НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:

Завідувач кафедри

_____ Олександр КОВАЛЬ

«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційні технології моніторингу довкілля»

спеціальності 122 «Комп'ютерні науки та інформаційні технології»
на тему: «Система аналізу часових змін лісових насаджень методом ДЗЗ»

Виконав (-ла):
студентка IV курсу, групи ТМ-62
Богач Анжеліка Геннадіївна _____

Керівник:
Старший викладач
Бандурка Олена Іванівна _____

Консультант:

_____

Рецензент:
Доцент кафедри АЕС і ІТФ, к.т.н.
Баранюк Олександр Володимирович

_____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 122 Комп’ютерні науки та інформаційні технології

Спеціалізація Інформаційні технології моніторингу довкілля

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр КОВАЛЬ
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту
Богач Анжеліці Геннадіївні
(прізвище, ім’я, по батькові)

1. Тема роботи «Система аналізу часових змін лісових насаджень методом ДЗЗ» керівник роботи Бандурка Олена Іванівна, старший викладач
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**

2. Строк подання студентом роботи _____
3. Вихідні дані до роботи мова програмування Python, середовище Spyder

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) провести аналіз предметної області, обрати технології для розробки, спроектувати моделі даних, що будуть використовуватися в системі, запропонувати архітектурне рішення системи, спроектувати графічний інтерфейс користувача_____
_____
_____
_____
_____

5. Перелік ілюстративного матеріалу: зображення видів класифікації космічних

знімків, діаграма прецедентів, графічне представлення інтерфейсу розробленої системи, приклади роботи програмного модулю

_____
_____
_____
____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ”___”_____201___ р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітки
1.	1.1.1.1.1 Затвердження теми роботи	25.10.19	
2.	Вивчення та аналіз задачі	03.02-11.02.20	
3.	Розробка архітектури та загальної структури системи	12.02-20.02.20	
4.	Розробка структур окремих підсистем	21.02-28.02.20	
5.	Програмна реалізація системи	02.03-11.03.20	
6.	Оформлення пояснювальної записки	12.03-29.05.20	
7.	1.1.1.1.2 Захист програмного продукту	10.06.20	
8.	1.1.1.1.3 Передзахист	10.06.20	
9.	Захист		

Студент _____ Анжеліка БОГАЧ
(підпис) (прізвище та ініціали,)

Керівник роботи _____ Олена БАНДУРКА
(підпис) (прізвище та ініціали,)

АНОТАЦІЯ

В даній роботі приведено програмне рішення системи аналізу часових змін лісових насаджень. Описано задачу та можливий спосіб її вирішення у вигляді спроектованої архітектури та виконаного програмного рішення.

Записка містить 87 сторінок, 11 рисунків, 5 додатків і 38 бібліографічних найменувань за «Переліком посилань».

Ключові слова: ДЗЗ, моніторинг довкілля, космічні знімки, Python, MySQL, Spyder.

ABSTRACT

This paper presents the software solution of the system of analysis of temporal changes of forest plantations. The problem and possible way of its solution in the form of the designed architecture and the executed software solution are described.

The note contains 87 pages, 11 figures, 5 attachments and 38 links.

Keywords: remote sensing, environmental monitoring, space imagery, Python, MySQL, Spyder.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ДЗЗ – дистанційне зондування Землі.

Python – мова програмування.

MySQL – база даних, яка задовільняє потреби програмного продукту та легка у використані разом з мовою програмування Python.

ШСЗ – штутні супутники Землі.

ГІС (GIS) – геоінформаційна система, яка виконує функції збору та аналізу даних.

ArcGIS – це програмне забезпечення для використання ГІС будь-якого рівня складності.

ВСТУП

Розвиток ІТ-технологій дозволяє нам розвивати наші можливості для покращення екологічної ситуації не тільки в межах нашої країни, але у світі в цілому. Одним з таких інструментів є дистанційне зондування Землі (ДЗЗ).

Здоров’я лісів значно погіршилось за останнє десятиліття через негативні наслідки глобальних змін клімату: підвищення температури та зменшення кількості опадів призвели до збільшення площі сушильних насаджень, а також до масового ураження шкідниками та хворобами. Наразі відслідковується подальше висихання насаджень більшості лісоутворюючих видів (ялина, ясен, дуб, граб, береза). Експерти найбільше стурбовані станом соснових насаджень, де триває катастрофічне пошкодження ослаблених лісів стебловими шкідниками.

Охорона та контроль лісових насаджень є обов'язковими, тому існує український закон про ліси, який регулює ведення лісового господарства, якщо закон України про охорону навколишнього середовища.

Метою Лісового кодексу України є регулювання правових заходів щодо підвищення продуктивності, охорони та відтворення лісів, посилення їх використання та задоволення потреб громади у лісових господарствах на основі знань та відповідності.

Державний лісовий облік та державний лісовий кадастровий фонд для можливої ефективної організації охорони та збереження лісів, раціональних можливих лісових ресурсів, рятування лісів, систематичного контролю за якісними та кількісними змінами лісового фонду та іншої інформації, зацікавленої у здійсненні лісових прав.

Саме тому терміновий облік лісів необхідний, адже за останній час ми втратили велику кількість лісової площі через стихійне лихо, а саме: пожежу. Оцінити конкретні масштаби трагедії дуже важко, тому що потрібно провести колосальну

роботу по обліку тих лісових насаджень, які вціліли на території України.

Моніторинг навколишнього середовища [2] є основним методом контролю за екологічним станом країни. Саме цей фактор є основним серед тих, яке потребує людство для продовження комфортного життя та розвитку наших досягнень.

На жаль, ми вже зараз помічаємо як руйнуємо комфортне життя тим, що використовуємо природні ресурси нашої планети. Тому ми повинні постійно аналізувати наш вплив на природу та намагатися нейтралізувати негативні наслідки нашої діяльності.

1 ЗАДАЧА РОЗРОБКИ ІНФОРМАЦІЙОЇ СИСТЕМИ АНАЛІЗУ ЧАСОВИХ ЗМІН ЛІСОВИХ НАСАДЖЕНЬ МЕТОДОМ ДЗЗ

Поставлена задача призначена для створення програмного продукту, який аналізує зміни лісів в часовому просторі. В зв’язку з відсутністю довгострокової інформації, труднощами доступу до архівів, вирішено застосувати класифікацію космічних знімків.

На основі цього функціоналом програмного продукту повинен бути:

- зберігання результатів класифікації у базі даних;
- аналіз отриманих даних;
- порівняння даних по рокам;
- легкість да доступність у користуванні програмним продуктом;
- перевірка отриманих результатів програмного продукту.

Розвиток супутникових систем протягом останніх років сприяє збільшенню даних в геоінформаційних системах. Постійно з’являються нові методи класифікації космічних знімків та аналізу даних. Зважаючи на те, що технології постійно змінюються, на даний момент немає змоги створити продукт, який міг би відповідати всім потребам користувача.

Програмний продукт такого виду повинен адаптуватися до специфічних вимог кожного з користувачів. Було принято рішення реалізувати базовий функціонал, який буде використовуватися кожним, але може бути розширений у майбутньому.

Одним із найскладніших завдань було поєднати дані класифікації використовуючи кросплатформлені можливості. Класифікація зображень – це дуже трудоємкий процес, який займає дуже багато часу через те, що відбувається порівняння кожного пікселя в зображені. Для коректної класифікації потрібно виділити полігони кожної породи аби програмний продукт зміг точно виділити потрібний піксель та віднести його до класу, який відповідає його стандартам.

Вхідні дані, які будуть зберігатися у базі даних, повинні відповідати певному формату, а саме:

— номер класу;

— назва класу;

— кількість дерев у обраній місцевості.

Тобто, занесена інформація повинна правильно трактуватися після класифікації космічного знімку. Структура таблиці дозволяє чітко вносити тільки потрібну інформацію, яка необхідна для аналізу екологічного стану. Тому такі дані відкидуються, щоб аналітичні потужності можна було зосередити на головному.

Після створення програмного продукту потрібно оцінити ефективність роботи. Потрібно порівняти результати аналізу системи з даними, які є в офіційних документах та джерелах.

Програмний продукт повинен бути розрахований на роботу з великим обсягом даних, адже саме швидкодія системи є необхідним чинником для роботи у цій галузі. Архітектура системи повинна бути зрозумілою для всіх, хто буде займатися розвитком продукту та окремих модулів у подальшому.

Після отримання завдання та уточнення всього необхідного функціоналу була розпочата робота над створенням швидкої, легкодоступної та простої у використанні системи аналізу часових змін лісових насаджень методом ДЗЗ.

1.1 Висновки до розділу

Було сформульовану задачу для створення системи аналізу часових змін лісових насаджень методом ДЗЗ, досліджено акутальність проблеми та нинішній стан проблематики у країні.

2 Дані дистанційного зондування Землі (ДЗЗ – Remote Sensing)

Для якісного виконання завдання потрібно ретельно розібратися з теорією та основними засобами обробки даних ДЗЗ.

2.1 Основні характеристики космічних знімків

Космічне сканування [3, 4] – один з провідних методів дистанційного зондування. Він виконується з використанням:

- штучних супутників Землі (ШЗС);
- міжпланетних автоматичних станцій;
- довготривалих орбітальних станцій;
- пілотованих космічних кораблів.

Космічну зйомку можна використовувати різними способами.

Залежно від характеру поверхні, ви можете бачити наступні картини:

- одиночне фотографування;
- маршрутну;
- прицільну;
- глобальну зйомку.

Незалежну (вибіркову) фотографію виконують космонавти ручними камерами. Зображення, як правило, багатообіцяючі з яскраво вираженими кутами.

Лінійна перевірка земної поверхні проводиться по трасі польоту супутника. Дальність стрільби залежить від висоти польоту та кута огляду системи зйомки.

Фокусне (селективне) сканування призначене для отримання зображень спеціально визначених частин земної поверхні поза маршрутом.

Глобальна візуалізація виконується з геостаціонарних та полярних супутників на орбіті. Чотири або п’ять геостаціонарних супутників на екваторіальній орбіті

забезпечують майже безперервні розвідувальні зображення по всій Землі (космічний патруль) за винятком полярних шапок.

Масштаб просторових зображень різний: від 1:1000 до 100 000 000, тобто він може змінюватися в сто разів. Найпоширеніші масштаби космічних зображень: від 1:200 000 до 1:10 000 000.

Масштаби космічних знімків залежать від:

- висоти фотографування;
- фокусної відстані апарату;
- коефіцієнта збільшення;
- кутів нахилу;
- кривизни земної поверхні.

Просторова роздільна здатність (або дозвіл на місцевості) визначається розміром найменшого (Δ), відтвореного на знімку, і визначається за формулою:

$$\Delta = m / 2N, \text{ де}$$

m – масштаб знімка;

N – роздільна здатність знімка, тобто число окремо фотографічно відтворюваних чорно-білих штрихів на відрізку довжиною 1 мм [5].

Віддалене збирання даних також використовується для створення баз даних для різних систем ГІС, навіть як «базовий» субстрат.

Обробка дистанційного зондування поділяється на підготовчу та тематичну і виконується в спеціалізованому програмному забезпеченні.

Попередня обробка даних для дистанційного зондування включає геометричні, радіометричні виправлення атмосферного зображення, географічну довідку зображення.

Після віддаленої обробки їх можна використовувати для створення високоякісних картографічних продуктів, таких як оновлення топографічної бази та розшифрування об’єктів інфраструктури.

Тематична обробка даних дистанційного зондування (дані дистанційного зондування) – це метод покращення зображення, що включає регулювання контрасту, зменшення шуму, вибір краю.

Тематична обробка даних дистанційного зондування виконується з метою вирішення конкретної проблеми, наприклад для моніторингу змін на території через місцезнаходження та ін. У процесі тематичної обробки даних відбувається класифікація об’єктів за космічними зображеннями за їх характеристиками.

Дистанційне зондування забезпечує більш детальну інформацію про місцевість, оскільки не всі дрібні об’єкти можуть відображатися на спеціалізованих картах завдяки узагальненню даних (невеликі озера, галявини, зміни порід дерев тощо).

Після обробки даних дистанційного зондування називають «дистанційними датчиками».

Якість та асортимент матеріалів дистанційного зондування залежить від наступних характеристик:

- просторового дозволу;
- спектрального дозволу;
- радіометричного дозволу;
- тимчасового дозволу [6].

2.2 Методи обробки даних ДЗЗ

Обробка даних ДЗЗ (image processing) – процес виконання операцій над аерокосмічними знімками, що включає їх корекцію, перетворення і поліпшення, дешифрування, візуалізацію [7].

Зростаючі обсяги доповнюють архів даних ДЗЗ, значно підвищуючи вимоги до швидкості та якості обробки даних. Розробляються методи обробки зображень ДЗЗ, які можна розділити на два типи: попередню обробку та спеціальну. За допомогою методів попередньої обробки вирішуються такі методи: посилення контрасту, зміна розміру, ортокорекція, радіометрична корекція, морфологічна обробка, видалення фону, шуму або інших небажаних об’єктів на зображенні, поліпшення зображення, використання фільтрів, сегментація тощо. Такі методи, як правило, є першим етапом аналізу зображення, за яким слід спеціальна обробка. Спеціальна обробка включає

методи, призначені для вирішення конкретних завдань, пов’язаних з кінцевими елементами, такі як пошук і розпізнавання об’єктів, класифікація об’єктів у зображенні. В обох групах особливий інтерес викликає використання методів, заснованих на використанні моделей нейронної мережі та високоефективних обчислювальних технологій через високу ефективність, продемонстровану на практиці [8].

Тому складно зробити основну причину геометричного дизайну [9]. Комбнація – це ртуть у маркуванні, що відрізняється для різних типів марок.

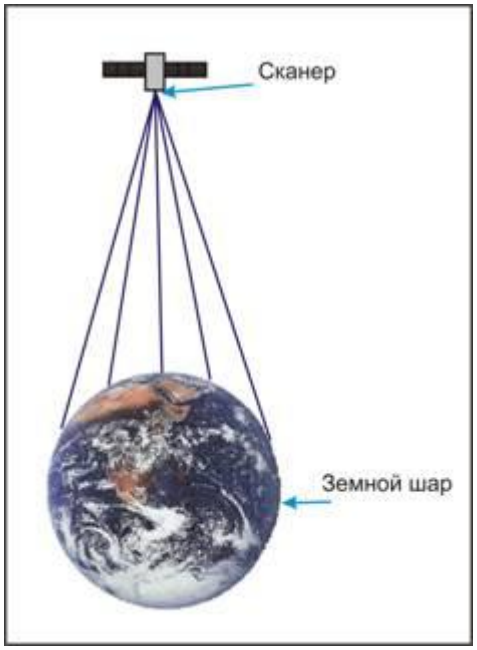


Рисунок 2.1 – Кривизна поверхні Землі

Геометричне спотворення зображень, спричинене викривленням земної поверхні, є результатом того, що точки відсканованої області не лежать в одній площині, а спостереження знаходиться не в самій глибокій площині, а під кутом до земної поверхні, як видно на рисунку 2.1. Тому, коли ви віддаляєтесь від осі сканування (там, де проводиться опитування внизу), спотворення форми та розмірів об’єктів збільшуються.

Спотворення форми предметів [10]. Прямою лінією поля буде крива на зображенні, квадрат, прямокутник тощо. Цей тип спотворень можна ігнорувати, якщо кут огляду сканера невеликий (MSS - Landsat, кут огляду приблизно 5,8°).

Викривлення масштабу. У випадку з зображеннями, зробленими за допомогою

оптико-механічного сканера (MODIS, AVHRR, ETM та MSS - Landsat, Aster (TIR)) - масштаб зменшиться після видалення зображення з центральної лінії. Це означає, що якщо ви візьмете два пікселі зображення: один із центральної області зображення, а другий збоку, піксель із бічної області міститиме більшу частину Землі, навіть якщо вони однакового розміру.

У випадку із зображеннями ПЗЗ (SPOT, IRS, Ikonos, супутникові датчики Астера (VNIR, SVIR)) масштаб не змінюється при русі від осі зображення.

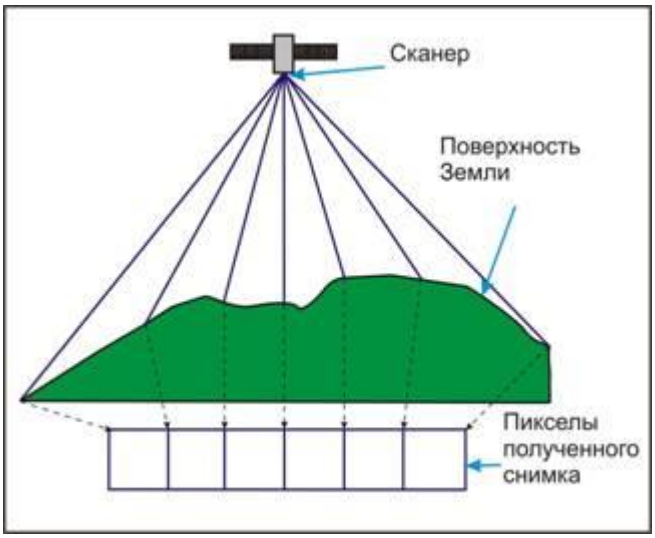


Рисунок 2.2 – Нерівність рельєфу Землі

На рисунку 2.2 ми бачимо, що нерівності рельєфу викликають ті ж спотворення, що й кривизна поверхні Землі, але завдання усунення їх складніше, через те, що форми рельєфу складніші, ніж форма Землі, яка близька до сфери.

