Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

РАБОТА ПРОВЕРЕНА
РЕЦЕНЗЕНТ,
_____
_____
«____»_____2017 Г.

ДОПУСТИТЬ К ЗАЩИТЕ
ЗАВЕДУЮЩИЙ КАФЕДРОЙ,
Д.Ф.-М.Н., ДОЦЕНТ
_____С.А.ЗАГРЕБИНА
«____»_____2017 Г.

Прогнозирование объемов продаж для интернет-магазина с помощью методов машинного обучения

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
_____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»
Направление «Прикладная математика и информатика»

ДОПУСТИТЬ К ЗАЩИТЕ
ЗАВЕДУЮЩИЙ КАФЕДРОЙ,
Д.Ф.-М.Н., ДОЦЕНТ
_____С.А.ЗАГРЕБИНА
«____»_____2017 Г.

ЗАДАНИЕ
на выпускную квалификационную работу студента
Степанченко Екатерины Вячеславовны
Группа ЕТ-485

1. Тема работы
   Прогнозирование объемов продаж для интернет-магазина с помощью методов машинного обучения утверждена приказом по университету от _____ 20_ _г. №_____
2. Срок сдачи студентом законченной работы (проекта) _____
3. Исходные данные к работе (проекту)
   - Данные о продажах интернет-магазина.
4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов)

4.1 Предобработка исходных данных
4.2 Описание принципов прогнозирования объемов продаж
4.3 Описание алгоритмов и методов машинного обучения, применяемых для прогнозирования объемов продаж
4.4 Программная реализация заданных алгоритмов при помощи методов машинного обучения
4.5 Тестирование алгоритмов на функциональность и эффективность
4.6 Анализ полученных результатов

5. Перечень графического материала (с точным указанием обязательных чертежей, плакатов в листах формата А1)

5.1 Презентация

5.2 Алгоритм случайного леса
5.3 Алгоритм метода опорных векторов
5.4 Работа с программой
5.5 Предобработка и анализ исходных данных
5.6 Прогнозирование объемов продаж
5.7 Выводы

6 Дата выдачи задания_____

Руководитель_____ _____ _____А.К.Богушов
(подпись)
Задание принял к исполнению_____Е.В.Степанченко
(подпись студента)

КАЛЕНДАРНЫЙ ПЛАН

Наименование этапов выпускной квалификационной работы (проекта)	Срок выполнения этапов работы (проекта)	Отметка о выполнении руководителя
1. Сбор материала и литературы по теме дипломной работы	30.01.2017 – 25.02.2017	
2. Постановка задач	25.02.2017 – 04.03.2017	
3. Предобработка исходных данных	04.03.2017 – 01.04.2017 –	
4. Программная реализация разработанных алгоритмов	01.04.2017 – 01.05.2017	
5. Подготовка пояснительной записки дипломной работы	01.05.2017 – 14.05.2017	
6. Оформление пояснительной записки	14.05.2017 – 25.05.2017	
7. Получение отзыва руководителя	30.05.2017	
8. Проверка работы руководителем, исправление замечаний	25.05.2017 – 31.05.2017	
9. Подготовка графического материала и доклада	31.05.2017 – 10.06.2017	
10. Нормоконтроль	10.06.2017	
11. Рецензирование, представление зав. кафедрой	10.06.2017	

Заведующий кафедрой _____/С.А.Загребина/

Руководитель работы _____/А.К.Богушов/

Студент _____/Е.В.Степанченко /

.

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(Национальный исследовательский университет)»
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

АННОТАЦИЯ

Степанченко Е.В. Прогнозирование объемов продаж для интернет-магазина с помощью методов машинного обучения/ Е.В. Степанченко. –Челябинск: ЮУрГУ, Институт естественных и точных наук, 2017. – 72 с., 17 ил., 0 табл., 6 прил., библиогр. список – 13 названий

В дипломной работе была разработана система прогнозирования объемов продаж для интернет-магазина с помощью методов машинного обучения. Были рассмотрены теоретические основы построения прогнозирования объемов продаж. Рассмотрены методы прогнозирования на основе алгоритмов случайного леса, градиентного бустинга и метода опорных векторов. Были написаны программы на языке программирования Python 3.5. Произведено тестирование и оценка эффективности разработанной программы.

Оглавление

ВВЕДЕНИЕ

На сегодняшний день одной из самых востребованных областей знаний является машинное обучение. Оно применяется в большом спектре решения прикладных задач, решение которых ранее считалось прерогативой человека. Цель решения таких задач состоит в прогнозировании и автоматизации принятия того или иного решения, в качестве анализируемых данных выступают большие объемы накопленной ранее информации.

Важным составляющим организации успешной деятельности коммерческой фирмы является составление качественного прогноза продаж. Одна из основных проблем в работе предприятия – это определение количества товара, которое необходимо иметь на складе в текущий и последующие моменты времени. Именно поэтому одной из приоритетных задач для коммерческих фирм является оптимизация загрузки склада [1].

Правильно составленный прогноз увеличивает эффективность ведения бизнеса путем контроля и оптимизации расходов, что, в свою очередь, помогает сформировать оптимальные (а не завышенные или заниженные) запасы продукции на складе. Таким образом в условиях рыночной экономики становится **актуальным** вопрос организации оперативного контроля и управления запасами материальных ресурсов на предприятии. Также на предприятии существует необходимость обработки большого количества номенклатуры. Решением данных проблем в определенной степени способствует внедрение автоматизированных систем, например таких как 1С–предприятие, которые позволяют наладить учет движения материальных ресурсов, то есть спрогнозировать какое количество товара из конкретной категории предприятие может реализовать потребителям в различные временные промежутки при условии сохранения интереса потребителей.

Задачи снижения товарного запаса и повышения оборачиваемости напрямую связаны с точностью прогнозирования продаж. При расчете страхового запаса среднее отклонение продаж от прогнозов является одной из основных состав-

ляющих. Поэтому повышение точности прогноза на 30–40% может дать повышение оборачиваемости на 15–20%, а также позволит увеличить продажи из-за снижения количества отсутствующего запаса или распроданного товара [2].

Обычно компания торгует большим количеством номенклатуры, но ходовыми являются примерно 20–25% от общего числа. В ряде направлений торговли можно выделить товары с сезонной составляющей или ввести праздничные товары, которые хорошо раскупаются перед праздниками. При составлении прогноза данные факторы нужно учитывать, так как они непосредственно будут влиять на качество прогноза.

Цель данной работы заключается в выявлении наиболее эффективного алгоритма прогнозирования объемов продаж на примере одного интернет-магазина с использованием методов машинного обучения.

Для достижения поставленной цели были решены следующие **задачи**:

1) изучить экономические показатели предприятия как факторы оценки объёмов продаж;

2) обосновать выбор метода прогнозирования продаж предприятия;

3) проанализировать финансовую деятельность предприятия с применением алгоритмов прогнозирования на основе методов машинного обучения;

4) рассчитать прогноз объёмов продаж продукции предприятия в краткосрочном периоде.

Предметом исследования является объем продаж одного интернет-магазина.

Объект исследования - интернет-магазин специализирующийся на продаже косметической продукции.

В качестве **методов** исследования выступают следующие:

- анализ экономических показателей деятельности компании;
- методы прогнозирования;
- методы машинного обучения;
- методы алгоритмизации и программирования.

В качестве **информационной базы** используются данные по объемам продаж интернет-магазина, работы Жак Вайнера, Гаруана Рич, Эндрю ЫНа, К.В. Воронцова, Д.П. Ветрова.

Работа состоит из введения, трех разделов, заключения, библиографического списка и приложения. Объем работы составляет 73 страницы, объем библиографии – 13 источников.

Первая глава работы описывает теоретический аспект методов прогнозирования применяемых на сегодняшний день, их взаимосвязь с методами машинного обучения, рассматриваются классы и задачи машинного обучения и аргументируется выбор алгоритмов для прогнозирования.

Во второй главе описана программная реализация системы прогнозирования и обработки исходных данных, описаны основные методы применяемые при обучении прогнозирования объемов продаж и продемонстрирована работа с программными модулями.

В третьей главе проводится вычислительный эксперимент, тестирующий реализованные модули на корректность, функциональность и точность выполнения поставленной задачи. Строится прогноз с применением трех алгоритмов, вычисляется ряд ошибок и проведя анализ выявляется самый эффективный алгоритм методов машинного обучения.

В заключении формулируются выводы по дипломной работе, оценивается степень выполнения поставленных задач.

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Понятие прогнозирования объемов продаж

Залог успеха компании обеспечивается лояльностью ее клиентов. Сокращая количество случаев дефицита товаров, торговая компания гарантирует себе эту лояльность. Эффективное прогнозирование позволяет подготовить исходные данные для отдела закупок, которые, в свою очередь, планируют своевременные поставки и обеспечивают постоянное наличие необходимого количества товара на складах компании. При прогнозировании используются как накопленный опыт прошлого, так и текущие допущения относительно развития событий в будущем.

Прогнозы разделяют на долгосрочные (сроком от одного года и более) и краткосрочные (неделя, квартал, полугодие и т.д.). Чем дольше период прогнозирования, тем вероятнее, что точность прогноза будет ниже, так как большое количество факторов могут влиять на ожидаемый результат в ту или иную сторону. Долгосрочный прогноз менее детализирован и служит основой для выбора долгосрочной стратегии развития предприятия.

Существует много подходов к классификации методов прогнозирования. Соответствующие методы прогнозирования во многом зависят от имеющихся данных. В частности, эти методы разделяют на количественные (формализованные) и качественные (интуитивные или экспертные).

Качественные методы основываются на опыте, интуиции, экспертных оценках специалистов в области принятия решений.

Качественные методы прогнозирования:

- Мнение жюри - коллективные экспертные оценки руководителей подразделений предприятия.
- Совокупное мнение сбытовиков работающих с потребителями и знающих их реакцию и поведение на рынке. Они знакомы с потребителями и могут принять в расчет их недавние действия быстрее, чем удается построить количественную модель.
- Модель ожидания потребителя основана на результатах опроса клиентов

компании в отношении их потребностей в будущем. Собрав все полученные таким путем данные и сделав поправки на пере- или недооценку исходя из собственного опыта, руководитель может предсказать совокупный спрос.

- Метод экспертных оценок - представляет анализ мнений специалистов из различных, но связанных областей деятельности. После заполнения анкет и ознакомления с мнением других экспертов специалисты делают новые оценки. Процедура может повторяться несколько раз для получения единого мнения по рассматриваемому вопросу.

Количественные методы прогнозирования используются тогда, когда есть основания считать, что деятельность предприятия в прошлом имела определенную тенденцию, которую можно продолжить в будущем, и когда имеющейся информации достаточно для выявления статистически достоверных тенденций или зависимостей, характеризующих производственную деятельность объекта управления. Исходной информацией служат как временные ряды прогнозируемого показателя, так и уровни факторных признаков.

Методы количественного прогнозирования:

- Метод экстраполяции (анализ временных рядов, трендов), при котором тенденции прошлого продлеваются в будущее развитие ситуации. Такой метод используется для оценки спроса на товары, объема сбыта, сезонности и др. Применение этого метода возможно лишь в ситуации, когда рыночная ситуация не изменяется слишком быстрыми темпами. Чем более достоверно предположение о подобии будущего прошлому, тем вероятнее точность прогноза. Для анализа временных рядов используют методы: 1) взвешенной средней; 2) экспоненциального сглаживания; 3) экстраполяции на основе аналитических показателей; 4) экстраполяции тренда.

- Анализ корреляций, рассматривающий зависимость между различными рассматриваемыми факторами и другими переменными. Метод используется для рассмотрения влияния нескольких переменных на прогнозируемый параметр. Применение такого метода является достаточно сложным и дорогостоящим, однако в упрощенном виде его можно использовать и для практического бизнеса.

- Имитационное компьютерное моделирование состоит в прогнозировании или реконструировании изменений некоторых сущностей путем прогона их компьютерных моделей.

- Каузальные методы прогнозирования используются тогда, когда прогнозируемая величина зависит от большого количества сложных факторов, которые можно использовать только при наличии вычислительной техники и соответствующего программного обеспечения. К ним относятся методы машинного обучения и нейронные сети. Методы машинного обучения представляют собой алгоритмы, которые сами настраиваются на данных. Используется машинное обучение в основном в задачах прогнозирования, когда нужно по входной информации (признакам) предсказать некоторую выходную величину.

Существует широкий диапазон количественных методов прогнозирования, часто разрабатываемых в конкретных дисциплинах для конкретных целей. Каждый метод имеет свои свойства, точность и затраты, которые необходимо учитывать при выборе конкретного метода. В большинстве задач количественного прогнозирования используются либо данные временных рядов (собираемые через регулярные интервалы времени), либо данные поперечного сечения (собранные в один момент времени). Данные временных рядов полезны, когда прогнозируются изменения, которые со временем меняются (например, цены акций, показатели продаж, прибыль и т.д.).

Выбор метода прогнозирования зависит от многих факторов - контекста прогноза, релевантности и доступности исторических данных, желаемой степени точности, прогнозируемого периода времени, стоимости/пользы прогноза для предприятия и доступного времени для проведения анализа.

Методы машинного обучения автоматизируют методы количественного прогнозирования объемов проданной продукции, ведь задача прогнозирования - кропотливая и рутинная работа. Она требует к себе внимания большого количества экспертов товарных групп, которые в совершенстве знают спрос на товар, особенности его ввода и вывода из ассортимента. Содержание большого штата экспертов для компании нерентабельно, а имеющиеся эксперты не всегда справ-

ляются с объемом задач, что, естественно, приводит к ошибкам прогнозирования и, как следствие, возможным значительным убыткам компании. Прогнозы строят и люди, но их точность существенно ниже. Большинство задач прогнозирования можно решить методом регрессии. В работе с данными равных машине быть не может [2].

1.2 Понятие машинного обучения

Потребность в изучении и анализе больших объемов данных возрастает с каждым годом. Этот вопрос актуален как для крупных компаний, так и для решения повседневных задач. В настоящее время машинное обучение активно используется в различных сферах деятельности человека. Обученные системы варьируются по сложности, но объединяет их то, что все они, основываясь на предыдущих данных для обучения, могут принимать решения. К таким системам относятся:

- системы распознавания лиц;
- рекомендательные системы;
- спам фильтры;
- системы прогнозирования;
- системы поисковых запросов и многие другие.

Системы машинного обучения применяются и в мобильных устройствах. Данные системы подстраиваются под владельца обучаясь на его действиях и выборе. Например, это такие мобильные системы, как Siri.

