МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего образования

УРАЛЬСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
имени первого Президента России Б. Н. Ельцина

ИНСТИТУТ ЕСТЕСТВЕННЫХ НАУК И МАТЕМАТИКИ
Кафедра Вычислительной Математики и Компьютерных Наук

ПРОГНОЗИРОВАНИЕ НАГРУЗКИ УДОСТОВЕРЯЮЩЕГО ЦЕНТРА

Направление подготовки 02.04.01 «Математика и компьютерные науки»

Образовательная программа «Современные проблемы компьютерных наук»

Допустить к защите:

Зав. кафедрой
д. ф.-м. н., проф. В. Г. Пименов

Выпускная квалификационная работа магистра

Левина Алексея Дмитриевича

Научный руководитель:
к. ф.-м. н., доц. С.И. Солодушкин

Екатеринбург
2018

РЕФЕРАТ

Задачей данной квалификационной работы было создание сервиса, позволяющего эффективно строить прогнозы нагрузки на Удостоверяющий центр. В процессе решения задачи были проанализированы исходные данные по количеству входящих заявок разных типов. В них выявлено наличие таких характеристик, как наличие сезонных признаков, резкие скачки нагрузки в праздничные дни и в периоды отчетности. По своей внутренней структуре данные были разбиты на два класса прогнозирования. Также был разработан алгоритм оценки эффективности различных методов прогнозирования и с их помощью были проанализированы несколько самых часто используемых моделей.

Из-за разнородности структуры исходных данных и наличия большого количества накопленной аналитической информации, которую сложно автоматически извлечь из обучающей выборки, было принято решение отказаться от полностью автоматизированных методов в пользу оценочных, опирающихся не только на исторические данные, но и априорные сведения о характеристиках прогноза. Среди оценочных методов в качестве модели была выбрана библиотека Facebook Prophet. Был проведен глубокий анализ возможностей и внутреннего устройства этой библиотеки для решения поставленной задачи.

Для настройки и хранения характеристик прогноза был разработан гибкий сервис настройки модели с богатым пользовательским интерфейсом. Цель этого сервиса – привлечь людей, не обладающих глубокими знаниями в статистике и прогнозировании, но владеющих аналитической информацией о структуре изучаемых данных, к настройке модели прогнозов. Для осуществления прогнозов, валидации и подсчета ошибки был написан REST WEB API сервер прогнозирования. После реализации необходимого минимума возможностей сервиса настройки модели, она была использована для настройки двух моделей, и был проведен последующий анализ эффективности прогнозов. Из полученных данных были сделаны выводы, в частности о том, что наличие априорных знаний о различных характеристиках нагрузки позволяет значительно улучшить эффективность прогноза.

Полученная система может применяться не только для прогнозирования нагрузки на Удостоверяющий Центр, но и как сервис для построения и анализа прогнозов самой разной природы и внутренней структуры.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

1. **АО «ПФ «СКБ Контур»** (далее – «**СКБ Контур»**) – Акционерное общество «Производственная фирма «СКБ Контур».
2. **Удостоверяющий центр (УЦ)** – организация, осуществляющая функции по созданию и выдаче сертификатов ключей проверки электронных подписей и иные функции удостоверяющего центра.
3. **Электронная подпись** – информация в электронной форме, которая присоединена к другой информации в электронной форме (подписываемой информации) или иным образом связана с такой информацией, и которая используется для определения лица, подписывающего информацию.
4. **Сертификат ключа проверки электронной подписи** (далее – **сертификат**) – электронный документ или документ на бумажном носителе, выданный Удостоверяющим центром либо Доверенным лицом Удостоверяющего центра и подтверждающий принадлежность ключа проверки электронной подписи владельцу сертификата ключа проверки электронной подписи.
5. **Сервисный центр** – подразделение Удостоверяющего центра или самостоятельное юридическое лицо/индивидуальный предприниматель, уполномоченные Удостоверяющим центром взаимодействовать с клиентами-заявителями.
6. **Оператор УЦ** – человек, осуществляющий проверку сертификатов.
7. **API –** прикладной программный интерфейс.

ВВЕДЕНИЕ

В современном мире все больше операций может быть осуществлено через информационно-телекоммуникационную сеть Интернет: от простого входа в личный аккаунт социальной сети до подачи документов в различные государственные органы. Все больше организаций предпочитают электронный документооборот бумажному. Для выполнения большинства подобных операций необходимо получить сертификат.

Выдача Сертификата осуществляется Удостоверяющими центрами в порядке и на условиях, установленных Федеральным законом «Об электронной подписи» от 06.04.2011 № 63-ФЗ. Одним из крупнейших Удостоверяющих центров в России является СКБ Контур.

Ежедневно в УЦ СКБ Контур приходит от нескольких сотен до нескольких тысяч заявок на выдачу сертификата, каждая из которых должна пройти проверку оператором УЦ. В связи с постоянным ростом клиентской базы, поток заявок увеличивается, что провоцирует также расширение штата операторов УЦ. Также растет и меняется список типов сертификатов, усложняя процесс распределения, так как не все операторы УЦ могут работать со всеми типами сертификатов. Кроме того, помимо проблемы увеличения штата в связи с ростом входящей нагрузки, остро стоит вопрос распределения штата в течение года, месяца и даже в пределах одного дня. Вручную контролировать весь этот процесс становится все труднее. При планировании необходимо учитывать такие факторы как рост среднего числа заявок, насыщенность рынка, наличие сезонных колебаний, рост и падение рынков, пики нагрузки в преддверье отчетности и многие другие. При этом выраженность каждого фактора зависит от типа заявки. Таким образом, возникает потребность в разработке сервиса прогнозирования нагрузки на операторов УЦ.

На текущий момент прогнозирование нагрузки осуществляется вручную путем визуального анализа данных за прошедший период. При этом используются сервисы, которые не представляют информацию в формате удобном для прогнозирования (Рисунок 1). Поэтому такие прогноз нельзя считать точными.

Задачей, которая легла в основу данной квалификационной работы, стало создание инструмента, который позволяет не только автоматизировать процесс прогнозирования, но и сделать сам прогноз практичным.

Статистика очереди УЦ

Всего проверено форм: **13169**

Количество | Условные единицы

Дата	Час:	0-1	1-2	2-3	...	21-22	22-23	23-0	Общая производительность
01.12.2017	Пришло Обработано В очереди	10 15 333	9 15 327	6 16 317		36 144 13	17 26 4	11 11 4	4262 4596
02.12.2017	Пришло Обработано В очереди	6 5 5	8 9 4	0 0 4		12 13 4	5 4 5	3 4 4	174 174
03.12.2017	Пришло Обработано В очереди	6 6 4	5 4 5	3 3 5		12 12 6	12 11 7	9 11 5	155 154
04.12.2017	Пришло Обработано В очереди	9 10 4	6 6 4	4 4 4		45 212 431	35 211 255	20 16 259	4333 4079
05.12.2017	Пришло Обработано В очереди	24 28 255	19 24 250	12 26 236		49 181 967	30 152 845	26 20 851	4758 4166

Рисунок 1 – пример визуализации работы УЦ за пять дней

Появление такого сервиса будет способствовать как снижению нагрузки на менеджеров, занимающихся планированием рабочего графика работы операторов УЦ, так и уменьшению периодов «простоя» в работе операторов УЦ. Также система станет более устойчивой к различного рода пикам нагрузки.

ОСНОВНАЯ ЧАСТЬ

1 Общее описание задачи

Задачу прогнозирования нагрузки на операторов УЦ можно математически описать как задачу регрессии – вычисление зависимости μ{Y | T}. Зависимость обычно выражают формулой $\hat{y}(t|T) = \mu(t) + \varepsilon,\ t \in T, \hat{y} \in Y,$

где $t, \hat{y}, \varepsilon$ – случайные величины, причем t и ε независимы. В нашем случае T – это временной ряд, элементами которого являются моменты времени (обычно взятые с одинаковым интервалом друг относительно друга), $\hat{y} \in \mathbb{R}$ – предсказанное количество пришедших заявок в УЦ в конкретный момент времени[1] (в нашем случае $\hat{y} \in \mathbb{R}^+$, так как заявок не может прийти отрицательное число, но на выбор метода решения этот факт влияния не окажет), $\varepsilon \in \mathbb{R}$ – случайная величина, разброс этого предсказания. Такие задачи в общем случае называют **задачами прогнозирования временного ряда**.

Существует множество формализованных методов для решения задач прогнозирования временного ряда. Однако, у них всех есть общие признаки. Для определения значений $u(t)$ в каждый момент времени t они обрабатывают известные пары значений(t, y) (для нас t – это один определенный период в прошлом, а y – это количество форм, пришедших в этот период). Эти пары значений называются **обучающей выборкой** (training data). Сам метод для решения, сформулированный в виде функционального представления, адекватно описывающее исследуемый процесс и являющееся основой для получения его будущих значений[1], называется **моделью прогноза** (prediction model). Процесс вычисления зависимости $u(t)$ называют процессом **обучения модели**.

Методика подхода к решению задач прогнозирования была сформулирована ещё J. Scott Armstrong [4]. Он систематизировал задачу, разбивая её на 6 стадий, выполняемых последовательно:

1. Сформулировать задачу.
2. Собрать информацию.
3. Выбрать методы.
4. Реализовать методы.

[1] Здесь и далее используется выражение «в конкретный момент времени», несмотря на то, что заявки не приходят одномоментно. Это намеренное упрощение: под «моментом времени» может пониматься как момент, так и интервал времени, без вреда для решения задачи.

5. Оценить методы.
6. Использовать прогноз.

Схематически эти стадии показаны на (Рисунок 1.). В данной работе используется именно этот подход.

Рисунок 1.1. Стадии решения задачи прогнозирования

Формулировка задачи

В общем виде задачу можно сформулировать так:

Разработать сервис для прогнозирования входящей нагрузки в Удостоверяющий Центр с учетом особенностей заявок.

При этом, необходимо определить особенности заявок. Для этого следует **проанализировать и классифицировать данные**, с которыми предстоит работать (пункт 2 схемы J. Scott Armstrong). Однако, после выполнения классификации данных, полученных знаний будет недостаточно для однозначного выбора метода прогнозирования. Методы необходимо сравнивать между собой. Значит, необходимо **определиться с метриками** оценки эффективности работы каждого из методов (пункт 5 схемы J. Scott Armstrong). Этим двум этапам и будут посвящены следующие два подраздела работы.

2 Методы оценки эффективности методов прогнозирования

Как уже писалось выше, все методы прогнозирования принимают на вход обучающую выборку – набор m пар значений (t_i, y_i)[1], – информацию о количестве пришедших форм в известные моменты времени в прошлом. На выходе они на основе этих данных моделируют зависимость $\hat{y}(t_i)$ для произвольного момента времени t_i. В общем случае (без учета моделирования ошибки), зависимость $\hat{y}(t)$ – функциональная: $\hat{y}(t) = \mu(t)$. После нахождения

[1] Здесь и далее будет использоваться запись чисел t и y с индексами i, чтобы проиллюстрировать принадлежность пары (t, y) i-й паре обучающей выборки

такой функции необходимо научиться оценивать её эффективность. Говоря математическим языком, хотелось бы найти функцию, которая каждому построенному прогнозу $\hat{y}(t)$ ставила бы в соответствие некоторое число $J \in \mathbb{R}$ – метрику ошибки прогноза. Тогда метод с наименьшим значением J был бы признан самым эффективным на данной обучающей выборке. Эту метрику J можно определить несколькими способами.

Можно посчитать $\hat{y} = \{y_1, y_2, \dots, y_m\}$ для каждого момента времени t_i в обучающей выборке, и определить ошибку d_i для каждого подсчитанного $\hat{y}_i$ как $d_i(y_i, \hat{y}_i) = |y_i - \hat{y}_i|$, либо как $d_i(y_i, \hat{y}_i) = |{y_i}^2 - {\hat{y}_i}^2|$. Тогда итоговую метрику можно взять просто как среднее арифметическое ошибок в каждой точке:

$$J(y, \hat{y}) = \frac{1}{m}\sum_{i}^{m} d_i(y_i, \hat{y}_i) = \frac{1}{m}\sum_{i}^{m} |{y_i}^2 - {\hat{y}_i}^2|$$

Такая оценка называется **ошибкой обучения**.

Но такая оценка не очень эффективна, так как на практике нам интересно узнать не насколько хорошо предсказанные данные соответствуют реальным данным, а насколько хорошо модель предсказывает данные в будущем. В самом деле, если использовать для оценки качества только эту метрику, то «победителем» всегда будет метод, генерирующий в точности значения y_i для t_i из обучающей выборки, его ошибка будет тождественно равна нулю. При этом ошибка прогноза этого метода может быть сколь угодно большой. Следовательно, для оценки качества метода прогнозирования, нужно накладывать ограничения на $\hat{y}(t)$ для значений t не только из обучающей выборки.

На практике применяется другой метод оценки качества прогноза, называемый **валидацией**. Он состоит в том, что выделяется определенный интервал из k дат в обучающей выборке (обычно из конца выборки, т.е. начиная с некоторого момента T и до конца данных) и исключается из обучающей выборки. Отсеченные пары значений (y, t) называются **валидационной выборкой**. Затем модель обучается на уменьшенном размере выборки. А потом, получив модель для $\hat{y}(t)$, считаются значения $\hat{y} = \{y_1, y_2, \dots, y_j, \dots, y_k\}$ для каждого значения t_j из валидационной выборки и определяется ошибка в момент времени t_j с использованием заранее оговоренной метрики $d_j(y_j, \hat{y}_j)$. Затем вычисляется среднее арифметическое k полученных ошибок:

$$J(y, \hat{y}) = \frac{1}{k}\sum_{j}^{k} d_j(y_j, \hat{y}_j)$$

Полученное таким образом значение $J(y, \hat{y})$ называется **ошибкой валидации**. Она показывает, насколько точный получился бы прогноз, если бы он был выполнен k периодов назад. Понятно, что реальное значение ошибки валидации зависит от выбора метрики d – функции, для каждого момента времени j в валидационной выборке определяющей «расстояние» между парой значений $(y_j, \hat{y}_j)$. Самый простой способ определить d_j – как модуль разности y_j и $\hat{y}_j$:

$$d_j(y_j, \hat{y}_j) = |y_j - \hat{y}_j|.$$

Тогда

$$J(y, \hat{y}) = \frac{1}{k}\sum_{j}^{k} d_j(y_j, \hat{y}_j) = \frac{1}{k}\sum_{j}^{k} |y_j - \hat{y}_j|$$

Ошибка J, полученная по такой формуле, носит название **средняя абсолютная ошибка (Mean Absolute Error, MAE)**. Понятно, что эта же формула может применяться и для вычисления ошибки обучения, и вообще для подсчета ошибки любого набора пар $(y_j, \hat{y}_j)$.

Способ подсчета ошибки валидации по алгоритму MAE можно использовать для оценки эффективности методов прогнозирования данных, он будет удовлетворять условию «чем меньше ошибка, тем эффективнее метод», но на практике у него есть один большой недостаток: он оперирует абсолютными значениями y, а значит, непригоден для сравнения прогнозов на разных обучающих выборках, и даже на разных участках одной и той же обучающей выборки. Этот недостаток заставляет отказаться от использования этого алгоритма в пользу других.

De Gooijer & Hyndman (2006)[2] дают обзор нескольким альтернативным метрикам подсчета ошибки прогноза. В частности, одна из самых популярных метрик носит название **Средняя Абсолютная Процентная Ошибка (Mean absolute percentage error, MAPE)**, и вычисляется по следующей формуле:

$$d_j(y_j, \hat{y}_j) = \left|\frac{y_j - \hat{y}_j}{y_j}\right|$$

$$J(y, \hat{y}) = 100\% \cdot \frac{1}{k}\sum_{j}^{k} d_j(y_j, \hat{y}_j) = 100\% \cdot \frac{1}{k}\sum_{j}^{k} \left|\frac{y_j - \hat{y}_j}{y_j}\right|$$

Полученная таким образом метрика, по сравнению с MAE, не зависит от обучающей выборки, так как считает не абсолютную, а относительную ошибку. Но и у неё есть недостатки:

- Она не определена для всех $y_j = 0$. Можно исключить все такие точки из рассмотрения (как обычно и делают при использовании этой метрики), но это снизит эффективность подсчета. Кроме того, для значений y_i, близких к 0, метрика получает неправдоподобно большую ошибку.
- Для прогнозов, ошибающихся в «меньшую» сторону, (т.е. для $\hat{y}_j$ в среднем меньших y_j), ошибка прогноза не может превышать 100%, но для прогнозов, ошибающихся в «большую» сторону она может расти бесконечно, в результате чего метрика является не симметричной.
- Метрика не объективно оценивает эффективность методов прогнозирования: она отдаёт предпочтения методам, занижающим прогнозы.