Оскільки космічні знімки роблять з великої висоти, то вплив форм рельєфу незначний, тому даний тип спотворень враховують лише для гірських областей.

«Відсутні пікселі» можуть виникати під час збору або передачі даних, значення яскравості всього рядка також можуть бути замінені значеннями сусідньої лінії (рис. 2.3). Такі явища можуть стати перешкодою для тематичної обробки зображень. Відсутні пікселі можуть бути відновлені інтерполяцією з певною помилкою.

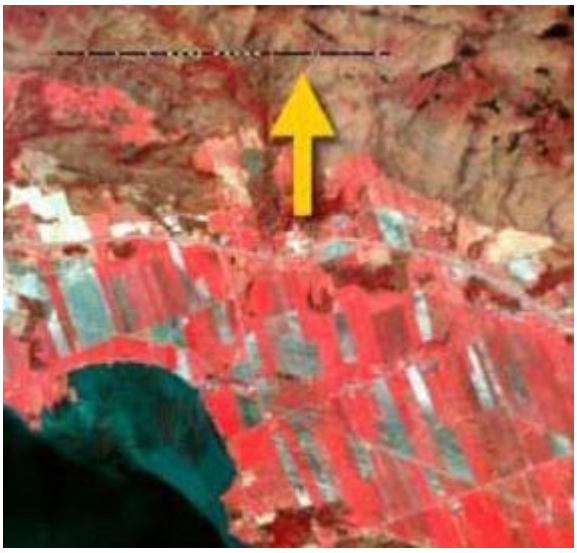


Рисунок 2.3 – Відсутні пікселі у зображеннях

Фільтрування - це перетворення, яке дозволяє поліпшити відтворення певних об'єктів, усуваючи додатковий випадковий шум.

Один з найпростіших способів фільтрації – перетворення в ковзному вікні.

При такому перетворенні перераховуються значення яскравості всіх пікселів зображення. Перерахунок відбувається для кожного пікселя таким чином: коли даний піксель є центральним у вікні, яке "рухається" по знімку, йому дається нове значення, яке є функцією від значень оточуючих його у вікні пікселів.

Розмір вікна може бути, наприклад 3x3 або 5x5 пікселів. Щоразу вікно зміщується на 1 піксель і рухається до тих пір, поки не пройде весь знімок.

Для всіх пікселів вікна дослідник встановлює вагові коефіцієнти виходячи з цілей дешифрування (рисунок 2.4).

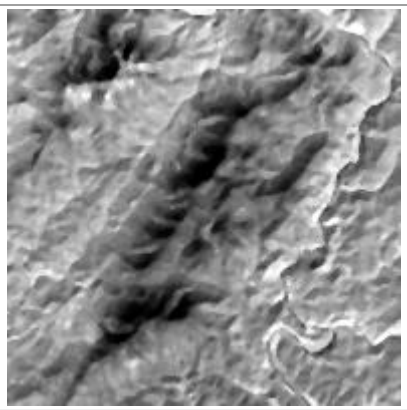
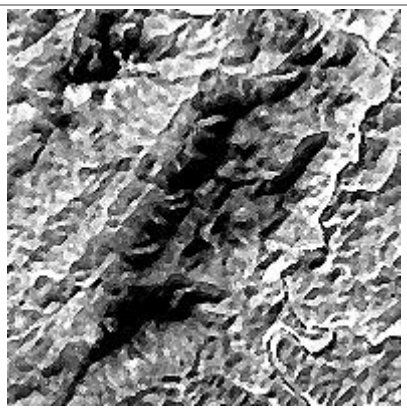
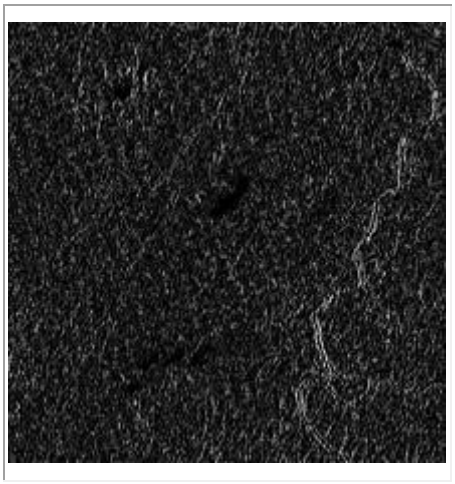
0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1	Если весовые коэффициенты всех пикселов в окне равны, изображение будет сглажено..	
-1 -1 -1 -1 3 -1 -1 -1 -1	Если весовой коэффициент центрального пиксела в окне значительно выше остальных, то изображение станет более резким.	
-1 3 -1 -1 3 -1 -1 3 -1	Если весовые коэффициенты вертикально расположенных пикселов в окне будут значительно выше остальных, на изображении выделятся линии меридионального направления.	

Рисунок 2.4 – Фільтрація зображення

Класифікація – це комп’ютерне декодування зображень або процес автоматичного поділу всіх пікселів зображення на групи (класи), які відповідають різним об’єктам.

Існують різні типи класифікацій:

— класифікація з навчанням;

— класифікація без навчання.

Класифікація з навчанням [11] – це процес порівняння значення яскравості кожного пікселя зі стандартами, в результаті чого кожен піксель належить до

найбільш відповідного класу об’єктів.

Класифікація навчання може використовуватися [12], якщо:

— заздалегідь відомо, які об’єкти є на зображенні;

— на малюнку зображено невелику кількість (до 30) занять;

— ці класи чітко виділяються на малюнку.

Процес класифікації з навчанням включає кілька етапів.

Класи класифікації з навчанням:

— визначення завдань обробки зображень та вибір методу класифікації;

— підбір опорних точок;

— класифікація та оцінка якості результатів.

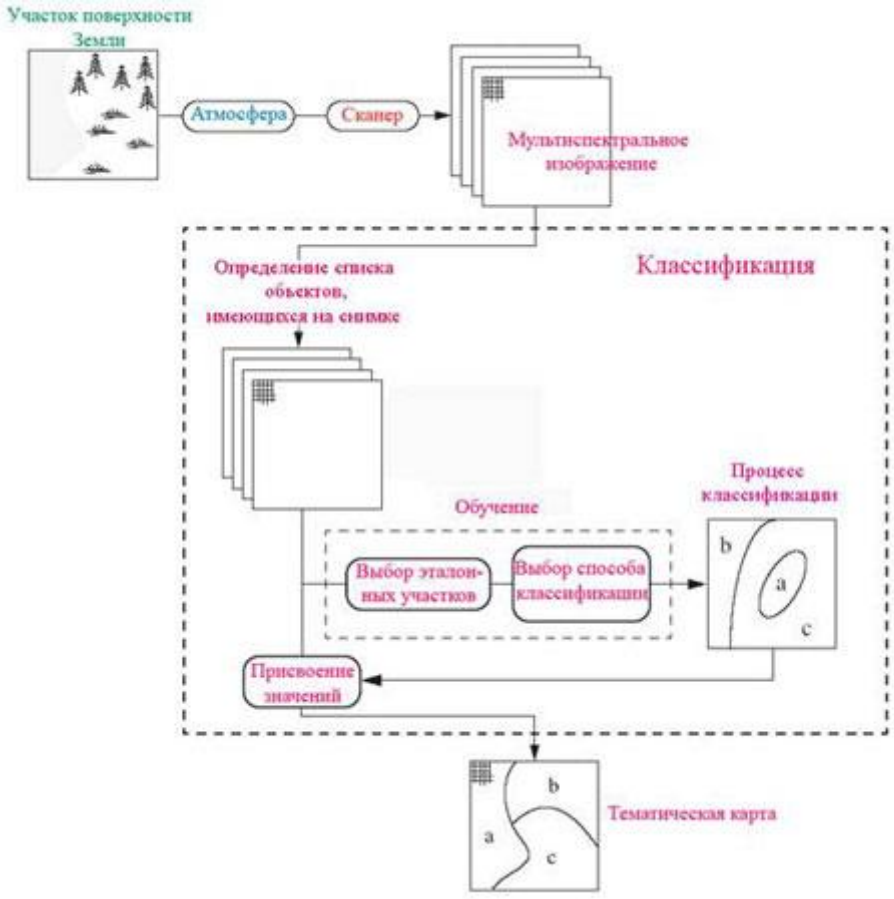


Рисунок 2.5 – Класифікація зображення

На рисунку 2.5 зображені етапи отримання та обробки мультиспектральних зображень.

2.3 Сучасні системи обробки даних ДЗЗ

Однією з найпоширеніших систем віддаленої обробки даних є програмне забезпечення ArcGIS [13].

Обробка даних для дистанційного зондування ДЗЗ – це сфера, яка активно розвивається протягом багатьох років і все більше інтегрується в ГІС. Останнім часом космічна інформація часто використовується в дослідницькій діяльності студентів.

Растрові дані – один з основних типів просторових даних у ГІС. Вони можуть представляти супутникові зображення аерофотознімків, регулярні цифрові моделі рельєфу, тематичні, отримані на основі аналізу геоінформаційної моделі ГІС.

Пакет GIS містить набір інструментів для роботи з растровими даними, які дозволяють обробляти ДЗЗ та виконувати подальші аналізи за допомогою аналітичних функцій GIS. Повна інтеграція з ArcGIS дозволяє швидко перетворювати просторово скоординовані растрові дані з однієї проекції карти в іншу, виконувати перетворення зображення та координацію прив’язки, перетворення реєстру в вектор і навпаки.

Раніші версії ArcGIS вимагали додаткового модуля аналізу зображень для професійної обробки растрових зображень. В останніх версіях ArcGIS до стандартного набору було додано ряд растрових функцій, багато з яких доступні у новому вікні аналізу зображень. Він містить чотири структурні елементи: вікно зі списком відкритих растрових шарів; Кнопка "Параметри", щоб встановити настройки за замовчуванням для деяких інструментів; дві секції інструментів ("Дисплей" та «Обробка»).

У розділі «Дисплей» поєднуються налаштування, які покращують візуальне сприйняття зображень на екрані монітора. У розділі «Обробка» представлений ряд функцій для роботи з растрами. Дослідження показали, що панель «Обробка вікон» у вікні «Аналіз зображень» значно спрощує роботу з растрами. Програма також підтримує керовану та неконтрольовану класифікацію цифрових зображень. Для аналізу можна використовувати функції інших модулів Spatial Analyst і 3D Analyst

[14].

Якщо у нас є майже єдине джерело даних для ГІС, які були оцифрованими картами, тепер зміни простору завжди використовуються для оновлення карти та початкового введення інформації. Такі райони на карті не були зачеплені 1950-х. Професійні критерії або критерії включаються в економічний сектор, повинні бути потрібні поточні картки. Інтенсивне використання системи віддаленого використання (на жаль, згодом порушене) визнало впродовж життя використання загальнодоступного рівня для комерційно доступних космічних зрушень, а також відносно широкий вибір обладнання для найпотужніших систем. З точки зору факторів чи факторів вони сприяли зростанню ДЗЗ.

Космічна фотографія має багато переваг, які роблять її все більш популярною. Це регулярність, простота порядку, існування архівів протягом багатьох років, найбільше висвітлення єдиного зображення, цифрова форма подання. Слід також зазначити, що діяльність Skanex розширює спектр даних, отриманих на приймальних станціях, та просуває нові види ДЗЗ на російському ринку. Тому ми вважаємо, що потреба у програмному забезпеченні для роботи з ДЗЗ та інтерес до нових методів дешифрування незабаром зросте.

У 1994 році DATA+ стала дистриб'ютором ERDAS та розповсюджує відомий програмний пакет для роботи з ДЗЗ під назвою ERDAS IMAGINE [15]. Кілька років тому компанія увійшла до складу міжнародної компанії Leica Geosystems, яка створила структурний підрозділ «ГІС та картографія». А DATE+ є дистриб'ютором Leica Geosystems.

Що стосується програмного забезпечення, Leica Geosystems дуже активно розробляє цілий ряд продуктів на базі ERDAS IMAGINE. Якщо попередні фотограмметричні рішення вже пропонувались у вигляді додаткових модулів (OrthoBASE, OrthoBASE Pro, Stereo Analyst), вони тепер виділяються в новому продукті під назвою Leica Photogrammetry Suite (LPS) [16]. Це означає, що в даний час розробляються дві групи продуктів на основі загальних технологій: ERDAS IMAGINE – як універсальне рішення в області обробки зображень і LPS – як спеціалізоване рішення для фотограмметрії. Цей підхід є цілком логічним,

враховуючи давню традицію відносної незалежності фотограмметрії та тематичної обробки зображень. А загальна архітектура та програмні компоненти гарантують, що ці родини тісно інтегруються в ті проекти, де це необхідно.

Взагалі кажучи, вже давно ведуться розмови про швидке злиття ГІС і систем обробки зображень. А тут ми бачимо картину начебто зворотний - виділення фотограмметричних рішень з єдиного перш сімейства ERDAS IMAGINE. Така ситуація обумовлена не тільки специфікою фотограмметричних задач. Розвиток можливостей персональних комп'ютерів дозволило реалізувати на них ті завдання, які раніше були під силу лише дорогим професійним апаратно-програмних комплексів. Яскравий приклад тому – LPS з його функціями побудови і редагування рельєфу місцевості. Тобто, і ГІС, і обробка зображень, і фотограмметрія - все три технології активно розвивають свій інструментарій, який стає все більш потужним і спеціалізованим. І це, природно, відсуває на більш віддалену перспективу питання повної інтеграції і створення єдиної, наскрізний технології.

Проте, зустрічний рух не припиняється. Leica Geosystems розробляла компоненти ArcGIS для роботи з растровими даними, а також модулі Image Analysis для ArcGIS і Stereo Analyst для ArcGIS. У той же час інформація з бази геоданих ESRI тепер може читатися і редагуватися в продуктах Leica Geosystems. У свою чергу, можливості використання растрових даних в продуктах ArcGIS останнім часом істотно розширилися, в тому числі за рахунок зберігання зображень в форматах ERDAS IMAGINE. Таким чином, незважаючи на активну власний розвиток, все три сімейства (IMAGINE, LPS і ArcGIS) продовжують розширювати можливості взаємодії. І це важливо для великих геоінформаційних проектів, де можливості інтеграції програмних продуктів істотно впливають на успішність реалізації і вартість рішень.

Експерти ДАТА + ведуть моніторинг не тільки вже відомих продуктів і технологій ESRI і Leica Geosystems, але також і всіх перспективних напрямків роботи з геоінформації. Так, розвиток можливостей персональних комп'ютерів і цифрових фотокамер дозволило реалізувати на їх базі тривимірне моделювання складних географічних об'єктів і цілих міст. Відповідно, в нашій компанії був створений

спеціальний проектний відділ для вирішення подібних завдань. Сучасні комп'ютери, дані космозйомки високого дозволу, цифрові фотокамери і програмне забезпечення дозволяють побудувати наскрізну технологію тривимірного моделювання, на виході якої – фотореалістичні віртуальні моделі міст або будь-яких інших територій. На відміну від своїх дерев'яних і пластмасових попередників, такі моделі підтримують просторовий аналіз, їх легко змінювати, додавати нові об'єкти і оцінювати, наскільки вони вписуються в існуюче середовище, програвати різні сценарії розвитку території і т.д. Така модель легко вміщається на жорсткому диску ноутбука і може бути показана в будь-якому місці і в будь-який час. При цьому вона зберігає свою географічно та може бути інтегрована з іншими ресурсами геоінформаційних систем.

Ще один перспективний напрям роботи з ДДЗ - об'єктно-орієнтоване дешифрування зображень. Дешифрування - необхідний крок для отримання корисної інформації з ДДЗ і наповнення нею бази даних ГІС. У цій області дуже довго домінували методи так званої по-піксельної класифікації, інтерпретують кожен піксель зображення незалежно від інших. Нова ж методика спочатку виділяє на зображенні об'єкти, як області відносної однорідності кольору, текстури і яскравості, і вже потім класифікує ці об'єкти як за традиційними спектрально-яркостное ознаками, так і за ознаками геометричним (форма, площа, орієнтація і ін.), Контекстним (входження в більш великі об'єкти або області, близькість до об'єктів певного класу і ін.) і текстурованим. Наша компанія пропонує реалізацію цього підходу у вигляді продукту eCognition німецької компанії Definiens Imaging. Цей продукт - єдина комерційно успішна технологія цього типу для автоматизованого дешифрування космознімків з високою роздільною здатністю, радарних, теплових і аерофотознімків [17].

2.4 Висновки до розділу

У другому розділі ми розглянули основні характеристики космічних знімків, які ми будемо використовувати для класифікації. Також провели аналіз, якими методами

ведеться космічна зйомка.

