מגיש: איליה קוטלוב 321851784 s3218517@campus.technion.ac.il

1. התהליך והחלטות משמעותיות:

1.1 בחירת מודל אוטומטי:
כחלק ממשימות הבונוס, מימשנו בחירת מודל אוטומטי כחלק אינטגרלי של משימות החובה.
לצורך כך כל הסקריפטים שעוסקים בmodel evaluation מרוכזים במחלקה שנקראת modelSelector.
מכיוון שיש לנו 3 משימות חיזוי, הSelector יאמן את כל המועמדים להיות המודל הנבחר על הtraining_set ויבדוק אותם על הvalidation_set.
ההבדל בין המשימות הוא במטריקות אשר ישמשו לבדיקות.
כל מודל יקבל score בסיום הvalidation עבור הביצועים שלו לכל משימת חיזוי.

1.2 מודלים שונים יכולים להיות טובים עבור משימות חיזוי שונות. כדי להתמודד עם זה החלטנו לכלול בModelSelector שלנו את האפשרות לבחור עבור משימות שונות את המודל המתאים, כך שמראש לא ידוע האם יבחר מודל יחיד לכל המשימות או מודלים שונים.

2. התאמת תרגיל בית 2 למשימות הנוכחיות:

2.1 מכיוון שבתרגיל זה ניתנו לנו הפיצ'רים הנכונים, בשונה מתרגיל בית 2 בו עשינו את תהליך הכנת הדאטא לכל הדאטא, כאן בחלק של ה data preparation בקובץ מתרגיל בית קודם מלכתחילה סיננו את המידע רק לפיצ'רים הטובים ולפיצ'רים שבהם השתמשנו להשלמת ערכים חסרים שהם high correlated:
כלומר סה"כ:

- Yearly_IncomeK, Number_of_differnt_parties_voted_for, Political_interest_Total_Score, Avg_Satisfaction_with_previous_vote, Avg_monthly_income_all_years, Most_Important_Issue, Overall_happiness_score, Avg_size_per_room, Weighted_education_rank

יחד עם:

['Avg_monthly_income_all_years', 'Avg_monthly_expense_when_under_age_21']

['Avg_size_per_room', 'Avg_education_importance', 'Avg_monthly_household_cost', 'Avg_environmental_importance']

['Phone_minutes_10_years', 'Weighted_education_rank', 'Avg_Residancy_Altitude']

ולאחר השלמת הערכים החסרים והscaling כפי שזה נעשה בתרגיל בית קודם,
השארנו רק את הפיצ'רים שניתנו במסגרת התרגיל.

2.2 שימוש בvalidation set:
לאחר בדיקת התוצאות של החלקים הבאים בתרגיל, ראינו כי יחד עם הvalidation set מתקבלות תוצאות טובות יותר בחיזויים.

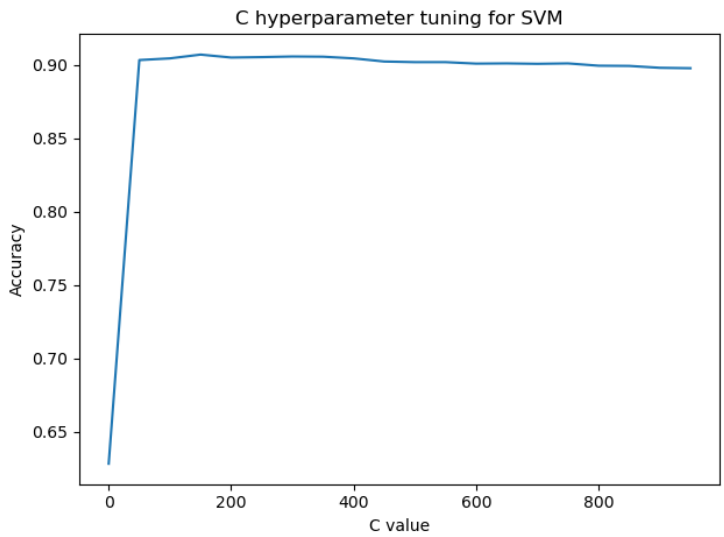
3. מודלים מועמדים:
בחלק זה נדון במודלים בהם בחרנו לניסויים שלנו. לחלק שבו צריך לחזות את המצביעים בצורה הסתברותית, אנחנו צריכים לחזות את ההסתברות ולא את הtag עצמו, אז ברצוננו לבחור במודלים שמספקים פונקציונליות זו.
הן על פי ההרצאות והן ממקורות חיצוניים, חקרנו ומצאנו כי SVM, KNN ו Random Forest יכולים להתמודד בצורה מיטבית עם בעיות multi-class classification, לכן בחרנו במודלים אלו יחד עם MLP, סה"כ 4 מודלים.

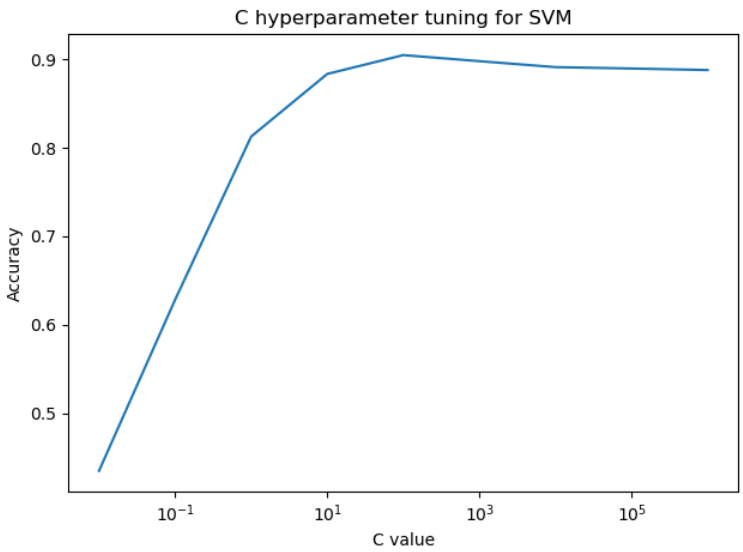
3.1 כיוונון היפר-פרמטרים עם k-cross validation:
בכדי לבחור את המועמדים הטובים ביותר, ביצענו 5-cross validation על הtraining set בלבד. תהליך הcross validation ממומש במחלקה crossValidator. בכל הניסויים השתמשנו במידת הf1_score לצורך הערכת הביצועים של המודלים.

