МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра автоматизованих систем управління



**Інформаційні технології смарт-систем**

**МЕТОДИЧНІ ВКАЗІВКИ**

до виконання практичних робіт

для студентів базового напрямку "Комп’ютерні науки"

**Львів - 2023**

**CRISP-DM**

**Процедура CRISP-DM** включає 6 кроків плюс дані, які пов’язані між собою ітеративним процесом.

***Розуміння бізнесу*** – на цьому етапі визначте результати проекту, оцініть поточну ситуацію, складіть план проекту

***Розуміння даних*** – опишіть і вивчіть дані, перевірте якість даних, бо часто з’ясовується, що дані неправильно або неповністю збережені в хмарі чи базі даних

***Підготовка* даних** – виберіть дані, очистіть дані (замініть пропущені значення, видаліть нетипові спостереження тощо), об’єднайте декілька наборів даних та/або отримайте агреговані значення

***Моделювання*** – оберіть техніку моделювання (регресія, нейронна мережа тощо), розділіть дані на набори даних для побудови моделей (набір train включає 75% даних для побудови моделей) і тестування (набір test включає 25% даних для вибору найкращої моделі або ансамблю моделей), побудуйте модель, оцініть ефективність моделі

***Оцінювання*** – оцінка результатів (на нових даних протягом 2-3 місяців), перегляд та затвердження моделей

***Деплоймент (розгортання)*** – результати моделей стають частиною щоденної ділової активності

Лабораторні роботи направлені на реалізацію проекту з використанням CRISP-DM. Кожна з робіт має на меті реалізацію одного або декількох етапів (інколи частково).

*Теоретична частина до практичної роботи №1*

**Етап розуміння бізнесу**

Одним із основних завдань на етапі розуміння бізнесу є встановлення бізнес-цілей проекту.

Це означає описати свою основну мету з точки зору бізнесу. Також можуть виникнути інші пов'язані питання, які ви хотіли б вирішити. Наприклад, вашою основною метою може бути утримання поточних клієнтів, передбачення того, коли вони схильні переходити до конкурента. Пов'язані бізнес -питання можуть бути такими: "Чи впливає використовуваний канал на те, залишаться або підуть клієнти?" або "Чи нижча плата за банкомат зменшить кількість цінних клієнтів, що йдуть з компанії?"

Не менш важливим є визначити цілі моделювання

Бізнес-мета визначає цілі в бізнес-термінології. Мета побудови моделі визначає цілі проекту в технічному плані. Наприклад, комерційною метою може бути "Збільшити продаж каталогів існуючим клієнтам". Ціллю моделювання може бути "Передбачте, скільки віджетів придбає клієнт, враховуючи їх покупки за останні три роки, демографічну інформацію (вік, зарплата, місто тощо) та ціну товару".

Для того, щоб оцінити чи досягнуто поставлені цілі необхідно визначити критерії та метрики оцінювання.

1. Критерії успіху бізнесу описують передбачувані результати проекту, які дозволяють досягти бізнес -цілей.
2. Критерії успіху моделювання визначають критерії успішного результату проекту в технічному плані - наприклад, певний рівень точності прогнозування. Як і у випадку з критеріями успіху бізнесу, може знадобитися описати їх суб’єктивно.

**Етап розуміння даних**

Основною метою цього етапу є ознайомлення з даними, звідки вони походять, як формуються, які бізнес обмеження на них впливають, як вони пов’язані, візуальний аналіз даних.

Цей етап передбачає формування декількох звітів.

*Звіт про збір первинних даних* - перелік джерел даних разом з їх місцезнаходженням, методи, які використовуються для їх отримання, та будь-які проблеми, що виникли. Записуйте проблеми, з якими ви зіткнулися, і будь -які досягнуті рішення. Це допоможе у майбутньому при розширенні цього проекту і у виконанні подібних майбутніх проектів.

*Звіт з описом даних* - опишіть отримані дані, включаючи їх формат, кількість (наприклад, кількість записів і полів у кожній таблиці), ідентичність полів та будь -які інші особливості розподілу, які були виявлені. Оцініть, чи отримані дані відповідають вашим вимогам.

Щоб ефективно приступити до проекту, подумайте про цінність створення точного звіту з описом даних за допомогою таких показників:

Кількість даних:

* Який формат даних?
* Визначте метод, який використовується для збору даних-наприклад, ODBC.
* Наскільки велика база даних (у кількості рядків та стовпців)?

Якість даних:

* Чи включають дані характеристики, які мають відношення до бізнес -питання?
* Які типи даних присутні (текстові, числові тощо)?
* Ви обчислювали основні статистичні дані щодо ключових атрибутів? Яке розуміння це дало в бізнес -питанні?
* Чи можете ви визначити пріоритетність відповідних атрибутів? Якщо ні, чи доступні бізнес -аналітики для надання додаткової інформації?

*Рис.1. Етапи дослідження даних*Кроки  дослідження даних


*Звіт про дослідження даних* - опис результатів вашого дослідження даних, включаючи перші висновки або початкову гіпотезу та їх вплив на решту проекту, графіки та діаграми, що вказують характеристики даних.

Для створення цього звіту використовуються методи інтелектуального аналізу даних та візуалізації даних. Вони можуть включати:

* Пропущені (порожні) значення ознак
* Типи даних кожної з ознак
* Розподіл ключових ознак (наприклад, цільового атрибута прогнозування)
* Масштаб значень ознак
* Відносини між парами або підгрупами атрибутів (кореляційний аналіз)
* Виявлення аутлаєрів (аномальних значень)
* Результати простих агрегацій
* Статистичний аналіз

*Звіт про якість даних -* перелічіть результати перевірки якості даних. Якщо проблеми з якістю існують, запропонуйте можливі шляхи їх вирішення. Рішення проблем якості даних, як правило, значною мірою залежать як від даних, так і від знань бізнесу. Дані рідко бувають ідеальними. Насправді, більшість даних містить помилки кодування, відсутні значення або інші невідповідності, які іноді ускладнюють аналіз.

Найчастіші такі типи проблем:

* Відсутні дані включають значення, порожні або кодовані як відповідь (наприклад, $ null $,? Або 999).
* Помилки даних - це зазвичай друкарські помилки, допущені при введенні даних.
* Похибки вимірювання включають дані, введені правильно, але засновані на неправильній схемі вимірювання.
* Невідповідності кодування зазвичай включають нестандартні одиниці вимірювання або суперечності величин, такі як використання як M, так і чоловічої статі для статі.
* Погані метадані включають невідповідність між видимим значенням поля та значенням, зазначеним у назві поля або визначенні.

Список запитань для формування звіту про якість даних:

1. Ви виявили відсутні атрибути та порожні поля? Якщо так, то чи є сенс у таких відсутніх значеннях?
2. Чи існують правописні невідповідності, які можуть спричинити проблеми при подальших злиттях чи перетвореннях?
3. Чи досліджували ви відхилення, щоб визначити, чи є вони "шумом" чи явищами, які варто аналізувати далі?
4. Ви проводили перевірку правдоподібності на значення? Записуйте будь -які явні конфлікти (наприклад, підлітки з високим рівнем доходу).
5. Ви думали про виключення даних, які не впливають на ваші гіпотези?
6. Чи зберігаються дані у файлах? Якщо так, то чи розділяються ліміти між файлами? Чи кожен запис містить однакову кількість полів?

Перш ніж готувати дані для моделювання, зверніть увагу на такі моменти:

* Наскільки ви добре розумієте дані?
* Чи всі джерела даних чітко визначені та доступні? Вам відомо про якісь проблеми чи обмеження?
* Ви визначили ключові атрибути з наявних даних?
* Ці атрибути допомогли вам сформулювати гіпотези?
* Ви відзначали розмір усіх джерел даних?
* Чи можете ви використовувати підмножину даних, де це доречно?
* Ви обчислювали основні статистичні дані для кожного цікавого атрибута? Чи з’явилася змістовна інформація?
* Чи використовували ви дослідницьку графіку, щоб отримати додаткове уявлення про основні атрибути? Чи змінило це уявлення якусь із ваших гіпотез?
* Які проблеми якості даних для цього проекту? Чи є у вас план вирішення цих питань?
* Чи зрозумілі кроки підготовки даних? Наприклад, чи знаєте ви, які джерела даних об’єднати, а які атрибути відфільтрувати чи вибрати?

*Теоретична частина до практичної роботи №2*

**Етап підготовки даних (Feature engineering)**

Типи вхідних даних, що використовуються для навчання ДС моделей зображені на схемі нижче.Diagram

Description automatically generated

*Рис.2. Типи даних*

Підготовка даних (інженерія ознак) є одним з найважливіших етапів і часто займає багато часу. Насправді, за оцінками, підготовка даних зазвичай займає 50-70% часу та зусиль проекту. Приділення достатньої енергії попереднім етапам розуміння бізнесу та розуміння даних може мінімізувати це, але вам все одно доведеться витратити чимало зусиль на підготовку даних для моделювання.

Одним з етапів підготовки даних є їх очищення. Це завдання передбачає підвищення якості даних до рівня, необхідного для обраних вами методів аналізу. Це може включати вибір чистих підмножин даних, вставлення відповідних типових значень або більш амбітні методи, такі як оцінка відсутніх даних шляхом моделювання. Очищення ваших даних передбачає уважніший погляд на проблеми у даних, які ви вирішили включити для аналізу.

*Звіт про очищення даних* - опишіть, які рішення та дії ви прийняли для вирішення проблем якості даних. Розгляньте будь-які зміни даних, зроблених для очищення, та їх можливий вплив на результати аналізу.

Під час написання звіту варто розглянути такі питання:

1. Які типи шуму (аномальні значення, відсутність значень, некоректне кодування категоріальних ознак) траплялися у даних?
2. Які підходи ви використовували для усунення шуму? Які методи були успішними?
3. Чи існують приклади чи ознаки, які неможливо використати? Обов’язково зверніть увагу на дані, виключені через шум.

Етапи інженерії ознак:

1. Створення нових ознак:

* з ознак формату дати
* зі змішаних ознак

1. Заміна пропущених значень
2. Кодування категоріальних ознак
3. Трансформація чисельних ознак
4. Дискретизація
5. Заміна аномальних значень
6. Масштабування ознак
7. Інше

*\*Послідовність етапів важлива*

Виконуючи лабораторну роботу №2 ви ознайомитесь та використаєте деякі з методів інженерії ознак.

**Відсутні дані:**

* Відсутні дані або відсутні значення виникають, коли даних немає збережених для певного спостереження у змінній.
* Відсутні дані є звичайним явищем у більшості наборів даних.
* Відсутні дані можуть мати значний вплив на висновки, які можна зробити з даних.

Причинами відсутніх даних можуть бути:

* Дані можуть бути пропущеними через некоректну процедуру збереження даних.
* Ознака створена в результаті ділення чисел і знаменник був 0.
* При зіставленні даних з поштовим індексом або датою народження для збільшення ознак, а поштовий індекс або дата народження відсутня в іншому джерелі даних, тому при об’єднанні нові змінні приймуть Null

Дані можуть бути пропущені абсолютно рандомно, або ж з певною закономірністю.

Повністю випадково дані відсутні (Missing Data Completely at Random (MCAR))

* Ймовірність пропуску однакова для всіх спостережень
* Немає абсолютно ніякого зв'язку між відсутніми даними та будь -якими іншими значеннями, що спостерігаються або відсутні, у наборі даних
* Ігнорування цих випадків не упереджує зроблені висновки

Випадково відсутні дані (Missing Data at Random (MAR))

* Імовірність відсутності спостереження залежить від наявної інформації

Diagram

Description automatically generated

*Рис.3. Приклад випадково відсутніх даних*

Дані відсутні не випадково (Missing Data not at Random (MNAR))

* Існує механізм або причина, чому в наборі даних є відсутні значення.

**Table

Description automatically generated**

*Рис.4. Приклад даних, що відсутні не випадково*

Щоб зрозуміти механізми, які використовувати для заміни відсутніх даних, нам потрібно ознайомитися з методами, що використовуються для збору даних. Але слід зауважити, що це не завжди можливо.

Імпутація - це акт заміни відсутніх даних статистичними оцінками відсутніх значень. Метою будь-якої техніки внесення даних є створення повного набору даних, який можна використовувати для навчання моделей машинного навчання.

Graphical user interface

Description automatically generated with medium confidence

*Рис.5. Техніки заміни відсутніх значень*

Заміна середнім/ медіанним значенням

Заміна середнім / медіанним значенням включає заміну всіх відсутніх значень (NA) у прикладах на середнє (якщо змінна має гаусівський розподіл) або медіану (якщо змінна має перекошений розподіл).

Зверніть увагу на наступне:

* Якщо змінна нормально розподілена, середнє значення, медіана та мода приблизно однакові. Тому заміна пропущених значень на середнє та медіану є еквівалентними. Заміна відсутніх даних на моду не є звичайною практикою для числових змінних.
* Якщо розподіл зміщений, середнє значення зміщується. Тому медіана - це найкращий варіант у більшості випадках.
* Для дискретних змінних, що використовуються як 'int' (для збереження пам'яті), середнє значення може бути не цілим числом, тому вся змінна буде повторно використана як 'float'. Щоб уникнути такої поведінки, замість цього ми можемо замінити NA медіаною. Медіана також буде цілим / дискретним значенням.

Які змінні можна зарахувати за допомогою середнього / медіанного значення?

* Середнє та медіану можна обчислити лише за числовими змінними, тому ці методи підходять лише для безперервних та дискретних числових змінних.

Припущення:

* Дані відсутні навмання (MCAR). Відсутні спостереження, швидше за все, виглядають як більшість спостережень у змінній (вона ж, середнє значення / медіана)
* Якщо дані відсутні абсолютно випадковим чином, то справедливо припустити, що пропущені значення, швидше за все, дуже близькі до середнього значення або медіани розподілу, оскільки вони являють собою найбільш часте / середнє спостереження.

Переваги:

* Легко реалізувати
* Швидкий спосіб отримання повних наборів даних
* Може бути інтегрований під час розгортання моделі

Обмеження:

* Спотворення вихідного розподілу змінних
* Спотворення початкової дисперсії
* Спотворення коваріації з іншими змінними набору даних

При заміні NA середнім значенням або медіаною, дисперсія змінної спотворюватиметься, якщо кількість NA буде великою по відношенню до загальної кількості спостережень, що призведе до заниження дисперсії.

Крім того, можуть вплинути оцінки коваріації та кореляції з іншими змінними у наборі даних.

