Міністерство освіти і науки України
Прикарпатський національний університет імені Василя Стефаника
Кафедра комп'ютерної інженерії та електроніки
(повна назва кафедри)

Стамбульська Романна Тарасівна
Stambulska Romanna

УДК 004:681.5

Спеціальність 123-Комп'ютерна інженерія
(шифр та назва спеціальності)

Кваліфікаційна робота
на здобуття освітньо-кваліфікаційного рівня магістр
(бакалавр, спеціаліст, магістр)

Аналіз та візуалізація даних в інженерії програмування на мові Python

Data analysis and visualisation in programming engineering with Python

Науковий керівник:
доктор технічних наук, професор
Когут І.Т.

Рецензент:
кандидат фіз.-мат. наук, професор
Никируй Л.І.

Івано-Франківськ
2022

Формат	Поз.	Позначення	Найменування	Кільк.	Прим.
		123.КІ(м)-21.11	Пояснювальна записка	1	

Змн.	Арк.	№ докум.	Підпис	Дата	123.КІ(м).21.11		
Розробила		Стамбльська Р.Т			Специфікація	Літ.	Арк. 2 / Аркушів 1
Перевірив		Когут І. Т.					
Н. Контр.							
Затвердив							

АНОТАЦІЯ

В магістерській кваліфікаційній роботі було проведено аналіз даних для виявлення закономірностей за допомогою зведеної статистики та елементів візуалізації на мові Python.

В пояснювальній записці було проведено огляд аспектів аналізу даних і особливостей його використання. Виконано ознайомлення з масивами даних BigData, і з методами класифікації типів даних. Вивчено основні можливості бібліотек мови Python для виконання аналізу та візуалізації даних. Використано здобуті теоретичні навики для реалізації власного проекту.

Загальний обсяг роботи - 64 сторінки, 22 рисунки, 1 таблиця.

ABSTRACT

In the master's qualification, data analysis was performed to identify patterns using summary statistics and visualization elements with Python.

The explanatory note reviewed the aspects of data analysis and the features of its use. Acquaintance with BigData data arrays and data type classification methods was performed. The main capabilities of Python language libraries for data analysis and visualization were studied. The acquired theoretical skills were used to implement one's own project.

Total volume of work - 64 pages, 22 figures, 1 table.

Міністерство освіти і науки України
Прикарпатський національний університет імені Василя Стефаника
Кафедра комп'ютерної інженерії та електроніки

Пояснювальна записка

до магістерської кваліфікаційної роботи на тему:

«Аналіз та візуалізація даних в інженерії програмування на мові Python»

					123.КІ(м).21.11		
Змн.	*Арк.*	*№ докум.*	*Підпис*	*Дата*			
Розробила		Стамбульська Р.Т.			Пояснювальна записка	*Літ.*	*Арк.* 4 / *Аркушів*
Перевірив		Когут І.Т.					
Н. Контр.							
Затвердив							

ЗМІСТ

ВСТУП

Сфера аналітики даних стала максимально важливою через неймовірно швидкі обсяги зростання цифрових даних. У світі, що все більше концентрується навколо інформаційних технологій, сотні трильйонів інформації щомиті передаються, приймаються та зберігаються через інтернет.

Процес аналізу даних полягає у впорядкуванні, очищенні і аналізі різних типів необроблених даних, що могли бути зібрані з різних за надійністю джерел, для отримання максимально корисних висновків, полегшення приймання рішень, оптимізації діяльності, максимізації прибутку. Розглядають декілька типів аналітики, ключовими питаннями яких є: «що сталося?», «чому щось сталося?», «що може статися?» або «що краще робити далі?» (Descriptive, Diagnostic, Predictive, Prescriptive analysis відповідно).

У цій галузі науки, основою вдалого результату є поєднане застосування як технічних інструментів, так і обов'язкових дисциплін – алгоритмів мов програмування, методів статистики і математики. Важливість знання інших дисциплін приходить в залежності від сфери застосування конкретного проекту аналізу даних, що розпочинається. Загалом, досвід у сферах, що аналізуються, завжди буде перевагою, бо це може допомогти краще зрозуміти проблеми, тип необхідних даних і засоби для їх збору, виявляти аномалії в результатах, розпізнавати проблеми, що виникають під час аналізу.

Процес аналізу даних дуже важко уявити без візуалізації. Людина краще сприймає інформацію не переглядаючи табличні дані, а через представлені зображення, ефективно знаходячи закономірності, зв'язки, тенденції. Це дозволяє вдосконалити процес прийняття більш стратегічно керованих рішень. В сферах бізнесу, маркетингу, інформаційних технологій може використовуватись навіть як інструмент для заохочення потенційних клієнтів.

Купівля ноутбука – це рішення, яке з часом приймає майже кожен. Бажання отримати хороше обладнання з відмінним співвідношенням вартості до якості, змушує потенційних покупців намагатись визначити ключові параметри й те, від

чого можна відмовитися. Саме тому, об’єктом дослідження виступає датасет ноутбуків, що найчастіше куплялись за останні два роки.

Метою цієї кваліфікаційної роботи є проведення дослідницького аналізу (Exploratory Data Analysis) всіх компонентів і визначення впливу технічних характеристик на вартість ноутбуків. Інструментами виконання є статистичні методи і мова програмування Python останньої версії, а в особливості її бібліотеки: pandas і numpy – для маніпулювання даними в процесах їх впорядкування, очищення та вибірки; matplotlib і seaborn – для створення візуалізаційних компонентів, що дозволять полегшити оцінювавння результатів.

РОЗДІЛ 1
АНАЛІЗ ДАНИХ ТА ЙОГО АСПЕКТИ

Людей завжди оточує колосальна кількість інформації, і з кожним днем появляється вдосталь нових даних. Не зважаючи на те, чи це звичайна дитина, що шукає відповіді на безліч своїх запитань, чи студент, який займається збором інформації для наукової роботи, чи це компанія, яка намагається краще ознайомитись із бажаннями потенційних клієнтів – всі вони мають змогу отримати необхідні дані.

Компанії швидко усвідомили яким чином можна скоротити витрати і час, розробляти нові продукти, та оптимально приймати рішення. Було виявлено, що для цього потрібно не багато – просто використовувати дані у правильному керунку. Саме тому, вони займаються збором та вивченням інформації, яка, при правильному використанні, допоможе виявити певні принципи, прогнози, і використовують це для переваги над конкурентами.

Проте, звичайним збором даних не можна досягти успіху. Акурат після цього починається головний етап – процес аналізу даних. Це клопітка робота, яка дозволить побачити певні закономірності і зрозуміти, які рішення необхідно прийняти для гарантованого бізнес-успіху. Отже, метою аналізу даних є організація та певне осмислення необробленої інформації, щоб отримати відповіді про її вплив на базові рішення.

Існує достатня кількість методів, що дозволяють вносити певні коригування для адаптації під необхідні потреби, і які використовуються для полегшення роботи з аналізом даних. Однак, потрібно пильно стежити за маніпулюванням наявними даними. На початкових етапах можна з легкістю отримати помилкові висновки, якщо працювати з хибними джерелами інформації.

1.1. Термін Big Data та особливості його використання.

Big Data (великі дані) – різноманітні дані, нескінченна кількість яких надходить з усе більшою швидкістю. Ці масиви даних є таких об’ємів, що унеможливлюють їх аналіз за допомогою звиклих технологій. Вони зайняли

центральну позицію, в основному, після появи пошукових систем, розважальних сервісів та соціальних мереж.

Близько сімнадцяти років тому було помічено, що за допомогою таких сервісів можна генерувати надзвичайну кількість даних користувачів. Саме в той період почали набирати популярності фреймворки для збору та аналізу даних (які дозволяли працювати з такою кількістю інформації легко і відносно дешево), та розподілена база даних NoSQL для зберігання інформації.

Розвиток технологій дозволив займатись збором інформації навіть побутовим пристроям, які підключені до інтернету. Вони здатні відстежувати людські вподобання і взаємодії, записувати їх і передавати до Big Data. Якщо ж розглядати факт появи машинного навчання, то це свідчить про ще більший обсяг генерування щоденних даних.

Для Big Data можна сформувати наступну характеристику, візуалізація якої зображено на рисунку 1.1:

1) Volume (обсяг). Щоб дані відносились до Big Data їх масиви мають бути значно більшими, ніж звичайні. Такі масиви переважно містять в собі неструктуровані дані з обмеженими структурованими та напівструктурованими даними. Відомо, що розміри клієнтської бази компаній еквівалентні обсягу даних, що надійшли до компанії. Такі дані збираються з веб-сторінок, пошукових історій, мобільних додатків, соціальних мереж.
2) Velocity (швидкість) – швидкість збирання даних. Часто, для пришвидшення збору даних використовують комбінації локальних та хмарних серверів. Сучасні інформаційні продукти мають на меті здаватись «розумними» для користувачів, тому їм необхідний доступ до всіх даних споживачів у режимі реального часу.
3) Variety (різноманітність). Поява Big Data зумовила появу нових типів неструктурованих даних (відео, аудіо, тексти), які для свого аналізу потребують більш складних інструментів та технологій очищення. Тільки після цього вони дозволяють отримати цінну інформацію.

4) Veracity (точність). Цей пункт визначає надійність отриманих даних. Наприклад, соціальні мережі, що містять в собі різноманітні блоги та публікації, що оздоблені хештегами, акронімами та людськими помилками при друку, можуть значно знизити надійність і точність наборів даних.

5) Value (цінність). Вже давно дані можна вважати як специфічну валюту з притаманною їй вартістю. Цінність даних визначається тим, яку інформацію з них здобули [1].

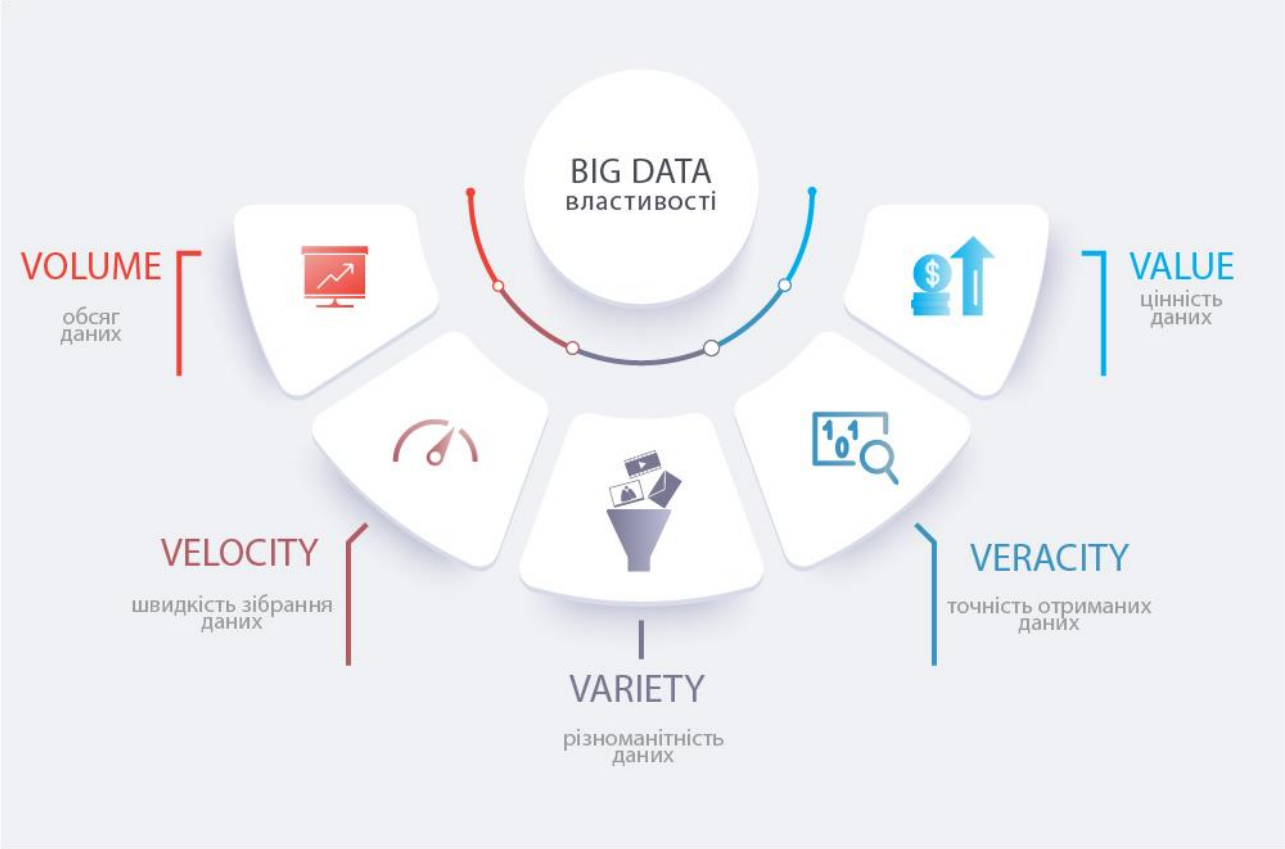


Рис. 1.1. Властивості Big Data.

Чим більше даних у нас є, тим більше інформації та деталей можна з них отримати. Тому, використання Big Data застосовується в різноманітних сферах:

1) У розробці програмних продуктів для розуміння вимог і сподівань клієнтів. Перед випуском нових продуктів розробляються прогностичні моделі. Наприклад, певні компанії застосовують дані, зібрані з сторінок соціальних мереж, тестових ринків і фокус-груп під час підготовки до запуску нових продуктів.

2) Для уникнення технічних збоїв та несправностей обладнання використовується прогнозоване технічне обслуговування. Для цього проводиться аналіз великої частини неструктурованих даних (відомості про помилки, записи про роботу та температурні показники) та структурованих даних (документована інформація про рік виготовлення обладнання, його модель) за допомогою необхідних аналітичних інструментів. Аналіз таких даних дозволяє зробити прогнози про можливі несправності, продовжити термін придатності, та підготуватись до технічного обслуговування обладнання.
3) Для взаємодії з клієнтом, тобто, можливості надання персоналізованих рекомендації для збільшення притоку користувачів, і залучання до своїх продуктів платоспроможної аудиторії.
4) В банках та різноманітних фінансових установах, які постійно мать боротись із шахрайством. Великі дані допомагають визначати шаблони і оцінювати схеми попередніх шахрайських транзакцій, щоб ефективно виявляти їх, а також зробити все для запобігання майбутнім.
5) У сферах машинного навчання. Щоб машина могла самостійно навчатись її потрібно забезпечити величезним об’ємом даних. Надійна інформація, що містить всі різновиди даних, допоможе машинам розвинути погляди на реальний світ і проблеми, для вирішення яких вони були призначені.
6) Для операційної ефективності та стимулювання інновацій. Здатність вивчати та передбачати майбутні тенденції попиту, аналізуючи відгуки клієнтів і дані, які стосуються найпопулярніших товарів та їх повернення, дозволяє удосконалити прийняття рішень стосовно виготовлення товарів. Вивчаючи взаємозв’язки між людьми та їхніми електронними пристроями, а також виробниками цих пристроїв, компанії можуть розробляти інноваційні продукти, досліджуючи поточні тенденції в продуктах і відповідаючи очікуванням клієнтів.

1.2. Типи даних.

Типи даних розподіляють на дві категорії – якісні і кількісні, а вони, в свою чергу, містять в собі певні підкатегорії.

Якісні дані – це ті, що не можуть бути виміряні. Вони спостерігаються тільки суб'єктивно з додаванням якісних характеристик до об'єктів. Якісні дані створюються шляхом класифікацій об'єктів за ознаками таких атрибутів, як запах, колір, текстура. Цей тип даних можна розділити на такі підкатегорії:

1) Двійкові дані – ті, що сигналізують про взаємовиключні події (можливість тільки одного з варіантів бути правильним і в подальшому бути застосованим). Варіант розділення на істину та хибу можна також називати біноміальним типом даних. Цей тип даних зазвичай використовують для розробки статистичних моделей при прогнозованому аналізі.
2) Номінальні дані – дані, що не мають неявного значення. Цей тип даних часто застосовується для оцінювання статистичних відмінностей у наборі даних за допомогою відповідних методів.
3) Впорядковані дані – ті, що справді мають в собі певне неявне значення (наприклад, малий, середній, великий), чи оцінюються рейтинговою шкалою.

Кількісні дані містять характеристики, що можуть бути об'єктивно виміряними. Класифікація об'єкта за допомогою вимірюваних ознак і надання йому числового значення викликає створення цього типу даних. Кількісні дані можна розділити на такі підкатегорії:

1) Неперервні дані – їх значення можуть бути виміряні на наступному нижчому рівні (для прикладу, кілометр – метр – сантиметр). Тобто, можна записати, що коробка з ноутбуком вагою 2,5 кілограми , або ж 2500 грам. Саме це відображає собою безперервний тип даних, який у більшості випадків використовується для тестування різних типів гіпотез, таких як оцінка точності ваги, вказаної на товарах.

2) Дискретні дані – це числові значення даних, що не можуть бути розділені, чи зменшені до вищого рівня точності. В цьому випадку можуть збиратись тільки неподільні числа (тобто, людина не може бути власником 2,5-ти ноутбуків), і вони називаються дискретним типом. Цей тип даних може використовуватись в поєднанні з безперервним типом даних для виконання регресійного аналізу (перевірка чи загальна вага коробки з ноутбуком взаємопов'язана з кількістю ноутбуків в середині неї).

Структуровані та неструктуровані дані

Розрізняють ще одну категорію даних: структуровані, напівструктуровані, та неструктуровані (рис. 1.2).

