גלאי חריגות פשוט ל time-series data
אבן דרך 2

הקדמה

סטודנטים יקרים שלום רב,

בפרויקט זה נממש אלגוריתם פשוט לגילוי חריגות ב time-series data. המטרה היא להגיע לתוצר הנדרש תוך כדי שמירה על עקרונות התכנות הפונקציונאלי.

במסמך זה נתאר את אבן הדרך השנייה.

ההגשה היא לבודדים במערכת הבדיקות האוטומטית בקורס FP ומטלה ex2.

בנוסף לקובצי המקור שהורדתם ממערכת הבדיקות, קיבלתם גם קובצי csv. שימרו אותם בתיקיית הפרוייקט (ולא בטעות ב src).

עליכם להגיש את קובצי ה scala הבאים:

Util, Line, TimeSeries, ZAnomalyDetector, SumSqrAnomalyDetector, LinearRegAnomalyDetector, HybridAnomalyDetector

תאריך ההגשה הוא 02.05.21

שימו לב:

בקורס זה יש רק 3 אבני דרך להגשה עם בדיקה אוטומטית, ללא בחינה או הגנה. כאשר ניתן להגיש ללא הגבלה קל לקבל 100 והערך של ציון זה פוחת. לכן מעתה מס' ההגשות במוד הגשה יוגבל. אתם כבר בשנה ג' וזו שפת JVM שעבורה מערכת ההפעלה לא יוצרת הבדלים, ובנוסף מוד ההגשה דומה עד מאד למוד האימון ולכן זו לא צריכה להיות בעיה.

אני כן מצפה שאם לא הצלחתם בהגשה במוד-הגשה אז תוסיפו בעצמכם בדיקות למוד האימון הלוקאלי שקבלתם כדי לאבחן את הבעיה, במקום להגיש אינספור פעמים למערכת במוד הגשה. בשיטה זו המאלצת אתכם במידת הצורך לכתוב בדיקות הלימוד יהיה לדעתי אפקטיבי יותר. אז שימו לב, מעתה:

- אין להגיש קוד עם **הדפסות דיבאג**, זה יגרור הפחתה של מלוא נקודות.
- במוד אימון תוכלו להגיש כמה פעמים שתרצו (עד לדדליין כמובן)
- אך במוד הגשה מותר לכם הפעם להגיש **רק 2 פעמים + פעם נוספת** למקרה של הגשה בטעות.
- כל הגשה במוד זה נחשבת, גם אם נפסלה בגלל שגיאת קומפילציה, ריצה או הדפסת דיבאג.

את אבן דרך 3 והאחרונה תצטרכו להגיש בשבוע האחרון לסמסטר.

בהצלחה!

Time Series

Time Series זו סדרה של נתונים שמשתנים על פני הזמן. מדובר במידע טבלאי בו כותרות העמודות מציינות את שמות המשתנים (features) שעבורם נאסף המידע. בכל שורה יופיעו הערכים של אותם features, כפי שהם משתנים לאורך זמן.

למשל אם קצב דגימת הנתונים הוא 10Hz אז כל 10 שורות מציינות מידע שנאסף במשך שנייה אחת. השורה ה 100 למשל מציינת את ערכי המשתנים בחלוף 10 שניות מתחילת הקלט ואילו השורה ה 101 תציין את הערכים בחלוף 10.1 שניות (10 + עשירית השנייה).

אנו אוספים נתונים בצורה זו עבור מערכות רבות ומגוונות, למשל נתוני טיסה של כלים בלתי מאוישים. לרוב נקבל את הקלט בצורה של קובצי CSV (ערכים המופרדים בפסיק) ונרצה למצוא חריגות באמצעות אלגוריתמים שונים.

בקובץ TimeSeries.scala עליכם להשלים טיפוס שמייצג Time Series כך שבהינתן (לבנאי) שם קובץ csv הסדרה תאפשר גישה לערכים ע"פ המתודות הנתונות. עליכם לבחור במבנה נתונים יעיל ומתאים לשם כך.

טיפ:

מתוך scala.io.Source תוכלו להשתמש בקוד הבא לקריאה של שורות מתוך קובץ:

```
val source=Source.fromFile(fileName)
source.getLines().foreach(line=>{

})
source.close()
```

תוכלו למצוא בדוקומנטציה דרכים קצרות יותר אולם בדרך זו אתם מקפידים על סגירת הקובץ.

משתנים שיש לממש:

- features – vector של String המכיל את שמות המאפיינים כפי שנלקחו מהשורה הראשונה בקובץ ה CSV.

