УДК 004.75, 004.855.5

РЕКОМЕНДАТОР ДЛЯ АЛГОРИТМОВ КЛАССИФИКАЦИИ PYSPARK.ML

А.Вертинская
Магистрант БГУ

Б.Зибицкер
CEO BEZNext,
Чикаго, США

А.Толстиков,
Старший преподаватель
кафедры *ВМ*

Белорусский государственный университет, Республика Беларусь
E-mail: tyndria23@gmail.com

А.Е.Вертинская
Окончила Белорусский государственный университет. Магистрант БГУ. Занимается сравнением алгоритмов классификации.

Б.Зибицкер
Президент компании BEZNext, почетный доктор БГУИР.

А.А.Толстиков
Старший преподаватель кафедры вычислительной математики факультета прикладной математики и информатики Белорусского государственного университета.

Аннотация. Существует множество алгоритмов машинного обучения, которые могут использоваться для решения самых разнообразных бизнес-задач. Некоторые алгоритмы имеют хорошую точность, но используют много процессорного времени. Другие быстро выдают предсказания, но потребляют много памяти. Выбор подходящего алгоритма и библиотеки машинного обучения в начале проектирования приложения является трудоемкой задачей, а в некоторых сферах - очень дорогой.. В этой статье мы сравним некоторые алгоритмы *двухклассовой классификации* и опишем методику выбора подходящего алгоритма в зависимости от требований задачи. Алгоритмы классификации широко используются в таких сферах, как медицина, компьютерная безопасность, компьютерное зрение, кредитный скоринг.

Мы рассмотрим результаты бенчмарк-тестов для нескольких алгоритмов классификации. Бенчмарк-тесты проводились для выборок с различным количеством объектов и признаков. На основе результатов тестов, характеризующих производительность и качество алгоритмов, были построены регрессионные модели.

Мы также рассмотрим результаты рекомендатора, который дает возможность упростить процесс выбора алгоритма классификации. Рекомендатор использует предсказанные метрики регрессионных моделей во взвешенной сумме, коэффициенты которой определяются требованиями бизнес-задачи. Взвешенная сумма является оценкой алгоритма, алгоритм с наибольшей оценкой выбирается как самый подходящий алгоритм классификации.

Ключевые слова: рекомендация, бенчмарк-тест, классификация, Pyspark.

Введение. Существует множество библиотек, которые в свою очередь содержат в себе различные алгоритмы классификации (см. Рисунок 1). Очень часто на этапе реализации модели необходимо выбрать алгоритм, который будет не только давать хорошую точность предсказаний, но и будет подходить по производительности.

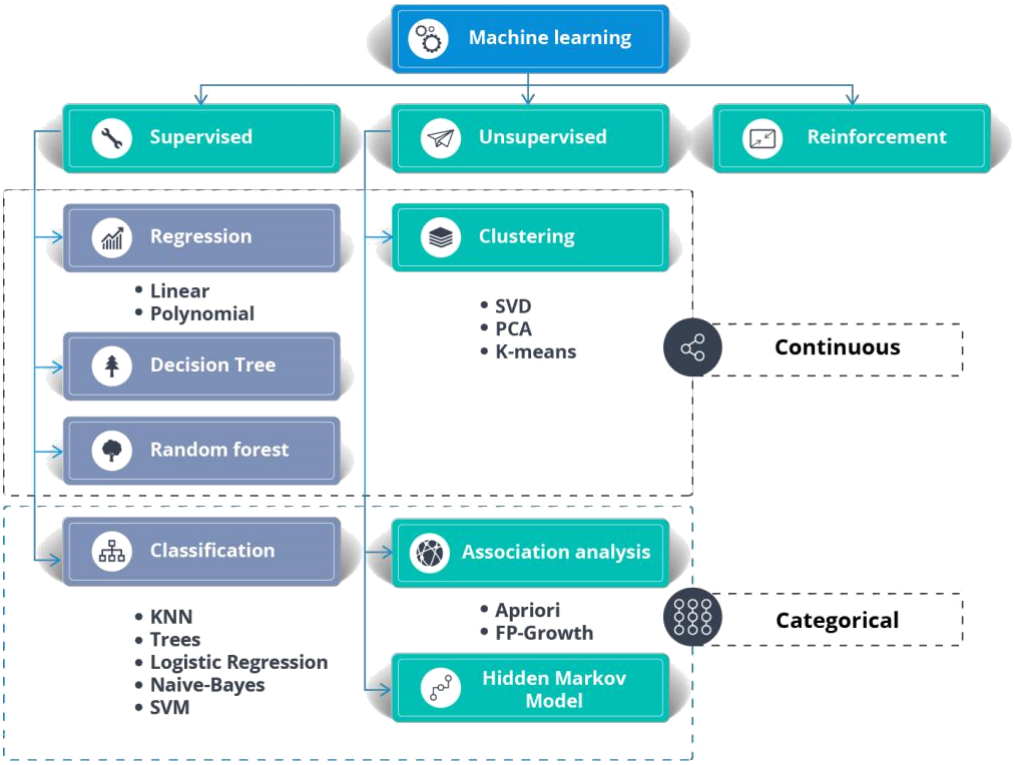

Рисунок 1. – Разнообразие алгоритмов машинного обучения [8]

Опишем процесс тестирования, который позволит быстро и недорого выбрать алгоритм классификации на начальном этапе проектирования приложения в зависимости от требования и характеристик задачи.

