САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет

Кафедра Системного Программирования

Кутькин Никита Андреевич

Использование методов машинного обучения для предсказания оттока клиентов телекоммуникационной компании

Бакалаврская работа

Допущена к защите.
Зав. кафедрой:
д. ф.-м. н., профессор Терехов А. Н.

Научный руководитель:
д. ф.-м. н., профессор Терехов А. Н.

Рецензент:
ведущий разработчик ООО "НМТ" Невоструев К. Н.

Санкт-Петербург
2015

SAINT-PETERSBURG STATE UNIVERSITY
Mathematics & Mechanics Faculty

Chair of Software Engineering

Nikita Kutkin

Using machine learning methods for predicting customer churn in telecommunication industry

Graduation Thesis

Admitted for defence.
Head of the chair:
professor Andrey Terekhov

Scientific supervisor:
professor Andrey Terekhov

Reviewer:
senior developer at "NMT" LLC Constantin Nevostruev

Saint-Petersburg
2015

Оглавление

Введение

В настоящее время в телекоммуникационной индустрии удержание существующих пользователей стоит дешевле чем привлечение новых. В связи с этим, важной является задача определения пользователей с высоким риском отказа от предоставляемых услуг: зная о том, что пользователь в ближайшее время собирается перейти к конкурентам, его можно попробовать удержать.

Для этого необходимо выявить скрытые закономерности между лояльностью пользователя и обезличенной информацией о нем (персональные данные, активность, трафик). Для подобных задач широко применяются методы анализа данных и машинного обучения.

В терминах машинного обучения данная задача может быть сведена к задаче бинарной классификации. Пользователи могут принадлежать одному из двух классов: собирающиеся или не собирающиеся в ближайшее время отказаться от предоставляемых услуг, а из персональных данных и трафика пользователя выделяется набор входных атрибутов. Для решения задачи требуется создание классификатора, который присваивает каждому набору входных атрибутов значение метки одного из классов. Классификатор входных значений является результатом работы этапа «обучения», в процессе которого на вход обучающего алгоритма подаются данные с уже известными значениями классов. Также, стоит отметить большое количество различных алгоритмов построения классификаторов и их входных параметров. Выбор наиболее подходящего алгоритма и его параметров – одна из основных сложностей любой задачи машинного обучения.

В данной работе для предсказания оттока клиентов используются реальные исходные данные, предоставленные одним из крупнейших российских операторов мобильной связи. Особенностью данных является их относительная сложность (наличие одновременно и персональной информации, и данных об активности, трафике), в связи с чем, возможно несколько вариантов обработки исходных данных (выделения векторов атрибутов), которые требуют изучения и сравнения.

Постановка задачи

Целью работы является построение классификатора для предсказания оттока пользователей телекоммуникационной компании.

В связи с необходимостью обучать и сравнивать множество классификаторов (из-за существования различных алгоритмов построения классификаторов и их входных параметров, вариативности обработки данных) стоит отметить необходимость системы для обучения и сравнения классификаторов.

Таким образом, можно выделить следующие задачи:

- Провести предварительную обработку и анализ исходных данных
- Реализовать систему для дальнейшей обработки данных и тестирования классификаторов (или использовать уже существующую)
- Сравнить результаты различных алгоритмов построения классификаторов

1. Обзор

1.1. Похожие задачи

Стоит отметить, что задача предсказания оттока клиентов в телекоммуникационной сфере не является новой, её решению посвящено достаточно большое количество научных работ. Первая датируется 1999 годом[2], но особенно много работ по данной теме приходится на последние годы (после 2010)[3].

Например, в 2012 году опубликовано исследование задачи предсказания оттока клиентов ирландской телекоммуникационной компании[4]. Для построения классификаторов использовались данные о 827124 реальных абонентах, 27124 из которых – уходящие. При сравнении различных алгоритмов машинного обучения лучшие результаты показали деревья решений и метод опорных векторов (метрики AUC от 0.85 до 0.9).

