ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В.ЛОМОНОСОВА

Казахстанский филиал
Направление 01.03.01 «Математика»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ПРЕДСКАЗАНИЕ КРЕДИТНОГО РИСКА С ПОМОЩЬЮ ГРАФОВОЙ МОДЕЛИ НЕЙРОННОЙ СЕТИ.

Студентка:
Сланова Айгерим Сакеновна

Научный руководитель:
кандидат физико-математических наук, доцент
Боков Григорий Владимирович

Допустить к защите:

_____
(подпись научного руководителя)
«__» мая 2020 г.

Нур-Султан
2020

Содержание

1 Введение

1.1 Актуальность темы исследования.

История банковских систем показывает, что одной из основных причин снижения потенциальной прибыли и возникновения финансовых трудностей - это неэффективная политика предоставления кредита. Большие потери на финансовом рынке происходят, главным образом из-за отсутствия надлежащего управления рисками. Примером недостаточного уровня управления рисками является финансовый кризис 2007 - 2008 годов. Одним из факторов, тесно связанных с кризисом, был пузырь на рынке недвижимости в США [1]. Кредит в течение этого периода был присвоен «субстандартным заемщикам» - лицамы, которые имели низкий доход или активы. Исходя из этого, базовые ставки по ипотечным кредитам и ипотекам с регулируемой процентом начали быстро расти [1]. Субстандартные кредиторы начали подавать на банкротство, так как все больше заемщиков не смогли быть платежеспособными. Отто Ван Хемерт и Юлия Демяник [3] в 2011 обнаружили, что в течение шести лет подряд до кризиса качество субстандартных кредитов ухудшалось. Оценки потерь, понесенных пузырем на рынке жилья, составляют триллионы долларов [2]. С конца прошлого десятилетия сфера управления банковскими рисками расцвела, и во многих секторах возросла важность оценки кредитного риска. Под кредитным риском обычно понимается вероятность невозврата заемщиком кредита в установленный срок.

В кредитной политике банки одновременно заинтересованы в двух вещах:

1. Увеличить число кредитов клиентам,

2. Сократить финансовые риски.

Поэтому банкам необходимы быстрые и надёжные средства проверки кредитоспособности клиентов.

Принимая во внимание, что банковский сектор прошел огромную эволюцию и что банки в настоящее время предлагают впечатляющий набор продуктов для выявления потенциальных «хороших» и «плохих» клиентов, следует помнить, что банки являются связующим звеном между заемщиками и кредиторами, и что деньги, предоставленные заемщикам, фактически являются деньгами, полученными банком от вкладов физических и юридических лиц. Соответственно, на банках лежит большая ответственность за правильный выбор заемщиков с высокой кредитоспособностью. Для достижения этой цели, банки используют несколько методов для оценки кредитного риска, которые делятся на следующие основные группы: нейронные сети, статистические методы, нечётко-множественные описания, экспертные методы. Выбор подхода зависит от многих факторов, к ним можно отнести: субъективные предпочтения банка, наличие и качество исходных данных, цели и задачи построения модели.

1.2 Состояние проблемы.

Современные концепции и идеи анализа кредитного скоринга появились в 1941 году благодаря Дюрану [4]. С тех пор трейдеры начали собирать информацию о претендентах на кредит и классифицировать их, чтобы принимать решение о предоставлении определенной суммы денег. По словам Лин С. Томас [5] кредитный скоринг - «набор моделей принятия решений и лежащих в их основе методов, которые помогают кредиторам в предоставлении кредита». В то же время, современные статистические методы и методы анализа данных внесли значительный вклад в область информатики и способны создавать модели для измерения уровня риска отдельного клиента, обусловленного его характеристиками, а затем классифицировать его как хорошего или плохого клиента в соответствии с уровнем риска.

Одним из методов, который сможет дать вероятностную оценку кредитного риска, является нейросетевой подход. Использование нейронных сетей в бизнес- приложениях ранее исследовано несколькими работами [9], [10], [11], [12], [13], [14]. Общий результат таких работ заключается в том, что в кредитной отрасли нейронные сети считаются одним из точных инструментов кредитного анализа. В своей работе Н. С. Лукашевич [15] отмечает следующие стороны нейросетевого подхода:

- положительные стороны:

  Автоматизация, поскольку получение результатов требует обработки большого объема данных в процессе обучения;

  Объективность. Субъективность заключается в выборе топологии сети и алгоритма обучения;

- отрицательные стороны:

  Точность зависит от качества исходных данных;

  Высокие временные затраты на обучение сети, сложны взаимосвязи факторов;

  Отсутствие объяснимости. Существует проблема в интерпретируемости весовых коэффициентов.

Нейросетевой метод является одним из традиционных в оценке кредитного риска реализованным в большинстве современных банковских программных продуктов.

1.3 Теоретическая и методологическая основа исследования.

Для оценки кредитоспособности банк перечисляет ряд вопросов, относящиеся к заемщику, который в свою очередь предоставляет свои ответы на основе набора возможных ответов, называемых атрибутами. Оценка заемщика осуществляется по многим качественным и количественным характеристикам. Например,

- Статус существующего текущего счета;

- Кредитная история;

- Сумма кредита;

- Сберегательный счет;

- и так далее.

Этот подход был исследован многими авторами. Среди них можно выделить таких, как:

- *Китара Ал-Шея*[6], которая предложила нейронную сеть обратного распространения, у которой нет циклов или петель, состоящую из трех слоев: входного слоя, скрытого слоя и выходного слоя;

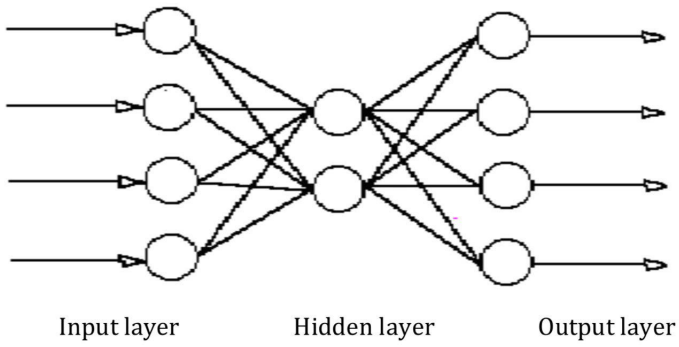

Рис. 1: Архитектура нейронной сети в модели Китары Ал-Шея.

- *Кашман*[7], который исследовал оптимальное количество скрытых слоев для нейронной сети обратного распространения;

- *Дэвида Уэст* [8], который сравнивал нейросетевой поход с другими методами.

В вышеперечисленных работах известные и доступные наборы данных Австралии [1] и Германии [2], взятые с сайта UCI Machine Learning Repository применялись в классификации, то есть в распределении данных по атрибутам, но не учитывалась взаимосвязь между атрибутами. Для более подробной характеристики заемщика необходимо больше информации. Например, как взаимодействуют атрибуты между собой в классификации. Данное взаимодействие атрибутов поможет устранить отсутствие объяснимости, на которое указал Н. С. Лукашевич в своей работе [15]. Ограничением этого является малое разнообразие в наборе данных, что также является следствием политики конфиденциальности персональных данных клиентов банка.

[1] http://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval)

[2] https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

1.4 Цели и задачи исследования.

Целью данной работы является исследование возможности применения технологий искусственного интеллекта и машинного обучения в выявлении связей между атрибутами (характеристиками) заемщиков, в управлении оценки кредитных рисков, оценки эффективности данного применения.

В соответствие с целью в работе поставлены следующие задачи:

- найти взаимосвязь между атрибутами и, опираясь на доступные источники в интернете, пояснить зависимость одного атрибута от другого, и наоборот;
- численно оценить взаимосвязь атрибутов и представить в виде таблицы значений;
- разработать модель обучения, учитывающую взаимосвязь атрибутов;
- подготовить данные для обучения модели;
- оценить точность и полноту классификации;

1.5 Научная новизна работы.

Научная новизна заключается в построении новой модели классификации, основанной на взаимосвязи характеристик клиентов банка.

2 Построение модели.

Построение модели состоит из нескольких этапов:

- Получить данные в доступном формате;
- Доопределить / исправить отсутствующие / аномальные значения данных;
- Подготовить данные для модели машинного обучения;
- Установите базовую модель;
- Разбиение данных на тренировочный и контрольный наборы;
- Обучить модель на тренировочных данных;
- Получить прогнозы по контрольным данным;
- Провести анализ по полученным результатам.

2.1 Набор данных.

В качестве входных данных для исследовательской цели используются немецкие данные[3] за 1994 год, которые классифицируют клиентов, на основе 20 социально-экономических атрибутов (характеристик) банка, как «хороший» и «плохой».

Данные содержат 14 категориальных и 6 числовых переменных, которые приведены на Рис. 2 и Рис. 3, и состоят из 1000 наблюдений на Рис. 4, которые делятся на 700 «хороших» и 300 «плохих» клиентов, пропорции этих классов отличаются - 70% и 30% соответственно, что говорит о несбалансированности данных. Несбалансированность негативно влияет на работу классификатора, так как он игнорирует малочисленный класс, что приводит плохому результату [16].

Для устранения возникшей проблемы проводится предварительная обработка данных (2.1.1) :

1. Поиск недостающих и аномальных данных;
2. Кодирование категориальных переменных;
3. Стандартизация и масштабирование данных;
4. Деление данных на тестовые и обучающие наборы;
5. Метод обработки несбалансированных данных.

2.1.1 Предварительная обработка данных.

Поиск недостающих и аномальных данных.

В данном наборе недостающие данные отсутствуют.

