Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Базовая кафедра геоинформационных систем

УТВЕРЖДАЮ
Заведующий кафедрой

_____ В. И. Харук
подпись
«___» _____ 2018 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.02 - Информационные системы и технологии

Цифровая карта объектов размещения отходов производства и потребления Красноярской агломерации

Руководитель _____ к.т.н., доц. каф. Б-ГИС, А.С. Савельев
подпись, дата

Выпускник _____ В.А. Шушарин
подпись, дата

Нормоконтролер _____ Е.В. Федотова
подпись, дата

Красноярск 2018

Федеральное государственное автономное
образовательное учреждение высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
Базовая кафедра геоинформационных систем

УТВЕРЖДАЮ
Заведующий кафедрой

_____ В. И. Харук
подпись
«___» _____ 2018 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту Шушарину Владимиру Андреевичу

Группа КИ-14-14Б Направление (специальность): 09.03.02

Информационные системы и технологии

Тема выпускной квалификационной работы: «Цифровая карта объектов размещения отходов производства и потребления Красноярской агломерации»

Утверждена приказом по университету №_____от_____2018 г.

Руководитель ВКР: Савельев А.С., доцент кафедры Б-ГИС, канд. техн. наук

Исходные данные для ВКР: документация и руководство пользователя Python, космические данные Landsat-8 и Sentinel-2A, кадастровая карта размещения отходов.

Перечень разделов ВКР: обзор используемого программного обеспечения, языка программирования, библиотек, спутниковых данных и алгоритмов; реализация и тестирование алгоритмов классификации; создание цифровой карты

Перечень графического материала: слайды презентации

Руководитель ВКР _____ А.С. Савельев
подпись

Задание принял к исполнению _____ В.А. Шушарин
подпись

«___» _____ 2018 г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Цифровая карта объектов размещения отходов производства и потребления» содержит 68 страниц текстового документа, 1 приложение, 31 использованный источник.

LANDSAT-8, SENTINEL-2A, PYTHON, КОСМИЧЕСКИЕ ДАННЫЕ, АТМОСФЕРНАЯ КОРРЕКЦИЯ, SACP, АЛГОРИТМ КЛАССИФИКАЦИИ, КОВАРИАЦИОННАЯ МАТРИЦА НЕСАНКЦИОНИРОВАННЫЕ СВАЛКИ.

В работе анализировались данные дистанционного зондирования Земли территории города Красноярска и его окрестностей.

Цель работы: создание цифровой карты, объектов размещения отходов производства и потребления.

Задачи:

- подготовить к работе космические снимки;

- разработать алгоритмы, классифицирующие объекты на территории Красноярской агломерации на космических изображениях;

- реализовать алгоритмы на языке Python, протестировать разработанные алгоритмы на различных наборах космических изображений;

- оценить эффективность алгоритмов классификации;

- создать цифровую карту, объектов размещения отходов производства и потребления.

В результате были разработаны алгоритмы, классифицирующие объекты на территории Красноярской агломерации, с использованием космических снимков спутников Landsat-8 и Sentinel-2A. Алгоритмы были протестированы и произведена оценка эффективности их работы. На основе результатов классификации была создана тематическая карта объектов размещения отходов производства и потребления Красноярской агломерации.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время является актуальной проблема несанкционированных свалок в окрестностях населённых пунктов. Человеческая деятельность оказывает глобальное влияние на биосферу по сравнению с началом XX века: вырубка леса, осушение рек и озёр, распашка степей, опустынивание земли, строительство зданий, дорог, плотин, загрязнение свалками и т. д. Отсутствие специально отведённых территорий под различные виды отходов ведёт к образованию несанкционированных свалок. Вследствие этого возникает необходимость определения этих территорий на картах. Для решения данной проблемы необходимо объехать все дороги близлежащие к населённым пунктам, для проверки и нанесения на карту несанкционированных свалок, либо использовать какие-либо воздушные суда или аппараты для облёта больших территорий близ населённых пунктов. К сожалению, и то, и другое занимает немалое количество времени, а также требуют определённых затрат ресурсов. В качестве решения данной проблемы было решено написать программный модуль, который позволяет, используя космические данные, полученные со спутников дистанционного зондирования Земли (ДЗЗ), идентифицировать территории с несанкционированными свалками. На основе полученных результатов работы программного модуля составляется цифровая тематическая карта объектов распределения отходов производства и потребления.

Для решения данной проблемы используется Геоинформационная система Quantum GIS, с встроенным в неё языком программирования Python

Цель работы: создание цифровой карты объектов размещения отходов производства и потребления.

Задачи:

- произвести загрузку и предварительную обработку космических данных;

- разработать алгоритмы, классифицирующие объекты на территории города Красноярска на космических изображениях;

- реализовать эти алгоритмы на языке Python, а также протестировать разработанные программы на различных наборах космических изображений;

- оценить эффективность алгоритмов классификации;

- создать цифровую карту отходов производства и потребления.

1 Обзор используемого программного обеспечения, языка программирования, библиотек, спутниковых данных и алгоритмов

1.1 Геоинформационная система QGIS

На данный момент существует множество различных Геоинформационных систем для создания цифровых тематических карт. Одной из них является Quantum GIS – свободная кроссплатформенная геоинформационная система. Она позволяет решать сложные задачи и управлять, отображать, редактировать и анализировать географическими данными. А также допускает использование различных инструментов в виде встроенных модулей. Панель инструментов предоставляет широкие возможности по обработке растровых и векторных типов данных. Интерфейс пользователя предоставляет множество полезных инструментов: идентификация и выбор объектов, редактирование, просмотр и поиск атрибутов, перепроецирование «на лету», компоновщик карт, изменение символики векторных и растровых слоёв. Данная ГИС поддерживает множество векторных и растровых форматов, имеется возможность создания, редактирования и экспорта пространственных данных с использованием инструментов оцифровки, модуля привязки растров и инструментов GPS. Возможен пространственный анализ с использованием модулей растровой алгебры, морфометрического анализа, гидрологического моделирования, сетевого анализа и т. д.

В геоинформационную систему QGIS встроены привязки, которые реализуют практически весь функционал QGIS на языке программирования Python а также предоставляет библиотеки, используемые при создании модулей. Эти модули могут адаптировать Геоинформационную системы под особые потребности, которые зависят от задач, решаемых пользователем.

При запуске Quantum GIS пользователю предстаёт графический интерфейс, который делится на 5 компонент, представленных на рисунке 1.

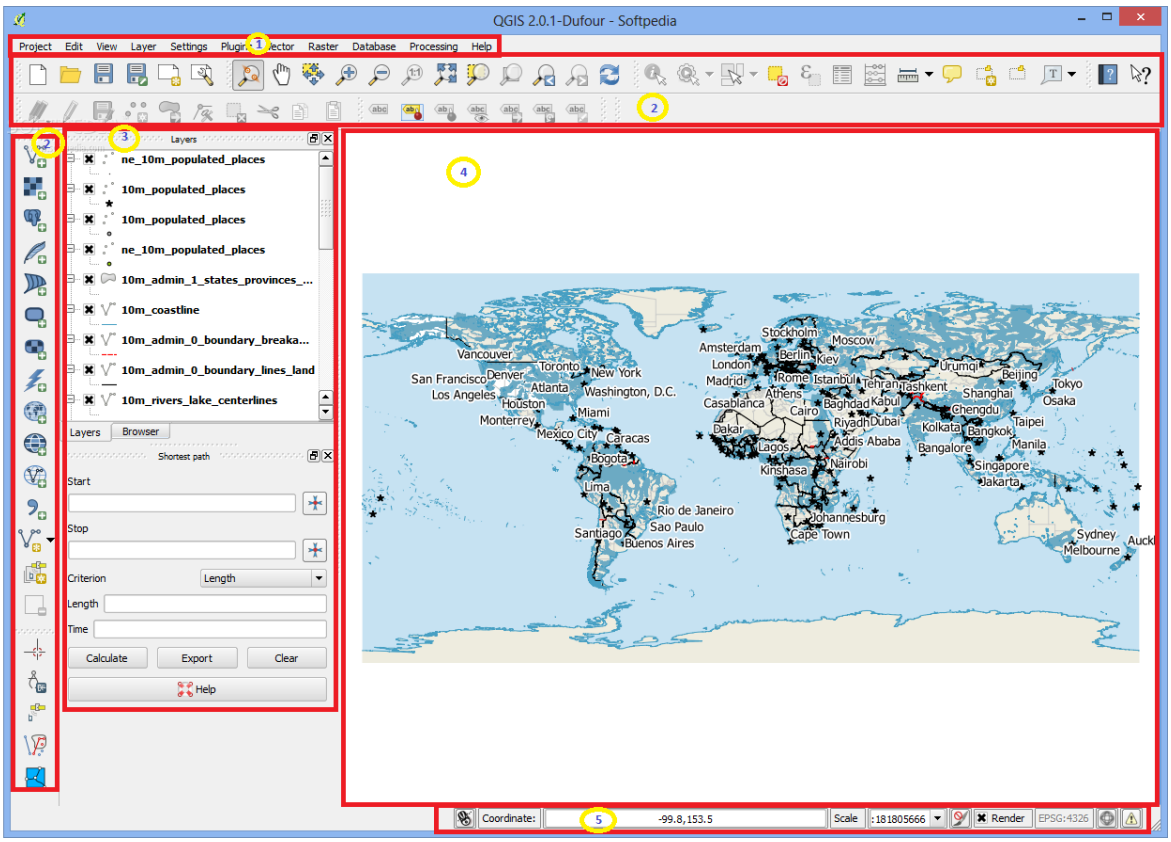


Рисунок 1 – Изображение графического интерфейса пользователя Quantum GIS. 1 - главное меню; 2 - набор инструментов; 3 - дополнительные панели; 4 - область карты; 5 - строка состояния.

Строка меню имеет иерархическую структуру и обеспечивает доступ к различным функциям QGIS.

Для выполнения задачи классификации объектов на карте существуют различные модули и расширения. Одним из таких расширений является Semi-automatic classification plugin. С его помощью можно загружать космические данные, сделанные с космических аппаратов: Landsat, MODIS, Sentinel и ASTER. Данный модуль позволяет контролировать классификацию объектов на изображениях дистанционного зондирования. Он предоставляет инструменты для загрузки, предварительной обработки и постобработки изображений. Этот модуль классифицирует объекты посредством следующих алгоритмов: maximum likelihood (максимального правдоподобия), minimum distance (минимального расстояния), spectral angle mapping (спектрального угла отображения).

1.2 Язык программирования Python

Для разработки и реализации алгоритмов классификации объектов нужно использовать встроенные в Quantum GIS возможности для программирования, таким образом, улучшив функциональность ГИС. Такой возможностью становится встроенный в QGIS язык программирования Python.

Это язык программирования высокого уровня с динамической типизацией, автоматическим управлением памятью и удобными высокоуровневыми структурами данных, такими как словари (хеш-таблицы), списки, кортежи. Python является в свою очередь интерпретируемым языком программирования, его интерпретаторы реализованы практически на всех платформах и операционных системах. Перенос программного кода из Linux в Windows и обратно заключается в простом копировании программного кода с одной машины на другую. В языке программирования Python поддерживаются классы, модули, обработка исключений, многопоточные вычисления, а также несколько парадигм программирования: структурное, функциональное, объектно-ориентированное и аспектно-ориентированное.

Синтаксис языка выразителен и прост в понимании и изучении. Он же языка исключает громоздкие конструкции, классов, методов или циклов и заменяет их наглядными пробелами и табуляцией. Это упрощает отслеживание прогресса создания программы, её отладку и доработку. Язык обладает высокой удобочитаемостью кода и большим множеством библиотек.

Консоль Python – встроенный модуль в QGIS, одновременно интерпретатор языка. С использованием консоли можно выполнять, как различные команды, не затрагивая при этом исходный код, так и выполнять лишь некоторые элементы программного кода, выполняющего какие-либо действия или функции.

1.3 Библиотеки Python.

1.3.1 Библиотека GDAL

Для работы с растровыми данными используются утилиты командной строки, входящие в состав библиотеки GDAL. На её основе разработан модуль GDAL Tools, предоставляющий графический интерфейс для работы с этой библиотекой.

Функциями библиотеки GDAL являются: перепроецирование, совмещение растров, обрезка растров, геотрансформация, объединение разных форматов растров, получение информации о количестве каналов растров, извлечение данных и метаданных и т. д. Она же может выполнять чтение из разных растровых форматов, таких как jpeg и GeoTTiff. При работе с пространственными данными и дистанционным зондированием, главным форматом является GeoTTiff, потому что в этом формате присутствует геопривязка и трансформация проекции, также можно накладывать на него другие изображения.

Библиотека GDAL очень полезна при работе с космическими данными, сделанными со съёмочной аппаратуры, расположенной на спутниках Земли. Являясь бесплатной и имея открытый исходный код, библиотека широко распространена в кругах ГИС пользователей.

1.3.2 Библиотека NumPy

Библиотека NumPy – расширение языка Python, добавляющее поддержку больших многомерных массивов и матриц, дополнительно присутствует множество высокоуровневых математических функций для оперирования массивами. Создание массивов одномерных, двухмерных, n-мерных, атрибутов, базовых операций, сложение, вычитание и умножение массивов, поэлементное применение функций и итерирование. Библиотека даёт возможность проводить

манипуляции с преобразованием форм массивов, объединение массивов из нескольких или же, наоборот, разбиение крупного массива на более мелкие массивы.

Математические алгоритмы, реализованные на интерпретируемых языках, часто работают медленнее тех же алгоритмов, реализованных на компилируемых языках. Библиотека NumPy предоставляет реализации вычислительных алгоритмов (в виде функций и операторов), оптимизированные для работы с многомерными массивами. В результате любой алгоритм, который может быть выражен в виде последовательности операций над массивами (матрицами) и реализованный с использованием NumPy, работает так же быстро, как эквивалентный код, выполняемый в MATLAB.

Основным объектом NumPy является однородный многомерный массив. Это таблица элементов (обычно чисел, символов или строк), всех одного типа, индексированных последовательностями натуральных чисел. Под «многомерностью» массива понимается то, что у него может быть несколько измерений или осей.

1.3.3 Библиотека PIL

Библиотека PIL (Python Imaging Library) – библиотека языка python, предназначенная для работы с растровой графикой. Возможности:

- поддержка бинарных, полутоновых, индексированных, полноцветных и CMYK изображений;

- поддержка форматов BMP, JPEG, TIFF, PDF, PNG, GIF и других на чтение и запись, а также некоторых форматов только на чтение;

- преобразование изображений из одного формата в другой; правка изображений (масштабирование, рисование, матричные операции и т.д.)

PIL идеально подходит для архивных изображений и пакетной обработки приложений. Библиотека может использоваться для создания эскизов,

преобразования между форматами файлов, печати изображений и т. д. Библиотека позволяет идентифицировать и читать большое количество форматов, а также, поддерживает запись в форматах наиболее часто используемых для обмена и представления.