Розглянули методи обробки даних ДЗЗ та виділи необхідний для нас. Використовуючи класифікацію зображень з навчанням, ми зменшуємо відсоток похибки нашого програмного забезпечення.

Також дізналися про вже існуючі інформаційні системи, проаналізували їх можливості та функціонал, переглянули документацію та вирішили, що саме потрібно для покращення нашого функціоналу та проаналізували, які технічні засоби задовольняють наші потреби.

3 ЗАСОБИ РОЗРОБКИ

Основною метою було обрати гнучку мову програмування, яка дозволила би в подальшому вдосконалювати програмний продукт та додавати новий функціонал. Саме тому було обрано Python як мову програмування та MySQL як базу даних.

3.1 Обґрунтування вибору технологій та їх короткий опис

Мова Python [18, 19] є незалежною, міжплатформною, відкритою мовою програмування, швидкою, потужною і легкою в освоєнні. Вона широко використовується і підтримується.

ESRI остаточно впровадив Python в ArcGIS [20] і розглядає цю мову в якості основного засобу, який задовольнить всі потреби користувачів. Перелічимо деякі переваги Python:

- простота вивчення та придатність як для початківців, так і для професіоналів;
- відмінно масштабований, він підходить як для великих проектів, так і для маленьких одноразових програм, відомих як скрипти;
- портативність і міжплатформність;
- можливість вставлення (написання скриптів в ArcGIS);
- стабільна і впевнена робота [21].

Програмний продукт створений за допомогою мови програмування Python, модулів, які можливо встановити та бази даних MySQL [22]. Графічний інтерфейс створено завдяки вставному модулю tkinter, який додатково встановлюється для мови Python.

Інформаційна система створена у вигляді програмного вікна, яке в подальшому можна буде встановлювати локально на ПК.

База даних MySQL була обрана через свою доступність, швидкість та легкий спосіб доступу та підключення.

3.2 Опис бази даних MySQL

База даних – це набір деяких даних, які зберігаються у впорядкованій формі. Базу даних можна порівняти з картотекою бібліотеки. Наша база даних є папкою на комп’ютері або сервері.

СУБД (система управління базами даних) – це програмне забезпечення по управлінню базами даних, таких як: Oracle Database, MS SQL Server, PostgreSQL [23] та MySQL, яку ми будемо використовувати.

MySQL – це реляційна система управління базами даних, яке є у вільному доступі. Відмінністю цієї системи від інших є можливість користування великою кількістю різних типів таблиць. Тобто користувач може користуватися звичними для нас таблицями, а також працювати з транзакціями на рівні окремих записів.

В рамках серверу таблиці не що інакше, як файл, у якому зберігаються дані конкретного типу. Таблиць в одній базі даних може бути скільки завгодно. І як вже попередньо зазначалося, в таблицях зберігають не все підряд, а конкретні дані, які необхідні або характеризують предметну область, а також дані конкретного типу. Різні таблиці нам потрібні для того, аби дані не змішувались і їх можна було легко відсортувати та знаходити.

Таблиці складаються із стовпчиків та рядків. В одному рядку зберігається інформація про один конкретний запис. Тобто, якщо у нас таблиця, яка містить в собі результати класифікації, то в одному рядку інформація про одну породу. У стовпчиках знаходиться конкретна інформація класифікації для кожної породи. Наприклад, номер класу деревної породи, назва породи та кількість дерев, тобто та інформація, яку ми потребуємо.

MySQL підтримує багато різних типів даних, основні з яких: числа, рядки та дати. При цьому, якщо якомусь стовпчику ми присвоїли певний тип даних, то у ньому

мають зберігатися дані саме такого типу.

MySQL реалізує клієнт-серверну модель. MySQL працює разом з декількома утилітними програмами, які підтримують адміністрування баз даних MySQL. Команди надсилаються до MySQLServer через клієнт MySQL, який встановлений на комп'ютері. MySQL спочатку був розроблений для швидкої обробки великих баз даних. Хоча MySQL зазвичай встановлюється лише на одній машині, він може надсилати базу даних у декілька локацій, оскільки користувачі мають доступ до неї через різні клієнтські інтерфейси MySQL. Ці інтерфейси відправляють оператори SQL на сервер, а потім відображають результати [24].

MySQL має наступні переваги перед PostgreSQL [25]:

- MySQL зазвичай набагато перевершує PostgreSQL за швидкістю роботи. Крім того, в MySQL 4.0 реалізований кеш запитів. Він дозволяє у багато разів збільшити швидкість обробки запитів для сайтів, на яких переважають неодноразово повторювані запити на читання;
- за кількістю користувачів MySQL також набагато перевершує PostgreSQL. Тому код тестується значно прискіпливіше і досвідченим шляхом доведено велика його надійність, ніж у PostgreSQL. MySQL частіше, ніж PostgreSQL, використовується на виробництві, в основному тому, що компанія MySQL AB (раніше - TCX DataKonsult AB) надає високоякісну комерційну технічну підтримку MySQL з моменту появи цієї системи на ринку, а у PostgreSQL до самого останнього часу ніякої підтримки не було;
- MySQL працює в середовищі Windows краще, ніж PostgreSQL. MySQL Server запускається як сьогодення (рідне) Windows-додаток (в NT / 2000 / XP - сервіс), в той час як PostgreSQL запускається в середовищі емуляції, Cygwin. Нам доводилося чути про недостатню стабільності роботи PostgreSQL в середовищі Windows, але самостійно ці відомості до сих пір ми перевірити не могли;

- MySQL оснащений великою кількістю API для інших мов і підтримується великою кількістю існуючих програм, ніж PostgreSQL. See section B Привнесені програми;
- MySQL працює на високонадійних промислових системах 24/7 (включених 24 години на добу 7 днів на тиждень). У більшості випадків ніяких «чисток» в MySQL проводити не потрібно. PostgreSQL ж поки що не може працювати в таких системах, так як іноді доводиться запускати VACUUM для звільнення зайнятого наслідками роботи команд UPDATE і DELETE простору і проводити статистичний аналіз, необхідний для досягнення максимальної продуктивності PostgreSQL. Запускати VACUUM необхідно і після кожного додавання до таблиці декількох стовпців. На напружено працюють системах VACUUM потрібно запускати більш часто, в найгірших випадках - по кілька разів на день. Але ж під час роботи VACUUM (а її робота може тривати години, якщо база даних досить велика) база практично «мертва». Втім, в PostgreSQL версії 7.2 виконання основних функцій цієї програми більше не призводить до блокування бази, і користувачі можуть продовжувати нормально працювати з нею. Нова команда VACUUM FULL береться за справу серйозніше: вона, як і в старих версіях, блокує таблицю і стискає копію таблиці на диску;
- у комплект поставки MySQL входять два тестових пакета, mysql-test-run і crash-me, а також пакет для замірів продуктивності. Тестова система постійно оновлюється, в неї додається код для тестування всіх нових можливостей і майже всіх відтворюваних помилок, які потрапили в поле нашого зору. Перед випуском кожної нової версії ми використовуємо ці пакети для тестування MySQL на декількох платформах. Наші тести значно перевершують за своїми можливостями всі існуючі в PostgreSQL аналоги, і забезпечують високу якість коду MySQL;
- Книг про MySQL вийшло значно більше, ніж про PostgreSQL. Книги про MySQL випустили видавництва O’Reilly, SAMS, Que і New Riders. Всі

можливості MySQL детально описані в документації, так як це є обов'язковою умовою включення нових можливостей в код;

– MySQL підтримує більше стандартних функцій ODBC [26], ніж PostgreSQL;
– MySQL має значно більш потужною реалізацією ALTER TABLE;
– в MySQL передбачена можливість створення таблиць без транзакцій, що необхідно додатків, що вимагають максимально можливої швидкості роботи. Ці таблиці можуть зберігатися в пам'яті, ставитися до типу HEAP-таблиць або дискових MyISAM;
– MySQL може працювати з двома підтримують транзакції оброблювачами таблиць, а саме – InnoDB [27] і BerkeleyDB. Так як всі системи підтримки транзакцій в різних умовах працюють по-різному, це дає розробнику можливість знайти найкраще рішення для умов, в яких буде працювати його система;
– команда злиття таблиць MERGE надає в ваше розпорядження унікальну можливість створити уявлення кількох ідентичних таблиць і працювати з ними як з одного. Це особливо зручно для роботи з журналами, розбитими, наприклад, по місяцях;
– можливість стиснення доступних тільки для читання таблиць, які не скасовує прямого доступу до їх записів, підвищує продуктивність системи, знижуючи кількість операцій зчитування з диска. Це особливо корисно при архівування;
– в MySQL реалізований повнотекстовий пошук;
– є можливість роботи з декількома базами через одне з'єднання (зрозуміло, в залежності від привілеїв користувача).
– система MySQL з самого початку розроблялася з розрахунку на многопоточность, а PostgreSQL використовує процеси. Перемикання контекстів і доступ до загальних даних декількома потоками здійснюється значно швидше, ніж окремими процесами. Таким чином MySQL Server в багатокористувацьких додатках отримує непогану

перевагу в продуктивності, а крім того, таким чином MySQL Server вдається значно ефективніше користуватися перевагами, наданими симетричними мультіпроцесорними системами (SMP);

– в MySQL реалізована значно потужніша система привілеїв, ніж в PostgreSQL. У той час як PostgreSQL забезпечує лише привілеї INSERT, SELECT і UPDATE/DELETE над базою або таблицею, MySQL надає можливість визначення повного набору різноманітних привілеїв на рівні бази, таблиці і стовпці. Крім того, MySQL дозволяє задавати привілеї для комбінацій хост/користувач;
– в MySQL використовується протокол зв'язку між клієнтом і сервером із стисненням даних, що збільшує продуктивність системи в умовах низькошвидкісних каналів зв'язку;
– наскільки нам відомо, тільки в реляційної системі баз даних MySQL Server використовується концепція «дескриптора таблиці». Завдяки цьому створюється можливість роботи з різними низькорівневими типами таблиць з ядра MySQL, причому кожна таблиця може бути оптимізована для різних характеристик продуктивності;
– усі типи таблиць у MySQL (крім InnoDB) реалізовані у вигляді файлів (одна таблиця на файл), що значно спрощує створення резервних копій, передач, видалення та навіть створення символічних зв'язків між базами даних та таблицями, навіть коли сервер не працює;
– наявність утиліти для відновлення і оптимізації таблиць MyISAM (найбільш поширеного типу таблиць в MySQL). Її використання потрібно тільки в разі фізичного пошкодження файлу даних (наприклад, в результаті апаратного збою). Дозволяє відновити більшу частину даних;
– оновлення (апгрейд) MySQL проходить абсолютно «безболісно». При модернізації MySQL немає потреби в копіюванні/відновленні даних, що доводиться робити при установці більшості оновлень PostgreSQL.

Недоліки MySQL в порівнянні з PostgreSQL:

- підтримка транзакцій в MySQL поки що не так добре перевірена, як в системі PostgreSQL;
- так як MySQL заснований на використанні потоків (threads), поки що ще не безпомилково працюють в деяких ОС, для забезпечення стабільної роботи доводиться або використовувати один з відкомпільованих пакетів;
- блокування таблиць, що застосовується в нетранзакціонних таблицях MyISAM, у багатьох випадках працює швидше, ніж блокування на рівні сторінок, рядків або контроль версій. Недолік цього підходу в тому, що якщо не враховувати механізм роботи блокування таблиць, один тривалий запит може надовго заблокувати таблицю. Зазвичай цього ефекту можна уникнути, прийнявши відповідні заходи при розробці програми;
- за допомогою UDF (user-defined functions, визначені користувачем функції) можливості MySQL можна розширити і доповнити звичайними SQL-функціями або їх об'єднаннями. Але це зробити не так просто, та й система не настільки гнучка в цьому відношенні, як PostgreSQL;
- в MySQL складніше організовувалися поновлення, що зачіпають кілька таблиць відразу. Втім, це було виправлено в MySQL 4.0.2 реалізацією багатотабличного UPDATE і в MySQL 4.1 - за допомогою підзапитів. В MySQL 4.0 можна одночасно видаляти дані з декількох таблиць [28].

Завдяки спеціальній мові запитів SQL [29] клієнт та сервер мають змогу взаємодіяти.

Основний набір операцій SQL:

- створення в базі даних нових таблиць;
- додавання до таблиці нових записів;
- зміна записів;
- видалення записів;
- вибірка записів із однієї або декількох таблиць.

3.3 Опис мови програмування Python

Python – це скриптова мова програмування. Він універсальний, тому підходить для вирішення різноманітних завдань і багатьох платформ, починаючи з iOS і Android і закінчуючи серверними операційними системами. Python використовується в веб-розробці, створенні десктопних і мобільних додатків, програмуванні ігор, а також в аналітиці та машинному навчанні.

Python іноді критикують за те, що він – інтерпретований і буває повільним, в порівнянні з компільованими мовами, такими як C. Однак використання байткодової компіляції і той факт, що переважна частина важкої роботи виконується програмними бібліотеками, означають, що продуктивність Python часто буває дивно хороша – до того ж є безліч прийомів, які дозволяють поліпшити продуктивність програм, якщо це необхідно.

Версії інтерпретатора Python з відкритим вихідним кодом знаходяться у вільному доступі для всіх основних операційних систем. Python ідеально підходить для всіх видів програмування, від оперативних разових сценаріїв до створення величезних і складних систем. Python може навіть виконуватися в інтерактивному режимі (в командному рядку), дозволяючи вводити одноразові команди і короткі програми і відразу отримувати результати. Це ідеальний сценарій для виконання оперативних обчислень або з'ясування того, як працює конкретна бібліотека.

Перше, на що розробник на Python звертає увагу, в порівнянні з іншими мовами, такими як Java або C ++, - це виразність мови: то, що на Java потребують 20-30 рядків програмного коду, на Python часто займає менше 10 рядків.

Сьогодні використовуються два основні різновиди Python: Python версії 2.x існує вже протягом багатьох років і все ще широко використовується, в той час як Python версії 3.x, яка не є назад несумісною з Python 2, стає все більш популярною, враховуючи, що ця версія Python взята за основу для подальшого розвитку.

Один з головних чинників, що стримують повсюдне прийняття Python 3, - це відсутність підтримки сторонніх бібліотек. Ця проблема була особливо гострою щодо

бібліотек Python, що використовуються для розробки геопросторових додатків, оскільки вони часто залежать від окремих розробників або мають вимоги, що не були сумісними з Python 3 протягом досить тривалого часу. Проте всі основні бібліотеки, які використовуються в цій книзі, тепер в рівній мірі можна виконувати, використовуючи Python 3.

Термін «розробка геопросторових додатків» означає процес написання комп'ютерних програм, які здатні отримувати доступ до такого типу інформації, управляти нею і її візуалізувати.

Внутрішні геодані представлені у вигляді серії координат, часто у вигляді значень широти і довготи. Крім того, нерідко також присутні додаткові атрибути, такі як температура, тип ґрунту, висота або назва орієнтира. Одиночний набір геоданих може описувати до декількох тисяч (або навіть мільйонів) точок даних.

У зв'язку з тим, що в роботі задіяно таку велику кількість даних, геопросторову інформацію прийнято зберігати в базах даних. Значна частина цієї книги буде присвячена прийомам зберігання вашої геоінформації в базі даних і отримання до неї доступу ефективним чином.

Геодані надходять в самих різних формах. Різні постачальники географічних інформаційних систем (ГІС), або, точніше, геопросторових інформаційних систем, за останні роки створили свої власні формати файлів, а різного роду організації встановили свої власні стандарти. І тому, щоб прочитати файли потрібного формату, необхідно при імпорті геоданих в вашу базу даних користуватися бібліотекою Python.

На жаль, не всі точки геоданих сумісні. Точно так само, як відстань величиною 2.8 одиниці може мати зовсім різні значення в залежності від того, використовуєте ви кілометри або милі, конкретне значення координати може вказувати на будь-яке число точок на викривленій поверхні Землі, в залежності від того, яка картографічна проекція використовується.

Картографічна проекція – це спосіб подачі земної поверхні в двох вимірах. Щоб порівняти або поєднати два набори геоданих, часто необхідно конвертувати дані з однієї проекції в іншу.

Десятиліття тому можливості розробки геододатків були в значній мірі більш

обмежені, ніж сьогодні. Нормою для роботи і візуалізації геоданих були професійні (і надзвичайно дорогі) географічні інформаційні системи. Інструментальні засоби з відкритим вихідним кодом там, де вони були доступні, були незрозумілі і складні у використанні. Більш того, все працювало виключно на настільному комп'ютері – поняття роботи з геоданих в Інтернеті було не більше, ніж віддаленій мрією.

У 2005 році компанія Google випустила два продукти, які повністю змінили характер геопрограмування: картографічна веб-служба Карти Google і безкоштовна програма перегляду карт на основі тривимірної моделі земної кулі Google Планета Земля дозволили будь-якому з веб-браузером або настільним комп'ютером переглядати геодані і з ними працювати. Не вимагаючи експертних знань і років напрацювання досвіду, вони дозволили навіть чотирирічній дитині безпосередньо переглядати і управляти інтерактивними картами світу.