3.2 כוונון הפרמטרים עבור SVM:
הפרמטרים הכי חשובים של הSVM הם הkernel ופרמטר הC. הkernel מגדיר את צורת ההפרדה של הhyper-plane והC פרמטר אחראי על סיבוכיותו. C גדול מגדיל את סיבוכיות המודל, מקטין את הBias ומגדיל את הoverfitting ולהיפך.Kernel לינארי בכלל לא התכנס, והמשיך לרוץ לנצח, sigmoid נתן תוצאות הרבה פחות טובות (פי יותר מחצי), poly נתן תוצאות

מעט פחות טובות באחוזים בודדים אך דרש פרמטר C גדול יותר ולכן גם זמן ריצה גדול יותר.

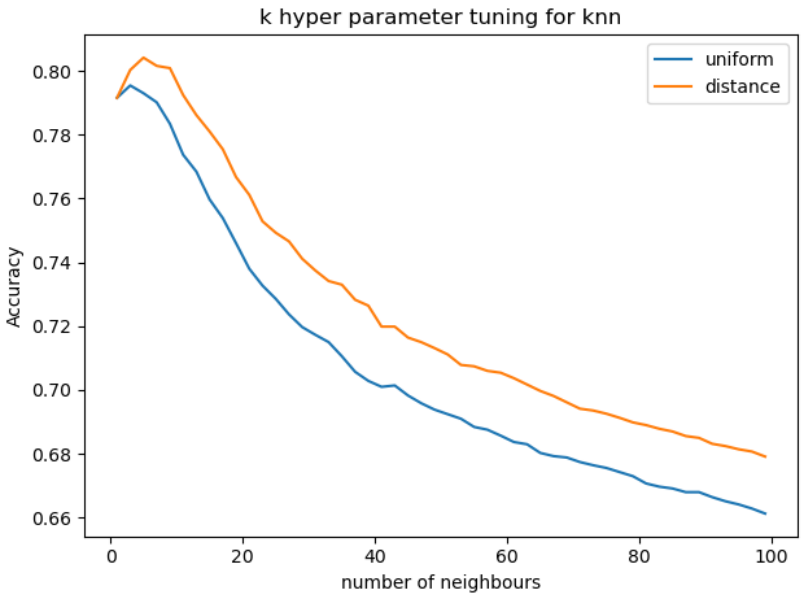
עבור RBF התקבלו התוצאות הבאות עבור כיוונון פרמטר הC:
וכפי שניתן לראות התוצאות הכי טובות התקבלו סביב 100<=C<=200
כאשר מעל 200 התנודות בדיוק הן קטנות מאחוז אחד, ובין 100 ל200 התנודות הן בגדר אחוז (89-90).





3.3 כיוונון פרמטרים עבור KNN:

נבדוק את שתי השיטות עבור פרמטר הweights: uniform, distance ולצורך כך בcrossValidator ניצור שני מסווגי KNN ונשווה את ביצועיהם. על מנת לבחור את הk הטוב ביותר נעשה k-fold cross validation עבור שניהם. התוצאות הכי טובות מתקבלות עבור k בטווח בין 3 ל-10 בחלוקות שונות והרצות שונות והתנודות הן בגדר 1-2 אחוזים, לכן נבחר k=5.
וכפי שניתן לראות תוצאות יותר טובות מתקבלות כשאר שיטת הwieghts היא distance.



3.4 כיוונון עבור Random Forest:

מסווג זה מתאים מספר מסווגים מסוג עץ Decision Tree Classifier לדגימות שונות של הדאטא ומשתמש בממוצעי התוצאות כדי לשפר את דיוק החיזוי וכדי להפחית את הoverfitting.
על פי מקורות חיצוניים שחקרנו, הפרמטרים הכי חשובים עבור מסווג זה הם:

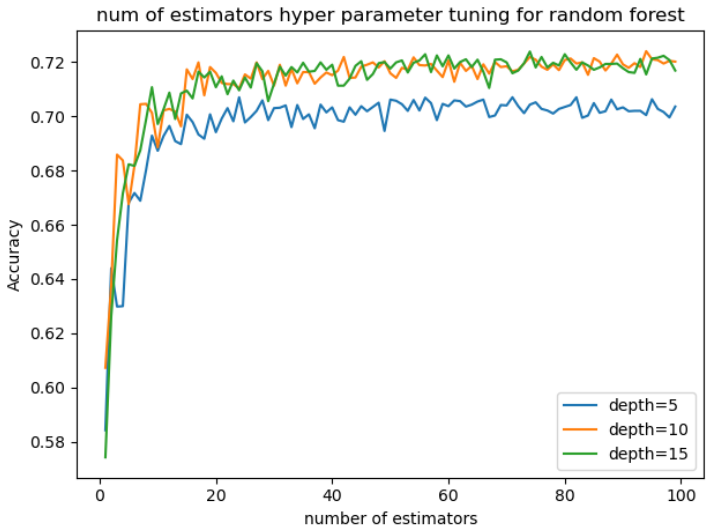
1. Number of Trees: באופן כללי ככל שיש יותר עצים כך המסווג טוב יותר, אבל מספר גדול של עצים יכול להאט את המערכת באופן משמעותי.
2. Maximum Depth: מייצג את העומק המקסימלי בכל עץ ביער. ככל שהעומק גדול יותר כך נגרם יותר overfitting, לכן בהתבסס על הגרפים

החלטנו לקחת עומק 8 כאופטימלי, מכיוון שעומקים גדולים יותר לא משפרים בצורה משמעותית את הביצועים.