Современные компании вне зависимости от своего размера постоянно сталкиваются с необходимостью обработки больших объемов информации, ручная обработка которых зачастую уже невозможна. И чтобы компания продолжала продуктивно работать в условиях конкуренции необходимо пользоваться системами анализа данных на основе машинного обучения.

Благодаря получаемой информации появляется возможность строить прогнозы из разных сфер человеческой деятельности. Например, можно спрогнозировать риски возникновения каких-либо заболеваний благодаря системам, собирающим и анализирующим данные о пациентах. Или проведение анализа финан-

совых данных может предсказать колебания стоимостей акций, активов, курса валют. Машинное обучение используется при больших объемах данных, так как нельзя применить какой-либо простой алгоритм для обработки таких данных.

1.2.1 Классы задач машинного обучения

Машинное обучение – метод обработки и анализа данных, основанный на создании и использовании компьютерных систем, которые обучаются не только на основе уже имеющихся данных, но и во время эксплуатации. Качество работы системы машинного обучения тем выше, чем дольше используется система и чем больше данных в ней содержится. Цель работы таких систем заключается в прогнозировании будущего состояния или поведения исследуемого объекта, результатов действия и оценка будущих тенденций.

Большую часть задач машинного обучения можно разделить на:

1) обучение без учителя;
2) обучение с учителем.

Под обучением с учителем понимают множество ситуаций и множеств возможных ответов, между которыми существует некоторая зависимость, но какая именно неизвестно. Известна обучающая выборка, которая так же называется конечной совокупностью и представляет собой взаимосвязанные пары «объект-ответ». Где множество X — объекты, примеры, образцы. А множество Y — ответы, отклики, «метки», классы.

В таком случае имеется некоторая зависимость $g : X \to Y$, позволяющая по $x \in X$ предсказать (или оценить вероятность появления) $y \in Y$. Зависимость известна только на объектах из обучающей выборки: $T = (x1, y1),(x2, y2), ...,(xn, yn)$.

Пара $(xi , yi) \in X \times Y$ - прецедент. Задача обучения по прецедентам: научиться по новым объектам $x \in X$ предсказывать ответы $y \in Y$.

На основе этих данных строится алгоритм, который должен быть способен для любого объекта выдать ответ максимально приближенный к точному. Точность ответов оценивается функционалом качества, который характеризует среднюю ошибку алгоритма на произвольной выборке. Учителем является либо обучающая

выборка, либо задающий правильные ответы. Данную задачу можно разделить на следующие задачи:

- задача классификации сводится к определению класса объекта по его характеристикам. В классической задаче классификации обучающая выборка представляет собой набор отдельных объектов $X = \{x_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $x_i = (x_i, 1, \dots x_i, d)$. В качестве исхода объекта X фигурирует переменная t, принимающая конечное число значений, обычно из множества $T = \{1, \dots, l\}$.

- задача регрессии позволяет определить по известным характеристикам объекта значение некоторого его параметра. В отличие от задачи классификации значением параметра является не конечное множество классов, а множество действительных чисел. В классической задаче регрессии обучающая выборка представляет собой набор отдельных объектов $X = \{x_i\}_{i=1}^{n}$, характеризующихся вектором вещественнозначных признаков $x_i = (x_i, 1, \dots x_i, d)$. В качестве исхода объекта X фигурирует непрерывная вещественнозначная переменная t.

- задача прогнозирования разбивается на два этапа. На первом этапе на основании набора данных с известными результатами строится модель. На втором этапе она используется для предсказания результатов на основании новых наборов данных. При этом требуется, чтобы построенные модели работали максимально точно. В классической задаче прогнозирования обучающая выборка представляет собой набор измерений $X = \{x[i]\}_{i=1}^{n}$, представляющих собой вектор вещественнозначных величин $x[i] = (x_1[i], \dots x_d[i])$, сделанных в определенные моменты времени. Требуется построить алгоритм (предиктор), который вернул бы точечную оценку $\{\hat{x}[i]\}_{i=n+1}^{n+q}$ или доверительный интервал $\{(x - [i],\ x + [i])\}_{i=n+1}^{n+q}$ прогноза на заданную глубину q. В отличие от задачи восстановления регрессии, здесь осуществляется прогноз в первую очередь по времени, а не по признакам.

Суть обучения без учителя состоит в том, что система получает набор начальных критериев на основе которых требуется обнаружить закономерности

между объектами обучающей выборки. В этом случае нет учителя и обучающая выборка состоит только из объектов, т.е. Y отсутствует. К данной задачи относятся следующие задачи:

- задачи кластеризации заключается в поиске независимых групп (кластеров) и их характеристик во всем множестве анализируемых данных. В классической задаче кластеризации обучающая выборка представляет собой набор отдельных объектов $X = \{x_i\}_{i=1}^{n}$, характеризующихся вектором веществennознachных признаков $x_i = (x_i, 1, \dots x_i, d)$. Требуется построить алгоритм (кластеризатор), который разбил бы выборку на непересекающиеся группы (кластеры) $X = U_{j=1}^{k} C_k$,

$$C_j \subset \{x_1, \dots, x_m\}, C_i \cap C_j = \emptyset.$$

- задача понижения размерности состоит в получении представления этой выборки в пространстве меньшей размерности $T = \{t_n\}_{n=1}^{N}, t_n \in R^d$.

Если выходная величина вещественная, то говорят о задаче регрессии, а если принимает конечное число дискретных значений — то о задаче классификации. Для решения задачи прогнозирования объемов продаж воспользуемся методом обучения с учителем задачи прогнозирования.

1.3 Используемые алгоритмы методов машинного обучения

В работе используются такие алгоритмы методов машинного обучения, как:

- алгоритм случайного леса;
- метод опорных векторов;
- градиентный бустинг.

Использование данных алгоритмов обусловлено материалами исследования Рич Гаруана «Эмпирическое сравнение контролируемых алгоритмов обучения» и исследование Жак Вайнера «Сравнение 14 различных семейств классификации алгоритмов» из которых следует, что данные алгоритмы являются наиболее эффективными, поэтому эти алгоритмы будут рассмотрены в рамках данной работы[3,4]. Стоит отметить, что приведены статьи как 2006 года, так и 2016 года, что говорит о том, что на протяжении 10 лет эти алгоритмы являются одними из самых эффективных алгоритмов методов машинного обучения.

1.3.1 Алгоритм случайного леса

Случайный лес – это универсальный алгоритм, предложенный Лео Брейманом и Адель Катлер и используемый в настоящее время в своем изначальном виде, так как никакие изменения его существенно не улучшили. Алгоритм заключается в использовании групп решающих деревьев и основан на двух основных методах: метод бэггинга Бреймана и метод случайных подпространств [5].

Деревом называется конечный связный граф с множеством вершин V, не содержащий циклов и имеющий выделенную вершину $v_0 \in V$, в которую не входит ни одно ребро. Данная вершина является корнем дерева. Вершина не имеющая выходящих ребер, называется терминальной или листом, соответствуют классам. Остальные вершины называются внутренними, они соответствуют признакам. Выходящие ребра связывают каждую внутреннюю вершину v с левой дочерней вершиной L_v и с правой дочерней вершиной R_v.

Дерево решений является направленным деревом и представляет собой процесс принятия решений с учетом возможных альтернатив и последствий на основе характеристик исследуемого объекта. Дерево решений является основным элементом леса, от того, каким образом построено дерево зависит качество работы и устойчивость всей финальной композиции.

Алгоритм построения дерева принятия решений:

Шаг 1. Выбрать очередной признак P и поместить его в текущую вершину.

Шаг 2. Для каждого значения выбранного признака v:

- из тестовых данных оставить только те, у которых P = v;
- рекурсивно построить дерево на выбранных данных, рассматривая оставшиеся признаки.

Шаг 3. Остановиться, если все тестовые объекты принадлежат одному классу, если закончились признаки или по другим критериям. Конец алгоритма.

Пример построенного дерева принятия решений представлен на рисунке 1.

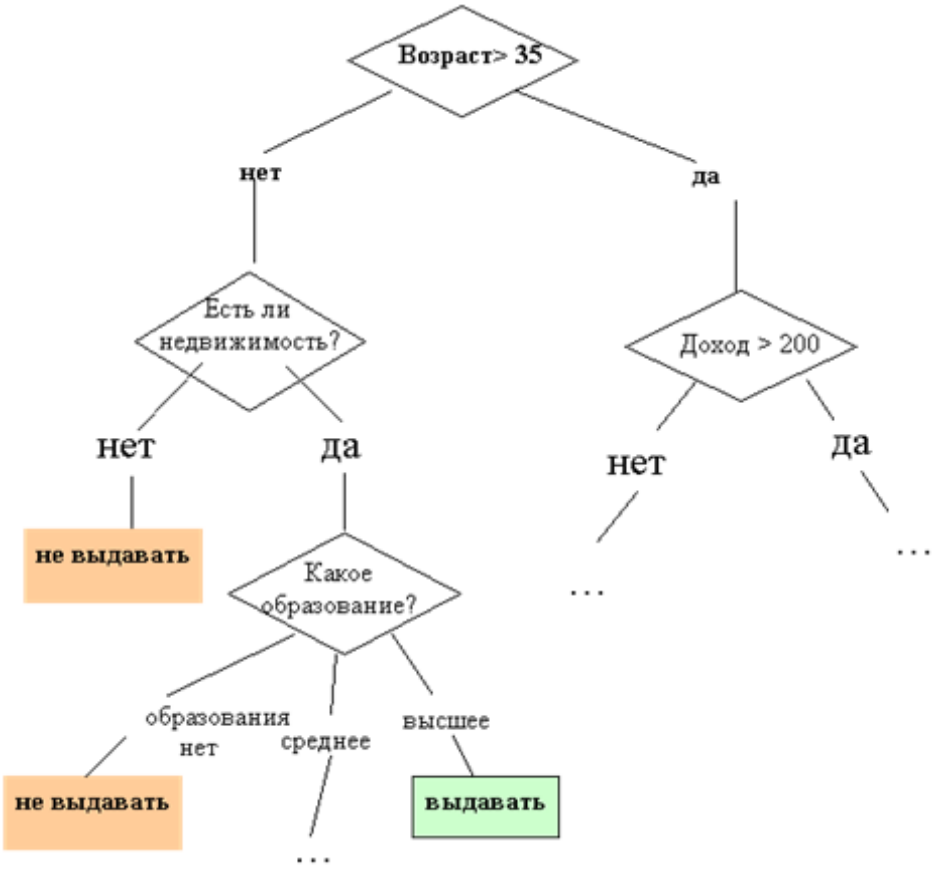

Рисунок 1 – Пример дерева принятия решений

Цель построения дерева решений состоит либо в классификации векторов x из $P(x)$, либо в оценке условного математического ожидания отклика при данном значении x. Процесс принятия решений начинается с корневой вершины и состоит в последовательном применении правил, связанных с вершинами дерева. Результатом этого процесса является определение терминальной вершины t такой, что $x \in X_t$. В случае классификации вектор x относится к наиболее часто встречающемуся классу в подвыборке D_t, соответствующей данной терминальной вершине, а в случае регрессии оценка условного математического ожидания отклика представляет собой среднее значение отклика в этой подвыборке.

Бэггинг – это способ построения композиций классификатора, в котором классификаторы обучаются независимо друг от друга. Благодаря такому подходу метод можно распараллелить. Бэггинг позволяет вычислять оценки обобщающей способности, оптимизировать число алгоритмов, оценить степень важности признака, подобрать параметры модели.

Универсальность алгоритма случайного леса состоит в том, что он показывает эффективную работу на большом спектре задач и применяется для решения задач классификации, селекции признаков, регрессии и т.д.

Алгоритм индукции случайного леса [6]:

Шаг 1. Для $i = 1, 2 \dots B$ (здесь B – количество деревьев в ансамбле) выполнить:

- Сформировать бутстреп выборку S размера i по исходной обучающей выборке $D = \{x_i, y_i\}_{i=1}^{l}$;
- По бутстреп выборке S индуцировать неусеченное дерево решений T_i с минимальным количеством наблюдений в терминальных вершинах равным $n_{\min}$, рекурсивно следуя следующему подалгоритму:

(а) из исходного набора n признаков случайно выбрать p признаков;

(б) из p признаков выбрать признак, который обеспечивает наилучшее расщепление;

(в) расщепить выборку, соответствующую обрабатываемой вершине на две подвыборки;

Шаг 2. В результате выполнения шага 1 получить ансамбль деревьев решений $\{T_i\}_{i=1}^{B}$;

Шаг 3. Предсказать новые наблюдения следующим образом:

(а) для регрессии: $\hat{f}_{rf}^{B}(x) = \frac{1}{B}\sum_{i=1}^{B} T_i(x)$;

(б) для классификации: пусть $\widehat{\omega}_i(x) \in \{\omega_1, \omega_2 \dots \omega_c\}$ – класс, предсказанный деревом решений T_i, т.е. $T_i(x) = \widehat{\omega}_i(x)$; тогда $\widehat{\omega}_{rf}^{B}(x)$ класс, наиболее часто встречающийся в множестве $\{\widehat{\omega}_b(x)\}_{i=1}^{B}$. Конец алгоритма.

Основным недостатком алгоритма случайного леса является большой размер структуры данных. Временная сложность алгоритма – $O(n * K)$, где K – количество деревьев.

Достоинства алгоритма случайного леса:

- высокая параллелизуемость и масштабируемость;
- эффективная обработка данных с большим числом признаков и классов;

- высокая точность классификации новых объектов;
- обучающая выборка может содержать признаки, измеряемые в разных шкалах: числовой, порядковый и номинальный;
- обработка как непрерывных, так и дискретных признаков.

Недостатки алгоритма случайного леса:

- качество полученной композиции сильно ухудшается в случае, когда на вход алгоритму подается мало размеченных данных, поскольку деревья не могут качественно выявить скрытые в данных закономерности;
- низкий уровень обработки категориальных признаков;
- медленная скорость обработки пропущенных значений;
- при увеличении числа деревьев повышается дисперсия;
- необходимость большого объема памяти для хранения модели.