Продукти Google не досконалі: картографічні проекції свідомо спрощені, призводять до помилок і проблем при відображенні на екрані накладень. Ці продукти є загальнодоступними виключно для некомерційного використання і майже не містять функціоналу, пов'язаного з виконанням геоаналіза. Незважаючи на ці обмеження, вони надали на процес розробки геододатків величезний вплив. Люди дізналися про відкриті можливості при використанні карт, а лежать в їх основі геодані стали настільки поширені, що тепер зазвичай включають вбудовані картографічні інструменти навіть в стільникові телефони.

Поряд з автономними інструментами і бібліотеками з'явилося багато геопросторових прикладних програмних інтерфейсів, англійський термін Application Programming Interface (API). Компанія Google запропонувала ряд програмних інтерфейсів, які можуть використовуватися для впровадження карт і виконання обмеженого геоаналіза на веб-сайті. Інші сайти, такі як геокодер OpenStreetMap, який ми використовували раніше, дозволяють виконувати різні геопросторові завдання, які було б важко вирішити, якби ви були обмежені використанням ваших власних даних і програмних ресурсів.

У міру того як від зростаючого числа джерел надходить все більше і більше геоданих, і оскільки одночасно з цим збільшується число інструментів і систем, які

можуть працювати з цими даними, нагальним стає завдання встановлення для геопросторових даних відповідних стандартів. Міжнародна організація по стандартизації Відкритий геопросторовий консорціум, англійський термін Open Geospatial Consortium (OGC), прагне займатися саме тим, що впроваджує ряд стандартних форматів і протоколів спільного використання та зберігання геоданих. Ці стандарти, включаючи GML, KML, GeoRSS, WMS, WFS і WCS, пропонують універсальну мову, на якій можна описувати геодані. Такі інструменти, як комерційні і відкриті геоінформаційні системи, безкоштовна програма для перегляду карт Google Планета Земля, різноманітні веб-API і спеціалізовані геопросторові бібліотеки на зразок динамічної бібліотеки OGR, – всі вони в змозі працювати з цими стандартами. По-справжньому важливий аспект інструментів геообробки полягає в їх здатності розпізнавати дані і транслювати їх між цими різними форматами.

У міру того як пристрої з вбудованими GPS-приймачами стають все більш масовими, стало можливим виконувати запис даних про місцезнаходження, виконуючи при цьому іншу задачу. Геолокація, тобто визначення і/або запис вашого місця розташування, в той час як ви робите щось ще, набуває все більшого поширення. Соціальна мережа Twitter, наприклад, тепер дозволяє робити запис і відображати своє місце перебування, коли ви публікуєте оновлення статусу. У міру наближення до свого офісу технологічно складне програмне забезпечення, що відстежує список поточних справ, тепер автоматично може приховати ті завдання, які не можуть бути виконані в даному місці. Крім того, ваш телефон може нагадати вам, хто з ваших друзів знаходиться неподалік, а результати пошуку можуть бути відфільтровані так, щоб показувати тільки прилеглі підприємства [30].

Це інтерпретована мова – він не компілюється, тобто до запуску вдає із себе звичайний текстовий файл. Програмувати можна практично на всіх платформах, мова добре спроектована та логічна.

Також є пакет ArcPy, який дозволяє виконувати функції системи GIS, які доступні в ArcGIS. Найзручніше використовувати ArcGIS для дистанційного зондування Землі.

Python можна використовувати для написання доповнень та скриптів для

готових програм. Наприклад, реалізувати логіку гри. Він також може бути використаний для створення інших модулів.

Сценарії часто написані на Python, які вбудовані в програми іншими мовами для автоматизації будь-якого завдання.

Python містить структури даних, такі як списки, n-кортежі та словники. Списки схожі на одновимірні поля (але ви можете використовувати Список містить списки - багатовимірні поля), n-кортежі - фіксовані списки, словники - та списки, але індекси можуть бути будь-якого типу, а не лише числові. "Поля" в Python можуть містити дані будь-якого типу, тобто в одному полі можуть бути числові, термінові та інші типи даних. Поля починаються з індексу 0, а останній елемент можна отримати з індексу -1.

Python - одна з найбільш широко використовуваних мов у Data Science. Використовується для написання алгоритмів машинного навчання та аналітичних програм. Завдяки цьому обслуговуються сховища даних та хмарні сервіси.

Незважаючи на всі переваги, ця мова має свої недоліки.

У Python є реалізація PyPy, близька за швидкістю Java, але не має всіх можливостей мови оригіналу. Python не підходить для завдань, які потребують великого обсягу пам'яті - краще вирішити їх вставками в C або C ++.

Ще один недолік – сильна залежність мови від системних бібліотек, що ускладнює передачу в інші системи. Для цієї мети існує інструмент Virtualenv, але він має і свої недоліки: надмірність методів повної ізоляції, барлінг, дублювання системних бібліотек.

Інша проблема полягає в тому, що Блокування глобального інтерпретатора (GIL) не дозволяє одночасно працювати декількома потоками Python у реалізації CPython. Однак GIL можна відключити на деякий час, як це робиться в математичному пакеті NumPy. [31]

Також з його допомогою можна аналізувати довільні (scrapping) дані з інтернету. Наприклад, в Google Python застосовують для індексації сайтів.

3.4 Висновки до розділу

Обрали технології розробки, які відповідають потребам програмного продукту і сприяють швидкодії системи, що значно впливає на швидкість аналізу даних. Детально розглянули кожні з технологій, структуру додаткиових програмних забезпечень та виділили недоліки та переваги кожної. Обрані технології також будуть сприяти подальшому розвитку програмного продукту та окремих модулів.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ІНФОРМАЦІНОЇ СИСТЕМИ

Ґрунтуючись на описаних в розділі 3 технологіях, в даному розділі описано архітектурне рішення, структура проекту, організація бази даних та алгоритми роботи серверів.

4.1 Опис архітектурного рішення

Система не має авторизації, так як у програмі відсутні рівні доступу. Кожен користувач має однаковий функціонал незалежно від прав доступу.

Для того, щоб розпочати роботу достатньо буде запустити програму та натиснути кнопки, які нам потрібні та цікаві.

Під час розробки програми було прийнято рішення розподілити функціонал на окремі кнопки, щоб зробити роботу простішою та швидшою. Тому кожен блок відведений під різну кнопку.

Для графічного інтерфейсу було використано модуль tkinter. Програма повинна мати зручний для користувача інтерфейс, саме тому ми використовуємо дану бібліотеку. Для зрозумілого користування нам потрібні кнопки, меню, поля для вводу та місце для виводу необхідної інформації. Тобто саме це формує зрозумілу та зручну взаємодію користувача та програми. Це зручно, адже користувач натискає на кнопки і відразу робить необхідні для нього маніпуляції.

Для мови програмування Python такі віджети включені в спеціальну бібліотеку - tkinter. Якщо її імпортувати в програму (скрипт), то можна користуватися її компонентами, створюючи графічний інтерфейс [32].

В сучасних операційних системах будь-який додаток користувача вкладено в вікно, яке назвемо головним, тому що в ньому розташовуються всі інші віджети.

Послідовність кроків при створенні графічного додатку має свої особливості. Програма повинна виконувати своє основне призначення, бути зручною для користувача, реагувати на його дії.

При створенні кнопки нам потрібник обробник, який виконає дію, яка нам необхідна. Алгоритм створюється у вигляді функції, в якій крок за кроком описуємо наші дії. Саме після виклику цієї функції наша кнопка відтворює те, для чого призначена.

Для того, щоб наші віджети з'явилися у вікні нашої програми потрібно використати метод pack.

Для відображення головного вікна необхідно визвати спеціальний метод mainloop.

Далі в програмі виводяться зображення та діаграми, саме для цього нам потрібна бібліотека PIL (Python Imaging Library). Через Pillow можна легко відкрити зображення і відобразити його на екрані через зовнішню програму [33]. Для цього нам потрібно імпортувати утиліту Image та скористуватися методом open, який зберігається в даній утиліті. Для відображення зображень у програмі використовуємо метод show, який ініціює вивід на екран.

Хоча розширену обробку зображення (розпізнавання обличчя, оптичний потік тощо), як OpenCV, неможливо виконати, проте може бути виконана проста обробка зображень, така як зміна розміру (масштабування), обертання та обрізка (часткове вирізання).

Оскільки PIL простіша і легша для розуміння, ніж OpenCV, її краще використовувати залежно від мети [34].

Саме зображення формату TIFF ми використовуємо для класифікації наших зображень, щоб отримати дані для нашої програми.

Графіки кореляції використовуються для візуалізації взаємозв'язку між 2 або більше змінними. Тобто, як одна змінна змінюється по відношенню до іншої.

Упорядкована гістограма ефективно передає порядок ранжирування елементів. Але, додавши значення показника над діаграмою, користувач отримує точну інформацію від самої діаграми [35].

NumPy – це бібліотека мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих (і дуже швидких) математичних функцій для операцій з цими масивами.

Основним об’єктом NumPy є однорідний багатовимірний масив (в numpy називається numpy.ndarray). Це багатовимірний масив елементів (зазвичай чисел), одного типу.

Найбільш важливі атрибути об’єктів ndarray:

- ndarray.ndim - число вимірювань (частіше їх називають "осі") масиву;
- ndarray.shape - розміри масиву, його форма. Це кортеж натуральних чисел, що показує довжину масиву по кожній осі. Для матриці з n рядків і m стовпів, shape буде (n, m). Число елементів кортежу shape одно ndim;
- ndarray.size - кількість елементів масиву. Очевидно, дорівнює добутку всіх елементів атрибута shape;
- ndarray.dtype - об’єкт, що описує тип елементів масиву. Можна визначити dtype, використовуючи стандартні типи даних Python. NumPy тут надає цілий букет можливостей, як вбудованих, наприклад: bool_, character, int8, int16, int32, int64, float8, float16, float32, float64, complex64, object_, так і можливість визначити власні типи даних, в тому числі і складові;
- ndarray.itemsize - розмір кожного елемента масиву в байтах;
- ndarray.data - буфер, який містить фактичні елементи масиву. Зазвичай не потрібно використовувати цей атрибут, так як звертатися до елементів масиву найпростіше за допомогою індексів [36].

Головною особливістю numpy є об’єкт array. Масиви схожі зі списками в python, виключаючи той факт, що елементи масиву повинні мати однаковий тип даних, як float і int. З масивами можна проводити числові операції з великим обсягом інформації в рази швидше і, головне, набагато ефективніше ніж зі списками [37].

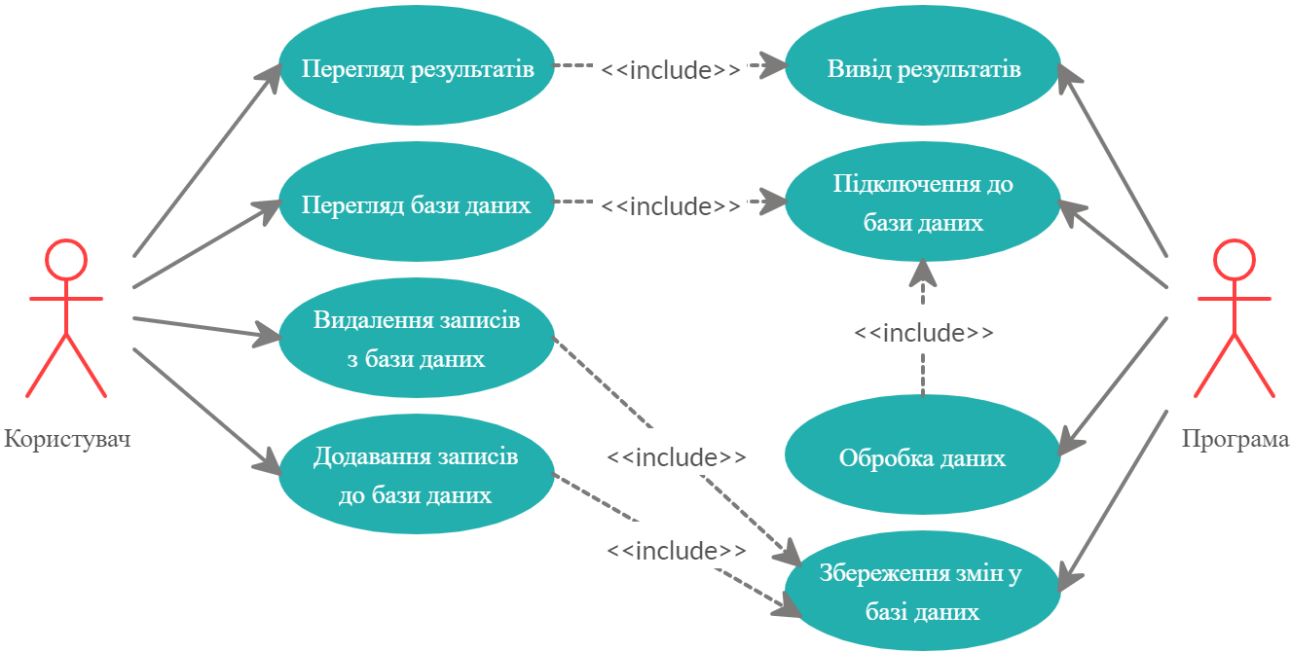


Рисунок 4.1 — Діаграма прецедентів для користувача та програми

На рисунку 4.1 зображено, як пов’язані дії користувача з діями в програмі. Кожна кнопка викликає свій модуль, який працює з кожною конкретною поставленою задачею. Ніякі зміни в програмі не будуть зроблені, поки користувач не використає кожну з функцій.

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Рисунок 4.2 – Формула Баєса

Ліва частина рівняння - апостеріорна оцінка ймовірності події A за умови настання події B (т. ч. умовна ймовірність).

P(A) - ймовірність події A (основна, завжди апріорна оцінка);

P(B | A) - ймовірність (також умовна), яку ми отримуємо з наших даних;

P(B) - константа нормування, яка обмежує ймовірність значенням 1.

Це коротке рівняння є основою баєсівського методу.

Якщо цю формулу використати для нашого випадку, враховуючи, що на ліс впливають пожежі та негоди, то ми отримаємо наступне:

$$P(\text{дерева}|\text{негода}) = \frac{P(\text{негода}|\text{дерева}) * P(\text{дерева})}{P(\text{негода})}, \text{де}$$

P(дерева|негода) – ймовірність появи кількості дерев при умові, що не відомо ймовірності впливу негоди на це.

Рівняння включає в себе в себе константу нормування P(негода) – ймовірність того, що негода (пожежа, мала кількість опадів, наводнення) була. Цю константу ми можемо знайти таким чином:

$$P(\text{негода}) = P(\text{негода}|\text{дерева}) * P(\text{дерева}) + P(\text{негода}|-\text{дерева}) * P(-\text{дерева})$$

P(негода|-дерева) – це ймовірність того, що сталась негода, коли нема дерев.

P(-дерева) – це ймовірність того, що дерев нема.

Це рівняння дозволить нам дізнатись наскільки вірна наша класифікація. Якщо наші ймовірності однакові, то результати класифікації рівні. Якщо ні – в нашій класифікації є похибки через ряд причин, які розглянемо далі.

4.2 Алгоритм отримання даних з бази даних MySQL

База даних нам потрібна для того, щоб програмний продукт отримав дані для аналізу та була можливість реалізувати увесь функціонал.

Під окремою кнопкою є інтерфейс для доступу до бази даних, де користувач може додати або видалити записи. Це потрібно для того, щоб не використовувати готові програми MySQL для доступу. Тому користувач, який не знає мови SQL не зобов’язаний витрачати свій час на вивчення цього питання. Програмний продукт робить це самостійно, користувачу потрібно лише ввести інформацію та натиснути необхідну кнопку.

MySQLdb – це бібліотека, що дозволяє підключитися в MySQL з Python, вона написана на мові C, воно безкоштовна в використанні і є відкритим вихідним кодом.

Спочатку ми встановлюємо бібліотеку та імпортуємо в наш проект за допомогою команди: import MySQLdb. Потім створюємо підключення за допомогою метода Connect:

conn=MySQLdb.connect(‘localhost’, ‘login’, ‘password’, ‘database’), де

- localhost – місце, де розташована ваша база даних. Так як вона у нас розташована локально, тому ми вказуємо саме localhost;
- login – логін, який вказували при встановлені MySQL. Якщо нічого не змінювали, то за замовчуванням він буде root;
- password – пароль, який вказали при встановлені MySQL;
- database – назва бази даних, до якої потрібно підключитися та в подальшому користуватися. У нашому випадку це база даних Diplom.

Далі нам необхідно створити об’єкт нашого підключення за допомогою метода cursor(), який є в нашому модулі MySQLdb.

Метод execute() виконує SQL запит, який ми вкажемо. Наша програма виконує запити SELECT, INSERT INTO та DELETE. Наш метод повертає нам об’єкт класу Connection. Для отримання цих даних нам потрібно використати метод fetchall().