Также хочется отметить использование различных алгоритмов классификации и методов обработки данных для решения похожих задач, что связано, прежде всего, со спецификой каждой конкретной задачи и особенностями имеющихся данных[3].

1.2. Задача классификации

Решаемая в данной работе задача классификации более формально ставится так[6]: имеется множество объектов, разделённых некоторым образом на классы. Каждый объект задаётся в виде числового вектора (вектора атрибутов) фиксированной длинны. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется обучающей выборкой. Классовая принадлежность остальных объектов не известна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества.

Более того, нас интересует задача бинарной классификации – задача, в которой объекты могут принадлежать одному из двух классов. В

данном случае:

- класс 0 – пользователи, не собирающиеся в ближайшее время отказаться от предоставляемых услуг (неуходящие или остающиеся пользователи)

- класс 1 – пользователи, собирающиеся в ближайшее время отказаться от предоставляемых услуг (уходящие пользователи)

Существует множество различных методов машинного обучения для построения классификаторов (алгоритмов классификации)[6].

1.3. Деревья решений

Одними из самых популярных классификаторов являются деревья решений[12]. В листьях такого дерева записаны значения целевой функции (или метки одного из классов), в остальных вершинах – условия, определяющие по какому ребру идти. Чтобы классифицировать новый объект, надо спуститься по дереву до листа и выдать соответствующее значение.

Данный метод имеет целый ряд преимуществ: быстрое обучение, способность работать с категориальными атрибутами, простота понимания и интерпретации, отсутствие необходимости масштабирования данных (нормализация или стандартизация).

Благодаря этим преимуществам метод хорошо подходит для сложных данных, подобных имеющимся у нас (наличие одновременно и персональной информации, и данных об активности, трафике). Поэтому деревья решений в чистом виде или методы, основанные на них, являются самыми популярными методами для работы с телекоммуникационными данными[3].

Из недостатков метода можно выделить склонность метода к переобучению и относительно низкую точность предсказания. Бороться с этими проблемами можно с помощью построения композиции деревьев решений (так называемый бустинг).

1.4. Бустинг

Бустинг – это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов[11]. Именно на алгоритмы бустинга на основе деревьев решений делается акцент в данной работе. Существуют несколько таких алгоритмов, самые популярные: Случайный лес[18], Ada boost[10], Gradient boosting[15].

Именно эти алгоритмы выбраны для использования (и сравнения) в данной работе.

1.5. Метод опорных векторов

Помимо деревьев решений и методов, основанных на них, существуют и другие алгоритмы классификации. Одним из самых популярных является метод опорных векторов[19], реализующий идею построения разделяющей гиперплоскости с максимальным зазором, предложенную В. Н. Вапником (рис. 1.5).

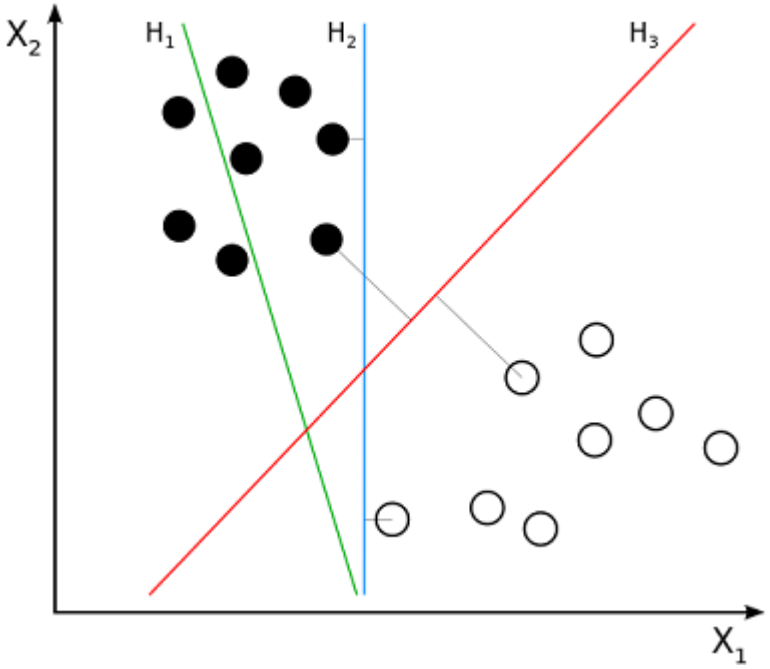

Рис. 1: Разделяющие гиперплоскости

1.6. Метрики

Поскольку данная работа предполагает сравнение различных классификаторов, необходимо ввести метрики, по которым будет вестись сравнение.