Таким образом, можно достаточно качественно классифицировать рассматриваемую выборку объектов при помощи всего одного дерева решений, если в качестве ответа для тестового объекта, попавшего в ячейку A_i, выдавать номер наиболее часто встречающегося в этой ячейке класса. Однако в реальных задачах часто встречаются погрешности в измерениях и объекты-выбросы, которые серьезно портят качество классификации одним конкретным деревом решений. Поэтому перед построением каждого нового дерева происходит сэмплирование с повторениями новой выборки $\{(x_i^k, y_i^k)\}_{i=1}^N$ из $\{(x_i, y_i)\}_{i=1}^N$, на которой происходит обучение дерева с номером k. После построения всех деревьев каждый тестовый объект z_i получает в качестве промежуточного ответа вектор меток, присвоенных ему каждым деревом, который преобразуется в финальную метку по методу простого голосования.

1.3.2 Алгоритм градиентного бустинга

Бустинг - итерационный алгоритм, реализующий “сильный” классификатор, который позволяет добиться произвольно малой ошибки обучения (на обучающей выборке) на основе композиции “слабых” классификаторов, каждый из которых лучше, чем просто угадывание, т.е. вероятность правильной классификации

больше 0.5. Ключевая идея: использование весовой версии одних и тех же обучающих данных вместо случайного выбора их подмножества. Основное отличие бустинга от бэггинга состоит в том, что обучающая выборка на каждой итерации определяется, исходя из ошибок классификации на предыдущих итерациях.

В общем же виде бустинг-алгоритм может быть представлен как жадный процесс минимизации любой штрафной функции. Однако проблема заключается в том, что для некоторой базовой модели может не существовать эффективного алгоритма обучения, работающего с учетом выбранной функции потерь. Данную проблему отчасти решает подход, носящий название градиентного бустинга, который позволяет без серьезных модификаций алгоритма обучения использовать любой дифференцируемый штраф $L(y, y')$.

Алгоритм градиентного бустинга [7]:

Входные данные: набор данных $\{(x_i, y_i)\}_{i=1,\dots,n}$; число итераций M, выбор функции потерь $L(y, f)$ с выписанным градиентом; выбор семейства функций базовых алгоритмов $h(x, \theta)$, с процедурой их обучения; дополнительные гиперпараметры $h(x, \theta)$ (например глубина дерева у деревьев решений); начальное приближение $f_0(x)$ заменено константой γ.

Выходные данные: итоговая модель градиентного бустинга $\hat{f}(x) = \sum_{i=0}^{M} \hat{f}_i(x)$.

Шаг 1. Инициализировать метод градиентного бустинга константным значением $\hat{f}(x) = \hat{f}_0,\ \hat{f}_0 = \gamma, \gamma \epsilon R,\ \hat{f}_0 = arg\ \min \sum_{i=1}^{n} L(y_i, \gamma)$.

Шаг 2. Для каждой итерации $t = 1, \dots, M$ повторять

1. Посчитать псевдо-остатки r_t: $r_{it} = -\left[\frac{dL(y_i, f(x_i))}{df(x_i)}\right]_{f(x)=\hat{f}(x)}, for\ i = 1, \dots, n.$

2. Построить новый базовый алгоритм $h_t(x)$ как регрессию на псевдо-остатках $\{(x_i, r_{it})\}_{i=1\dots n}$.

3. Найти оптимальный коэффициент p_t при $h_t(x)$ относительно исходной функции потерь $p_t = arg\ \min \sum_{i=1}^{n} L(y_i, f(x_i) + p * h(x_i, \theta))$.

4. Сохранить $\hat{f}_t(x) = p_t * h_t(x)$.

5. Обновить текущее приближение $\hat{f}(x)$: $\hat{f}(x) \leftarrow \hat{f}(x) + \hat{f}_t(x) = \sum_{i=0}^{t} \hat{f}_i(x)$.

Шаг 3. Скомпоновать итоговую модель градиентного бустинга $\hat{f}(x)$:

$\hat{f}(x) = \sum_{i=0}^{M} \hat{f}_i(x)$. Конец алгоритма.

Сложность алгоритма градиентного бустинга можно определить как $O(N^2)$, но при этом стоит учитывать, что чем больше итераций, тем больше больше базовых алгоритмов для голосования и соответственно тем больше сложность алгоритма.

Достоинства алгоритма случайного леса:

- подходит для задач регрессии, классификации и ранжирования;
- может использоваться произвольная функция потерь;
- возможно рассмотрение любого семейства базовых алгоритмов;

Недостатки алгоритма случайного леса:

- большое количество итераций и базовых алгоритмов увеличивают сложность алгоритма;
- алгоритм является плохо распараллеливаемым;
- результаты работы бустинга могут быть сложно интерпретируемы.

Таким образом, алгоритм градиентного бустинга выполянет M итераций, на каждой из которых происходит обучение базовой модели $L(y_i, \gamma)$, путем подбора оптимальных параметров γ. Следует отметить, что для настройки каждой базовой модели используется функция потерь $\Psi(y, y')$, вообще говоря, отличная от $L(y, y')$. Именно благодаря этому факту алгоритм является универсальным по отношению к применяемому штрафу.

1.3.3 Алгоритм метода опорных векторов

Метод опорных векторов относится к группе граничных методов. Он определяет классы при помощи границ областей и основан на концепции гиперплоскостей, которые определяют границы гиперповерхностей. Каждый объект данных представлен, как вектор (точка в p-мерном пространстве (последовательность p чисел)). Основная идея метода опорных векторов в поиск гиперплоскости с максимальным зазором, разделяющей между собой вектора разных классов. Задача поиска параметров для такой гиперплоскости $w * x = b$, где x, w −скалярное

произведение векторов x и w; b – дискриминантная функция, сводится к задаче квадратичного программирования

Для построения модели метода опорных векторов нужно взять обучающие входные данные, отобразить их в многомерное пространство, а затем использовать регрессию, чтобы найти гиперплоскость (гиперплоскость - это поверхность в n-мерном пространстве, которая разделяет его на подпространства), которая лучше всего разделяла бы классы входных данных. После обучения модели она способна классифицировать новые входные данные в один из классов при помощи разделяющей гиперплоскости.

Разделяющей гиперплоскостью будем называть гиперплоскость, которая отделяет группу объектов, имеющих различную классовую принадлежность. Тогда, оптимальная разделяющая гиперплоскость — это гиперплоскость, максимизирующая ширину разделяющей полосы и лежащая в середине этой полосы. Иными словами, оптимальная разделяющая гиперплоскость максимизирует зазор (margin) между плоскостью и данными из обучающей выборки. Если классы линейно разделимы и каждый содержит не менее одного элемента, то оптимальная разделяющая гиперплоскость единственна. Пример границы полосы представлен на рисунке 2.

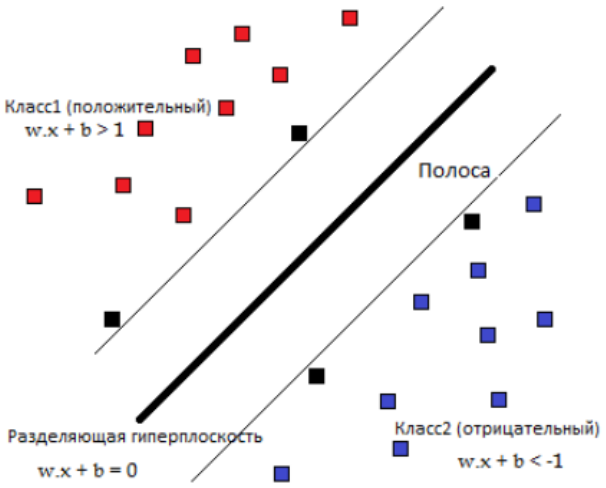

Рисунок 2 – Пример разделяющей полосы

Определение ширины разделяющей полосы между данными является задачей оптимизации, где $M = \frac{2}{||w||} = \frac{2}{w_1^2+w_2^2+\cdots+w_m^2}$.

Формулировка задачи оптимизации: $\frac{2}{||w||} \to \max_w$ при условии

$$\begin{cases} y_i = +1: w^{\mathrm{T}}x + b \geq 1 \\ y_i = -1: w^{\mathrm{T}}x + b \leq -1 \end{cases}$$

Где y_i – метки классов, элементы множества $Y = -1, +1$;

x, w −скалярное произведение векторов x и w;

$b -$ дискриминантная функция.

Получим задачу квадратической оптимизации с линейными ограничениями:

$$\begin{cases} \frac{||w||}{2} = \frac{1}{2}(w_1^2 + w_2^2 + \cdots + w_m^2) \to \min_w \\ y_i(w^{\mathrm{T}}x + b) \geq 1 \end{cases}$$

В методе опорных векторов рассматривается задача обучения по прецедентам $\{X, Y, y^*, X^l\}$ где X — пространство объектов, Y — множество ответов, $y^*: X \to Y$ — целевая зависимость, значения которой известны только на объектах обучающей выборки $X^l = (x_i, y_i)_{i=1}^l, y_i = y^*(x_i)$. Требуется построить алгоритм $a: X \to Y$, аппроксимирующий целевую зависимость на всём пространстве X.

Алгоритм метода опорных векторов [8]:

Шаг 1. Выбрать ядро $K(x_i, x_j)$:

- линейное: $K(x_i, x_j) = x_i^t x_j$;
- полиномиалное: $K(x_i, x_j) = (1 + x_i^t x_j)^p$.

Шаг 2. Выбрать значение штрафа C.

Шаг 3. Выбрать значение параметра ядра (σ, p, B_0, B_1).

Шаг 4. Решить задачу квадратического программирования и получить опорные векторы a_i:

$$\begin{cases} -L(\lambda) = -\sum_{i=1}^n \lambda_i + \frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j c_i c_j (x_i * x_j) \to \min_\lambda \\ \lambda_i \geq 0,\ 1 \leq i \leq n \\ \sum_{i=1}^n \lambda_i c_i = 0 \end{cases}$$

Шаг 5. Построить разделяющую функцию g используя опорные векторы:

$g(x) = \sum_{i=1}^n a_i K(x_i, x_j) + b.$

Шаг 6. Проверить качество классификации, используя скользящий контроль.

Шаг 7. Если качество не удовлетворяет, то переход к шагу 2, иначе завершение алгоритма.

Стоит отметить, что суммирование идёт не по всей выборке, а только по опорным векторам. Именно это свойство разреженности отличает метод опорных векторов от других линейных разделителей.

Метод опорных векторов характеризуется высокой вычислительной сложностью, алгоритм в общем виде имеет алгоритмическую сложность $O(N^3)$.

Достоинства метода опорных векторов:

- наиболее быстрый метод нахождения решающих функций;
- метод сводится к решению задачи квадратичного программирования в выпуклой области, которая всегда имеет одно решение;
- метод находит разделяющую полосу максимальной ширины, что позволяет осуществлять более точную классификацию;
- позволяет рассматривать различные виды нелинейности, изменяя ядра или их параметры.

Недостатки метода опорных векторов:

- неустойчивость к различным шумам во входных данных;
- необходимость подборки параметров C, Σ;
- при линейной неразделимости классов не существует общего подхода к выбору ядра и построению спрямляющего пространства;
- отсутствует отбор признаков.

Выводы по главе один

Качественно составленный прогноз ожидаемого объема продаж помогает решить проблему оперативного контроля и управления запасами материальных ресурсов на предприятии. Существует большое количество методов прогнозирования, каждый из которых имеет свои достоинства и недостатки. Внедрение автоматизированных систем, а в частности применение методов машинного обучения

для составления прогноза, автоматизируют процесс прогнозирования, обработки и анализа большого количества номенклатуры на предприятии.

Большую часть задач машинного обучения можно разделить на обучение без учителя и обучение с учителем. Суть обучения без учителя состоит в том, что система получает набор начальных критериев на основе которых требуется обнаружить закономерности между объектами обучающей выборки. А под обучением с учителем понимают множество ситуаций и множество возможных ответов, между которыми существует некоторая зависимость. Известна обучающая выборка, которая представляет собой взаимосвязанные пары «объект-ответ». Для задач прогнозирования используется обучение учителем.

Машинное обучение применяется для решения ряда задач: классификации, регрессии и прогнозирования. Для составления прогноза наиболее эффективными алгоритмами будут такие алгоритмы, как случайный лес, градиентный бустинг и метод опорных векторов.

Задача прогнозирования разбивается на два этапа. На первом этапе на основании набора данных с известными результатами строится модель. На втором этапе она используется для предсказания результатов на основании новых наборов данных. При этом требуется, чтобы построенные модели работали максимально точно. Степень достоверности прогнозов можно сравнить с реальными показателями и, сделав выводы, приступить к следующему прогнозу уже с существующими данными.

2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРОГНОЗИРОВАНИЯ ОБЪЕМОВ ПРОДАЖ И АНАЛИЗА ИСХОДНЫХ ДАННЫХ

Система прогнозирования объемов продаж была реализована в виде библиотеки с возможностью вызова из консоли на языке программирования Python версии 3.5. Техническое задание на разработку и руководство пользователя приведены в приложениях Б и В соответственно.

Средства, использованные во время разработки:

- операционная система MS Windows 7;
- среда разработки Pycharm Community Edition 2016;
- набор библиотек Anaconda 4.3.1;
- система контроля версий Git 2.13.0.

2.1 Описание программных модулей

Проект состоит из следующих частей:

- модуль analysis;
- модуль converter;
- модуль predictionSales.

Модуль analysis, представленный в приложении В, позволяет оценить корректность заполнения данных, проверить их тип, удалить ненужные данные, очистить или заполнить пустые значения, провести анализ и отследить тенденции развития товара или товарной группы, выделить наиболее часто продаваемые товары, выделить режим работы предприятия, построить графики для визуализации тенденций продаж.

Модуль converter, представленный в приложении Г, позволяет обработать исходные данные и привести их к соответствующему виду использования при дальнейшем обучении. Данные проверяются на корректность и обрабатываются возможные ошибки заполнения данных, такие как: незаполненные части, нулевые значения, не подходящий для обучения тип данных, замена строковых обозначений численными, разбиение даты на отдельные столбцы (год, месяц, день). Полученный файл сохраняется под названием out формата csv в рабочей директории.

Для осуществления работы программы необходимо было импортировать следующие внешние модули и методы:

- модуль argparse назначение которого в обработке опций и аргументов командной строки, с которой вызывается скрипт. Основные действия будут выполнятся классом ArgumentParser модуля argparse. Из данного класса воспользуемся методом: add_argument.

Синтаксис add_argument:

```
add_argument(name or flags...[, action][, nargs][, default] [, choices] [, help])
```

Параметры:

name или flags – имя опции.

action – тип действия, которое будет выполнятся при вызове опции.

nargs – количество аргументов к опции, которые могут быть использованы.

default – действие по-умолчанию, если опция пропущена (не вызвана), применяется ко всем опциям, которым не назначен отдельный ключ default.

choices – контейнер допустимых значений для аргумента.

help – краткое описание опции, используется в –help.

