1.2 Базовые понятия машинного обучения

Прежде чем говорить о конкретных моделях, опишем задачу прогнозирования успеха стартапа с точки зрения машинного обучения.

В результате исследования мы желаем научиться предсказывать успех стартапа на основе доступной нам о нем информации. Таким образом, речь идет о проблеме бинарной классификации – для каждой компании у нас есть два варианта: она или скорей будет успешной, или не будет.

Наши данные содержат разноплановую информацию о компаниях, в том числе и ту, которая отвечает за выполнение условий успеха – финансирование серии B, первичное размещение и поглощение. Значит, решаемая задача относится к обучению с учителем, поскольку мы сможем показать алгоритму примеры правильной классификации, которые он должен будет научиться обобщать.

Поскольку классификатор, который мы получим, будет принимать данные конкретной компании на вход и давать 1 (успех) или 0 (неуспех) на выход, то его можно рассматривать как функцию, действующую из n-мерного евклидова пространства в множество, включающее 0 и 1. n равно количеству признаков (фич), на которых будет учиться алгоритм.

Заметим, что классификатор, являясь функцией, будет лишь оценкой истинной функции, которая нас интересует. Это связано с тем, что обучение алгоритма будет происходить на выборке, хоть и достаточно большой, а не на всей популяции.

Так как мы хотим получить не просто классификатор, а хороший классификатор, нам необходимы метрики, которые дадут возможность сравнивать модели между собой и выбирать наиболее подходящую.

Рассмотрим основные метрики.

Доля правильно предсказанных объектов (accuracy) – это отношение правильно предсказанных объектов ко всем объектам, в отношении которых осуществлялось предсказание. Эта метрика кажется разумной, однако часто может вводить в заблуждение, если используется вне связки с другими. Так,

если классы сильно несбалансированны (как в настоящем исследовании, где доля успешных компаний меньше 10%), то классификатор, выдающий 0 для всех компаний, то есть обычная константа, будет иметь долю правильно предсказанных значений выше 90%.

Следующая метрика это точность (precision). Точность определяется как отношение правильно предсказанных 1 (успешных компаний) к объектам, которые были предсказаны как 1. Точность можно понимать как условную вероятность того, что объект является 1, если известно, что предсказана 1 (успех).

Полнота (recall), или чувствительность (sensitivity), это отношение правильно предсказанных 1 ко всем 1 (успешным компаниям) в выборке. То есть это условная вероятность предсказать 1 в отношении объекта, являющегося 1 (успешной компанией).

F1-мера является гармоническим средним метрики и полноты и позволяет учитывать показатели обеих метрик сразу.

Также используется такая метрика как специфичность (specificity). Она является зеркальным отражением полноты и представляет условную вероятность верной классификации объекта, если он является 0.

Стоит упомянуть и ROC-кривую. Она представляет собой график, на одной оси которого полнота, на другой вероятность неверной классификации объекта (FP rate), если он является 0, что равно (1-специфичность). В бинарной классификации алгоритм может использовать разные пороги для классификации объекта как 0 и 1. Например, если порог 0.3, то классифицируем все, что больше 0.3, как 1. А если 0.7, то все, что больше 0.7, как 1. Таким образом, разные пороги будут давать разные метрики, в этом и заключается идея ROC-кривой. Она позволяет посмотреть, какие комбинации полноты и (1-специфичность) способен достигать алгоритм.

Чем ближе точка к вершине (1;0), тем лучше. Поскольку модель характеризуется не одной точкой, а их совокупностью при разных порогах, то если ROC-кривая модели одной модели лежит выше кривой другой

модели, можно заключить, что первая модель однозначно лучше другой. Обычно интерпретация не настолько простая, потому что одна модель может себя прекрасно показывать, если нам нужна высокая чувствительность, в то время как другая может доминировать на области с низким FP уровнем.

Наконец, применяется метрика AUC ROC – площадь под ROC-кривой. Она позволяет посчитать, насколько в целом хороша модель, не учитывая, что для нас наибольший вес могут иметь весьма конкретные участки на ROC-кривой. Идеальная модель имеет площадь, равную единице.

Приведем изложенное выше воедино:

$$\text{Accuracy} = \frac{TP + TN}{total}$$

$$\text{True Positive Rate (TPR)} = \frac{TP}{TP + FN}$$

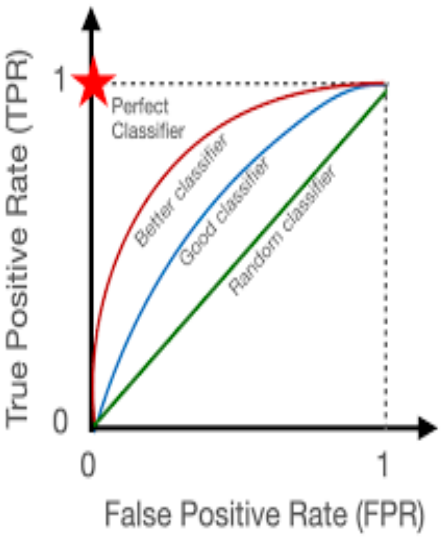
$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{AUC Score} = \frac{\text{Area under ROC Curve}}{\text{Total Area}}$$



В настоящем исследовании мы будем полагаться прежде всего на F1, поскольку эта метрика является достаточно универсальной и подходит для бинарной классификации несбалансированных классов.

Перед тем, как перейти к описанию моделей, которые будут использованы в настоящей работе, отметим, что каждая модель характеризуется структурой, функцией потерь и методом оптимизации функции потерь. Например, мы можем иметь линейную модель с сигмоидной активацией, логистической функцией ошибки и градиентным спуском соответственно, что соответствует модели логистической регрессии. Изменив функцию потерь или метод оптимизации, мы получаем уже несколько иную модель.