Каждый объект выборки принадлежит одному из двух классов. Бинарный классификатор присваивает объекту метку одного из этих классов. Таким образом все объекты выборки делятся на четыре группы. Размеры этих групп характеризуют работу классификатора на данной выборке. Группы для данной задачи:

- TP – *true positives* – правильно предсказанные уходящие абоненты
- TN – *true negatives* – правильно предсказанные остающиеся абоненты
- FP – $false$ *positives* – остающиеся абоненты, предсказанные уходящими
- FN – $false$ *negatives* – уходящие абоненты, предсказанные остающимися

На основе размеров этих групп вводятся метрики $precision$ и $recall$[20]:

$$precision = \frac{TP}{TP + FP} \tag{1}$$

$$recall = \frac{TP}{TP + FN} \tag{2}$$

Эти метрики многое говорят о работе классификатора, но для сравнения требуется единственное число, характеризующее эту работу. В качестве такой метрики может выступать $fscore$[13] – по сути, взвешенное среднее $precision$ и $recall$:

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{3}$$

Значение $fscore$ также лежит в отрезке $[0; 1]$, коэффициент β определяет предпочтение между $precision$ ($\beta < 1$) или $recall$ ($\beta > 1$).

Метрика $fscore$ проста в интерпретации, но её недостатком является то, что её значение характеризует работу классификатора при фиксированном значении порога решающего правила. Существуют метрики, лишенные данного недостатка, например, AUC[7] – площадь под графиком кривой ошибок. Кривая ошибок – это кривая, показывающая значения $recall$ и $fall\text{-}out$ при варьировании порога решающего правила, где $fall\text{-}out$ определяется следующим образом:

$$fall\text{-}out = \frac{FP}{FP + TN} \tag{4}$$

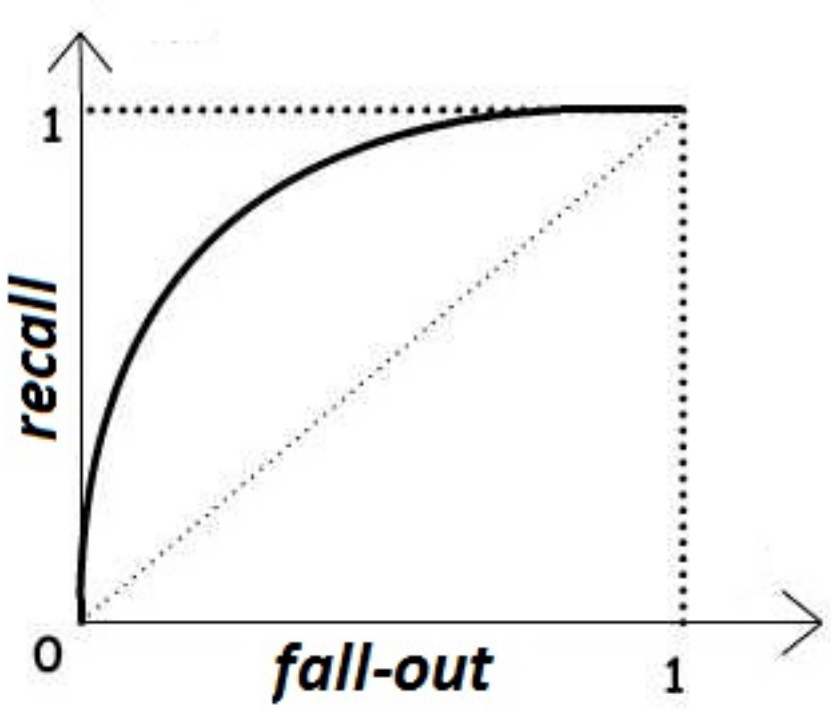

Рис. 2: Кривая ошибок

AUC также принимает значения в отрезке $[0; 1]$, чем выше значение, тем лучше. Связь между AUC и $fscore$ существует, но она не тривиальна[9]. Метрика AUC также является стандартной метрикой сравнения бинарных классификаторов[21].