Библиотека Python Image Library включает в себя некоторые базовые функции обработки изображении: операции с точками, фильтрация и преобразование цветового контраста, изменение размера изображения, поворот и различные аффинные преобразования.

1.4 Спутниковые данные

Под спутниковыми данными или данными дистанционного зондирования Земли (ДЗЗ) понимаются растровые изображения участков земной поверхности, а также файлы с пространственными данными о снимке, прошедшие обработку и сделанные со съёмочной аппаратуры, расположенной на космических носителях: пилотируемые корабли, орбитальные станции и искусственные спутники Земли. Такие данные являются очень востребованными из-за своей актуальности, объективности, охвата большой территории и, в большинстве своём, доступности для приобретения как физическими, так и юридическими лицами.

На данный момент на орбиту Земли и других космических тел выведено около 1000 действующих искусственных спутников Земли, запущенные более чем 70 различными странами мира. Различают следующие типы спутников:

- астрономические спутники;
- спутники ДЗЗ;
- космические станции;
- метеорологические спутники;
- разведывательные спутники;
- навигационные спутники;
- спутники связи.

С каждым годом это количество растёт и за всё время, с начала покорения космического пространства человеком, орбита Земли наполнилась большим скоплением космического мусора.

Количество спутников ДЗЗ, действующими в данный момент является несколько десятков спутников. Космические данные используются для решения задач в различных отраслях: мониторинг пожароопасной обстановки, мониторинг сельскохозяйственных угодий, обнаружение незаконных вырубок и обнаружение незаконных построек в охраняемых зонах и т. д.

Съёмочная аппаратура, установленная на космическом носителе, ведёт съёмку по принципу активного зондирования. То есть она излучают электромагнитные сигналы с целью сканирования объектов, а затем сенсор имеет возможность обнаружить и измерить отражённые сигналы.

Исходя из того, какая съёмочная аппаратура установлена на космических носителях, космические данные обладают следующими свойствами:

Пространственное разрешение – величина, характеризующая размер наименьших объектов, различимых на изображении, сопоставимые с величиной единичного элемента разрешения или пикселя. Чем меньше значения разрешения, тем более чётко отображаются объекты на растровых изображениях. Изображения, полученные со спутников, обладают разрешением от километров до 50 сантиметров, что является самым лучшим пространственным разрешением на данный момент.

Радиометрическое разрешение – величина, которая определяет количество градаций значений цвета, что соответствует переходу от яркости абсолютно белого к абсолютно чёрному и выражается в количестве бит на пиксель. Чем больше число бит, тем больше градаций цвета: 6 бит – 64 градации, 8 бит – 256 градаций, 11 бит – 2048 градаций и т. д. В настоящий момент сенсоры спутников обладают радиометрическим разрешением 8 бит и более. Градации цвета позволяют лучше и более детально различать очень светлые и тёмные области космических снимков.

Спектральное разрешение – величина, указывающая на диапазон участка спектра электромагнитных волн, регистрируемый сенсором. Весь диапазон длин волн, регистрируемый спутниками ДЗЗ, делится на участки: радиоволны, тепловое излучение, ИК-излучение и видимый свет. Человеческий глаз способен воспринимать лишь видимое излучение в диапазоне от 400 до 750 нанометров (только цвета). Современные сенсоры различают в сотни раз превосходящие диапазоны длин волн и точно выявляют объекты и явления по заранее известным спектрограммам. Возможности сенсоров позволяют не только распознавать, что за объекты присутствуют на космических снимках, а также оценить их состояние в данный период времени. Например, используя ближний инфракрасный диапазон, оценивается состояние растительность, её вегетационный период. Использование тепловых снимков позволяет оценивать пожароопасную обстановку поверхности Земли.

Временное разрешение – величина, указывающая то, с какой периодичностью идёт съёмка одного и того же участка Земли. Периодичность, с которой идёт повторная составляет период от нескольких дней до нескольких часов. Этот параметр зависит от того, по какой орбите движется спутник и находится ли на той же орбите спутники того же семейства. В зависимости от того, какая задача решается, временное разрешение может играть как важную, так и незначительную роль. При мониторинге чрезвычайных ситуаций важно использовать наиболее актуальные космические снимки, то есть чем чаще делаются снимки, тем легче оценивать текущую обстановку, например, природных бедствий. Однако, если снимки нужны для мониторинга явлений, не изменяющихся быстро и резко, например, вырубки лесов, то достаточно ограничиться снимками, делающимися раз в несколько дней.

Исходя из целей, для которых были запущены спутники ДЗЗ, они обладают различным пространственным, временным, радиометрическим и спектральным разрешением. Таким образом, ориентируясь на цели, которые нужно достигнуть и задачи, которые важно выполнить, пользователь должен использовать подходящие для этого космические снимки. Далее в работе

описываются космические носители, наиболее подходящие для достижения цели данной работы.

1.4.1 Постобработка космических данных

После того, как космический аппарат сделал снимок поверхности Земли, им были получены цифровые данные двумерного изображения в каком-либо спектральном диапазоне в виде матрицы, в данный момент, количество диапазонов, в которых ведётся съёмка, может составлять до десятков, в итоге получаются космические снимки одной и той же территории, сделанные в различных диапазонах. Дополнительно к файлу каждого снимка, прикрепляется файл с метаданными: время съёмки, формат, разрешение, статистические данные, информация о проекции и т. д. Для создания точных карт на этапе постобработки идёт также геотрансформация, для привязки изображения к проекции и определённым координатам.

Кроме этого, из-за характеристик съёмочных приборов, могут возникать дефекты в изображениях в виде полос, которые показаны на рисунке 2.

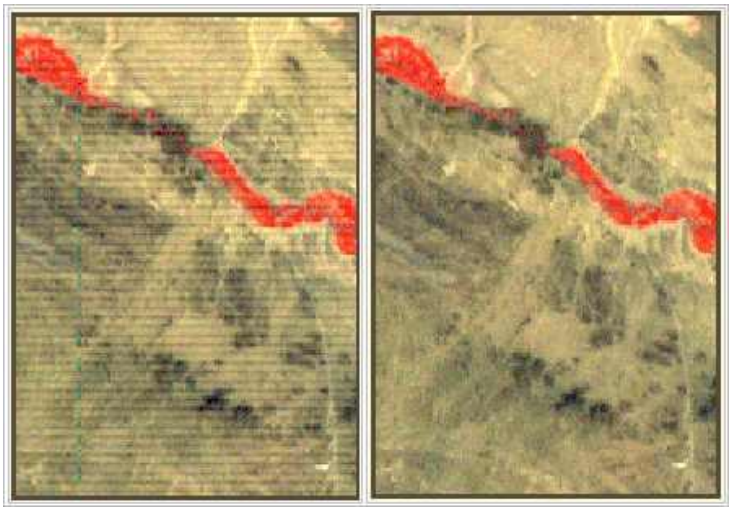


Рисунок 2 – Изображение до и после использования радиометрической коррекции

Человеческое восприятие такие полосы на изображении не будет принимать во внимание, он практически безошибочно определит, что они не являются реальным отображением поверхности Земли. Однако машина не обладает таким восприятием и будет принимать во внимание каждый пиксель изображения и сделает ошибочные выводы на их основе. Для этого и нужна радиометрическая коррекция, преобразующая дискретные уровни сигнала в их реальные значения. Существует два вида коррекции: статистическая и с использованием параметров и характеристик приборов. Статистическая коррекция позволяет корректировать изображения на основе их же статистических значений. А вторая корректирует изображение на основе корректировочных параметров самого прибора, на основе длительных испытаний.

Существует ещё один вид коррекции, которая исключает влияние атмосферы на космические снимки и называется атмосферной. На данные ДЗЗ могут повлиять различные факторы, связанные с атмосферой (рисунок 3).

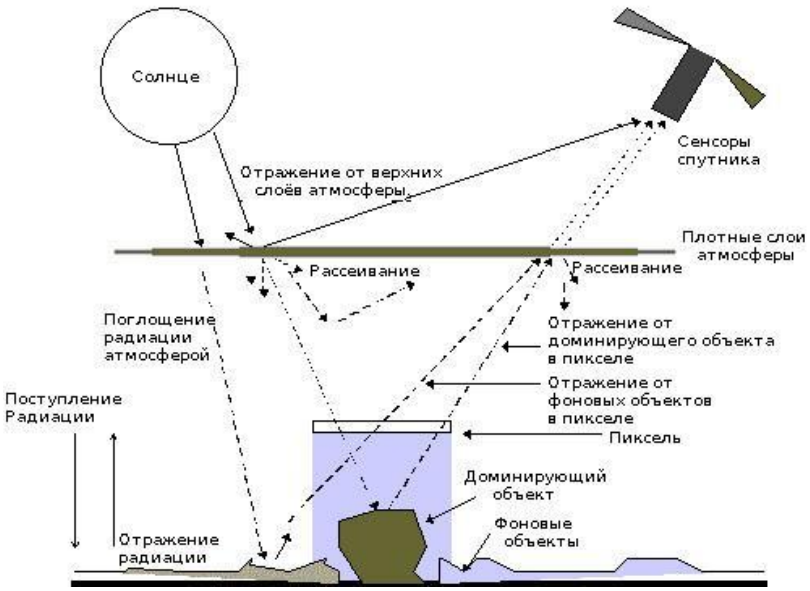


Рисунок 3 – Схема факторов, влияющих на попадание отражённой солнечной радиации на сенсоры спутника [16]

Атмосферная коррекция наиболее важный этап предварительной обработки космического снимка и, одновременно, это самая сложная задача. Она подразумевает под собой уменьшение влияния на снимок атмосферы и перевод значений радиации, дошедшей до сенсоров спутника, в значения реально отражённого от Земли спектрального излучения солнечного света. Влияние атмосферы на космический снимок проявляется в целом ряде факторов: угол падения и отражения солнечных лучей, прозрачность атмосферы, газовый фактор и дымка

Одним из алгоритмов атмосферной коррекции является алгоритм Dark Object Substraction (DOS), ориентированный на объекты скрытые тенью облаков и связанный с атмосферным рассеиванием. Суть алгоритма состоит в нахождении яркости однопроцентного тёмного объекта с последующей коррекцией минимума значений каждого пикселя изображения относительно спектральной яркости найденного объекта.

1.4.2 Спутники семейств Landsat и Sentinel

Для обработки и анализа поверхности Земли используются космические снимки, сделанные с помощью съёмочного оборудования, расположенного на спутниковых носителях. Одними из таких спутников, производящих ДЗЗ, являются спутники семейства Landsat и Sentinel.

Программа Landsat является американским проектом по получению спутниковых снимков поверхности Земли. Первый спутник этой серии, Landsat-1, был запущен в 1972 году. Последним спутником из этой серии на данный момент является Landsat-8. Он был запущен 11 февраля 2013 году и находился в режиме тестирования до 30 мая 2013 года. Спутник снимает около 700 сцен в день, с временным разрешением равным приблизительно 16 дням. Landsat-8 использует 2 различных сенсора, чтобы собирать данные в девяти коротковолновых диапазонах и в двух длинноволновых тепловых диапазонах (таблица 1).

Программа Sentinel предполагает реализацию пяти миссий под руководством Европейского космического агентства и включает последовательный запуск целой группы спутников в период с 2014 года по 2020 год. Каждая миссия включает два взаимодополняющих спутника, одной из таких миссий является Sentinel-2. Дата запуска спутников Sentinel-2A – 2013 год, Sentinel-2B – 2015 год. Их совместная работа проводит повторные съёмки каждые 5 дней на экваторе и каждые 2-3 дня в средних широтах. Sentinel-2 оснащён мультиспектральным сенсором для съёмок в ближней инфракрасной и коротковолновой инфракрасной зонах спектра, включающий в себя 13 спектральных каналов (таблица 1).

Таблица 1 – Спектральные каналы данных Landsat-8 и Sentinel-2

Band number	Landsat-8		Sentinel-2	
	спектральный диапазон, мкм	пространственное разрешение, м	спектральный диапазон, мкм	пространственное разрешение, м
1	0,433-0,453	30	0,433-0,453	60
2	0,45-0,515	30	0,455-0,52	10
3	0,525-0,6	30	0,54-0,575	10
4	0,63-0,68	30	0,65-0,68	10
5	0,845-0,885	30	0,7-0,715	20
6	1,56-1,66	30	0,735-0,75	20
7	2,1-2,3	30	0,773-0,793	20
8	0,5-0,68	15	0,787-0,902	10
9	1,36-1,39	30	0,935-0,955	60
10	10,3-11,3	100	1,36-1,39	60
11	11,5-12,5	100	1,565-1,655	20
12	-	-	2,1-2,28	20
8A	-	-	0,855-0,875	20

Все снимки, сделанные с космических аппаратов, отправляются в крупный банк пространственных данных. Данные ДЗЗ, полученные со спутника Sentinel-2, управляются Европейским Космическим Агентством и распространяются по банкам данных, расположенным в его филиалах: Париж, Нордвейк, Дармштадт, Кёльн, Фраскати и т. д. Доступ к данным продукта Sentinel предусмотрен путём простой регистрации и обеспечен программой

Copernicus, поддерживающей свободную, полную и открытую политику данных.

Данные ДЗЗ, полученные с космического аппарата Landsat-8. Оперируются ведомством NASA, данные размещаются в их пространственных банках данных, к которым предоставляется свободный доступ, космические данные публикуются как общественное достояние.

1.5 Алгоритмы классификации

1.5.1 Классификация с использованием сигнатур

Классификация-процесс сортировки элементов изображения (пикселей) на конечное число сегментов (классов) на основе значений их атрибутов. Пиксель относят к определённому сегменту, если он удовлетворяет условию классификации.

Например, если взять космический снимок, сделанный в трёх коротковолновых диапазонах: синем (450-510 нм), зелёном (530-590 нм) и красном (640-670 нм), совместить их в одно растровое изображение и выбрать любую точку (пиксель) на изображении. Используя трёхмерную систему координат, можно найти область пикселей подобных заданному пикселю (рисунок 4).

Чтобы определить эту область, нужно найти математическое ожидание для данного пикселя по каждому из трёх каналов, а затем и среднеквадратичное отклонение.

