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:

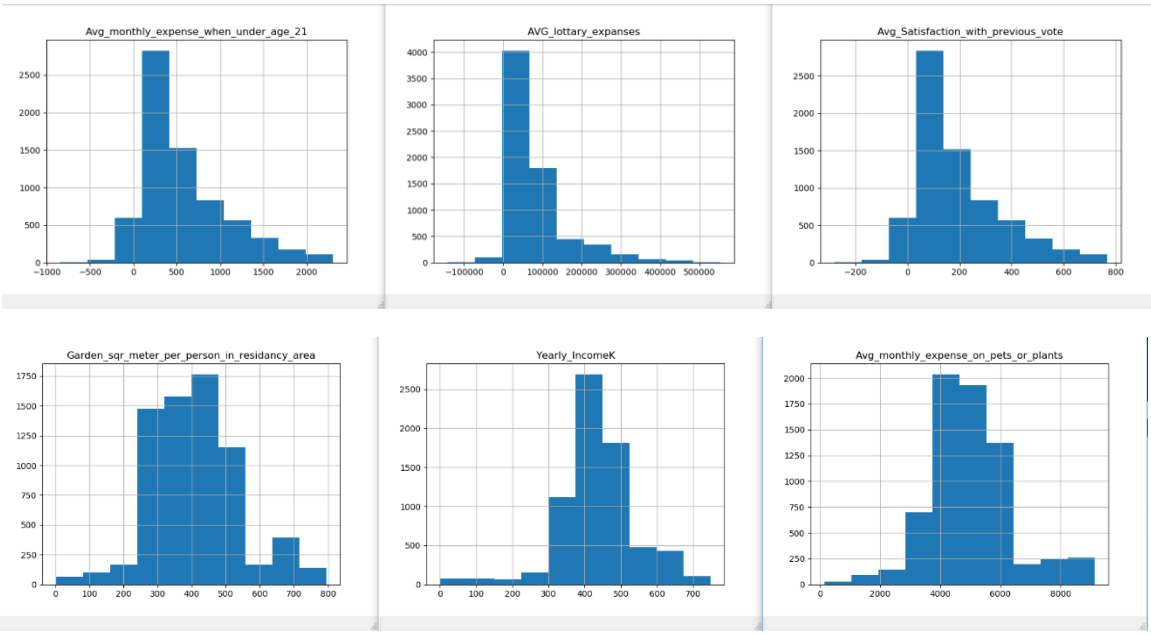
Avg_monthly_expanse_when_under_21, Avg_lottary_expanse, Avg_Satisfaction_with_previous_vote

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.

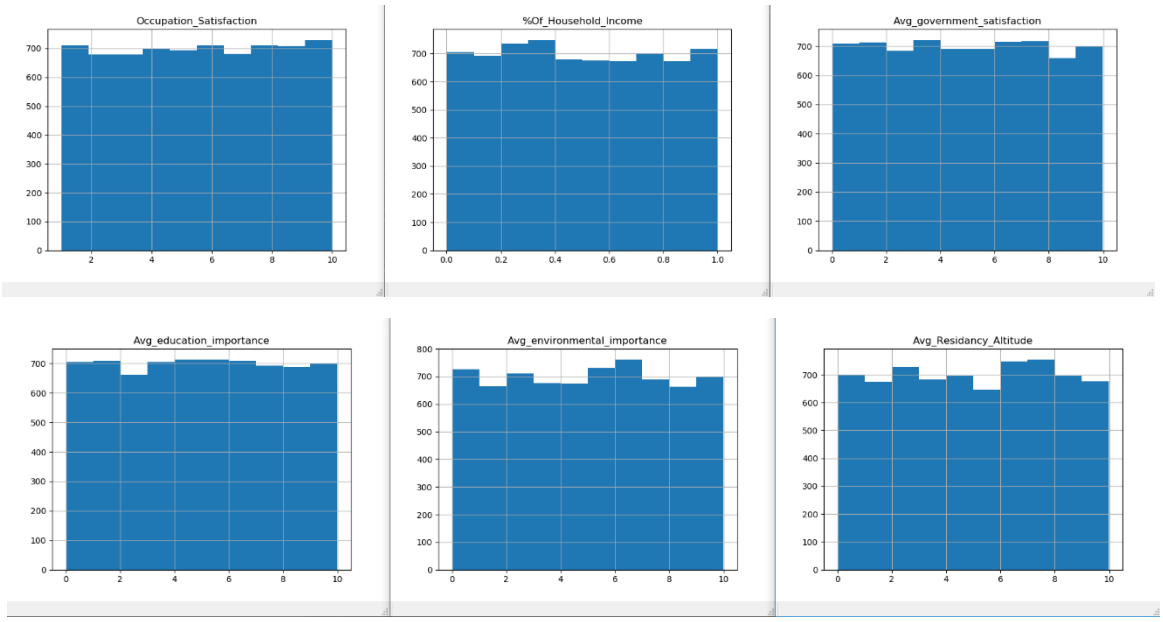
נורמליזציה – Scaling:

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:

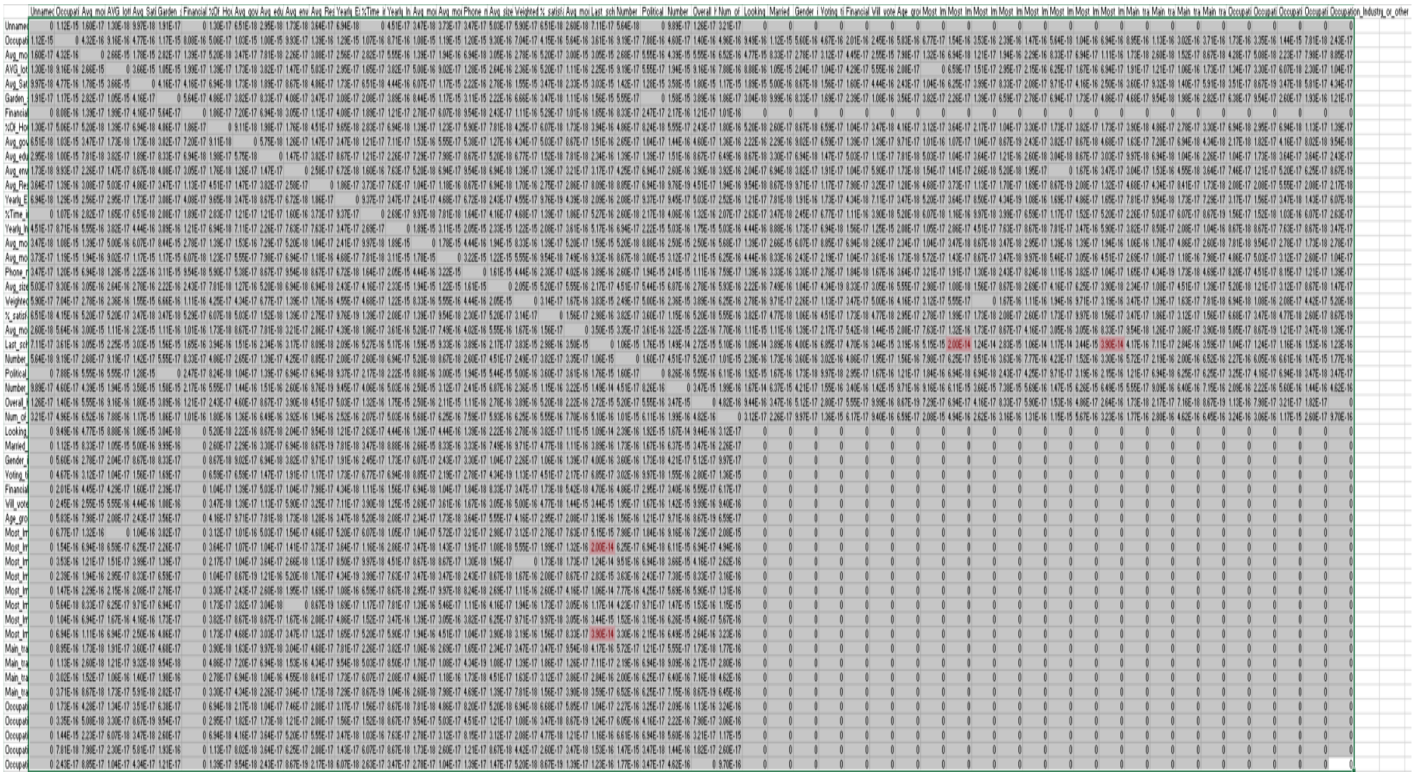


3. תכונה שההיסטוגרמה שלנה נראית אחידה – הפכנו למשתנה אחיד בתחום [-1,1] בעזרת Min-Max scaling. רשימת התכונות הללו נמצאת בקובץ Consts.setUniformFeatures.
מספר דוגמאות לתכונות עם התפלגות אחידה שבחרנו:



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-$1.95 \cdot 10^{-14}$ וכל הערכים בטבלה קטנים מ- $1.95 \cdot 10^{-13}$, כלומר ההפרש הכי גבוה חסום מלמעלה על ידי $1.95 \cdot 10^{-13}$.

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.

כפי שניתן לראות הקורלציה לפני ואחרי היא כמעט ללא שינוי ולכן לא נהרסה התלות הלינארית.