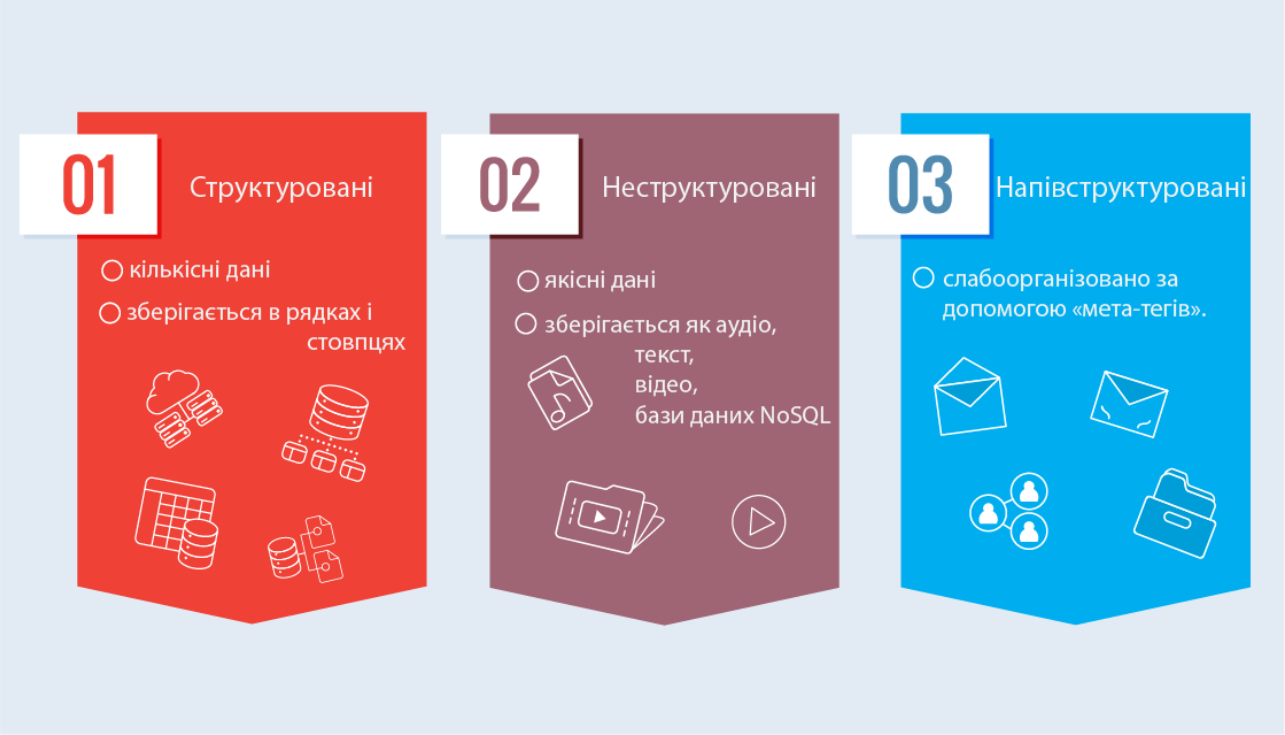


Рис. 1.2. Категорії даних.

Структурованими називають дані, що були заздалегідь визначені та відредаговані до певної структури перед публікацією до сховища даних (це називають схемою при записі). Структуровані дані часто є кількісними даними, використовують свої переваги здебільшого при записі. Як приклад такого типу даних можна розглядати реляційні бази даних, де вся інформація занесена в точно

визначені поля для того, щоб до них було легко звертатись. Для структурованого типу даних можливе використання мови програмування SQL (Structured Query Language), що обробляє реляційні бази даних.

Структуровані дані віддавна були основою для систем контролю банкоматів. Вони можуть бути створені людиною або машиною. До створених машиною структурованих даних можна віднести статистику веб-журналів, чи дані про продажі, штрих-коди. Класичним прикладом структурованих даних, які були створені людьми, є електронні таблиці.

Можна визначити наступні переваги структурованих даних:

1) Вони дуже легко використовуються алгоритмами машинного навчання. Організований характер структурованих даних дає змогу спрощено робити запит до необхідної інформації.
2) Якщо бізнес-користувач розуміє тему, до якої відносяться дані, то він може легко їх використовувати у своїх потребах. Людині не потрібно володіти глибокими знаннями стосовно взаємозв'язків даних, що дозволяє відкрити перед собою етап самообслуговування.
3) Структуровані дані мають більшу кількість інструментів, які були випробувані на них для аналізу даних. Це пояснюється тим, що цей тип використовувався значно довше, бо був єдиним варіантом. Отже, при роботі з ними є більше можливостей для вибору засобів роботи.

До недоліків цього типу даних можна віднести недостатню гнучкість даних, а саме:

1) Хоча і визначення даних під час запису є великою перевагою для структурованих даних, проте, дані з попередньо визначеною структурою можуть бути використані тільки за своїм призначенням. Цей фактор здатен обмежувати гнучкість та варіанти використання.
2) Збереження цього типу даних у сховищах даних (системах для зберігання даних із жорсткими схемами). Якщо виникає потреба внесення певних змін, то потрібно проводити оновлення всіх цих структурованих даних відповідно до нових вимог, що викликає великі

грошові затрати і втрату часу. Частина витрат може бути зменшена за допомогою хмарного сховища даних, оскільки це забезпечує більшу масштабованість і усуває витрати на технічне обслуговування, що були пов’язані з розміщенням обладнання на місці.

Неструктуровані дані вважаються тими, що зберігаються у рідному форматі та не підлягають обробці, поки не виникає необхідність у їх використані, що називають схемою при читанні. Ці дані доступні у безлічі файлових форматах (дані з соціальних мереж, фото, відео, листи електронних пошт, супутникові зображення). Неструктуровані дані вважаються якісним типом, а не кількісним, що означає, що вони є більш характерними та категоричними за своєю природою. Використовує свої переваги цей тип даних зазвичай при зчитуванні.

Цей тип даних вдало підходить для того, щоб визначати ефективність маркетингових кампаній або для виявлення потенційних тенденцій купівлі через соціальні мережі та веб-сайти з оглядами. Він також може бути дуже корисним для підприємства, допомагаючи контролювати дотримання політики, оскільки його можна використовувати для виявлення шаблонів у чатах або підозрілих тенденцій електронної пошти.

Кількість неструктурованих даних значно перевищує структуровані дані. Його відсоток від загальної кількості світової інформації вже перевищує 80%, а зростання на цьому не зупиняється. Тому, доцільно вважати, що не використовуючи цей тип даних, компанії можуть втрачати колосальну частину бізнес-аналітики.

Можна визначити наступні переваги неструктурованих даних:

1) Дані не визначаються, доки не знадобляться, так як неструктуровані дані збережено у власному форматі до їх використання. Це дозволяє збільшити варіанти використання, бо їх призначення є адаптованим, а також, займатись підготовкою та аналізом лише необхідних даних. Можливість не обмежуватись певним форматом, зберігаючи більшу різноманітність форматів файлів у базі даних, дозволяє компаніям черпати величезну кількість інформації.

2) Швидкі темпи накопичення даних дозволяє уникати попереднього визначення даних, що означає їх швидший та легший збір.
3) Можливість зберігати масивні неструктуровані дані у так званих хмарних озерах даних, які дозволяють визначати ціни на зберігання за принципом «оплатити за використання», що допомагає знизити витрати та забезпечує легку масштабованість.

До недоліків цього типу даних можна віднести те, що для успішного їх використання, та можливості оцінити весь потенціал потрібні спеціальні знання та спеціальні інструменти:

1) Великим недоліком є необхідність експертних знань в галузі даних для їх підготовки та аналізу. Звичайний бізнес-спеціаліст не зможе користуватись цим типом даних у їх звичайному стані через їх невизначену природу. Необхідне розуміння теми, області даних та того, як пов’язувати всю інформацію, щоб зробити її корисною.
2) Також, є велика необхідність у використанні специфічних інструментів для маніпулювання, щоб опрацьовувати неструктуровані дані. Деякі з них ще знаходяться не у закінченому стані, а решта змушують менеджера даних працювати з обмеженим вибором продуктів для цього типу даних.

Напівструктуровані дані можна вважати чимось середнім між двома вищеперерахованими типами. Це структуровані дані, які не можна вписати у структуру реляційної бази даних, проте використовує системи тегів, чи ж якісь інші ідентифіковані маркери, відокремлюючи різні елементи та роблячи пошук можливим. Напівструктуровані дані мають більше ієрархії та є більш конкретними, ніж неструктуровані дані.

Як приклад, до цього типу даних можна віднести звичайні фотографії. Фото зроблені на телефон містять неструктурований вміст зображення, а також структуровану інформацію (позначений час, місце розташування). Іноді їх називають даними, що мають структуру, яка самоописується.

1.3. Етапи аналізу даних.

Для вдалого аналізу даних необхідно мати правильно поставлений процес та відповідні інструменти, що будуть налаштованими для цього конкретного аналізу даних. Це дозволить зробити процес, з багатьма складними етапами, значно зрозумілішим та простішим. Для того, щоб полегшити роботу і бути впевненим у прийнятих рішеннях, можна розділити процес на декілька етапів (рис. 1.3).

Рис. 1.3. Етапи проведення аналізу даних.

1.3.1. Розуміння потреб.

Першим етапом у аналізі даних є розуміння потреб для аналізу даних. Необхідно зрозуміти навіщо вам потрібен цей аналіз, та що саме потрібно для досягнення поставленої мети.

Зазвичай, не виникає проблем із браком інформації, а великою проблемою стає саме її надлишок. Якщо мати справу з надмірною кількістю даних, то це тільки ускладнить прийняття вірного рішення на їх основі.

Саме тому, потрібно бути впевненим, що отримані дані будуть відповідати на ті питання, на які хочемо отримати відповідь, а також, важливо знати, як можна зробити точні висновки з даних, з якими працюємо. Питання мають бути стислими, зрозумілими, та могти стверджувати, чи відхиляти деякі з потенційних рішень конкретної проблеми. Наприклад, «як змінюються певні показники з часом?» і «чи корелюють ці показники з іншими, і наскільки сильно?».

1.3.2. Збір даних.

Другим етапом є збір даних. Перш ніж займатись збором нових даних, потрібно визначити з якою інформацією працювати. Під рукою вже можуть бути готові бази даних, чи інші джерела, і спочатку потрібно пройтись по них. Цей процес був би легшим, вже міг би дати певну інформацію, та заощадити час і кошти. Якщо ж необхідна додаткова інформація, то можна перейти до інших джерел.

Ключовою частиною цього є визначення того, які дані потрібні. Це можуть бути кількісні (числові) дані, наприклад цифри продажів, або якісні (описові) дані, такі як відгуки людей. В бізнес-аналітиці розрізняють три категорії даних: першої сторони (дані відстежені безпосередньо від клієнтів, і впорядковані чітко визначеним чином), дані другої сторони (отримані від інших організацій, і ще можуть вважатись доволі надійними), та сторонні дані (зібрані та узагальнені з безлічі джерел сторонньою організацією).

Важливо зберігати дані якомога організованішими. Залежно від можливостей, організацію даних можна виконати на звичній користувачам електронній таблиці Excel, або за допомогою багатьох інструментів і програм, що можуть обробляти статистичні дані. Дані потрібно класифікувати та впорядкувати згідно вимог. Також, можна працювати із журналом, в якому записані дати збору, і додавати в нього примітки про джерела збору, і певну виконану нормалізацію. Це може допомогти в роботі з майбутніми висновками.

Можна виділити багато інструментів, якими можна скористатися для полегшення роботи. Що точно може знадобитись, незалежно від галузі та сфери роботи, це платформа керування даними (data management platform). Ця

платформа є частиною програмного забезпечення, що може ідентифікувати та збирати дані з різних джерел, перш ніж маніпулювати ними.

1.3.3. Очищення даних.

Щоб аналіз даних був виконаний вірно, дуже важливо бути переконаним, що в наборі даних немає будь-яких помилок, дублікатів, надмірних даних, неправильного формату, не приносять відсутні значення, що змусить їх дуже важко проходити через алгоритми. Необхідна попередня обробка, щоб переконатися, що проходить аналіз лише високоякісних даних, а ресурси використовуються ефективно.

Першим чином варто переконатися, що дані мають однаковий формат, і найкращий спосіб впоратися з цим – це помістити всю інформацію в стандартизовану базу даних, яку можна переглянути.

Наступним кроком важливо перевірити наявність відсутніх значень, бо при отриманні інформації з реального світу вони можуть трапитись в одній із частин. Якщо їх небагато, то вони можуть бути видаленими, або заміненими на середні значення з інших стовпців цього рядку. Спеціаліст, який займається аналізом, сам вирішує, що зробити з відсутніми значеннями.

Після цього потрібно вирішити проблему з можливими повторюваними значеннями. Якщо таких значень багато, то вони можуть спотворювати багато результатів, що будуть отримані. Іноді можуть бути присутні дублікати, і їх може бути зменшено до двох, а іноді всі записи будуть збережені унікально. Вибір за спеціалістом.

1.3.4. Аналіз підготовлених даних.

Третім етапом є сам аналіз очищених даних. В залежності від бажаного результату, для аналізу можна використовувати різні підходи (рис. 1.4).

Прогнозний аналіз (Predictive analysis) – дозволяє спрогнозувати майбутні тенденції, чи плани та зростання бізнесу за допомогою попереднього досвіду, статистики та різноманітних методів моделювання. Такий аналіз може використовуватись у багатьох сферах (прогнозування погоди, створення ігор, маркетингові стратегії).

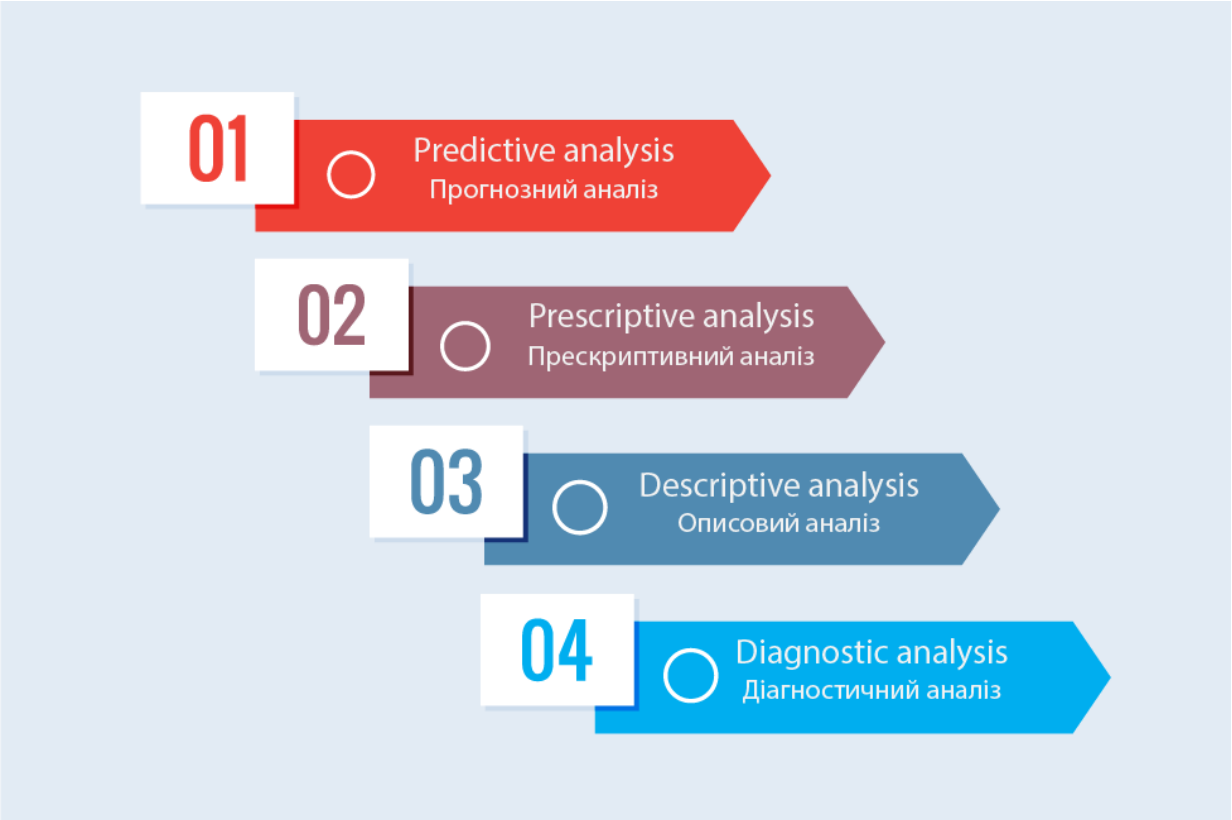


Рис. 1.4. Підходи аналізу даних.

Цей типу аналізу покращується із розвитком машинного навчання, і прогнози стають все точнішими. Прогнозний аналіз значно впливає на зниження витрат, та зменшення ризиків інвесторів і фінансових спеціалістів. Також, його можна проводити вручну або за допомогою алгоритмів машинного навчання. Одним із інструментів прогнозної аналітики є регресійний аналіз, який може визначити зв'язок між двома змінними (однолінійна регресія), або трьома, чи більше змінними (множинна регресія). Зв'язки між змінними записуються як математичне рівняння, яке допомагає передбачити результат, якщо одна змінна зміниться.

Іншим інструментом можуть бути класифікаційні моделі, що розподіляють дані по категоріях на основі історичних знань, і вивчають кореляції між даними та мітками та категоризують будь-які нові дані. Такі моделі можна легко перенавчити за допомогою нових даних, тому вони використовуються в багатьох різноманітних галузях.

Прескриптивний аналіз (Prescriptive analysis) – дозволяє відповідати на питання – що потрібно зробити зараз, щоб досягти певної мети в майбутньому.

Цей вид аналізує дані про минулу продуктивність за допомогою машинного навчання, тому в ньому визначається доволі мізерна кількість людського внеску.

Такий аналіз охоплює множину, як простих можливостей (прийняття рішень, пропозиції коригування цін, автоматизація торгів, визначення рівня запасів), так і складніших (безупинний та автоматизований процес обробки нових даних для збільшення точності і отримання кращих рішень, із можливістю ілюстрування наслідків кожного з варіантів), і найскладніших (підтримання механізмами потокової обробки з аналізом потенційних рішень, їх взаємодію та вплив на результат).

