Київський національний університет імені Тараса Шевченка
Факультет радіофізики, електроніки та комп'ютерних систем
Кафедра комп'ютерної інженерії

ПРОГНОЗУВАННЯ КІЛЬКОСТІ ВІДВІДУВАЧІВ ТОРГІВЕЛЬНОГО ЦЕНТРУ З ВИКОРИСТАННЯМ РЕГРЕСІЙНИХ МОДЕЛЕЙ

Дипломна робота магістра
студента 2 року навчання
спеціальність: 123 «Комп'ютерна інженерія»
Олексія МОРМУЛЯ

Науковий керівник
кандидат ф-м. наук Олександр БАРАБАНОВ,
доцент кафедри комп'ютерної інженерії

Рецензент
кандидат ф-м. наук Анатолій ШКАВРО
доцент кафедри нанофізики конденсованих
середовищ інституту високих технологій

До захисту допускаю: Завідувач кафедрою

Юрій БОЙКО

Ухвалено на засіданні кафедри "____" _____ 2022 р., протокол № ___

Київ - 2022

Реферат

Враховуючи можливості використання та аналіз даних статистики відвідування торгівельного центру, існує необхідність використання прогнозування для отримання кількості відвідувачів на майбутні дні. З необхідністю статистики для використання електроенергії, запасу складу магазинів на певний майбутній період необхідно спрогнозувати кількість відвідувачів в даний відрізок часу.

Ці дані можуть бути представлені як багатовимірний часовий ряд і використовуватись не тільки для моделювання, але і для отримання майбутнього прогнозу в різних галузях де використовуються дані минулих років, також пора року, погоду, кількість відвідувачі, температура навколишнього середовища.

Випускна кваліфікаційна робота магістра містить: 48 ст., 5 рис., 16 джерел, 1 додаток.

Ключові слова: Python, ARIMA, Polynomial Regression, Linear Regression, часові ряди, прогнозування.

ЗМІСТ

Скорочення та умовні позначення

ARIMA (Auto Regressive Integrated Moving Average) – Інтегрована модель авторегресії змінного середнього.

Трейдинг – в перекладі з англійської обмін одних цінностей на інші, простіше кажучи – торгівля. Але не будь-яка.

PR (Polynomial Regression) – поліноміальна регресія.

MAPE – середня абсолютна відсоткова помилка.

RMSE – середньоквадратична помилка.

Спектральний аналіз – є фізичним методом якісного та кількісного визначення складу речовин, він оснований отриманням і дослідженням спектрів електромагнітного випромінювання.

Кореляційний аналіз – “correlation”, відношення чи співвідношення відповідності речей або понять.

SES (simple exponential smoothing) – просте експоненційне згладжування, метод математичного перетворення, що застосовується для прогнозування часових рядів.

Вступ

Актуальність роботи:

Протягом останніх років теорія часових рядів швидко розвивалася. Виникають нові необхідності використання моделей для практичного прогнозування і використання яких на сьогодні набуває все більшу актуальність. У зв’язку з інтенсивним поширенням застосування прогнозування для отримання передбачення майбутнього з’явилася потреба у здійснення прогнозування на основі сучасних статистичних або математичних методів, що є складовими ефективного прийняття рішень, зокрема при аналізі фінансових часових рядів для отримання прогнозування даних використання електроенергії, а також закупівлі продукції з використанням кількості відвідувачів торгівельного центру за певний період.

1 Аналітичний огляд літератури

1.1 Часові ряди

Часовий ряд – це послідовність значень досліджуваної ознаки, впорядкована у хронологічному порядку, або сукупності значення випадкових процесів, взятих на рівних проміжках часу (*t*) [1]. Точність будь-якої інформації системи прогнозування надається ефективність управління процесом прогнозування. Під процесом прогнозування вважається аналіз і оцінювання, на основі визначених наукових підходів, тенденцій розвитку певних процесів або явищ, використовуючи наявну інформацію про перебіг цього процесу або явища в минулому, тобто ретроспективні дані [2].

Часові ряди активно досліджуються з різними цілями. В одному випадку вони використовуються для передбачення майбутнього (наприклад кількість відвідувачів або трейдингу [3]), в іншому випадку буде достатньо отримати опис характерного особливостей ряду. Найпоширенішими методами аналізу часових рядів є: кореляційний аналіз, спектральний аналіз, моделі авторегресії, багатоканальні моделі авторегресії і змінного середнього, сезонна модель Бокса-Дженкінса, прогноз експоненціально-зваженим змінним середнім.

Одними з методів дослідження в сучасному аналізі часових рядів є лінійна та поліноміальна регресії, метод ARIMA, а також Холта-Вінтера.

1.2 Прогнозування методом часових рядів

Прогнозування методом часових рядів – є одним із популярніших підходів до прогнозування та розвитку економічних процесів, наприклад замовлення споживання електроенергії, об’ємів торгівельних операцій, об’ємів виробництва а також накопичення або замовлення продукції на складах, формування бюджетів підприємств, торгівельних центрів, магазинів та держави, прогнозування а також менеджмент економічних або фінансових ризиків та інше [4]. Загалом методи прогнозування можуть мати в собі три класи:

1. Прогнозування в основі суджень – прогнозування, яке ґрунтується з суб’єктивних судженнях (оцінках), поглиблених знаннях певної області а також інакшій інформації, яка має відношення до процесу прогнозування – або передбачення;
2. Методи прогнозування на основі використання часового ряду однією змінної, тобто, на основі авторегресії, авторегресії з змінним середнім (АРКС) та авторегресії з середнім змінним а також модель тренду;
3. Метод прогнозування в основі для використання кількох змінних часових рядів.

В третьому випадку ендогенна змінна, яка робить прогнозування, має залежність від декількох регресорів або екзогенних змінних у елементу рівняння. Загалом прогнозування може також поєднувати в собі два або три з наведених вище методів.

Прогнозування часового ряду можна в цілому розділити на два типи [5]:

1. Одновимірним прогнозуванням часових рядів, якщо використовувати лише попередні значення часового ряду, щоб передбачити його майбутні значення;
2. Багатоваріантним прогнозуванням часових рядів, якщо використовувати для прогнозування інші прогнози, аніж ряди (наприклад екзогенні змінні).

Прогнозування часових рядів означає прогнозування або передбачення майбутньої вартості за певний період часу. Це тягне за собою розробку моделей на основі попередніх даних і їх застосування для спостережень і спрямування майбутніх стратегічних рішень.

Майбутнє прогнозується або оцінюється на основі того, що вже сталося. Часовий ряд додає залежність від часового порядку між спостереженнями. Ця залежність є одночасно і обмеженням, і структурою, яка забезпечує джерело додаткової інформації.

Прогнозування часових рядів — це техніка для передбачення подій через послідовність часу. Він передбачає майбутні події, аналізуючи тенденції минулого, виходячи з припущення, що майбутні тенденції будуть подібними до історичних тенденцій.

Після того, як буде визначено, які прогнози потрібні, необхідно поставити задачу побудови ймовірнісної, математичної або логічної моделі, яка має мету покрити високу якість прогнозування на заданому горизонті та знайти або зібрати дані, на яких будуть базуватися прогнози. Дані, необхідні для прогнозування, можуть уже існувати.

Звісно, існують обмеження в роботі з непередбачуваним і невідомим. Прогнозування часових рядів не є безпомилковим і не підходить для всіх ситуацій. Оскільки, насправді, не існує чітких правил щодо того, коли можливо або неможливо використовувати прогнозування, тому потрібно знати обмеження аналізу та те, що можуть підтримувати моделі. Не кожна модель підходить для кожного набору даних або відповість на кожне запитання.

Хороше прогнозування працює з чистими даними з позначкою часу і може визначити справжні тенденції та закономірності в історичних даних. Аналітики можуть визначити різницю між випадковими коливаннями або викидами, і можуть відокремити справжню інформацію від сезонних коливань. Аналіз

часових рядів показує, як дані змінюються з часом, і якісне прогнозування може визначити напрямок, у якому змінюються дані.

1.3 Лінійна регресія

За визначенням [6]: Лінійна регресія володіє важливими інструментами аналітики, що включає в собі методики статичного обчислення для побудови лінії тренду в наборі псевдо випадкових точок даних. Лінія “тренду”, в якій присутні будь які дані: від прогнозування кількості товарів для закупівлі або споживання електроенергії до кількості людей, які відвідують торгівельний центр за певний період часу.

Іноді залежну змінну також називають ендогенною змінною, прогностичною змінною або регресатом. Незалежні змінні також називають екзогенними змінними, провісниками або регресатами. Однак лінійний регресійний аналіз складається не тільки з підгонки лінійної лінії через хмару точок даних. Він складається з 3 етапів:

- Аналіз кореляції та спрямованості даних;
- Оцінка моделі, тобто підгонка лінії;
- Оцінка валідності та корисності моделі.

Існує три основні види використання регресійного аналізу:

- Причинно-наслідковий аналіз;
- Прогноз впливу;
- Прогнозування тенденцій.

Крім кореляційного аналізу, який зосереджується на силі зв'язку між двома або більше змінними, регресійний аналіз передбачає залежність або причинно-наслідковий зв'язок між однією або кількома незалежними та однією залежною змінною.

Однобічна або проста лінійна регресія [7] - це найпростіший випадок лінійної регресії з однією незалежною змінною, $\boldsymbol{x} = x$. Застосовуючи просту лінійну регресію, як правило, починають з заданого набору пар вводу-виводу (x-y). Однобічна залежність виражається з допомогою функції, яка називається

функцією регресії математичної залежності. Регресія з двома змінними має рівняння:

$$f(x) = y \tag{1.3.1}$$

В лінійній регресії одна незалежна змінна *x*, і одна залежна змінна *y*, і для заданого значення можливо оцінити середнє значення результату і записати його як умовне очікування. Регресія має в собі специфічний вид асоціації і може бути лінійним або нелінійним.