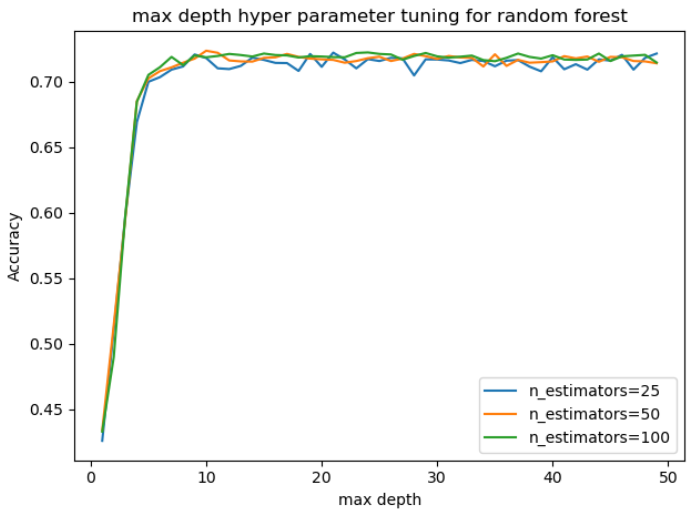
3. Minimum samples split: מייצג את כמות הדגימות המינימלית הנדרשת כדי לפצל צומת בעץ. הדיוק הכי טוב מתקבל עבור הערך 0.01, כלומר 70 דוגמאות (מתוך 7000 של הסט).

פרמטרים אחרים נראים לנו פחות משמעותיים (ופחות אינטואיטיביים), לכן בחרנו לעבוד עם 3 הפרמטרים האלו.

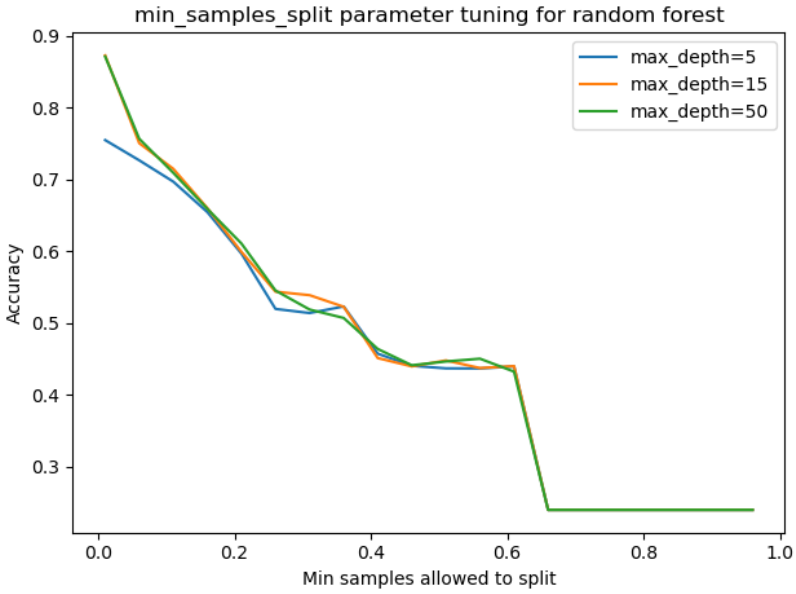
n tuning



depth tuning



min samples tuning



3.5 :MLP = Multi-layer Perceptron

המודל הנ"ל ממזער את פונקציית הlog loss באמצעות LBFGS או stochastic gradient descent. בחלק זה נדון בפרמטרים הכי חשובים עבורו ועבור הערכים שבחרנו.

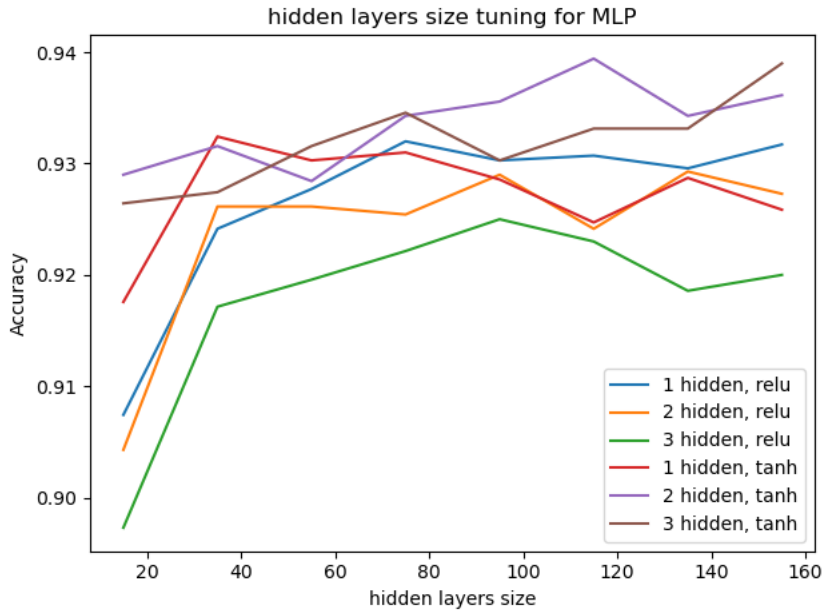
א. Number of Hidden Layers: החלטנו לבחון 3 אופציות בחלק זה, 1,2,3 שכבות נסתרות.

ב. Hidden Layer Size: גודל זה קובע את את הגבול העליון של כמות הנויירונים החבויים אשר לא יגרמו ל-overfitting:

$$N_h = \frac{N_s}{\alpha * (N_i + N_o)}$$

כאשר N_0 הוא מספר הנויירונים של הפלט, N_i הוא מספר הנויירונים של הקלט, N_s הוא הוא מספר דוגמאות האימון וalpha הוא פאקטור שרירותי בין 2 ל10. במקרה שלנו, עבור alpha = 10, הגבול העליון של הנויירונים הוא 166 בשכבה הנסתרת. לבסוף, שגודל השכבה היעיל הוא 95.

ג. Activation Function: בדקנו שתי אופציות: Tanh ו-ReLU, כאשר התוצאות הטובות יותר התקבלו עבור Tanh.