Прескриптивний аналіз можна вважати найкращим інструментом при створенні бізнес-аналітики будь-якої з організацій. Завдяки роботі штучного інтелекту зменшується ризик людської помилки, чи упередженого ставлення, дані обробляються значно швидше, а завдяки досконалим алгоритмам забезпечується точніша аналітика. Проте, мають місце і недоліки при описі цього аналізу: 1) він ефективний, якщо вхідні дані мають вірні припущення і не було допущено помилок; 2) не підходить для довготривалої аналітики, бо результати втрачають свою надійність.

Описовий аналіз (Descriptive analysis) – дозволяє описувати конкретні набори даних, та надавати невеликі узагальнення про показники даних за допомогою статичних методів. Цей тип виділяється поміж інших тим, що не буде робити прогнозування майбутніх перспектив.

Найвідомішим у описовому аналізі є визначення центру, що застосовується у математиці, чи статистиці (середнє значення, медіана). Середнє значення можна визначити при додаванні всіх цифр у наборі даних, і діленні їх на загальну кількість цифр. Тоді, режимом набору даних буде значення, яке появлятиметься найчастіше, а медіаною – цифра, розташована в середині набору даних.

Дуже часто описовий аналіз використовується для перетворення складних кількісних даних з великого масиву у невеликі зрозумілі описи (наприклад, середня академічна успішність, коли відбувається усереднення всіх балів для

забезпечення загального розуміння особистого середнього балу). Також, цей аналіз дозволяє відкинути менш важливі дані, визначити подібність між змінними, та знайти помилки, на відміну від інших методів, що більш зосереджені на обчисленні закономірностей.

Діагностичний аналіз (Diagnostic analysis) – дозволяє визначити, які конкретно фактори стоять за трендами, подіями, явищами. Розширює собою описовий аналіз, бо також працює на основі попередніх даних. Відрізняється ж тим, що він прагне виявляти та пояснювати випадки-аномалії та викиди.

Цей вид аналізу дозволяє застосовувати нестандартний погляд на вплив зовнішніх факторів на закономірності даних, знаходження нових джерел інформації, порівнянні зроблених висновків з набором даних. Його ще називають аналізом першопричини завдяки його акцентуванні на причинах та їх наслідках.

Діагностичний аналіз для своєї роботи використовує різні техніки:

1) деталізація даних – для отримання детальнішої інформації про аспекти впливу на події, надаючи різні види даних у звітах, діаграмах і електронних таблицях. Найчастіше застосовується для вивчення детальної інформації, яка лежить за підсумковими даними, показаними у звітах;

2) «data mining» – для пошуку значних обсягів даних, щоб виявити найпоширеніші фактори та асоціації в них. Збір даних можна здійснювати власноруч, чи автоматично за допомогою машинного навчання. Він є важливою частиною успішної аналітики;

3) кореляційний аналіз – для перевірки того, як сильно змінні пов’язані між собою, тобто, обчислює рівень зміни однієї змінної через зміну іншої. Висока кореляція дає зрозуміти про сильний зв’язок між двома змінними, а низька кореляція означає, що змінні слабко пов’язані між собою.

1.3.5. Трактування результатів.

Після проведення аналізу даних необхідно оцінити результати, або ж надати їх особам, що будуть відповідати за це. Саме тут приходить на допомогу візуалізація цих даних, яка є найкращим методом для розуміння складних зв’язків, що були присутні всередині масиву всіх проаналізованих даних.

Візуальні елементи дозволяють перетворити інформацію для людини у такий формат, що дасть змогу швидше опрацьовувати необхідне. Такий підхід, також, може допомогти вирішити деякі бізнес-проблеми вчасно даючи відповіді, порівняно з аналізом звітів та електронних таблиць.

Графічні елементи допомагають визначити закономірності та зв'язки, що відображаються поміж даними. Також, можуть допомогти виявляти тенденції ринку, що можуть бути використані у своїх інтересах, і стануть перевагою над конкурентами не залежно від сфери роботи. Дозволяють вплинути на вирішення деяких потенційних проблем, перш ніж вони перетворяться на справді великі проблеми для компанії.

При застосуванні правильного типу візуалізації і її належного застосування, можна помітити деякі викиди, що час від часу появляються в наборі даних. Такі викиди можуть спровокувати відтік клієнтів, прибутку, вплинути на якість виробів.

Етап візуалізації часто повинен враховувати фактор кардинальності стовпців, які мають бути реалізовані. Якщо кардинальність висока, то буде багато унікальних значень, якщо ж вона низька – стовпець містить повторювані значення (наприклад, коли вказано дані про стать людини).

РОЗДІЛ 2
МОВА ПРОГРАМУВАННЯ PYTHON ТА ЇЇ БІБЛІОТЕКИ, ЯК ЗАСІБ ДЛЯ АНАЛІЗУ ДАНИХ

Вже стало звичною справою, що мова програмування Python отримала застосування у безлічі сфер. З моменту її створення у 1991 році, досі продовжує набирати популярність завдяки багатьом позитивним факторам, незважаючи на деякі пов'язані складнощі. Її можна з легкістю інтегрувати з іншими мовами програмування, як C, C++, Java.

Мова ідеально підходить для початківців, і гарантує, що для деяких потреб зробить написання програмного коду максимально спрощеним, незалежно від того, чи бажаючий вивчати має хоч якийсь досвід у програмуванні. Зарезервовані слова, що використовуються в Python, є зрозумілими – англійськими. Її синтаксис спирається на ідею об'єктно-орієнтованого програмування. Обсяг коду буде максимально компактнішим за інші (код на Python займатиме всього п'яту частину обсягу коду на C++ для виконання одного й того ж завдання). Також, перевагою вважається значна кількість ресурсів для вивчення мови (курси на навчаючих платформах, сайтах, чи YouTube; книги, для вивчення з нуля, або для поглибленого вивчення; сторінки в соціальних мережах, що дозволяють знайти корисні ресурси).

Буде помилковим вважати, що оскільки ця мова розроблена, щоб допомогти початківцям, то вона буде не достатньо потужною для виконання серйозних проектів. Python має багато можливостей і може виконувати складні проекти, що потребують більшої потужності. Відповідні бібліотеки та розширення дозволять займатись математикою, науками, глибоким навчанням, машинним навчанням.

Ця мова може працювати з будь-якими форматами даних, що забажає розробник. Можна без проблем імпортувати таблиці SQL не перетворюючи їх у певний формат, що точно працюватиме. Наприклад, Python працює з такими найпоширенішими форматами даних:

- CSV (comma-separated values) – значення розділені комою, відносяться до типу електронних таблиць. Кожен рядок представляє собою певні значення, які називають записом, що може містити декілька полів, розділених комою.
- XLSX – формат створений Microsoft Excel. В ньому дані розподілені за клітинками та стовпцями на аркуші. Кожен такий файл може містити один, чи декілька аркушів.
- ZIP – архівний файл, який застосовується для збору кількох файлів в один з метою використання меншого місця для зберігання. Цей формат стиснення забезпечує стискання без втрат і може бути повністю відновленим після його розпакування.
- TXT – формат файлу із звичайним неструктурованим текстом, з яким не пов’язані жодні метадані. Він доволі просто зчитується програмами, але його важко інтерпретувати.
- JSON (JavaScript Object Notation) – відкритий текстовий стандарт для обміну даними через всесвітню мережу, що використовується для передачі структурованих даних. Він легко зчитується мовами програмування.
- XML – розширювана мова розмітки з певними правилами кодування даних. Розроблена для надсилання інформації через інтернет і не використовує попередньо визначені теги.
- HTML – стандартна мова розмітки для створення веб-сторінок. Має попередньо визначені теги і не чутлива до регістру.
- PDF (Portable Document Format) – інтерпретує та відображає текстові документи разом із вбудованою графікою і може бути захищений паролем.
- DOCX – формат для текстових даних, який можна редагувати. Його формат можна змінювати на будь-який інший. Має можливість вбудовувати зображення, таблиці, посилання.

- MP3 – найпоширеніший мультимедійний файл для кодування цифрового звуку. Складається з кількох кадрів, що можуть бути розділені на заголовок і блок даних. Заголовок визначає початок дійсного кадру, а блоки містять аудіоінформацію щодо частот і амплітуд. Використовує формат кодування Moving Picture Experts Group – 1.
- MP4 – найпоширеніший формат для зберігання відео та фільмів. Він містить кілька кадрів, які відтворюються за певний період часу. Може містити субтитри.
- Зображення – складаються з пікселів та пов'язаними з ними метаданими. Кадри складаються з двовимірного масиву значень пікселів, які можуть бути будь-якої інтенсивності. Метадані, пов'язані із зображенням, можуть бути типом зображення або розмірами в пікселях [2].

Коли йде мова про Python, то можна побачити, що традиційна бібліотека має більш ніж достатньо функцій, які знадобляться для виконання проектів, і не потрібно покладатись на зовнішні бібліотеки. Бібліотеки Python – це набір класів і модулів, що можуть бути використані для виконання конкретних завдань, усуваючи необхідність їх написання з нуля. Вони дозволяють зосередитись на вирішенні реальних проблем за рахунок повторного використання коду, що вже був створений іншими людьми для своїх потреб.

Python має доволі велику спільноту, що зосереджена в усіх частинах світу, і з різним досвідом програмування. Це є великою перевагою для початківців – наявність спільноти забезпечує корисні поради, відповіді на запитання, полегшення у вирішення проблем, з якими можна зіткнутися, коли мова заходить про певні алгоритми.

Проте, Python не стане вдалим рішення абсолютно для всіх потреб. Беручи до уваги те, що ця мова інтерпретована, і виконується поступово за рядками, то програми працюватимуть значно повільніше, ніж в компільованих мовах, де попередньо весь код перетворюється у машинний. Також, тестування в цій мові

програмування дещо складніше, бо всі помилки виявляються під час виконання, викликаючи синтаксичні помилки.

2.1. Бібліотеки для аналізу даних.

Згідно з опитуванням, мова програмування Python займає третє місце серед найбільш використовуваних мов програмування, і часто стає вибором в галузях аналізу даних, глибокого навчання, наукою про дані, та машинним навчанням. Це зумовлено і чималою кількістю бібліотек з якими може працювати фахівець. Бібліотеки допоможуть обробляти проекти з наукою про дані, і потрібно просто переконатись, що обрано саме ті, котрі працюватимуть для конкретної моделі, чи алгоритму.

Без бібліотек важко виконати певні необхідні завдання, що пов'язані з даними, а робота з передбаченням необхідних результатів може стати практично неможливою. Кожна з них дещо відрізняється від інших і виконує частину роботи іншим методом. Деякі краще підходять для збору даних, а інші для їх аналізу. Тому, перш за все, потрібно ознайомитись з найпопулярнішими бібліотеками (рис. 2.1), та їх можливостями, щоб розуміти, яку обирати під потреби проекту.

Рис. 2.1. Найпопулярніші бібліотеки мови Python для аналізу даних.

NumPy (Numerical Python) – одна з найкращих для завантаження бібліотек з відкритим кодом, що зосереджена на математичних обчисленнях та обробці масивів. Бібліотека містить функції лінійної алгебри; об'єкт багатомірного масиву; перетворення Фур'є, розширені можливості випадкових чисел; засоби для запису та зчитування набору даних у вигляді масивів; інструменти для інтеграції з іншими мовами програмування низького рівня. NumPy також сумісний з іншими бібліотеками мови Python, та є невід'ємною частиною багатьох завдань, що виконуються під час аналізу даних, слугуючи основою для багатьох функцій і типів даних.

Python підтримує структури даних тільки у одному вимірі, тому, списки, кортежі та словники є одновимірними контейнерами даних. Вбудовані типи також не піддаються складним обчисленням. Це стає обмеженням для певних завдань, що пов'язані із аналізом даних, та створенням моделей, які роблять масиви надзвичайно необхідною структурою. Бібліотека NumPy дозволяє частково вирішити проблему з повільним виконанням складних математичних алгоритмів, пропонуючи багатовимірні масиви та функції, які будуть ефективно працюють з масивами, змінюючи код, який відноситься до внутрішніх циклів [3]. Виміри NumPy звуть осями, а клас масиву називають ndarray.

SciPy (Scientific Python) – бібліотека мови Python з відкритим кодом і побудована на основі NumPy (тобто, масиви для роботи в SciPy надаються завдяки бібліотеці NumPy), проте, містить більш функціональні версії модулів лінійної алгебри. Вважається однією з найкорисніших бібліотек для оптимізації, інтеграції, статистики, інтерполяції, алгебраїчних та диференціальних рівнянь.

SciPy надає набір алгоритмів високого рівня для маніпулювання та візуалізації даних, а об'єднавши його з NumPy дозволяє виконувати обробку зображень. Бібліотека здатна надати більше впливу на інтерактивні сеанси Python завдяки високорівневим командам для обробки даних, та візуалізацією для користувача. Інтерактивний сеанс Python в поєднанні з SciPy стане середовищем, що набуває рівня технологій обробки даних.

Pandas – бібліотека в якій присутні висока продуктивність засобів роботи з масивами, та можливості маніпулювання даними, які властиві реляційним базам даних, чи електронним таблицям. Застосовується, коли потрібно полегшити роботу із збором даних, їх сортуванням, очищенням і обробкою. Бібліотека спеціалізується на обробці числових таблиць і часових рядів, що часто зустрічаються у науці про дані. Pandas побудований на основі NymPy і застосовується в SciPy для статистичного аналізу, а в Matplotlib для побудови функцій.

Pandas постачається одразу із Python і розроблений для максимального спрощення роботи з реляційними даними. Також, ця бібліотека справно проводить швидку візуалізацію даних, маніпулювання та агрегацію даних, і решту завдань з галузі науки про дані.

Matplotlib – найпопулярніша бібліотека для візуалізації даних і графіків серед користувачів мови Python. Дозволяє побудувати велику різноманітність графіків, починаючи з гістограм, до лінійних та теплових графіків за допомогою лише кількох рядків програмного коду[4].

Візуальні елементи спрощують роботу із складними зв’язками, коли відбувається робота з даними. Тому, Matplotlib буде одним з найкращих інструментів для створення та обробки як простих, так і складних візуальних елементів за короткий термін. Проте, ця бібліотека доволі низькорівнева, і для досягнення візуальних ефектів вищого рівня потребує написання додаткового коду [3].

Бібліотека Matplotlib може бути визначена своєю універсальністю – застосовується у різноманітних оболонках. Може використовуватись для полегшення форматування візуальних елементів. Досвідчений користувач зможе налаштовувати дизайн діаграм зважаючи на свої вподобання за допомогою об’єктно-орієнтованого інтерфейсу, чи набору функцій.

Scikit-Leran – вважається однією з найзручніших бібліотек, на теперішній час, для допомоги з наступними функціями:

- обробки зображень;

- для вивчення, вдосконалення, та виконання моделей машинного навчання;
- виконання алгоритмів класифікації;
- лінійна та логістична регресії;
- кластеризація (метод k-середніх, спектральна кластеризація).

Бібліотека базується на інших бібліотеках мови Python – NumPy, matplotlib, і SciPy, та додає окремий набір алгоритмів для популярних завдань машинного навчання та аналізу даних. Scikit-Leran є інструментом аналізу даних та інтелектуального аналізу. Він має відкритий код і може використовуватись для розробки керованих і неконтрольованих алгоритмів машинного навчання.

Scikit-Leran зручний у використанні, бо надає стислий та зручний інтерфейс, коли приходить час працювати над найбільш поширеними алгоритмами машинного навчання. Це спрощує впровадження алгоритмів у виробничу систему. Також, бібліотека популярна для наукових досліджень через хорошу документацію, високу продуктивність і гнучкий API. Містить навчальні набори даних, що дозволяє уникнути етапи збору та очищення даних.

TensorFlow – бібліотека, яка була розроблена компанією Google для їх внутрішнього використання у своїх дослідженнях, та розробці власних продуктів. З того часу, став доступним для використання у своїх потреб іншими спеціалістами. Використання бібліотеки можливе для машинного навчання, обробки природної мови, класифікації зображень, створенні нейронних мереж. Однією з переваг можна вважати здатність працювати з розподіленими обчисленнями.

Необхідно зазначити, що TensorFlow надає інструменти та ресурси, які дозволяють навчати та розгортати моделі будь-де, незалежно від того, яку мову чи платформу використовують. Головний API для роботи з цією бібліотекою був розроблений саме для мови Python, проте, існують варіації і для інших мов програмування [5].

Можна виділити п’ять рівнів системи TensorFlow: апаратний рівень, рівень ядра, API низького рівня, API середнього рівня, API високого рівня (рис. 2.2). На

нижньому рівні, апаратному, бібліотека може залучати до обчислюваних ресурсів CPU, GPU, TPU. Наступний рівень є ядрами, що реалізують C++ і можуть працювати на розподілених платформах. На третьому рівні містяться написані на Python оператори, які надають низькорівневі інструкції API. Четвертий рівень містить компоненти моделі, що забезпечують функції втрат, оптимізатори, конвеєри даних, стовпці функцій, і були реалізовані на мові Python. На останньому рівні знаходяться API високого рівня, реалізовані на Python [6].

