**תרגיל מס' 2**

**שם מגיש:** מיכאל נוביצקי

**מס' ת''ז:** 311773915

**שם מגיש:** אלון קווארט

**מס' ת''ז:** 201025228

**חלק ראשון – Mandatory assignment**

פיצול הדאטה:

פיצלנו את הדאטה לפי Stratified sampling על מנת שכל אחת מהקבוצות – אימון, ולידציה ומבחן, ייצגו באופן טוב יותר את הדאטה הכולל. פיצלנו את הדאטה ל-3x2 סטים עם הגדלים הבאים: train = 70%, validation = 15%, test = 15%.

הפיכת תכונות נומינאליות לנומריות:

הבחנו בין שלושה מקרים אפשריים:

1. תכונות בוליאניות – Looking\_at\_poles\_results, Married, Gender, Voting\_time, Financial\_agenda\_matters, התאמנו את הערכים 0 ו-1.
2. תכונות עם סדר מסויים – Will\_vote\_only\_large\_party, Age\_group, התאמנו את הערכים -1, 0, 1
3. תכונות ללא סדר – Most\_Important\_Issue, Main\_transportation, Occupation, מילאנו בעזרת שיטת Hot Spot.

מילוי מידע חסר – Imputation:

עבור מילוי התאים החסרים השתמשנו בשיטת Closest Fit על מנת למלא ערכים לפי השכן הכי קרוב בעזרת מתודה שכתבנו. את מילוי המידע החסר בכל סט(אימון, ולדיציה ומבחן) עשינו רק על סמך המידע שנמצא בסט האימון.

ניקוי רעש וערכים חריגים – Data Cleansing:

הפכנו ערכים שליליים לאותם ערכים בערך מוחלט בתכונות הבאות שבהם לא חשבנו שיש הגיון בערכים שליליים:

Avg\_monthly\_expanse\_when\_under\_21, Avg\_lottary\_expanse, Avg\_Satisfaction\_with\_previous\_vote

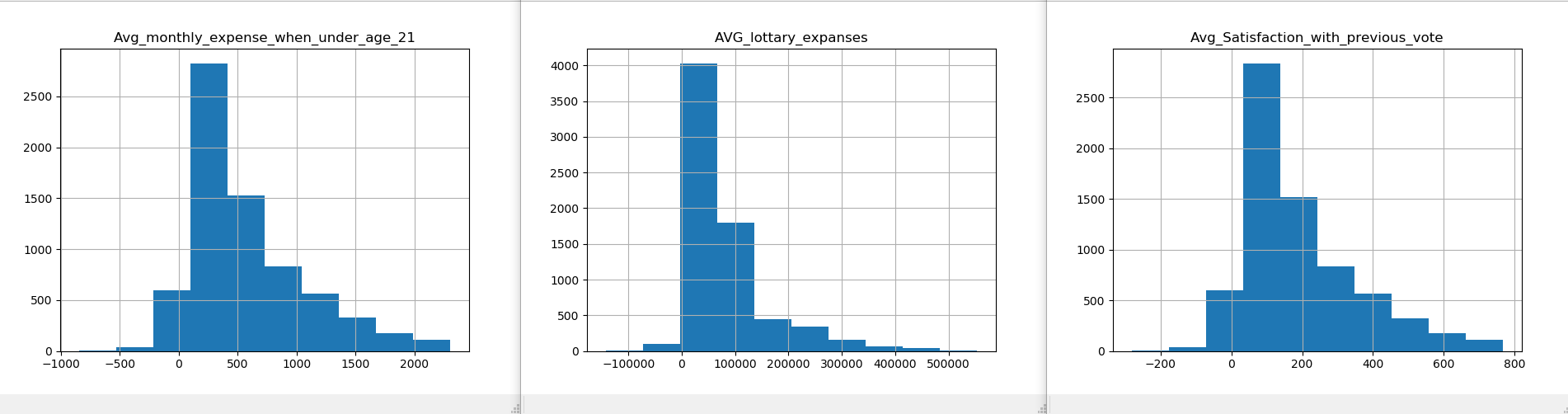
השווינו בין הקורלציה של התכונות עם תכונות אחרות לפני השינוי ואחרי השינוי והשינוי בקורלציה שקיבלנו היה מאוד נמוך ולכן השינוי כנראה לא משפיע כמעט ובכלל על הדטהסט.