Аномальные значения (выбросы) являются значениями, которые сильно отличаются от большинства значений в наборе данных. Выбросы могут быть аномально высокими или низкими значениями, из-за этого они искажают статистические характеристики данных, такие как среднее и дисперсию, которые играют важную роль в стандартизации и нормализации данных (2.1.1). К выбросам, к примеру, можно отнести возраст клиента, если минимальное его значение будет равно несовершеннолетнему. Отсутствие аномальных данных объясняется отсутствием выбросов в диапазоне числовых значений на Рис. 5.

Кодирование категориальных переменных.

[3] https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

№	Атрибуты	Вид
1	статус текущего счета	(качественный) A11: ... < 0 DM A12 : 0 <= ... < 200 DM A13 : ... >= 200 DM/заработная плата не менее за 1 год A14 : нет текущего счета
2	срок кредита	(количественный) продолжительность в месяцах
3	кредитная история	(качественный) A30 : не брал(-а) кредитов/все кредиты возвращены должным образом A31 : все кредиты в этом банке выплачены должным образом A32 :существующие кредиты возвращены должным образом до настоящего времени A33 : задержка в выплате в прошлом A34 : критический счет/другие существующие кредиты (не в этом банке)
4	назначение кредита	(качественный) A40 : автомобиль (новый) A41 : автомобиль (б/у) A42 : Мебель / оборудование A43 : радио/ телевизор A44 : бытовая техника A45 : ремонт A46 : образование A47 : (отпуск- не существует?) A48 : переквалификация A49 : бизнес A410 : другое
5	сумма кредита	(количественный)
6	наличие сберегательного счета/ облигаций	A61 : ... < 100 DM A62 : 100 <= ... < 500 DM A63 : 500 <= ... < 1000 DM A64 : .. >= 1000 DM A65 : неизвестно / нет сберегательного счета
7	занятость по настоящее время	(качественный) A71 : безработный A72 : ... < 1 год A73 : 1 <= ... < 4 года A74 : 4 <= ... < 7 лет A75 : .. >= 7 лет
8	процентная ставка от располагаемого дохода	(количественный)
9	Личный статус и пол	(качественный) A91 : мужчина: разведенный/ разведенный A92 : женщина : разведенная /разведенная/замужем A93 : мужчина : не женат A94 : мужчина: женат/вдовец A95 : женщина : не замужем
10	Другие должники / поручители	(качественный) A101 : нету A102 : созаявитель A103 : поручитель

Рис. 2: Атрибуты входных данных.

Набор данных состоит из комбинации категориальных и числовых переменных. В этом наборе данных численные переменные, такие как *срок кредита*, *сумма кредита*, *процентная ставка* и т.д., являются непрерывными признаками. Однако переменные *статус текущего счета*, *кредитная история*, *назначение кредита* и т.д.,

11	Настоящее место жительства с	(количественный)
12	наличие собственности	A121 : недвижимость A122 : если нет A121 : соглашение о сбережениях строительного общества / страхование жизни A123 : если нет A121/A122 : автомобиль или другое, нет в характеристики 6 A124 : неизвестный/ нет собственности
13	Возраст	(количественный)
14	другие виды взноса	(качественный) A141 : банк A142 : магазины A143 : нету
15	жилье (свое\аренда\бесплатно)	(качественный) A151 : аренда A152 : собственное A153 :бесплатное
16	количество кредитов, существующих в этом банке	(количественный)
17	Работа	(качественный) A171 : безработный / неопытный - иногородний A172 : не квалифицированный - постоянный житель A173 : квалифицированный сотрудник / должностное лицо A174 : руководство / частный предприниматель / высококвалифицированный сотрудник/ должностное лицо
18	количество людей, ответственных за исполнение обязательств	(количественный)
19	наличие контактного телефона	(качественный) A191 : отсутствует A192 : да, зарегистрирован под именем клиента
20	иностранный гражданин	(качественный) A201 : да A202 : нет

Рис. 3: Атрибуты входных данных.

являются категориальными на Рис. 2 и Рис. 3. Вместо диапазона все они имеют фиксированный список возможных значений и обозначают качественный признак, а не непрерывный.

На данный момент наиболее распространенным способом представления категориальных переменных в [21] является *прямое кодирование (one-hot-encoding)* [4].

Идея [5] на Рис. 6, лежащая в основе прямого кодирования, заключается в том, что для кодируемого категориального признака создаются N новых признаков, где N — количество категорий. Каждый i-й новый признак — бинарный характеристический

[4] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

[5] https://dyakonov.org/2016/08/03/python

	checking	duration	history	purpose	amount	savings	employed	installp	marital	coapp	...	property	age	other	housing	existcr	job	depends	telephon	foreign	good_bad
0	A11	6	A34	A43	1169	A65	A75	4	A93	A101	...	A121	67	A143	A152	2	A173	1	A192	A201	1
1	A12	48	A32	A43	5951	A61	A73	2	A92	A101	...	A121	22	A143	A152	1	A173	1	A191	A201	2
2	A14	12	A34	A46	2096	A61	A74	2	A93	A101	...	A121	49	A143	A152	1	A172	2	A191	A201	1
3	A11	42	A32	A42	7882	A61	A74	2	A93	A103	...	A122	45	A143	A153	1	A173	2	A191	A201	1
4	A11	24	A33	A40	4870	A61	A73	3	A93	A101	...	A124	53	A143	A153	2	A173	2	A191	A201	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	A14	12	A32	A42	1736	A61	A74	3	A92	A101	...	A121	31	A143	A152	1	A172	1	A191	A201	1
996	A11	30	A32	A41	3857	A61	A73	4	A91	A101	...	A122	40	A143	A152	1	A174	1	A192	A201	1
997	A14	12	A32	A43	804	A61	A75	4	A93	A101	...	A123	38	A143	A152	1	A173	1	A191	A201	1
998	A11	45	A32	A43	1845	A61	A73	4	A93	A101	...	A124	23	A143	A153	1	A173	1	A192	A201	2
999	A12	45	A34	A41	4576	A62	A71	3	A93	A101	...	A123	27	A143	A152	1	A173	1	A191	A201	1

1000 rows × 21 columns

Рис. 4: Входные данные.

```
duration :  4  -  72
amount :  250  -  18424
installp :  1  -  4
resident :  1  -  4
age :  19  -  75
existcr :  1  -  4
depends :  1  -  2
good_bad :  0  -  1
```

Рис. 5: Диапазон числовых переменных.

признак i-й категории. Например, некоторый признак может принимать 10 разных значений. В этом случае One-Hot-Encoding подразумевает создание 10 признаков, все из которых равны нулю за исключением одного. На позицию, соответствующую численному значению признака ставится 1.

Итог кодирования категориальных переменных - это увеличение размера атрибутов на Рис. 7: было 20, стало 61.

Стандартизация и масштабирование данных.

Некоторые классификаторы, которые описаны в (2.3), чувствительны к масштабированию данных. Чтобы итоговое представление данных было более подходящим для использования классификаторов, часто используется один из методов корректировки данных:

- Стандартизация. Она гарантирует, что для каждого признака среднее будет равно 0, а дисперсия будет равна 1, в результате чего все признак будут иметь один и тот же масштаб. Стандартизация данных происходит с помощью библиотеки scikit-learn с классом StandardScaler .

- Масштабирование данных. Данные состоят из переменных с различными мас-

```
checking :  ['A11' 'A12' 'A14' 'A13']
checking :  [0 1 3 2]
history :  ['A34' 'A32' 'A33' 'A30' 'A31']
history :  [4 2 3 0 1]
purpose :  ['A43' 'A46' 'A42' 'A40' 'A41' 'A49' 'A44' 'A45' 'A410' 'A48']
purpose :  [4 7 3 0 1 9 5 6 2 8]
savings :  ['A65' 'A61' 'A63' 'A64' 'A62']
savings :  [4 0 2 3 1]
employed :  ['A75' 'A73' 'A74' 'A71' 'A72']
employed :  [4 2 3 0 1]
marital :  ['A93' 'A92' 'A91' 'A94']
marital :  [2 1 0 3]
coapp :  ['A101' 'A103' 'A102']
coapp :  [0 2 1]
property :  ['A121' 'A122' 'A124' 'A123']
property :  [0 1 3 2]
other :  ['A143' 'A141' 'A142']
other :  [2 0 1]
housing :  ['A152' 'A153' 'A151']
housing :  [1 2 0]
job :  ['A173' 'A172' 'A174' 'A171']
job :  [2 1 3 0]
telephon :  ['A192' 'A191']
telephon :  [1 0]
foreign :  ['A201' 'A202']
foreign :  [0 1]
```

Рис. 6: Кодирование категориальных переменных.

	duration	amount	installp	resident	age	existcr	depends	checking_A11	checking_A12	checking_A13	...	job_A172	job_A173	job_A174	telephon_A191	telephon_A192	foreign_A201	foreign_A202
0	6	1169	4	4	67	2	1	1	0	0	...	0	1	0	0	1	1	0
1	48	5951	2	2	22	1	1	0	1	0	...	0	1	0	1	0	1	0
2	12	2096	2	3	49	1	2	0	0	0	...	1	0	0	1	0	1	0
3	42	7882	2	4	45	1	2	1	0	0	...	0	1	0	1	0	1	0
4	24	4870	3	4	53	2	2	1	0	0	...	0	1	0	1	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	12	1736	3	4	31	1	1	0	0	0	...	1	0	0	1	0	1	0
996	30	3857	4	4	40	1	1	1	0	0	...	0	0	1	0	1	1	0
997	12	804	4	4	38	1	1	0	0	0	...	0	1	0	1	0	1	0
998	45	1845	4	4	23	1	1	1	0	0	...	0	1	0	0	1	1	0
999	45	4576	3	4	27	1	1	0	1	0	...	0	1	0	1	0	1	0

1000 rows × 61 columns

Рис. 7: После кодирования категориальных переменных.