Така задача простої регресії є найбільш поширеною та вивченою. Вивчення властивості параметрів, що отримуються різними методами ймовірності характеристики факторів та випадкових похибок моделі.

Одним з найпопулярніших методів оцінки параметрів лінійної регресії є метод найменших квадратів [8]. Робота, яка вважається першою роботою, що використовувала метод найменших квадратів належить Лежандру. В 1805 р. стаття “Нові методи визначення орбіт комет” було написано “Після того як повністю використані умови задачі, необхідно визначити коефіцієнти так, щоб величини їх помилок були найменшім із можливих. Для цього нами вказаний простий спосіб, який полягає в знаходженні мінімуму суми квадратів помилок”.

В 1809 р. Гаусс у відомій своїй роботі на обчислення орбіт дав інше обґрунтування закону розподілення помилок. Окрім того, Гаусс наполягав використання методу найменших квадратів з 1795 р.

Метод найменших квадратів, в майбутньому, зв’язане з іншим іменем, таким як з іменем Лапласа, який в 1812 році в роботі “Аналітична теорія ймовірностей” продемонстрував, що метод дозволяє знайти незміщені оцінки незважаючи на тип вхідного розподілу даних.

Гаусс в своїх роботах публікував міркування, що пов’язані з даним методом в 1821 році. В роботах не було зв’язаних понять, таких як дисперсія,

але попри все, не прибігаючи до звичайної матричної алгебри, мав змогу довести, що серед класу оцінок, які надходять:

- Незміщеними, за рахунок оцінками параметрів.
- Комбінаціями вхідних даних з методом лінійності.

Важлива характеристика оцінок, що були отримані методом найменших квадратів, є у незалежності від типу розподілу.

Загальний варіант теореми був доведений в 1912р. А. Марковим і в даний час відома як теорема Гауса-Маркова [9]. Теорема є центральним варіантом в методі найменших квадратів.

Проста лінійна регресія є параметричним тестом , що означає, що він робить певні припущення щодо даних. Ці припущення:

- Однорідність дисперсії: розмір помилки в нашому прогнозі суттєво не змінюється у значеннях незалежної змінної.
- Незалежність спостережень : спостереження в наборі даних були зібрані за допомогою статистично дійсних методів вибірки, і між спостереженнями немає прихованих зв'язків.
- Нормальність : дані відповідають нормальному розподілу .
- Лінійна регресія робить одне додаткове припущення:
- Зв'язок між незалежною та залежною змінною є лінійною : лінія, яка найкраще підходить через точки даних, є прямою лінією (а не кривою чи якимось фактором групування).

Великий вклад в розвиток даного методу був здійснений в 1934р. Ейткеном, який узагальнив теорему на випадок картельованих результатів спостережень з різними методами дисперсії. Робота М. Мерримана [10] має в собі історичний розвиток методу найменших квадратів та дає критичні зауваження, що можливо використовувати в роботі.

1.4 Множинна лінійна регресія

Регресійні моделі використовуються для опису зав’язків між змінними шляхом підбору лінії до спостережуваних даних. Регресія дозволяє оцінити, як змінюється залежна змінна в міру зміни незалежної змінної.

Множинна лінійна регресія використовується для оцінки зв’язку між двома або більше незалежними змінними та однією залежною змінною. Використання множинної лінійної регресії дає змогу знайти:

- Наскільки сильний зв'язок між двома або більше незалежними змінними та однією залежною змінною (наприклад, як кількість відвідувачів, температура погода впливають на споживання електроенергії в торгівельному центрі).
- Значення залежної змінної при певному значенні незалежних змінних (наприклад, очікуваний результат кількості відвідувачів).

У множинної лінійної регресії припущення ті ж самі, що і у лінійної регресії:

- Однорідність дисперсії – розмір помилки в нашому прогнозі суттєво не змінюється у значеннях незалежної змінної.
- Незалежність спостережень – спостереження в наборі даних були зібрані за допомогою статистично дійсних методів, і немає прихованих зав’язків між змінними.

Для множинній лінійній регресії можливо, що деякі незалежні змінні фактично корелюють одна з одною, тому важливо перевірити їх перед розробкою моделі регресії. Якщо дві незалежні змінні дуже сильно корелюють, то в регресійній моделі слід використовувати лише одну з них.

Результат множинної лінійної регресії це лінія, яка найкраще підходить через точки даних, і є прямою лінією, а не кривою чи якимось фактором групування.

Формула множинної лінійної регресії виглядає:

$$y = \beta_0 + \beta_1 X_1 + \cdots + \beta_n X_n + \varepsilon \quad (1.4.1)$$

Де y – прогнозоване значення залежної змінної, β_0 переріз у (значення y, коли всі інші параметри встановлені на 0), $\beta_1 X_1$ коефіцієнт регресії (β_1) першої незалежної змінної (X_1) (він же вплив збільшення значення незалежної змінної на передбачене значення y, $\beta_n X_n$ коефіцієнт регресії останньої незалежної змінної, ε помилка моделі (а також скільки варіацій є в оцінці y.

Для знаходження лінії, яка найкраще підходить для кожної незалежної змінної, множинна лінійна регресія обчислює три речі:

- Коефіцієнти регресії, які призводять до найменшої загальної помилки моделі.
- t – статистика загальної моделі.
- Пов’язане значення p (наскільки ймовірно, що t – статистика виникла б випадково, якби нульова гіпотеза про відсутність зв’язку між незалежною та залежною змінними була вірною).

1.5 Поліноміальна регресія

Поліноміальна регресія – це форма лінійної регресії, де лише через нелінійний зв'язок між залежними та незалежними змінними ми додаємо деякі поліноміальні терміни до лінійної регресії, щоб перетворити її в поліноміальну регресію. Поліноміальна регресія чутлива до викидів (різких змін), тому наявність одного або двох викидів також може погану вплинути на продуктивність. Отже, поліноміальна регресія розглядається як феноменальний випадок лінійної регресії.

Припустимо, що ми маємо X як незалежні дані і У як залежні дані. Перед подачею даних у режим на етапі попередньої обробки ми перетворюємо вхідні змінні в поліноміальні терміни, використовуючи певний ступінь. Припускаємо поліноміальну залежність між вихідними та вхідними даними і, отже, оцінену функцію регресії поліномів.

Іншими словами, крім лінійних доданків, таких як $b_1\ x_1$, функція регресії f може включати нелінійні доданки, такі як $b_2\ x_1\ ^2$, $b_3\ x_1\ ^3$ або навіть $b_4\ x_1\ x_2$, $b_5\ x_1\ ^2x_2$ тощо.

Найпростіший приклад поліноміальної регресії має єдину незалежну змінну, а оцінна функція регресії є поліномом ступеня 2:

$$f\ (x) = b_0\ + b_1\ x + b_2\ x^2, \tag{1.5.1}$$

Необхідність поліноміальної регресії полягає:

• Якщо ми застосовуємо лінійну модель до лінійного набору даних, то це дає нам хороший результат, в порівнянні з простою лінійною регресією, але якщо ми застосовуємо ту ж модель без будь-яких змін до нелінійного набору даних , то це призведе до появи виходу за рамки. Завдяки цьому функція втрат збільшиться, відсоток помилок буде високим, а точність знизиться.

• Тому для таких випадків, коли точки даних розташовані не лінійно, ми використовуємо модель поліноміальної регресії.

Припущення поліноміальної регресії:

- Поведінку залежної змінної можна пояснити лінійним, або криволінійним, адитивним зв’язком між залежною змінною та набором з *k* незалежних змінних (*xi, i=1* до *k*).
- Зв'язок між залежною змінною і будь-якою незалежною змінною є лінійною або криволінійною (зокрема поліноміальною).
- Незалежні змінні незалежні одна від одної.
- Похибки незалежні, нормально розподілені з нульовим середнім значенням і постійною дисперсією (*OLS*).

Недоліки поліноміальної регресії:

Однак, як і лінійна регресія, поліноміальна регресія не є універсальним інструментом.

- Навіть один випадок у графіку даних може серйозно зіпсувати результати.
- PR-моделі схильні до переобладнання. Якщо використовується достатня кількість параметрів, можна підібрати що завгодно.
- Як наслідок попереднього, моделі PR можуть погано узагальнюватися за межами використовуваних даних.

Модифікований метод багатовимірної поліноміальної регресії полягає в тому, що має фіксацію невеликої кількості певних змінних, а також одночасні зміни інших параметрів, які не є фіксованими. Цей спосіб приводить до одновимірних регресій з вищими ступенями та має більшу точність оцінок невідомих коефіцієнтів регресії.

Поліноміальна регресія є простим, але потужним інструментом для прогнозної аналітики. Це дозволяє розглядати нелінійні зв’язки між змінними і робити висновки, які можна оцінити з високою точністю.

1.6 Модель ARIMA

Модель ARIMA (AutoregRessive Integrated Moving Average) [13] – один із найпоширеніших методів аналізу та прогнозування часових рядів. Ця модель дозволяє обробити дані тимчасового ряду, щоб краще зрозуміти цей ряд або передбачити його розвиток.

Авторегресивні моделі концептуально подібні до лінійної регресії, це припущення має місце і для моделі ARIMA. Дані часових рядів необхідно зробити стаціонарними, щоб усунути будь-яку очевидну кореляцію та колінеарність з минулими даними. У стаціонарних даних часових рядів властивості або значення вибіркового спостереження не залежать від відмітки часу, за яким воно спостерігається. Наприклад, якщо помітити, що населення збільшується вдвічі щороку або збільшується на фіксовану величину, то ці дані є нестаціонарними. Будь-яке дане спостереження сильно залежить від року, оскільки значення населення залежатиме від того, наскільки воно віддалено від довільного минулого року. Ця залежність може викликати неправильне зміщення під час навчання моделі з даними часових рядів.