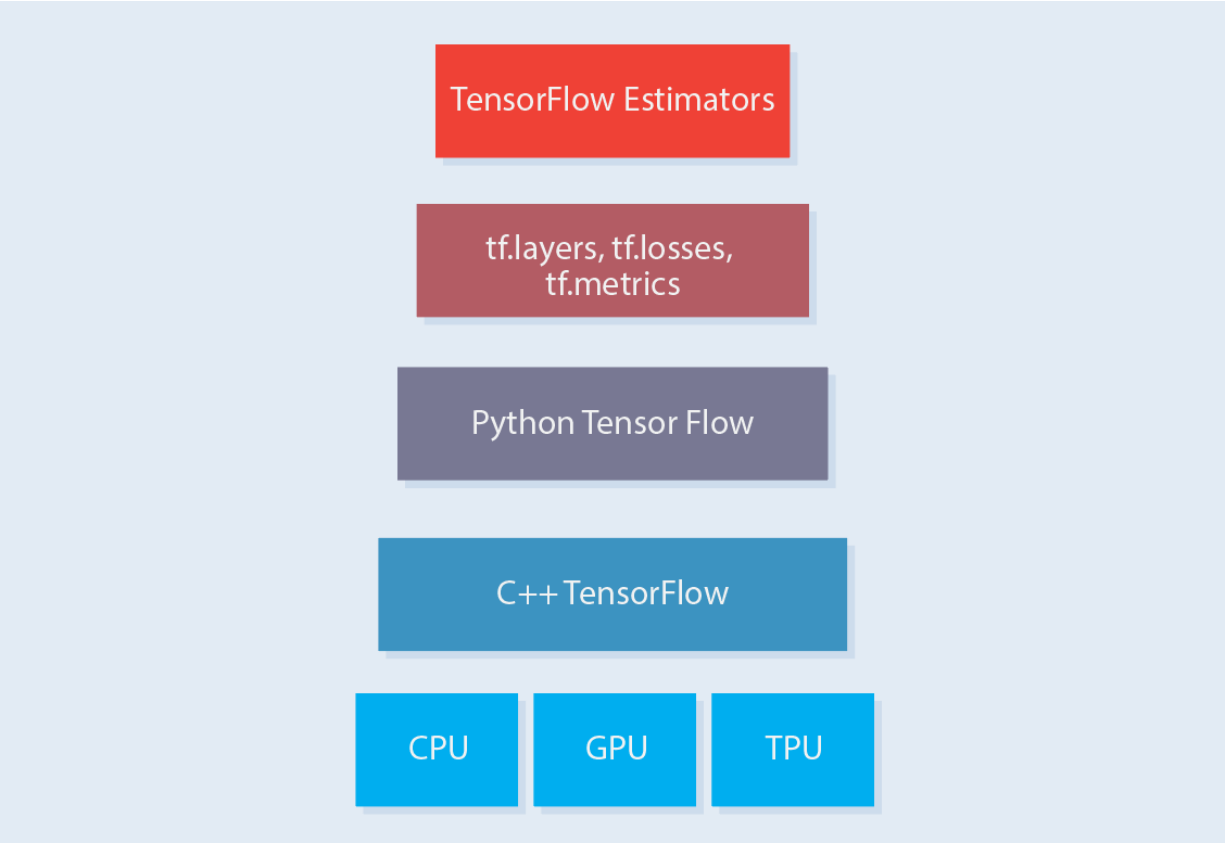


Рис. 2.2. Ієрархія TensorFlow.

Keras – мінімалістична бібліотека з відкритим вихідним кодом, та деякими можливостями вищого рівня, написана на мові Python. Як серверну частину він використовує бібліотеки TensorFlow, або Theano. Його було розроблено під час дослідницької роботи проекту ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System). Бібліотека чудово підходить для роботи з нейронними мережами. Окрім стандартних нейронних мереж підтримує згорточні та рекурентні мережі. [7].

Дизайн Keras максимально мінімалістичний для того, щоб можна було якомога швидше проводити роботу із моделями та переходити від ідеї до результату. Створення мереж використовуючи Keras може здаватись легким навіть для початківців. Незважаючи на легкість використання та простоту, бібліотека вважається достатньо потужною для серйозного моделювання. Ідея Keras базується на багатьох шарах, а що стане необхідним у майбутньому, реалізуватиметься навколо цих шарів. Перший шар відповідає за вхід даних, що підготовлені у тензорах (математичних об'єктах, що не залежать від зміни системи координат), а останній шар відповідатиме за результат [3].

2.2. Вибір середовища розробки.

Програмний код, по своїй суті, є набором тексту з інструкціями для виконання комп'ютером. Саме тому, найпростішим варіантом створення коду буде будь-який ресурс, який здатен реєструвати текст. Для написання та редагування коду, написаного на мові Python, існує велика кількість засобів. Для освоєння базисів мови, чи написання простих програм можна використовувати звичайну оболонку, проте для написання великих програм краще скористатись відповідним редактором чи середовищем, кожне з яких відрізняється від інших унікальними функціями та інтерфейсом [8].

Інтегроване середовище розробки (IDE) є програмним забезпеченням, яке дозволяє використовувати редактор коду, автоматизацію збірки, тестування та налагоджування в одному інтерфейсі, що дозволяє значно максимізувати процес розробки. Надаючи багато інструментів в одному місці звільняє від потреби постійно перемикатись між різними засобами. Певні середовища дозволяють встановлювати додаткові плагіни для потужніших можливостей. Існує два типи середовищ:

- з підтримкою однієї мови програмування (IDLE, PyCharm, PyDev);
- з підтримкою декількох мов програмування (Eclipse, Komodo, NetBeans, Aptana).

Все більш популярними стають хмарні інтегровані середовища завдяки своїй здатності давати доступ до проектів звідусіль. Деякі з них пропонують спільну роботу в команді, та підтримку багатьох мов програмування. Як обмеження, потрібне підключення до мережі [9].

Як недолік, більшість інтегрованих середовищ можуть бути складними у використанні початківцями через переповнення різними плагінами. Також, потрібно звертати увагу на можливості комп'ютера – деякі середовища вимагають багато ресурсів.

Редактори коду нагадують текстові редактори, проте, з особливими розширеними функціями, бо написання коду однозначно складніший процес, ніж написання звичайного тексту. Перевагою редакторів коду є значна гнучкість цих інструментів – не потрібно багато ресурсів для повноцінного використання, і навіть доступна можливість використовувати веб-версію певних з них. Вони здатні виконувати дії для зручності, та підвищення продуктивності розробників:

- різнобарвне підсвічення зарезервованих синтаксичних елементів – дозволяє робити код більш читабельним, та бачити чи допущені синтаксичні помилки;
- автоматичне проставлення відступів – дозволяє не витрачати час на форматування величезних блоків коду;
- завершення синтаксичних елементів – автоматизує написання повторюваних рядків коду, чи парне зіставлення всіх типів дужок, що дозволяє пришвидшити кодування.

Проте, інтеграція кількох потоків в один проект, поміщення власного коду в написаний іншими, чи завдання, які вимагають компіляції можуть викликати проблеми, і тут краще скористатись інтегрованим середовищем розробки [10].

Jupyter Notebook є серверною програмою з можливістю інтерактивного програмування, і використовується різними мовами, основною з яких є Python. Робота з цим інструментом розробки нагадує запис нотаток в звичайні документи: блокнот може містити написаний код, коментарі, візуалізацію, та вихідні дані в одному файлі. Збереження всієї інформації в одному файлі є перевагою з точки

зору його розповсюдження. Перевагою Jupyter Notebook над інтерактивним середовищем Python є змога динамічного редагування коду, і отримання нових миттєвих результатів, що інтерактивний режим вищезгаданої мови не дозволяє [11].

Не можна сказати, що Jupyter Notebook – це редактор коду, чи повноцінне інтегроване середовище розробки. Блокнот забезпечує щось середнє між швидким і легким у використанні редактором, та багатофункціональними IDE. Із Jupyter можна швидко працювати, і швидко отримувати результат, а дослідження та очищення даних виконується в найзручнішому вигляді [12].

Документи у Jupyter Notebook містять вхідні дані, результат, та додатковий текст, що не призначений для виконання. Це дає змогу чергувати код з пояснювальним текстом, проводити контроль версій, ділитись результатами з командою, оскільки ці документи є внутрішніми файлами JSON. Також, можливий експорт у формати HTML, LaTeX, PDF, чи слайди презентації. Будь-який документ Jupyter Notebook може відображатись як статична веб-сторінка за загальнодоступною URL-адресою, якщо поділитись нею через «nbviewer» [13].

Jupyter є доволі популярним серед спеціалістів у сфері аналізу даних, машинного навчання, науки про дані, роботи з статистикою. Стає корисним у всіх напрямах, коли необхідно отримувати проміжкові результати поетапно, або ж постійно працювати з візуалізацією результатів. При повторному відкритті файлу буде збережено не тільки весь написаний код, а й усі запущені результати, що є перевагою, коли проводиться аналіз математичних операцій.

Як недоліки такого середовища можна визначити те, що написані частини коду мають дуже складні глобальні взаємозв’язки, а при редагуванні значень можна легко заплутатись звідки брались дані. Також, при перенесенні коду з Jupyter Notebook в інше середовище потрібно буде витратити додатковий час для налаштування і переписування коду[11].

Спробувати роботу з цим середовищем можна без попереднього встановлення, обравши на офіційному сайті відповідну опцію. Найпростішим варіантом встановлення є через термінал наступною командою **$ pip3 install**

jupyter. Щоб розпочати роботу з блокнотом потрібно прописати в терміналі команду **$ jupyter notebook**. Після цього, у браузері відкриється вкладка з середовищем, де можна створювати нові блокноти, відкривати та завантажувати попередньо створені.

Новий документ Jupyter Notebook складається з послідовності чарунок. Одна чарунка – це багаторядкове поле для вводу тексту. Поведінка виконання написаного у чарунці визначається типом цієї чарунки. Тип чарунки можна визначити за допомогою комбінації на клавіатурі, чи через панель інструментів, і їх є три: code, markdown і raw. За замовчуванням, присвоєно саме тип code [13].

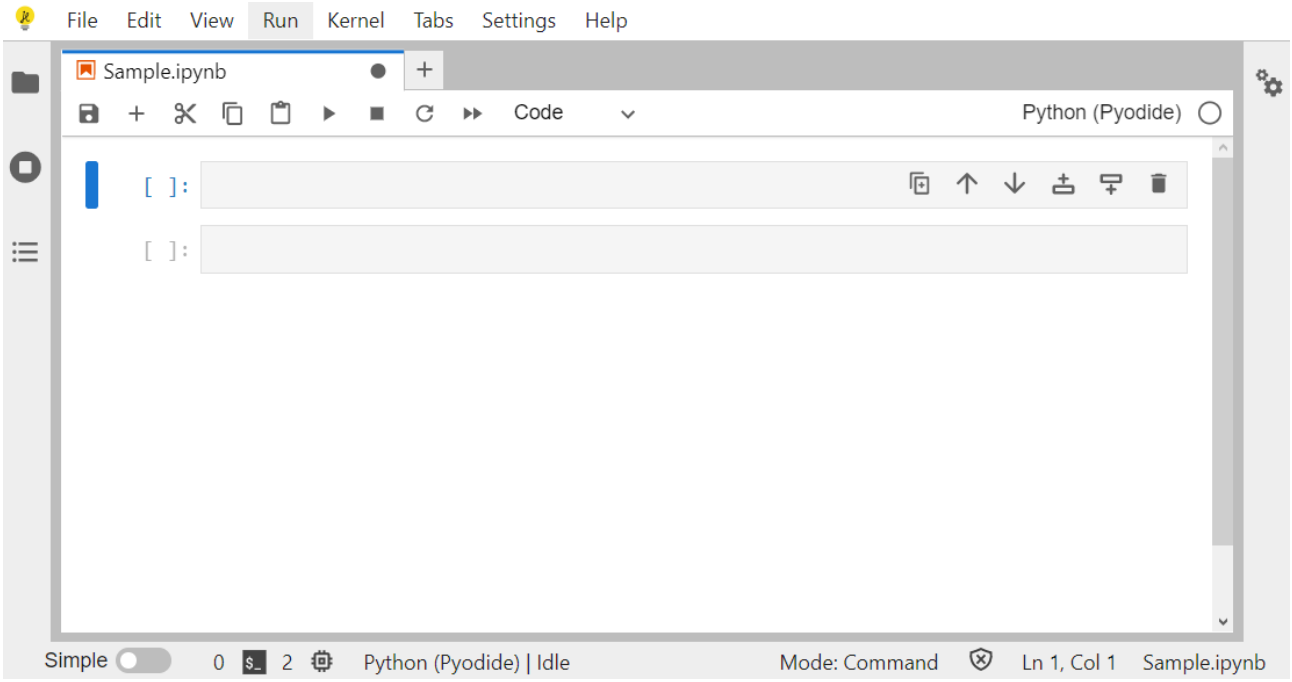


Рис. 2.3. Інтерфейс Jupyter Notebook.

Для того, щоб виконати написаний у чарунці програмний код, можна скористатись трьома методами: 1) через список меню; 2) кнопкою на панелі інструментів; 3) комбінацією клавіш Shift + Enter. Після команди виконання, код надсилається до ядра, а результати відображаються в новій вихідній чарунці. Ядро обирається в залежності від мови програмування, за замовчуванням, для Python – це IPython, і визначає яким чином буде виконуватись код . Не весь код може давати результати, тому не всі чарунки матимуть вихідні дані. Дані можуть

бути різного типу – від звичайних числових та текстових, до таблиць, графіків, діаграм, зображень.

Чарунки типу markdown не містять код, і дозволяють додавати коментарі чи текстове оформлення у процесі роботи. Це досягається завдяки використанні спрощеної мови розмітки Markdown. Мова має максимально простий, обмежений синтаксис, і не є заміною для HTML. За допомогою синтаксису, будь-який текст може бути відформатований (наприклад, застосування жирного накреслення, курсиву, списків та таблиць). За допомогою Markdown можна створювати у блокноті повноцінні інтерактивні статті. Для впорядкованої структури документу можна використовувати заголовки розмітки, які складаються від одного до шести символів решітки та назви заголовку. Один символ решітки відповідає за заголовок першого рівня, два – другого рівня, і так до шостого.

В блокноті доступна можливість інтроспекції об’єктів – отримання загальної інформації про конкретний об’єкт, написавши після змінної знак питання. За допомогою цього можна отримати тип, методи, атрибути, рядки документації (якщо вона існує). Якщо ж додати два знаки питання, то можна отримати вихідний код функції [14].

Середовище Jupyter Notebook має за мету максимально полегшити контроль над поведінкою проектів, та вирішення поширених проблем. Тому, існують командні утиліти, які пропонуються ядром IPython, і звуться «чарівними». Існують два типи цих команд: однорядкові – позначаються одним символом відсотку; багаторядкові – позначаються двома символами відсотку. Використовуючи команду %automagic можна увімкнути автоматичний режим, що дозволить використовувати «чарівні» команди навіть без знаку відсотку. Проте, таке підходить тільки якщо не визначена однойменна змінна, бо команди мають найнижчий пріоритет і втрачають свою видимість. Результат деяких функцій може бути присвоєний змінній. Всі команди можна знайти ввівши команду %magic, або в офіційній документації, а список найчастіше використовуваних подано у таблиці 1.

Таблиця 1.

Найбільш використовувані утиліти IPython

Командна утиліта	Виконувана дія
%%html	перетворює вміст чарунки на html-сценарій
%magic	відображення документації про всі утиліти
%%timeit	відображення часу виконання
%hist	відображення історії прописаних команд
%paste	виконати відформатований код з буферу обміну
%reset	видалити всі визначені змінні з інтерактивного простору
%run	виконати Python
%pinfo	відображає детальну інформацію про об’єкт
%who	відображає список всіх глобальних змінних

Утиліта, яка робить використання IPython надзвичайно популярним у сфері аналізу та візуалізації даних – це, безумовно, %matplotlib. Вона забезпечує створення графіків та діаграм у чарунках блокноту, а не за межами блокноту, чи через окремі вікна браузеру (рис. 2.4).

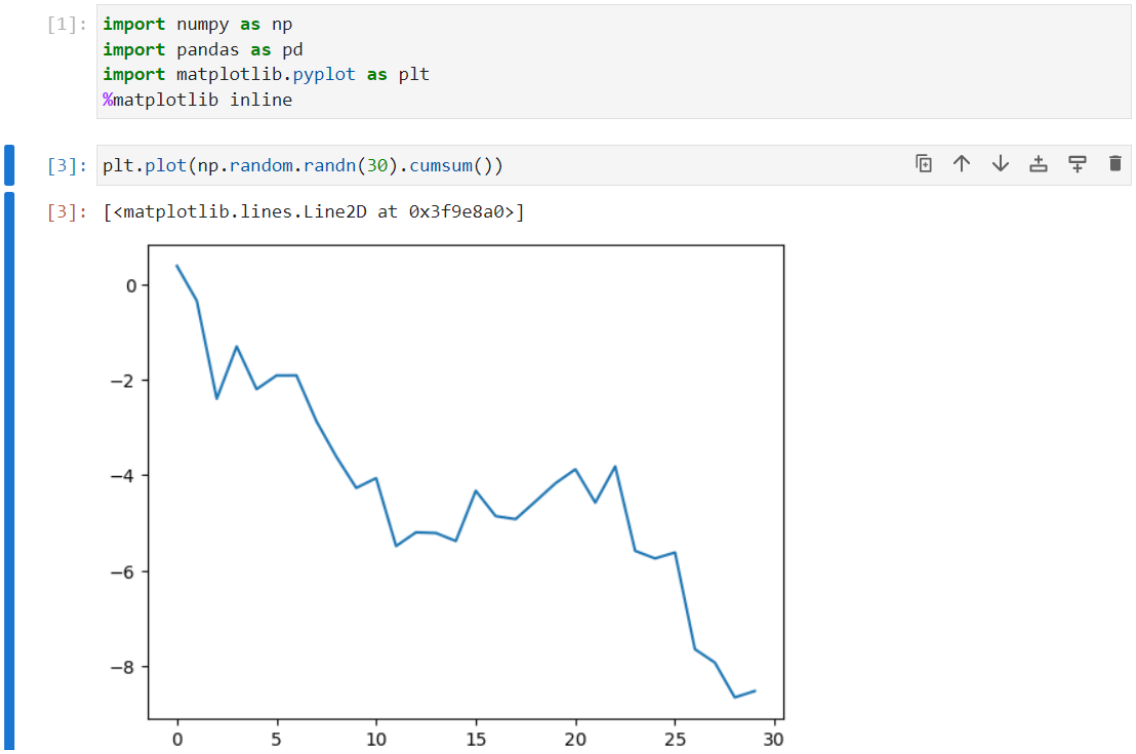


Рис. 2.4. Створення візуалізації утилітою matplotlib у Jupyter Notebook.

