Разделы

1 Деревья принятия решений

Дерево принятия решений является графом, в узлах которого записаны условия определяющие переход в следующую вершину, а в листьях хранятся значения целевой функции (рисунок 1). Таким образом, решение задачи сводится к спуску из корня дерева до его листа.

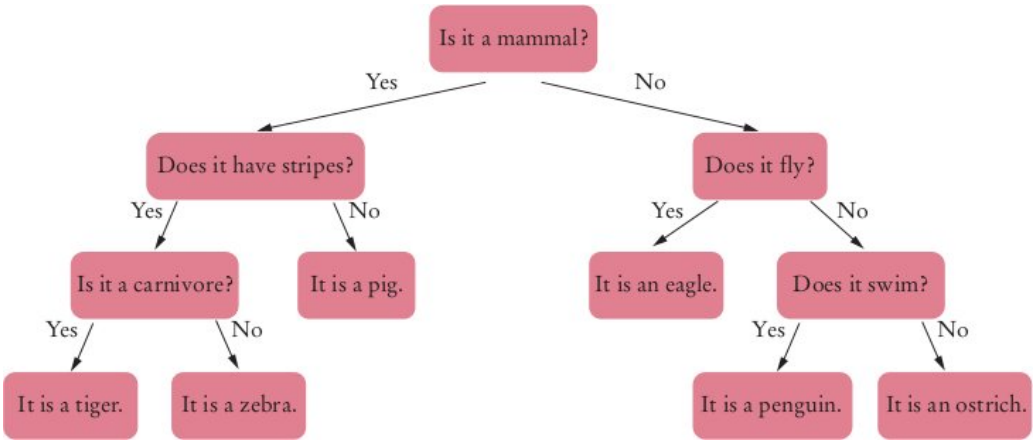


Рисунок 1 – Пример дерева решений классификации животных

В зависимости от решаемой задачи в листьях дерева могут храниться значение одного из двух основных типов: либо это конкретный класс рассматриваемой предметной области при решении задачи классификации, либо число (регрессионная модель), определяющее вероятность наступления некоторого события (победа команды в предстоящем матче) или прогнозируемую величину исследуемой функции (цена от продажи квартиры). В качестве входных данных дерево решений принимает массив параметров с опорной информацией, используемой в узлах дерева для определения перехода, например, для расчета вероятности выигрыша футбольной команды это будут: информация о положении в турнирной таблице, погоде и предыдущих играх.

1.1 Общий подход формирования дерева

Деревья решений имеют очень простую и наглядную структуру, что позволяет формировать их вручную для небольшого набора данных. Тем не менее, существуют разные схемы для автоматического обучения и выращивания деревьев. Каждый из таких методов имеет свои преимущества и недостатки, но один из подходов стал основой большинства других решений, поскольку

оказался очень быстродействующим и простым в реализации. Этот алгоритм известен под названием рекурсивного секционирования [1] и действует по принципу инкрементного построения дерева.

Процесс выращивания начинается с пустого дерева и полного набора данных. Первым этапом решается задача поиска приемлемой точки разбиения исходной выборки на подмножества. С точки зрения реализации алгоритма, применяемый способ выбора атрибута разделения не имеет значения. Таким образом создается первый узел в дереве, разбивающий набор данных на несколько подмножеств, после чего данный процесс выполняется повторно, применительно к каждому из подмножеств, для создания поддеревьев. Критерий прекращения выращивания дерева выбирается в зависимости от решаемой задачи.

Для создания узлов принятия решений необходимо установить какими должны быть критерии секционирования. Как правило, при выборе атрибутов разбиения применяется жадный алгоритм, предусматривающий выбор наилучшего из приемлимых атрибутов.

1.2 Функция оценки

Основная идея при рекурсивном построении дерева — это создание «хорошего» разбиения для текущего узла. Чтобы оценить качество получаемых поддеревьев и выбрать среди возможных вариантов самое оптимальное, вводят специальную функцию оценки.

Для конкретности, рассмотрим пример построения дерева принятия решений задачи классификации. На вход подается тестовая выборка $< x_i, b_i >$, где x_i — i-ый вектор параметров, соответствующий классу b_i. Определено множество предикатов H и каждый не листовой элемент дерева связан с некоторым $h \in H$.

На рисунке 2 показан пример классификатора, разбивающего двумерное пространство точек на четыре части. Множество предикатов задано функциями вида $H = \{x_i > \alpha | \alpha \in \mathbb{R}\}$.