2. Данные

Оператором мобильной связи были предоставлены данные на 411802 абонентов, содержащие пол, возраст, регион подключения, дату регистрации и данные по активности абонента за пятнадцать календарных месяцев. Данные по активности включают в себя количество отправленных СМС сообщений, количество мегабайт входящего трафика, агрегированную длительность исходящих и входящих вызовов за каждый месяц. При этом исходящие вызовы были разбиты на следующие шесть категорий:

1. Исходящие на городские номера в пределах области подключения
2. Исходящие на мобильные номера оператора в пределах области подключения
3. Исходящие на мобильные номера прочих операторов в пределах области подключения
4. Исходящие на мобильные номера оператора за пределы области подключения
5. Исходящие на городские номера или номера прочих операторов за пределы области подключения
6. Международные исходящие вызовы

Таким образом данные по активности абонента можно представлять в виде девяти временных рядов, по пятнадцать значений в каждом.

3. Обработка данных

3.1. Анализ активности абонента

Для произвольного абонента и произвольного месяца требуется по общей информации об абоненте и данных о его активности за несколько предыдущих месяцев научиться предсказывать собирается абонент прекратить пользоваться услугами оператора в этом месяце (уходящий абонент) или нет (остающийся абонент).

Так как в работе для построения классификатора предполагается использование алгоритмов машинного обучения с учителем, одним из необходимых этапов обработки данных является разделение абонентов на два класса – уходящие и остающиеся абоненты.

Уходящими считаем абонентов, не проявивших никакой активности в течение месяца. Последней месяц, в который абонент проявлял активность назовём месяцем ухода, предсказание интересно делать на месяц, следующий за ним. Всех прочих абонентов считаем остающимися, выбор месяца на который делаем предсказание несущественен. Соответственно, из активности за несколько месяцев, предшествующих интересующему, нужно выделить вектор атрибутов.

При выполнении данного этапа обработки данных исключаются абоненты с недостаточным количеством месяцев активности подряд (слишком быстро ушедшие или зарегистрировавшиеся в один из последних месяцев).

Особенно наглядно видно является ли абонент уходящим при использовании маски активности, где единицам соответствуют месяцы, в которые абонент проявлял активность (рис. 3.1).

Такой подход направлен на максимизацию количества ушедших абонентов в итоговой выборке.

3.2. Строковые атрибуты

Также для использования алгоритмов машинного обучения строковые атрибуты требуется преобразовать в числовые. Пол и регион

