Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

Навчально-науковий інститут деревообробних та комп'ютерних технологій і дизайну
(повне найменування інституту, назва факультету (відділення))

Кафедра інформаційних технологій
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до дипломної роботи
другий (магістерський)
(рівень вищої освіти)

на тему: **Інформаційна система моделювання процесу росту та прогнозування врожайності сільськогосподарських культур**

Виконала: студентка VI курсу, групи КН-61м
спеціальності
122 – "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)
Поліщук С. І.
(прізвище та ініціали)

Керівник Пірко І. Б.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Львів – 2022 р.

Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)

ННІ деревообробних та комп'ютерних технологій і дизайну
Кафедра інформаційних технологій
Рівень вищої освіти другий (магістерський)
Спеціальність 122 "Комп'ютерні науки"
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____Крошний І. М.
"____" _____2022 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТЦІ

Поліщук Софії Ігорівні
(прізвище, ім'я, по батькові)

1. Тема роботи **Інформаційна система моделювання процесу росту та прогнозування врожайності сільськогосподарських культур**

керівник роботи Пірко І. Б., канд. фіз.-мат. наук, доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від 13.12. 2021 року № С-617
2. Термін подання студентом роботи 12. 12. 2022 р.
3. Вихідні дані до роботи:

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
4.1. Стан проблемної області.
4.2. Інформаційне забезпечення.
4.3. Математичне забезпечення.
4.4. Програмне забезпечення.
4.5. Розроблення стартап проекту.
4.6. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
презентація у Microsoft PowerPoint.

6. Дата видачі завдання 20 грудня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Огляд літературних даних та інших джерел згідно досліджуваної теми	20.12-30.01.2022	
2	Аналіз досліджуваної теми та вибір відповідних варіантів її розробки	01.02-27.02. 2022	
3	Постановка задачі та її формалізація	01.03-01.04. 2022	
4	Вибір та обґрунтування методів і засобів проведення дослідження	02.04-30.04. 2022	
5	Розроблення концептуальної схеми реалізації завдання	01.05-01.06. 2022	
6	Програмна реалізація завдання	02.06-30.10. 2022	
7	Тестування програмного продукту та отриманих результатів	01.11-15.11. 2022	
8	Розробка пояснювальної записки магістерської роботи	16.11-30.11. 2022	
9	Корегування пояснювальної записки згідно вимог, розроблення презентації	01.12-11.12. 2022	

Студентка _____ Поліщук С. І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Пірко І. Б.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Пояснююча записка складається з 90 сторінок, 24 рисунків, 2 додатків та 4 таблиць, 20 літературних джерел.

В дипломній роботі розглянуто основні фактори, що впливають на формування врожаю, кореляційний зв’язок між факторними ознаками (сонячна активність, опади, добрива) та врожайністю культур, параметри відбору факторних ознак для побудови регресійних моделей урожайності.

Засобами Python та його бібліотек для аналізу даних Scikit-learn, Pandas та візуалізації результатів досліджень Matplotlib отримано інформаційно-аналітичну систему, з допомогою якої можна моделювати процеси росту рослин та прогнозувати майбутню врожайність цих культур. Отримано показники для оцінки адекватності екстраполяційних моделей урожайності сільськогосподарських культур та оцінки на їх основі точності прогнозів.

Ключові слова: урожайність, прогнозування, Python, Scikit-learn, Pandas, Matplotlib.

SUMMARY

The explanatory note consists of 90 pages, 24 figures, 2 attachments and 4 tables.

The main factors that are added to the formation of crops, the correlation between factor signs (drowsy activity, fall, kindness) and crop yields, parameters for the selection of factor signs for inducing regression models of productivity are considered.

With the help of Python and libraries for the analysis of Pandas data and visualization of the results obtained by Matplotlib, an information-analytical system was created, with the help of which it is possible to model the growth process of growing and predict the future crop yield. Indications were taken to assess the adequacy of extrapolation models for the analysis of time series in the yield of agricultural crops and assessments based on the accuracy of forecasts.

Keywords: yield, forecasting, Python, Scikit-learn, Pandas, Matplotlib.

ПЕРЕЛІК СКОРОЧЕНЬ І СПЕЦІАЛЬНИХ ТЕРМІНІВ

- АКФ – автокореляційна функція;
- БД – база даних;
- ВІ – вегетаційний індекс;
- ДМУ – дійсно можливий урожай по біокліматичних показниках та умовах вологозабезпечення;
- ІУ – індекс урожайності (ІУ);
- ООП – об’єктно-орієнтоване програмування;
- ПУ – потенціальний урожай, який отримують по приходу фотосинтетичної активної радіації;
- УВ – урожай у виробництві: рівень урожайності, який отримують в виробництві;
- ФАР – фотосинтетично активна радіація;
- GUI – Graphical User Interface;
- FAO – Food and Agriculture Organization of the United Nations;
- LAI – Leaf Area Index;
- ML – Machine Learning;
- NDVI – Normalized Difference Vegetation Index;
- SAFY (Simple algorithm for yield estimates) – модель розвитку рослинних культур.

ЗМІСТ

ВСТУП

Актуальність дипломної роботи

Серед багатьох показників особливої уваги заслуговує врожайність сільськогосподарських культур. Це комплексний показник, з одного боку він є вихідною інформацією для побудови планів та прогнозів, з іншого, це один з основних результуючих показників сільськогосподарського виробництва. Урожайність сільськогосподарських культур є показником дуже складним з погляду прогнозу, оскільки формування врожаю пов’язане як з впливом виробничих факторів, так і погодніх умов та біологічних систем. Завчасний прогноз урожайності сільськогосподарських культур є основою для своєчасного та ефективного коригування структури сільськогосподарського виробництва, його розміщення та перерозподілу ресурсів.

Предметом дослідження є розробка інформаційної системи моделювання процесу росту та прогнозування врожайності сільськогосподарських культур.

Об’єктом дослідження є ріст рослин та урожайність сільськогосподарських культур.

Мета роботи – розробка та реалізація інформаційно-аналітичної системи моделювання процесу росту та прогнозування врожайності сільськогосподарських культур.

Відповідно до мети дослідження поставлено наступні **завдання**:

- провести огляд літератури по даній тематиці, проаналізувати існуючі системи росту та прогнозування урожайності сільськогосподарських культур;
- розробити математичну модель росту рослин;
- реалізувати програмну модель для прогнозування динаміки урожайності сільськогосподарських культур;
- в рамках програмної моделі провести дослідження динаміки росту рослин та прогнозування їх урожайності.

Наукова новизна одержаних результатів

Наукова новизна полягає у розробці алгоритму побудови прогнозу врожайності культур. Запропонований прийом побудови прогнозу дозволяє враховувати динаміку врожайності, так і її циклічні особливості. Виявлено динамічний характер кореляційних зв’язків між факторними ознаками та врожайністю культур, обґрунтовано параметри відбору факторних ознак для побудови регресійних моделей. Отримано показники для екстраполяційних моделей та оцінки точності прогнозу.

Практичне значення одержаних результатів

З метою забезпечення достовірності висновків та результатів досліджень було проаналізовано показники врожайності різних сільськогосподарських культур. По них було розраховано моделі екстраполяційних прогнозів. Отримані дані свідчать про достовірність отриманих результатів. Розроблено механізм побудови прогнозу врожайності культур. Цей алгоритм відрізняється простою реалізацією, є універсальним з погляду його використання на різних рівнях розробки прогнозів.

РОЗДІЛ 1. АНАЛІЗ СТАНУ ПРОБЛЕМНОЇ ОБЛАСТІ

1.1. Прогнозування та програмування урожайності сільськогосподарських культур

Програмування врожаю слід відрізняти від прогнозування та планування. Програмування передбачає лише вирішальні фактори, що впливають на формування врожайності: технологія обробки, зрошення, сорт, захист рослин від хвороб, шкідників, бур'янів. Розробка проблеми програмування у всій її складності можлива в майбутньому при накопиченні даних про продукційний процес формування врожаю як у кількісному, так і в якісному відношенні.

Планування врожаю як правило здійснюється від досягнутого рівня з використанням бажаних показників росту продуктивності рослинництва на найближчий період та розглядається як перший етап програмування. Він базується на середньостатистичних даних щодо врожайності в даному господарстві або на даному полі за багато років із перевищенням останнього на величину, що очікується від зміни рівня агротехніки.

Прогнозування представляє собою розрахунок теоретично можливого наростання врожаю, що забезпечується кліматичними, ґрунтовими та матеріально-технічними ресурсами. Воно дає можливість передбачити кінцевий результат з обробки культури у певних ґрунтово-кліматичних умовах.

Прогнозування врожаїв – це науково обґрунтоване передбачення продуктивності сільськогосподарських культур на ряд років або на перспективу. При використанні методу кореляційно-регресійного аналізу у прогнозуванні врожаїв користуються лінійною формою рівняння:

$$Y = a + bx, \tag{1.1}$$

де $У$ – середній урожай в році, ц/га; $а$ – вільний член рівняння, $в$ – коефіцієнт регресії, $х$ – фактор часу.

При програмуванні, крім наукового прогнозу величини та якості врожаю, заздалегідь намічається майбутній хід його формування, тобто ріст та розвиток рослин. У цьому цілеспрямовано здійснюється оптимізація основних факторів формування врожаю.

Програмування врожаю – це науково обґрунтоване прогнозування поетапного його формування, оптимізація основних факторів росту та розвитку та управління процесом формування врожаю на основі поточної інформації, що швидко обробляється на комп'ютерах за спеціальними програмами. При програмуванні врожайності сільськогосподарських культур необхідно мати відповідні математичні моделі, задані режими технології вирощування сільськогосподарських культур, володіння навичками користування комп'ютерною технікою для оперативного визначення необхідних агротехнічних прийомів для отримання програмованої врожайності.

Програмування врожаїв передбачає повну реалізацію потенційної продуктивності сорту за оптимізації основних факторів життєдіяльності рослин та раціональне використання ресурсів клімату та грунтів за умови лімітування продуктивності посівів яким-небудь фактором.

На даний час існують такі методи розрахунку врожайності:

- метод екстраполяції закономірностей, які склалися;
- біологічні методи;
- методи, які базуються на використанні узагальнених агрокліматичних ресурсів: вологозабезпечення посівів, біокліматичного потенціалу;
- математично-статистичний метод (регресійні моделі кількісних зв'язків урожаю з факторами, які його забезпечують);
- оптимізаційні моделі;
- імітаційні моделі;
- статистичні моделі.

Програмування врожайності починається з обґрунтування величини можливого врожаю, на який потрібно орієнтуватися. Урожай формується у процесі фотосинтезу. Рівень врожайності залежить від біологічних властивостей культури або сорту, рівня агротехніки та метеорологічних умов.

При програмуванні урожаю любої сільськогосподарської культури визначають три рівні врожайності:

- потенційний урожай (ПУ) – по приходу фотосинтетично активної радіації (ФАР);
- дійсно можливий урожай (ДМУ) – по біокліматичних показниках та умовах вологозабезпечення;
- урожай у виробництві (УВ) – рівень урожайності, який отримують у виробництві.

Потенційний урожай – це теоретично можливий максимальний урожай, який можна отримати в ідеальних метеорологічних умовах (достатньо води, тепла, світла). Він залежить від приходу ФАР та потенційної продуктивності культури або сорту.

Дійсно можливий урожай – це максимальний урожай, який може бути отриманий за реальних середньорокових кліматичних умовах. На основі узагальнення результатів багаторічних дослідів встановлено, що ДМУ становить 60-80 % від ПУ.

Урожай у виробництві значно нижчий за ДВУ. Це пояснюється тим, що ФАР та метеорологічні умови максимально не використовуються для формування врожаю. Причини цього – незадовільний прогноз погоди, недоліки в агротехніці та організації виробництва, наявність хвороб, шкідників та бур’янів у посівах сільськогосподарських культур.

Основне завдання програмування врожаїв – наближення УВ до ДМУ та ДМУ до УВ. Програмування врожаїв має бути спрямоване на здійснення наступних переходів: ПУ $\rightarrow$ ДМУ $\rightarrow$ УВ. Для цього необхідно провести заходи щодо покращення узгодженості потреб рослин з умовами зовнішнього середовища, при цьому слід мати на увазі і економічну ефективність заходів, що проводяться.

Якість програмування врожаю у виробництві слід оцінювати не за абсолютним значенням отриманого врожаю, а, по різниці між ДВУ і УВ. Ця різниця є величиною врожаю, що недоотримується через неповне використання потенційних можливостей підвищення врожаю. Ефективність програмування врожаю тим вища, що менша різниця між ДМУ та УВ, тобто менший недобір урожаю. В ідеальному випадку УВ повинен дорівнювати ДМУ.

1.2. Інформаційно-аналітичні системи для програмування урожайності

Агроекологічні моделі відображають вплив грунтових і погодних умов на продукційний процес рослин, на їх ріст, розвиток і формування кінцевого врожаю. Сьогодні науковий напрямок комп'ютерного моделювання агроекосистем вже остаточно сформувався. Як лідери світового ринку моделювання продукційного процесу рослин можна вказати голандську школу моделювання (сімейство моделей WOFOST) або американську школу (сімейство CERES).

Поява сучасних високошвидкісних комп'ютерів з великою пам'яттю дозволяють докорінно змінити ситуацію в галузі інформаційної підтримки прийнятих рішень. Для прийняття рішень можуть бути використані бази даних, а також бази знань, які основані на використанні імітаційних динамічних моделей агроекосистем.

Це дозволить видозмінити всю технологію прийняття рішень шляхом програвання сценаріїв майбутнього з урахуванням поточної та прогнозованої обстановки. До останньої, перш за все, відносяться метеорологічні умови, що багато в чому визначають продукційний процес рослин та його результат.

Прикладом такої технології є європейський проект MARS – Monitoring Agricultural Statistics with Remote Sensing, який розроблений в Об'єднаному дослідному центрі Європейської комісії. Його головною метою є кількісна оцінка площ, що займають різні культури в кожному регіоні (країні), моніторинг посівів, оперативний прогноз врожаїв, швидка та оперативна оцінка загальної продукції країн Європи по найважливіших культурах. В основу прогностичних розрахунків покладено європейську динамічну модель WOFOST. В Агрофізичному інституті протягом останніх двох десятиліть розвивалася теорія моделювання агроекосистем, розроблені та реалізовані динамічні моделі низки найважливіших сільськогосподарських культур, проведена їх апробація у ряді регіонів країни.

В основу цієї моделі покладено попередні розробки, що склали основу сімейства моделей AGROTOOL. Динамічна модель описує продукційний процес рослин від моменту посіву до повного дозрівання. Вона має блокову структуру (рис. 1.1) і включає в себе опис наступних процесів, що мають місце в системі "ґрунт - рослинний покрив - приземний шар повітря":

- радіаційний режиму посіву, що включає в себе моделювання поглиненої посівом радіації, теплової радіації;
- фотосинтез та фотодихання;
- розвиток рослин;
- розподіл накопичених продуктів фотосинтезу по органах рослини, ріст рослин та формування врожаю;
- динаміка ґрунтової вологи;
- прогнозування темпів розвитку рослин;
- прогнозування урожаю.

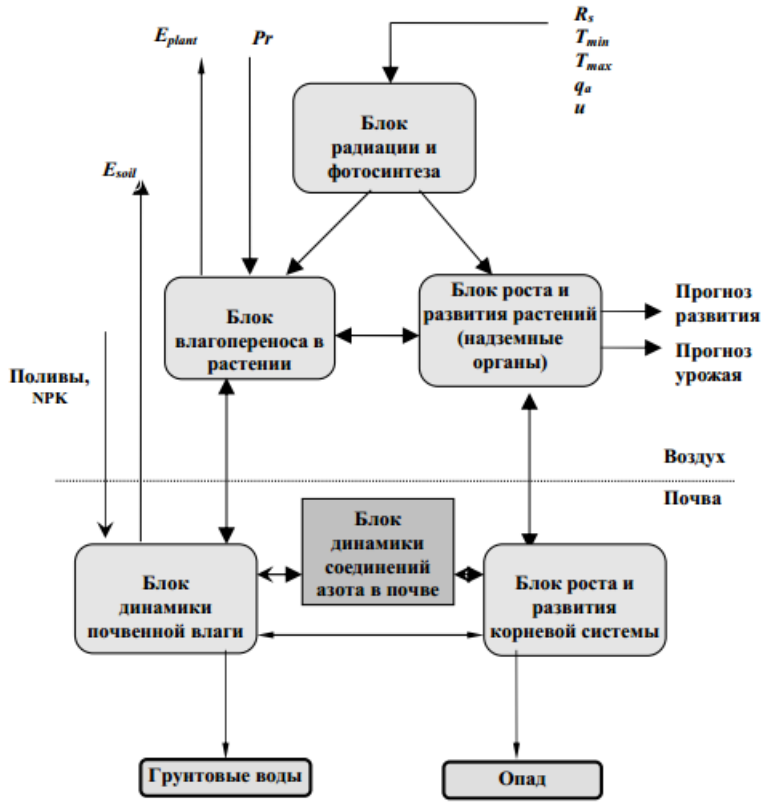


Рис. 1.1. Блок-схема моделі.

Ця інформаційна система дозволяє розв’язувати наступні завдання:

- аналіз результатів продукційного процесу минулого року вегетації;
- оцінка біжучого стану агроценозу;
- динамічне прогнозування очікуваних результатів року.

1) Аналіз результатів продукційного процесу минулого року вегетації. У цьому режимі можна вибрати рік вегетації з набору років, у яких є експериментальна

інформація, і отримати можливість оцінити обрану стратегію управління або порівняти результати моделювання з фактичними даними.

2) Оцінка поточного стану агроценозу. При цьому погодні умови до дати розрахунку відомі і можна розрахувати поточний стан посіву – доступну для рослин вологу або листовий індекс.

3) Прогнозування ходу вегетації. Початкові умови та фактичні дані про погоду та режими поливу вводяться до дня прогнозу. Користувач отримує інформацію про перебіг вегетаційного періоду.

Урожайність сільськогосподарських культур являє собою комплексний індикатор, оскільки, з однієї сторони, це вихідна інформація для побудови планів і прогнозів, а з іншої – один з основних результуючих індикаторів агропідприємства. Прогнозування урожайності на основі супутникових даних є перспективним напрямком, тому що використовувані методи прогнозування відрізняються своєю об’єктивністю та оперативністю. З їх основними перевагами можна отримати дані не тільки на широких площах, але і на важкодоступних ділянках, а також наглядність отриманої інформації та можливість повторного зйомки тієї ж місцевості через потрібні інтервали часу.

Рис. 1.2. Графічний інтерфейс інформаційної системи SoftFarm.

В якості основного показника для побудови прогнозної моделі застосовується вегетаційний індекс NDVI. Як правило, використовуються максимальні значення його ряду, які мають високу динамічну кореляцію з урожайністю. Тим не менше, якщо врахувати лише фактичний максимум вегетації, який у різні роки досягається

нерівномірно, то знижується можливість раннього прогнозування, а значить і його практична цінність. Щоб цього уникнути, беруться до уваги інші показники, які характеризують метеорологічні та кліматичні особливості досліджуваного регіону. Чим повніша та достовірніша інформація, внесена в систему для аналізу, тим більш точно буде складено прогноз. У результаті вивчення даних про значимість та їх вплив на врожайність, SoftFarm при розрахунках враховує тип і попередню культуру, дати посівів і збору, фактичну урожайність, а також суми опадів і суми активних температур за минулий рік і з початку поточного року.

Рис. 1.3. Проведення досліджень в інформаційній системі SoftFarm.

На першому етапі прогнозування необхідно досліджувати змінність вегетаційного індексу NDVI в різні часові періоди, а також розраховувати його середню багаторічну динаміку, яка застосовується для аналізу оцінки стану посівів. При розрахунках значень використовуються попередньо оброблені дані по посівних площах, які безпосередньо впливають на точність моделі прогнозування. Далі для вибраних територій формуються динамічні ряди значень NDVI по конкретному полю та відповідній культурі. Веб-сервіс SoftFarm має велику базу даних по посівах та урожайності, що становить 2500000 га по всій Україні. Система порівнює показники вегетації з аналогічними графіками в базі даних і при появі нового супутникового знімку кожного разу оновлює прогноз.

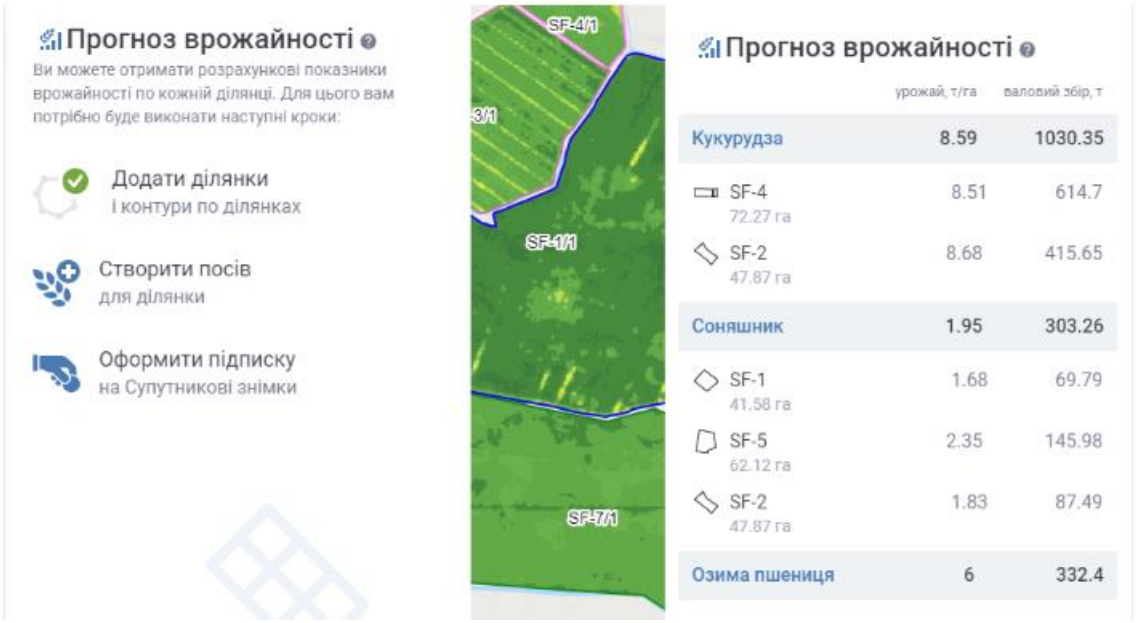


Рис. 1.4. Прогноз врожайності сільськоосподарських культур в системі SoftFarm.