<u>בחירת תכונות – Feature Selection:</u>

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 (השתמשנו במסווג Random Forest):

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_lottary_expanses, Political_interest_Total_Score, Main_transportation_Public_or_other

סה"כ נבחרו 9 תכונות.

זוהי לא רשימת התכונות הסופית שבחרנו, השתמשנו בשיטות נוספות שנפרט עליהם בסוף הקובץ.

תחת הכותרת "בחירה סופית"

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.

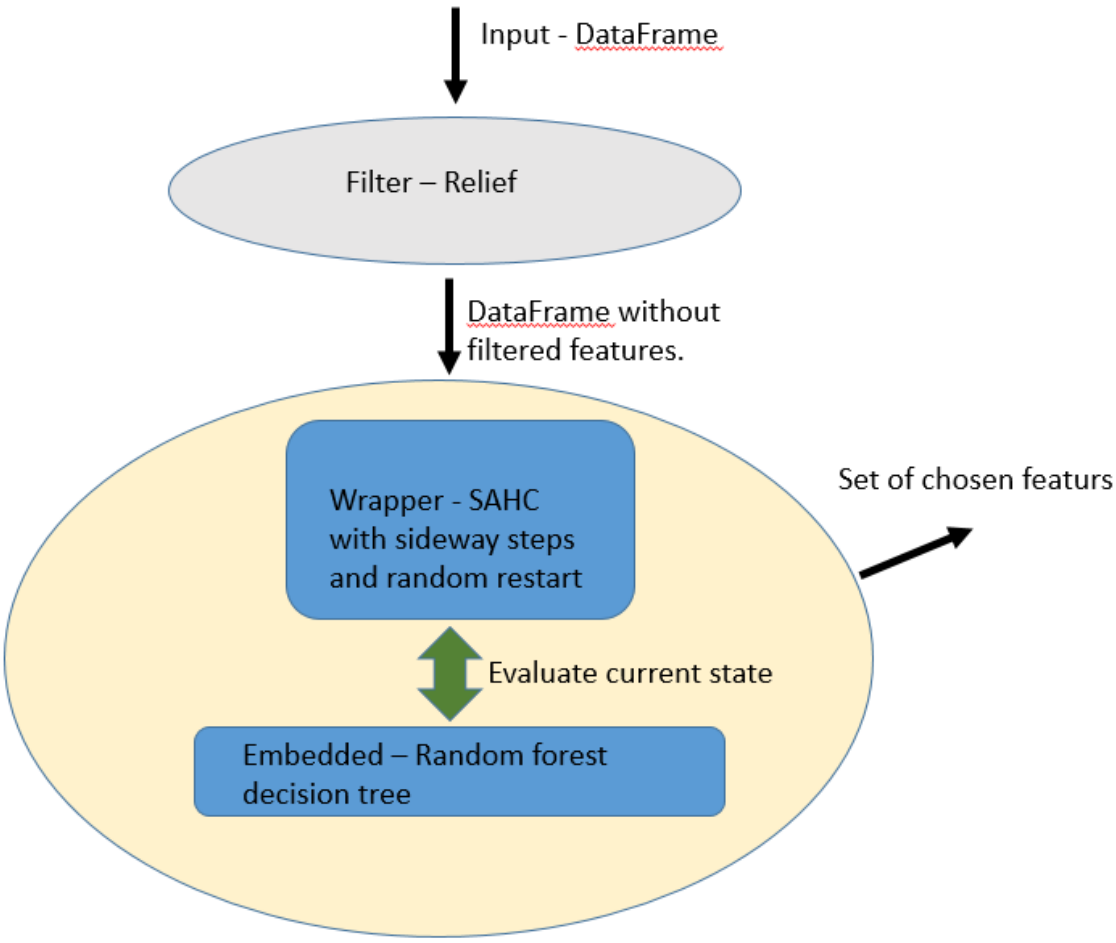
הקבצים לאחר הטרנספורמציה מוגשים יחד עם התרגיל. מכיוון שהשתמשנו ב-Closest Fit, הדאטה שהשתמשנו בו בחלק הראשון הוא רק הדאטה של האימון ולכן התוצאות זהות למבוקש בחלק זה.