Мы будем оценивать следующие метрики алгоритмов: время ответа, метрики качества, загрузка процессора, использования памяти. Веса именно этих метрик в рекомендаторе можно будет менять в зависимости от *требований задачи*.

Каждая задача, для которой мы хотим выбрать наиболее подходящий алгоритм, по сути представляет собой набор данных, который описывается двумя признаками: **число объектов и число признаков.** Именно по этим признаками мы будем предсказывать метрики алгоритмов. Поэтому чтобы иметь возможность обучить регрессионные модели, мы должны собрать значения всех метрик для различного числа объектов и признаков.

Обучающая выборка представляет собой *множество объектов*, для которых известно, к каким классам они относятся. Каждый объект описывается *набором признаков*.

Опишем этапы тестирования алгоритмов и реализации рекомендатора:

–Подготовка данных с различным числом объектов и признаков.

–Подготовка и запуск бенчмарк-тестов.

–Сбор метрик: время обучения и ответа, точность, загрузка процессора, использования памяти.

–Обучение регрессионных моделей на собранных метриках.

–Предсказание метрик для конкретной задачи.

–Подсчет взвешенной суммы для каждого алгоритма и выбор наиболее подходящего.

Технологии, данные и методы. Для тестирования алгоритмов классификация использовался фреймворк **Spark** - платформа для параллельной обработки данных, которая позволяет повышать производительность за счет сокращений операций вводы / вывода, которые компенсируются оптимизированной работой в оперативной памяти. Данная

технология дала возможность уменьшить количество времени и ресурсов, потраченных на тестирование алгоритмов классификации.

Тесты запускались на кластере IBM, где использовались четыре Spark-узла для параллльных вычислений, каждый из которых имел 8 гигабайтов оперативной памяти.

Данные генерировались с помощью функции *make_classification* пакета библиотеки *sklearn.datasets*. Согласно документации функция создает кластеры точек, нормально распределенных вокруг вершин гиперкуба, размерность которого равна количеству признаков. Далее функция присваивает каждому классу (в нашем случае классов всего 2) равное количество кластеров. Таким образом она распределяет точки по классам. Функция также вводит взаимозависимость между признаками и добавляет различные типы шума к данным.

Каждая сгенерированная выборка имеет размер × , где *n* - это количество объектов в диапазоне [1200, 1000000], а *m* - это количество признаков в диапазоне [4, 300], структура выборки представлена в Таблице 1.. Всего использовалось около 130 выборок различной размерности.

Таблица 1. – Структура выборки для бенчмарк-теста

	Признак 1	Признак 2	...	Признак m
Объект 1				
Объект 2				
...				
Объект n				

Для оценки точности алгоритмов двухклассовой классификации использовались следующие метрики:

–площадь под ROC-кривой (Area Under ROC Curve, AUC-ROC);

–площадь под PR-кривой (Precision-Recall Curve, PR-ROC).

ROC-кривая – это график, отображающий соотношение между долей истинно-положительных предсказаний, и долей ложно-положительных предсказаний при варьировании порога решающего правила. Чем больше площадь под ROC-кривой, тем качественнее предсказания классификатора, при этом значение 0,5 означает непригодность выбранного метода классификации [6, 7]. Если положительный класс существенно меньше по размеру, то AUC-ROC может давать необъективную оценку качества.

Избавиться от указанной проблемы с несбалансированными классами можно перейдя от ROC-кривой к Precision-Recall кривой. Она определяется аналогично ROC-кривой, только по осям откладываются полнота и точность [7]. Полнота определяется как соотношение между истинно-положительными предсказаниями и всеми положительными предсказаниями, в том числе, неверными. Точность определяет долю истинно-положительных предсказаний среди всех положительных объектов.

На рисунке 2 представлены примеры графиков ROC-кривой и PR-кривой.