Використання результатів прогнозування дозволяє оцінити доходи, що дає можливість коригувати плани господарства з розподілу ресурсів. Завдяки прогнозу врожайності в SoftFarm у фермера буде розуміння витрат на реалізацію заходів, які будуть спрямовані на збільшення врожайності.

Враховуючи важливість прогнозування, тепер цей модуль знаходиться на головній сторінці веб-сервісу і являється ще одним із інструментів підвищення ефективності прийняття рішень.

Інформаційно-аналітична система SoftFarm дозволяє проводити:

- прогнозування врожайності;
- оцінку загального стану посівів;
- виявлення проблемних участків.

1.3. Супутниковий моніторинг урожайності сільськогосподарських рослин

Дане завдання базується на відбивальній здатності листової поверхні рослин. Тобто сонячне випромінювання, попадаючи на листя рослин, частина його поглинається, інша частина – відбивається. Якщо взяти випромінювання в червоній області спектру, то вого дуже добре поглинається хлорофілом. Кліткова структура листя навпаки досить добре відбиває інфрачервоне випромінювання. Ця комбінація поглинання червоного та відбивання інфрачервоного дозволяє зробити висновки про

здоров’я рослин, про їх стан. На основі цих всіх різних комбінацій розробляють різні вегетаційні індекси, одним з яких є NDVI.

Рис. 1.5. Датчик для вимірювання біомаси рослин та урожайності.

Стандартні області застосування даного пристрою: зчитування даних та агрономічні дослідження, вимірювання біомаси рослин та нерівномірності рослинного покриву, визначення потенціалу урожаю. Датчик дозволяє отримати значення в реальному часі для рослинних культур.

Вхідними даними служить інформація про границі полів та культурах на них. Для кожного поля були отримані щотижневі усереднені дані про вегетаційний індекс NDVI. Отриманий часовий ряд був згладжений методом поліноміальної апроксимації. Також були отримані дані про середньодобову температуру.

Для оцінки стану посівів необхідна статистика, яку можна отримати на основі багаторічних даних, які враховуватимуть можливу варіацію метеоумов. Для різних районів та років можуть спостерігатися значні варіації динаміки розвитку культур, що не дозволяє порівнювати багаторічні часові ряди.

Відомо, що швидкість розвитку рослин залежить від температури повітря, а для досягнення рослинами певної фази росту необхідно накопичення певної суми температур. Приведення часових рядів вегетаційних індексів до єдиної шкали накопичених температур дозволяє оцінювати відхилення посівів від норми у фазах їх розвитку.

Супутниковий моніторинг посівів – це технологія дистанційного спостереження за розвитком посівів, що базується на аналізі супутникових зображень. Основні завдання: моніторинг станів посівів протягом вегетаційного періоду, обслуговування

великих територій, ідентифікація зон неоднорідностей розвитку посівів, підтримка при формуванні точних прогнозів врожайності, дослідження біофізичних властивостей рослин.

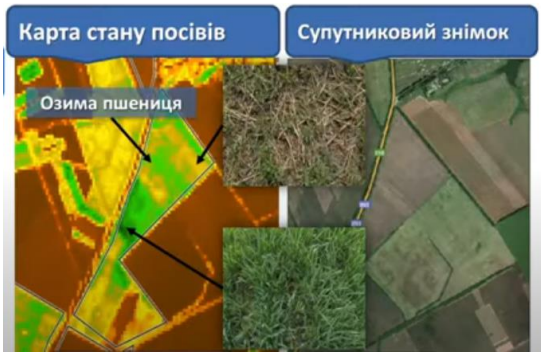


Рис. 1.6. Дистанційне дослідження стану посівів.

За супутниковими даними можна ідентифікувати ступінь розвитку рослинності в тій чи іншій частині поля. Так на рис. 1.6 представлено окремі фрагменти та відповідні фотографії для посівів пшениці від слабкого до достатньо високого ступеня розвитку вегетації.

Розглянемо основні завдання супутникового моніторингу.

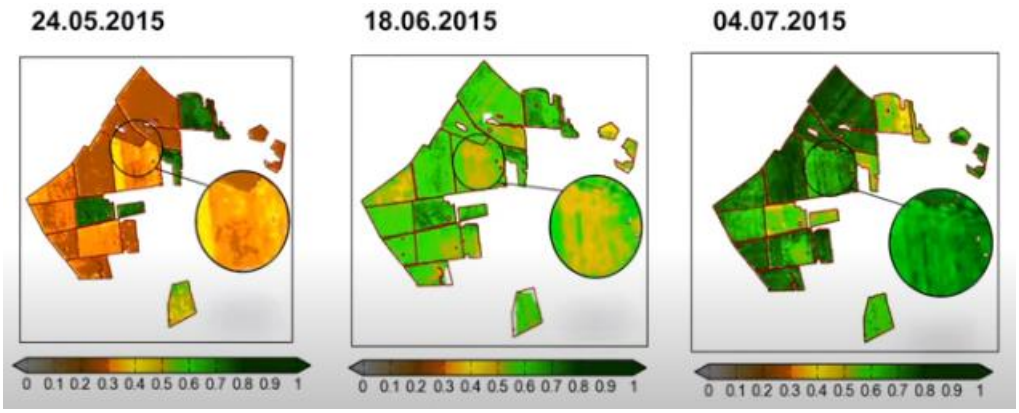


Рис. 1.7. Моніторинг динаміки розвитку посівів на прикладі тематичних карт вегетаційного індексу NDVI.

В даному прикладі розглядаються дані за тематичними картами NDVI станом на 24 травня, 18 червня та 4 липня 2015 р. для одного і того ж поля. Це дозволяє відслідкувати динаміку розвитку стану посівів протягом вегетаційного періоду. Великою перевагою використання саме супутникового моніторингу є те, що дана технологія легко адаптується для різних масштабів.

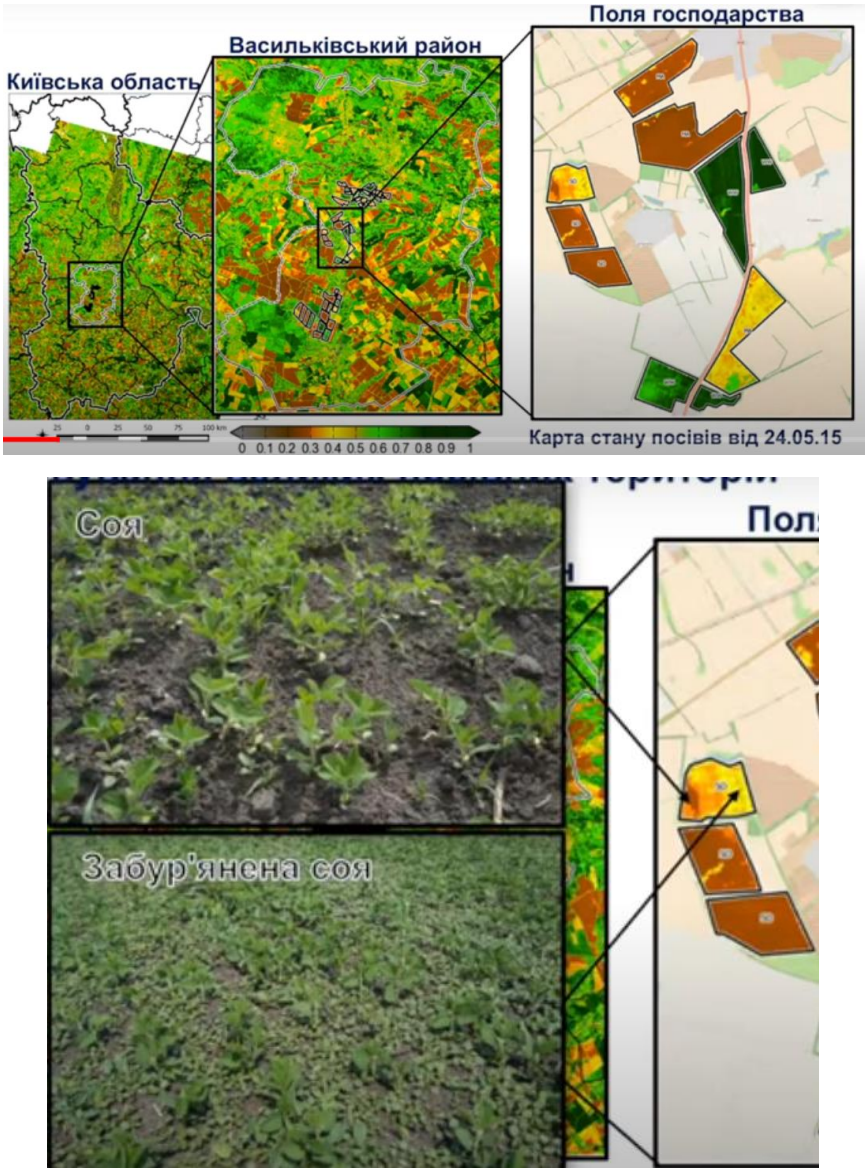


Рис. 1.8. Оцінка динаміки полів.

Можна оцінювати динаміку полів (рис. 1.8). Цей приклад є доволі характерним, оскільки інтерпретація продуктів супутникового агромоніторингу потребує значного експертного досвіду та знання предметної області. В даному випадку вегетаційний індекс у ділянки поля є нижчим, однак насправді ступінь розвитку самої культури є доволі непоганим. Супутниковий моніторинг спирається на використання вегетаційних індексів.

Sen2Agri – це система Sentinel-2 для сільського господарства; він призначений для автоматичного створення ключових продуктів для моніторингу сільського господарства на основі даних Sentinel-2 і Landsat-8. Для маркування посівних угідь вони використовують комплекси складних методів, включаючи комп’ютерний зір та нормовані індекси рослинності.

Cropio – це супутникова система управління полями, яка полегшує дистанційний моніторинг сільськогосподарських угідь. Система забезпечує оновлення в режимі реального часу про поточні умови поля та посівів, визначає рівні рослинності та визначає проблемні ділянки, надає точні прогнози погоди.

Geosys – це компанія, яка використовує зображення, отримані супутниками Landsat 8 і Sentinel 2. Крім того, вони також використовують зображення з приватних супутників та інших постачальників наукових даних.

Descartes Labs використовує супутникові зображення для моделювання складних систем, таких як лісове та сільське господарство. Вони обробляють потоки даних у масштабі, щоб забезпечити миттєвий доступ до готових до аналізу зображень в інтерфейсі з можливістю пошуку та на вимогу.

ВИСНОВКИ ДО РОЗДІЛУ 1

В першому розділі проаналізовано предметну область, засоби та технології проектування інформаційно-аналітичної системи для дослідження росту та прогнозування урожайності сільськогосподарських культур.

З допомогою цієї інформаційно-аналітичної системи можна буде моделювати динаміку росту рослин та прогнозувати майбутню урожайність.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1. Аналіз часових рядів урожайності сільськогосподарських культур

Розглянемо тимчасові ряди даних NDVI для деяких видів рослинних культур. Вегетаційний індекс (BI) – це показник, який розраховується в результаті операцій з різними спектральними діапазонами (каналами), і який має відношення до параметрів рослинності в даному пікселі фотографії. Розрахунок більшої частини вегетаційних індексів базується на двох найбільш стабільних участках кривої спектральної відбивальної здатності рослин.

На червону ділянку спектра (0,62-0,75 мкм) приходиться максимум поглинання сонячної радіації хлорофілом листя, а на ближню інфрачервону (0,75-1,3 мкм) припадає максимальне відбивання енергії структурою листя. Тобто висока фотосинтетична активність веде до більш низьких значень коефіцієнта відбивання в червоній зоні спектру і набагато більшим значенням в ближній інфрачервоній. Відношення цих показників дозволяє чітко вирізняти рослинність від всіх інших об’єктів.

Як видно на рис. 2.1, для зеленого листя (зелений графік) крива відбивальних характеристик мінімальна.

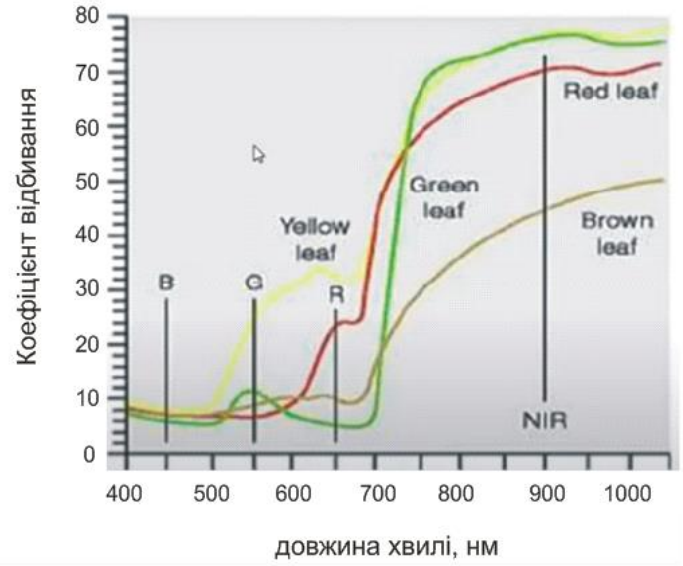


Рис. 2.1. Спектри відбивання при різному вмісті хлорофілу в листі.

Завдяки цьому, використовуючи вегетаційний індекс, можна надійно ідентифікувати рослини в період вегетації. NDVI (Normalized Difference Vegetation Index) – це нормалізований відносний вегетаційний індекс. Він найбільше поширений

у сільському господарстві та характеризує густоту рослинності та дозволяє аграріям оцінити ріст рослин, наявність бур'янів чи хвороб рослин, а також спрогнозувати продуктивність полів.

Показники цього індекса формуються через супутникові знімки зеленої маси рослин на полях, яка поглинає електромагнітні хвилі у видимому червоному діапазоні і відбиває їх у ближньому інфрачервоному:

$$\mathrm{NDVI} = \frac{\mathrm{NIR} - \mathrm{RED}}{\mathrm{NIR} + \mathrm{RED}}, \qquad (2.1)$$

де NIR – інтенсивність випромінювання, яке відбите від поверхні рослин в ближньому інфрачервоному каналі, RED – інтенсивність випромінювання, яке відбите від поверхні рослин в червоному каналі.

В даній роботі проведено дослідження залежності індекса NDVI від сезонних змін в циклі росту основних типів сільськогосподарських культур.

Для проведення аналізу урожайності полів використано космічні знімки поверхні Землі з допомогою супутників SENTINEL-2. Дослідження проводилися з використанням даних дистанційного зондування Землі. Завдання полягало в отриманні числових значень індексів NDVI конкретних полів на вибраній території протягом певного проміжку часу та програмної візуалізації залежності індексів NDVI від часу з подальшим аналізом отриманих результатів.

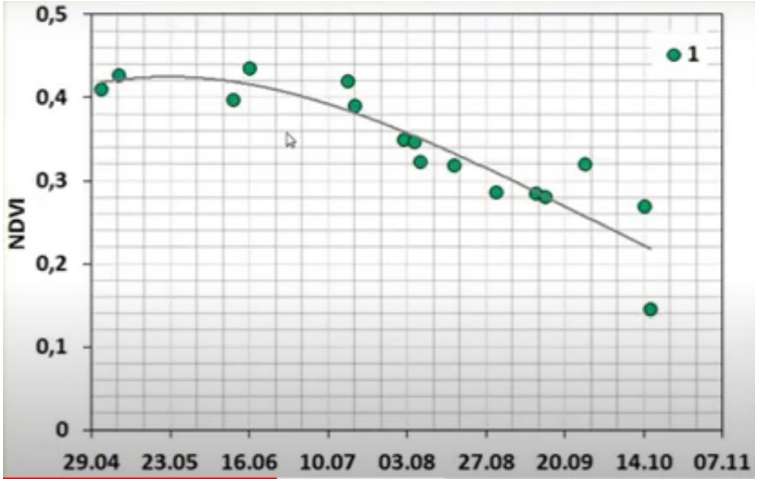


Рис. 2.2. Графік індексу NDVI для пшениці.

З приходом перших теплих днів в квітні спостерігається стрімкий ріст культури. Тому на графіку не спостерігається яскраво вираженого піку вегетації культури. З травня по червень NDVI знаходиться приблизно в однакових межах в районі 0,4.

Далі спостерігається його зниження, що можна пояснити з поступовим засиханням рослин, починаючи з липня місяця.

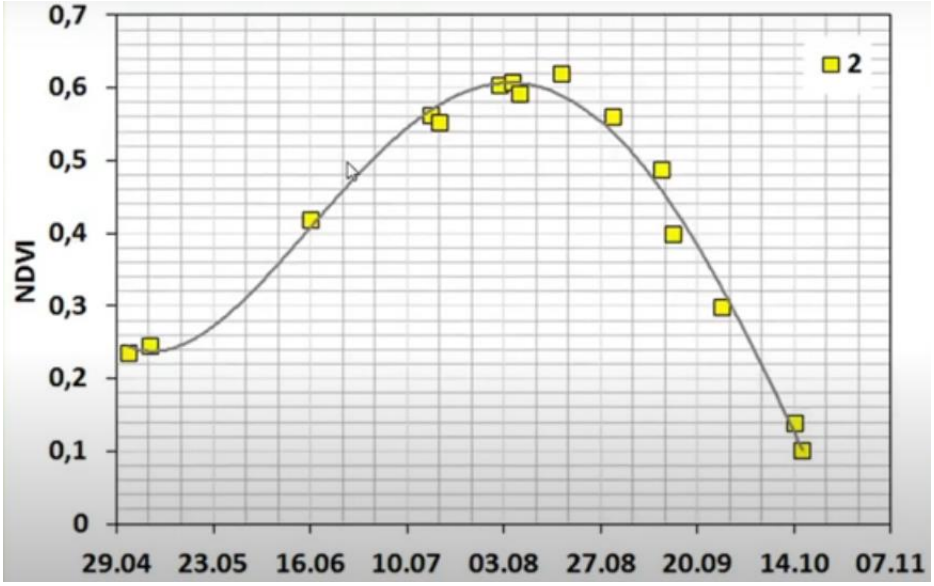


Рис. 2.3. Графік індексу NDVI для полів кукурудзи.

На рис. 2.3 по росту кривої вегетаційної індикації можна зробити висновок про постійний ріст кукурудзи. Максимум вегетації закінчується на початку серпня. NDVI при цьому досягає рівня 0,6. Після цього спостерігається зниження цього індекса, що пов’язане з появою врожаю кукурудзи переважно жовтого кольору та висиханням листя. Подальше зниження індекса пояснюється засиханням даної культури.

Виходячи з аналізу отриманих даних, були виділені основні етапи розвитку рослин. Індекс NDVI являється універсальним індексом для оцінки вегетації рослин, з допомогою якого можна відслідковувати ріст культур і при потребі своєчасно втручатися в нього для отримання максимального результату урожайності.

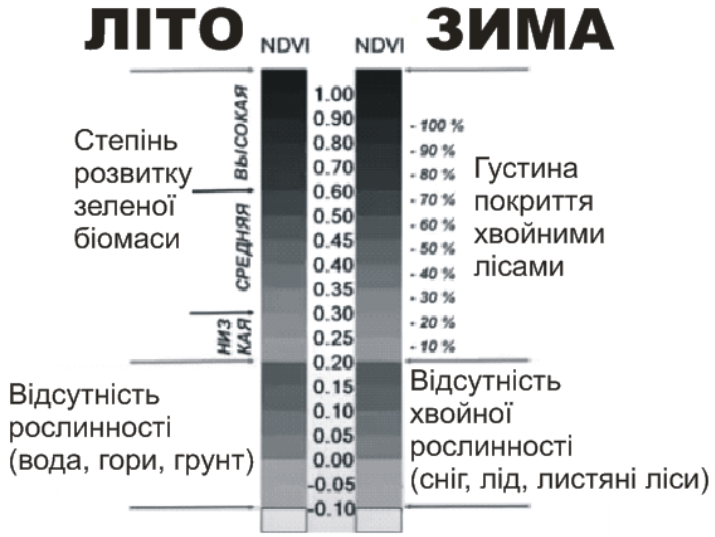


Рис. 2.4. Значення індексу NDVI та відповідні йому типи рослинного покрову.

Як відомо, відбивання рослинним покривом в червоній та ближній інфрачервоній областях електромагнітного спектру тісно пов’язана з його зеленою фітомасою. Для того, щоб кількісно оцінити стан рослинності, використовують даний індекс. Він змінюються в діапазоні від -1 до 1 (рис. 2.4). Для зеленої рослинності відбивання в червоній області завжди є меншим, ніж в ближній інфрачервоній, за рахунок поглинання світла хлорофілом, тому цей коефіцієнт для рослинності не може бути від’ємним.

Характерною ознакою рослинності та її стану є спектральна відбивна здатність, що характеризується великими відмінностями у відображенні випромінювання різних довжин хвиль. Знання про зв’язок структури та стану рослинності з її спектральними відбиваючими властивостями дозволяють використовувати аерокосмічні знімки для ідентифікації типів рослинності та їх стану.

Спектральні індекси, що використовуються для вивчення та оцінки стану рослинності, отримали назву вегетаційних індексів. Вегетаційний індекс (BI) – показник, що розраховується в результаті операцій з різними спектральними діапазонами (каналами) даних дистанційного зондування, який має відношення до параметрів рослинності в даному пікселі фотографії.