2.3. Структури Pandas і NumPy як засобів для візуалізації даних.

Візуалізація надзвичайно важлива для виявлення закономірностей, викидів, збору та розуміння необроблених даних, і представлення висновків для зовнішньої аудиторії [4]. Вона дозволяє перенести всю кількісну інформацію в повноцінну графічну форму. Візуалізація даних є дуже поширеним явищем у повсякденних сферах. Найпоширенішими різновидами діаграм є наступні:

- bar chart – пропорційні відносно значень стовпці, що показують порівняння між окремими категоріями. Можуть бути направлені як вертикально, так і горизонтально. Часто застосовується для демонстрації тенденцій.
- histogram – стовпці, які показують частоту появи кожного значення в наборі даних. Завжди використовується для чисельних даних.
- boxplot – стандартизований спосіб відображення розподілу даних на основі підсумку мінімуму, першого квартилю, медіани, третього квартиолю і максимуму. Застосовується в описувальній статистиці. Показує чи згруповані дані, та на скільки вони симетричні.
- scatter plot – точки, які описують значення двох числових змінних. Використовується для спостереження за зв’язками між змінними
- pie chart – діаграма, яка зображує коло, розбите на пропорційні відносно значень сегменти. Сегменти представляють відношення кількісті або відсотку однієї частини до інших.
- heat map – діаграма, в якій представлення даних відбувається використовуючи системи кольорового кодування. Дозволяє візуалізувати найбільш важливі області з набору даних.

NumPy – призначений для чисельних операцій та наукових розрахунків. Спроектований саме для комфортної роботи з великими масивами даних. Не потрібно застосовувати цикли для перебору кожного елементу масиву, бо всі методи і математичні операції орієнтовані на масиви цілком. Його алгоритми працюють в сотню раз швидше аналогічних, що написані на чистому Python, без застосування бібліотеки, і не потребують так багато ресурсів. Об’єкт ndarray –

репрезентація n-мірного масиву, який являє собою контейнер для однорідних структур даних, тому, на відміну від списків і словників, повинен містити елементи тільки одного типу даних.

Найпростішим методом створення масиву в NumPy – через списки або кортежі методом **np.array()**. Окрім цього методу, існують стоворення на основі послідовностей чисел, послідовностей списків, згенерованих випадкових чисел, нулів та одиниць (коли ще не відомі дані, але масив вже потрібен). Якщо не перевизначати тип елементів під час створення, то метод самостійно спробує визначити необхідний тип для такого масиву. NumPy підтримує знакові та беззнакові цілі числа, числа з плаваючою комою різної точності, комплексні числа, булевий тип, ASCII-рядки і Unicode-рядки фіксованої довжини.

Matplotlib – бібліотека, що є розширеною частиною NumPy і забезпечує побудову всіх видів діаграм та загальну візуалізацію. Вона пропонує методи, які дозволяють швидше побачити шаблони та тенденції з великих обсягів інформації. Зазвичай використовується для побудови тільки двовимірних графіків і обмеженою підтримкою створення тривимірних графіків.

При створенні діаграм буває необхідність у використанні як загальних команд, так і конкретизованих. Matplotlib дозволяє отримати повний контроль над даними і використовувати команди високого рівня, і мати можливість, при потребі, скористатись низькорівневими. Саме тому, організація бібліотеки є ієрархічною. На найвищому рівні поміщено середовище кінцевої машини де прості функції використовуються для додавання частин графіка до осей малюнка. Наступний рівень є об’єктно-орієнтованим інтерфейсом де використовуються тільки явні створення фігур та об’єктів осей, і в подальшому застосовуються для більшості побудов. Після цього, йде рівень найбільшого контролю, де можна залишити цілком об’єктно-орієнтований підхід без графічного інтерфейсу [15].

Візуальні елементи, у випадку лінійних графіків, створюються шляхом прописання методу **plt.plot()**, який приймає в якості аргументу масив чисел, які відповідають спочатку координаті X, а тоді координаті Y. Якщо ж метод приймає тільки один аргумент, то дані передаватимуться тільки на координату Y, а вісь

абцис спроектується автоматично – це зручно коли діапазон чисел на ній не важливий. Функція **plt.show()** відповідає за відображення графіку. Зберігати всі створені діарами можна за допомогою методу **savefig(назва_файлу.тип_файлу)**.

Діаграму типу boxplot можна зручно візуалізувати методом **plt.boxplot().** На цій діаграмі (рис. 2.5), оранжевими смужками позначено медіани елементів трьох масивів даних – середнього значення більшості елементів. З обох сторін рамками є перший та третій квартель (25 та 75 відсотків відповідно). Вусиками, як правило, є максимальні та мінімальні значення масивів. Червоні цятки на діаграмі (встановлення кольору та тип маркеру було визначено словником red_circle) позначають викиди – значення, що знаходяться за межами статистичної важливості [16].

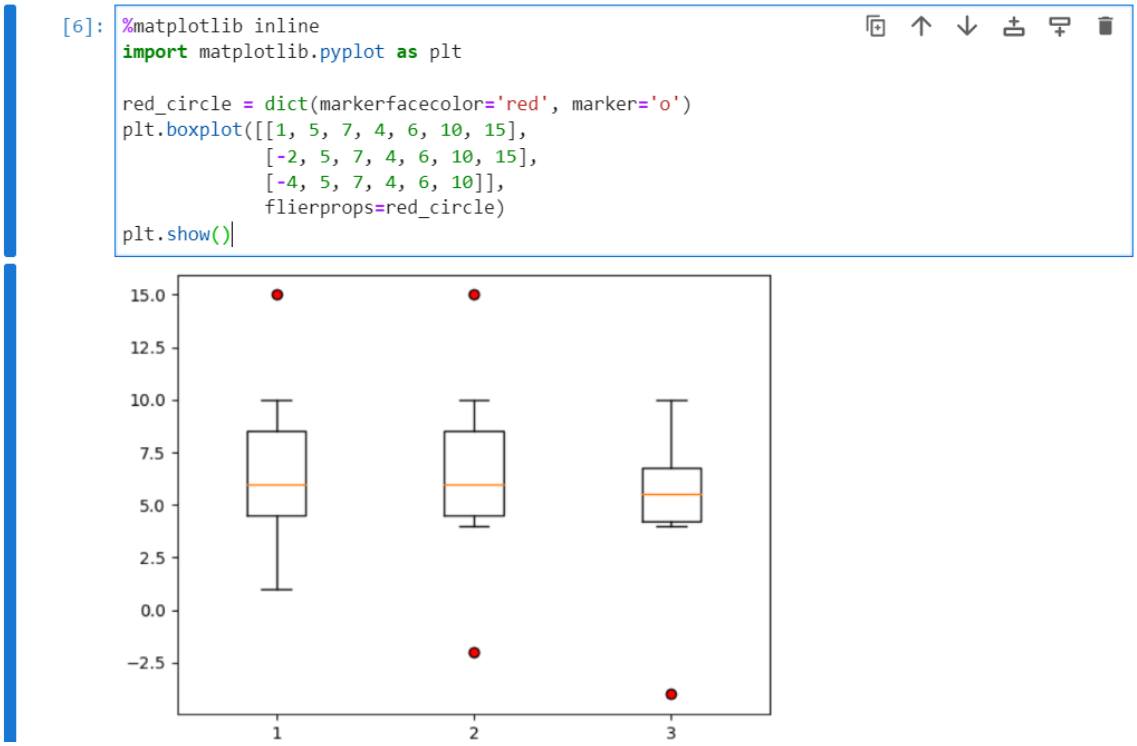


Рис. 2.5. Візуалізація діаграми boxplot методами Matplotlib.

Pandas призначений для інтеграції в безліч наукових середовищ з багатьма іншими бібліотеками сторонніх розробників. Для виконання аналізу даних, користуючись бібліотекою Pandas, необхідно дослідити дві основні структури даних: DataFrame та Series, які здатні обробляти більшу частину випадків використання. Бібліотеку можна розглядати як вмістилище для даних все менших

розмірів. DataFrame – це контейнер для Series, а Series – для скалярів. Бібліотека пропонує велику кількість методів для модифікації цих структур.

Series – це одновимірна структура даних, що нагадує один стовпець матриці, і описується тільки індексами рядків. Може містити дані будь-якого типу, наприклад, числа та рядки. Найпоширенішим варіантом створення такого об'єкту є у вигляді списку Python, конструктором **pd.Series ()**.

DataFrame – це двовимірна структура даних, яка складається з рядків і стовпців, та нагадує своєю структурою електронну таблицю. Зазвичай, цей об'єкт є найчастіше використовуваний у Pandas. Найпоширенішим варіантом створення такого об'єкту є через словник Python, який містить списки однієї довжини, конструктором **pd.DataFrame()**. Іншими методами створення є через список словників, структури Series, інші двовимірні структури даних, двовимірні масиви. Ключі словника варто сприймати як заголовки стовпців, а значення в кожному списку як стовпці. Індекс та стовпці вважаються осями, де індексу присвоєно вісь «0», а стовпцям «1».

Важливо мати доступ до даних, що зберігаються в різноманітних форматах, щоб виконувати повноцінний аналіз даних. Для цього Pandas пропонує наступний метод – **pd.read_тип-файлу()**, який в якості аргументу приймає шлях до необхідного файлу, та автоматично формує з нього DataFrame без використання додаткових функцій. Зчитування можливе для таких основних типів файлу: csv, table, excel (з двох сторінок одночасно), hdf, html, json, sas, sql, stata, feather.

Під час використання методу зчитування можна застосувати аргументи для додаткового фільтрування, щоб зробити отримані дані більш читабельними та придатними для використання:

- sep – визначення роздільника в рядках. Може бути символом, послідовністю символів, регулярним виразом.
- header – визначення рядку, який містить імена колонок. Без додаткових аргументів буде прирівнюватсь до першого рядку, якщо він не пустий.

- names – визначення списку імен стовпців, що сторюватимуть рядок заголовку. Не може містити в собі дублікати.
- index_col – визначення колонки, яка виконуватиме роль індексації рядків. Якщо наступні рядки містять менше стовпців, ніж перший рядок, вони заповнюються NaN.
- nrows – визначення кількості рядків для читання. Застосовується для обробки фрагментів даних з дуже великого об’єму.
- chunksize – визначення кількості рядків при ітерації (читання невеликими порціями дуже громіздких файлів).
- encoding – визначення кодування даних.

Для візуалізації Pandas використовує Matplotlib, але користувачі можуть застосовувати для цього і внутрішні функції. Функція **plot()** дає змогу створювати різні види діаграм, встановлюючи відповідне значення параметру kind. Для прикладу, точкова діаграма формується таким методом – **df.plot(kind='scatter', x='a', y='b')**. В цьому випадку, параметри x та y відповідають за осі. Якщо точкова діаграма надто щільна, то пропонується використання шестикутної точкової діаграми **hexbin**. Формується така діаграма аналогічним чином і в якості аргументу може приймати аргумент **gridsize**, який контролює кількість шестикутників за напрямком осі x. Результати таких візуалізацій можна побачити на рисунку 2.6.

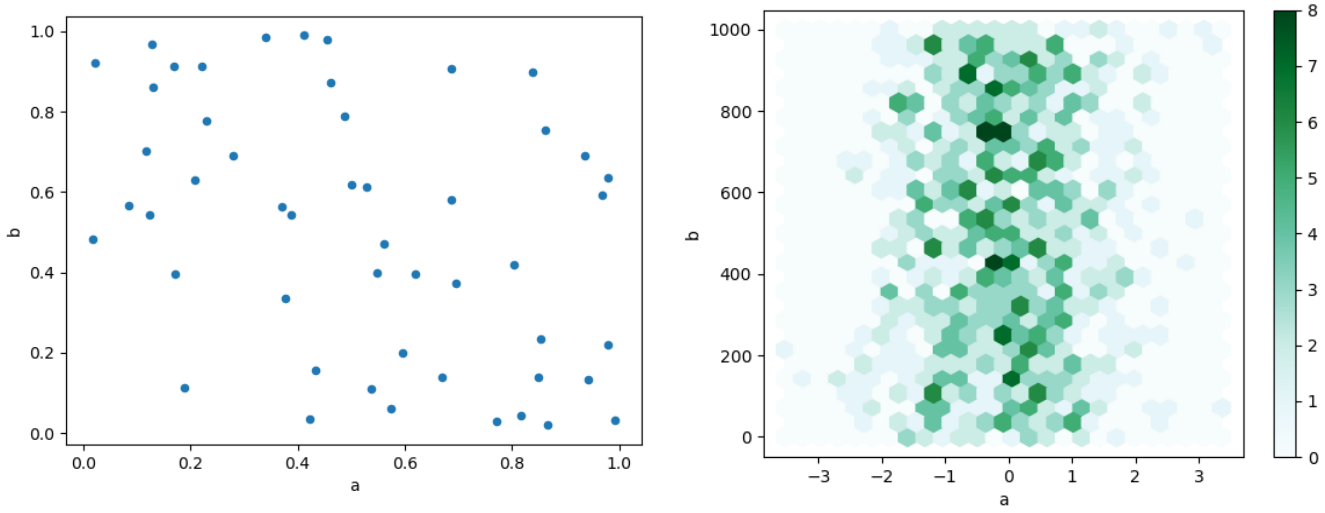


Рис. 2.6. Візуалізація внутрішніми функціями Pandas.

РОЗДІЛ 3
АНАЛІЗ ВПЛИВУ ТЕХНІЧНИХ ХАРАКТЕРИСТИК НА ВАРТІСТЬ НОУТБУКІВ

Датасет – це підготовлений ряд статистичних даних, які, охоплюють одну конкретну тематику і всі елементи взаємопов'язані. Термін зазвичай асоціюється з аналізом даних, машинним навчанням, глибоким навчанням, і штучним інтелектом. У сферах науки та аналізу даних дозволяють зберігати, аналізувати і регулярно використовувати в своїх роботах інформацію. Датасет може бути табличним – містить структуровані дані, що упорядковані за рядками і стовпцями, та нетабличним – всі дані містяться в дужках і є неструктурованими. В залежності від збережених даних, визначається тип датасету:

- числовий – містить тільки інформацію, яка виражається числами;
- категоральний – містить тільки характеристику об'єктів і може бути дихотомічним (кожна змінна має лише одне з двох значень, наприклад, істина і хиба) або політомним (кожна змінна має більше можливих значень);
- двовимірний – містить набір тільки з двома типами змінними для проведення аналізу зв'язків між ними;
- багатовимірний – містить набір більше двох типів змінних;
- кореляційний – містить набір даних, які мають зв'язок одне з одним.

Датасети повністю відрізняються від бази даних, яка за своєю сутністю є набором датасетів. База даних зазвичай є більшою і може охоплювати ширший діапазон інформації, а для маніпулюванням нею потрібні складні комп'ютерні системи. Існує багато категорій баз даних, і вони поділені за методами організації, розміщення і зберігання даних.

Для аналізу було обрано відкритий дата сет з ресурсу Kaggle, що наповнений загальними відомостями та технічними характеристиками ноутбуків, які куплялись найчастіше за останні два роки. Інструментами виконання є мова Python останньої версії, і її бібліотеки (pandas, numpy, matplotlib, seaborn) для маніпулювання та візуалізації даних.

Датасет містить колонки з наступними компонентами:

- Company – виробник ноутбуків (тип колонки String).
- Product – серія та модель ноутбуку (тип String).
- TypeName – тип ноутбуку. Існує п’ять типів в залежності від їх розмірів і продуктивності: Notebook – найтиповіший ноутбук з діагоналлю екрану 15.6 дюйми та вагою трохи більше двох кілограм; Ultrabook – ноутбук меншого розміру і ваги, в наслідок чого, поступається продуктивністю; Gaming – більших розмірів та ваги через масивні радіатори для охолодження компонентів, що продуктивніші за звичайні ноутбуки; 2-in-1 Convertible – особливість конструкції даного типу дозволяє використовувати його як планшет завдяки меншим розмірам і сенсорній панелі; Workstation – найважчий тип ноутбуку, який призначений для складних завдань, що вимагають максимум ресурсів.
- Inches – розмір екрану, що описується довжиною його діагоналі і вимірюється в дюймах.
- ScreenResolution – розширення екрану, що є горизонтальним і вертикальним набором пікселів.
- Cpu – модель центрального процесора. Процесор є найважливішою частиною, яка відповідає за кожну дію будь-якої цифрової обчислювальної системи. В середині себе містить тисячі крихітних транзисторів, які контролюють потік електроенергії через інтегральні схеми. Провідними виробниками процесорів є компанії Intel та AMD.
- Ram – кількість оперативної пам’яті. Енергозалежна пам’ять комп’ютера, яка зосереджена на зберіганні даних, що в конкретний момент використовує процесор. При її переповненні сповільнюється робота комп’ютера.
- Memory – кількість пам’яті в HDD або SSD. HDD – енергонезалежні пристрої для зберігання даних, які функціонують за допомогою магнітного накопичувача з швидко обертовими пластинами. SSD –

новітня технологія, що всі дані зберігає в інтегральних схемах. Працює SSD значно швидше, тихіше, використовує менше енергії, і є довговічнішим, що робить його значно дорожчим за HDD.

- GPU – модель графічного процесору. Графічний процесор є електронною схемою, що призначена для обробки даних графічного типу. Може бути двох типів: інтегрованим – вбудованим з центральним процесором; дискретним – встановлений, на окремій електронній схемі, що приєднана слотом PCI Express [17].
- OpSys – операційна система ноутбуку. Вона виконує керування ресурсами і апаратним забезпеченням комп'ютера.
- Weight – вага ноутбуку в кілограмах.
- Price_euros – ціна ноутбуку, що визначена в євро.