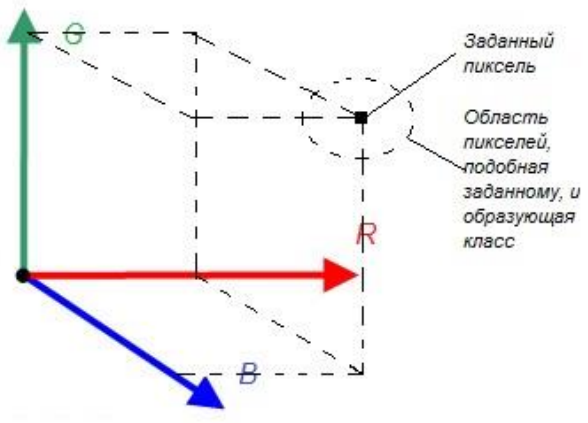


Рисунок 4 – Схема каналов RGB, образующих трёхмерную систему координат, демонстрирующую создание класса или сигнатуры

Таким образом, все пиксели, находящиеся на расстоянии меньшем, чем среднеквадратичные отклонения по каждому из каналов, будут вместе образовывать один класс. Таким методом будет образовано конечное количество классов. Классы также можно назвать сигнатурами, то есть набором атрибутов, обеспечивающих идентификацию того или иного участка на изображении.

1.5.2 Классификация методом k-ближайших соседей

Метод k-ближайших соседей – простейший метрический алгоритм, который основан на оценивании сходства объектов с экземплярами классов (рисунок 5).

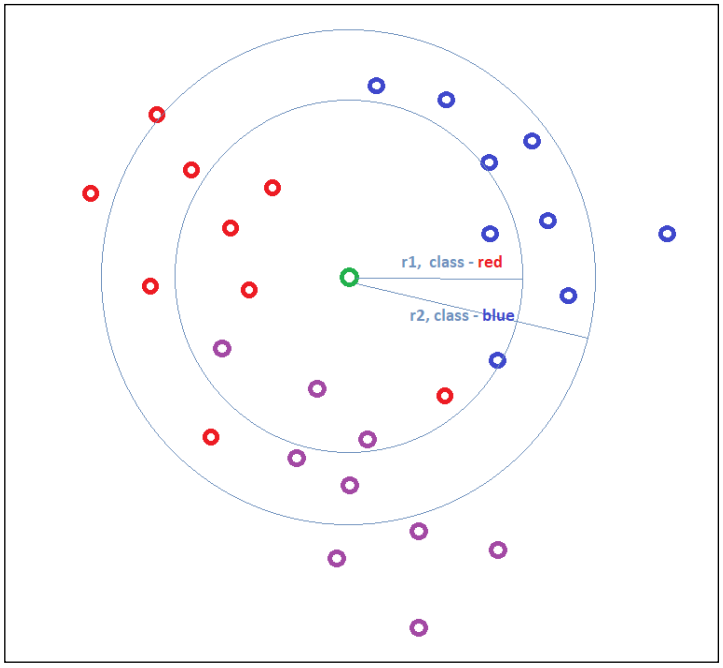


Рисунок 5 – Схема действия алгоритма классификации методом k-ближайших соседей

Данный алгоритм подразумевает собой отнесение объекта к тому классу, объектов которого находится в определённом радиусе наибольшее количество. Каждый класс объекта формируется на основе некоторого количества экземпляров или примеров, классификация объектов идёт путём сравнения его с экземплярами каждого класса. Чем больше количество экземпляров класса, тем точнее объект будет отнесён к определённому классу. Однако, для точности количество экзепляров каждого класса, должно приблизительно равняться друг другу. Для избежания неопределённости (при равном количестве соседей), можно присвоить каждому экземпляру определённый вес, который зависит от их зависимости.

Метод ближайшего соседа обладает простой реализацией, однако, при слишком большом числе соседей (>100), будет работать не очень быстро. Также, при большом наборе данных, присутствует сложность в подборе подходящих весов и выборке незначительных признаков. Подбор подходящего радиуса поиска соседей для оптимального решения задачи занимает также много времени.

1.5.3 Байесовский классификатор

Байесовский классификатор – простой вероятностный классификатор, который основан на теореме Байеса, формула которой показана на рисунке 6. Также называется наивным, так как наивно предполагается, что события наступают независимо друг от друга.

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)},$$

Рисунок 6 – Формула теоремы Байеса

Смысл формулы заключается в том, чтобы переставить местами причину и следствие. То есть, зная, с какой вероятностью причина ведёт к некоему следствию, можно рассчитать вероятность того, что именно эта причина привела к этому действию. Использование этой теоремы как классификатора, заменяет в её смысле причину на параметры, а следствие на класс. Таким образом, формула теоремы Байеса примет вид, показанный на рисунке 7, подходящий для классификации объектов (пикселей) на изображении.

$$P(Class\ A|Feature\ 1, Feature\ 2) = \frac{P(Feature\ 1|Class\ A) \cdot P(Feature\ 2|Class\ A) \cdot P(Class\ A)}{P(Feature\ 1) \cdot P(Feature\ 2)} \quad (1)$$

Рисунок 7 – Формула классификатора Байеса

Формула выше читается так: вероятность того, что пиксель принадлежит к классу А на основе параметров 1 и 2 равно отношению вероятности параметра 1, принадлежащего к классу А, умноженной на вероятность параметра 2, принадлежащего к классу А, умноженной на вероятность события А к произведению вероятности параметра 1 и вероятности параметра 2.

В качестве примера попробуем классифицировать качество производимых смартфонов, где параметрами являются страны производители. В качестве

образцов используются 1000 смартфонов 3 различных производителей, информация о которых записана в таблице 2

Таблица 2 – Частотная таблица производимых смартфонов

	Китай	Япония	Корея	Всего
Высокое качество	70	130	100	300
Среднее качество	100	160	140	400
Низкое качество	120	90	90	300
Всего	290	380	330	1000

Предположим, что неизвестный смартфон произведён страной-производителем Япония. Для определения вероятности класса качества при том, что страна-производитель является Японией (P), высчитывается вероятность, что страна-производитель не Китай (a), вероятность, что страна-производитель Япония (b), вероятность, что страна-производитель не Корея (c), вероятность этого класса качества (d) и подставляется в формулу (1):

P(высокое)=a1*b1*c1*d1=(230/300)*(130/300)*(100/300)*(300/1000)
=0.03322

P(среднее)=a2*b2*c2*d2=(300/400)*(160/400)*(260/400)*(400/1000)
=0.078

P(низкое)=a3*b3*c3*d3=(180/300)*(90/300)*(210/300)*(300/1000)
=0.0378

В итоге получается, что смартфон, произведённый страной-производителем Япония, является смартфоном среднего качества.

Использование Байесовского классификатора не требует построения каких-либо сложных алгоритмов и большого количества данных для обучения. Хоть он и достаточно простой, но классификатор работает хорошо в различных прикладных задачах. Если же у какого-либо класса присутствует значение

свойства, которое не встречается в наборе образцов для обучения, тогда оценка вероятности будет равна нулю и приведёт к получению объектов, не имеющих класса.

1.5.4 Классификация методом расстояния Махаланобиса

Расстояние Махаланобиса – это мера расстояния между двумя векторами величин, с помощью него можно определять сходство между известной выборкой и случайным вектором значений. Расстояние Махаланобиса отличается от расстояния Евклида, так как учитывает корреляцию между переменными и инвариантно масштабу (рисунок 8), то есть изменение расстояний никак не повлияет на уравнение, для этого в подсчёте расстояний используются ковариационные матрицы.

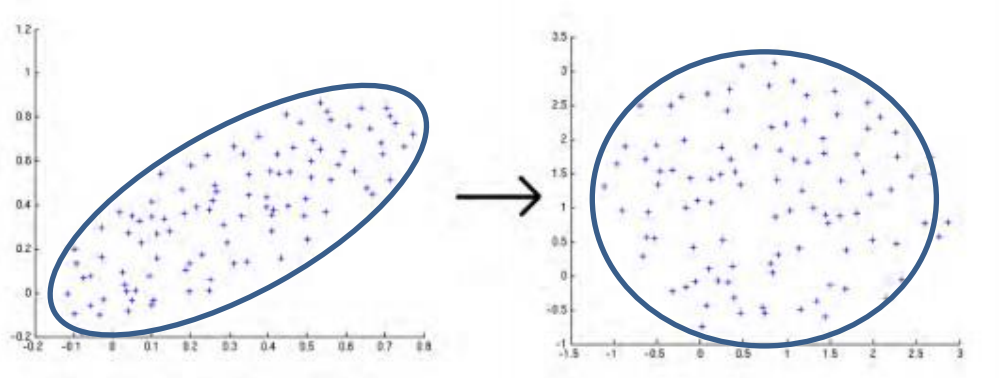


Рисунок 8 – Преобразование пространства при использовании метрики Махаланобиса

Ковариационная матрица представляет собой матрицу попарных ковариаций элементов одного или двух случайных векторов. Она является аналогом дисперсии случайной величины для случайных векторов. Ковариационная матрица вместе с математическим ожиданием полностью определяют её распределение.

Классификация методом расстояния Махаланобиса определяет различие между векторами и не зависит от масштаба, учитывает распределение значений

параметров некоторых пикселей в эталонных участках. Для подсчёта расстояния используется формула расстояния Махаланобиса, учитывающая ковариационную матрицу распределения, математическое ожидание и случайный многомерный вектор, изображённая на рисунке 9.

$$D_M(x) = \sqrt{(x-\mu)S^{-1}(x-\mu)^T}$$

Рисунок 9 – Формула расстояния Махаланобиса

Расстояние Махаланобиса в задаче определения принадлежности заданной точки одному из N классов предполагает для начала определение матриц ковариаций для каждого класса. Это делается на основе известных выборок каждого класса. Также это расстояние используется для нахождения выбросов или же для назначения объектов выборки весов значимости.

2 Реализация и тестирование алгоритмов классификации

2.1 Подготовка исходных данных

Для того чтобы приступить к работе, требуется найти подходящие космические данные и загрузить их. Загрузка данных выполняется посредством использования модуля Quantum GIS, Semi-automatic classification plugin (SACP). А в качестве области, по которой идёт поиск, используется территория Красноярской агломерации. Время съёмки должно быть актуальным на сегодняшний момент, то есть время съёмки не в зимний период 2017-2018 годов.

Для загрузки космического снимка, сделанного спутниками серии Landsat-8 или Sentinel-2A, выбирается соответствующая вкладка в панели инструментов SACP. Параметры, которые необходимы для поиска изображений (рисунок 10):

- координаты территории поиска: вводятся вручную или выделяется нужная область, используя кнопку, с изображённым на ней плюсом;
- дата совершения снимков территории: выбираются нужные границы даты;
- max cloud cover (%): вводится максимальная облачность в процентах, которая допускается на территории поиска;
- satelites: здесь выбирается версия спутника, сделавшего снимки (чем выше версия, тем больше каналов содержит изображение);
- результаты поиска: вводится максимальное количество отображаемых результатов поиска;
- фильтр: позволяет отфильтровать изображения по ImageID;
- bands (каналы): во вкладке Download options происходит выбор каналов в которых были произведены снимки.

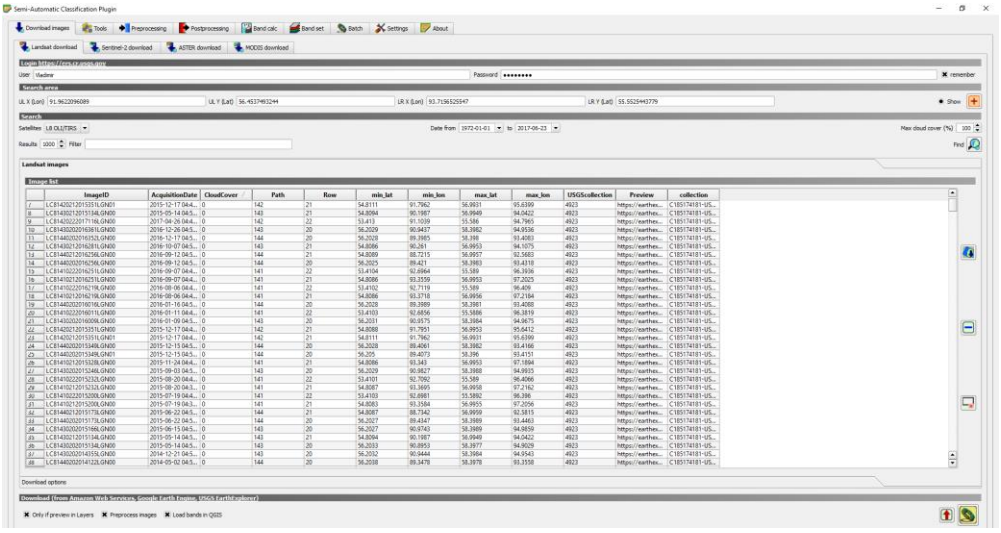


Рисунок 10 – Результаты поиска космических снимков

Из полученного списка выбирается территория, отражающая Красноярскую агломерацию с минимальным присутствием облаков и, далее, происходит скачивание всех, важных для классификации каналов данного космического снимка. Каналы со спутника Landsat-8: 2,3,4,5,6,7. Каналы со спутника Sentinel-2A: 2,3,4,5,6,7,8,8A,11,12.

После загрузки космических снимков, отображающих территорию города Красноярска и его окрестностей, не является лишним выполнить атмосферную коррекцию с помощью соответствующего инструмента SACP. Атмосферная коррекция проводится алгоритмом Dark Object Subtraction (DOS). Основное предположение состоит в том, что на изображении некоторые пиксели находятся в полной тени и их излучение, полученное со спутника, связано с атмосферным рассеиванием. Это предположение сочетается с тем фактом, что очень мало объектов на земной поверхности являются абсолютно чёрными. Поэтому допускается то, что однопроцентный минимальный коэффициент отражения лучше, чем нулевой процент.

По выполнению данного алгоритма космические данные были скорректированы таким образом, чтобы атмосфера оказывала минимальное влияние на отражённую от плоскости Земли энергию, которая важна при идентификации объектов на снимке.

После загрузки космических снимков Sentinel-2A, были получены снимки с охватом территории города Красноярска, состоящие из нескольких отдельных зон снимков, визуализирующих территорию Красноярска, как мозаичное изображение. Чтобы не заниматься каждой зоной по отдельности, нужно объединить их в единый космический снимок: объединить каждый канал зоны с теми же каналами зоны.

В итоге было получено 2 снимка со спутника Sentinel-2A и Landsat-8 города Красноярска. Однако их вид не совсем подходит для работы с ним, нужно произвести обрезку изображения до лучшей визуализации и работы с ним. Обрезку по всем каналам также можно выполнить с использованием инструмента SACP и, либо выделить область обрезки, либо выполнить обрезку по имеющемуся shape-файлу.

Таким образом, после предварительной обработки космических снимков, были подготовлены изображения города Красноярска и его окрестностей, на которых уже можно проводить идентификацию и классификацию объектов.

Алгоритмы классификации разрабатываются с использованием встроенных в Quantum GIS модуля Python и редактора для написания кода на этом языке программирования. Алгоритмы классификации должны найти объекты производства и потребления на территории города Красноярска и в его окрестностях. Алгоритмы, использующиеся для классификации предполагает обучение по каким-либо наборам данных. В качестве этих наборов данных были выбраны следующие классы объектов: лесная территория (tree), область с полевой территорией (field), водные ресурсы (water), строительные объекты (built), территории с отходами (dump).

