

Оглавление

[Введение 2](#_Toc515123284)

[1 Машинное обучение 3](#_Toc515123285)

[1.1 Введение в машинное обучение 3](#_Toc515123286)

[1.2 Общие принципы машинного обучения 3](#_Toc515123287)

[2 Обзор средств машинного обучения языка python 6](#_Toc515123288)

[2.1 Преимущества python 6](#_Toc515123289)

[2.2 Краткий обзор основных библиотек 6](#_Toc515123290)

[3 Построение классифицирующей нейронной сети 9](#_Toc515123291)

[3.1 Реализация персептрона на Python 9](#_Toc515123292)

[3.2 Тренировка персептронной модели 10](#_Toc515123293)

[3.3 Алгоритм случайные леса 13](#_Toc515123294)

[3.4 Алгоритм k ближайших соседей 15](#_Toc515123295)

[Использованная литература 17](#_Toc515123296)

ВВЕДЕНИЕ

В предложенной курсовой работе рассматривается тема реализация алгоритмов классификации с помощью средств языка Python. В настоящее время данная тема очень актуальна, что подтверждается массой различных применений.

Объектом данной курсовой работы является совокупность знаний об построение исследуемых алгоритмов, и анализ прикладного программного обеспечения для применения на практике.

Предметом исследования выступает механизм реализации, классифицирующей модели средствами языка Python.

Целью работы является реализация модели персептрона, а также реализация некоторых алгоритмов классификации.

Задачей является изучение алгоритмов и их реализация.

Для изучения данной темы использовались такие методы как сравнительной анализ теоритического материала, метод аналогии при изучении машинного обучения, а также моделирование нейронной сети.

В курсовой работе предлагается реализация электронного дидактического ресурса, который позволяет изучить понятие задачи классификации и алгоритмов построения, а также предлагается реализация построения этих алгоритмов.

1 Машинное обучение

1.1 Введение в машинное обучение

Машинное обучение ­ технология, которая строит саму себя. Это новое явление в нашем мире [2]. Во второй половине ХХ века машинное обучение развилось в подобласть искусственного интеллекта, которая охватывала разработку самообучающихся алгоритмов. Эти алгоритмы разрабатываются с целью обработки данных и выполнения различных прогнозов более простыми способами чем построение модели в ручном режиме [1].

Полезнее рассматривать машинное обучение как средство созданий моделей данных. Создается математическая модель для исследования данных с последующим постепенным улучшение качества прогнозных моделей и принятие решений, управляемых данными.

Задачи «обучения» начинаются с появлением у этих моделей настраиваемых параметров, которые можно приспособить для отражения наблюдаемых данных, таким образом, программа обучается на данных  [4].

Наиболее успешные алгоритмы машинного обучения являются те, которые автоматизируют процессы принятия решений путем обобщения известных примеров. В этих методах, пользователь предоставляет алгоритму пары объект-ответ, а алгоритм находит способ получения ответа по объекту. В частности, алгоритм способен выдать ответ для объекта, которого он никогда не видел раньше, без какой-либо помощи [2].

Каждый год в мире появляются сотни новых алгоритмов с обучением, но все они основаны на небольшом наборе фундаментальных идей [4].

1.2 Общие принципы машинного обучения

Существует три типа машинного обучения: обучение с учителем (контролируемое), обучение без учителя (неконтролируемое, или спонтанное) и обучение с подкреплением.

­ Машинное обучение с учителем

Основная задача обучения с учителем (supervised learning) состоит в том, чтобы на маркированных тренировочных данных извлечь модель, которая позволяет делать прогнозы о ранее не встречавшихся или будущих данных. Здесь термин «с учителем» относится к подмножеству образцов, в которых нужные выходные сигналы (метки) уже известны. [1]Оно разделяется далее на задачи классификации и задачи регрессии [2].

Задача классификации ­ это подкатегория методов машинного обучения с учителем, суть которой заключается в идентификации категориальных меток классов для новых экземпляров на основе предыдущих наблюдений1•

Метка класса представляет собой дискретное, неупорядоченное значение, которое может пониматься как принадлежность группе экземпляров. Извлеченная алгоритмом обучения с учителем прогнозная модель может присваивать новому, немаркированному экземпляру любую метку класса, которая была опреде­лена в тренировочном наборе данных.