В кінці нам обов’язково потрібно закрити наше підключення до бази даних та серверу, тому ми використовуємо метод close().

Також особливістю користування базою даних через Python є те, що всі дані ми отримуємо в форматі tuple (кортеж). Кортеж – це колекція, отримана та незмінна. На Python кортежі написані з круглими дужками. Кортеж захищений від змін, як навмисних (що погано), так і випадкових (що добре). Навіщо вони нам, якщо їх неможливо змінити? Ми маємо змогу робити всі операції над списками, що не змінюють список (додавання, множення на число, методи index () і count () і деякі інші операції). Можна також по-різному змінювати елементи місцями і так далі.

Тому діставши значення ми маємо змогу тільки аналізувати та працювати з ними. Змінити записи в базі даних ми можемо лише через правильні запити. Саме через це в програмі є окремі блоки для роботи з базою даних, аби все працювало правильно та чітко.

Для того, щоб дані з бази даних у нашій програмі мали схожий вигляд таблиці, ми використовуємо віджет treeview, який знаходиться в модулі tkinter. Цей віджет дещо відрізняється від схожих модулів в інших мовах програмування, де таблиця інтерфейсу самостійно підстроюється під таблицю бази даних. У Python таблицю потрібно настроювати вручну за допомогою методів. Тому настроївши таблицю у

коді, ми зможемо використовувати лише такий каркас. Навіть якщо ми змінимо структури таблиці напряму в MySQL, у нашому інтерфейсі нічого не зміниться, поки ми не змінимо каркас у програмі. Кількість колонок ми встановлюємо завдяки методу columns. Далі, використовуючи метод column, ми задаємо ширину стовпчика. Для того, щоб підписати стовпці ми використовуємо метод heading().

Щоб наші дані можна було додати у наш створений каркас таблиці, нам необхідно вставити наші кортежі через метод insert(), який є в віджеті treeview. Далі виводимо на екран за допомогою методу pack().

4.3 Географічні інформаційні системи

Абревіатура ГІС позначає географічні інформаційні системи, англійський термін Geographic Information Systems (GIS). Зазвичай під ними маються на увазі складні обчислювальні системи, пов’язані зі зберіганням геоданих, управлінням ними і їх візуалізацією. Крім того, ця абревіатура може також використовуватися для позначення більш загального поняття - геоінформатики, тобто науки, на базі якої застосовуються системи ГІС.

Робота з геоданих ускладнюється тим, що ви маєте справу з математичними моделями земної поверхні. У багатьох випадках легко уявити Землю як сферу, на якій можна розмістити дані. Це було б просто, якщо б не одне «але»: результати будуть неточними - Земля більше схожий на приплюснутий сфероїд, ніж на ідеальну сферу. Ось ця відмінність, а також ряд інших математичних складнощів і надають процесу подання точок, ліній і площ на земній поверхні досить складний характер. Розглянемо деякі ключові поняття ГІС, які необхідно знати при роботі з геоданих.

Географічне положення, або місце розташування, - це точка на земній поверхні. Один з найбільш поширених методів вимірювання географічного положення передбачає використання координат, що визначаються двома числами: широтою і довготою. Що ці цифри означають і в чому їх користь?

Будь-яку задану точку на земній поверхні можна з’єднати лінією, що виходить із

центру Землі.

Широта точки – це кут, з яким лінія проходить в напрямку північ-південь щодо площині екватора.

У той же час довгота точки – це кут, з яким ця лінія проходить в напрямку схід-захід щодо довільної початкової точки (зазвичай береться географічне положення Королівської обсерваторії в Грінвічі, Англія).

Прийнято, що позитивні значення широти знаходяться в північній півкулі, тоді як її негативні значення – в південній півкулі. Точно так же позитивні значення довготи лежать на схід від Грінвіча, і негативні значення – західніше від нього. Таким чином, широти і довготи покривають всю Землю.

Горизонтальні лінії, що представляють точки рівній широти, називаються паралелями, а вертикальні лінії, що представляють точки рівній довготи, називаються меридіанами. Меридіан в нульовий довготі часто називають головним меридіаном. За визначенням паралель в нульовий широті – це екватор Землі.

Працюючи зі значеннями широти і довготи, слід пам’ятати про два моменти:

- про західні довготи в основному негативні, але бувають ситуації (особливо це стосується даних, які обмежені територією США), коли західні довготи дані як позитивні значення;
- про значення довготи змінюють знак і напрямок в точці ± 180°. А саме, подорожуючи на схід, довгота буде 177, 178, 179, 180, -179, -178, -177 і т. д., що може вкрай заплутати базові обчислення відстані, якщо покладатися тільки на свої сили, а не на програмну бібліотеку, яка може зробити цю роботу за вас.

Значення широти і довготи відноситься до так званого геодезичного положенню. Геодезичне положення ідентифікує розрахункову точку на поверхні Землі, незалежно від того, що може перебувати в цьому місці. Значна частина даних, з якими ми будемо працювати, містить геодезичні положення. Проте можна зустріти і інші методи опису місця розташування на поверхні Землі. Наприклад, місце проживання фізичної особи – це, зрозуміло, просто вуличний адресу, який являє собою ще один абсолютно допустимий (хоч і менш точний, з наукової точки зору) спосіб визначення

географічного положення. Аналогічним чином місцезнаходження юридичної особи містить інформацію про те, в межах яких адміністративно-територіальних кордонів (таких як виборчий округ, регіон або місто) воно знаходиться, так як в деяких контекстах ця інформація представляє важливість.

Відстань між двома точками на земній поверхні можна уявити по-різному. Ось кілька прикладів:

- кутова відстань – це кут між двома променями, що виходять з центру Землі через дві точки. Кутові відстані зазвичай використовуються в сейсмології; з ними можна зустрітися під час роботи з геоданих;
- лінійна відстань – це саме те, що люди зазвичай мають на увазі, коли говорять про відстані, – наскільки далеко дві точки стоять один від одного на земній поверхні. Його часто називають відстанню «по прямій». Незабаром ми обговоримо це поняття більш детально, хоча слід усвідомити, що лінійні відстані не такі прості, як здається;
- пройдену відстань, або відстань ходу (traveling distance) - з лінійними відстанями («по прямій») все зрозуміло, але не дуже-то багато набереться людей, які вміють літати, як птахи. Інший корисний метод вимірювання відстані полягає у вимірюванні того, як далеко потрібно фактично промандрували, щоб дістатися від однієї точки до іншої, при цьому зазвичай рухаючись по дорозі або іншому очевидному маршруту.

Якщо ви припускаєте, що Земля сферична, то обчислити відстань по дузі великого кола між будь-якими двома точками відносно просто; для цього часто використовується формула гаверсінуса. Є і більш складна методика, яка дозволяє більш точно уявити фігуру Землі, втім, формули гаверсінуса буде досить для більшості випадків.

У вересні 1999 року орбітальний модуль Mars Climate Orbiter досяг зовнішніх країв марсіанської атмосфери, Долетівши в космосі протягом 286 днів. Створення апарату обійшлося в цілому в 327 мільйонів доларів. Коли він наблизився до своєї завершальної орбіти, похибка в розрахунку змусила його летіти занадто низько, і в результаті орбітальний модуль був загублений. Розслідування показало, що

акселератори апарату обчислювали прискорення, використовуючи англійські одиниці виміру, в той час як комп'ютер космічного апарату працював з метричними одиницями. Результатом з'явилися катастрофа для НАСА і жорстке нагадування всім нам про важливість розуміння того, в яких одиницях знаходяться ваші дані.

Геопросторові значення можуть вимірюватися в самих різних одиницях. Зрозуміло, відстані можуть вимірюватися в одиницях метричної та англійської системи заходів, проте конкретна відстань може насправді вимірюватися різними одиницями, в тому числі використовуючи наступні:

- дюйм;
- сантиметр;
- міліметр;
- міжнародний фут;
- геодезична миля США;
- миля;
- метр;
- ярд;
- кілометр;
- міжнародна морська миля;
- статутна миля США;
- британська морська миля.

Працюючи з даними, які описують відстані, завжди важливо знати, в яких одиницях вони вимірюються. Більш того, вам часто доведеться конвертувати дані з однієї одиниці вимірювання в іншу.

Кутові заходи теж можуть перебувати в різних одиницях: в градусах або радіанах, і, значить, знову ж вам часто доведеться конвертувати ці одиниці з однієї в іншу.

Якщо підходити строго, то одиниці виміру широти і довготи ідентичні, і тим не менш вам часто доводиться мати справу з різними форматами відображення їх значень. Традиційно значення довготи і широти записуються в форматі «градуси/ хвилини/секунди».

Процес створення двовимірної карти з тривимірної фігури Землі іменується картографічною проекцією. Картографічна проекція – це математичне перетворення, яке «розгортає» тривимірну фігуру Землі і поміщає її на двовимірну площину.

Існують сотні різних картографічних проекцій, але жодна з них не досконала. І дійсно, математично неможливо уявити тривимірну поверхню Землі на двовимірній площині, не вводячи свого роду спотворення; фокус полягає в тому, щоб вибрати таку проекцію, при якій спотворення ніяк не позначається на вашій роботі. Наприклад, деякі проекції точно позначають конкретні ділянки земної поверхні, додаючи головне спотворення в інші її ділянки; такі проекції годяться для картування строго певної ділянки Землі і ніякого іншого. Ще одні проекції спотворюють фігуру країни, залишаючи її площа без змін, в той час як інші проекції роблять протилежне.

За характером спотворень розрізняють три основні групи картографічних проекцій: циліндричні, конічні і площинні. Коротко розглянемо кожну з них.

Є простий спосіб розібратися в циліндричних проекціях – треба уявити, що Земля виглядає як сферичний китайський ліхтар зі свічкою в середині. Якщо помістити цю землю-ліхтар в паперовий циліндр, то свічка «спроектує», тобто перенесе, земну поверхню на внутрішню частину циліндра. І тоді цей циліндр можна «розгорнути», щоб отримати двовимірне зображення Землі.

Зрозуміло, це спрощення – насправді в картографічних проекціях джерела світла для відображення земної поверхні на площину зовсім не використовуються. Щоб досягти того ж самого ефекту, натомість використовуються складні математичні перетворення.

Конічну проекцію отримують шляхом відображення земної поверхні на конус. Потім конус «розгортають», щоб отримати остаточну карту. Ось кілька найпоширеніших типів конічних проекцій: рівновелика проекція Албере, рівнокутна (конформна) конічна проекція Ламберта і рівновіддалена проекція.

Площинна, або азимутальная, проекція на увазі відображення поверхні Землі безпосередньо на площину. Площинні проекції центруються навколо єдиної точки і зазвичай поверхні всієї Землі не показують. Однак ж вони підкреслюють сферичну природу Землі. За багатьма аспектами площинні проекції зображують Землю такою,

якою її можна було б спостерігати з космосу. Ось кілька основних типів площинних проекцій: гномоніческа проекція, рівновелика азимутальна проекція Ламберта і ортогональна проекція.

Як зазначалося раніше, досконалої проекції в природі не існує – кожна проекція деяким чином спотворює земну поверхню. І дійсно, математик Карл Гаус довів, що математично неможливо спроектувати тривимірну геометричну фігуру, таку як сфера, на площину, не ввівши деяких спотворень. Саме з цієї причини існує стільки різних типів картографічних проекцій: деякі проекції більше підходять для якоїсь конкретної мети, але ніяка проекція не може охопити все. Під час створення геопросторових даних або роботи з ними завжди важливо знати, яка картографічна проекція застосовувалася при створенні цих даних. Не знаючи проекції, ви не зможете виконувати ні візуалізації даних, ні точних розрахунків.

Поняття системи координат тісно пов’язано з картографічними проекціями. Є два типи систем координат, в яких необхідно добре розбиратися: картографічні (спроектовані) і географічні (неспроектовані).

Прикладом географічної системи координат є значення широти і довготи. Ці географічні координати прямо вказують на точку, розташовану на поверхні Землі.

Географічні координати корисні тим, що вони можуть точно представляти конкретну точку на поверхні Землі. Разом з тим вони сильно ускладнюють розрахунки відстані і виконання інших геопросторових обчислень. На відміну від них, в картографічній системі координат прямокутні координати вказують на точку, розташовану на площині з двома осями, яка зображує поверхню Землі.

В картографічної (прямокутної) системі координат, як зрозуміло з назви, спочатку використовується картографічна проекція, яка перетворює Землю в двовимірну (картезианьску) координатну площину, після чого на цю площину поміщаються точки. Щоб працювати з картографічною системою координат, потрібно знати, яка проекція використовувалася для створення лежить в основі карти.

В картографічних і географічних системах координат, крім того, передбачається наявність набору початкових (опорних) точок (reference points), які дозволяють визначати, де конкретна точка знаходиться. Наприклад, в географічній

системі координат (широти і довготи) нульове значення довготи представлено нульовим (Грінвічському) меридіаном, тобто лінією, що проходить з півночі на південь через Грінвічську обсерваторію в Англії. Аналогічним чином нульове значення широти виражено нульовою паралеллю, тобто лінією, що проходить навколо екватора.

На відміну від них, для картографічних систем координат зазвичай задають початок відліку координат і одиниці виміру карти. У деяких системах координат також використовуються значення північного зміщення (false northing) і східного зміщення (false easting), щоб скорегувати положення початку відліку координат.

У загальних рисах геодезичний датум – це математична модель Землі, яка використовується для опису розташування на її поверхні. Датум складається з набору опорних точок, або базових геодезичних параметрів, часто в комбінації з моделлю фігури Землі. Опорні точки використовуються для опису місцезнаходження інших точок на поверхні Землі, в той час як модель фігури Землі використовується при відображенні поверхні Землі на двовимірну площину карти. Таким чином, датум використовуються як в картографічних проекціях, так і в системах координат.

Хоча в усьому світі кількість що знаходяться в користуванні датум налічується сотнями, все ж більшість з них має відношення виключно до локальних територій, і лише три вважаються основними. Це так звані опорні датум (reference datums), які охоплюють ширші території і з якими ви, ймовірно, зустрінетеся, коли будете працювати з геоданих:

- NAD 27: північноамериканський датум 1927 роки (North American Datum). Він складається з визначення фігури Землі (використовуючи модель під назвою Сфероїд Кларка 1866) і набору базових параметрів, зосереджених навколо геодезичного центру Ранчо Міда (Meades) в шт. Канзас. NAD 27 можна розглядати як локальний датум з зоною покриття, що включає Північну Америку;
- NAD 83: північноамериканський датум 1983 року. У ньому використана складніша модель фігури Землі (геодезична опорна система 1980 року,

GRS 80). NAD 83 можна вважати локальним датум з зоною покриття, що включає США, Канаду, Мексику і Центральну Америку;

– WGS 84: світова геодезична система 1984 (World Geodetic System). Це глобальні датум, чия зона покриття - вся поверхня Землі. У ньому використана ще одна модель фігури Землі (гравітаційна модель Землі 1996 року, EGM 96) і базові параметри на основі Міжнародного опорного меридіана IERS. Датум WGS 84 дуже популярний. При роботі з геоданих, які охоплюють територію США, WGS 84 практично збігається з NAD 83. Правда, на відміну від нього, WGS 84 використовується супутниками Системи глобального позиціонування GPS, і тому всі дані, отримані від модулів GPS, будуть використовувати цей датум.

На сьогоднішній день найпоширенішим датум є глобальний датум WGS 84, разом з тим у багатьох годинних передбачається використання інших датум. Маючи справу із значенням координати, завжди важливо знати, який датум використовувався під час розрахунку цієї координати. Наприклад, конкретна точка, виражена в датум NAD 27, може бути на відстані в сотні футів від тієї ж самої координати, вираженої в датум WGS 84. Таким чином, життєво важливо знати, який датум використовується для конкретного набору геоданих, і в разі необхідності конвертувати в інший датум.

Формат даних ГІС, як правило, підтримує наступні дані:

– геопросторові дані з описом географічних об’єктів, або геооб’єктів, що те ж саме;

– додаткові метадані, які дають опис самих геопросторових даних, в тому числі використовувані датум і проекцію, систему координат і одиниці вимірювання даних, дату останнього оновлення файлу даних і т. д.;

– атрибути, які надають додаткову інформацію про описувані географічні об’єкти. Наприклад, геооб’єкт, що позначає місто, може мати такі атрибути, як назва, населення, середня температура і т.д.;

– візуально відображається інформація, в тому числі колір або стиль лінії, що використовуються при зображенні об’єкта.

У ГІС використовуються два основних типи даних: дані в растровому форматі

і дані в векторному форматі. Растрові формати зазвичай використовуються для зберігання побітово адресованих графічних зображень, таких як відскановані паперові карти чи аерофото і космічні знімки. У свою чергу, векторні формати представляють просторові дані за допомогою точок, ліній і багатокутників. Використання векторних форматів в ГІС додатках більш поширене, оскільки такі дані компактніше і ними легко управляти [38].

4.4 Висновки до розділу

У цьому розділі розглянули структуру створеного програмного продукту, а також алгоритми деякого функціоналу. Також розібрали програмні модулі, які необхідні для коректного функціонування системи та її швидкодії.