Концентрація всіх пропущених значень на середньому / медіанному значенні може призвести що приклади (спостереження), які є звичайними явищами у розподілі, будуть сприйняті як аномалії.

Коли використовувати середнє / медіанне внесення?

* Дані відсутні навмання
* Не більше 5% змінної містить відсутні дані

*Хоча теоретично вищезазначені умови мають бути виконані, щоб мінімізувати вплив цієї техніки внесення, на практиці дуже часто використовується, навіть у тих випадках, коли дані не є MCAR і є багато відсутніх значень. Причиною цього є простота техніки.*

Заміна відсутніх даних довільним (Arbitrary value) значенням

Врахування довільного значення полягає в заміні всіх появ відсутніх значень (NA) у межах змінної на довільне значення. Зазвичай використовуються довільні значення 0, 999, -999 (або інші комбінації 9s) або -1 (якщо розподіл позитивний).

Які змінні можна замінити довільним значенням?

* І категоріальні, і числові відсутні дані можуть бути замінені довільними значеннями.

Для категоріальних змінних це еквівалент заміни всіх екземплярів NA на додаткову мітку, що є дуже поширеною практикою.

Припущення:

* Дані відсутні випадково.

Якщо це так, ми хочемо позначити відсутні значення іншим (довільним) значенням, замість того, щоб замінити ці входження на середнє або медіану, які представляють найпоширеніше значення.

Переваги:

* Легко реалізувати
* Швидкий спосіб отримання повних наборів даних
* Може бути інтегрований у виробництво (під час розгортання моделі)
* Висвітлює важливість "відсутності", якщо вона є

Обмеження

* Спотворення вихідного розподілу змінних
* Спотворення початкової дисперсії
* Спотворення коваріації з іншими змінними набору даних

Якщо довільне значення є в кінці розподілу, воно може маскувати або створювати аномалії.

Потрібно бути обережним, щоб не обрати довільне значення, занадто схоже на середнє або медіану (або будь -яке інше загальне значення розподілу змінної)

Коли використовувати:

* Заміну NA на довільні значення слід використовувати, коли є підстави вважати, що NA відсутні випадково i ми хочемо відзначити той факт, що спостереження відсутнє.

Проблема підходу полягає у вирішенні того, яке довільне значення вибрати.

Заміна відсутніх даних на значення кінця розподілу

У попередньому пункті ми ознайомилися з заміною відсутніх даних на довільне значення. Однак визначення довільного значення може бути трудомістким, і це, як правило, ручна робота. Ми можемо автоматизувати цей процес, автоматично вибравши довільні значення з кінців розподілів ознак.

Як вибрати значення в кінці?

* Якщо змінна нормально розподілена, ми можемо використовувати середнє значення плюс/мінус число в 3 рази більше стандартного відхилення.
* Якщо змінна зміщена, ми можемо використовувати правило наближення IQR.
* Ми також можемо вибрати значення min / max і помножити його на певну кількість разів, наприклад 2 або 3.

Які змінні можна замінити довільним значенням?

* Цей метод підходить для числових змінних.

Припущення:

* MNAR - значення не відсутнє навмання

Якщо значення відсутнє не навмання, ми не хочемо замінювати його середнім значенням / медіаною, і тому робити це спостереження схожим на більшість наших спостережень. Натомість ми хочемо позначити це спостереження як інше, і тому ми призначаємо значення, яке знаходиться в кінці розподілу, де спостереження рідко представлені у сукупності.

Переваги:

* Легко реалізувати
* Швидкий спосіб отримання повних наборів даних
* Може бути інтегрований під час розгортання моделі
* Висвітлює важливість "відсутності", якщо вона є

Недоліки

* Спотворення вихідного розподілу змінних
* Спотворення початкової дисперсії
* Спотворення коваріації з іншими змінними набору даних
* Ця техніка може маскувати справжні аномалії у розподілі

*Я не бачила, щоб цей метод використовувався часто у ДС, проте цей метод використовується у фінансових компаніях. При збиранні фінансової історії клієнтів, щоб не вважати, що дані відсутні випадково, відсутні дані замінюються значенням в кінці розподілу.*

Заміна відсутніх значень на категорію, що найчастіше зустрічається (Frequent category imputation | Mode imputation)

Які змінні можна заміняти цією технікою?

* Хоча медіану або найчастіше значення категорії можна обчислити як для числових, так і для категорійних змінних, на практиці ми використовуємо цей прийом лише для категоріальних змінних. Причина в тому, що для числових змінних середнє значення або медіана мають тенденцію краще представляти середнє значення вибірки.

Припущення:

* Дані відсутні навмання (MCAR)

Відсутні спостереження, швидше за все, виглядають як більшість спостережень у змінній (вона ж - мода)

Обґрунтування полягає в тому, що значення, яке слід було б побачити для відсутнього спостереження, є, швидше за все, найчастішим значенням.

Переваги:

* Легко реалізувати
* Швидкий спосіб отримання повних наборів даних
* Може бути інтегрований під час розгортання моделі

Обмеження:

* Спотворення зв’язку найчастішої категорії з іншими змінними в наборі даних
* Може призвести до надмірного представлення найпоширенішої категорії, якщо є велика кількість NA

Коли використовувати?

* Дані відсутні навмання
* Не більше 5% змінної містить відсутні дані

Заміна відсутніх даних довільним (Arbitrary value) значенням для категоріальних змінних

Це найбільш широко використовуваний метод заміни відсутніх даних для категоріальних змінних. Цей метод полягає у розгляді відсутніх даних як додаткової категорії. Усі відсутні спостереження згруповані у категорії "Відсутня".

По суті, це еквівалент заміни числових змінних довільним значенням.

Краса цієї техніки полягає в тому, що вона нічого не передбачає про те, що дані відсутні. Вона дуже добре підходить, коли кількість відсутніх даних велика.

Переваги:

* Легко реалізувати
* Швидкий спосіб отримання повних наборів даних
* Може бути інтегрований під час розгортання моделі
* Висвітлює важливість "відсутності", якщо вона є
* Припущення щодо даних не зроблені

Обмеження

* Якщо кількість NA невелика, створення додаткової категорії може призвести до перенавчання моделей.

Заміна відсутніх даних випадковими значеннями

Заміна випадковою вибіркою в принципі подібне до заміни на середнє значення, в тому сенсі, що воно спрямоване на збереження статистичних параметрів вихідної змінної, дані про які відсутні.

Випадкова вибірка для заміни складається з випадкового спостереження з наявних спостережень змінної та використання цього випадково значення для заповнення NA. При внесенні випадкової вибірки береться стільки випадкових спостережень, скільки відсутніх змінних у змінній.

Завдяки випадковим вибірковим спостереженням змінної для тих випадків, коли є дані, ми гарантуємо збереження середнього значення та стандартного відхилення змінної.

Застосовуючи цей метод для категоріальних змінних ми гарантуємо збереження частоти різних категорій / міток у змінній.

Які змінні я можу зарахувати шляхом випадкової вибіркової імпутації?

* Визначення випадкової вибірки можна застосувати як до числових, так і до категорійних змінних.

Припущення

* Дані відсутні абсолютно випадково (MCAR). Якщо це так, має сенс замінити відсутні дані значеннями, витягнутими з вихідного розподілу змінних.

З імовірнісної точки зору, значення, які є більш частими, наприклад середнє значення, медіана чи найчастіша категорія, для категоріальних змінних, будуть вибиратися частіше - оскільки їх більше для вибору, але інші значення також будуть обрані. Таким чином, дисперсія та розподіл змінної зберігаються.

Ідея полягає в тому, щоб замінити сукупність відсутніх значень на сукупність значень з однаковим розподілом вихідної змінної.

Переваги:

* Легко реалізувати
* Швидкий спосіб отримання повних наборів даних
* Зберігає дисперсію змінної

Обмеження:

* Випадковість

На взаємозв’язок замінених змінних з іншими змінними може вплинути, якщо багато NA.

Підхід важкий для розгортання, оскільки нам потрібно зберігати оригінальний навчальний набір для вибірки значень і заміни NA у майбутніх спостереженнях.

Коли використовувати випадкову зразок імпутації?

* Дані відсутні навмання
* Не більше 5% змінної містить відсутні дані
* Добре підходить для лінійних моделей, оскільки не спотворює розподіл, незалежно від % NA

Якщо його використовувати в поєднанні з відсутнім показником, як ми побачимо в наступному пункті, цей метод можна використовувати, коли дані також відсутні випадково, або коли є багато відсутніх спостережень.

Додавання змінної для виявлення NA

У попередніх пунктах ми дізналися, як замінити пропущені значення на середнє, медіану або шляхом додавання випадкового значення. Ці методи передбачають, що дані відсутні абсолютно випадково (MCAR).

Існують інші методи, які можна використовувати, коли значення відсутні не випадково. Однак ці методи внесення змін різко вплинуть на розподіл змінних і тому не підходять для лінійних моделей.

Отже, що ми можемо зробити, якщо дані не є MCAR і ми хочемо використовувати лінійні моделі?

Якщо дані відсутні випадковим чином, непогано замінити відсутні спостереження середнім значенням / медіаною і позначити ці відсутні спостереження також іншим показником. Індикатор відсутності - це додаткова двійкова змінна, яка вказує, чи були дані відсутні для спостереження (1) чи ні (0).

Для яких змінних я можу додати відсутній індикатор?

* Ми можемо додати відсутній показник як до числових, так і до категоріальних змінних.

Примітка

* Додавання відсутнього показника ніколи не використовується окремо. Він завжди використовується разом з іншим прийомом заміни даних. Ми також можемо використовувати внесення випадкових вибірок разом з додаванням відсутнього показника як для категоріальних, так і для числових змінних.

Припущення:

* Дані відсутні випадково
* Відсутні дані є передбачувальними

Переваги:

* Легко реалізувати
* Охоплює важливість відсутніх даних, якщо вони є

Обмеження:

* Розширює простір функцій
* Щоб видалити NaN, ще потрібно приписати вихідну змінну

Додавання відсутнього індикатора збільшить кількість ознак. Якщо набір даних містить 10 об’єктів, і всі вони мають відсутні значення, після додавання відсутнього показника ми матимемо набір даних із 20 ознаками. Можливо, це не проблема у наборах даних з десятками до кількох сотень змінних, але якщо наш вихідний набір даних містить тисячі змінних, то, створивши додаткову змінну для позначення NA, ми отримаємо дуже великі набори даних.

Крім того, дані, як правило, відсутні для одного і того ж спостереження за кількома змінними, що часто призводить до того, що багато змінних показників, які відсутні, насправді є подібними або ідентичними один одному.

**Кодування (Encoding) категоріальних ознак**

One Hot кодування полягає у кодуванні кожної категорії різними булевими змінними (також називаються dummy змінними), які приймають значення 0 або 1, що вказує на те, що категорія присутня в спостереженні.

Наприклад, для категоріальної змінної "стать", з можливими категоріями "жіноча" та "чоловіча", ми можемо генерувати булеву змінну "жіноча", яка приймає 1, якщо людина "жінка" або 0, (або ми можемо генерувати змінна "чоловіча", яка приймає 1, якщо людина є "чоловіком" і 0 інакше).

Для категоричної змінної "кольору" з значеннями "червоний", "синій" і "зелений", ми можемо створити 3 нових змінних, що називаються "червоними", "синіми" та "зеленими". Ці змінні будуть приймати значення 1, якщо спостереження є зазначеним кольором або 0 інакше.

*Кодування в K-1 змінних*

Зверніть увагу, однак, що для змінної "кольору", створюючи 2 бінарні змінні, скажімо "червоний" і "синій", ми вже кодуємо всю інформацію:

* Якщо спостереження червоне, він буде захоплений змінною "червоним" (червоний = 1, синій = 0)
* Якщо спостереження синє, він буде захоплений змінною "синій" (червоний = 0, синій = 1)
* Якщо спостереження зелене, це буде захоплено комбінацією "червоного" та "синього" (червоний = 0, синій = 0)

Нам не потрібно додавати третю змінну "зелену", щоб захопити, що спостереження зелене.

Більш загалом, категоріальна змінна повинна бути закодована шляхом створення двійкових змінних K-1, де K - кількість різних категорій. У випадку статі, K = 2 (чоловіки / жінки), тому нам потрібно створити лише 1 (k - 1 = 1) бінарну змінну. У випадку кольору, який має 3 різних категорій (k = 3), нам потрібно створити 2 (k - 1 = 2) двійкові змінні, щоб захопити всю інформацію.

One hot кодування в двійкові змінні K-1 враховує, що ми можемо використовувати меншу розмірність і все ще охопимо всю інформацію: якщо спостереження становить 0 у всіх бінарних змінних, то він повинен бути у не присутній категорії. Тому, кодування категоричних змінних у K-1 бінарні змінні, як це дозволяє уникнути введення надлишкової інформації.

Виняток: кодування в K змінних

Деякі випадки, коли краще кодувати змінні в K-манекенні змінні:

* При використанні алгоритмів на основі дерева
* При виборі функцій за допомогою рекурсивних алгоритмів
* Коли цікаво визначити важливість кожної окремої категорії

Алгоритми на основі дерева, на відміну від більшості алгоритмів машинного навчання, не оцінюють весь набір даних під час навчання. Вони випадково витягують підмножину ознак з даних, встановлених у кожному вузлі для кожного дерева.

Якщо ми плануємо зробити вибір ознак за допомогою рекурсивних підходів, або якщо ми хочемо оцінити важливість кожної окремої категорії категоріальної змінної, то нам також буде потрібно весь набір бінарних змінних (K), щоб дозволити моделі машинного навчання вибрати, які з них мають найбільший вплив на результати прогнозування.

Переваги методу:

* Простий для реалізації
* Не робить припущення про розподіл або категорії категоріальної змінної
* Зберігає всю інформацію змінної
* Підходить для лінійних моделей

Обмеження:

* Розширює простір ознак
* Не додає додаткової інформації під час кодування
* Багато dummy змінних можуть бути ідентичними, додаючи надлишкову інформацію
* Якщо наші набори даних містять кілька категоріальних змінних з багатьма категоріями, ми скоро опинимося з наборами даних з тисячами стовпців, які можуть зробити навчання наших алгоритмів повільним та неефективним.

One hot encoding категорій, що найчастіше зустрічаються в наборі даних

Недоліками попереднього підходу є велика різноманітність категорій та рідкісні категорії, що можуть призвести до певних ознак, які з'являються лише у наборі навчальних даних. Це може призвести до перенавчання моделі або ж рідкісні категорії, які зустрічаються лише в тестовому наборі даних і під час передбачення виникають помилки.

Щоб уникнути цих ускладнень, ми можемо створювати змінні лише для найчастіших категорій.

