Федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет бизнеса и менеджмента

КУРСОВАЯ РАБОТА

Предиктивный анализ потока абитуриентов на образовательные программы НИУ ВШЭ

по направлению подготовки Бизнес-информатика

образовательная программа «Бизнес-информатика»

Выполнил:

Студент группы 185

Поликарпов Кирилл Николаевич

Ф.И.О.

Руководитель:

доцент, Ефремов Сергей Геннадьевич

степень, звание, должность Ф.И.О.

Москва 2020

Содержание

Введение

Последние несколько лет бурно развивается область интеллектуального анализа данных (Data Science), а ее инструменты и методы нашли свое применение во многих областях жизни и науки. В данной работе задаюсь вопросом о том, как можно использовать накопленный объем данных университетом для того, чтобы решать задачи, которые стоят перед ним, эффективно распределяя человеческие и финансовые ресурсы в существующих процессах.

Национальный исследовательский университет «Высшая школа экономики» является одним из популярных университетов России, поступать в который каждый год приходит большое количество абитуриентов со всей страны. Приемная комиссия, принимая заявки на поступление в течение нескольких месяцев, накапливает достаточно большую базу данных, в которой хранится информация о каждом поступающем. **Актуальность исследования** заключается в том, что, обладая таким источником информации, университет может извлекать из него скрытую информацию на пользу себе с помощью инструментов интеллектуального анализа данных.

Целью курсовой работы стоит выяснить, какие факторы больше всего влияют на решение абитуриента о поступлении на ту или иную программу. Это позволит университету более грамотно распределять бюджет на социальную помощь и стипендии для максимизации потока абитуриентов. Кроме этого, это также может давать администрации университета возможность регулировать число и качество поступивших студентов. Решение проблемы выявления мотивации поступающих сопровождается одновременно и с решением другой задачи, а именно, прогнозированием вероятностей поступления каждого из абитуриентов. А это уже в свою очередь позволяет оценить общее число поступающих в университет заранее. **Основными задачами**, поставленными для достижения цели можно считать:

- изучить и проанализировать актуальную информацию о предиктивном анализе поступления абитуриентов в университет;

- проанализировать основные понятия и модели машинного обучения;

- собрать данные, на основе которых будет построена модель;

- отобрать модель, показывающую лучшую метрику качества на тестовых данных;

- разработать практические рекомендации по возможному улучшению модели.

Объектом исследования являются данные абитуриентов при подаче документов в университет. Разработка моделей строилась на основе реальных данных абитуриентов, поступающих в Высшую школу экономики.

Предметом исследования предсказание поступления абитуриента в университет на основе ряда факторов.

После анализа похожих работ по данной проблеме можно сказать, что она изучена и раскрыта достаточно слабо из-за малого количества исследований, где большая доля приходится на научные работы зарубежных авторов.

Результаты данной работы получены с помощью алгоритмов машинного обучения и техник извлечения данных. Рассмотрены следующие алгоритмы машинного обучения: метод k-ближайших соседей (kNN), логистическая регрессия (logistic regression), метод опорных векторов (SVM), решающее дерево (decision tree) и случайный лес (random forest).

1. Анализ существующих работ по данной теме

1.1. Статья «Predicting Student Enrollment Based on Student and College Characteristics»

Ahmad Slim, Don Hush, Tushar Ojah и Terry Babbitt поставили перед собой цель понять, какой абитуриент больше всего склонен к поступлению в университет по ряду признаков, а какой нет, какие факторы больше всего способны повлиять на его решение.

Для проведения этого анализа ими были взяты настоящие данные абитуриентов, которые поступили в University of New Mexico (UNM) в разные года в разные сезоны.

Перед тем, как использовать данные для предсказания, авторы занялись их предобработкой.

Признаки в данных являются теми факторами, на основе которых алгоритмы моделей определяют то, поступит абитуриент или нет. Авторы выделили порядка 60 таких факторов, одна доля которых характеризует университет, другая – абитуриентов. Примеры соответствующих признаков:

- GENDER: бинарная перемеренная, дающая информацию про пол абитуриента;
- ETHNICITY: категориальная переменная, говорящая о том, к какой этнической группе относится абитуриент;
- SCORE: дискретная количественная переменная, характеризующая уровень знаний абитуриента;
- FIRST_GENERATION: бинарная переменная, которая говорит о том, что по крайней мере один родитель закончил университет или колледж;
- PARENT_INCOME: непрерывная количественная переменная, характеризующая уровень дохода родителей;

- STUDENT_INCOME: непрерывная количественная переменная, характеризующая уровень дохода абитуриента в случае наличия работы;
- INSTITUTIONAL_MONEY: непрерывная количественная переменная, которая показывает размер финансовой помощи абитуриенту со стороны университета;
- FEDERAL_MONEY: непрерывная количественная переменная, которая показывает размер финансовой помощи со стороны государства;

По их мнению, лучшей метрикой качества для оценки модели является построение матрицы ошибок (confusion matrix), рассматривая такие метрики качества как точность (precision) и полноту (recall).

Рассматривая проблему классификации поступающих, авторы выделяют два подхода: классификация на индивидуальном уровне (individual level) и на уровне групп (cohort level). Индивидуальный уровень подразумевает под собой то, что модель на основе значений признаков для каждого абитуриента делает предсказание. Затем, суммируя тех абитуриентов, которых модель посчитала, что поступят в университет, получаем общее число поступивших. Классификация на уровне же групп говорит, что абитуриенты разделены на группы по какому-то признаку (этот признак может включать себя множество отдельных изначальных признаков). На основе этого признака идет предсказание того, будет ли зачислена данная группа или нет. Найдя те группы абитуриентов, которых модель определила как те, что будут зачислены, будет найдено общее количество зачисленных, зная количество абитуриентов в каждой из групп.

В подходе классификации на индивидуальном уровне используются алгоритмы логистической регрессии (logistic regression) и опорных векторов (support vector machine).

