הטכניון, מכון טכנולוגי לישראל

הפקולטה למדעי המחשב

מבוא למערכות לומדות

236756

סמסטר אביב תשע"ט

תרגיל מספר: 2

תאריך הגשה: 02/05/19

תא להחזרה: 68

מגישים:

עידו יחזקאל	8 6 3 7 9 3 4 0 2	idoye
שם מלא	מספר ת.ז.	דואר אלקטרוני ב- t2

אמיר אביבי	3 7 8 3 8 1 5 0 3	saavivi
שם מלא	מספר ת.ז.	דואר אלקטרוני ב- t2

:Mandatory Part – Data Preparation

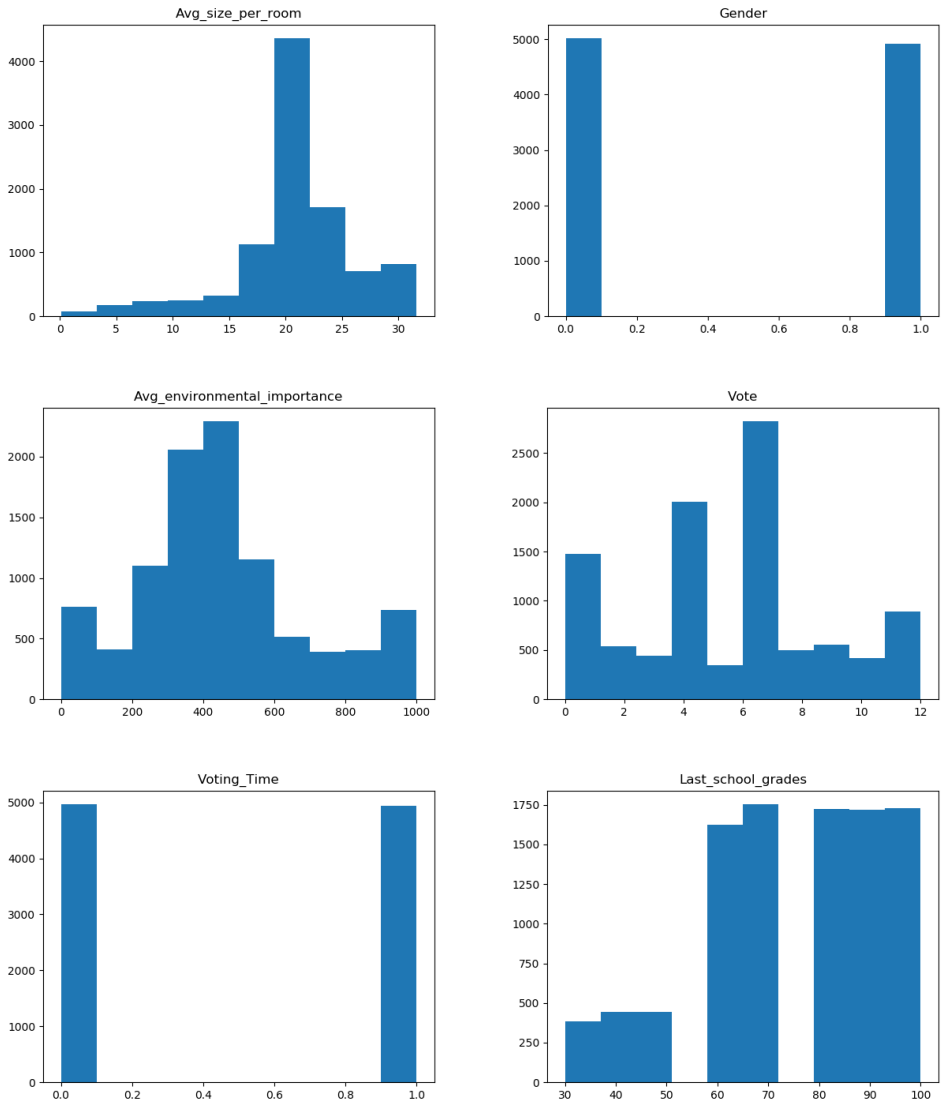
בחלק זה נתאר את הפעולות שביצעו על המידע הניתן לנו וזאת על מנת להבין אותו לקראת אלגוריתם למידה כלשהו.