Использование функции оценки при построении дерева должно увеличить точность создаваемого классификатора. Определим ошибку в каждой вершине v как $E(v)$, следовательно суммарную ошибку по всему дереву можно

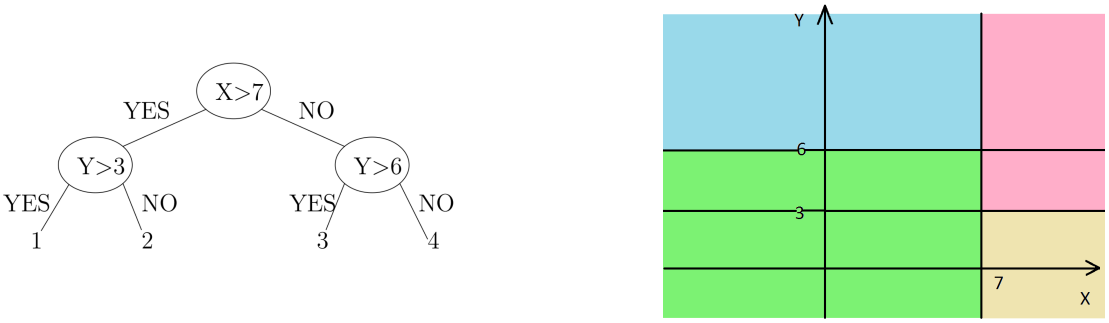


Рисунок 2 – Разбиение двумерного пространства точек деревом принятия решений; класс 1 — розовый цвет, класс 2 — желтый, класс 3 — голубой и класс 4 — зеленый

вычислить как $E(T) = \sum_{v \in T} Prob(v)E(v)$, где T — дерево принятия решений, а $Prob(v)$ — вероятность перехода в вершину v, т. е. если по предикату H из вершины v осуществляется переход в одно из двух поддеревьев с корнями в вершинах v_1 и v_2, вероятности для них можно рассчитать по формулам:

$$Prob(v_1) = P(H)$$

$$Prob(v_2) = 1 - P(H)$$

где $P(H)$ — вероятность перехода из вершины v в v_1. На очередном этапе рекурсивного алгоритма выращивания в качестве функции оценки можно использовать $E(T)$, где T — это дерево с предполагаемым разбиением и корнем в текущей вершине. Однако, такой подход не всегда работает, например, если в исходной вершине v наблюдалась ошибка $E(v) = 0.2$ и один из вариантов ее разделения предполагает добавление двух новых вершин v_1 и v_2 с равновероятными переходами $P(H) = 0.5$ и вероятностью ошибки $E(v_1) = 0.4$ и $E(v_2) = 0$, то итоговая ошибка $E(v)$ никак не изменится $E(v) = 0.5 * 0.4 + 0.5 * 0 = 0.2$, хотя в результате такого разбиения одна из веток дает сто процентный результат классификации.

1.3 Построение сверху-вниз

Функцию $E(T)$ можно использовать для оценки качества уже построенного дерева, а не для выбора одного из множества поддеревьев на этапе разбиения. Таким образом, выращивая дерево, можно говорить о решении задачи минимизации функции $E(T_i)$, где T_i — множество всех возможных

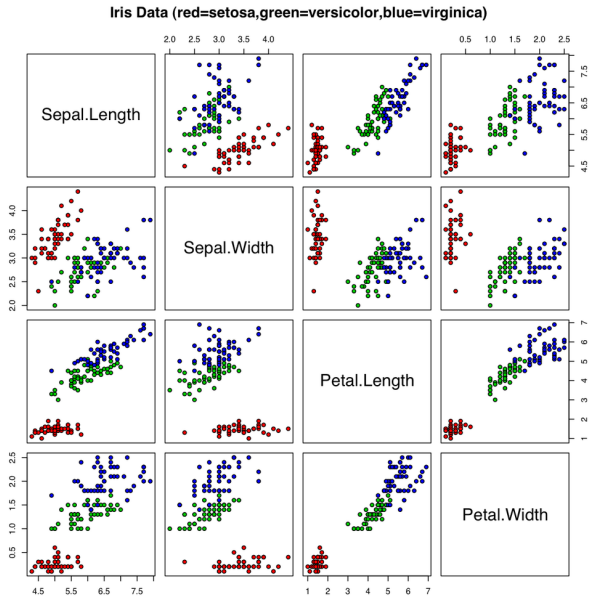
классификационных деревьев.