Тестировались следующие алгоритмы двухклассовой классификации:

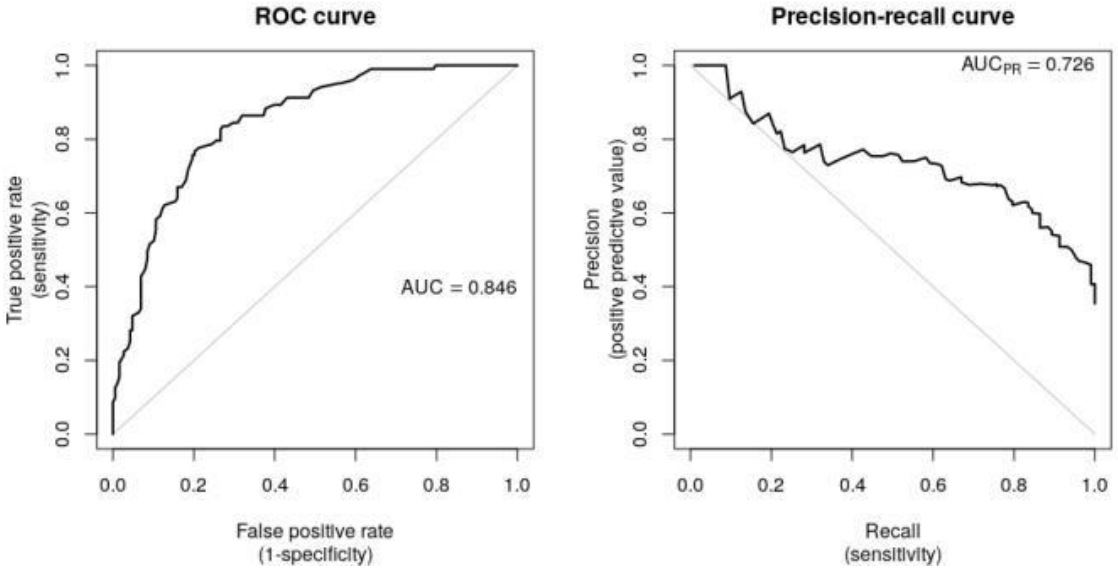

Рисунок 2. – Чем больше площадь под кривыми, тем качественнее классификатор

Тестировались следующие алгоритмы двухклассовой классификации:
- логистическая регрессия (Logistic regression);
- случайный лес (Random forest);
- дерево решений (Decision tree);
- градиентный бустинг над деревьями (Gradient-boosted tree);
- метод опорных векторов (Linear Support Vector Machine).

После выполненых тестов для каждого алгоритма и каждой метрики создается следующая выборка: признаки - это количество объектов и количество признаков исходной выборки, значение на этом объекте - это значение конкретной метрики (использование процессора, памяти, время выполнения теста, метрики качества).

Используя регрессионные модели мы можем предсказать значение каждой метрики каждого алгоритма для определенного количество объектов и признаков. Далее мы считаем взвешенную сумму и получаем оценку алгоритма, по которой мы можем отсортировать все алгоритмы.

Регрессионная модель представляет собой градиентный бустинг для регрессии *sklearn.ensemble.GradientBoostingRegressor* библиотеки *sklearn.*

Методика сравнения. Для каждого из алгоритмов классификации выполнялось следующее:

1) Алгоритм обучался на каждой выборке 3 раза.

2) На выходе каждого теста мы имели следующие метрики: время ответа, использование памяти и процессора, метрики качества.

3) Поскольку для рекомендации алгоритма мы должны предсказывать метрики по двум признакам: числу объектов и признаков - нам нужно для каждой метрики каждого алгоритма собрать выборки размером $n_{A,S} \times m_{A,S}$, где $n_{A,S}$, - это число объектов, равное количеству проведенных Spark-тестов для этого алгоритма, а m_A - это размерность признакового пространства, равное 2-м (число признаков и число объектов).

4) Для каждой такой выборки мы обучаем регрессионную модель, которая предсказывает значение метрики по размерности выборки.

5) Предсказав метрики для каждого алгоритма и определив коэффициенты при каждой метрике (коэффициенты определяются в зависимости от требования задачи), мы можем подсчитать взвешенную сумму, которая будет являться оценкой алгоритма.

6) На основе оценок алгоритмов мы можем выбрать наиболее подходящий.

Результаты бенчмарк-тестов. Рассмотрим результаты тестов на графиках. Проанализируем такие метрики как площадь под PR-кривой, ROC-кривой, использование памяти, процессора, время обучение и ответа, количество операций ввода / вывода.

Рассмотрим **метрики качества**.

На рисунке № 3 – представлен график метрики площадь под PR-кривой, на рисунке № 4 – площадь под ROC-кривой.

При небольшом пространстве признаков градиентный бустинг работает лучше всех.

Неплохое качество предсказаний также показывает случайный лес.

При увеличении количества объектов качество предсказаний на выборках в среднем уменьшается.

Заметим, что логистическая регрессия и метод опорных векторов дает не очень хорошую точность при небольшом пространстве признаков.