Налічують досить багато варіантів вегетаційних індексів. Вони підбираються експериментально, виходячи з відомих особливостей кривих спектральної відбивної здатності рослинності та грунтів.

Розрахунок більшої частини вегетаційних індексів базується на двох найбільш стабільних ділянках кривої спектральної відбивної здатності рослин (рис. 2.4). На червону зону спектра (0,62-0,75 мкм) припадає максимум поглинання сонячної радіації хлорофілом, але в ближню інфрачервону зону (0,75-1,3 мкм) припадає максимальне відбивання сонячної енергії структурою листя.

Для рослинності індекс NDVI приймає додатні значення, чим більша зелена фітомаса, тим він більший. На значення індексу впливає також видовий склад рослинності, грунт під рослинністю. Для зеленої рослинності індекс зазвичай набуває значень від 0,2 до 0,8. Розрахунок індексу для кожного пікселя космічного знімка по червоній та ближній інфрачервоній спектральних зонах дозволяє отримати зображення – карту NDVI.

NDVI дозволяє виявити проблемні зони пригніченої рослинності, дозволяючи приймати рішення, створені для підвищення врожайності. Ділянки з різним станом рослинності або об’ємом зеленої фітомаси можуть будуть відображені різними кольорами. За допомогою статистичної обробки карт NDVI крім визначення кількості фітомаси можна також виділити площі посіву різних сільськогосподарських культур. Також на їх основі можливе отримання чисельних даних для використання у розрахунках оцінки та прогнозування врожайності та продуктивності, біологічної різноманітності.

2.2. Мова програмування Python

Для реалізації програм розрахунку індексів і візуалізації результатів була обрана високорівнева мова програмування Python. Дана мова програмування орієнтована на підвищення продуктивності розробника і читабельність коду. Стандартна бібліотека мови включає великий об’єм корисних функцій.

Python відкриває доступ до структурного, функціонального, об’єктно-орієнтованого програмування. Автоматичне управління пам’яттю, зручні високорівневі структури, динамічна типізація (визначення типу змінної тільки під час виконання програми), підтримка багатопоточних обчислень (можливість запросити тип і структуру об’єкту під час виконання програми) є його основними архітектурними особливостями.

Перевагою даної мови є багата стандартна бібліотека. Писати кросплатформені програми дозволяє набір модулів для роботи з операційною системою. Крім стандартної бібліотеки існують багато інших бібліотек для різних сфер діяльності (обробка зображень, чисельні методи).

При розрахунку індексів NDVI потрібно було обробляти великі масиви даних і проводити з ними обчислення. Для цих цілей була використана бібліотека мови Python – NumPy.

2.3. Бібліотека для візуалізації результатів досліджень Matplotlib

Бібліотека Matplotlib в Python допомагає відображати дані на графіках.

Елемент графіка Figure представляє собою вікно, в якому побудований графік. Axes – це область, на якій відображаються дані. Це може бути вісь X, вісь Y і т.д. Spines – це лінії, які з'єднують точки Axes.

Бібліотеку Matplotlib можна встановити з допомогою модуля pip:

```
pip install matplotlib
```

Для побудови лінійного графіку використовують два списки Python в якості джерела даних для точок графіку:

```
import matplotlib.pyplot as plt
year = [1950, 1975, 2000, 2018]
population = [2.12, 3.681, 5.312, 6.981]
plt.plot(year, population)
plt.show()
```

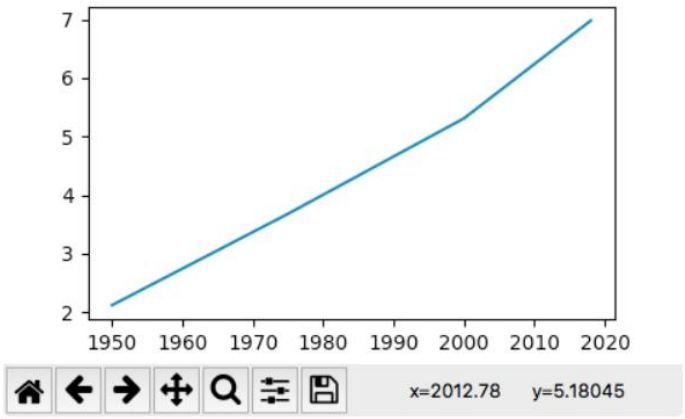


Рис. 2.8. Побудова графіку.

```
plt.xlabel('Year')
plt.ylabel('Population')
plt.title('World Population')
```

Графік, який приведений на рис. 2.8, показує точки, які фактично не були передані в масиві, оскільки він показує лінію. Якщо потрібно побудувати самі точки на графіку без лінії, це допоможе зробити діаграма розсіювання:

```
import matplotlib.pyplot as plt
year = [1950, 1975, 2000, 2018]
population = [2.12, 3.681, 5.312, 6.981]
plt.scatter(year, population)
plt.show()
```

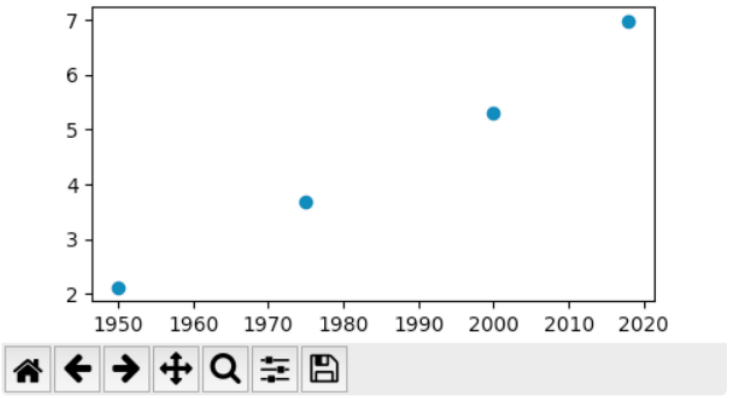


Рис. 2.9. Точковий графік (діаграма розсіювання).

Розглянемо як побудувати в Python гістограму.

В той час як графіки інформують про те, як змінюються дані, гістограма описує, як ці дані розподіляються. Чим більше значень в діапазоні, тим вище смуга діапазону. Для цього використовують функцію hist() для побудови гістограми. У неї є два важливі параметри: список значень для побудови, кількість діапазонів для розподілу цих точок.

```
import matplotlib.pyplot as plt
values = [0, 1.2, 1.3, 1.9, 4.3, 2.5, 2.7, 4.3, 1.3, 3.9]
plt.hist(values, bins = 4)
plt.show()
```

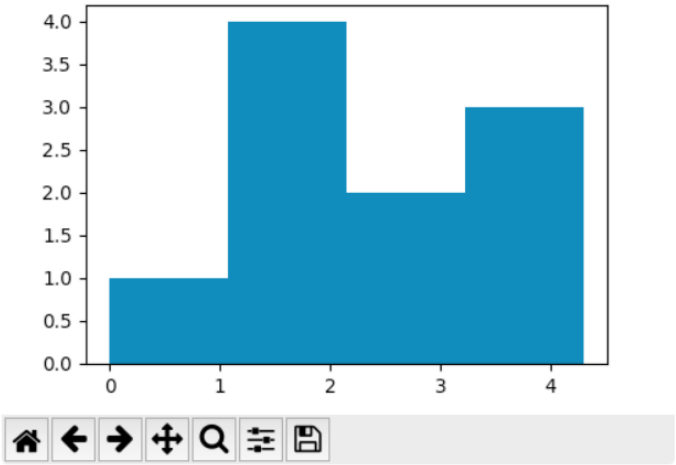


Рис. 2.10. Побудова гістограми.

Налаштування графіків

Для того щоб змінити перший графік, потрібно, щоб вісь Y починалася з 0. За це відповідає такий код:

```
plt.yticks([0, 2, 4, 6, 8, 10])
```

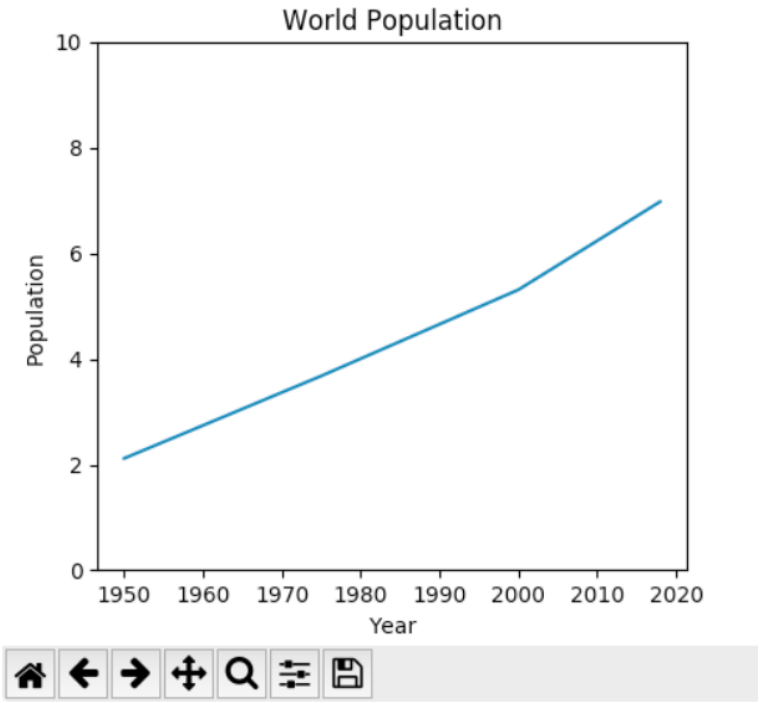


Рис. 2.11. Зміна графіку.

За побудову декількох кривих на одному графіку відповідає такий фрагмент коду:

```
import numpy as np
import matplotlib.pyplot as plt
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
cos, sin = np.cos(X), np.sin(X)
plt.plot(X, cos)
plt.plot(X, sin)
plt.show()
```

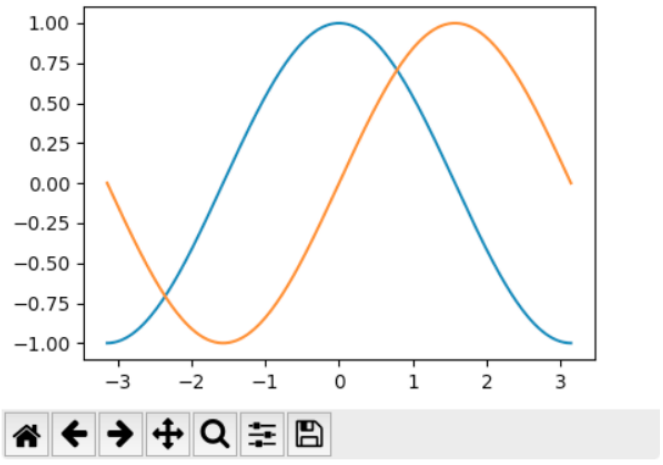


Рис. 2.12. Побудова двох кривих в одному вікні графіку.

Щоб змінити колір кривих та добавити написи, потрібно добавити такий код:

```
import numpy as np
import matplotlib.pyplot as plt
X = np.linspace(-np.pi, np.pi, 256, endpoint=True)
cos, sin = np.cos(X), np.sin(X)
plt.plot(X, cos, color='blue', label="cosine")
plt.plot(X, sin, color='red', label="sine")
plt.legend(loc='upper left', frameon=False)
plt.show()
```

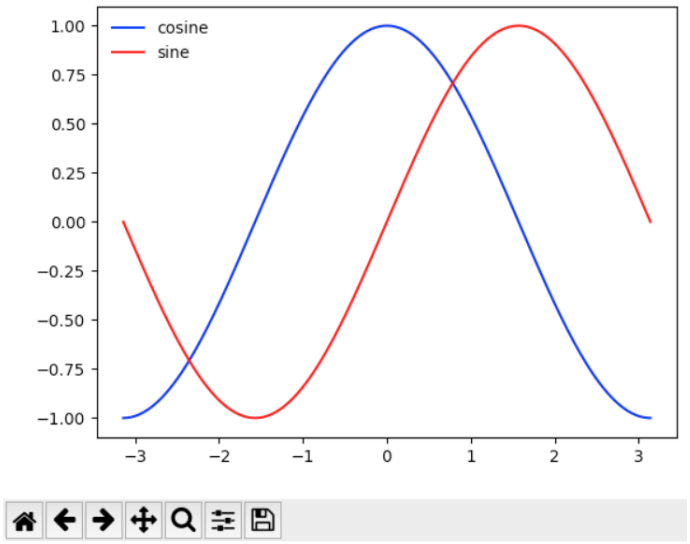


Рис. 2.13. Зміна кольору кривих та добавлення легенди до графіку.

За побудову графіка гістограми відповідає такий код:

```
import matplotlib.pyplot as plt; plt.rcdefaults()
import numpy as np
import matplotlib.pyplot as plt
names = ('Tom', 'Dick', 'Harry', 'Jill', 'Meredith', 'George')
y_pos = np.arange(len(names))
speed = [8, 7, 12, 4, 3, 2]
plt.bar(y_pos, speed, align='center', alpha=0.5)
plt.xticks(y_pos, names)
plt.ylabel('Speed')
plt.title('Person')
plt.show()
```

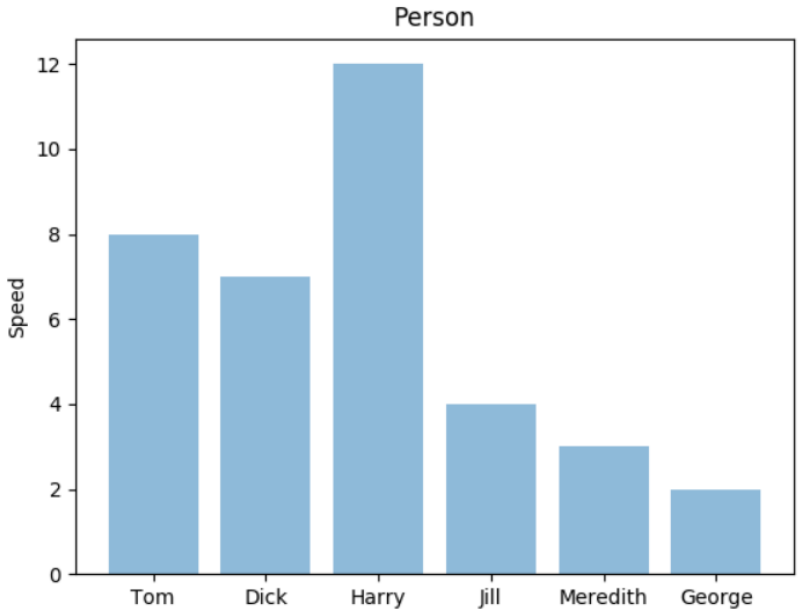


Рис. 2.14. Побудова гістограми.

За створення кругової діаграми відповідає наступний фрагмент коду:

```
import matplotlib.pyplot as plt
names = 'Tom', 'Dick', 'Harry', 'Jill', 'Meredith', 'George'
speed = [8, 7, 12, 4, 3, 2]
colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue', 'red',
'blue']
explode = (0.1, 0, 0, 0, 0, 0)
plt.pie(speed, explode=explode, labels=names, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal')
plt.show()
```

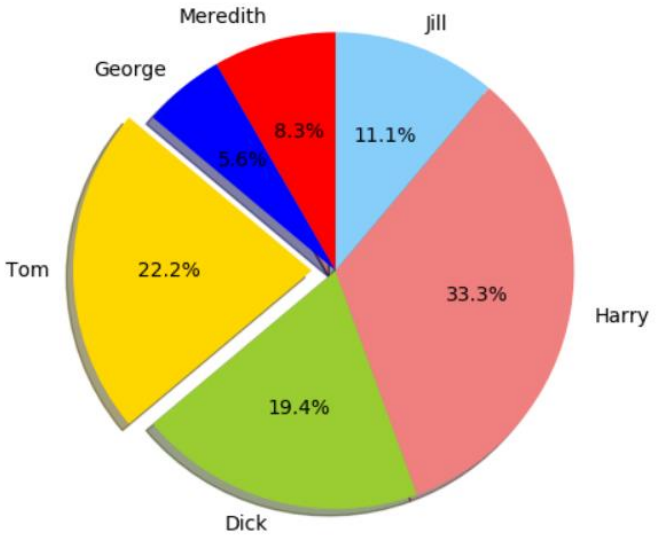


Рис. 2.15. Побудова кругової діаграми.

За створення теплових карт відповідає наступний фрагмент коду:

```
import numpy as np
import numpy.random
import matplotlib.pyplot as plt
temperature = np.random.randn(4096)
anger = np.random.randn(4096)
heatmap, xedges, yedges = np.histogram2d(temperature, anger,
bins=(64,64))
extent = [xedges[0], xedges[-1], yedges[0], yedges[-1]]
plt.clf()
plt.ylabel('Anger')
plt.xlabel('Temp')
plt.imshow(heatmap, extent=extent)
plt.show()
```

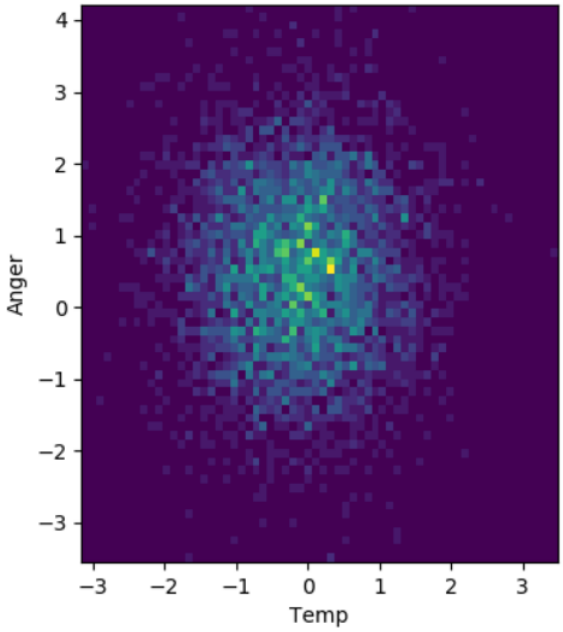


Рис. 2.16. Побудова теплових карт.

ВИСНОВКИ ДО РОЗДІЛУ 2

У другому розділі приведено відомості про часові ряди для прогнозування урожайності. Описано принципи їх функціонування, технології їх розробки.

Розглянуто засоби, з допомогою яких проектується дана інформаційна система. Приведено відомості про особливості проектування програмного продукту засобами Python, проектування графічного інтерфейсу засобами бібліотеки PyGui, особливості візуалізації результатів з допомогою бібліотеки Matplotlib.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1. Математична модель росту рослин для прогнозування урожайності

В математичній моделі розглядається можливість визначення фітомаси рослин і прогнозування урожайності на прикладі моделі їх розвитку. Розглянемо можливість прогнозування врожайності на основі мінімальних польових даних і на основі простих моделей розвитку рослин, тобто даних, які можна в основному отримати дистанційно (по космічних, метеорологічних даних) і використовувати на тих полях, для яких відсутня статистична інформація про врожайність. Тобто на основі якогось одного року без яких-небудь даних за попередні роки.

Модель розвитку рослин – це динамічна модель, яка описує процеси росту і розвитку рослини в цілому з врахуванням основних факторів зовнішнього середовища, які впливають на ці процеси і які орієнтовані на практичне використання в сільському господарстві.

Найбільш відомі моделі розвитку рослин: WOFOST, DASSAT, APSIM, MONICA, AquaCrop та багато інших. В нашому випадку будемо використовувати модель SAFY, яка трохи простіша від попередніх.

Рис. 3.1. Моделі росту рослин.

Модель SAFY (Simple algorithm for yield estimates) – це модель розвитку рослинних культур, в основі якої лежить рівняння Монтейта:

$$\Delta DAM = APAR \cdot ELUE \cdot F_T(T_a). \quad (3.1)$$

$$\mathrm{DAM_{tot}} = \sum_{x}^{y} \Delta \mathrm{DAM}; \tag{3.2}$$

$$\mathrm{GY = HI \cdot DAM_{tot}}, \tag{3.3}$$

де ΔDAM – щоденний приріст фітомаси; $\mathrm{DAM_{tot}}$ – загальна фітомаса; APAR – поглинута фотосинтетична активна радіація; ELUE – максимальна ефективність поглинання світла; $F_T(T_a)$ – функція оптимальної температури; x, y – перший та останній день росту зеленої фітомаси; HI – індекс урожайності; GY – фактична урожайність.

В основі даної моделі лежить рівняння Монтейта, яке показує залежність приросту фітомаси від ефективності поглинання світла рослинами. Щоденний приріст фітомаси ΔDAM залежить від поглинутої фотосинтетичної радіації (APAR), максимальної ефективності поглинання світла (ELUE) та від функції оптимальної температури $F_T(T_a)$.

Завдяки тому що ми взнаємо щоденний приріст фітомаси, можна вияснити, яка сумарна фітомаса виросла на даному полі протягом сезону. Завдяки відповідності між приростом фітомаси і кінцевою урожайністю є деякий індекс урожайності (ІУ), який дозволяє через кінцеву фітомасу обчислити урожайність на досліджуваному полі.

Функція оптимальної температури $F_T(T_a)$ обчислювалася на основі відхилень температурних характеристик від оптимальних для рослини в ту чи іншу сторону. Тобто якщо фактична температура була меншою чи більшою за оптимальну температуру, або виходила за її межі, то дана функція змінювалася.

T_{min} – мінімальна температура для розвитку культури;

T_{max} – максимальна температура для розвитку культури;

T_{opt} – оптимальна температура для розвитку культури;

T_a – фактична температура повітря.