Пусть для некоторого предиката $h \in H$, терм $T(l, h)$ определяет дерево, в котором у листа l существует два новых дочерних элемента, переход в которые осуществляется по h. Тогда алгоритм построения можно определить следующим образом: на вход подается множество H и максимальный размер дерева t, построение начинается с дерева T_0, состоящим из одной вершины, и далее по алгоритму

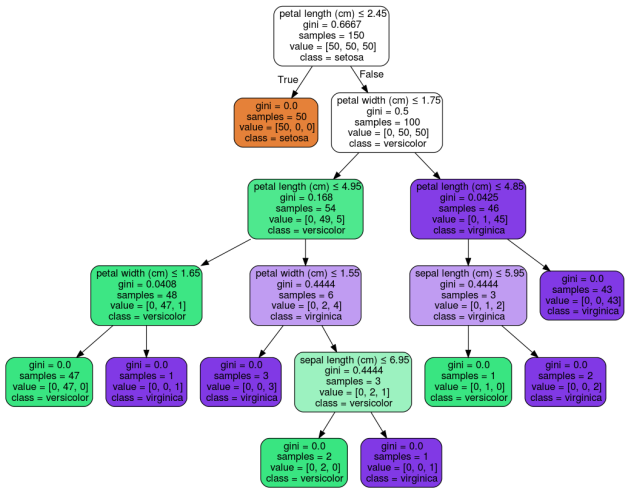
1. Закончить выращивание, если размер дерева больше установленного ограничения t

2. Построить новое дерево $T_{i+1} \leftarrow min_{l \in T, h \in H}(E(T_i) - E(T_i(l, h)))$

По такому принципу работают алгоритмы CART и C4.5, названный в 2008 году алгоритмом машинного обучения №1 [2]. Эти алгоритмы доступны в качестве готовых библиотек для языка программирования Python. С их помощью можно легко вырастить дерево для задачи классификации ирисов Фишера (рисунок 3) — набора данных, на примере которого Рональд Фишер в 1936 году продемонстрировал работу разработанного им метода дискриминантного анализа [3].



а)



б)

Рисунок 3 – а) Диаграмма рассеяния ирисов Фишера
б) Дерево принятия решений для классификации ирисов Фишера

1.4 Процедура обучения

При использовании автоматических методов построения деревьев принятия решений отдельное внимание необходимо уделять обучающей выборке чтобы не допустить переобучения (overfitting) — ситуации, когда во входном наборе данных обнаруживаются некоторые случайные закономерности, отсутствующие в генеральной совокупности. В противном случае, полученная модель является недостаточно общей.

Один из подходов борьбы с проблемой переобучения заключается в заготовке качественных наборов данных, разделенных на три группы:

- Обучающее множество, которое используется в алгоритме рекурсивного секционирования.
- Аттестационное множество, для усовершенствования дерева решений, полученного в результате обучения.
- Проверочное множество, позволяющее выполнить окончательную проверку результатов обучения для подтверждения применимости полученного дерева решений.

1.5 Отсечение веток

Отсечение частей дерева решений (pruning) является алгоритмом улучшения качества и осуществляется на этапе обработки, который следует за обучением. Для его работы необходимо иметь аттестационное множество, отличное от обучающей выборки. Метод основан на предположении, что в дереве существуют ребра, убрав которые произойдет увеличение точности.

Для того чтобы осуществить подобную проверку, необходимо знать, какое промежуточное значение накапливается в каждом узле принятия решений. Далее происходит тестирование дерева на всем аттестационном множестве с подсчетом успешно классифицированных наборов данных в каждой вершине. Если окажется, что некоторый узел имеет лучшую оценку, чем сумма всех его сыновей, то происходит усечение дерева (рисунок 4).