Задача регрессии ­ это предсказание значений непрерывной целевой переменной (регрессионный анализ). Имеется несколько предикторных (объясняющих) переменных и непрерывная (результирующая) переменная отклика, и алгоритм пытается найти между этими переменными связь, которая позволит предсказывать результат.

­ Машинное обучение без учителя

Обучение без учителя (unsupervised learning) включает моделирование признаков набора данных без каких-либо меток. Эти модели включают такие за дачи, как кластеризация (clustering) и понижение размерности (dimensionality reduction) [2].

Кластеризация ­ это метод разведочного анализа данных, который позволяет организовать груду информации в содержательные подгруппы (кластеры), не имея никаких предварительных сведений о принадлежности группе. Каждый кластер, который может появиться во время анализа, обозначает группу объектов, которые обладают определенной степенью подобия и одновременно больше отличаются от объектов в других кластерах, поэтому кластеризацию также иногда называют «классификацией без учителя».

Еще одна подобласть обучения без учителя представлена методом снижения размерности. При этом методе метки или другая информация определяются исходя из структуры самого набора данных [2]. Смысл задачи

понижения размерности состоит в том, чтобы отбросить нерелевантные или избыточные признаки. Задача разбивается на две группы, отбрасывание признаков, выделение признаков [3].

Кроме того, существуют так называемые методы частичного обучения (semi­supervised learning), располагающиеся примерно посередине между машинным обучением с учителем и машинным обучением без учителя. Методы частичного обучения бывают полезны в случае наличия лишь неполных меток [2].

2 Обзор средств машинного обучения языка python

2.1 Преимущества python

Python один из самых популярных языков программирования для науки о данных. Он обладает огромным количеством полезных дополнительных библиотек. Учитывая, что производительность таких интерпретируемых языков, как Python, для вычислительноемких задач хуже, чем у языков программирования более низкого уровня, были разработаны дополнительные библиотеки, такие как NитРу и SciPy, которые опираются на низкоуровневые реализации на Foгtran и С для быстродействующих и векторизованных операций на многомерных массивах.

Для выполнения задач программирования машинного обучения следует обращаться к библиотеке scikit-learn, которая на сегодня является одной из самых популярных и доступных библиотек машинного обучения с открытым исходным кодом [1].

В Python есть библиотеки для загрузки данных, визуализации, статистических вычислений, обработки естественного языка, обработки изображений и многого другого. Этот обширный набор инструментов предлагает специалистам по работе с данными (data scientists) большой набор инструментов общего и специального назначения. Одним из основных преимуществ использования Python является возможность напрямую работать с программным кодом с помощью терминала или других инструментов типа Jupyter Notebook [5].

2.2 Краткий обзор основных библиотек

Библиотека NumPy ­ это один из основных пакетов для научных вычислений в Python. Он содержит функциональные возможности для работы с многомерными массивами, высокоуровневыми математическими функциями (операции линейной алгебры, преобразование Фурье, генератор псевдослучайных чисел). В scikit-learn массив NumPy – это основная структура данных. scikit-learn принимает данные в виде массивов NumPy [5].

Pandas ­ более новый пакет, надстройка над библиотекой NumPy, обеспечивающий реализацию класса DataFrame. Объекты DataFrame ­ многомерные массивы с метками для строк и столбцов, а также зачастую с неоднородным типом данных и/или пропущенными данными [2].

Библиотека scikit-learn это проект с открытым исходным кодом, это означает, что его можно свободно использовать и распространять, и любой человек может легко получить исходный код. Проект scikit-learn постоянно развивается и совершенствуется. Он содержит ряд современных алгоритмов машинного обучения, документацию по каждому алгоритму [1].

Лучше всего представлять используемые в библиотеке Scikit-Learn данные в виде таблиц. Простейшая таблица ­ двумерная сетка данных, в которой строки представляют отдельные элементы набора данных, а столбцы некоторые атрибуты, связанные с каждым из этих элементов. Примером может служить рассматриваемый в данной работе набор данных Iris проанализированный Рональдом Фишером в 1936 году.