Чтобы получить наборы данных для обучения нужно распознать соответствующие объекты на космическом снимке. Так как на чёрно-белых снимках различных каналов сложно распознать объекты классов. Перед этим нужно визуализировать космический снимок в естественных цветах. Создать такой снимок можно посредством использования инструмента SACP, Band Set. Band Set представляет собой создание виртуального слоя, используя

комбинации из 3 каких-либо исходных каналов, для возможности менять визуальное отображение слоя не прибегая к созданию нового. Чтобы снимок был визуализирован в естественных цветах, используется комбинация каналов 2-3-4, показанная на рисунке 11.

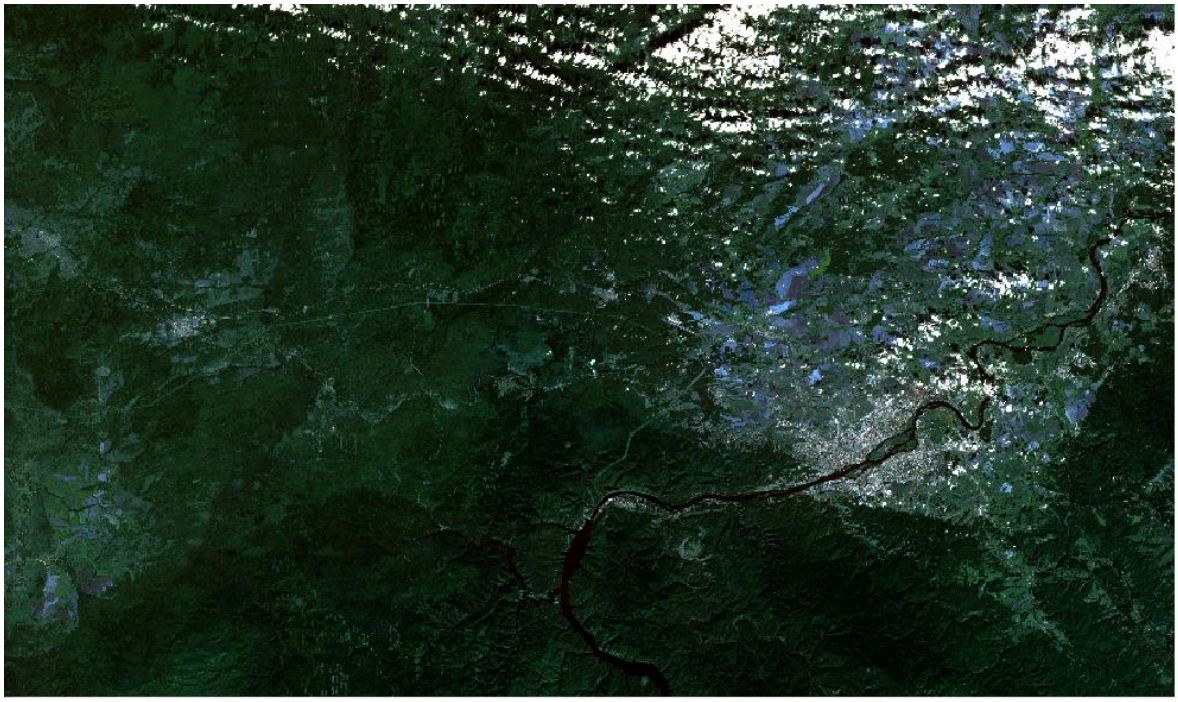


Рисунок 11 – Космический снимок Landsat-8, визуализированный в реальных цветах с использованием комбинации каналов 2-3-4

Разные комбинации каналов дают возможность визуально выделить для пользователей те или иные объекты на карте. Например, комбинация 6-5-4 даёт возможность анализировать сельскохозяйственные угодья и растительный покров. Комбинация 7-6-4 позволяет анализировать состояние атмосферы и дым. Комбинация 5-6-4 отображает растительность в различных тонах, даёт возможность изучения почв и растительного покрова. Комбинация 4-3-2 распределяет к анализу состояния водных объектов, оценивать глубины. Создавая определённую комбинацию каналов, можно найти наилучшее изображение объектов под конкретную прикладную проблему.

Найдя объекты, подходящие под описание классов, нужно как-либо выделить их на этом снимке. Для этого, с использованием инструментов

Quantum GIS, создаётся векторный слой или shape-файл, описывающий область, в которой находятся объекты определённого класса. Тот же инструмент для обрезки каналов позволит вырезать соответствующие зоны для каждого класса. Итогом таких манипуляций будет каталог образцов, нужный для работы алгоритмов классификации с обучением.

После выделения нужных областей на снимке как образцов, нужно загрузить информацию о них из каталогов прямо в программную среду, в которой ими и будут проходить манипуляции с помощью языка Python.

Для открытия изображений формата tif используется библиотека PIL, а для преобразования изображения в массив данных потребовалась библиотека NumPy. Отдаю дань уважения создателям данных библиотек. Используя несколько вложенных циклов, позволило перевести изображения в таблицу. При первом переводе в таблице были выявлены значения NaN, то есть значения являющиеся ничем, которые необходимо исключить посредством проверки условием сравнения с самим собой. Метод для перевода образцов в таблицу под названием input был написан.

При первых попытках работы с каталогом данных возникли проблемы, а конкретно, время, затрачиваемое на открытие файлов из каталогов и избыточность в их использовании: слишком большое количество пикселей для некоторых классов. Было решено выгрузить определённое количество значений пикселей для каждого класса в единую таблицу формата csv.

В итоге была создана csv-таблица: каждая строка представляет собой информацию об одном пикселе, где в первом столбце название класса, к которому он принадлежит, а в остальных информация о коэффициентах отражённой энергии из всех известных каналов. С такими данными легче работать, и нет трудностей в их быстрой выгрузке в программную среду, также размер файла отличается: таблица около 100 килобайт, а каталоги около 2 мегабайт.

Для начала классификации, нужно получить массив данных по классифицируемому изображению. Для первичных проверок работы

программы область классификации всего города Красноярска является слишком большой: слишком много пикселей (около 8 000 000), что потребует значительного количества времени. Для тестирования лучше взять небольшую часть снимка, где есть объекты всех классов: область северо-восточнее города Красноярска – Железногорск, территории которого можно увидеть на рисунке 12.

Рисунок 12 – Космический снимок Sentinel-2A, классифицируемой зоны города Железногорска в реальных цветах

Данный образец можно использовать для пробных классификаций. Напишем программный код по переводу данных из изображения в массив данных. Такое преобразование возможно посредством использования либо библиотеки NumPy, либо библиотеки GDAL. Однако во втором необходимо присутствие ещё одного файла формата xml с метаданными по снимку.

Использование такого метода на выход даст трёхмерный массив данных по классифицируемому изображению, где x и y это координаты пикселей, а z – номер канала изображения.

Csv-таблицу легко использовать, поэтому нужно написать метод выводящий его данные в два массива: первый массив представляет собой

список классов для каждого пикселя, а второй – список значений пикселей. Метод считывает таблицу построчно и отправляет в массив.

2.2 Реализация алгоритма классификации методом k-ближайших соседей

Пора приступить к написанию самих алгоритмов классификации. Первым алгоритмом для реализации будет алгоритм k-ближайших соседей. С помощью циклов придётся пройти по всем пикселям изображения, одновременно придётся сравнивать каждый пиксель со всеми значениями из массива образцов. Если какие-либо из образцов классов будут в пределах определённого Евклидова расстояния, то они будут являться соседями этого пикселя. В итоге будет получена матрица количества соседей для каждого класса. У какого класса будет наибольшее количество их, к тому классу и будет отнесён данный пиксель. В качестве параметра расстояния поиска соседей опробуется число tetta=0.16.

Метод классификации присваивает каждому пикселю новой матрицы определённое значение в соответствии с присвоенным ему классом. Однако, для вывода матрицы в виде изображения, потребуется такое создать. Для создания изображения формата tif используется библиотека GDAL. Она поможет перенести из классифицируемого изображения в создаваемый такие параметры, как: проекция, трансформация, размер, название.

Итак, для вывода первого классифицированного изображения под названием nearest.tif с использованием наборов данных в количестве по 200 пикселей для каждого класса и с параметром максимального расстояния поиска соседей, равным 0.16, что является безразмерной величиной. Классификация выполнена через некоторое количество времени, результат её показан на рисунке 13.

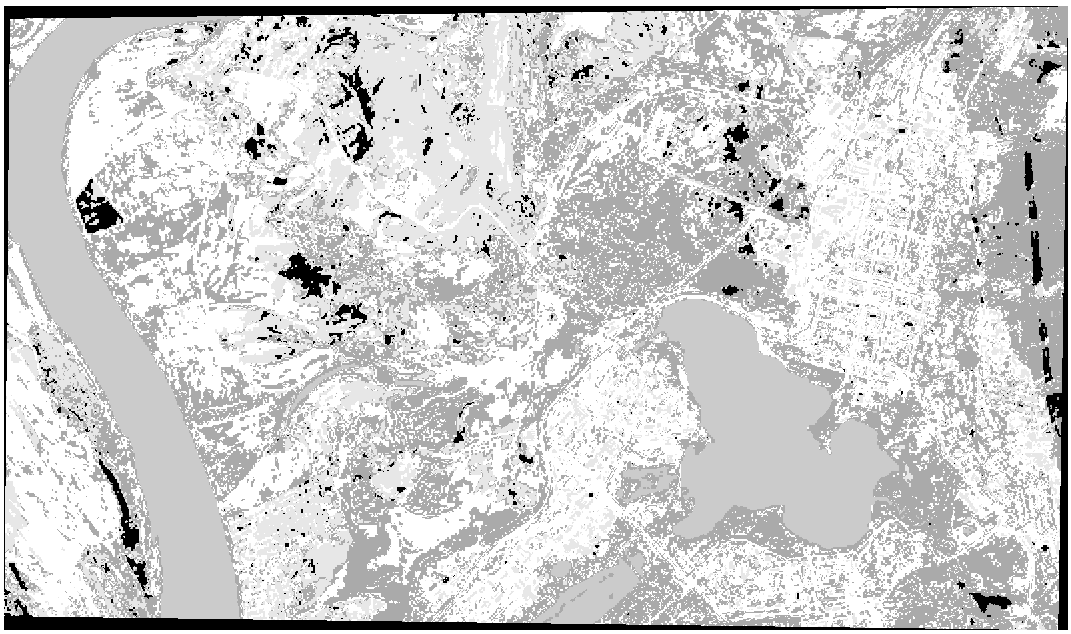


Рисунок 13 – Чёрно-белый классифицированный снимок Sentinel-2A города Железногорска

На выходе было получено чёрно-белое изображение классификации космического снимка, где полностью чёрные области и точки обозначают не классифицированные объекты. Но проанализировать эту классификацию не является возможным, для более простого восприятия того, где какие объекты были распознаны, нужно перейти к цветной их визуализации. Для сопоставления классификации с реальной картой местности, нужно подключить API карты-подложки OpenStreetMap, включающей карту улиц и настроить прозрачность (рисунок 14).

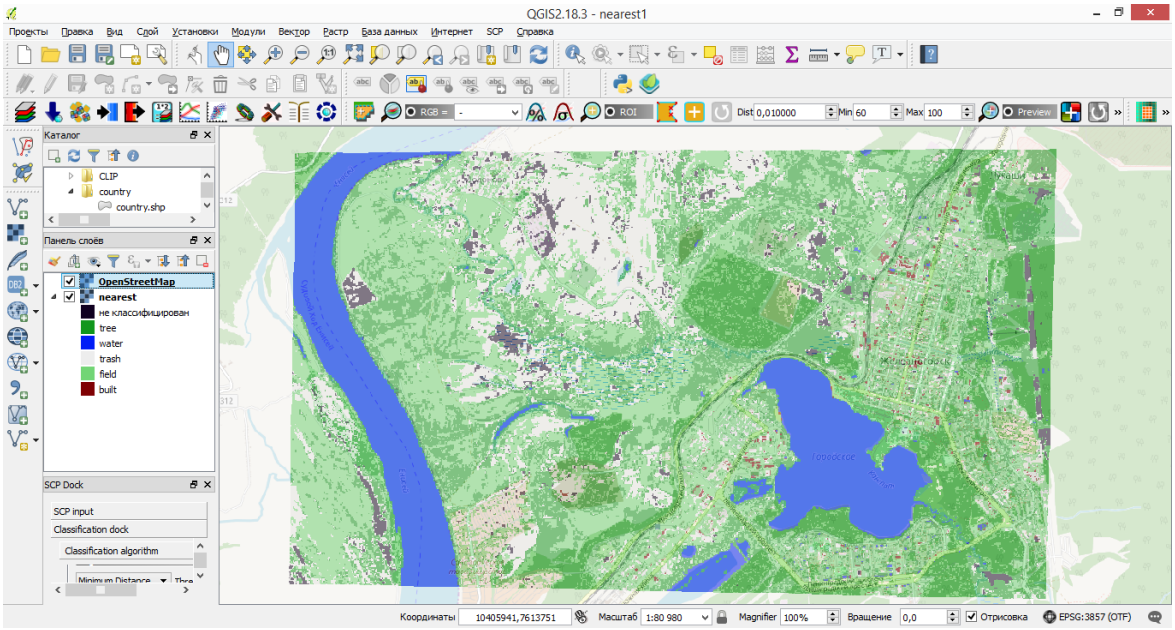


Рисунок 14 – Снимок Sentinel-2A с цветной классификацией объектов города Железногорска (200 образцов класса) методом k-ближайших соседей

Как видно из результатов классификации, были точно определены водные ресурсы на территории города Красноярска, так как коэффициенты отражённой энергии сильно отличаются от коэффициентов других объектов на снимке. Остальные объекты имеют некоторые не соответствия и пересечения с реальной ситуацией в городе. Слишком много объектов были классифицированы как отходы (в данном случае строительный мусор), большинство строительных объектов были классифицированы как поле, лесные ресурсы тоже классифицированы не идеально, однако по точности классификации являются вторыми после водных ресурсов. Алгоритм является рабочим, но не идеальным. Количество образцов каждого класса равнялось 200, далее, для увеличения качества классификации, нужно увеличить этот параметр до 300. Выполним классификацию и переведём результат в цветовую гамму (рисунок 15).