Щоб усунути цю кореляцію, ARIMA використовує розбіжність (differencing), щоб зробити дані стаціонарними. Розрізнення, у найпростішому випадку, передбачає отримання різниці двох сусідніх точок даних.

ARIMA використовує три основні параметри *(p, d, q)*, які виражаються цілими числами. Тому модель також записується як ARIMA *(p, d, q)*. Водночас ці три параметри враховують сезонність, тенденцію та шум у наборах даних:

- *p* – порядок авторегресії *(AR)*, що дозволяє додати попередні значення часового ряду та означає кількість відставання *Y*, яке буде використано як орієнтири (наприклад: завтра, ймовірно, буде багато людей, якщо останні дні було людно).

• d - порядок інтегрування (I; тобто порядок різниць вихідного часового ряду). Він додає в модель поняття різниці часових рядів (визначає кількість минулих часових точок, які потрібно відняти з поточного значення).

• q – порядок змінного середнього (MA), який дозволяє встановити похибку моделі як лінійну комбінацію значень помилок, що спостерігалися раніше.

Для відстеження сезонності використовується сезонна модель ARIMA – ARIMA*(p, d, q)(P,D,Q)s* [14]. Тут *(p, d, q)* – несезонні параметри, описані вище, а (P, D, Q) слідують тим самим визначенням, але застосовуються до сезонної складової часового ряду. Параметр s визначає періодичність часового ряду (4 – квартальні періоди, 12 – річні періоди тощо).

Сезонна модель ARIMA може здатися складною через численні параметри. У наступному розділі ви дізнаєтесь, як автоматизувати процес визначення оптимального набору параметрів для сезонної моделі часових рядів ARIMA.

Інтервали прогнозування для моделей ARIMA базуються на припущеннях, що залишки некорельовані та нормально розподілені. Якщо будь-яке з цих припущень не виконується, то інтервали передбачення можуть бути неправильними. З цієї причини завжди будуйте графік ACF та гістограму залишків, щоб перевірити припущення, перш ніж створювати інтервали передбачення.

Загалом, інтервали прогнозування з моделей ARIMA збільшуються зі збільшенням горизонту прогнозу. Для стаціонарних моделей (тобто $cd = 0$) вони зближаться, так що інтервали прогнозування для довгих горизонтів по суті однакові. Для $d \geq 1$, інтервали прогнозування будуть продовжувати зростати в майбутньому.

Як і в більшості розрахунків інтервалів передбачення, інтервали на основі ARIMA, як правило, занадто вузькі. Це відбувається тому, що враховано лише

різницю в помилках. Існують також зміни в оцінках параметрів і порядку моделі, які не були включені в розрахунок. Крім того, розрахунок передбачає, що історичні закономірності, які були змодельовані, зберігатимуться протягом прогнозованого періоду.

Модель ARIMA [15]— це модель, де ряд часу віднімається принаймні один раз, щоб зробити його нерухомим, і ми поєднуємо терміни авторегресивного (AR) і змінним середнього (MA). Отже, ми отримали таке рівняння:

$$Y_t = a + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots + \beta_p y_{t-p} \epsilon_t + \emptyset_1 \epsilon_{t-1} + \emptyset_2 \epsilon_{t-2} + \cdots + \emptyset_q \epsilon_{t-q} \quad (1.6.1)$$

Тут майбутнє значення y(t) обчислюється на основі помилок ϵ_t, допущених попередньою моделлю. Таким чином, кожен наступний термін дивиться на крок далі в минуле, щоб включити помилки, допущені цією моделлю, у поточні обчислення. На основі вікна, яке ми готові зазирнути, встановлюється значення q. Таким чином, вищенаведену модель можна незалежно позначити як змінне середнє порядок q або просто MA(q).

Модель ARIMA словами:

Прогнозований Y_t= константа + затримки лінійної комбінації Y (до p lags) + лінійна комбінація передбачуваних помилок із затримкою (до q затримок). Таким чином, метою цієї моделі є знаходження значень p, q і d .

Використання моделі ARIMA має переваги при умовах:

Коли минулі значення у наших даних впливають на поточні чи майбутні значення або можуть передбачити майбутні тенденції на основі останніх коливань. У цьому випадку прогнозування часових рядів є рішенням такої проблеми регресії. Кілька інших моделей прогнозування часових рядів покладаються на включення послідовних змін або останніх змін у дані для прогнозування майбутніх тенденцій. З іншого боку, деякі інші моделі використовують чисто статистичні величини, які часто включають тенденції з

історичних даних, які можуть бути не такими релевантними в теперішніх або майбутніх значеннях. Ці припущення та підходи мають вагоме обґрунтування, але часто бувають невдалими в реальному житті.

ARIMA включає ці ідеї у свій комбінований підхід авторегресії та змінного середнього до моделювання стаціонарних даних часових рядів. Цей підхід визначає важливість минулих коливань, включає загальні тенденції та має справу зі згладжуванням ефекту викидів або тимчасових аномальних змін у даних. Таким чином, ARIMA підходить для врахування історичних тенденцій, сезонності, випадковості та іншої нестатичної поведінки.

1.7 Модель Холта-Вінтера

Реальні дані, такі як дані про попит у будь-якій галузі, зазвичай мають багато сезонності та тенденцій. При прогнозуванні попиту в таких випадках потрібні моделі, які враховують тенденцію та сезонність у даних, оскільки рішення, прийняте бізнесом, буде ґрунтуватися на результатах цієї моделі. Для таких випадків метод Холта-Вінтера є одним із багатьох методів прогнозування часових рядів, які можна використовувати для прогнозування.

Експоненціальне згладжування Холта-Вінтера [16], назване на честь двох його учасників: Чарльза Холта та Пітера Вінтера, є одним із найстаріших методів аналізу часових рядів, який враховує тенденцію та сезонність під час прогнозування. Цей метод має 3 основні аспекти виконання передбачень. Він має середнє значення з тенденцією та сезонністю. Три аспекти – це 3 типи експоненціального згладжування, і, отже, метод утримання Вінтера також відомий як потрійне експоненціальне згладжування.

Три основні аспекти передбачення:

- Експоненціальне згладжування : просте експоненціальне згладжування, як випливає з назви, використовується для прогнозування, коли набір даних не має тенденцій або сезонності.
- Метод згладжування Холта: метод згладжування Холта, також відомий як лінійне експоненціальне згладжування, також широко відомою моделлю згладжування для прогнозування даних, які мають тенденцію.
- Метод Вінтер-згладжування: Вінтер метод згладжування дозволяє нам включати сезонність, роблячи прогноз разом із тенденцією.

Просте експоненціальне згладжування (SES) підходить для даних часових рядів без трендів або сезонних компонентів.

Ця модель обчислює прогнозні дані за допомогою середньозважених величин. Одним із важливих параметрів, які використовуються в цій моделі, є

параметр згладжування: α, і ви можете вибрати значення від 0 до 1, щоб визначити рівень згладжування. Коли $\alpha = 0$, прогнози дорівнюють середнім історичним даним. При $\alpha = 1$ прогнози будуть дорівнювати значенню останнього спостереження.

Ви можете вибрати конкретне α (наприклад, у прикладі коду я використовував 0.8) або використати модуль Python 'statsmodels', щоб автоматично знайти оптимізоване значення для набору даних. Зазвичай я використовую підхід автоматичної оптимізації, який дає нам найменшу помилку, але якщо ви хочете бути більш консервативним або агресивним, ви можете вказати α .

Візуалізація результатів для моделі прогнозу простого експоненційного згладжування (SES) показує різницю між вказаним α та автоматично оптимізованим α. Оскільки більшість даних часових рядів мають певну тенденцію або сезонність, цю модель можна використовувати, щоб отримати відчуття базової лінії для порівняння.

Метод лінійного тренду Холта підходить для даних часових рядів із компонентом тренду, але без сезонного компонента.

Розширюючи метод SES , метод Холта допомагає прогнозувати дані часових рядів, які мають тенденцію. На додаток до параметра згладжування рівня α , введеного в методі SES, метод Холта додає параметр згладжування тренду β^* . Як і для параметра α , діапазон β^* також знаходиться між 0 і 1. У порівнянні з SES, Холт фіксує більше тенденцій даних.

Сезонний метод Холта-Вінтера підходить для даних часових рядів із тенденційними та/або сезонними компонентами.

Модель Холта-Вінтера розширює Холт, щоб дозволити прогнозувати дані часових рядів, які мають як тенденцію, так і сезонність, і цей метод включає цей параметр згладжування сезонності: γ .

Існує два загальні типи сезонності: адитивна та мультиплікативна.

• Адитивна: *xt = Trend + Seasonal + Random*. Сезонні зміни в даних залишаються приблизно однаковими з часом і не коливаються по відношенню до загальних даних.

• Мультиплікативна: *xt = Trend * Seasonal * Random*. Сезонні зміни змінюються по відношенню до загальних змін у даних. Отже, якщо дані мають тенденцію до зростання, сезонні відмінності також зростають пропорційно.

Візуалізація результатів для методу Холта-Вінтера показує адитивну порівняно з тенденціями адитивних + погашених тенденціях. Однак RMSE не краще, ніж результати простої моделі SES. Також можна сказати, що прогноз починає знижуватися до кінця.

Отже, метод Холта-Вінтера враховує середнє значення разом із тенденцією та сезонністю під час прогнозування часових рядів.