штабами, для оптимизации работы классификаторов машинного обучения важно, чтобы все имели одинаковый масштаб. Этот метод сдвигает данные таким образом, что все признаки находились строго в диапазоне от 0 до 1. Для изменения масштаба данных используется класс MinMaxScaler из библиотеки scikit-learn.

				числа M:																		
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	статус текущего счета	checking	1	0	0.8	0.6	0	0	0.8	0.9	0	0	0	0	0.5	0.3	0	0.6	0.7	1	0	0	0.7
2	срок кредита в месяцах	duration	2	0.8	0	0	0	0.9	0.9	1	0	0	0.8	0.7	0.6	0.5	0	0.3	0	1	0.6	0.7	0.7
3	кредитная история	history	3	0.8	0.8	0	0	0.8	0.9	1	0.3	0	0.7	0.7	0.5	0.6	0.7	0.5	0.8	1	0.4	0.7	0.7
4	назначение	purpose	4	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0.7
5	сумма кредита	amount	5	1	0.8	0.8	1	0	0.7	0.8	0.8	0	0.9	0.8	0.6	0.8	0.7	0.6	1	1	0.7	0	0.7
6	сберегательный счет\облигации	savings	6	0.8	0.9	0	0	0	0	0.8	0.8	0	0	0	0.5	0.3	0	0.6	0.3	0.8	0	0	0.7
7	занятость по настоящее время	employed	7	0	0	0	0	0	0	0	0	0	0	0.5	0	0.7	0	0.5	0	1	0	0	0.7
8	процентная ставка от располагаемого дохода	installp	8	0.8	0	0	0	0	0.9	0.8	0	0	0	0.7	0	0	0	0	0	1	0	0.7	0.7
9	личный статус и пол	marital	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7
10	другие должники\поручитель	coapp	10	0	0.8	0	0	0.7	0	0	0	0.8	0	0	0	0	0	0	0.7	1	1	0	0.7
11	настоящее место жительства с (количество)	resident	11	0	0	0	0	0	0	0	0	1	0	0	0.9	0	0	1	0	0	0	0	0.7
12	собственность	property	12	0	0	0	0	0	0	0	0	0	0	0.4	0	0.5	0	1	0	0.9	0	0	0.7
13	возраст в цифрах	age	13	0	0	0	0	0	0	0	0	0.8	0	0	0	0	0	0	0	0	0	0	0.7
14	другие виды взноса	other	14	0	0	0.7	0.5	0.8	0.4	0.9	0.9	0	0.9	0	0	0.5	0	0.7	0.9	0.7	0	0	0.7
15	жилье (свое\аренда\бесплатно)	housing	15	0	0	0	0	0	0	0	0	0.7	0	0.8	0.9	0.8	0	0	0	0.7	0	0	0.7
16	количество существующих кредитов в этом банке	existr	16	0.6	0	0.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7
17	стаж работы	job	17	0	0	0	0	0	0	1	0	0.6	0	0	0	0.8	0	0	0	0	0	0.8	0.7
18	количество людей, ответственных за исполнение обязательств	depends	18	0	0.7	0.5	0.4	0.8	0	0	0	0.5	1	0	0	0	0	0	0	0	0	0.8	0.7
19	телефон	telephon	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7
20	иностранный рабочий	foreign	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

числа N

как числа N засисят от чисел M

Рис. 8: Таблица отношений.

Деление данных на тестовые и обучающие наборы.

Разбиение данных на обучающий и тестовый набор производится в пропорциях. Применяется функция train_test_split из модуля sklearn.model_selection. Вводятся параметры:

- test_size = 0.40, что означает, что деление данных на обучающий/тестовый набор происходит в отношении 60%:40%;
- random_state = 1, начальное значение для генерации случайных чисел (по дефолту его нет, но для воспроизводимых результатов укажем любое число типа int).

Остальные параметры по умолчанию[6].

Метод обработки несбалансированных данных.

Для решения несбалансированности используется один из методов обработки несбалансированных данных SMOTE (метод передискретизации синтетического меньшинства), идея которого основана на генерации некоторого количества искусственных примеров, которые были бы «похожи» на имеющиеся в меньшем классе, но при этом не дублировали их [17].

Вводится параметр random_state = 1, остальные параметры по умолчанию[7].

2.1.2 Пояснение.

Представим 20 атрибутов в виде вершин графа и создадим таблицу отношений между атрибутами. Взаимосвязь атрибутов будем оценивать в обе стороны. Например,

[6] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[7] https://imbalanced-learn.org/stable/generated/imblearn.over_sampling.SMOTE.html

как один влияет на другого, и наоборот. Их зависимости будут варьироваться и будут равны числу из отрезка $[0, 1]$.

Далее, будут приведены несколько пояснений зависимостей между характеристиками. Предложенные мной коэффициенты от 0 до 1, будут расположены в скобках и преобразованы в виде Таблицы отношений на Рис. 8.

1. Сумма кредита от кредитной истории[8](0.8).
   Банки в обязательном порядке проверяют кредитную историю заёмщика. Для этого сотрудники делают запрос в Бюро кредитных историй, где собирается информация от всех кредитных организаций, в том числе и в сфере микрокредитования.

2. Кредитной истории от пола[9] (0).
   Зависимости нет, мужчины и женщины одинаково возвращают кредиты.

3. Кредитной истории от возраста (0.6).
   Показатель хуже всего у людей в возрасте до 25 лет. Лучшими заемщиками являются люди в возрасте 30 - 40 лет.

4. Кредитной истории от места жительства[10] (0.5).
   Более высокие оценки будут у жителей крупных городов. Связано это с материальным положением. Частая смена места жительства негативно влияет на оценку заемщика.

5. Кредитной истории от наличия собственности[11] (0.5).
   Наличие собственности является гарантом, что заемщик справится со взятыми на себя обязательствами.

6. Кредитной истории от стажа работы (1) и занятости по настоящее время (1).[12]
   Одним из основных требованием банков является непрерывный трудовой стаж на последнем месте работы минимум три месяца. Надежным заемщиком для кредитного учреждения является человек с большим стажем, измеряемым в годах, на последнем месте работы. Отрицательно влияет на оценку большие перерывы в стаже.

7. Стаж работы от пола (личного статуса) (0.6)[13]. Трудовой стаж идет только тем женщинам, которые непосредственно работали до выхода в декрет, но и его длительность ограничена шестью годами.

[8] https://pravo.ru/news/208378/
[9] https://www.kp.by/daily/26808/3843790/ находятся зависимости 2-4
[10] https://www.banki.ru/news/daytheme/?id=10908537
[11] https://obzor.city/article/357693
[12] https://obzor.city/article/357693
[13] https://www.gazeta.ru/business/2019/10/31/12787910.shtml

8. Между сроком кредита в месяцах и суммой кредита (1).[14]
   Срок кредита напрямую связан с его суммой. Чем выше его сумма, тем больше срок.

9. Между поручительством и сроком кредита[15] (0.8).
   Если срок большой, то в качестве подтверждения требуют предоставления поручительства. И, наоборот, банк предоставит большую сумму кредита и, следовательно, большой срок, если будет участвовать в деле поручитель.

10. Сумма кредита от назначения (1).[16].
    Сумма кредита зависит от целей, которых клиент не может достичь за счёт собственных финансовых средств. В банке предпринимают меры, чтобы заемные деньги использовались строго по назначению, если это не потребительский кредит.

11. Между назначением кредита и остальными характеристиками.[17]
    В основном на назначение кредита влияет занятость заемщика по настоящее время (1), возраст(1), пол (1) и стаж работы(1).

12. Кредитная история от срока кредита (0.8) и суммы кредита(0.8)[18].
    В ответ на банковский запрос с согласием заемщика бюро кредитных историй (БКИ) отправляет информацию о взятых кредитах заемщиком за последние 10 лет, включая те, с которыми он полностью рассчитался. Данные о просрочке разбиты на периоды: просрочки сроком до 1 месяца, просрочки сроком от 1 месяца до 3, просрочки сроком от 3 месяцев до 6 и т.д. Поэтому просрочка по кредиту в банке сроком 2 дня или 30 дней в отчете БКИ отразится одинаково - просрочка сроком до 1 месяца.

13. Между процентной ставкой от располагаемого дохода и другими характеристиками.[19]
    Процент по кредиту вычисляют относительно дохода заемщика. Отношение суммы ежемесячных платежей по предполагаемому кредиту к сумме ежемесячного дохода заемщика (после вычета всех налогов). Применяется банками при расчете суммы возможного займа. В качестве дохода можем взять характеристики, такие как статус текущего счета(0.8), занятость по настоящее время(0.8) и стаж работы(0.8).

14. Влияние местожительства и наличия контактного телефона на срок кредита(0.7), процентную ставку(0.7), количество существующих кредитов в этом банке(0.7) и

[14] https://mos-zalog.ru/kredit-pod-zalog/sroki-kreditovanija/
[15] https://mos-zalog.ru/kredit-pod-zalog/sroki-kreditovanija/
[16] https://www.e-xecutive.ru/wiki/index.php/%D0%9A%D1%80%D0%B5%D0%B4%D0%B8%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5
[17] https://www.kommersant.ru/doc/3888369
[18] https://dolgam.net/berem-v-dolg-pravilno/kreditnaya-istoriya-grazhdanina/
[19] http://www.banki.ru/wikibank/koeffitsien_platej-dohod/

кредитную историю (0.7). [20](С остальными характеристиками предположительно возьмем (0)).
Большинство банков требуют уведомлять их о смене места проживания, прописки, работы, контактных номеров телефонов, имени, фамилии. Эти данные нужны банкирам, чтобы правильно идентифицировать клиента и иметь возможность с ним связаться в любой момент. Неуведомление банка о смене места жительства, работы или семейного положения может повлечь для заемщика серьезные финансовые санкции – от увеличения кредитной ставки до требования полностью погасить кредит.