На основе модели логистической регрессии авторы проводят отбор признаков с помощью алгоритма forward selection. Таким образом получают следующие наиболее коррелирующие признаки:

- STATE_AWARD_ORIGINAL: непрерывная количественная переменная, характеризующая размер стипендии, которая дается поступающим из штата New Mexico;
- FIRST_DECISION_DIFF: бинарная переменная, которая говорит о том, если заявка о зачислении была рассмотрена после февраля (март, апрель, июнь и июль), то индекс равен 1, иначе 0;
- SUCCESS: бинарная переменная, характеризующая денежную выплату, которую получают абитуриенты, которым нужна финансовая помощь, в первом семестре, где получают 1, иначе 0;
- GPA: непрерывная количественная переменная, дающая представление о GPA (0-5) по окончании школы;
- RESIDENCY_STATE: категориальная переменная, говорящая о том, откуда студента: 0 – in-state (из New Mexico), 1- проживает в штатах Texas, California, Arizona и Colorado, 2 – non-resident, 3 – international;
- FAFSA_BDEADLINE: бинарная переменная, характеризующая то, подал ли поступающий заявку на FAFSA до дедлайна – 1, иначе 0;
- LOW_INCOME: бинарная переменная, характеризует социально-экономический статус родителей, если доход низкий – 1, иначе 0;
- BRIDGE: бинарная переменная, характеризующая денежную выплату, которую получают первокурсники в первом семестре в индивидуальном порядку в зависимости от уровня знаний, где 1 получает, иначе 0;
- APP_AFEB: бинарная переменная, говорящая о том, когда абитуриент подал заявку. Если заявка была отправлена после февраля, то признак принимает значение 1, иначе 0;

- FED_AWARD_ORIGINAL: непрерывная количественная переменная, характеризующая размер денежной выплаты со стороны государства поступающим из федеральных штатов;

В подходе классификации на уровне групп используется вероятностный подход и анализ временных рядов. В вероятностном подходе авторами была выбрана модель semi-supervised learning, а для анализа временных рядов – модель ARIMA.

1.2. Статья «Improving Student Enrollment Prediction Using Ensemble Classifiers»

Stephen Kahara, Geoffrey Muchiri ставят перед собой вопрос о том, как привлечь больше абитуриентов на технические специальности такие как science, technology, engineering и mathematics (STEM), и выяснить, какие факторы влияют на решение о поступлении, каким абитуриентам нужна финансовая помощь с помощью техник и инструментов educational data mining (EDM).

В своей работе авторы используют модель композиции классификаций (ensemble classification), обосновывая это тем, что данная модель, комбинируя в себе разные модели, дает лучший результат метрики качества, чем наилучший результат включенных моделей классификаций по-отдельности. Авторы пишут, что такое преимущество данной модели достигается за счет того, что модель, объединяя в себе разные модели позволяет компенсировать трудности и ошибки одной входящей в нее модели с помощью другой.

Для построения модели они используют методологию Cross Industry Standard Process for Data Mining (CRISP-DM), в которой выделяются следующие этапы.

Авторы статьи не имели готовых данных для проведения анализа и предсказания на его основе. Для решения этой проблемы данные были собраны с помощью анкетирования определенных студентов в University of Technology, Kenya за академический год 2016-2017. Целевую аудиторию разделили на две группы: STEM и non-STEM. В самом опросе были выделены следующие атрибуты со следующими возможными ответами:

- Career Flexibility: {Yes, No};
- High School Final Grade: {A, A-, B+, B, B-, C+};
- Math Grade: {A, A-, B+, B, B-, C+};
- Pre-University Awareness: {Yes, No};
- Teacher Inspiration: {Yes, No};
- Financial Aid: {Yes, No};
- Extracurricular: {Yes, No};
- Societal Expectation: {Yes, No};
- Parent Career: {STEM, non-STEM};
- Self-Efficacy: {Yes, No};
- Career Earning: {Yes, No};
- Gender: {Male, Female};
- Age: {Below 20 Years, 20-25 Years, 26-30 Years, 31 and above Years};
- Family Income: {Less than 10000, 10001-20000, 20001-30000, 30001-40000, 40001-50000, 50001 and above}.

В исследовании авторы используют такие алгоритмы как J48 algorithm, naive Bayes algorithm, CART и bagging в программном обеспечении WEKA. Первые три алгоритма являются отдельно взятыми классификаторами в то время, как Bagging является ансамблем. Несмотря на то, что наилучшую метрику качества показал алгоритм J48, оценка качества первых трех алгоритмов сильно скачает из-за их восприимчивости к шуму в данных и склонности к переобучению, а алгоритм bagging нивелирует эти эффекты в связи с тем, что использует кросс-валидацию и выдает результат на основе объединения

нескольких алгоритмов классификации. Поэтому авторами был выбран этот алгоритм.

В заключении работы приходят к выводу, что наиболее коррелирующими факторами по коэффициенту корреляции Пирсона при принятии решения абитуриентом о поступлении являются High Scholl Grade (0.981), Career Flexibility (0.842), Math Grade (0.763), Self-Efficacy (0.714) и Teacher Inspiration (0.692).

2. Теоретическая часть

2.1 Формальная постановка задачи машинного обучения

Формализуем то, как будет устроено описание моделей и алгоритмов. Машинное обучение – это наука о том, как восстановить функцию зависимости по точкам. Представим это отображением множества объектов во множество ответов: $X \to Y$. Задача машинного обучения заключается в том, чтобы найти такой аппроксимирующий алгоритм:

$$a: X \to Y,$$

который бы был наиболее всего приближен ко множеству ответов Y.

В общем виде записать пространство всех возможных объектов (samples) и пространство всех возможных ответов (targets) можно следующим образом:

$$X = \{x_1, \dots, x_l\} \text{ и } Y = \{y_1, \dots, y_l\}, i = 1, \dots, l,$$

а представление выборки, которая состоит из объектов и ответов: $(x_i, y_i)_{i=1}^{l}$.

2.1.1. Множество признаков

Каждый объект описывается множеством признаков (features). То, как описывается множество объектов во множестве признаков, можно представить следующим образом:

$f_j: X \to D_j, j = 1, \dots, d$, где

$x = (x_1, \dots, x_d)$ – признаковое описание объекта x.

Обычно запись представления множества признаков и множества объектов представляют в виде матрицы «объекты – признаки»:

$$\begin{pmatrix} f_1(x_1) & \cdots & f_d(x_1) \\ \vdots & \ddots & \vdots \\ f_1(x_l) & \cdots & f_d(x_l) \end{pmatrix}$$

Существует достаточно большое количество видов признаков. Как правило, выделяют следующие:

1. Бинарные признаки: $D_j = \{0, 1\}$;
2. Вещественные (количественные) признаки: $D_j = R$;
3. Номинальные (категориальные) признаки: $D_j -$ неупорядоченное множество;
4. Порядковые признаки: $D_j -$ упорядоченное множество.

2.1.2. Множество ответов

Модели классификации отличает от других то, что множество ответов – это конечное число категориальных ответов. Для моделей классификации выделяют следующие виды ответов (где k количество классов):

1. Бинарная классификация: $Y = \{-1, +1\}$;
2. Многоклассовая классификация: $Y = \{1, \dots, k\}$;
3. Классификация с пересекающимися классами: $Y = \{0, 1\}^k$.