Поскольку характер данной работы скорей прикладной, нежели теоретический, мы не будем в точности описывать технические составляющие алгоритмов, а ограничимся общей идеей, стоящей за ними.

1.3 Используемые модели

Первой рассматриваемой моделью будет логистическая регрессия. Она представляет собой линейную модель, к которой применяется сигмоида.

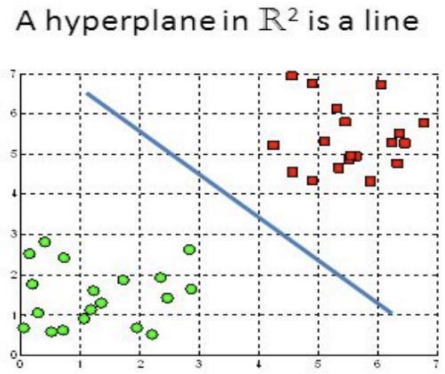
Линейность предполагает, что каждый признак получает определенный вес, а затем все признаки суммируются. Чем больше абсолютное значение веса отдельного параметра, тем больше он влияет на результат.

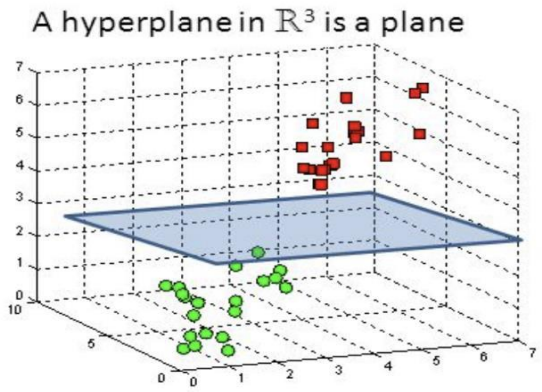
Сигмоида является сглаживающей функцией, действующей из множества действительных чисел в [0,1]. После применения сигмоиды получившееся значение можно интерпретировать как вероятность успеха или неуспеха отдельной компании.

Итак, модель простая по структуре, но несмотря на это она часто показывает хорошие результаты при решении разнообразных задач, а потому и сейчас активно используется.

Следующая рассматриваемая модель это метод опорных векторов. Ее суть в том, чтобы найти уравнение разделяющей гиперплоскости. То есть мы имеем n-мерное пространство и хотим найти n-1-мерное подпространство.

При этом предпочтительна такая гиперплоскость, чтобы объекты из разных классов находились по разные стороны от нее на максимальном расстоянии. Проиллюстрируем идею метода на примере двухмерного и трехмерного пространств:





Еще одним методом является алгоритм случайного леса. Он относится к методам решающих деревьев, при котором на основе одной выборки строится множество деревьев. Для того, чтобы они различались, вносится элемент случайности – каждое дерево строится не на всей совокупности наблюдений и с использованием не всей совокупности признаков. В результате имеется множество деревьев-классификаторов, мнение каждого из которых учитывается для формирования окончательно прогноза.

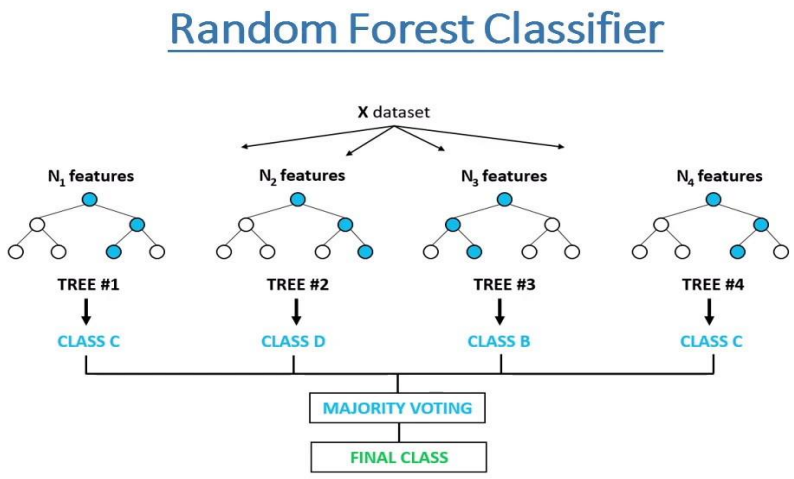
Напомним, что само решающее дерево мы можем понимать как граф, вершины которого содержат логические тесты, цель которых максимально уменьшать энтропию.

Достоинство решающих деревьев в том, что они являются сложным алгоритмом, способным за счет увеличения количества вершин давать все лучшее и лучшее приближение искомой функции.

В этом одновременно и их недостаток – из-за способности расти в глубину и ширину вероятно переобучение, когда дерево начинает находить в выборке закономерности, которые в действительности отсутствуют в популяции. Эту проблему можно охарактеризовать как наличие у оценки небольшого смещения, но высокой дисперсии. Так как случайный лес

агрегирует множество решающих деревьев, а не одно, то происходит уменьшение дисперсии и склонности к переобучению.