ראשית נתאר את הפעולות שעזרו לנו להבין טוב יותר את המידע עליו אנחנו עובדים:

1. לאחר טעינת המידע ראשית הבנו מה הן התכונות המספריות ומה הן התכונות הנומינליות וקיבלנו את החלוקה הבאה:

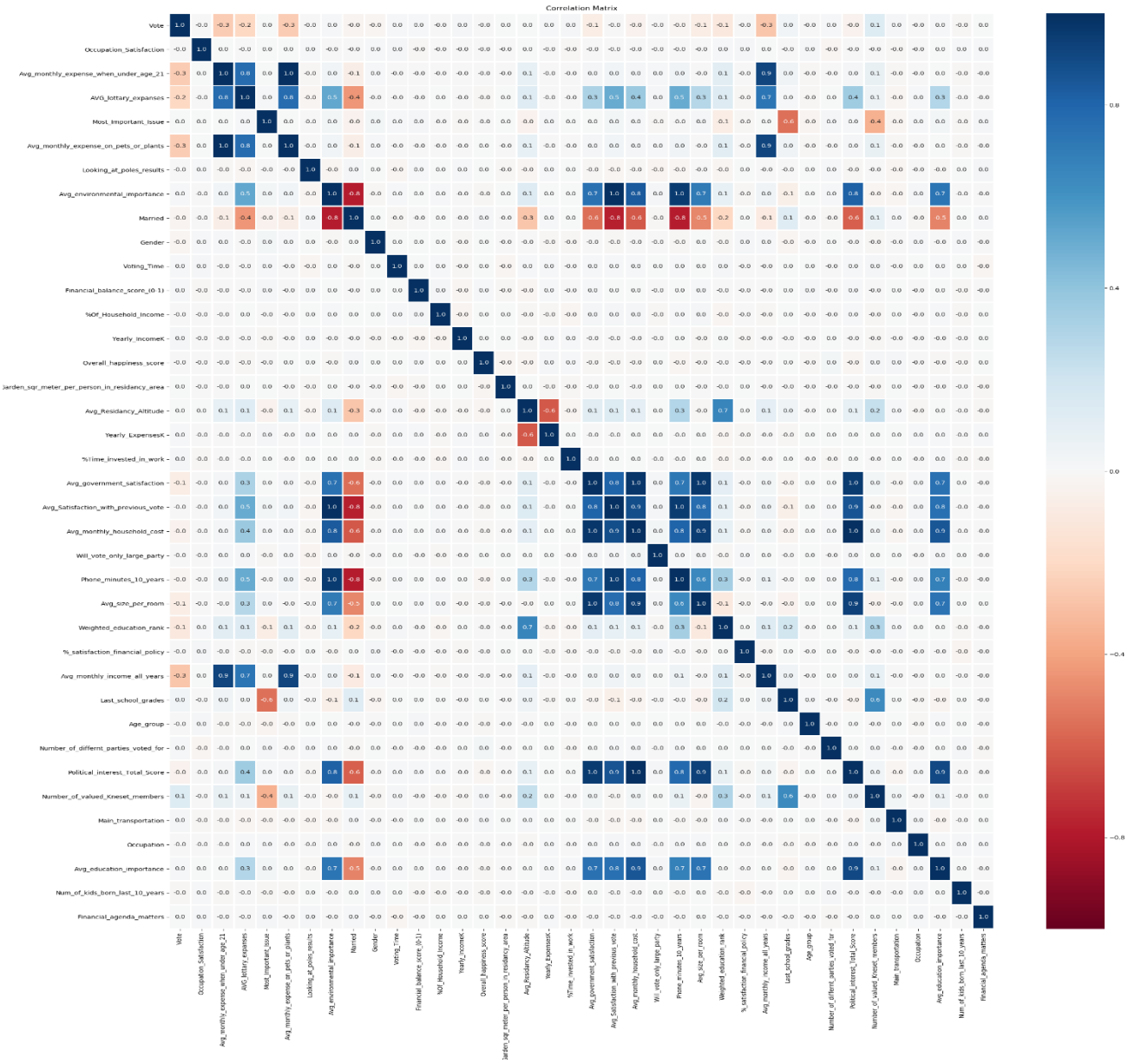
Numerical Features	**Nominal Features**
Occupation_Satisfaction	*Most_Important_Issue*
Avg_monthly_expense_when_under_age_21	*Looking_at_poles_results*
AVG_lottary_expanses	*Married*
Avg_Satisfaction_with_previous_vote	*Gender*
Garden_sqr_meter_per_person_in_residancy_area	*Voting_Time*
Financial_balance_score_(0-1)	*Will_vote_only_large_party*
Avg_government_satisfaction	*Age_group*
Avg_education_importance	*Main_transportation*
Avg_environmental_importance	*Occupation*
Yearly_ExpensesK	*Financial_agenda_matters*
Of_Household_Income%	*Vote*
Avg_Residancy_Altitude	
Yearly_ExpensesK	
Time_invested_in_work%	
Yearly_IncomeK	
Avg_monthly_expense_on_pets_or_plants	
Avg_monthly_household_cost	
Phone_minutes_10_years	
Avg_size_per_room	
Weighted_education_rank	
satisfaction_financial_policy_%	
Avg_monthly_income_all_years	
Last_school_grades	
Number_of_differnt_parties_voted_for	
Political_interest_Total_Score	
Number_of_valued_Kneset_members	
Overall_happiness_score	
Num_of_kids_born_last_10_years	

