**ניתוח אלגוריתם - Gaussian Process Regression עם - Ridge ו – Nystroem Kernel Approximation**

**מבוא**

בפרויקט זה נעשה שימוש ברגרסיה גאוסיאנית משולבת עם Ridge Regression, תוך שימוש בשיטת Nystroem Kernel Approximation. המטרה הייתה לבצע חיזוי מחירי טיסות בהתבסס על מאפיינים שונים ולבחון אילו פרמטרים מניבים את התוצאות הטובות ביותר.

**למה לא Gaussian Process רגיל?**

רגרסיה גאוסיאנית רגילה היא אלגוריתם רב עוצמה, אך כאשר כמות הנתונים גדלה, מורכבות החישוב גדלה לרמת O(n3)O(n^3) מה שהופך את השיטה ללא ישימה עבור מאגרי נתונים גדולים כמו זה שלנו (500,000 דגימות). כדי להתגבר על בעיה זו, השתמשנו ב-**Nystroem Kernel Approximation** שמאפשר גישה חישובית יעילה יותר באמצעות דגימת תת-קבוצה של הנקודות כדי לבצע קירוב לפונקציית הקרנל.

**רגרסיית - Ridge**

רגרסיית Ridge היא הרחבה של רגרסיה לינארית רגילה עם רגולריזציה L2L2. היא מוסיפה עונש על גודל המשקלים במודל כדי למנוע אוברפיטינג. בפרויקט השתמשנו ב-α=0.05\alpha=0.05, שהוא פרמטר הרגולריזציה שנבחר לאחר Grid Search.

**שימוש ב-Nystroem Kernel Approximation**

כדי להתמודד עם החישובים הכבדים של קרנלים, השתמשנו בשיטת Nystroem, שמאפשרת לקרב את מטריצת הקרנל ע"י דגימה של מספר מופחת של רכיבים.

**כיצד מתקבלת המשוואה של המודל ומה קורה לנתונים בפועל?**

כאשר אנחנו משתמשים ברגרסיית Ridge עם Nystroem Kernel Approximation אנחנו בעצם יוצרים טרנספורמציה לא-לינארית של הנתונים, ומריצים עליה מודל לינארי. בואו נראה מה קורה מתמטית לכל שלב.

**פרמטרים שנבדקו:**

* **n\_components:** מספר הרכיבים בהם נעשה שימוש לקירוב הקרנל. לאחר בדיקות, נמצא כי ncomponents=7000 מניב ביצועים מיטביים.
* **קרנלים שנבדקו:**

**RBF (Radial Basis Function)** - קרנל נפוץ המתאים לבעיות לא לינאריות.

**Polynomial** - קרנל פולינומי המתאים ליחסים מסובכים יותר בין הנתונים.

**Sigmoid** - קרנל המדמה רשתות נוירונים אך נמצא כפחות מתאים.

**Laplacian** - קרנל דומה ל-RBF אך נותן משקל חזק יותר לערכים קרובים מאוד, ונמצא כמניב תוצאות טובות ביותר.

בסופו של דבר, הקרנל **Laplacian** נמצא כמתאים ביותר, ולכן נבחר.

**הערכת Overfitting / Underfitting**

* התוצאות מראות כי המודל מספק ביצועים טובים הן על סט האימון והן על סט הבדיקה, ללא פער משמעותי, מה שמצביע על כך שאין אוברפיטינג משמעותי.
* הקרנל **Laplacian**  והרגולריזציה ב-Ridge עוזרים להימנע מאוברפיטינג ע"י שליטה במורכבות המודל.
* אם היה אוברפיטינג, ניתן היה להפחית את **n\_components** או להגדיל את **alpha** ברגרסיית Ridge כדי להפחית מורכבות המודל.

**מסקנות**

באמצעות שילוב של Nystroem Kernel Approximation ורגרסיית Ridge הצלחנו להתמודד עם גודל הנתונים הגדול ולשמור על ביצועים טובים. הבחירה בקרנל Laplacian נמצאה כמתאימה ביותר, והרגולריזציה α=0.05\alpha=0.05 ברגרסיית Ridge נתנה תוצאות מאוזנות ומדויקות.

ניתן לשפר את המודל בעתיד על ידי ניסוי קרנלים נוספים, שינוי מספר הרכיבים ב-Nystroem, או שימוש בשיטות Feature Selection מתקדמות יותר.