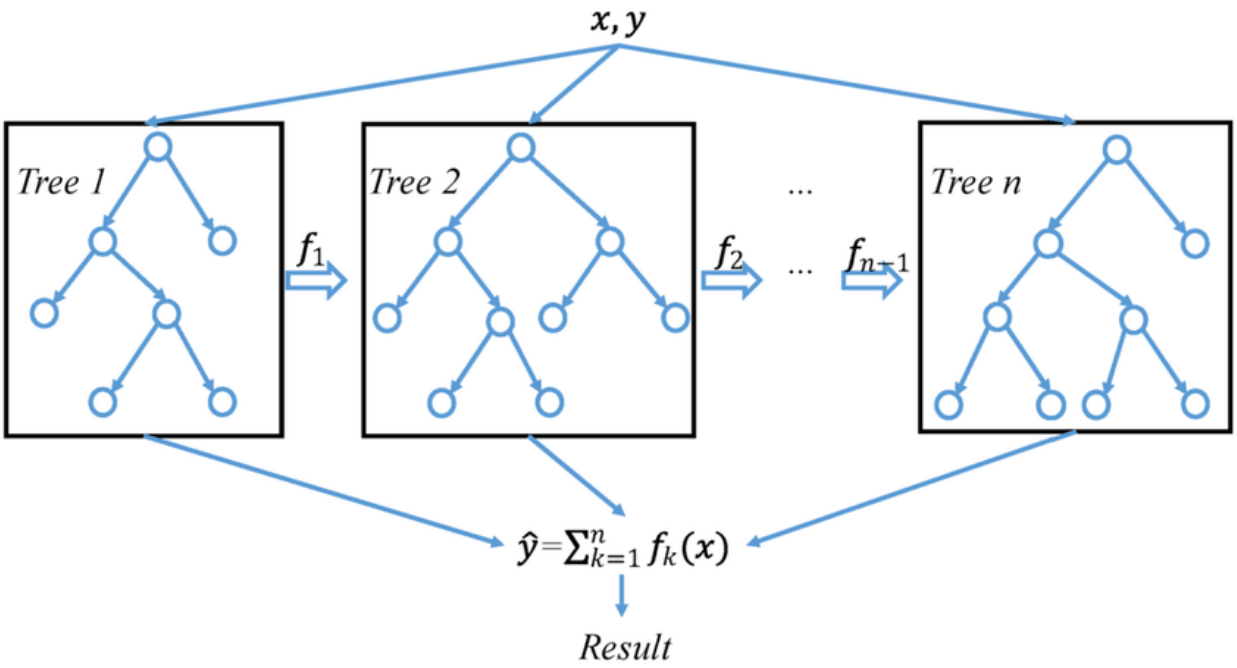
Приведем иллюстрацию действия случайного леса:



Мы будем использовать и алгоритм Extreme gradient boosting. Это один из доминирующих алгоритмов машинного обучения на настоящее время. Он строится на основе последовательного добавления в итоговую модель слабых подмоделей, где на каждой итерации минимизируется функция потерь посредством нахождения антиградиента в отношении текущей комбинации подмоделей [Mason et al., 1999]. В качестве слабых подмоделей применяются решающие деревья.

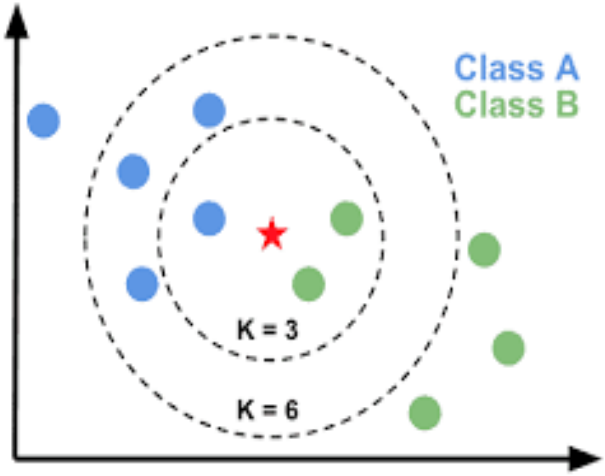
К недостаткам обычно относят слабую интерпретируемость полученного результата и потенциально высокие вычислительные затраты.

Проиллюстрируем принцип действия XGBoost:



Наконец, уделим внимание алгоритму k-ближайших соседей. Он опирается на предпосылку, что схожие объекты находятся рядом друг с другом. Так как каждая компания описывается вектором в n-мерном пространстве, то в отношении интересующей компании мы можем найти k ближайших компаний, метки которых нам известны, и путем определения преобладающей метки принять решение относительно неизвестной компании. Близость можно оценивать по-разному, стандартной метрикой для нее является Евклидова, или метрика Минковского с p=2.

Проиллюстрируем принцип действия kNN:



Глава 2. Практическая часть

2.1 Описание методологии

В настоящей работе для решения проблемы бинарной классификации будут использоваться модели:

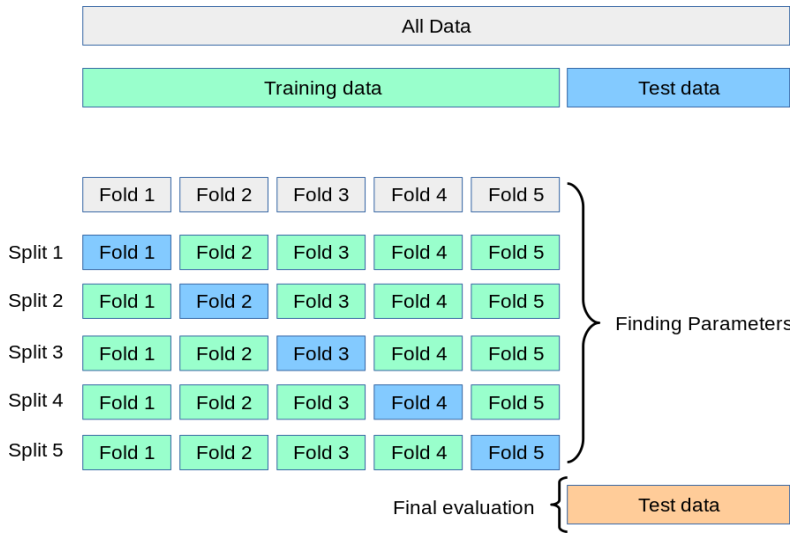
- Логистическая регрессия (Logistic regression)
- Метод опорных векторов (SVM)
- Метод K ближайших соседей (kNN)
- Случайный лес (Random Forest)
- eXtreme Gradient Boosting (XGB)

Для первых четырех моделей будем использовать имплементации из библиотеки scikit-learn, в качестве библиотеки для XGB будем использовать одноименную библиотеку xgboost.