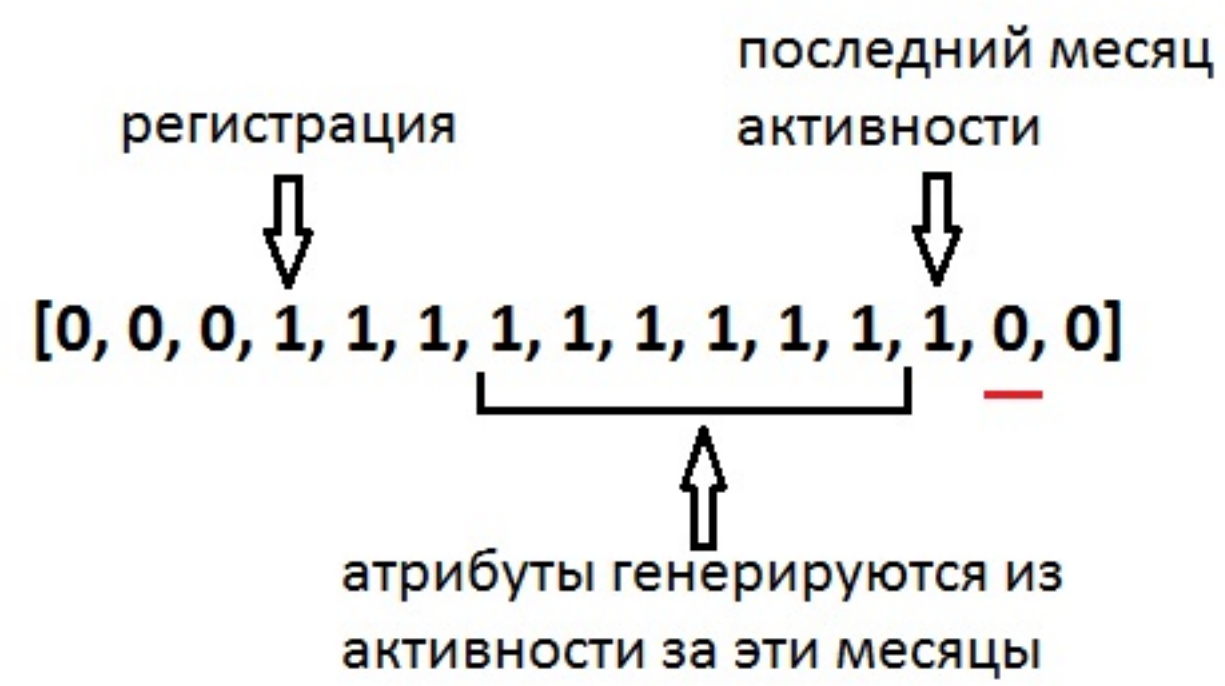

Рис. 3: Пример маски активности

подключения являются строковыми категориальными атрибутами, поэтому каждый из них при обработке изначально преобразовывается в натуральное число (номер своей категории), а затем может преобразовываться в набор бинарных признаков, что важно, например, для метода опорных векторов[14]. Дата регистрации преобразовывается в количество дней, прошедших с регистрации до месяца, на который мы пытаемся делать предсказание.

3.3. Новые атрибуты

Помимо использования уже существующих значений в качестве атрибутов, можно генерировать новые атрибуты на основе уже существующих атрибутов или какой-либо внешней информации. В данной задаче, так как часть данных представлена в виде временных рядов, можно попробовать использовать следующие атрибуты:

- Минимум
- Максимум

- Математическое ожидание
- Дисперсия
- Коэффициент эксцесса
- Коэффициент асимметрии

3.4. Масштабирование

Также стоит отметить необходимость масштабирования данных при использовании метода опорных векторов. Обычно используется стандартизация – придание значениям каждого атрибута свойств стандартного нормального распределения: нулевое математическое ожидание, единичное среднеквадратичное отклонение[14]. Новые значения считаются по следующей формуле:

$$x_{new} = \frac{x - \mu}{\sigma} \tag{5}$$

Где μ – математическое ожидание, σ – среднеквадратичное отклонение.

Также может использоваться нормализация:

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{6}$$

Где x_{min} и x_{max} – минимальное и максимальное значение атрибута соответственно.

3.5. Разделение выборки

Так как в работе предполагается сравнение алгоритмов машинного обучения с учителем, необходимо разделить исходную выборку на три части: тренировочную, валидационную и тестовую. Тренировочная выборка используются для обучения классификаторов, результаты на валидационной выборке служат для сравнения различных классификаторов (и выбора лучшего), итоговый результат – результат лучше-

го классификатора на тестовой выборке. Подобное разделение выборки применяется для минимизации переобученности финального классификатора. Размер тренировочной выборки выбран равным 60% исходной, размеры валидационной и тестовой выборки – 20% каждая.

3.6. Вариативность обработки данных

Отдельно хочется отметить большую вариативность обработки данных. Необходимо выбрать количество месяцев, активность за которые используется при генерации атрибутов (например, 3, 6 или 9). Необходимо выбрать набор новых атрибутов для временных рядов. Необходимо выбрать один из вариантов масштабирования (нормализация, стандартизация, без масштабирования).