Vladik Kreinovich, Hung T. Nguyen и Rujira Ouncharoen (2014)[3] предлагают использовать альтернативную метрику для оценки эффективности прогнозов:

$$J(y, \hat{y}) = 100\% \cdot \frac{1}{k}\sum_{j}^{k} \frac{|y_j - \hat{y}_j|}{(|y_j| + |\hat{y}_j|)/2}$$

Эта метрика носит название **Симметричная Средняя Абсолютная Процентная Ошибка (Symmetric mean absolute percentage error, SMAPE)** и является более эффективной метрикой для оценки методов прогнозирования.

Она лишена проблемы неограниченного роста в «большую» сторону, которая была у MAPE: её значения находятся строго внутри отрезка $[0\%, 200\%]$. Более того, для лучшей интерпретируемости можно использовать видоизмененную версию оценки без деления на 2 в знаменателе:

$$J(y, \hat{y}) = 100\% \cdot \frac{1}{k} \sum_{j=0}^{k} \frac{|y_j - \hat{y}_j|}{|y_j| + |\hat{y}_j|}$$

Ее значения лежат в интервале от 0% до 100%.

Для значений $y_j = 0$ эта метрика равна 100%. Этот результат определен, в отличие от MAPE, но также может сильно исказить среднюю оценку, поэтому значения $y_j = 0$ обычно исключают из рассмотрения.

К сожалению, третья проблема MAPE так же актуальна и для SMAPE: вопреки своему названию, она не является идеально симметричной, но в этот раз она отдаёт предпочтения «завышающим» прогнозам: к примеру, для пары $(y_j = 100, \hat{y}_j = 110)$ она получает результат $J = 4.76\%$, а для пары $(y_j = 100, \hat{y}_j = 90)$ результат будет 5.26%.

Несмотря на свои недостатки, алгоритм SMAPE выдает приемлемый уровень качества как для сравнения методов прогнозирования на одной и той же, так и на различных обучающих выборках, поэтому далее при подсчете как ошибок валидации, так и ошибок обучения, будет использоваться именно он.

3 Анализ исходных данных

В УЦ ежедневно приходит огромное число заявок, и, какой метод для прогнозирования не был бы выбран, неэффективно строить прогноз для всех данных сразу, так как они имеют очень разнородную структуру. Имеет смысл сгруппировать эти данные по какому-либо признаку, чтобы данные в каждой группе имели схожие характеристики. И в дальнейшем прогноз выполнять отдельно для каждой группы. В качестве такого признака был выбран тип заявки. Тип заявки – это тип выпускаемого сертификата. Различные типы требуются различным группам людей, для различных нужд и в различное время. Поэтому очевидно, что заявки с различным типом будут различаться по характеристикам. Утверждение же, что все заявки одного типа имеют одинаковые характеристики, хоть и тоже не верно в общем случае, но для поставленной задачи такой аппроксимации достаточно. Для анализа данных был выбран один конкретный тип заявок и построен график обучающей выборки – количества пришедших заявок в каждый период времени в истории (Рисунок). В качестве длины периода взят один день: так как прогнозы по дням являются самыми востребованными и

самыми сложными из всех типов прогнозов. В дальнейшем длина периода обучающей выборки будет именоваться термином «**масштаб прогноза**».

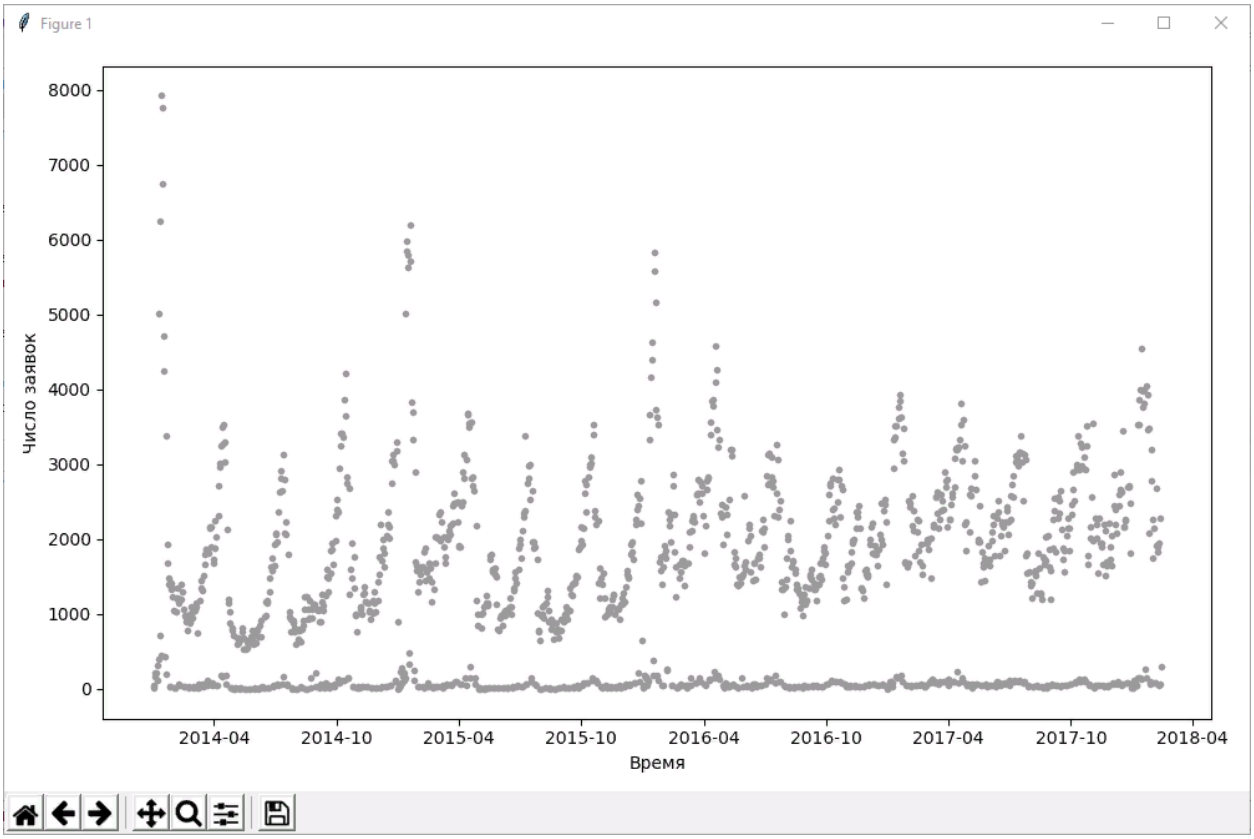


Рисунок 3.1. График обучающей выборки одного типа заявок

Из этого графика сразу можно выделить несколько закономерностей данных обучающей выборки:

- Наличие годовой и квартальной сезонности: можно явно увидеть рост данных при приближении отчетности, приходящейся на конец квартала, а также спад ближе к новому году.
- Наличие растущего тренда: график медленно, но явно, смещается вверх.

Также видно, что значения графика очень кучно расположены ближе к нулю, и очень разреженно в верхней половине графика. Поэтому чтобы лучше его визуализировать, полезно будет прологарифмировать все значения обучающей выборки (**Ошибка! Источник ссылки не найден.**)

На этом графике, как и на предыдущем, явно прослеживается два класса данных: условно, значения в одном классе среднем больше $e^6 \approx 104$, а во втором – меньше. Появилось предположение, что это следствие влияния недельной сезонности: данные в первом классе – количество форм, приходящих в будние дни, а меньше 104 – в выходные, когда

нагрузка резко снижается. Для лучшей визуализации раскрасим каждую точку в цвет дня недели, в который мы делали изменение (Рисунок 3.3).

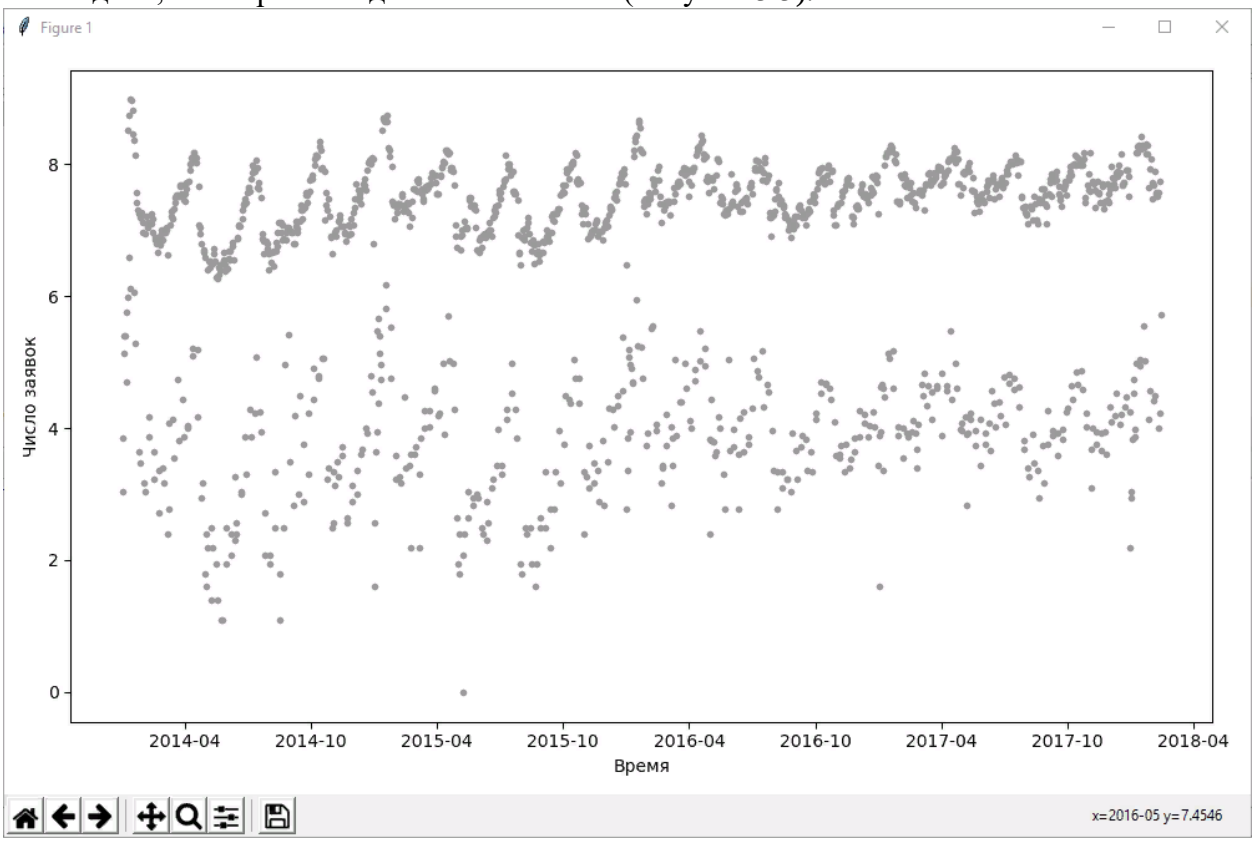


Рисунок 3.2. Прологарифмированный график обучающей выборки

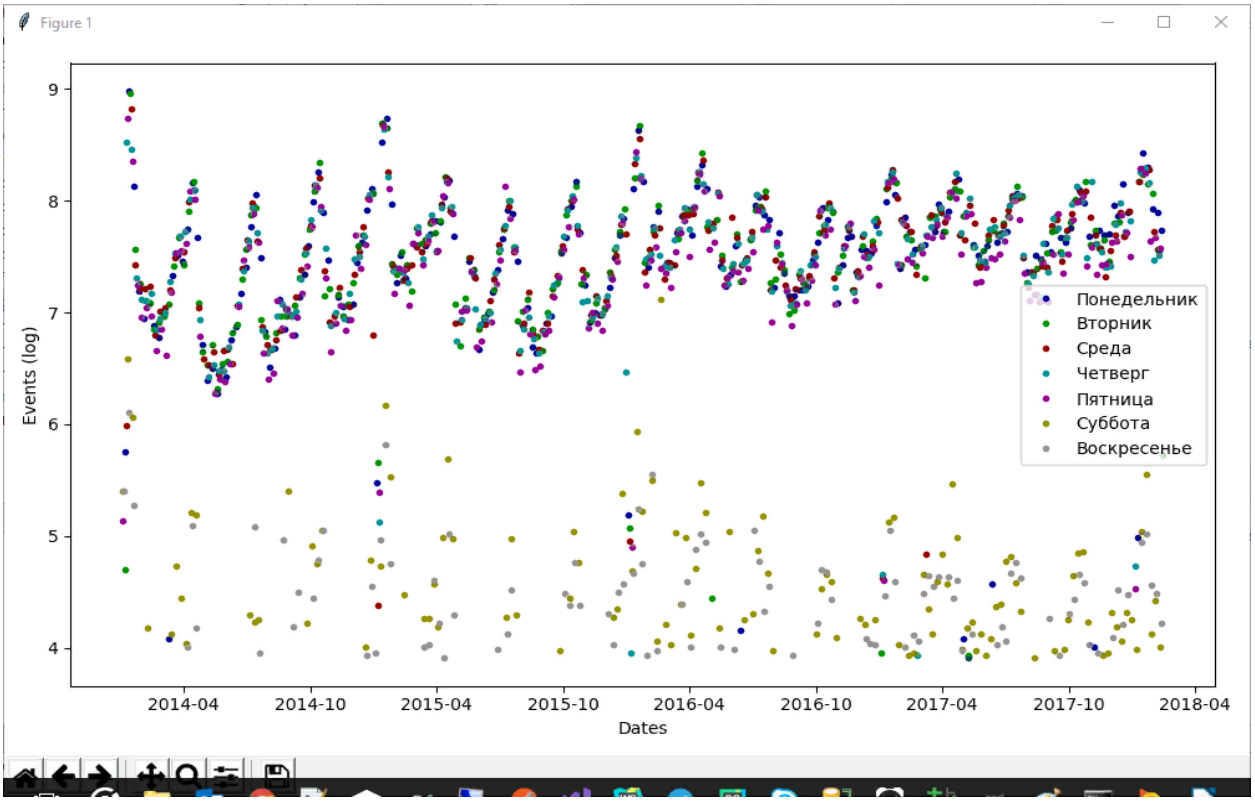


Рисунок 3.3. График с визуализацией различных дней недели

Нижняя часть графика агрегирует по большей части данные за субботу и воскресенье, в то время, как остальные дни недели находятся в верхней части графика, из чего можно сделать вывод о присутствии в данных недельного типа сезонности.

Если приглядеться, на графике можно заметить, что не все будние дни приходятся на верхнюю часть графика: часть из них «оседает» внизу, вместе с данными выходных дней. Это влияние ничего иного, как праздничных дней: в течение праздников нагрузка снижается практически до нагрузки выходных дней. Также есть и обратный процесс, хоть он и почти не заметен: часть точек выходных дней присутствует в верхней части графика. Это происходит из-за переносов рабочих дней на выходные в производственном календаре.

И, наконец, на этом графике явно наблюдается ещё один интересный факт: «верхняя» и «нижняя» часть графика, т.е. данные будней и выходных, колеблются с практически идентичной амплитудой в течение года. Это позволит нам в будущем сделать один важный статистический вывод о характеристике обучающих данных, а пока что просто оставим прологарифмированные данные для более удобной их визуализации.

4 Формальная постановка задачи

Проанализировав исходные данные, можно, наконец, дать строгое формальное определение поставленной задаче:

- Написать и реализовать алгоритм прогнозирования нагрузки на УЦ: сравнить различные методы прогнозирования и выбрать из них единственный, дающий минимальную ошибку.
- Выбранный метод должен учитывать различие входящих заявок по типам, и в каждом типе он должен уметь определять максимальное количество следующих особенностей:
    - Наличие тренда.
    - Влияние сезонных факторов.
    - Наличие динамически растущих и снижающихся рынков.
    - Снижение активности в праздничные дни.
    - Восприимчивость к насыщенности рынка.
- Разработанный сервис должен уметь автоматически определять эти особенности для каждого типа заявок и/или уметь принимать их «извне» (априори). Во

втором случае нужно разработать и реализовать удобный интерфейс для определения их из исторических данных и внесения их в модель прогнозирования.

- Определить и реализовать методы автоматической оценки эффективности настроенной модели прогноза
- Разработать, реализовать и протестировать на удобство все пользовательские сценарии работы менеджеров с написанным сервисом

5 Разбор и сравнение методов прогнозирования

Для сравнения методов прогнозирования использовалась та же самая обучающая выборка из 3 подраздела. С одним отличием: предварительно из нее были удалены аномальные данные - точки со слишком большими и слишком мелкими значениями. Некоторые из методов прогнозирования слишком восприимчивы к таким «выбросам». Предварительное их удаление позволит выбирать из более широкого спектра моделей. В дальнейшем можно будет точно так же дать аналитикам возможность удалять выбросы в интерфейсе сервиса для каждой обучающей выборки, так что это не станет проблемой.

Данные получившейся в результате обучающей выборки можно видеть на (Рисунок 5.1).