Для каждой модели мы будем находить оптимальные гиперпараметры и после оценивать ее параметры на обучающей выборке, затем измерять финальное качество на тестовой, чтобы иметь возможность сравнить модели и определить, какая из них дает самые лучшие результаты качества предсказаний для нашей задачи. Сначала исходный массив (матрица, в строках которой - информация об отдельной компании, в столбцах - признаки и метка) разбиваем на обучающую и тестовую части в пропорции 3:1 (Тестовая часть составляет 25% исходной выборки). Затем на обучающей части с помощью кроссвалидации на 5 фолдах мы подберем для каждой модели оптимальные параметры, после чего построим предсказания для объектов из тестовой выборки и получим интересные нам метрики.

Отметим, что кроссвалидация позволяет справляться с проблемой переобучения при поиске оптимальных гиперпараметров, поскольку в процессе сами части обучающей выборки начинаются играть роль тестовых.

Схема валидации моделей представлена ниже:



2.2 Предобработка данных

Особенностью исследовательской работы заключается в использовании данных с платформы Crunchbase. Она аккумулирует большие данные по публичным и частным компаниям по всему миру, существует больше десяти лет и пользуется популярностью у исследователей. Также она предоставляет бесплатный доступ к данным для проведения исследований.

Исходные данные, которые были получены, представлены в виде 17 таблиц. В формате csv. Первый этап предварительной обработки данных заключался в исследовании зависимостей и консолидации данных. Мы выявили, что основные признаки, которые могут быть наиболее полезными – в таблицах organizations (основная), acquisitions, funding_rounds, ipos. Так, к основной таблице organizations мы присоединяли остальные, в процессе убирая нерелевантные признаки и проводя необходимые трансформации. К примеру, для компаний у которых было несколько сделок поглощений, выходов на IPO, B серий, оставляем только первые по времени.

Поскольку мы работаем с изначально не очищенными данными, они требуют аккуратной технической обработки. После объединения таблиц отметим следующие особенности в данных:

Достаточно высокая доля пропущенных значений для многих признаков. При этом важно отметить, что, учитывая особенности

объединения таблиц, некоторые признаки действительно предполагают наличие большого количества пропущенных значений и фактическими пропусками не являются. Другие же признаки действительно имеют множество фактических пропусков, поэтому такие наблюдения из выборки будут удалены.

Также характерно небольшое количество числовых переменных, относительно много категориальных. Работая с моделями машинного обучения, мы предпочитаем признаки в числовой формате, поэтому будем осуществлять трансформации из категориальных в числовые.

Подчеркнем, что есть данные в формате даты. Мы будем трансформировать абсолютные временные значения в относительные, так как иначе может возникать смещение в оценке.

Важным аспектом нашего исследования является вопрос того, что считать успехом. Как было замечено ранее, мы определяем успешность компании как выполнение хотя бы одного из условий ниже:

- компания вышла на IPO (первичное публичное предложение)
- компания вышла на финансирование B серии
- компанию была поглощена

Можно рассматривать данные прокси успеха по отдельности, но на данный момент ограничимся предложенным комбинированным вариантом как наиболее общим и находившим применение в ряде более ранних исследований.

Продемонстрируем использованные трансформации категориальных переменных:

- Country_code – страна, в которой зарегистрирована компания. Изначально представляет собой список стран – то есть категориальную переменную с большим количеством уникальных значений. После обработки мы получили бинарную переменную, принимающую значение 1, если страна входит в топ-5 по популярности (по количеству зарегистрированных компаний) и 0 в обратном случае.

- Category_group_list – категория компании (отрасль, вид деятельности). Изначально представлена в виде списка всех областей деятельности, в которых задействована компания. Создаем переменную is_tech: 1 для IT-стартапов, 0 для всех остальных. К IT относим сферы: Internet Services, Information Technology, Software, Hardware, Mobile, Messaging and Telecommunications, Data and Analytics, Biotechnology, Artificial Intelligence, Gaming, Payments, Apps.
- Employee_count – количество сотрудников. Изначально представлена как категориальная в виде интервалов (1-10, 10-50 и далее). Задаем для каждого интервала порядковую переменную, которая равна 1 для первого интервала и увеличивается на 1 для каждого следующего.

Для данных, которые представлены в формате дат, предлагаем следующую последовательность для обработки значений:
-Перевести из даты (string) в год (num)
-Выделить только те наблюдения, которые входят в заданный временной промежуток, для которого проводится исследование.
-Перейти к относительным значениям для тех признаков, где это имеет смысл: например, фиксирование возраста компании на момент B серии, на момент IPO, на момент поглощения.

После проведения полной обработки данных получаем количество наблюдений в итоговом датасете порядка 400 тысяч (в изначальном датасете после объединения таблиц – свыше 1,5 миллиона наблюдений). Далее выборка была разделена в пропорции 3:1 на обучающую и тестовую.