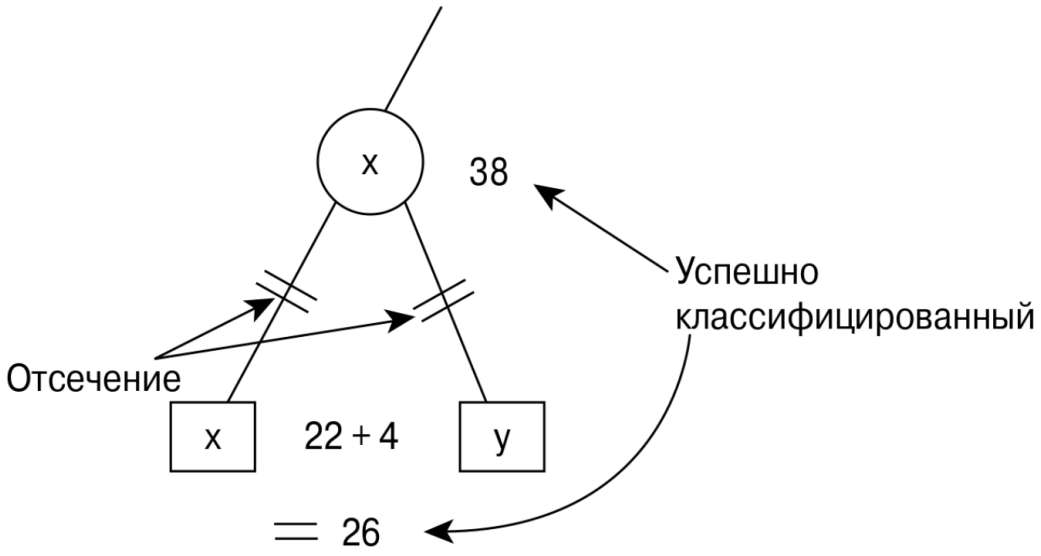


Рисунок 4 – Отсечение ребер дерева после подсчета успешно классифицированных примеров

1.6 Случайный лес

Метод случайного леса (random forest) основан на использовании комитета (ансамбля) деревьев принятия решений. Он был предложен Leo Breiman [4], одним из создателей алгоритма для построения бинарного дерева решений (CART). Метод дает высокое качество результата, лучше чем у нейронных сетей [5], эффективно обрабатывает данные с большим числом признаков классов, обладает возможностью оценивать значимости отдельных признаков в модели.

При обучении создается выборка из N примеров, а также задаются размерность пространства признаков M и дополнительный параметр m (для задач классификации обычно выбирается $m \approx sqrt(M)$). Все деревья комитета строятся независимо друг от друга по следующей процедуре:

- Генерируется случайное подмножество с повторениями размером N из множества обучающей выборки.

- Выращивается решающее дерево, классифицирующее примеры данного подмножества, причём в ходе создания очередного узла выбирается признак не из всего пространства признаков размерности M, а лишь из m случайно выбранных.

- Дерево строится до полного исчерпания подмножества примеров и не обрабатывается процедурами улучшения результата (такими как отсечение веток).

Классификация объектов проводится путём голосования: каждое дерево комитета относит классифицируемый объект к одному из классов, и побеждает тот, за который проголосовало наибольшее число деревьев. Размерность леса подбирается таким образом, чтобы минимизировать ошибку классификатора на тестовой выборке.

Случайные леса могут быть естественным образом использованы для оценки важности переменных в задачах регрессии и классификации. Меняя параметры выборки, и определяя точки максимумов можно установить какие из них являются более важными для тренировочного набора.

Использование случайного леса для задачи классификации изображений вместе с алгоритмами выделения регионов интереса (ROI — Region Of Interest) дает хорошие результаты [6].

2 Нейронные сети

Под нейронными сетями понимают математическую модель самообучающейся системы, имитирующей деятельность нервных клеток в биологических нейронных сетях. Понятие возникло при изучении процессов, протекающих в мозге, и при попытке их моделирования. Несмотря на большое разнообразие вариантов нейронных сетей, все они состоят из большого числа связанных между собой однотипных элементов — нейронов, иммитирующих одноименные клетки живых организмов.

Искусственный нейрон, так же, как и живой, состоит из синапсов, связывающих его входы с ядром, которое осуществляет обработку входных сигналов. Связь с нейронами следующего слоя происходит через аксон. Каждый синапс имеет вес, определяющий насколько соответствующий вход нейрона влияет на его состояние, вычисляемое по формуле 1,

$$S = \sum_{i=1}^{n} x_i w_i \tag{1}$$

где n — число входов нейрона, x_i — значение i-го входа нейрона, w_i — вес i-го синапса. Значение аксона вычисляется как $f(S)$. Функция f называется активационной и может быть своей у каждого нейрона сети. На рисунке 5 показан пример нейросети, состоящей из трех слоев: входные нейроны (зеленый цвет), скрытые (синий) и выходной нейрон (желтый).