Пример использования данного метода представлен в листинге 1.

```
from argparse import ArgumentParser
parser = ArgumentParser(description='Converts client dataset')

parser.add_argument('data', default='in.csv', help='Input file')
parser.add_argument('--out', '-o', default='out.csv', help='Output file')
parser.add_argument('--company', default='converter', help='Company name')
args = parser.parse_args()
```

Листинг 1 – Метод add_argument модуля argparse

- модуль pandas - библиотека для анализа и обработки данных.

Модуль predictionSales, представленный в приложении Д, позволяет подготовить данные (разбить на тестовую и тренировочную выборку) и обучить данные на ряде моделей (случайный лес, градиентный бустинг, метод опорных векторов) с возможностью изменения параметров (интервал прогнозирования, разбиение выборки, объект обучения), вывести ошибку отклонения прогнозных значе-

ний от реальных. Для осуществления работы программы необходимо было импортировать следующие внешние модули и методы:

- модуль argparse назначение которого в обработке опций и аргументов командной строки, с которой вызывается скрипт.
- модуль enum необходим для ограничения множества допустимых значений для некоторого типа данных. Содержит несколько классов Enum, IntEnum (константы могут иметь только тип int) и декоратор unique, который проверяет набор констант на дубликаты.
- модуль collections предоставляет специализированные типы данных, на основе словарей, кортежей, множеств, списков. Класс collections.namedtuple позволяет создать тип данных, ведущий себя как кортеж, с тем дополнением, что каждому элементу присваивается имя, по которому можно в дальнейшем получать доступ.
- модуль RandomForestRegressor представляет собой множество решающих деревьев. В задаче регрессии их ответы усредняются для улучшения точности прогнозирования и контроля над процессом. Все значения используются по умолчанию.

Синтаксис RandomForestRegressor:

```
class sklearn class sklearn.ensemble.RandomForestRegressor(n_estimators=10, criterion='mse', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, bootstrap=True, n_jobs=1, random_state=None, verbose=0, warm_start=False)
```

Параметры:

n_estimators – количество деревьев в лесу (по умолчанию 10);

criterion – функция для измерения критерия расщепления. Для решения задачи регрессии реализованы два критерия: “mse” (средняя квадратичная ошибка) и “mae” (средняя абсолютная ошибка). По умолчанию “mse”.

max_features – число признаков для выбора расщепления. При увеличении max_features увеличивается время построения леса, а деревья становятся «более однообразными». По умолчанию он равен $\sqrt{n}$, где n – количество деревьев.

max_depth – максимальная глубина деревьев. При увеличении глубины возрастает качество на обучении, но увеличивается время выполнения построения.

min_samples_split – минимальное число объектов, при котором выполняется расщепление. При увеличении параметра качество на обучении падает, но время построения случайного леса сокращается. Значение по умолчанию равно двум.

min_samples_leaf – ограничение на число объектов в листьях. Значение по умолчанию равно единице, но для задач регрессии рекомендуется использовать значение равное пяти.

min_weight_fraction_leaf – минимальная взвешенная доля от общей суммы весов (всех входных объектов) должна быть в листе. По умолчанию имеет значение равное нулю, а это значит, что все листья имеют одинаковый вес.

max_leaf_nodes – максимальное количество листьев. По умолчанию ограничения отсутствуют.

bootstrap – применять ли бустрэп для построения дерева. По умолчанию бустрэп применяется.

n_jobs – количество ядер для построения модели и предсказаний. По умолчанию значение равно единице.

random_state – начальное значение для генерации случайных чисел. По умолчанию значение отсутствует.

verbose – вывод подробного отчета по построению деревьев. По умолчанию равен нулю.

warm_start – использует уже натренированную модель и добавляет деревьев в ансамбль. По умолчание значение отсутствует.

- модуль GradientBoostingRegressor строит аддитивную модель, позволяет оптимизировать произвольные дифференцируемые функции потерь.

Синтаксис GradientBoostingRegressor:

```
class sklearn.ensemble.GradientBoostingRegressor(loss='ls', n_estimators=100, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3, init=None, random_state=None, verbose=0, max_leaf_nodes=None, warm_start=False)
```

Параметры:

loss – функция потерь, подлежащая оптимизации. По умолчанию используется “ls” – метод наименьших квадратов.

n_estimators – количество ступеней для ускорения. При увеличение параметра увеличивается производительность. Значение по умолчанию равняется ста.

max_depth – максимальная глубина индивидуальных оценок регрессии. При увеличении глубины возрастает качество на обучении, но увеличивается время выполнения построения. По умолчанию равняется трем.

criterion – функция для измерения критерия расщепления. Для решения задачи регрессии реализованы два критерия: “freedman_mse” (средняя квадратичная ошибка с оценкой улучшения Фридмена) и “mae” (средняя абсолютная ошибка). По умолчанию “ freedman_mse”.

min_samples_split – минимальное число объектов, при котором выполняется расщепление. При увеличении параметра качество на обучении падает, но время построения градиентного бустинга сокращается. Значение по умолчанию равно двум.

min_samples_leaf – ограничение на число объектов в листьях. Значение по умолчанию равно единице.

min_weight_fraction_leaf – минимальная взвешенная доля от общей суммы весов (всех входных объектов) должна быть в листе. По умолчанию имеет значение равное нулю, а это значит, что все листья имеют одинаковый вес.

subsample – доля образцов, которые будут использоваться для подгонки. Если значение меньше 1.0, это приводит к стохастическому усилению градиента. По умолчанию значение равно 1.0.

max_leaf_nodes – максимальное количество листьев. По умолчанию ограничения отсутствуют.

init – объект оценки, который используется для вычисления начальных прогнозов. По умолчанию отсутствует.

verbose – вывод подробного отчета по построению деревьев. По умолчанию равен нулю.

warm_start – использует уже натренированную модель и добавляет деревьев в ансамбль. По умолчание значение отсутствует.

- модуль SVR при построении разделяющей линии разделяет элементы выборки на отдельные классы. Для решения задач регрессии использует ядра при этом отсутствуют локальные минимумы.

Синтаксис SVR:

```
class sklearn.svm.SVR(kernel='rbf', degree=3, gamma='auto', tol=0.001, C=1.
0, shrinking=True, cache_size=200, verbose=False, max_iter=-1)
```

Параметры:

C – предельный параметр C ошибки. По умолчанию равен единице.

kernel – указывает тип ядра, который будет использоваться в алгоритме. По умолчанию задан “rbf” – радиальная базисная функция.

degree – степень многочлена полиномиальной функции ядра. По умолчанию значение равно трем.

gamma – коэффициент выбранного ядра. Для регрессии ядро “rbf”.

shrinking – использовать ли сокращенные эвристики. По умолчанию стоит их использование.

cache_size – размер кеша ядра, указывается в MB.

verbose – вывод подробного отчета. По умолчанию равен нулю.

max_iter – предел итераций при решении. По умолчанию равен минус единице, что означает, что ограничения на итерации отсутствуют.

2.2 Работа с программой

При запуске программы из консоли необходимо ввести параметры, передающие входные данные. Вывод справки по вводу параметров приведен на рисунке 3, а пример ввода параметров представлен на рисунке 4. Результатом работы программы является прогноз представленный в консоли или сохраненный в файл prediction.csv.

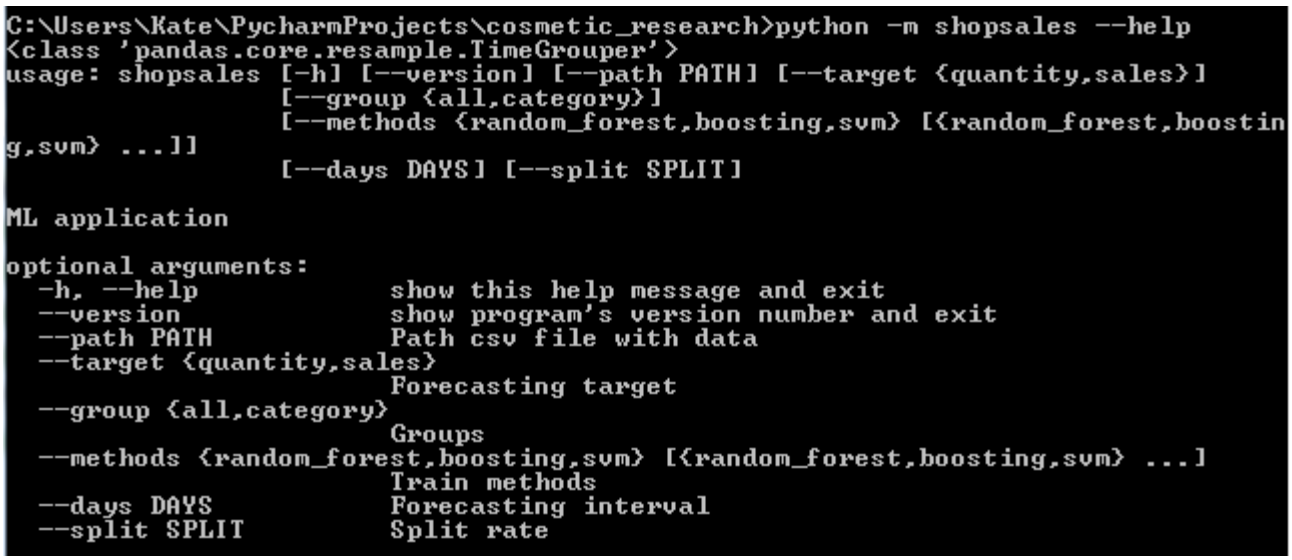

Рисунок 3 – Справка по вводу параметров

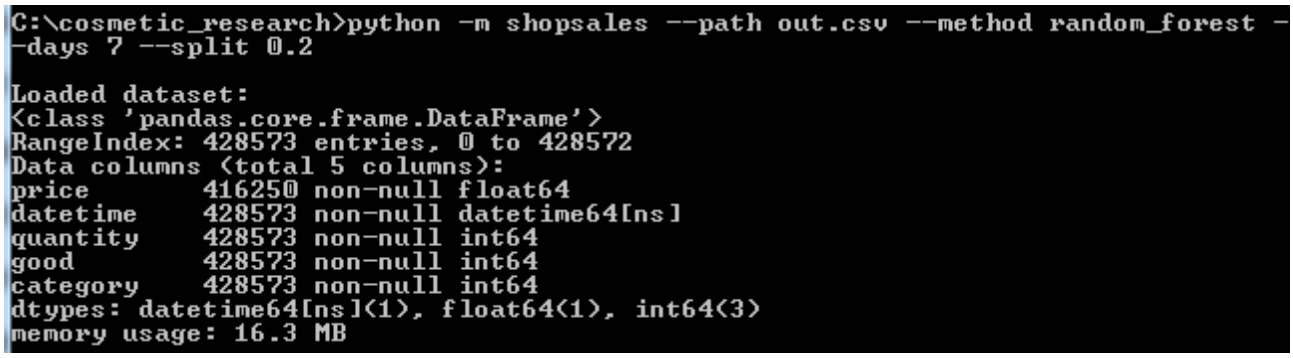

Рисунок 4 – Запуск программы с заданными параметрами

Где задаваемые параметры:

–m – выбор загружаемого модуля;

—path – путь (наименование) файла;

—method – выбор метода прогнозирования (random_forest, gradient_boosting, svr);

—days – по сколько дней берем выборку для обучения;

—split – разделение данных на тестовую и тренировочную выборку.

—target – объект прогноза.

Общий алгоритм работы программы predictionSales представлен на рисунке 4.

Рисунок 4 – Общий алгоритм работы программы predictionSales

Выводы по главе два

Система прогнозирования состоит состоит из следующих частей: модуль analysis; модуль converter; модуль predictionSales.

Модуль analysis позволяет провести предварительный анализ входных данных, оценить корректность их заполнения, проверить тип, удалить ненужные данные, очистить или заполнить пустые значения, провести анализ и отследить тенденции развития товара или товарной группы, выделить наиболее часто продаваемые товары, выделить режим работы предприятия, построить графики для визуализации тенденций продаж.

Модуль converter позволяет обработать исходные данные и привести их к соответствующему виду использования при дальнейшем обучении. Данные проверяются на корректность и обрабатываются возможные ошибки заполнения данных, такие как: незаполненные части, нулевые значения, не подходящий для обучения тип данных, замена строковых обозначений численными, разбиение даты на отдельные столбцы (год, месяц, день). Полученный файл сохраняется под названием out формата csv в рабочей директории.

Модуль predictionSales позволяет подготовить данные (разбить на тестовую и тренировочную выборку) и обучить данные на ряде моделей (случайный лес, градиентный бустинг, метод опорных векторов) с возможностью изменения параметров (интервал прогнозирования, разбиение выборки, объект обучения), вывести ошибку отклонения прогнозных значений от реальных.

Программа может работать автономно посредством вызова из консоли с передачей параметров и выбором метода обучения. Результаты работы программы (составление прогноза и вычисление ошибки) выводятся в файл формата csv в рабочую директорию.

3 ВЫЧИСЛИТЕЛЬНЫЙ ЭКСПЕРИМЕНТ

3.1 Предварительная обработка и анализ исходных данных

Загрузим исходные данные в Jupyter Notebook (интерактивная оболочка для языка программирования Python) и визуализируем данные в виде упорядоченной таблицы для начала работы и предварительного анализа. Загруженное количество данных равно 428573 строки, 8 колонок. Пример загруженных данных приведен на рисунке 1. Пример данных исходного файла представлен в приложении 1.

	ID_order	DateTime	ID_good	Quantity	Price	Hour	DayOfWeek	ID_category
0	0000-034809	01.09.2016 0:01	12810002	1	250	NaN	4	дизайн
1	0000-034809	01.09.2016 0:01	12810001	1	250	NaN	4	дизайн
2	0000-034809	01.09.2016 0:01	10730019	1	600	NaN	4	топ
3	0000-034809	01.09.2016 0:01	12570116	1	379,99	NaN	4	гель-лак
4	0000-034809	01.09.2016 0:01	12480003	1	350	NaN	4	топ
5	0000-034809	01.09.2016 0:01	12570024	1	380	NaN	4	гель-лак
6	0000-034809	01.09.2016 0:01	10730148	1	470,02	NaN	4	гель-лак
7	0000-034809	01.09.2016 0:01	10730223	1	600	NaN	4	база
8	0000-034809	01.09.2016 0:01	10750352	1	240	NaN	4	гель-лак
9	0000-034808	01.09.2016 0:02	10240006	1	60	NaN	4	пилки

Рисунок 1 – Таблица исходных данных

В представленной таблице используются следующие обозначения:

ID_order – идентификатор заказа показывающий, количество и стоимость отдельных товаров в конкретном заказе;

DateTime – время и дата покупки товара;

ID_good – идентификатор отдельно взятого товара;

Quantity – количество проданного конкретного товара;

Price – стоимость одной единицы конкретного товара;

Hour – время продажи товара (интернет- магазин работает круглосуточно);

DayOfWeek – день недели в который была совершена покупка;

ID_category – категория, к которой относится проданный товар.