2.1.3. Функция потерь, функционал качества, модель

На самом деле задача обучения моделей с помощью алгоритмов сводится к задаче оптимизации, которая приводит к тому, что обучение проходит наиболее эффективным способом. Из этого приходим к тому, что во время обучения на обучающей выборке нам нужно выбрать такой алгоритм, который бы давал наиболее точные ответы. Измерить точность ответа алгоритма на одном отдельном объекте позволяет функция потерь. Функция потерь – величина ошибки алгоритма $a \in A$ на объекте $x \in X$ – $L(a, x)$. Однако оптимизировать ошибку нужно не для каждого отдельного объекта, а для всех объектов обучающей выборки – процесс оптимизации функционала качества (эмпирического риска). Именно минимизировать меру этой величины и стоит задача, так как чем меньше ошибка, тем лучше наша модель обучилась на обучающих выборке. Формальная запись выглядит следующим образом:

$$Q(a, X^l) = \arg\min_{a \in A} \frac{1}{l} \sum_{i=1}^{l} L(a, x_i).$$

Модель – это семейство параметрических функций алгоритмов:

$$A = \{a(x) = g(x, w) | w \in W\},$$

где w – искомый параметр нашей модели среди всего множества допустимых значений параметра W.

Наиболее популярными алгоритмами оптимизации при обучении являются градиентный и стохастический градиентный спуски, которые позволяют эффективно подбирать параметры модели, достигая минимума функционала качества при обучении модели.

2.2. Методология ведения проектов по машинному обучению

Ведение проекта по анализу данных достаточно трудоемкий процесс, контролировать и вести который достаточно сложно. Наиболее популярным для этого фреймворком является CRISP-DM. В своей книге Robert Nisben, John Elder и Gary Miner выделяют следующие этапы: понимание бизнеса (business understanding), понимание данных (data understanding), подготовка данных (data preparation), моделирование (modeling), оценка (evaluation) и развертывание (deployment).

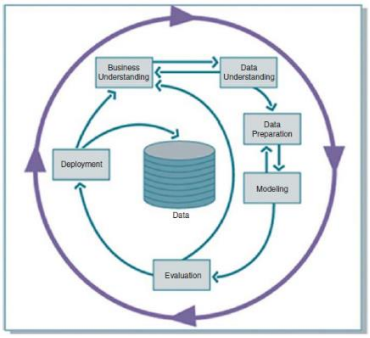


Рисунок 2.0.1 – Фазы процесса CRISP-DM

2.2.1. Понимание бизнеса

Перед тем, как приступить к работе с данными необходимо четко понять, какая стоит задача. Важно правильно услышать и разобрать желания заказчика, формализуя полученные бизнес-цели и составляя предварительный план по их достижению. Выделив бизнес-цели, необходимо также составить цели анализа данных, которые приведут к достижению бизнес-целей. Поскольку предварительный план и бизнес-цели построены только на понимании желаний заказчика, далее необходимо изучить существующие ограничения, ресурсы и риски, которые необходимо учесть при окончательном формировании целей и плана проекта. Определиться с целями и требованиями проекта также позволит выделение заинтересованных лиц.

2.2.2. Понимание данных

Данная фаза подразумевает под собой сбор данных и предварительное знакомство с ними. Этап начинается с поиска источников данных и выгрузки их в единую структуру. По окончании формирования данных необходимо «поверхностно» описать данные и провести исследовательский (разведочный, exploratory data analysis) анализ данных, в котором рассматриваются распределения, общие закономерности и аномалии в данных с помощью инструментов визуализации и статистики. После разведочного анализа переходим к оценке качества данных на наличие пропусков и их корректность. По выявлению каких-либо ошибок в данных необходимо составить список по тому, какие существуют причины их появления и каким образом можно устранить данные ошибки.

2.2.3. Подготовка данных

Данный этап включает в себя все методы и инструменты, которые позволят получить из сырого набора данных финальный, который уже можно будет использовать при обучении моделей. Подготовка данных является довольно важным этапом, так как то, насколько качественно будут предобработаны данные зависит и то, какие результаты будут получены при построении моделей. Для приведения данных к итоговому виду на этом шаге используются такие процессы как отбор признаков, очистка данных, создание новых признаков, объединение данных и приведение данных в единый формат. При отборе выделяют подходы: фильтрации (filter methods), методы-обертки (wrapper methods) и понижение с помощью моделей (embedded methods). Выбирают тот способ, который приводит к наибольшему качеству модели. Обычно приведение данных в единый формат состоит из двух этапов: кодирование категориальных признаков и масштабирование признаков. Кодирование категориальных признаков позволяет решить проблему того, чтобы на вход в функцию обучения шла матрица с вещественными значениями, а не со строчными значениями при

некоторых признаках, иначе некоторые алгоритмы не смогут обучиться. Наиболее популярными способами кодирования являются one-hot encoding, ordinal encoding и mean-target encoding. Масштабирование данных же позволяет решить проблему масштаба признаков. Для этого используют нормализацию, стандартизацию или minmaxscaler.

2.2.4. Моделирование

На данном этапе к полученным данным применяем различные подходы моделирования, строя алгоритмы моделей и подбирая оптимальные параметры для них. Наиболее популярным подходом, который позволяет подобрать наиболее оптимальные параметры для модели является кросс-валидация (cross-validation). Также при обучении важно учитывать то, что модель может просто переобучиться. Чтобы избежать этого, часто используют регуляризации L1 (Lasso) или L2 (Ridge), которые позволяет снизить риск переобучения. По итогу построения каждой модели необходимо провести оценку ее качества с помощью метрик качества.

2.2.5. Оценка

На этом этапе отбираем лучшую модель с точки зрения метрики качества. В задаче классификации обычно используются такие метрики как доля положительных ответов (accuracy), точность (precision), полнота (recall), F-мера, AUC-PRC и AUC-ROC. Каждая из этих метрик качеств имеет свои преимущества и недостатки и выбирается в зависимости от задачи, которая поставлена. Смотрим на то, удалось ли добиться поставленных бизнес-целей, а также на какие-либо недостатки модели с точки зрения бизнеса, которые ранее не принимались во внимание при разработке. Только после проверки всех целей принимается решение о запуске модели.

2.2.6. Развертывание

Цикл разработки проекта интеллектуального анализа данных заканчивается этапом развертывания. Он подразумевает подготовку финального отчета о проделанной работе и запуск модели уже в реальном секторе бизнеса. Также этот этап может включать в себя мониторинг и поддержку данного проекта со стороны разработчика.

2.3. Метод k-ближайших соседей

Метод k-ближайших соседей (kNN) является самым простым алгоритмом машинного обучения, который строится на идеи гипотезы компактности, говорящей о том, что похожие объекты, как правило, имеют похожие ответы.

2.3.1. Сравнение объектов

Для того, чтобы сравнивать объекты и говорить о их схожести, подсчитывают расстояние между объектами в признаковом пространстве, используя следующие метрики:

1. Евклидово расстояние: $\rho(x,z) = \sqrt{\sum_{j=1}^{d}(x_j - z_j)^2}$;
2. Манхэттенское расстояние: $\rho(x,z) = \sum_{j=1}^{d}|x_j - z_j|$;
3. Расстояние Минковского: $\rho(x,z) = \sqrt[p]{\sum_{j=1}^{d}(x_j - z_j)^p}$.