5 Робота користувача з програмною системою

5.1 Системні вимоги для додаткове програмне забезпечення

Для правильної та зручної роботи з програмним продуктом є деякі вимоги до апаратного забезпечення та додаткові програмні забезпечення.

Для роботи мінімально необхідні: процессор Intel Core i3, 4 Гб оперативної пам’яті, 128 ГБ вільного місця на жорсткому диску.

Додатково необхідно встановити програмне забезпечення: MySQL, Python 2.

5.2 Результати виконання програми

Скріншоти для демонстрації роботи програмного продукту.

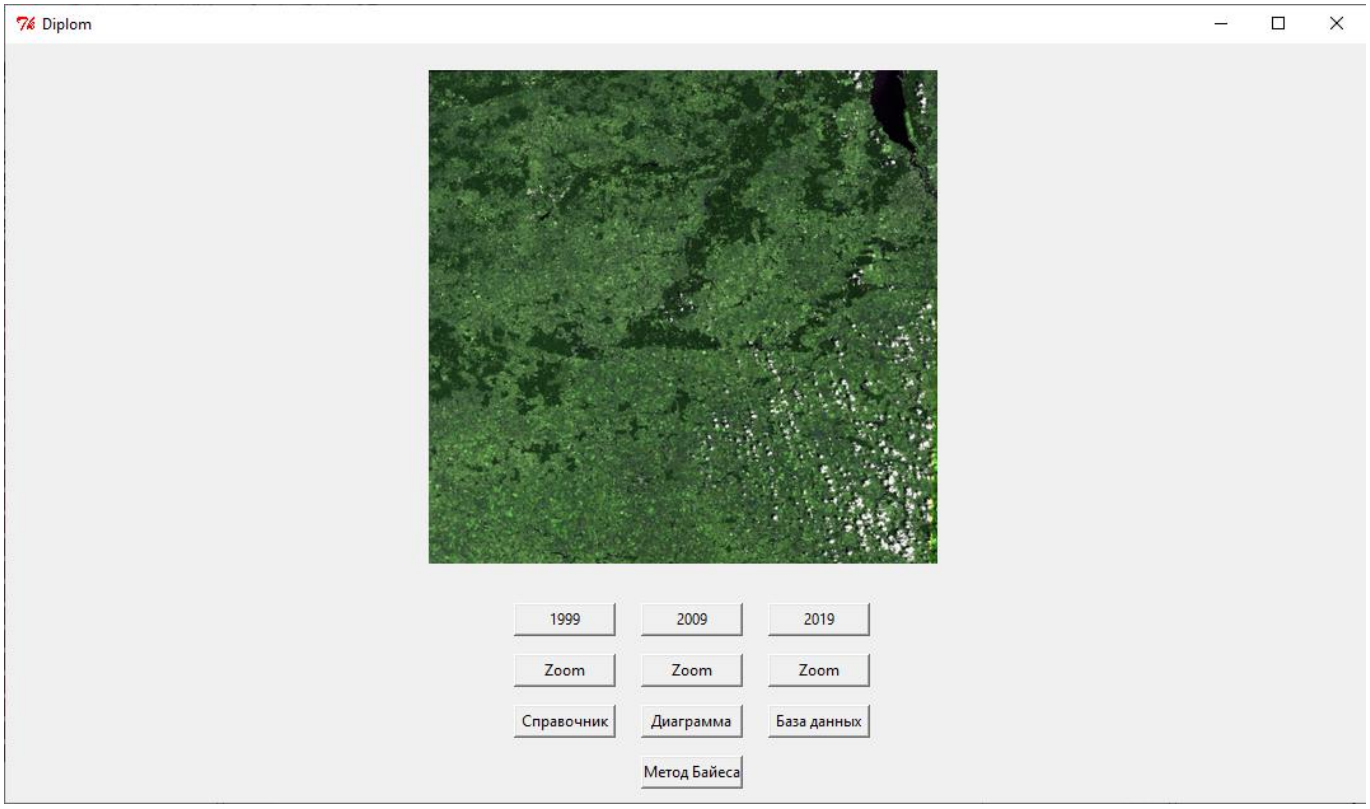


Рисунок 5.1 — Головна сторінка інтерфейсу користувача

На рисунку 5.1 ми бачимо, що є кнопки, при натисканні яких виконується

функціонал програми. Кнопки «1999», «2009», «2019» виводять мініатюрні зображення, за допомогою яких проводили класифікацію лісного покриву. Кнопки «Zoom» дають змогу приблизити зображення та розглянути їх більш детально. Кожна така кнопка знаходиться під відповідним роком зображення. Тобто при натисканні кнопки «Zoom» під кнопкою «1999» ми збільшуємо зображення саме 1999 року.

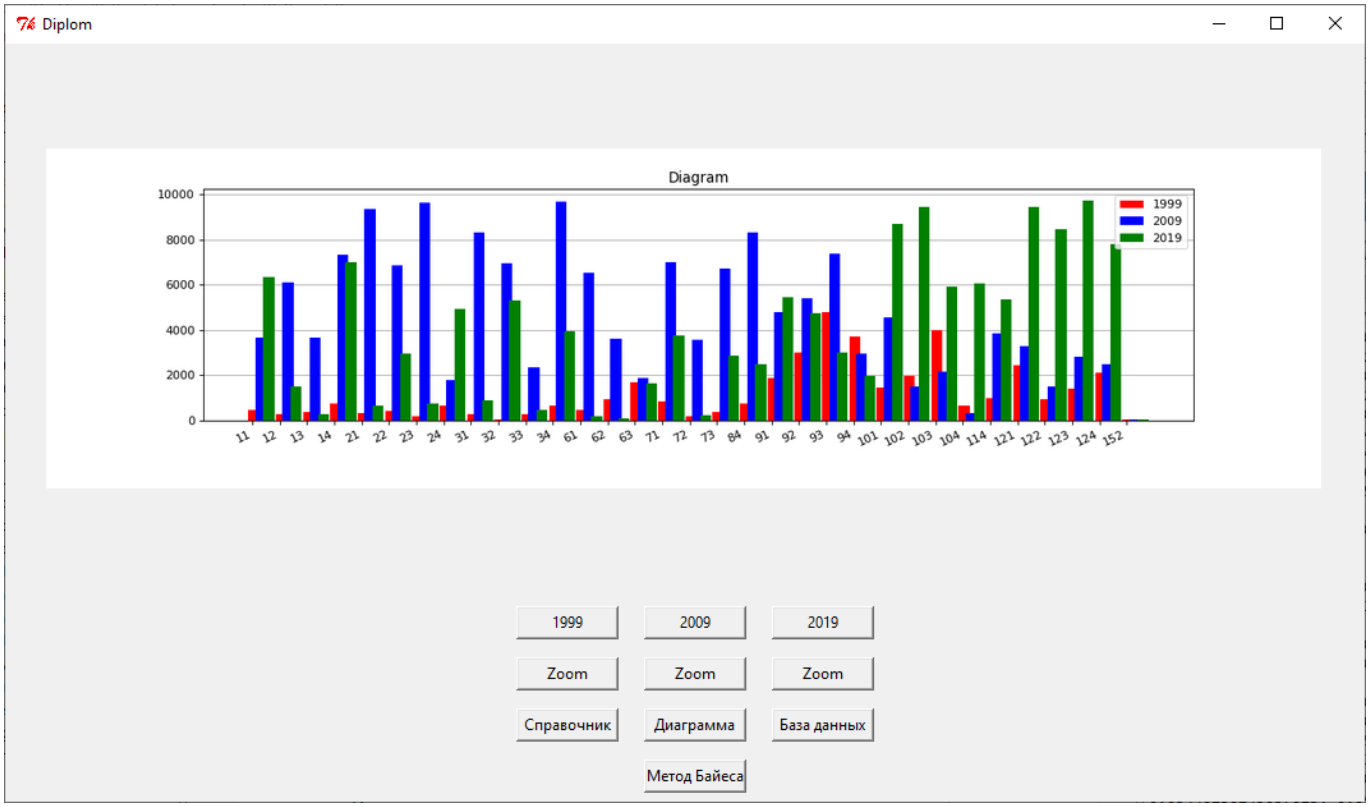


Рисунок 5.2 — Діаграма порівняння лісового покриву за 3 роки

На рисунку 5.2 зображено, що отримає користувач натиснувши кнопку «Діаграма». Дані, отримані з бази даних, дозволяють нам побудувати діаграму, на якій видно зміну кількості порід за 1999, 2009 та 2019 роки. Знизу підписано класами порід, щоб було легше працювати з діаграмою. Щоб було зрозуміліше, яка порода стоїть за номером класу породи, користувач може натиснути кнопку «Довідник» і буде виведено інше вікно, в якому буде перелік порід та номер його класу. Це додаткове вікно можна переміщувати для більшої зручності користування.

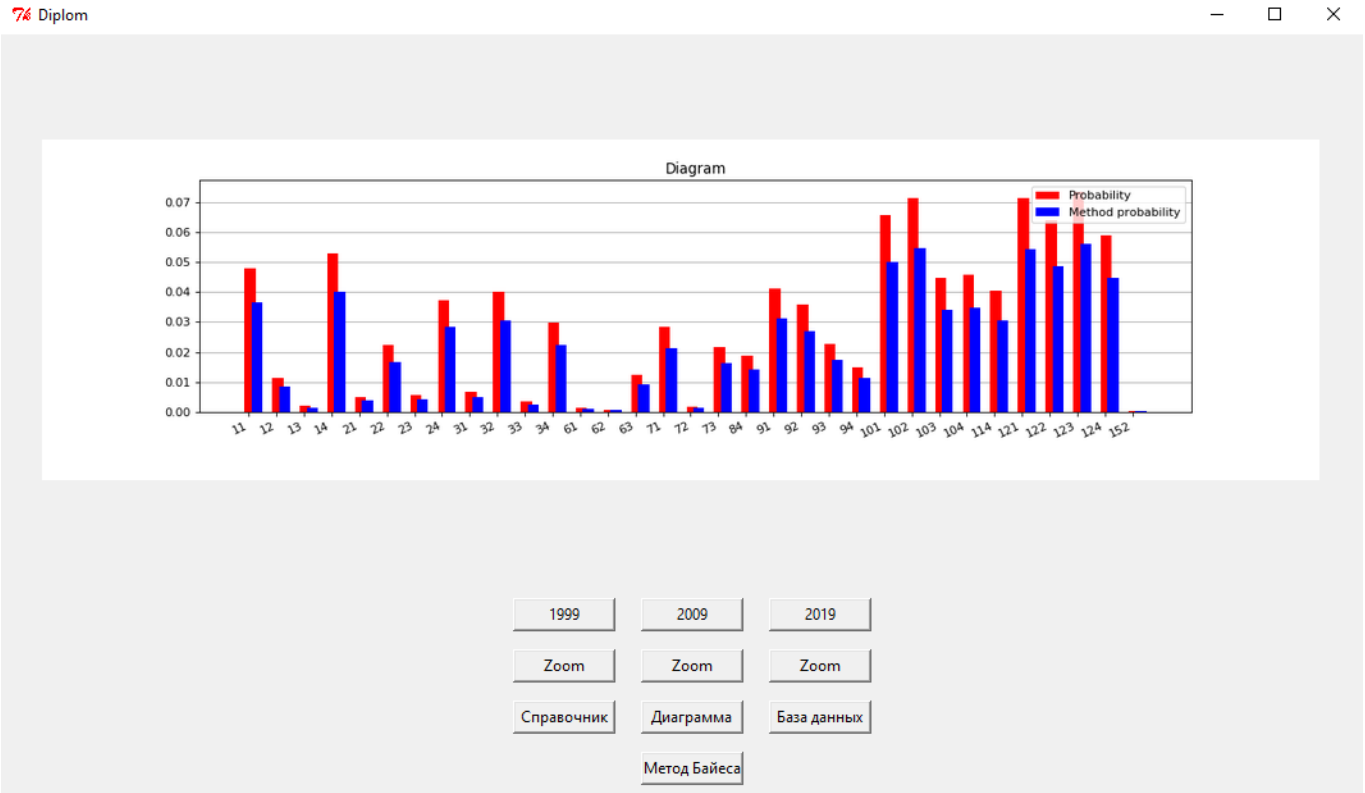


Рисунок 5.3 — Діаграма порівняння ймовірності кількості дерев методом Баєса

На рисунку 5.3 зображено, що отримає користувач натиснувши кнопку «Метод Баєса». Червоним кольором показано ймовірність, яка у нас вийшла за 2019 рік. Синім – ймовірність, яка мала би бути за методом Баєса. Тобто ми можемо порівняти отримані результати з тими, які ми мали б отримати теоритично. Тобто програмне забезпечення може давати похибку. Ця похибка може бути через некоректну класифікацію, невірні дані та похибку при обчислені.

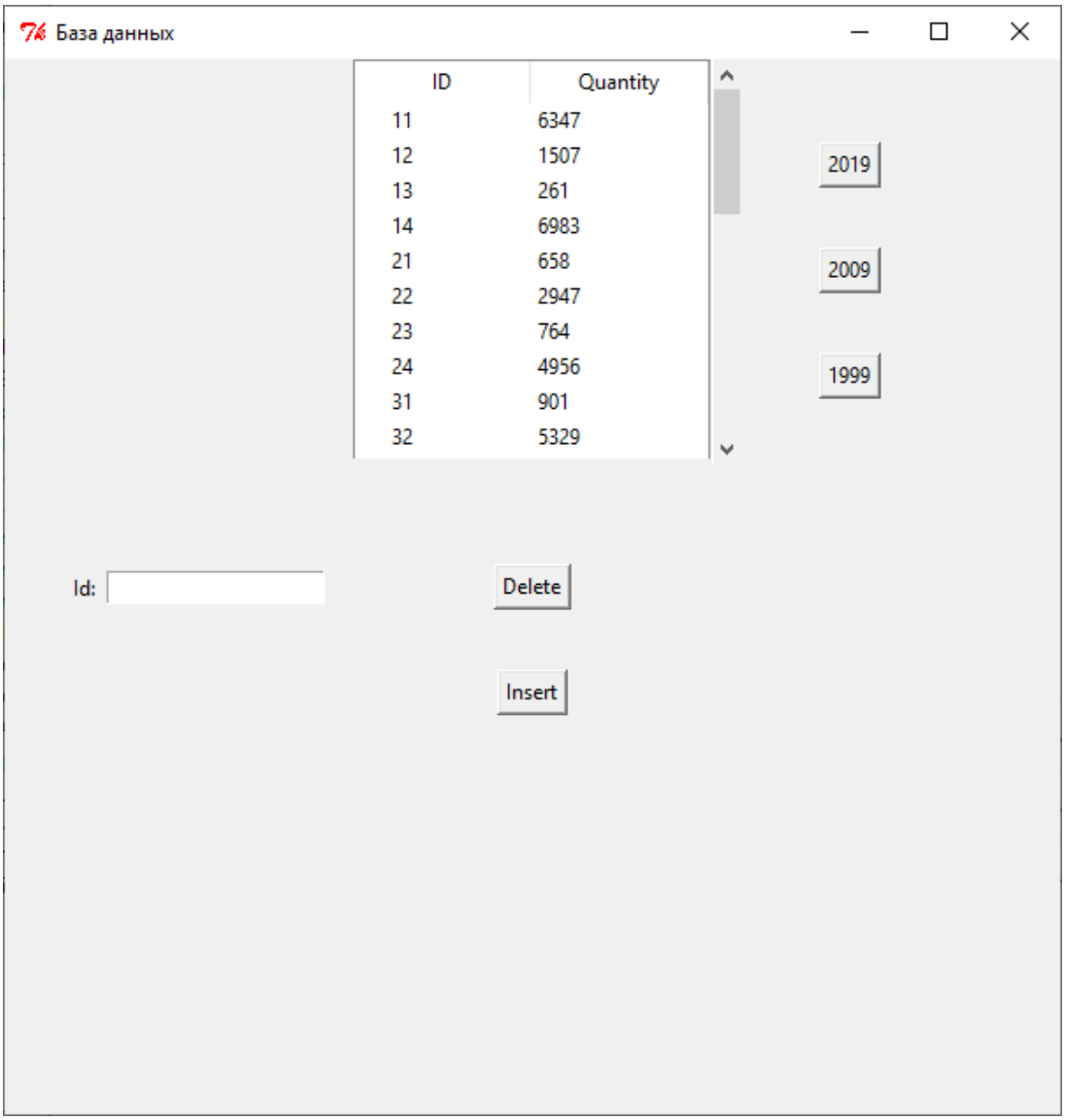


Рисунок 5.4 — Інтерфейс користувача для доступу до бази даних

На рисунку 5.4 зображено, що отримає користувач натиснувши кнопку «База даних». З’являється додаткове вікно, в якому виводиться інформація з бази даних. Якщо ввести в поле для вводу Id запису та натиснути кнопку «Delete», то запис видалиться з бази даних. Натиснувши кнопку «Insert» з’явиться ще одне додаткове вікно для внесення даних до бази даних, що зображено на рисунку 5.5.