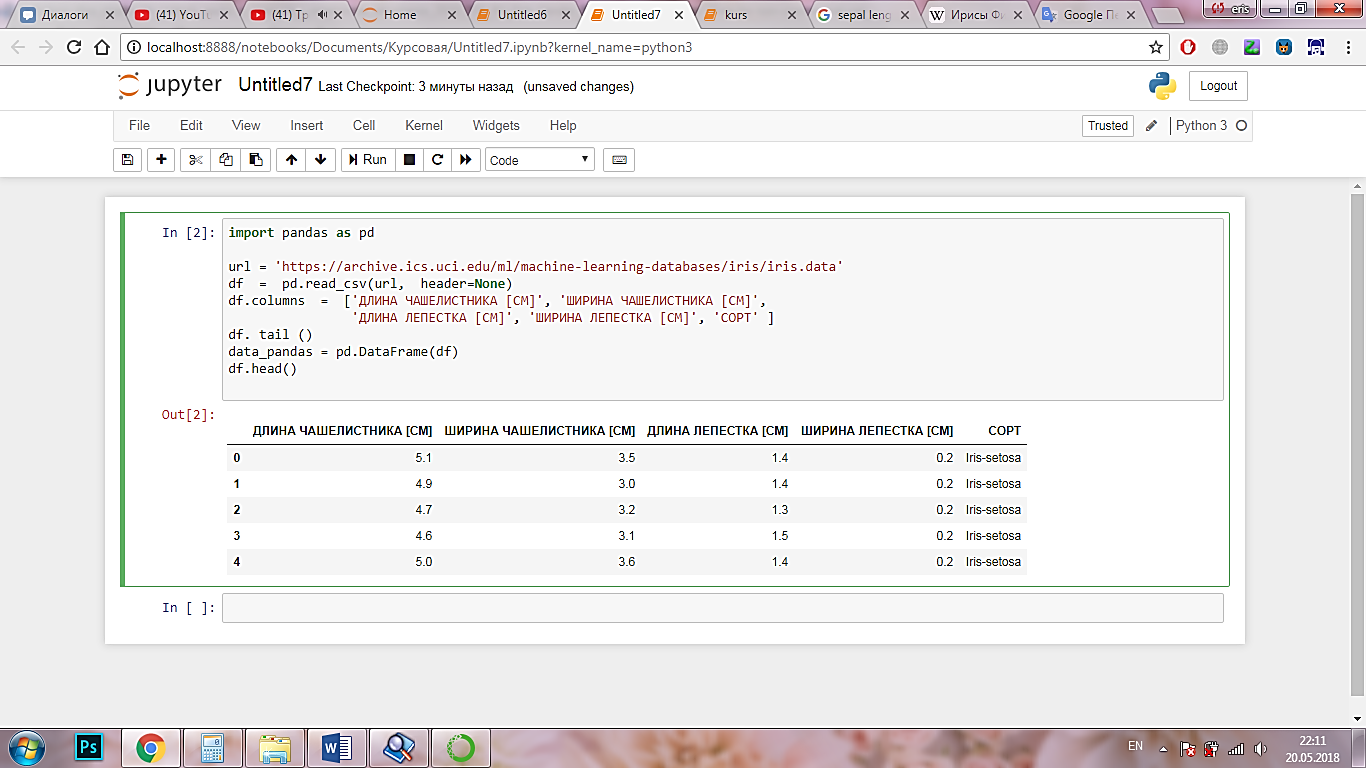


Рисунок 1 ­ Вызов таблицы набора Iris

Библиотека SciPy – это набор функций для научных вычислений в Python. Она содержит процедуры линейной алгебры, математическую оптимизацию функций, обработку сигналов, специальные математические функции и статистические функции. Scikit-learn использует набор функций SciPy для реализации своих алгоритмов.

Matplotlib – это основная библиотека для построения научных графиков в Python. Она включает функции для создания высококачественных визуализаций типа линейных диаграмм, гистограмм, диаграмм разброса.

Pandas – библиотека Python для обработки и анализа данных. Она построена на основе структуры данных. DataFrame библиотеки pandas представляет собой таблицу, похожую на электронную таблицу Excel. В отличие от NumPy, который требует, чтобы все записи в массиве были одного и того же типа, в pandas каждый столбец может иметь отдельный тип (например, целые числа, даты, числа с плавающей точкой и строки).