המתודות שעליכם לממש:

- getValues – בהינתן שם המשתנה היא תחזיר ב O(1) אופציה ל Vector[Double] המכיל את ערכי המשתנה ב time series. אם שם המשתנה לא קיים ב time series אז יש להחזיר None.
- getValue – בהינתן שם המשתנה ואינדקס זמן (time step) היא תחזיר ב O(1) אופציה ל Double – ערך המשתנה בזמן זה או None אם יש בעיה בשם המשתנה או האינדקס.
- getValues – בהינתן שם משתנה וטווח זמן (Range) היא תחזיר כאופציה את ערכי המשתנה בטווחי הזמן או None אם שם המשתנה או ערכי הטווח אינם נכונים.

טיפ: אנו משתמשים ב Vector מפני שהוא Immutable. אך בעת הבנייה של אובייקט ה Time Series תוכלו להשתמש במבנה נתונים יעיל יותר..

אלגוריתם לגילוי חריגות

עבור האלגוריתמים הבאים כדאי להשתמש בחלק מהפונקציות שכתבתם ב Util.scala במטלה הקודמת.

בנוסף תוכלו להוסיף לשם פונקציות עזר נוספות.

זכרו את עקרון הפרדה. אם למשל אלגוריתם עושה שימוש בחישוב מסוים, אז כדאי שזו תהיה פונקציה כללית שהאלגוריתם עושה בה שימוש, ולא קוד מוטמע בתוך האלגוריתם. כך גם אלגוריתמים אחרים יוכלו להשתמש בפונקציה זו.

כעת נממש מספר אלגוריתמים מאד פשוטים לגילוי חריגות. אלגוריתמים אלו הם:

- אלגוריתמי **offline** – הם מקבלים את כל ה time series מתחילתה ועד סופה, ומוצאים היכן היו חריגות (בדיעבד). זאת בניגוד לאלגוריתמי Online שמוצאים חריגה בזמן שהיא מתרחשת.
- אלגוריתמים **לומדים**. הם מתבססים על קלט קודם עם דוגמאות תקינות.
  - הם מחזירים מודל שמשקף כיצד צפוי להיראות קלט תקין.
- לרוב, מבוססים על **קורלציה**. ההנחה היא שמשתנים קורלטיביים עשויים להעיד מה הערך הצפוי של האחד בהינתן הקלט של השני. המרחק מהצפי הזה מהווה מדד לגודל החריגה.

נתון לכם trait בשם AnomalyDetector שמגדיר את המתודות הבאות:

- המתודה learn אשר בהינתן TimeSeries של קלט תקין (ללא חריגות) היא תחזיר מודל כמפה מ String ל String שתשמש אותנו עבור מיפוי פרמטרים-ערכים שגלאי החריגות יעשה בהם שימוש.
  - לדוגמה ["thresholds", "0.9,0.85"].
  - השימוש במחרוזות נותן לכם גמישות לכתוב כל דבר שתרצו, איך שתרצו.

- המתודה detect אשר בהינתן מודל ו TimeSeries של קלט שעלול להכיל חריגות, היא תחזיר וקטור של זוגות Touple2[String,Int] – מחרוזת לתיאור החריגה והזמן (שורה) שבו היא התרחשה.

שימו לב שאם היה זה Object Oriented אז המודל היה נשמר באובייקט כ state ולא ניתן כפרמטר ל detect. אולם, אנו ממשים זאת בצורה פונקציונאלית.

את האלגוריתמים הבאים יש לממש כסוג של AnomalyDetector.

אלגוריתם Z

נתחיל באלגוריתם הפשוט ביותר.

בעת הלמידה אלגוריתם זה עובר על כל עמודה ב Time Series ומחשב את ערך ה z score בערך מוחלט של כל ערך x ביחס לשאר הערכים באותה העמודה (כולל את x), ושומר את המקסימום לכל עמודה. ערך זה מהווה סף שאם הוא נחצה נדווח על חריגה. תוכלו לשמור את המודל (המפה המוחזרת) באיזו דרך שתרצו.

בעת הגילוי הוא בודק לכל עמודה לכל ערך x האם ה zscore שלו בערך מוחלט גדול מהמקסימום שלמדנו עבור אותה העמודה. אם כן, אז תדווח חריגה כאשר ה String זה שם העמודה בה החריגה קרתה, וה Int זה מספר השורה.