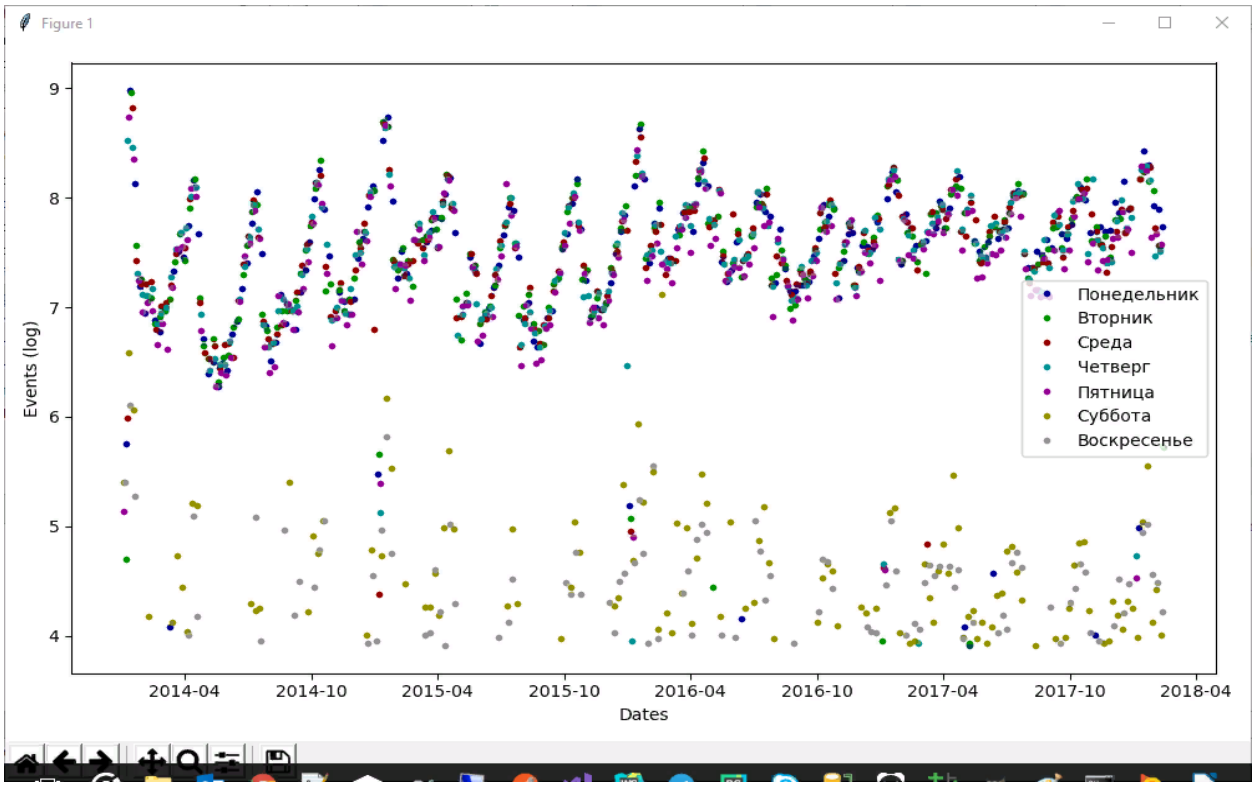


Рисунок 5.1. Обучающая выборка с удаленными выбросами

Рассмотренные методы включали в себя:

- **Полиномиальная регрессия**. Это расширение метода линейной регрессии. Он, также, как и алгоритм линейной регрессии использует градиентный спуск, чтобы приблизить обучающую выборку к некоторой функции. Разница с линейной регрессией состоит в том, что функция может быть не линейной, но любым полиномом, коэффициенты к которому определяются автоматически с помощью регуляризации. К сожалению, он не подходит для прогнозирования временных серий, так как не умеет адекватно реагировать на сезонность, его невозможно научить учитывать пиковые периоды, к тому же для полиномов большой степени он работает слишком долго.
- **Нейронные сети**. Проблема с ними состоит в том, что они плохо обнаруживают периодичность в данных и требуют длительной настройки. К тому же, нейронные сети намного хуже решают задачи регрессии, чем задачи классификации данных.
- **Случайное блуждание с наивной сезонностью**. Базовый метод для оценки всех остальных. В каждый момент времени значение $\hat{y}$ равно значению у один, или больше, сезонов назад. Этот метод очень напоминает те, что используются для прогнозирования входящей нагрузки аналитиками УЦ в текущий момент, поэтому он будет предоставлять хорошую «базовую» метрику для оценки всех остальных прогнозов.
- **auto_arima**. Базовый метод для прогнозирования временных серий с сезонностью. Метод auto_arima пакета **pyramid** для python обучает набор из нескольких моделей ARIMA (Autoregressive integrated moving average) и выбирает из них наилучшую.
- **ets.** Этот алгоритм из библиотеки **forecast** для языка **R** обучает набор моделей экспоненциального сглаживания и выбирает из них лучшую.
- **tbats** – алгоритм из той же библиотеки **forecast** языка **R** обучает модель TBATS с недельной и годовой сезонностью.

Результаты сравнения работы этих методов представлены на .

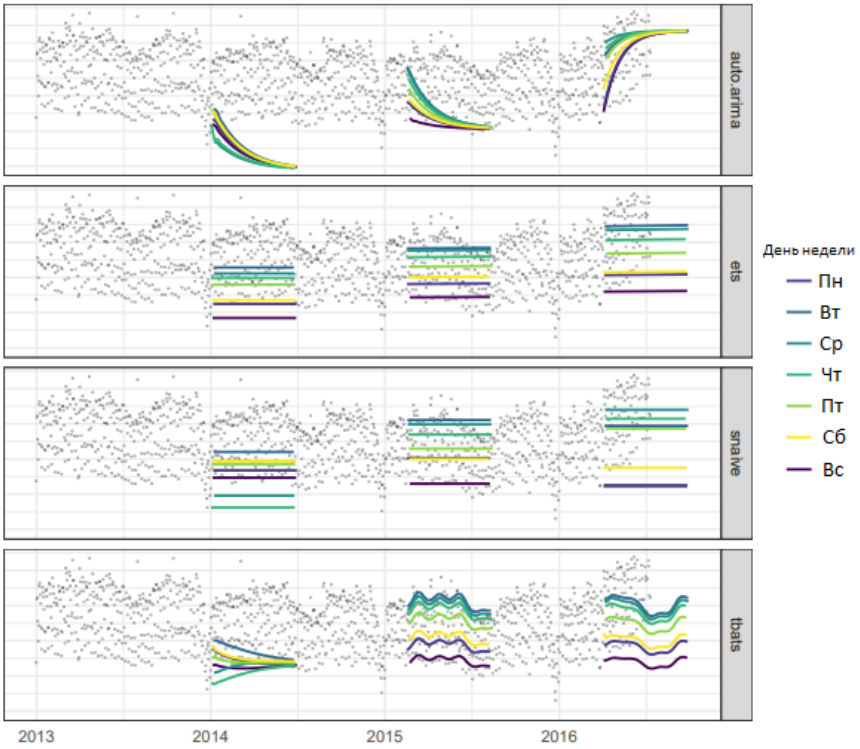


Рисунок 5.2. Сравнение различных методов предсказания временных серий[1]

Метод **auto_arima** показал хороший результат, но в случае, когда в конце обучающей выборки наблюдается сезонный спад (например, перед новым годом), он генерирует прогноз с слишком большой ошибкой.

Методы **«Экспоненциальное сглаживание»** и **«TBATS»** могут уловить недельную сезонность, но не годовую. Также они, как и ARIMA, намного сильнее, чем должны, реагируют на ежегодный спад тренда в конце года.

Также у вышеперечисленных методов есть один большой общий недостаток: их сложно корректировать не знакомому со статистическими терминами аналитику. К примеру, первый параметр в методе ARIMA – это максимальные порядки интегрированности, компоненты авторегрессии и компоненты скользящего среднего: их очень сложно обернуть в понятные аналитику величины, с которыми он может экспериментировать, чтобы уменьшить ошибку алгоритма. В итоге получается, что нужен человек, разбирающийся в устройстве метода/методов прогнозирования, который будет настраивать модели прогноза. В связи с этим возникают следующие проблемы:

[1] Здесь и далее из-за большого влияния недельной сезонности все графики прогнозов будут визуализироваться с разбиением по дням недели: из одной функции агрегируют семь новых, каждая из которых соответствует включает только точки в конкретный день недели.

- Среди факторов, влияющих на модель прогноза много таких, которые известны для аналитиков заранее, «априори». Среди них: наличие/отсутствие определенных типов сезонностей, насыщенность рынка, влияние определенных периодов в году на предсказываемое значение. В итоге для настройки модели становится необходимым постоянное сотрудничество между программистом и аналитиком. В такой ситуации проще и дешевле написать сервис, с помощью которого аналитик сам мог бы настраивать модель. Но из-за сложности интерпретации параметров модели это становится затруднительно.
- Параметры модели прогноза имеют свойство изменяться с течением времени. Соответственно, модели прогноза нужно тоже постоянно корректировать. Более того, рынок постоянно меняется, появляются новые типы сертификатов, соответственно, приходится постоянно создавать и настраивать модели.
- Необходимо, чтобы система умела сама себя поддерживать. Для этого она должна быть как можно более простой: если программист, создававший систему, покинет компанию, процесс передачи знаний по работе сервиса должен быть как можно более легким. Плюс система должна уметь автоматически обнаруживать ошибки и предлагать способы их устранения.

Учитывая вышесказанное, можно сделать вывод, что ни один из рассмотренных методов прогнозирования не может быть использован для решения поставленной задачи. Поэтому необходимо рассмотреть альтернативные методы прогнозирования, которые были бы намного проще в настройке, и при этом генерировали схожие по эффективности прогнозы. Один из таких методов реализован в библиотеке **Facebook Prophet**.

6 Обзор Facebook Prophet

6.1 Альтернативный подход к процессу прогнозирования

Facebook Prophet – библиотека, написанная на языках Python и R реализующая комплексный метод прогнозирования временных серий. От других схожих библиотек она отличается подходом к решению задачи. Если в схеме J. Scott Armstrong все этапы выполнялись дата-инженером со знаниями теории статистики и навыками программирования, то в модели Prophet пункты 3-6 выполняются специальным человеком с компетенциями аналитика: **«аналитик-в-контексте» (analysist-in-the-loop)** – аналитик, обладающий глубокими знаниями в предметной области осуществляемых прогнозов (Рисунок 6.1)

Рисунок 6.1. Модификация схемы J. Scott Armstrong

Более того, в схему добавляется ещё один этап, помогающий в масштабировании процесса прогнозирования: “автоматическая оценка методов”. В более общем виде схема выглядит следующим образом: (Рисунок 6.2)

Рисунок 6.2. Схема процесса прогнозирования Facebook Prophet

Эта схема работы отличается от традиционных автоматизированных методов прогнозирования временных серий, которые настраиваются редко, и исключительно людьми с широкими знаниями в области статистики. Эта схема заимствует многие черты у так называемых “оценочных” методов прогнозирования (judgmental forecasts) [4]. Но при этом этот метод оставляет многие процессы автоматизируемыми, что приближает его к традиционным статистическим моделям. В результате получившаяся модель находится примерно посередине между оценочными и статистическими, заимствую преимущества из обоих подходов.

Для обеспечения работы данной модели, библиотека предоставляет программный интерфейс для прогнозирования, который, с одной стороны, осуществляет автоматическое построение прогнозов с достаточно хорошей базовой эффективностью (без настройки), а также предоставляет набор понятных любому аналитику характеристик модели, регулирование которых позволяет ещё сильнее повысить эффективность прогнозов.

6.2 Оценка базовой эффективности модели

Чтобы положить начало оценке эффективности работы Facebook Prophet для нашей задачи, был построен график для той же обучающей выборке, что и на Рисунке 5.2. Получившийся результат и оценка прогноза показана на Рисунок 6.3.

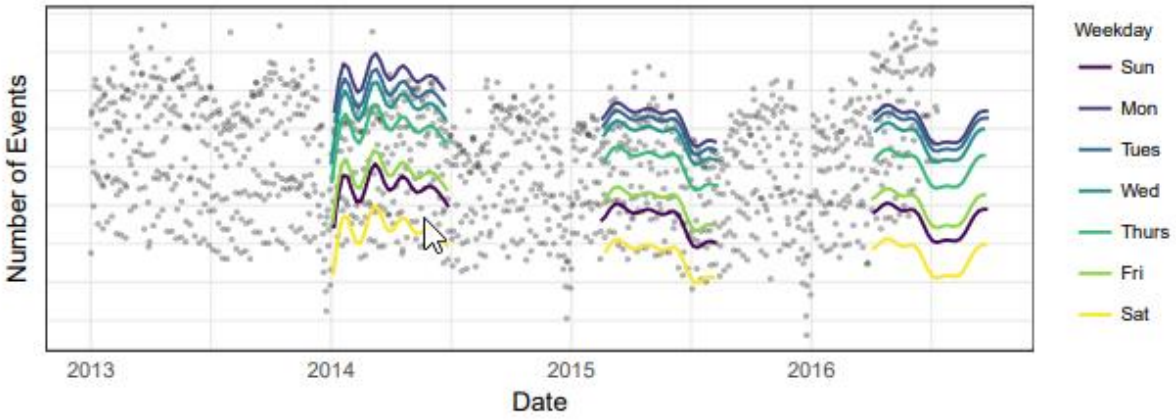


Рисунок 6.3. Визуализация прогнозов Facebook Prophet

Этот прогноз получился намного более реалистичным по сравнению с предыдущими.

6.3 Описание работы библиотеки

В своей основе библиотека использует декомпозируемую аддитивную модель временного ряда[5] с тремя главными компонентами: **трендом**, **сезонностью** и **праздничными днями**:

- **Тренд** может быть либо линейным, либо насыщаемым. В обоих случаях скорость роста тренда не постоянна и может меняться несколько раз, в точках, вычисляемых автоматически.

  Функция кусочно-линейного тренда:

  $$g(t) = (k + a(t)^T \vee \delta)t + (m + a(t)^T \vee \gamma)$$

  Функция логистического (насыщаемого) тренда:

$$g(t) = \frac{C(t)}{1 + exp\left(-(k + a(t)^T\delta)\left(t - (m + a(t)^T\gamma)\right)\right)}$$

Где C(t) – функция верхней границы тренда.

В обоих формулах присутствует $\boldsymbol{\delta}$ – вектор изменений скорости роста. $\boldsymbol{\delta \sim Laplace(0, \tau)}$. Параметр $\boldsymbol{\tau}$ – коэффициент масштаба, таким образом, отвечает непосредственно за гибкость алгоритма выбора тренда.

- **Сезонность** моделируется рядами Фурье:

$$s(t) = \sum_{n=1}^{N}\left(\alpha_n cos\left(\frac{(2\pi nt)}{P}\right) + b_n sin\left(\frac{(2\pi nt)}{P}\right)\right)$$

N – порядок ряда – задается вручную для каждого типа сезонности отдельно.

Компонент сезонности берется как

$\boldsymbol{s(t) = X(t)\beta}$,

где $\boldsymbol{X(t)}$ – матрица векторов сезонности, а $\boldsymbol{\beta \sim Normal(0, \sigma^2)}$.

Отсюда получаем параметр **σ** – коэффициент масштаба, напрямую влияющий на выраженность периодичности в нашей модели.

- **Праздничные дни.** Для каждого типа праздничного дня генерируется параметр k_i – среднее изменение в прогнозе для каждого дня этого типа в прошлом и затем формируется матрица регрессоров $Z(t) = [1(t \in D1), \ldots, 1(t \in DL)]$, показывающая, к каким праздничным дням относится текущий день. Компонент праздничных дней, таким образом:

$$h(t) = Z(t)\boldsymbol{\kappa},$$

где $\boldsymbol{\kappa} \sim Normal(0, \boldsymbol{\nu^2})$,

где $\boldsymbol{\nu}$ – коэффициент масштаба, влияющий на выраженность всех праздничных дней в прогнозе.

Кроме праздничных дней, можно также составить список **регрессоров** – компонент, представляющих собой длительные периоды в прошлом и будущем (в отличие от праздничных дней, которые модулируют краткосрочные «пиковые» изменения), и при этом могут иметь различную величину влияния для каждого дня действия.

Объединяя все три компонента, получаем аддитивную модель:

$$\boldsymbol{\hat{y}(t) = g(t) + s(t) + h(t) + \varepsilon_t}$$

Затем с помощью алгоритма L-BFGS (в библиотеке PyStan) эта модель обучается на входных данных, и на выходе мы получаем максимальную апостериорную оценку для всех компонент модели плюс разброс для компоненты тренда. Также есть возможность провести процедуру полного Байесовского вывода, чтобы получить неопределенность для всех компонент, но это увеличит время выполнение прогноза во много раз.