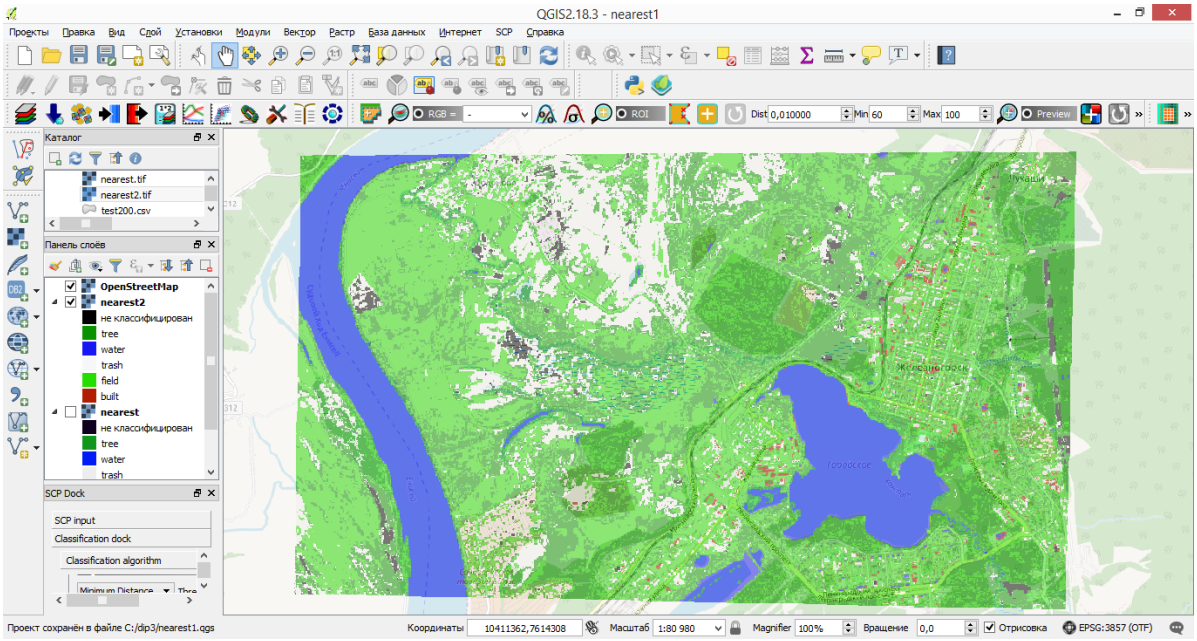


Рисунок 15 – Снимок Sentinel-2A с цветной классификацией объектов города Железногорска (300 образцов класса) методом k-ближайших соседей

В результате увеличения количества образцов каждого класса, качество классификации увеличилось, но в малой степени: появилось меньше не классифицированных объектов, больше было распознано строительных объектов. Можно и дальше улучшать его действие, посредством увеличения количества образцов, однако и время выполнения алгоритма будет увеличиваться: на 200 образцов было затрачено около 6 часов, а процент объектов не классифицированных равняется 2,3 %; на 300 образцов было затрачено уже 9 часов, а процент не классифицированных объектов составляет 1,2 %. Следовательно, если сделать это количество равным 400 на классификацию уйдёт около 12 часов. Классификация методом k-ближайших соседей с использованием Евклидовой метрики не является лучшим и от неё нужно отказаться сразу.

2.3 Реализация классификатора Байеса

Следующим алгоритмом для реализации является байесовский алгоритм классификации. Для этого используются значения пикселей. Перед началом

работы алгоритма, создаётся таблица значений пикселей, присваиваемых при попадании их в диапазоны коэффициентов отражённой энергии в каждом канале (таблица 3).

Таблица 3 – Таблицы, отображающие трёхмерную матрицу данных, важную при работе классификатора Байеса.

Пиксели, которые попали в диапазон коэффициентов отражения											
	B2	B3	B4	B5	B6	B7	B8	B8A	B11	B12	total
tree	298	295	296	271	287	285	296	286	273	280	300
water	296	298	297	292	287	270	293	280	290	294	300
dump	295	294	295	276	288	292	295	291	280	276	300
field	295	292	297	284	287	287	296	287	291	290	300
built	285	288	289	292	294	294	290	284	291	286	300
											1500
Пиксели, которые не попали в диапазон коэффициентов отражения											
	B2	B3	B4	B5	B6	B7	B8	B8A	B11	B12	total
tree	2	5	4	29	13	15	4	14	27	20	300
water	4	2	3	8	13	30	7	20	10	6	300
dump	5	6	5	24	12	8	5	9	20	24	300
field	5	8	3	16	13	13	4	13	9	10	300
built	15	12	11	8	6	6	10	16	9	14	300
Значения весов для каждого диапазона											
	B2	B3	B4	B5	B6	B7	B8	B8A	B11	B12	
tree	36,35	18,29	34,56	13,14	3,8	2,91	1,9	2,25	3,95	10,33	
water	55,98	71,77	75,65	93,3	59,55	49,98	73,65	51,83	87,47	96,52	
dump	2,81	2,93	2,11	3,88	2,69	2,17	1,8	1,88	2,82	3,3	
field	28,86	26,66	19,04	19,44	4,93	4,2	4,16	4,25	11,38	11,57	
built	1,3	1,2	1	1,31	1,26	1,2	1,09	1,25	1,57	1,47	
Значения пикселей, присваиваемые при ПОПАДАНИИ в определённый диапазон, в каждом канале											
	B2	B3	B4	B5	B6	B7	B8	B8A	B11	B12	
tree	36,11	17,99	34,1	11,87	3,64	2,76	1,87	2,15	3,59	9,64	
water	55,23	71,29	74,89	90,81	56,97	44,98	71,93	48,37	84,55	94,59	
dump	2,76	2,87	2,07	3,57	2,58	2,11	1,77	1,82	2,63	3,04	
field	28,38	25,95	18,85	18,4	4,72	4,02	4,1	4,07	11,04	11,18	
built	1,24	1,15	0,96	1,28	1,23	1,18	1,05	1,18	1,52	1,4	
Значения пикселей, присваиваемые при НЕПОПАДАНИИ в определённый диапазон, в каждом канале											
	B2	B3	B4	B5	B6	B7	B8	B8A	B11	B12	
tree	0,0067	0,0167	0,0133	0,0967	0,0433	0,05	0,0133	0,0467	0,09	0,0667	
water	0,0133	0,0067	0,01	0,0267	0,0433	0,1	0,0233	0,0667	0,0333	0,02	
dump	0,0167	0,02	0,0167	0,08	0,04	0,0267	0,0167	0,03	0,0667	0,08	
field	0,0167	0,0267	0,01	0,0533	0,0433	0,0433	0,0133	0,0433	0,03	0,0333	
built	0,05	0,04	0,0367	0,0267	0,02	0,02	0,0333	0,0533	0,03	0,0467	

В качестве пикселей, которые не попали в диапазоны, были выбраны те, которые находятся в области, находящейся у границ диапазонов коэффициентов и размером в пять процентов от реального её размера. Использование этих данных, как вероятности непопадания значений пикселей в диапазон исключает нулевые значения вероятностей. С использованием образцов классов строится трёхмерная матрица, которая является обучающими данными для классификатора. Так как диапазоны образцов данных слишком разнятся и одни даже входят в другие, это даёт широким диапазонам большое преимущество. Чтобы свести это преимущество к нулю, нужно создать таблицу весов, которые присваиваются при попадании в определённый диапазон. Веса высчитываются относительно самого максимального диапазона среди всех каналов каждого класса. То есть, чем меньше диапазон коэффициентов отражения, тем больше вес пикселя в этом диапазоне.

После формирования обучающих данных, можно приступать к классификации пикселей на космическом снимке. В данном случае эти данные значительно меньше по сравнению с данными в алгоритме k-ближних, поэтому время его работы значительно ниже. Однако, реализация работы алгоритма по созданию обучающих данных и обхода массива изображения занимает значительно больше времени из-за его более сложной структуры действия.

Классификатор Байеса выполнял свою работу не только с использованием обучающих данных, но и новых обучающих данных, которые дополняли основные при проходе массива изображения и пополняли таблицу значений, которые попадали или не попадали в диапазоны определённых каналов. Выполнение алгоритма было сравнительно недолгим и составляло всего лишь 3 минуты, несмотря на использование большого количества циклов.

Итогом классификации было растровое изображение, где каждому пикселю был задан определённый класс. Для вывода массива в это изображение используется тот же метод, написанный до этого для алгоритма k-ближних. Посредством обработки растра в Quantum GIS, на выходе получается его цветная визуализация, показанная на рисунке 16.

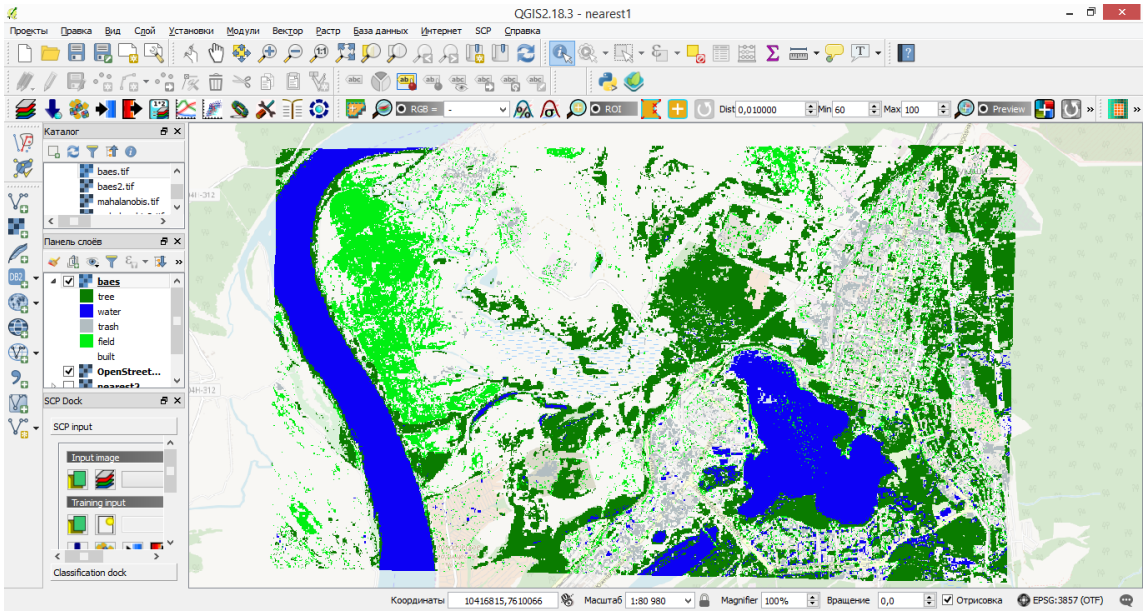


Рисунок 16 – Снимок Sentinel-2A с цветной классификацией объектов города Железногорска (200 образцов класса) классификатором Байеса

Тестирование алгоритма классификации методом Байеса привело к тому, что классам, имеющим большие диапазоны поиска, было присвоено значительное количество пикселей, неверно идентифицированных. В данном случае таким является класс объектов built. Чтобы избежать неверной идентификации объектов, следует добавить больше классов, а также уменьшить диапазон класса built, разбив его на подклассы.

Исключая объекты built, можно заметить, что остальные классы объектов в большей степени были идентифицированы, верно: водные ресурсы, поля и зелёные насаждения. Объекты строительного мусора были замечены на территории населённого пункта, однако являются лишь строительными объектами. Алгоритм работает хорошо на малых диапазонах на фоне больших, а диапазон коэффициентов отражённой энергии для класса built, в данном случае лучше считать не классифицированными объектами и скрыть их прозрачностью.

Увеличив выборку образцов класса с 200 до 300, эффективность классификации объектов была в некоторой степени ухудшена (рисунок 17).

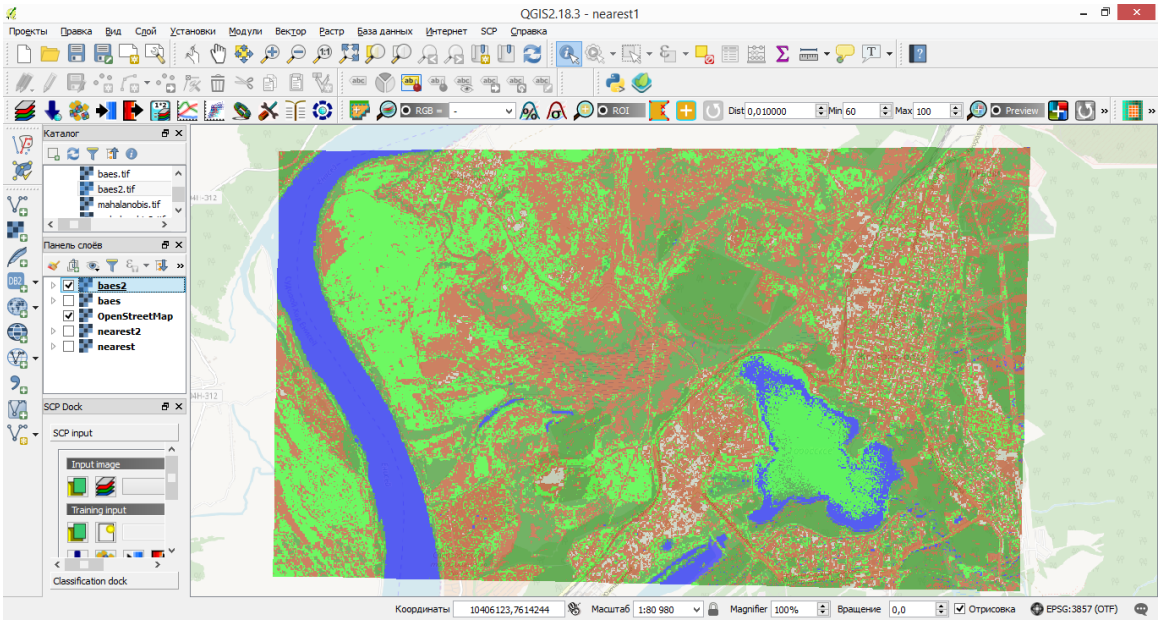


Рисунок 17 – Снимок Sentinel-2A с цветной классификацией объектов города Железногорска (300 образцов класса) классификатором Байеса

Результат, полученный после классификации, говорит о том, что теперь меньше объектов были классифицированы как строительные объекты (красный цвет) и больше как поля и лесные угодья, посредством увеличения экземпляров класса. Однако водохранилище города Железногорска было классифицировано как класс field. Причиной этому послужило то, что образцом являлась именно река, а, также то, что присутствует значительная разность в значениях коэффициентов отражения на реке и в водохранилище: взглянув на космический снимок можно заметить даже разницу в цвете. Объекты класса dump практически не изменились, так как диапазон значений этого класса не очень большой.

2.4 Классификация с использованием расстояния Махаланобиса

Классификация с использованием расстояния Махаланобиса предполагает использование ковариационных матриц, которые характеризуют расположение классов объектов. Каждый класс определяется разбросом его векторов значений, и описываются областями вида подобного эллипсоидам. Для

реализации работы алгоритма, нужно создать ковариационные матрицы для каждого класса образцов выборок. Их создание предполагает манипуляции с матричными данными: матричное умножение, транспонирование матрицы и вычисление обратной матрицы.