15. Взаимосвязь между иностранным гражданином и другими характеристиками. (0.7)[21, 22]. Законы не запрещают банкам кредитовать граждан других стран. Но банки опасаются выдавать иностранцам кредиты без залога, потому что они ничем не обеспечены. Поэтому к иностранным гражданам требования гораздо жестче. Следовательно, является ли заемщик гражданином данной страны или нет имеет большое влияние на выдачу кредита.

2.2 Предлагаемая модель.

В данном разделе будет предложена графовая модель предсказаний кредитного риска, состоящая из методов №2, №3 и №4, и произведено сравнение с методом №1 других авторов.

1. Метод «чистых» данных;
2. Метод ненормированной связи данных с весами взаимоотношений;
3. Метод нормированной связи данных с весами взаимоотношений;
4. Метод расширенных данных: объединение данных метода №1 и метода №3;

2.2.1 Метод «чистых» данных.

Метод реализован с помощью описанных ранних работ Варша Айтал и Рошан Давид Джатанна [18] и Друв Дханеш Танавала [19], построение модели которого состоит из стандартных этапов.

Графовая модель предсказаний кредитного риска.

[20] http://dengi.ua/clauses/90284_Chem_opasno_zaemshhiku_molchanie_o_peremenah_v_zhizni_i_karere.html
[21] https://www.credy.kz/kredity-inostrancam/
[22] https://journal.tinkoff.ru/ask/bank-ne-daet-kredit/

Свёрточные нейронные сети.

Свёрточные нейронные сети (CNN) обычно используются для классификации изображений и обработки аудио. Идея свёрточной нейронной сети основана на свёртке. Свёртка – это фильтр, который проходит по изображению, обрабатывает его и выделяет признаки, которые можно использовать для того, чтобы понять, что у изображений есть общего. После фильтрации признаки изображений будут более очевидными. И тогда можно их использовать, чтобы распознать предметы.

Введем обозначения:

- Граф $G = (V, E)$, где V - множество вершин, каждую вершину задает атрибут, E - множество упорядоченных пар вершин графа;

- Матрица смежности графа - это матрица A, $\dim(A) = n \times n$, где n - это количество атрибутов (узлов графа), в которой значение ее элемента a_{ij}, равно числу ребер, связывающих вершины v_i и v_j, где $i, j = \overline{1, n}$;

- Матрица степеней графа - диагональная матрица D, $\dim(D) = n \times n$, в которой её элемент d_{ii} содержит информацию о степени i-ой вершины, то есть о количестве ребер прикрепленных к вершине, где $i = \overline{1, n}$.

- Нормализованная матрица смежности A задается путем деления столбцов A на степени соответствующих узлов.

- Набор входных данных - матрица X, $\dim(X) = k \times n$, где k - количество входных данных

Скрытый слой сети имеет вид

$$H^i = f(H^{i-1}, A),$$

где H^i соответствует выходам i -го слоя, $H^0 = X$, f - нелинейная функция.

$$f(H^{i-1}, A) = \sigma(AH^{i-1}),$$

где σ - нелинейная функция активации.

Чтобы матрица смежности для каждого узла учитывала сам узел, кроме того, что суммирует все векторы признаков всех соседних узлов, к матрице A прибавляется единичная матрица I.

$$\tilde{A} = A + I$$

Матрица смежности $\tilde{A}$ не нормализована. Следовательно, умножение на $\tilde{A}$ полностью изменит масштаб вектор-признаков. Признаки могут быть нормализованы по степени узла путем умножения матрицы смежности $\tilde{A}$ на обратную матрицу матрицы степени D, то есть умножение $D^{-1}\tilde{A}$ соответствует взятию среднего значения атрибутов соседних узлов, включая исходный узел.

2.2.2 Метод ненормированной связи набор данных с весами взаимоотношений.

Набор данных клиентов банка (German data set) является входными данными. Для взаимосвязи входных данных с весами зависимостей атрибутов используется принцип работы свёрточной сети, а точнее её аналог для работы с графовыми данными на первом слое $i = 0$. В качестве матрицы смежности будет выступать таблица отношений. Входными данными для обучения модели метода №2 являются результатами формулы (1).

$$\tilde{A}X \tag{1}$$

2.2.3 Метод нормированной связи данных с весами взаимоотношений.

Реализация данного метода происходит путем добавления нормированной матрицы смежности, в данном случае таблицы отношений. Следовательно, входными данными для обучения модели метода №3 являются результатами формулы (2).

$$D^{-1}\tilde{A}X \tag{2}$$

2.2.4 Метод расширенных данных: объединение данных метода №1 и метода №3.

Данный метод реализуется путем добавления преобразованных данных из метода №3 к методу №1 «чистым» данным.

2.3 Обучение.

Для обучения модели рассмотрим несколько классификаторов и их краткое описание:

1. Деревья:
   (a) Дерево решений;
   (b) Случайный лес;
   (c) Лес сверхслучайных (сильнорандомизированных) деревьев.
2. Метод опорных векторов:
   (a) Нелинейный классификатор опорных векторов;
   (b) Линейный классификатор опорных векторов.
3. Градиентый бустинг;
   (a) Стохастический градиентный спуск ;

Деревья.

Дерево решений.
Дерево решений рассматривается как серия вопросов да/нет, решения которых принимаются так же, как и человеком: задаются вопросы о доступных данных до тех пор, пока не приходим к определённому решению. Визуализация работы дерева решений представлена на Рис. 9.

Используется класс DecisionTreeClassifier в библиотеке Scikit-learn с параметрами:

- min_samples_split = 10 — минимальное количество объектов, необходимое для разделения внутреннего узла.
- min_samples_leaf = 5 — минимальное число объектов в листе;
- random_state = 42 — результаты будут одинаковыми при каждом запуске разбиения для получения воспроизводимых результатов;

  остальные параметры по умолчанию, с которыми можно ознакомиться на сайте[23]:
- max_depth: максимальная глубина дерева — точка, на которой останавливается разбиение узлов (по дефолту — None);
- max_features: число признаков для поиска лучшей точки для разбиения. Чем больше число, тем лучше результат. Но в этом случае обучение займет больше времени (по дефолту — None);
- min_impurity_split: порог для ранней остановки роста дерева. Узел разобьется только в том случае, если его точность будет выше указанного порога (по дефолту — None);

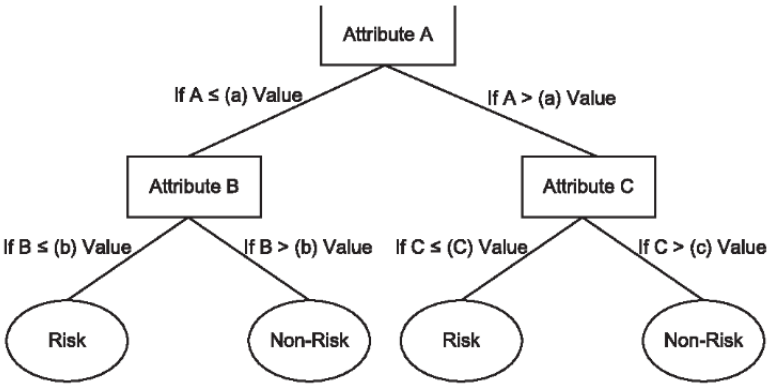

Рис. 9: Дерево решений.

[23] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

Случайный лес.
Алгоритм случайного леса на Рис. 10 сочетает в себе две основные идеи:

- метод бэггинга [24]Бреймана.
- метод случайных подпространств, предложенный Тин Кам Ху.

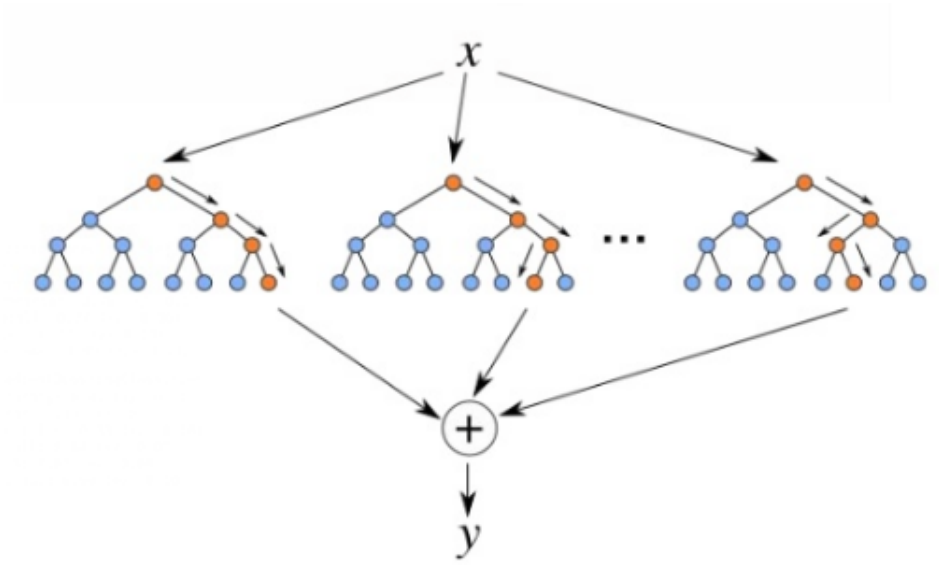

Рис. 10: Случайный лес.

Основная идея заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим.