Ця процедура також називається One Hot Encoding (OHE) of Frequent Categories.

OHE частих або найвищих категорій еквівалентно групуванню всіх інших категорій у нову спільну категорії “Інше”.

Переваги OHE найвищих категорій

* Простота в реалізації
* Не розширює критично простір ознак
* Підходить для лінійних моделей

Обмеження

* Не додає будь-якої інформації, яка може зробити змінну більш прогнозною
* Не зберігає інформацію про проігноровані категорії

Часто категоріальні змінні мають кілька домінуючих категорій, а інші категорії додають мало інформації. Тому OHE з найвищих категорій - це проста і корисна техніка. Кількість найчастіших категорій встановлюється довільно. Це число можна вибрати довільно або отриманий від аналізу даних.

Кодування цілим числом (Integer Encoding)

Кодування цілим числом полягає у заміні категорій цифрами від 1 до N (або 0 до N-1, залежно від реалізації), де N - кількість різних категорій змінної. Номери призначаються довільно.

Переваги:

* Простий для реалізації
* Не розширює простір ознак

Обмеження:

* Не надає жодної інформації про значення категорій
* Не підходить для лінійних моделей.

Цей метод краще підходить для нелінійних методів таких як дерева рішень.

Кодування базуючись на цільовій змінній

Ми оглянули прості методи що не роблять припущень щодо категоріальних змінних та щодо цільової змінної та працюють взагалі в різних сценаріях.

Однак є методи, які дозволяють нам додати інформацію, одночасно кодувати категорії категоріальних змінних. Ці методи містять:

* Кодування категорій відповідно до цільової змінної (Ordering the labels according to the target)
* Заміна категорій середнім значенням цільової змінної (mean encoding / target encoding)
* Заміна категорій на ймовірність цільової змінної (для класифікації)
* Заміна категорій на кількість (вагу) прикладів з цими категоріями (Weight of evidence)

Всі вищевказані методи мають щось спільне:

* Кодування керується цілями
* Вони створюють монотонний взаємозв'язок між ознаками і цільовою змінною.

Монотонні відносини - це відносини, які виконують одне з наступних:

(1) якщо значення однієї змінної збільшується, то значення іншої змінної також збільшується; або (2) якщо значення однієї змінної збільшується, значення іншої змінної зменшується.

Переваги

* Додавання інформації в категорії, створюючи більш прогнозовані особливості
* Створення монотонного взаємозв’язку між ознакою і цільовою змінною, тому підходить і для лінійних моделей
* Не розширює простір ознак

Обмеження

* Схильні викликати перенавчання
* Важко перевірити поточними бібліотеками

Кодування з використанням частоти (Count or frequency encoding)

У цьому підході, ми замінюємо категорії за рахунок підрахунку спостережень, які показують цю категорію в наборі даних. Аналогічним чином, ми можемо замінити категорію частотним відсотком спостережень у наборі даних. Тобто, якщо 10 наших 100 спостережень показують колір синього кольору, ми замінимо синій на 10, якщо виконуємо кодування підрахунку, або на 0,1, якщо замінити на частоту.

Припущення техніки полягає в тому, що кількість спостережень, показаних кожною змінною, містить додаткову інформацію щодо категорії.

Переваги

* Простий в реалізації
* Не розширює функціональний простір

Недоліки

* Якщо 2 різні категорії з'являються таку ж кількість разів у наборі даних, тобто вони з'являються в тій же кількості спостережень, вони будуть замінені на те ж число: може втратити цінну інформацію.

Наприклад, якщо є 10 спостережень для категорії синій та 10 спостережень для категорії червоний, обидва будуть замінені на 10.

**Опрацювання аномалій**

Аномалія (оutlier) - це точка даних, яка суттєво відрізняється від інших даних. "Outliver - це спостереження, яке настільки відрізняється від інших спостережень, щоб викликати підозри, що він був створений іншим механізмом".

Статистичні показники, такі як середня та дисперсія, дуже сприйнятливі до викидів. Крім того, деякі моделі машинного навчання чутливі до аномалій. Таким чином, залежно від того, який алгоритм ми хочемо використовувати, ми часто видаляємо аномалії з наших змінних.

Як ми можемо попередньо обробляти аномалії?

* Trimming: видалення аномалій з набору даних
* Ставитися до аномалій як до відсутніх даних та застосовувати будь-який підхід щодо відсутніх даних
* Цензурування: аномалії замінюються найбільшим або найменшим нормальним значенням ознаки
* Дискретизація: заміна всіх ознак на дискретні частини, а аномалій на найбільший або найменший дискрет.

Для видалення аномалій з набору даних нам потрібно лише вирішити що таке аномалій. Ми можемо використати гаусівський розподіл для нормально розподілених змінних.

Перевагою є швидка реалізація.

Обмеженням є те, що аномалії для однієї змінної можуть містити корисну інформацію в інших змінних. І можемо видалити велику частину датасету.

**Масштабування значень ознак**

Масштаб ознак є важливим фактором при побудові моделей машинного навчання. коротко:

* Коефіцієнти регресії лінійних моделей безпосередньо залежать від масштабу змінної.
* Метод градієнтного спуск сходиться швидше, коли ознаки знаходяться на аналогічних масштабах
* масштабування ознак дозволяє зменшити час пошуку векторів підтримки для SVMs
* Евклідові відстані чутливі до величини ознаки.
* Деякі алгоритми, такі як PCA вимагають ознак, щоб мають центр розподілу в 0.

Моделі машинного навчання, які чутливі до масштабу ознак:

* Linear and Logistic Regression
* Neural Networks
* Support Vector Machines
* KNN
* K-means clustering
* Linear Discriminant Analysis (LDA)
* Principal Component Analysis (PCA)

Функція масштабування

Масштабування ознак відноситься до методів або технік, що використовуються для нормалізації незалежних змінних в даних, або, іншими словами, способи, щоб встановити діапазон значень ознак в одному масштабі. Масштабування ознак, як правило, останній крок в підготовці даних, виконується безпосередньо перед навчанням моделей.

Є кілька методів Feature Scaling:

* Стандартизація
* Середня нормалізація
* Масштабування до мінімальних і максимальних значень - MinMaxScaling
* Масштабування до максимального значення - MaxAbsScaling
* Масштабування для квантилів і медіана - RobustScaling
* Нормалізація на одиницю довжини вектора

Стандартизація

Стандартизація включає в себе центрування значень ознаки в нулі, і стандартизація дисперсії до 1. Процедура включає в себе віднімання середньої кожне спостереження з подальшим розподілом на стандартному відхиленні:

z= (х - x\_mean) / std

Результат перетворення Z, який називається Z-оцінка, і означає наскільки стандартне відхилення даного спостереження відхиляється від середнього значення. Z-оцінка вказує місце розташування спостереження (прикладу) в розподілі (в кількості стандартних відхилень відносно середнього розподілу). Знак Z-оцінки (+ або -) вказує, чи є спостереження вище (+) або нижче (-) середнього.

Форма стандартизованого (або нормалізованою Z оцінкою) розподілу буде ідентична вихідного розподілу змінної. Якщо вихідний розподіл є нормальним, то стандартизований розподіл буде нормальним. Іншими слова, стандартизація змінних не нормалізує розподілу даних.

У двох словах, стандартизація:

* центрує середнє в 0
* масштабує дисперсію до 1
* зберігає форму початкового розподілу
* мінімальні і максимальні значення різних змінних можуть відрізнятися
* нечутлива до аномалій

Добре для алгоритмів, що вимагають ознак з центром в нулі.

Масштабування до мінімальних і максимальних значень - MinmaxScaling

Цей підхід стискає значення від 0 до 1.

X\_scaled = (x - x.min / (x.max - x.min)

Результатом вищезгаданого перетворення є розподіл зі значеннями в межах від 0 до 1. Але середнє значення не зосереджене в нулі, а стандартне відхилення змінюється для кожної ознаки. Форма масштабованого розподілу буде схожа на початкову змінну, але дисперсія може змінюватися, тому не ідентична. Цей підхід масштабування також чутливий до викидів.

У двох словах, MinmaxScaling:

* не центрує середнє значення в 0
* дисперсія змінюється
* не може зберегти форму оригінального розподілу
* Мінімальні та максимальні значення - 0 і 1.
* Чутлива до аномальних значень

*Теоретична частина до практичної роботи №3*

**Етап моделювання**

Моделювання зазвичай проводиться в декілька ітерацій. Як правило, запускається кілька моделей з використанням параметрів за замовчуванням, а потім налаштовують параметри або повертаються до фази підготовки даних для маніпуляцій, необхідних відповідно до обраної ними моделі.

Виберіть техніку моделювання

В якості першого кроку моделювання потрібно вибрати техніку моделювання, яку будете використовувати. Хоча ви, можливо, вже вибрали інструмент на етапі розуміння бізнесу, на цьому етапі ви обираєте конкретну техніку моделювання, наприклад побудова дерева рішень або генерація нейронної мережі із зворотним поширенням. Якщо застосовується кілька методів, виконайте цей етап окремо для кожної техніки.

Відповідь на питання про те, який тип моделювання вибирати, не настільки очевидний, як може здатися на перший погляд. У кожній конкретній ситуації необхідно врахувати цілий ряд факторів та нюансів: Яке співвідношення ефективності моделей та її складності вам потрібно? Які ресурси доступні для навчання моделей? Чи повинна модель бути компактною та зручною для зберігання? Також важливо зрозуміти, що ми можемо помилитися у своєму виборі. Інколи виникає необхідність порівняти різні методи машинного навчання на практиці, щоб зрозуміти, який з них вам більше підходять. Якщо у вас є така можливість, навчіть кілька різних моделей, щоб вибрати ту, яка оптимально відповідає вашим потребам.

Вирішуючи, яку модель (моделі) використовувати, подумайте, як впливають на ваш вибір такі проблеми:

1. Чи вимагає модель розбиття даних на тестові та навчальні набори?
2. У вас достатньо даних для отримання достовірних результатів для певної моделі?
3. Чи модель вимагає певного рівня якості даних? Чи можна досягти цього рівня за наявними даними?
4. Ваші дані належного типу для певної моделі? Якщо ні, чи можна здійснити необхідні перетворення?

*Звіт щодо методу моделювання* - задокументуйте техніку моделювання, яку потрібно використовувати.

Починаючи звужувати обрані вами підходи моделювання, запишіть про процес прийняття рішень. Документуйте будь-які припущення щодо даних, а також будь-які маніпуляції з даними, які відповідають вимогам моделі.

Наприклад, як логістична регресія, так і нейронна мережа вимагають, щоб типи даних були повністю відомі перед виконанням. Подібним чином, моделі прогнозування, такі як C5.0, можуть мати користь від збалансування даних при прогнозуванні правил для рідкісних подій. Створюючи цей тип прогнозування, ви часто можете отримати кращі результати подаючи в модель більш збалансовану підмножину.

*Припущення моделювання* - Багато методів моделювання роблять конкретні припущення щодо даних, наприклад, що всі атрибути мають рівномірний розподіл, відсутні пропущені значення, атрибут класу має бути числовим тощо. Запишіть усі зроблені припущення.

***Як обрати тип моделі для вирішення задачі***

Перш за все, щоб вибрати алгоритм для свого проекту, вам потрібно знати, які існують. Давайте поглибимо ваші знання про різні класифікації.

Алгоритми за стилем навчання

Можна згрупувати алгоритми за їх стилем навчання.

* Supervised learning

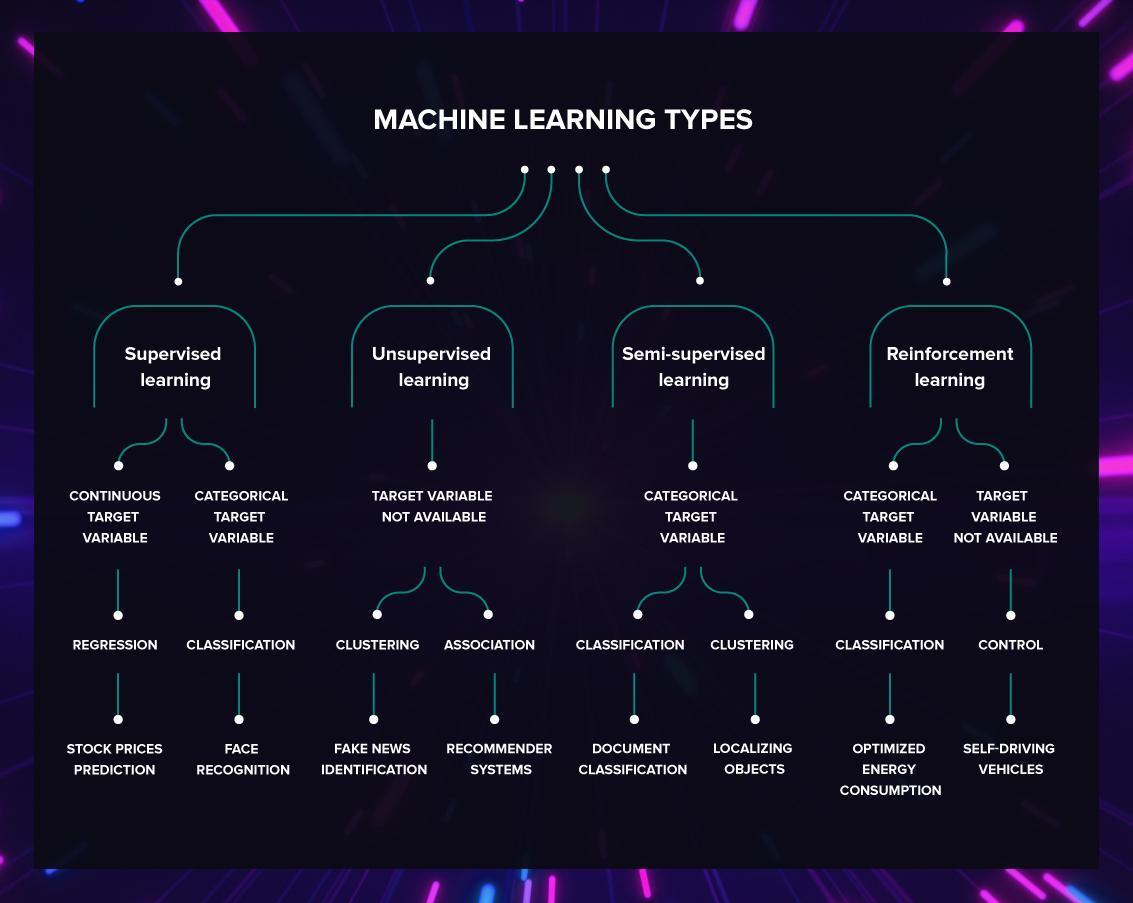
У разі навчання з вчителем моделям потрібен «учитель», який їх «виховує». У цьому випадку фахівець з машинного навчання збирає набір даних і маркує їх. Потім необхідно передати машині навчальний набір та мітки (результат, яких хочемо передбачити). Наступний крок - подивитися, як машині вдається обробити дані тестування. Якщо є допущені помилки, програміст виправляє їх і повторює дію, поки алгоритм не запрацює точно.