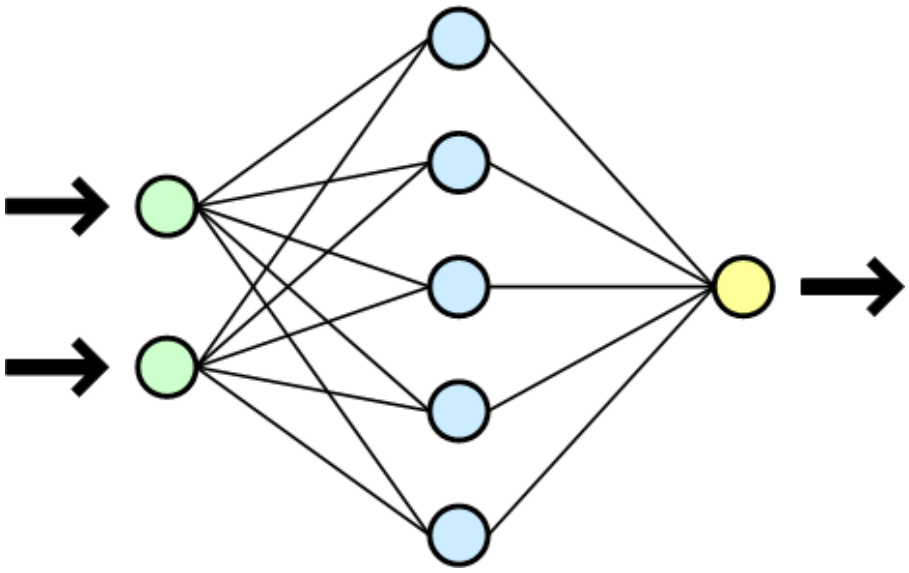


Рисунок 5 – Схема простой нейросети

Нейронные сети активно используются в задачах распознавания образов и предсказывания. Их часто применяют в области машинного обучения, называющейся глубокое обучение (deep learning), занимающейся моделированием высокоуровневых абстракций данных, используя системы со сложной архитектурой из множества нелинейных трансформаций. Под глубиной понимается максимальная длина между входным и выходным узлами графа вычислений модели.

2.1 Нейронные сети обратного распространения

Алгоритм обратного распространения ошибки (back propagation) является одним из методов обучения многослойных нейронных сетей прямого распространения, называемых также многослойными персептронами. Данный подход предполагает два прохода по всем слоям сети: прямой, при котором вычисляется реакция на заданный входной образ, и обратный, редактирующий синаптические веса с целью максимального приближения выходного сигнала сети к желаемому.

Рассмотрим работу алгоритма на примере обучения нейронной сети, представленной на рисунке 6.