Используется класс RandomForestClassifier в библиотеке Scikit-learn с параметрами:

- random_state = 42;

  по умолчанию, с которыми можно ознакомиться на сайте [25] :
- min_samples_split — минимальное количество объектов, необходимое для разделения внутреннего узла. Можно задать числом или процентом от общего числа объектов (по дефолту — 2);
- min_samples_leaf — минимальное число объектов в листе. Можно задать числом или процентом от общего числа объектов (по дефолту — 1);
- n_estimators — число деревьев в "лесу"(по дефолту – 100);

[24] *http://www.machinelearning.ru/wiki/index.php?title=Бэггинг*

[25] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

- criterion — функция, которая измеряет качество разбиения ветки дерева (по дефолту выбран критерий "gini" можно выбрать "entropy" ;
- max_features — число признаков, по которым ищется разбиение. Вы можете указать конкретное число или процент признаков, либо выбрать из доступных значений: "auto"(все признаки),"sqrt" , "log2" . По дефолту стоит "auto" ;
- max_depth — максимальная глубина дерева (по дефолту глубина не ограничена);
- min_weight_fraction_leaf — минимальная взвешенная доля от общей суммы весов (всех входных объектов) должна быть в листе (по дефолту имеют одинаковый вес);
- max_leaf_nodes — максимальное количество листьев (по дефолту нет ограничения - None);
- min_impurity_split — порог для остановки наращивания дерева (по дефолту 1e-7);
- bootstrap — применять ли бустрэп для построения дерева (по дефолту True);
- oob_score — использовать ли out-of-bag объекты для оценки R^2 (по дефолту False);
- warm_start — использует уже натренированую модель и добавляет деревьев в ансамбль (по дефолту False);
- class_weight — вес каждого класса (по дефолту все веса равны 1, но можно передать словарь с весами, либо явно указать "balanced тогда веса классов будут равны их исходным частям в генеральной совокупности; также можно указать "balanced_subsample тогда веса на каждой подвыборке будут меняться в зависимости от распределения классов на этой подвыборке.

Лес сверхслучайных деревьев является модификацией случайного леса [26].
Это метод ансамблевого обучения, который объединяет результаты нескольких не коррелированных деревьев решений, собранных в «лесу», для вывода результата классификации. По своей концепции он очень похож на классификатор случайных лесов и отличается от него только способом построения деревьев решений в лесу.

Используется класс ExtraTreesClassifier в библиотеке Scikit-learn с параметрами:

- random_state = 42;
- остальные параметры по умолчанию, как и у RandomForestClassifier, с которыми можно ознакомиться на сайте [27],

[26] https://www.pvsm.ru/python/250970
[27] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html

Метод опорных векторов.

Основная идея метода опорных векторов на Рис. 11 заключается в построении гиперплоскости, разделяющей объекты выборки оптимальным способом. Алгоритм работает в предположении, что чем больше расстояние (зазор) между разделяющей гиперплоскостью и объектами разделяемых классов, тем меньше будет средняя ошибка классификатора. Отличия между линейным и нелинейным методами опорных векторов в ядрах и подходах : "один против всех" и "один против одного"соответственно. Следовательно, делают разные предположения в отношении проблемы оптимизации, что приводит к разным характеристикам в одной и той же задаче.

1. Для реализации нелинейного классификатора опорных векторов используется класс SVC в библиотеке Scikit-learn с параметрами:
   - kernel = 'rbf' - тип ядра, который будет использоваться в классификаторе;
   - probability = True - логическое значение, определяющее следует ли на этапе обучения вычислять значения, которые в дальнейшем позволят оценить вероятности принадлежности нового объекта каждому из рассматриваемых классов (в случае классификации) и восстановить распределение ошибки предсказания в заданной точке;
   - C = 2 - параметр регуляризации;
   - class_weight = "balanced" - для параметра C «сбалансированный» режим для автоматической регулировки весов;
   - остальные параметры по умолчанию, с которыми можно ознакомиться на сайте [28].
2. Для реализации линейного классификатора опорных векторов используется класс LinerSVC в библиотеке Scikit-learn с параметрами:
   - C = 2;
   - class_weight = "balanced";
   - dual = False - выбор алгоритма для решения задачи двойной или первичной оптимизации;
   - остальные параметры по умолчанию, которые указаны на сайте [29]

Градиентый бустинг[30].

[28] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC
[29] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html
[30] https://lambda-it.ru/post/busting-s-pomoshchiu-adaboost-i-gradient-boosting

Бустинг [31] - это процедура последовательного построения композиции алгоритмов машинного обучения, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов.

Градиентый бустинг - это техника построения ансамблей, идея на Рис. 12 которой состоит в том, что следующая модель будет учиться на ошибках предыдущей, то есть модели построены не независимо, а последовательно. Они могут быть выбраны из широкого ассортимента моделей, например, деревья решений, регрессия и т.д. Из-за того, что модели обучаются на ошибках, совершенных предыдущими, требуется меньше времени для того, чтобы добраться до реального ответа.

Для реализации градиентого бустинга используется класс GradientBoostingClassifier в библиотеке Scikit-learn с параметрами по умолчанию, с которыми можно ознакомиться на сайте [32], например:

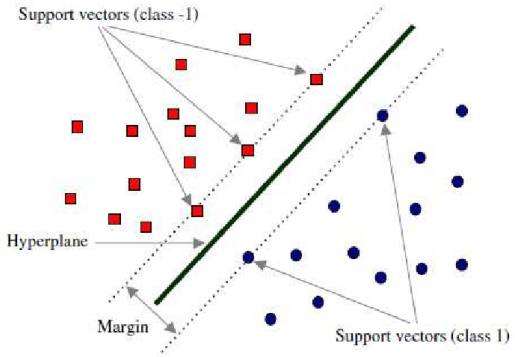

Рис. 11: Метод опорных веторов.

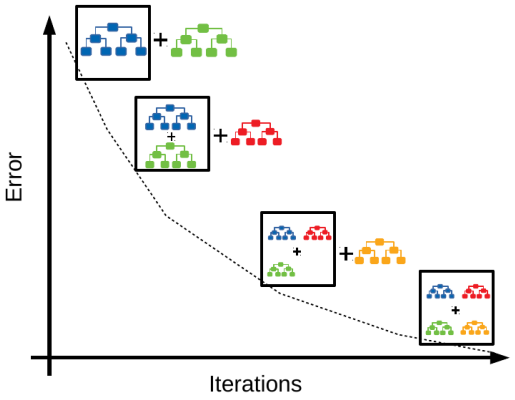

Рис. 12: Градиентный бустинг.

- min_samples_split = 2 - минимальное количество наблюдений, необходимое для разделения внутреннего узла;
- random_state = None ;
- min_samples_leaf = 1 - минимальное количество наблюдений в листовом узле;
- max_depth = 3 - максимальная глубина индивидуальных регрессионных оценок;
- verbose = 0 - выключен подробный вывод;

Стохастический градиентный спуск.

Градиентный спуск — метод нахождения локального минимума функции потерь с помощью движения вдоль градиента.

[31] http://www.machinelearning.ru/wiki/index.php?title=Бустинг

[32] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

Переобучение - это явление, когда построенная модель хорошо объясняет примеры из обучающей выборки, но относительно плохо работает на примерах, не участвовавших в обучении.

Подход к борьбе с переобучением градиентного бустинга, который заключается в том, что каждый базовый алгоритм обучается не на всей выборке, а на некоторой ее случайной подвыборке. Такой подход еще называется стохастическим градиентным спуском.

Для реализации градиентого бустинга используется класс SGDClassifier в библиотеке Scikit-learn с параметрами по умолчанию, с которыми можно ознакомиться на сайте[33], например:

- random_state = None;
- alpha = 0.0001 - гиперпараметр служит для двойной цели. Это и параметр регуляризации, и начальная скорость обучения по расписанию по умолчанию;
- loss = 'hinge' - функция потерь.

2.3.1 Оценка и улучшение качества модели.

Перекрестная проверка.

Перекрестная проверка представляет собой метод аналитической оценки модели и её поведения на данных. В перекрестной проверке данные разбиваются несколько раз и строится несколько моделей. Вариант перекрестной проверки, используемый в работе - k-блочная перекрестная-проверка, в которой k - это параметр. k = 10, то есть разбиение данных происходит в десятиблочной перекрестной проверке. Данные сначала разбиваются на 10 частей (примерно) одинакового размера, называемых блоками. Далее строится последовательность моделей. Первая модель обучается, используя блок 1 в качестве тестового набора, а остальные блоки (2-10) выполняют роль обучающего набора. Модель строится на основе данных, расположенных в блоках 2-10, а затем на данных блока 1 оценивается ее точность. Затем происходит обучение второй модели, в качестве тестового набора использует блок 2 , а обучающего набора - блоки 1, 3-10. Этот процесс повторяется для блоков 3-10. Для каждого из 10 разбиений (splits) данных на обучающих и тестовый наборы вычисляется правильность. В итоге фиксируются 10 значений правильности. Процесс показан на Рис. 13. Наиболее распространенный способ подытожить правильность, вычисленную в ходе перекрестной проверки, - это вычисление среднего значения.

Используется функция cross_val_score из модуля sklearn.model_selection с параметром:

- cv = 10, который обозначает количество блоков.

[33] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

и для вычисления вспомогательных метрик оценки качества классификации (2.4) вводится параметр scoring равным:

- 'accuracy' - коэффициент правильности (2.4);
- 'precision' - точность (2.4);
- 'neg_mean_absolute_error' - коэффициент ошибки (2.4);
- 'recall' - полнота (2.4);
- 'f1' - F_1-мера (2.4);

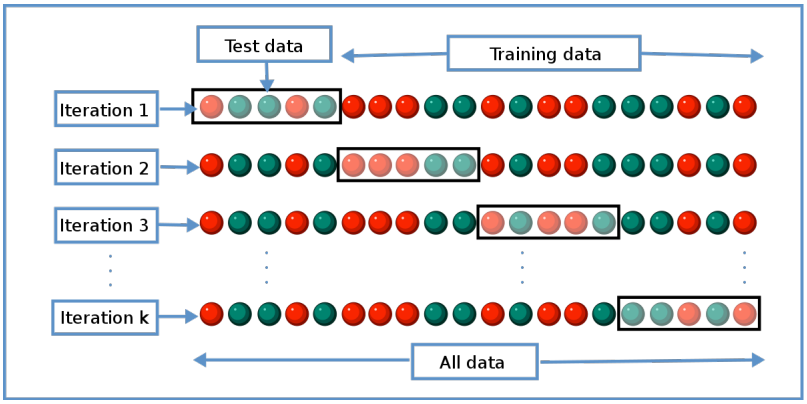

Рис. 13: Перекрестная проверка (при k=10).