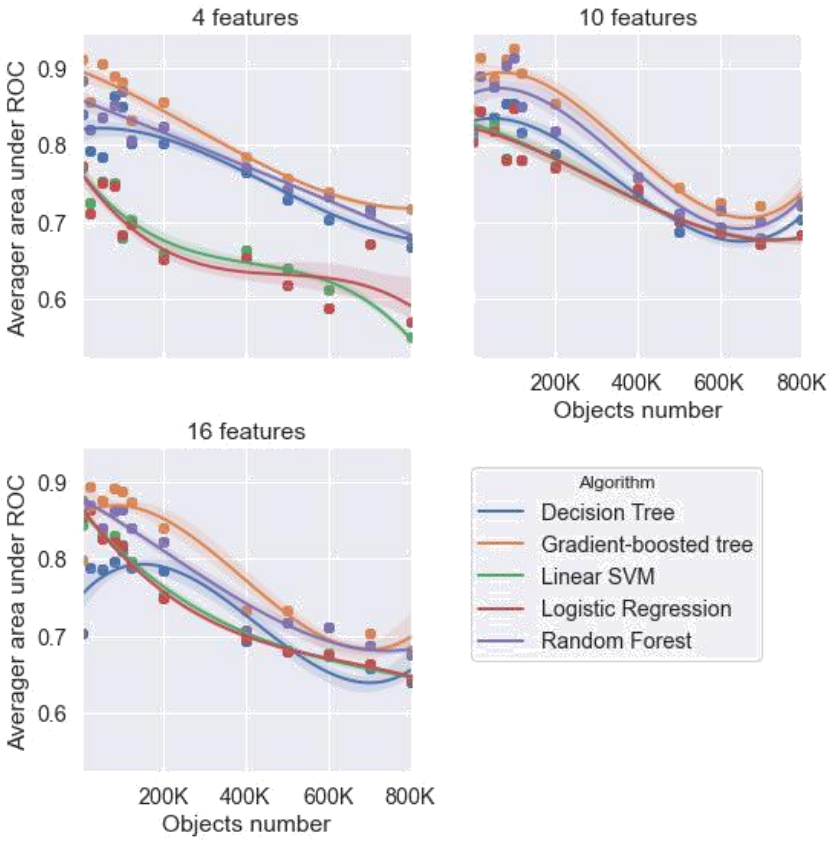

Рисунок *3.* – Градиентный бустинг и случайный лес имеют высокие значения площади под PR-кривой на небольшом пространстве признаков

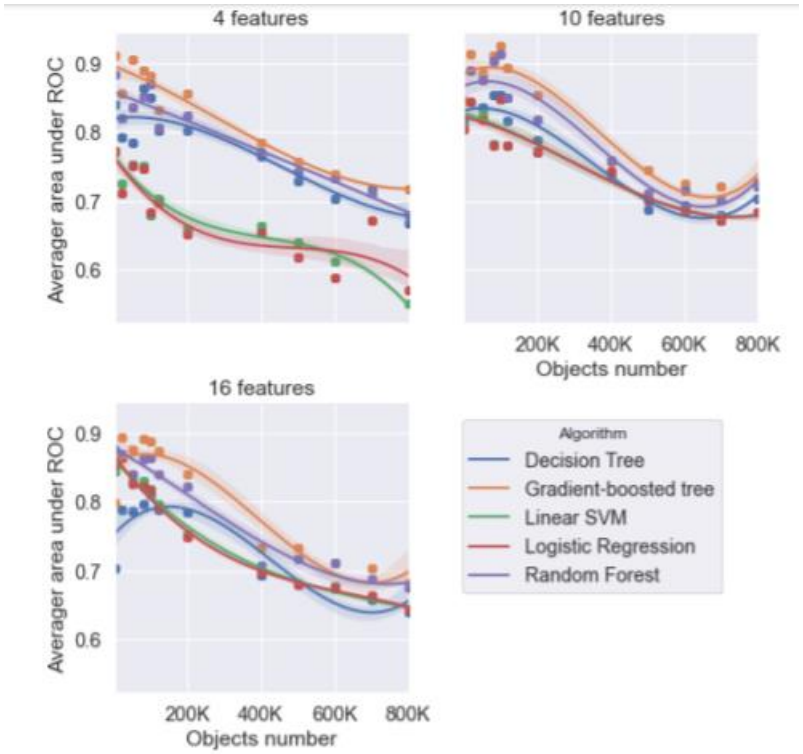

Рисунок 4. –Логистическая регрессия и метод опорных векторов плохо предсказывают на небольшом пространстве признаков

Рассмотрим метрики качества при увеличении пространства признаков.