Используя формулу для нахождения расстояния Махаланобиса можно вычислить на основе ковариационных матриц классов, расстояние от центра масс класса до определённого пикселя. Однако если даже найти класс, от которого пиксель находится на наименьшем расстоянии, есть вероятность, что он не принадлежит к данному классу. Это возникает из-за разницы в размерах областей каждого класса. Таким образом, при написании алгоритмов классификации, нужно использовать расстояние Махаланобиса от определяемого пикселя и максимальное расстояние Махаланобиса класса.

Выполняется, реализованный на этой основе код и полученное изображение отображается в Quantum GIS, выделяя каждый класс объектов разным цветом.

В результате классификации по расстоянию Махаланобиса было создано растровое отображение всех объектов классов на территории города Железногорска. Некоторая часть объектов класса built была классифицирована как dump (серый цвет), а большая часть как field (светло-зелёный цвет). Причиной, по которой этому классу не было присвоено ни одного пикселя, является слишком большой диапазон значений его образцов данных. Также, территория, на которой находится водохранилище города Железногорска, была классифицировано как лесная территория. Возможная причина этого недостаточное количество данных о водных ресурсах. В качестве второй попытки классификации, как обычно используются вместо 200, 300 образцов каждого класса (рисунок 18).

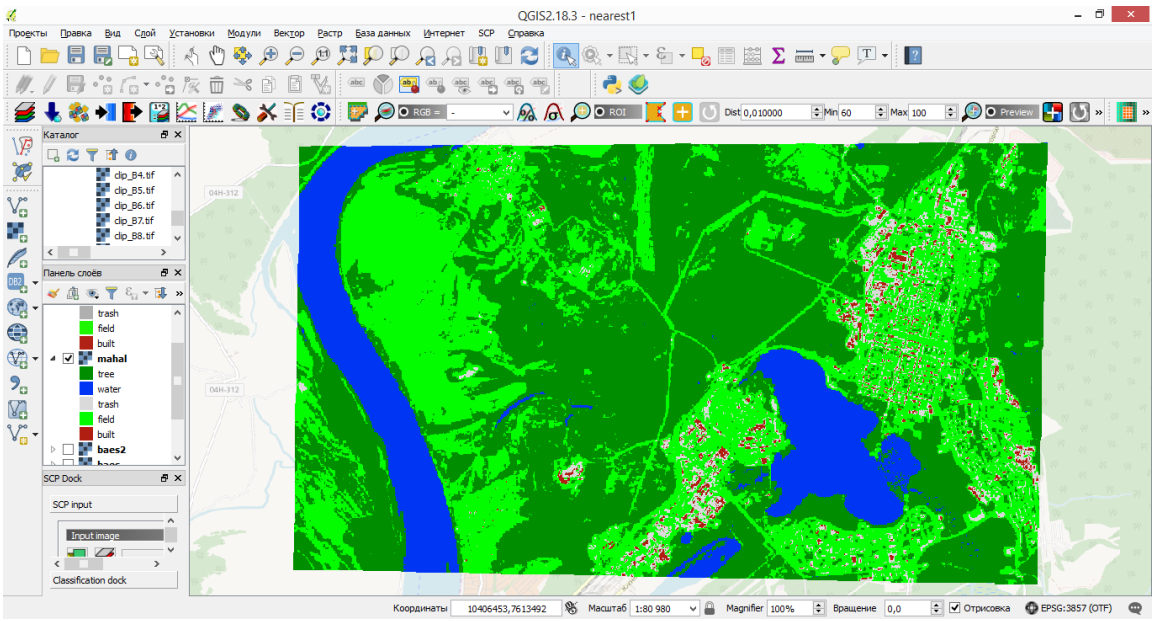


Рисунок 18 – Снимок Sentinel-2A с цветной классификацией объектов города Железногорска (300 образцов класса) с использованием расстояния Махаланобиса

После повторной попытке классификации территория водохранилища города Железногорска была чётко определена. Некоторые области лесных угодий теперь идентифицированы как обычные поля.

Чтобы на карте появилось больше мест класса built, нужно уменьшить диапазон значений класса built, а также подразделить его на классы, что повлечёт за собой более точную классификацию.

2.5 Классификация с использованием плагина SACP

Плагин SACP поддерживает классификацию с обучением, осуществляющую с помощью обучения по некоторым эталонам с созданием для каждого из них соответствующей сигнатуры, которые и используются в дальнейшем для определения центров классов.

Правила классификации, как и сами сигнатуры, могут быть параметрические и непараметрические.

Выделение сигнатур подразумевает под собой выделение определённых районов интереса, используя которые можно будет классифицировать изображение. Сигнатура характеризует собой набор атрибутов, обеспечивающих идентификацию того или иного участка на изображении. Это так называемый эталон или область с заданным классом, использующаяся для дальнейшей классификации с обучением по этим эталонам. Добавить эталоны можно несколькими способами:

— инструмент ROI-point, polygon;

— инструмент ROI-region grow;

— выбор из пространства признака (Feature space) связанного с классифицируемым изображением, пространство признака строится по двум осям (каналам изображения). По каждой оси откладывается значение от 0 до 255.

Итак, для того, чтобы начать работу по добавлению объектов на космическом снимке, нужно:

1 Выбрать файл, на котором и будут выделяться объекты (сигнатуры);

2 Создать файл training.scp, который и будет хранить всю информацию о созданных сигнатурах;

Для начала выделения объектов в панели инструментов нужно нажать Create ROI polygon (создать район интереса), выделяется полигоном часть реки Енисей. После, для выделенной области задаётся id и название и жмётся кнопка save temporary ROI in training input. После сохранения лист сигнатур изменится. Далее производится выделение жилой зоны города Красноярска. Для этого используется инструмент захвата пикселей Activate ROI pointer (активировать точечный поиск района интереса). Перед точечным поиском требуется ввести некоторые параметры: радиус захвата пикселей, минимальная область поиска пикселей и максимальная область поиска пикселей. Инструмент 13 осуществит поиск небольшого района инфраструктуры города. Также, для области задаётся id и название и происходит сохранение.

Чтобы произвести визуализацию классифицированного изображения, первым делом нужно выбрать алгоритм классификации Spectral angle mapping и ввести параметр, отвечающий за размер изображения “1000”, которое хотелось бы классифицировать. В панели инструментов жмём на иконку activate classification prewiev pointer (активировать предварительную точечную классификацию). Далее можно произвести классификацию участка города Красноярска, исходя из имеющегося листа сигнатур, сделав клик в его середину. Также, задав прозрачность, можно определить сходства между исходным изображением и результатом классификации (рисунок 19).

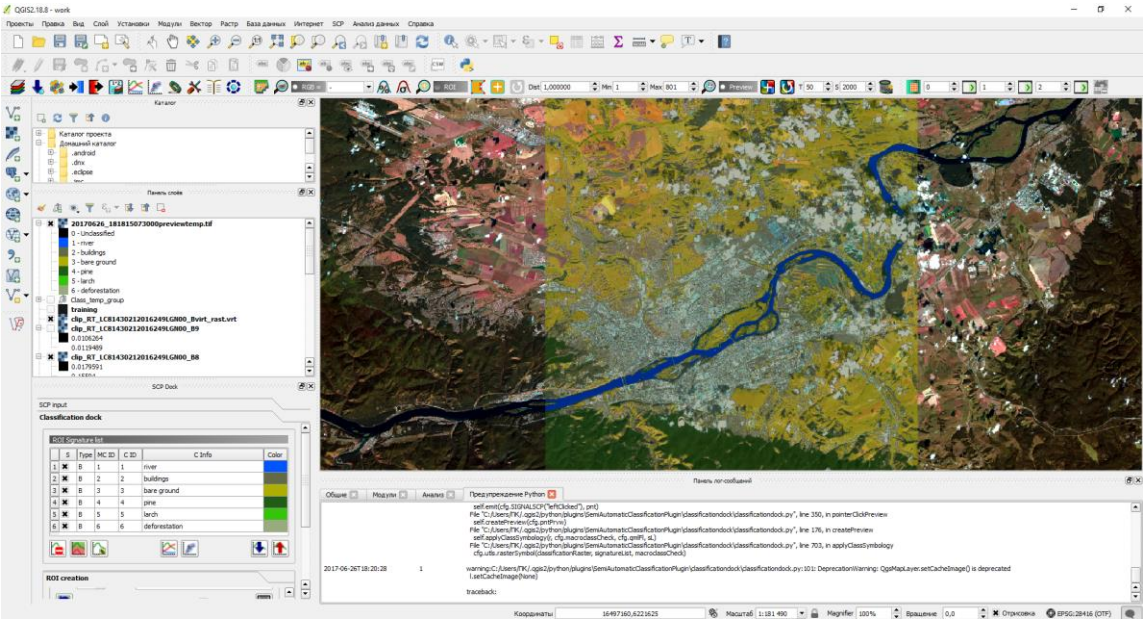


Рисунок 19 – Результат предварительной классификации участка города Красноярска на космическом снимке спутника Landsat-8 (сентябрь, 2016 год)

Сохранение полученной классификации происходит в виде файлов в формате tif. В данной работе было получено 8 файлов данного формата. Шесть из них представляют собой чёрно белые изображения, отражающие степень присутствия объекта класса на той или иной территории. Седьмой файл отражает совокупность всех шести растров. А последний файл отображает итоговую классификацию с цветной визуализацией изображения.

2.6 Общая схема действия реализованных алгоритмов

В результате написания кода алгоритмов классификации на Python, была создана совокупность методов, отвечающих за те или иные функции, схма которых изображена на рисунке 20.

Рисунок 20 – Общая схема действия реализованных на Python, алгоритмов

Схема чётко показывает то, каким образом данные переходят из одного метода в другой. На этапе классификации выбирается нужный метод и идёт идентификация объектов. Чтобы постоянно не менять название применяемого метода классификации можно использовать какой-либо параметр в виде числового или текстового значения в качестве входных данных, например, B – классификатор Байеса, M – классификация расстоянием Махаланобиса, K – метод k-ближайших соседей.

Таким образом, вся схема из отдельных методов может быть объединена в один единый метод, на вход которому идут следующие параметры: массив номеров каналов космического снимка, массив названий идентифицируемых

классов, название директории, в которой находится классифицируемый снимок и образцы изображений классов, название используемого классификатора, ограничение количества образцов для каждого класса и параметр минимального расстояния для алгоритма k-ближайших соседей, если используется именно он.

2.7 Оценка алгоритмов классификации

Оценка алгоритмов классификации предполагает анализ эффективности их действия. Оцениваться работа будет посредством применения разработанных алгоритмов, на уже известных образцах классов, из которых и были сформированы таблицы с образцами классов формата csv. Для реализации оценки пишется новый метод, применяющий различные классификации, а далее, подсчитывающий количество пикселей, верно и неверно классифицированных. Относительно общего их количества высчитывается процентная составляющая того, насколько точно была проведена классификация. Метод для реализации оценки также был написан на языке python.

Для оценки эффективности определённого алгоритма классификации, нужно ввести название соответствующего метода классификации. Чтобы исключить постоянное переименование методов, можно в качестве параметра в метод оценки добавлять название алгоритма, который нужно проверить.

После выполнения метода оценки выходными данными являются количество объектов классов, которые были идентифицированы на образцах классов, посредством которой и можно оценить процент верной идентификации. Итак, оценим эффективность трёх алгоритмов классификации: k-ближайших соседей, классификатор Байеса, расстоянием Махаланобиса с использованием обучающей выборки в 300 образцов на каждый класс, результаты оценки показаны на таблице 4.

Таблица 4 – Матрицы ошибок алгоритмов классификации

	Классификация методом k-ближайших					
	Количество идентифицированных пикселей, %					
class	tree	water	dump	field	built	Total, pixels
tree	67,38	0	0	32,62	0	13743
water	0	100	0	0	0	44199
dump	0	0	84,14	12,38	3,48	517
field	10,25	0	0,84	88,91	0	10897
built	0,16	0	32,99	50,62	16,24	3165
	Классфкация методом Байеса					
	Количество идентифицированных пикселей, %					
class	tree	water	dump	field	built	Total, pixels
tree	84,33%	0	0,01	0,01	15,64	13743
water	0	100	0	0	0	44199
dump	0	0	74,85	0	25,15	517
field	0,01	0	0,65	94,7	4,64	10897
built	0,06	0	47,8	0,44	51,69	3165
	Классификация методом расстояния Махаланобиса					
	Количество идентифицированных пикселей, %					
class	tree	water	dump	field	built	Total, pixels
tree	99,91	0,07	0	0,02	0	13743
water	1,01	98,99	0	0	0	44199
dump	0	0	56,87	7,54	35,59	517
field	0,11	0	0,96	98,93	0	10897
built	0	0	53,65	11,06	35,29	3165

После выполнения алгоритма были сформированы таблицы для каждой классификации и определена процентная составляющая верно и неверно идентифицированных пикселей. По данным таблицы, все алгоритмы хорошо идентифицируют водные ресурсы (water). У всех классификаций присутствуют ошибки в правильной идентификации объектов строительных сооружений (built) и свалок (dump). Так как объектами поиска являются именно территории с отходами, то наибольшей эффективностью по их поиску является метод k-ближайших соседей, а на втором месте классификатор Байеса. Чтобы использовать метод ближайшего соседа, следует уменьшить количество соседей, которое проверяется, до минимума.

3 Создание цифровой карты

Создание цифровой карты, объектов размещения отходов производства и потребления предполагает использование результатов работы одного из нескольких реализованных алгоритмов классификации. Тот алгоритм, который по оценкам являлся наиболее эффективным и должен использоваться для создания цифровой карты. Так как диапазоны отражённой энергии для различных видов отходов значительно разнится, лучше будет выделить для каждого небольшого диапазона отдельный класс. Конкретные классы должны быть определены в результате обработки территорий пользователем для выделения образцов отходов. Любые алгоритмы классификации могут не иметь возможности определить, что является отходами, а что просто обычный продукт человеческой деятельности: материал, из которого сделан продукт и материал как отход могут совершенно не отличаться значениями коэффициентов отражённой энергии. То есть даже если классификация показывает, что на территории населённого пункта есть какие-либо отходы, они могут не являться такими.

Для создания цифровой карты размещения различных типов отходов нужны результаты классификации идентифицируемой территории. В качестве такой территории была выбрана область города Красноярска, её окрестности, а также области близлежащих городов-спутников. В качестве алгоритма классификации выбран Байесовский классификатор как самый эффективный, который был применён для классификации территории города Красноярска (рисунок 21).