Незважаючи на найкращий результат прогнозу, метод Холта-Вінтера все ж має певні недоліки. Одним з основних обмежень цього алгоритму є мультиплікативна ознака сезонності. Проблема мультиплікативної сезонності полягає в тому, як працює модель, коли ми маємо часові рамки з дуже малими сумами. Часовий проміжок з точкою даних 10 або 1 може мати фактичну різницю 9, але є відносна різниця приблизно в 1000%, тому сезонність, яка виражена як відносний термін, може різко змінитися, і про неї слід подбати. під час побудови моделі.

Алгоритм Холта-Вінтера має широкі сфери застосування. Вона використовується в різних бізнес-задачах в основному через дві причини, одна з яких полягає в простому підході до реалізації, а інша полягає в тому, що модель буде розвиватися в міру того, як змінюються наші вимоги.

Модель часового ряду Холта-Вінтера є дуже потужним алгоритмом передбачення, незважаючи на те, що є однією з найпростіших моделей. Він може обробляти сезонність у наборі даних, просто обчислюючи центральне значення, а потім додаючи або помножуючи його на нахил і сезонність. Потрібно переконатися, що налаштували правильний набір параметрів, і найкраще підходить модель. Необхідно перевіряти ефективність моделі, використовуючи значення MAPE (середня абсолютна відсоткова помилка) або значення RMSE (середньоквадратична помилка), і точність може залежати від бізнес-проблеми та набору даних, доступних для навчання та тестування моделі.

2 Постановка та формалізація задачі

Проблема знаходження кількості відвідувачів торгівельного центру з використанням регресійних моделей є економічно важливою. Існують області діяльності людини, які використовують данні на основі відвідування закладів, та використовують для аналізу та прогнозування кількості витрат електроенергії, а також при закупівлі товару для магазинів торгівельного центру на певний період. В рамках ймовірностей ця задача має формулювання як оцінка регресійних моделей в результаті статичних експериментах і в практичних задачах являє собою певну область застосунку прикладного аналізу даних регресійного характеру.

Після вивчення результатів роботи моделей, було зроблено корегування похибки через ситуацію з отриманими даними. Реалізація регресійних моделей аналізувала дані (в нашому випадку дані були 2020 та 2021 року) аналогічного прогнозованого тижня минулого року, аналогічного тижня минулого місяця, дані тижня до прогнозованого, а також один день до прогнозу. Для покращення результату, через складність та обмеження відвідування в період карантинів, було прийнято рішення додати покращуючи коефіцієнти та інтерфейс користувача, де можливо обрати обмеження на дані.

Основна особливість вищеперерахованого методі – є певна спрощеність при знаходженні більш точної регресійної моделі, а також покращення користування програмою за допомогою інтерфейсу.

3 Алгоритм реалізації регресивних моделей

3.1 Підготовка вхідних даних до використання

Запис в масив даних кількості відвідувачів по індексу (місяць, день, години), розбивання на аналогічні частини для більш точного прогнозу без використання погодних умов. Після аналізу вхідних даних, було вирішено використовувати прогноз на періоди п’ять або сім днів, тобто робочий або повний тиждень, за відрізками годин а не по днях в цілому.

Для прискорення організації коректного відображення всіх даних по індексу в масиві, було перевірено документ на наявність пропущених елементів та при необхідності заповнення відсутніх даних.

3.2 Реалізація простої лінійної регресії

Визначення даних для роботи. Вхідні дані (регресорів, *x*) і вихідні дані (*y*) мають бути масивами або подібними об’єктами. Спосіб надання даних для регресії *x*, тобто по годинах кожен день:

```
for hour in range(0, 24):
  ListHour.append(hour)
  ListHour.append(hour)
  ListHour.append(hour)
  ListHour.append(hour)
x = np.array(ListHour).reshape((-1, 1))
```

(Заповнення масиву по кількості вхідних даних за годину (в нашому випадку 4 вхідних даних))

Створення моделі лінійної регресії та підгонки за наявними даними має команду `model = LinearRegression()`, цей оператор створює змінну model, як екземпляр `LinearRegression`. Також можливо надати кілька необов’язкових параметрів для LinearRegression:

- `fit_intercept`. Є логічним параметром (True за замовчуванням), який вирішує, чи обчислювати перехоплення b_0(True) чи вважати його рівним нулю (False).
- `normalize`. Є логічним значенням (False за замовчуванням), який вирішує, нормалізувати вхідні змінні чи ні.
- `copy_X`. Є логічним значенням (True за замовчуванням), який вирішує, копіювати чи перезаписувати вхідні змінні.
- `n_jobs`. Є цілим числом або None (за замовчування) і представляє кількість завдань, що використовуються в паралельних обчисленнях. None зазвичай означає одну роботу і *-1* використовувати всі процесори.

В даній моделі використовуються значення за замовчуванням для всіх параметрів.

Для використання моделі потрібно “зателефонувати” .fit() на model.

```
model = LinearRegression().fit(x, ListDataTemp)
```

Де `ListDataTemp` масив вихідних значень *y*.

Після встановлення моделі, для отримання результату, щоб перевірити, чи працює модель задовільно, та інтерпретувати її з коефіцієнтом на тиждень прогнозування (так як прогнозування проводилось на період з можливістю обмеження відвідування):

```
y_pred = model.predict(x)

ListDataPredict.extend(y_pred * coefForYYMMD[0])

Де coefForYYMMD є коефіцієнтом.
```

Для демонстрації результату роботи моделі лінійної регресії було обрано метод графічного зображення на рис. 3.1, а також запис в excel документ, для зручності використання виведених даних.

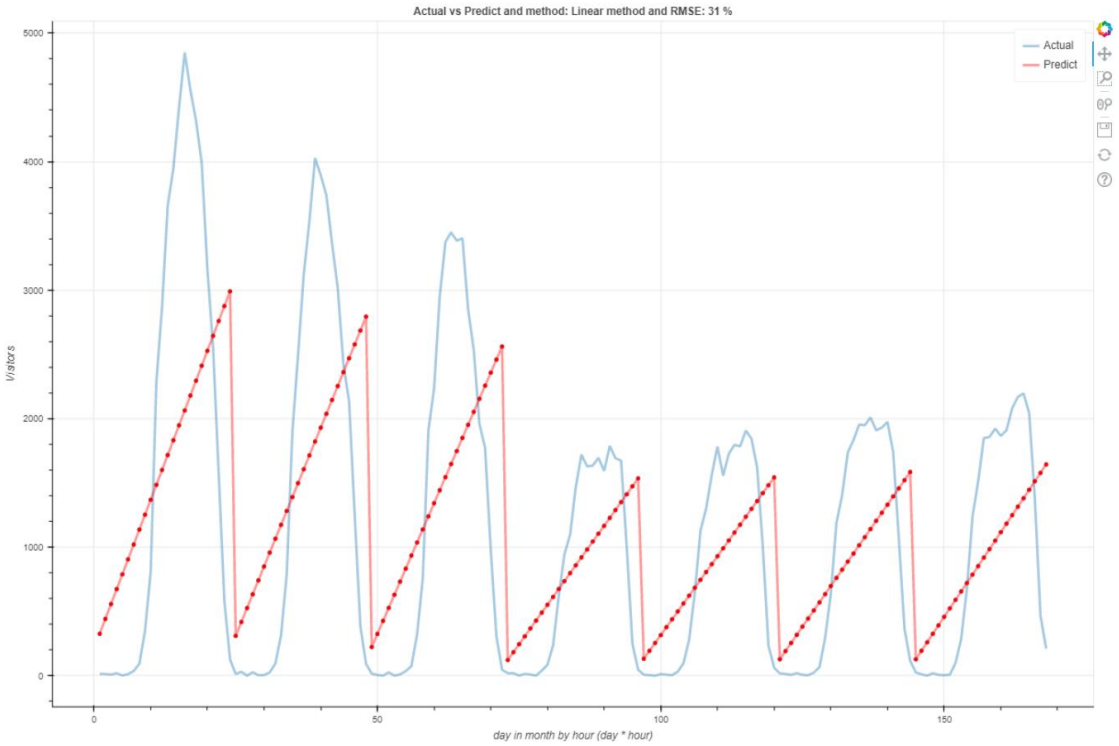

Рис. 3.1 Графік порівняння на 7 днів 5 місяця з використанням лінійної регресії.

Після отримання результату роботи моделі лінійної регресії та спроб покращення для більшості випадків, вдалося опустити коефіцієнт похибки до 31%. Через те, що результат простої лінійної регресії є лінія, графічне порівняння має специфічних вигляд, але для використання отриманих даних не потребує коректності графічного представлення, а тільки загальний результат роботи моделі. Надалі буде проведено роботу над покращенням загального результату роботи лінійної моделі, для зниження коефіцієнту похибки.

3.3 Реалізація поліноміальної лінійної регресії

Реалізація поліноміальної регресії схожа на лінійну регресію. Є лише один додатковий крок: необхідно перетворити масив вхідних даних, щоб включити нелінійні терміни.

Для створення екземпляру класу PolynomialFeatures потрібна команда:

```
pf = PolynomialFeatures(degree=5, include_bias=False)
```

pf це змінна, що посилається на екземпляр PolynomialFeatures, який можна використовувати для перетворення вхідних даних *x*.

Також є можливість надати кілька необов’язкових параметрів для PolynomialFeatures:

- `degree`. Це ціле число (2 за замовчуванням), яке представляє ступінь функції поліноміальної регресії.
- `interaction_only`. Є логічним значенням (False за замовчуванням), який визначає, чи включати лише функції взаємодії чи всі функції.
- `include_bias`. Є логічним значенням (True за замовчуванням), який визначає, чи включати стовпець зміщення (перехоплення) одиниць чи ні.