Якщо $T_{min} < T_a < T_{opt}$, то:

$$F(T_a) = 1 - \left(\frac{T_{opt} - T_a}{T_{opt} - T_{\min}} \right). \tag{3.4}$$

Якщо $T_{max} > T_a > T_{opt}$, то:

$$F(T_a) = 1 - \left(\frac{T_a - T_{opt}}{T_{max} - T_{opt}} \right). \quad (3.5)$$

Якщо $T_a < T_{min}$, чи $T_a > T_{max}$, то:

$$F(T_a) = 0. \quad (3.6)$$

Поглинута фотосинтетична радіація являється добутком частки фотосинтетично поглинутої радіації, яка помножена на коефіцієнт кліматичної постійної та на сумарну сонячну радіацію:

$$APAR = FAPAR \cdot \varepsilon_c \cdot R_g. \quad (3.7)$$

Розглянемо принцип роботи моделі SAFY, завдяки чому вона дозволяє прогнозувати урожайність. Справа в тому, що накопичення зеленої фітомаси у рослинності не відбувається безпосередньо до кінця сезону. Тобто за один чи півтора місяці до безпосереднього збору урожаю зелена фітомаса рослин зупиняє свій ріст і далі в основному йде процес достигання плодів. Дана модель має залежність між зеленою фітомасою і фактичною урожайністю, це означає, що не обов'язково чекати ці півтора місяці, щоб зробити розрахунки кінцевої урожайності. Видно на графіку щоденний приріст фітомаси, тобто як і в який день рослини виростали. Починаючи з моменту, коли виростання починає зменшуватися, цей показник вже не враховують.

Рис. 3.2. Щоденний приріст фітомаси по днях.

Ця модель розраховує дані щоденно. Тобто дані, які були недоступні для щоденного аналізу, інтерполювалися. Це зокрема дані FAPAR.

В результаті отримали, що теоретична точність моделі при використанні описаної методики складає не менше 85-90 %. Однак для підтвердження її потрібно проведення

серії польових досліджень в різних агрокліматичних умовах і на прикладі різних культур.

3.2. Часові ряди врожайності культур

Питання аналізу часових рядів врожайності, встановлення їх мінливості розглядалися багатьма дослідниками. Розглядали різні аспекти цієї проблеми – від аналізу різних складових часових рядів до можливих шляхів прогнозування урожайності на основі використання закономірностей, які закладені в самих часових рядах.

Було потрібно з допомогою теорії випадкових функцій найкращим чином визначити тенденцію врожайності, дослідити закономірності їх мінливості з часом і в просторі. Це дозволило б успішно проводити розробку статистичних методів прогнозів, які синтезують оцінки очікуваної тенденції урожайності та відхилення урожайності від тренду. Ці дані отримують з допомогою динамічних моделей продуктивності рослин.

Аналіз часових рядів урожайності проводився по такій схемі (рис. 3.1). На першому етапі здійснювалося виділення тенденції урожайності, оцінювалась правильність вибору видів тренду, перевірялася гіпотеза про те, що випадкова компонента представляє собою стаціонарний випадковий процес. Другий етап включав аналіз тенденції урожайності, виділення типів ходу тенденції часових рядів урожаїв. На третьому етапі досліджувалася випадкова компонента, проводився її автокореляційний та спектральний аналізи.

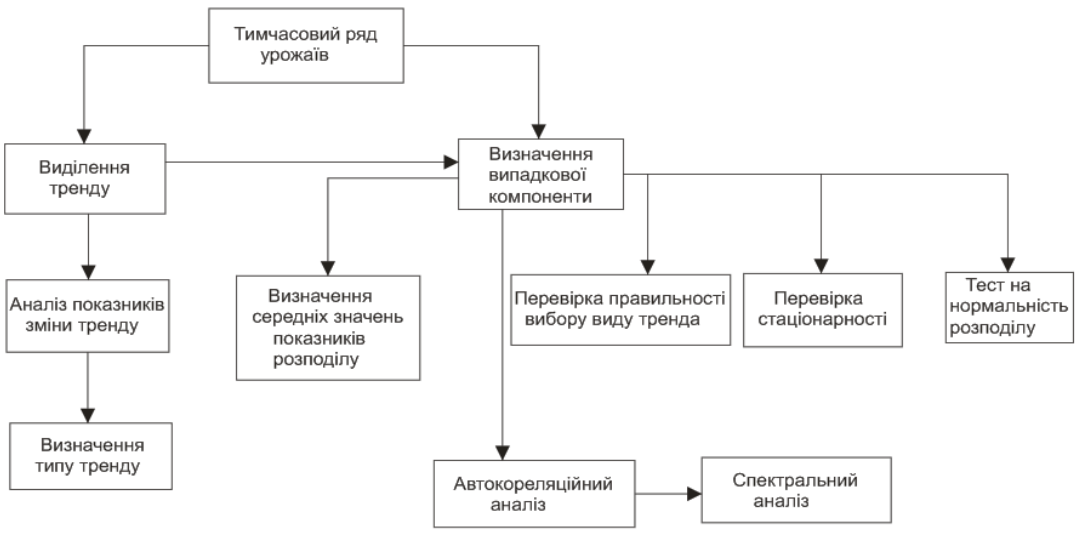


Рис. 3.1. Схема аналізу часових рядів урожайності.

Твердження про наявність тенденції у часових рядах урожайності, тобто про їх нестаціонарність, являється загальновизнаним фактом. Застосування для аналізу часового ряду теорії випадкових функцій вимагає приведення цього ряду до стаціонарної випадкової послідовності. Виділення тенденції часового ряду може бути виконане різними способами. Необхідною умовою при цьому є оцінка коректності визначення тренду і дослідження випадкової компоненти на стаціонарність. Вибір виду тренду врожайності представляє собою багатоваріантну процедуру, правильність здійснення якої вимагає кількісної оцінки.

Завдяки успіхам у розвитку фрактального аналізу та методів штучного інтелекту з'явилися нові методи аналізу та прогнозування динаміки часових рядів. Важливою властивістю часових рядів є наявність у них довготривалої пам'яті. Це означає, що поточне значення ряду залежить не тільки від попередніх йому значень, а й від віддалених значень. Деякі фрагменти часового ряду можуть повторюватися з невеликими змінами. Результати досліджень служать основою прогнозування часових рядів урожайності.

Моделі часових рядів є ефективним інструментом дослідження складних систем. Тимчасовий ряд можна записати у загальному вигляді:

$$\{y_t\}\ t=1, 2, ..., n, \tag{3.8}$$

де t – рівновіддалені моменти спостережень (година, доба, місяць, квартал, рік тощо), y_t – рівні ряду. Часові ряди не можна ототожнювати із звичайними випадковими вибірками. Статистичні характеристики вибірки не змінюються при довільній випадковій перестановці елементів. Основна інформація про систему, що спричинила часовий ряд, полягає у фіксованій часовій послідовності рівнів часового ряду. У більшості випадків послідовні рівні ряду є незалежними. Тому аналіз та прогнозування часових рядів вимагають спеціальних методів, що враховують зазначені особливості.

Для аналізу та прогнозування часових рядів важливим є поняття стаціонарного часового ряду. Стаціонарний часовий ряд – це процес, для якого математичне очікування і дисперсія існують і є постійними величинами, не змінюються в часі, а автокореляційна функція залежить тільки від різниці між двома моментами часу $t_1 - t_2 = \tau$ і залежить від конкретного періоду часу.

Більшість часових рядів аграрних показників не є стаціонарними. В аналізі часових рядів прийнято представляти часовий ряд y_t у вигляді суми детермінованої складової та випадкового відхилення від неї:

$$y_t = \nu_t + s_t + c_t + \varepsilon_t \quad t = 1, 2, ..., n \tag{3.9}$$

Тренд ν_t є невипадковою складовою часового ряду, яка змінюється повільно, та відображає вплив деяких постійних факторів. Складова s_t описує сезонні коливання – систематичні зміни рівнів часового ряду, що мають періодичний характер протягом одного року. Циклічні коливання c_t – це систематичні періодичні зміни рівнів часового ряду більш тривалих інтервалів часу. Тренд та циклічні компоненти є детермінованими компонентами часового ряду. Для виділення детермінованої систематичної компоненти використовують згладжування часового ряду.

Окремі компоненти адитивної моделі (3.9) можуть бути відсутніми. Обов'язковою є наявність лише випадкової складової ε_t, яка завжди супроводжує процес та визначає характер відповідного часового ряду. Аналіз випадкової компоненти є важливою передумовою прогнозування часових рядів. Це пов'язано з тим, що в короткотерміновому та середньотерміновому прогнозуванні результати прогнозу значною мірою визначаються випадковою компонентою, тоді як у довготерміновому прогнозуванні головну роль відіграє тренд та циклічна компонента. Прогнозування випадкової складової часового ряду здійснюється методами теорії випадкових процесів. У прогнозуванні врожайності розглядають статистику серій наростання врожайності, яка дозволяє судити про ймовірність подальшої поведінки на основі аналізу поточного стану системи, яка виражається через значення двох-трьох останніх приростів урожайності.

Для моделювання тенденції зміни часових рядів урожайності застосовуються методи механічного та аналітичного вирівнювань. Найбільш простий прийом механічного вирівнювання ряду – це метод ковзаючого середнього. Для побудови аналітичного виразу тренду найчастіше використовують метод найменших квадратів.

Для аналізу властивостей динамічної системи необхідно позбутися властивого їй тренду. Для виявлення та моделювання тренду необхідно провести попередній аналіз часового ряду. Першим етапом аналізу є побудова графіка досліджуваного процесу. Візуальний аналіз часто допомагає встановити структуру часового ряду. Однак у

більшості випадків необхідні статистичні інструменти, які допомагають встановити стаціонарність ряду, вид тренду, наявність циклічної компоненти, метод серій.

За наявності тренду та циклічної складової значення кожного наступного рівня ряду залежить від попередніх значень. Повну інформацію про наявність у часовому ряду автокореляції надає автокореляційна функція (АКФ).

Система моделей та методів аналізу одномірних часових рядів урожайності побудована на основі системного підходу до математичного моделювання систем та базується на принципі цілісності об’єкта дослідження, дозволяє підвищити глибину аналізу та покращити точність прогнозування врожайності культур.

Більшість з відомих методів середньотермінового та довготермінового прогнозування врожайності ґрунтуються на принципах теорії часових рядів. За останні десятиліття були розроблені нові методи та підходи до прогнозування врожайності. Серія робіт була присвячена розробці нових методів середньотермінового прогнозування та вдосконалення та адаптації відомих методів прогнозування врожайності. Деякі з цих методів вже активно використовуються (метод авторегресії, метод найближчих сусідів, метод нейронних мереж, метод ковзаючого середнього).

Однак слід мати на увазі, що навіть використання відомих методів прогнозування рядів урожайності має свої особливості, без урахування яких прогнозування не може бути ефективним. Насамперед це зумовлено головними особливостями динаміки врожайності – реверсивністю та циклічністю. Тому використання кожного методу прогнозування вимагає тривалого етапу підготовчої роботи, який починається з передпрогнозного аналізу та продовжується шляхом комп’ютерних експериментів, які дозволяють підібрати оптимальну конфігурацію та значення параметрів прогнозних моделей. Дослідження вчених доводять, що з перерахованих вище методів середньотермінового прогнозування зберігають хороші прогнозні властивості при розширенні прогнозного горизонту до 10 років, тобто у разі довготермінового прогнозування.

Однією з головних ознак детермінованої поведінки системи є циклічність. Для лінійних консервативних систем характерна строго періодична циклічність. Більшість природних та економічних систем відносяться до класів нелінійних дисипативних

систем та нелінійних автоколивальних систем. Для таких об’єктів характерні коливання зі змінними значеннями періоду та амплітуди.

Дослідження показали, що часовим рядам врожайності притаманні короткі цикли (4 роки), середні цикли (17–23 роки) та довгі цикли (41–56 років). Короткий цикл швидше за все викликаний циклічністю погодно-кліматичних факторів, середній може бути пояснений у рамках моделі "врожайність-родюча здатність", яка є моделлю типу "хижак-жертва", довгий цикл має технологічні причини. Найбільш чітко виражений ефект циклічності для областей степової зони. Для областей західного регіону України циклічність урожайності менш помітна, що пояснюється впливом клімату. Ефективним способом моделювання часових рядів з ефектом циклічності є лінійно-гармонійна модель врожайності.

Прогнозування врожайності є складним завданням, оскільки динаміка врожайності має змішаний детерміновано-стохастичний характер. Найбільш сильними стохастичними збуреннями є посухи, які можуть призвести до катастрофічного зниження врожайності.

Адекватність моделей можна підвищити, якщо зменшити ефект випадкових збурень. З цією метою використовують згладжування часових рядів. Найбільш вживаними методами згладжування є метод ковзаючого середнього та метод дискретного перетворення Фур’є. Оптимальним варіантом згладжування рядів урожайності є згладжування модифікованим методом ковзаючого середнього з шириною вікна 9 років та поступовим звуженням вікна на краях ряду. Згладжений ряд оптимально відображає детерміновану складову початкового ряду. Основним інструментом прогнозування є лінійно-гармонійна модель для згладженого ряду. Такий підхід дозволяє підвищити прогнозні якості лінійно-гармонійної моделі та дозволяє виконувати прогнози від одного року до кількох років.

ВИСНОВКИ ДО РОЗДІЛУ 3

Модель росту рослин SAFY, яка розраховувалася по даних космічних зйомок, метеоданих та довідкових параметрах, може бути застосована для прогнозу урожайності культур за 1-1,5 місяці до їх дозрівання.

В цій моделі можна використовувати дані дистанційного зондування та метеорологічних умов для визначення фітомаси та урожайності без врахування статистичних даних за попередні роки.

Прогнозування урожайності стає можливим до закінчення стадії активної вегетації культур, тобто за один-півтора місяці до фактичного збору урожаю.

РОЗДІЛ IV. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1. Підхід машинного навчання до прогнозування врожайності сільськогосподарських культур

Прогнозування врожайності сільськогосподарських культур є важливою проблемою сільського господарства. Сільськогосподарська врожайність насамперед залежить від погодних умов (дощ, температура), пестицидів, а точна інформація про історію врожайності сільськогосподарських культур є важливим аргументом для прийняття рішень щодо управління сільськогосподарськими ризиками та майбутніх прогнозів. У цьому проекті передбачено 10 найбільш споживаних врожаїв у всьому світі шляхом застосування методів машинного навчання. В даній дипломній роботі будемо прогнозувати врожайність 10 найбільш поширених культур у всьому світі: Cassava; Maize; Plantains and others; Potatoes; Rice, paddy; Sorghum; Soybeans; Sweet potatoes; Wheat; Yams.

Розроблена інформаційно-аналітична система має передбачати врожайність культур в даному місці за деякими залежними величинами (вхідними параметрами). Потрібно знати назву району та сезонні культури, які вирощуються, а також температуру в даній місцевості, швидкість вітру, тиск та вологість повітря, тип грунту, кількість параметрів N, P, K в грунті. Перш за все потрібно імпортувати бібліотеку Pandas. Ця бібліотека використовується для імпорту наборів даних. Є три типи наборів даних. Це csv, txt, xlsx-файли. Після того як імпортували дані, вони будуть представлені в Pandas у вигляді набору даних.

Незважаючи на велигий сільськогосподарський потенціал, Україна має одну з найнижчих урожаїв з гектара в Європі. Понад 2 млн г. в Україні використовуються неефективно, а майже 5 млн г. не використовуються взагалі. Згадану неефективність можна було б подолати, якби цей сектор керувався всіма даними, які він виробляє щодня. Збір та розповсюдження цих даних дозволить проаналізувати їх та оптимізувати аграрний сектор української економіки.

4.2. Розробка інформаційної системи для прогнозування врожайності сільськогосподарських культур

В роботі застосовуються методи машинного навчання для прогнозування врожайності з використанням загальнодоступних даних FAO та Світового банку даних World Data Bank.

Отримання та робота з даними

Після імпорту необхідних бібліотек урожайність десяти найбільш споживаних культур у світі була завантажена з веб-сайту FAO. Зібрані дані включають країну, товар, рік, починаючи з 1961 до 2016, і значення врожайності.

```
In [1]: import numpy as np
        import pandas as pd
```

```
In [2]: df_yield = pd.read_csv('yield.csv')
        df_yield.shape
Out[2]: (56717, 12)
```

```
In [3]: df_yield.head()
```

Out[3]:

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
0	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1961	1961	hg/ha	14000
1	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1962	1962	hg/ha	14000
2	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1963	1963	hg/ha	14260
3	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1964	1964	hg/ha	14257
4	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1965	1965	hg/ha	14400

```
In [4]: df_yield.tail()
```

Out[4]:

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
56712	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2012	2012	hg/ha	24420
56713	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2013	2013	hg/ha	22888
56714	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2014	2014	hg/ha	21357
56715	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2015	2015	hg/ha	19826
56716	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2016	2016	hg/ha	18294

Дивлячись на стовпці в файлі csv, можна перейменувати Value на hg/ha_yield, щоб було легше розпізнати, що це значення врожайності культур. Також можна видалити такі непотрібні стовпці, як код регіону, домен, код товару тощо.

Перейменуємо стовпці:

```
df_yield = df_yield.rename(index=str, columns={"Value": "hg/ha_yield"})
df_yield.head()
```

Out[5]:

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	hg/ha_yield
0	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1961	1961	hg/ha	14000
1	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1962	1962	hg/ha	14000
2	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1963	1963	hg/ha	14260
3	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1964	1964	hg/ha	14257
4	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1965	1965	hg/ha	14400

Видалимо не потрібні стовпці:

```
df_yield = df_yield.drop(['Year Code','Element Code','Element','Year Code','Area Code','Domain Code','D
df_yield.head()
```

Out[6]:

	Area	Item	Year	hg/ha_yield
0	Afghanistan	Maize	1961	14000
1	Afghanistan	Maize	1962	14000
2	Afghanistan	Maize	1963	14260
3	Afghanistan	Maize	1964	14257
4	Afghanistan	Maize	1965	14400

Для отримки статистичних даних про набір даних:

In [7]:
```
df_yield.describe()
```

Out[7]:

	Year	hg/ha_yield
count	56717.000000	56717.000000
mean	1989.669570	62094.660084
std	16.133198	67835.932856
min	1961.000000	0.000000
25%	1976.000000	15680.000000
50%	1991.000000	36744.000000
75%	2004.000000	86213.000000
max	2016.000000	1000000.000000

З клітинки вище ми знаємо, що фрейм даних починається з 1961 року і закінчується в 2016 році, це всі наявні дані ФАО.

In [8]:
```
df_yield.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 56717 entries, 0 to 56716
Data columns (total 4 columns):
Area           56717 non-null object
Item           56717 non-null object
Year           56717 non-null int64
hg/ha_yield    56717 non-null int64
dtypes: int64(2), object(2)
memory usage: 2.2+ MB
```

Кліматичні дані: кількість опадів

До кліматичних факторів відносяться кількість опадів і температура. Вони є абіотичними компонентами, включаючи пестициди та ґрунт, екологічних факторів, які впливають на ріст і розвиток рослин.