4. אמצעי מידה לביצועים:

4.1 אמצעי מדידת רב המצביעים: על מנת לחזות לאיזו מפלגה יצביע רב המצביעים, אנחנו מציעים את המידה הכי פשוטה שעולה לראש: binary score. אם הרב שנחזה הוא כמו בvalidation set, אז הscore הוא 1, אחרת 0. השיטה הזו מסוגלת לתת לנו מודלים שונים שיכולים לחזות את המפלגה המנצחת. על מנת לעשות את השיטה יותר סלקטיבית החלטנו להשתמש בf1 score כמידה משנית.

4.2 אמצעי מדידת חלוקת הקולות:
במשימה זו אנחנו רוצים לחזות בצורה מיטבית את חלוקת כל הקולות בtest set בין המפלגות השונות. במילים אחרות, אנחנו רוצים להשוות בין היסטוגרמת הקולות שנחזו לבין היסטוגרמת הקולות כפי שהם מופיעים בtest set, לכן המטריקה שהשתמשנו בה היא מרחק אוקלידי בין שתי ההסיטוגרמות:

$$D = \sqrt[2]{\sum_{i=0}^{12}(histPredicted_i - histTrue_i)^2}$$

לפי מידה זו, התפלגות קולות שנחזו תהיה רחוקה מאוד מהתפלגות הקולות האמיתית אם היא תהיה שונה מאוד ממנה, ולהיפך. לכן מודל זה ייחשב כהכי טוב למשימה זו.

4.3 אמצעי מדידת המצביעים הכי סבירים (שיצביעו):
במשימה זו עלינו לחזות את ההסתברות של כל מצביע להצביע לכל מפלגה, אך לא את הלייבל המדוייק. זו הסיבה שהגישה הראושנה הייתה להשתמש במטריקות שונות.
נסמן ב-P_i את הסט של המצביעים שנחזה הסתברותם להצביע למפלגה ה-i מעל threshhold, T_ii בתור המצביעים שבאמת הצביעו למפלגה ה-i, אזי:

א. החיתוך בין P_i ו-T_i הוא כמות המצביעים שלהם בפועל תסופק תחבורה, וככל שהחיתוך גדול יותר, כך הscore גבוהה יותר.

ב. P_i – T_i הם המצביעים שנחזו לא נכון, אשר לא יצביעו למפלגה הנתונה, אבל יקבלו נסיעה חינמית. תוצאה גבוה תקטין את הscore.

ג. T_i – P_i הם מצביעים שמצביעים למפלגה הנתונה, אבל לא תסופק להם תחבורה ולכן אולי לא יצביעו כלל. תוצאה גדולה כמובן היא רעה ותקטין את הscore.

לכן לבסוף נקבל את הנוסחה הבאה:

$$Score = \sum_{i=0}^{12} |P_i \cap T_i| - |P_i - T_i| - |T_i - P_i|$$

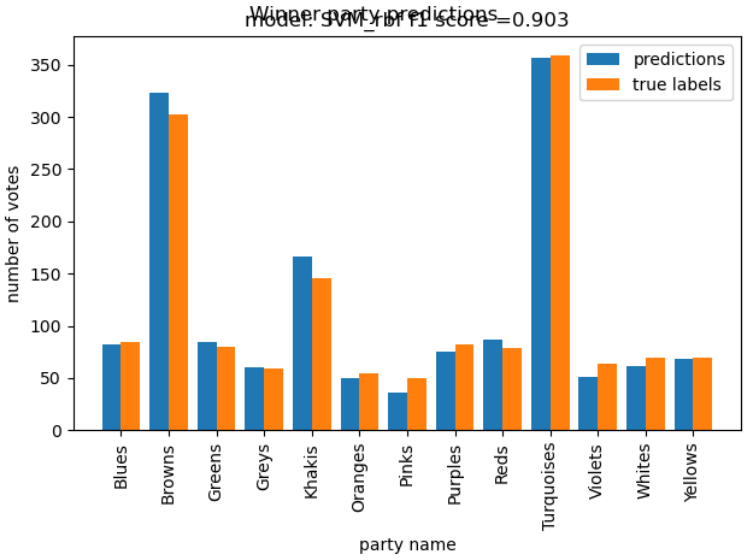
במשך עבודתינו שמנו לב כי f1 score ממושקל תואם באופן טוב מאוד לscore הגמיש שלנו. לכן החלטנו להשתמש ב f1 score ממושקל כמידה לטובת משימה זו.

5. תוצאות החיזוי והמודל הנבחר:

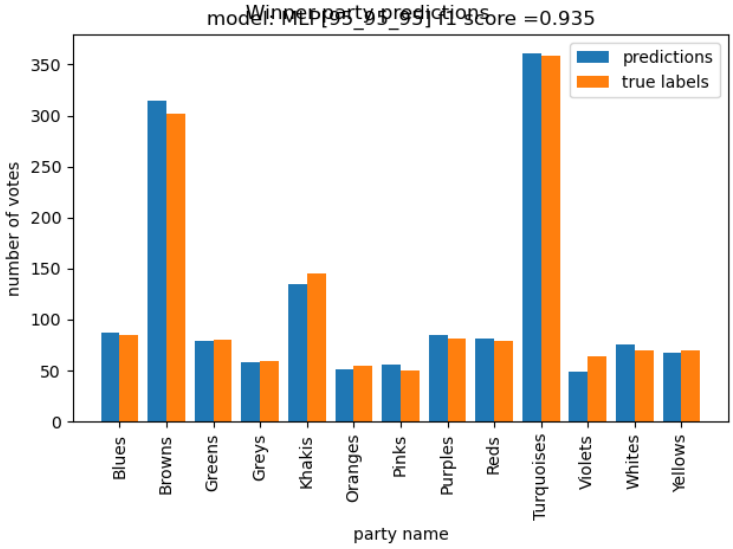
בחלק זה נדון בתוצאות כל משימת חיזוי, איכות התוצאות בהתאם למידות שנבחרו להערכת התוצאות ואילו מודלים נבחרו למשימות החיזוי השונות.

5.1 כדי להמחיש בצורה וויזואלית את תוצאות החיזוי על הvalidation set נראה את ההיסטוגרמות של תוצאות האמת אל מול תוצאות החיזוי. על פי הגרפים, 3 מודלים חיזו בצורה נכונה את הזוכים. מודל הMLP קיבל את תוצאת הf1 score הגבוה ביותר, לכן נבחר להיות המודל הטוב ביותר עבור משימה זו.