Початковим кроком в аналізі даних є імпорт всіх необхідних бібліотек та отримання доступу до даних, що у форматі CSV, методом **read_csv**, і виведення їх у DataFrame. Формат файлу не передбачає отримання явної інформації про тип даних, тому метод **read_csv** виконує виведення типу. Переконатись, що дані відображаються, і їх типи визначено правильно, можна викликом методу **info**. Як видно на рисунку 3.1, метод надає підсумок про кількість рядків, колонок, типи даних, використання пам'яті, і ненульові значення у кожному стовпці. Датасет містить 1303 рядки, що відповідають окремим ноутбукам, і 13 описових колонок.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   laptop_ID         1303 non-null   int64
 1   Company           1303 non-null   object
 2   Product           1303 non-null   object
 3   TypeName          1303 non-null   object
 4   Inches            1303 non-null   float64
 5   ScreenResolution  1303 non-null   object
 6   Cpu               1303 non-null   object
 7   Ram               1303 non-null   object
 8   Memory            1303 non-null   object
 9   Gpu               1303 non-null   object
 10  OpSys             1303 non-null   object
 11  Weight            1303 non-null   object
 12  Price_euros       1303 non-null   float64
dtypes: float64(2), int64(1), object(10)
memory usage: 132.5+ KB
```

Рис. 3.1. Вивід інформації про набір даних.

3.1. Підготовка датасету до аналізу.

Перед виконанням дослідження даних, проводиться дуже важливий етап – підготовки і очищення. Необхідно обробити відсутні дані, перевести назви колонок до нижнього регістру, видалити колонку з порядковими номерами (індексація проводиться автоматично) відповідними методами бібліотеки Pandas. В сфері аналізу даних обробка відсутніх значень (являють собою пусті рядки, або значення Na і Null) зустрічається дуже часто і займає велику кількість часу. Обраний датасет не потребує обробки відсутніх значень, проте, перевірка на наявність дублікатів показала 28 таких рядків. Скинути дублюючі значення можна застосувавши метод **drop_duplicates**.

Щоб зрозуміти, які з колонок потребують додаткових змін, краще використати метод **head**, що дозволяє вивести DataFrame і побачити n-ну кількість рядків з однієї, чи декількох колонок (рис. 3.2).

	company	product	typename	inches	screenresolution	cpu	ram	memory	gpu	opsys	weight	price_euros
0	Apple	MacBook Pro	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	1339.69
1	Apple	Macbook Air	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	898.94
2	HP	250 G6	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	575.00
3	Apple	MacBook Pro	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg	2537.45
4	Apple	MacBook Pro	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg	1803.60
5	Acer	Aspire 3	Notebook	15.6	1366x768	AMD A9-Series 9420 3GHz	4GB	500GB HDD	AMD Radeon R5	Windows 10	2.1kg	400.00

Рис. 3.2. Табличний вивід перших рядків методом head.

Перша колонка, що вимагає змін – screenresolution, яка містить багато детальної інформації про екрани, і її доцільно розділити на три окремі:

1) роздільна здатність екрану – досягається використанням методу **extract** разом із регулярним виразом **\d+x\d+** для зіставлення тексту та шаблону, і виділення окремих цифрових значень з усього рядка;

2) інформація, чи в ноутбука тип матриці IPS Panel;

3) інформація, чи екран ноутбука підтримує сенсорне використання.

Після розподілу даних, первинна колонка screenresolution залишається порожньою і може бути видалена з датасету методом **drop**: df_.drop('screenresolution', axis=1).

З колонок, що наповнені інформацією про оперативну пам'ять та вагу ноутбуків варто винести одиниці вимірювання, і помістити їх до заголовку, а сам тип чарунок перевести в числовий. Це стане значною перевагою для подальшого аналізу. Виконується це методами **replace** і **astype** відповідно: df_['ram'] = df_['ram'].str.replace('GB', '') df_['ram']=df_['ram'].astype(int).

Колонка cpu містить частоту процесора і його модель. Ці дані потрібно розділити на три частини:

1) частота процесора – за допомоггою регулярних виразів видобути числові дані, одиницю вимірювання помістити в заголовок, а зробити тип чарунок числовим;

2) модель процесора;

3) виробник процесору.

Схожа за типом інформації, колонка gpu потребує менших маніпуляцій – тільки видобування виробника графічного процесору: df_["gpu_brand"] = df_['gpu'].str.extract(r"^([\w\-]+)")

Для колонки memory необхідно більшого процесу очищення. Дані вказані з різними одиницями вимірювання і їх всі потрібно конвертувати в гігабайти, а тип чарунок зробити числовим. Також, деякі з рядків наповнені великою кількістю додаткових параметрів. Для детальнішого аналізу краще розподілити наступні частини в чотири окремі колонки:

1) основний тип первинного накопичувача;

2) ємність первинного накопичувача;

3) вторинний тип накопичувача (якщо він є);

4) ємність вторинного накопичувача.

Весь розподіл відбувався через використання методів бібліотеки pandas та регулярних виразів. Після виконання всіх маніпуляцій, датасет став наповнений додатковою кількістю колонок і меншою кількістю рядків (рис. 3.3), саме це дозволяє проаналізувати значно більше окремих частин, та отримати точніші результати впливу компонентів на ціну і один на одного.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1274 entries, 0 to 1274
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   company           1274 non-null   object
 1   product           1274 non-null   object
 2   typename          1274 non-null   object
 3   inches            1274 non-null   float64
 4   cpu               1274 non-null   object
 5   ram               1274 non-null   int64
 6   gpu               1274 non-null   object
 7   opsys             1274 non-null   object
 8   weight            1274 non-null   float64
 9   price_euros       1274 non-null   float64
 10  resolution        1274 non-null   object
 11  touch_screen      1274 non-null   object
 12  IPS Panel_screen  1274 non-null   object
 13  cpu_frq(GHz)      1188 non-null   float64
 14  cpu_brand         1274 non-null   object
 15  memory1_type      1274 non-null   object
 16  memory2_type      208 non-null    object
 17  memory1_capacity  1274 non-null   float64
 18  memory2_capacity  1274 non-null   float64
 19  gpu_brand         1274 non-null   object
dtypes: float64(6), int64(1), object(13)
memory usage: 209.0+ KB
```

Рис. 3.3. Вивід очищеної інформації про набір даних.

3.2. Дослідницький аналіз датасету.

Розпочати варто з визначення найпопулярнішого виробника ноутбуків (рис. 3.4). Для отримання візуалізації, використовуючи елементи Matplotlib, спочатку необхідно визначити клас фігури, що є вмістилищем вищого рівня для осей, – так зване полотно. Полотно може бути тільки одне, або ж декілька незалежних один від одного. На відміну від звичної нумерації в мові програмування Python, що розпочинається з нуля, нумерація полотен розпочинається з одиниці. Вигляду полотен можна надавати невелику модифікацію, застосовуючи наступні параметри: номер полотна; розмір фігури (за замовчуванням, він вказується послідовністю значень в дюймах, проте, можна застосувати й інші одиниці вимірювання); роздільна здатність зображення; колір полотна; колір навколо полотна; наявність рямки та її розмір.

Осі є простими компонентами для створення окремих ділянок на полотні, коли виникає потреба у зображенні декількох різних графіків. Є можливість розміщувати їх на будь-якій частині полотна, і навіть одних на одних. Щоб видалити зайві осі можна скористатись методом **delaxes**.

Зробити кількісну візуалізацію зручно використовуючи тип графіку countplot бібліотеки seaborn. Це дозволить підрахувати кількість записів в конкретній категорії, а кінцевий результат зобразити у вигляді стовпчиків розподілу, схожих на гістограму. Цьому типу графіку можна передавати наступні параметри: вхідні дані; порядок побудови стовпчиків; вертикальний, чи горизонтальний тип орієнтації ділянки; колір елементів; палітра кольорів; насиченість елементів. Додатково, через цикл виводиться кількість елементів над стовпцями, задається їх розташування, розмір та сімейство шрифту.

Додаткове форматування відміток на осях **x** та **y** надає їм потрібний вигляд і забезпечує кращу читабільність даних. Вони можуть бути різного типу та розподілятись за вказаним кратним значенням. як параметри передано ромір шрифту і його нахил, щоб уникнути перекриття одних відміток іншими.

```
fig =plt.figure(figsize=(10,10))
ax = plt.axes()

sns.countplot(df["company"],
              order=df["company"].value_counts().index,
              palette='coolwarm');
target = df["company"].value_counts()
values = target.values

for i,g in enumerate(values):
    plt.text(i-0.25,g+2, f"{g}", {"family":"sans-serif", "size":12})

plt.xticks(fontsize=14, rotation=40)

ax.spines["top"].set_visible(False)
ax.spines["right"].set_visible(False)
ax.set(ylabel='')
ax.set(xlabel='')
ax.set(facecolor = "none")
```

Рис. 3.4. Лістинг визначення розподілу виробників.

Варто визначити, як цінова категорія ноутбуків змінюється, в залежності від бренду його виробника. Для такої мети чудово підійде тип графіку boxplot – відмінний статистичний інструмент для порівняння розподілу кількісних даних між змінними, чи рівнями категоріальної змінної. Вони часто застосовуються для візуалізації в дослідницькому аналізі та науці про дані, щоб продемонструвати діапазон розповсюдженості, викиди, спотвореність даних. Позиція кожного елементу такої діаграми (медіани, двох квартилів, мінімуму та максимуму, викидів) формується шляхом обчислень, відштовхуючись від медіанного значення. Діапазон значень, що визначений прямокутником, називається інтерквартильним діапазоном. Функція **sns.boxplot** може приймати такі вибіркові параметри: вхідні дані; тип орієнтації ділянки; палітра кольорів; максимальна довжина інтерквартильного діапазону. Порівняти результати розподілу діаграми можна на рисунку 3.5.

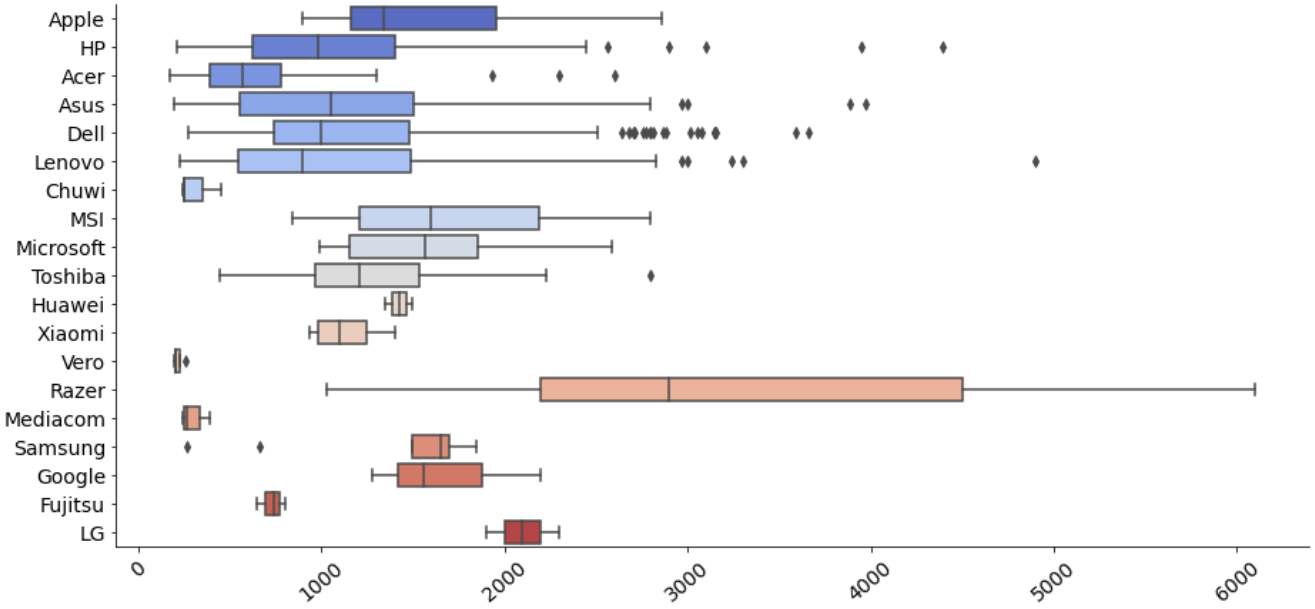


Рис. 3.5. Розподіл вартості ноутбуків в залежності від бренду.

Отже, датасет вмісткість якого складає 1274 рядків, надає дані про дев'ятнадцять виробників ноутбуків. Найпопулярнішими є компанії Dell, HP та Lenovo – 66.5 % від всього датасету. Загальний діапазон цін є в межах від двісті до шести тисяч євро. Середня ціна коливається в межах тисячі євро, а із загальних результатів значно виділяються дві компанії – Razer та MSI. Найнижча середня

ціна у ноутбуків бренду Vero, Chuwi, Mediacom, а найдешевшим серед найпоширеніших брендів є Acer. Ціна вище чотирьох тисяч євро визначена тільки у чотирьох ноутбуків з усього датасету, тому, для отримання точніших статистичних результатів їх краще відкинути.

Датасет містить дані про шість типів ноутбуків. Необхідно визначити, який з них найпопулярніший, а якому віддають перевагу окремі бренди. Розподіл за кількістю та ціновим діапазоном найкраще виконати методами **countplot** та **boxplot** відповідно. Відсоткову перевагу виробництв брендів краще візуалізувати круговою діаграмою, що може відображати тільки один тип даних, а площа її секторів представляє відсоток частин даних.

Генерується кругова діаграма методом **pie**, а як параметри приймає масив значень із даними, список пояснень, аспекти модифікації (додання тіней, повертання сегментів діаграми за годинниковою стрілкою, ширина та колір сегментів, розділяючі лінії). Список пояснень виводиться методом **legend**, і показує дані, що знаходяться на осі **y**, та зразки відповідних їм кольорів з діаграми. Метод **legend** може приймати парамтери кількості стовпців (за замовчуванням один стовпець), координат свого розміщення, зміну розміру та сімейства шрифту.

Порівняти результати розподілу кругової діаграми для перших чотирьох брендів можна на рисунку 3.6. Одразу стає зрозуміло, що певні виробники зосереджені на продажі тільки одного типу ноутбуків (Apple – Ultrabook; MSI – Gaming; Chuwi - Notebook). Також, з діаграми видно, що компанії виробляють більше ноутбуків типу Notebook, ніж будь-який інший тип, тому це дає зрозуміти, що більшість покупців зацікавлені саме в ньому.

Згідно отриманих результатів, ноутбук Workstation має найвищу медіану серед розподілу ціну, що і не дивно для такого потужного типу, завдання якого обробляти складні наукові дані. Найдорожчим виявився тип Gaming, не зважаючи на те, що його середнє значення менше ніж у Workstation. Найпопулярнішим типом ноутбуків є Notebook, що займає більше п’ятдесяти

відсотків даних датасету, а найменш популярним є тип Netbook – в датасеті тільки 25 таких значень. Середня вартість Notebook та Netbook є найменшою.

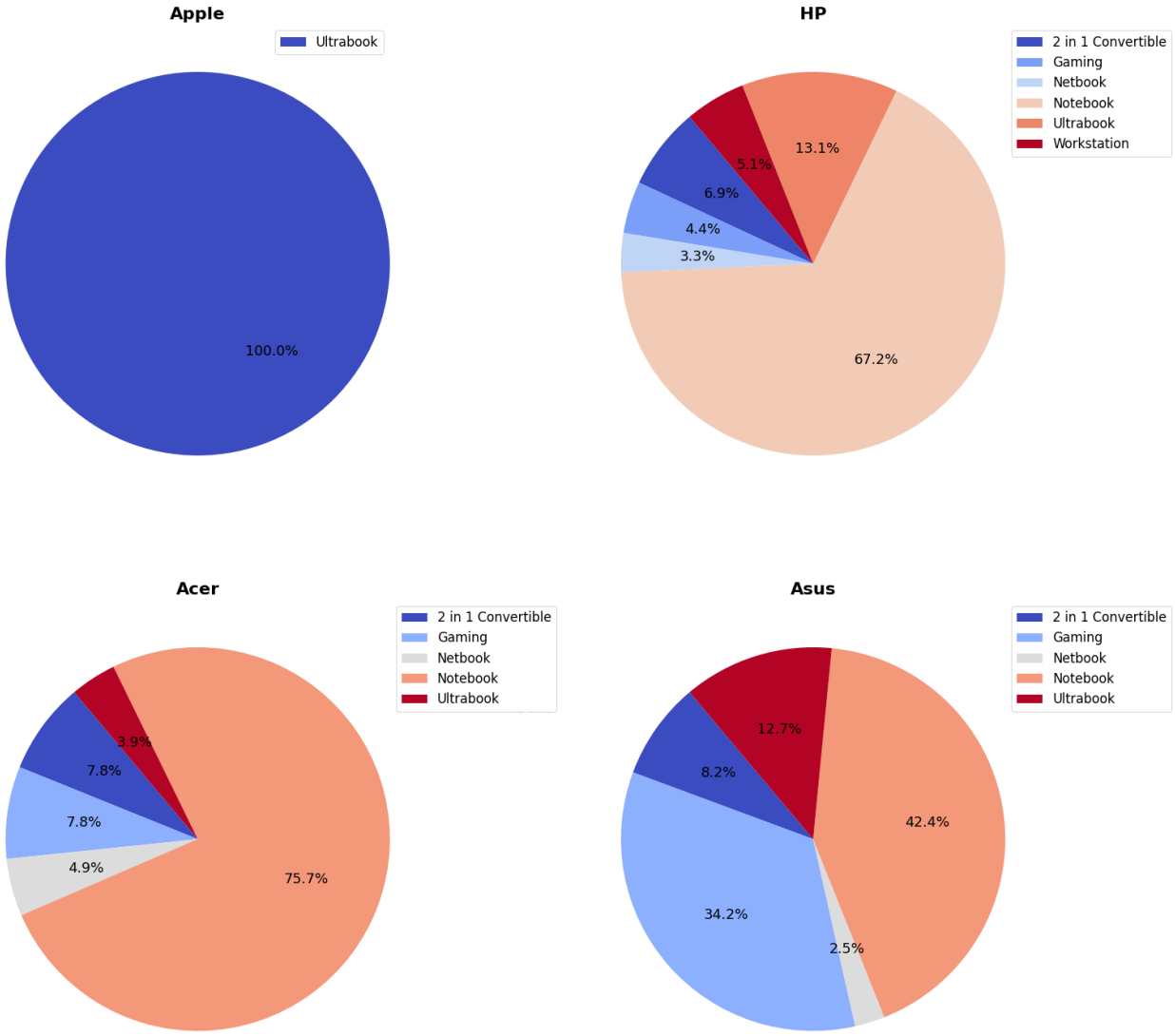


Рис. 3.6. Розподіл найпоширеніших типів проданих ноутбуків брендами Apple, HP, Acer та Asus.