Опади мають серйозний вплив на сільське господарство. Для цього проекту інформацію про кількість опадів за рік було зібрано зі Світового банку даних World Data Bank.

```
In [9]: df_rain = pd.read_csv('rainfall.csv')
        df_rain.head()
```

Out[9]:

	Area	Year	average_rain_fall_mm_per_year
0	Afghanistan	1985	327
1	Afghanistan	1986	327
2	Afghanistan	1987	327
3	Afghanistan	1989	327
4	Afghanistan	1990	327

```
In [10]: df_rain = df_rain.rename(index=str, columns={" Area": 'Area'})
```

Потрібно переконатися, що імена стовпців уніфіковані в усіх датафреймах, що важливо для злиття після очищення. Для перевірки типу даних:

```
df_rain.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6727 entries, 0 to 6726
Data columns (total 3 columns):
Area                            6727 non-null object
Year                            6727 non-null int64
average_rain_fall_mm_per_year   5953 non-null object
dtypes: int64(1), object(2)
memory usage: 210.2+ KB
```

З клітинки вище видно, що тип average_rain_fall_mm_per_year є об’єктом, тому потрібно перетворити його значення на значення з плаваючою крапкою. Для цього конвертуємо average_rain_fall_mm_per_year з об’єкта в float:

```
df_rain = df_rain.convert_objects(convert_numeric=True)
df_rain.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6727 entries, 0 to 6726
Data columns (total 3 columns):
Area                            6727 non-null object
Year                            6727 non-null int64
average_rain_fall_mm_per_year    5947 non-null float64
dtypes: float64(1), int64(1), object(1)
memory usage: 210.2+ KB
```

Далі потрібно видалити будь-які порожні рядки з набору даних і об'єднати датафрейм урожайності з датафреймом про опади за стовпцями року та області:

```
In [13]: df_rain = df_rain.dropna()
```

```
In [14]: df_rain.describe()
```

Out[14]:

	Year	average_rain_fall_mm_per_year
count	5947.000000	5947.000000
mean	2001.365899	1124.743232
std	9.526335	786.257365
min	1985.000000	51.000000
25%	1993.000000	534.000000
50%	2001.000000	1010.000000
75%	2010.000000	1651.000000
max	2017.000000	3240.000000

Дані про кількість опадів починаються з 1985 року і закінчуються в 2016 році. Далі потрібно об'єднати датафрейм урожайності з датафреймом опадів за стовпцями року та області:

```
yield_df = pd.merge(df_yield, df_rain, on=['Year','Area'])
```

Для отримання даних про розмір даатфрейму та інформацію про дані, які в ньому знаходяться:

```
In [16]: yield_df.shape
```

```
Out[16]: (25385, 5)
```

```
In [17]: yield_df.head()
```

Out[17]:

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year
0	Afghanistan	Maize	1985	16652	327.0
1	Afghanistan	Potatoes	1985	140909	327.0
2	Afghanistan	Rice, paddy	1985	22482	327.0
3	Afghanistan	Wheat	1985	12277	327.0
4	Afghanistan	Maize	1986	16875	327.0

Видно, що тепер роки починаються з першого датафрейму про врожайність, початковим роком був 1961 рік, тепер 1985 рік, тому що саме тоді починаються дані про кількість опадів.

```
In [18]: yield_df.describe()
```

Out[18]:

	Year	hg/ha_yield	average_rain_fall_mm_per_year
count	25385.000000	25385.000000	25385.000000
mean	2001.278787	68312.278353	1254.849754
std	9.143915	75213.292733	804.449430
min	1985.000000	50.000000	51.000000
25%	1994.000000	17432.000000	630.000000
50%	2001.000000	38750.000000	1150.000000
75%	2009.000000	94286.000000	1761.000000
max	2016.000000	554855.000000	3240.000000

Розглянемо дані про пестициди, які бли використані в даній роботі. Пестициди, використані для кожного товару та країни, також були зібрані з бази даних FAO:

```
In [19]: df_pes = pd.read_csv('pesticides.csv')
         df_pes.head()
```

Out[19]:

	Domain	Area	Element	Item	Year	Unit	Value
0	Pesticides Use	Albania	Use	Pesticides (total)	1990	tonnes of active ingredients	121.0
1	Pesticides Use	Albania	Use	Pesticides (total)	1991	tonnes of active ingredients	121.0
2	Pesticides Use	Albania	Use	Pesticides (total)	1992	tonnes of active ingredients	121.0
3	Pesticides Use	Albania	Use	Pesticides (total)	1993	tonnes of active ingredients	121.0
4	Pesticides Use	Albania	Use	Pesticides (total)	1994	tonnes of active ingredients	201.0

```
In [20]: df_pes = df_pes.rename(index=str, columns={"Value": "pesticides_tonnes"})
         df_pes = df_pes.drop(['Element','Domain','Unit','Item'], axis=1)
         df_pes.head()
```

Out[20]:

	Area	Year	pesticides_tonnes
0	Albania	1990	121.0
1	Albania	1991	121.0
2	Albania	1992	121.0
3	Albania	1993	121.0
4	Albania	1994	201.0

Для отримання статистичних даних про даний даатфрейм:

```
In [21]: df_pes.describe()
```

Out[21]:

	Year	pesticides_tonnes
count	4349.000000	4.349000e+03
mean	2003.138883	2.030334e+04
std	7.728044	1.177362e+05
min	1990.000000	0.000000e+00
25%	1996.000000	9.300000e+01
50%	2003.000000	1.137560e+03
75%	2010.000000	7.869000e+03
max	2016.000000	1.807000e+06

```
In [22]: df_pes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4349 entries, 0 to 4348
Data columns (total 3 columns):
Area                 4349 non-null object
Year                 4349 non-null int64
pesticides_tonnes    4349 non-null float64
dtypes: float64(1), int64(1), object(1)
memory usage: 135.9+ KB
```

Після цього потрібно об’єднати датафрейм пестицидів із датафреймом урожайності:

```
yield_df = pd.merge(yield_df, df_pes, on=['Year','Area'])
yield_df.shape
```

Out[23]: (18949, 6)

```
In [24]: yield_df.head()
```

Out[24]:

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes
0	Albania	Maize	1990	36613	1485.0	121.0
1	Albania	Potatoes	1990	66667	1485.0	121.0
2	Albania	Rice, paddy	1990	23333	1485.0	121.0
3	Albania	Sorghum	1990	12500	1485.0	121.0
4	Albania	Soybeans	1990	7000	1485.0	121.0

Середня температура

Середня температура для кожної країни була зібрана з даних Світового банку World Bank Data:

```
In [33]: avg_temp=  pd.read_csv('temp.csv')
```

```
In [34]: avg_temp.head()
```

Out[34]:

	year	country	avg_temp
0	1849	Côte D'Ivoire	25.58
1	1850	Côte D'Ivoire	25.52
2	1851	Côte D'Ivoire	25.67
3	1852	Côte D'Ivoire	NaN
4	1853	Côte D'Ivoire	NaN

Для отримання статистичних даних про даний датафрейм:

```
In [35]: avg_temp.describe()
```

Out[35]:

	year	avg_temp
count	71311.000000	68764.000000
mean	1905.799007	16.183876
std	67.102099	7.592960
min	1743.000000	-14.350000
25%	1858.000000	9.750000
50%	1910.000000	16.140000
75%	1962.000000	23.762500
max	2013.000000	30.730000

Таким чином, середня температура починається з 1743 і закінчується в 2013, з деякими порожніми рядками, які потрібно видалити:

```
In [36]: avg_temp = avg_temp.rename(index=str, columns={"year": "Year", "country":'Area'})
         avg_temp.head()
```

Out[36]:

	Year	Area	avg_temp
0	1849	Côte D'Ivoire	25.58
1	1850	Côte D'Ivoire	25.52
2	1851	Côte D'Ivoire	25.67
3	1852	Côte D'Ivoire	NaN
4	1853	Côte D'Ivoire	NaN

```
In [37]: yield_df = pd.merge(yield_df,avg_temp, on=['Area','Year'])
         yield_df.head()
```

Out[37]:

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	Albania	Maize	1990	36613	1485.0	121.0	16.37
1	Albania	Potatoes	1990	66667	1485.0	121.0	16.37
2	Albania	Rice, paddy	1990	23333	1485.0	121.0	16.37
3	Albania	Sorghum	1990	12500	1485.0	121.0	16.37
4	Albania	Soybeans	1990	7000	1485.0	121.0	16.37

Для отримання даних про розмір даного датафрейму:

```
In [38]: yield_df.shape
```

Out[38]: (28242, 7)

Для отримання статистичних даних про даний даатфрейм:

```
In [39]: yield_df.describe()
```

Out[39]:

	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
count	28242.000000	28242.000000	28242.00000	28242.000000	28242.000000
mean	2001.544296	77053.332094	1149.05598	37076.909344	20.542627
std	7.051905	84956.612897	709.81215	59958.784665	6.312051
min	1990.000000	50.000000	51.00000	0.040000	1.300000
25%	1995.000000	19919.250000	593.00000	1702.000000	16.702500
50%	2001.000000	38295.000000	1083.00000	17529.440000	21.510000
75%	2008.000000	104676.750000	1668.00000	48687.880000	26.000000
max	2013.000000	501412.000000	3240.00000	367778.000000	30.650000

```
In [40]: yield_df.isnull().sum()
```

Out[40]:
```
Area                            0
Item                            0
Year                            0
hg/ha_yield                     0
average_rain_fall_mm_per_year   0
pesticides_tonnes               0
avg_temp                        0
dtype: int64
```

Як видно, тепер в даному датафреймі відсутні нульові значення

Дослідження даних

yield_df — кінцевий отриманий датафрейм.

```
In [41]: yield_df.groupby('Item').count()
```

Out[41]:

Item	Area	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
Cassava	2045	2045	2045	2045	2045	2045
Maize	4121	4121	4121	4121	4121	4121
Plantains and others	556	556	556	556	556	556
Potatoes	4276	4276	4276	4276	4276	4276
Rice, paddy	3388	3388	3388	3388	3388	3388
Sorghum	3039	3039	3039	3039	3039	3039
Soybeans	3223	3223	3223	3223	3223	3223
Sweet potatoes	2890	2890	2890	2890	2890	2890
Wheat	3857	3857	3857	3857	3857	3857
Yams	847	847	847	847	847	847

Для отримання статистичних даних даатфрейму yield_df:

```
In [42]: yield_df.describe()
```

Out[42]:

	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
count	28242.000000	28242.000000	28242.00000	28242.000000	28242.000000
mean	2001.544296	77053.332094	1149.05598	37076.909344	20.542627
std	7.051905	84956.612897	709.81215	59958.784665	6.312051
min	1990.000000	50.000000	51.00000	0.040000	1.300000
25%	1995.000000	19919.250000	593.00000	1702.000000	16.702500
50%	2001.000000	38295.000000	1083.00000	17529.440000	21.510000
75%	2008.000000	104676.750000	1668.00000	48687.880000	26.000000
max	2013.000000	501412.000000	3240.00000	367778.000000	30.650000

Можна помітити високу дисперсію значень для кожного стовпця:

```
In [43]: yield_df['Area'].nunique()
Out[43]: 101
```

Фрейм даних містить 101 країну, упорядковану за 10 найвищими об’ємами виробництва:

```
In [44]: yield_df.groupby(['Area'],sort=True)['hg/ha_yield'].sum().nlargest(10)
Out[44]: Area
         India             327420324
         Brazil            167550306
         Mexico            130788528
         Japan             124470912
         Australia         109111062
         Pakistan           73897434
         Indonesia          69193506
         United Kingdom     55419990
         Turkey             52263950
         Spain              46773540
         Name: hg/ha_yield, dtype: int64
```

Індія має найвищу врожайність у наборі даних. Включення елементів у групу за:

```
In [45]: yield_df.groupby(['Item','Area'],sort=True)['hg/ha_yield'].sum().nlargest(10)
```

```
Out[45]: Item            Area
         Cassava         India             142810624
         Potatoes        India              92122514
                         Brazil             49602168
                         United Kingdom     46705145
                         Australia          45670386
         Sweet potatoes  India              44439538
         Potatoes        Japan              42918726
                         Mexico             42053880
         Sweet potatoes  Mexico             35808592
                         Australia          35550294
         Name: hg/ha_yield, dtype: int64
```

Індія є найбільшим виробником маніоки та картоплі. Картопля домінуюча культура в наборі даних, вона є найвищою в 4 країнах. Кінцевий датафрейм починається з 1990 року і закінчується 2013 роком, це дані за 23 роки для 101 країни. Тепер, досліджуючи зв'язки між стовпцями фрейму даних, хороший спосіб швидко перевірити кореляції між стовпцями — це візуалізація кореляційної матриці як теплової карти.

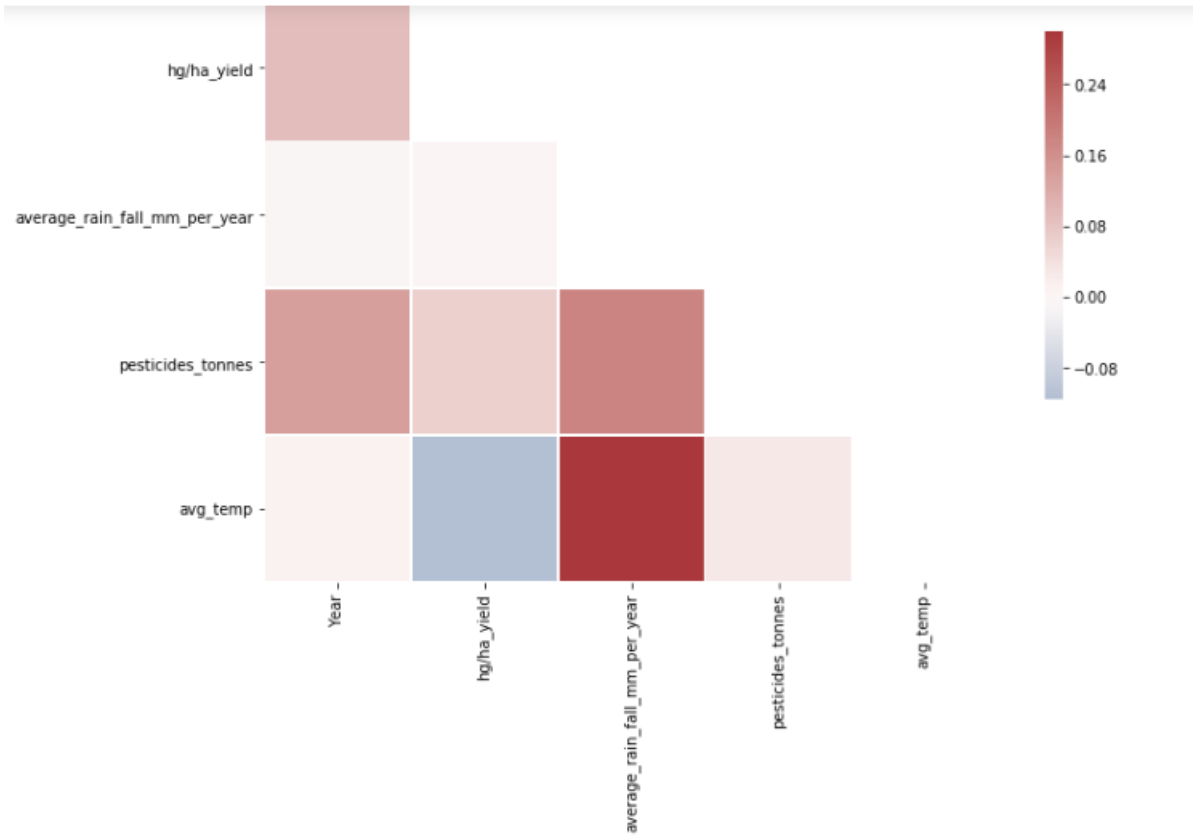
```
In [46]: import sklearn
         import seaborn as sns
         import matplotlib.pyplot as plt
```

Для цього треба створити спеціальну карту кольорів, а саме побудувати теплову карту з маскою та правильним співвідношенням сторін:

```
correlation_data=yield_df.select_dtypes(include=[np.number]).corr()

mask = np.zeros_like(correlation_data, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

f, ax = plt.subplots(figsize=(11, 9))
```

```
cmap = sns.palette="vlag"
```

```
sns.heatmap(correlation_data, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5});
```



З отриманої карти кореляції видно, що немає кореляції між будь-якими стовпцями у датафреймі

Попередня обробка даних

Попередня обробка даних — це техніка, яка використовується для перетворення необроблених даних у чистий набір даних. Іншими словами, щоразу, коли дані збираються з різних джерел, вони збираються в необробленому форматі, який неможливий для аналізу.

```
In [48]: yield_df.head()
```

Out[48]:

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	Albania	Maize	1990	36613	1485.0	121.0	16.37
1	Albania	Potatoes	1990	66667	1485.0	121.0	16.37
2	Albania	Rice, paddy	1990	23333	1485.0	121.0	16.37
3	Albania	Sorghum	1990	12500	1485.0	121.0	16.37
4	Albania	Soybeans	1990	7000	1485.0	121.0	16.37

Для цього проводять в першу чергу кодування категоріальних змінних. У датафреймі є два категоріальні стовпці, категоріальні дані – це змінні, які містять значення міток, а не числові значення. Кількість можливих значень часто обмежується фіксованим набором, як у цьому випадку значення культур і країн. Багато алгоритмів машинного навчання не можуть безпосередньо працювати з даними міток. Вони вимагають, щоб усі вхідні та вихідні змінні були числовими.

Це означає, що категоріальні дані повинні бути перетворені в числову форму. Для цього проводять кодування — процес, за допомогою якого категоріальні змінні перетворюються у форму, яку можна надати алгоритмам машинного навчання для кращої роботи з прогнозуванням. З цією метою було використано кодування One-Hot Encoding для перетворення даних цих двох стовпців на єдиний числовий масив.

Категоріальне значення представлятиме після цього числове значення запису в наборі даних. Це кодування створить двійковий стовпець для кожної категорії та створить нову матрицю з результатами.

In [49]:
```
from sklearn.preprocessing import OneHotEncoder
```

In [50]:
```
yield_df_onehot = pd.get_dummies(yield_df, columns=['Area',"Item"], prefix = ['Country',"Item"])
features=yield_df_onehot.loc[:, yield_df_onehot.columns != 'hg/ha_yield']
label=yield_df['hg/ha_yield']
features.head()
```

Out[50]:

	Year	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argentina	Country_Armenia
0	1990	1485.0	121.0	16.37	1	0	0	0	0
1	1990	1485.0	121.0	16.37	1	0	0	0	0
2	1990	1485.0	121.0	16.37	1	0	0	0	0
3	1990	1485.0	121.0	16.37	1	0	0	0	0
4	1990	1485.0	121.0	16.37	1	0	0	0	0

5 rows × 115 columns

In [51]:
```
features = features.drop(['Year'], axis=1)
```

In [52]:
```
features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28242 entries, 0 to 28241
Columns: 114 entries, average_rain_fall_mm_per_year to Item_Yams
dtypes: float64(3), uint8(111)
memory usage: 3.9 MB
```

In [53]:
```
features.head()
```

Out[53]:

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argentina	Country_Armenia	Count
0	1485.0	121.0	16.37	1	0	0	0	0	
1	1485.0	121.0	16.37	1	0	0	0	0	
2	1485.0	121.0	16.37	1	0	0	0	0	
3	1485.0	121.0	16.37	1	0	0	0	0	
4	1485.0	121.0	16.37	1	0	0	0	0	

5 rows × 114 columns

Подивившись на набір даних вище, видно, що він містить функції, які сильно відрізняються за величинами, одиницями вимірювання та діапазоном. Щоб позбутися цього, потрібно привести всі функції до одного рівня величин. Цього можна досягти, провівши масштабування даних MinMaxScaler:

In [54]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
features=scaler.fit_transform(features)
```

Після видалення стовпця року на додаток до масштабування всіх значень у об’єктах, отриманий масив виглядатиме так:

```
In [55]: features

Out[55]: array([[4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
                 0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
                [4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
                 0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
                [4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
                 0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
                ...,
                [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
                 0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
                [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
                 1.00000000e+00, 0.00000000e+00, 0.00000000e+00],
                [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
                 0.00000000e+00, 1.00000000e+00, 0.00000000e+00]])
```

Поділ даних на начальну та тестові вибірки

Набір даних буде розділено на два набори даних: навчальний набір і тестовий набір даних. Дані зазвичай мають тенденцію до розподілу нерівності, оскільки навчання моделі зазвичай вимагає якомога більшої кількості даних. Загальні розподіли – 70/30 або 80/20 для навчання/тесту.

Навчальний набір даних – це початковий набір даних, який використовується для навчання алгоритму для навчання та створення правильних прогнозів (70 % набору даних – навчальний набір)

Однак тестовий набір даних використовується для оцінки того, наскільки добре алгоритм ML навчено за допомогою навчального набору даних. Не можна повторно використовувати навчальний набір даних на етапі тестування, оскільки алгоритм ML вже знає очікуваний результат, що перешкоджає меті тестування алгоритму (30 % набору даних є тестовим набором даних):

```
In [56]: from sklearn.model_selection import train_test_split
         train_data, test_data, train_labels, test_labels = train_test_split(features, label, test_size=0.3, random_state=42)

yield_df.to_csv('yield_df.csv')

In [58]: from sklearn.model_selection import train_test_split
         train_data, test_data, train_labels, test_labels = train_test_split(features, label, test_size=0.3, random_state=42)
```

Порівняння та вибір моделей

Перш ніж прийняти рішення про алгоритм для використання, спочатку потрібно оцінити, порівняти та вибрати найкращий, який підходить для цього конкретного набору даних.

Зазвичай, працюючи над проблемою машинного навчання з заданим набором даних, потрібно застосовувати різні моделі та методи, щоб вирішити задачу

оптимізації та підібрати найточнішу модель, яка не буде ані надмірною, ані недоповненою моделлю.

Для цього проекту використано такі моделі:

- Gradient Boosting Regressor;
- Random Forest Regressor;
- SVM;
- Decision Tree Regressor.

In [59]:
```
from sklearn.metrics import r2_score
def compare_models(model):
    model_name = model.__class__.__name__
    fit=model.fit(train_data,train_labels)
    y_pred=fit.predict(test_data)
    r2=r2_score(test_labels,y_pred)
    return([model_name,r2])
```

In [60]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn import svm
from sklearn.tree import DecisionTreeRegressor

models = [
    GradientBoostingRegressor(n_estimators=200, max_depth=3, random_state=0),
     RandomForestRegressor(n_estimators=200, max_depth=3, random_state=0),
    svm.SVR(),
   DecisionTreeRegressor()
]
```

In [61]:
```
model_train=list(map(compare_models,models))
```

In [62]:
```
print(*model_train, sep = "\n")
```

```
['GradientBoostingRegressor', 0.8965731164462923]
['RandomForestRegressor', 0.6842532317855172]
['SVR', -0.20353376480360752]
['DecisionTreeRegressor', 0.9590644980817136]
```

Показник оцінки встановлюється на основі функції оцінки регресії R^2 (коефіцієнт детермінації), яка представлятиме частку дисперсії для елементів (культур) у регресійній моделі. Оцінка R^2 показує, наскільки добре терміни (точки даних) відповідають кривій або лінії.

R^2 — це статистичний показник від 0 до 1, який обчислює, наскільки лінія регресії схожа на дані, до яких вона підігнана. Якщо це 1, модель 100 % передбачає дисперсію даних; якщо це 0, модель не передбачає жодної дисперсії.

З наведених вище результатів Decision Tree Regressor має найвищу оцінку R^2 96 %, GradientBoostingRegressor займає друге місце.

Також потрібно обчислити скоригований R^2, який вказуватиме на те, наскільки добре члени відповідають кривій або лінії, але коригується відповідно до кількості членів у моделі. Якщо додати все більше й більше змінних до моделі, скоригований

коефіцієнт R^2 зменшуватиметься. Якщо додати більше корисних змінних, скоригований R^2 збільшиться. Скоригований R^2 завжди буде меншим або дорівнює R^2.