Примечательно также то, что после выполнения прогноза библиотека возвращает для анализа не только предсказанные значения, но и отдельно каждую из предсказанных компонент прогноза. Это позволяет визуализировать их на графике, чтобы лучше понять, какая из компонент требует улучшения (Рисунок 6.)

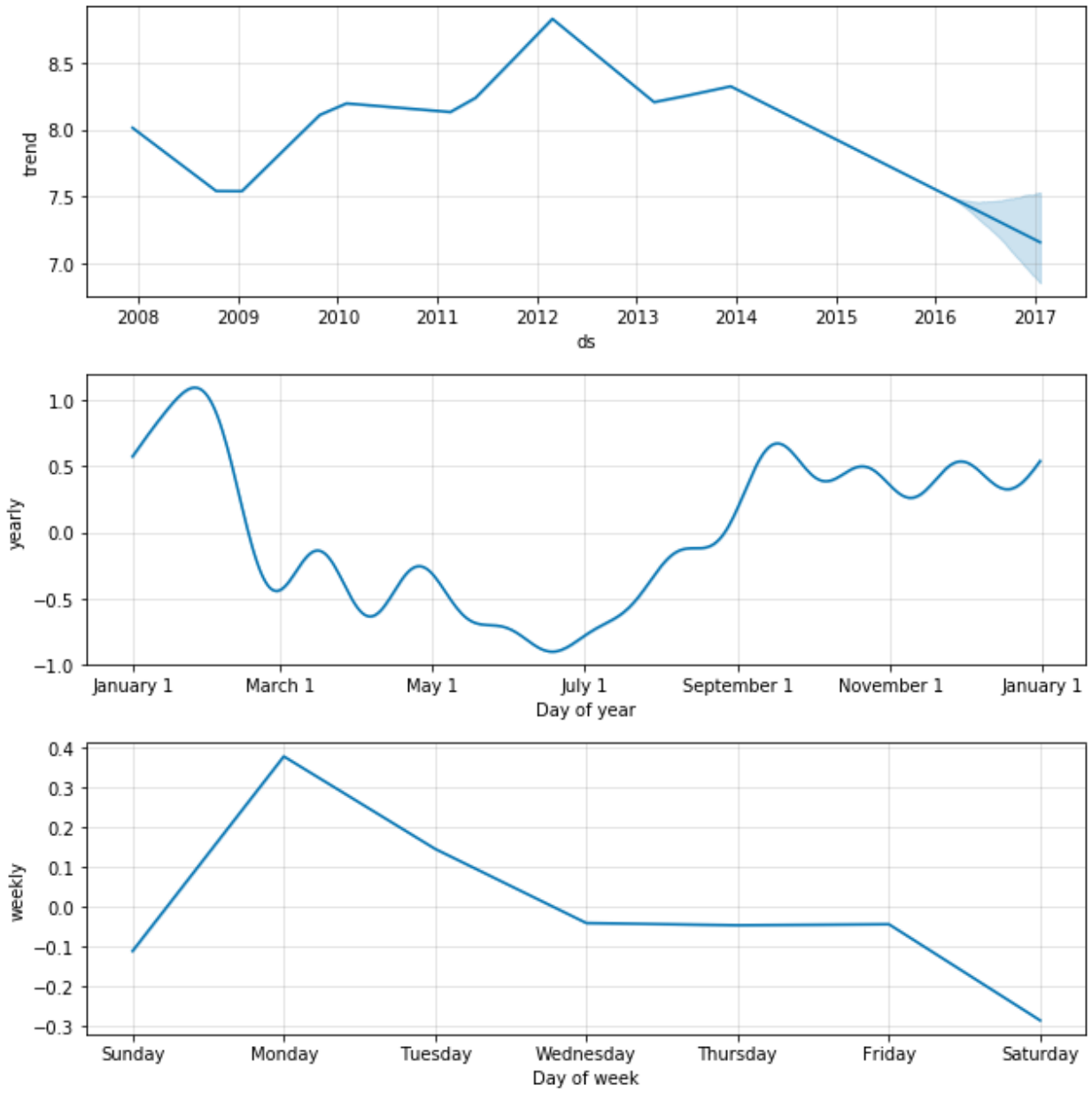


Рисунок 6.4. Просмотр отдельных компонент прогноза

6.4 Аддитивные и мультипликативные модели

Ранее было замечено, что годовые сезонные колебания в обучающей выборки имеют разную амплитуду для будних и выходных дней. В самом деле, если построить прогноз на исходной, не прологарифмированной выборке, мы увидим, что сезонные колебания получаются слишком слабыми для будних дней и слишком сильными для выходных (Рисунок 6.5).

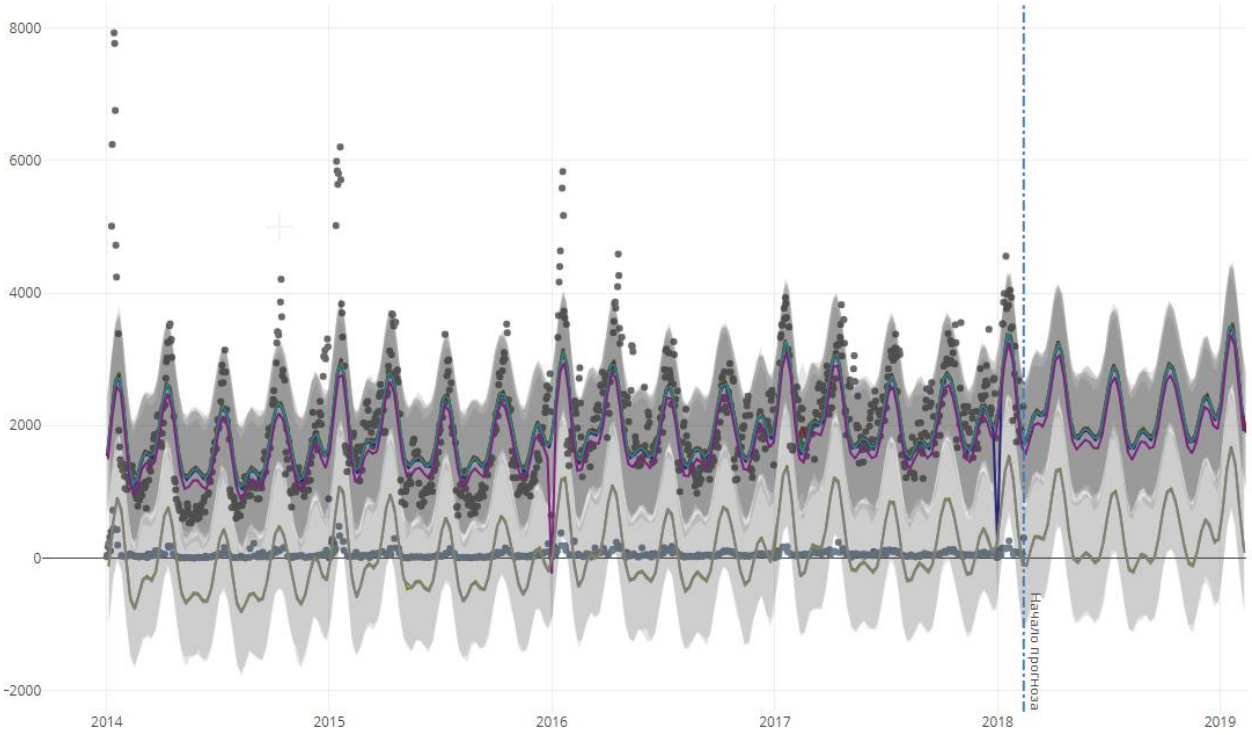


Рисунок 6.5. Прогноз для не логарифмированной выборки

Однако, если прологарифмировать данные, то сезонные колебания в обучающей выборке приобретут одинаковую амплитуду как для будних дней, так и для выходных (Рисунок).

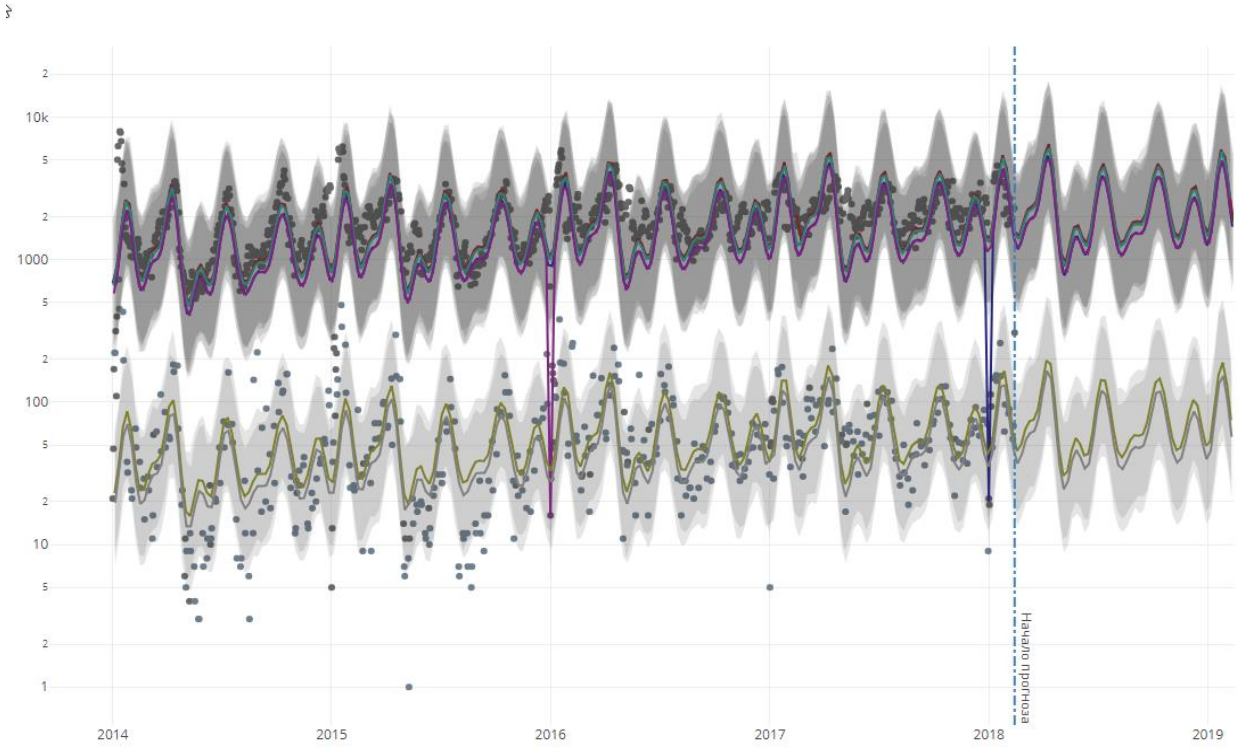


Рисунок 6.6. Прогноз для прологарифмированной выборки

Это не просто совпадение, а признак, указывающий на характер обучающей выборки. Это означает, что сезонность в обучающей выборке имеет не аддитивный:

$$\widehat{\boldsymbol{y}}(\boldsymbol{t}) = \boldsymbol{g}(\boldsymbol{t}) + \boldsymbol{s}(\boldsymbol{t}) + \boldsymbol{h}(\boldsymbol{t})^{1},$$

а мультипликативный характер:

$$\widehat{\boldsymbol{y}}(\boldsymbol{t}) = \boldsymbol{g}(\boldsymbol{t}) * \boldsymbol{s}(\boldsymbol{t}) * \boldsymbol{h}(\boldsymbol{t}),$$

где

$$s(t) = \prod_{i=1}^{N_S} s_i(t)$$

Иными словами, при изменении одной компоненты, прогноз меняется не линейно, а пропорционально.

[1] Для упрощения формул отбросим компонент неопределённости

Операция взятия логарифма из обучающей выборки меняет тип прогноза. Этому существует математическое объяснение.

Взяв логарифм значений $y(t)$ в обучающей выборки и обучив модель, мы получим три аддитивных компоненты прогноза:

$$\overline{\boldsymbol{y}}(\boldsymbol{t}) = \overline{\boldsymbol{g}}(\boldsymbol{t}) + \overline{\boldsymbol{s}}(\boldsymbol{t}) + \overline{\boldsymbol{h}}(\boldsymbol{t}),$$

где $\overline{\boldsymbol{y}}(\boldsymbol{t}), \overline{\boldsymbol{g}}(\boldsymbol{t}), \overline{\boldsymbol{s}}(\boldsymbol{t}), \overline{\boldsymbol{h}}(\boldsymbol{t})$ – аддитивная функция и её компоненты, полученные на прологарифмированной обучающей выборке.

$$\overline{\boldsymbol{g}}(\boldsymbol{t}) = \boldsymbol{log}\big(\boldsymbol{g}(\boldsymbol{t})\big),$$

$$\overline{\boldsymbol{s}}(\boldsymbol{t}) = \boldsymbol{log}\big(\boldsymbol{s}(\boldsymbol{t})\big),$$

$$\overline{\boldsymbol{h}}(\boldsymbol{t}) = \boldsymbol{log}\big(\boldsymbol{h}(\boldsymbol{t})\big),$$

где $\boldsymbol{g}(\boldsymbol{t}), \boldsymbol{s}(\boldsymbol{t}), \boldsymbol{h}(\boldsymbol{t})$ – компоненты, полученные при обучении на изначальной (не логарифмированной) выборке. $\widehat{\boldsymbol{y}}(\boldsymbol{t}) = \boldsymbol{g}(\boldsymbol{t}) + \boldsymbol{s}(\boldsymbol{t}) + \boldsymbol{h}(\boldsymbol{t})$

Применив свойства логарифма, получим:

$$\overline{\boldsymbol{y}}(\boldsymbol{t}) = \overline{\boldsymbol{g}}(\boldsymbol{t}) + \overline{\boldsymbol{s}}(\boldsymbol{t}) + \overline{\boldsymbol{h}}(\boldsymbol{t}) = \boldsymbol{log}\big(\boldsymbol{g}(\boldsymbol{t})\big) + \boldsymbol{log}\big(\boldsymbol{s}(\boldsymbol{t})\big) + \boldsymbol{log}\big(\boldsymbol{h}(\boldsymbol{t})\big)$$
$$= \boldsymbol{log}\big(\boldsymbol{g}(\boldsymbol{t}) * \boldsymbol{s}(\boldsymbol{t}) * \boldsymbol{h}(\boldsymbol{t})\big)$$

$$\overline{\boldsymbol{y}}(\boldsymbol{t}) = \boldsymbol{log}\big(\boldsymbol{g}(\boldsymbol{t}) * \boldsymbol{s}(\boldsymbol{t}) * \boldsymbol{h}(\boldsymbol{t})\big)$$

Применяя операцию экспоненцирования к обоим частям этого уравнения, получим:

$$\boldsymbol{e}^{\overline{\boldsymbol{y}}(\boldsymbol{t})} = \boldsymbol{g}(\boldsymbol{t}) * \boldsymbol{s}(\boldsymbol{t}) * \boldsymbol{h}(\boldsymbol{t})$$

Итого, если посчитать $\overline{\boldsymbol{y}}(\boldsymbol{t})$ – функцию прогноза на прологарифмированной обучающей выборки, а затем применить к ней операцию экспоненцирования, мы получим модель с мультипликативными компонентами, в частности, сезонностью. Этот алгоритм для превращения аддитивной модели в мультипликативную получил название “log trick” активно используется для прогнозирования временных серий с сезонностью.

Одинаковый вид годовой сезонности вне зависимости от дня недели указывает на мультипликативную зависимость годового и недельного трендов, а значит, прогноз нужно моделировать мультипликативной зависимостью компонент.

Применение Facebook Prophet для решения первоначальной задачи

Всего для каждой модели прогноза нам доступны для изменения следующие параметры:

- $\boldsymbol{C(t)}$ – верхняя граница тренда: если тренд имеет насыщаемый тип, нам необходимо указать верхнюю границу для каждой точки как в обучающей выборке, так и в прогнозе.
- $\boldsymbol{\delta}$ – точки смены тренда: если у аналитика имеется информация о датах, когда наблюдались серьезные изменения в тренде, есть возможность их указать, чтобы повысить эффективность прогноза. Если дат нет, они будут выбраны автоматически.
- $\boldsymbol{\tau}$ – коэффициент масштаба для автоматического выбора точек смены тренда. Чем больше τ – тем более гибким получается тренд, что приводит к меньшей ошибке в обучающей выборке, но и тем большую неопределенность мы получаем в будущем.
- **Сезонность**: для каждой модели можно указывать произвольное число компонентов сезонности с разными интервалами и порядком ряда Фурье.
- $\boldsymbol{\sigma}$ – коэффициент масштаба сезонности. Чем больше коэффициент, тем больше модель подвержена сезонным изменениям.
- **Праздничные дни**: дни, когда в прошлом наблюдался серьезный спад/подъем модели, можно обозначить как «праздничные», чтобы автоматически делать поправку на них в будущем.
- $\boldsymbol{\nu}$ – коэффициент масштаба праздничных дней. Чем он больше, тем сильнее праздничные дни в будущем будут влиять на прогноз.
- Также есть особый параметр, который не используется в модели прогнозирования напрямую, но тоже влияет на точность прогноза. Это выбросы – точки, значения в которых являются аномальными вследствие влияния случайных факторов, которые не следует учитывать в модели. Модель Prophet очень устойчива к

«пробелам» в обучающей выборке, поэтому удаление выбросов не станет для ней проблемой, и нам не нужно будет интерполировать данные на месте пропусков, как в случае с некоторыми другими моделями.