Аnaconda дистрибутив Python, предназначенный для крупномасштабной обработки данных, прогнозной аналитики и научных вычислений. Anaconda уже включает NumPy, SciPy, matplotlib, pandas, IPython, Jupyter Notebook и scikit-learn.

3 Построение классифицирующей нейронной сети

3.1 Реализация персептрона на Python

Работу с алгоритмами машинного обучения начнем реализации персептрона Розенблатта. Применяя объектно-ориентированный подход, определим интерфейс персептрона как класс языка Python.

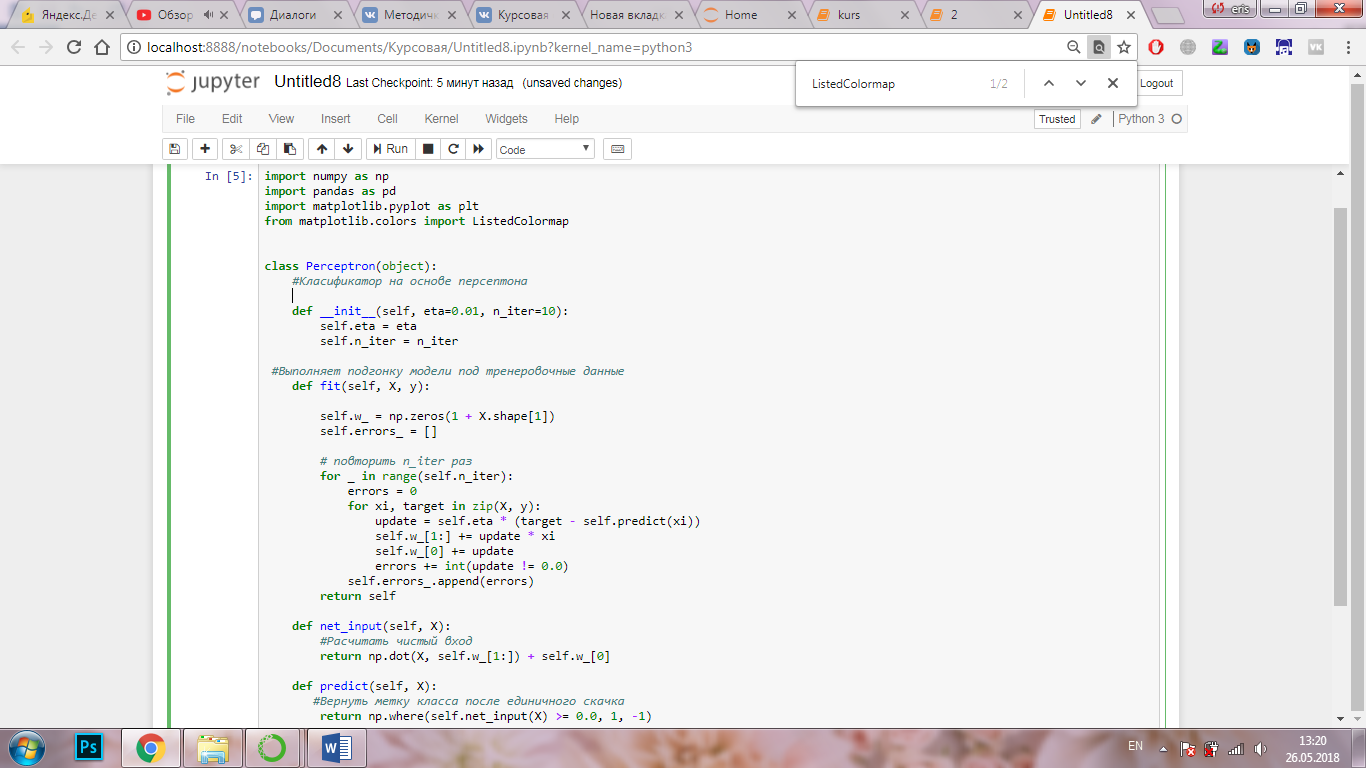


Рисунок 2 ­ Реализация персептрона

Новые объекты персептрона смогут обучаться на данных при помощи метода fit (выполнить подгонку модели) и делать прогнозы при помощи метода predict (распознать). Задается тем обучения eta и число эпох n\_iter (проходов), по тренировочному набору. При помощи метода для выполнения подгонки fit инициализируются веса в атрибуте self.w\_ нулевым вектором , где m число признаков в наборе данных, в нулевой позиции которого учитываются веса (порог).