На рисунке № 5 и 6 можно заметить, решающее дерево классифицирует объекты хуже всех. Неплохо себя показывает логистическая регрессия, метод опорных векторов.

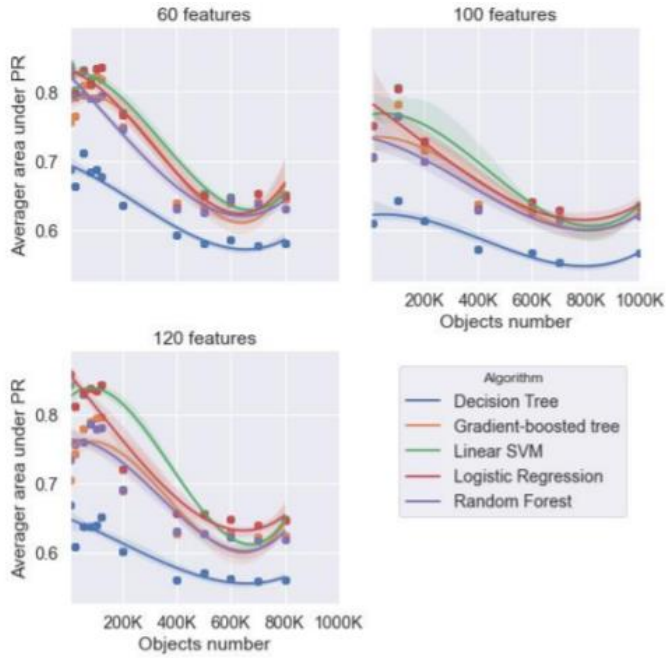

Рисунок 5. – При увеличении пространства признаков логистическая регрессия и метод опорных векторов увеличивают свои показатели

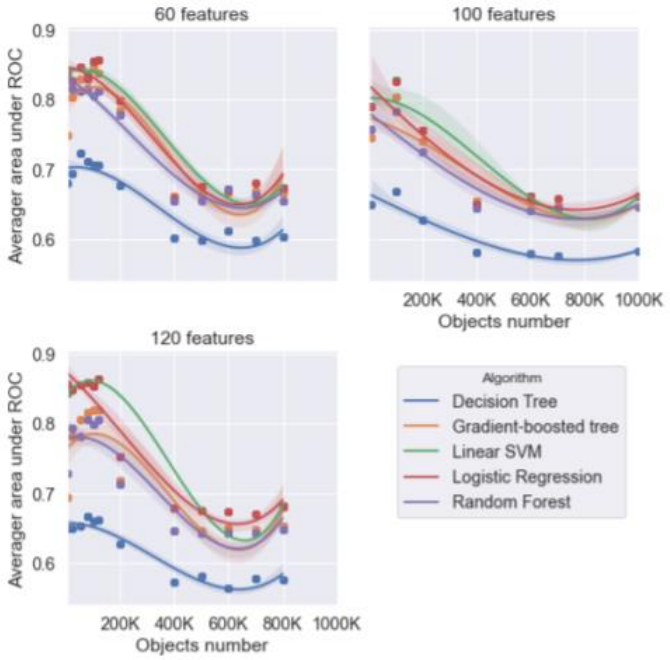

Рисунок *6*. – Дерево решений отстает по метрикам предсказаний, хорошо предсказывает логистическая регрессия

Рассмотрим метрики **времени обучения и ответа**.

На рисунке № 7 представлен график метрики времени обучения.

Метод опорных векторов обучается долго относительно остальных алгоритмов. При увеличении количества объектов время обучения градиентного бустинга над решающими деревьями также увеличивается.

Необычно, что время обучения метода опорных векторов при увеличении выборки уменьшается, это может быть связано с тем, что алгоритмы распаралелливались.

При росте количества объектов немного увеличивается время обучения случайного леса и решающего дерева.

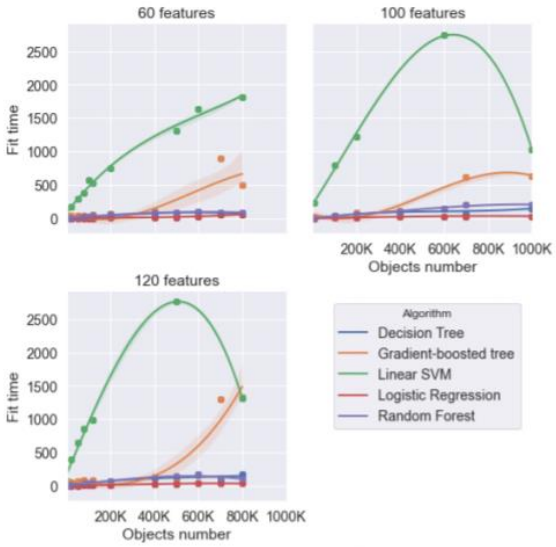

Рисунок *7*. – Метод опорных векторов обучается относительно долго, время обучения градиентного бустинга увеличивается при росте количества объектов.