B:

נתאר סכמה עבור בחירת תכונות:

1. בשלב הראשון נפלטר תכונות שאינן קונסיסטנטיות במרחב על ידי אלגוריתם Relief.
2. בשלב השני נבצע חיפוש SAHC with sideway steps and random restart. ניתן להגביל את כמות הפעמים שיבוצע random restart , לחילופין להגבילו בזמן או להפוך אותו ל-Anytime (האלגוריתם ימשיך בחיפוש עד לקבלת סיגנל חיצוני).
3. . נתאר את מרחב החיפוש:
   - קבוצת כל המצבים – כל תתי הקבוצות של תכונות.
   - מצב התחלתי – קב' מצבים אקראית כלשהי בגודל אקראי.
   - קבוצת המצבים הסופיים – יכול להיות כל מצב כי זאת בעיית אופטימיזציה.
   - פונקציית העוקב – הוספת איבר אחד בקבוצה הנוכחית.
   - פונקציית המחיר – ניתן לקבוע את המחיר על הצמתים או כיחס הפוך להפרש תועלת המצבים על קשת המחברת ביינהם. את התועלת בכל מצב נקבע על ידי הערכת ביצועי יער החלטה שייבנה על ידי התכונות במצב הנ"ל. (Embedded)
   במידה ויש שוויון, נבחר במצב עם מס' תכונות קטן יותר לפי עקרון התער של אוק'אם.

נתאר זאת בדיאגרמה:


Input - DataFrame
Filter – Relief
DataFrame without filtered features.
Wrapper - SAHC with sideway steps and random restart
Evaluate current state
Embedded – Random forest decision tree
Set of chosen featurs

:C

זיהוי קשרים בין תכונות לסיווג:

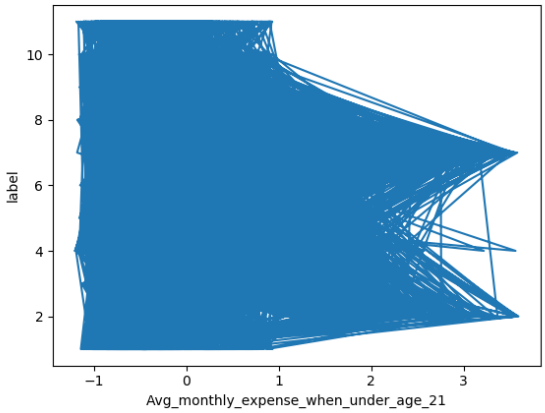
תחילה נציג את המיפוי שבו השתמשנו להמרת הסיווג למספר על מנת שהגרפים יהיו מובנים

Oranges : 11, Greens : 10, Pinks : 9, Purples : 8, Blues: 7, Whites : 6, Browns : 5, Yellows : 4