2.4 Анализ и результаты.

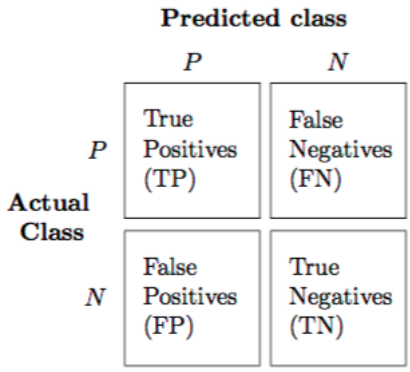

Рис. 14: Матрица ошибок.

Одним из наиболее развернутых способов, позволяющих оценить качество бинарной классификации, является использование матрицы ошибок. Матрица ошибок

имеет размер 2×2, строки которой соответствуют фактическим классам, а столбцы спрогнозированным классам. Матрица ошибок на Рис. 14 состоит из четырех значений:

Истинно-положительный (TP): правильный положительный прогноз;

Ложно-положительный (FP): неправильный положительный прогноз, то есть ошибка 1-го рода, когда модель предсказала положительный результат, а на самом деле он отрицательный;

Истинно-отрицательный (TN): правильный негативный прогноз

Ложно-отрицательный (FN): неправильный негативный прогноз, то есть ошибка 2-го рода, когда модель предсказала отрицательный результат, но на самом деле он положительный.

Оценить качество можно с помощью следующих метрик:

Коэффициент правильности. Коэффициент правильности (accuracy) рассчитывается как число всех правильных прогнозов, деленное на общее количество набора данных. Наилучшая точность равна 1, а наихудшая - 0.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Коэффициент ошибки. Коэффициент ошибок (error) рассчитывается как число всех неправильных прогнозов, деленное на общее количество набора данных. Наилучшая частота ошибок - 0, а наихудшая - 1.

$$error = \frac{FP + FN}{TP + TN + FP + FN}$$

Полнота. Полнота (recall) рассчитывается как количество правильных положительных прогнозов, деленное на общее количество положительных результатов. Наилучшая полнота равна 1, а наихудшая - 0.

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

Точность. Точность (precision) рассчитывается как количество правильных положительных прогнозов, деленное на общее количество положительных прогнозов. Наилучшая точность - 1, а наихудшая - 0.

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

F_1-мера. F_1-мера (в общем случае F_β) - это сочетание точности и полноты. Она дает некоторый компромисс между ними двумя.

β - в данном случае определяет вес точности в метрике, и при $\beta = 1$ это среднее гармоническое. Оценка F_1 достигает своего наилучшего значения в 1 и худшее в 0.

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

$$F_1 = \frac{2 \times precision \times recall}{precision + recall}$$

Метод №1	Accuracy	Error	Precision	Recall	F1-S
LinearSVC	0.77	0.23	0.79	0.74	0.76
SVC	0.77	0.23	0.79	0.74	0.76
DecisionTreeClassifier	0.76	0.24	0.79	0.76	0.77
RandomForestClassifier	**0.82**	**0.18**	0.82	0.84	**0.91**
ExtraTreesClassifier	**0.82**	**0.18**	0.84	0.84	0.83
GradientBoostingClassifier	**0.82**	**0.18**	**0.85**	0.82	0.82
SGDClassifier	0.74	0.26	0.77	**0.89**	0.82

Таблица 1: Метод «чистых» данных.

Метод №2	Accuracy	Error	Precision	Recall	F1-S
LinearSVC	**0.75**	**0.25**	0.76	0.71	0.74
SVC	0.60	0.40	0.59	0.61	0.60
DecisionTreeClassifier	0.66	0.34	0.69	0.66	0.67
RandomForestClassifier	**0.75**	**0.25**	**0.79**	0.72	0.75
ExtraTreesClassifier	0.74	**0.25**	0.78	0.71	0.74
GradientBoostingClassifier	**0.75**	**0.25**	0.77	0.82	**0.78**
SGDClassifier	0.64	0.36	0.74	**0.85**	0.73

Таблица 2: Метод ненормированной связи данных с весами взаимоотношений.

Приведенные таблицы суммируют результаты различных методов с данными. Видно, что при работе с таблицей отношений атрибутов клиента в методах №2, №3 и №4 происходит рост точности предсказания. Результаты выделенные жирным шрифтом, полученные методом № 4, превосходят в сравнении с результатами метода №1.

На Рис. 15 показано, как коррелируют атрибуты в методе расширенных данных. Причина положительной корреляции между добавочными атрибутами, в правом нижнем красном квадрате на Рис. 15, является умножение на матрицу смежности

Метод №3	Accuracy	Error	Precision	Recall	F1-S
LinearSVC	**0.76**	**0.24**	**0.78**	0.73	0.75
SVC	0.60	0.40	0.60	0.63	0.61
DecisionTreeClassifier	0.68	0.32	0.72	0.69	0.70
RandomForestClassifier	0.73	0.26	**0.78**	0.73	0.74
ExtraTreesClassifier	0.73	0.27	0.76	0.70	0.73
GradientBoostingClassifier	0.74	0.26	0.77	**0.80**	**0.77**
SGDClassifier	0.66	0.37	0.74	**0.80**	0.65

Таблица 3: Метод нормированной связи данных с весами взаимоотношений.

Метод № 4	Accuracy	Error	Precision	Recall	F1-S
LinearSVC	0.78	0.22	0.80	0.74	0.76
SVC	0.77	0.23	0.80	0.72	0.76
DecisionTreeClassifier	0.75	0.25	0.78	0.77	0.77
RandomForestClassifier	0.82	0.17	0.83	**0.87**	0.85
ExtraTreesClassifier	**0.87**	**0.14**	**0.88**	0.86	**0.87**
GradientBoostingClassifier	0.81	0.19	0.83	0.84	0.83
SGDClassifier	0.70	0.35	0.81	0.76	0.78

Таблица 4: Метод расширенных данных.

$\tilde{A}$ в формуле (1) и (2), так как при умножении происходит суммирование соседних узлов. В правом верхнем и левом нижнем углах основным цветом является светло-голубой, который указывает на то, что корреляция между исходными атрибутами и добавочными атрибутами равна нулю.

3 Заключение.

3.1 Вывод.

В данной работе была разработана модель по улучшению точности предсказания кредитного риска с помощью графо-структурированных связей атрибутов клиента. Опираясь на полученные результаты, можно сделать вывод что предложенная модель дает дополнительную информацию о клиенте, с помощью которой лучше предсказывает кредитный риск.

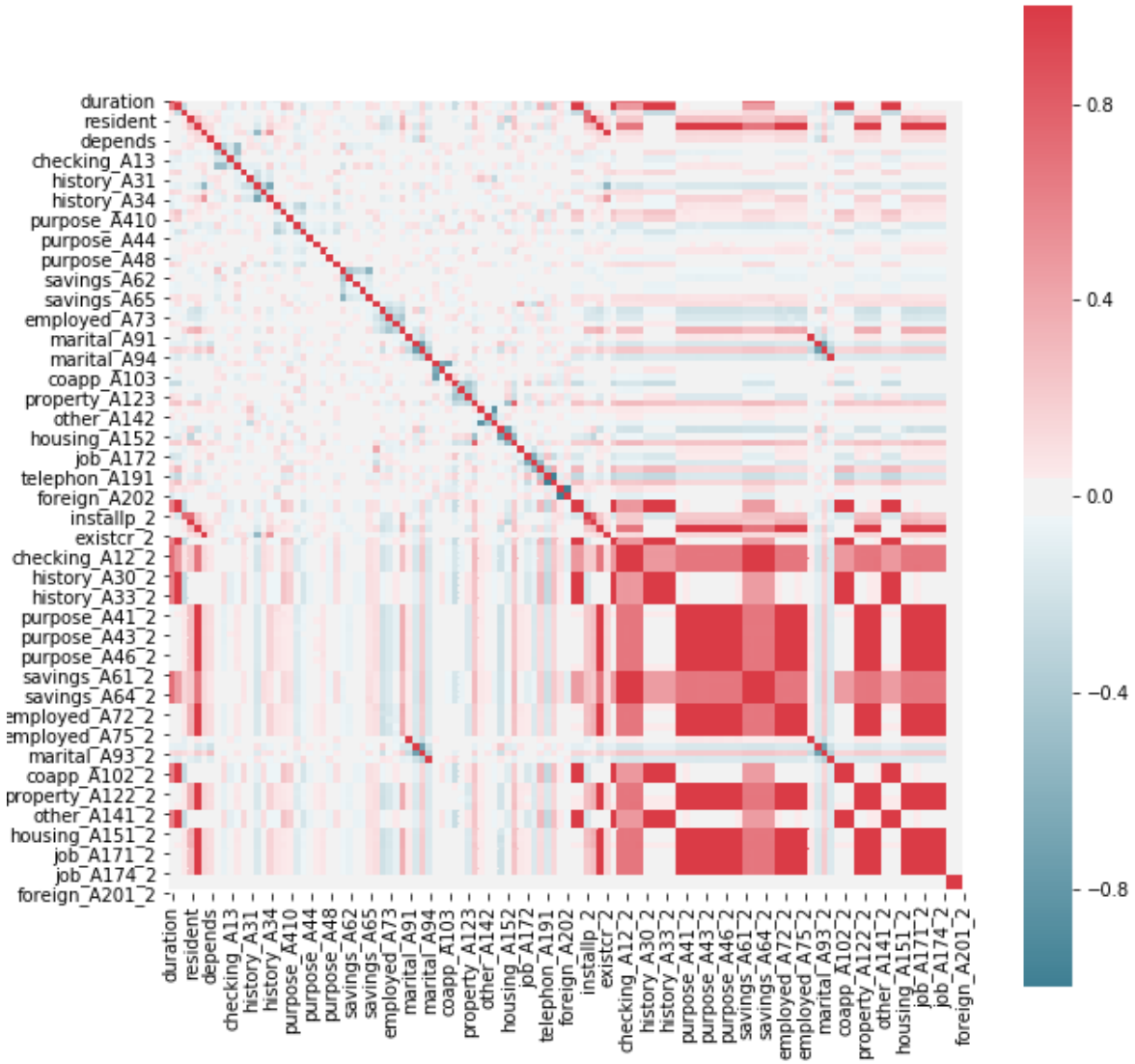

Рис. 15: Корреляция характеристик в методе расширения данных.