Столбец DateTime имеет компиляцию из двух числовых значений – даты и времени. Для дальнейшей обработки и анализа данных нужно удалить время. Создадим

новый столбец GoodTime, куда будет занесена только дата. По завершению обработки даты удалим изначальный некорректный и вспомогательный столбцы. Для этого используются следующие команды, указанные в листинге 1.

```
# отрезаем от столбца Даты часы
data_new['Year'] = data_new.DateTime.apply(lambda x: x.split(' '))
# создаем столбец с исправленной датой
data_new['GoodTime'] = data_new.Year.apply(lambda x: (x[0]))
# удаление колонок
data_new=data_new.drop(["DateTime"], axis = 1)
data_new=data_new.drop(["Year"], axis = 1)
data_new[:5]
```

Листинг 1 – Обработка и создание столбца с корректной датой

Результат работы представлен на рисунке 2.

	ID_order	ID_good	Quantity	Price	Hour	DayOfWeek	ID_category	GoodTime
0	0000-034809	12810002	1	250	NaN	4	дизайн	01.09.2016
1	0000-034809	12810001	1	250	NaN	4	дизайн	01.09.2016
2	0000-034809	10730019	1	600	NaN	4	топ	01.09.2016
3	0000-034809	12570116	1	379,99	NaN	4	гель-лак	01.09.2016
4	0000-034809	12480003	1	350	NaN	4	топ	01.09.2016
5	0000-034809	12570024	1	380	NaN	4	гель-лак	01.09.2016

Рисунок 2 – Скорректированная дата

На основе скорректированной даты разобьем столбец GoodTime на отдельные значения (дни, месяцы, годы). Так же удалим столбец ID_order, так как при дальнейшем прогнозировании он не понадобится. В листинге 2 отображены используемые команды.

```
data_new['Year'] = data_new.GoodTime.apply(lambda x: x.split('.'))
data_new['Month'] = data_new.Year.apply(lambda x: int(x[1]))
data_new['Day'] = data_new.Year.apply(lambda x: int(x[0]))
data_new['Year'] = data_new.Year.apply(lambda x: int(x[2]))
data_new = data_new.drop(["ID_order"], axis = 1)
data_new[:5]
```

Листинг 2 – Разбиение даты на отдельные значения.

Результат работы представлен на рисунке 3.

	ID_good	Quantity	Price	Hour	DayOfWeek	ID_category	GoodTime	Month	Day	Year
0	12810002	1	250	NaN	4	дизайн	01.09.2016	9	1	2016
1	12810001	1	250	NaN	4	дизайн	01.09.2016	9	1	2016
2	10730019	1	600	NaN	4	топ	01.09.2016	9	1	2016
3	12570116	1	379,99	NaN	4	гель-лак	01.09.2016	9	1	2016
4	12480003	1	350	NaN	4	топ	01.09.2016	9	1	2016

Рисунок 3 – Разбиение даты на отдельные столбцы

Проверим данные в колонках на корректное заполнение. Для этого выведем уникальные значения и их количество по каждой колонке и проанализируем полученные результаты. Воспользуемся командами, отображенными в листинге 3.

```
print("Уникальные значения ID")
print(data_new.ID_good.unique())
print("Всего уникальных значений:"+ str(len(data.ID_good.unique())))
print("")
print("Уникальные значения количества")
print(data_new.Quantity.unique())
print("Всего уникальных значений:"+ str(len(data.Quantity.unique())))
print("")
print("Уникальные значения цены")
print(data_new.Price.unique())
print("Всего уникальных значений:"+ str(len(data.Price.unique())))
print("")
print("Уникальные значения времени")
print(data_new.Hour.unique())
print("Всего уникальных значений:"+ str(len(data.Hour.unique())))
print("")
print("Уникальные значения дня недели")
print(data_new.DayOfWeek.unique())
print("Всего уникальных значений:"+ str(len(data.DayOfWeek.unique())))
print("")
print("Уникальные значения категории товара")
print(data_new.ID_category.unique())
print("Всего уникальных значений:"+ str(len(data.ID_category.unique())))
print("")
```

Листинг 3 – Выделение уникальных значений и их количество по каждой колонке.

Результаты работы представлены на рисунке 4.

```
Уникальные значения ID
[12810002 12810001 10730019 ..., 18000438 14001949 10720079]
Всего уникальных значений:17197

Уникальные значения количества
[  1   2   3   5   4  13  12   8   6  10  20  50  15   7  18  37  30  42
  11   9  26  23  19  21  28  44  16  14  33  22  41  35  40  17  43  25
  45  65  53  29 100  62  51]
Всего уникальных значений:43

Уникальные значения цены
['250' '600' '379,99' ..., '455,9' '579,35' '254,29']
Всего уникальных значений:18531

Уникальные значения времени
[ nan   1.   2.   3.   4.   5.   6.   8.   9.  10.  11.  12.  13.  14.  15.
  16.  17.  18.  19.  20.  21.  22.  23.   7.]
Всего уникальных значений:24

Уникальные значения дня недели
[4 5 6 7 1 2 3]
Всего уникальных значений:7

Уникальные значения категории товара
['дизайн' 'топ' 'гель-лак' 'база' 'пилки' 'масло для кутикулы'
 'инструменты' 'слайдер' 'кисти' 'бонд' 'сопутствующие' 'гель-краска'
 'жидкость' 'для кутикулы' 'лампа LED' 'стразы' 'трафареты' 'стемпинг'
 'гель' 'для волос' 'аппаратный маникюр' 'термонаклейки' 'типсы и формы'
 'депиляция' 'аэрография' 'лечение ногтей' 'однофазный гель-лак' 'для ног'
 'акрил' 'для бровей и ресниц' 'оборудование' nan 'лампа УФ' 'спа' 'лак'
 'для рук и тела' 'декоративная косметика' 'набор для начинающих'
 'сертификаты' 'лампа гибрид']
Всего уникальных значений:40
```

Рисунок 4 – Уникальные значения по каждому столбцу

Проанализируем результаты по каждому столбцу в отдельности. Уникальные значения ID по столбцу исходных данных ID_good заполнены корректно, все данные представляют собой положительное восьмизначное значение.

Уникальное значение количества по столбцу исходных данных Quantity заполнены корректно, все данные положительны. Можно заметить, что самое большое количество проданных товаров за одну продажу составило 100 единиц конкретного товара.

Уникальное значение цены по столбцу исходных данных Price заполнены некорректно. Следует проверить тип данных и округлить значения в большую сторону.

Уникальное значение времени по столбцу исходных данных Hour заполнены некорректно. По данным можно сделать вывод, что магазин работает круглосуточно, но вместо значений в двенадцать часов ночи присутствуют пробелы, что необходимо исправить путем замены пробелов на корректное число.

Уникальное значение дня недели по столбцу исходных данных DayOfWeek заполнены корректно. Можно сделать вывод, что вся неделя является рабочей без фиксированных выходных дней.

Уникальное значение категории товара по столбцу исходных данных ID_category заполнены некорректно. При необходимости работы с этими данными может потребоваться преобразования значений из качественного признака в количественный.

По результатам вышеописанного анализа проверим тип данных по каждому столбцу. Для этого воспользуемся кодом представленном в листинге 4.

```
print ("Тип данных ID " + str(type(data_new.ID_good[0])))
print ("Тип данных Quantity " + str(type(data_new.Quantity[0])))
print ("Тип данных Price " + str(type(data_new.Price[0])))
print ("Тип данных Hour " + str(type(data_new.Hour[0])))
print ("Тип данных DayOfWeek " + str(type(data_new.DayOfWeek[0])))
print ("Тип данных ID_category " + str(type(data_new.ID_category[0])))
```

Листинг 4 – Проверка типа данных по каждому столбцу.

Полученный результат представлен на рисунке 5.

```
Тип данных ID <class 'numpy.int64'>
Тип данных Quantity <class 'numpy.int64'>
Тип данных Price <class 'str'>
Тип данных Hour <class 'numpy.float64'>
Тип данных DayOfWeek <class 'numpy.int64'>
Тип данных ID_category <class 'str'>
```

Рисунок 5 – Типы данных каждого столбца

Проанализировав уникальные значения и типы данных по каждому столбцу, можно сделать вывод, что столбец Price следует преобразовать из формата строки в формат вещественного числа с плавающей запятой, а затем преобразовать до целого числа. Столбец ID_category нужно преобразовать из вещественного признака в числовой. Для этого выделим уникальные значения колонки ID_category, выведем их количество и произведем присваивание каждой категории уникального номера и проведем замену по всем исходным данным. Пример присваивания строковому типу данных численных значений представлен в листинге 5, пример присваиваемых данных в формате «ключ» - «значение» представлен на рисунке 6.

```
categories_ids = {v: i for i, v in enumerate(df['ID_category'].
unique())}
    df['category'] = df['ID_category'].map(lambda x: categories_ids[x])
    print('Categories converted')
good2category = {i[1]['ID_good']: i[1]['category'] for i in
source.iterrows()}
```

Листинг 5 – Пример присваивания значений

```
{'дизайн': 0,
 'топ': 1,
 'гель-лак': 2,
 'база': 3,
 'пилки': 4,
 'масло для кутикулы': 5,
 'инструменты': 6,
 'слайдер': 7,
 'кисти': 8,
 'бонд': 9,
 'сопутствующие': 10,
 'гель-краска': 11,
 'жидкость': 12,
 'для кутикулы': 13,
 'лампа LED': 14,
```

Рисунок 6 – Присвоенные значения

Все преобразования выполняются при помощи модуля converter, представленного в приложении Г. Полученные результаты сохраняются в файле out.csv в рабочей директории проекта. Ну рисунке 7 показан запуск и процесс работы программы converter, а на рисунке 8 отражены обработанные данные, сохраненные в файле out.csv. В качестве разделителя выступает знак запятой.

```
C:\cosmetic_research>python -m converter data/sales.csv
Dataset loaded from data/sales.csv
Prices converted
Time converted
Categories converted
Remove unused columns
Save to file out.csv
Data was successfully converted
```

Рисунок 7 – Запуск и работа программы converter

Где задаваемые параметры:

–m – выбор загружаемого модуля;

data/ – путь к папке, где лежит файл с исходными данными для обработки.

1	price,datetime,quantity,good,category
2	250.0,2016-09-01 00:01:50,1,12810002,0
3	250.0,2016-09-01 00:01:50,1,12810001,0
4	600.0,2016-09-01 00:01:50,1,10730019,1
5	379.99,2016-09-01 00:01:50,1,12570116,2
6	350.0,2016-09-01 00:01:50,1,12480003,1
7	380.0,2016-09-01 00:01:50,1,12570024,2
8	470.02,2016-09-01 00:01:50,1,10730148,2
9	600.0,2016-09-01 00:01:50,1,10730223,3
10	240.0,2016-09-01 00:01:50,1,10750352,2

Рисунок 8 – Данные после конвертации

Добавим колонку день недели и сгруппируем данные по продажа за каждый день и построим соответствующий график. Пример группировки и построения графика представлен в листинге 7, график продаж по каждой категории отображен на рисунке 9.

```
table = source.groupby('DayOfWeek').DayOfWeek.sum()
table = table.sort(inplace=False, ascending=False)
table.head(30).plot(kind='bar')
```

Листинг 7 – Группировка продаж по дню недели

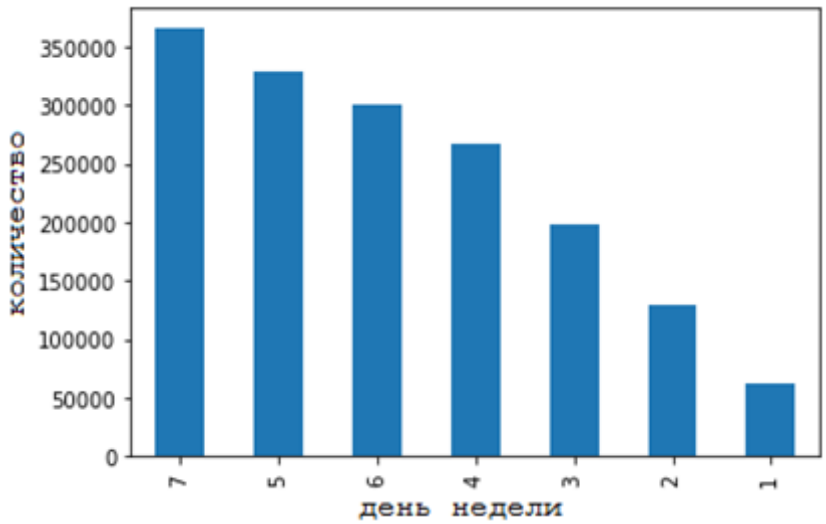

Рисунок 9 – Группировка продаж по дню недели

Из графика, представленного на рисунке 9, можно увидеть, что интернет-магазин работает без фиксированных выходных дней и наибольший объем про-

даж предприятия приходится на воскресенье, а наименьший объем продаж приходится на понедельник.

Аналогично сгруппируем данные и построим график продаж по каждой категории, результат представлен на рисунке 10. И построим график продаж по наиболее часто продаваемому id товара. Построение графика по id товару представлено на рисунке 11.