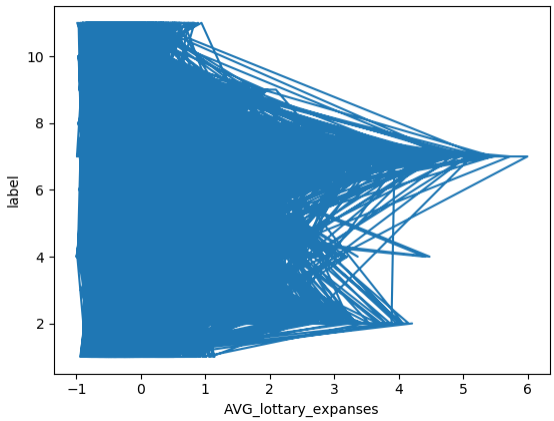
Reds : 3, Turquoises : 2, Greys : 1

כעת נפרט את המסקנות שלנו לגבי חלק מהתכונות שבדקנו:

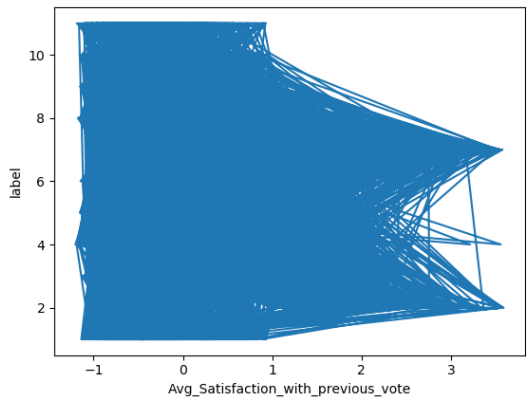
- אנשים מתחת לגיל 21 שמוציאים הכי הרבה כסף בחודש מצביעים בעיקר למפלגת הטורקיז ולמפלגת הכחולים.
- בקרב האנשים שמוציאים הכי קצת כסף בחודש ההצבעה אחידה בקירוב לכל המפלגות

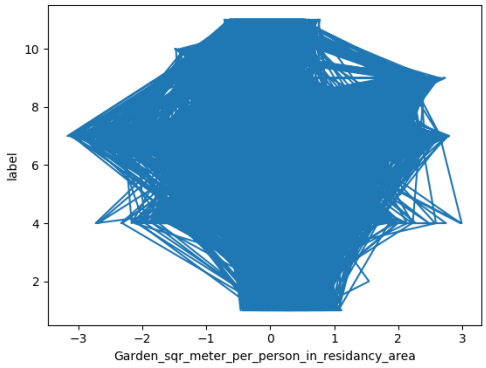


- אנשים שמוציאים הכי הרבה כסף על לוטו מצביעים בעיקר למפלגה הכחולה.
- בקרב האנשים שמוציאים הכי קצת כסף בחודש ההצבעה אחידה בקירוב לכל המפלגות

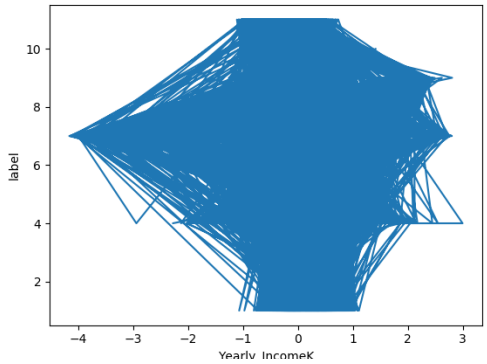


- האנשים שהכי מרוצים עם ההצבעה הקודמת שלהם מצביעים בעיקר לכחולים ולטורקיז

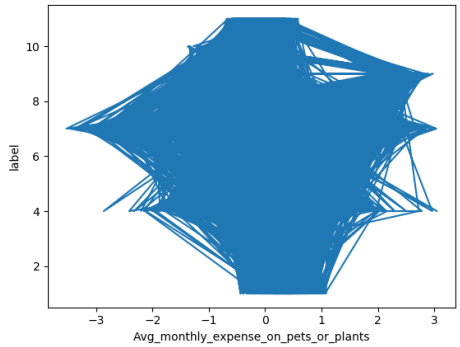




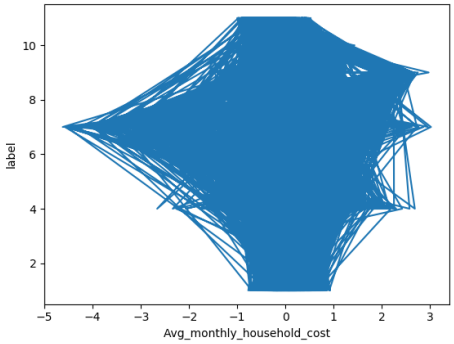
- האנשים שיש להם הכי פחות מ"ר מצביעים לכחולים
- האנשים שיש להם מ"ר בינוני מצביעים בצורה אחידה
- רוב האנשים שיש להם הרבה מ"ר מצביעים לרוב לורודים ולכחולים