In [63]:
```
yield_df_onehot = yield_df_onehot.drop(['Year'], axis=1)
```

In [64]:
```
yield_df_onehot.head()
```

Out[64]:

	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argentina	Country_Ar
0	36613	1485.0	121.0	16.37	1	0	0	0	
1	66667	1485.0	121.0	16.37	1	0	0	0	
2	23333	1485.0	121.0	16.37	1	0	0	0	
3	12500	1485.0	121.0	16.37	1	0	0	0	
4	7000	1485.0	121.0	16.37	1	0	0	0	

5 rows × 115 columns

Встановимо тестові дані у стовпці з датафрейму і виключимо значення «hg/ha_yield», де модель ML має передбачати урожайність.

```
test_df=pd.DataFrame(test_data,columns=yield_df_onehot.loc[:, yield_df_onehot.columns != 'hg/ha_yield'].columns)
```

```
cntry=test_df[[col for col in test_df.columns if 'Country' in col]].stack()[test_df[[col for col in test_df.columns if 'Country'
cntrylist=list(pd.DataFrame(cntry).index.get_level_values(1))
countries=[i.split("_")[1] for i in cntrylist]
itm=test_df[[col for col in test_df.columns if 'Item' in col]].stack()[test_df[[col for col in test_df.columns if 'Item' in col]]
itmlist=list(pd.DataFrame(itm).index.get_level_values(1))
items=[i.split("_")[1] for i in itmlist]
```

In [66]:
```
test_df.head()
```

Out[66]:

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argentina	Country_Armenia	Count
0	0.183443	0.110716	0.542078	0.0	0.0	0.0	0.0	0.0	
1	0.458451	0.000413	0.627257	0.0	0.0	0.0	0.0	0.0	
2	0.183443	0.106159	0.518228	0.0	0.0	0.0	0.0	0.0	
3	1.000000	0.224154	0.890971	0.0	0.0	0.0	0.0	0.0	
4	0.458451	0.000355	0.625213	0.0	0.0	0.0	0.0	0.0	

5 rows × 114 columns

In [67]:
```
test_df.drop([col for col in test_df.columns if 'Item' in col],axis=1,inplace=True)
test_df.drop([col for col in test_df.columns if 'Country' in col],axis=1,inplace=True)
test_df.head()
```

Out[67]:

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	0.183443	0.110716	0.542078
1	0.458451	0.000413	0.627257
2	0.183443	0.106159	0.518228
3	1.000000	0.224154	0.890971
4	0.458451	0.000355	0.625213

In [68]:
```
test_df['Country']=countries
test_df['Item']=items
test_df.head()
```

Out[68]:

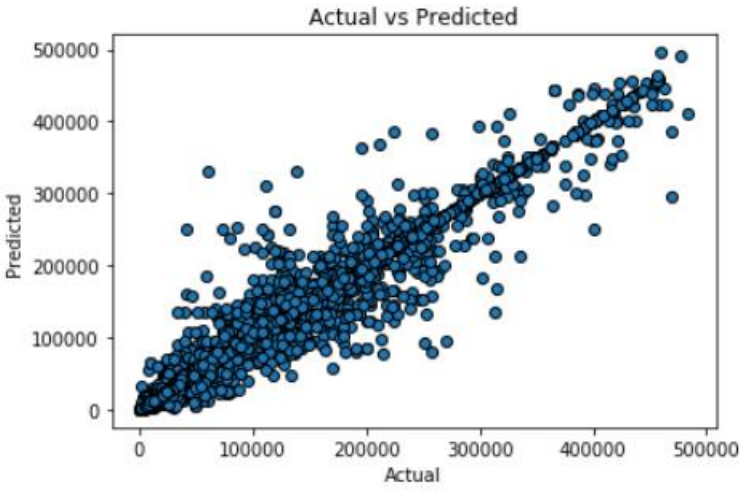
	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country	Item
0	0.183443	0.110716	0.542078	Spain	Rice, paddy
1	0.458451	0.000413	0.627257	Madagascar	Wheat
2	0.183443	0.106159	0.518228	Spain	Sorghum
3	1.000000	0.224154	0.890971	Colombia	Potatoes
4	0.458451	0.000355	0.625213	Madagascar	Sweet potatoes

```
In [71]: clf=DecisionTreeRegressor()
         model=clf.fit(train_data,train_labels)

         test_df["yield_predicted"]= model.predict(test_data)
         test_df["yield_actual"]=pd.DataFrame(test_labels)["hg/ha_yield"].tolist()
         test_group=test_df.groupby("Item")
         test_group.apply(lambda x: r2_score(x.yield_actual,x.yield_predicted))
```

```
Out[71]: Item
         Cassava                0.926622
         Maize                  0.894963
         Plantains and others   0.817372
         Potatoes               0.910895
         Rice, paddy            0.896925
         Sorghum                0.802025
         Soybeans               0.847838
         Sweet potatoes         0.840751
         Wheat                  0.922150
         Yams                   0.928241
         dtype: float64
```

Надалі проводять порівняння фактичних значень моделі з прогнозованими:

```
fig, ax = plt.subplots()

ax.scatter(test_df["yield_actual"], test_df["yield_predicted"],edgecolors=(0, 0, 0))

ax.set_xlabel('Actual')
ax.set_ylabel('Predicted')
ax.set_title("Actual vs Predicted")
plt.show()
```



На отриманому результаті показано відповідність передбачень у вигляді лінії. Можна побачити, що оцінка R^2 відмінна. Це означає, що отримано приємливу модель для прогнозування врожайності сільськогосподарських культур для певної країни. Додавання додаткових функцій, наприклад кліматичних даних; вітер і забруднення, економічна ситуація в даній країні тощо, ймовірно, можуть покращити прогнози моделі.

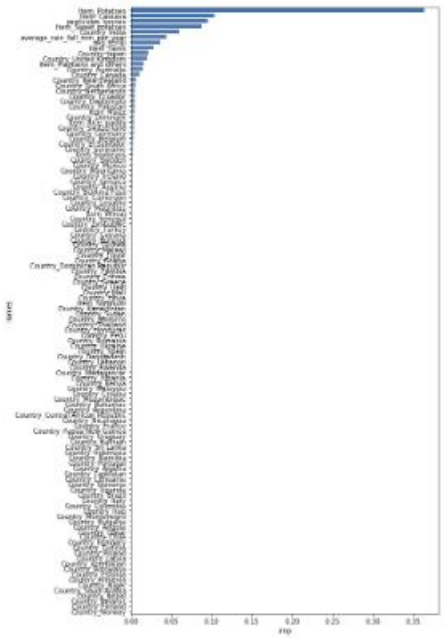
In [86]:
```
def adjusted_r_squared(y,yhat,x):
    score=1- (((1-(r2_score(y,yhat)))*(len(y)-1))/(len(y)-x.shape[1]-2))
    return score

test_group.apply(lambda x: adjusted_r_squared(x.yield_actual,x.yield_predicted,x))
```

Out[86]:
```
Item
Cassava                 0.925669
Maize                   0.894284
Plantains and others    0.807946
Potatoes                0.910346
Rice, paddy             0.896068
Sorghum                 0.800251
Soybeans                0.846604
Sweet potatoes          0.839268
Wheat                   0.921607
Yams                    0.925723
dtype: float64
```

4.3. Результати роботи моделі

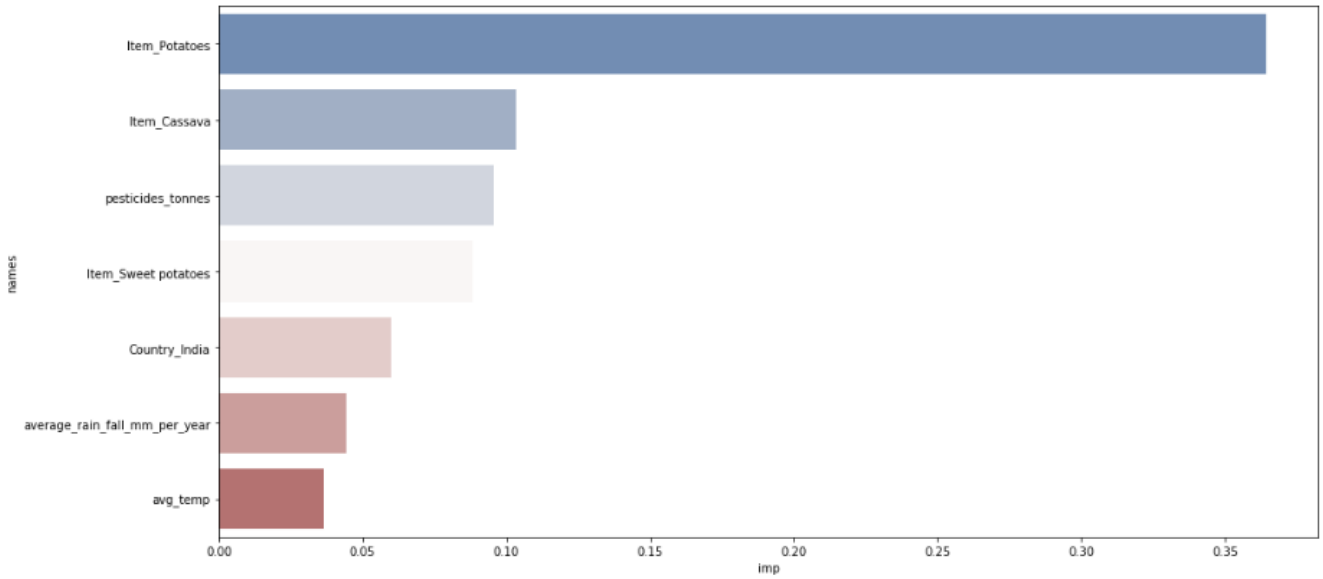
In [87]:
```
varimp= {'imp':model.feature_importances_,'names':yield_df_onehot.columns[yield_df_onehot.columns!="hg/ha_yield"]}
```

In [88]:
```
a4_dims = (8.27,16.7)

fig, ax = plt.subplots(figsize=a4_dims)
df=pd.DataFrame.from_dict(varimp)
df.sort_values(ascending=False,by=["imp"],inplace=True)
df=df.dropna()
sns.barplot(x="imp",y="names",palette="vlag",data=df,orient="h",ax=ax);
```



Найважливіші фактори, що впливають на посіви:

```
a4_dims = (16.7, 8.27)

fig, ax = plt.subplots(figsize=a4_dims)
df=pd.DataFrame.from_dict(varimp)
df.sort_values(ascending=False,by=["imp"],inplace=True)
df=df.dropna()
df=df.nlargest(7, 'imp')
sns.barplot(x="imp",y="names",palette="vlag",data=df,orient="h",ax=ax);
```

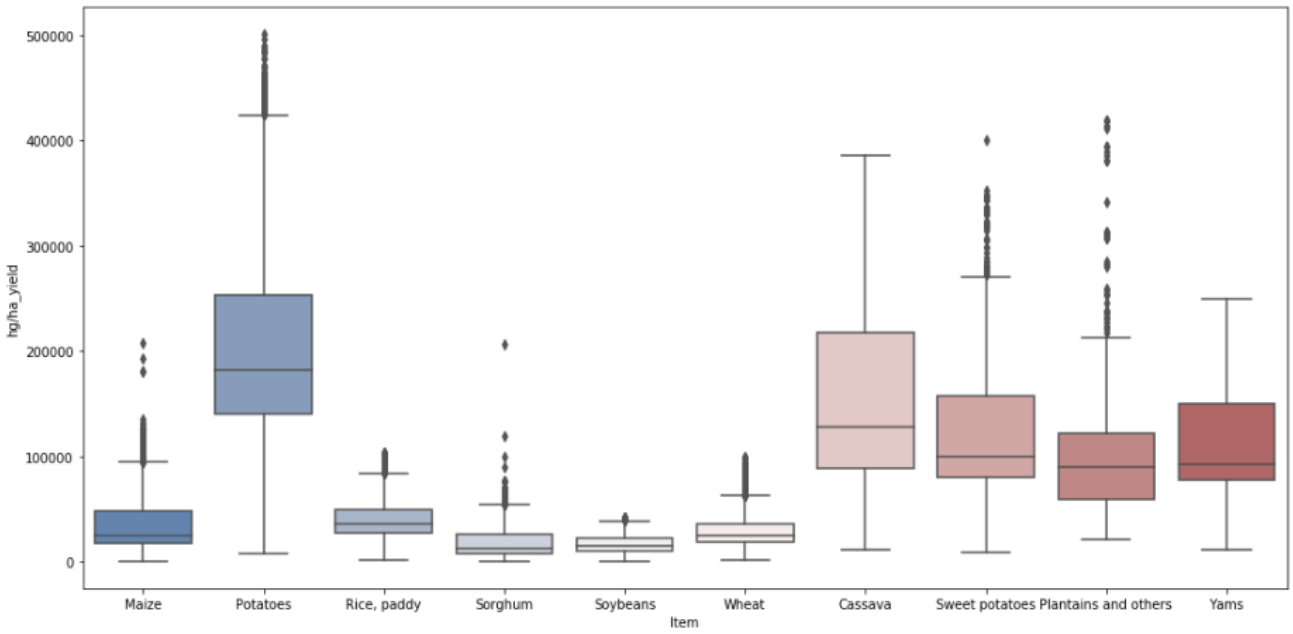


Культура картопля має найбільше значення для прийняття рішень для моделі, де це найвищі культури в наборі даних. Маніока також, тоді, як і очікувалося, ми бачимо вплив пестицидів, де це третя найважливіша характеристика, а потім, якщо культурою є солодка картопля, ми бачимо одні з найвищих культур за важливістю характеристик у наборі даних.

Якщо культуру вирощують в Індії, це має сенс, оскільки Індія має найбільшу кількість посівів у наборі даних. Потім йдуть опади і температура. Перше припущення щодо цих характеристик було правильним, оскільки всі вони суттєво впливають на очікувану врожайність сільськогосподарських культур у моделі.

Побудуємо графік Boxplot, який показуватиме врожайність для кожної культури:

```
a4_dims = (16.7, 8.27)

fig, ax = plt.subplots(figsize=a4_dims)
sns.boxplot(x="Item",y="hg/ha_yield",palette="vlag",data=yield_df,ax=ax);
```



4.4. Розроблення графічного інтерфейсу інформаційної системи

Графічний інтерфейс для інформаційної системи прогнозування урожайності спроектовано з допомогою графічної бібліотеки PySimpleGUI. В даній роботі розроблено графічний інтерфейс для системи машинного навчання, яка може спрогнозувати кращий урожай за даних умов. Ця інформаційна система дасть прогноз, яку найкраще культуру вирощувати на даному полі. Інтерфейс даної системи представлено на рис. 4.1.

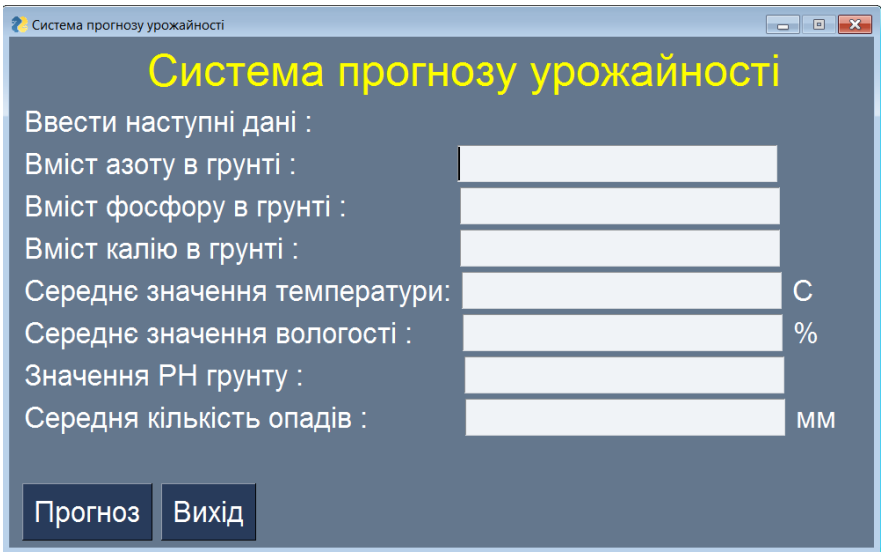


Рис. 4.1. Інтерфейс інформаційної системи прогнозування урожайності.

В наступні текстові поля вводять дані по вмісту азоту, фосфору та калію в грунті. Нижче потрібно ввести значення теммператури, вологості, середньої кількості опадів в

даній місцевості, а також значення PH грунту. Після цього слід натиснути кнопку Прогноз.

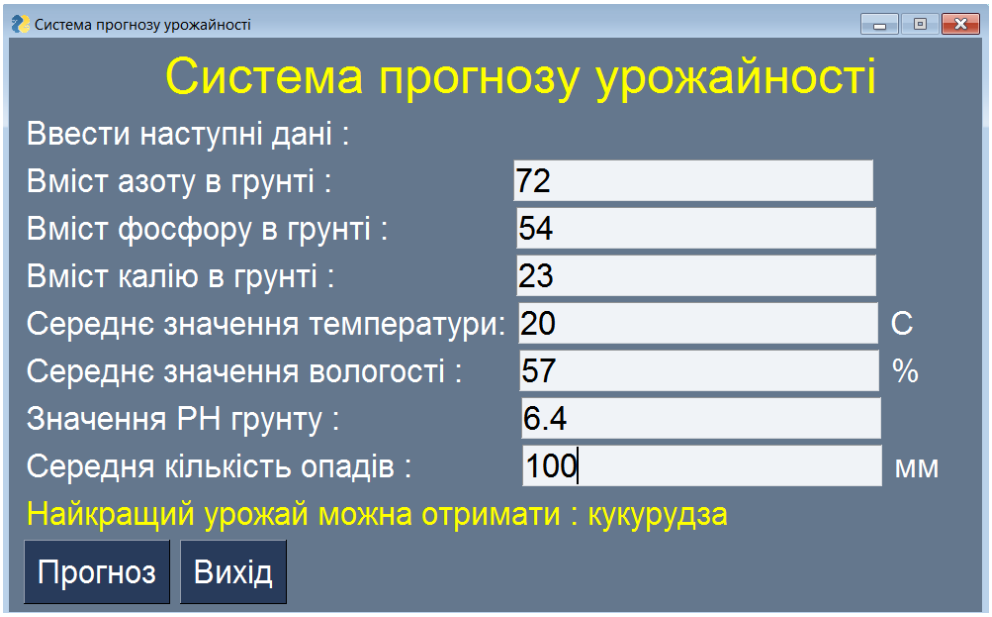


Рис. 4.2. Результати роботи інформаційної системи.

Через деякий час появиться поруч повідомлення: найкращий урожай за даних умов можна отримати, посадивши кукурудзу.

В датасеті знаходяться дані про 22 культури, які можна вирощувати на полях, 7 ознак (вхідні параметри моделі) та одна мітка (вихідний параметр – вид культури). Загалом в даному датасеті є 2200 записів. В цій моделі машинного навчання використовується метод K-Nearest Neighbors (KNN) для прогнозування найкращого урожаю.

Для початку імпортують бібліотеки, які необхідні для подальшої роботи: це Numpy, Pandas, Scikit-Learn, PySimpleGUI. Після цього імпортують датасет з даними про урожай crop. Він зберігається у файлі Excel. Ці дані будуть використані для навчання моделі. Дані, які були імпортовані, містять два типи значень. Значення всіх об’єктів представлені в числовій формі, але назви культур представлені в текстовій формі. В машинному навчанні, коли потрібно провести навчання моделі, потрібно всі значення представити в числовій формі. Для цього назви культур переводять в числову форму. Це робиться з допомогою кодування міток. Це важливий етап попередньої обробки набору даних.

ВИСНОВКИ ДО РОЗДІЛУ 4

Розроблено та реалізовано інформаційну-аналітичну систему для прогнозування врожайності сільськогосподарських культур. Реалізовано програмну модель прогнозування, в рамках даної моделі проведено прогноз динаміки урожайності культур.

Проаналізовано широкий набір факторів, що впливають на вегетацію сільськогосподарської культури, а отже і на врожайність. На їх основі побудовано регресійні моделі, які допомогли оцінити найбільш суттєві ознаки для прогнозування. Дану інформаційно-аналітичну систему можна застосувати для прогнозування врожайності сільськогосподарських культур.

Проведено розрахунок кореляції між різними ознаками та цільовим значенням урожайності на поле з використанням коефіцієнта кореляції Пірсона. Проведено навчання різних моделей регресії для прогнозування врожайності на основі даних про поля та погодних особливостей.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

5.1. Опис проекту інформаційної системи

Перш ніж почати просувати свій проект як стартап, потрібно зробити кілька кроків. Перший – створити інформаційну карту проекту. На цій карті показано параметри проекту та орієнтовний бюджет, який потрібно виділити на нього. Сюди входять зарплати розробника, які розраховані на певні цикли розробки, та оренда обладнання. Інформаційна карта наведена в таблиці. 5.1.

Табл. 5.1. Карта інформаційно-аналітичної системи

Назва номінації	Python програма, модель машинного навчання
Назва проекту	Інформаційна система моделювання процесу росту та прогнозування врожайності сільськогосподарських культур
Назва ВНЗ, факультету, спеціальності	НЛТУ, кафедра інформаційних технологій, 122 «Комп'ютерні науки»
Прізвище, ім'я, по-батькові	Поліщук Софія Ігорівна
Цілі і задачі проекту	Мета проекту – розробка та реалізація інформаційно-аналітичної системи моделювання процесу росту та прогнозування врожайності сільськогосподарських культур. Задачі проекту: • провести огляд літератури по даній тематиці, проаналізувати існуючі системи росту та прогнозування урожайності сільськогосподарських культур; • розробити математичну модель росту рослин; • реалізувати програмну модель для прогнозування динаміки урожайності

	сільськогосподарських культур; • в рамках програмної моделі провести дослідження динаміки росту рослин та прогнозування їх урожайності.
Короткий зміст проекту	Розглянуто основні фактори, що впливають на формування врожаю, кореляційний зв'язок між факторними ознаками (сонячна активність, опади, добрива) та врожайністю культур, параметри відбору факторних ознак для побудови регресійних моделей урожайності. Засобами Python та його бібліотек для аналізу даних Scikit-learn, Pandas та візуалізації результатів досліджень Matplotlib отримано інформаційно-аналітичну систему, з допомогою якої можна моделювати процеси росту рослин та прогнозувати майбутню врожайність цих культур. Отримано показники для оцінки адекватності екстраполяційних моделей урожайності сільськогосподарських культур та оцінки на їх основі точності прогнозів.
Терміни виконання проекту	12 місяців
Бюджет проекту	200 000 грн.

5.2. Стратегія проекту

У загальному випадку успішний стартап зі штучного інтелекту або машинного навчання концентрує зусилля на одному або кількох завданнях зі списку:

- зменшення необхідного обсягу людської праці або звільнення від нього в галузях, які раніше вважалися складними для автоматизації;

- використання «вільного місця», яке утворилося завдяки появі нових можливостей (мова про нові продукти або послуги, раніше надто дорогі або неможливі);
- збільшення цінності традиційних додатків завдяки впровадженню техніки машинного навчання в них.