** שימו לב, הספירה של השורות מתחילה מבחינתנו מ 0 לא כולל הכותרת. כך שאם החריגה קרתה בשורה 21 בקובץ ה csv יש לדווח 19.

עליכם לממש אלגוריתם זה כ object ZAnomalyDetector בקובץ ZAnomalyDetector.scala

אלגוריתם רגרסיה ליניארית (LinearRegAnomalyDetector)

בעת הלמידה אלגוריתם זה מחשב לכל משתנה (feature, עמודה) איזה משתנה (אחר) הוא הקורלטיבי ביותר אליו. לשם כך הוא ישתמש בקורלציית פירסון בערך מוחלט.

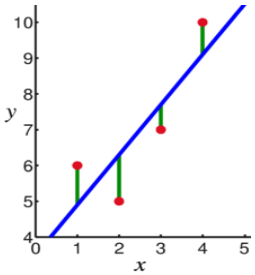
** שימו לב שאם חישבתם את הקורלציה בין A ל B אז אין טעם לחשב את הקורלציה בין B ל A שכן זה אותו הדבר. לכן אופן הסריקה הוא כבמטריצה משולשית עליונה – כל עמודה i יש להשוות לשאר העמודות מ j=i+1 ועד הסוף.

לדוגמה, אם יש לנו את העמודות A,B,C,D,E אז עבור A יש לחפש את הקורלציה המקסימאלית מבין B,C,D,E ואילו עבור C לדוגמה יש לחפש ב D,E בלבד. וכן, זה בסדר שלמשל A יהיה הכי קורלטיבי ל C ואילו ל C נמצא ש D יהיה הכי קורלטיבי אליו (למרות שהקורלציה ל A חזקה יותר).

כאשר הקורלציה בערך מוחלט גדולה או שווה ל 0.9 אז המשתנים ייחשבו כקורלטיביים. מבין המשתנים הקורלטיביים האלגוריתם מצמיד לכל משתנה את זה הקורלטיבי ביותר אליו.

כעת לכל זוג משתנים קורלטיביים האלגוריתם מחשב את קו הרגרסיה. כך, אם משתנה X קורלטיבי ל Y אז באמצעות קו הרגרסיה ניתן לחשב בהינתן קלט של ערך ל X היכן צפוי להיות הערך של Y.

אולם, הערך האמתי של Y לא יימצא בדיוק על קו הרגרסיה אלא במרחק מה ממנו. לכן בעת הלמידה לא נחשב רק את קו הרגרסיה לכל זוג משתנים קורלטיביים, אלא נחשב גם את המרחק המקסימלי ממנו שראינו בקלט התקין שישמש אותנו כסף. תוכלו לשמור את המודל (המפה המוחזרת) באיזו דרך שתרצו.



לדוגמה הביטו בתרשים מימין, המשתנה Y נתגלה כקורלטיבי ביותר ל X ולכן הם נבחרו כזוג. הקלט של הנקודות $(x \in X, y \in Y)$ מופיעות באדום. הן יצרו את קו הרגרסיה המופיע בכחול. לכל נקודה כזו יש מרחק מסוים מקו הרגרסיה; מרחק זה צבוע בירוק. זהו המרחק בערך מוחלט של ההפרש בין y לבין f(x) כש f היא משוואת קו הרגרסיה.

כעת בעת גילוי החריגות האלגוריתם מסתכל על כל זוג משתנים X,Y שזווגו בעת הלמידה. הוא מודד את המרחק בין כל נק' $(x \in X, y \in Y)$ לקו הרגרסיה שנלמד. אם המרחק הזה עולה על המרחק המקסימלי שנמדד עבור אותו הזוג בעת הלמידה אז תוכרז חריגה.

תיאור החריגה הוא שמות המשתנים מופרדים ע"י פסיק. סדר השמות הוא לפי סדר ההופעה בקובץ ה CSV, למשל "A,C" ולא "C,A".

אלגוריתם סכום מרחקים ריבועיים (SumSqrAnomalyDetector)

הגדרות:

- מרחק – יוגדר אצלנו כמרחק אוקלידי דו-ממדי:
  - $dist(p_1, p_2) = \sqrt[2]{(p_1.x - p_2.x)^2 + (p_1.y - p_2.y)^2}$
- בהינתן אוסף של נקודות P ניתן למדוד לכל נקודה $p_i \in P$ את סכום המרחקים הריבועיים לשאר הנקודות ב P.
  - $sqrSum(p_i, P) = \sum_{j \neq i} dist(p_i, p_j)^2$

תיאור האלגוריתם:

באופן **זהה** לאלגוריתם הקודם, בעת הלמידה אלגוריתם זה מצמיד זוגות משתנים שהם קורלטיביים ביותר. אולם, לכל זוג משתנים כאלו X,Y הפעם האלגוריתם מודד את סכום המרחקים הריבועיים שכל נקודה $(x \in X, y \in Y)$ יוצרת. הסכום המקסימאלי שנצפה עבור כל זוג X,Y מהווה סף עבור אותו הזוג.