Набор этих параметров (вместе с выбросами) в дальнейшем будем называть **Моделью прогноза**.

Вернёмся к формулировке задачи. Теперь можно каждому требованию к сервису прогнозирования проставить в соответствие способ реализации и моделирования этого требования с помощью библиотеки Facebook Prophet:

- **Наличие тренда** моделируется настраиваемой компонентой тренда.
- **Влияние сезонных факторов** моделируется настраиваемыми компонентами сезонности.
- **Наличие динамически растущих и снижающихся рынков** моделируется точками смены тренда и, частично, регрессорами.
- **Снижение активности в праздничные дни** моделируется компонентой праздничных дней.
- **Восприимчивость к насыщенности рынка** моделируется компонентой логистического тренда с указанием насыщенности рынка.

7. Определение структуры сервиса прогнозирования

Исходя из вышесказанного, можно сделать вывод, что сервис прогнозирования должен состоять из двух частей:

Сервис осуществления прогнозов и **сервис настройки модели**. Первый должен осуществлять все операции, связанные с прогнозированием, валидацией (процесс валидации описан в Методы оценки эффективности методов прогнозирования); второй же должен хранить и предоставлять удобный и понятный пользовательский интерфейс для редактирования моделей прогнозирования.

Пользователь напрямую общается только с сервисом настройки модели. Упрощённая пользовательская модель взаимодействия пользователя с сервисом прогнозирования представлена на (Рисунок)

Рисунок 7.1. Упрощённая пользовательская модель

Для качественной работы сервиса прогнозирования мало просто предоставить интерфейс для настройки параметров модели, даже если эти параметры интуитивно понятны. Нужно также дать возможность сохранять эти параметры, чтобы не настраивать их каждый раз заново. Также нужно предоставить какой-то интерфейс для оценки ошибки модели, чтобы аналитик мог понять, насколько полученная модель хорошо настроена, и, если нет, то что необходимо в ней изменить.

7.1 Классы прогнозирования

В начале работы было сделано допущение, что заявки одинакового типа имеют примерно одинаковые параметры модели прогноза. На самом деле это предположение можно расширить: можно выделить целые классы типов заявок с одинаковыми моделями прогноза. Тогда при проведении валидации и настройки модели можно использовать обучающую выборку, равную сумме значений обучающих выборок, входящих в класс прогнозирования. Очевидно, что это сэкономит большое количество времени, которое бы тратилось

на дублирование одинаковых моделей прогноза для разных типов заявок. Эти классы эквивалентности по признаку совпадения моделей прогнозирования были названы **классами прогнозирования**. Классы прогнозирования – основная структура данных сервиса прогнозирования.

Вместе с аналитиками была проведена работа по объединению типов заявок в классы прогнозирования. В итоге все типы заявок получилось разбить всего на два класса (Таблица 7.1).

Класс прогнозирования	Сезонность	Тренд
Сертификаты УЦ	Годовая, недельная	Не насыщаемый
Сертификаты для сервисов Контура	Годовая, квартальная, недельная	Насыщаемый

Таблица 7.1. Классы прогнозирования

7.2 Масштаб прогноза

В требованиях к сервису прогнозирования присутствует необходимость строить прогнозы в разном масштабе, например, для прогноза на год, лучше делать прогноз по дням, в то время как для прогноза на день/неделю точно нужно разбивать прогноз по часам. И некоторые параметры класса прогнозирования изменяются при переходе от масштаба по дням к масштабу по часам. В самом деле, выбросы и точки смены тренда – это теперь точки на почасовой шкале, а в модели появляется ещё один тип сезонности: дневная. Поэтому был сделан вывод, что в рамках одного класса некоторые параметры модели нужно хранить отдельно для каждого масштаба. Масштаб прогноза, как и класс прогнозирования, выбирается пользователем перед переходом к редактированию модели.

7.3 Процесс валидации

Во второй главе был описан алгоритм оценки прогноза при помощи алгоритма валидации. Схема работы этого алгоритма:

1. Отсекаем часть обучающей выборки, объявляем отсеченную часть **валидационной выборкой.**
2. Обучаем модель на уменьшенной обучающей выборке.

3. Строим прогноз для значений из валидационной выборки.
4. Считаем ошибку как разницу между значениями валидационной выборки и предсказанными значениями по метрике SMAPE.

Но для аналитика мало лишь обнаружить ошибку. Нужно также позаботиться о том, чтобы пользователь мог понять, что именно в модели прогноза нуждается в улучшении, чтобы прогноз был более эффективным, и генерировал меньшую ошибку.

Прежде, чем переходить к методам определения способов модели, важно разбить все ошибочные модели на два класса: **недоубученные (underfitted) модели** и **переобученные (overfitted) модели**. Также их называют моделями с **высоким предубеждением (high bias)** и **высокой вариацией (high variance)** соответственно. Недообученные модели генерируют высокие ошибки как обучения, так и валидации. Это означает, что в модели недостаточно сведений, чтобы аккуратно смоделировать предсказываемый процесс. Обычно это исправляется добавлением дополнительных параметров в модель. Переобученные модели же генерируют маленькую ошибку обучения, но большую ошибку валидации. Это означает, что модель слишком сильно старается повторить форму обучающей выборки вместо того, чтобы выделить из неё некоторую общую модель. Типичные способы борьбы с переобучением – удаление лишних параметров из модели и увеличение размера обучающей выборки (больше точек – более общую модель по ним получается построить).

Таким образом, при наблюдении слишком большой ошибки валидации, нужно сначала обратить внимание на ошибку обучения, и исходя из её величины определить, пере- или недообучена модель и только потом принимать решения о способе её модификации.

Для определения способа корректировки модели в процессе валидации визуализируются следующие вещи:

- Величина ошибки обучения и ошибки валидации, посчитанные методом SMAPE. На их основе принимается заключение о недо- или переобученности модели.
- График обучающей выборки с наложенным на него графиком прогноза как в «прошлом» (в обучающей выборке), так и в «будущем» (в валидационной выборке.

- Графики всех компонент прогноза: тренда, сезонностей и праздничных дней, позволяющих дать понятие о влиянии параметров модели на свои соответствующие компоненты прогноза.

Примерный алгоритм принятия решений о корректировки модели на основе графиков валидации представлен на (Рисунок 2)

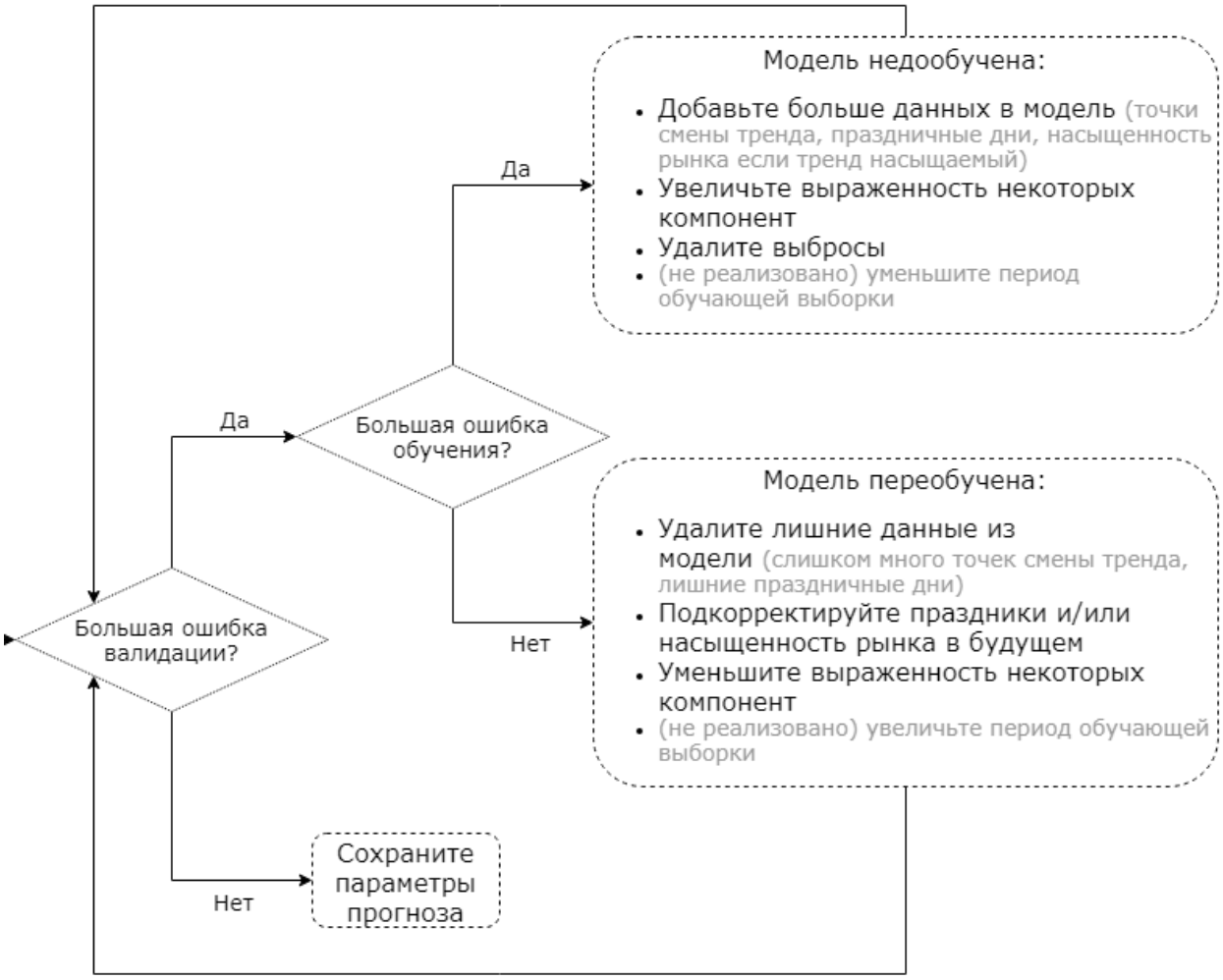


Рисунок 2.2. Алгоритм принятия решения о модификации модели на основе данных валидации.

Также, сравнивая прогноз с обучающей выборкой, можно быстро принять решение о добавлении (или удалении) в модель новых точек смены тренда, о наличии насыщаемого роста (и примерной насыщенности), о реальном влиянии конкретных праздников и сезонных компонент.

7.4 Процесс кросс-валидации

Для масштабирования сервиса прогнозирования необходимы методы автоматического определения моделей с большой ошибкой. Для этого был реализован алгоритм симуляции исторических прогнозов, или «кросс-валидации». Суть его состоит в том, что одна и та же модель используется для составления нескольких прогнозов в прошлом. Затем считаются отдельные ошибки валидации для каждого прогноза, а также средняя ошибка, равная среднему арифметическому ошибок всех прогнозов. Прогнозы визуализируются на графике для более удобного определения причины ошибки.

Формально метод описывается следующим образом:

на вход поступает три параметра:

- h – горизонт прогнозирования – длина временного интервала, на который необходимо получить прогноз.
- (опц.) t – длина тренировочного периода – длина временного интервала обучающей выборки. По умолчанию равен $h * 3$.
- (опц.) c – временное расстояние между датами отсечения. Дата отсечения – дата, на которой заканчивается обучающая выборка. По умолчанию равно $h/2$.

Пусть обучающая выборка для алгоритма охватывает временной период $[T_s;\ T_f]$.

Описание алгоритма:

1. Сначала формируем первую точку отсечения как $T_f - h$. Период прогнозирования таким образом становится равен $[T_f - h; T_f]$. Тренировочный период же равен $[T_f - (h + t); T_f - h]$.
2. Прогнозируем данные на тренировочный период, считаем ошибку.
3. Пусть $k := 1$ – количество сдвигов графика (в точности равно количеству точек отсечения на графике минус 1).
4. Пока $T_s \leq T_f - (h\ +\ t\ +\ (k\ *\ c))$:

    4.2 Сдвиг $\tau := k * c$.

    4.3 Период прогнозирования: $[T_f - h - \tau;\ T_f - \tau]$.

    4.4 Тренировочный период: $[T_f - (h + t + \tau);\ T_f - (h + \tau)]$.

    4.5 Прогнозируем данные на тренировочный период, считаем ошибку

    4.6 $k :=\ k\ +\ 1$.

После завершения работы алгоритма мы получаем $k + 1$ спрогнозированных временных интервалов, для каждого из которых мы можем подсчитать ошибку.

Этот алгоритм можно также выполнить несколько раз для различных горизонтов прогнозирования, чтобы более полно оценить точность модели. Но эта возможность пока не реализована.

7.5 Структура сервиса прогнозирования

Как уже было отмечено ранее, сервис прогнозирования был разбит на две части: **сервис осуществления прогнозов** (далее – **сервер прогнозирования**) и **сервис настройки модели.** На самом деле классификацию можно продлить, разбив сервис настройки модели на серверную и клиентскую часть. Получившаяся архитектура представлена на Рисунок 7.33.

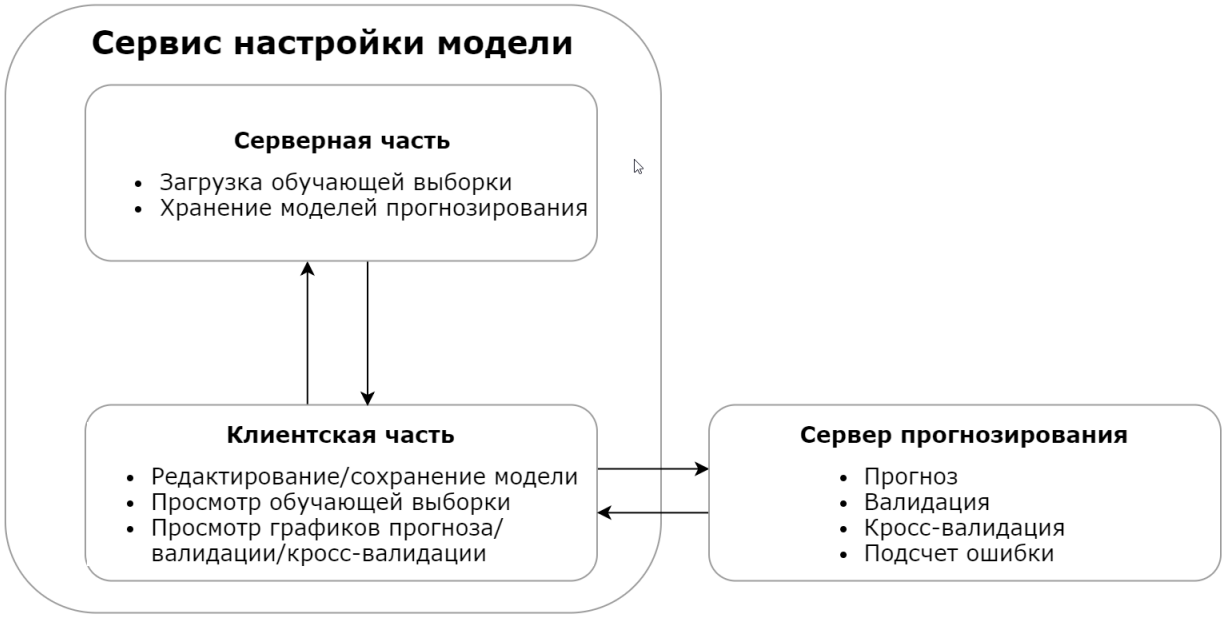
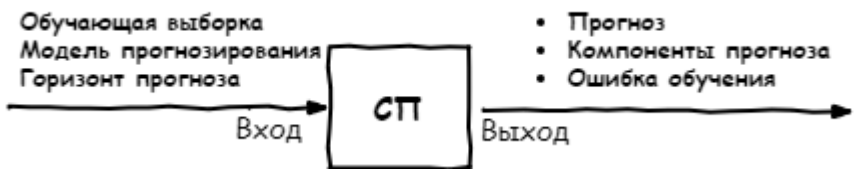


Рисунок 7.3. Алгоритм принятия решения о модификации модели на основе данных валидации.

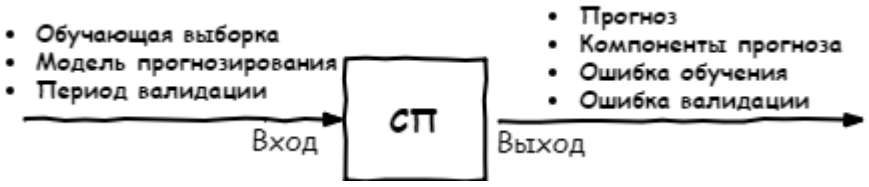
Рассмотрим подробнее каждую составляющую часть сервиса прогнозирования.