Є причини, через які платформи машинного навчання існують окремо. Такі гіганти, як Google і Facebook, інвестують чималі суми в їх розвиток та широко використовують принципи відкритого коду під час роботи над ними. У таких умовах конкуренція з найбільшими світовими компаніями, які до того ж мають унікальні пропрієтарні набори даних, стає вкрай скрутною, якщо не неможливою.

Крім цього, недолік фахівців з обробки даних призводить до того, що клієнти часто неспроможні достатньо використовувати всі переваги великих платформ і алгоритмів. В результаті стартап з горизонтальною платформою може продавати лише професійні послуги, допомагаючи кожному клієнту визначати та досягати специфічних цілей.

Нарешті, перед розробкою IT-платформи варто подумати про складність виходу на ринок. Поведінка покупців у різних галузях відрізняється, потрібно враховувати і наявність різних каналів просування. Все це ускладнює роботу з просування комплексного продукту.

Якщо модель машинного навчання можна застосувати в різних індустріях, перед вибором спеціалізації варто врахувати наступні змінні.

- вартість впровадження;
- додаткова цінність.

Чим ще цінне рішення на основі машинного навчання, окрім заміни людини? Підвищення якості та задоволеності клієнтів, менша кількість помилок.

Скорочення частки людської праці - основний ефект застосування штучного інтелекту, що закономірно викликає серйозну протидію. Чи люди втратять роботу в результаті впровадження вашої технології? Так, одне з ключових побоювань серед IT-компаній - скорочення часу на налаштування ПЗ через автоматизацію.

Отже, щоб отримати максимум із розробок у галузі штучного інтелекту та машинного навчання, потрібно:

- уникати областей, у яких великі компанії мають структурні переваги;

- вибрати плацдарм, у якому новий продукт вирішує важливу проблему, а покупці не схильні до опору;
- вибрати цільову галузь, ґрунтуючись на ступені її готовності до впровадження ІТ та машинного навчання та відсутності великих законодавчих проблем у адаптації технології.

5.3. Розробка програми стартап проекту

Табл. 5.2. Основні переваги та концепції інформаційної системи

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами
1	моделювання показників урожайності	створення власних функціональних одиниць	гнучкість та свобода для кінцевого споживача
2	визначення параметрів росту сільсько-господарських культур	можливість підключення датчиків росту рослин	можливість використання даних дистанційного зондування Землі
3	зручний інтерфейс	інтерфейс, який містить потрібні дані для можливості прогнозування урожайності рослин	можливість прогнозування урожайності в майбутньому

Табл. 5.3. Опис рівнів інформаційно-аналітичної системи урожайності сільськогосподарських рослин

рівні товару	**сутність та її складові**
1. Програмний продукт за задумом	моделювання для прогнозування врожайності сільськогосподарських рослин
2. Програмний продукт, який має бути реально виконаний	програмний продукт для моделювання росту рослин та прогнозування їх урожайності

3. Підкріплення	служба для підтримки системи прийняття рішень за допомогою цієї інформаційно-аналітичної системи

Табл. 5.4. Визначення меж для встановлення ціни на інформаційно-аналітичну систему

№ п/п	рівень цін на товари замінники	рівень цін на товари-аналоги	рівень доходів цільової групи споживачів	верхня та нижня межі встановлення ціни на товар/послугу
1	наперед не задано	наперед не задано	200$+	300/150 $

ВИСНОВКИ ДО РОЗДІЛУ 5

Розроблено та реалізовано інформаційно-аналітичну систему для моделювання росту сількогосподарських культур та прогнозування їх врожайності, в якій враховуються погодні умови, стан грунту, агрономічні фактори. Ця інформаційна система може бути реалізована на практиці. Її перевагою є те, що альтернативних програмних продуктів немає, а якщо є, то дуже мало. При використанні реклами та оголошень для просування програмного продукту можна досягти успіху та отримати пристойний дохід.

В п’ятому розділі розроблено стартап даного проекту. Ця інформаційна система може бути реалізована на практиці. Її перевагою є те, що альтернативних програмних продуктів немає, а якщо є, то дуже мало. При використанні реклами та оголошень для просування даного програмного продукту можна досягти успіху та отримати пристойний дохід.

СПИСОК ЛІТЕРАТУРИ

1. Маккинни У. Python и анализ данных. – М.: ДМК Пресс, 2015. – 482 с.

2. Элбон Крис. Машинное обучение с использованием Python. Сборник рецептов. – СПб: БХВ-Петербург, 2019. – 384 с.:

3. Мартин Освальдо. Байесовский анализ на Python. – М.: ДМК Пресс, 2020. – 341 с.

4. Хилл Кристиан. Научное программирование на Python. – М.: ДМК Пресс, 2021. – 647 с.

5. Грас Джоэл. Data Science. Наука о данных с нуля. – СПб: БХВ-Петербург, 2017. – 336 с.

6. Рашка Себастьян. Python и машинное обучение. – М.: ДМК Пресс, 2017. – 420 с.

7. Коэльо Л. П., Ричарт В. Построение систем машинного обучения на языке Python. – М.: ДМК Пресс, 2016. – 302 с.

8. Силен Д., Мейсман А., Али М. Основы Data Science и Big Data. Python и наука о данных. – СПб.: Питер, 2017. – 336 с.

9. Харрисон Мэтт. Машинное обучение: Карманный справочник. Краткое руководство по методам структурированного машинного обучения на Python. – СПб.: Диалектика, 2020. – 320 с.

10. Абдрахманов М. И. Pandas: Работа с данными. – Devpractice Team, 2020. – 170 с.

11. Полевой А. Н. Прикладное моделирование и прогнозирование продуктивности посевов: – М: Гидрометеоиздат. 1988.

12. Уланова Е. С., Забелин В. Н. Методы корреляционного и регрессионного анализа в агрометеорологии. – Л.: Гидрометеоиздат, 1990.

13. Агеев В. В., Есаулко А. Н. Горбатко Л. С. и др. Математико-нормативное обеспечение программирования урожая. Ставрополь: Ставропольский государственный аграрный университет, 2004. – 182 с.

14. Можаев Н. И., Серикпаев П. А., Стыбаев Г. Ж. Программирование урожаев сельскохозяйственных культур. Учебное пособие. – Астана: Фолиант, 2013. – 160 с.

15. Каюмов М. К. Программирование продуктивности полевых культур. Справочник. – М.: Росагропромиздат, 1989. – 368 с.

16. Шахова О. А., Якубышина Л. И. Программирование урожая сельскохозяйственных культур. Учебное пособие. – Тюмень: Титул, 2018. – 96 с.

17. Загайтов И. Б. Прогноз колебаний природных условий сельскохозяйственного производства и всемирная статистика урожаев. – Воронеж: ВГАУ, 1998.

18. Личко К. П., Абельдяев Н. Ф. Прогнозирование урожайности сельскохозяйственных культур (экстраполяционные приемы). – М.: ТСХА, 1988.

19. Полевой А. Н. Методическое пособие по разработке динамико-статистических методов прогнозирования урожайности сельскохозяйственных культур. – М.: Гидрометеоиздат. 1981.

20. Полевой А. Н. Теория и расчет продуктивности сельскохозяйственных культур. – М.: Гидрометеоиздат. 1983.

ВИСНОВКИ

Засобами Python та його бібліотек для аналізу даних Scikit-Learn, Pandas та візуалізації результатів досліджень Matplotlib отримано інформаційну систему, з допомогою якої можна моделювати процеси росту рослин та прогнозувати майбутню врожайність цих культур. Отримано показники для оцінки адекватності екстраполяційних моделей урожайності сільськогосподарських культур та оцінки на їх основі точності прогнозів.

В першому розділі проаналізовано предметну область, засоби та технології проектування інформаційно-аналітичної системи для дослідження росту та прогнозування урожайності сільськогосподарських культур. З допомогою цієї інформаційно-аналітичної системи можна буде моделювати динаміку росту рослин та прогнозувати майбутню урожайність.

У другому розділі приведено відомості про часові ряди для прогнозування урожайності. Описано принципи їх функціонування, технології їх розробки. Розглянуто засоби, з допомогою яких проектується дана інформаційна система. Приведено відомості про особливості проектування програмного продукту засобами Python, проектування графічного інтерфейсу засобами бібліотеки PyGui, особливості візуалізації результатів з допомогою бібліотеки Matplotlib.

В третьому розділі використано як математичну модель модель росту рослин SAFY, яка розраховувалася по даних космічних зйомок, метеоданих та довідкових параметрах, вона використовується для прогнозу урожайності сільськогосподарських культур за 1-1,5 місяці до їх дозрівання.

В цій моделі можна використовувати дані дистанційного зондування та метеорологічних умов для визначення фітомаси та урожайності без врахування статистичних даних за попередні роки. Прогнозування урожайності стає можливим до закінчення стадії активної вегетації культур, тобто за один-півтора місяці до фактичного збору урожаю.

В четвертому розділі розроблено та реалізовано інформаційну-аналітичну систему для прогнозування врожайності сільськогосподарських культур. Реалізовано програмну модель прогнозування, в рамках даної моделі проведено прогноз урожайності культур. Проаналізовано широкий набір факторів, що впливають на

вегетацію сільськогосподарських культур та на врожайність. На їх основі побудовано регресійні моделі, які допомогли оцінити найбільш суттєві ознаки для прогнозування. Проведено розрахунок кореляції між різними ознаками та цільовим значенням урожайності на поле. Проведено навчання різних моделей регресії для прогнозування врожайності на основі даних про поля та погодних особливостей.

ДОДАТКИ

ДОДАТОК А

1.ipynb

```
[ ] import numpy as np
    import pandas as pd
```

```
[ ] df_yield = pd.read_csv('yield.csv')
    df_yield.shape
```

```
(56717, 12)
```

```
[ ] df_yield.head()
```

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
0	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1961	1961	hg/ha	14000
1	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1962	1962	hg/ha	14000
2	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1963	1963	hg/ha	14260
3	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1964	1964	hg/ha	14257
4	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1965	1965	hg/ha	14400

```
df_yield.tail()
```

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	Value
56712	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2012	2012	hg/ha	24420
56713	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2013	2013	hg/ha	22888
56714	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2014	2014	hg/ha	21357
56715	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2015	2015	hg/ha	19826
56716	QC	Crops	181	Zimbabwe	5419	Yield	15	Wheat	2016	2016	hg/ha	18294

```
df_yield = df_yield.rename(index=str, columns={"Value": "hg/ha_yield"})
df_yield.head()
```

	Domain Code	Domain	Area Code	Area	Element Code	Element	Item Code	Item	Year Code	Year	Unit	hg/ha_yield
0	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1961	1961	hg/ha	14000
1	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1962	1962	hg/ha	14000
2	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1963	1963	hg/ha	14260
3	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1964	1964	hg/ha	14257
4	QC	Crops	2	Afghanistan	5419	Yield	56	Maize	1965	1965	hg/ha	14400

```
[ ] df_yield = df_yield.drop(['Year Code','Element Code','Element','Year Code','Area Code','Domain Code','Domain','Unit','Item Co
    df_yield.head()
```

	Area	Item	Year	hg/ha_yield
0	Afghanistan	Maize	1961	14000
1	Afghanistan	Maize	1962	14000
2	Afghanistan	Maize	1963	14260
3	Afghanistan	Maize	1964	14257
4	Afghanistan	Maize	1965	14400

```
df_yield.describe()
```

	Year	hg/ha_yield
count	56717.000000	56717.000000
mean	1989.669570	62094.660084
std	16.133198	67835.932856
min	1961.000000	0.000000
25%	1976.000000	15680.000000
50%	1991.000000	36744.000000
75%	2004.000000	86213.000000
max	2016.000000	1000000.000000

```
[ ] df_yield.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 56717 entries, 0 to 56716
Data columns (total 4 columns):
Area           56717 non-null object
Item           56717 non-null object
Year           56717 non-null int64
hg/ha_yield    56717 non-null int64
dtypes: int64(2), object(2)
memory usage: 2.2+ MB
```

```
[ ] df_rain = pd.read_csv('rainfall.csv')
    df_rain.head()
```

	Area	Year	average_rain_fall_mm_per_year
0	Afghanistan	1985	327
1	Afghanistan	1986	327
2	Afghanistan	1987	327
3	Afghanistan	1989	327
4	Afghanistan	1990	327

```
[ ] df_rain = df_rain.rename(index=str, columns={" Area": 'Area'})
```

```
[ ] df_rain.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6727 entries, 0 to 6726
Data columns (total 3 columns):
Area                            6727 non-null object
Year                            6727 non-null int64
average_rain_fall_mm_per_year    5953 non-null object
dtypes: int64(1), object(2)
memory usage: 210.2+ KB
```

```
[ ] df_rain['average_rain_fall_mm_per_year'] = df_rain['average_rain_fall_mm_per_year'].apply(pd.to_numeric, downcast='float', er
    df_rain.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6727 entries, 0 to 6726
Data columns (total 3 columns):
Area                            6727 non-null object
Year                            6727 non-null int64
average_rain_fall_mm_per_year    5947 non-null float64
dtypes: float64(1), int64(1), object(1)
memory usage: 210.2+ KB
```

```
[ ] df_rain = df_rain.dropna()
```

```
[ ] df_rain.describe()
```

	Year	average_rain_fall_mm_per_year
count	5947.000000	5947.000000
mean	2001.365899	1124.743232
std	9.526335	786.257365
min	1985.000000	51.000000
25%	1993.000000	534.000000
50%	2001.000000	1010.000000
75%	2010.000000	1651.000000
max	2017.000000	3240.000000

```
[ ] yield_df = pd.merge(df_yield, df_rain, on=['Year','Area'])
```

```
[ ] yield_df.shape
```

```
(25385, 5)
```

```
[ ] yield_df.head()
```

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year
0	Afghanistan	Maize	1985	16652	327.0
1	Afghanistan	Potatoes	1985	140909	327.0
2	Afghanistan	Rice, paddy	1985	22482	327.0
3	Afghanistan	Wheat	1985	12277	327.0
4	Afghanistan	Maize	1986	16875	327.0

```
[ ] yield_df.describe()
```

	Year	hg/ha_yield	average_rain_fall_mm_per_year
count	25385.000000	25385.000000	25385.000000
mean	2001.278787	68312.278353	1254.849754
std	9.143915	75213.292733	804.449430
min	1985.000000	50.000000	51.000000
25%	1994.000000	17432.000000	630.000000
50%	2001.000000	38750.000000	1150.000000
75%	2009.000000	94286.000000	1761.000000
max	2016.000000	554855.000000	3240.000000

```
[ ] df_pes = pd.read_csv('pesticides.csv')
    df_pes.head()
```

	Domain	Area	Element	Item	Year	Unit	Value
0	Pesticides Use	Albania	Use	Pesticides (total)	1990	tonnes of active ingredients	121.0
1	Pesticides Use	Albania	Use	Pesticides (total)	1991	tonnes of active ingredients	121.0
2	Pesticides Use	Albania	Use	Pesticides (total)	1992	tonnes of active ingredients	121.0
3	Pesticides Use	Albania	Use	Pesticides (total)	1993	tonnes of active ingredients	121.0
4	Pesticides Use	Albania	Use	Pesticides (total)	1994	tonnes of active ingredients	201.0

```
[ ] df_pes = df_pes.rename(index=str, columns={"Value": "pesticides_tonnes"})
    df_pes = df_pes.drop(['Element','Domain','Unit','Item'], axis=1)
    df_pes.head()
```

	Area	Year	pesticides_tonnes
0	Albania	1990	121.0
1	Albania	1991	121.0
2	Albania	1992	121.0
3	Albania	1993	121.0
4	Albania	1994	201.0

```
[ ] df_pes.describe()
```

	Year	pesticides_tonnes
count	4349.000000	4.349000e+03
mean	2003.138883	2.030334e+04
std	7.728044	1.177362e+05
min	1990.000000	0.000000e+00
25%	1996.000000	9.300000e+01
50%	2003.000000	1.137560e+03
75%	2010.000000	7.869000e+03
max	2016.000000	1.807000e+06

```
[ ] df_pes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4349 entries, 0 to 4348
Data columns (total 3 columns):
Area                 4349 non-null object
Year                 4349 non-null int64
pesticides_tonnes    4349 non-null float64
dtypes: float64(1), int64(1), object(1)
memory usage: 135.9+ KB
```

```
[ ] yield_df = pd.merge(yield_df, df_pes, on=['Year','Area'])
    yield_df.shape
```

```
(18949, 6)
```

```
[ ] yield_df.head()
```

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes
0	Albania	Maize	1990	36613	1485.0	121.0
1	Albania	Potatoes	1990	66667	1485.0	121.0
2	Albania	Rice, paddy	1990	23333	1485.0	121.0
3	Albania	Sorghum	1990	12500	1485.0	121.0
4	Albania	Soybeans	1990	7000	1485.0	121.0

```
[ ] avg_temp=  pd.read_csv('temp.csv')
```

```
[ ] avg_temp.head()
```

	year	country	avg_temp
0	1849	Côte D'Ivoire	25.58
1	1850	Côte D'Ivoire	25.52
2	1851	Côte D'Ivoire	25.67
3	1852	Côte D'Ivoire	NaN
4	1853	Côte D'Ivoire	NaN

```
[ ] avg_temp.describe()
```

	year	avg_temp
count	71311.000000	68764.000000
mean	1905.799007	16.183876
std	67.102099	7.592960
min	1743.000000	-14.350000
25%	1858.000000	9.750000
50%	1910.000000	16.140000
75%	1962.000000	23.762500
max	2013.000000	30.730000

```
[ ] avg_temp = avg_temp.rename(index=str, columns={"year": "Year", "country":'Area'})
    avg_temp.head()
```

	Year	Area	avg_temp
0	1849	Côte D'Ivoire	25.58
1	1850	Côte D'Ivoire	25.52
2	1851	Côte D'Ivoire	25.67
3	1852	Côte D'Ivoire	NaN
4	1853	Côte D'Ivoire	NaN

```
[ ] yield_df = pd.merge(yield_df,avg_temp, on=['Area','Year'])
    yield_df.head()
```

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	Albania	Maize	1990	36613	1485.0	121.0	16.37
1	Albania	Potatoes	1990	66667	1485.0	121.0	16.37
2	Albania	Rice, paddy	1990	23333	1485.0	121.0	16.37
3	Albania	Sorghum	1990	12500	1485.0	121.0	16.37
4	Albania	Soybeans	1990	7000	1485.0	121.0	16.37

```
[ ] yield_df.shape
```

```
(28242, 7)
```

```
[ ] yield_df.describe()
```

	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
count	28242.000000	28242.000000	28242.00000	28242.000000	28242.000000
mean	2001.544296	77053.332094	1149.05598	37076.909344	20.542627
std	7.051905	84956.612897	709.81215	59958.784665	6.312051
min	1990.000000	50.000000	51.00000	0.040000	1.300000
25%	1995.000000	19919.250000	593.00000	1702.000000	16.702500
50%	2001.000000	38295.000000	1083.00000	17529.440000	21.510000
75%	2008.000000	104676.750000	1668.00000	48687.880000	26.000000
max	2013.000000	501412.000000	3240.00000	367778.000000	30.650000

```
[ ] yield_df.isnull().sum()
```

```
Area                            0
Item                            0
Year                            0
hg/ha_yield                     0
average_rain_fall_mm_per_year   0
pesticides_tonnes               0
avg_temp                        0
dtype: int64
```

```
[ ] yield_df.groupby('Item').count()
```

Item	Area	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
Cassava	2045	2045	2045	2045	2045	2045
Maize	4121	4121	4121	4121	4121	4121
Plantains and others	556	556	556	556	556	556
Potatoes	4276	4276	4276	4276	4276	4276
Rice, paddy	3388	3388	3388	3388	3388	3388
Sorghum	3039	3039	3039	3039	3039	3039

```
[ ] yield_df.describe()
```

	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
count	28242.000000	28242.000000	28242.00000	28242.000000	28242.000000
mean	2001.544296	77053.332094	1149.05598	37076.909344	20.542627
std	7.051905	84956.612897	709.81215	59958.784665	6.312051
min	1990.000000	50.000000	51.00000	0.040000	1.300000
25%	1995.000000	19919.250000	593.00000	1702.000000	16.702500
50%	2001.000000	38295.000000	1083.00000	17529.440000	21.510000
75%	2008.000000	104676.750000	1668.00000	48687.880000	26.000000
max	2013.000000	501412.000000	3240.00000	367778.000000	30.650000

```
[ ] yield_df['Area'].nunique()
```

```
101
```

```
[ ] yield_df.groupby(['Area'],sort=True)['hg/ha_yield'].sum().nlargest(20)
```

```
Area
India                327420324
Brazil               167550306
Mexico               130788528
Japan                124470912
Australia            109111062
Pakistan              73897434
Indonesia             69193506
United Kingdom        55419990
Turkey                52263950
Spain                 46773540
South Africa          41333132
Germany               38780463
Egypt                 36828848
Canada                34706922
```

```
[ ] yield_df.groupby(['Item','Area'],sort=True)['hg/ha_yield'].sum().nlargest(20)
```

```
Item            Area
Cassava         India             142810624
Potatoes        India              92122514
                Brazil             49602168
                United Kingdom     46705145
                Australia          45670386
Sweet potatoes  India              44439538
Potatoes        Japan              42918726
                Mexico             42053880
Sweet potatoes  Mexico             35808592
                Australia          35550294
Cassava         Brazil             33671231
Potatoes        Pakistan           32969754
Sweet potatoes  Japan              32794236
Potatoes        Turkey             30530955
Yams            Japan              29165394
Sweet potatoes  Brazil             28266502
Potatoes        South Africa       27341980
```

```
[ ] import sklearn
    import seaborn as sns
    import matplotlib.pyplot as plt
```

```
[ ] corr_data=yield_df.select_dtypes(include=[np.number]).corr()

    mask = np.zeros_like(corr_data, dtype=np.bool)
    mask[np.triu_indices_from(mask)] = True

    f, ax = plt.subplots(figsize=(11, 9))

    # Generate a custom diverging colormap
    cmap = sns.light_palette("navy")

    # Draw the heatmap with the mask and correct aspect ratio
    sns.heatmap(corr_data, mask=mask, cmap=cmap, vmax=.3, center=0,
                square=True, linewidths=.5, cbar_kws={"shrink": .5});
```

```
[ ] yield_df.head()
```