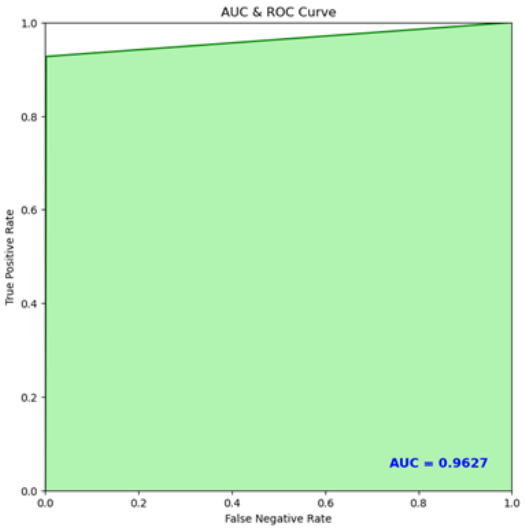
2.3 Сравнение моделей

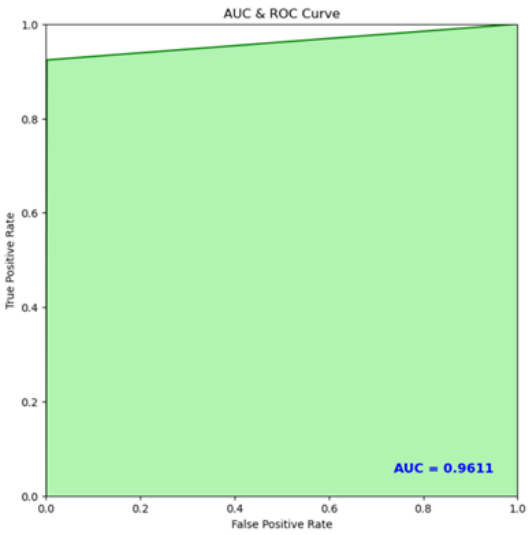
Начнем с рассмотрения алгоритма k-ближайших соседей.

Основным гиперпараметром является k – количество ближайших соседей, также можно придавать или не придавать больший вес мнению тех соседей, которые находятся ближе, чем другие участвующие в определении метки соседи. Соответственно, гиперпараметр weights=uniform или weights=distance. С помощью GridSearchCV мы подобрали на обучающей выборке k = 7 для модели без весов и k = 6 для модели с весами. После проверки качества модели на тестовой выборке получаем более высокие метрики для модели без весов:

	accuracy	F1	ROC-AUC
weights = 'uniform', k = 7	0.9939	0.9489	0.9627
weights = 'distance', k = 6	0.9937	0.9467	0.9611

Таким образом, имеет смысл остановиться на модели без весов, для неё график ROC выглядит следующим образом (слева), для модели с весами график представлен справа:

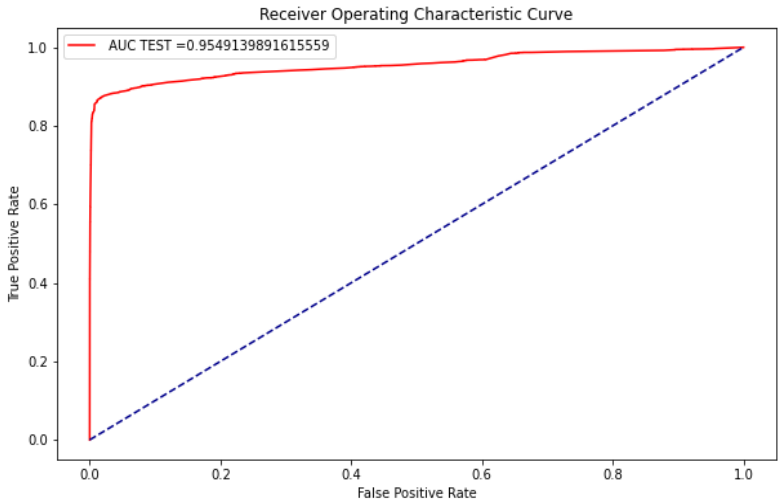




Перейдем к модели логистической регрессии

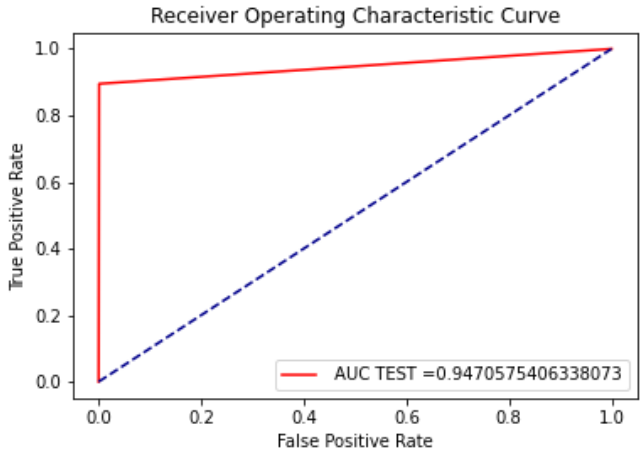
Была использована l1 регуляризация, при которой модель склонна не полагаться на признаки, которые вносят незначительный вклад в результат модели, то есть имеют небольшие веса. Для оптимизации был использован liblinear.

- Лучший результат при параметре: C = 3.16228, что является слабее используемой в библиотеке по умолчанию регуляризации C=1
- Accuracy = 0.9831
- F1 = 0.853
- AUC = 0.9549



Следующая модель это метод опорных векторов.