Быстрее всего классифицирует объекты логистическая регрессия, что видно на рисунке № 8.

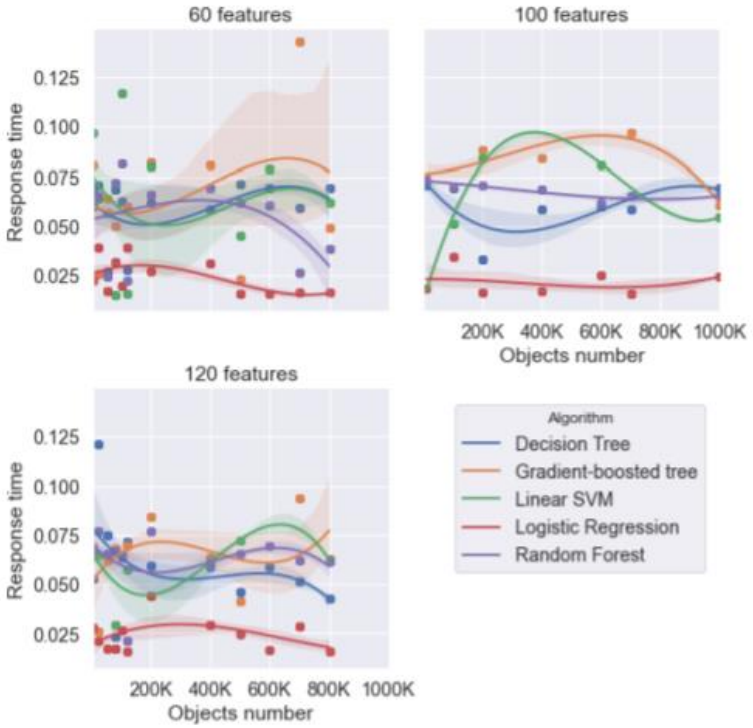

Рисунок *8.* – Логистическая регрессия быстрее всех классифицирует объекты

Рассмотрим **метрику использования памяти и процессора** для алгоритмов классификации на рисунке 9.

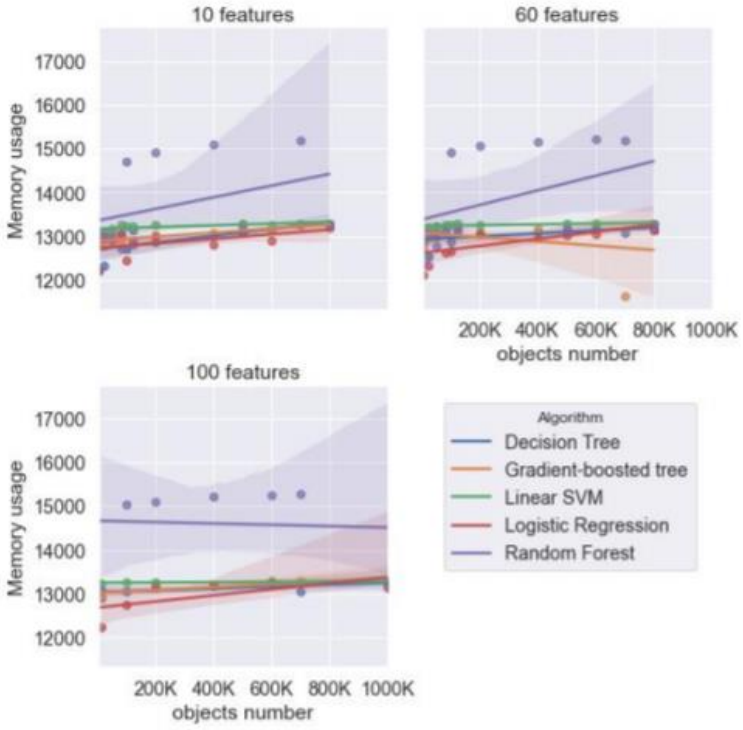

Рисунок *9.* – Случайный лес использует относительно много памяти

Больше всего памяти для классификации использует случайный лес.

Что касается использования процессора, больше всего времени потребляет метод опорных векторов, сравнение алгоритмов по этой метрике можно увидеть на рисунке № 10.