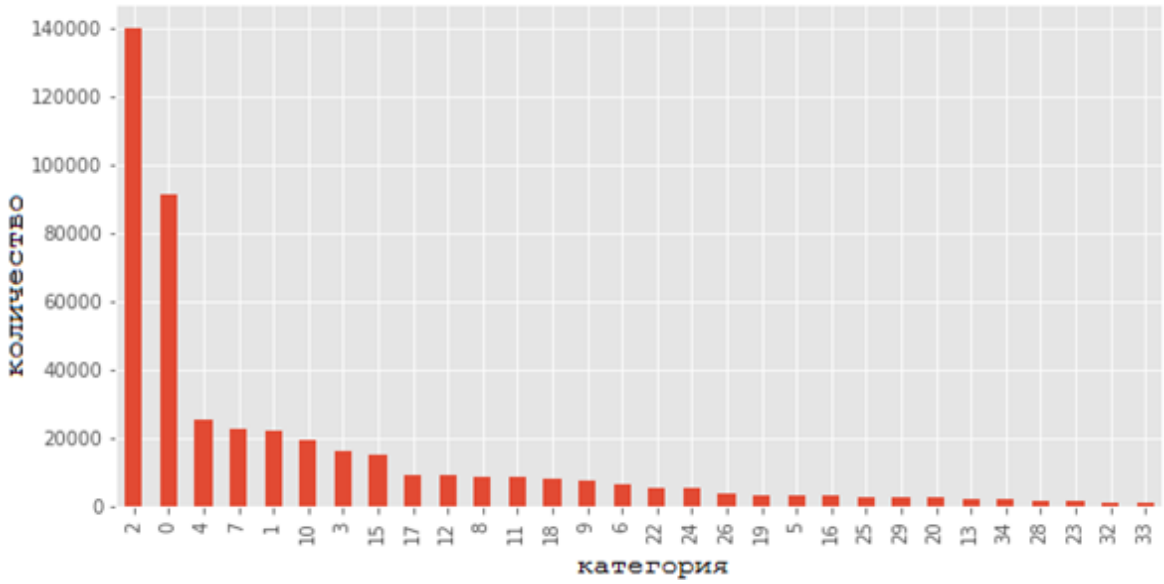

Рисунок 10 – Продажа товаров по категориям

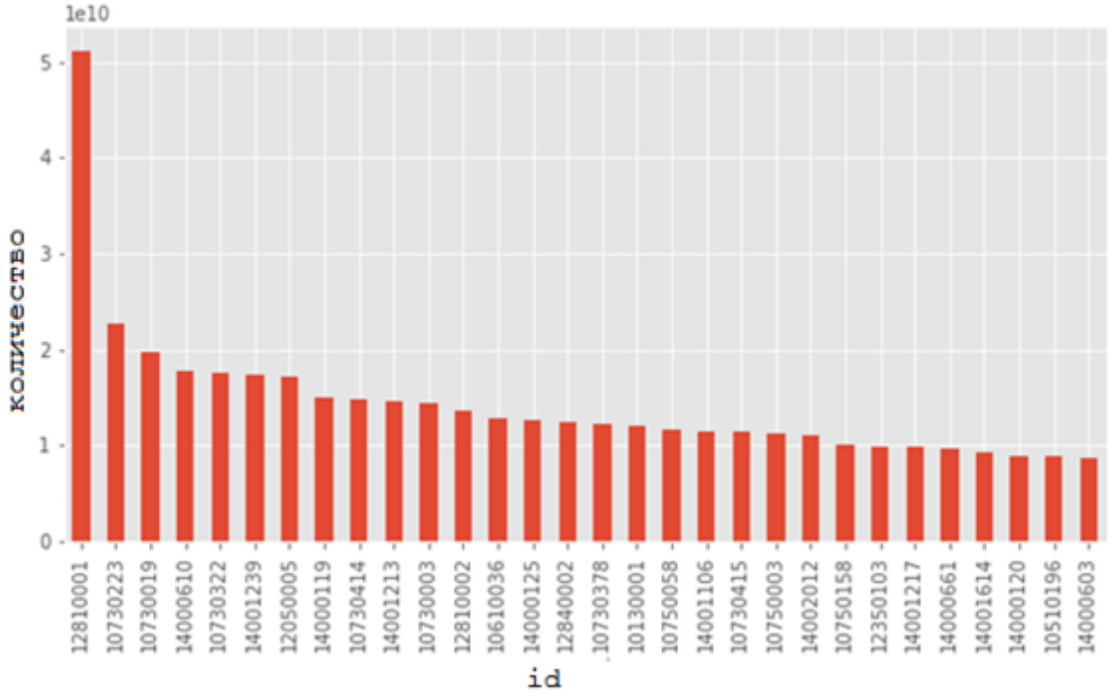

Рисунок 11 – Объем продаж по id

На основании графика, отображенного на рисунке 10 можно сделать вывод, что самой популярной продаваемой категорией является категория под номером 2 (гель-лак). А на основании графика, отображенного на рисунке 11 можно сделать вывод, что самым продаваемым товаром является товар с номером id 1281001. Также на данном графике представлены самые популярные товарные позиции и все они, примерно разделены равномерно.

Также можем проанализировать динамику изменения объема продаж любой категории или id. В качестве примера рассмотрим изменения динамики объема продаж самого продаваемого товара с id 1281001. Для этого создадим словарь данных и сгруппируем данные по продажам данного товара за все дни, указывая весь временной промежуток. Для анализа тенденции изменения продаж конкретного товара нужно обращаться к нему по его id номеру. Пример группировки и построение графика представлены в листинге 8, таблица продаж выбранного товара за каждый день приведена на рисунке 12, график динамики изменения продаж отображен на рисунке 13.

```
df_dict = {k:v.reset_index(name='Quantity')
                for k,v in source.groupby([pd.Grouper(freq='D',
key='DateTime'),'ID_good'])['Quantity'].sum().groupby(level=1)}
df_dict.keys()
df_dict[12810001]
d[:, 12810001].plot()
```

Листинг 8 – Создание словаря DataFram и группировка продаж за каждый день

	DateTime	ID_good	Quantity
0	2016-09-01	12810001	84
1	2016-09-02	12810001	65
2	2016-09-03	12810001	48
3	2016-09-04	12810001	40
4	2016-09-05	12810001	44
5	2016-09-06	12810001	59
6	2016-09-07	12810001	56
7	2016-09-08	12810001	64
8	2016-09-09	12810001	54
9	2016-09-10	12810001	61

Рисунок 12 – Пример продаж за каждый день товара id 1281001

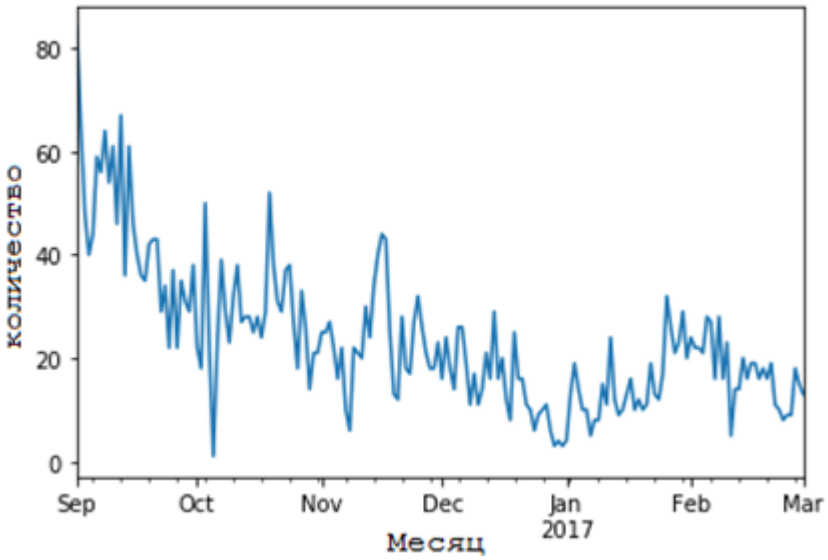

Рисунок 13 – Динамика продаж товара id 1281001

По построенному графику динамики продаж товара id 1281001, отображенного на рисунке 13, можно сделать вывод, что динамика продаж уменьшается. Аналогично можно просмотреть динамику продаж любой категории или id товара.

3.2 Прогнозирование объема продаж

Воспользуемся модулем shopsales и составим прогноз на несколько временных промежутков с применением разных алгоритмов методов машинного обучения, таких как: случайный лес, градиентный бустинг, метод опорных векторов. Составление прогнозе можно получить как в командной строке, так и сохранить в виде отдельного файла. Пример запуска программы с заданием параметров, определением типа данных и использованием памяти представлен на рисунке 14.

```
C:\cosmetic_research>python -m shopsales --path out.csv --method random_forest -
--days 7 --split 0.2

Loaded dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 428573 entries, 0 to 428572
Data columns (total 5 columns):
price       416250 non-null float64
datetime    428573 non-null datetime64[ns]
quantity    428573 non-null int64
good        428573 non-null int64
category    428573 non-null int64
dtypes: datetime64[ns](1), float64(1), int64(3)
memory usage: 16.3 MB
```

Рисунок 14 – Запуск программы с заданными параметрами

Где задаваемые параметры:

–m – выбор загружаемого модуля;

—path – путь (наименование) файла;

—method – выбор метода прогнозирования (random_forest, gradient_boosting, svm);

—days – по сколько дней берем выборку для обучения;

—split – разделение данных на тестовую и тренировочную выборку.

—target – объект прогноза.

Пример результата работы программы прогноза объема количества продаж товара в последующие 16 месяцев приведен на рисунке 15, ошибки прогноза на рисунке 16.

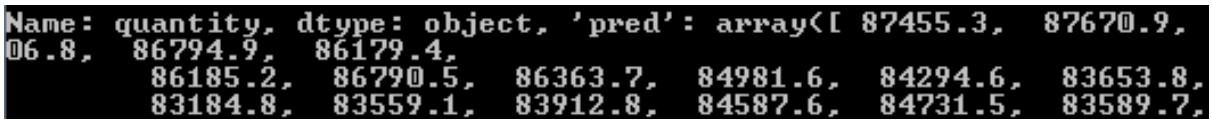

Рисунок 15 – Объем продаж на 16 месяцев

Рисунок 16 – Ошибка прогноза от тестовой выборки

Ошибки прогноза рассчитываются по следующим формулам:

Абсолютная средняя ошибка:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_t - \hat{y}_t|}{y_t},$$

где:

y_t – фактический объем продаж за анализируемый период;

$\hat{y}_t$ – значение прогнозной модели за анализируемый период;

n – количество периодов.

Процент средней ошибки вычисляется простым усреднением ошибок на каждом шаге:

$$\bar{e} = \frac{\sum e}{n},$$

где:

e – ошибка прогнозируемой величины от реальной;

n – количество периодов.

Среднеквадратичное отклонение:

$$S = \sqrt{\frac{\sum_{i=1}^{n}(x - \bar{x})^2}{n}},$$

где:

$x -$ отдельные значения;

$\bar{x} -$ среднее арифметическое по выборке;

n – количество периодов.

Выведем ошибки прогноза трех реализуемых методов (случайного леса, градиентного бустинга и метода опорных векторов). Результат работы программы salesPredict с заданным параметром –method all (выводящий ошибки по все методам) представлен на рисунке 17.

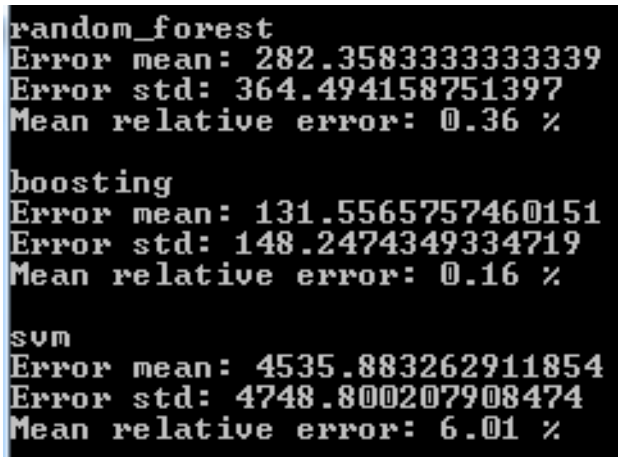

Рисунок 17 – Ошибки при обучении алгоритмов

Проведя ряд тестов с фиксированным days и измененным split и наоборот можно убедиться, что наиболее эффективным алгоритмом для составления прогноза объемов продаж является алгоритм метода градиентного бустинга.

Выводы по главе три

В результате вычислительно эксперимента приведены возможности программных модулей и составлены графики, отображающие тенденции развития то-

варных категорий или отдельных товаров. В ходе проведения эксперимента были получены ошибки отклонения реальных значений от прогнозируемых. В целом, все эти ошибки достаточно низкие, что говорит о хорошей начальной подготовки данных, которые были обработаны в модуле converter, и о возможности применения всех построенных моделей на практике.

После сравнения результатов полученных в ходе вычислительного эксперимента, можно сделать вывод: наиболее эффективным алгоритмом для прогнозирования объемов продаж является алгоритм градиентного бустинга, так как его отклонение прогнозируемых данных от реальных наиболее незначительное и не превышает 2%. Самые высокие ошибки показал метод опорных векторов. Стоит отметить, что все протестированные алгоритмы имеют небольшую ошибку, которая не превышает 10%. Это значит, что данные алгоритмы можно применять на практике и они являются достаточно эффективными. Уменьшить ошибку возможно с внедрением оптимизационных параметров.

ЗАКЛЮЧЕНИЕ

В работе рассмотрены основные методы прогнозирования объемов продаж. Задача прогнозирования разбивается на два этапа. На первом этапе на основании набора данных с известными результатами строится модель. На втором этапе она используется для предсказания результатов на основании новых наборов данных. При этом требуется, чтобы построенные модели работали максимально точно. Степень достоверности прогнозов можно сравнить с реальными показателями и, сделав выводы, приступить к следующему прогнозу уже с существующими данными.

Для задачи прогнозирования объемов продаж подходят такие алгоритмы, как случайный лес, градиентный бустинг, метод опорных векторов.

Разработанная система прогнозирования состоит из трех модулей. Данные модули автономными и используются для разных целей, а именно:

Модуль analysis позволяет провести предварительный анализ входных данных, оценить корректность их заполнения, проверить тип, удалить ненужные данные, очистить или заполнить пустые значения, провести анализ и отследить тенденции развития товара или товарной группы, выделить наиболее часто продаваемые товары, выделить режим работы предприятия, построить графики для визуализации тенденций продаж.

Модуль converter позволяет обработать исходные данные и привести их к соответствующему виду использования при дальнейшем обучении. Данные проверяются на корректность и обрабатываются возможные ошибки заполнения данных, такие как: незаполненные части, нулевые значения, не подходящий для обучения тип данных, замена строковых обозначений численными, разбиение даты на отдельные столбцы (год, месяц, день). Полученный файл сохраняется под названием out формата csv в рабочей директории.