Список литературы

[1] Holt, Jeff. (2009). A Summary of the Primary Causes of the Housing Bubble and the Resulting Credit Crisis: A Non-Technical Paper. The Journal of Business Inquiry. 8. 120-129. and the Resulting Credit Crisis: A Non-Technical Paper.

[2] Dean Baker. The housing bubble and the financial crisis. 2008. Center for Economic and Policy Research, USA.

[3] Demyanyk, Y. S. and O. Van Hemert. 2008. “Understanding the Subprime Mortgage Crisis.” http://ssrn.com/abstract=1020396

[4] Durand, David. Risk Elements in Consumer Instalment Financing. Pp. xx, 163. New York: National Bureau of Economic Research, 1941. The ANNALS of the American Academy of Political and Social Science, 218(1), 237–237.

[5] Thomas, L. C. (2000). A survey of credit and behavioural scoring: forecasting financial isk of lending to consumers. International Journal of Forecasting, 16, 149–172.

[6] Qeethara Al-Shayea. Evaluating Credit Risk Using Artificial Neural Networks. September 2011 - www.researchgate.net

[7] Adnan Khashman. Neural networks for credit risk evaluation: Investigation of different neural models and learning schemes

[8] David West. Neural network credit scoring models. Computers & Operations Research

[9] Ahn, B. S., Cho, S. S., & Kim, C. Y. (2000). The integrated methodology of rough set theory and artificial neural network for business failure prediction. Expert Systems with Applications, 18(2), 65–74.

[10] Baesens, B., Gestel, T. V., Stepanova, M., Van den Poel, D., & Vanthienen, J. (2005). Neural network survival analysis for personal loan data. Journal of the Operational Research Society, 56(9), 1089–1098.

[11] Becerra-Fernandez, I., Zanakis, S. H., & Walczak, S. (2002). Knowledge discovery techniques for predicting country investment risk. Computers and Industrial Engineering, 43(4), 787–800.

[12] Hsieh, N. C. (2005). Hybrid mining approach in design of credit scoring model. Expert Systems with Applications, 28, 655–665.

[13] Huang, Z., Chen, H., Hsu, C. J., Chen, W. H., & Wu, S. (2004). Credit rating analysis with support vector machines and neural networks: A market comparative study. Decision Support Systems, 37(4), 543–558.

[14] Huang, W., Nakamori, Y., & Wang, S. Y. (2005). Forecasting stock market movement direction with support vector machine. Computers and Operations Research, 32(10), 2513–2522.

[15] Лукашевич Н.С. Сравнение нейросетевых и статистических методов оценки кредитного риска. Финансы и кредит. 2011. №1 (433). URL: https://cyberleninka.ru/article/n/sravnenie-neyrosetevyh-i-statisticheskih-metodov-otsenki-kreditnogo-riska

[16] Никулин Владимир Николаевич, Канищев Илья Сергеевич, Багаев Иван Владимирович. Методы балансировки и нормализации данных для улучшения качества классификации // КИО. 2016. №3. URL: https://cyberleninka.ru/article/n/metody-balansirovki-i-normalizatsii-dannyh-dlya-uluchsheniya-kachestva-klassifikatsii.

[17] Сергей Царьков. Различные стратегии сэмплинга в условиях несбалансированности классов. https://basegroup.ru/community/articles/imbalance-datasets

[18] Aithal, V., & Jathanna, R. D. (2019). Credit risk assessment using machine learning techniques. International Journal of Innovative Technology and Exploring Engineering, 9(1), 3482-3486. https://doi.org/10.35940/ijitee.A4936.119119

[19] Thanawala, Dhruv Dhanesh, "CREDIT RISK ANALYSIS USING MACHINE LEARNING AND NEURAL NETWORKS Open Access Master's Report, Michigan Technological University, 2019. https://digitalcommons.mtu.edu/etdr/856

[20] Huang, C. L., Chen, M. C., & Wang, C. J. (2007). Credit scoring with a data mining approach based on support vector machines. Expert Systems with Applications, 33(4), 847–856.

[21] Andrea C. Muller & Sarah Guido.(2017). Introduction to Machine Learning with Python. A Guide for Data Scientists, 392.

4 Приложение.

4.1 Код метода расширенных данных.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from collections import defaultdict
from sklearn.preprocessing import LabelEncoder
from numpy import asarray
import seaborn as sns
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import SMOTE
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from sklearn import tree, svm
from sklearn.ensemble import GradientBoostingClassifier,
from sklearn.ensemble import RandomForestClassifier, ExtraTreesClassifier
from sklearn.linear_model import SGDClassifier
import math
```

```
names = ['checking','duration','history', 'purpose',
'amount','savings','employed','installp',
 'marital', 'coapp', 'resident', 'property', 'age',
'other', 'housing', 'existcr', 'job',
 'depends', 'telephon', 'foreign', 'good_bad' ]
data = pd.read_csv('GermanData.csv', header = None, names = names )
df = data.drop('good_bad', axis = 1) #входные данные
from collections import Counter
print('shape of data =',data.shape)
target = data.values[:,-1]
#print(target)
#-----------------------------------------------------------------
counter = Counter(target)
for k,v in counter.items():
    per = v/len(target) * 100
    print('Class=%d, Count=%d, Percentage=%.3f%%' % (k, v, per))
#-----------------------------------------------------------------
print('\n')
#from matplotlib import pyplot
num_ix = df.select_dtypes(include=['int64', 'float64']).columns
print('количественные признаки: ', num_ix)

#сделаем замену выходных значений 1-Good и 2-Bad на 1-good и 0
data.good_bad.replace([1,2], [1,0], inplace=True)
print('\n')
#создадим список быстрого доступа с метками категориальных переменных
catvars = ['checking', 'history', 'purpose', 'savings',
 'employed', 'marital', 'coapp', 'property',
  'other', 'housing', 'job','telephon', 'foreign']
#создадим список быстрого доступа с метками числовых переменных
numvars = ['duration', 'amount',
'installp', 'resident', 'age',
'existcr','depends', 'good_bad']
print('количество недостающих значений: ', data.isnull().sum().sum())
#количество недостающих значений = 0
#диапозон значений числовых переменных
print('диапозон значений числовых переменных')
for x in range(len(numvars)):
    print(numvars[x],": ", data[numvars[x]].min()," - ",data[numvars[x]].max())
print('\n')
# Кодирование переменной
d = defaultdict(LabelEncoder)
lecatdata = data[catvars].apply(lambda x: d[x.name].fit_transform(x))
# трансформация:
for x in range(len(catvars)):
    print(catvars[x],": ", data[catvars[x]].unique())
    print(catvars[x],": ", lecatdata[catvars[x]].unique())
print('\n')
#One hot encoding
#создаем фиктивные переменные для каждой категории каждой категориальной переменной
```

```
dummyvars = pd.get_dummies(data[catvars])
print(dummyvars)
print('\n входные значения: \n')
ndata = pd.concat([data[numvars].drop(['good_bad'], axis=1), dummyvars], axis = 1)
print(ndata)
print('\n')
#граф связи:
np.set_printoptions(linewidth=120, threshold=np.inf)
col = ['duration','amount','installp','resident', 'age',
'existcr', 'depends','checking_A11','checking_A12',
'checking_A13','checking_A14','history_A30','history_A31',
'history_A32','history_A33','history_A34','purpose_A40',
'purpose_A41','purpose_A410','purpose_A42','purpose_A43',
'purpose_A44','purpose_A45','purpose_A46','purpose_A48',
'purpose_A49','savings_A61','savings_A62','savings_A63',
'savings_A64','savings_A65','employed_A71','employed_A72',
'employed_A73','employed_A74','employed_A75','marital_A91',
'marital_A92','marital_A93','marital_A94','coapp_A101','coapp_A102',
'coapp_A103','property_A121','property_A122','property_A123',
'property_A124','other_A141','other_A142','other_A143',
'housing_A151','housing_A152','housing_A153','job_A171','job_A172',
'job_A173','job_A174','telephon_A191','telephon_A192',
'foreign_A201','foreign_A202' ]
gdata = pd.read_csv('rash_tab.csv', names = col)
#rash_tab.csv - расширенная таблица
print('\n таблица отношений: \n')
print(gdata)
print('\n')
#BECA
#превращаем из файла rash_tab.csv в массив (таблицу отношений)
A= asarray(gdata)
#print(A.shape)
#print(A)
#D - матрица степени графа
count = asarray(gdata)
for i in range(0,61):
    for j in range(0,61):
        if A[i][j] >0.0:
            count[i][j] = 1.0
#print(count)