После инициализации весов метод fit в цикле просматривает отдельные образцы тренировочного набора и обновляет веса согласно правилу обучения персептрона. Метки классов распознаются методом predict, который также вызывается в методе fit. Метод fit может применяться и для распознавания меток классов новых данных уже после подгонки модели. Кроме того, в списке self.errors\_ собираем число ошибочных распознаваний классов в каждой эпохе, для анализа алгоритма.

3.2 Тренировка персептронной модели

На наборе данных Ирисы Фишера реализуем классифицирующий алгоритм. Этот набор данных содержит данные измерений 150 цветков ириса трех видов: ирис щетинистый (lris setosa), ирис виргинский (Iris viгginica) и ирис разноцветный (Iris veгsicoloг) по четырем характеристикам (признакам): длина чашелистика, ширина чашелистика, длина лепестка, ширина лепестка. Согласно рисунку 3 можно увидеть зависимость пар данных.

Реализация модели будет строится на описанного в библиотеке scikit-learn персептона, аналогичного описанному ранее.

Выясним значение математической зависимости между признаками. Наглядно ее можно показать с помощью тепловой карты зависимости признаков. Данный метод выдает Значения коэффициента корреляции. :Согласно рисунку 3 наиболее высокая зависимость между переменными «ширина лепестка» и «длина лепестка» 0.96.

Следовательно, матрицу признаков Х составят третий и четвертый столбцы, а  вектору *у* ­ метки классов, которые соответствуют видам цветков (0,1,2).

Для оценки алгоритма произвольным образом с помощью метода train\_test\_split из модуля model\_selection библиотеки scikit-learn разделим массивы Х и у на тестовые данные в размере 30% от общего объема (45 образцов) и тренировочные данные в размере 70% (105 образцов).



Рисунок 3 ­ Зависимость признаков



Рисунок 4 ­ Тепловая карта зависимости признаков

Многие алгоритмы машинного обучения и оптимизации в целях улучшения качества также требуют выполнения масштабирования признаков. Выполним стандартизацию дспомощью StandardScaler из модуля preprocessing библиотеки scikit-learn. Этот метод вычисляет параметры µ (эмпирическое среднее) и S (стандартное отклонение) для каждой размерности признаков из тренировочных данных.

Подобрав темп обучения так, что бы модель была достаточно натренированной. После проведения тренеровки видно, что персептрон ошибочно классифицирует 4 из 45 образцов цветков. Ошибка классификации на тестовом наборе данных составляет 0.089.

В библиотеке scikit-leaгn также реализовано большое разнообразие различных метрик оценки качества работы, которые доступны благодаря модулю metrics. Возможно вычислить оценку верности классификации с использованием персептрона на тестовом наборе.