4. Система тестирования

4.1. Требования

Были выбраны несколько алгоритмов построения классификаторов, для каждого из этих алгоритмов существуют несколько входных параметров, существенно влияющих на точность классификации, и, подбирающихся, по сути, перебором[17]. Также, как уже отмечалось, существуют несколько возможных вариантов обработки исходных данных, причём, некоторые специфичны для конкретных алгоритмов.

Таким образом, получаем большое общее количество классификаторов, которые необходимо обучить и протестировать. При времени обучения одного классификатора от 10 до 40 минут, общее время, требуемое для обучения, измеряется сутками.

Становится очевидна необходимость системы для обработки данных, обучения и тестирования классификаторов. Можно выделить следующие требования такой системы:

1. Возможность перебора параметров обработки данных
2. Возможность перебора алгоритмов построения классификаторов и их входных параметров
3. Возможность остановки и возобновления работы, просмотр промежуточных результатов
4. Распараллеливание

4.2. Существующие решения

Во многих библиотеках машинного обучения для перебора входных параметров алгоритмов реализованы методы поиска в сетке[16][5]. Подобные методы не поддерживают возможность перебора различных вариантов обработки исходных данных и просмотр промежуточных результатов, не удовлетворяя, таким образом, представленным требованиям.

Помимо поиска в сетке существуют также методы для подбора алгоритмов машинного обучения и их входных параметров, предоставляющие более широкие возможности, например, Auto-WEKA[1]. Но, к сожалению, возможность различной обработки исходных данных также не предусматривается.

Поэтому, было принято решение самостоятельно реализовать систему для тестирования различных классификаторов, удовлетворяющую необходимым требованиям.

4.3. Схема работы

Выбор языка программирования, по сути, определяется выбором библиотеки машинного обучения. В данной работе в качестве такой библиотеки используется scikit-learn[22], основные критерии выбора – популярность и хорошая документация[23]. Соответственно, язык программирования для системы тестирования – Python[8].

Общую схему работы системы можно представить в виде диаграммы (рис. 4.3).