2.3.2. Обучение kNN

Обучение модели kNN строится на том, что модель запоминает всю обучающую выборку. При поступлении нового объекта x алгоритм сортирует объекты множества X из обучающей выборки по расстоянию до этого объекта:

$$\rho(x, x_1) \leq \cdots \leq \rho(x, x_l).$$

После выбирает k ближайших объектов к нему и относит его к тому классу, объектов которого больше среди его соседей:

$$a(x) = \arg\max_{y \in Y} \sum_{i=1}^{k} [y_i = y].$$

Параметр k является гиперпараметром, который подбирается вручную и от которого во многом зависит то, какая модель в итоге получится.

2.3.3. Взвешенный kNN

После выбора k соседей возникает проблема того, что никак не учитывается то, насколько далеки или близки эти соседи к данному объекту. Для решение данной проблемы дополнительно вводится вес i-ого соседа объекта x:

$$a(x) = \arg\max_{y \in Y} \sum_{i=1}^{k} [y_i = y]\, w(i, x).$$

Наиболее популярны весом в модели kNN является Парзеновское окно, которое описывается следующим образом: $w_i = K(\frac{\rho(x, x_i)}{h})$, где K – ядро, а h – ширина окна.

2.4. Линейная классификация

Модель линейной классификации можно представить следующим образом (есть единичный признак) для случая бинарной классификации:

$$a(x) = sign \sum_{j=1}^{d} w_j x_j = sign\langle w, x\rangle.$$

Линейный классификатор проводит гиперплоскость, разделяя таким образом объекты по двум классам: $\langle w, x\rangle = 0$. Если $\langle w, x\rangle < 0$, то объект

относится к одному классу, а если $\langle w, x\rangle > 0$, то объект относится к другому классу.

Понятие отступа позволяет нам понять, верно ли модель классификации дает ответ или нет: $M_i = y_i\langle w, x_i\rangle$. Если $M_i > 0$, то объект классифицирован верно, если $M_i < 0$ – нет. А абсолютное значение отступа говорит о том, насколько модель уверенна в классификации объекта, так чем дальше объект от разделяющей гиперплоскости, тем больше отступ.

2.4.1. Обучение линейных классификаторов

Для моделей линейной классификации классическим и простым примером функционала ошибки является доля ошибки (error rate):

$$Q(x, X) = \frac{1}{l}\sum_{i=1}^{l}[\langle w, x_i\rangle \neq y_i] = \frac{1}{l}\sum_{i=1}^{l}[y_i\langle w, x_i\rangle < 0].$$

Однако заметим, что в данном случае функционал ошибки это просто индикатор, который нельзя продифференцировать, что затрудняет процесс оптимизации при обучении. Обойти эту проблему позволяет верхняя оценка дифференцируемой функцией:

$$L(M) = [M < 0] \leq \tilde{L}(M)$$

$$0 \leq \frac{1}{l}\sum_{i=1}^{l}[y_i\langle w, x_i\rangle < 0] \leq \frac{1}{l}\sum_{i=1}^{l}\tilde{L}(\langle w, w_i\rangle) \to \min_w.$$

Выделяют следующие примеры верхних оценок для данного случая:

1. $\tilde{L}(M) = \log(1 + e^{-M})$ – логистическая;
2. $\tilde{L}(M) = \max(0, 1 - M)$ – кусочно-линейная;
3. $\tilde{L}(M) = e^{-M}$ – экспоненциальная;
4. $\tilde{L}(M) = \frac{2}{1+e^{M}}$ – сигмоидная.

2.4.2. Логистическая регрессия

Особенностью данного вида модели является возможность оценивать вероятности классов. Функционал ошибки для логистической регрессии выглядит следующим образом:

$$Q(a,X) = \frac{1}{l}\sum_{i=1}^{l} \log\left(1 + \exp(-y_i\langle w, x_i\rangle)\right) \rightarrow \min_w.$$

Идея получения вероятностей идет из того, что у нас есть модель $b(x) = \langle w, x\rangle$, выходы которой переводятся на отрезок $[0,1]$ с помощью сигмоиды:

$$\delta(\langle w, x\rangle) = \frac{1}{1+\exp\left(-\langle w,x\rangle\right)}.$$

Тогда говорим, что модель $b(x)$ предсказывает вероятности, если среди объектов с $b(x) = p$ доля положительных равна p.

2.4.3. Метод опорных векторов

Идея данного класса алгоритмов машинного обучения заключается в том, чтобы максимизировать отступ $\frac{1}{\|w\|}$ от гиперплоскости до ближайшего объекта класса. Модель классификации опорных векторов описывается следующим функционалом ошибки:

$$Q(a,X) = C\sum_{i=1}^{l} \max(0, 1 - y_i\langle w, x_i\rangle + w_0) + \|w\|^2 \rightarrow \min_{w,w_0}.$$

2.5. Решающее дерево

Решающее дерево – это модель машинного обучения, которая делает предсказания на основе покрывающего набора логических правил конъюнкций. Преимущество такого подхода в том, что он позволяется довольно легко

интерпретировать то или иное предсказание моделью, а также дает возможность находить нелинейные закономерности.

Деревья имею бинарный вид, у которых во внутренних вершинах содержаться предикаты $[x_j < t]$, а в листьях – прогнозы модели $c \in Y$.

Модель представлена следующим образом:

$$a(x) = \sum_{j=1}^{J} c_j [x \in R_j].$$

2.5.1. Критерий информативности

При построении решающего дерева алгоритм в каждой его вершине понимает, какое разбиение удачное, а какое нет, благодаря критерию информативности. Обычно используется такие критерии как критерий Джини или критерий энтропии.

Критерий энтропии – это мера неопределенности распределения. Она позволяет понять, насколько вершина хаотична, то есть то, насколько в ней перемешаны классы. Вычисляется по следующей формуле:

$$H(p_1, \dots, p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i,$$

где p_i – доля объектов i – ого класса, которая попала в данную вершину.

2.5.2. Обучение решающего дерева

При разбиении вершины задача алгоритма так ее разбить, чтобы значение функционала ошибки было наименьшим:

$$Q(R, j, t) = H(R_l) + H(R_r) \rightarrow \min_{j,t}.$$

Однако, у данного функционала есть проблема, он может так разбить выборку, что хоть и энтропия будет наименьшей, но при этом не будет учтено

то, какое количество объектов попало в каждую из дочерних вершин, поэтому обычно используют следующий функционал с весами:

$$Q(R, j, t) = \frac{|R_l|}{|R|} H(R_l) + \frac{|R_r|}{|R|} H(R_r) \to \min_{j,t}.$$

Сам алгоритм такого обучения называется жадным, потому что на каждом шаге он берет наилучший предикат для данной вершины, не меняя при этом уже построенное дерево. Такой подход приводит к переусложнению структуры дерева, что может приводить к переобучению.

Константный прогноз в каждом из листьев определятся чаще всего как самый популярный класс: $c_v = \arg\max_{k \in Y} \sum_{(x_i, y_i)} [y_i = k]$, или как распределение вероятностей классов для объектов листовой вершины: $c_{vk} = \frac{1}{|R|} \sum_{(x_i, y_i)} [y_i = k]$.