Перед застосуванням pf, його потрібно доповнювати .fit():

```
pf.fit(x)
```

Після встановлення, потрібно перетворити вхідний масив за допомогою .transform(). Він приймає вхідний масив як аргумент і повертає змінений масив.

```
x_ = pf.transform(x)
```

Створення моделі виглядає, як і в лінійній регресії:

```
model = LinearRegression().fit(x_, ListDataTemp)
```

Отримання результат прогнозування за допомогою поліноміальної регресії має вигляд, схожий на використання лінійної регресії. Для цього потрібен лише змінений вхідний текст замість оригіналу:

```
y_pred = abs(model.predict(x_))

ListDataPredict.extend(y_pred * coefForYYMMD[0])
```

На рис. 3.2 результати роботи поліноміальної регресії.

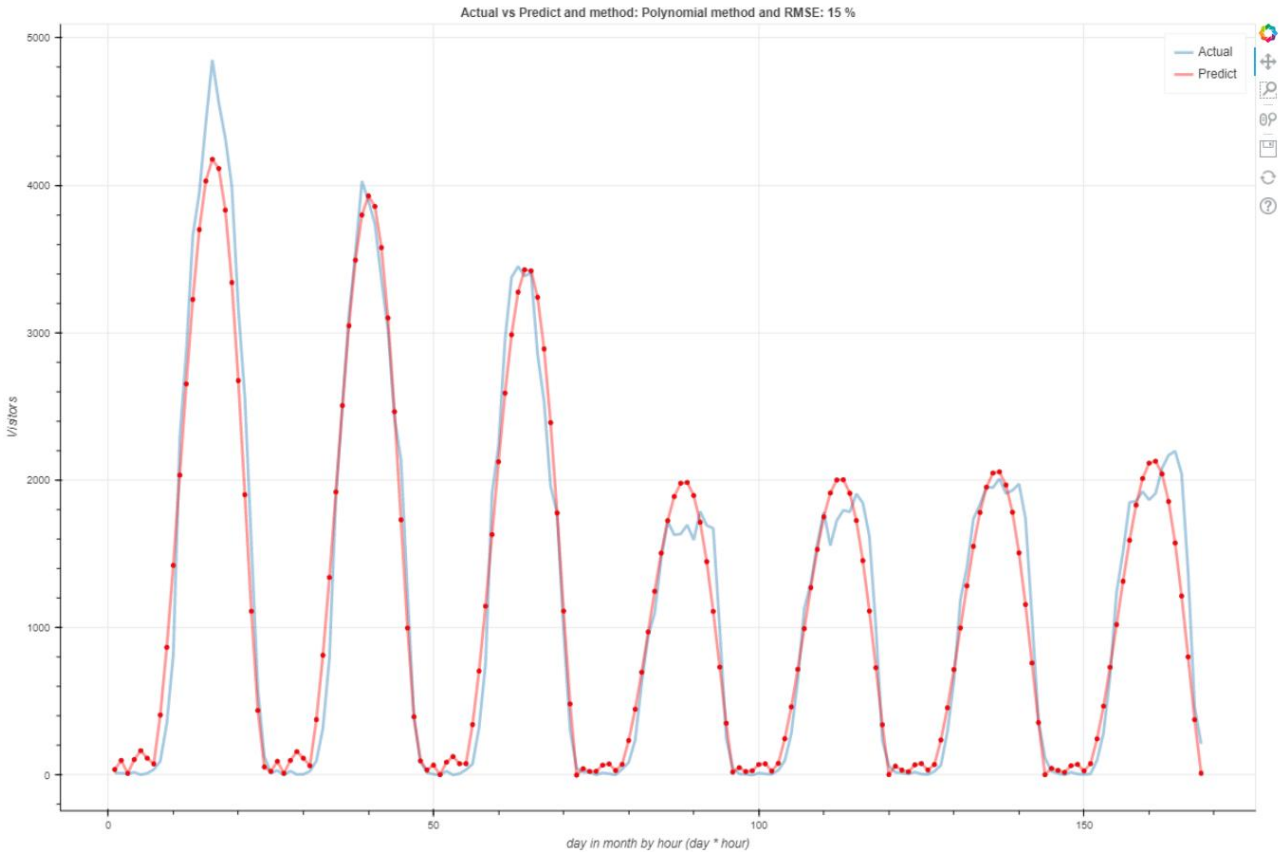

Рис. 3.2 Графік порівняння на 7 днів 5 місяця з використанням поліноміальної регресії.

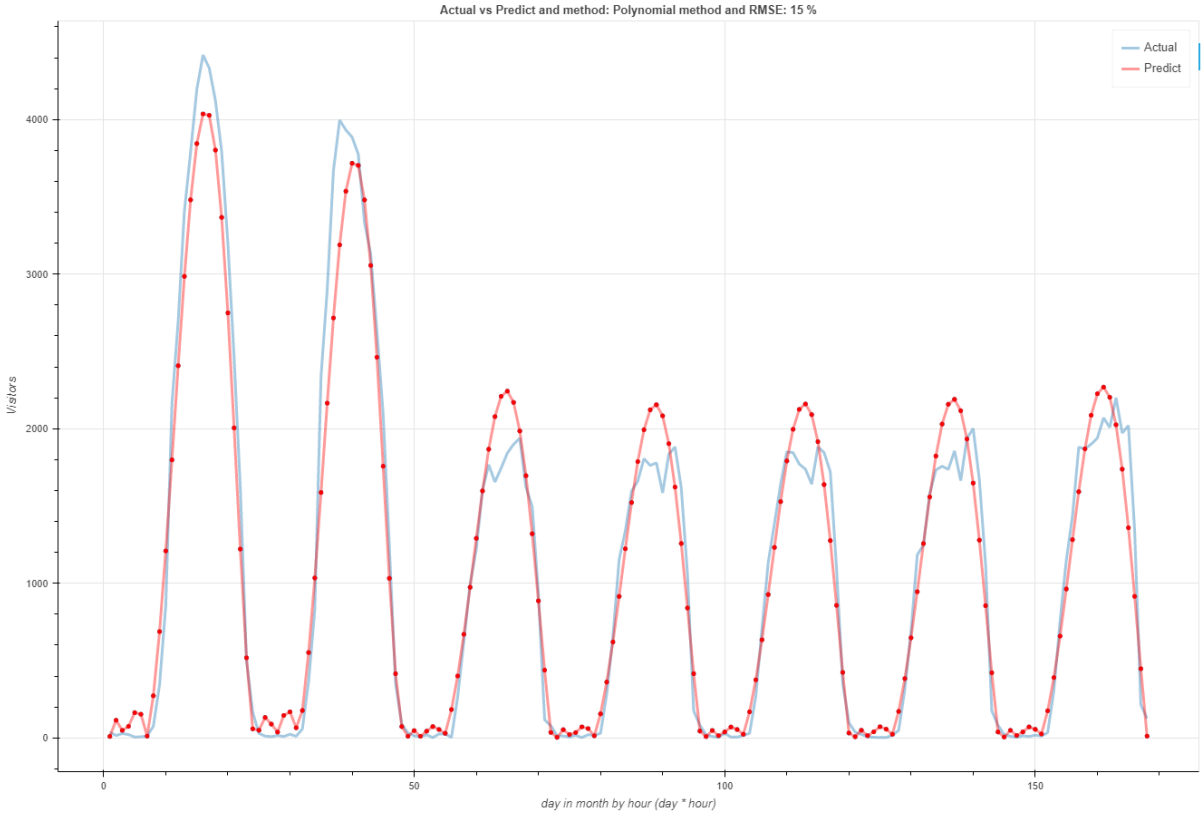

Рис. 3.3Графік 6 місяця з використанням поліноміальної регресії.

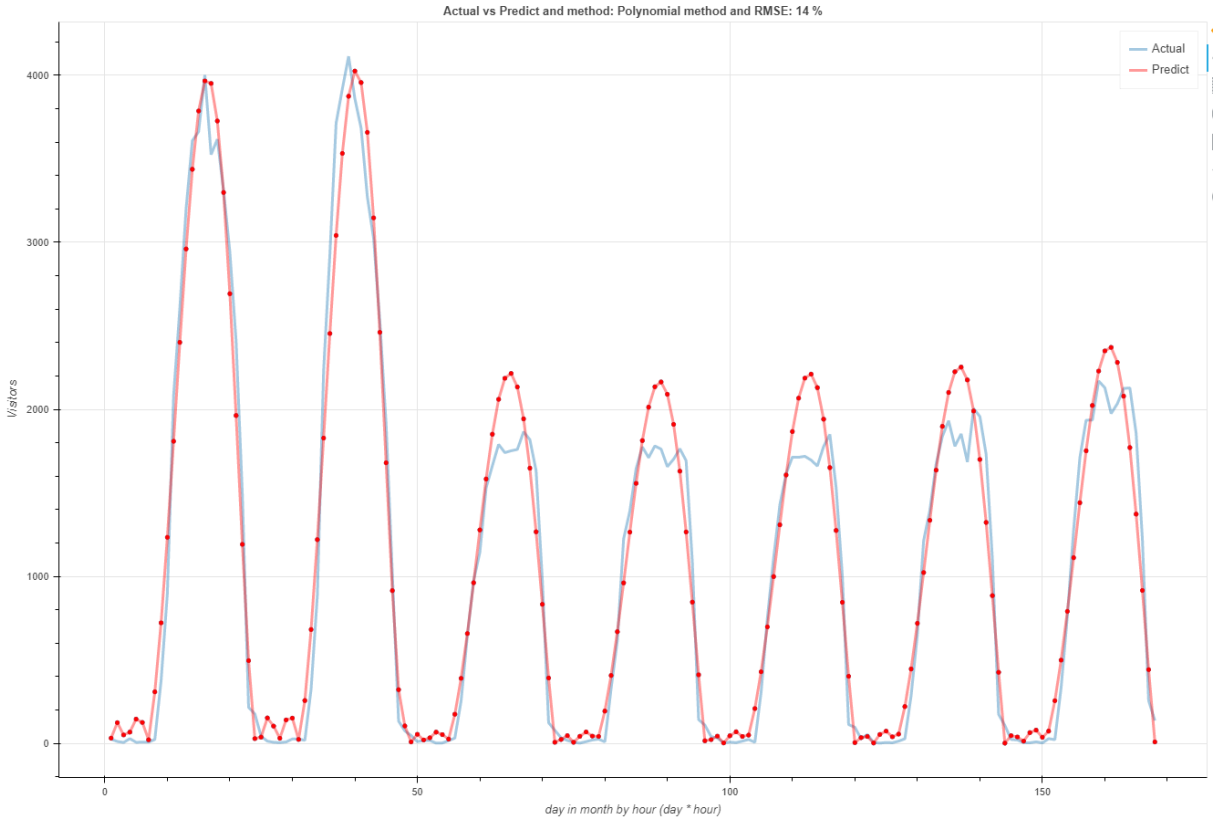

Рис. 3.4 Графік 7 місяця з використанням поліноміальної регресії.