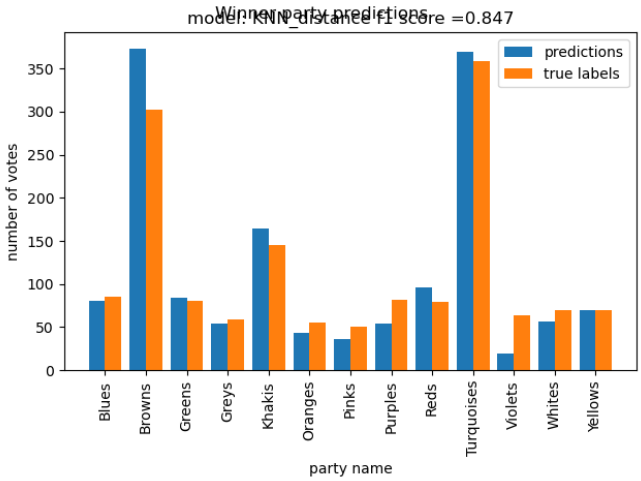
SVM



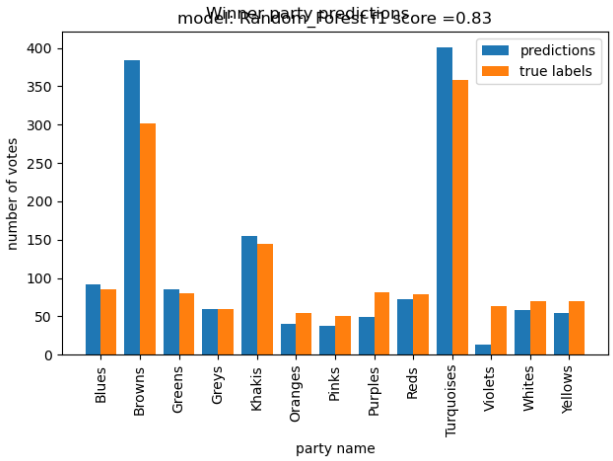
MLP



KNN



Random Forest

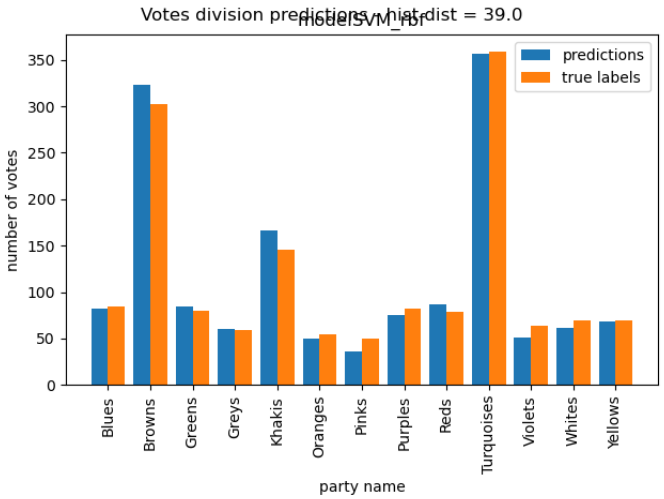


5.2 התפלגות הקולות:

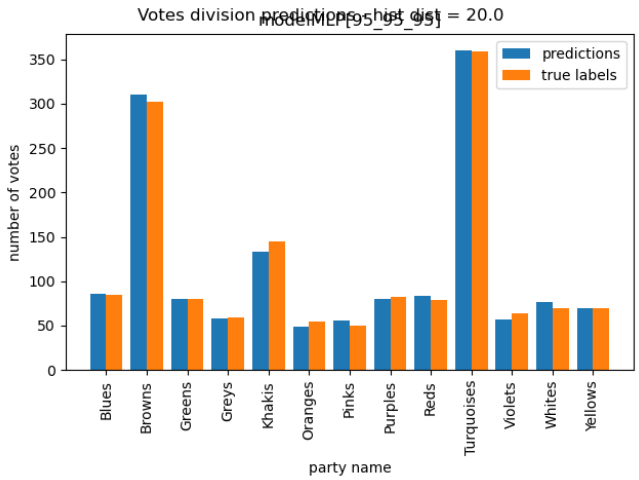
כדי להמחיש בצורה ויזואלית את תוצאות החיזוי על הvalidation set נראה את ההיסטוגרמות של תוצאות האמת אל מול תוצאות החיזוי. בהתאם למידת המרחק האוקלידי שנבחרה, אנחו מעוניינים במרחק הכי קטן, מכיוון שהמרחק הכי קטן תואם לשוני הכי קטן בין היסטוגרמות ההתפלגות של החיזוי והאמת.

לסיכום, נראה כי הMLP מקבל את התוצאות הטובות ביותר עם score של 20.