- האנשים עם ההכנסה הכי נמוכה מצביעים לכחולים
- אנשים עם הכנסה בינונית מצביעים בצורה אחידה
- אנשים עם הכנסה גבוהה מצביעים לרוב לוורודים ולכחולים

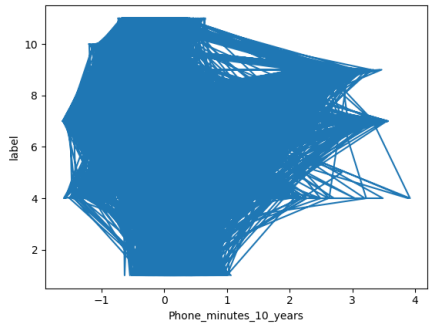


- המדד זהה כמעט לחלוטין למדד של ההכנסה השנתית שראינו בגרף הקודם.

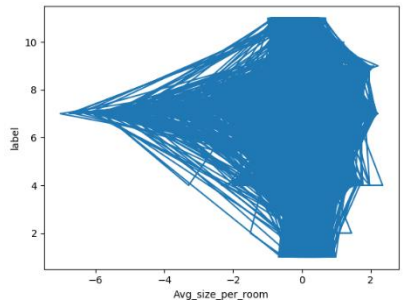


- אנשים שמוציאים הכי קצת על משק בית מצביעים לכחולים
- אנשים שמוציאים בצורה בינונית מצביעים בצורה אחידה
- אנשים שמוציאים הכי הרבה מצביעים לורודים או לכחולים

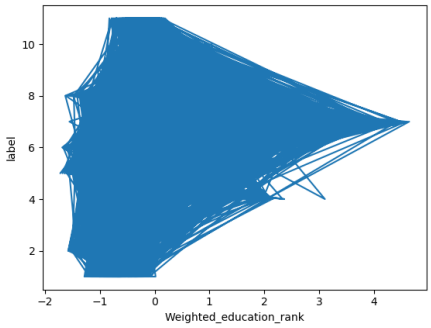
- אנשים שמדברים הכי הרבה בטלפון מצביעים לורודים או לכחולים



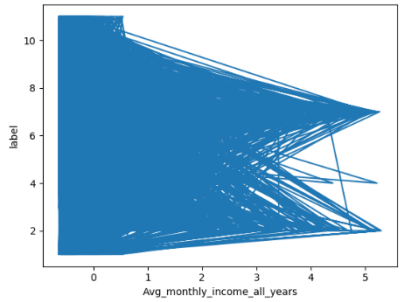
- אנשים שיש להם חדר קטן מצביעים לכחולים



- האנשים הכי משכילים מצביעים לכחולים



- המדד זהה כמעט לחלוטין למדד של האנשים שהכי מרוצים עם ההצבעה הקודמת שלהם



כפי שניתן לראות, חלק מהגרפים נראים מאוד דומים אחד לשני ולכן אפשר להניח שקיימת תלות בין התכונות שמוצגות בגרפים אלו. בנוסף, רוב הגרפים "המעניינים יותר" שניתן להסיק מהם מידע הם הגרפים של משתנים שהחלטנו שהם גאוסיים

:D

מימשנו את אלגוריתם Relief בתוך הקובץ relief.py. בקובץ זה ישנם 3 פונקציות:

- relief_nearst_miss:
  שגרה זאת מקבלת נקודה ומוצאת נקודה בעלת תיוג שונה, הקרובה ביותר לנקודת שניתנה.
- Relief_nearst_hit:
  שגרה זאת מקבלת נקודה ומוצאת נקודה בעלת תיוג זהה, הקרובה ביותר לנקודה שניתנה אך זרה לה.
- relief_alg:
  זאת השגרה המרכזית, שגרה זאת מקבלת את הפרמטרים הבאים:
  - X – ערכי התכונות ב- DataFrame
  - Y - הסיווג המתאים ב- DataFrame
  - N – מספר הדוגמאות שיבחרו להשתתף במישקול.
  - τ - ערך הסף אותו תכונה צריכה לעבור כדי להיחשב כרלוונטית.