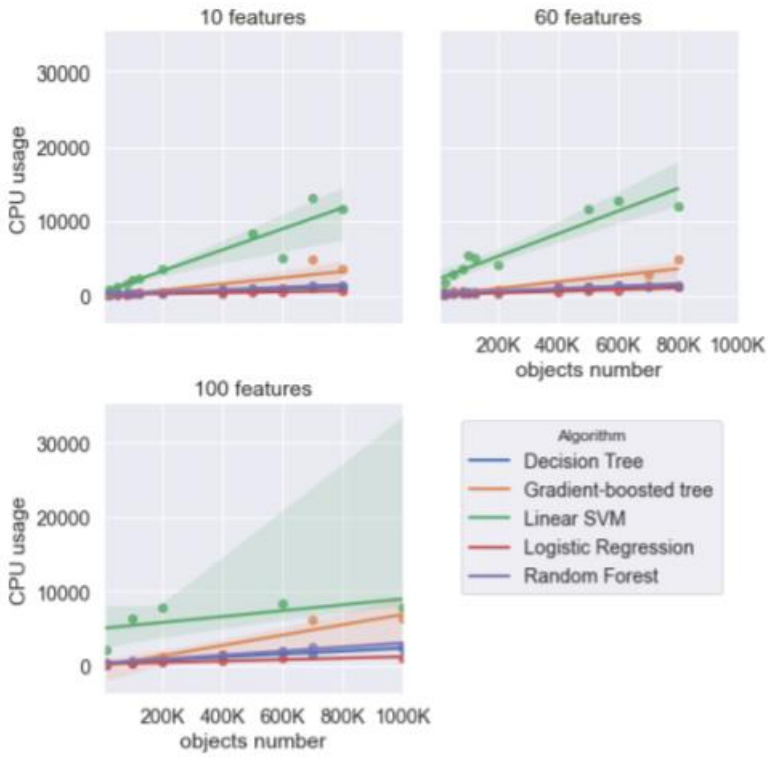

Рисунок *10.* – Метод опорных векторов использует относительно много процессорного времени

Рекомендатор. Около 650 проведенных тестов позволяют собрать метрики качества производительности. Для каждого алгоритма мы имеем выборки следующего вида (Таблица 2):

Таблица 2. –Пример обучающей выборки для регрессионной модели

n - кол-во объектов	m - кол-во признаков	метрика
100000	48	0.793048015556
100000	150	0.81262611612
...	...	...
800000	4	0.840813346872

Регрессионные модели, обученные на таких выборках, предсказывают метрики для размера датасета (*n,m*). Для построения моделей использовался градиентный бустинг для регрессии *sklearn.ensemble.GradientBoostingRegressor* библиотеки *sklearn*.

Для каждого алгоритма мы можем предсказать все метрики, соответственно оценка алгоритма представляет собой взвешенную сумму метрик:

$$score = w_1 \times areaUnderRoc + w_2 \times areaUnderPR + w_3 \times responseTime + \\ + w_4 \times fitTime + w_5 \times cpuUsage + w_6 \times memoryUsage$$

Коэффициенты определяются согласно требованиям задачи.

После подсчета оценок алгоритмы сортируются по убыванию (от худшего к лучшему).

Мы проверили качество регрессионных моделей на датасете *Breast Cancer Wisconsin* [9].

Для этого мы запустили каждый алгоритм на кластере, собрали метрики качества и производительности.

Наш датасет имеет размер (n,m) = (*683, 9*), где *n* – количество объектов в выборке,а *m* – количество признаков. То-есть предсказывать метрики мы будем именно для этих параметров.

Рассмотрим несколько вариантов наборов весов для взвешенной суммы.

Вариант 1. Пусть веса равны ⅙, всего у нас шесть метрик.

Посчитав взвешенную сумму для каждого алгоритма с предсказанными и реальными метриками, мы отсортировали алгоритмы **от лучшего к худшему** (Таблицу 3):

Таблица 3. – Сравнение предсказанных и истинных оценок алгоритмов

№	Предсказанные оценки	Истинные оценки
1	Логистическая регрессия	Логистическая регрессия
2	Градиентный бустинг над деревьями	Метод опорных векторов
3	Метод опорных векторов	Градиентный бустинг над деревьями
4	Дерево решений	Дерево решений
5	Случайный лес	Случайный лес

При одинаковых весах можно отметить, что некоторые метрики были предсказаны хорошо.

Вариант 2. Пусть нам важны лишь метрики качества и время обучения, пусть веса при данных метриках буду равны 0.3, а остаток 0.1 поделим поровну между оставшимися метриками.

Таблица 4. – Сравнение предсказанных и истинных оценок алгоритмов

№	Предсказанные оценки	Истинные оценки
1	Логистическая регрессия	Логистическая регрессия
2	Градиентный бустинг	Метод опорных векторов
3	Дерево решений	Дерево решений
4	Случайный лес	Градиентный бустинг
5	Метод опорных векторов	Случайный лес

Здесь мы придали вес трем метрикам: метрикам качества и времени обучения.

По предсказаниям метод опорных векторов занял последнее место, т.к. во время тестирования обучения данного алгоритма занимало относительно много времени.