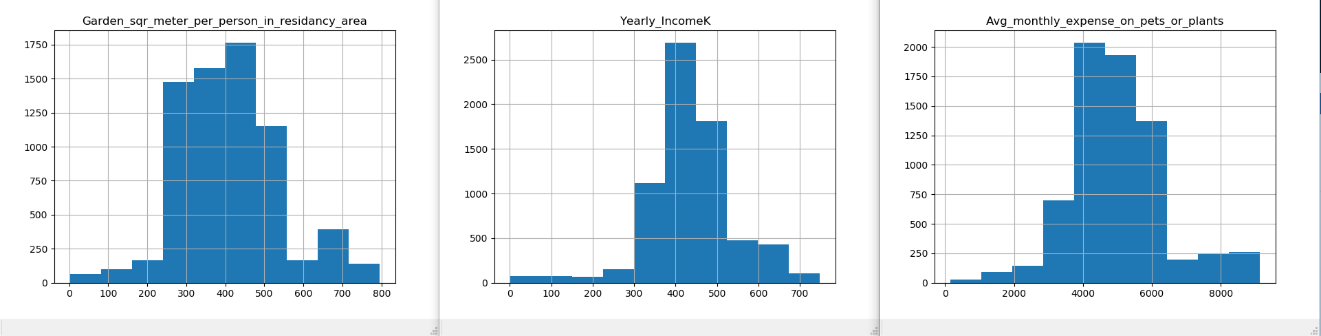
נורמליזציה – Scaling:

הבחנו בין שלושה מקרים אפשריים:

1. תכונה בוליאנית או תכונה טרינארית – אין צורך לנרמל.
2. תכונה שההיסטוגרמה שלה נראית גאוסית – הפכנו למשתנה גאוסי בעזרת Z-Score. רשימת התכונות הללו נמצאת בקובץ Consts.setGaussianGeatures.

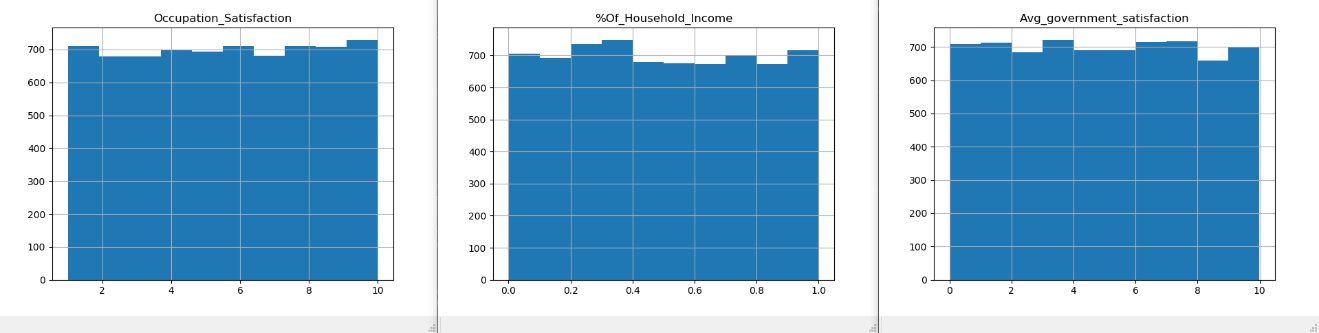
מספר דוגמאות לתכונות גאוסיות שבחרנו:

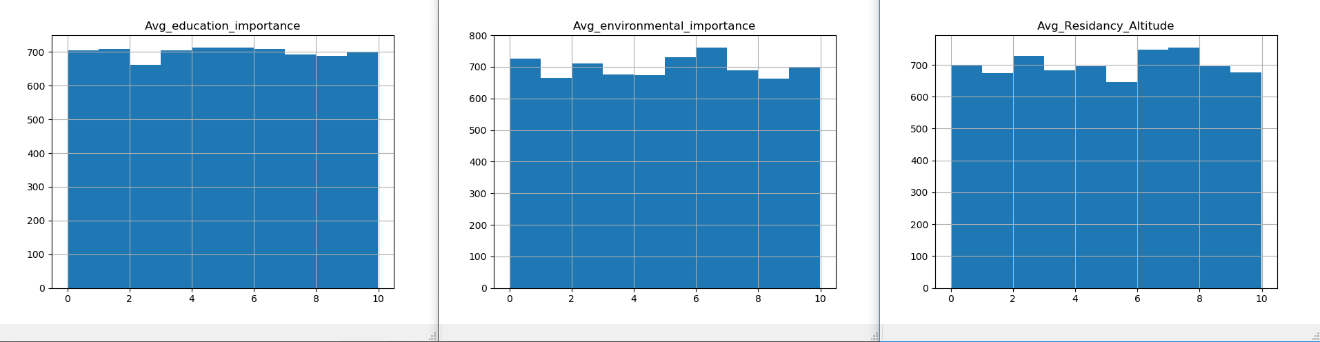




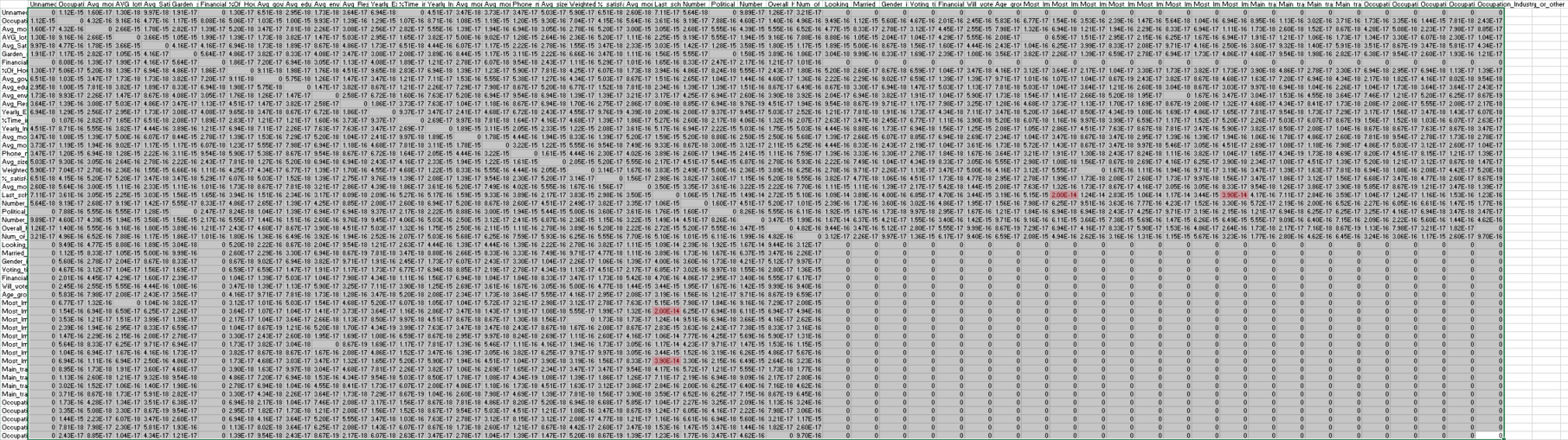
1. תכונה שההיסטוגרמה שלנה נראית אחידה – הפכנו למשתנה אחיד בתחום [-1,1] בעזרת Min-Max scaling. רשימת התכונות הללו נמצאת בקובץ Consts.setUniformFeatures.

מספר דוגמאות לתכונות עם התפלגות אחידה שבחרנו:





לאחר שסיימנו לנרמל את התכונות השווינו בין מט' הקורלציה לפני הנרמול ואחרי, כלומר החסרנו בין מטריצות הקורלציה ולקחנו ערך מוחלט לכל איבר. כפי שניתן לראות מהתוצאות, השינוי הוא קטן מאוד ולכן לא פגענו בתלויות הלינאריות בין התכונות.

**טבלת הפרש בערך מוחלט בין מט' הקורלציה לפני השינוי לבין מט' הקורלציה אחרי השינוי**

בטבלה ניתן לראות משבצות מודגשות בצבע אדום, כל ערך במשבצת כזאת גדול מ- וכל הערכים בטבלה קטנים מ- , כלומר ההפרש הכי גבוה חסום מלמעלה על ידי .

האלכסון שווה ל-0 מכיוון שהקורלציה של משתנה עם עצמו שווה ל-1 בכל אחת מהטבלאות ולכן ההפרש שווה ל-0.