  שגרה זאת מודדת מעיין אנטרופיה לכל תכונה, כלומר, אם נתבונן במישור הנוצר בין התכונה והתיוגים, שגרה זאת תיתן אמד ל"רעש" או מנגד להומוגניות של נקודות במרחב. ככל שמקבצי תיוגים זהים קרובים יותר, וככל שתיוגים זרים רחוקים יותר, ציון התכונה גדל.
  עבור $N = 100, \tau = 0$ נתקבלו התכונות הבאות:

0 Occupation_Satisfaction
1 Garden_sqr_meter_per_person_in_residancy_area
2 Yearly_IncomeK
3 Avg_monthly_expense_on_pets_or_plants
4 Avg_monthly_household_cost
5 Avg_size_per_room
6 Weighted_education_rank
7 Last_school_grades
8 Number_of_differnt_parties_voted_for
9 Political_interest_Total_Score
10 Number_of_valued_Kneset_members
11 Overall_happiness_score
12 Looking_at_poles_int
13 Married_int
14 Gender_int
15 Voting_time_int
16 Financial_agenda_matters_int
17 Will_vote_only_large_party_int
18 Age_group_int
19 Most_Important_Issue_Social
20 Most_Important_Issue_Education
21 Most_Important_Issue_Environment
22 Most_Important_Issue_Financial
23 Most_Important_Issue_Healthcare
24 Most_Important_Issue_Foreign_Affairs

25 Most_Important_Issue_Other
26 Most_Important_Issue_Military
27 Main_transportation_Car
28 Main_transportation_Public_or_other
29 Main_transportation_Foot_or_bicycle
30 Main_transportation_Motorcycle_or_truck
31 Occupation_Services_or_Retail
32 Occupation_Public_Sector
33 Occupation_Student_or_Unemployed
34 Occupation_Hightech
35 Occupation_Industry_or_other

בסך הכל עבור מדגם של 100 דוגמאות לכל תכונה נתקבלו 36 תכונות (כאשר חלקן נוצרו מפיצול תכונה חסרת סדר אחת מלכתחילה. למרות העובדה כי התוצאות היו יציבות עבור מדגמים בגדלים אלו, מרחב האוכלוסייה ממנה נלקח המדגם הינו בגודל 10,000 ולכן גדלים אלו אינם בהכרח מייצגים את אופי התכונה. בכדי לשפר את אופי האמד בצענו הרצה בעלת 1,000 דגימות לכל תכונה. בהרצה זאת קיבלנו את אותן התכונות בתוספת תוכנה אחת.

חלק שלישי – SFS Implementation(Bonus)

הסבר:

מימשנו אלגוריתם SFS שנמצא בקובץ sps.py ומכיל שתי פונקציות:

- sfsAux – הפונקציה שנקראת מבחוץ על ידי המשתמש ומקבלת את הפרמטרים הבאים:

Clf- המסווג שעליו מאמנים את הדאטה

X – ערכי התכונות ב- DataFrame

y- הסיווג המתאים ב- DataFrame

k- מספר הפיצ'רים לבחירה

הפונקציה בעצם קוראת למתודה הראשית sequential_forward_selection שמממשת את אלגוריתם ה-sfs עם הפרמטרים הנתונים ומחשבת את הערך שעבורו קיבלנו את התוצאה הטובה ביותר. הערך המוחזר הוא רשימת התכונות שעבורם התקבלה התוצאה הטובה ביותר, הערך של אורך הרשימה נע בין 1 ל-k.

- sequential_forward_selection – הפונקציה הראשית שמבצעת feature selection ועובדת באופן הבא:

בכל איטרציה האלגוריתם מעריך את הביצועים על ידי ביצוע cross_validation עם 3 חלקים ומנסה למקסם את פונקציית ה-accuracy ונבחרת התכונה שנתנה את הדיוק הכי גבוה.