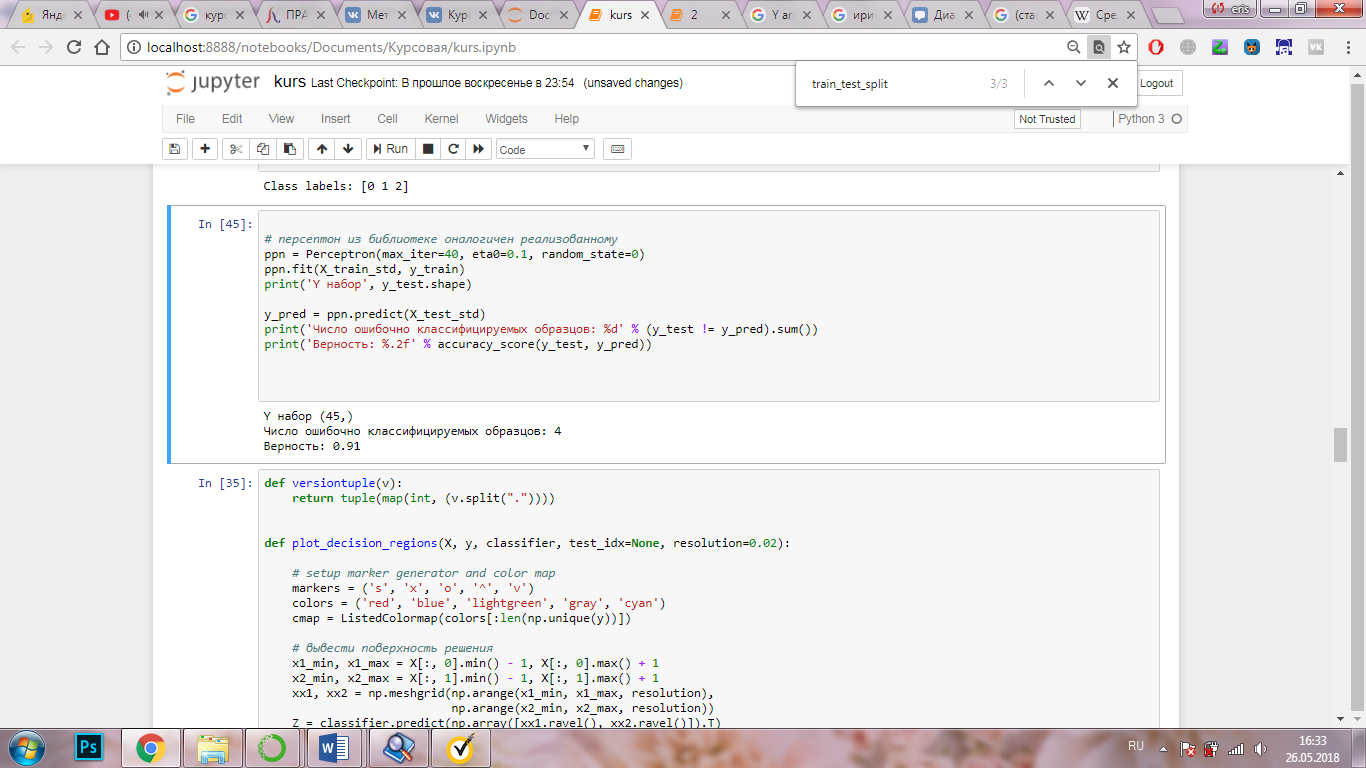


Рисунок 5 ­ Тренировки модели

Применяя алгоритм к тестовым данным видно, что полностью разделить три класса цветков линейной границей решения не получается, что видно согласно рисунку 6.

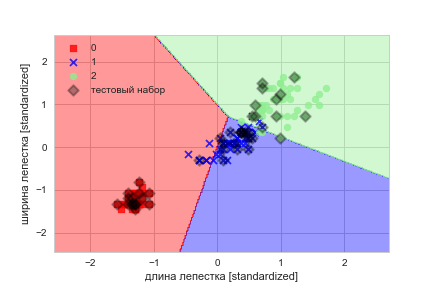


Рисунок 6 ­ График областей решений

Минусом алгоритма обучения персептрона является то, что он никогда не сходится на наборах, данных, которые не полностью линейно разделимы.

3.3 Алгоритм случайные леса

Рассмотрим другой алгоритм обучения случайные леса (гandom foгests). Он обладает хорошая классификационная способность и масштабируемость. Алгоритм метода:

1) Случайным образом отобрать из тренировочного набора данных n образцов с возвратом;

2) Вырастить дерево решений из выборки. В каждом узле:

а) случайным образом отобрать d признаков без возврата

б) расщепить узел, используя признак, который обеспечивает наилучшее рас­щепление согласно целевой функции.

в) повторить шаги 1 и 2 k число раз;

г) для назначения метки класса агрегировать прогноз из каждого дерева на основе большинства голосов.

Реализацию классификатора на основе случайного леса RandomForestClassifier в библиотеке scikit-learn, размер выборки выбирается так, чтобы он был эквивалентным числу образцов в исходном тренировочном наборе, что обычно обеспечивает хороший компромисс между смещением и дисперсией. В библиотеке scikit-learn уже существует реализация, которую можно использовать.

При количестве деревьев равное 10, точность алгоритма достаточно велика.