**פירוט מורחב:**

Gaussian Process (GP) - תהליך גאוסי הוא מודל הסתברותי שמתאר התפלגות של פונקציות רציפות, כמו לאורך ציר הזמן או במרחב, בהנחה שכל אוסף סופי של משתנים מקריים מתפלג נורמלית רב-ממדית.

Gaussian Process Regression (GPR) משמש בעיקר לבעיות רגרסיה, כלומר חיזוי ערכים רציפים, ומציע יתרונות ייחודיים כמו אומדן אי-ודאות וגמישות במידול נתונים מורכבים. האלגוריתם נפוץ בתחומים כמו חיזוי סדרות זמן, מחירים וטמפרטורה, אופטימיזציה של פונקציות לא ידועות (Black Box Optimization) והערכת רמת ביטחון בתחזיות, מה שהופך אותו לכלי חיוני בניתוח רפואי, זיהוי הונאות ויישומים נוספים.

**מאפיינים מרכזיים של האלגוריתם:**

* ייצוג פונקציות כמשתנים מקריים: במקום לקבוע נוסחה קבועה לרגרסיה, GPR מגדיר התפלגות על פונקציות, המאפשרת חיזוי נקודות חדשות עם רמת אי-ודאות.
* חיזוי לאורך ציר זמן: האלגוריתם מתאים במיוחד לבעיות ניבוי סדרות זמן, שכן הוא ממזער את השונות בין נקודות מדידה ומשפר את הדיוק של התחזיות.
* התאמה דינמית: המודל אינו פרמטרי ואינו מוגבל לצורה מתמטית קבועה מראש, אלא מתאים את עצמו לנתונים הקיימים.

רכיבים מרכזיים בהתפלגות הגאוסיאנית - בתהליך גאוסיאני, החיזוי מבוסס על שלושה רכיבים מרכזיים המגדירים את מבנה ההתפלגות ומאפשרים חיזוי מדויק תוך מדידת אי-ודאות.

* וקטור ממוצעים (μ): מייצג את הערכים הצפויים עבור כל נקודת נתונים. משמש כבסיס לחיזוי.
* סטיית תקן (σ): משקפת את רמת אי-הוודאות סביב הממוצע. ככל שסטיית התקן גדולה יותר, כך רמת הביטחון בתחזית נמוכה יותר.
* מטריצת קו-וריאציה (Σ): מתארת את התלות בין נקודות הנתונים. ככל שהקו-וריאנס בין נקודות גבוה יותר, כך החיזוי מדויק יותר.
* מרכיב נוסף הוא הקרנל (Kernel)

**פונקציות גרעין (Kernel Functions) בתהליך גאוסיאני:**

הקרנל (Kernel) הוא רכיב מרכזי בתהליך הגאוסיאני, שכן הוא מכתיב כיצד מחושבת מטריצת הקו-וריאנס (Σ) וקובע את הקשרים בין נקודות שונות במרחב הנתונים. תפקידו למדוד דמיון בין נקודות – ככל שהן קרובות יותר, כך הקשר ביניהן חזק יותר. בכך, הקרנל משפיע ישירות על מבנה מטריצת הקו-וריאנס, מה שקובע את איכות התחזיות והיכולת של המודל ללמוד דפוסים מתוך הנתונים

תפקיד הקרנל:

* קובע את התלות בין נקודות נתונים על ידי מדידת מידת הדמיון ביניהן.
* מאפשר למודל להתמודד עם מבנים לא לינאריים על ידי שינוי ייצוג הנתונים.
* משפיע על גמישות המודל ודיוק התחזיות בהתאם לסוג הקרנל שנבחר.

קרנלים נפוצים:

* RBF (Radial Basis Function): מתאים לקשרים חלקים ורציפים, הקרנל הנפוץ ביותר.
* Linear Kernel: מתאים למערכות לינאריות.
* Polynomial Kernel: מתאים לקשרים פולינומיאליים.
* Matern Kernel: מתאים לנתונים רועשים ומחוספסים.
* White Kernel: משמש לייצוג רעש בנתונים

אתגרים בעבודה עם קרנלים:

* מורכבות גבוהה – קשה לפרש אינטואיטיבית קרנלים מורכבים.
* בחירה לא טריוויאלית – לרוב מתבצעת בניסוי וטעייה.
* עלות חישובית גבוהה – מודלים גאוסיאניים דורשים משאבים חישוביים משמעותיים, במיוחד עם דאטה גדול.