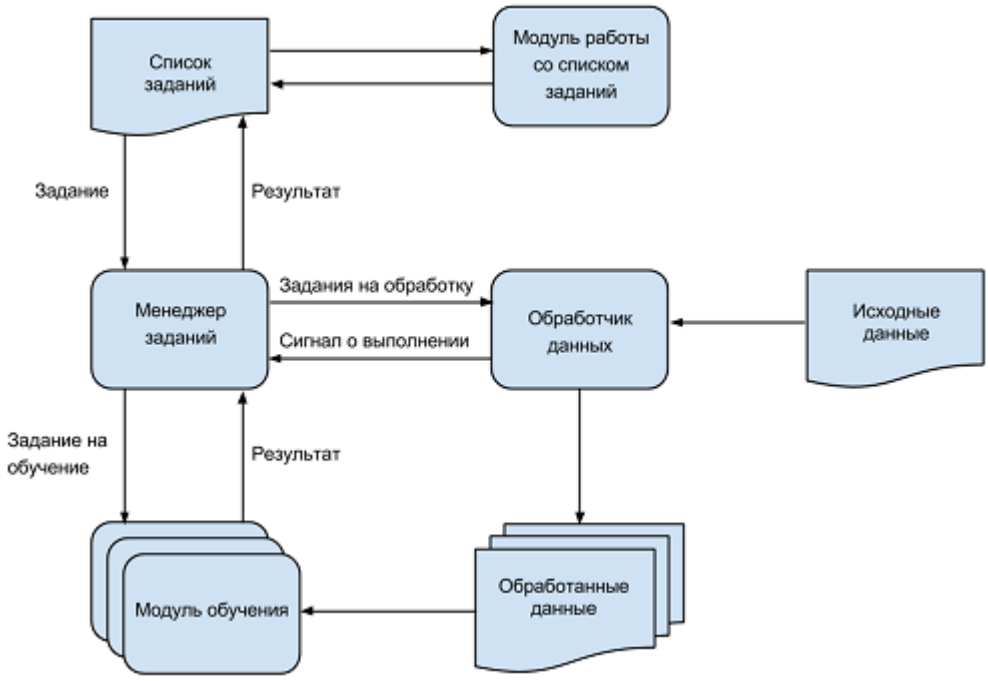

Рис. 4: Схема работы системы

Заданием для системы тестирования называется набор, состоящий из входных параметров для обработки исходных данных, алгоритма построения классификатора, его входных параметров и результатов классификатора (результаты заполняются после выполнения задания и представляют собой значения интересующих метрик).

Задания для системы формируют список заданий, сериализованный список хранится в виде файла на жестком диске.

Модуль работы со списком заданий служит для редактирования списка заданий и просмотра результатов (для просмотра результатов не требуется выполнение всех заданий – возможен просмотр промежуточных результатов)

При запуске системы для выполнения заданий управление передается менеджеру, менеджер считывает список заданий, формирует список требуемых вариантов обработки исходных данных и передает этот список обработчику данных.

Для каждого элемента списка обработчик считывает необработанные исходные данные, выполняет требуемую обработку, сериализует полученные данные и сохраняет их по адресу, полученному из параметров для обработки.

После завершения обработки менеджер формирует очередь заданий на обучение, состоящую из еще не выполненных заданий системы тестирования, создает пока что пустую очередь выполненных заданий, и запускает параллельно несколько модулей обучения.

Каждый модуль обучения в ходе своей работы повторяет следующую последовательность действий: берёт из очереди задание на обучение, считывает указанный в задании вариант обработанных исходных данных, выполняет обучение указанного в задании классификатора (с указанными входными параметрами) на обучающей части выборки, получает результаты работы полученного классификатора на обучающей и валидационной частях выборки, обновляет задание полученным результатом, записывает выполненное задание в соответствующую очередь.

Извлечение выполненных заданий из этой очереди и обновление

списка заданий полученными результатами также осуществляет менеджер.

5. Сравнение классификаторов

5.1. Метрики

Для сравнения классификаторов приоритетной выбрана метрика AUC. Также, в силу специфики задачи, между $precision$ и $recall$ предпочтение отдаётся $precision$: мы хотим минимизировать количество неуходящих абонентов, классифицированных неправильно. Поэтому используется $fscore$ со значением $\beta = 0.5$

5.2. Параметры обработки данных

Для каждого использованного алгоритма построения классификаторов, в соответствии с приоритетной метрикой на валидационной части выборки, был выбран классификатор со следующими параметрами обработки исходных данных:

	количество месяцев	персональные данные	новые атрибуты	масштабирование
Gradient boosting	3	использовать	добавить	нет
Ada boost	3	использовать	добавить	нет
Случайный лес	3	использовать	добавить	нет
Метод опорных векторов	3	использовать	заменить временные ряды	нормализация

Исходя из этого можно сделать несколько интересных наблюдений:

- Использование персональных данных абонента (пол, возраст, дата регистрации и т.д.) улучшает результат
- Использование новых атрибутов, полученных из временных рядов (минимум, максимум, математическое ожидание и т.д.) улучшает результат

- Использование более чем трех месяцев, активность за которые используется для генерации атрибутов, не улучшает результат

5.3. Сравнение алгоритмов

Были получены результаты работы этих классификаторов на тестовой части выборки:

	AUC	$precision$	$recall$	$F_{0.5}$
Gradient boosting	0.88	0.72	0.65	0.70
Ada boost	0.86	0.68	0.60	0.66
Случайный лес	0.85	0.62	0.72	0.64
Метод опорных векторов	0.80	0.64	0.57	0.62

Метод опорных векторов показал худший результат. Случайный лес и Ada boost показали более высокие, сравнимые между собой результаты. Лучший же результат продемонстрировал Gradient boosting.