תוכלו לשמור את המודל (המפה המוחזרת) באיזו דרך שתרצו.

בעת הגילוי האלגוריתם בוחן כל זוג משתנים שנתגלו כקורלטיביים ביותר בעת הלמידה. גם הפעם לכל נקודה שהם יוצרים יימדד סכום המרחקים הריבועיים, אלא שהפעם אם נוצר סכום שעולה על המקסימום שצפינו עבור אותו הזוג בעת הלמידה אז תוכרז חריגה.

** הקפידו שהחלק שזהה בין האלגוריתמים יופיע פעם אחת בקוד. בניגוד לתכנות מונחה עצמים ששם נשתמש בירושה, כאן זו יכולה להיות פשוט פונקציה כללית בתוך Util למשל, שתחזיר רשימה של Tuple3 (שלשה) עם שני האינדקסים הקורלטיביים וערך הקורלציה.

אלגוריתם היברידי (HybridAnomalyDetector)

- האלגוריתם של הרגרסיה הליניארית טוב בעיקר למקרים בהם יש קורלציה גובהה, שכן פירסון מודד קורלציה ליניארית ולכן ניתן לאפיין את התנהגות המשתנים ע"י קו לניארי.
- במקרה כזה האלגוריתם של סכום המרחקים הריבועיים יתנהג פחות טוב שכן נקודות שנמצאות בקצוות הקו עלולות להיחשב כחריגות. אולם, כאשר הקורלציה נמוכה ופיזור הנקודות יראה יותר כעיגול נפוח ופחות כקו, אז האלגוריתם הזה הופך לרלוונטי יותר; עם וריאציה מסוימת.
- כאשר ישנו משתנה שאינו קורלטיבי לאף אחד אחר אז האלגוריתמים לעיל כלל לא מסתכלים על משתנה זה, ולכן במקרה כזה ניתן להשתמש ב z score כברירת מחדל.

לכן ניצור את האלגוריתם ההיברידי הבא:

בעת הלמידה נמדוד את הקורלציה בין כל שני משתנים בדומה לאופן הסריקה בפעמים הקודמות. אלא שהפעם:

- אם הקורלציה בערך מוחלט שווה או גבוהה מ 0.9 האלגוריתם ילמד את קו הרגרסיה ואת המרחק המקסימלי ממנו.
- אם הקורלציה בערך מוחלט גבוהה מממש מ 0.5 אך קטנה ממש מ 0.9 האלגוריתם ימצא את הנקודה שעבורה סכום המרחקים הריבועיים הוא מינימלי. נק' זו תיחשב כמרכזו של מעגל. הנקודה הרחוקה ביותר ממנה (אוקלידית) תגדיר את רדיוס המעגל שייחשב כסף.
- אם לא נמצאה קורלציה גבוהה מ 0.5 אז המשתנה ייחשב כלא קורלטיבי והאלגוריתם ימדוד את ערך ה z score המקסימלי כסף עבור גילוי החריגות.

בעת הגילוי האלגוריתם יסתכל על המשתנים השונים בהתאם למה שלמד עליהם, ובהתאמה:

- עבור משתנים עם קורלציה גבוהה, אם המרחק מקו הרגרסיה עולה על המקסימום שנלמד תוכרז חריגה.
- עבור המשתנים עם קורלציה בינונית, תחושב (שוב) הנק' בעלת הערך המינימלי של סכום המרחקים הריבועיים (שתיחשב כמרכז מעגל). נמדוד את המרחק בינה לבין כל נקודה אחרת, ואם המרחק עולה על הרדיוס שלמדנו אז תוכרז חריגה.
- עבור משתנים לא קורלטיביים, תוכרז חריגה אם נצפה ערך z score גבוה מזה שנלמד

** ייתכן מצב למשל ש A ממש קורלטיבי ל B, ובסריקה של B מול כל השאר ל B לא תמצא קורלציה. לאלגוריתם זה לא באמת משנה, הוא יסתכל על הזוג A,B וגם על B כבודד עם z score.

בהצלחה!