**איך האלגוריתם עובד?**

1. הנחת קשרים בין נקודות - האלגוריתם מניח שיש קשר סטטיסטי בין כל הנקודות במרחב. קשר זה נקבע על ידי פונקציית הקרנל (Kernel), שמודדת את מידת הדמיון בין נקודות שונות. ככל ששתי נקודות קרובות יותר במרחב, כך הקשר ביניהן חזק יותר.
2. חיזוי ערכים חדשים - בהינתן נקודות עם ערכים ידועים, האלגוריתם משתמש בקרנל כדי להעריך את מידת הדמיון שלהן לנקודה חדשה. על בסיס קשרים אלו, ניתן לבצע חיזוי של הערך החדש בצורה רציפה ומדויקת, תוך שקלול אי-הוודאות של התחזית.
3. מטריצת הקו-וריאנס - מטריצת Covariance Matrix מתארת את התלות בין הנקודות בנתונים. קשרים אלה מאפשרים למודל להשתמש במידע על נקודות קיימות כדי לאמוד ערכים של נקודות חדשות. ככל שהקו-וריאנס בין נקודות גבוה יותר, כך החיזוי יהיה אמין יותר.

למידה אקטיבית: למידה אקטיבית היא שיטה חיצונית לשיפור מודלים כמו Gaussian Process Regression (GPR), אך אינה חלק מובנה מהאלגוריתם עצמו. בתהליך זה, המודל מתחיל עם כמות קטנה של נתוני אימון (Train Set) כדי לחסוך במשאבים חישוביים ולבצע חיזוי ראשוני, תוך שימוש בסט בדיקה (Test Set) גדול יחסית להערכת ביצועיו. אם המודל שוגה בתחזיותיו, מתבצע עדכון דינמי, שבו מוסיפים נתונים חדשים לסט האימון כך שהמודל מתאים את עצמו לנקודות חדשות ולומד מהטעויות. שיטה זו חוסכת משאבים, משפרת את הדיוק בהדרגה, ומותאמת במיוחד למצבים שבהם איסוף נתונים הוא יקר או מוגבל (כגון בדיקות רפואיות). עם זאת, GPR אינו מבצע למידה אקטיבית כברירת מחדל, וניתן להשתמש בו גם ללא תהליך זה.

**יתרונות וחסרונות של Gaussian Process Regression (GPR)**

יתרונות:

* מדידת אי-ודאות: האלגוריתם לא רק מספק חיזוי אלא גם טווח ביטחון, מה שמאפשר להעריך את רמת הביטחון בתחזיות.
* גמישות גבוהה: יכול להתאים כמעט לכל פונקציה באמצעות Kernels שונים, מה שמאפשר עבודה עם מבנים נתונים מורכבים.
* התאמה אישית עם Kernels: ניתן לבחור או לשלב Kernels שונים כדי לשפר את הדיוק ולהתאים את המודל לבעיה.
* למידה אקטיבית: המודל מתעדכן ומשתפר ככל שנוספים לו נתונים חדשים, תכונה שימושית באופטימיזציה בייזיאנית.
* חיזוי בנתונים חסרים: האלגוריתם יכול לבצע חיזוי גם כאשר חסרים נתונים מסוימים, על בסיס תלות בין הנקודות.
* אומדן אי-ודאות: מאפשר זיהוי תחומים שבהם המודל פחות בטוח, שימושי בתחומים קריטיים כמו רפואה ופיננסים.

חסרונות:

* זמן חישוב ארוך: בעל מורכבות חישובית O(n³), ולכן אינו יעיל עבור מערכי נתונים גדולים במיוחד.
* קושי בפרשנות: קשה להסביר את הקשרים בין הנתונים בהשוואה למודלים פרמטריים פשוטים, והצלחתו תלויה בבחירת Kernel נכונה.
* Overfitting: ללא בחירה נכונה של Kernel והגבלות מתאימות, המודל עלול להתאים את עצמו לרעש בנתונים ולהוביל לתחזיות שגויות.
* בעייתיות בנתונים עם שינויים חדים: GPR מתקשה להתמודד עם שינויים פתאומיים, מאחר שהוא מתבסס על פונקציות חלקות כמו RBF.
* מגבלות חישוביות בנתונים מרובי ממדים: ככל שמספר המדדים (Features) גדל, חישוב מטריצת הקו-וריאנס הופך למסובך יותר, מה שמקשה על עבודה עם דאטה רחב מאוד.