Заключение

В рамках данной работы рассмотрены различные методы предсказания оттока пользователей в телекоммуникационной сфере, произведено их сравнение. В частности, были выполнены следующие задачи:

- Проведены предварительная обработка и анализ данных
- Реализована система для дальнейшей обработки данных и тестирования классификаторов
- Проведено сравнение результатов различных алгоритмов построения классификаторов

Список литературы

[1] Auto-WEKA. — URL: http://www.cs.ubc.ca/labs/beta/Projects/autoweka/ (дата обращения: 22.05.2015).

[2] Churn Reduction in the Wireless Industry. — URL: http://www.cs.colorado.edu/~mozer/Research/Selected%20Publications/reprints/churn_nips.pdf (дата обращения: 22.05.2015).

[3] Customer Churn Prediction in Telecommunication A Decade Review and Classification. — URL: http://www.researchgate.net/profile/Nabgha_Hashmi/publication/257920014_Customer_Churn_Prediction_in_Telecommunication_A_Decade_Review_and_Classification/links/00b495261475ba6758000000.pdf (дата обращения: 22.05.2015).

[4] Customer churn prediction in telecommunications. — URL: http://www.sciencedirect.com/science/article/pii/S0957417411011353 (дата обращения: 22.05.2015).

[5] Grid Search. — URL: http://scikit-learn.org/stable/modules/grid_search.html (дата обращения: 22.05.2015).

[6] MachineLearning.ru. Классификация. — URL: http://www.machinelearning.ru/wiki/index.php?title=%D0%9A%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F (дата обращения: 22.05.2015).

[7] MachineLearning.ru. Кривая ошибок. — URL: http://www.machinelearning.ru/wiki/index.php?title=ROC-%D0%BA%D1%80%D0%B8%D0%B2%D0%B0%D1%8F (дата обращения: 22.05.2015).

[8] Python. — URL: https://www.python.org/ (дата обращения: 22.05.2015).

[9] The Relationship Between Precision-Recall and ROC Curves. —

URL: https://www.biostat.wisc.edu/~page/rocpr.pdf (дата обращения: 22.05.2015).

[10] Wikipedia. Ada boost // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/AdaBoost (дата обращения: 22.05.2015).

[11] Wikipedia. Boosting // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Boosting_%28machine_learning%29 (дата обращения: 22.05.2015).

[12] Wikipedia. Decision tree // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Decision_tree_learning (дата обращения: 22.05.2015).

[13] Wikipedia. F score // Википедия, свободная энциклопедия. — URL: http://en.wikipedia.org/wiki/F1_score (дата обращения: 22.05.2015).

[14] Wikipedia. Feature scaling // Википедия, свободная энциклопедия. — URL: http://en.wikipedia.org/wiki/Feature_scaling (дата обращения: 22.05.2015).

[15] Wikipedia. Gradient boosting // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Gradient_boosting (дата обращения: 22.05.2015).

[16] Wikipedia. Grid search // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Hyperparameter_optimization#Grid_search (дата обращения: 22.05.2015).

[17] Wikipedia. Hyperparameter optimization // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Hyperparameter_optimization (дата обращения: 22.05.2015).

[18] Wikipedia. Random forest // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Random_forest (дата обращения: 22.05.2015).

[19] Wikipedia. SVM // Википедия, свободная энциклопедия. — URL: https://en.wikipedia.org/wiki/Support_vector_machine (дата обращения: 22.05.2015).

[20] Wikipedia. Sensitivity and specificity // Википедия, свободная энциклопедия. — URL: http://en.wikipedia.org/wiki/Sensitivity_and_specificity (дата обращения: 22.05.2015).

[21] An introduction to ROC analysis. — URL: http://www.sciencedirect.com/science/article/pii/S016786550500303X (дата обращения: 22.05.2015).

[22] scikit-learn. — URL: http://scikit-learn.org/stable/ (дата обращения: 22.05.2015).

[23] scikit-learn documentation. — URL: http://scikit-learn.org/stable/user_guide.html (дата обращения: 22.05.2015).