* Unsupervised learning

Цей тип машинного навчання не потребує вихователя. Комп'ютер отримує набір даних без міток. Передбачається знаходження закономірностей самостійно. Люди можуть трохи керувати машиною по ходу процесу, надаючи також набір позначених навчальних даних. У цьому випадку це називається *semi-supervised learning*.

* Reinforcement learning

Навчання з підкріпленням відбувається в середовищі, де модель повинна працювати. Навколишнє середовище виступає в ролі вчителя, який надає машині позитивний чи негативний відгук, що називається підкріпленням.



*Рис.6. Класифікація методів машинного навчання за типом навчання*

Для того щоб зрозуміти який з типів навчання нам потрібно використати необхідно детально оглянути доступні дані та проблему, що ми вирішуємо.

1. Якщо наш набір даних містить історичні дані з нашим бажаним значенням – використовуємо *алгоритми навчання з вчителем*. Тобто якщо ми маємо історичну вибірку цін квартир та їх характеристик і нашим завдання є спрогнозувати ціну на квартиру за її характеристиками - використовуємо алгоритм навчання з вчителем.
2. Якщо набір даних не містить нашу цільову змінну, а лише набір інформації про об’єкти і завданням є щось на зразок “Виявити аномальні...”, “Згрупувати”, “Знайти закономірності” – використовуємо *алгоритми навчання без вчителя*. Наприклад ми маємо інформацію про географічні координати активації стартових пакетів певної мобільної мережі, але не маємо розмічених даних щодо назви місця і нам потрібно згрупувати місця активації то для вирішення такого завдання оберемо алгоритми навчання без вчителя.

Наші дані можуть бути частково розмічені (певні набори геолокацій таки матимуть назву магазину чи району), *тоді ми застосовуємо змішаний тип навчання* і додаємо в нашу модель інформацію, що можемо отримати з розмічених даних.

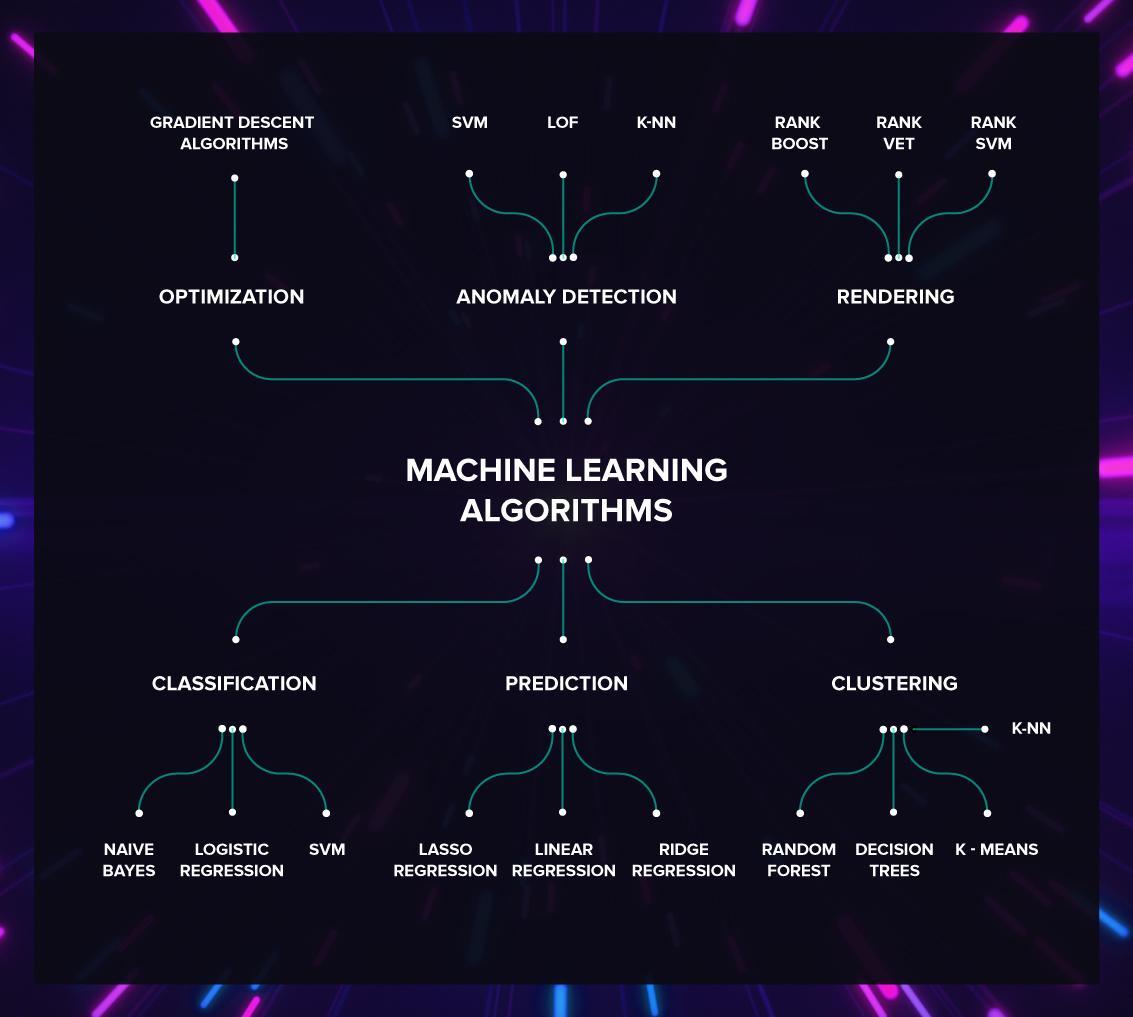
1. Якщо ми не маємо набору історичних даних, але можемо отримати відгук середовища застосування моделі і знаємо певні правила, за якими потрібно поводитись в цьому середовищі – використовуємо *алгоритм навчання з підкріпленням*. Наприклад ми хочемо навчити модель грати в Pocket Man. Ми знаємо, що наш персонаж може рухатися в різні сторони і при цьому залишатися живим і отримувати винагороду, або ж померти. Також ми можемо отримати інформацію про навколишнє середовище. На основі цих даних ми можемо застосувати алгоритми навчання з підкріпленням і отримати модель, що може чудово грати в Pocket Man.

Методи машинного навчання, згруповані за типами проблем

Інший спосіб поділу методів на групи ґрунтується на проблемах, які вони вирішують. Ми поговоримо про класифікацію, регресію, оптимізацію та інші групи алгоритмів.

Ось найпопулярніші алгоритми ML. Іноді вони належать до кількох груп, тому що вони ефективні у вирішенні кількох проблем.

* Логістична регресія (Logistic Regression)
* Лінійна регресія (Linear Regression)
* Дерево рішень (Decision Tree)
* SVM
* Наївний Байєсівський класифікатор (Naive Bayes)
* k-NN
* K-середніх (K-Means)
* Нейромережі (Neural networks)
* Випадковий ліс (Random Forest)
* Алгоритми зменшення розмірності (Dimensionality Reduction)



*Рис.7. Класифікація алгоритмів машинного навчання за типами задач, що вони вирішують*

Класифікація

Класифікація допомагає нам вирішувати широкий спектр проблем. Це дозволяє приймати більш обґрунтовані рішення, розбиратися зі спамом, передбачати, чи поверне позичальник позику, або позначити друзів на фотографії у Facebook.

Ці алгоритми передбачають дискретні мітки змінних. Дискретна цільова змінна має злічену кількість можливих значень і може бути класифікована. Точність прогнозу залежить від обраної вами моделі.

Прикладом задачі класифікації є передбачити чи буде абонент наступний місяць користуватися послугами вашої компанії чи змінить мобільного оператора. Або ж вказати до якого типу

Типовими алгоритмами класифікації є логістична регресія, дерева рішень, ансамблі на основі дерев рішень, Naive Bayes та SVM.

Кластеризація

Іноді вам потрібно розділити об’єкти на категорії, але ви не знаєте, що це за категорії. Класифікація використовує попередньо визначені класи для призначення об’єктів. З іншого боку, кластеризація дозволяє виявляти подібності, які можуть бути між об’єктами, а потім групувати їх відповідно до спільних характеристик. Це механізм, який стоїть за виявленням шахрайства, аналізом документів, групуванням клієнтів тощо. Кластеризація широко використовується в продажах та маркетингу для сегментації клієнтів та персоналізованої комунікації.

K-NN, k-середніх, дерева рішень, випадковий ліс можна використовувати для задач кластеризації.

Передбачення

Спроба з'ясувати зв'язок між двома або більше безперервними змінними є типовою задачею регресії.

*Примітка: Якщо змінна може приймати будь -яке значення між своїм мінімальним і максимальним значенням, вона називається безперервною змінною.*

Прикладом такого завдання є прогнозування цін на житло на основі розміру, місцезнаходження та інших характеристик квартири. Ціна будинку в цьому випадку є безперервною числовою змінною.

Лінійна регресія є найпоширенішим алгоритмом у цій галузі. Алгоритми багатоваріантної регресії, Ridge Regression та регресії LASSO, Дерева рішень використовуються, коли потрібно моделювати зв’язок між більш ніж двома змінними.

Оптимізація

Програмне забезпечення для машинного навчання дозволяє вам забезпечувати підхід, орієнтований на дані, до постійного вдосконалення практично в будь-якій галузі. Ви можете застосувати аналітику використання продукту, щоб дізнатися, як нові функції продукту впливають на попит. Складне програмне забезпечення, оснащене емпіричними даними, допомагає виявити неефективні заходи, що дозволяє уникнути невдалих рішень.

Наприклад, можна використовувати гетерархічну систему управління виробництвом, щоб покращити спроможність динамічної виробничої системи адаптуватися та самоуправлятися. Методи машинного навчання виявляють найкращу поведінку в різних ситуаціях у режимі реального часу, що призводить до постійного вдосконалення системи.

Алгоритми градієнтного спуску зазвичай використовуються в ML для роботи з оптимізацією.

Виявлення аномалій

Фінансові установи щороку втрачають близько 5% доходу від шахрайства. Побудувавши моделі на основі історичних транзакцій, інформації в соціальних мережах та інших джерел даних, можна виявити аномалії, поки не пізно. Це допомагає виявляти та запобігати шахрайським операціям у режимі реального часу, навіть для невідомих раніше видів шахрайства.

Типовими алгоритмами виявлення аномалій є SVM, LOF, k-NN, k-середнє.

Рейтинг

Ви можете застосувати машинне навчання для побудови моделей ранжування. MLR зазвичай передбачає застосування керованих, напівнаглядових або підсилювальних алгоритмів. Прикладом завдання ранжування є такі пошукові системи, як SearchWiki від Google.

Прикладами алгоритмів ранжування є RankNet, RankBoost, RankSVM та інші.

Рекомендація

Системи рекомендацій пропонують користувачам цінні пропозиції. Цей метод приносить корисність користувачам, а також приносить користь компаніям, оскільки мотивує їхніх клієнтів купувати більше або вивчати більше вмісту.

Елементи ранжуються відповідно до їх актуальності. Користувачу відображаються найбільш релевантні. Релевантність визначається на основі історичних даних. Ви знаєте, як це працює, якщо ви коли-небудь дивилися що-небудь на Youtube або Netflix. Системи пропонують вам відео, подібне до того, що ви вже переглядали.

Основними алгоритмами, що використовуються для систем-рекомендацій, є алгоритми спільної фільтрації та системи на основі вмісту.

**Як вибрати методи машинного навчання для вирішення вашої проблеми**

Як знайти найкращий алгоритм машинного навчання для вашої проблеми? Можна використовувати три основні підходи.

*Навчання на основі завдань*

Категоризуйте свою проблему. Можна класифікувати завдання за вхідними та результуючими даними.

За вхідними:

* Якщо у вас є набір розмічених даних або ви можете підготувати такий набір, це сфера навчання з вчителем.
* Якщо вам все-таки потрібно визначити взаємозалежність, це проблема навчання без вчителя.
* Якщо вам потрібна модель для взаємодії з середовищем, ви застосуєте алгоритм навчання з підкріпленням.

За результатами:

* Якщо на виході моделі є число, це проблема регресії.
* Якщо на виході моделі є клас і кількість очікуваних класів відома, це проблема класифікації.
* Якщо результатом моделі є клас, але кількість очікуваних класів невідома, це проблема кластеризації.
* Якщо вам потрібно покращити продуктивність, це оптимізація.
* Якщо ви хочете, щоб система пропонувала варіанти на основі історії дій, це проблема рекомендацій.
* Якщо ви хочете отримати інсайти з даних, застосуйте моделі розпізнавання образів.
* Якщо ви хочете виявити проблеми, використовуйте алгоритми виявлення аномалій.



*Рис.8. Як вибрати метод машинного навчання залежно від початкової задачі*

*Зрозумійте свої дані*

Процес вибору алгоритму не обмежується категоризацією проблеми. Вам також потрібно придивитися до ваших даних, оскільки вони відіграють важливу роль у виборі правильного алгоритму для вирішення проблеми. Деякі алгоритми нормально функціонують з меншими наборами вибірок, тоді як інші вимагають величезної кількості прикладів у датасеті. Деякі алгоритми працюють з категоріальними даними, тоді як інші працюють лише з числовим ознаками.