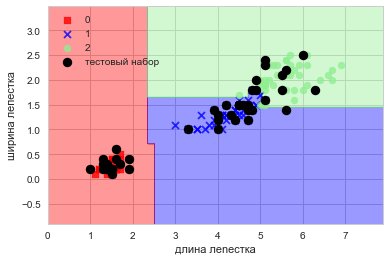


Рисунок 7 ­ График областей решения методом гandom foгests из 10 деревьев

Если увеличить количество деревьев, то точность алгоритма значительно увеличится.

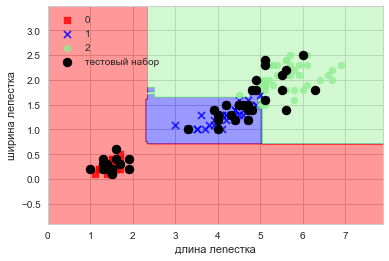


Рисунок 8 ­ График областей решения методом гandom foгests из 50 деревьев

3.4 Алгоритм k ближайших соседей

Алгоритм k ближайших соседей ­ это классификатор на основе k ближайших соседей (k-nearest neighbor classifieг, KNN), который особенно интересен тем, что он существенно отличается от других алгоритмов обучения.

Алгоритм может быть резюмирован следующими шагами:

1. выбрать число k и метрику расстояния;
2. найти k ближайших соседей образца, который мы хотим классифицировать;
3. присвоить метку класса мажоритарным голосованием.

В тренировочном наборе данных k образцов находятся те, которые являются самыми близкими к классифицируемой точке. Метка класса новой точки данных затем определяется мажоритарным голосованием среди ее k ближайших соседей.

Основное преимущество такого подхода с запоминанием состоит в том, что классификатор немедленно адаптируется по мере сбора новых тренировочных данных. Минусом является вычислительная сложность классифицирования новых образцов. Кроме того, нет возможсти отбросить тренировочные образцы, поскольку нет тренировочного шага.

Правильный выбор числа k крайне важен для нахождения хорошего равновесия между переобучением и недообучением.

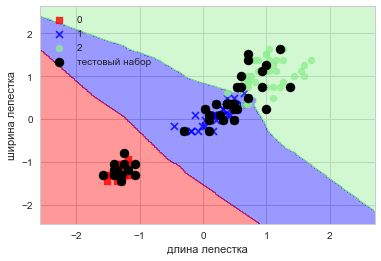


Рисунок 9 ­ График областей решения методом k ближайших соседей

ЗАКЛЮЧЕНИЕ

Цель курсовой работы – является реализация модели персептрона, и реализация некоторых алгоритмов классификации ­ достигнута.

В рамках курсовой работы изучены алгоритмы построения модели машинного обучения задачи классификации. Реализованы такие алгоритмы классификации как: персептроная модель, алгоритм случайного леса и алгоритм k ближайших соседей.

Важно отметить, что используемый алгоритм классификации напрямую завит от исходных данных. Если данные линейно неразделимы, то применение персептонной модели недопустимо.

В курсовой работе предлагается исследование библиотек, предназначенных для машинного обучения языка python NumPy, scikit-learn. Проведено успешное построение нечеткой нейронной сети. Данные библиотеки имеют открытый код, что позволяет рассматривать аппарат построения математической модели как средство работы с данными.

Также возможно использование данного теоритического материала для дальнейших исследований в области машинного обучения, и применения его для создания программного обеспечения.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Рашка С., Python и машинное обучение / Рашка С., пер. с англ. А. В. Логунова. М.: ДМК Пресс, 2017 ­ 418 с.
2. Вандер Плас Дж., Python для сложных задач: наука о данных и машинное обучение. / Вандер Плас Дж. СПб.: Питер, 2018 ­ 576 с.
3. Коэльо Л., Построение снстем машишюrо обучения нa языке Pytho/ Коэльо Луне Перо, Вилл Ричарт. М.: ДМК Пресс, 2016 ­ 302 с.
4. Домингос П., Верховный алгоритм: как машинное обучение изменит наш мир / Педро Домингос ; пер. с англ. В. Горохова. М. : Манн, Иванов и Фербер, 2016 ­ 336 с.
5. А.Мюллер, Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными / А.Мюллер, С.Гвидо. СПб.: ООО “Альфа книга” , 2017 ­ 480 с.
6. Дэви Силен, Основы Data Science и Big Data. Python и наука о данных / Дэви Силен, Арно Мейсман. СПб.: Питер, 2017 ­ 336с