הערך המוחזר הוא שלשה:

bestIndexes – רשימת האינדקסים של התכונות הטובות ביותר

bestFeatures – רשימת התכונות הטובות ביותר לפי סדר הבחירה

bestScores- מילון שהמפתחות בו הם מס' האיטרציה פחות 1 והערך הוא התוצאה עבור האיטרציה.

<u>יתרונות:</u>

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

<u>חסרונות</u>

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.

<u>פירוט על התכונות שנבחרו</u>

- רשימת התכונות הנבחרות לאחר כל השלבים בחלק הראשון עבור Random Forest:
- Will_vote_only_large_party_int, Garden_sqr_meter_per_person_in_residancy_area, Overall_happiness_score, Avg_size_per_room , Number_of_valued_Kneset_members, Last_school_grades, Avg_monthly_income_all_years, Political_interest_Total_Score, Main_transportation_Motorcycle_or_truck

סה"כ נבחרו 9 תכונות.

- רשימת התכונות הנבחרות לאחר כל השלבים בחלק הראשון עבור SVM:

- Garden_sqr_meter_per_person_in_residancy_area, Will_vote_only_large_party_int, Overall_happiness_score, Yearly_IncomeK, Number_of_valued_Kneset_members, Last_school_grades, Weighted_education_rank, Avg_monthly_expense_when_under_age_21, Most_Important_Issue_Foreign_Affairs,

Married_int, Avg_monthly_household_cost, Avg_monthly_expense_on_pets_or_plants, Phone_minutes_10_years

סה"כ נבחרו 13 תכונות.

קיבלנו שוני בגודל הסט וגם בחלק מהתכונות. יש סה"כ 5 תכונות משותפות בין שני המסווגים. ההבדל נובע מדרך הפעולה של המסווגים ומהעובדה ש-SFS מוטה לפי המסווג שנשלח אליו כפרמטר. למרות זאת, בחלק הראשון שמנו לב שלמשל שהגרפים של התכונות Avg_monthly_income_all_years ו- Avg_monthly_expense_when_under_age_21 דומים וניתן לראות שכל אחד מהמסווגים בחר אחת מהם ולכן למרות שאין הרבה תכונות משותפות, עדין יש דמיון בין חלק מהתכונות שייחודיות לכל אחד מהמסווגים.

<u>בחירה סופית:</u>

את הבחירה הסופית של התת קבוצה של התכונות עשינו על ידי ביצוע Relief על הדאטה לאחר שעבר imputation, data cleansing ו- scaling כפי שפירטנו בחלק הראשון. לאחר מכן, הורדנו את התכונות שיש ביניהם קורלציה שגדולה מ-0.95 ולבסוף, השתמשנו באלגוריתם SFS שכתבנו עבור מסווג SVC ומסווג Random Forest.

- רשימת התכונות הנבחרות לאחר כל השלבים בחלק הראשון עבור SVM:

- Garden_sqr_meter_per_person_in_residancy_area, Will_vote_only_large_party_int, Overall_happiness_score, Yearly_IncomeK, Number_of_valued_Kneset_members, Last_school_grades, Weighted_education_rank, Married_int, Most_Important_Issue_Foreign_Affairs, Avg_monthly_household_cost

סה"כ נבחרו 10 תכונות.

- רשימת התכונות הנבחרות לאחר כל השלבים בחלק הראשון עבור Random Forest:
- 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, Most_Important_Issue_Environment, Most_Important_Issue_Education, Most_Important_Issue_Financial

סה"כ נבחרו 9 תכונות.

בחרנו את הרשימת תכונות שקיבלנו כשהשתמשנו באלגוריתם SVC בגלל מספר סיבות:

1. ישנם 7 תכונות משותפות בין הסטים ולכן סט התכונות של SVC כמעט מכיל את סט התכונות של Random Forest.
2. איבוד תכונה רלוונטית הוא מאוד משמעותי ויכול לפגוע בשלבים הבאים.
3. ניתן להתגבר על תכונות עודפות בשלבים מתקדמים יותר.