2. על מנת לטפל במידע באופן נכון יותר רצינו לקבל גם את ההתפלגות של כל אחת מהתכונות ולכן הפקנו עבור כל תכונה היסטוגרמה של ערכים על מנת להבין את ההתפלגות, להלן כמה היסטוגרמות שהפקנו:



גרפים אלו המחישו עברונו את ההתפלגות של כל תכונה ואילו תכונות מפולגות באופן אחיד ואילו מפולגות באופן דומה להתפלגות נורמלית.

3. בנוסף גם רצינו לדעת האם קיימת קורלציה לינארית בין התכונות לצורך השלמת המידע החסר ולכן הפקנו את מפת החום הבאה כאשר עבור כל שתי תכונות מצויינת הקורלציה ביניהן(כחול-קורלציה חיובית, אדום-קורלציה שלילית. ככל שהצבע קהה יותר הקורלציה מתקרבת ל1 בערך מוחלט):



4. לפי קריאת שמות התכונות הבנו כי אף תכונה לא צריכה להכיל ערך שלילי.

5. לאחר הבנת מאפייני המידע טוב יותר ניגשנו לבצע מניפולציות על המידע לשם הכנתו לצורך אלגוריתם למידה כלשהו:

1. טעינת קובץ המידע, זיהוי תכונות נומינליות ומיפוי ערכיהם אל מספרים שלמים, זאת על מנת לבצע מניפולציות בקלות יותר.
2. חלוקת המידע לשלושה קבצים נפרדים: train, validation and test ויצאו לקבצים לפני מניפולציות כלשהן.
3. השלב הבא הינו טיפול ב- outliers שכלל שתי מניפולציות עיקריות:
   - הסרת כל הערכים השליליים והמידע והחלפתם ב *nan*.
   - עבור כל התכונות שמתפלגות נורמלית, ביצוע סטנדרטיזציה לפי Z , והחלפת כל מידע אשר מעל z threshold שהינו 4.5 ב *nan*, המוטיבציה להחלפה זו נובעת מהמרחק הגדול של תכונות כאלו מהתפלגות התכונה.
4. כעת ניגשנו להשלמת המידע החסר, בצענו זאת על ידי כמה טכניקות כפי שלמדנו בביתה כאשר:
   - מכיוון שכבר למדנו את כיצד המידע שלנו מתנהג יכלנו להבין כי קיימות תכונות בעלות קורלציה לינארית גבוהה מאוד ביניהן ולכן יצרנו מילון עבור כל התכונות בעלי קורלציה מעל 0.9, חישבנו מה המקדם הלינארי האפקטיבי בין שני תכונות אלו והשלמנו את הערכים החסרים לפיו.
   - כעת פנינו לשיטה אחרת להשלמת ערכים חסרים והיא closest fit עבור כל אחד מהסטים של המידע (train, test, validation) יצרנו תת אוסף של מידע ללא חוסרים כלל והשלמנו שורות חסרות על סמך שורות קרובות ביותר לפי מדידת מרחק כפי שנלמד בביתה.
   - לאחר מכן ביצענו אלגוריתם Expectation Maximization על כל אחד מהסטים.
   - ולבסוף עבור כל אחד מהסטים השלמנו את הערכים החסרים בעזרת מניפולציה הסתברותית: עבור תכונה נומינלית הערך הושלם על פי הרוב ואילו עבור תכונה מספרית ממוצע.
   - וידאנו כי כל המידע שלם.
5. כעת ניגשנו לביצוע נרמול וסטנדרטיזציה למידע כאשר הידע המקדים על התכונות אפשר לנו לבצע זאת בצורה חכמה יותר:
   - עבור תכונות המתפלגות באופן אחיד ביצענו נרמול בטווח [-1,1] .
   - עבור תכונות התפלגות נורמלית ביצענו סטנדרטיזציה לפי Z.
   - עבור תכונות נומינליות לא התבצעה מניפולציה כלל.
6. בחירת התכונות הרלוונטיות התבצעה על פי מספר שיטות שונות:
   - Filter method- עבור התכונות המספריות הסרנו את התכונות אשר השונות (variance) שלהן קטנה מהסף 0.2 מתוך הנחה כי תכונות אלו לא תורמות הרבה מידע ולכן ניתן לוותר עליהן, לקחנו השראה מאלגוריתם PCA. התכונות שירדו הן:

- `Avg_government_satisfaction`
- `Avg_environmental_importance`

- Wrapper Method- על ידי שימוש ב SGDClassifier ו SelectKBest של sklearn הצלחנו למצוא את קבוצת התכונות בעלת mutual information מרבי ובעלת גודל לכל היותר של 17 אשר מניבה דיוק מרבי בעזרת אלגוריתם סיווג זה.
- קבוצת התכונות הנבחרת הינה:

- Avg_monthly_expense_when_under_age_21
- AVG_lottary_expanses
- Avg_Satisfaction_with_previous_vote
- Avg_education_importance
- Avg_Residancy_Altitude
- %Time_invested_in_work
- Avg_monthly_expense_on_pets_or_plants
- Avg_monthly_household_cost
- Phone_minutes_10_years
- Avg_size_per_room
- Weighted_education_rank
- Avg_monthly_income_all_years
- Last_school_grades
- Political_interest_Total_Score
- Number_of_valued_Kneset_members
- Most_Important_Issue
- Married

:First Bonus – Relief Algorithm

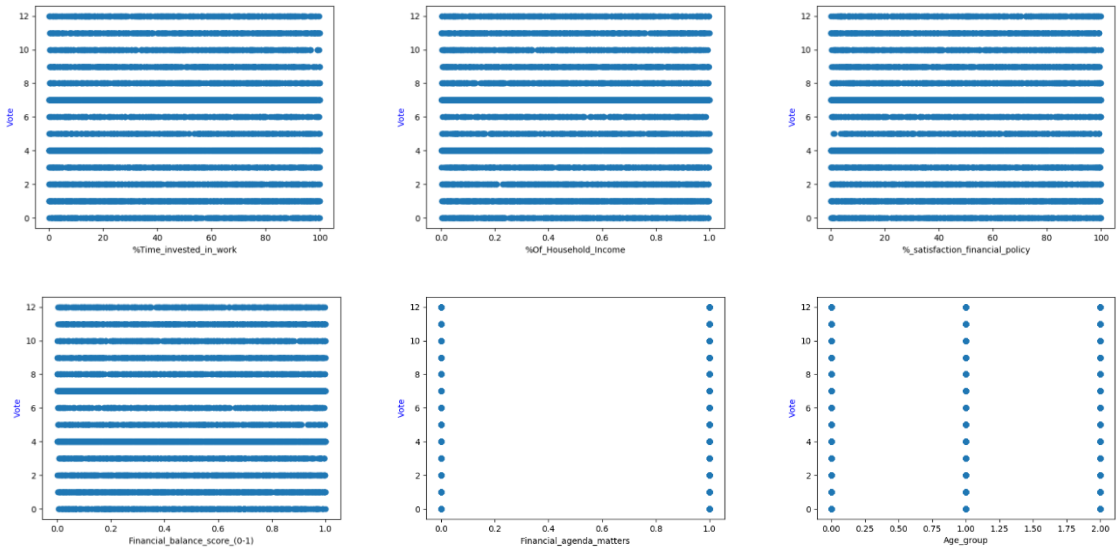
כל המימושים והמסקנות לבונוס זה מסתמכים על המאמר:

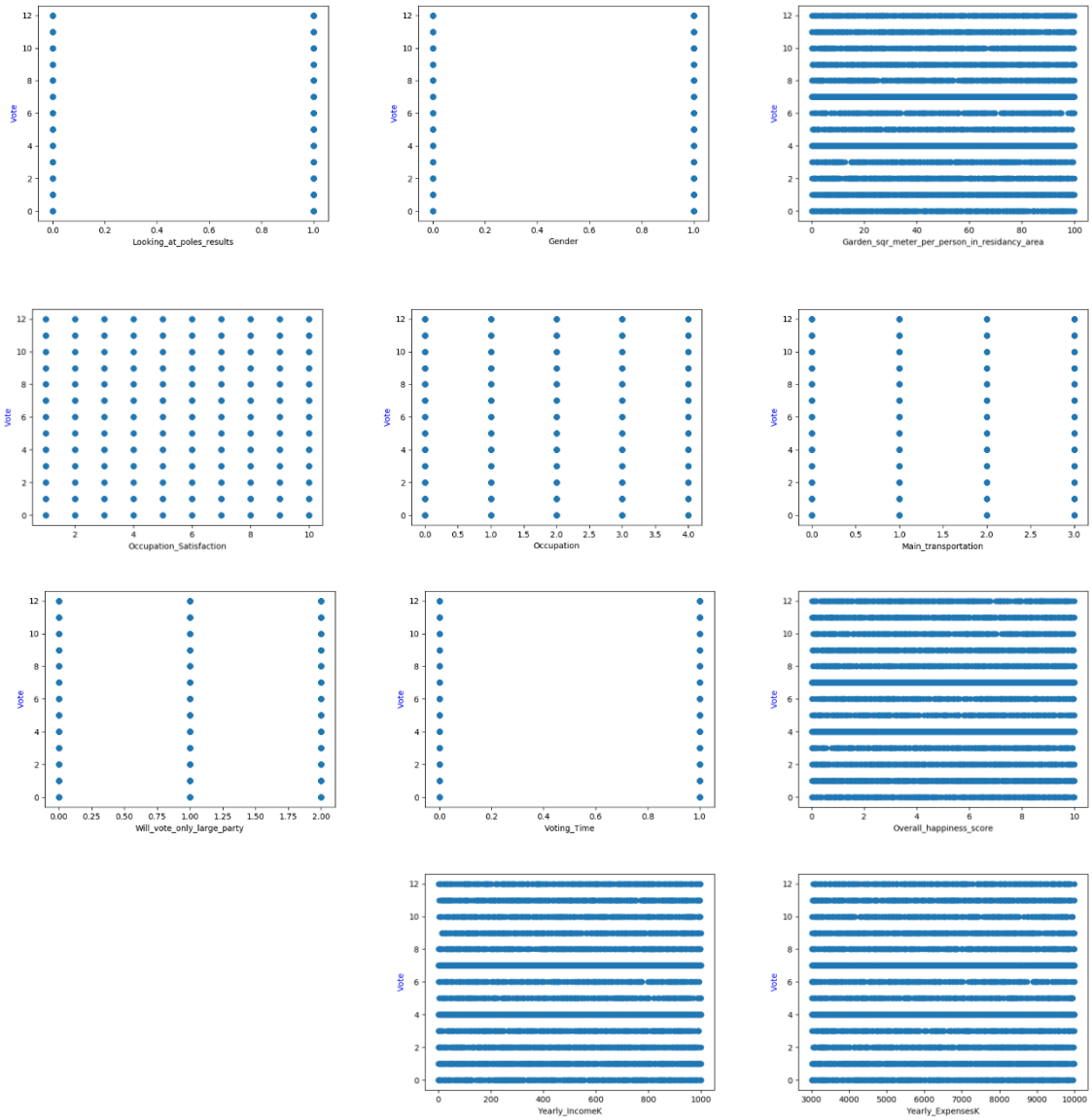
$Kira\&Rendell(1992) - A\ practical\ approach\ to\ feature\ selection$

1. זיהוי קשרים בין תכונות לבין סיווג, בהינתן כי הסיווג ממופה למספרים הבאים:

```
'Blues': 0,
'Browns': 1,
'Greens': 2,
'Greys': 3,
'Khakis': 4,
'Oranges': 5,
'Pinks': 6,
'Purples': 7,
'Reds': 8,
'Turquoises': 9,
'Violets': 10,
'Whites': 11,
'Yellows': 12
```

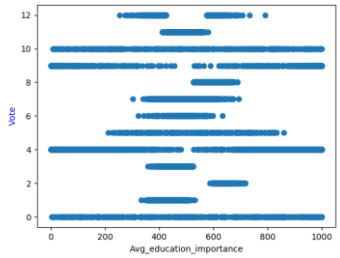
תכונות אשר לא הצלחנו לייחס להם קשר להן בעלי גרף המתפזר באופן אחיד על פני הסיווג לדוגמא:



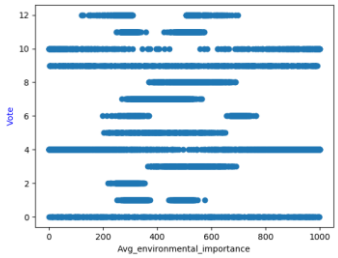


לעומת זאת לתכונות הבאות מצאנו קשרים לסיווג:

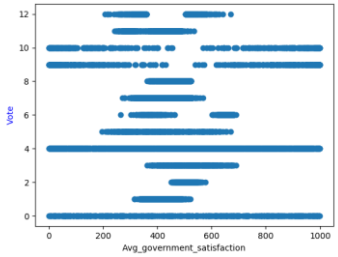
- Avg_education_importance- הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל כולם אמנם שאר הצבעים נפוצים רק אצל ערכים ממוצעים:



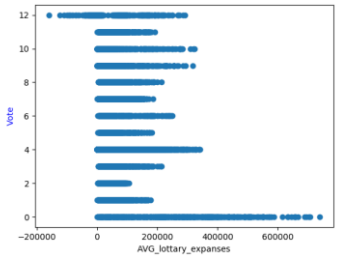
- Avg_environmental_importance- הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל כולם אמנם שאר הצבעים נפוצים רק אצל ערכים ממוצעים:



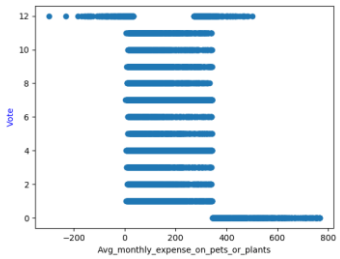
- Avg_government_satisfaction- הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל כולם אמנם שאר הצבעים נפוצים רק אצל ערכים ממוצעים:



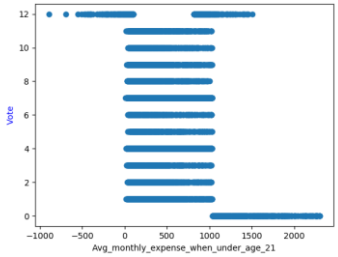
- AVG_lottary_expanses – בעלי הוצאה גבוהה נוטים לבחור כחול ובעלי הוצאה נמוכה נוטים לבחור צהוב:



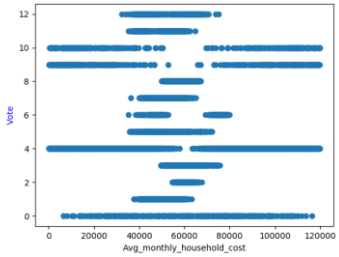
- Avg_monthly_expense_on_pets_or_plants- בעלי הוצאה גבוהה נוטים לבחור כחול ובעלי הוצאה נמוכה נוטים לבחור צהוב:



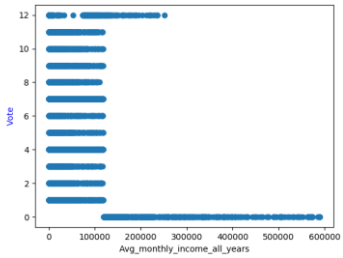
- Avg_monthly_expense_when_under_age_21- בעלי הוצאה גבוהה נוטים לבחור כחול ובעלי הוצאה נמוכה נוטים לבחור צהוב:



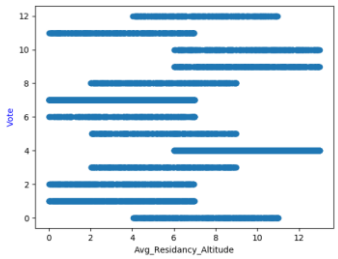
- Avg_monthly_household_cost- הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל רוב הערכים אמנם שאר הצבעים נפוצים רק אצל ערכים ממוצעים:



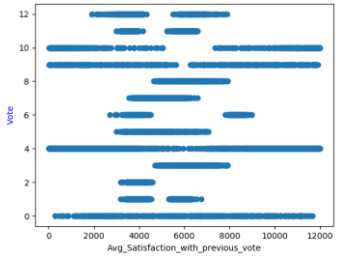
- Avg_monthly_income_all_years- בעלי הכנסה גבוהה בוחרים רק כחול:



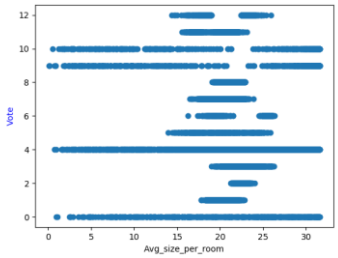
- Avg_Residancy_Altitude- מגמה מעורבת בעיקר לכל קבוצת גבהים קיימת קבוצת צבעים:



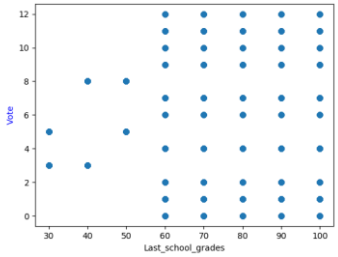
- Avg_Satisfaction_with_previous_vote – הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל כולם אמנם שאר הצבעים נפוצים רק אצל ערכים ממוצעים:



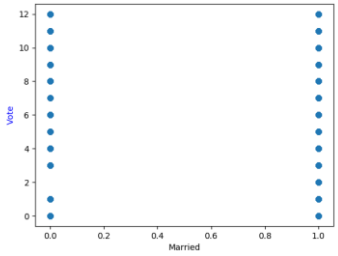
- Avg_size_per_room- הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל כולם ובעלי ערכים קטנים בוחרים רק בהם:



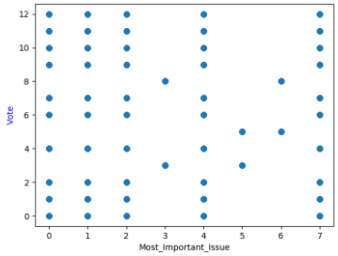
- Last_school_grades- בעלי ערכים קטנים בוחרים רק באפור, כתום ואדום:



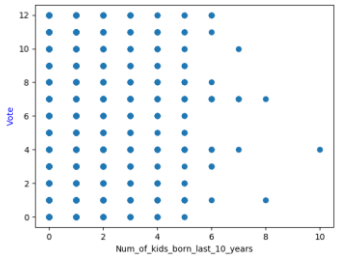
- Married- רווקים (לא נשואים) לא בוחרים בירוק (בניגוד למציאות 😉):



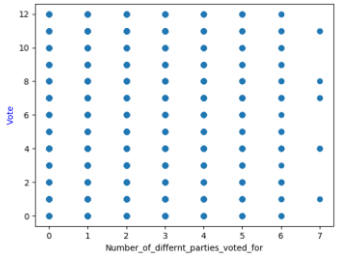
- Most_Important_Issue- הצבעים אפור, כתום ואדום נפוצים רק אצל ערכים מסוימים:



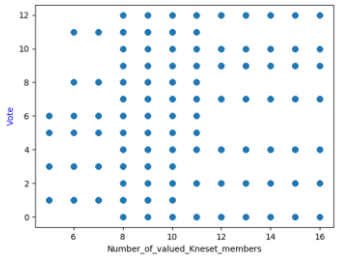
- Num_of_kids_born_last_10_years- ערכים גבוהים נוטים לבחור בחום, חאקי, סגול וסיגל:



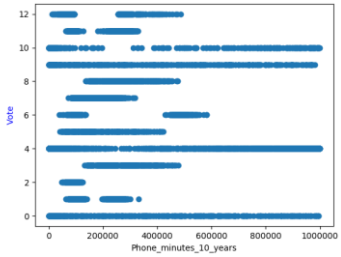
- Number_of_differnt_parties_voted_for- ערכים מקסימליים נוטים לבחור בחום, חאקי, סגול, אדום ולבן:



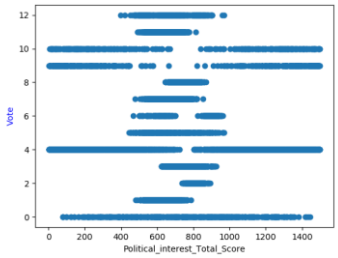
- Number_of_valued_Kneset_members- מגמה מעורבת בערכים נמוכים וגבוהים:



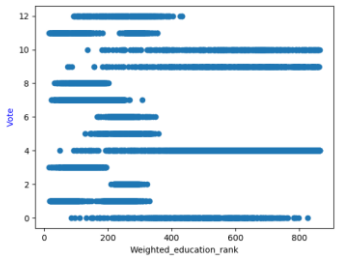
- Phone_minutes_10_years- ערכים גבוהים בוחרים רק בצבעים כחול, חאקי, טורקיז וסיגל, שאר הצבעים נפוצים בעיקר אצל ערכים נמוכים:



- Political_interest_Total_Score- הצבעים כחול, חאקי, טורקיז וסיגל נפוצים אצל רוב הערכים אמנם שאר הצבעים נפוצים רק אצל ערכים ממוצעים:



- Weighted_education_rank- ערכים גבוהים בוחרים רק בצבעים כחול, חאקי, טורקיז וסיגל, שאר הצבעים נפוצים בעיקר אצל ערכים נמוכים:



2. האלגוריתם מומש בקובץ *bonus_relief.py* בקובץ זה קיימת הפונקציה *relief* אשר מקבלת:
   - x_train: train data frame
   - y_train: train labels data frame
   - local_nominal_feature: the nominal features to examine
   - local_numerical_features: the numerical features to examine
   - N , num_of_iter: number of iterations
   - τ , threshold: threshold

   וממשת את האלגוריתם ולבסוף מחזירה את התכונות הנבחרות.

עבור N=1,000,τ=1 התקבלו התכונות הבאות:

```
'Looking_at_poles_results'
'Gender'
'Overall_happiness_score'
'Avg_Residancy_Altitude'
'Yearly_ExpensesK'
'%Time_invested_in_work'
'Phone_minutes_10_years'
'%_satisfaction_financial_policy'
'Avg_monthly_income_all_years'
'Number_of_differnt_parties_voted_for'
'Number_of_valued_Kneset_members'
'Main_transportation'
'Occupation'
'Financial_agenda_matters'
```

יתרונות:
- קל למימוש.
- מתמודד עם מספר גדול של דוגמאות.
- ניתן לשלוט בבחירת תכונות על ידי קביעת סף.
- משתמש בניתוח סטטיסטי בלבד
- יעילות זמן ריצה פולינומיאלית

  $(\Theta(pmn)[based\ on\ the\ above\ paper, section\ 6 - Conclusion]$

  נובעת מ:

  1. לא מחפשים במרחב תתי הסטים של התכונות בצורה מפורשת
  2. ויתור על מינימאליות תת סט התכונות המוחזרות.
- יחסית חסין לרעש
- לא מושפע מתלויית בין הפיצ'רים.