7.5.1 Сервер прогнозирования

- **Прогноз:**



- **Валидация:**



- **Кросс-валидация:**

7.5.2 Сервис настройки модели: серверная часть

Сервис настройки модели хранит данные моделей прогнозирования и обучающие выборки для каждого класса и масштаба прогнозирования. Данные хранятся в реляционной базе данных. Взаимоотношения между отношениями в базе представлены на (Рисунок

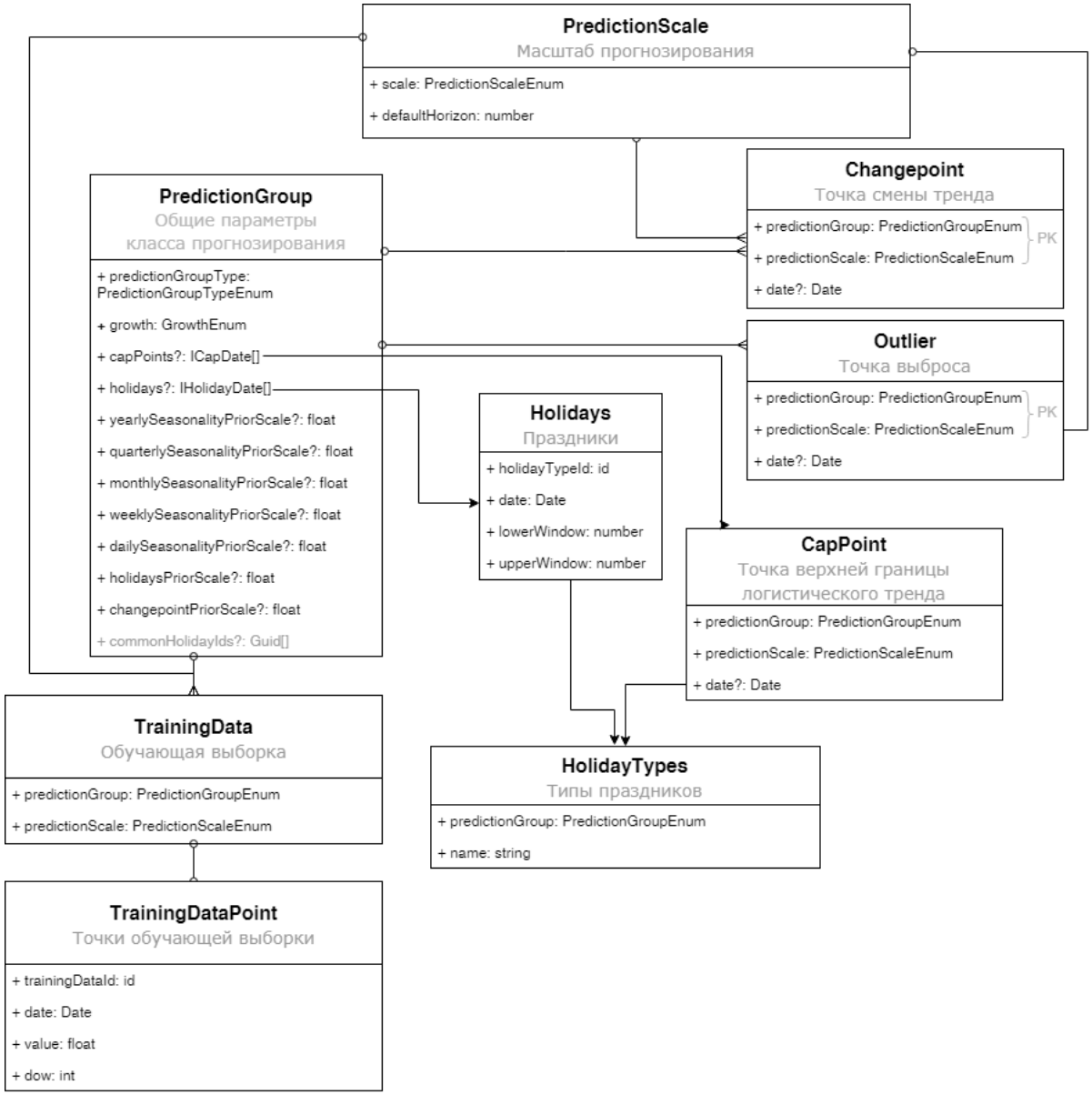


Рисунок 7.4. Структура отношений в базе данных сервиса настройки модели.

Как видно из схемы, параметры Changepoints (точки смены тренда), CapPoints (значения насыщенности тренда) и Outliers (выбросы) хранятся отдельно для разных типов масштаба прогноза. Вместо значений насыщенности тренда для каждой даты в обучающей выборке хранится лишь малое количество значений для отдельно выбранных дат. Также видно, что типы праздников хранятся отдельно от самих праздников, а также то, что каждый тип праздника может принадлежать какому-либо классу прогнозирования либо быть общим типом. В таком случае он включается в каждую модель прогнозирования.

Также сервис настройки модели предоставляет API для сохранения и загрузки моделей прогнозирования и обучающей выборки для каждого класса прогнозирования.

7.5.3. Сервис настройки модели: клиентская часть

Клиентская часть сервиса настройки модели – это единственная часть, с которой непосредственно взаимодействуют пользователи. Она позволяет выполнять следующие задачи:

- Загрузка и визуализация обучающей выборки для каждого класса и масштаба прогнозирования.
- Визуализация прогнозов, валидации и кросс-валидации выбранного класса прогнозирования.
- Предоставления интерфейса для редактирования модели прогноза, т.е.:
    - Удаление выбросов.
    - Редактирование параметров, связанных с компонентой тренда: типа тренда, емкости тренда для тренда насыщаемого типа и точек смены тренда.
    - Редактирование компонент сезонности.
    - Редактирование праздничных дней.
- Наложение параметров модели прогноза на обучающую выборку для более точной их настройки (к примеру, визуализация выбросов или подсвечивание праздничных дней).
- Сохранение отредактированной модели на сервер.

Все задачи выполняются прямолинейно и не заслуживают отдельного описания, кроме одной: процесса установки верхней границы насыщаемого тренда.

Для тренда логистического типа во временной серии для каждой даты необходим параметр ёмкости, отражающий верхнюю границу тренда в эту дату. Причём он нужен как во временном ряде обучающей выборки, так и во временном ряде предсказания. Для установки этих границ в интерфейсе был придуман следующий алгоритм:

- Используем виджет для редактирования временного ряда с обязательным доп. параметром ёмкости для каждой даты.
- Если во временном ряде указана одна дата, значение ёмкости из этой даты считается значением ёмкости для каждой даты во временном ряде.
- Для двух и более дат для вычисления значения ёмкости мы проводим отрезок между каждыми двумя соседними точками на графике, для левой и правой

точки отрезок дополняется до луча. Значения ёмкости интерполируются как значения отрезка в этой точке.

Экстраполированные значения ёмкости визуализируются на графике обучающей выборки.

Структура общения “Клиент-сервер представлена на Рисунке 7.5

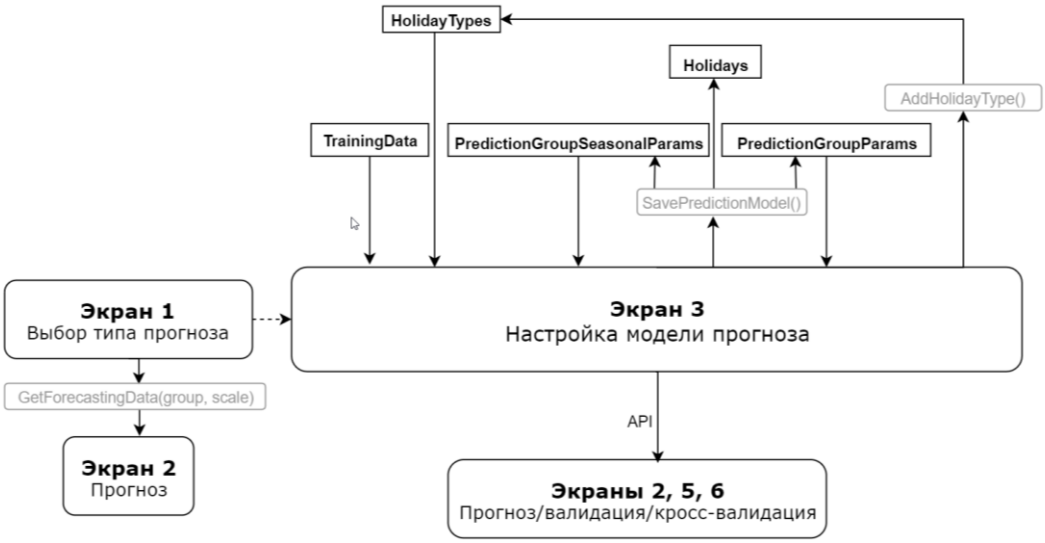


Рисунок 7.5. Структура общения клиент-сервер

Скриншоты интерфейса сервиса настройки модели приведены в Приложении 1.

7.5.4. Техническое описание компонентов сервиса настройки модели

Сервер прогнозирования написан на языке **Python**. Он имеет формат REST Web-API сервера. Для реализации Web-API был использован фреймворк **Flask**. Спецификациии методов хранятся на сервисе Swagger в формате Open API 2.0. Для операций с данными используются Python-библиотеки **scikit-learn**, **pandas** и **numpy**. Для визуализации графиков в процессе разработки использовалась библиотека **matplotlib**.

Серверная часть сервиса прогнозирования написана на фреймворке **ASP.net**. В качестве базы данных используется реляционная база данных **MS SQL**. В качестве движка для общения с базой данных используется **Entity Framework**.

Клиентская часть сервиса прогнозирования написана с использованием библиотеки **React**. Визуализация графиков осуществляется с помощью библиотеки **Plotly.js**. Для управления данными используется state-manager **Redux**. В качестве библиотеки контролов используется внутренняя библиотека контролов СКБ Контур «**React-UI**».

8. Настройка модели и сравнение результатов

Для проверки эффективности работы разработанного метода, были предварительно настроены модели для обоих классов прогнозирования. Модели настраивались без участия аналитиков, исключительно по визуальным данным графика и априорным сведениям о каждом классе прогнозирования, собранных во время их настройки. Прогноз проводился с помощью трех моделей:

- Наивный сезонный подход – предположение, что данные в любой момент времени в точности равны данным один сезон (в данном случае за сезон брался один год) назад
- Прогноз **Facebook Prophet** с моделью по умолчанию
- Прогноз **Facebook Prophet** с заранее настроенной моделью

Данные проверялись процессом валидации. Для прогнозов Facebook Prophet производилась также кросс-валидация данных.

Результаты этих сравнений для двух классов прогнозирования представлены на таблице 8.1

Класс прогнозирования	**Наивный подход**	**Prophet без настройки**	**Prophet с настройкой**
Сертификаты УЦ	43.97%	35.17% к/в: 39.28%, 26.27%, 23.88%	31.1% к/в: 35.34%, 28.17%, 21.68%
Сертификаты для сервисов Контура	32.21%	35.17% к/в: 22.42%, 20.68%, 18.93%	32.62% к/в: 22.79%, 20.33%, 18.81%

Таблица 8.1. Ошибки различных методов прогнозирования

Сами графики построенных прогнозов изображены на Рисунке 8.1

Из этих данных можно сделать несколько выводов:

- В среднем ошибка Facebook Prophet намного меньше ошибки наивного метода. Но иногда это правило нарушается. Это происходит из-за того, что модель была неэффективно обучена: корректировка параметров приведет к дальнейшему уменьшению ошибки
- Ошибки кросс-валидации в среднем меньше ошибок валидации. Этот, хоть и парадоксальный факт объясняется тем, что валидация использует всю обучающую вы-

борку, а кросс-валидация – только самую актуальную её часть. Если для большинства методов машинного обучения увеличение обучающей выборки может привести только к уменьшению ошибки, для методов прогнозирования временных серий это не так. Чем ближе находятся данные к моменту начала прогноза, тем более они актуальные, поэтому включение в прогноз более «старых» данных может привести как к улучшению, так и к ухудшению прогноза, в зависимости от ситуации. Отсюда получаем вывод, что для валидации и прогнозирования необходимо использовать только часть обучающей выборки по длине сравнимой с горизонтом прогнозирования.

- Ошибки даже на слабо настроенной модели меньше, чем на не настроенной. Значит, дальнейшая настройка модели приведет к ещё более низким ошибкам и большей эффективности прогноза.

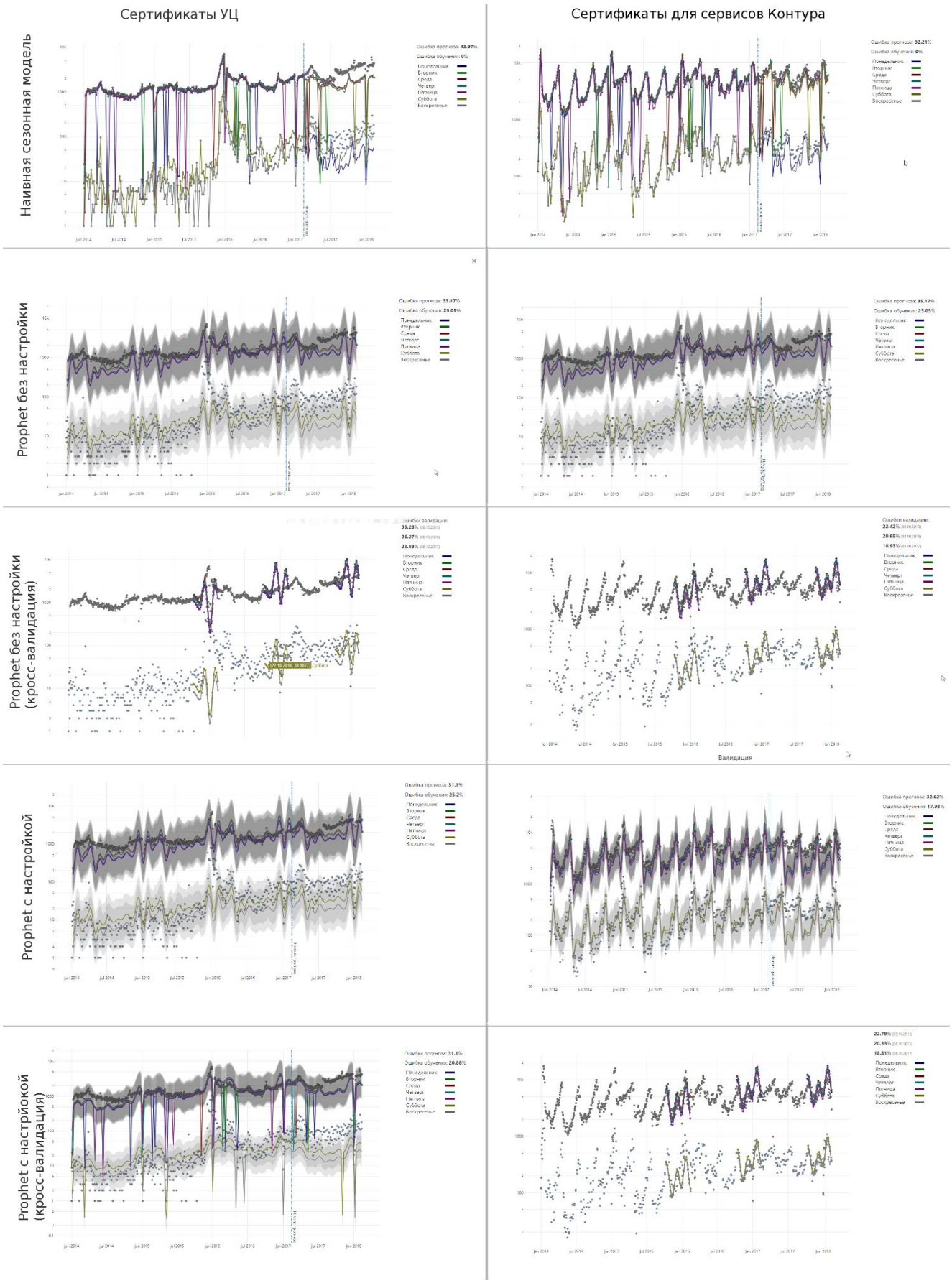


Рисунок 8.13. Сравнение графиков прогнозов

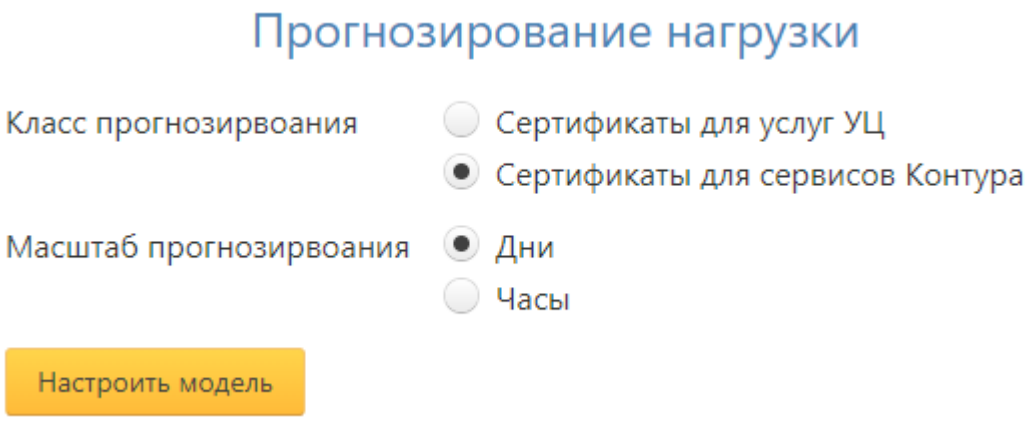
ЗАКЛЮЧЕНИЕ

В процессе решения задачи были проанализированы различные методы прогнозирования временных серий и выбран единственный, наиболее полно удовлетворяющей поставленным требованиям. Был разработан и реализован прототип многофункционального сервиса для прогнозирования временных серий, в частности, данных по нагрузке на удостоверяющий центр. Была разработана модель взаимодействия с сервисом как в краткосрочном, так и в долгосрочном плане. Был сделан вывод о необходимости разработки интуитивно понятного пользовательского интерфейса для настройки сервиса. Была разработана внутренняя структура сервиса, и структура пользовательского интерфейса, основная цель которого – максимальное облегчение процесса настройки модели и наблюдения динамики прогнозов. Был разработан сервер прогнозирования и сервис настройки модели с использованием современных языков и технологий программирования. Были реализованы методы как автоматической, так и визуальной оценки эффективности настроенной модели прогноза. С использованием разработанного интерфейса были настроены два класса прогнозирования, и затем была произведена повторная оценка эффективности реализованных методов. Основные выводы из проделанной работы можно сформулировать следующим образом:

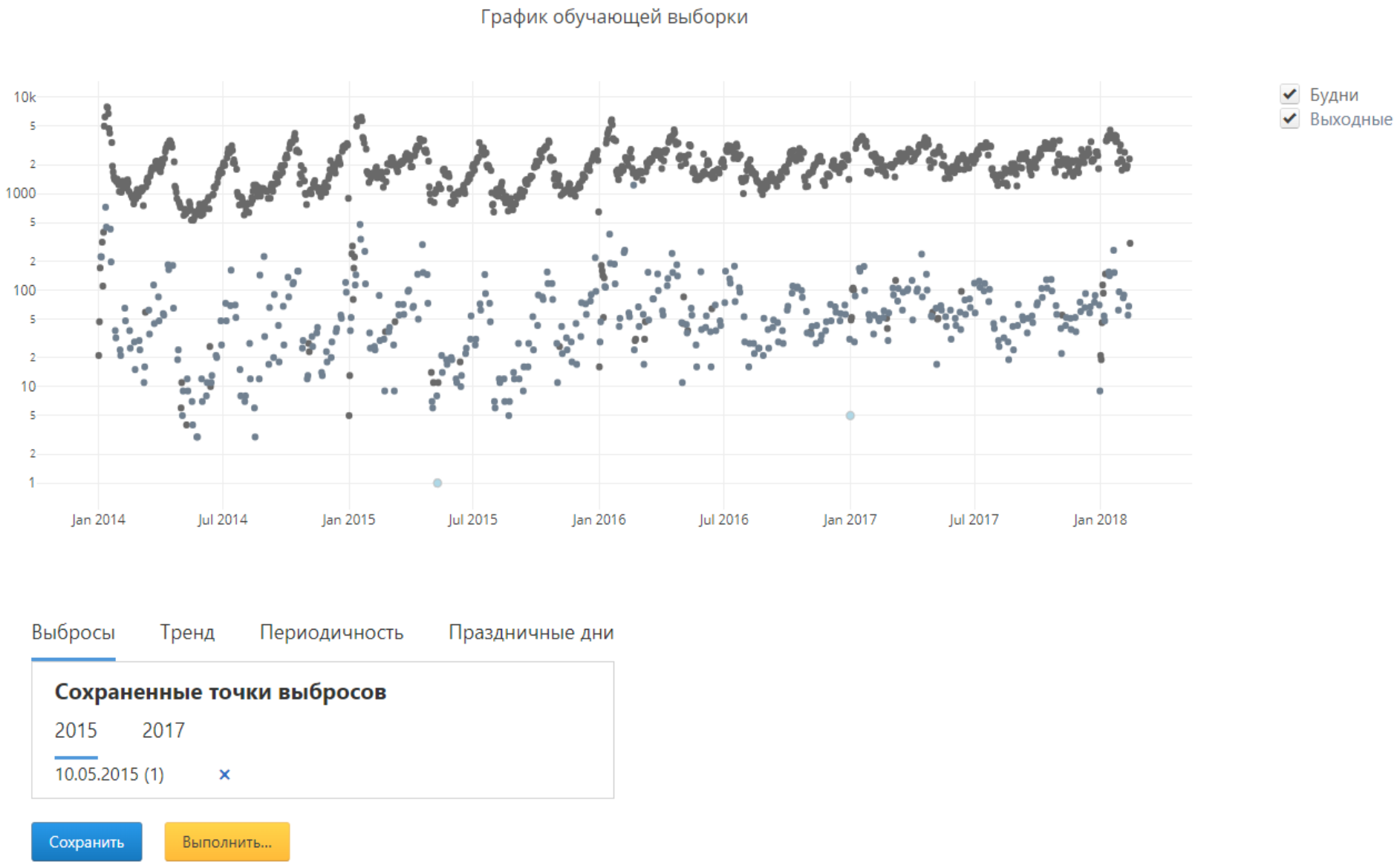
- Нагрузка на Удостоверяющий Центр имеет слишком разнородную структуру, чтобы эффективно её прогнозировать ручными методами
- Библиотека Facebook Prophet лучше других методов прогнозирования подходит для решения задачи прогнозирования данных временных серий
- Для получения более эффективных прогнозов необходимо участие в процессе аналитиков с большими знаниями в предметной области.

Полученные выводы и созданный инструмент в дальнейшем можно использовать для прогнозирования большого числа временных серий, имеющих самую разную природу и характеристики. Полученный сервис можно превратить в отдельный внутренний продукт, сочетающий в себе высокую гибкость настройки с простотой использования.

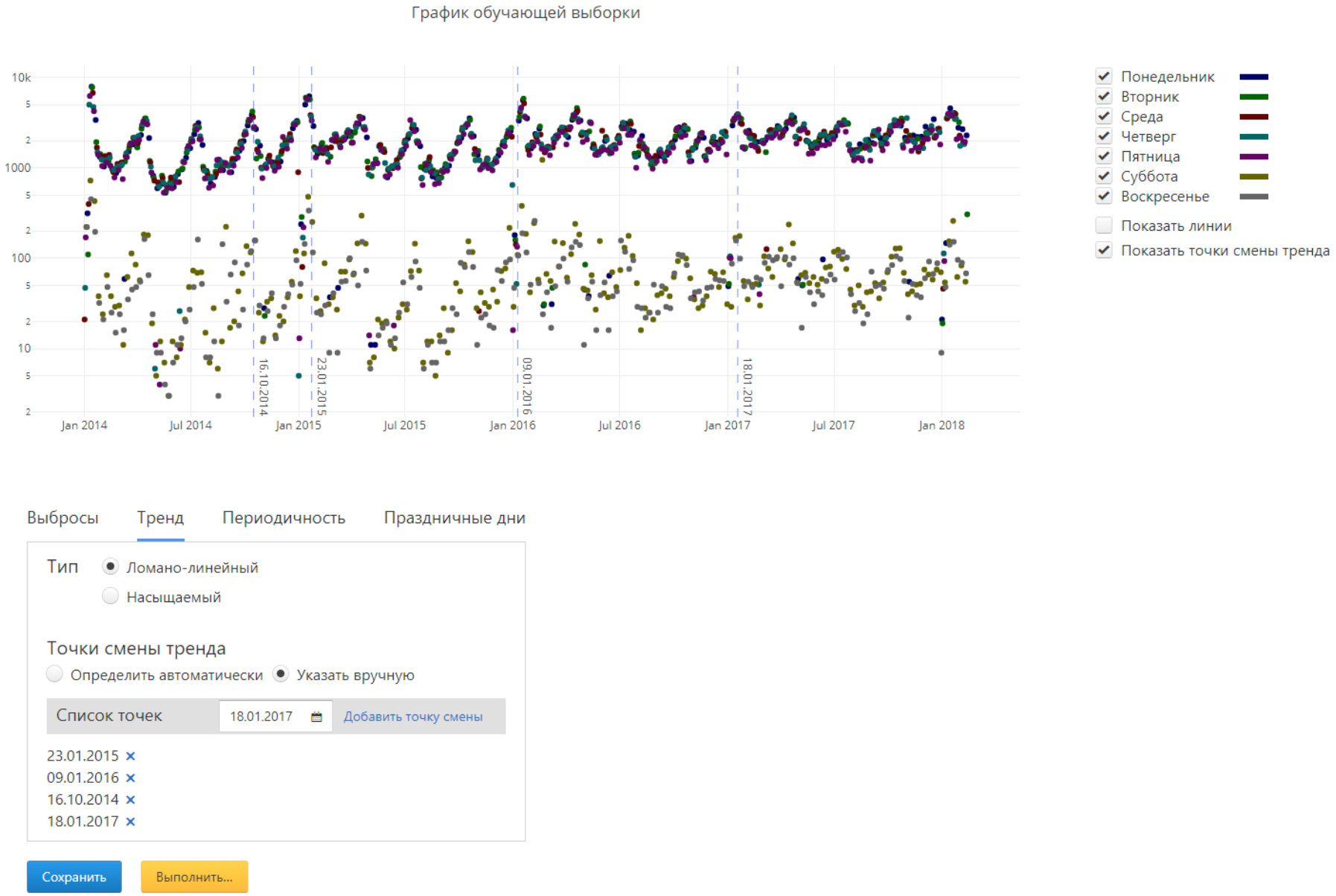
Приложение 1. Интерфейс сервиса настройки модели