3.2.1. Дослідження розмірів та розширення екрану.

Від екрану ноутбука залежить на скільки буде комфортним процес роботи, тому, можна припустити, що цей параметр матиме достатній вплив на вартість. Найпопулярнішим розміром екрану є 15.6 дюйми – займає 51% із всього набору даних, медіана його ціни є найнижчою, присутній у продукції майже кожної з компаній (рис. 3.7). Наступний за популярністю є екран розміром 14 дюймів, третє місце належить 17.3 дюймам, чия медіана вартості є найбільшою, а четверте 13.3 дюймам. Інші розміри екранів зустрічаються в одиниць компаній.

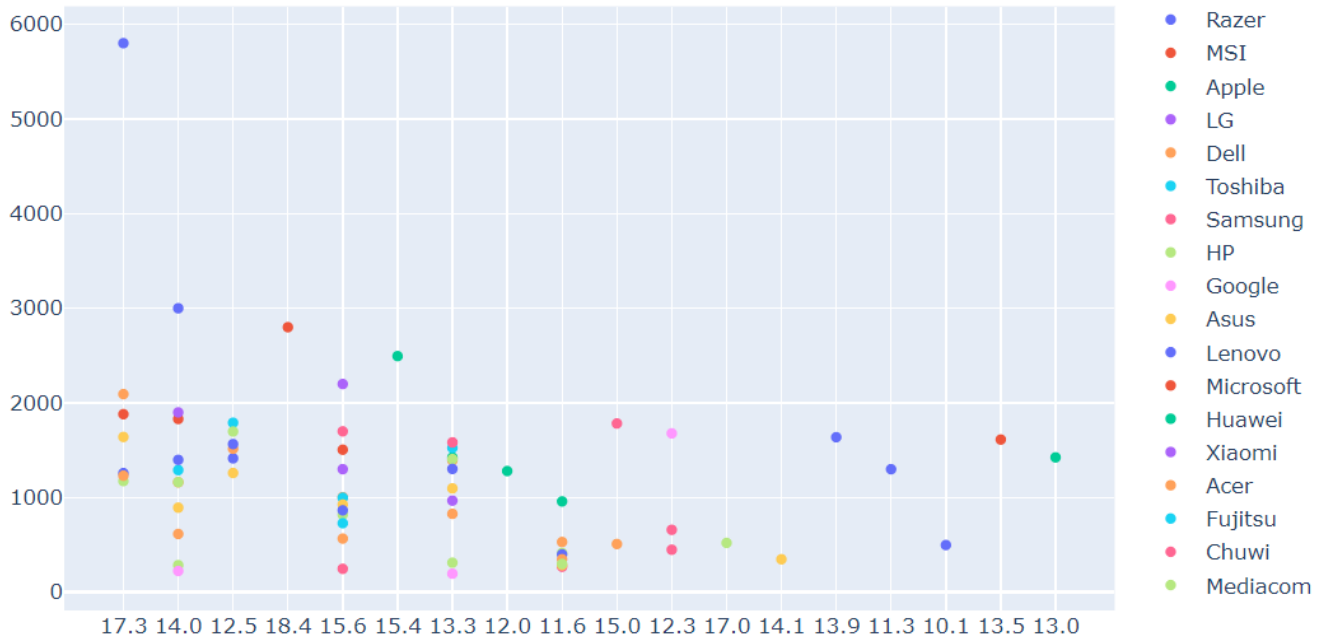


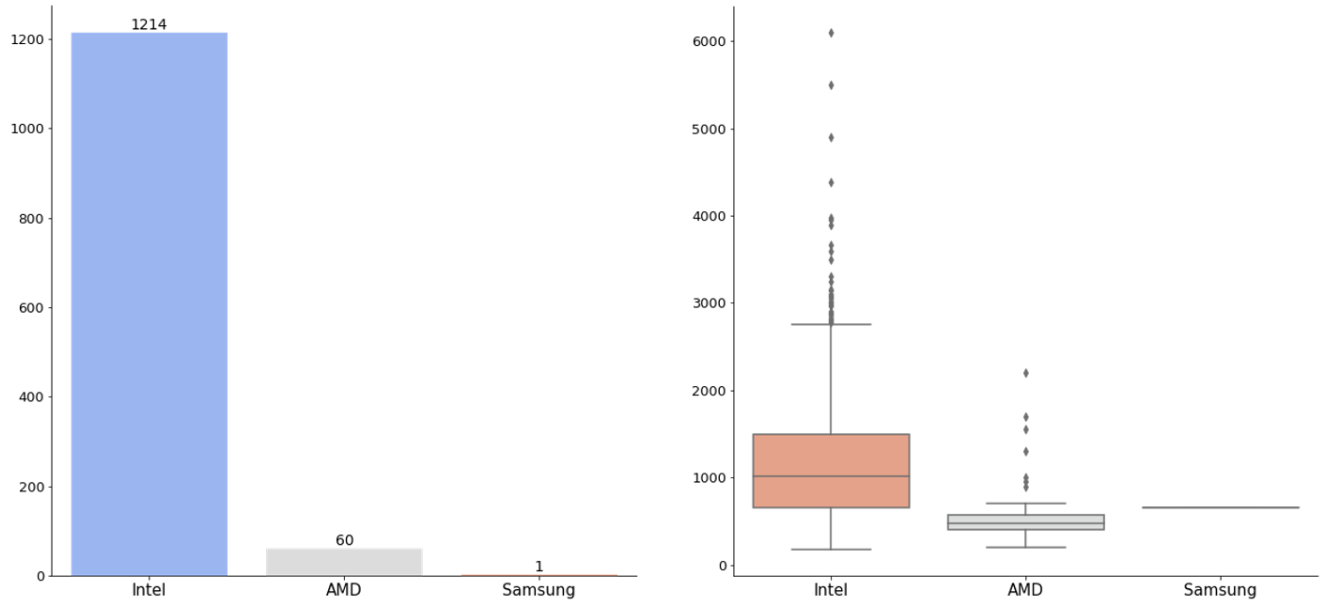
Рис. 3.7. Середня ціна ноутбука для кожної компанії, в залежності від розміру екрану в дюймах.

Роздільна здатність наводиться у форматі «ширина х висота». Чим вище її значення, тим краща якість зображення. У наборі подано п'ятнадцять різних роздільних здатностей, з яких найпопулярнішою є стандартна FullHD (1920 х 1080). Середня ціна ноутбуків з такою роздільною здатністю екрану в межах 1200 євро. Найгіршою є роздільність з розмірами 1366x768, і медіана розподілу її вартості є найменшою. Згідно результатів аналізу, можна отримати висновок про те, що чим вище роздільна здатність екрана, тим вище середня ціна.

3.2.2. Дослідження центрального процесора.

Головними гравцями на ринку центральних процесорів є компанії Intel та AMD. Окрім цих двох гігантів, свою нішу на ринку також займають інші компанії: Samsung, ARM, Qualcomm. Однак, що стосується ринку ноутбуків, то місце лідера із значним відривом займає компанія Intel. Серед проаналізованих варіантів, як видно на рисунку 3.8 (а), центральні процесори від Intel були встановлені в 1214 моделей ноутбуків, а це в цілих двадцять раз більше за кількість ноутбуків на платформі AMD – таких було тільки шістдесят. Також, в наборі даних була одна модель ноутбука із процесором від компанії Samsung,

однак, цю модель можна не враховувати під час підрахунку середніх значень, задля точності дослідження. На рисунку 3.8 (б), зображено діаграму розподілу вартості ноутбуків, які мали в собі центральний процесор тієї, чи іншої компанії. Через надзвичайно велику кількість ноутбуків на платформі Intel, компанія має найбільший діапазон середньої ціни, а також велику кількість викидів. Середня ціна ноутбука, який працює із процесором від Intel, становить тисячу євро. Стосовно AMD, то ноутбуки з даними процесорами мають невеликий діапазон середньої ціни, що ставить дані рішення в сегмент середніх, або бюджетних ноутбуків. Також, як видно на діаграмі, медіана середньої ціни в компанії амд розташована на значенні шестисот євро.



а б

Рис. 3.8. Дослідження центрального процесора.

Проведено аналіз найпопулярніших моделей центрального процесора тієї чи іншої компанії. Однак, перші п’ятнадцять позицій належать моделям процесорів компанії Intel, що і не дивно, враховуючи ті значення, що були представлені вище. Проте, лідирує модель процесора середнього сегменту – це обумовлено тим, що дана модель, через свою невелику вартість та хорошу продуктивність, може розміщуватись в ноутбуках разної цінової категорії. Наступні чотири позиції займають процесори, які відносяться до більш продуктивних рішень. Таку їхню популярність можна пояснити тим, що в даний

момент ноутбуки типу gaming та workstation є досить актуальними, однак, дані моделі досягли такої популярності у використанні саме завдяки gaming ноутбукам. Невисокі температури під час офісних та домашніх задач, а також, висока продуктивність під час ресурсозатратних задач, і роблять дані моделі популярними.

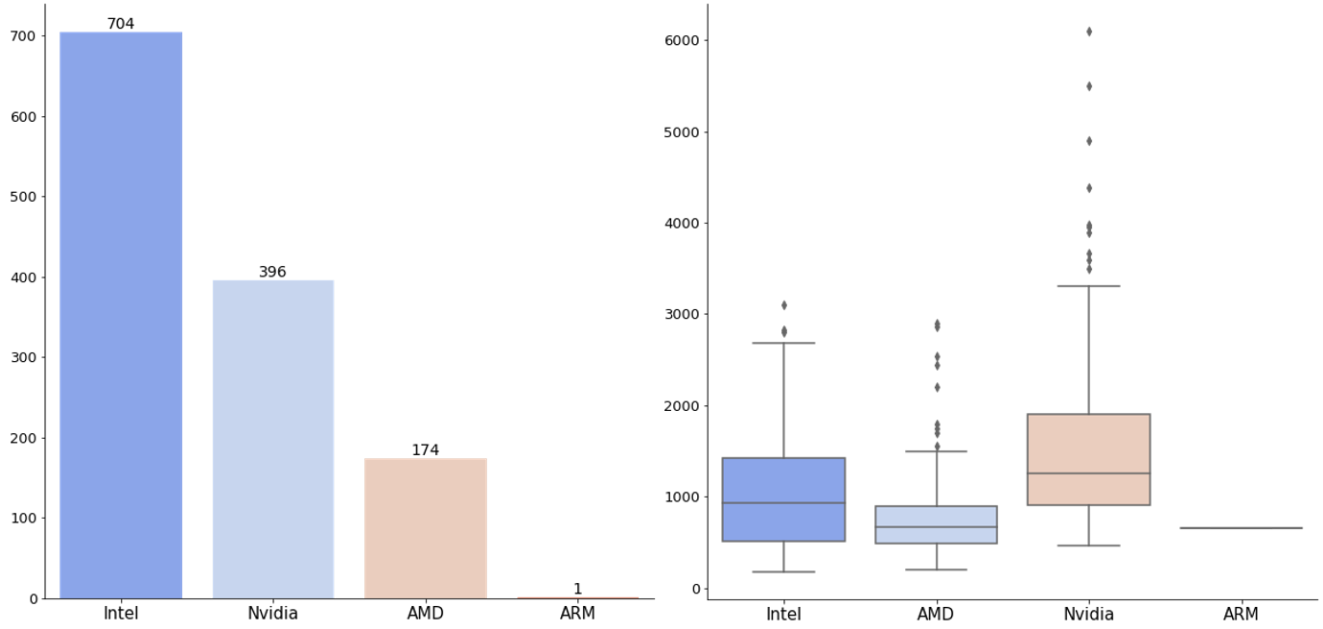
3.2.3. Дослідження графічного процесора.

Як видно на рисунку 3.9 (а), тільки три компанії користуються попитом, коли мова заходить про відео прискорювачі. Серед проаналізованих ноутбуків різного типу, саме Intel лідирує за кількістю встановлених відеоадаптерів. Nvidia та AMD займають другу і третю позицію в цьому списку, зі значним відставанням від Intel. Такий результат обумовлений тим, що Intel, в більшій мірі, виробляє мікропроцесори із вбудованим відео ядром, що дозволяє розробку ноутбуків бюджетного сегменту без дискретного відеоадаптера, а популярність та розміри компанії тільки збільшують частоту використання їхніх продуктів. Також, під час аналізу датасету було знайдено ноутбук із відеоадаптером від компанії ARM, однак він використовується тільки в одній моделі ноутбука, тому дана модель не бралась до уваги задля точності результатів аналізу.

Також, на рисунку 3.9 (б), видно діаграму розподілу цін ноутбуків, які використовують в своїй будові відео прискорювачі розглянутих вище компаній. Тут можна побачити, що ноутбуки із відео прискорювачами від компаній Intel і Nvidia мають найбільший діапазон значень ціни, також Nvidia має найвищу середню ціну, тримаючись на позначці близько 1200 євро. Другим за середньою вартістю йде Intel, його значення розмістилось на позначці близько 950 євро. AMD серед проаналізованих ноутбуків має найменший діапазон цін, а також найнижчу середню ціну в 700 євро.

Після аналізу загального використання, визначено, що перші три позиції серед найчастіше використовуваних відеоадаптерів займє Intel зі своєю інтегрованою графікою, що й не дивно, враховуючи ціну даних рішень. На наступних трьох позиціях розмістились дискретні відеокарти від Nvidia, це

обумовлене популярністю ігрових ноутбуків, а також невисокій ціні та хорошій продуктивності серед інших дискретних рішень.



а б

Рис. 3.9. Дослідження графічного процесора.

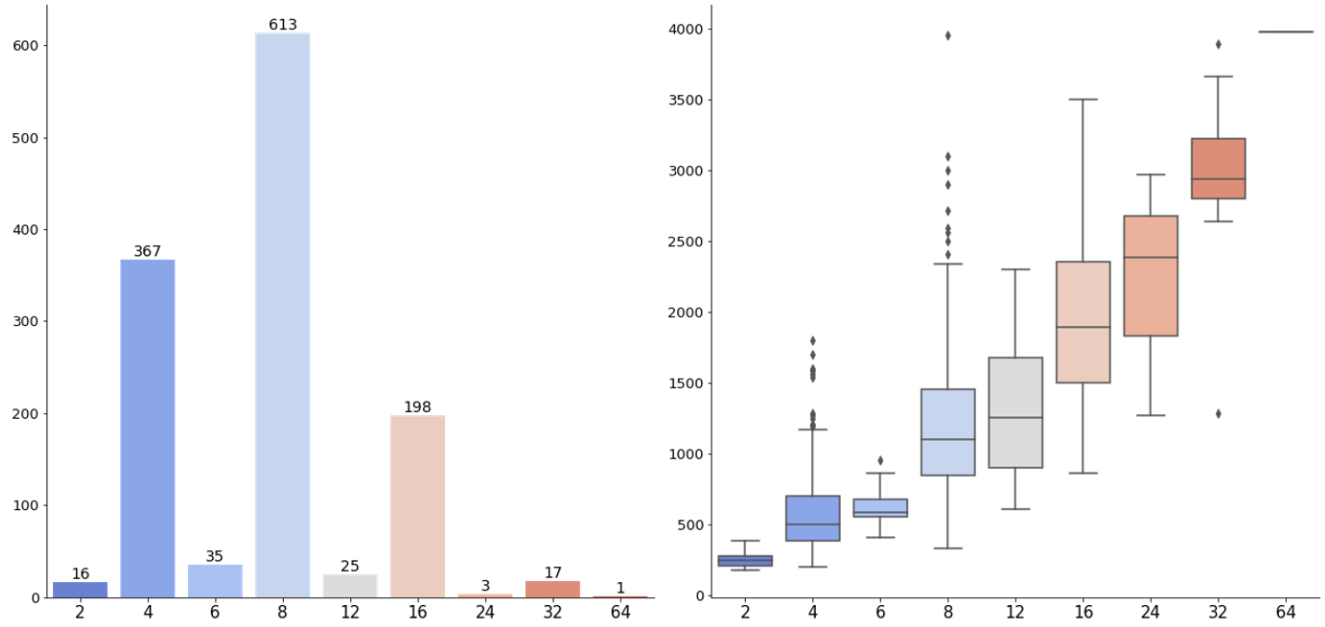
3.2.4. Дослідження оперативної пам’яті.

Оперативна пам’ять має значний вплив на роботу персонального комп’ютера - саме від її обсягу і частоти залежить загальна швидкодія, та рівень продуктивності при багатозадачності. Її можна вважати одним із головних елементів ноутбука, коли мова стосується маркетингу. Проте, така пам’ять є обмеженою, і під час використання багатозатратних програм її може не вистачати, що стане причиною зависання. Також, при нестачі пам’яті, може відбутись автоматчний процес очищення з неї менш важливих завдань.