	Area	Item	Year	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	Albania	Maize	1990	36613	1485.0	121.0	16.37
1	Albania	Potatoes	1990	66667	1485.0	121.0	16.37
2	Albania	Rice, paddy	1990	23333	1485.0	121.0	16.37
3	Albania	Sorghum	1990	12500	1485.0	121.0	16.37
4	Albania	Soybeans	1990	7000	1485.0	121.0	16.37

```
[ ] yield_df_onehot = pd.get_dummies(yield_df, columns=['Area',"Item"], prefix = ['Country',"Item"])
    features=yield_df_onehot.loc[:, yield_df_onehot.columns != 'hg/ha_yield']
    label=yield_df['hg/ha_yield']
    features.head()
```

	Year	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_
0	1990	1485.0	121.0	16.37	1	0	0	
1	1990	1485.0	121.0	16.37	1	0	0	
2	1990	1485.0	121.0	16.37	1	0	0	
3	1990	1485.0	121.0	16.37	1	0	0	
4	1990	1485.0	121.0	16.37	1	0	0	

5 rows × 115 columns

```
[ ] features = features.drop(['Year'], axis=1)
```

```
[ ] features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28242 entries, 0 to 28241
Columns: 114 entries, average_rain_fall_mm_per_year to Item_Yams
dtypes: float64(3), uint8(111)
memory usage: 3.9 MB
```

```
[ ] features.head()
```

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argent
0	1485.0	121.0	16.37	1	0	0	
1	1485.0	121.0	16.37	1	0	0	
2	1485.0	121.0	16.37	1	0	0	
3	1485.0	121.0	16.37	1	0	0	
4	1485.0	121.0	16.37	1	0	0	

5 rows × 114 columns

```
[ ] from sklearn.preprocessing import MinMaxScaler
    scaler=MinMaxScaler()
    features=scaler.fit_transform(features)
```

```
[ ] features
```

```
array([[4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [4.49670743e-01, 3.28894097e-04, 5.13458262e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       ...,
       [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
        1.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [1.90028222e-01, 6.93361288e-03, 6.28960818e-01, ...,
        0.00000000e+00, 1.00000000e+00, 0.00000000e+00]])
```

```
[ ] from sklearn.model_selection import train_test_split
    train_data, test_data, train_labels, test_labels = train_test_split(features, label, test_size=0.3, random_state=42)
```

```
[ ] yield_df.to_csv('yield_df.csv')
```

```
[ ] from sklearn.model_selection import train_test_split
    train_data, test_data, train_labels, test_labels = train_test_split(features, label, test_size=0.3, random_state=42)
```

```
[ ] from sklearn.metrics import r2_score
    def compare_models(model):
        model_name = model.__class__.__name__
        fit=model.fit(train_data,train_labels)
        y_pred=fit.predict(test_data)
        r2=r2_score(test_labels,y_pred)
        return([model_name,r2])
```

```
[ ] from sklearn.ensemble import RandomForestRegressor
    from sklearn.ensemble import GradientBoostingRegressor
    from sklearn import svm
    from sklearn.tree import DecisionTreeRegressor

    models = [
        GradientBoostingRegressor(n_estimators=200, max_depth=3, random_state=0),
         RandomForestRegressor(n_estimators=200, max_depth=3, random_state=0),
        svm.SVR(gamma='auto'),
       DecisionTreeRegressor()
    ]
```

```
[ ] model_train=list(map(compare_models,models))
```

```
[ ] print(*model_train, sep = "\n")
```

```
['GradientBoostingRegressor', 0.8965731164462923]
['RandomForestRegressor', 0.6842532317855172]
['SVR', -0.20353376480360752]
['DecisionTreeRegressor', 0.9604317192857756]
```

```
[ ] yield_df_onehot = yield_df_onehot.drop(['Year'], axis=1)
```

```
[ ] yield_df_onehot.head()
```

	hg/ha_yield	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	C
0	36613	1485.0	121.0	16.37	1	0	0	
1	66667	1485.0	121.0	16.37	1	0	0	
2	23333	1485.0	121.0	16.37	1	0	0	
3	12500	1485.0	121.0	16.37	1	0	0	
4	7000	1485.0	121.0	16.37	1	0	0	

5 rows × 115 columns

```
[ ] test_df=pd.DataFrame(test_data,columns=yield_df_onehot.loc[:, yield_df_onehot.columns != 'hg/ha_yield'].columns)
    cntry=test_df[[col for col in test_df.columns if 'Country' in col]].stack()[test_df[[col for col in test_df.columns if 'Count
    cntrylist=list(pd.DataFrame(cntry).index.get_level_values(1))
    countries=[i.split("_")[1] for i in cntrylist]
    itm=test_df[[col for col in test_df.columns if 'Item' in col]].stack()[test_df[[col for col in test_df.columns if 'Item' in c
    itmlist=list(pd.DataFrame(itm).index.get_level_values(1))
    items=[i.split("_")[1] for i in itmlist]
```

```
[ ] test_df.head()
```

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country_Albania	Country_Algeria	Country_Angola	Country_Argent
0	0.183443	0.110716	0.542078	0.0	0.0	0.0	
1	0.458451	0.000413	0.627257	0.0	0.0	0.0	
2	0.183443	0.106159	0.518228	0.0	0.0	0.0	
3	1.000000	0.224154	0.890971	0.0	0.0	0.0	
4	0.458451	0.000355	0.625213	0.0	0.0	0.0	

5 rows × 114 columns

```
[ ] test_df.drop([col for col in test_df.columns if 'Item' in col],axis=1,inplace=True)
    test_df.drop([col for col in test_df.columns if 'Country' in col],axis=1,inplace=True)
    test_df.head()
```

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp
0	0.183443	0.110716	0.542078
1	0.458451	0.000413	0.627257
2	0.183443	0.106159	0.518228
3	1.000000	0.224154	0.890971
4	0.458451	0.000355	0.625213

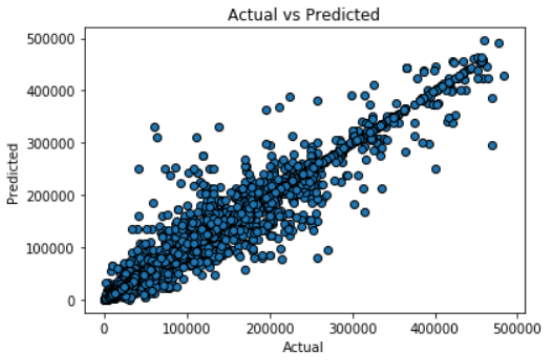
```
[ ] test_df['Country']=countries
    test_df['Item']=items
    test_df.head()
```

	average_rain_fall_mm_per_year	pesticides_tonnes	avg_temp	Country	Item
0	0.183443	0.110716	0.542078	Spain	Rice, paddy
1	0.458451	0.000413	0.627257	Madagascar	Wheat
2	0.183443	0.106159	0.518228	Spain	Sorghum
3	1.000000	0.224154	0.890971	Colombia	Potatoes
4	0.458451	0.000355	0.625213	Madagascar	Sweet potatoes

```
[ ] clf=DecisionTreeRegressor()
    model=clf.fit(train_data,train_labels)

    test_df["yield_predicted"]= model.predict(test_data)
    test_df["yield_actual"]=pd.DataFrame(test_labels)["hg/ha_yield"].tolist()
    test_group=test_df.groupby("Item")
    test_group.apply(lambda x: r2_score(x.yield_actual,x.yield_predicted))
```

```
Item
Cassava                 0.928131
Maize                   0.894730
Plantains and others    0.774214
Potatoes                0.911193
Rice, paddy             0.894121
Sorghum                 0.803444
Soybeans                0.815715
Sweet potatoes          0.839861
Wheat                   0.924442
Yams                    0.927334
dtype: float64
```

```
[ ] fig, ax = plt.subplots()
    ax.scatter(test_df["yield_actual"], test_df["yield_predicted"],edgecolors=(0, 0, 0))
    ax.set_xlabel('Actual')
    ax.set_ylabel('Predicted')
    ax.set_title("Actual vs Predicted")
    plt.show()
```

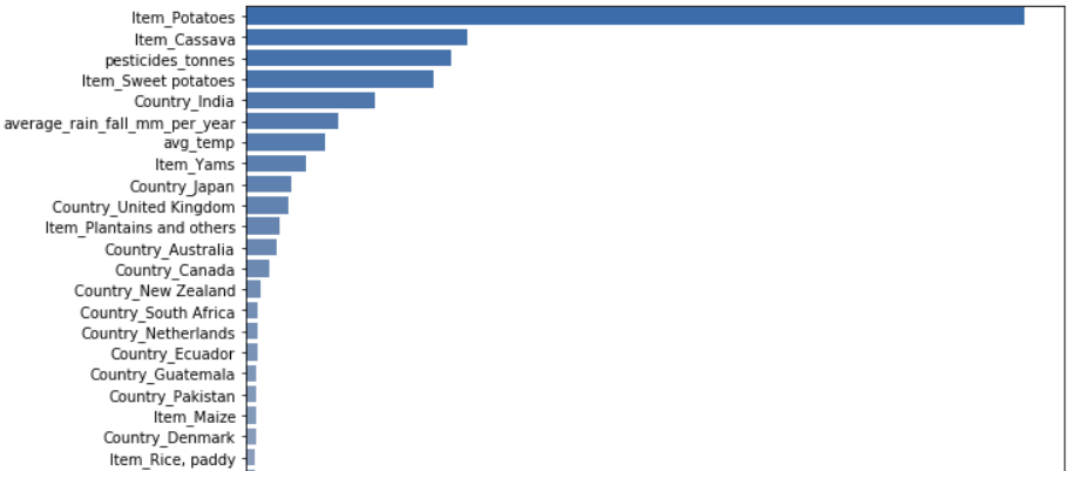



```
[ ]  def adjusted_r_squared(y,yhat,x):
         score=1- (((1-(r2_score(y,yhat)))*(len(y)-1))/(len(y)-x.shape[1]-2))
         return score

     test_group.apply(lambda x: adjusted_r_squared(x.yield_actual,x.yield_predicted,x))
```

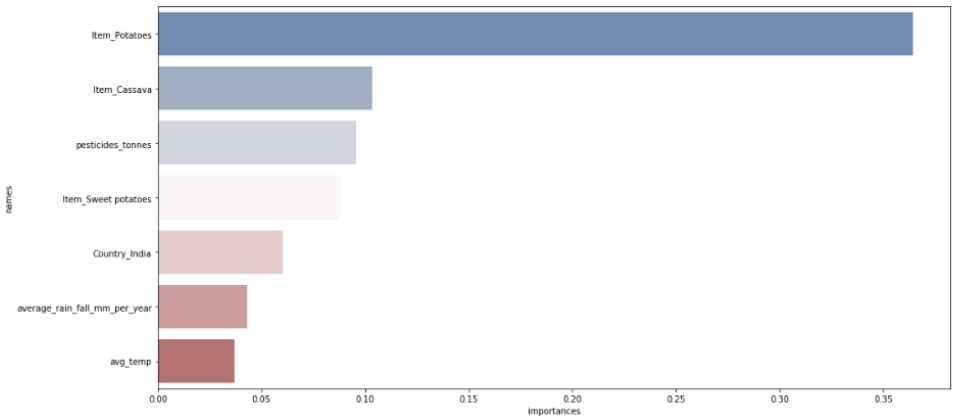
```
Item
Cassava                 0.927198
Maize                   0.894050
Plantains and others    0.762561
Potatoes                0.910646
Rice, paddy             0.893241
Sorghum                 0.801683
Soybeans                0.814220
Sweet potatoes          0.838370
Wheat                   0.923915
Yams                    0.924784
dtype: float64
```

```
[ ]  varimp= {'importances':model.feature_importances_,'names':yield_df_onehot.columns[yield_df_onehot.columns!="hg/ha_yield"]}
```

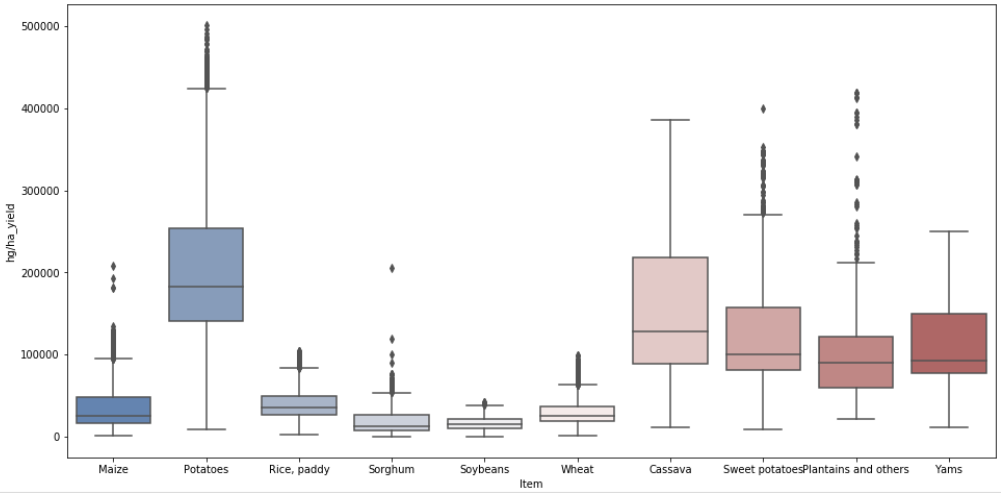
```
[ ]  a4_dims = (10,30)

     fig, ax = plt.subplots(figsize=a4_dims)
     df=pd.DataFrame.from_dict(varimp)
     df.sort_values(ascending=False,by=["importances"],inplace=True)
     df=df.dropna()
     sns.barplot(x="importances",y="names",palette="vlag",data=df,orient="h",ax=ax);
```




```
[ ]  a4_dims = (16.7, 8.27)
     fig, ax = plt.subplots(figsize=a4_dims)
     df=pd.DataFrame.from_dict(varimp)
     df.sort_values(ascending=False,by=["importances"],inplace=True)
     df=df.dropna()
     df=df.nlargest(7, 'importances')
     sns.barplot(x="importances",y="names",palette="vlag",data=df,orient="h",ax=ax);
```




```
[ ]  a4_dims = (16, 8.)
     fig, ax = plt.subplots(figsize=a4_dims)
     sns.boxplot(x="Item",y="hg/ha_yield",palette="vlag",data=yield_df,ax=ax);
```



ДОДАТОК Б

2.py

```
import pandas as pd
from sklearn import preprocessing
from sklearn.neighbors import KNeighborsClassifier
import numpy as np
import PySimpleGUI as sg

excel = pd.read_excel('Crop.xlsx', header = 0)
print(excel)
print(excel.shape)

le = preprocessing.LabelEncoder()
crop = le.fit_transform(list(excel["CROP"]))

NITROGEN = list(excel["NITROGEN"])
PHOSPHORUS = list(excel["PHOSPHORUS"])
POTASSIUM = list(excel["POTASSIUM"])
TEMPERATURE = list(excel["TEMPERATURE"])
HUMIDITY = list(excel["HUMIDITY"])
PH = list(excel["PH"])
RAINFALL = list(excel["RAINFALL"])

features = list(zip(NITROGEN, PHOSPHORUS, POTASSIUM, TEMPERATURE, HUMIDITY, PH,
RAINFALL))
features = np.array([NITROGEN, PHOSPHORUS, POTASSIUM, TEMPERATURE, HUMIDITY, PH,
RAINFALL])

features = features.transpose()
print(features.shape)
print(crop.shape)

model = KNeighborsClassifier(n_neighbors=3)
model.fit(features, crop)
layout = [[sg.Text('Система прогнозу урожайності', font=("Helvetica", 30), text_color =
'yellow')],
          [sg.Text('Ввести наступні дані :', font=("Helvetica", 20))],
          [sg.Text('Вміст азоту в грунті :                    ', font=("Helvetica", 20)),
sg.Input(font=("Helvetica",20), size = (20,1) )],
          [sg.Text('Вміст фосфору в грунті:', font=("Helvetica", 20)),
sg.Input(font=("Helvetica", 20),size = (20,1))],
          [sg.Text('Вміст калію в грунті:', font=("Helvetica", 20)),
sg.Input(font=("Helvetica", 20),size = (20,1))],
```

```
        [sg.Text('Середнє значення температури:', font=("Helvetica", 20)),
sg.Input(font=("Helvetica", 20),size = (20,1)), sg.Text('C', font=("Helvetica", 20))],
        [sg.Text('Середнє значення вологості:', font=("Helvetica", 20)),
sg.Input(font=("Helvetica", 20),size = (20,1)), sg.Text('%', font=("Helvetica", 20))],
        [sg.Text('Значення PH грунту:', font=("Helvetica", 20)),
sg.Input(font=("Helvetica", 20),size = (20,1))],
        [sg.Text('Середня кількість опадів:', font=("Helvetica", 20) ),
sg.Input(font=("Helvetica", 20),size = (20,1)),sg.Text('мм', font=("Helvetica", 20))],
        [sg.Text(size=(50,1),font=("Helvetica",20) , text_color = 'yellow', key='-
OUTPUT1-' )],
        [sg.Button('Прогноз', font=("Helvetica", 20)),sg.Button('Вихід',
font=("Helvetica", 20))] ]
window = sg.Window('Система прогнозу урожайності', layout)

while True:
      event, values = window.read()
      if event == sg.WINDOW_CLOSED or event == 'Quit':
            break
      print(values[0])
      nitrogen_content = values[0]
      phosphorus_content = values[1]
      potassium_content = values[2]
      temperature_content = values[3]
      humidity_content = values[4]
      ph_content = values[5]
      rainfall = values[6]
      predict1 = np.array([nitrogen_content,phosphorus_content, potassium_content,
temperature_content, humidity_content, ph_content, rainfall])
      print(predict1)
      predict1 = predict1.reshape(1,-1)
      print(predict1)
      predict1 = model.predict(predict1)
      print(predict1)
      crop_name = str()
if predict1 == 0:
            crop_name = 'яблука'
      elif predict1 == 1:
            crop_name = 'банани'
      elif predict1 == 2:
            crop_name = 'blackgram'
      elif predict1 == 3:
            crop_name = 'нут'
      elif predict1 == 4:
            crop_name = 'кокос'
      elif predict1 == 5:
```

```
            crop_name = 'кава'
      elif predict1 == 6:
            crop_name = 'бавовник'
      elif predict1 == 7:
            crop_name = 'виноград'
      elif predict1 == 8:
            crop_name = 'джут'
      elif predict1 == 9:
            crop_name = 'квасоля'
      elif predict1 == 10:
            crop_name = 'lentil'
      elif predict1 == 11:
            crop_name = 'кукурудза'
      elif predict1 == 12:
            crop_name = 'манго'
      elif predict1 == 13:
            crop_name = 'боби'
      elif predict1 == 14:
            crop_name = 'mungbeans'
      elif predict1 == 15:
            crop_name = 'диня'
      elif predict1 == 16:
            crop_name = 'апельсин'
      elif predict1 == 17:
            crop_name = 'papaya'
      elif predict1 == 18:
            crop_name = 'pigeonpeas'
      elif predict1 == 19:
            crop_name = 'гранат'
      elif predict1 == 20:
            crop_name = 'рис'
      elif predict1 == 21:
            crop_name = 'кавун'

if int(humidity_content) >=1 and int(humidity_content)<= 33 :
            humidity_level = 'low humid'
      elif int(humidity_content) >=34 and int(humidity_content) <= 66:
            humidity_level = 'medium humid'
      else:
            humidity_level = 'high humid'

      if int(temperature_content) >= 0 and int(temperature_content)<= 6:
            temperature_level = 'cool'
      elif int(temperature_content) >=7 and int(temperature_content) <= 25:
            temperature_level = 'warm'
```

```
else:
      temperature_level= 'hot'

if int(rainfall) >=1 and int(rainfall) <= 100:
      rainfall_level = 'less'
elif int(rainfall) >= 101 and int(rainfall) <=200:
      rainfall_level = 'moderate'
elif int(rainfall) >=201:
      rainfall_level = 'heavy rain'

if int(nitrogen_content) >= 1 and int(nitrogen_content) <= 50:
      nitrogen_level = 'less'
elif int(nitrogen_content) >=51 and int(nitrogen_content) <=100:
      nitrogen_level = 'not to less but also not to high'
elif int(nitrogen_content) >=101:
      nitrogen_level = 'high'

if int(phosphorus_content) >= 1 and int(phosphorus_content) <= 50:
      phosphorus_level = 'less'
elif int(phosphorus_content) >= 51 and int(phosphorus_content) <=100:
      phosphorus_level = 'not to less but also not to high'
elif int(phosphorus_content) >=101:
      phosphorus_level = 'high'

if int(potassium_content) >= 1 and int(potassium_content) <=50:
      potassium_level = 'less'
elif int(potassium_content) >= 51 and int(potassium_content) <= 100:
      potassium_level = 'not to less but also not to high'
elif int(potassium_content) >=101:
      potassium_level = 'high'

if float(ph_content) >=0 and float(ph_content) <=5:
      phlevel = 'acidic'
elif float(ph_content) >= 6 and float(ph_content) <= 8:
      phlevel = 'neutral'
elif float(ph_content) >= 9 and float(ph_content) <= 14:
      phlevel = 'alkaline'

print(crop_name)
print(humidity_level)
print(temperature_level)
print(rainfall_level)
print(nitrogen_level)
print(phosphorus_level)
print(potassium_level)
```

```
        print(phlevel)

        window['-OUTPUT1-'].update('Найкращий урожай, який можна виростити:'+ crop_name)

window.close()
```