Після отримання результату роботи моделі поліноміальної регресії, було проведено аналіз для зниження коефіцієнту похибки загального результату роботи до 15%. Графічний спосіб представлення результату, прогнозованих даних відвідування за допомогою поліноміальної регресії з дійсними отриманими значеннями, зручний для попереднього порівняння та аналізу для покращення роботи коду. Загальний результат порівняння має похибку 15%, надалі буде проведено роботу над покращенням алгоритму та підбір коефіцієнтів для збільшення точності.

При порівнянні результату простої лінійної з поліноміальним регресійним методом, наглядно видно особливості поліноміальної регресії, такі як зв'язок між незалежною та залежною змінною, коли залежна змінна пов’язана з незалежною змінною п’ятого ступеня. Так як вхідні дані та результат є нелінійними даними, то лінійна регресія не може спрогнозувати лінію, яка найкраще підходить, у відмінності від поліноміальної регресії, яка допомагає виявити криволінійний зв'язок між незалежними та залежними змінними.

3.4 Реалізація методу ARIMA

Підбір даних для навчання моделі ARIMA був обраний на основі року до необхідного прогнозованого тижня. Під час запуску алгоритм знаходження найкращих коефіцієнтів був зупинений через нехватку оперативної пам’яті, тому було обрано параметри при даних одного місяця.

Наразі параметри p = 1, d = 1, q = 1, P = 1, D = 1, Q = 1, S = 12. Опис параметрів міститься в розділі 1.6, вибір параметрів було обрано підбором.

```
model = ARIMA(TempListForARIMA, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12)).fit()
```

На рис. 3.5 результат роботи методу ARIMA.

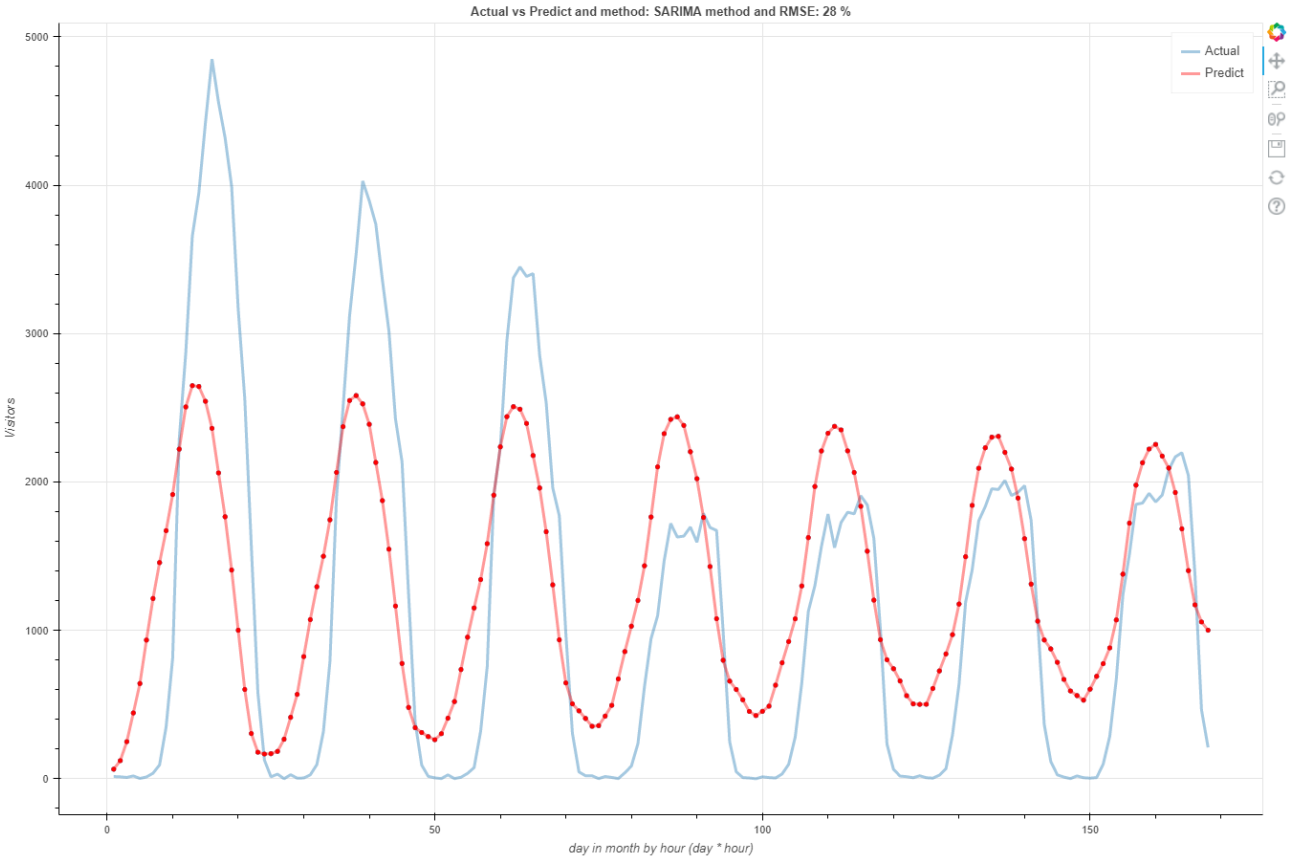

Рис. 3.5 Графік порівняння на 7 днів 5 місяця з використанням моделі ARIMA.

Після отримання результату роботи моделі ARIMA проаналізовано різницю між очікуваними та отриманими даними на основі коефіцієнту

похибки, який складає 28%, що є непоганим результатом, хоч і графічно має не достатньо необхідний вигляд, але загальний результат, котрий необхідно отримати, має потенціал. Надалі буде проведено роботу над покращенням, а також знаходженням параметрів, для зменшення коефіцієнту.

При порівнянні отриманих результатів лінійної, поліноміальної регресії, а також ARIMA виявлено метод отримання найкращого результату – поліноміальна регресія.

3.5 Реалізація методу Холт-Вінтера

Вхідні дані для використання методу Холта-Вінтера було обрано аналогічно, як для лінійної та поліноміальної регресії, що дає змогу напряму порівнювати результат роботи даних регресійних моделей.

Результат роботи методу Холта-Вінтера на рис. 3.6 схожий на результат роботи лінійної регресії з рис. 3.1.

```
model = ExponentialSmoothing(np.asarray(ListDataTemp), seasonal_periods=7, trend='add', seasonal='add').fit()
```