В результаті дослідження цього типу пам’яті, спочатку було побудовано діаграму, що на рисунку 3.10 (а). Як видно із діаграми, чітко виділяються три найпопулярніші об'єми оперативної пам'яті, а саме: чотири, вісім та шістнадцять гігабайт. Як не дивно, саме середнє значення із цих тьрох є найпопулярнішим, це пояснюється тим, що ноутбуки, які мають вісім гігабайт оперативної пам'яті належаться до середнього цінового сегменту. Для більшості людей такий об'ємним вважається золотою серединою, якщо ноутбук не використовується у

високопродуктивних задачах. Друге місце займає об'єм в чотири гігабайти пам'яті. Такий об'єм втановлюється тільки в бюджетні рішення і для багатьох користувачів вже може цього не вистачати, однак, така популярність серед виробників нотбуків напряму залежить від ринкової вартості даного об'єму.

Якщо звернути свою увагу на діаграму 3.10 (б), то можна побачити розподіл середніх значень вартості різних об'ємів оперативної пам'яті. На даному типі діаграми спостерігається закономірне зростання ціни в залежності від об'єму. Відповідно, чим більший об'єм оперативної пам'яті, тим дорожчий ноутбук.



а б

Рис. 3.10. Дослідження оперативної пам’яті.

Найбільший діапазон вартісних значень мають найпопулярніші об'єми. Медіана середньої ціни ноутбуків із восьма гігабайтами пам'яті більш ніж в два рази вища за медіану середньої ціни ноутбуків із чотирьма нінабайтами пам'яті (1200 євро та 500 євро відповідно). Що стосується третього за популярністю об'єму, а саме шістнадцяти гігабайт, то даний об'єм встановлюється в ноутбуки середньої та преміальної цінової категорії. Саме тому, ноутбуки із такою кількістю оперативної пам'яті можна зустріти по ціні від 800 до 3500 євро, а медіана розташована на значенні 1800 євро. Через не достатню кількість ноутбуків, об'єми в 24 та 64 гігабайти можна виключити із підрахунків, щоб дослідження мало більш точні результати.

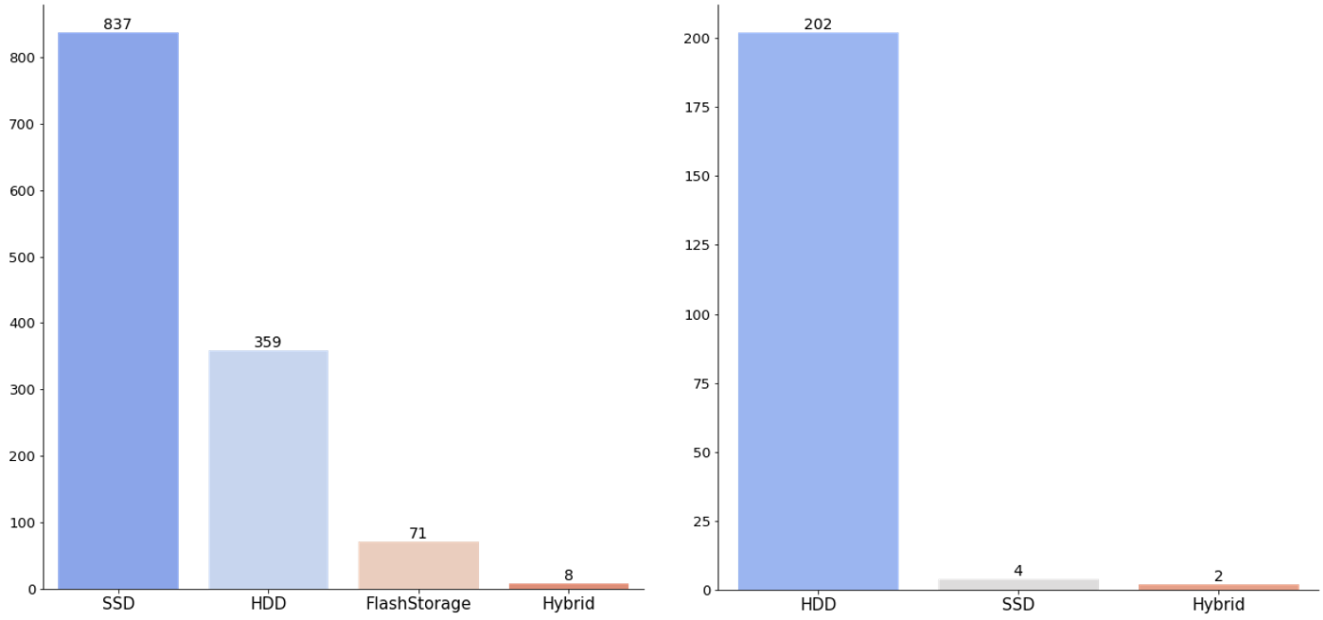
3.2.5. Дослідження накопичувачів.

Накопичувачі в системі комп'ютера не відіграють таку велику роль, як процесор чи оперативна пам'ять, під час оцінки продуктивності, проте, тип накопичувача відіграє велику роль під час читання та запису даних. На рисунку 3.11 (а) зображено побудовану діаграму, де відображається кількісне використання типів накопичувачів в ноутбуках в якості основого. Як можна побачити, серед проаналізованих продуктів накопичувачі типу SSD лідирують, вони були встановлені у 837 моделях ноутбуків датасету, як основний накопичувач. Такий результат пояснюється зниженням вартості на даний тип, внаслідок збільшення конкуренції на ринку SSD накопичувачів, а швидкість читання/запису, яка може становити від 500 до 7000 Мб/с, в залежності від моделі, тільки підвищує попит на даний тип накопичувача.

Наступним по кількості використання йде накопичувач типу HDD. Цій технології накопичувачі вже багато років, однак вона все ще користується деяким попитом за рахунок великих об'ємів пам'яті при низькій ціні. Третю та четверту позицію займають досить непопулярні рішення. Флеш пам'ять не користується попитом через свою повільну роботу та інтегрованість із системною платою комп'ютера, тобто її неможливо замінити без спеціального обладнання. Щодо гібридного накопичувача, тобто поєднання HDD і SSD, то ця технологія не користується популярністю по тій причині, що швидкості роботи не досягають до швидкостей SSD, а об'єм накопичувача не відрізняється від звичайного HDD. Єдина відмінність SHDD від HDD - це наявність в першому швидкісного кешу, який може зберігати в собі невеликі обсяги даних, однак мало в яких сферах SHDD можуть бути критично необхідними, що і пояснює їхню непопулярність.

Будь-який комп'ютер не часто має в своїй будові тільки один накопичувач даних, переважно їх декілька. Проте виробники ноутбуків найчастіше встановлюють в свої продукти тільки один накопичувач, залишаючи простір для бажань користувача, але є моделі в яких виробник сам втановив додатковий накопичував, тому на рисунку 3.11 (б) зображена діаграма використання в ноутбуках типу накопичувача в якості додаткового. Найчастіше встановлюється

накопичувач типу HDD, завдяки своїм об'ємам, що дозволяє використовувати основний накопичувач для системних файлів, а додатковий для менш важливих або об'ємних даних. SSD та SHDD в якості додаткового накопичувача використовуються настільки рідко, що ці дані можна віднести до похибки.



а б

Рис. 3.11. Дослідження накопичувачів.

Основна доля накопичувачів була типу SSD, найпопуляршіний і також можна вважати найоптимальніший об'єм SSD - це 256 Гб. Як додатковий накопичувач найпопуляршіний варіант в цьому наборі даних об'ємом в 1000 Гб, такий результат обумовлений великою кількістю HDD накопичувачів, які дуже рідко мають об'єм, що не дорівнював би 1000 Гб.

3.2.6. Дослідження міри кореляції компонентів.

Кореляція – статистична залежність, що показує міру пов’язаності між кількісними величинами. Застосовується кореляція для прогнозування, перевірки теорій, надійності тестування. Характер зв’язку між різними величинами може бути різний, тому необхідно визначити коефіцієнт їх залежності. Коефіцієнт кореляції коливається від -1 до +1, і чим ближче він до нуля, тим слабшою є лінійна залежність. Як результат кореляції, можна отримати тільки три можливі результати:

- позитивна кореляція - це взаємозв'язок між двома змінними, в яких їх значення мають тенденцію до збільшення разом.
- негативна кореляція - це взаємозв'язок між двома змінними, в яких значення однієї збільшується, в той час, як значення іншої змінної зменшуються.
- нульова кореляція – відбувається, коли між двома змінними немає жодного зв’язку.

Діаграма типу heatmap використовуються для того, щоб продемонструвати двовимірне візуальне зображення зв’язку однієї змінної з іншими у вигляді акуратної сітки кольорових комірок. Завдяки такому типу візуалізації можна з легкістю зрозуміти результат, саме тому, ця діаграма найкраще підходить для візуалізації кореляційних зв’язків. Варто зазначити, що будь-які нечислові значення будуть ігноруватись.

Бібліотека Seaborn дозволяє створювати діаграми heatmap, які можна конфігурувати за допомогою вісімнадцяти аргументів, відповідно до потреб. Основні аргументи наступні: **annot=True** – відображення коефіцієнту кореляції на кожній з комірок сітки; **annot_kws** – словник, що приймає значення для контролю сімейства і розміру шрифтів; **fmt** – форматування коефіцієнту кореляції до певної кількості знаків після крапки, що інколи забезпечує кращу читабельність; **square=True** – перетворює комірки сітки з форми прямокутника на квадрат; **cbar=False** – вимкнення вимірювальної шкали; **cbar_kws** – визначення орієнтації шкали; **vmin** – мінімальне значення вимірювальної шкали; **vmax** – максимальне значення вимірювальної шкали.

Для набору даних ноутбуків було створено матрицю кореляції, що на рисунку 3.12. Матриця дозволила отримати наступні відомості:

- найвищий позитивний коефіцієнт кореляції з ціною має оперативна пам’ять – 0.74. Також, на ціну має достатньо великий вплив ємність SSD – 0.68.

- Іншими важливими параметрами є тип центрального процесору і розширення екрану (x_res і y_res) – їх кореляційні коефіцієнти достатні, щоб викликати зміни вартості.
- Розмір екрану має високий позитивний коефіцієнт кореляції з вагою, що і не дивно – чим більший девайс, тим і більшою є його вага.

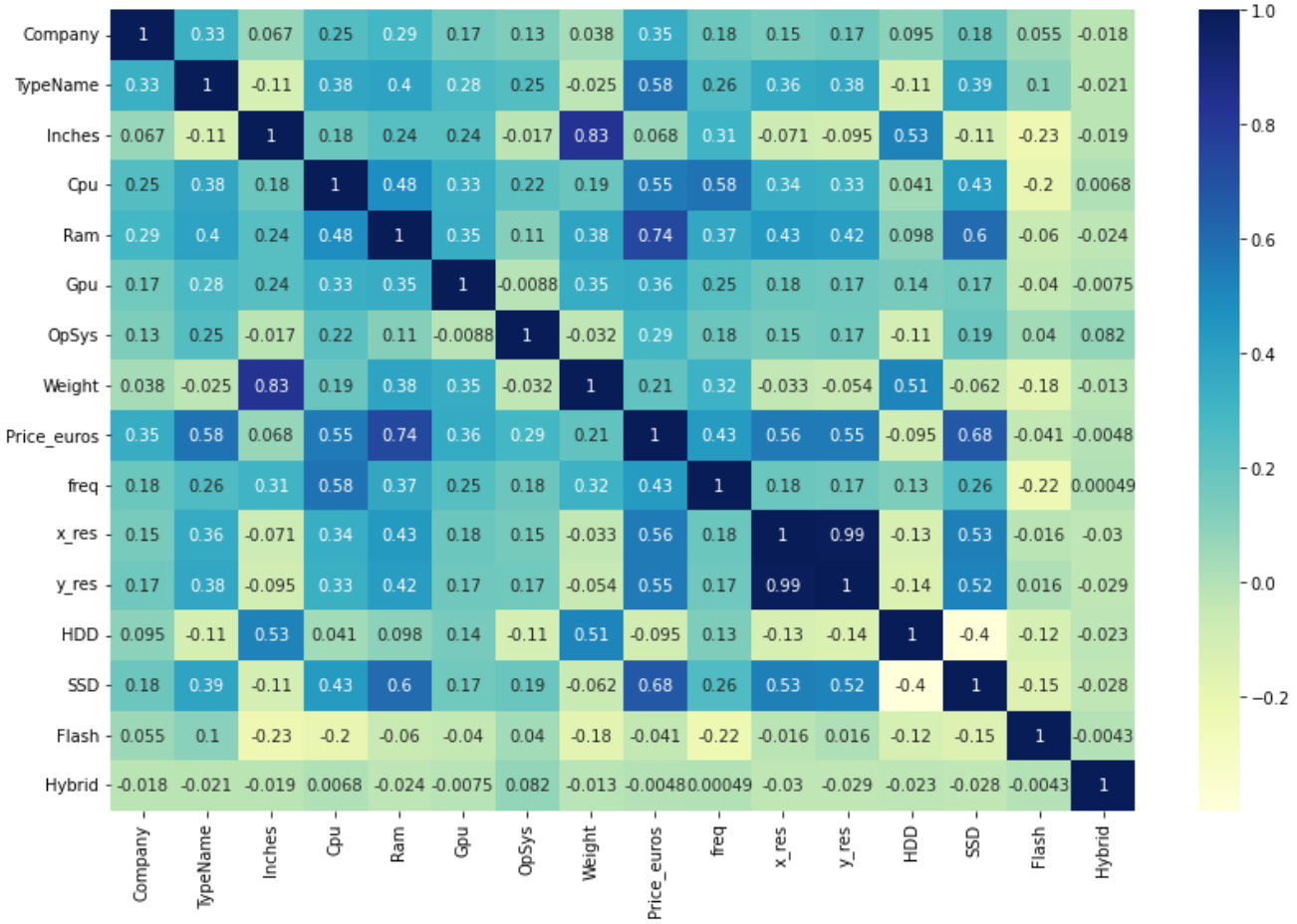


Рис. 3.12. Матриця кореляції.

ВИСНОВКИ

Отже, як результат виконання цієї дипломної роботи, було реалізовано наступні завдання:

1) Проаналізовано сучасний стан необхідності обробки величезних масивів різноманітних даних для отримання необхідно- корисної інформації для користувача. Зокрема, проведено аналіз стуктури масивів даних BigData, показано його структуру, існуючі можливості і перспективи їх використання.
2) Наведено типи даних та їх характеристики, визначено основні етапи проведення аналізу збирання та обробки даних.
3) Показано, що перспективною мовою програмування для опрацювання та візуалізації даних, (як високоінформативного методу) є мова Python, що відрізняється як простотою у користуванні, так і водночас, достатньо великими можливостями для створення програмних продуктів щодо обробки різноманітних масивів даних (від наукових до комерційних) і їх наступного представлення у інформативно-наглядній візуально-графічній формі.
4) Використано теоретичні навики для проведення очищення і попередньої обробки набору даних для забезпечення якості результатів. Виконано дослідницький аналіз компонентів датасету для виявлення закономірностей за допомогою зведеної статистики та елементів візуалізації. Створено матрицю кореляційних зв’язків між характеристиками та вартістю ноутбуків і, як результатом, є висновок, що найбільший вплив на вартість має кількість оперативної пам’яті (лінійна залежність доводить, що і в майбутньому цей компонент буде найважливішим).

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Python for Data Analysis. A Basic Guide for Beginners to Learn the Language of Python Programming Codes Applied to Data Analysis with Libraries Software Pandas, Numpy, and IPython [Текст] / Oliver R. Simpson, 2019. – 137 с.
2. How to read most commonly used file formats in Data Science [Електронний ресурс]. – Режим доступу: https://www.analyticsvidhya.com/blog/2017/03/read-commonly-used-formats-using-python/
3. A Python Data Analyst's Toolkit [Текст] / Gayathri Rajagopalan. 2021. – 409 с.
4. Python libraries for data analysis [Електронний ресурс]. – Режим доступу: https://makemeanalyst.com/data-science-with-python/python-libraries-for-data-analysis/
5. TensorFlow [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/TensorFlow
6. Chapter 3: Hierarchy of TensorFlow [Електронний ресурс]. – Режим доступу: https://lyhue1991.github.io/eat_tensorflow2_in_30_days/english/Chapter3/
7. Keras [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Keras
8. 10 Best Python IDE & Code Editors [Електронний ресурс]. – Режим доступу: https://hackr.io/blog/best-python-ide
9. What is an IDE in Programming? An IDE Definition for Developers [Електронний ресурс]. – Режим доступу: https://www.freecodecamp.org/news/what-is-an-ide-in-programming-an-ide-definition-for-developers/
10. IDE vs Code Editor--Why and When to Use Them [Електронний ресурс]. – Режим доступу: https://www.jobsity.com/blog/ide-vs-code-editor-why-and-when-to-use-them
11. Что такое Jupyter-ноутбук и как его использовать [Електронний ресурс]. – Режим доступу: https://highload.today/jupyter-notebook/

12. Why Jupyter Notebooks? [Електронний ресурс]. – Режим доступу: https://problemsolvingwithpython.com/02-Jupyter-Notebooks/02.02-Why-Jupyter-Notebooks/

13. The Jupyter Notebook [Електронний ресурс]. – Режим доступу: https://jupyter-notebook.readthedocs.io/en/stable/notebook.html

14. Python for Data Analysis. Second edition [Текст] / Wes McKinney. O'Reilly Media, Inc., 2020. – 540 с.

15. Usage - matplotlib documentation [Електронний ресурс]. – Режим доступу: https://matplotlib.org/2.0.2/faq/usage_faq.html

16. Краткое руководство по Matplotlib [Електронний ресурс]. – Режим доступу: https://pyprog.pro/mpl/mpl_types_of_graphs.html

17. What Is a GPU? [Електронний ресурс]. – Режим доступу: https://www.intel.com/content/www/us/en/products/docs/processors/what-is-a-gpu.html