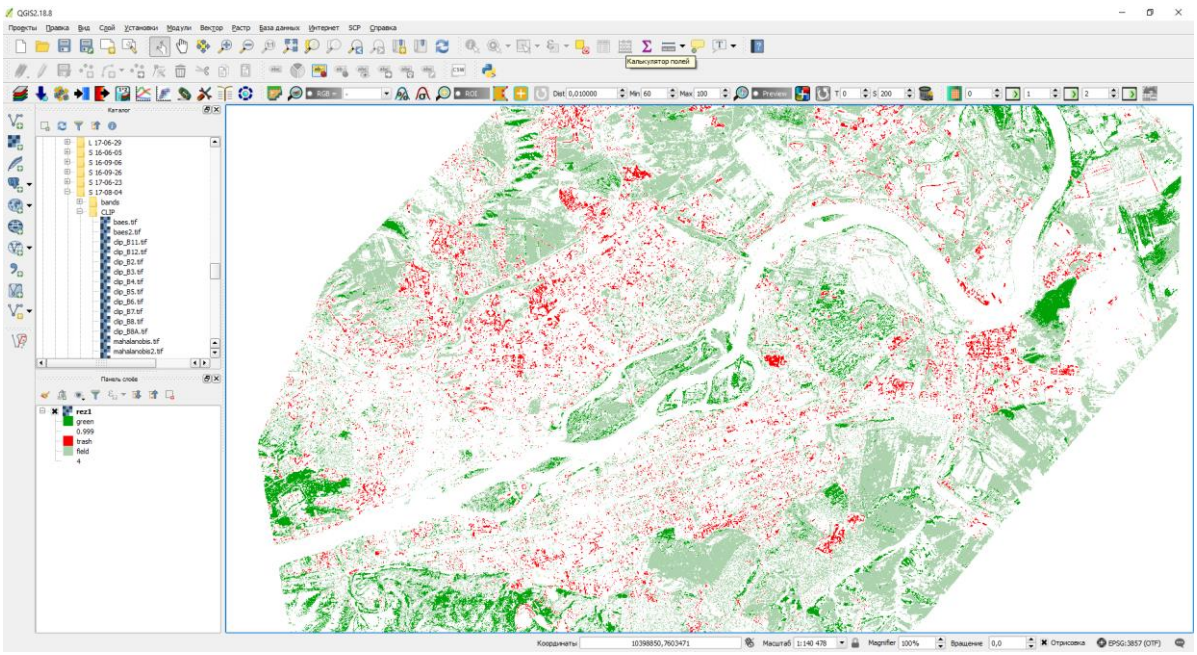


Рисунок 21 – Результат классификации Байесом, территории города Красноярска

Особое внимание лучше уделять территориям за пределами населённого пункта, то есть анализировать места, в пределах лесных массивов, полей, близ рек и некоторых дорог. Именно здесь важно найти места отходов, созданные человеком. Регистрация территорий размещения отходов позволит упростить работу для экологических служб и ведомств, действующих на территории города Красноярска и Красноярского края.

3.1 Создание тематической карты на основе кадастра отходов

Достижение цели, по созданию тематической карты размещения различных видов отходов предполагает использование как основы некой основы данных. В качестве этой основы выступает кадастр размещения отходов на территории города Красноярска и в его окрестностях. Он представляет собой векторный слой с точечным размещением тех или иных видов отходов на различных территориях, включающий таблицу атрибутов с информацией о каждом объекте. В данный слой включена информация об объектах

размещения отходов с различных промышленных предприятий: РУСАЛ, ТЭЦ, Сибтяжмаш, Красцветмет, ГХК, КраМЗ и так далее; это различные шламонакопители, шлакохранилища, золоотвалы, терриконы, полигоны бытовых и промышленных отходов, а также несанкционированные свалки. Некоторые, более различимые области с отходами выделяются в качестве образцов: бытовые отходы, промышленные отходы, несанкционированные свалки, шлаковые отходы, шламовые отходы, карбидный ил, золоотвалы, отходы графитного и древесного производства.

Области используются как образцы для обучения в работе алгоритма классификации, и насчитывают всего около 150 образцов. Используя их, проводится классификация методом ближайшего соседа. Результат классификации – чёрно-белый растр. Для лучшей визуализации, в его свойствах настраивается стиль отображения псевдоцветной на основе значений класса, присвоенного при классификации.

Для понимания того, какого местоположение территории свалок, относительно других объектов карты, нужна карта-подложка. Такой является карта OpenStreetMap. Она является абсолютно свободной и бесплатной. Сложив всё вместе, а также добавив саму кадастровую карту размещения отходов, была получена тематическая карта размещения отходов производства и потребления на территории города Красноярска, которая показана на рисунке 22.

Рисунок 22 – Карта размещения отходов производства и потребления на территории Ленинского района, города Красноярска

Полученная карта позволяет проанализировать территории, на которых были идентифицированы те, или иные виды отходов. Как было замечено: кадастровая карта размещения отходов не описывает точного размещения, а лишь показывает примерное местоположение территорий с отходами. Классификация позволила определить это распределение, а также найти дополнительно новые объекты на основе имеющихся образцов.

3.2 Анализ тематической карты

Идентифицированные объекты-пиксели в частности представляют собой любые антропогенные воздействия на почвы: полигоны под отходы, объекты рекультивации, земли, имеющие разные примеси вредных веществ.

Проанализировав территории города Красноярска, было выяснено, что такому влиянию больше всего подвержены промышленные районы, в то время

как в Центральном районе не было замечено значительных областей (рисунок 23).

Рисунок 23 – Классифицированный космический снимок Центрального района города Красноярска

На изображении преобладают лишь шламовые объекты в реке Енисей. В частности все пиксели, где отсутствует сильное течение, были идентифицированы как шлам.

Используя космические google-снимки с более высоким разрешением, можно проанализировать идентифицированные области. Например, золошлакоотвал с ТЭЦ-2 на кадастровой карте показывал лишь пару точек, а посредством классификации было определено их распределение, которое показывает, что все отходы помещены в водоотстойник (рисунок 24).

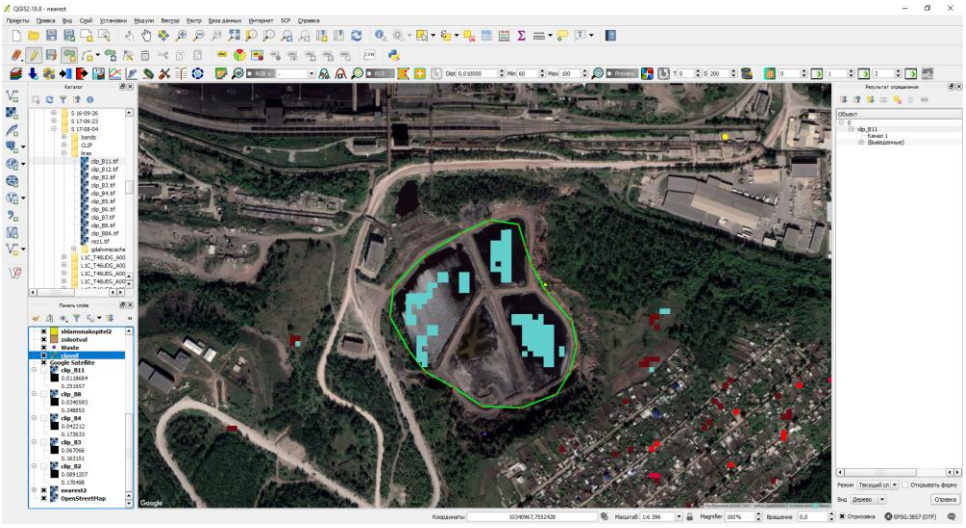


Рисунок 24 – Космический снимок территории шлакозолоотвала отходов с ТЭЦ-2

Множество почв было выделено как области с несанкционированными свалками. Эти места были в наибольшей степени подвержены антропогенному влиянию, то есть, количество примесей позволяет считать такие объекты свалками (рисунок 25, А, Б).

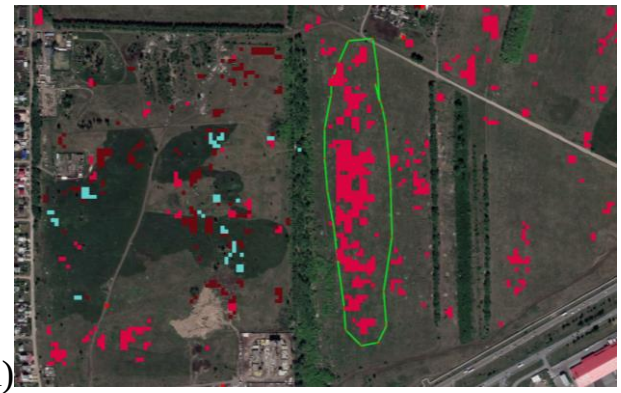
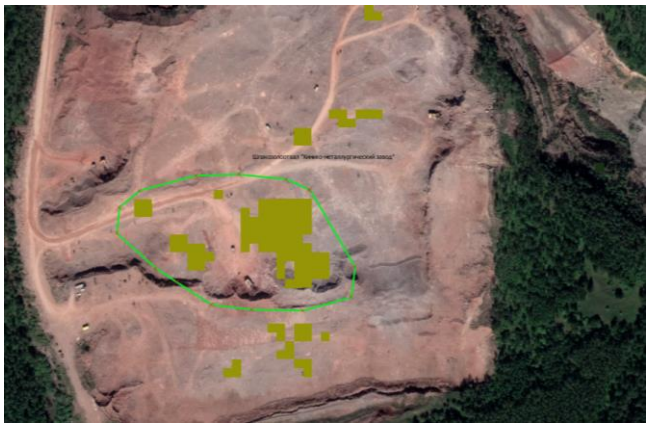
А)  Б) 

Рисунок 25 – Снимок территорий, подверженных антропогенному влиянию. А, Б – идентифицированные участки загрязнений

Также в процессе работы не были выделены области, полностью описывающие распределение некоторых видов отходов. Причин может быть несколько. Во-первых, коэффициенты отражённой энергии представляют собой значения, имеющие большую дисперсию, то есть небольшие отличия в

количестве примесей могут повлиять на классификацию. Во-вторых, количество образцов, на основе которых идёт идентификации не является достаточным, так как для каждого класса было отделено около 10 пикселей, а увеличение этого числа повлечёт за собой увеличение времени работы алгоритма. В-третьих, идентифицируемые пиксели представляют собой области 15 на 15 метров, и представляют усреднённые значения коэффициентов отражённой энергии. Примеры таких территорий показаны на рисунке 26, А, Б, В.

А) 

Б) 

В) 

Рисунок 26 – Космические снимки идентифицированных территорий отходов. А, Б, В – различные территории поиска свалок

Созданная тематическая карта визуально представляет области распределения различных видов отходов. Однако присутствуют не идентифицированные территории. Во избежание этого следует либо создать дополнительные классы объектов, либо добавить эти данные в качестве новых образцов в уже существующих классах. Как было сказано ранее, увеличение количества образцов или классов повлечёт за собой увеличение времени работы алгоритма классификации «k-ближайших соседей». Для более быстрой работы, нужно более мощное аппаратное обеспечение.

На данном этапе работы можно сделать вывод, что алгоритм позволяет наиболее оптимально определять объекты тех или иных классов. Если свести задачу к поиску лишь 2-3 класса объектов отходов, можно значительно уменьшить время работы алгоритма, и даже увеличить его эффективность. Использование космических данных с более высоким разрешением позволит также увеличить эффективность, но может увеличить и время работы.

Использование данных Landsat-8 и Sentinel-2A допустимо в задаче идентификации. Однако из-за их недостаточно высокого разрешения, идентифицируемые классы могут смешиваться, Также его не хватает чтобы отсеивать объекты по текстурным признакам.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной работы были реализованы три алгоритма классификации на языке python в Quantum GIS. Была произведена оценка эффективности действия алгоритмов и выбран наиболее оптимальный. На основе результатов классификации была создана цифровая карта объектов размещения отходов производства и потребления. В процессе выполнения работы была изучена предметная область, в пределах которой и создаются цифровые карты отходов.

По окончанию разработки была создана на основе кадастровой карты отходов тематическая карта размещения этих отходов, а также выполнен её анализ. Полученная карта представляет собой обобщённый вариант уже имеющегося кадастра.

Карты, полученные в результате такой работы, могут оказаться полезными для решения проблем несанкционированных свалок вблизи населённых пунктов. Посредством них можно найти места с несанкционированными свалками, не затрачивая при этом большого количества времени и денег. Таким образом, меры по устранению несанкционированных свалок будут предприниматься наиболее оперативно.

Задача по загрузке и предварительной обработке космических снимков к работе выполнена. Задача по разработке алгоритмов классификации выполнена, но возможно доработка для большей эффективности. Задача по реализации алгоритмов на языке Python и их тестирование выполнены, но возможно изменение кода для улучшения эффективности. Задача по оценке эффективности алгоритмов классификации выполнена. Задача по созданию цифровой карты объектов размещения отходов производства и потребления Красноярской агломерации выполнена частично, создана лишь цифровая карта объектов размещения отходов производства и потребления города Красноярска.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Цифровая карта. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: https:// ru.wikipedia.org/wiki/Цифровая_карта
2. Система координат. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: https:// ru.wikipedia.org/wiki/Система_координат
3. Python. Программирование для смартфонов [Электронный ресурс]. – Режим доступа: http://dimonvideo.ru/articles/1368/
4. Sentinel-2. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: http://ru.wikipedia.org/wiki/Sentinel-2
5. Мониторинг земель [Электронный ресурс]. – Режим доступа: http://biofile.ru/bio/35187.html
6. Дистанционное зондирование Земли. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: https:// ru.wikipedia.org/wiki/Дистанционное_зондирование_Земли
7. Система электронного обучения СФУ [Электронный ресурс]. – Режим доступа: https://e.sfu-kras.ru/
8. NumPy. Python 3 для начинающих [Электронный ресурс]. – Режим доступа: http://pythonworld.ru/numpy/1.html
9. Разработка простого решения для QGIS на Python. GIS-Lab [Электронный ресурс]. – Режим доступа: http://gis-lab.info/qa/qgis-dev-python.html
10. Разработка Python. Хабр [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/post/150302/
11. GDAL. GIS-Lab [Электронный ресурс]. – Режим доступа: http://gis-lab.info/
12. Космические снимки. GIS-Lab [Электронный ресурс]. – Режим доступа: http://gis-lab.info/qa/open-rs-control.html/
13. Свалка. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Свалка

14. OpenStreetMap. Русская энциклопедия Традиция [Электронный ресурс]. – Режим доступа: https://traditio.wiki/OpenStreetMap

15. Semi-Automatic Classification Plugin Documentation [Электронный ресурс]. – Режим доступа: http://semiautomaticclassificationmanual-v4.readthedocs.io/en/latest

16. Атмосферная коррекция по методу DOS. GIS-Lab [Электронный ресурс]. – Режим доступа: http://gis-lab.info/qa/atcorr-dos.html/

17. Методологический аппарат для классификации изображений. GIS-Lab [Электронный ресурс]. – Режим доступа: http://gis-lab.info/qa/genclass-erdas.html

18. Европейское космическое агентство. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/w/index.php?title=Европейское_космическое_агентство&oldid=91923924

19. НАСА. Сетевая энциклопедия Wikipedia [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/w/index.php?title=НАСА&oldid=90767156

20. Чабан, Л. Н. Теория и алгоритмы распознавания образов. М.,МИИГАИК, 2004 г. – 128 с.