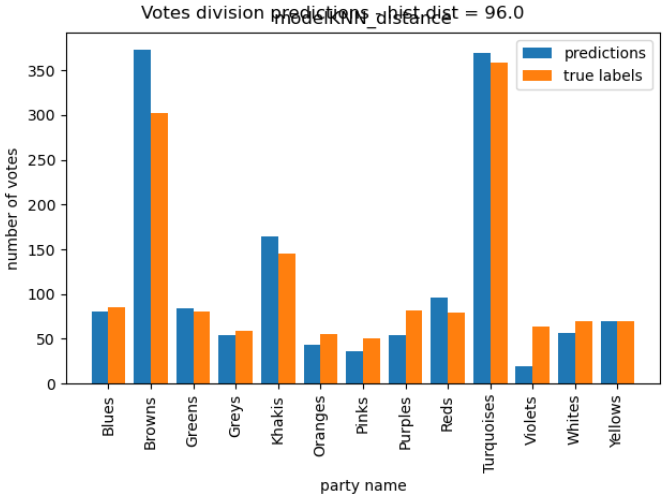
SVM



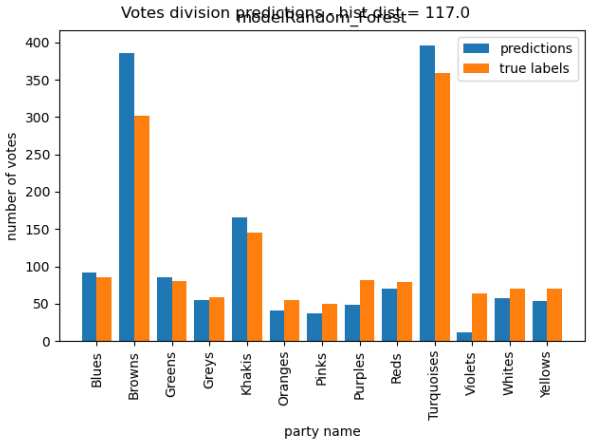
MLP



KNN



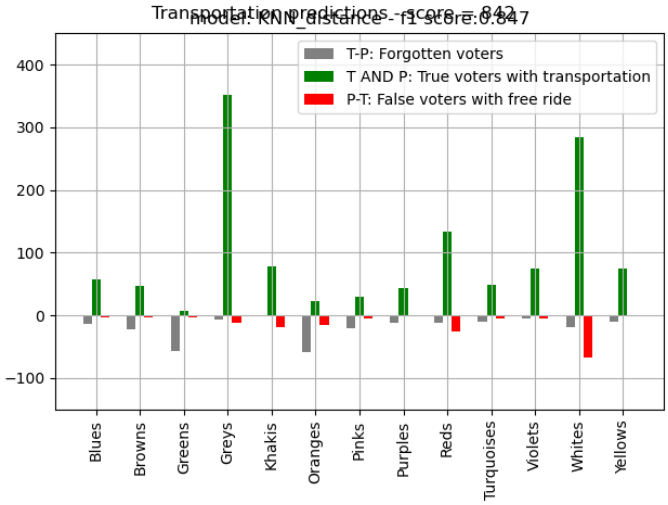
Random Forest

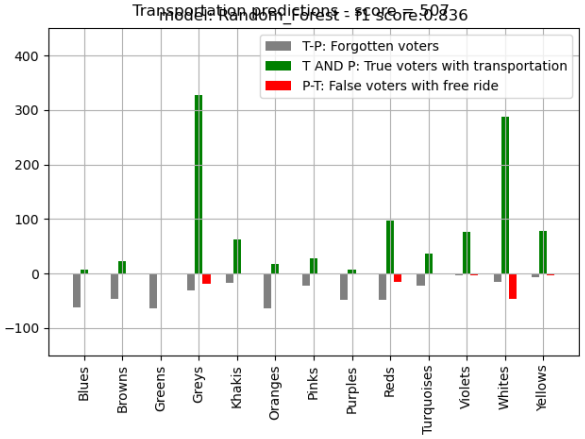


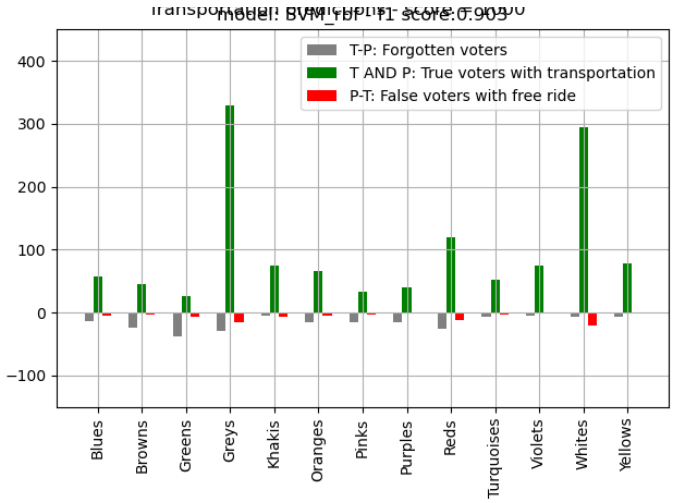
5.3 טרנספורטציית המצביעים:

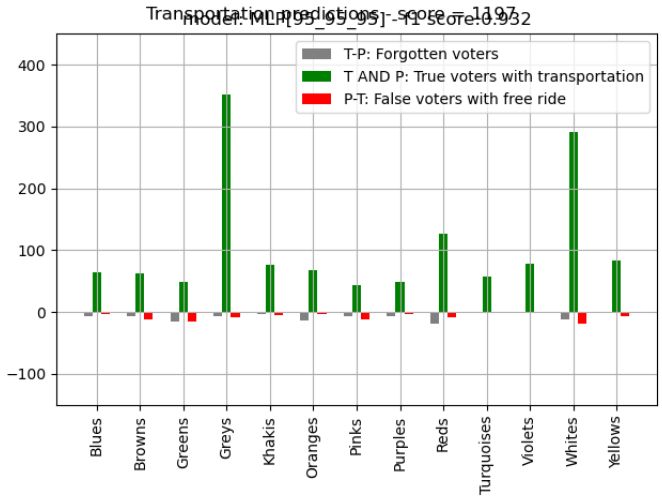
כדי להמחיש בצורה וויזואלית את תוצאות החיזוי על הvalidation set נראה את ההיסטוגרמות בצורה של bar charts בהתאם למידה שבה בחרנו למשימה זו.

כמו שתיארנו בתיאור המידה, הבאר הירוק יתרום לscore תרומה חיובית והאפור והאדום יתרמו תרומה שלילית. כפי שניתן לראות התוצאות של Random Forest הן התוצאות הגרועות ביותר, לפי כך רבים מהאנשים שלא יצביעו לטובת המפלגות יוסעו לשווא, לעומת זאת תוצאות חיזוי MLP גבוהות, כלומר voters true יקבלו הסעות בהתאם למה שהיינו רוצים. אדם נחש בתור מצביע למפלגה רק אם ההסתברות להצבעה היא מעל 0.5. המימוש שלנו מאפשר את הפונקציונליות לקביעת הthreshold שיכול להשפיע רק על הקולות שעליהם מתבצע החיזוי, אך לא על המודל הטוב ביותר שנבחר, מכיון שנבחר לפי f1 score.