Розуміння ваших даних вимагає певних кроків:

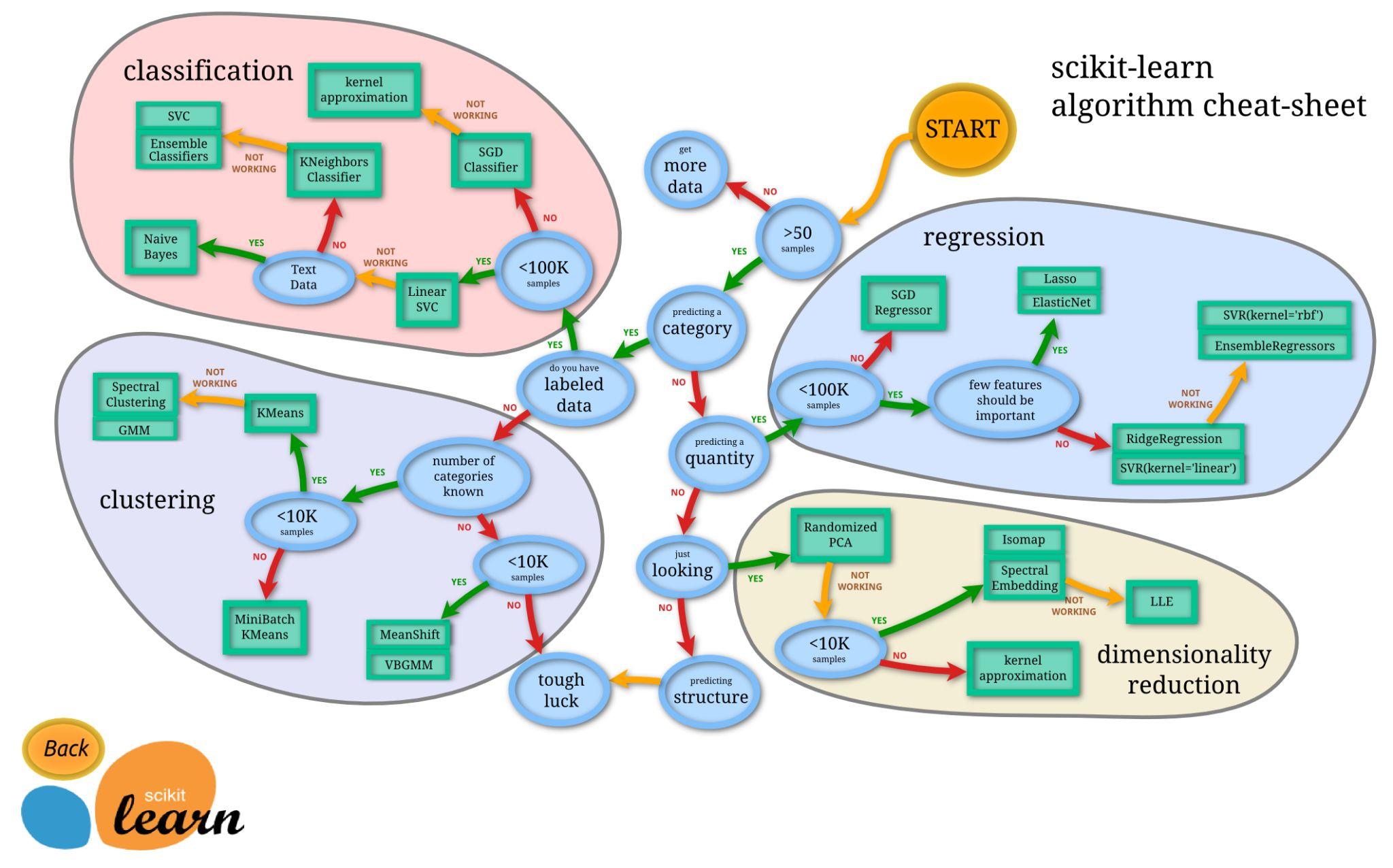
1. Опрацювання. Складовими елементами опрацювання даних є попередня обробка, профілювання, очищення, збирання даних з різних внутрішніх та зовнішніх джерел.
2. Інженерія ознак. Вам потрібно перетворити необроблені дані в ознаки, які можуть представляти основну проблему для моделей прогнозування. Це допомагає підвищити точність і швидше отримати бажані результати.
3. Вибір алгоритму - це комплексне завдання, яке вимагає аналізу різноманітних факторів.

*Інші речі, які можуть вплинути на вибір моделі:*

* Точність моделі;
* Інтерпретація моделі;
* Складність моделі;
* Масштабованість моделі;
* Час, необхідний для побудови, навчання та валідації моделі;
* Час, необхідний для прогнозування за допомогою моделі;
* Чи модель відповідає вашим бізнес-цілям.

*Підхід проб і помилок*

Іноді проблема є надто складною, і ви не знаєте, з чого почати. Більше однієї моделі здаються підходящими, і важко передбачити, яка з них виявиться найбільш ефективною. У цьому випадку ви можете протестувати декілька моделей і оцінити їх.



*Рис.9. Дерево рішень для вибору ML моделі*

Валідація моделі

Перш ніж побудувати модель, вам потрібно створити процедуру або механізм для перевірки якості та достовірності моделі. Наприклад, у контрольованих задачах інтелектуального аналізу даних, таких як класифікація, зазвичай використовується точність як показник якості для моделей. Як правило, ви розділяєте набір даних на навчальний і тестові набори, навчаєте модель на навчальному наборі і оцінюєте її якість на окремому тестовому або валідаційному наборі.

Створення комплексного проекту тестування складається з двох частин:

* Опис критеріїв "правильності" моделі
* Визначення даних, на яких ці критерії будуть перевірятися

Правильність моделі можна оцінити кількома способами. Для моделей класифікації, таких як C5.0 та C&R Tree, вимірювання точності (правильності) зазвичай оцінюють коефіцієнт коректних передбачень конкретної моделі. Для моделей регресії – оцінюють відхилення прогнозованого значення від істинного. Для моделей що навчаються без вчителя, таких як кластерні мережі Кохонена, вимірювання можуть включати такі критерії, як відстань між прогнозованими кластерами та дисперсія у кластері.

*Дизайн тесту* - це опис кроків для тестування навчених моделей. Оскільки моделювання - це ітераційний процес, важливо знати, коли припинити коригування параметрів та спробувати інший метод чи модель. Основним компонентом плану є визначення способу поділу наявного набору даних на набори даних навчання, тестування та валідації.

Створюючи проект тесту, врахуйте наступні питання:

1. Які дані будуть використані для тестування моделей? Ви розділили дані на навчальні/тестові набори?
2. Як ви можете оцінити модель (наприклад, дерево рішень для задачі регресії)?
3. Скільки разів ви готові повторно запустити модель зі зміненими параметрами, перш ніж спробувати інший тип моделі?

Оцініть модель

Коли у вас є набір початкових моделей, уважніше подивіться на них, щоб визначити, які з них є точними чи достатньо ефективними, щоб бути остаточними. Остаточна модель може означати кілька речей, таких як "готова до розгортання" або "виявляє неочевидні закономірності". Використання плану тестування, який ви створили раніше, може допомогти зробити цю оцінку з точки зору вашої організації.

Інтерпретуйте моделі відповідно до ваших знань у галузі, критеріїв успіху видобутку даних та бажаного дизайну тесту. Судіть про успішність застосування методів моделювання та виявлення технічно, потім зверніться до бізнес-аналітиків та експертів у галузі, щоб обговорити результати у бізнес-контексті. Це завдання розглядає лише моделі, тоді як фаза оцінки також враховує всі інші результати, які були отримані в ході проекту.

На цьому етапі ви повинні ранжувати моделі та оцінювати їх відповідно до критеріїв оцінки. Ви повинні максимально враховувати цілі та критерії успіху бізнесу тут. Перевірте результати на основі свого розуміння проблеми бізнесу. Зверніться до аналітиків даних чи інших експертів, які можуть зрозуміти актуальність конкретних результатів.

Виходячи з того, що ви дізналися під час оцінки моделі, настав час ще раз поглянути на моделі. Тут у вас є два варіанти:

1. Налаштуйте параметри існуючих моделей.
2. Виберіть іншу модель для вирішення проблеми.

В обох випадках ви повернетесь до завдання тренування моделей і повторите, поки результати не будуть успішними. Не турбуйтеся про повторення цього кроку. Це хороший аргумент для побудови кількох моделей одночасно та порівняння результатів перед коригуванням параметрів для кожної.

*Для чого потрібно оцінювати модель?*

Одним з основних завдань при побудові будь -якої моделі машинного навчання є оцінка її продуктивності. Це принципово, а також дуже важко.

Отже, як можна оцінити успіх моделі машинного навчання? Як знати, коли припинити навчання та оцінювання та сказати, що модель натренована добре?

Хоча підготовка даних та навчання моделі машинного навчання є ключовим етапом у процесі машинного навчання, не менш важливо виміряти продуктивність цієї навченої моделі. Наскільки добре модель узагальнює невидимі дані, це те, що визначає адаптивні та неадаптивні моделі машинного навчання.

Використовуючи різні показники для оцінки ефективності, ми повинні мати можливість покращити загальну спроможність прогнозування нашої моделі, перш ніж ми розгорнемо її для виробництва.

Без належного оцінювання МЛ моделі з використанням різних показників може виникнути проблема, коли результати передбачення моделі на нових даних будуть некоректними.

Це відбувається тому, що інколи наші моделі не навчаються, а навпаки запам'ятовують; отже, вони не можуть добре узагальнювати нові дані. Для початку давайте визначимо ці три важливі терміни:

* *Навчання*: стан, коли навчена ML модель з майже однаковою точністю/похибкою передбачає нові дані, та дані на яких навчалася.
* *Запам'ятовування* (*перенавчання*): стан, коли навчена ML модель з високою точністю/ малою похибкою передбачає дані, на яких навчалася, але не може передбачити результат на нових даних.
* *Узагальнення*: Можна визначити як здатність ML моделі застосовувати навчання до нових даних. Без узагальнення немає навчання, лише запам’ятовування. Але зауважте, що узагальнення є залежним від цілі моделювання - наприклад, добре навчена модель розпізнавання зображень на зображеннях тварин у зоопарках може погано узагальнити зображення автомобілів та будівель.

На зображеннях нижче показано, як покладаючись на точність моделі під час навчання, ми можемо зіткнутись з поганою роботою під час перевірки.

Diagram

Description automatically generated

*Рис.10. Результати навчання моделі.*

***Метрики валідації моделі***

Показники оцінювання прив'язані до завдань машинного навчання. Існують різні показники для завдань класифікації, регресії, ранжування, кластеризації, тощо. Деякі показники, такі як точність, корисні для кількох завдань. Класифікація, регресія та ранжування є прикладами навчання з вчителем, яке становить більшість моделей машинного навчання.

1. Метрики для класифікації

Під час прогнозування класів можна виділити чотири види результатів.

True Positive (Істино позитивне) - це коли ви передбачаєте, що спостереження належить до класу, і воно насправді належить до цього класу.

True Negative (Істино негативне) - це коли ви передбачаєте, що спостереження не належить до класу, і воно насправді не належить до цього класу.

False Positive (Хибно позитивні) - коли ви передбачаєте, що спостереження належить до класу, а насправді це не так.

False Negative (Хибно негативні) - коли ви передбачаєте, що спостереження не належить до класу, а насправді воно є.

Ці значення утворюють матрицю невідповідностей (confusion matrix). Матриця невідповідностей є прикладом для випадку бінарної класифікації. Матриця повинна бути згенерована після того, як зробили прогнози для своїх тестових даних, а потім визначили кожне передбачення як один із чотирьох можливих результатів, описаних вище.

Graphical user interface, application

Description automatically generated

*Рис.11. Матриця невідповідностей для бінарної класифікації*

Ви також можете розширити цю матрицю невідповідностей, щоб побудувати прогнози класифікації для багатьох класів. Нижче наведено приклад матриці для класифікації прикладів з набору квітів Ірис.

Application

Description automatically generated with medium confidence

*Рис.12. Матриця невідповідностей для мультикласифікації*

Матриця невідповідностей є основою для інших типів метрик.

*Рис.13. Метрики класифікації, що базуються на матриці невідповідностей*Graphical user interface, application

Description automatically generated

*Accuracy (Точність)* визначається як відсоток правильних прогнозів для тестових даних. Її можна легко обчислити, поділивши кількість правильних передбачень на кількість загальних передбачень.

*Precision (прогностична значущість позитивного результату)* визначається як частка відповідних прикладів (істино позитивних) серед усіх прикладів, для яких передбачалося, що вони належать до певного класу.

*Recall* *(рівень хибного виявлення)* визначається як частка прикладів, для яких передбачалося, що вони належать до класу, щодо всіх прикладів, які дійсно належать до класу.

Наступний рисунок візуалізує різницю між *Precision* та *Recall*.

A picture containing chart

Description automatically generated

*Рис.14. Візуалізація матриці невідповідностей*

A picture containing chart

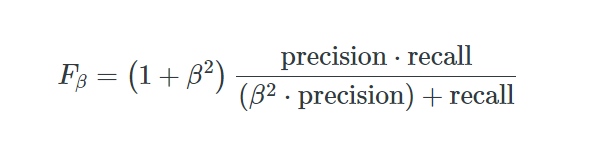
Description automatically generated

*Рис.15. Різниця між Precision та Recall*

*Precision* та *Recall* корисні у випадках, коли класи розподілені нерівномірно. Загальний приклад - це розробка алгоритму класифікації, який передбачає, чи є у когось захворювання. Якщо лише невеликий відсоток населення (скажімо, 1%) має це захворювання, ми могли б побудувати класифікатор, який завжди передбачає, що у людини немає цієї хвороби, і ми б побудували модель, яка на 99% точна і 0% корисна .

Однак, якби ми виміряли *Precision* та *Recall* цього марного предиктора, було б зрозуміло, що з нашою моделлю щось не так. У цьому прикладі Recall гарантує, що ми не забуваємо про людей, які хворіють на цю хворобу, тоді як Precision гарантує, що ми не помилково класифікуємо занадто багато людей як хворі, коли вони цього не роблять. Очевидно, ви не хотіли б моделі, яка неправильно передбачає, що у людини є рак (це спричинить потребу у болісному і дорогому лікуванні хвороби, якої вона не мала), але ви також не хочете неправильно передбачити, що людина не хвора на рак, хоча насправді хворіє. Таким чином, важливо оцінити як *Precision* так і *Recall*.

Зрештою, приємно мати одну цифру для оцінки моделі машинного навчання так само, як ви отримуєте єдину оцінку на тесті в університеті. Таким чином, є сенс поєднати показники *Precision* та *Recall*; загальний підхід до об'єднання цих показників відомий як *f-оцінка.*

**

Параметр β дозволяє нам контролювати компроміс важливості між *Precision* та *Recall.* β <1 більше зосереджується на *Precision*, тоді як β> 1 більше зосереджується на *Recall*.

У деяких випадках необхідно отримати найкращу *Precision* та *Recall* одночасно, без компромісів. Тоді ми використовуємо β =1 , така метрика називається F1 і виглядає так:

Diagram

Description automatically generated

Чим вище оцінка F1, тим більшою є спрогнозована здатність моделі класифікації. Оцінка, близька до 1, означає ідеальну модель, однак, оцінка, близька до 0, свідчить про зниження прогнозних можливостей моделі.