В итоге, получаем дерево, которое разбивает признаковое пространство на $R_1, \dots, R_J$ областей, где каждая область R_j соответствует конкретному листу с прогнозом c_j.

На практике данная модель редко используется. Но она является тем строительным кирпичиком, который позволяет строить мощные композиции такие как случайный лес, градиентный бустинг и др.

2.6. Композиция моделей

Композиция моделей комбинирует разное количество базовых моделей в одну, позволяя таким образом добиться лучшего качества, чем при рассмотрении каждой модели по-отдельности. Понятие устойчивости модели позволяет нам частично ответить на вопрос, в чем преимущество такой композиции. Она убирает эффект того, что модель при небольшом изменении обучающей выборки значительно меняется.

2.6.1. Идея композиции моделей

В общем виде имеем: $b_1(x), \dots, b_N(x)$ – базовых моделей, которые хотя бы немного лучше модели случайного угадывания. В случае задачи классификации наша модель композиции будет строиться на идеи голосования большинством, то есть для каждого объекта мы выбираем тот класс, за который проголосовало большинство базовых алгоритмов. Тогда получаем следующую модель:

$$a(x) = \arg\max_{y \in Y} \sum_{i=1}^{N} [b_n = y].$$

Существует два подхода к обучению композиции моделей:

1. Независимое обучение базовых моделей на случайных подвыборках – бэггинг (bagging) и случайные подпространства.
2. Последовательное обучение базовых моделей – бустинг (boosting).

2.6.2. Бэггинг

Бэггинг – композиция базовых моделей, обучающихся независимо на случайных подмножествах объектов, которые генерируются бутстрапом. Бутстрапом называется выборка с возвращением, то есть, имея обучающую выборку X^l, мы берем случайным образом последовательно l объектов, каждый раз возвращая объект в нее. Как правило, в нашей новой выборке получится около 63% уникальных объектов.

2.6.3. Случайный лес

Случайный лес – это довольно мощная универсальная модель, которая позволяет построить композицию деревьев так, чтобы уменьшить между ними связь (корреляцию ошибок), увеличивая тем самым качество модели. Идея данного алгоритма заключается в том, что если во время жадного построения дерева рассматривается разбиение объектов по всем признакам, то теперь в каждой вершине каждого дерева мы берем случайное подмножество признаков

размером q и уже из этого множества выбирается наилучший предикат. Таким образом уменьшается корреляция между моделями деревьев. Эмпирически было получено, что наилучшим числом признаков для классификации является $q = \sqrt{d}$, где d – количество всех признаков.

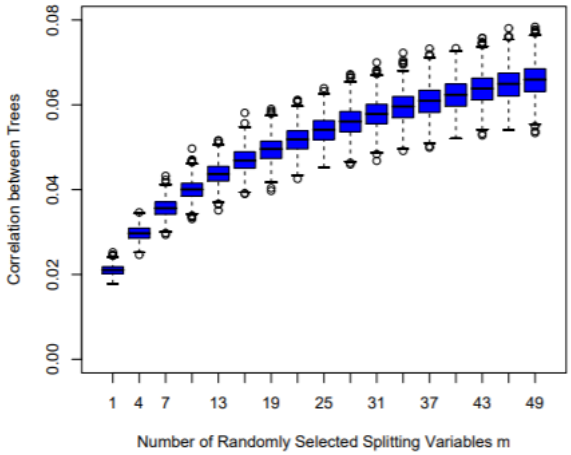


Рисунок 2.2 - Зависимость между корреляцией деревьев и количеством признаков в случайном подмножестве [4]

3. Практическая часть

Данные были получены из базы данных приемной комиссии на период подачи документов за 2019 год по разным программам. В данной таблице выделены следующие признаки по каждому абитуриенту: ”№ п/п”, “Регистрационный номер”, “Фамилия, имя, отчество”, “Подлинник/Копия документа об образовании”, “Медаль / диплом с отличием”, “Право поступления без вступительных испытаний”, “Поступление на места в рамках квоты для лиц, имеющих особое право”, “Поступление на места в рамках квоты целевого приема”, “Наличие согласия на зачисление”, “Математика”, “Иностранный язык”, “Русский язык”, “Экзамен 4”, “Балл за итоговое сочинение”, “Балл за иные достижения”, “Итоговая сумма баллов по индивидуальным достижениям”, “Сумма конкурсных баллов”, “Дата предоставления подлинника”, “Форма обучения”, “Преимущественное право”, “Требуется общежитие на время обучения”, “Договор”, “Договор оплачен”, “Скидка по сумме баллов ЕГЭ”,

“Возврат документов”, “Все выбранные конкурсы”, “Приказ о зачислении”, “Основание зачисления / выбытия”, “Гражданство”.

Будем строить модель для абитуриентов, которые подали документы на конкретную программу. При рассмотрении всех абитуриентов со всех программ вместе упускаем то, что на каждой из программ, во-первых, могут быть разные предметы при поступлении, во-вторых, разный порог при зачислении и/или разные олимпиады при приеме, а, в-третьих, разное количество бюджетных и платных мест. Рассмотрим построение модели для программы «Бизнес-информатика» Факультета бизнеса и менеджмента.

Данные приемной комиссией были собраны с помощью программы Microsoft Office Excel и далее сохранены в формате XLSX. Исследование проходило с помощью языка Python в среде Jupyter Notebook (Anaconda3) (см. **Приложение 1**). С помощью библиотеки Pandas проходила обработка данных. Для построения моделей использовалась библиотека Scikit-learn. Для визуализации данных применялись такие библиотеки как Matplotlib и Seaborn.

В первую очередь необходимо выделить признаки, на основе которых предиктивная модель будет строить прогноз, и целевой признак, который мы и будем прогнозировать. При просмотре целевого признака была обнаружена проблема, связанная с тем, что не для всех абитуриентов проставлены приказы о зачислении. Для того, чтобы восстановить его был взят список студентов 1 курса из текущего рейтинга за 2019/2020 учебного года (1-2 модули). Если данный студент был в рейтинге, то заносим информацию в данную переменную как 1, иначе 0.