Это говорит о том, что для предсказания метрик скорее всего недостаточно как количества данных, так и их разнообразия.

Вариант 3. Пусть нам важен быстрый ответ, который потребляет не очень много памяти, возможно не самый точный. Для метрики времени ответа и использования памяти положим вес равный 0.25, для метрик качества - 0.20, соответственно для остальных метрик - 0.05.

Таблица 5. – Сравнение предсказанных и истинных оценок алгоритмов

№	Предсказанные оценки	Истинные оценки
1	Логистическая регрессия	Логистическая регрессия
2	Градиентный бустинг	Метод опорных векторов
3	Метод опорных векторов	Градиентный бустинг
4	Случайный лес	Дерево решений
5	Дерево решений	Случайный лес

Стоит отметить, что для задачи *Breast Cancer Wisconsin* регрессионные модели довольно неплохо предсказали метрики качества, однако плохое качество предсказаний мы получили для метрик времени ответа, иногда - использование памяти и процессорного времени.

Для улучшения предсказаний можно исследовать следующие направления:

1) Протестировать выборки на большом количестве сгенерированных и *реальных* выборках различной размерности.

2) Возможно, сконцентрироваться на определенной области, где обучение на похожих данных будет в какой-то мере гарантировать хорошее качество предсказаний.

3) Усовершенствовать механизм сбора и обработки метрик производительности.

4) Усовершенствовать регрессионные модели.

Заключение. Мы провели множество бенчмарк-тестов на выборках разной размерности для следующих алгоритмов классификации:

−логистическая регрессия (Logistic regression);

−случайный лес (Random forest);

−дерево решений (Decision tree);

−градиентный бустинг над деревьями (Gradient-boosted tree);

−метод опорных векторов (Linear Support Vector Machine).

Мы сравнили такие метрики, как время ответа, качество предсказаний, использование памяти и процессора для вышеуказанных алгоритмов.

На основе собранных данных, полученных при выполнении бенчмарк-тестов, для каждого алгоритма мы построили регрессионные модели, которые позволяют предсказать каждую из метрик для пары (n,m), где n – это количество объектов выборки, а m – это количество признаков.

Рекомендатор представляет собой набор регрессионных моделей, которые дают возможность оценить протестированные алгоритмы для некоторой выборки размера (n,m) путем взвешенной суммы предсказанных метрик. Коэффициенты при метриках определяются согласно требованиям задачи.

Таким образом, рекомендатор позволяет быстро оценить потенциал алгоритмов для конкретной бизнес-задачи.

На основе распределенных тестов мы построили модели, которые предсказывают метрики тестов для определенного количества объектов и признаков. Мы также проверили модели на датасете Breast Cancer Wisconsin и сравнили ранжирование алгоритмов.

Список литературы

[1.] Zibitsker B., Selection of Machine Learning Algorithms and Libraries for Big Data Applications: conference CMG impact 2017 material.

[2.] Введение в машинное обучение [Электронный ресурс]. – Режим доступа: https://www.coursera.org/learn/vvedenie-mashinnoe-obuchenie.

[3.] Воронцов, К. Математические методы обучения по прецедентам [Электронный ресурс]. – Режим доступа: http://www.machinelearning.ru/ wiki/images/6/6d/Voron-ML-1.pdf.

[4.] Воронцов, К. Лекции по логическим алгоритмам классификации [Электронный ресурс]. – Режим доступа: http://www.machinelearning.ru/ wiki/images/3/3e/Voron-ML-Logic.pdf.

[5.] What is Apache Spark? // Microsoft documentation [Electronic resource]. – 2019. – Mode of access: https://docs.microsoft.com/en-us/dotnet/spark/what-is-spark. – Date of access: 12.03.2020.

[6.] ROC-кривая // Википедия [Электронный ресурс]. – 2020. – Режим доступа: https://ru.wikipedia.org/wiki/ROC-кривая. – Дата доступа: 04.03.2020.

[7.] Соколов Е., Семинары по выбору моделей / Е. Соколов // Профессиональный информационно-аналитический ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных [Электронный ресурс]. – 2015. – Режим доступа: http://www.machinelearning.ru/wiki/images/1/1c/Sem06_metrics.pdf. – Дата доступа: 02.04.2020.

[8.] Upasana, Machine Learning Algorithms / Upasana // Edureka [Electronic resource]. – 2020. – Mode of access: https://www.edureka.co/blog/machine-learning-algorithms/. – Date of access: 02.04.2020.

[9]. Wolberg W.H. , Street W.N., Mangasaria O.L. Breast Cancer Wisconsin (Diagnostic) Data Set / W.H. Wolberg, W.N. Street, O.L. Mangasaria // [Electronic resource]. – 1995. – Mode of access: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic). – Date of access: 16.04.2020.