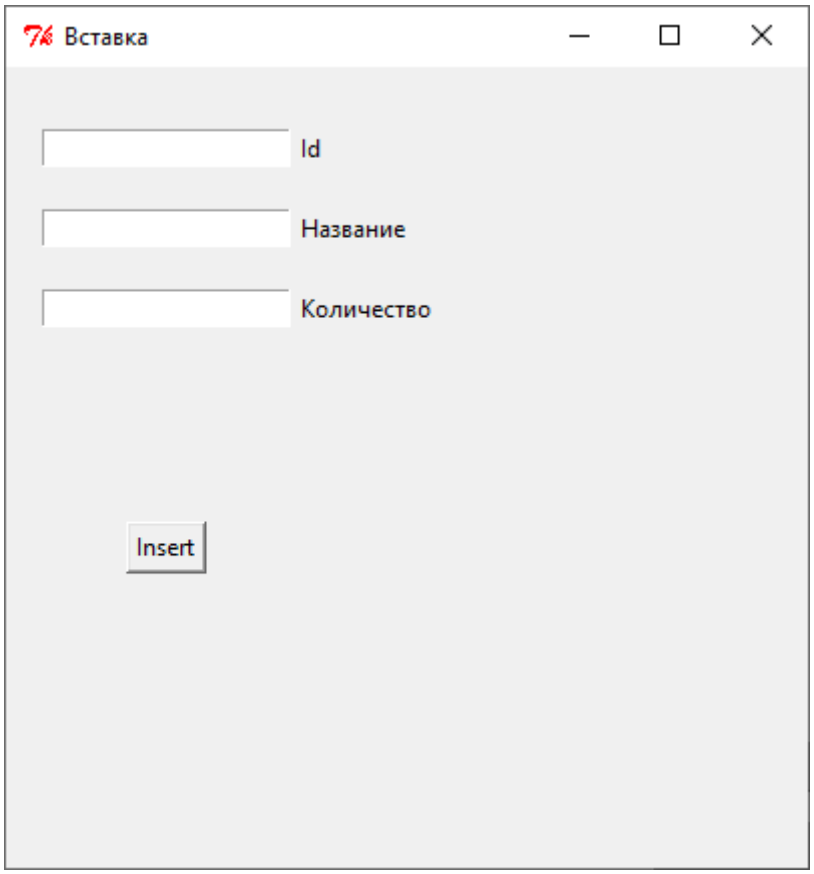


Рисунок 5.5 — Додаткове вікно для внесення нових записів до бази даних

На рисунку 5.5 ми бачимо поля для вводу та кнопку «Insert». Записавши дані в поля вводу та натиснувши кнопку, запис додається до існуючих записів у базі даних.

5.3 Висновки до розділу

У цьому розділі відображено графічний інтерфейс користувача, описано весь функціонал та пояснення для кожної кнопки. Інтерфейс представляє собою набір вікон, які дозволяють не плутатись та легко знаходити потрібну функцію системи.

ВИСНОВКИ

Під час виконання даної роботи, відповідно до вимог, було проаналізовано предметну область. Ретельно вибрано технології для створення програмного продукту. У результаті роботи, отримали програмне забезпечення, яке дозволяє проаналізувати результати класифікації космічних знімків. Також завдяки методу Баєса можемо перевірити наскільки отримані значення точні. Надалі систему можна покращити завдяки додаванню нового функціоналу, такого як: впровадження штучного інтелекту, який би сам робив аналіз та оцінку класифікації; самостійна розробка алгоритму класифікації зображень без додаткових програмних забезпечень.

Програмний продукт дає наглядно проаналізувати стан лісництва на території України протягом не одного десятиліття. Також потрібно розуміти, що в результатах можлива своя похибка, адже ми не маємо можливості використати зображення 100% якості, особливо, коли такі космічні знімки наразі платні. Також класифікація не дає 100% точності. Тому результати є приближені, але цього достатньо аби зробити попередню оцінку. Якщо покращити алгоритм класифікації та працювати зі знімками вищої якості, то програма буде давати більш точніші результати. Після цього збільшиться попит на ринку на даний продукт.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Лесной кодекс Украины [Електронний ресурс] – Режим доступу до ресурсу: http://pravoved.in.ua/section-kodeks/148-lku/1261-glava-001.html.
2. Моніторинг навколишнього середовища [Електронний ресурс] – Режим доступу до ресурсу: http://energetika.in.ua/ua/books/book-5/part-3/section-4/4-5
3. Satellite Observations Aid Bison Management [Електронний ресурс] – Режим доступу до ресурсу: https://earthobservatory.nasa.gov/images/146389/satellite-observations-aid-bison-management
4. The Teledesic Network: Using Low-Earth-Orbit Satellites to Provide Broadband, Wireless, Real-Time Internet Access Worldwide [Електронний ресурс] – Режим доступу до ресурсу: https://web.archive.org/web/20160306014740/http://www.isoc.org/inet96/proceedings/g1/g1_3.htm
5. Константиновская Л. В. Дистанционные методы контроля / Людмила Васильевна Константиновская., 2000. – 216 с.
6. Космические снимки (данные ДЗЗ) [Електронний ресурс] – Режим доступу до ресурсу: http://www.geocentre-consulting.ru/products/index?section=78.
7. Обработка данных ДЗЗ - Этапы обработки данных [Електронний ресурс] – Режим доступу до ресурсу: http://mapexpert.com.ua/index_ru.php?id=26&table=Menu.
8. Современные методы интеллектуальной обработки данных ДЗЗ / Н. С.Абрамов, Д. А. Макаров, А. А. Талалаев, В. П. Фраленко. – №9. – С. 417–442.
9. Comparison of image data acquired with AVHRR, MODIS, ETM [Електронний ресурс] – Режим доступу до ресурсу: https://www.sciencedirect.com/science/article/pii/S0273117703905448
10. Обзор космических съемочных систем высокого разрешения [Електронний ресурс] – Режим доступу до ресурсу: http://vinek.narod.ru/satellites.html
11. GIS (Geographic information system): data acquisition and processing techniques [Електронний ресурс] – Режим доступу до ресурсу: https://www.forhom.com/en/trouver-la-meilleure-solution/formations-cles-en-main/gis-geographic-information-system-data-acquisition-and-processing-techniques
12. Документация для ArcGIS [Електронний ресурс] – Режим доступу до ресурсу: https://doc.arcgis.com/ru/
13. A quick guide to ArcGIS — how do we begin? [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/kontinentalist/a-quick-guide-to-arcgis-how-do-we-begin-ded5efd41240

14. Обработка данных дистанционного зондирования Земли в ГИС пакете [Електронний ресурс] – Режим доступу до ресурсу: https://cyberleninka.ru/article/n/obrabotka-dannyh-distantsionnogo-zondirovaniya-zemli-v-gis-pakete-arcgis/viewer
15. KS IMAGINE Tutorials [Електронний ресурс] – Режим доступу до ресурсу: https://community.hexagongeospatial.com/t5/IMAGINE-Tutorials/tkb-p/KS_ERDAS_IMAGINE_Tutorials
16. Leica Geosystems GIS & Mapping Releases Leica Photogrammetry [Електронний ресурс] – Режим доступу до ресурсу: https://www.pobonline.com/articles/88659-leica-geosystems-gis-mapping-releases-leica-photogrammetry-suite-12-17-03
17. Андрианов В. Ю. Новые технологии дистанционного зондирования и работы с ДДЗ [Електронний ресурс] / В. Ю. Андрианов. – 2005. – Режим доступу до ресурсу: https://www.esri-cis.ru/news/arcreview/detail.php?ID=1740&SECTION_ID=47.
18. Building cross-platform, easy to maintain software architectures with Python, JSON [Електронний ресурс] – Режим доступу до ресурсу: https://codeburst.io/building-cross-platform-easy-to-maintain-software-architectures-with-python-json-and-javascript-fd1c68af9613
19. ArcGIS API for Python [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/Esri/arcgis-python-api
20. Python MySQL [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/python/python_mysql_getstarted.asp
21. The Python Tutorial [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/2/tutorial/index.html.
22. Connecting to MySQL Using Connector/Python [Електронний ресурс] – Режим доступу до ресурсу: https://dev.mysql.com/doc/connector-python/en/connector-python-example-connecting.html
23. PostgreSQL vs. MySQL [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresqltutorial.com/postgresql-vs-mysql/
24. Лекція 2 - Вивід даних на PHP та MySQL [Електронний ресурс] – Режим доступу до ресурсу: http://fcit.tneu.edu.ua/navchannja/pidhotovka-do-pratsevlashtuvannia/66-web-rozrobka/php-oop/556-vyvid-danykh-na-php-ta-mysql.
25. PostgreSQL and MySQL: Differences In Performance, Syntax, And Features [Електронний ресурс] – Режим доступу до ресурсу: https://blog.panoply.io/postgresql-vs.-mysql
26. Install mysql odbc drivers [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@joelzhang/install-mysql-odbc-drivers-in-ubuntu-18-04-39241072326a
27. Running the InnoDB recovery process [Електронний ресурс] – Режим доступу до ресурсу: https://www.a2hosting.com/kb/developer-corner/mysql/repairing-mysql-databases-and-tables#Running-the-InnoDB-recovery-process
28. Вестра Э. Разработка геоприложений на языке Python / Эрик Вестра. –

Москва: ДМК Пресс, 2017. – 446 с. – (3).
29. MySQL User Defined Function [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/24404/
30. Вестра Э. Разработка геоприложений на языке Python / Эрик Вестра. – Москва: ДМК Пресс, 2017. – 446 с. – (3).
31. Савченко С. Ю. Технологія розробки системи комплексного забезпечення життєдіяльності району міста [Електронний ресурс] / С. Ю. Савченко – Режим доступу до ресурсу: https://er.nau.edu.ua/bitstream/NAU/41935/1
32. Введення в tkinter. Урок 1 [Електронний ресурс] – Режим доступу до ресурсу: http://ni.biz.ua/8/8_8/8_83795_vvedenie-v-tkinter-urok-.html.
33. Pillow обработка изображений в Python на примерах [Електронний ресурс] – Режим доступу до ресурсу: https://python-scripts.com/pillow.
34. How to use Pillow (PIL: Python Imaging Library) [Електронний ресурс] – Режим доступу до ресурсу: https://note.nkmk.me/en/python-pillow-basic/.
35. 50 оттенков matplotlib — The Master Plots [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/468295/.
36. NumPy, часть 1: начало работы [Електронний ресурс] – Режим доступу до ресурсу: https://pythonworld.ru/numpy/1.html.
37. NumPy в Python. Часть 1 [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/352678/.
38. Вестра Э. Разработка геоприложений на языке Python / Эрик Вестра. – Москва: ДМК Пресс, 2017. – 446 с. – (3).

ДОДАТОК 1

Система аналізу часових змін лісових насаджень методом ДЗЗ

СПЕЦИФІКАЦІЯ

КПІ ім. Ігоря Сікорського ТМ62193_20Б

Аркушів 2

Київ 2020

Позначення	Найменування	Примітки
Документація		
УКР.КПім.ІгоряСікорського_ТЕФ_АПЕПС_ТМ62_20_4-1	Записка.docx	Текстова частина дипломної роботи
	Діаграми.vsdx	UML діграми
Компоненти		
УКР.КПім.ІгоряСікорського_ТЕФ_АПЕПС_ТМ62_20_4-1	gui.py	Модуль основного тіла програми
УКР.КПім.ІгоряСікорського_ТЕФ_АПЕПС_ТМ62_20_4-1	gui.py	Виконання алгоритму