X = asarray(ndata)
D = np.array(np.sum(count, axis=0))
D = D**-1
D = np.matrix(np.diag(D))
#print(D)
D_h = D*A #D^{-1}A
#входные данные клиента под номером 1
ans_h = np.dot(D_h,X[0])
for i in range(1,1000):
```

```
    C = np.dot(D_h,X[i])
    ans_h = np.vstack((ans_h,C))
print(ans_h.shape)
#расширение данных:
ans = np.hstack((ndata,ans_h))
col_2 = ['duration','amount','installp','resident','age','existcr',
'depends','checking_A11','checking_A12','checking_A13',
'checking_A14','history_A30','history_A31','history_A32',
'history_A33','history_A34','purpose_A40','purpose_A41',
'purpose_A410','purpose_A42','purpose_A43','purpose_A44',
'purpose_A45','purpose_A46','purpose_A48','purpose_A49',
'savings_A61','savings_A62','savings_A63','savings_A64',
'savings_A65','employed_A71','employed_A72','employed_A73',
'employed_A74','employed_A75','marital_A91','marital_A92',
'marital_A93','marital_A94','coapp_A101','coapp_A102',
'coapp_A103','property_A121','property_A122','property_A123',
'property_A124','other_A141','other_A142','other_A143',
'housing_A151','housing_A152','housing_A153','job_A171',
'job_A172','job_A173','job_A174','telephon_A191',
'telephon_A192','foreign_A201', 'foreign_A202' ,
'duration_2','amount_2','installp_2','resident_2',
'age_2','existcr_2','depends_2','checking_A11_2',
'checking_A12_2','checking_A13_2','checking_A14_2',
'history_A30_2','history_A31_2','history_A32_2','history_A33_2',
'history_A34_2','purpose_A40_2','purpose_A41_2','purpose_A410_2',
'purpose_A42_2','purpose_A43_2','purpose_A44_2','purpose_A45_2',
'purpose_A46_2','purpose_A48_2','purpose_A49_2','savings_A61_2',
'savings_A62_2','savings_A63_2','savings_A64_2','savings_A65_2',
'employed_A71_2','employed_A72_2','employed_A73_2','employed_A74_2',
'employed_A75_2','marital_A91_2','marital_A92_2','marital_A93_2',
'marital_A94_2','coapp_A101_2','coapp_A102_2','coapp_A103_2',
'property_A121_2','property_A122_2','property_A123_2',
'property_A124_2','other_A141_2','other_A142_2','other_A143_2',
'housing_A151_2','housing_A152_2','housing_A153_2','job_A171_2',
'job_A172_2','job_A173_2','job_A174_2',
'telephon_A191_2','telephon_A192_2','foreign_A201_2','foreign_A202_2' ]
df=pd.DataFrame(data=ans[0:,0:], index=[i for i in range(data.shape[0])],
 columns=[i for i in col_2])
#корреляция:
f, ax = plt.subplots(figsize=(122, 122))
corr = df.corr()
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool),
cmap=sns.diverging_palette(220, 10, as_cmap=True),square=True, ax=ax)
plt.savefig('testplot.png')
#print(ans.shape)
numdata_std = pd.DataFrame(StandardScaler().fit_transform(ans))
print('numdata_std:\n',numdata_std)
numdata_minmax = pd.DataFrame(MinMaxScaler().fit_transform(ans))
print('numdata_minmax:\n', numdata_minmax)
print('\n')
```

```
X_minmax = numdata_minmax
y_minmax = data['good_bad']
X_train_minmax, X_test_minmax, y_train_minmax,
y_test_minmax = train_test_split(X_minmax,y_minmax,test_size=0.40, random_state=1)

X_std = numdata_std
y_std = data['good_bad']
X_train_std, X_test_std, y_train_std,
y_test_std = train_test_split(X_std,y_std,test_size=0.40, random_state=1)

X_clean = ans
y_clean = data['good_bad']
X_train_clean, X_test_clean, y_train_clean,
y_test_clean = train_test_split(X_clean,y_clean,test_size=0.40, random_state=1)

sm = SMOTE()
X_train_clean_res, y_train_clean_res = sm.fit_sample(X_train_clean, y_train_clean)
X_train_std_res, y_train_std_res = sm.fit_sample(X_train_std, y_train_std)
X_train_minmax_res, y_train_minmax_res = sm.fit_sample(X_train_minmax, y_train_minmax)

print("Before/After clean")
unique, counts = np.unique(y_train_clean, return_counts=True)
print(dict(zip(unique, counts)))
unique, counts = np.unique(y_train_clean_res, return_counts=True)
print(dict(zip(unique, counts)))
print("Before/After std")
unique, counts = np.unique(y_train_std, return_counts=True)
print(dict(zip(unique, counts)))
unique, counts = np.unique(y_train_std_res, return_counts=True)
print(dict(zip(unique, counts)))
print("Before/After minmax")
unique, counts = np.unique(y_train_minmax, return_counts=True)
print(dict(zip(unique, counts)))
unique, counts = np.unique(y_train_minmax_res, return_counts=True)
print(dict(zip(unique, counts)))


def get_eval(clf, X_train, y_train,y_test,y_pred):
    # Cross Validation to test and anticipate overfitting problem
    scores1 = cross_val_score(clf, X_train, y_train, cv=10, scoring='accuracy')
    scores2 = cross_val_score(clf, X_train, y_train, cv=10, scoring='precision')
    scores3 = cross_val_score(clf, X_train, y_train, cv=10, scoring='roc_auc')
    scores4 = cross_val_score(clf, X_train, y_train, cv=10,
     scoring='neg_mean_absolute_error')
    scores5 = cross_val_score(clf, X_train, y_train, cv=10, scoring='recall')
    scores6 = cross_val_score(clf, X_train, y_train, cv=10, scoring='f1')
    # The mean score and standard deviation of the score estimate
    print("Accuracy: %0.2f (+/- %0.2f)" % (scores1.mean(), scores1.std()))
    print("Error: %0.2f (+/- %0.2f)" % (abs(scores4.mean()), scores4.std()))
    print("Precision: %0.2f (+/- %0.2f)" % (scores2.mean(), scores2.std()))
```

```
    print("Recall: %0.2f (+/- %0.2f)" % (scores5.mean(), scores5.std()))
    print("F1-S: %0.2f (+/- %0.2f)" % (scores6.mean(), scores6.std()))
    print("roc_auc: %0.2f (+/- %0.2f)" % (scores3.mean(), scores3.std()))
    # Create and print confusion matrix# Create and print confusion matrix
    abclf_cm = confusion_matrix(y_test,y_pred)
    print(abclf_cm)
    return

def dectreeclf(X_train, y_train,X_test, y_test):
    print("DecisionTreeClassifier")
    dec_tree = tree.DecisionTreeClassifier(min_samples_split=10,
    min_samples_leaf=5).fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = dec_tree.predict(X_test)

    # Export visualization as file
    #visualize_tree(dec_tree,X_train,y_train)

    # Get Cross Validation and Confusion matrix
    get_eval(dec_tree, X_train, y_train,y_test,y_pred)
    return

def linearsvmclf(X_train, y_train, X_test, y_test, C):
    print("LinearSVC")
    lin_svc = svm.LinearSVC(C=C, class_weight="balanced",
    dual=False).fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = lin_svc.predict(X_test)

    # Get Cross Validation and Confusion matrix
    get_eval(lin_svc, X_train, y_train,y_test,y_pred)
    return

def svmclf(X_train, y_train, X_test, y_test, C):
    print("SVC")
    #C = 0.7  # SVM regularization parameter
    svc = svm.SVC(kernel='rbf',probability=True,
    C=C,class_weight="balanced").fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = svc.predict(X_test)

    # Get Cross Validation and Confusion matrix
    get_eval(svc, X_train, y_train,y_test,y_pred)
    return

def gradientboostingclf(X_train, y_train, X_test, y_test):
    print("GradientBoostingClassifier")
```

```
    gbclf = GradientBoostingClassifier().fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = gbclf.predict(X_test)

    # Get Cross Validation and Confusion matrix
    get_eval(gbclf, X_train, y_train,y_test,y_pred)
    return

def randomforestclf(X_train, y_train,X_test, y_test):
    print("RandomForestClassifier")
    randomforest = RandomForestClassifier().fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = randomforest.predict(X_test)

    # Get Cross Validation and Confusion matrix
    get_eval(randomforest, X_train, y_train,y_test,y_pred)
    return

def sgdclf(X_train, y_train,X_test, y_test):
    print("SGDClassifier")
    sgd = SGDClassifier().fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = sgd.predict(X_test)

    # Get Cross Validation and Confusion matrix
    get_eval(sgd, X_train, y_train,y_test,y_pred)
    return

def extratreesclf(X_train, y_train,X_test, y_test):
    print("ExtraTreesClassifier")
    extratrees = ExtraTreesClassifier().fit(X_train, y_train)

    # Predict target variables y for test data
    y_pred = extratrees.predict(X_test)

    # Get Cross Validation and Confusion matrix
    get_eval(extratrees, X_train, y_train,y_test,y_pred)
    return


dectreeclf(X_train_clean_res, y_train_clean_res,X_test_clean, y_test_clean)

linearsvmclf(X_train_std_res, y_train_std_res,X_test_std, y_test_std,2)

svmclf(X_train_minmax_res,y_train_minmax_res,X_test_minmax,y_test_minmax,2)

gradientboostingclf(X_train_clean_res, y_train_clean_res,
```

```
X_test_clean, y_test_clean)

randomforestclf(X_train_clean_res, y_train_clean_res,
X_test_clean, y_test_clean)

extratreesclf(X_train_clean_res, y_train_clean_res,
X_test_clean, y_test_clean)

sgdclf(X_train_minmax, y_train_minmax, X_test_minmax, y_test_minmax)
```