Рассмотрим признаки таблицы:

- Подлинник/Копия документа об образовании (is_original) – бинарная переменная. Если “Подлинник” – 1, иначе 0;
- Медаль / диплом с отличием (medal) – бинарная переменная. Если “Да” – 1, иначе 0;

- Право поступления без вступительных испытаний (is_olympiad) – бинарная переменная, означающая тип олимпиады, на основании которой поступает студент. Наличие олимпиады при поступлении – 1, иначе 0;
- Поступление на места в рамках квоты для лиц, имеющих особое право (is_quota_special_right) – бинарная переменная. Если “+” – 1, иначе 0;
- Поступление на места в рамках квоты целевого приема (is_quota_target_reception) – бинарная переменная. Если “+” – 1, иначе 0;
- Наличие согласия на зачисление (is_consent) – бинарная переменная. Если “Да” – 1, иначе 0;
- Математика (math_points) – количественная переменная, которая показывает количество набранных баллов по математике на ЕГЭ;
- Иностранный язык (foreign_points) – количественная переменная, которая показывает количество набранных баллов по иностранному языку на ЕГЭ;
- Русский язык (russian_points) – количественная переменная, которая показывает количество набранных баллов по русскому языку на ЕГЭ;
- Балл за итоговое сочинение (composition_points) – количественная переменная, которая показывает количество баллов за итоговое сочинение в 11 классе;
- Балл за иные достижения (achievement_points) – количественная переменная, которая показывает количество баллов за иные достижения;
- Итоговая сумма баллов индивидуальным достижениям (total_individual_points) – категориальная переменная, которая показывает общую сумму индивидуальных баллов;
- Сумма конкурсных баллов (total_points) – сумма всех баллов;
- Дата предоставления подлинника (grant_date_original) – временная переменная, которая показывает то, когда был предоставлен подлинник в приемную комиссию;
- Форма обучения (educational_form) – категориальная переменная, которая какая форма обучения у абитуриента;

- Преимущественное право (is_preemptive right) – бинарная переменная, которая показывает то обладает ли абитуриент преимущественным правом (1) или нет (0);
- Требуется общежитие на время обучения (is_dormitory) – бинарная переменная, которая говорит о том, нужно ли общежитие абитуриенту (1) или нет (0);
- Договор (is_contract) – бинарная переменная, которая показывает заключен ли договор (1) с абитуриентом или нет (0);
- Договор оплачен (is_pay_contract) – бинарная переменная, которая показывает то, оплачен ли (1) договор или нет (0) абитуриентом;
- Скидка по сумме баллов ЕГЭ (discount) – количественная переменная, которая показывает размер скидки, который получит абитуриент по сумму баллов ЕГЭ;
- Возврат документов (is_return_contract) – бинарная переменная, которая показывает было ли возвращение (1) документов или нет (0);
- Все выбранные конкурсы (alternatives) – возможные альтернативы поступления абитуриентом;
- Приказ о зачислении (enrollment) – категориальная переменная, которая показывает номер приказа о зачислении абитуриента;
- Основание зачисления / выбытия (reason_for_enrollment) – категориальная переменная, которая характеризует причину зачисления абитуриента;
- Гражданство (citiz) – категориальная переменная, которая характеризует то, какое гражданство у абитуриента.

Перед предобработкой данных был проведен разведочный анализ данных (см. **Приложение 2**), в результате которого было определено то, что есть избыточные признаки, пропуски в данных, а также было получено то, что распределения количественных переменных - нормальное. В процессе предобработки данных пропуски заполнялись значениями по смыслу, поскольку были допущены не случайно, но также были такие признаки, где

пропуски было восстановить невозможно. Были удалены неинформативные признаки и признаки, в результате которых возникала проблема мультиколлинерности. Были подкорректированы значения данных у признаков и изменены типы данных у тех признаков, которые были не верно определены. Дополнительно проведена оптимизация памяти с помощью изменения типов данных. Кодирование категориальных признаков проходило с помощью метода One-Hot Encoding, а для масштабирования данных использовалась стандартизация (см. **Приложение 3**).

В ходе анализа целевой переменной было получена, что присутствует проблема дисбаланса классов. После отбора итоговой модели для ее устранения использовался подход взвешивания классов.

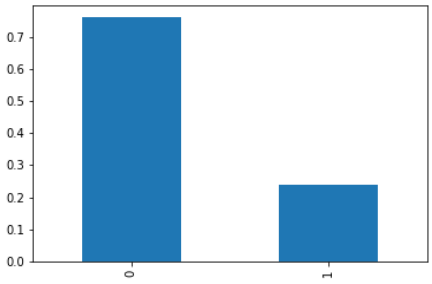


Рисунок 3.1 – Дисбаланс классов

Построение моделей проходило с помощью кросс-валидации. Сначала для каждой модели были подобраны оптимальные гиперпараметры, а уже после снова проходила кросс-валидация на 10 фолдах с целью получения среднего значения метрики качества при данных гиперпараметрах. Обучение и оценка моделей проходила с помощью метрики качества F1, где больший вес дали полноте по сравнению с точностью ($\beta = 2$), так как нам важнее верно

определить поступит или нет абитуриент, чем не верно предсказать, что он поступит. Формула F1 следующая:

$$F1 = (1 + \beta^2)\frac{precision * recall}{\beta^2 * precision + recall}$$

В ходе моделирования средняя метрика качества на всех классах моделей была высокая и почти не отличалась. Ее значение было на уровне 0.89. В связи с этим была взята модель логистической регрессии как довольно простой и надежной. Построены графики PR и ROC – кривых:

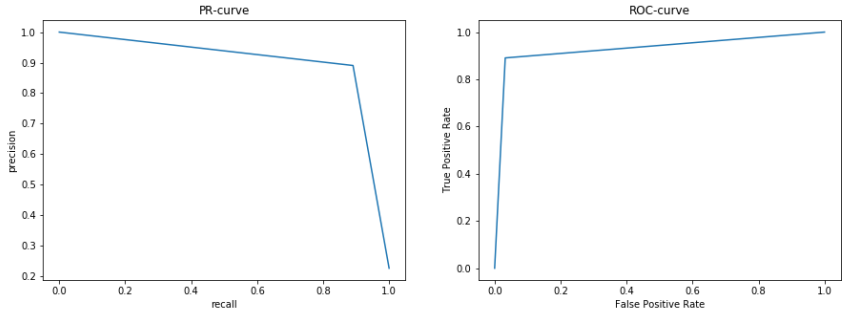


Рисунок 3.2. PR и ROC - кривые

На тестовых данных модель показала качество – 0.891, что является показателем хорошей предиктивной способности модели. Построенная модель смогла предсказать 64 из 64 студентов, которые собирались и поступили в университет. Также модель прошла проверку на адекватность, где константная случайная модель показала качество всего 0.224.

С помощью подхода понижения с помощью моделей были получены веса признаков (см. **Приложение 4**). Рассмотрим первые 10 признаков и их важность (вес) в порядке убывания.

Признак	Вес
is_original	2.87
discount_70	1.33
discount_50	0.66
discount_25	0.5
is_preemptive_right	0.34
citiz_Социалистическая Республика Вьетнам	0.3
no_alternatives	0.3
russian_points	0.28
is_olympiad	0.24

is_return_contract	0.23
citiz_Республика Казахстан	0.2

Для визуализации объектов тестовой выборки на двумерной плоскости по классам были использованы подходы PCA и t-SNE (см. **Приложение 5**).