ДОДАТОК 2

Система аналізу часових змін лісових насаджень методом ДЗЗ

ТЕКСТ ПРОГРАМНОГО МОДУЛЮ

КПІ ім. Ігоря Сікорського ТМ62193_20Б 12-2

Аркушів 12

Київ 2020

Тест програми

```
from tkinter import *
import tkinter.ttk as ttk
from tkinter import messagebox as mb
from PIL import ImageTk, Image
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import MySQLdb
import numpy

root = Tk()
root.geometry('1100x1100')
root.title('Diplom')

canvas = Canvas(root,width=1000,height=1000)
canvas.pack()

conn=MySQLdb.connect('localhost', 'root', '1234', 'Diplom')
mycursor=conn.cursor()

mycursor.execute("SELECT Quantity FROM irpin2019")
temp2019 = mycursor.fetchall()
res2019 = []

mycursor.execute("SELECT Quantity FROM irpin2009")
temp2009 = mycursor.fetchall()
res2009 = []

mycursor.execute("SELECT Quantity FROM irpin1999")
temp1999 = mycursor.fetchall()
res1999 = []

for element in temp2019:
```

```
    res2019.append(element[0])

for element in temp2009:
    res2009.append(element[0])

for element in temp1999:
    res1999.append(element[0])

theSum = 0
for i in res2019:
    theSum = theSum + i

res2019ver = []

for elem in res2019:
    temp = elem / float(theSum)
    res2019ver.append(temp)

print res2019ver

ver2019 = []

for a in res2019ver:
    tmp = (0.3 * a) / (0.3 * a + 0.4 * (1 - a))
    ver2019.append(tmp)

print ver2019

data_names = ['11', '12', '13', '14', '21', '22', '23', '24',
              '31', '32', '33', '34', '61', '62', '63', '71',
              '72', '73', '84', '91', '92', '93', '94',
              '101', '102', '103', '104', '114', '121', '122', '123',
              '124', '152']

dpi = 80
```

```
fig = plt.figure(dpi = dpi, figsize = (1200 / dpi, 384 / dpi) )
mpl.rcParams.update({'font.size': 10})

plt.title('Diagram')

ax = plt.axes()
ax.yaxis.grid(True, zorder = 1)

xs = range(len(data_names))

plt.bar([x + 0.05 for x in xs], res1999,
    width = 0.4, color = 'red', alpha = 1, label = '1999',
    zorder = 2)
plt.bar([x + 0.3 for x in xs], res2009,
    width = 0.4, color = 'blue', alpha = 1, label = '2009',
    zorder = 2)
plt.bar([x + 0.6 for x in xs], res2019,
    width = 0.4, color = 'green', alpha = 1, label = '2019',
    zorder = 2)
plt.xticks(xs, data_names)

fig.autofmt_xdate(rotation = 25)

plt.legend(loc='upper right')
fig.savefig('bars.png')
mycursor.close()
conn.close()

def load_image(name):
    img = Image.open(name)
    img.thumbnail((400, 400), Image.ANTIALIAS)
    return ImageTk.PhotoImage(img)
def load_image1(name):
    img = Image.open(name)
    img.thumbnail((1000, 1000), Image.ANTIALIAS)
```

```
    return ImageTk.PhotoImage(img)

def set_image(image):
    canvas.delete("all")
    canvas.create_image(500,215,image=image)

image = load_image("irpin1999.jpg")
image1 = load_image("irpin2009.jpg")
image2 = load_image("irpin2019.jpg")
bar = load_image1("bars.png")

def cat1():
    set_image(image)

def cat2():
    set_image(image1)

def cat3():
    set_image(image2)

def zoom1999():
    path='C:\Users\user\Desktop\Python\diplom\irpin1999.jpg'
    im=Image.open(path)
    im.show()

def zoom2009():
    path='C:\Users\user\Desktop\Python\diplom\irpin2009.jpg'
    im=Image.open(path)
    im.show()

def zoom2019():
    path='C:\Users\user\Desktop\Python\diplom\irpin2019.jpg'
    im=Image.open(path)
    im.show()
```

```
def Diagr():
    set_image(bar)


def Sprav():
    window = Toplevel(root)
    window.geometry('420x400')
    window.title('Справочник')
    t8 = Label(window)
    t8.place(x=20, y=10)
    t8['text'] = "11 - Сосна молодняки"
    t9 = Label(window)
    t9.place(x=20, y=30)
    t9['text'] = "12 - Сосна середньовікові"
    t10 = Label(window)
    t10.place(x=20, y=50)
    t10['text'] = "13 - Сосна прстигаючі"
    t11 = Label(window)
    t11.place(x=20, y=70)
    t11['text'] = "14 - Сосна стиглі та перестиглі"
    t12 = Label(window)
    t12.place(x=20, y=90)
    t12['text'] = "21 - Ялина молодняки"
    t13 = Label(window)
    t13.place(x=20, y=110)
    t13['text'] = "22 - Ялина середньовікові"
    t14 = Label(window)
    t14.place(x=20, y=130)
    t14['text'] = "23 - Ялина прстигаючі"
    t15 = Label(window)
    t15.place(x=20, y=150)
    t15['text'] = "24 - Ялина стиглі та перестиглі"
    t16 = Label(window)
    t16.place(x=20, y=170)
    t16['text'] = "31 - Дуб молодняки"
    t17 = Label(window)
```

```
t17.place(x=20, y=190)
t17[‘text’] = "32 - Дуб середньовікові"
t18 = Label(window)
t18.place(x=20, y=210)
t18[‘text’] = "33 - Дуб прстигаючі"
t19 = Label(window)
t19.place(x=20, y=230)
t19[‘text’] = "34 - Дуб стиглі та перестиглі"
t20 = Label(window)
t20.place(x=20, y=250)
t20[‘text’] = "61 - Граб молодняки"
t21 = Label(window)
t21.place(x=20, y=270)
t21[‘text’] = "62 - Граб середньовікові"
t22 = Label(window)
t22.place(x=20, y=290)
t22[‘text’] = "63 - Граб прстигаючі"
t23 = Label(window)
t23.place(x=20, y=310)
t23[‘text’] = "71 - Ясен молодняки"
t24 = Label(window)
t24.place(x=20, y=330)
t24[‘text’] = "72 - Ясен середньовікові"
t25 = Label(window)
t25.place(x=20, y=350)
t25[‘text’] = "73 - Ясен прстигаючі"
t26 = Label(window)
t26.place(x=20, y=370)
t26[‘text’] = "84 - Акація стиглі та пересиглі"
t27 = Label(window)
t27.place(x=220, y=10)
t27[‘text’] = "91 - Береза молодняки"
t28 = Label(window)
t28.place(x=220, y=30)
t28[‘text’] = "92 - Береза середньовікові"
```

```
t29 = Label(window)
t29.place(x=220, y=50)
t29[‘text’] = "93 - Береза прстигаючі"
t30 = Label(window)
t30.place(x=220, y=70)
t30[‘text’] = "94 - Береза стиглі та перестиглі"
t31 = Label(window)
t31.place(x=220, y=90)
t31[‘text’] = "101 - Осика молодняки"
t32 = Label(window)
t32.place(x=220, y=110)
t32[‘text’] = "102 - Осика середньовікові"
t33 = Label(window)
t33.place(x=220, y=130)
t33[‘text’] = "103 - Осика прстигаючі"
t34 = Label(window)
t34.place(x=220, y=150)
t34[‘text’] = "104 - Осика стиглі та перестиглі"
t35 = Label(window)
t35.place(x=220, y=170)
t35[‘text’] = "114 - Тополя стиглі та перестиглі"
t36 = Label(window)
t36.place(x=220, y=190)
t36[‘text’] = "121 - Вільха молодняки"
t37 = Label(window)
t37.place(x=220, y=210)
t37[‘text’] = "122 - Вільха середньовікові"
t38 = Label(window)
t38.place(x=220, y=230)
t38[‘text’] = "123 - Вільха прстигаючі"
t39 = Label(window)
t39.place(x=220, y=250)
t39[‘text’] = "124 - Вільха стиглі та перестиглі"
t40 = Label(window)
t40.place(x=220, y=270)
```

```
    t40[‘text’] = "152 - Інші"


def show_message():
    con=MySQLdb.connect(‘localhost’, ‘root’, ‘1234’, ‘Diplom’)
    mycur=con.cursor()
    messageNum = message.get()
    print messageNum
    mycur.execute("DELETE FROM irpin2019 WHERE ID=%s", (messageNum,))
    con.commit()
    mycur.close()
    con.close()


def insert_db():
    conStr=MySQLdb.connect(‘localhost’, ‘root’, ‘1234’, ‘Diplom’)
    mycurIn=conStr.cursor()
    idTemp = IdInsert.get()
    nameTemp = nameInsert.get()
    quantityTemp = quantityInsert.get()
    print idTemp
    print nameTemp
    print quantityTemp
    mycurIn.execute("INSERT INTO irpin2019 (ID, Class_name, Quantity) VALUES (%s, %s, %s)", (idTemp, nameTemp,
quantityTemp,))
    conStr.commit()
    mycurIn.close()
    conStr.close()


def show_insert():
    windowInsert = Toplevel(root)
    windowInsert.geometry(‘400x400’)
    windowInsert.title(‘Вставка’)
    IdInsert_entry = Entry(windowInsert, textvariable=IdInsert)
    IdInsert_entry.place(relx=.2, rely=.1, anchor="c")
    label2 = Label(windowInsert, text="Id")
    label2.place(relx=.36, rely=.075)
```

```
    nameInsert_entry = Entry(windowInsert, textvariable=nameInsert)
    nameInsert_entry.place(relx=.2, rely=.2, anchor="c")
    label3 = Label(windowInsert, text="Название")
    label3.place(relx=.36, rely=.175)
    quantityInsert_entry = Entry(windowInsert, textvariable=quantityInsert)
    quantityInsert_entry.place(relx=.2, rely=.3, anchor="c")
    label4 = Label(windowInsert, text="Количество")
    label4.place(relx=.36, rely=.275)
    insert_but = Button(windowInsert, text="Insert", command=insert_db)
    insert_but.place(relx=.2, rely=.6, anchor="c")
    dbShow()

def dbShow():
    windowDB = Toplevel(root)
    windowDB.geometry('600x600')
    windowDB.title('База данных')
    tree = ttk.Treeview(windowDB, selectmode='browse')

    scrolltable = ttk.Scrollbar(windowDB, orient="vertical", command=tree.yview)
    scrolltable.place(x=200+200+2, y=0, height=210+20)
    tree.configure(yscrollcommand=scrolltable.set)

    tree["columns"]=("one")
    tree.column("one", width=100 )
    tree.heading("one", text="Quantity")
    tree.column("#0", width=100)
    tree.heading("#0",text="ID")

    conn1=MySQLdb.connect('localhost', 'root', '1234', 'Diplom')
    mycursor1=conn1.cursor()

    mycursor1.execute("SELECT Quantity FROM irpin2019")
    tempq2019 = mycursor1.fetchall()
    quantity2019 = []
```

```
for element in tempq2019:
    quantity2019.append(element[0])

print quantity2019

mycursor1.execute("SELECT ID FROM irpin2019")
tempId2019 = mycursor1.fetchall()
Id2019 = []

for element in tempId2019:
    Id2019.append(element[0])

print Id2019

i = 0

for element in quantity2019:
    tree.insert("" , i,    text=Id2019[i], values=(element))
    i = i + 1

mycursor1.close()
conn1.close()

message_entry = Entry(windowDB, textvariable=message)
message_entry.place(relx=.2, rely=.5, anchor="c")
label1 = Label(windowDB, text="Id:")
label1.place(relx=.06, rely=.484)

message_button1 = Button(windowDB, text="Delete", command=show_message)
message_button1.place(relx=.5, rely=.5, anchor="c")

message_button2 = Button(windowDB, text="Insert", command=show_insert)
message_button2.place(relx=.5, rely=.6, anchor="c")

tree.pack()
```

```
    windowDB.mainloop()

message = StringVar()
IdInsert = StringVar()
nameInsert = StringVar()
quantityInsert = StringVar()

btn1 = Button(root, text="1999", width=10, command=cat1)
btn1.place(x=400, y=440)

btn2 = Button(root, text="2009", width=10, command=cat2)
btn2.place(x=500, y=440)

btn3 = Button(root, text="2019", width=10, command=cat3)
btn3.place(x=600, y=440)

btn4 = Button(root, text="Zoom", width=10, command=zoom1999)
btn4.place(x=400, y=480)

btn5 = Button(root, text="Zoom", width=10, command=zoom2009)
btn5.place(x=500, y=480)

btn6 = Button(root, text="Zoom", width=10, command=zoom2019)
btn6.place(x=600, y=480)

btn7 = Button(root, text="Диаграмма", width=10, command=Diagr)
btn7.place(x=500, y=520)

btn8 = Button(root, text="Справочник", width=10, command=Sprav)
btn8.place(x=400, y=520)

btn9 = Button(root, text="База данных", width=10, command=dbShow)
btn9.place(x=600, y=520)

        root.mainloop()
```

ДОДАТОК З

Реалізація програмного рішення інформаційної системи аналізу часових змін лісових насаджень методом ДЗЗ

ОПИС ПРОГРАМНОГО МОДУЛЮ

КПІ ім. Ігоря Сікорського ТМ62193_20Б 13-2

Аркушів 8

Київ 2020

АНОТАЦІЯ

Додаток містить опис інформаційної системи аналізу часових змін лісових насаджень методом ДЗЗ.

В додатку виконуються такі функції:

— перегляд обробленої інформації із класифікації;

— перегляд знімків обраної зони лісових насаджень за 1999, 2009 та 2019 роки;

— Перегляд записів із бази даних**;**

— видалення та додавання записів із/до бази даних**.**

Вхідні та вихідні дані отримуються засобами реалізованими Python та модулем MySQL.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ

У цьому додатку описано інформаційну систему аналізу часових змін лісових насаджень методом ДЗЗ.

Модулі системи описані в ДОДАТКУ 2.

Додаток працює в операційних системах, таких як Windows7, Windows8, Windows10.

Компоненти необхідні для установки: Python, Spyder, Anaconda, MySQL.

Використана мова програмування — Python.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблена інформаційна система призначена для аналізу часових змін лісових насаджень методом ДЗЗ.

Дана система може бути використана структурами, які займаються моніторингом навколишнього середовища і які бажають зменшити час аналізу змін лісових насаджень.

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Інформаційна система являє собою програму зі зручним інтерфейсом, який дозволить швидко аналізувати дані класифікації.

Потрібно встановити всі додаткові програмні додатки та запустити програму. Після цього ви можете працюати з програмою, змінювати та додавати інформацію, робити аналіз даних та зберігати зміни.

ВИКОРИСТОВУВАНІ ТЕХІЧНІ ЗАСОБИ

Компоненти необхідні для установки інформаційної системи: Python, Spyder, MySQL, Anaconda.

Використана мова програмування для інформаційної системи — Python.

Розроблена інформаційна система працює в операційних системах Windows7, Windows8, Windows10 та потребує встановлення компонентів: Python, Spyder, MySQL, Anaconda.

ВИКЛИК І ЗАВАНТАЖЕННЯ

Для запуску необхідно викликати cmd та написати команду «python path\gui.py», де path – шлях до папки, де знаходиться папка з програмою. Для встановлення бази даних потрібно встановити MySQL, завантажити скрипт та виконати його. Після цього програма готова до використання.

ДОДАТОК 4

Розробка прикладного програмного забезпечення для аналізу часових змін методом Байеса

ТЕКСТ ПУБЛІКАЦІЇ

КПІ ім. Ігоря Сікорського ТМ62193_20Б

Аркушів 2

Київ 2020

УДК 004.9

Студент 4 курсу, гр. ТМ-62 Богач А.Г.
Ст.викл. Бандурка О.І.

РОЗРОБКА ПРИКЛАДНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АНАЛІЗУ ЧАСОВИХ ЗМІН ЛІСОВИХ НАСАДЖЕНЬ МЕТОДОМ БАЙЕСА

Питання масової вирубки лісових насаджень на території України стало нагальною проблемою. Масове винищення природних лісових ресурсів наносить невиправну шкоду екології та економічним чинникам країни.

Вирішення даної проблеми потрібує прийняття невідкладних заходів. Тому метою є створення прикладного програмного забезпечення з аналізом, оцінкою та прогнозуванням ситуації певних лісових зон. На основі методу Байеса будуть оцінюватись критичні зміни лісових насаджень в часовому вимірі. Даний програмний продукт дасть змогу економити час на аналіз космічних знімків, лише завантажуючи їх у програму та отримувати обґрунтований результат.

$$P\left(\frac{S_i}{k_j}\right) = P(S_i)P\left(\frac{k_j}{S_i}\right) = P(k_j)P\left(\frac{S_i}{k_j}\right), \text{де}$$

$P(S_i)$ – ймовірність появи стану S_i,

$P(k_j)$ – ймовірність появи признаку k_j,

$P(k_j/S_i)$ – ймовірність появи признаку k_j у об’єктів зі станом S_i.

Для вирішення цієї задачі пропонується використати інструменти програмного продукту ArcGIS, що надає можливості для створення карт, оброки просторових запитів, базового моделювання та аналізу даних та підключенння додаткових модулів. Для створення ймовірного методу, який при наявних ознаках відносить до одного з можливих варіантів, завдяки формулі Байеса створено інтерпретованою об`єктно-орієнтованою мовою Python з використанням ArcPy.

Важливим аспектом використання програмного забезпечення організації зможуть зменшити витрати часу на проведення оцінювання збитків через вирубку лісових насаджень, автоматизувати оцінку становища лісових угідь з плином часу. Таким чином програмний продукт є зручним засобом для вирішення проблеми, яка постає перед клієнтом: обробка, класифікація та аналіз природних та екологічних зон країни.

Перелік посилань:

1. Вестра Э. Разработка геоприложений на языке Python / Эрик Вестра., 2016. – 446 с.
2. What is ArcPy? [Електронний ресурс]. – 2010. – Режим доступу до ресурсу: https://www.esri.com/arcgis-blog/products/arcgis-desktop/analytics/what-is-arcpy/.

ДОДАТОК 5

Класифікація знімків для аналізу часових змін лісових насаджень

ТЕКСТ ПУБЛІКАЦІЇ

КПІ ім. Ігоря Сікорського ТМ62193_20Б

Аркушів 2

Київ 2020

УДК 004.9

Студент 4 курсу, гр. ТМ-62 Богач А.Г.; студент 4 курсу, гр. ТМ-62 Баб`як В.В.
Асист. Швайко В.Г.

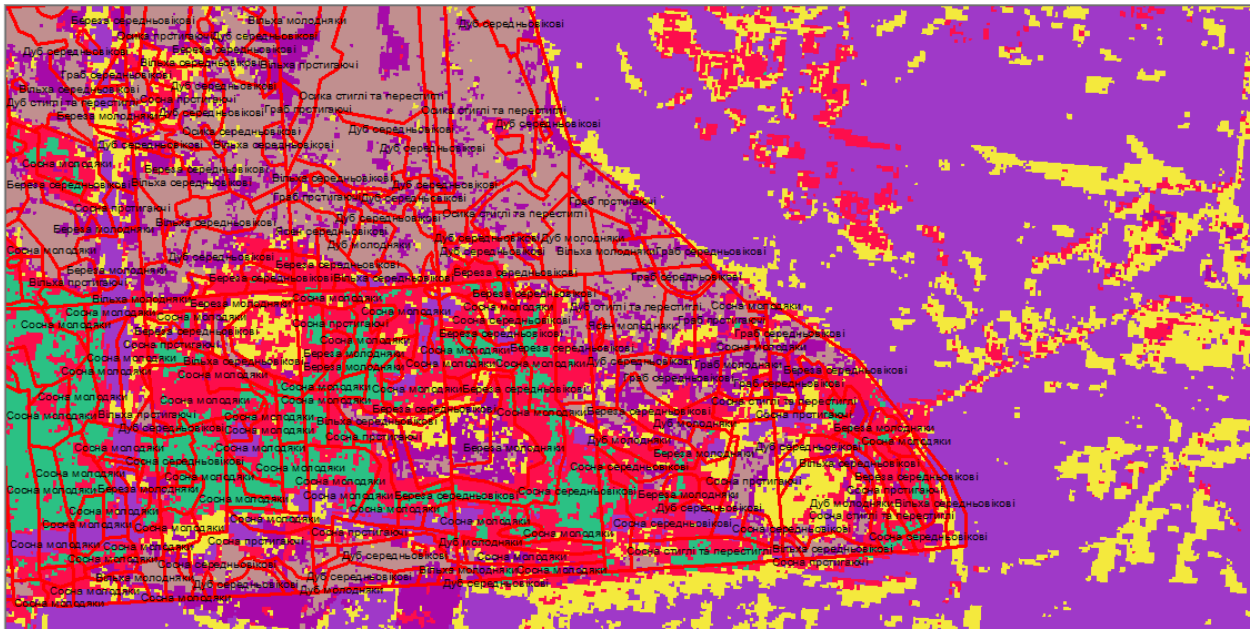
КЛАСИФІКАЦІЯ ЗНІМКІВ ДЛЯ АНАЛІЗУ ЧАСОВИХ ЗМІН ЛІСОВИХ НАСАДЖЕНЬ

Зважаючи на негативні зміни екологічних зон через вирубку лісових насаджень необхідно провести їх аналіз на протязі певного часу. Саме тому нам потрібна класифікація космічних знімків.

Вирішення даної проблеми надасть нам змогу контролювати ці зміни та вчасно зупинитись, щоб не нанести шкоди екологічним та економічним чинникам України. Процес класифікації – це процес отримання класів з багатоканального растрового зображення. Саме тому ми використовуємо космічні знімки визначеної місцевості зі зміною у десять років. Процес класифікації – це багатокроковий робочий процес, який потребує часу.

Для вирішення цієї задачі ми використовуємо інструменти програмного продукту ArcGIS. Створюємо базу даних кварталів та полігонів, завантажуємо файл з описами порід дерев на обраній ділянці. Виділяємо класи для кожної породи та позначаємо її власним Id кодом. Після того, як ми оброили карту та виділили кожен клас, проводимо автоматичну класифікацію зображення. В результаті отримуємо карту виділів кожної породи. Для кожного класу відповідає індивідуальний колір, що спрощує процес аналізу класифікації.

Ця робота дозволяє нам швидко класифікувати зображення та проаналізувати зміни лісових угідь.



Богач А.Г. _____ Баб`як В.В. _____ Швайко В.Г. _____
(Підпис) (Підпис) (Підпис)