*AUC-ROC (площа під кривою ROC)*

Крива ROC (крива робочих характеристик моделі) - це графік, що показує ефективність моделі класифікації на всіх порогах класифікації. Ця крива зображує два параметри:

*True Positive Rate (Істинно позитивний коефіцієнт)*- кількість істинно позитивних передбачень, поділена на суму істинно позитивних і хибно негативних (всіх прикладів, що є позитивними). Він описує наскільки добре модель передбачає позитивний клас, коли реальний результат позитивний.

*False Positive Rate (Хибний позитивний коефіцієнт)* - кількість хибно позитивних передбачень, поділена на суму кількості хибно позитивних та істино негативних передбачень (усіх прикладів, що мають реальне негативне значення).

Криві ROC зображують графік між True positive rate та false-positive rate. Ці графіки генеруються за різними пороговими значеннями класифікації. Отже, якщо у нас низький поріг класифікації (threshold), ми можемо класифікувати більше позицій як позитивні, збільшуючи при цьому як хибнопозитивні, так і істино позитивні результати. Наступний малюнок показує типову криву ROC.

Diagram

Description automatically generated

*Рис.16. Приклад кривої ROC*

Точки на кривій ROC можна обчислити, оцінивши модель машинного навчання, таку як логістична регресія, але це буде неефективно. Вирішенням цієї проблеми є алгоритм на основі сортування, відомий як AUC.

AUC - це абревіатура від зони під кривою. Він обчислює всю площу 2D під кривою ROC.

Більш інтуїтивно зрозумілим способом, це графік FPR (*False Positive Rate*) на осі x та TPR (*True Positive Rate*) на осі y для різних порогових значень від 0,0 до 1.

Графік AUC ROC - одна з найпопулярніших метрик, що використовуються для визначення можливостей прогнозування моделі машинного навчання. Нижче наведено кілька причин використання AUC ROC:

* Криві різних моделей машинного навчання можна перевірити за допомогою різних порогів.
* Можливості прогнозування моделі узагальнюються за площею під кривою (AUC).
* AUC вважається масштабованим варіантом, він вимірює ранг прогнозів, а не його абсолютні значення
* AUC завжди зосереджується на якості навичок моделі щодо прогнозування, незалежно від того, який поріг вибрано.

1. Метрики для регресії

Показники оцінки для регресійних моделей сильно відрізняються від наведених вище метрик, які ми обговорювали для моделей класифікації, тому що зараз ми передбачаємо в безперервному діапазоні замість дискретної кількості класів. Якщо ваша регресійна модель передбачає, що ціна будинку становитиме 400 тисяч доларів, а продаватиметься за 405 тисяч доларів, це досить хороший прогноз. Однак у прикладах класифікації нас цікавило лише те, чи було передбачення правильним чи неправильним, чи не було можливості сказати, що прогноз був «досить хорошим». Таким чином, у нас є різний набір оціночних показників для регресійних моделей.

*Середня абсолютна помилка (Mean Absolute Error)* - це середнє значення різниці між вихідними та передбачуваними значеннями. Це дає нам міру того, наскільки прогнози були далекі від реального результату. Однак вона не дає нам уявлення про напрямок помилки, тобто про те, чи наше передбачення більше від істиного чи навпаки менше. Математично це представлено у вигляді:

Text

Description automatically generated

*Середня квадратична помилка (Mean Squared Error)* дуже схожа на середню абсолютну помилку, єдина відмінність полягає в тому, що MSE бере середнє значення квадрату різниці між вихідними значеннями та прогнозованими значеннями. Перевага MSE в тому, що простіше обчислити градієнт (необхідно для навчання моделі), тоді як середня абсолютна помилка вимагає складних інструментів лінійного програмування для обчислення градієнта. Оскільки ми приймаємо квадрат помилки, ефект більших помилок стає більш вираженим, ніж менша помилка, отже, тепер модель може більше зосередитися на більших помилках.

Text

Description automatically generated

*Корінь з середньоквадратичної помилки (Root mean squared error)* є найпопулярнішою метрикою, що використовується у задачах регресії. RMSE визначається стандартним відхиленням помилок прогнозування. Ці помилки передбачення іноді називають залишками. Залишки - це в основному вимірювання відстані точок даних від лінії регресії.

Graphical user interface, application

Description automatically generated

Простіше кажучи, RMSE показує, наскільки добре зосереджена точка даних навколо лінії регресії. З RMSE передбачається, що залишки є неупередженими і слідують нормальному розподілу. Нижче наведено деякі цікаві моменти, пов'язані з коренем середньоквадратичної помилки.

* RMSE працює ефективно, коли ми маємо справу з великим обсягом даних.
* Відповідно до математичної формули RMSE, «квадратний корінь» показує велике відхилення числа.
* Перед використанням RMSE переконайтеся, що у наборі даних немає аномалій, оскільки на RMSE сильно впливають викиди.
* RMSE має вищий ваговий коефіцієнт, а також карає помилки порівняно з іншими оціночними показниками (важливо при тренуванні моделі).

1. Метрики для кластеризації

Порівняно з класифікацією, важко визначити якість результатів кластеризації. Показник оцінки не може залежати від міток, а лише від “правильності” поділу. Більше того, коли ми використовуємо кластеризацію, ми зазвичай не маємо справжніх міток спостережень.

*Коефіцієнт оцінки силуету (silhouette score)* обчислюється за допомогою середньої відстані між кластерами та середньої відстані найближчого кластера для кожного прикладу з вибірки (датасету).

A picture containing diagram

Description automatically generated

Відстань «Силует» показує, наскільки відстань між об’єктами одного класу відрізняється від середньої відстані між об’єктами з різних кластерів. Значення оцінки силуету лежать між -1 і +1. Тому, чим вище значення силуету, тим кращі результати кластеризації.

Також з оцінкою силуету ми також можемо визначити оптимальну кількість кластерів, взявши кількість кластерів, які максимізують коефіцієнт силуету.

***Розділення ваших даних***

Основою всіх методів перевірки є поділ ваших даних під час навчання моделі. Причина цього - зрозуміти, що станеться, якщо ваша модель зіткнеться з даними, яких вона раніше не бачила.

Якщо всі дані використовуються для навчання моделі, і коефіцієнт помилок оцінюється на основі результату порівняно з фактичним значенням з того самого набору даних навчання, ця помилка називається помилкою повторного заміщення. Цей метод називається методом перевірки повторного заміщення (resubstitution).

1)Утриманий набір (Holdout set)

Щоб уникнути помилки повторного заміщення, дані розбиваються на два різні набори, позначені як навчальний та тестовий. Це може бути розділення 70/30 або 80/20 (тобто 70% всіх наявних ми включаємо в тренувальну вибірку і 30% в тестовому).

Перевага такого підходу в тому, що ми можемо побачити, як модель реагує на раніше невидимі дані. Однак що робити, якщо в одній підмножині наших даних є лише люди певного віку чи рівня доходу? Зазвичай це називається упередженістю вибірки (sampling bias): *випадок, коли розподіл різних класів даних у навчальній та тестовій вибірці буде різний.* Щоб виправити це, навчальний та тестовий набір даних створюється з однаковим розподілом різних класів даних. Цей процес називається стратифікацією (stratification).

Щоб вирішити цю проблему, можна створити також додатковий набір даних. Дані розділяються на 3 набори: навчальний, валідаційний, тестовий. Співвідношення відповідно 70/10/20. Валідаційний (Holdout set) набір використовується для перевірки моделі ще в процесі навчання. Тоді як тестовий набір – після навчання моделі.

A picture containing icon

Description automatically generated

*Рис. 17. Поділ навчальних даних.*

2.Перехресна перевірка k-Fold ( k-Fold validation)

Щоб мінімізувати упередженість вибірки, ми можемо думати про валідацію моделі дещо інакше. Що робити, якщо замість того, щоб робити єдиний поділ, ми зробимо багато розділень і перевірятимемо модель на всіх комбінаціях цих поділів?

Тут з’являється перехресна перевірка k-кратності. Вона розбиває дані на k наборів, потім навчає дані на k-1 наборі і перевіряє на одному наборі, який не було подано для навчання. Це відбувається для всіх комбінацій і усереднює результат для кожної вибірки.

Table

Description automatically generated with medium confidence

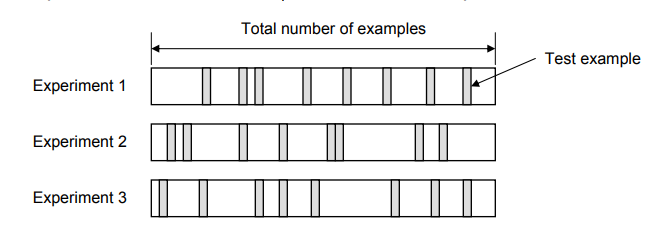
*Рис.18. K-fold validation*

Перевагою є те, що всі приклади використовуються як для навчання, так і для перевірки, і кожен приклад використовується один раз для перевірки. Ми зазвичай вибираємо або i = 5, або k = 10, оскільки вони знаходять хороший баланс між обчислювальною складністю та точністю перевірки.

ПОРАДА: Оцінки кожної підвибірки з методів перехресної перевірки містять більше інформації, ніж здається на перший погляд. Вони в основному використовуються для простого отримання середньої точності/похибки моделі. Однак можна також подивитися на дисперсію або стандартне відхилення отриманих метрик, оскільки це дасть інформацію про стабільність моделі на різних вхідних даних.

3. Випадкова підвибірка

У цій техніці кілька наборів даних випадковим чином вибираються з набору даних і об’єднуються для формування тестового набору даних. Решта даних формує навчальний набір. Наступна діаграма представляє цю техніку. Частота помилок моделі - це середнє значення частоти помилок кожної ітерації.



*Рис.19. Методика перевірки випадкової підвибірки*

4. Перехресна перевірка з одноразовим використанням (Leave-one-out Cross-Validation LOOCV)

Варіантом k-Fold валідації є перехресна перевірка з одноразовим використанням (LOOCV). LOOCV використовує кожну вибірку в даних як окремий тестовий набір, тоді як усі решта зразків формують навчальний набір. Цей варіант ідентичний k-кратній перехресній перевірці, якщо k = n (кількість спостережень).

A picture containing text, electronics, keyboard, mounted

Description automatically generated

*Рис.20. Leave-one-out Cross-Validation*

ПРИМІТКА: LOOCV обчислювально дуже затратна, оскільки моделі потрібно пройти навчання n разів. Використовуйте цей метод лише в тому випадку, якщо дані невеликі або якщо ви можете впоратися з такою кількістю обчислень.

5. Перехресна перевірка "залишити одну групу" (Leave-one-group-out Cross-Validation (LOGOCV))

Проблема з k-Fold валідацією виникає коли є потреба, щоб кожен набір містив лише одну групу. Припустимо, що у вас є набір даних із 20 компаній та їх клієнтів, і ви хочете передбачити успіх цих компаній. Щоб зберегти набори даних "чистими" (містять лише одну компанію), необхідно створити набір для кожної компанії. Таким чином, ви створюєте версію k-Fold валідації та LOOCV, де залишаєте одну компанію/групу.

A screenshot of a computer

Description automatically generated with low confidence

*Рис.21. Leave-one-group-out Cross-Validation*

5. Перехресна валідація часового ряду

Тепер, що було б, якби ви використовували k-Fold для даних часових рядів? Перенавчання (overfit) буде серйозною проблемою, оскільки ваші дані про навчання можуть містити інформацію з майбутнього. Важливо, щоб усі ваші дані про навчання відбувалися до ваших тестових даних (відносно часових показників).

Одним із способів перевірки даних часових рядів є використання k-Fold валідації та перевіряти, що в кожній складовій навчальні дані мають місце перед даними тесту.

A picture containing bar chart

Description automatically generated

*Рис.22. Валідація для моделей часового ряду*

Обов’язково впорядковуйте свої дані відповідно до індексу часу/дати, який ви використовуєте. Таким чином поділ на роботи буде створено з врахуванням часових показників. Наприклад ви маєте дані щоденних покупок парасольок. Тоді вам необхідно відсортувати дані за зростанням дати і здійснювати поділ на відсортованих даних.

Порівняння моделей

Коли ви вважаєте одну модель кращою за іншу? Якщо точність однієї моделі перевищує іншу, чи це достатня причина для вибору найкращої моделі?

Для виконання практичних робіт ми будемо вважати цю умову достатньою, але в реальному житті інколи потрібно використати складніші перевірки.

Як data scientist, я хочу переконатися, що я розумію, чи є модель насправді значно точнішою за іншу. На щастя, існує багато методів, які застосовують статистику до вибору моделей машинного навчання. Одними з таких методів є тест Уілкоксона (Wilcoxon signed-rank test), Тест Макнемара та 5x2CV paired t-test.

Перевірка моделі може бути складним питанням, оскільки для вибору правильної процедури вона потребує значного розуміння даних.

*Теоретична частина до практичної роботи №4*

**Оптимізація моделі**

У моделей машинного навчання існують два види параметрів:

1. Параметри моделі - це значення, які модель вивчає під час тренування на основі вхідних даних (наприклад, ваги в нейронній мережі). Ці параметри є частиною моделі і визначають, як модель повинна реагувати на вхідні дані. Параметри моделі зазвичай налаштовуються під час тренування з використанням оптимізаційних алгоритмів, таких як стохастичний градієнтний спуск.
2. Гіперпараметри - це значення, які встановлюються до початку процесу навчання моделі та визначають архітектуру та процес навчання моделі (наприклад, кількість прихованих шарів в нейронній мережі, розмір міні-пакету для навчання). Гіперпараметри визначають, як модель повинна бути налаштована для кращої продуктивності.

Основна різниця між параметрами та гіперпараметрами полягає в тому, що параметри навчаються під час тренування моделі, тоді як гіперпараметри встановлюються до початку процесу навчання та не змінюються в процесі тренування моделі. Гіперпараметри зазвичай встановлюються на основі експертного досвіду, здорового глузду та експериментів.

Оптимізація гіперпараметрів є важливим етапом у процесі розробки моделей машинного навчання. Гіперпараметри визначають архітектуру та процес навчання моделі, тому їх правильне налаштування може значно покращити продуктивність моделі.

Оптимізація гіперпараметрів допомагає знайти найкращі значення для гіперпараметрів моделі, що може покращити точність прогнозування та зменшити ризик перенавчання або недонавчання моделі. Крім того, правильно налаштовані гіперпараметри можуть покращити швидкість навчання та ефективність використання ресурсів.