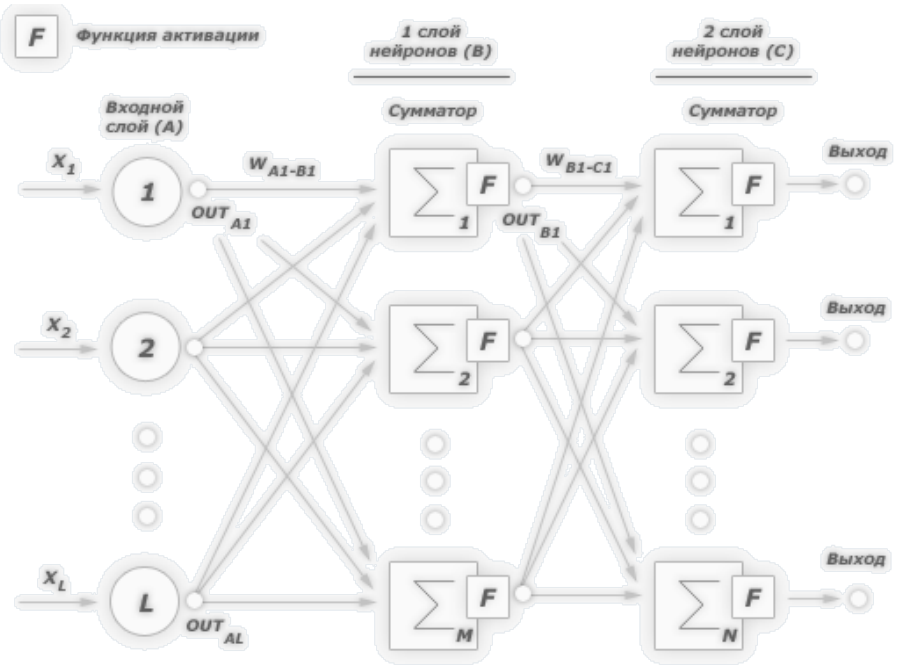


Рисунок 6 – Пример двухслойного персептрона

В качестве активационной, используется сигмоидальная логистическая функция (2), где α — параметром наклона, изменяя который можно управлять крутизной графика. Сигмоид нормирует входной сигнал, возвращая значение из диапазона от нуля до единицы. Для алгоритма обратного распространения ошибки от активационной функции требуется только выполнения условия всюду дифференцируемости. Сигмоид удовлетворяет этому требованию, а его дополнительное преимущество состоит в автоматическом контроле усиления: для слабых сигналов кривая вход-выход имеет сильный наклон, дающий большое усиление, но когда величина сигнала становится больше, усиление падает. Таким образом, большие сигналы воспринимаются сетью без насыщения, а слабые проходят без чрезмерного ослабления.

$$f(x) = \frac{1}{1 + e^{-\alpha x}} \tag{2}$$

Определим понятие обучающей пары — это два вектора, один из которых задает входной сигнал для нейронной сети, а второй отвечает за требуемый выход. Обучение происходит на множестве обучающих пар, его целью является такая настройка синаптических весов, чтобы суммарное отклонение (функция ошибки) от желаемого результата было минимально. Ошибку нейронной сети для каждого теста обучающей выборки можно вычислить по методу наименьших квадратов (3),

$$E(w) = \frac{1}{2} \sum_{i=1}^{p} (y_i - d_i)^2 \tag{3}$$

где y_i — значение i-го выхода нейросети, d_i — ожидаемое значение, p — число нейронов в выходном слое.

Таким образом, ставится задача многомерной минимизации, для решения которой используется метод градиентного спуска. На основании каждой обучающей пары происходит корректировка синаптических весов сети w_{ij} на некоторое число 4,

$$\Delta w_{ij} = -\eta \frac{\delta E}{\delta w_{ij}} \tag{4}$$

где $0 < \eta < 1$ — множитель, задающий скорость движения вдоль направления аниградиента.

2.2 Сверточные нейронные сети

Используемые обозначения

ANN — Artificial Neural Network, искуственная нейронная сеть (тоже самое что и NN)
Back propagation — алгоритм обратного распространения ошибки обучения многослойных нейронных сетей
CART — Classification And Regression Trees, алгоритм для построения бинарного дерева решений
Data mining — автоматические методы интеллектуального анализа
Decision tree — деревья принятия решений
Deep learning — область машинного обучения, занимающаяся моделированием высокоуровневых абстракций данных, используя системы со сложной архитектурой из множества нелинейных трансформаций
Fine tuning — алгоритмы частичной корректировки коэффициентов модели при ее дообучении
Information extraction — методы автоматического выделения информации
Logistic function — сигмоидальная функция
Logistic neuron — нейрон с сигмоидальной активационной функцией
ML — Machine Learning, машинное обучение
NN — Neural Network, нейронная сеть
Pruning — отсечение веток дерева решений, алгоритм улучшения качества
OCR — Optical Character Recognition, задача распознавания рукописного текста
Overfitting — переобучение
Random forest — комитета решающих деревьев для задач классификации и регрессии
ROI — Region Of Interest

Список литературы

[1] *Quinlan, J. R.* Decision trees and decision-making / J. R. Quinlan // *Systems, Man and Cybernetics, IEEE Transactions on.* — 1990. — Vol. 20, no. 2. — Pp. 339–346.

[2] Top 10 algorithms in data mining / X. Wu, V. Kumar, J. R. Quinlan et al. // *Knowledge and Information Systems.* — 2008. — Vol. 14, no. 1. — Pp. 1–37.

[3] *Fisher, R. A.* The use of multiple measurements in taxonomic problems / R. A. Fisher // *Annals of eugenics.* — 1936. — Vol. 7, no. 2. — Pp. 179–188.

[4] *Breiman, L.* Random forests / L. Breiman // *Machine learning.* — 2001. — Vol. 45, no. 1. — Pp. 5–32.

[5] *Niculescu-Mizil, A.* An empirical comparison of supervised learning algorithms using different performance metrics: Tech. rep. / A. Niculescu-Mizil, R. Caruana: Cornell University, 2005.

[6] *Bosch, A.* Image classification using random forests and ferns / A. Bosch, A. Zisserman, X. Munoz // Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on / IEEE. — 2007. — Pp. 1–8.