חסרונות:
- דרוש מספר איטרציות גדול על מנת לקבל תוצאות ברורות.
- לא תמיד ברור איזה סף לקבוע על מנת לקבל תת קבוצה טובה.
- לעומת SFS לא בוחן מסווג ספציפי ולכן נוכל לקבל תוצאות שונות על מסווגים שונים.

- מושפע מהחסרונות שנובעים מכל חישוב על בסיס "מרחק", לדוגמא:

1. תכונות נומנליות שיכולות להיות חסרות משמעות משפיעות באופן דרסטי על המרחק ל $nearest\ hit\backslash miss$.
2. מעניק לכל התכונות משקל שווה בחישוב המרחק(כתלות בבעיה, תכונה זו יכולה להיות מועילה מאוד, או מזיקה מאוד).

:First Bonus – SFS Algorithm

3. האלגוריתם מומש בקובץ *bonus_sfs.py* בקובץ זה קיימת הפונקציה *sfs_algo* אשר מקבלת:
   - x_train: train data frame
   - y_train: train labels data frame
   - clf: classifier to examine
   - subset_size: user required subset size not mandatory

מממשת את האלגוריתם על ידי ניקוד של k-cross fold validation (k=3) ולבסוף מחזירה את קבוצת התכונות הנבחרות, גודל הקבוצה נקבע על ידי פרמטר של המשתמש או אם לא ניתן אז כאשר אף תכונה לא משפרת את הסיווג.

בחנו את האלגוריתם על ידי שני מסווגים שונים:

- SGDClassifier תוצאות:

```
SVM Classifier accuracy score before SFS is: 0.7723601652376123
SVM Classifier selected features are: [
'Weighted_education_rank', 'Avg_monthly_income_all_years', 'Last_school_grades',
'Number_of_valued_Kneset_members', 'Phone_minutes_10_years',
'Avg_education_importance', 'Avg_government_satisfaction', 'Married',
'AVG_lottary_expanses', 'Avg_Residancy_Altitude', 'Yearly_ExpensesK',
'Avg_size_per_room']
SVM Classifier accuracy score after SFS is: 0.8070339800786065
```

- KNN תוצאות:

```
K Neighbors Classifier accuracy score before SFS is: 0.5826105372698062
K Neighbors Classifier selected features are: [
'Avg_environmental_importance', 'Avg_government_satisfaction',
'Avg_education_importance', 'Avg_Satisfaction_with_previous_vote',
'Last_school_grades']
K Neighbors Classifier accuracy score after SFS is: 0.8313137509347758
```

יתרונות:

- קל למימוש.
- מתמודד עם מספר גדול של דוגמאות.
- ניתן לשלוט בגודל תת קבוצת התכונות הנבחרות.
- התכונות הנבחרות מותאמות לאלגוריתם למידה ספציפי.
- החלטתיו נקבעות על ידי המדד שמוערך אלגוריתם הלמידה ולא בצורה מוחלטת, כל מתכנת יכול לבחור את המדד אותו הוא רוצה למקסם.
- סקלבילי – ניתן לייצר סט תוצאות(תת סט של פיצ'רים) הולכות וגדולות ע"י שימוש באלגוריתמי קלסיפיקציה שונים, ובכך לקבל תמונה מציאותית יותר לגבי אופן התנהגות הבעיה.

חסרונות:

- זמן ריצה תלוי במספר התכונות ולכן נעדיף להפעילו בהינתן מספר קטן של תכונות.
- החסרון העיקרי - **לא ניתן לחזור אחורה ולבטל בחירה של תכונה גם אם היא redundant לחלוטין.**
- סכנה לoverfit- מכיוון שהתכונות הנבחרות נבחרות ביחס לאחוזי הדיוק על סט הולדיצה, ישנה סכנה ברורה שהתכונות הנבחרות יהיו רגישות יותר ל overfit.
- חמדני – בוחר להכניס \ לא להכניס את התכונה הבאה רק אם משפרת את יכולת החיזוי ולכן לא לוקח בחשבון תלויות בין תכונות(בעיה בה זה יכול לבוא לידי ביטוי היא לדוגמא בעיית ה XOR)
- עוד חסרון שנובע מחמדנות- מחפש במרחב תתי התכונות באופן אינקרמנטלי(עולה), ולכן סביר להניח שבמרחב תכונות גדול יפספס תתי סטים של תכונות בעלות אחוזי דיוק גבוהים יותר.