6. תשובות סופיות:

לשלושת המשימות החיזוי הטוב ביותר מתבצע באמצעות MLP[95,95,95].

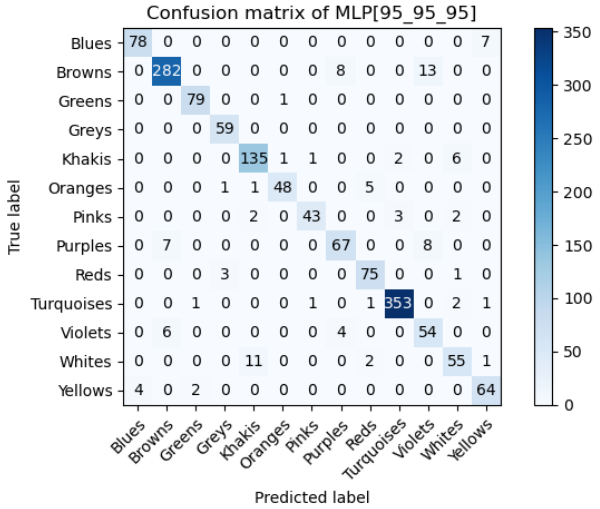
6.1 המפלגה שתזכה לפי מודל זה היא: Turquoises

```
MLP[95_95_95]  prediction -  Turquoises  party will win the elections.
```

6.2 התפלגות הקולות לפי המודל זה היא:

```
MLP[95_95_95]  prediction - Vote division:
Party  Blues : 5.5 %
Party  Browns : 19.7 %
Party  Greens : 5.5 %
Party  Greys : 4.2 %
Party  Khakis : 9.9 %
Party  Oranges : 3.3 %
Party  Pinks : 3.0 %
Party  Purples : 5.3 %
Party  Reds : 5.5 %
Party  Turquoises : 23.9 %
Party  Violets : 5.0 %
Party  Whites : 4.4 %
Party  Yellows : 4.9 %
```

6.3 Confusion Matrix & Test error:




```
Model  MLP[95_95_95]  reached  7.2 % error.
```

7. בנוסף לגישת ה"different model for each task" אנחנו רוצים לנסות את שיטת המודל היחיד לכל המשימות.
בחלק זה נענה על אותן השאלות כפי שעשינו עבור החלק הקודם.

7.1 איך נבחר את המודל הכי טוב:
בחרנו לממש תהליך בחירת מודל אוטומטי באותה המחלקה modelSelector אשר בוחרת מודל אחד עבור כל המשימות ע"י ביצוע שני שלבים:

א. איבלואציה (evaluation) של כל המודלים המועמדים על הvalidation set ובחירת כל המודלים אשר צדקו בשאלת המפלגה המנצחת. אנחנו רואים במשימה זו את המשימה הכי חשובה, מכיוון שהיא עוקת בתוצאת הבחירות העיקרית.
ב. איבלואציה (evaluation) של כל המודלים המועמדים על הvalidation set על מנת לחזות את המצביעים הכי סבירים עבור משימת הטרנספורטציה, ונבחר את המודל שקיבל את הביצועים הכי טובים בהתאם למטריקה עבור משימה זו. מכיוון שהזכייה בבחירות היא המשימה הכי חשובה, המשימה השניה הכי חשובה היא לספק לאותה מפלגה שאמורה לנצח את החיזוי הכי טוב למצביעיה, כדי שיגיעו להצביע והמפלגה המנצחת באמת תנצח.

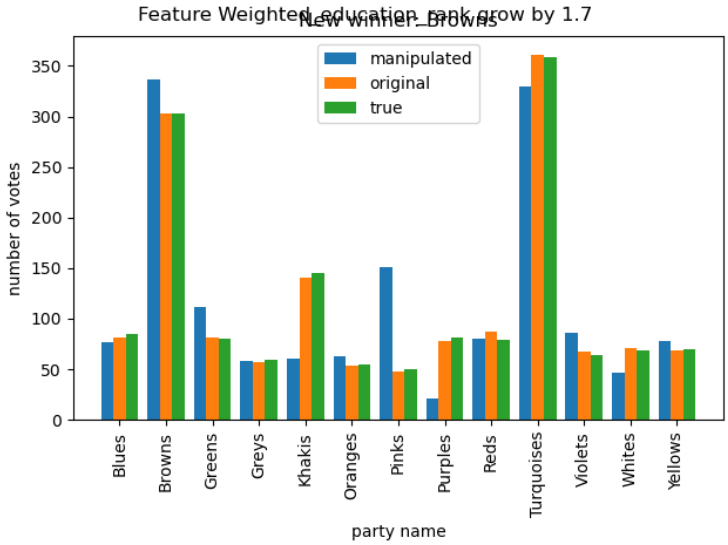
לכן סה"כ בחרנו במודל הMLP. לא התייחסנו בתהליך זה לvote devision מכיוון שלא ראינו חשיבות לכך, כמו החשיבות לשני המשימות בבחירה למודל היחיד כפי שתיארנו, אך כפי שראינו MLP חוזה גם את התפלגות הקולות בצורה מיטבית, וכמו שראינו בחלק הקודם, התוצאות התקבלו בצורה בלתי תלוייה, ולכן הן זהות גם עבור חלק זה.

8. ההשוואה בין הגישות:
כפי שניתן לראות, בשתי הגישות בחרנו באותו המודל מכיוון שקיבלנו את אותן התוצאות.

א. היתרון של גישת one for all היא פשטות המימוש.
ב. במקרה שלנו אותו המודל נבחר לכל שלושת המשימות, מה שלא בהכרח תמיד יקרה. במקרה שיש מודלים שונים שטובים עבור משימות שונות, אנחנו עושים tradeoff בין דיוק החיזוי לבין פשטות המימוש.