ניתן לראות בחלק הימני התחתון בטבלה "בלוק" של משבצות, שהערך בכל אחת מהן שווה ל-0. זה בדיוק הבלוק של התכונות שמילאנו בשיטת ה- Hot Spot והערכים לא שונו בשלב ה-scaling ולכן ההפרש לפני ואחרי שווה בדיוק ל-0.

בחירת תכונות – Feature Selection:

לאחר שביצענו scaling לדאטה, בחרנו תכונות בשיטת filter ובשיטת wrapper:

* בשיטת filter השתמשנו במטריצת קורלציה על מנת להוציא מהדטה סט תכונות עם קורלציה הגבוהה מ- 0.95 (הוצאנו באקראי את אחת מהתכונות עבור כל זוג).
* מימשנו אלגוריתם SFS והשתמשנו בו על מנת לבחור תכונות בשיטת ה- wrapper. האלגוריתם נמצא בקובץ sfs.py (פירוט מורחב של האלגוריתם ניתן למצוא בחלק השלישי של התרגיל).

התכונות הנבחרות לאחר כל השלבים בחלק הראשון:

Will\_vote\_only\_large\_party\_int, Garden\_sqr\_meter\_per\_person\_in\_residancy\_area, Overall\_happiness\_score, Yearly\_IncomeK, Number\_of\_valued\_Kneset\_members, Last\_school\_grades, Avg\_monthly\_expense\_when\_under\_age\_21, Most\_Important\_Issue\_Education, Most\_Important\_Issue\_Environment

סה''כ נבחרו 8 תכונות (Most\_Important\_Issue פוצלה בשיטת Hot Spot) . זוהי לא רשימת התכונות הסופית שבחרנו, השתמשנו בשיטות נוספות שנפרט עליהם בהמשך.

**חלק שני – Additional Assignments (Bonus)**

A:

בחלק הראשון ביצענו את הכנת הדאטה על סט האימון, סט הולידציה וסט המבחן. על מנת להשלים ערכים חסרים בעזרת Closest Fit השתמשנו רק דאטה מסט האימון כפי שנכתב בפיאצה על ידי המדריך.

הקבצים לאחר הטרנספורמציה מוגשים יחד עם התרגיל. מכיוון שהשתמשנו ב-Closest Fit, המידע

B:

C:

D:

מימשנו את אלגוריתם Relief בתוך הקובץ relief.py המקבל את הפרמטרים הבאים:

X – ערכי התכונות ב- DataFrame

y- הסיווג המתאים ב- DataFrame

tau- מספר הפיצ'רים לבחירה

**חלק שלישי – SFS Implementation(Bonus)**

הסבר:

מימשנו אלגוריתם SFS שנמצא בקובץ sps.py המקבל את הפרמטרים הבאים:

Clf- המסווג שעליו מאמנים את הדאטה

X – ערכי התכונות ב- DataFrame

y- הסיווג המתאים ב- DataFrame

k- מספר הפיצ'רים לבחירה

בכל איטרציה האלגוריתם מעריך את הביצועים על ידי ביצוע cross\_validation עם 5 חלקים ומנסה למקסם את פונקציית ה-accuracy.

בנוסף, בכל איטרציה מודפסת רשימת התכונות שנבחרו באותה איטרציה ותוצאת ה-score.

הערך המוחזר הוא רשימת התכונות הנבחרות לפי הסדר שבו הם נבחרו.

יתרונות:

1. ניתן להשתמש באלגוריתם עם כל מודל למידה (מתקבל כפרמטר למתודה) ולכן אוניברסלי ופשוט.

חסרונות

1. זמן חישוב גבוה – לכל תת קבוצה שנבדקת נבנה, מאומן ונבדק מסווג.
2. לא ניתן להסיר תכונה שכבר נבחרה גם במקרה שהיא נהפכה להיות מיותרת לחלוטין כי ניתן רק להוסיף תכונות.
3. התת קבוצה הנבחרת מוטה לפי המסווג שנשלח לפונקציה ולכן, תת קבוצה שנבחרה עבור מסווג מסויים, לא בהכרח תתאים למסווג שונה.

פירוט על התכונות שנבחרו

**להשלים**