Заключение

Таким образом, в ходе разработки модели машинного обучения удалось построить такую модель, которая бы давала метрику качества на тестовых данных больше 0.8. Первым делом были изучены зарубежные исследования по похожей теме для того, чтобы понимать в каком возможном направлении необходимо было двигаться при ее решении. Далее были разобраны основные подходы в предобработки данных и построении моделей, а также рассмотрена соответствующая методология, позволяющая вести подобного рода проекты.

Необходимые данные были получены из тех, что были собраны приемной комиссией на период поступления студентов в университет.

Были рассмотрены такие модели как kNN, логистическая регрессия, случайное дерево и случайный лес. В качественной финальной модели была отобрана логистическая. С помощью нее была возможна оценка вероятностей при поступлении абитуриентов, а также на ее основе были отобраны наиболее важные признаки как принес ли абитуриент оригинал, величина скидки, имеет ли он преимущественное право, количество баллов по русскому языку и является ли абитуриент гражданином Вьетнама.

В качестве рекомендации хотелось бы выделить более специфические признаки как доход семьи, из какого региона абитуриент, подавал ли он документы в другие вузы, его интересы, какое образование у родителей и др. при

анализе, а также использование более сложных алгоритмов как градиентный бустинг и с использование базовых моделей с градиентным спуском.

Список литературы

1. Ahmad Slim, Don Hush, Tushar Ojah и Terry Babbitt. Predicting Student Enrollment Based on Student and College Characteristics [Электронный ресурс] / Educational Data Mining Conference. – 2018. Режим доступа: http://educationaldatamining.org/files/conferences/EDM2018/papers/EDM2018_paper_136.pdf, свободный – (03.05.2020).
2. Stephen Kahara, Geoffrey Muchiri. Improving Student Enrollment Prediction Using Ensemble Classifiers [Электронный ресурс]. – 2018. Режим доступа: https://www.researchgate.net/publication/325078017_Improving_Student_Enrollment_Prediction_Using_Ensemble_Classifiers, свободный – (03.05.2020).
3. Nisbet, R., Elder, J., and Miner, G. Handbook of Statistical Analysis and Data Mining Applications. – Amsterdam: Elsevier, 2009.
4. Hastie, T., Tibshirani R., and Friedman, J. The Elements of Statistical Learning. – California: Springer, 2017.
5. http://www.machinelearning.ru/wiki/index.php?title=Заглавная_страница
6. Luuk Derksen. Visualising high-dimensional datasets using PCA and t-SNE in Python [Электронный ресурс]. – 2016. Режим доступа: https://towardsdatascience.com/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b, свободный (17.05.2020).
7. Олег Глушко. Обработка пропусков в данных – часть 1 [Электронный ресурс]. – 2016. Режим доступа: https://basegroup.ru/community/articles/missing, свободный (17.05.2020).
8. Josh Devlin. Руководство по использованию pandas для анализа больших наборов данных [Электронный ресурс]. – 2019. Режим доступа: https://habr.com/ru/company/ruvds/blog/442516/, свободный (17.05.2020).

Приложения

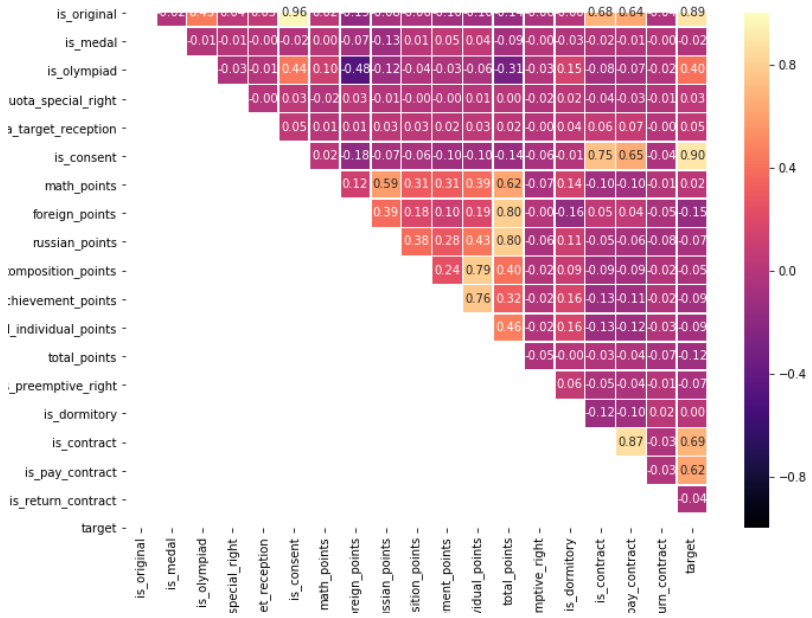
Приложение 1

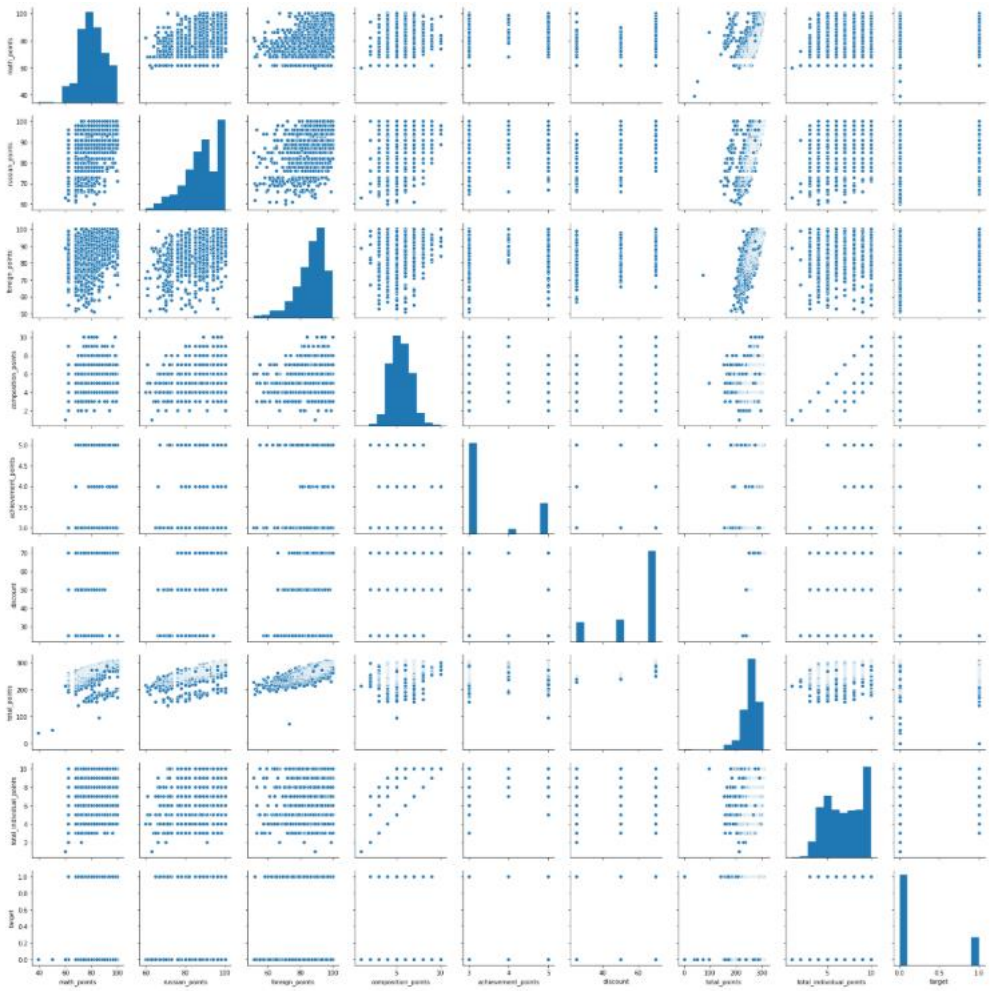
Ссылка на код реализации проекта

https://github.com/knyht/data-science-projects/blob/master/research_reliability_of_borrowers/research_reliability_of_borrowers.ipynb