פתרונות אפשריים לבעיות חישוביות:

* סינון מאפיינים לא רלוונטיים כדי לצמצם את מורכבות הנתונים.
* שימוש ב-Kernels פשוטים יותר כדי להקטין את עומס החישוב.
* בחירה נכונה של length-scale כדי לאזן בין גמישות המודל לבין מניעת Overfitting.

**שלבי עבודה במחברת:**

כשמשתמשים בקרנלים במודלים כמו Gaussian Process Regression (GPR), צריך לחשב מטריצת קו-וריאנס (Covariance Matrix), שמתארת את הקשרים בין כל הזוגות של הנתונים.

הבעיה היא שאם יש 500,000 דוגמאות, מטריצת הקרנלים תהיה בגודל 500,000 × 500,000, וזה דורש חישובים כבדים מאוד של O(n³) (זמן חישוב אקספוננציאלי). כלומר - חישוב כזה נהיה כבד מאוד חישובית עבור מערכי נתונים גדולים. ולכן קודם כל חשוב לציין שלא עבדנו באופן ישיר עם GPR בגלל סיבוכיות חישוב גבוהה שלו על דאטה גדול - O(n³).

החלטנו להשתמש במקום בקירוב קרנלים (Nystroem Kernel Approximation) שמאפשר להפוך את החישוב לאפשרי על דאטה גדול מבלי לוותר על היתרונות של שימוש בקרנלים.

כלומר, במקום לחשב את כל המטריצה הפתרון שעשינו הוא להשתמש בקירוב קרנלים. בשיטה זו - במקום לחשב את כל המטריצה מגדירים בעזרת הפרמטר n\_components נקבע כמה תכונות ישמרו מתוך המטריצה המקורית .

ככל שהערך גבוה יותר, הייצוג של הקרנל יותר מדויק אבל החישוב יותר כבד. ולכן המטרה היא למצוא את האיזון- מספיק n\_components כדי לייצג נכון את הנתונים אבל לא יותר מדי כדי להימנע מחישוביות גבוהה מדי. במקרה שלנו השתמשנו ב7000.

הקשר בין Ridge Regression לקירוב קרנלים (Nystroem Kernel Approximation) הוא בכך שלאחר קירוב הקרנלים, מתקבלת מערכת חדשה של תכונות (Features) שמייצגות את הקרנל המקורי אך עם כמות מופחתת של חישובים. עם זאת, כיוון שהנתונים מועברים למרחב חדש שבו עשויים להיווצר הרבה מאוד משתנים חדשים (עד n\_components), מודל רגרסיה רגיל עלול לסבול מ-Overfitting בשל כמות התכונות הגבוהה. כדי להתמודד עם זה, נעשה שימוש ברגרסיית Ridge (רגולריזציה L2), שמאפשרת שליטה על משקלי המשתנים החדשים, מונעת התאמה מוגזמת לרעש בנתונים, ושומרת על מודל גמיש אך יציב.

בשלב הראשון בקוד- בחירת גישה אלגוריתמית:

כפי שהסברנו קודם בגלל גודל מערך הנתונים (500,000+ טיסות), Gaussian Process Regression (GPR) לא היה ישים עקב סיבוכיות חישובית גבוהה, המחייבת חישוב מטריצת קרנל ענקית. כדי להתגבר על כך, השתמשנו בקירוב קרנלים באמצעות Nystroem Kernel Approximation, שמאפשר לחשב קרנלים על תת-מדגם קטן יותר במקום על כל הנתונים, תוך שימור הדפוסים המרכזיים.

לאחר קירוב הקרנלים, נדרשנו להשתמש במודל שיבצע רגרסיה על הנתונים החדשים שנוצרו. מכיוון שכמות התכונות גדלה בעקבות קירוב הקרנלים, ישנו סיכון ל-Overfitting. כדי להתמודד עם זה, השתמשנו ב-Ridge Regression, המספק רגולריזציה (L2) ומונע שהמודל יתאים את עצמו לרעש בנתונים.