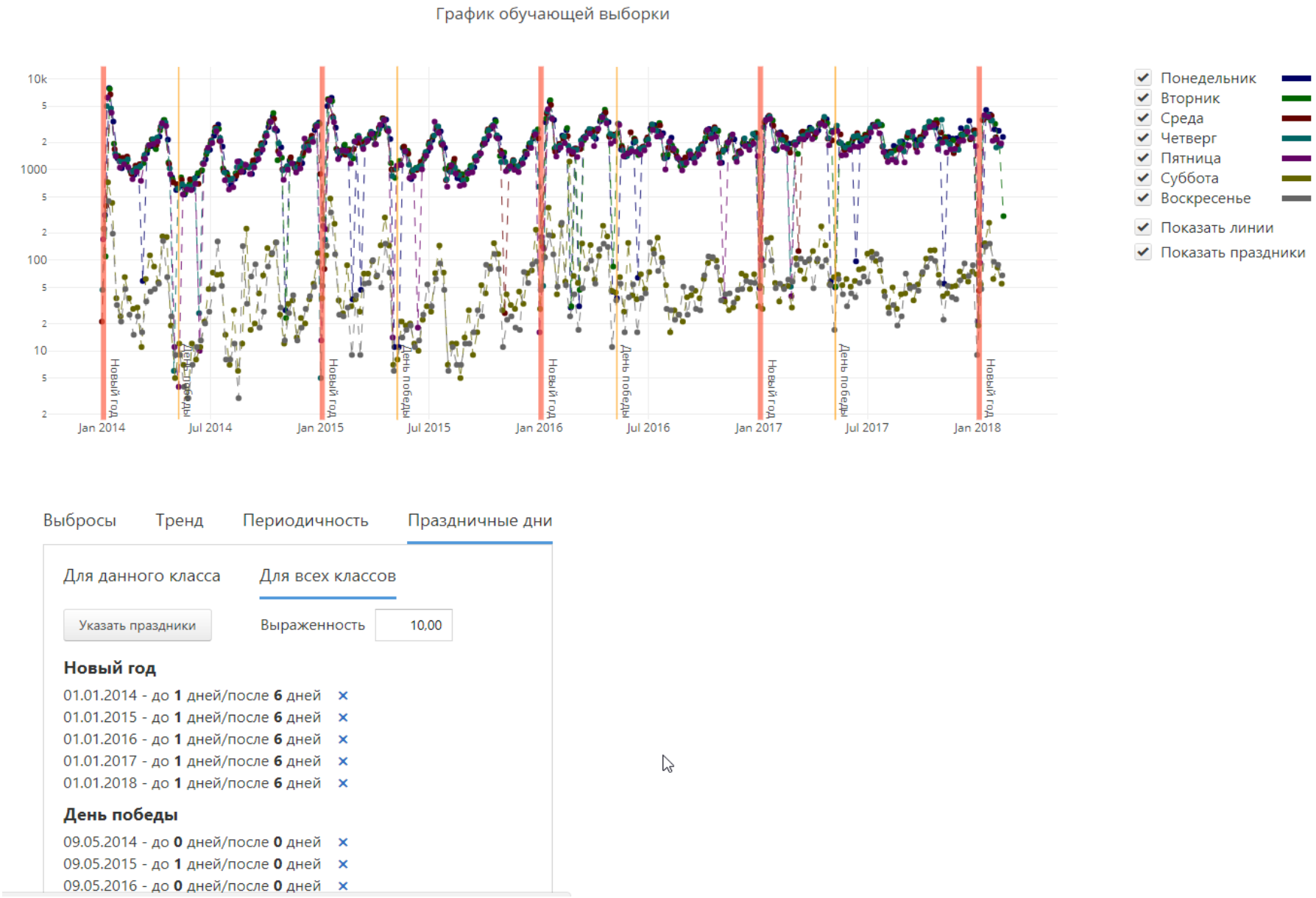
1. Домашняя страница сервиса



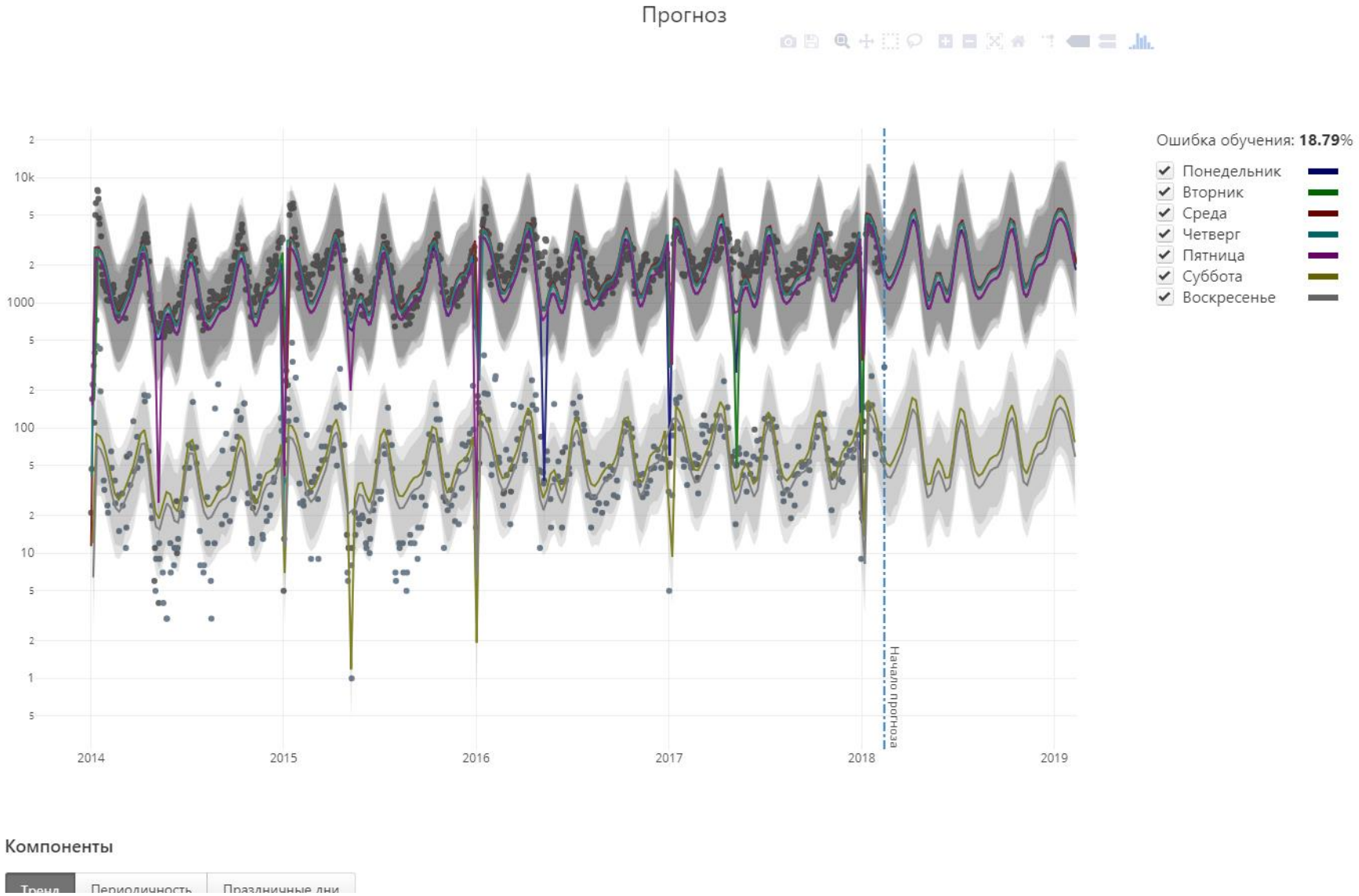
2. Интерфейс настройки выбросов



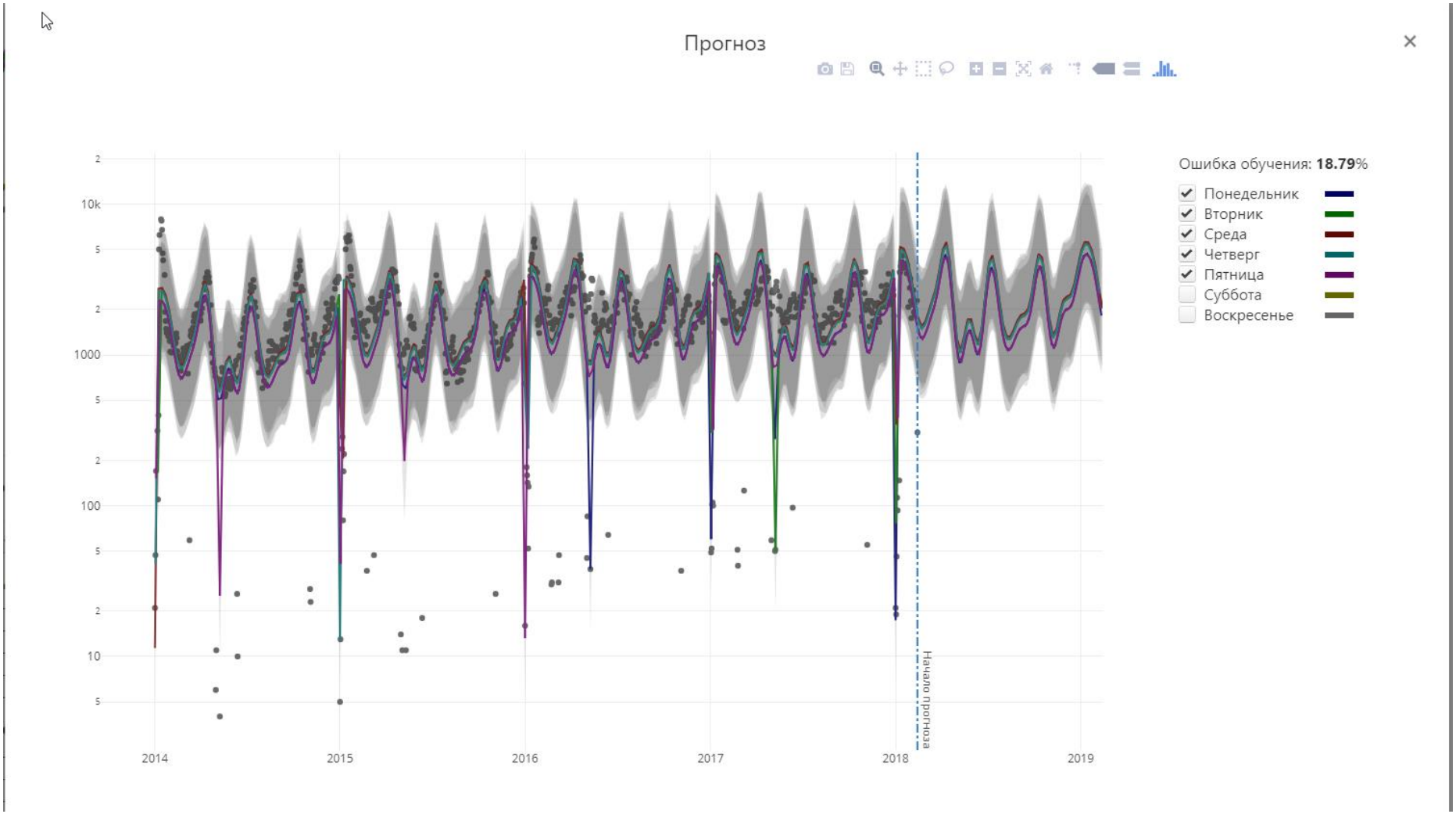
3. Настройка тренда



4. Настройка праздничных дней



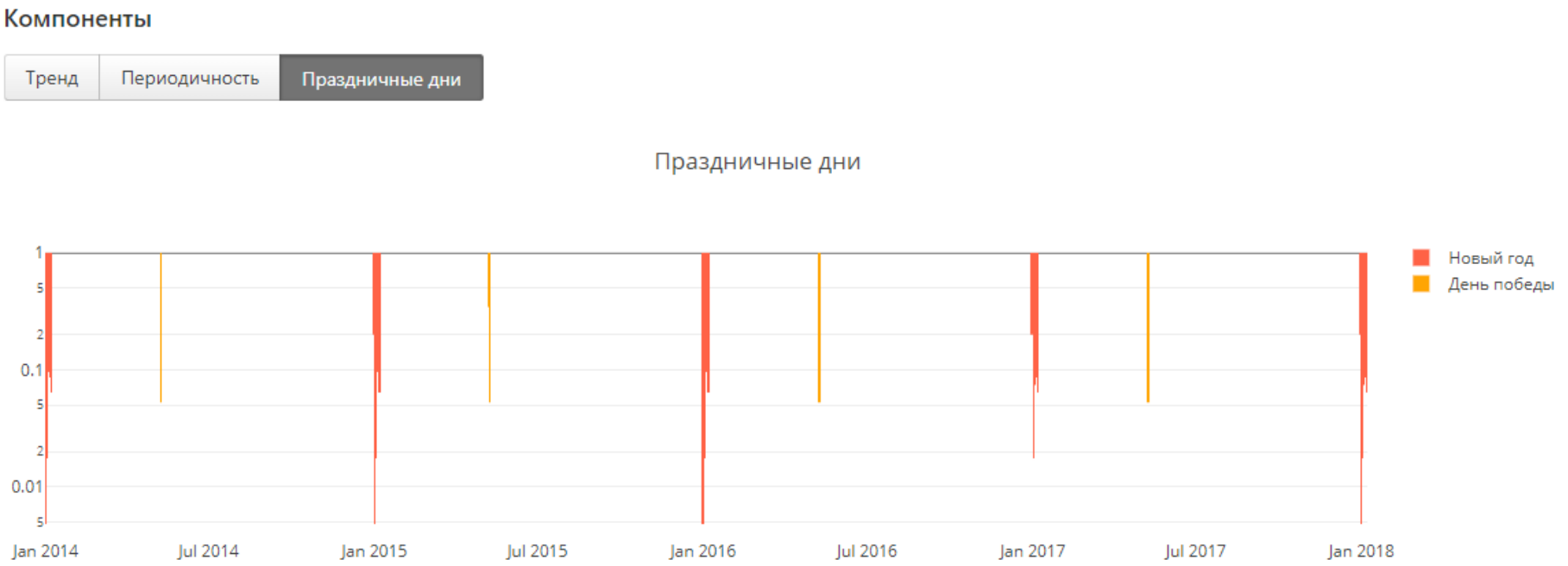
5. Просмотр прогноза



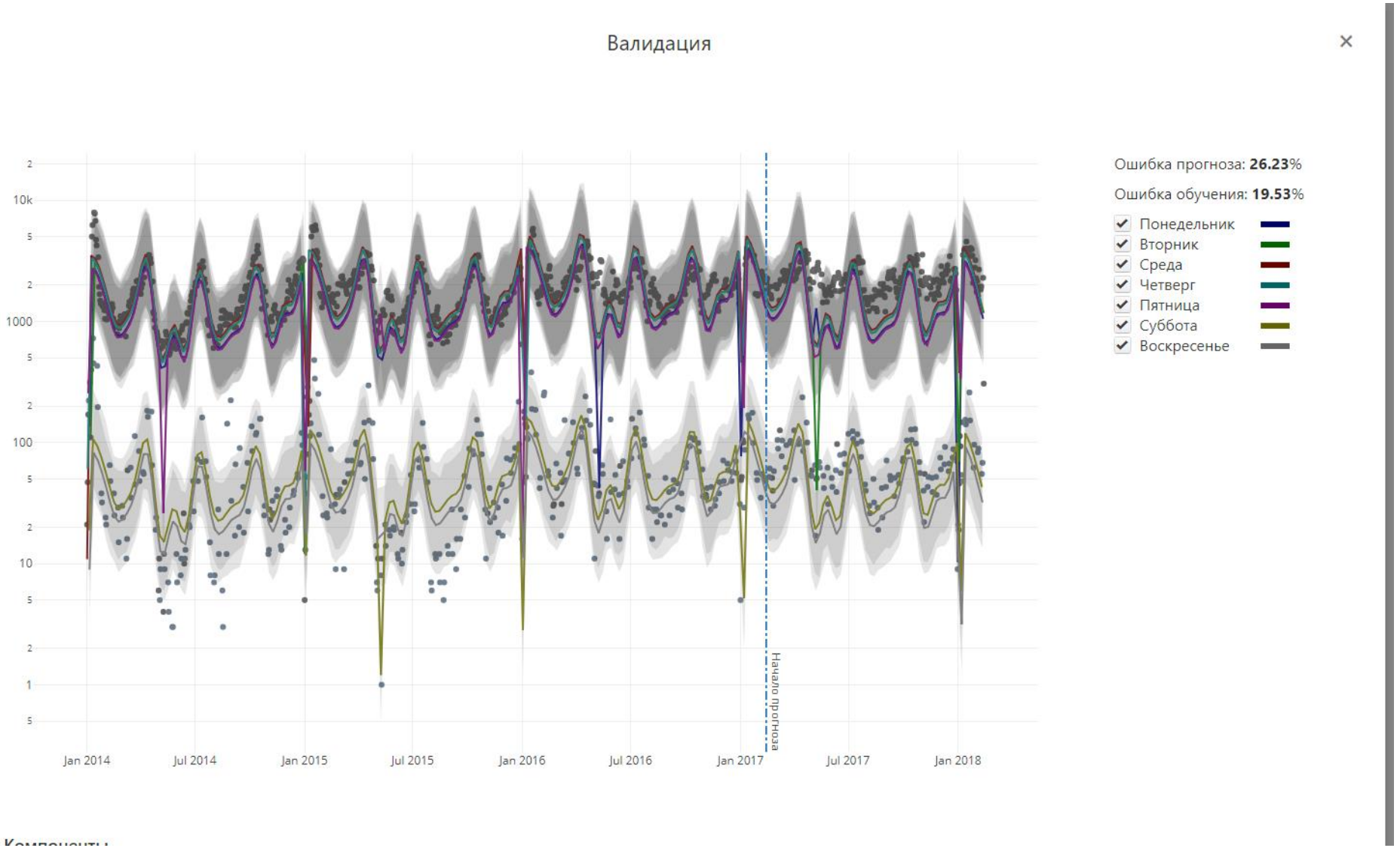
6. Прогноз без выходных

7. Просмотр компоненты тренда

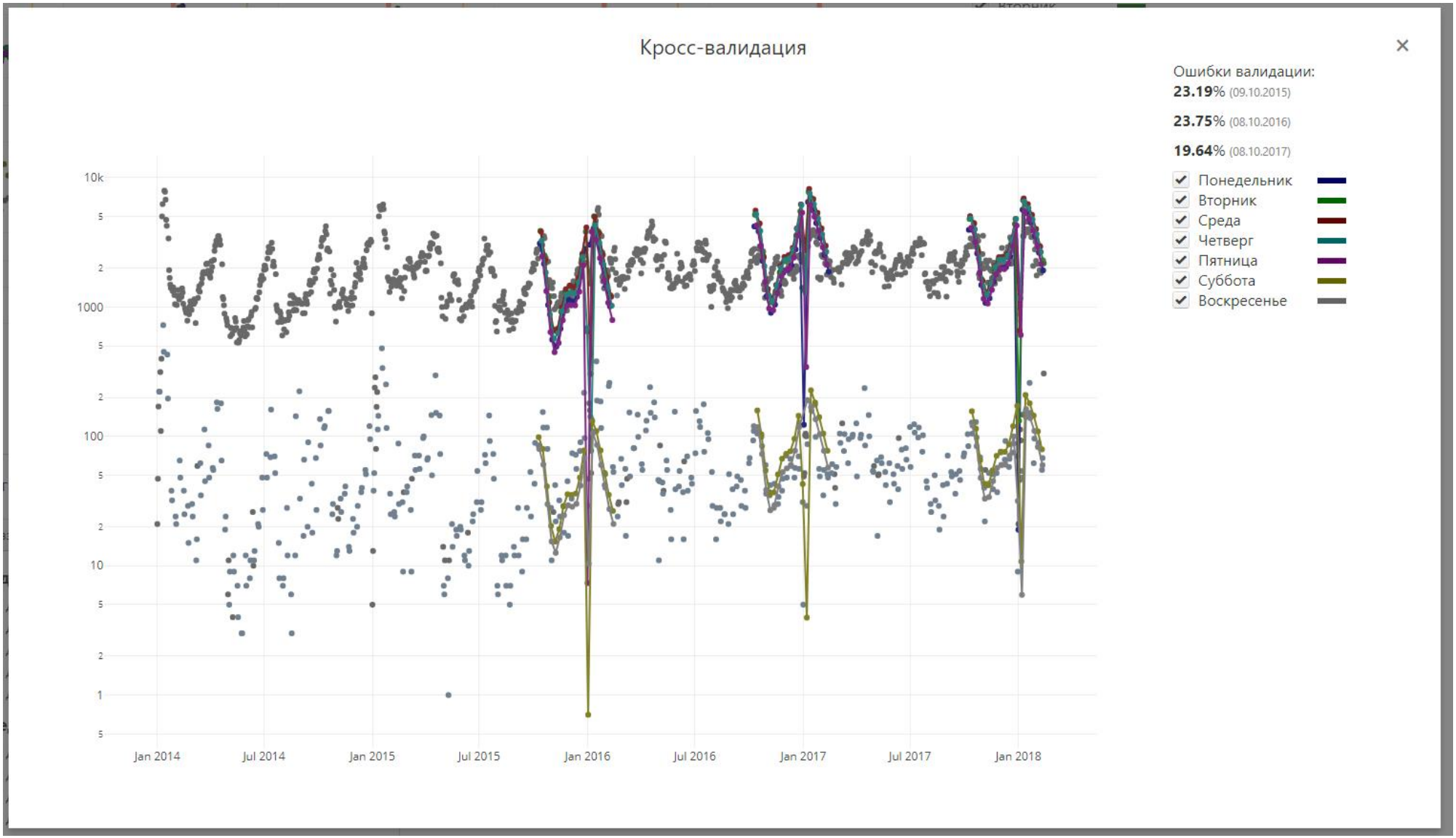
8. Просмотр сезонных компонентов



9. Просмотр компоненты праздничных дней



10. График валидации



11. График кросс-валидации

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Чучуева И.А. МОДЕЛЬ ПРОГНОЗИРОВАНИЯ ВРЕМЕННЫХ РЯДОВ ПО ВЫБОРКЕ МАКСИМАЛЬНОГО ПОДОБИЯ, диссертация… канд. тех. наук / Московский государственный технический университет им. Н.Э. Баумана. Москва, 2012.

[2] De Gooijer, J. G. & Hyndman, R. J. (2006), '25 years of time series forecasting', International Journal of Forecasting 22(3), 443–473.

[3] Vladik Kreinovich, Hung T. Nguyen и Rujira Ouncharoen. How to Estimate Forecasting Quality: A System-Motivated Derivation of Symmetric Mean Absolute Percentage Error (SMAPE) and Other Similar Characteristics / Vladik Kreinovich; University of Texas at El Paso El Paso, TX 79968, USA, 2014.

[4] J. Scott Armstrong PRINCIPLES OF FORECASTING: A Handbook for Researchers and Practitioners / J. Scott Armstron; University of Pennsylvania, The Wharton School Philadelphia, Pennsylvania USA, 2001

[5] Harvey, A. & Peters, S. (1990), 'Estimation procedures for structural time series models', Journal of Forecasting 9, 89–108