Модуль predictionSales позволяет подготовить данные (разбить на тестовую и тренировочную выборку) и обучить данные на ряде моделей (случайный лес, градиентный бустинг, метод опорных векторов) с возможностью изменения пара-

метров (интервал прогнозирования, разбиение выборки, объект обучения), вывести ошибку отклонения прогнозных значений от реальных.

Модуль predictionSales выдает прогноз на заданный временной промежуток. Он принимает на вход данные о объемах продаж, проводит первичный анализ данных, приведение данных к виду пригодному для обучения и обучает алгоритм, выдавая прогноз и оценивая его качество. На данных о продажах интернет-магазина был проведен вычислительный эксперимент, продемонстрировавший корректность работы при различных задаваемых параметрах. В ходе проведения эксперименты были получены ошибки отклонения реальных значений от прогнозируемых. В целом, все эти ошибки достаточно низкие, что говорит о хорошей начальной подготовки данных и о возможности применения всех построенных моделей на практике.

Наиболее эффективным алгоритмом является алгоритм градиентного бустинга. Это обусловлено тем, что алгоритм на каждой итерации строит базовый алгоритм, который действительно эффективен лишь на части подвыборки. На каждом шаге алгоритма новое слагаемое считается опираясь не на всю обучающую выборку, а лишь на случайную подвыборку фиксированного размера.

Правильно составленный прогноз увеличивает эффективность ведения бизнеса путем контроля и оптимизации расходов, что, в свою очередь, помогает сформировать оптимальные (а не завышенные или заниженные) запасы продукции на складе.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК:

1. Наркевич, Л.В. Прогнозирование объема продаж торговой организации с учетом колебаний спроса // Л.В. Наркевич, К.П. Боровикова- Вестник Белорусско – Российского университета, 2013. – 123 с.

2. Spielmaker, K. J. On Shelf Availability: A Literature Review & Conceptual Framework / K. J. Spielmaker – University of Arkansas, 2012. – 20 p.

3. Frey, C.B. The future of employment: how susceptible are jobs to computerization /C. B. Frey, M. A. Osborne – Department of Engineering Science, 2013. – 72 p.

4. Caruana, R. An Empirical Comparison of Supervised Learning Algorithms / R. Caruana, A. Niculescu-Mizil. – ICML '06 Proceedings of the 23rd international conference on Machine learning, 2006. – 8 p.

5. Jacques. W., Comparison of 14 different families of classification algorithms on 115 binary datasets / W. Jacques. – 2016. – 36 p.

6. Hastie, T. Chapter 15. Random Forests / The Elements of Statistical Learning: Data Mining, Inference, and Prediction. / T. Hastie, R. Tibshirani, J. Friedman – Springer-Verlag, 2009. – 746 p.

7. Басовский, Л.Е. Прогнозирование и планирование в условиях рынка: Учебное пособие / Л.Е. Басовский – М.: ИНФРА-М, 2007. – 223 с.

8. Breiman, L. Random forests // Machine Learning. — 2001. – 32 p.

9. Siroky D. Navigating Random Forests and related advances in algorithm modeling // Statictic Surveys, 2009. – 163 p.

10. Friedman, J.H. Greedy Function Approximation: A Gradient Boosting Machine. // Annals of Statistics, 2001. – 1123 p.

11.Садовникова, Н.А. Анализ временных рядов и прогнозирование. Учебное пособие / Н.А. Садовникова, Р.А. Шмойлова. – Москва: Московский государственный университет экономики, статистики и информатики, 2001. – 67 с.

12.Четыркин, Е.М. Статистические методы прогнозирования. Учебное пособие / Е.М. Четыркин – М.: Статистика, 1975. – 183 с.

13.Афанасьев, В.Н. Анализ временных рядов и прогнозирование: Учебник / В.Н. Афанасьев, М.М. Юзбашев. – М.: ФиС, ИНФРА-М, 2012. – 320 с.

ПРИЛОЖЕНИЕ А

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

Прогнозирование объемов продаж для интернет-магазина
с помощью методов машинного обучения
ТЕХНИЧЕСКОЕ ЗАДАНИЕ
НА РАЗРАБОТКУ ПРОГРАММНОГО КОМПЛЕКСА «ПРОГНОЗ»
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
_____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

1 ВВЕДЕНИЕ

1.1 Наименование программного изделия

Полное наименование программы – «Программа прогнозирования объемов продаж с помощью методов машинного обучения для интернет-магазина». Краткое наименование – программа прогноза объема продаж.

1.2 Область применения

Программа предназначена для построения прогноза объемов продаж в указанном временном промежутке (от нескольких дней до года) и предварительной обработке данных, которые используются для построения прогноза. Программа может применяться для работы с файлами, сохраненными в текстовом формате, предназначенный для представления табличных данных (csv формат), при условии структурированности данных: конкретизации данных в отдельные группы/столбцы, наличие разделяющих символов между данными.

2 ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

2.1 Документ, на основании которого ведется разработка

Программа прогноза объемов продаж разрабатывается в выпускной квалификационной работе в соответствии с учебным планом кафедры МиКМ («Математическое и компьютерное моделирование») ЮУрГУ.

2.2 Организация, утвердившая этот документ, и дата его утверждения

Задание на выпускную квалификационную работу и разработку программного обеспечения утверждено старшим преподавателем кафедры Богушовым А. К.

2.3 Наименование темы разработки

Наименование темы разработки – SalesForecast.

3 НАЗНАЧЕНИЕ РАЗРАБОТКИ

Разработка является выпускной квалификационной работой.

4 ТРЕБОВАНИЯ К ПРОГРАММЕ

4.1 Требования к функциональным характеристикам

4.1.1 Состав выполняемых функций

4.1.1.1 Программа должна предусматривать выполнение посредством вызова из консоли с передачей параметров либо путем импорта библиотеки.

4.1.1.2 Программа прогноза объемов продаж должна обеспечить предварительную обработку уже имеющихся данных и на их основании построить прогноз объема продаж на конкретный срок.

4.1.2 Организация входных и выходных данных

При вызове программы посредством консоли на вход подаются аргументы:

- --path – путь к файлу, содержащий данные после обработки в модуле converter. Файл должен иметь расширение csv;
- --target –величина, представляющая собой прогнозируемый объект (quantity или sales);
- --group – выбор по каким колонкам группируются объекты (category, id, all);
- --method – выбор метода прогнозирования (random_forest, gbm, svrr);
- --days – по сколько дней разбивается выборка;
- --split – в каком соотношении разбивается исходная выборка на обучающую и тестовую.

Выходные данные должны представлять собой прогноза объема продаж в единицах выбранной прогнозируемой величины.

4.1.3 Временные характеристики и размер занимаемой памяти

Время выполнения программы и объем занимаемой памяти зависит от объема входных данных.

4.2 Требования к надежности

4.2.1 Требования к надежному функционированию

Программа должна нормально функционировать при бесперебойной работе ЭВМ.

4.2.2 Контроль входной и выходной информации

В программе должен присутствовать контроль формата входных данных в виде модуля converter для предобработки исходных данных.

4.2.3 Время восстановления после отказа

После отказа программа может требовать повторного ввода данных и времени на выполнение.

4.3 Условия эксплуатации

Программа должна храниться в виде двух маркированных копий: эталонной и рабочей. Периодическая перезапись информации должна осуществляться согласно вынесенной маркировке. Условия хранения дисков с программой должны соответствовать нанесенной на них маркировке.

4.4 Требования к составу и параметрам технических средств

Программа должна корректно работать на персональном компьютере с процессором Pentium и выше.

4.5 Требования к информационной и программной совместимости

4.5.1 Требования к информационным структурам на входе и выходе

Требования к информационным структурам на входе и выходе определены в п.4.1.2.

4.5.2 Требования к методам решения

Требования к методам решения определены в п.4.1.1.1. Выбор остальных методов решения осуществляется разработчиком без согласования с заказчиком.

4.5.3 Требования к языкам программирования

Программа должна быть написана на языке программирования Python 3.5.

4.5.4 Требования к программным средствам, используемым программой

Для работы необходима операционная система Windows XP и выше либо Linux. Должен быть предустановлен интерпретатор языка Python версии 3.5. Также должны быть установлены библиотеки numpy, scipy и scikit-learn, pandas.

4.6 Требования к маркировке и упаковке

Диски с эталонным и рабочим экземплярами программы должны иметь маркировку, состоящую из надписи «SalesPrediction», надписи «эталон» или «рабочая», даты последней перезаписи программы. Упаковка должна соответствовать условиям хранения диска. На упаковке должны быть указаны условия транспортировки и хранения диска.

4.7 Требования к транспортировке и хранению

Условия транспортировки и хранения должны соответствовать п.4.6.

5 ТРЕБОВАНИЕ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

Состав программной документации должен включать следующие документы:

1) Технический проект программы по ГОСТ 19.404-79 в машинном исполнении;

2) Описание программы по ГОСТ 19.402-78 на компакт-диске;

3) Текст программы по ГОСТ 19.401-78 на компакт-диске;

4) Руководство программиста по ГОСТ 19.504-79 на компакт-диске в виде файла README.TXT.

Пояснительная записка «Технический проект программы» должна содержать следующие разделы.

1. Раздел «Входные данные» (характер, организация и предварительная подготовка входных данных).

2. Раздел «Выходные данные» (характер и организация выходных данных).

3. Раздел «Описание логической структуры» при технологии объектно-ориентированного программирования должен включать следующие материалы:

- Описание связей программы с другими программами;
- Описание внутренних массивов и переменных, которые используются в межмодульном обмене данных;
- Расшифровка наименований модулей;
- Описание функционирования программы с учетом ее модульного деления;
- Описание модулей программы.

4. Раздел «Используемые технические средства» (типы ПК, на которых возможно выполнения программы; устройства, используемые при выполнении программы).

5. Раздел «Вызов и загрузка» (виды носителей программы, их используемый объем; способы вызова программы с соответствующих носителей данных; входные точки в программу – запуск программы).

6. Раздел «План мероприятий по разработке и внедрению программы».

6 ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

Технико-экономические показатели должны определяться заказчиком без участия исполнителя.

7 СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

Разработка программы должна выполняться по следующим этапам:

1) Разработка, согласование и утверждение технического проекта программы с пояснительной запиской – 5 недель;

2) Разработка рабочего проекта программы с комплексным тестированием – 6 недель;

3) Приемка-сдача с исправлением обнаруженных недостатков в программе и программной документации – 2 недели;

4) Внедрение.

8 ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ

8.1 Виды испытаний

Испытания программы и верификация документации должны проводиться в присутствии заказчика. Проверочные тесты должны готовиться заказчиком.

8.2 Общие требования к приемке

Приемка программы должна осуществляться заказчиком. Программа должна считаться годной, если она удовлетворяет всем пунктам данного технического задания.

ПРИЛОЖЕНИЕ А

ПРИМЕР ИСХОДНЫХ ДАННЫХ

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

Прогнозирование объемов продаж для интернет-магазина
с помощью методов машинного обучения
ПРИМЕР ИСХОДНЫХ ДАННЫХ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
_____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

ПРИЛОЖЕНИЕ Б. Пример исходных данных.

ID_order	DateTime	ID_good	Quantity	Price	Hour	DayOfWeek	ID_category
0000-034809	01.09.2016 0:01	12810002	1	250		4	дизайн
0000-034809	01.09.2016 0:01	12810001	1	250		4	дизайн
0000-034809	01.09.2016 0:01	10730019	1	600		4	топ
0000-034809	01.09.2016 0:01	12570116	1	379,99		4	гель-лак
0000-034809	01.09.2016 0:01	12480003	1	350		4	топ
0000-034809	01.09.2016 0:01	12570024	1	380		4	гель-лак
0000-034809	01.09.2016 0:01	10730148	1	470,02		4	гель-лак
0000-034809	01.09.2016 0:01	10730223	1	600		4	база
0000-034809	01.09.2016 0:01	10750352	1	240		4	гель-лак
0000-034808	01.09.2016 0:02	10240006	1	60		4	пилки
0000-034808	01.09.2016 0:02	10160015	1	99		4	масло для кутикулы
0000-034808	01.09.2016 0:02	10230025	1	59		4	пилки
0000-034808	01.09.2016 0:02	10180001	1	137		4	инструменты
0000-034808	01.09.2016 0:02	10170026	1	589		4	инструменты
0000-034808	01.09.2016 0:02	10230023	1	59		4	пилки
0000-034808	01.09.2016 0:02	10250006	1	224		4	инструменты
0000-034807	01.09.2016 0:08	12160151	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	12160204	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	12160205	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	14000269	1			4	дизайн
0000-034807	01.09.2016 0:08	12160203	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	12160206	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	12160085	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	12160075	1	179		4	гель-лак
0000-034807	01.09.2016 0:08	12160214	1	179		4	гель-лак
0000-034806	01.09.2016 0:09	10120032	1	50		4	слайдер
0000-034806	01.09.2016 0:09	10120725	1			4	слайдер
0000-034806	01.09.2016 0:09	10210048	1	39		4	кисти
0000-034806	01.09.2016 0:09	10120732	1			4	слайдер
0000-034806	01.09.2016 0:09	10121011	1	50		4	слайдер
0000-034806	01.09.2016 0:09	14000140	1	250		4	дизайн
0000-034806	01.09.2016 0:09	10120044	1	50		4	слайдер
0000-034806	01.09.2016 0:09	10120781	1			4	слайдер
0000-034806	01.09.2016 0:09	10120744	1			4	слайдер
0000-034806	01.09.2016 0:09	10120679	1	50		4	слайдер
0000-034806	01.09.2016 0:09	14000130	1	250		4	дизайн
0000-034806	01.09.2016 0:09	10120283	1	50		4	слайдер
0000-034806	01.09.2016 0:09	14000119	1	250		4	дизайн
0000-034806	01.09.2016 0:09	10120814	1	50		4	слайдер
0000-034806	01.09.2016 0:09	14000121	1	250		4	дизайн
0000-034806	01.09.2016 0:09	12810001	1	250		4	дизайн
0000-034806	01.09.2016 0:09	14000143	1	250		4	дизайн
0000-034806	01.09.2016 0:09	10121027	1	50		4	слайдер
0000-034806	01.09.2016 0:09	10120667	1	50		4	слайдер
0000-034806	01.09.2016 0:09	10121054	1	50		4	слайдер
0000-034806	01.09.2016 0:09	10120426	1	50		4	слайдер
0000-034806	01.09.2016 0:09	10120030	1	50		4	слайдер
0000-034806	01.09.2016 0:09	14000126	1	250		4	дизайн

ПРИЛОЖЕНИЕ В

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

Прогнозирование объемов продаж для интернет-магазина
с помощью методов машинного обучения
РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ
К РАЗРАБОТКЕ ПРОГРАММНОГО КОМПЛЕКСА «ПРОГНОЗ»
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
_____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

1 ОБЩИЕ СВЕДЕНИЯ

Программа предназначена для построения прогноза объемов продаж в указанном временном промежутке (от нескольких дней до года) и предварительной обработке данных, которые используются для построения прогноза. Программа может применяться для работы с файлами, сохраненными в текстовом формате, предназначенный для представления табличных данных (csv формат), при условии структурированности данных: конкретизации данных в отдельные группы/столбцы, наличие разделяющих символов между данными.