שלב שני – נרמול הנתונים:

בשל השילוב בין תכונות מספריות רציפות (כגון משך טיסה ויום יציאה) לבין תכונות קטגוריות (כגון חברות תעופה שהומרו ל-One-Hot Encoding), בוצע נרמול רק על התכונות המספריות כדי למנוע השפעה לא מאוזנת על המודל. לשם כך, השתמשנו ב-StandardScaler(), המבצע נרמול סטטיסטי (Z-score normalization), כך שכל משתנה מספרי מתוקנן לממוצע 0 וסטיית תקן 1. נרמול זה חיוני במיוחד בשיטות מבוססות קרנלים (Kernel-based methods), כמו GPR, שכן סקיילינג נכון של המשתנים משפיע ישירות על אופן חישוב הקרנלים ועל דיוק המודל.

שלב שלוש - אימון, Fine-Tuning וניתוח ביצועי המודל:

לאחר שהגדרנו את מבנה המודל, שלב האימון מתבצע על סט האימון (X\_train\_scaled).

מכיוון שהמודל הוכשר על נתונים מנורמלים, התחזיות (y\_pred\_scaled) מתקבלות בסקייל של StandardScaler(), ולכן יש להמיר אותן חזרה לטווח המחירים המקורי (NIS) באמצעות inverse\_transform().

בנוסף, ביצענו כוונון היפר פרמטרים (GridSearchCV) כדי למצוא את השילוב האופטימלי של פרמטרי Nystroem Kernel Approximation ו-Ridge Regression. בתהליך זה:

* נבדקו קרנלים שונים (rbf, poly, sigmoid, laplacian), מספר רכיבים (n\_components), דרגות פולינום (degree), ורמות רגולריזציה (alpha).
* לאחר מציאת הפרמטרים האופטימליים, המודל הוכשר מחדש עם הנתונים המלאים, בוצע חיזוי, הוחזרו המחירים לטווח המקורי (inverse\_transform), והמודל הוערך באמצעות מדדים סטנדרטיים.

לאחר כוונון היפר פרמטרים מוצלח, נבחרו הפרמטרים האופטימליים: קרנל Laplacian, דרגה 3, n\_components=500, ו- ridge\_\_alpha=0.01. כתוצאה מכך, הביצועים השתפרו משמעותית, עם דיוק גבוה יותר וצמצום השגיאות.

שלב ארבע – שיפור המודל ע"י הגדלת n\_components וכוונון רגולריזציה

בשלב זה הגדלנו את הn\_components ל-7000 והצלחנו לשפר את ביצועי המודל. עם עלייה ב-R² ל-0.8542 וירידה ניכרת במדדי השגיאה (MSE, RMSE, MAE). מאחר שהמחשב אינו יכול להתמודד עם n\_components=10000, זהו המקסימום שנוכל לבדוק מבחינת גודל הייצוג של הקרנל.

בנוסף, בשלב זה בוצע כוונון של ridge\_\_alpha כדי למצוא את רמת הרגולריזציה האופטימלית למודל, כך שישיג איזון בין דיוק לבין Generalization. נבדקו ערכים שונים, כאשר עבור כל ערך המודל אומן מחדש ונבדק על סט הבדיקה, המטרה הייתה לזהות את הערך שממזער שגיאות ומגדיל את יכולת ההסבר של המודל. נמצא כי α = 0.05 מספק את התוצאות הטובות ביותר.

לסיכום: המודל שבנינו, המבוסס על Nystroem Kernel Approximation + Ridge Regression, סיפק חיזוי מחירי טיסות מדויק ויעיל, עם שגיאות נמוכות.

בעזרת שימוש בקרנל Laplacian אפשר ללכוד קשרים לא-לינאריים, בעוד שרגרסיית Ridge עם α=0.05 מנעה Overfitting ושיפרה את יציבות המודל. התוצאות מראות איזון טוב בין דיוק לבין הכללה, ללא הטיה משמעותית, אך ייתכן שיפור נוסף על ידי כוונון נוסף של קרנלים, רגולריזציה או הוספת מאפיינים חדשים.