*\*\*Норм?

* **Для линейных моделей.** Данные должны быть отнормированы, так модель будет и быстрее обучаться и лучше предсказывать, а все номинальные переменные должны быть обязательно закодированы дамми-переменными.
* **Для моделей с деревьями решений.** Номинальные переменные не обязаны быть представлены в виде дамми-переменных, более того это может добавить проблемы в обучении в виде увеличения затрат времени на обучение.
* **Для xgboost**

Соответственно, было сформировано три итоговые выборки, которые различаются, описанными выше свойствами, перейдем к обучению моделей.

\*\*\* надо сказать про то что нот гивен у мужиков и телок это отдельная хуйня

## Создание моделей

На данный момент существует множество различных методов, моделей, алгоритмов обучения, которые помогают решать задачу классификации. Машинное обучение в этом плане продвинулось уже намного дальше классической логистической регрессии, байесовского классификатора, или были созданы ансамблевые методы: Adaboost, градиентный бустинг. Наука продвинулась и были придуманы нейросети и различные методы глубинного обучения.

Но машинное обучение тем и привлекательно, что никогда не знаешь, какой метод и с какими параметрами лучше подойдет для тех или иных данных, ведь не редки случаи, когда наиболее эффективными моделями являются самые простые. Поэтому в ходе исследования были рассмотрены несколько моделей:

* Простые классические модели:
  1. Логистическая регрессия
  2. Дерево решений
  3. Байес
  4. SVM
* Ансамблевые методы
  1. Adaboost
  2. Гладиентный бустинг
  3. Случайный лес
* Другое
  1. Нейросеть
  2. XGBoost

Практически для каждой задачи самой главной задачей является подбор гиперпараметров, мы будем подбирать лучшие гиперпараметры, используя поиск по сетке (GridSearch), который вычисляет модель с какими гиперпараметрами предсказывает лучше всего.

Для того чтобы определять какая модель лучше, а какая хуже необходимо подобрать метрику качества относительно которой мы будем сравнивать модели. Подобных метрик много среди наиболее популярных стоит отметить accuracy, recall (полнота) , precission (верность). Стоит от

\*\*\*итд мы в итоге выбрали f

## Простые классические модели

### Логистическая регрессия

\*\*\*Логистическая регрессия это блаблабла

Рассмотрим, какие гиперпараметры можно настроить для данной модели и какой диапазон значений для них мы будем рассматривать. (Таблица)

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Диапазон | Лучшее |
| penalty | Регуляризация. При добавлении какого типа регуляризации в формулу ошибки модель обучится лучше. | l1, l2 |  |
| c | Коэффициент при регуляризации. | 10^-5….10^5 |  |
| Solver | Какой способ оптимизации использовать, при поиске минимума функции ошибки. Стоит отметить, что некоторые из этих способов оптимизации не работают при l1-регуляризации, так как она содержит сумму модулей. | ‘newton-cg’, ‘lbfgs’, ‘liblinear’, ‘sag’, |  |
| max\_iter | Максимально допустимое количество шагов, используемое для нахождения минимума функции ошибки | 10,50,100,200 |  |

### Дерево решений

\*\*\*написать подводку

\*\*\*не уверен в правильнсоти описаннахы гиепрпараметров

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Диапазон | Лучшее |
| criterion | Мера неопределенности, подсчитываемая для формирования условие разветвления | Энтропия,  Индекс джинни |  |
| splitter | \*\*\* хз | ['best', 'random'] |  |
| max\_depth | Максимальная глубина дерева, не допуская слишком большое число, мы спасаемся от переобучения. | 5,10,20,40,100, 500 |  |
| min\_samples\_leaf | Минимальное количество наблюдений, содержащихся в одном листе. Регулирование данного показателя так же помогает избавиться от переобучения | 1,5, 10,20,50 |  |
| min\_samples\_split | Минимальное количество наблюдений необходимое для добавления нового узла | 2,5,10,20 |  |

### Байес

### SVM

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Диапазон | Лучшее |
| C |  | np.logspace(-2,3, 5) |  |
| gamma | Количество деревьев в случайном лесе. Если будет слишком большое, то есть риск переобучения, а при маленьком недообучения | np.logspace(-3,1) |  |
| kernel |  | [ 'poly', 'rbf', 'sigmoid'] |  |
| decision\_function\_shape |  | ['ovo', 'ovr'] |  |

## Ансамблевые методы

### Случайный лес

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Диапазон | Лучшее |
| criterion | Мера неопределенности, подсчитываемая для формирования условие разветвления | Энтропия,  Индекс джинни |  |
| n\_estimators | Количество деревьев в случайном лесе. Если будет слишком большое, то есть риск переобучения, а при маленьком недообучения | 10,50,100,200 |  |
| max\_depth | Максимальная глубина дерева, не допуская слишком большое число, мы спасаемся от переобучения. | 2,4,10 |  |
| min\_samples\_leaf | Минимальное количество наблюдений, содержащихся в одном листе. Регулирование данного показателя так же помогает избавиться от переобучения | 1,10,20 |  |
| min\_samples\_split | Минимальное количество наблюдений необходимое для добавления нового узла | 2,10,20 |  |

### Градиентный бустинг

learning\_rate\_array=np.logspace(-3,2)

loss\_array=['deviance', 'exponential']

n\_estimators\_array=[100,200, 300]

max\_depth\_array=[3,10]

min\_samples\_split\_array=[2,10]

min\_samples\_leaf\_array=[1,10]

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Диапазон | Лучшее |
| learning\_rate |  | np.logspace(-3,2) |  |
| n\_estimators | Количество деревьев в случайном лесе. Если будет слишком большое, то есть риск переобучения, а при маленьком недообучения | 10,50,100,200 |  |
| loss |  | ['deviance', 'exponential'] |  |
| max\_depth | Максимальная глубина дерева, не допуская слишком большое число, мы спасаемся от переобучения. | 2,4,10 |  |
| min\_samples\_leaf | Минимальное количество наблюдений, содержащихся в одном листе. Регулирование данного показателя так же помогает избавиться от переобучения | 1,10,20 |  |
| min\_samples\_split | Минимальное количество наблюдений необходимое для добавления нового узла | 2,10,20 |  |

### Adaboost

n\_estimators\_array=[50,100,200, 500]

algorithm\_array =['SAMME', 'SAMME.R']

learning\_rate\_array=np.logspace(-3,2)

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Диапазон | Лучшее |
| learning\_rate |  | np.logspace(-3,2) |  |
| n\_estimators | Количество деревьев в случайном лесе. Если будет слишком большое, то есть риск переобучения, а при маленьком недообучения | [50,100,200, 500] |  |
| algorithm |  | [['SAMME', 'SAMME.R'] |  |
|  |  |  |  |

## Другие

### Нейросеть

### XGboost

# Вывод