Процес оптимізації гіперпараметрів може бути складним та часовим, тому для цього використовуються різні методи, такі як перехресна перевірка (cross-validation), решітчастий пошук (grid search), випадковий пошук (random search), байєсовська оптимізація (Bayesian optimization) та інші. Кожен з цих методів має свої переваги та недоліки та може використовуватися в залежності від конкретної задачі та обмежень ресурсів.

**Як налаштувати гіперпараметри? Як знайти найкращі гіперпараметри?**

Вибір правильної комбінації гіперпараметрів вимагає розуміння гіперпараметрів та бізнес-використання моделі. Технічно, є два способи налаштування гіперпараметрів.

1. Ручне налаштування гіперпараметрів.

Ручне налаштування гіперпараметрів передбачає експериментування з різними наборами гіперпараметрів вручну, тобто кожне випробування з набором гіперпараметрів буде виконуватись вами. Ця техніка потребує надійного трекера експериментів, який може відстежувати різні змінні, відображати зображення, логи та системні метрики.

*Переваги ручної оптимізації гіперпараметрів:*

* Ручне налаштування гіперпараметрів означає більший контроль над процесом.
* Якщо ви досліджуєте або вивчаєте вплив гіперпараметрів на ваги мережі, то мануальне налаштування буде розумним варіантом.

*Недоліки оптимізації гіперпараметрів вручну:*

* Мануальний підбір є трудомістким процесом, оскільки може бути багато спроб, і відстеження їх може виявитися коштовним та зайняти багато часу.
* Це не є дуже практичним підходом, коли є багато гіперпараметрів, які потрібно врахувати.

1. Автоматизована настройка гіперпараметрів

Автоматизований підбір гіперпараметрів використовує вже існуючі алгоритми для автоматизації процесу. Слід дотримуватися наступних кроків:

* Спочатку визначте набір гіперпараметрів та межі значень цих гіперпараметрів (зауваження: кожен алгоритм потребує, щоб цей набір був конкретною структурою даних, наприклад, словниками, що є поширеними при роботі з алгоритмами).
* Потім алгоритм виконує важку роботу за вас. Він запускає ті експерименти та отримує найкращий набір гіперпараметрів, які дадуть оптимальні результати.

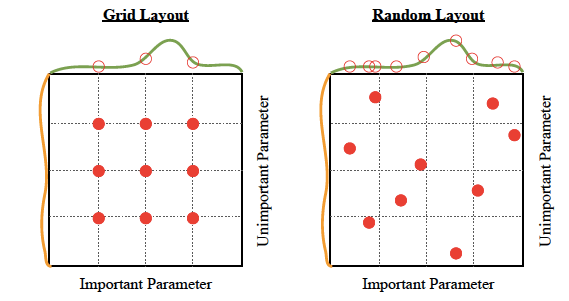
**Методи налаштування гіперпараметрів**

1. Випадковий пошук

У методі випадкового пошуку ми створюємо сітку можливих значень для гіперпараметрів. Кожна ітерація спробує випадкову комбінацію гіперпараметрів з цієї сітки, реєструє продуктивність і, нарешті, повертає комбінацію гіперпараметрів, що забезпечила найкращу продуктивність.

1. Пошук за сіткою

У методі пошуку за сіткою ми створюємо сітку можливих значень для гіперпараметрів. Кожна ітерація спробує комбінацію гіперпараметрів в певному порядку. Модель обчислюється на кожній можливій комбінації гіперпараметрів і реєструє продуктивність моделі. Нарешті, повертає найкращу модель з найкращими гіперпараметрами.



*Рис. 23 Випадковий пошук та пошук гіперпараметрів по сітці.*

1. Оптимізація Байєса

Налаштування та знаходження правильних гіперпараметрів для вашої моделі - це задача оптимізації. Ми хочемо мінімізувати функцію втрат нашої моделі, змінюючи параметри моделі. Оптимізація Байєса допомагає знайти мінімальну точку за мінімальну кількість кроків. Оптимізація Байєса також використовує функцію здобутку, яка направляє вибірку в області, де ймовірно поліпшення порівняно з поточним кращим спостереженням.

1. Деревоорієнтовані оцінювачі Парзена

Ідея оптимізації на основі дереваорієнтованих оцінювачів Парзена подібна до оптимізації Байєса. Замість того, щоб знаходити значення p (y | x), де y - функція, яку потрібно зменшити (наприклад, втрату перевірки), а x - значення гіперпараметра, така оптимізація моделює P (x | y) та P (y). Одним з великих недоліків деревоорієнтованих оцінювачів Парзена є те, що вони не моделюють взаємодії між гіперпараметрами. Однак така оптимізація дуже добре працює на практиці і був перевірений у більшості областей.

**Підбір оптимальних гіперпараметрів моделі машинного навчання** повинен базуватися на об'єктивних метриках, які відображають якість моделі. Однак, вибір правильної метрики може бути залежний від типу завдання, яке вирішується.

Наприклад, якщо метою є класифікація, то основною метрикою може бути точність (accuracy), F1-міра (F1-score) або AUC-ROC (Area Under the Receiver Operating Characteristic curve). Якщо ж метою є регресія, то основними метриками можуть бути середня квадратична помилка (Mean Squared Error, MSE), середня абсолютна помилка (Mean Absolute Error, MAE), коефіцієнт детермінації (R-squared) та інші.

При виборі метрики слід також враховувати особливості конкретного завдання та даних, на яких буде тренуватися модель. Наприклад, якщо класи нерівномірно розподілені, то точність може бути непоказовою метрикою, і в такому випадку можна розглянути F1-міру або AUC-ROC.

Крім основної метрики, також можна врахувати додаткові метрики, які допоможуть краще зрозуміти якість моделі. Наприклад, для задачі класифікації можна виводити матрицю помилок (confusion matrix) або ROC-криву, а для задачі регресії - діаграму розкиду (scatter plot) або графік залишкових помилок (residual plot).

У будь-якому випадку, метрика повинна бути об'єктивною, повторюваною та відображати якість моделі.

Основні пункти, на які слід звернути увагу при оптимізації гіперпараметрів моделі:

1. Вибір метрики: важливо вибрати метрику, яка відображає цілісний ефект моделі на даному завданні. Ця метрика має бути якомога більш чутливою до зміни гіперпараметрів моделі.
2. Призначення області пошуку: визначення простору гіперпараметрів, які слід оптимізувати, та їх діапазонів.
3. Вибір методу оптимізації: слід вибрати метод оптимізації, який підходить до вашого завдання та обмежень. Різні методи оптимізації можуть мати різні переваги та недоліки залежно від конкретного завдання.
4. Обережність з перенавчанням: перенавчання може виникнути, коли модель підлаштовується до даних тренування занадто докладно, що може призвести до погіршення результатів на нових даних. Для уникнення перенавчання можна використовувати крос-валідацію та інші методи.
5. Врахування бізнес-вимог: Підбір гіперпараметрів повинен відповідати бізнес-вимогам. Наприклад, якщо вам потрібно зробити прогнози на кілька наступних місяців, то гіперпараметри повинні бути підібрані так, щоб модель була стійкою до змін в даних, а не тільки здатна прогнозувати точно на попередній тестовій вибірці.
6. Експерименти з гіперпараметрами повинні бути ретельно задокументовані: Кожен експеримент з гіперпараметрами повинен бути детально задокументований, включаючи вхідні дані, гіперпараметри та результати. Це допоможе зрозуміти, які гіперпараметри працюють краще для вашої задачі та уникнути повторних експериментів в майбутньому.
7. Критичне мислення: При підборі гіперпараметрів потрібно мати критичне мислення та не приймати занадто спрощених рішень. Наприклад, вибір гіперпараметрів за допомогою рандомної стратегії може призвести до кращих результатів у деяких випадках, але це не означає, що він є оптимальним для вашої конкретної задачі. Підбір гіперпараметрів є складним процесом, який потребує врахування багатьох факторів, тому важливо бути критичним і не приймати занадто простих рішень.

**Незбалансовані класи**

Незбалансовані класи в машинному навчанні відносяться до випадку, коли кількість прикладів в кожному класі в навчальному наборі даних не рівна. Така нерівномірність може мати наслідком погану точність алгоритмів машинного навчання, оскільки модель може бути зіскочена на клас з більшою кількістю прикладів, а клас з меншою кількістю прикладів може бути неправильно класифікований.

Наприклад, в задачі виявлення шахрайства в банківських транзакціях може бути тисячі легітимних транзакцій на одну шахрайську транзакцію. Це призводить до незбалансованості класів, де клас "шахрайство" має меншу кількість прикладів, ніж клас "легітимні транзакції".

У таких ситуаціях потрібно використовувати спеціальні методи для роботи з незбалансованими даними, такі як зменшення кількості прикладів у більшому класі, збільшення кількості прикладів у меншому класі, або використання ваг класів для підвищення важливості меншого класу.

Визначення того, що клас є незбалансованим, залежить від конкретної задачі та контексту. Зазвичай, коли один клас переважає над іншим, наприклад, коли кількість прикладів одного класу у наборі даних значно більша або менша, ніж кількість прикладів іншого класу, то можна говорити про незбалансованість класів.

Наприклад, якщо маємо набір даних, де 95% прикладів належать до класу A, а тільки 5% - до класу B, то можна вважати це прикладом незбалансованості класів. Однак, в іншій задачі, можливо, що таке розподілення класів не є проблемою. Тому, немає єдиної відповіді на запитання, на скільки відсотків один клас може переважати над іншим щоб вважати це незбалансованими класами.

Існує кілька підходів для роботи з незбалансованими класами:

1. Збільшення прикладів мінорних класів - цей підхід полягає в створенні додаткових зразків для меншості класів Це може бути зроблено шляхом аугментації даних.
2. Зменшення прикладів переважаючих класів - це може бути зроблено шляхом сабсемплінгу даних.
3. Використання ваг класів - цей підхід полягає в тому, щоб дати більші ваги мінорним класам в порівнянні з більшістю. Це може бути зроблено шляхом встановлення ваги для кожного класу.
4. Використання алгоритмів, які підтримують незбалансовані класи - деякі алгоритми, такі як дерева рішень, можуть підтримувати незбалансовані класи. Наприклад, деякі дерева рішень можуть мати параметр, що дозволяє встановлювати мінімальну кількість зразків у листі, який може допомогти уникнути перенавчання на більшості класів.
5. Використання метрик, які підходять для незбалансованих класів - для оцінки моделі важливо використовувати метрики, які приділяють увагу різниці між класами, такі як матриця помилок, F1-оцінка, ROC-крива..

Chart, bar chart, waterfall chart

Description automatically generated

*Рис.24. Зменшення та збільшення даних при незбалансованих цільових класах*

*Теоретична частина до практичної роботи №5*

**Розгортання моделі**

Розгортання - це процес використання ваших нових знань для вдосконалення вашої організації. Це може означати офіційну інтеграцію, таку як реалізація моделі. Крім того, розгортання може означати, що ви використовуєте інформацію, отриману від інтелектуального аналізу даних, щоб започаткувати зміни у вашій організації. Наприклад, можливо, ви виявили у своїх даних тривожні закономірності, які вказують на зміну поведінки клієнтів старше 30 років. Ці результати можуть не бути офіційно інтегрованими у ваші інформаційні системи, але вони, безперечно, будуть корисними для планування та прийняття маркетингових рішень.

Загалом, фаза розгортання CRISP-DM включає два види діяльності:

1. Планування та моніторинг розгортання результатів
2. Виконання підсумкових завдань, таких як підготовка остаточного звіту та огляд проекту

Залежно від вимог вашої організації вам може знадобитися виконати один або обидва ці кроки.

Робочий процес розгортання залежить від інфраструктури бізнесу та проблеми, яку ви прагнете вирішити. Модель прогнозування може бути ядром нової окремої програми або може бути включена до наявного програмного забезпечення.

Чому розгортання моделі важливо?

● Щоб почати використовувати модель машинного навчання, її потрібно ефективно впровадити у виробництво, щоб ML передбачення могли використовувати інші підсистеми.

● Щоб максимізувати цінність моделі машинного навчання, нам потрібно мати можливість надійно витягати прогнози та ділитися ними з іншими системами.

Машинне навчання як сервіс *(Machine learning as a service MLaaS)* - це автоматизована або напівавтоматизована хмарна платформа з інструментами для попередньої обробки даних, навчання моделі, тестування та розгортання, а також прогнозування. Перші три MLaaS - це Google Cloud AI, Amazon Machine Learning та Azure Machine Learning від Microsoft.

Розгортання на платформах MLaaS автоматизоване. Наприклад, результати прогнозів можна поєднати з внутрішніми або іншими хмарними корпоративними інфраструктурами за допомогою REST API.

Вам також слід подумати про те, як вам потрібно отримувати аналітичні результати: у режимі реального часу або через встановлені інтервали.

Для роботи ML моделі на виробництві потрібні різні компоненти:

● Інфраструктура

● Середовище (програми та компоненти)

● Дані

● Документація

● Конфігурація

Пакетне прогнозування (Batch prediction)

Цей варіант розгортання підходить, коли ваші прогнози не потрібні постійно. Вибираючи цей тип розгортання, ви отримуєте одне передбачення для групи спостережень. Модель навчається на статичному наборі даних і дає прогноз. Ви можете розгорнути модель на своєму сервері, на хмарному сервері, якщо вам потрібно більше обчислювальних потужностей, або використовувати для цього MlaaS. Розгортання не потрібно, якщо потрібен єдиний (разовий) прогноз. Наприклад, ви можете вирішити проблему класифікації, щоб дізнатися, чи певна група клієнтів приймає вашу пропозицію чи ні.

*Batch prediction потрібно буде реалізувати у практичній роботі.*

Веб -сервіс

Такий робочий процес машинного навчання дозволяє отримувати прогнози майже в режимі реального часу. Однак модель обробляє один запис із набору даних одночасно і робить для цього прогнози. Можливе розгортання моделі за допомогою платформ MLaaS, внутрішніх або хмарних серверів.

Прогнозування в режимі реального часу

Цей тип розгортання говорить сам за себе. За допомогою потокової (streaming) аналітики в режимі реального часу ви можете миттєво аналізувати потокові дані та швидко реагувати на події, які відбуваються в будь-який момент. Прогнозування в режимі реального часу дозволяє обробляти сенсорні або ринкові дані, дані з Інтернету речей або мобільних пристроїв, а також із мобільних або настільних програм та веб-сайтів. Оскільки цей метод розгортання вимагає обробки великих об’ємів вхідних даних, було б розумно використовувати Apache Spark або покладатися на платформи MlaaS. Apache Spark-це фреймворк для кластерних обчислень з відкритим вихідним кодом. Кластер - це набір комп’ютерів, об’єднаних у систему за допомогою програмного забезпечення та мереж. Завдяки високій продуктивності кластера, його можна використовувати для обробки великих даних, швидкого написання програм у Java, Scala або Python.