Приложение 2

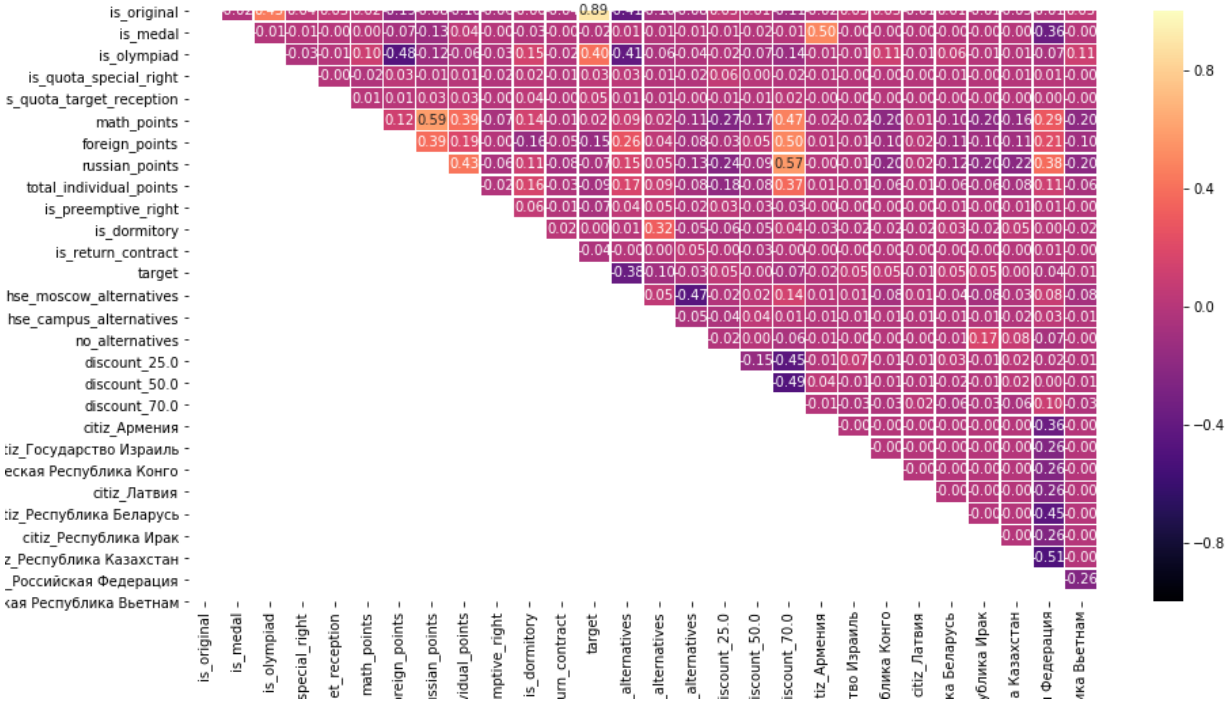
Разведочный анализ данных





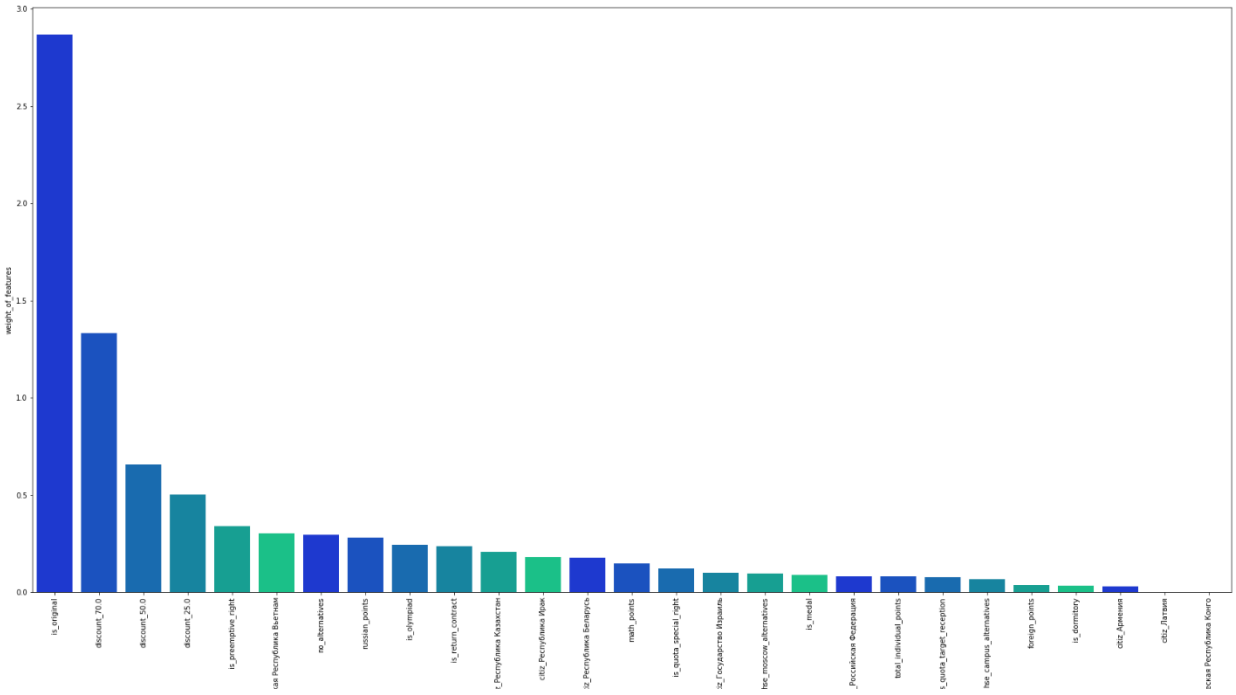
Приложение 3

Матрица корреляций после предобработки данных



Приложение 4

Диаграмма весов признаков



Приложение 5

График разбиения объектов тестовой выборки с помощью PCA и t-SNE