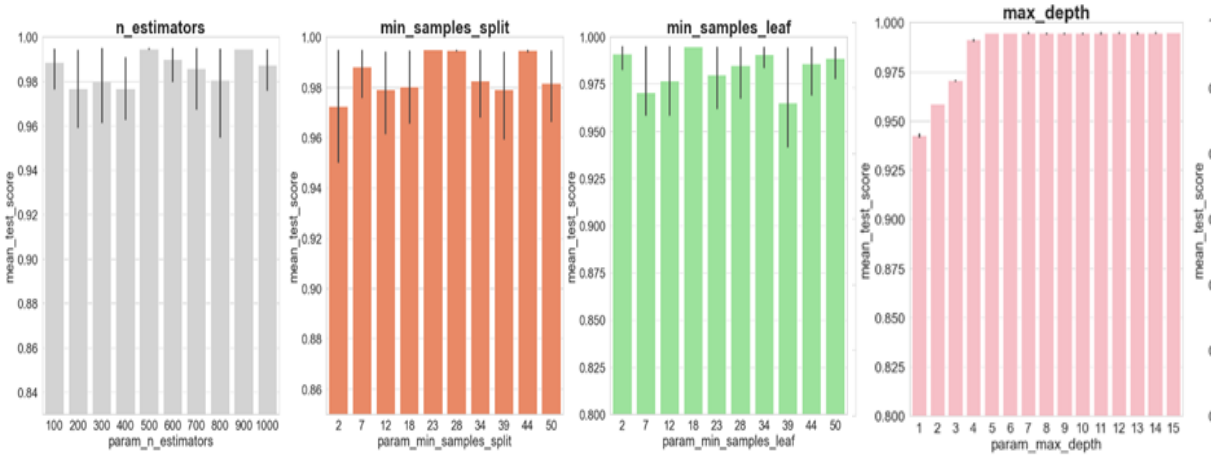
- Лучший результат - при параметрах C = 5, gamma = 0,125, где параметр C отвечает за силу регуляризации l2 (слабей, чем по умолчанию C=1) и gamma за коэффициент при ядре.
- F1 = 0.93768
- Accuracy = 0.99275
- AUC = 0.947



Приводим результаты модели случайного леса.

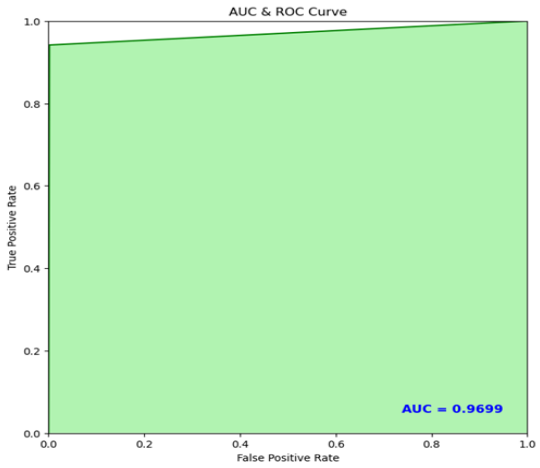
С помощью RandomizedSearchCV были проанализированы метрики случайных векторов гиперпараметров с целью сужения потенциальной области поиска гиперпараметров на GridSearchCV. Получаем, что оптимальное количество деревьев находится в окрестностях 500 или 900, глубина дерева лежит около 5, 6 или 15. Минимальное число объектов при котором идёт дальнейшее расщепление находится в окрестностях 23, 28 или 44, ограничение числа объектов в листьях – 17,18,19.

Графики представлены только для n_estimators, min_samples_split, min_samples_leaf и max_depth:



По итогам поиска гиперпараметров на GridSearchCV получаем, что оптимальным вектором гиперпараметров является:

n_estimators = 867,
max_depth = 15,
max_features = 'sqrt',
criterion = 'gini',
min_samples_leaf = 15,
min_samples_split = 25,
bootstrap = False



Итоговые метрики, полученные на тестовой выборке, получились следующими:

accuracy	F1	ROC-AUC
0.9948	0.9567	0.9699

Следующая модель это XGBoost.

Получаем оптимальные гиперпараметры модели:

'eta': 0.001, 'max_depth': 5, 'scale_pos_weight': 1,

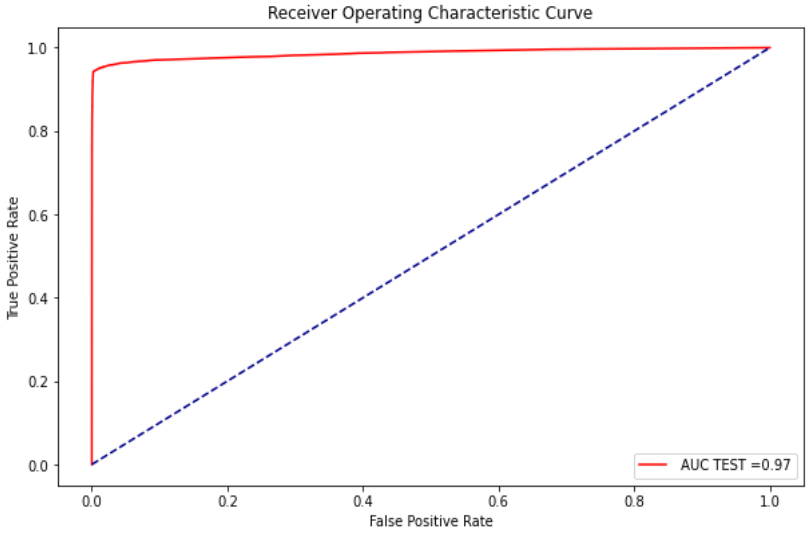
где Eta – размер шага (градиентного спуска)

Max_depth – максимальная глубина дерева

Scale_pos_weight – параметр, регулирующий веса наблюдений с меткой 1 и с меткой 0

Результаты оценки качества по разным метрикам для XGB:

	Среднее на 5-fold CV	Тестовая выборка
Accuracy	-	0.9948
F1-score	0.9570	0.9569
ROC-AUC	-	0.9700
Average precision	-	0.9194
Cohen-Kappa coefficient	-	0.9541
Matthews coeficient	-	0.9542



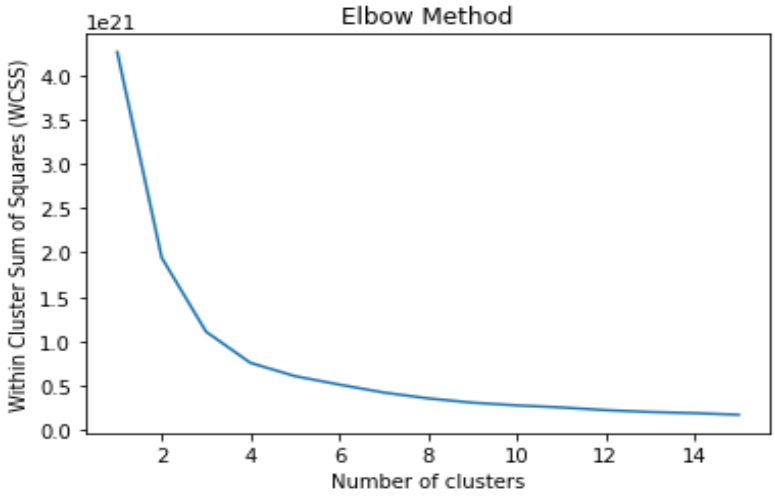
Также, для результатов предсказаний этого классификатора мы рассчитали значения метрик Average precision, коэффициента корреляции Метьюза и коэффициента Каппа-Коэна. Эти метрики используются для оценки классификаторов при несбалансированных выборках: коэффициенты корреляции Метьюза и Каппа-Коэна принимают значения от -1 до 1. Значение равное 1 характеризует идеальный алгоритм классификации, 0 – характеризует случайный классификатор, -1 – обратный (inverse prediction). Полученные результаты, равные 0,95, свидетельствуют о высокой предиктивной силе модели.

Поскольку XGB оказывается лучшей моделью, проверим, возможно ли улучшить результат посредством предварительного создания нового признака на основе кластеризации k-means

k-means, как k-nearest neighnours, полагается на предпосылку, что схожие объекты должны находиться рядом друг с другом. Отличие в том, что

это обучения без учителя, метки алгоритму не доступны и он должен сам сформировать k кластеров, где количество кластеров является гиперпараметром.

Используя elbow метод, выбираем k=5.



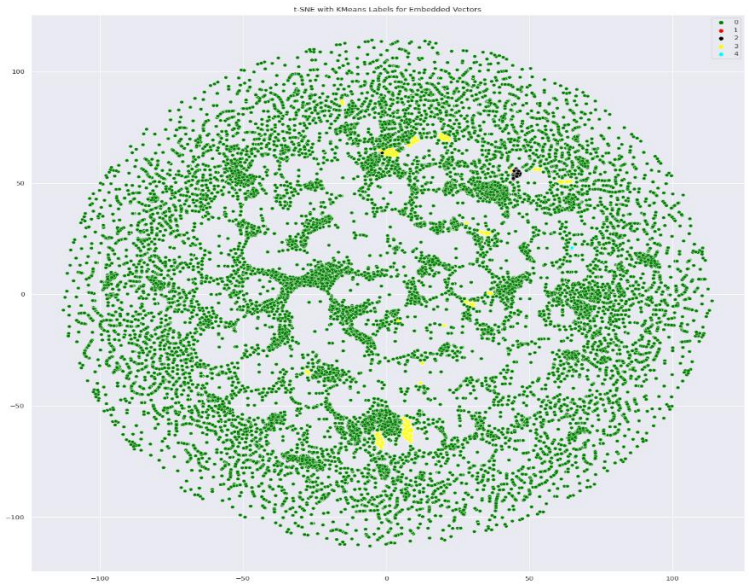
Результаты оценки качества для XGB с кластеризованным признаком:

	Среднее на 5-fold CV	Тестовая выборка
Accuracy	-	0.9948
F1-score	0. 0.9569	0.9568
ROC-AUC	-	0.9700
Average precision	-	0.9194
Cohen-Kappa coefficient	-	0.9541
Matthews coeficient	-	0.9542