2 ПРАВИЛА ВВОДА ВХОДНЫХ ДАННЫХ

При запуске программы из консоли необходимо ввести параметры, передающие входные данные. Пример запуска программы с заданием параметров, определением типа данных и использованием памяти представлен на рисунке 1.

Обязательным параметром является filename – путь к файлу формата csv с данными, обработанными в модуле converter.

При вызове программы посредством консоли на вход подаются аргументы:

- --path – путь к файлу, содержащий данные после обработки в модуле converter. Файл должен иметь расширение csv;
- --target –величина, представляющая собой прогнозируемый объект (quantity или sales);
- --group – выбор по каким колонкам группируются объекты (category, id, all);
- --method – выбор метода прогнозирования (random_forest, gbm, svrr);
- --days – по сколько дней разбивается выборка;
- --split – в каком соотношении разбивается исходная выборка на обучающую и тестовую.

Выходные данные должны представлять собой прогноза объема продаж в единицах выбранной прогнозируемой величины.

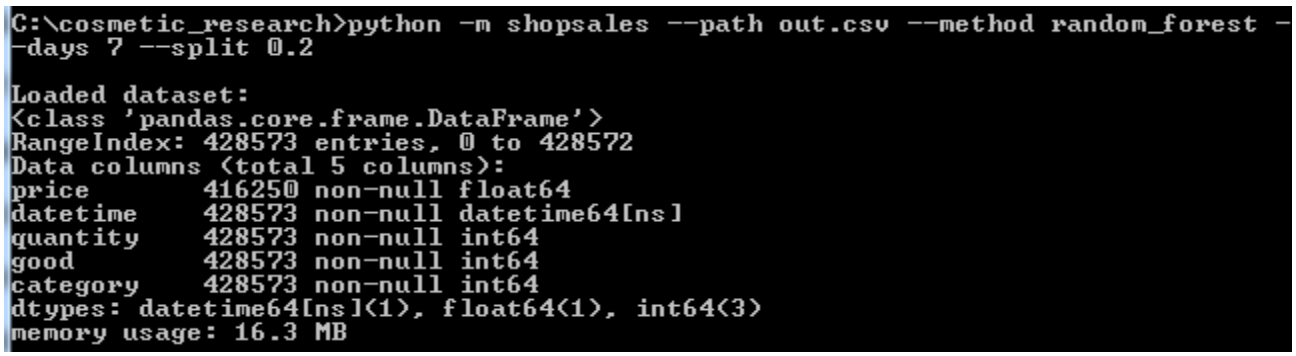

Рисунок 1 – Пример запуска программы с задаваемыми параметрами

3 РАБОТА С ПРОГРАММОЙ

Пользователю предоставляется возможность работать с программой посредством вызова из консоли.

В случае работы с использованием консоли необходимо ввести входные данные так, как описано в главе 3. Выходные данные будут выведены в файл out.csv.

ПРИЛОЖЕНИЕ Г

АНАЛИЗ ИСХОДНЫХ ДАННЫХ

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

Прогнозирование объемов продаж для интернет-магазина
с помощью методов машинного обучения
АНАЛИЗ ИСХОДНЫХ ДАННЫХ
РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА «ПРОГНОЗ»
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
_____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

ПРИЛОЖЕНИЕ Г. Анализ исходных данных.

Файл dataAnalysis.py

```
%matplotlib inline
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
plt.style.use('ggplot')
plt.rcParams['figure.figsize'] = (10, 5)

# загрузка и вывод на экран исходных данных с учетом разделителя
data = pd.read_csv(r"D:\ucheba\diplom\sales\proba\sales.csv", sep='|')

# просматриваем столбец ID_category на его уникальные значения и их количе-
ство
print(data.ID_category.unique())
print(len(data.ID_category.unique()))

# отрезаем от столбца даты часы и создаем столбец с исправленной датой
data_new = data.copy()
data_new['Year'] = data_new.DateTime.apply(lambda x: x.split(' '))
data_new['GoodTime'] = data_new.Year.apply(lambda x: (x[0]))
# удаляем ненужные столбцы
data_new=data_new.drop(["DateTime"], axis = 1)
data_new=data_new.drop(["Year"], axis = 1)
data_new=data_new.drop(["ID_order"], axis = 1)
data_new=data_new.drop(["Hour"], axis = 1)

# разбиваем дату на 3 новых колонки: год, месяц, день
data_new['Year'] = data_new.GoodTime.apply(lambda x: x.split('.'))
data_new['Month'] = data_new.Year.apply(lambda x: int(x[1]))
data_new['Day'] = data_new.Year.apply(lambda x: int(x[0]))
data_new['Year'] = data_new.Year.apply(lambda x: int(x[2]))

# по каждому столбцу выводим уникальные значения и их количество
print("Уникальные значения ID")
print(data_new.ID_good.unique())
print("Всего уникальных значений:"+ str(len(data.ID_good.unique())))
print("")
print("Уникальные значения количества")
print(data_new.Quantity.unique())
print("Всего уникальных значений:"+ str(len(data.Quantity.unique())))
print("")
print("Уникальные значения цены")
print(data_new.Price.unique())
print("Всего уникальных значений:"+ str(len(data.Price.unique())))
print("")
print("Уникальные значения времени")
print(data_new.Hour.unique())
```

```
print("Всего уникальных значений:"+ str(len(data.Hour.unique())))
print("")
print("Уникальные значения дня недели")
print(data_new.DayOfWeek.unique())
print("Всего уникальных значений:"+ str(len(data.DayOfWeek.unique())))
print("")
print("Уникальные значения категории товара")
print(data_new.ID_category.unique())
print("Всего уникальных значений:"+ str(len(data.ID_category.unique())))
print("")
#выводим тип данных по каждой категории
print ("Тип данных ID " + str(type(data_new.ID_good[0])))
print ("Тип данных Quantity " + str(type(data_new.Quantity[0])))
print ("Тип данных Price " + str(type(data_new.Price[0])))
print ("Тип данных DayOfWeek " + str(type(data_new.DayOfWeek[0])))
print ("Тип данных ID_category " + str(type(data_new.ID_category[0])))

# меняем строковый тип данных на float
data_new['Price'] = data_new['Price'].map(lambda x: float(x.replace(',',
'.').replace(' ', '')) if type(x) == str else x)
# удаление пустых значений
data_new['ID_category'] = data_new['ID_category'].replace('      ', np.nan,
regex=True)
data_new['ID_category'] = data_new['ID_category'].dropna(axis=0, how='any')

# присваивание каждому строковому значению в столбце «категория» численного
«ключа»
categories_ids = {v: i for i, v in enumerate(source['ID_category']
.unique())}
source['category'] = source['ID_category'].map(lambda x: categories_ids[x])
good2category= {i[1]['ID_good']: i[1]['category'] for i in
source.iterrows()}
# группировка и построение графика самого продаваемого товара по id
table = names.groupby('Good_id').quantity.sum()
table = table.sort(inplace=False, ascending=False)
table.head(30).plot(kind='bar')
# группировка и построение графика дня недели с наибольшими продажами
table = names.groupby('weekday').quantity.sum()
table = table.sort(inplace=False, ascending=False)
table.head(30).plot(kind='bar')
```

ПРИЛОЖЕНИЕ Д

КОНВЕРТИРОВАНИЕ ИСХОДНЫХ ДАННЫХ

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

Прогнозирование объемов продаж для интернет-магазина
с помощью методов машинного обучения
КОНВЕРТИРОВАНИЕ ИСХОДНЫХ ДАННЫХ
РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА «ПРОГНОЗ»
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
_____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

ПРИЛОЖЕНИЕ Д. Конвертирование исходных данных.

Файл main.py

```
from argparse import ArgumentParser
import importlib

parser = ArgumentParser(description='Converts client dataset')

# ключи для получения информации в консольном окне
parser.add_argument('data', default='in.csv', help='Input file')
parser.add_argument('--out', '-o', default='out.csv', help='Output file')
parser.add_argument('--company', default='imkosmetik', help='Company name')

args = parser.parse_args()
converter = importlib.import_module('converter.' + args.company)
converter.run(args.data, args.out)
print('Data was successfully converted')
```

Файл converter.py

```
import pandas as pd
# загрузка исходных данных и исправление кодировки
def run(input_path, output_path):
    df = pd.read_csv(input_path, sep=';', encoding='cp1251')
    print('Dataset loaded from {}'.format(input_path))
# преобразование строкового типа в float
df['price'] = df['Price'].map(lambda x: float(x.replace(',', '.').replace
(' ', '')) if type(x) == str else x)
    print('Prices converted')
    df['datetime'] = pd.to_datetime(df['DateTime'], dayfirst=True).
    print('Time converted')
    df['quantity'] = df['Quantity']
    df['good'] = df['ID_good']
    # преобразование обозначения категории в число
    categories_ids = {v: i for i, v in enumerate(df['ID_category']
.unique())}
    df['category'] = df['ID_category'].map(lambda x: categories_ids[x])
    print('Categories converted')
    # удаление исходных столбцов и замена их конвертированными
    print('Remove unused columns')
    df = df.drop(['Price', 'DateTime', 'Hour', 'DayOfWeek', 'ID_order',
'Quantity', 'ID_category', 'ID_good'], axis=1)
    #сохранение полученного файла
    print('Save to file {}'.format(output_path))
    df.to_csv(output_path, index=False)
```

ПРИЛОЖЕНИЕ Е

СОСТАВЛЕНИЕ ПРОГНОЗА

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Кафедра «Математическое и компьютерное моделирование»

Прогнозирование объемов продаж для интернет-магазина
с помощью методов машинного обучения
СОСТАВЛЕНИЕ ПРОГНОЗА
РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА «ПРОГНОЗ»
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ– 01.03.02.2017.130.16.000 ВКР

Нормоконтролер,
Доцент каф. МиКМ, к.ф.-м.н.,
_____Т.А. Макаровских
_____2017 г.

Руководитель работы,
Старший преподаватель,
_____А.К.Богушов
____2017 г.

Автор работы
Студентка группы ЕТ-485
_____Е.В.Степанченко
_____2017 г.

Челябинск, 2017

ПРИЛОЖЕНИЕ Е. Прогнозирование.

Файл main.py

```
from argparse import ArgumentParser
import shopsales
from shopsales.enums import Target, TrainMethod, GroupData
from shopsales.data import load_data
from shopsales.model import MultiModel

parser = ArgumentParser(prog=shopsales.__name__, description='ML applica-
tion')
# ключи для получения информации в консольном окне
parser.add_argument('--version', action='version', version='%(prog)s
{}'.format(shopsales.version))
parser.add_argument('--path', default='data.csv', help='Path csv file with
data')
# задание объекта прогнозирования
parser.add_argument('--target', default=Target.quantity.name,
                    choices=[i.name for i in Target], help='Forecasting
target')
# выделение группировки
parser.add_argument('--group', default=GroupData.all.name,
                    choices=[i.name for i in GroupData], help='Groups')
parser.add_argument('--method', default=TrainMethod.all.name,
                    choices=[i.name for i in TrainMethod], help='Train
method')
# интервал прогнозирования
parser.add_argument('--days', default=30, help='Forecasting interval')
# разбиение выборки
parser.add_argument('--split', default=0.2, help='Split rate')

args = parser.parse_args()
data = load_data(args.path)
print('\nLoaded dataset:')
data.info()

target = Target[args.target]
method = TrainMethod[args.method]
group = GroupData[args.group]
model = MultiModel(data, target, method, group)

model.fit()
model.stats()
```

Файл data.py

```
# обработка даты
import pandas as pd
```

```
def load_data(path):
    data = pd.read_csv(path)
    if 'datetime' in data.columns:
        data['datetime'] = pd.to_datetime(data['datetime'])

    return data
```

Файл enums.py

```
from enum import IntEnum

class Target(IntEnum):
    quantity = 1
    sales = 2

class TrainMethod(IntEnum):
    all = 0
    random_forest = 1
    gradient_boosting = 2
    svm = 3

class GroupData(IntEnum):
    all = 0
    category = 1
```

Файл model.py

```
from collections import namedtuple
import pandas as pd
from sklearn.ensemble.forest import RandomForestRegressor
from sklearn.ensemble.forest import GradientBoostingRegressor
from sklearn.ensemble.forest import svm
from sklearn.metrics import mean_squared_error
from shopsales.enums import TrainMethod, Target

class MultiModel:
    def __init__(self, data, target: Target, method: TrainMethod, days=30,
split=0.2):
        self.data = data
        self.split_rate = 0.2
        self._target = target
        self.method = method
        self.days = days
        self.split = split
        self.fit_result = None

    def fit(self):
        if self.method == TrainMethod.random_forest:
```

```
            model = RandomForestRegressor()
        train, test = self.train_test()
        target = self._target.name
        x = train[[i for i in train.columns.values if i != target]]
        y = train[target]
        model.fit(x, y)

        target = self._target.name
        x = test[[i for i in test.columns.values if i != target]]
        y_real = test[target]
        y_pred = model.predict(x)
        return {
            'real': y_real,
            'pred': y_pred,
            'rmse': mean_squared_error(y_real, y_pred),
        }
    print(type(pd.Grouper(key='datetime', freq='D')))

    def train_test(self):
#группировка количества проданного товара по дням
        groups_by_day = self.data.groupby([pd.Grouper(key='datetime',
freq='D')])
        quantity_by_day = groups_by_day['quantity'].sum()
        n = quantity_by_day.size - self.days

        train = pd.DataFrame(columns=['start_day_of_year', 'quantity'])
        for i in range(n):
            interval = quantity_by_day[i:i + self.days]
            train.loc[i] = [interval.index[0].dayofyear, interval.sum()]

        n = int(train.size * self.split)
        train = train[:-n]
        test = train[-n:]

        return train, test
```