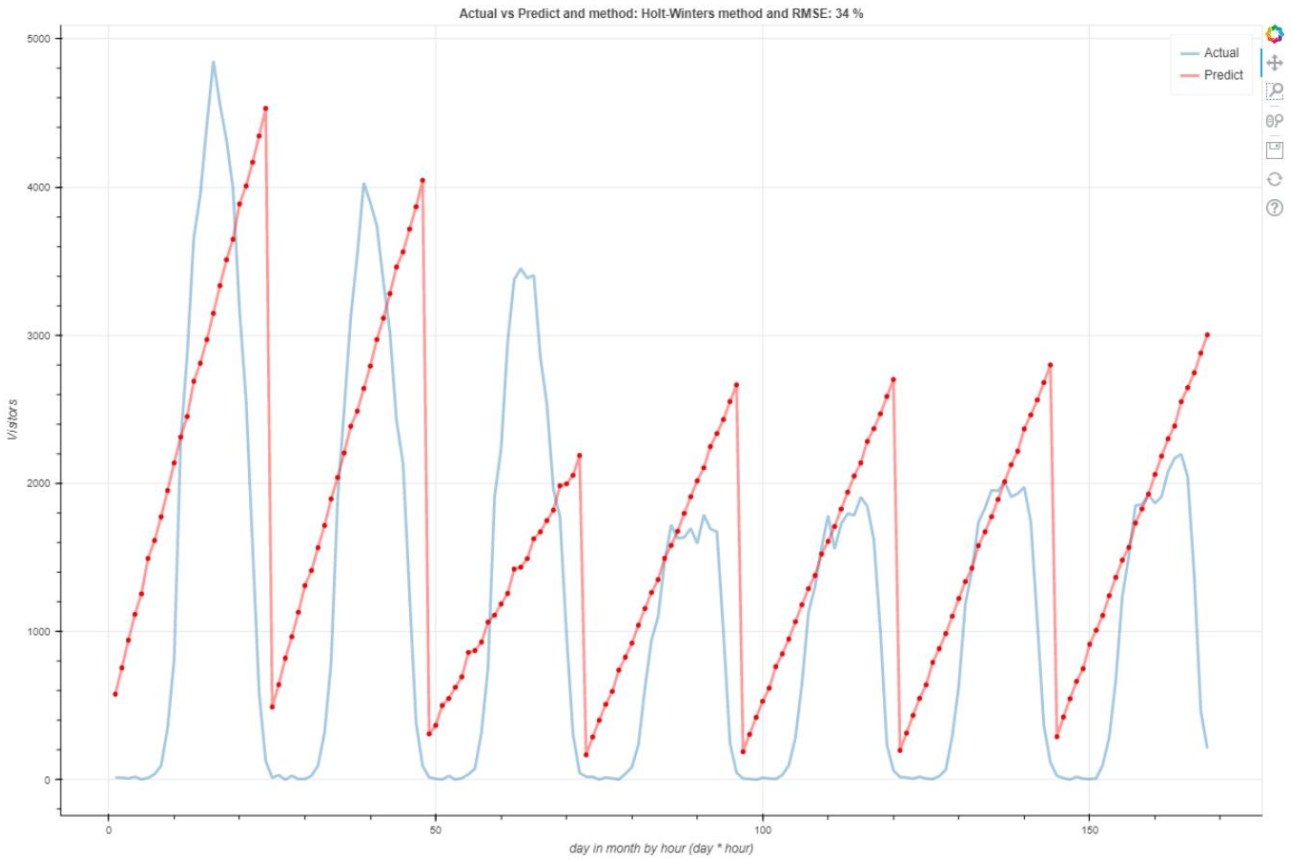

Рис. 3.6 Графік порівняння моделі Холта-Вінтера 5 місяця.

Після отримання результату роботи методу Холт-Вінтера та спроб покращення для більшості випадків, вдалося опустити коефіцієнт похибки до 34%. Через те, що результат роботи даного методу не є необхідним та задовільним, був проведений аналіз помилок використання. Графічне порівняння має не коректний вигляд та помилки, такі як високі результати

відмінності від дійсних значень певних годин, а також практично пряму лінію. Загальне порівняння результату даних має високу похибку в 34%, дана похибка була отримана при покращенні результату. Надалі будуть проведені роботи з покращенням використання даного методу і отриманням більш точних результатів, як графічних так і загальних даних.

3.6 Інтерфейс, способи виведення та результат роботи

Для представлення інтерфейсу користувача було взято бібліотеку tkinter Python. Tkinter – стандартний інтерфейс Python для набору інструментів GUI. Запуск з командного рядка відкриває вікно, що демонструє простий для користування інтерфейс:

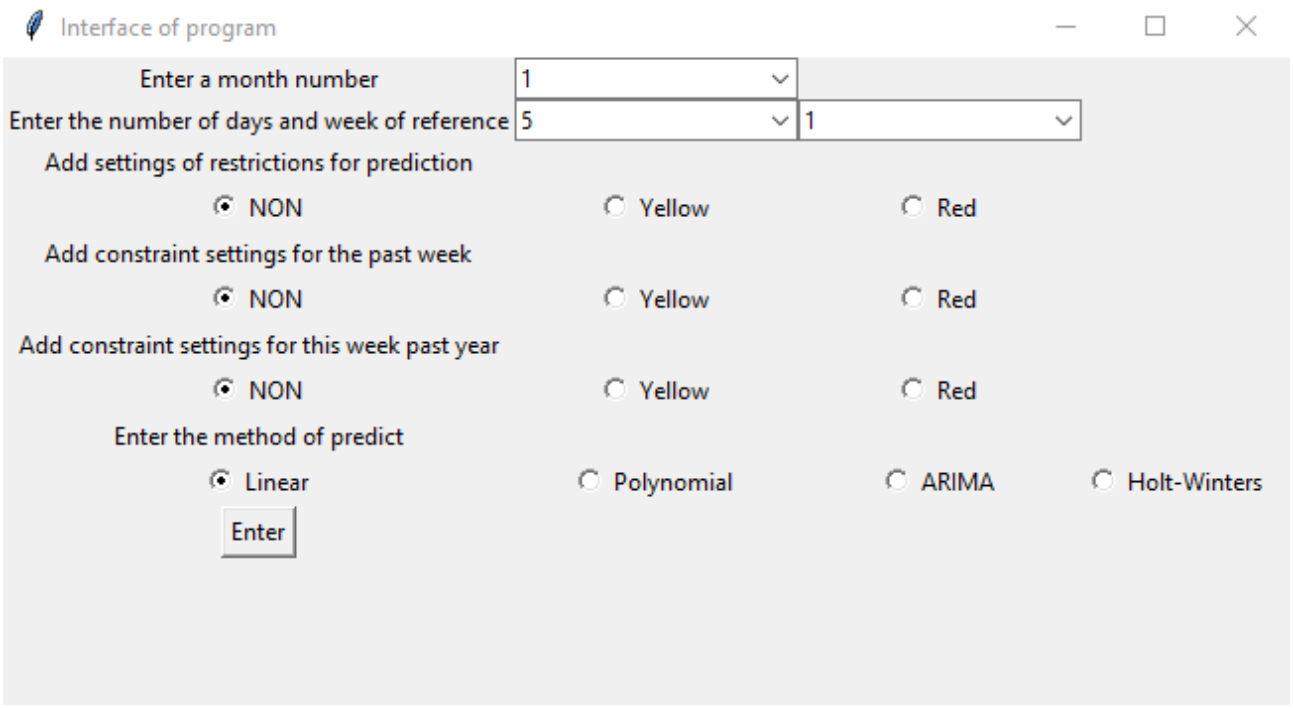

Рис. 3.7 Інтерфейс користувача.

Для гнучкості користування було додано вибір обмежень на певний період вхідних даних, а також вибір дати прогнозування. Спочатку потрібно обрати номер місяця, кількість днів (п’ять або сім, тобто робочий або повний тиждень), номер тижня в місяці, а також обмеження на вхідні дані за даними критеріями (на тиждень аналогічний поточному в минулому році, в минулому місяці, а також минулий тиждень і використання минулого дня для наступного). В кінці обрати модель для прогнозування і вивести результати за допомогою графічного і excel документа з порівнянням отриманого результату з очікуваним.

Графічний спосіб представлення результату роботи реалізований за допомогою бібліотеки bokeh.plotting. І запис в excel документ через бібліотеку Pandas.

Для покращення результату додано вагові коефіцієнти червоної зони 0.075 вихідних та 0.35 робочі дні а також жовтої зони карантину 0.5 та 0.6 відповідно.

Вхідні дані зчитуються з excel документів рис. 3.8 та зберігаються в масив.

січень	4	понеділок	20		1440
січень	4	понеділок	21		750
січень	4	понеділок	22		138
січень	4	понеділок	23		135
січень	5	вівторок	0		59
січень	5	вівторок	1		16
січень	5	вівторок	2		13
січень	5	вівторок	3		5
січень	5	вівторок	4		7
січень	5	вівторок	5		7

Рис. 3.8 Excel документ з вхідними даними.

Порівняння середньоквадратичних похибок в результаті роботи регресійних моделей представлені в таблиці 3.9 та обрано найкращий метод.

Лінійна регресія	31%
Поліноміальна регресія	**15%**
Модель ARIMA	28%
Модель Холта-Вінтера	34%

Таблиця 3.9 Порівнянні середньоквадратичних похибок.

Висновки

В даній роботі було розглянуто регресійні моделі: лінійна, поліноміальна регресії, метод ARIMA а також Холт-Вінтера.

Розроблено програмне забезпечення мовою програмування Python з використанням бібліотеки matplotlib, ARIMA, pandas, math, sklearn та tkinter для прогнозування кількості відвідувачів торгівельному центру за допомогою даних регресійних моделей.

Методом для отримання найкращого результату роботи прогнозування програмою є поліноміальна регресія з п’ятим ступенем поліному та середньоквадратичною похибкою 15 відсотків, в порівнянні з лінійною моделлю – 31 відсоток, ARIMA – 28 відсотків та Холта-Вінтера – 34 відсотки, при вхідних даних періоду аналогічного місяця минулого року, минулого місяця, а також минулий день і минулий тиждень, з ваговими коефіцієнтами червоної зони 0.075 вихідних та 0.35 будні а також жовтої зони карантину 0.5 та 0.6 відповідно.

Список використаних джерел

1. Мокін Б. І., Мокін В. Б., Мокін О. Б. Математичні методи ідентифікації динамічних систем (2010). – Навчальний посібник. – Вінниця : ВНТУ, 2010. – 260 с.

2. Бакай Є. І., Кабачій В. В., Маслій Р. В. Модель прийняття рішень для фінансових часових рядів на основі пари середніх з використанням оцінки різних часових вимірів / Вісник Вінницького політехнічного інституту. – 2017. – № 1. – С. 70-77.

3. Квєтний Р. Н. Імовірнісні нейронні мережі в задачах ідентифікації часових рядів [Електронний ресурс] / Р. Н. Квєтний, В. В. Кабачій, О. О. Чумаченко // Наукові праці Вінницького національного технічного університету. – 2010. – № 3. – 6 с. – Режим доступу до журн. : http://www.nbuv.gov.ua/e-journals/VNTU/2010-3/2010-3.html .

4. Прогнозування часових рядів [Електронний ресурс] – Режим доступу до документу: https://core.ac.uk/download/pdf/288837986.pdf.

5. Модель ARIMA – Повний посібник із прогнозування часових рядів у Python [Електронний ресурс] – Режим доступу до посібника: https://www.machinelearningplus.com/time-series/ARIMA-model-time-series-forecasting-python/.