Веб-сервіс і прогнозування в реальному часі відрізняються кількістю даних для аналізу, які система отримує за раз.

Потокове навчання (Stream learning)

Потокове навчання передбачає використання динамічних моделей машинного навчання, здатних вдосконалюватись та оновлюватися в реальному часі. Ви можете розгорнути модель, здатну до самонавчання, якщо вам потрібно часто аналізувати дані. Apache Spark або MlaaS забезпечать вам високі обчислювальні можливості та дадуть змогу розгорнути модель самонавчання.

Інструменти: MlaaS (Google Cloud AI, Amazon Machine Learning, Azure Machine Learning), фреймворки ML (TensorFlow, Caffe, Torch, scikit-learn), фреймворки кластерних обчислень з відкритим кодом (Apache Spark).

Проекти ML іноді можуть бути заплутаними, тому важливо починати з малого і поступово збільшувати складність проекту, якщо це необхідно.

**Один з можливих варіантів структури ML проекту:**

* project\_name/ - ім’я проекту чи підпроекту
  + src/ - файли з аналізом даних, тестуванням різних моделей, результатами валідації
  + tests/ - юніт тести (не є обов’язковою)
  + models/ - для зберігання натренованих моделей
  + data/ - вхідний датасет
  + pipeline/ - скрипти тренування та передбачення
  + docs/ - документація проекту
  + [Readme.md](http://readme.md/)
  + …

**Пункти, які необхідно врахувати для скрипту тренування моделі.**

1. Скрипт для тренування моделі повинен містити наступні кроки:

* вичитку датасету;
* опрацювання даних (робота з пропущеними значеннями та аномаліями), кодування категоріальних ознак, масштабування ознак та інші перетворення даних; поділ датасету;
* тренування моделі;
* валідація результатів та виведення їх у файл чи лог;
* генерація важливості ознак та виведення їх в додатковий файл чи лог;
* збереження натренованої моделі.

1. Всі значення, що використовуються для опрацювання даних та генерації нових ознак мають бути збережені.
2. Навчену модель обов’язково потрібно зберегти для подальшого використання.
3. Частини коду, що стосуються вибору моделі додавати не потрібно, але перевірку обраної моделі все ж варто залишити.

**Пункти, які необхідно врахувати для скрипту побудови передбачення.**

1. Скрипт для передбачення містить такі частини:

* зчитування нових даних для передбачення;
* опрацювання даних (робота з пропущеними значеннями), кодування категоріальних ознак, масштабування ознак та інші перетворення даних;
* завантаження натренованої моделі;
* передбачення результату;
* форматування та збереження результату передбачення.

1. Нові вхідні дані мають проходити той же набір перетворень, що й дані для навчання. Тобто всі етапи з інженерії ознак мають бути застосовані в тому ж порядку і з тими ж параметрами. Окрім відкидання значень (якщо такий крок був застосований) - цей крок для нових вхідних значень пропускаємо. Якщо ви здійснювали заміну пропущених значень на середнє значення навчальної вибірки, то це значення необхідно зберегти і для нового вхідного набору даних використовувати теж середнє значення навчальної вибірки. Те саме стосується і кодування категоріальних ознак – вам необхідно зберегти словник кодування і використовувати один і той самий щоразу. У разі використання різних словників виникає ситуація, коли ми навчали модель на даних, в яких “яблуко” кодувалось цифрою 1, а в нових вхідних даних ми закодували “яблуко” цифрою 5 – ми отримаємо хибне передбачення.
2. Функції з етапу огляду даних до вихідного коду включати не потрібно.
3. Код (скрипт) має містити сценарій дій для порожнього набору вхідних значень та інших ситуацій.

Незалежно від обсягу проекту машинного навчання, його реалізація є трудомістким процесом, що складається з тих самих основних кроків з визначеним набором завдань. Розподіл ролей у командах з вивчення даних необов’язковий і може залежати від масштабу проекту, бюджету, часових рамок та конкретної проблеми. Наприклад, фахівці, які працюють у невеликих командах, зазвичай поєднують обов’язки кількох членів команди.

Навіть якщо ключова мета проекту - розробка та впровадження моделі прогнозування - досягнута, проект продовжується. Data scientists мають відстежувати, чи точність результатів прогнозування відповідає вимогам до продуктивності, і вдосконалювати модель, якщо це необхідно. Переконайтеся, що ви відстежуєте продуктивність розгорнутої моделі, якщо ви не використовуєте динамічну модель у виробництві. Один із способів перевірити, чи модель все ще працює на повну силу, - це зробити тест A/B. Показники ефективності, що використовуються для оцінки моделі, також можуть стати цінним джерелом зворотного зв'язку. Чим швидше дані застаріють у вашій галузі, тим частіше вам доведеться перевіряти продуктивність вашої моделі.

Завдання до практичних робіт

[\*Приклади функцій, що потрібно використати та інших допоміжних матеріалів для реалізації знаходяться у папці *example.*](https://drive.google.com/drive/folders/1aRUKMeWrsaUdhnMWvipwu0AqVhyAGguq?usp=sharing)

**Практична робота 1**

1. Відповідно до вашого варіанту опишіть можливі три бізнес-цілі, цілі моделювання та критерії успіху до них відповідно. Для цього завдання візьміть до уваги лише пункт «Ваш клієнт».
2. Ознайомившись із пунктом «Запит клієнта» додайте до попередніх цілей задану та сформулюйте цілі моделювання і критерії успіху.
3. Проведіть аналіз даних вашого підприємства (файл «train.csv») та сформуйте звіт про дослідження даних та звіт про якість даних.  
   Приклади функцій, що використовуються для аналізу даних ви можете знайти у файлі *example.ipynb* Для проведення аналізу необхідно встановити python3 та jupyter notebook (послідовність дій вказана у файлі *Встановлення python*)
4. Наведіть приклади цікавих, на вашу думку, фактів з аналізу даних.

**Практична робота 2**

1. Здійсніть підготовку даних вашого підприємства до подальшого моделювання використовуючи python/jupyter:

* Замініть відсутні значення, якщо такі присутні та обґрунтуйте вибір методу(ів)
* Проведіть кодування (Categorical Encoding) категоріальних ознак
* Опрацюйте аномальні значення
* Використайте підходи масштабування ознак якщо це доцільно для ваших даних

1. Сформуйте “Звіт про очищення даних”
2. Обґрунтуйте методи інженерії даних, які були використані для підготовки даних вашого підприємства.

**Практична робота 3**

1. Виберіть техніку моделювання для вирішення проблеми вашого підприємства.
2. Складіть звіт моделювання та опишіть припущення для обраних моделей.
3. Виберіть декілька метрик для ваших моделей з попередньої практичної та обґрунтуйте свій вибір.
4. Виберіть метод поділу даних на навчальні та тестові. Опишіть основи такого рішення.
5. Здійсніть тренування обраних моделей на даних вашого підприємства.
6. Здійсніть валідацію моделей на основі обраних метрик та поділу даних.
7. Оберіть найкращу модель.
8. Виведіть та про аналізуйте важливість ознак використовуючи метаінформацію однієї з натренованих моделей.
9. Поясніть вибір техніки моделювання та аналіз важливості ознак. Поясніть результати тренування моделей відповідно до ваших метрик.

**Практична робота 4**

1. Виберіть декілька метрик для ваших моделей для оптимізації гіперпараметрів.
2. Виберіть метод оптимізації гіперпараметрів, гіперпараметри та їх можливі значення. Опишіть основи такого рішення.
3. Здійсніть оптимізацію гіперпараметрів моделі на основі обраних метрик та гіперпараметрів.
4. Оберіть оптимальний результат.
5. Поясніть результати пошуку гіперпараметрів моделі відповідно до ваших метрик.
6. Застосуйте підходи роботи з незбалансованими класами якщо є така необхідність. Обгрунтуйте свій вибір.

**Практична робота 5**

1. Створіть структуру вашого проекту. Додайте ваші файли/файл з практичних робіт 1-4 у папку src.
2. Розділіть дані з вашого файлу «train.csv» на train та new\_input у співвідношенні 90:10
3. Створіть python скрипт для тренування моделі та запустіть його на train даних
4. Створіть файл для побудови передбачень та запустіть його на new\_input даних.
5. Створіть Readme.md та інші файли, яких потребує структура проекту.
6. Опишіть структури проекту.

Наведіть розгорнутий висновок щодо етапів побудови ML проекту, закономірностей в даних для тренування, аналізу важливості ознак, порівняння результатів різних моделей. Які з етапів вам видались складними та чому? Що сподобалось? Ваші загальні враження? Чому навчилися виконуючи практичні роботи.

[Варіанти завдань *(за загальним списком групи)*](https://drive.google.com/drive/folders/1QpE9Cx_bFj3MxZV2rJACwKyXfxUujB4Q?usp=sharing)

**1 варіант** https://www.kaggle.com/datasets/yasserh/loan-default-dataset/data

Опис Компанії-Замовника: JPMorgan Chase & Co. - один з найбільших і найвідоміших банків у світі, який активно використовує аналітичні інструменти для оцінки кредитного ризику.

Запит: Керівництво банку шукає спосіб оптимізації процесу оцінки кредитоспроможності клієнтів для зниження ризику неповернення кредитів.

Пропозиція Реалізації: Розробити передову аналітичну модель на основі машинного навчання, яка використовуватиме історичні дані кредитування для прогнозування ймовірності неповернення позики (default). Ця модель допоможе установі більш точно оцінювати ризики та приймати обґрунтовані рішення щодо надання кредитів, тим самим знижуючи відсоток неплатоспроможних кредитів.

**2 варіант** https://www.kaggle.com/datasets/parulpandey/palmer-archipelago-antarctica-penguin-data/data

Опис Компанії-Замовника: Scott Polar Research Institute (SPRI) - Британський інститут, що є частиною Кембриджського університету, який займається всесторонніми дослідженнями полярних регіонів.

Запит: Організація потребує розробки алгоритму, здатного класифікувати різні види пінгвінів на основі зібраних даних, щоб поліпшити розуміння різноманіття та розподілу цих видів у регіоні.

Пропозиція Реалізації: Створення моделі машинного навчання для класифікації видів пінгвінів. Модель повинна враховувати різні фізичні характеристики, такі як розміри тіла, маса та гендерні особливості, для точного визначення виду кожного пінгвіна. Ця класифікація допоможе визначити основні видові групи пінгвінів у регіоні та їх розподіл, сприяючи кращому збереженню цих видів та розумінню їхнього екологічного статусу.

**3 варіант** https://www.kaggle.com/competitions/petfinder-adoption-prediction/data

Опис Компанії-Замовника: Best Friends Animal Society: Організація в США, відома своїми зусиллями у просуванні усиновлення домашніх тварин і зниженні кількості тварин, що евтаназуються в притулках.

Запит: Організація шукає розв'язок для прогнозування ймовірності усиновлення тварин, що знаходяться в притулку, з метою підвищення їхніх шансів на знаходження нового дому.

Пропозиція Реалізації: Розробка моделі машинного навчання, для передбачення ймовірності усиновлення тварин з притулку. Модель повинна аналізувати різноманітні характеристики тварин, такі як вік, порода, здоров'я, поведінка, та інші значимі фактори, що можуть впливати на шанси тварини бути усиновленою. Цей аналіз допоможе притулку краще розуміти потреби тварин та ефективніше підходити до процесу усиновлення, збільшуючи шанси кожної тварини знайти новий дім.

**4 варіант** https://archive.ics.uci.edu/dataset/697/predict+students+dropout+and+academic+success

Опис Компанії-Замовника: Стенфордський Університет (Stanford University): Знаний своїми інноваціями в галузі освіти та дослідженнями, які зосереджені на підвищенні ефективності навчального процесу та підтримці студентів.

Запит: Установа шукає способи аналізу академічної успішності та прогнозування ризику відрахування студентів, щоб ефективніше керувати навчальним процесом і надавати додаткову підтримку студентам, які цього потребують.

Пропозиція Реалізації: Створення аналітичної моделі машинного навчання для ідентифікації ключових індикаторів, що впливають на академічний успіх та ризик відрахування студентів. Модель аналізуватиме різні аспекти студентського життя і навчання, включаючи академічні досягнення, соціальні та економічні фактори, а також поведінкові патерни, для прогнозування потенційних ризиків та визначення студентів, що можуть потребувати додаткової підтримки. Це дозволить освітній установі вживати своєчасних заходів для запобігання відрахувань і поліпшення загальної академічної успішності.

**5 варіант** https://archive.ics.uci.edu/dataset/848/secondary+mushroom+dataset

Опис Компанії-Замовника: Royal Botanic Gardens, Kew: Відомий ботанічний сад у Великобританії, який має велику колекцію грибів і активно займається їх дослідженням.

Запит: Організація шукає методи для точної класифікації та аналізу грибів, зокрема щодо їхньої їстівності, отруйності, а також інших біологічних характеристик, на основі зібраних даних.

Пропозиція Реалізації: Розробити модель машинного навчання для класифікації видів грибів. Модель повинна враховувати різні морфологічні характеристики грибів, такі як форма, колір, розмір, а також інші важливі біологічні ознаки, для визначення їх їстівності, отруйності та інших властивостей. Це дозволить дослідникам краще зрозуміти різноманітність грибів та їхні характеристики, а також сприятиме безпеці використання грибів у харчуванні та медицині.

**6 варіант** https://www.kaggle.com/datasets/adilashrafi/bank-marketing-classification-task

Опис Компанії-Замовника: HSBC Holdings plc: Міжнародний банківський конгломерат, який активно використовує цифровий маркетинг та персоналізовані кампанії для просування своїх продуктів.

Запит: Банк шукає способи для підвищення результативності своїх маркетингових кампаній, зокрема в контексті залучення клієнтів до нових банківських продуктів.

Пропозиція Реалізації: Розробка класифікаційної моделі машинного навчання для прогнозування реакції потенційних клієнтів на різні маркетингові ініціативи. Вона повинна оцінювати ефективність різних підходів, враховуючи демографічні дані клієнтів, їхню історію взаємодій з банком, а також відгуки на попередні рекламні акції. Це допоможе банку більш цілеспрямовано підходити до своїх клієнтів, збільшуючи ефективність маркетингових заходів та залучення нових клієнтів.