21. Chavez, P. S. Image-based atmospheric correction—revisited and improved. Photogrammetric Engineering and Remote Sensing, 1996 г. - С. 1025 – 1036.

22. Жиленев, М. Ю. Обзор применения мультиспектральных данных ДЗЗ и их комбинаций при цифровой обработке, ГЕОМАТИКА №3, 2009 г. – 163 с.

23. Форсайт, Д. А. Компьютерное зрение. Современный подход.: Пер. с англ. / Д. А. Форсайт, Ж. Понс. – М. : Издательский дом Вильямс, 2004 г. – 928 с.

24. Дворкин, Б. А. Новейшие и перспективные спутники дистанционного зондирования земли/ Б. А. Дворкин, С. А. Дудкин// Геоматика. -2013. -№2. – С. 16-36.

25. Копанева, И. М. Обработка многозональных снимков с использованием геоинформационных технологий/ И. М. Копанева, Е. А. Рублева// Теория и практика гармонизации взаимодействия природных, социальных и производственных систем региона. -2017. – С. 343-347.

26. Лаврова, О. Ю. Средства и методы работы с данными спутникового дистанционного зондирования/ О.Ю. Лаврова, М.И. Митягина, А.Г. Костяной// Спутниковые методы выявления и мониторинга зон экологического риска морских акваторий. – Москва, 2016. – С. 50-100.

27. Адамович, Т. А. Новая группировка спутников дистанционного зондирования земли Sentinel/ Т.А. Адамович, В.П. Савиных// Биодиагностика состояния природных и природно-техногенных систем. – 2017. – С. 42-45.

28. Адамович, Т. А. Использование различных комбинаций спектральных каналов космических снимков спутника landsat 8 для оценки природных сред и объектов/ Т.А. Адамович, Т.Я. Ашихмина, Г.Я. Кантор //Теоретическая и прикладная экология. -2017. -№2. – С. 9-18.

29. Богданов, А. П. Разработка методики мониторинга состояния лесов на основе использования данных мультиспектральной космосъемки/ А.П. Богданов, Р.А. Алешко// Проблемы экологического мониторинга и моделирования экосистем. -2017. -№1. – С. 98-110.

30. Деркачёва, А. А. Эффективность атмосферных коррекций гиперспектральных снимков HYPERION в регионах с развитым растительным покровом/ А. А. Деркачёва, О. В. Тутубалина// Современные проблемы дистанционного зондирования земли из космоса. -2014. -№4. – С. 360-368.

31. Донцов, А. А. Разработка технологии организации каталогов спутниковых данных/ А. А. Донцов, Н. В. Волков, А. А. Лагутин// Известия алтайского государственного университета. -2014. -№1-2. – С. 172-175.

ПРИЛОЖЕНИЕ А

Код алгоритмов классификации объектов

```
import time
import os
from math import *
import numpy as n
from PIL import Image
from osgeo import gdal
from osgeo import osr
import sys
bn=['2','3','4','5','6','7','8','8A','11','12'] #channels, that have  for identification and classification
ob=['tree','water','trash','field','built'] #classes, that identificated on the space image
csv='test200.csv' #name of csv table
loc='F:\CLIP'
loc2='E:\dip\S 17-08-04\CLIP'
loc3='F:\\'
#ar=[]
#csvAr=[]
#csvCl=[]
tetta=0.16
n1=200

def input(csv, loc, bn, ob, n1):      #input information about classes in csv-table,
    os.chdir(loc)
    outcsv=open(csv, 'w')
    for o in ob:
        os.chdir(loc+o)
```

```
        im=Image.open(o+'_clip_b2.tif')
        imar=n.array(im)
        i=0
        t=str(o)+';'
        n2=0
        for row in imar:
            j=0
            for el in row:
                t1=t
                for b in bn:
                    os.chdir(loc+o)
                    im=Image.open(o+'_clip_b'+b+'.tif')
                    imar1=n.array(im)
                    if imar1[i,j]==imar1[i,j]:
                        t1=t1+str(imar1[i,j])+';'
                j+=1
                if t1!=t:
                    t1=t1+'\n'
                    outcsv.write(t1)
                    n2+=1
                    if n2==n1:break
            i+=1
            if n2==n1:break
    outcsv.close()


def imageInArray(loc2, bn): # image go in massiv
    ar=[]
    os.chdir(loc2)
    for b in bn:
```

```
        raster=gdal.Open('clip_B'+b+'.tif')
        band=raster.GetRasterBand(1)
        band=band.ReadAsArray()
        ar.append(band)
    return ar

def imageInArray2(loc2, bn, o): # image go in massiv
    ar=[]
    os.chdir(loc2)
    for b in bn:
        im=Image.open(o+'_clip_b'+b+'.tif')
        imar=n.array(im)
        ar.append(imar)
    return ar

def csvInArray(csv, loc): #csv table in massiv
    os.chdir(loc)
    csv=open(csv, 'r')
    Ar=[]
    Cl=[]
    while True:
        n=csv.readline().rstrip()
        if not n:break
        D=n.split(';')[1:len(bn)+1]
        D=[float(el) for el in D]
        Ar.append(D)
        N=n.split(';')[0]
        Cl.append(N)
    return Ar, Cl
```

```
def classification(ar, bn, tetta, csvAr, csvCl): # classification method n-nearest
    arOut=ar[0]
    N=0
    kol=0
    for i in range(len(ar[0])):
        for j in range(len(ar[0][0,])):
            obraz=[0,0,0,0,0]
            D=[]
            if ar[0][i,j]==ar[0][i,j]:
                for k in range(len(bn)):
                    D.append(ar[k][i,j])
                for n in range(len(csvAr)):
                    s=0
                    for k in range(len(bn)):
                        s+=(D[k]-csvAr[n][k])**2
                    s=sqrt(s)
                    if s<=tetta:
                        obraz[klass(csvCl[n], ob)]+=1
                arOut[i,j]=max(obraz)
                if arOut[i,j]==0:
                    N+=1
                kol+=1
        print i
    print N, kol #amount of pixels in image(kol) and pixels no classificated(N)
    return arOut


def assesment(ob, loc, bn, csvAr, csvCl, tetta): #assesment of work classificcation
    value=[0, 1, 2, 3, 4]
    o1=0
```

```
    for o in ob:
        obraz=[0,0,0,0,0]
        os.chdir(loc+o)
        arOs=imageInArray2(loc+o, bn, o)
#       arOut2=classification(arOs, bn, tetta, csvAr, csvCl)
#       arOut2=classificationBaes(arOs, bn, ob, csvAr, csvCl)
        arOut2=classificationMahalanobis(arOs, bn, ob, csvAr, csvCl)
        n1=0
        for i in range(len(arOut2)):
            for j in range(len(arOut2[0,])):
                if arOut2[i,j]==arOut2[i,j]:
                    for v in range(len(value)):
                        if round(arOut2[i,j], 1)==value[v]:
                            obraz[v]+=1
                            break
                    n1+=1
        print 'kol-vo', n1
        print o, obraz[o1]
        s=0
        for v in range(len(obraz)):
            if v!=o1:
                s+= obraz[v]
        print 'another', s
        print obraz
        o1+=1
    print 'end'


def outputClassification(arOut, loc2):
    os.chdir(loc2)
```

```
        raster=gdal.Open('clip_B2.tif')
        outPr=raster.GetProjectionRef()
        outTr= list(raster.GetGeoTransform())
        outRS = (ar[0].shape[1], ar[0].shape[0])
        driver = gdal.GetDriverByName('GTiff')
        dst_ds = driver.Create( 'mahal2.tif', outRS[0], outRS[1], 1,
gdal.GDT_Float32 )
        dst_ds.SetGeoTransform(outTr)
        dst_ds.SetProjection(outPr)
        outB= dst_ds.GetRasterBand(1)
        outB.SetNoDataValue(-99)
        outB.WriteArray(arOut)
        outB.FlushCache()
        print 'end'

    def klass(n, ob): #define class
        r=0
        for i in range(len(ob)):
            if ob[i]==n:
                r=i
        return r

    def max(obraz): # define maximal
        t=1.0/len(obraz)
        max=0
        r=0
        for i in range(len(obraz)):
            if obraz[i]>max:
                max=obraz[i]
                r=i+1
```

```
    if r==0:
        r-=1
    return (r+1)*t


def classificationBaes(ar, bn, ob, csvAr, csvCl): # classification method Baes
    arOut = ar[0]
    arTable=n.zeros((5, len(ob), len(bn)))
    #table[0]-pixels in diapazon
    #table[1]-maximal znachenie of pixel in diapazon
    #table[2]-minimal znachenie of pixel in diapazon
    #table[3]-pixels not in diapazon
    #table[4]-znachenie of ves pixel in this diapazon
    for i in range(len(csvAr)):
        if arTable[0, klass(csvCl[i], ob), 0]==0:
            arTable[0, klass(csvCl[i], ob),]+=1
            arTable[1, klass(csvCl[i], ob),]=csvAr[i]
            arTable[2, klass(csvCl[i], ob),]=csvAr[i]
        else:
            arTable[0, klass(csvCl[i], ob),]+=1
            for b in range(len(bn)):
                if arTable[1, klass(csvCl[i], ob), b]<csvAr[i][b]:
                    arTable[1, klass(csvCl[i], ob), b]=csvAr[i][b]
                if arTable[2, klass(csvCl[i], ob), b]>csvAr[i][b]:
                    arTable[2, klass(csvCl[i], ob), b]=csvAr[i][b]
    obFull=arTable[0,:,0]
    max=0
    for o in range(len(arTable[0])):
        for b in range(len(arTable[0,0])):
            if arTable[1, o, b]-arTable[2, o, b]>max:
                max=arTable[1, o, b]-arTable[2, o, b]
```

```
        for o in range(len(arTable[0])):
            for b in range(len(arTable[0,0])):
                arTable[4, o, b]=max/(arTable[1, o, b]-arTable[2, o, b])
        for o in range(len(ob)):
            for b in range(len(bn)):
                diff=arTable[1, o, b]-arTable[2, o, b]
                arTable[1, o, b]-=(diff*0.05)
                arTable[2, o, b]+=(diff*0.05)
        for i in range(len(csvAr)):
            for b in range(len(bn)):
                if arTable[1, klass(csvCl[i], ob), b]<csvAr[i][b] or arTable[2, klass(csvCl[i], ob), b]>csvAr[i][b]:
                    arTable[0, klass(csvCl[i], ob), b]-=1
                    arTable[3, klass(csvCl[i], ob), b]+=1
        full=len(csvCl)
        for i in range(len(ar[0])):
            for j in range(len(ar[0][0,])):
                if ar[0][i,j]==ar[0][i,j]:
                    o1=[0,0]
                    for o in range(len(arTable[0])):
                        p=1
                        for b in range(len(arTable[0,0])):
                            if ar[b][i,j]<=arTable[1, o, b] and ar[b][i,j]>=arTable[2, o, b]:
                                p*=arTable[0, o, b]/obFull[o]*arTable[4, o, b]
                            else:
                                p*=arTable[3,o,b]/obFull[o]
                        p*=obFull[o]/full
                        if o1[0]<p:
                            o1[0]=p
                            o1[1]=o
```

```
                full+=1
                obFull[o1[1]]+=1
                for b in range(len(arTable[0,0])):
                    if ar[b][i,j]<=arTable[1, o1[1], b] and ar[b][i,j]>=arTable[2, o1[1], b]:
                        arTable[0, o1[1], b]+=1
                    else:
                        arTable[3, o1[1], b]+=1
                arOut[i,j]=o1[1]
        return arOut

    def classificationMahalanobis(ar, bn, ob, csvAr, csvCl):
        arOut=ar[0]
        arTable=n.zeros((2, len(ob), len(bn)))
        csvAr1=n.array(csvAr)
        for o in ob:
            for i in range(len(csvCl)):
                if csvCl[i]==o:
                    arTable[0, klass(csvCl[i], ob),]+=1
                    for b in range(len(bn)):
                        arTable[1, klass(csvCl[i], ob), b]+=csvAr[i][b]
        arTable[1]=arTable[1]/arTable[0]
        for i in range(len(csvCl)):
            for b in range(len(bn)):
                csvAr[i][b]=csvAr[i][b]-arTable[1, klass(csvCl[i], ob), b]
        arr=[]
        for o in ob:
            arr1=[]
            for i in range(len(csvCl)):
                if csvCl[i]==o:
```

```
                arr1.append(csvAr[i])
            arr1=n.array(arr1) #list in matrix
            arr1=n.dot(arr1.T, arr1) #multiplication matrix and matrix(T)
            arr1=arr1/(arTable[0, klass(o, ob), 0] -1) #kovariacionnaya matrix for
every class
            arr.append(arr1)
        arTable[0, :, 1]=0
        for i in range(len(csvCl)):
            D=n.array(csvAr1[i])
            t=n.linalg.inv(arr[klass(csvCl[i], ob)])
            RM=n.dot((D-arTable[1, klass(csvCl[i], ob), ]).T, t)
            RM=n.dot(RM, (D-arTable[1, klass(csvCl[i], ob), ]))
            RM=sqrt(RM)
            print RM
            if RM>arTable[0, klass(csvCl[i], ob), 1]:
                arTable[0, klass(csvCl[i], ob), 1]=RM
        for i in range(len(ar[0])):
            for j in range(len(ar[0][0,])):
                if ar[0][i,j]==ar[0][i,j]:
                    D=[]
                    b=[0, 99999]
                    for k in range(len(bn)):
                        D.append(ar[k][i,j])
                    for o in range(len(ob)):
                        t=n.linalg.inv(arr[klass(o, ob)])
                        RM=n.dot((D-arTable[1, o, ]).T, t)
                        RM=n.dot(RM, (D-arTable[1, o, ]))
                        RM=sqrt(RM)/arTable[0, o, 1]
                        if b[1]>RM:
                            b[1]=RM
```

```
                    b[0]=o
                arOut[i,j]=b[0]
    return arOut

st=time.time()
#input(csv, loc, bn, ob, n1)
#print time.time()-st

ar = imageInArray(loc, bn)
print time.time()-st

csvAr,csvCl=csvInArray(csv, loc3)
print time.time()-st

arOut1=classification(ar, bn, tetta, csvAr, csvCl)
print time.time()-st

arOut1=classificationBaes(ar, bn, ob, csvAr, csvCl)
print time.time()-st

arOut1=classificationMahalanobis(ar, bn, ob, csvAr, csvCl)
print time.time()-st

outputClassification(arOut1, loc)
print time.time()-st

assesment(ob, loc3, bn, csvAr, csvCl, tetta)
print time.time()-st
```