6. Стаття з сайту Read Python - «Лінійна регресія в Python» [Електронний ресурс] – режим доступу: (https://realpython.com/linear-regression-in-python/#conclusion).

7. Стаття з сайту proglib - «Лінійна регресія в Python» [Електронний ресурс] – режим доступу: https://proglib.io/p/linear-regression.

8. Стаття з сайту matplotlib, тема- «matplotlib.pyplot» [Електронний ресурс] – режим доступу до статті: https://matplotlib.org/3.3.3/api/_as_gen/matplotlib.pyplot.html.

9. Стаття з сайту GitHub - «Prediction of Bildings Energy» [Електронний ресурс] – режим доступу: https://cs109-energy.github.io/building-energy-consumption-prediction.html.

10. Стаття з сайту hindawi.com - «Epileptic Seizures Prediction Using Machine Learning Methods» [Електронний ресурс] – режим доступу: https://www.hindawi.com/journals/cmmm/2017/9074759/.

11. Все, що потрібно знати про поліноміальну регресію [Електронний ресурс] – Режим доступу до статті: https://www.analyticsvidhya.com/blog/2021/07/all-you-need-to-know-about-polynomial-regression/.

12. Aleksandar Peckov, "A MACHINE LEARNING APPROACH TO POLYNOMIAL REGRESSION", Doctoral Dissertation, Jozef Stefan International Postgraduate School, 2012.

13. Прогнозування часових рядів за допомогою ARIMA в Python3 [Електронний ресурс] – режим доступу: https://www.8host.com/blog/prognozirovanie-vremennyx-ryadov-s-pomoshhyu-arima-v-python-3/.

14. Аналіз часових рядів [Електронний ресурс] – Режим доступу до статті: https://dokumen.pub/time-series-analysis-9780387759593-038775959x.html.

15. Приклад моделі ARIMA Python [Електронний ресурс] – Режим доступу до статті: https://towardsdatascience.com/machine-learning-part-19-time-series-and-autoregressive-integrated-moving-average-model-ARIMA-c1005347b0d7.

16. Джеффрі Стрікленд Аналіз часових рядів і прогнозування за допомогою Python & R. – (2020) . – 448 с.

17. Посилання на Git репозиторій коду програми, режим доступу: «https://github.com/MOlexiy/diplom».

Додаток А Код програми

Код створення та заповнення масиву з індексами, для використання з можливістю по місяцям, дням а також годинам:

```
 def GetDayInMonth(year, month):
  day = calendar.monthrange(year, month)[1]
  return day
def CreateEmptyList(year):
  list = [[]]
  for month in range(1, 13):
    List1 = [[]]
    days = GetDayInMonth(year, month)
    for day in range(1, days+1):
      List2 = []
      for hour in range(1, 25):
        List3 = [hour]
        List2.extend(List3)
      List1.append(List2)
    list.append(List1)
  return list
def CreateTestList(List):
  row = 7
  percent = 0
  for month in range(1, 13):
    days = GetDayInMonth(year1, month)
```

```
        for day in range(1, days+1):
          for hour in range(0, 24):
            if sheet_1[row][6].value == hour:
              List[month][day][hour] = sheet_1[row][8].value
              row += 1
            else:
              List[month][day][hour] = random.randint(0, 20)
        percent += 4
        print(percent, '%')
      print('end read exel')
      return List
    def CreatePredictList(List):
      row = 7
      percent = 52
      for month in range(1, 13):
        days = GetDayInMonth(year2, month)
        for day in range(1, days + 1):
          for hour in range(0, 24):
            if  sheet_2[row][5].value  ==  hour:  #  Так  как  ексель  с
пропущенными данными, то мы их добавляем рандомайзером от 1 до 10
              List[month][day][hour] = sheet_2[row][7].value
              row += 1
            else:
              List[month][day][hour] = random.randint(0, 20)
        percent += 4
```

```
        print(percent, '%')
      print('end read second exel')
      return List
```

Код графічного виведення результату роботи:

```
def PrintGraphics(x, y, y_pred, text, rmse):
      rmse = str(rmse)
      method = ""
      if text == 0:
        method = "Linear method"
      elif text == 1:
        method = "Polynomial method"
      elif text == 2:
        method = "SARIMA method"
      elif text == 3:
        method = "Холт-Вінтера method"
      p = figure(title="Actual vs Predict and method: " + method + " and
RMSE: " + rmse + " %", width=1350, height=900)
      p.title.align = 'center'
      p.circle(x, y_pred)
      p.line(x, y, legend_label='Actual', line_width=3, line_alpha=0.4)
      p.circle(x, y_pred, color="red")
      p.line(x,     y_pred,     color="red",     legend_label='Predict',
line_width=3, line_alpha=0.4)
      p.xaxis.axis_label = 'day in month by hour (day * hour)'
      p.yaxis.axis_label = 'Visitors'
```

```
  show(p)
```

Код інтерфейсу та виклику логіки програми:

```
def clicked():
  enterMonth = int(combo0.get())
  countDays = int(combo1.get())
  numberWeek = int(combo2.get()) - 1
  text0 = int(selected0.get())
  text1 = int(selected1.get())
  text2 = int(selected2.get())
  text3 = int(selected3.get())
  lvl0 = getLvl(text0)
  lvl1 = getLvl(text1)
  lvl2 = getLvl(text2)
  Regress(enterMonth, countDays, numberWeek, lvl0, lvl1, lvl2,
text3)
# Display menu
window.title("Interface of program")
window.geometry('650x330')
lbl0 = Label(window, text="Enter a month number")
lbl0.grid(column=0, row=0)
combo0 = Combobox(window)
combo0['values'] = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)
combo0.current(0)
```

```
    combo0.grid(column=1, row=0)

    lbl1 = Label(window, text="Enter the number of days and week of reference")

    lbl1.grid(column=0, row=1)

    combo1 = Combobox(window)

    combo1['values'] = (5, 7)

    combo1.current(0)

    combo1.grid(column=1, row=1)

    combo2 = Combobox(window)

    combo2['values'] = (1, 2, 3, 4)

    combo2.current(0)

    combo2.grid(column=2, row=1)

    lbl2 = Label(window, text="Add settings of restrictions for prediction")

    lbl2.grid(column=0, row=2)

    selected0 = IntVar()

    rad0 = Radiobutton(window, text='NON', value=0, variable=selected0)

    rad1 = Radiobutton(window, text='Yellow', value=1, variable=selected0)

    rad2 = Radiobutton(window, text='Red', value=2, variable=selected0)

    rad0.grid(column=0, row=3)

    rad1.grid(column=1, row=3)

    rad2.grid(column=2, row=3)

    lbl3 = Label(window, text="Add constraint settings for the past week")
```

```
    lbl3.grid(column=0, row=4)

    selected1 = IntVar()

    radb0 = Radiobutton(window, text='NON', value=0, variable=selected1)

    radb1     =     Radiobutton(window,     text='Yellow',     value=1,
variable=selected1)

    radb2 = Radiobutton(window, text='Red', value=2, variable=selected1)

    radb0.grid(column=0, row=5)

    radb1.grid(column=1, row=5)

    radb2.grid(column=2, row=5)

    lbl4 = Label(window, text="Add constraint settings for this week
past year")

    lbl4.grid(column=0, row=6)

    selected2 = IntVar()

    radiobutton0    =    Radiobutton(window,    text='NON',    value=0,
variable=selected2)

    radiobutton1    =    Radiobutton(window,    text='Yellow',    value=1,
variable=selected2)

    radiobutton2    =    Radiobutton(window,    text='Red',    value=2,
variable=selected2)

    radiobutton0.grid(column=0, row=7)

    radiobutton1.grid(column=1, row=7)

    radiobutton2.grid(column=2, row=7)

    lbl5 = Label(window, text="Enter the method of predict")

    lbl5.grid(column=0, row=8)

    selected3 = IntVar()
```

```
    predictBut0 = Radiobutton(window, text='Linear', value=0, variable=selected3)

    predictBut1 = Radiobutton(window, text='Polynomial', value=1, variable=selected3)

    predictBut2 = Radiobutton(window, text='ARIMA', value=2, variable=selected3)

    predictBut3 = Radiobutton(window, text='Холт-Вінтера', value=3, variable=selected3)

    predictBut0.grid(column=0, row=9)

    predictBut1.grid(column=1, row=9)

    predictBut2.grid(column=2, row=9)

    predictBut3.grid(column=3, row=9)

    btn = Button(window, text="Enter", command=clicked)

    btn.grid(column=0, row=10)

    # end Display menu

    window.mainloop()
```

Код роботи регресійних методів:

```
     if text3 == 0:

      model = LinearRegression().fit(x, ListDataTemp)

      y_pred = model.predict(x)

      ListDataPredict.extend(y_pred * coefForYYMMD[0])

    elif text3 == 1:

      x_ = pf.transform(x)

      model = LinearRegression().fit(x_, ListDataTemp)
```

```
      y_pred = abs(model.predict(x_))
      ListDataPredict.extend(y_pred * coefForYYMMD[0])
        elif text3 == 3:
          model     =     ExponentialSmoothing(np.asarray(ListDataTemp),
seasonal_periods=7, trend='add', seasonal='add').fit()
          y_pred = abs(model.forecast(24))
          ListDataPredict.extend(y_pred * coefForYYMMD[0])
      if text3 == 2:
        model    =    ARIMA(TempListForARIMA,    order=(1,    1,    1),
seasonal_order=(1, 1, 1, 12)).fit()
        countOfDay = int(24 * countDays)
        yhat  =  abs(model.predict(start=5713,  end=5712  +  countOfDay,
alpha=0.05, dynamic=False))
        ListDataPredict.extend(yhat * coefForYYMMD[0])
```