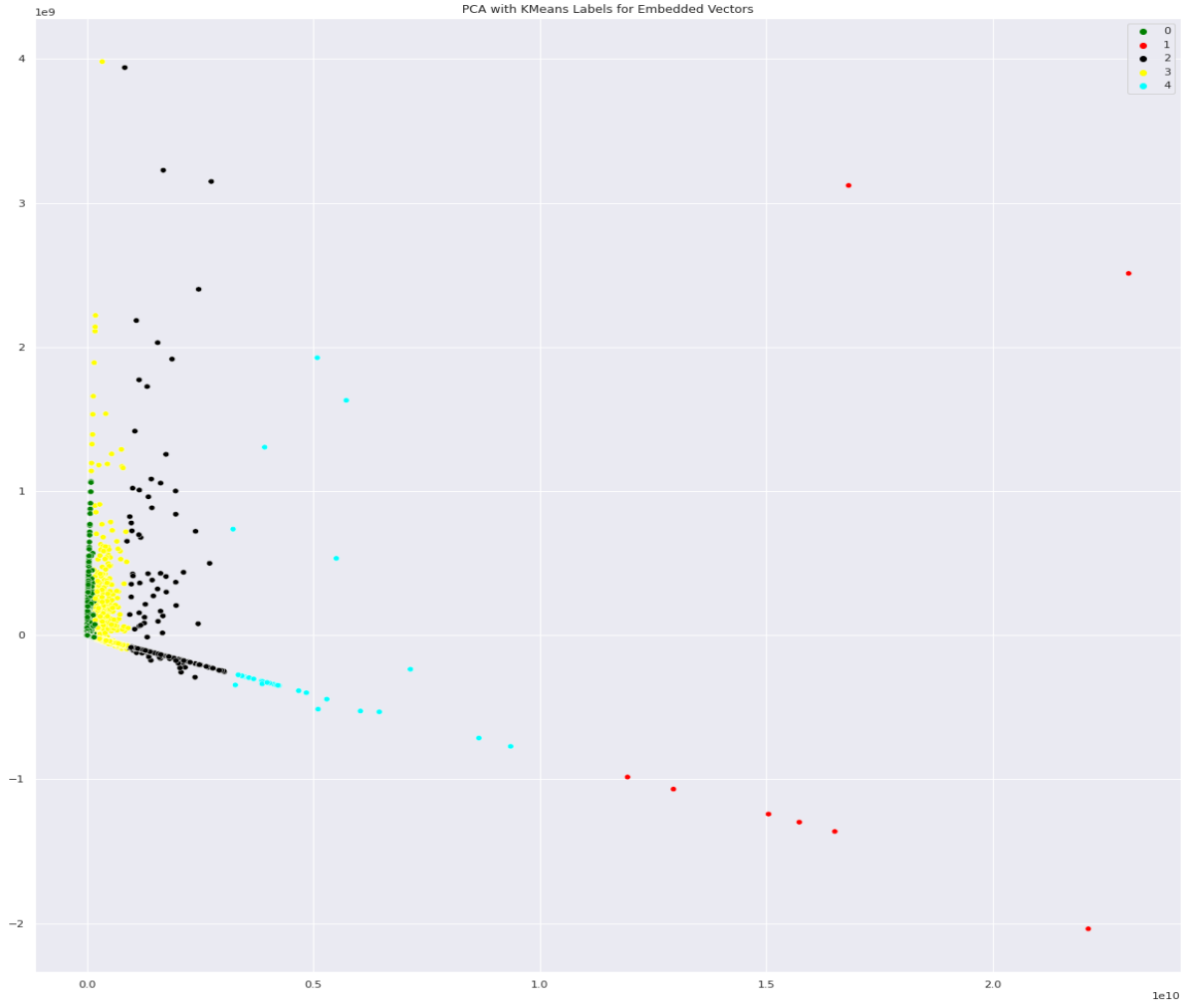
Наличие нового признака не дало какой-либо прирост в качестве модели.

Визуализируем результаты k-means кластеризации путем сокращения размерности до 2.

Используем технику понижения размерности t-SNE:

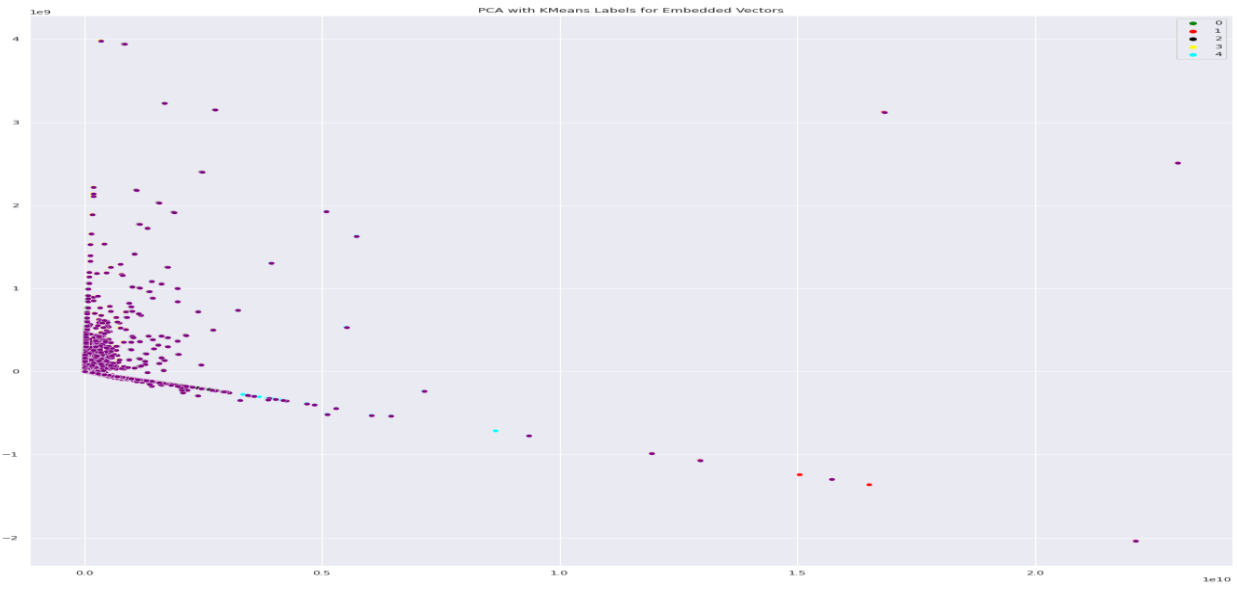


Несложно заметить, что кластеры не имеют четких границ, поэтому попробуем визуализировать через PCA:



PCA позволяет увидеть, что k-means действительно определил близкие друг к другу объекты и образовал 5 различимых кластеров.

Мы также можем отметить стартапы на графике:



В результате стартапы, фиолетовым, оказываются разбросаны по всем кластерам, а не сконцентрированы в каком-то одном.

Из этого можем сделать вывод, что просто кластеризация не может решать задачу бинарной классификации на наших данных так же, как это делают методы обучения с учителем.