9. Feature Manipulation – Bonus:

בחלק הזה אחנו נחפש פי'צרים שיכולים שבאמצעות מניפולציות עליהם נוכל לגרום למפלגה אחרת לנצח. על מנת להשלים משימה זו, יצרנו את המחלקה featureManipulator המוצא באופן אוטומטי פיצ'רים כאלו. לפני הכל אנחנו מספקים את המודל שחוזה הכי טוב את המפלגה המנצחת, לאחר מכן בצורה איטרטיבית הfeature Manipulator מפחית או מגדיל עמודה אחת של הvalidation set או ה-test set, בערך קבוע C. בכל איטרציה אנחנו מתחילים בערכים קטנים של C ובכל פעם מגדילים אותם, עד שהמודל חוזה שמפלגה אחרת תנצח. ניתן לראות בתוצאות ומהגרף, כי אם נגדיל את משקל Weighted Education Rank פי 1.7 או את Political_Interest_Total_Score, זה יגרום לBrowns לנצח.



וכמו כן התוצאות המלאות:

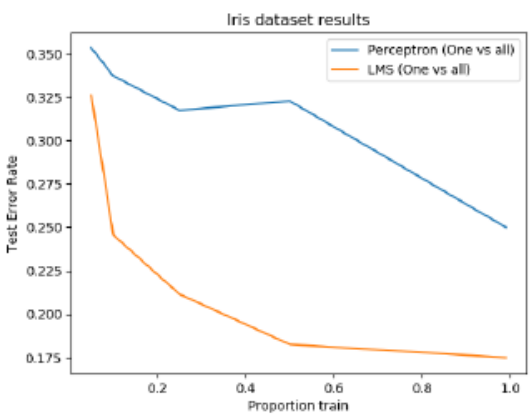
```
If  Number_of_differnt_parties_voted_for  will grow by  4.2 , that will cause  Browns  to win
If  Political_interest_Total_Score  will grow by  1.7 , that will cause  Browns  to win
If  Avg_Satisfaction_with_previous_vote  will grow by  7.5 , that will cause  Browns  to win
If  Avg_monthly_income_all_years  will grow by  8.1 , that will cause  Blues  to win
If  Overall_happiness_score  will grow by  7.5 , that will cause  Reds  to win
If  Avg_size_per_room  will grow by  6.5 , that will cause  Greys  to win
If  Weighted_education_rank  will grow by  1.7 , that will cause  Browns  to win
```

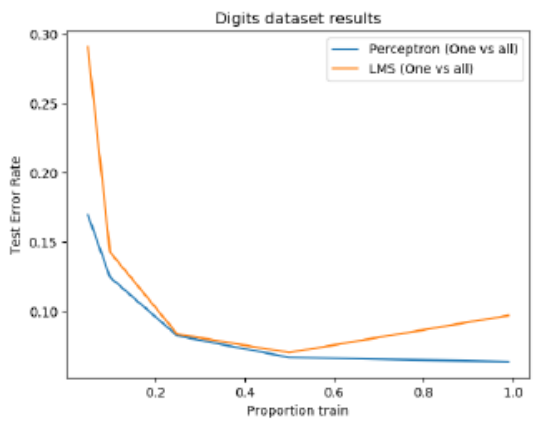
10. מימוש אלגוריתם Adaline:

בחלק זה מימשנו את אלגוריתם Adaline. בהתחלה מימשנו אלגוריתם איטרטיבי אשר עבד בצורה טובה, אבל היה מאוד איטי. לכן הגענו למסקנה, כי הפיתרון הכי טוב הוא אנליטי, שגם אותו מימשנו.
לאחר מכן מימשנו בטכניקת 1-vs-all את אלגוריתם Adaline עבור סיווג Multi-class. עבור הדאטא של Iris אימנו 3 מסווגים בינאריים ועבור Digits אימנו 10 מסווגים.

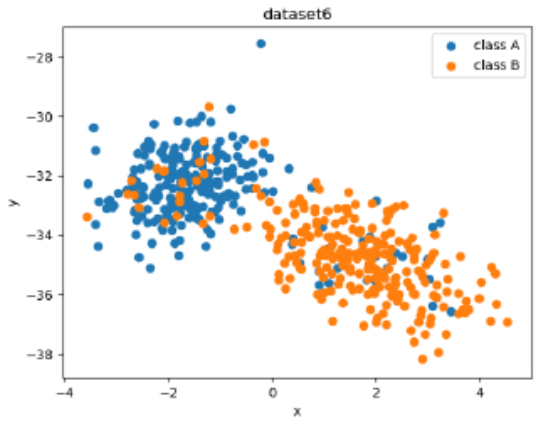
10.1 תוצאות על Iris וDigits:

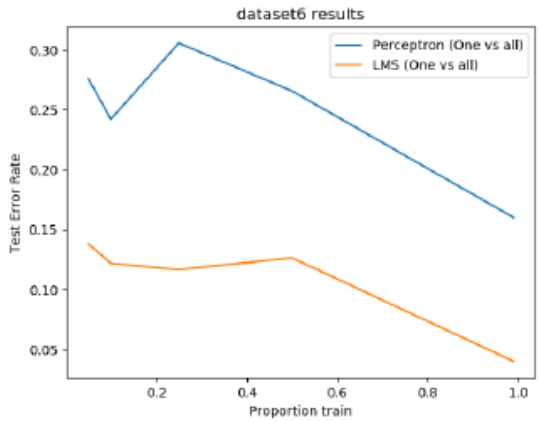
התאמנו את כמות המסווגים הנדרשת לכמות המחלקות של הדאטא וקיבלנו את הגרפים הבאים:





10.2 תוצאות על דאטא מג'ונרט:

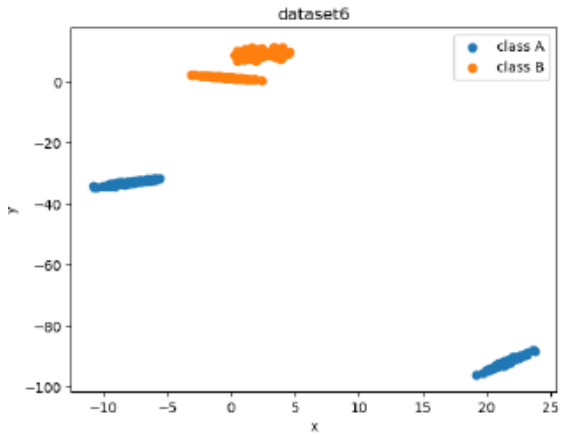


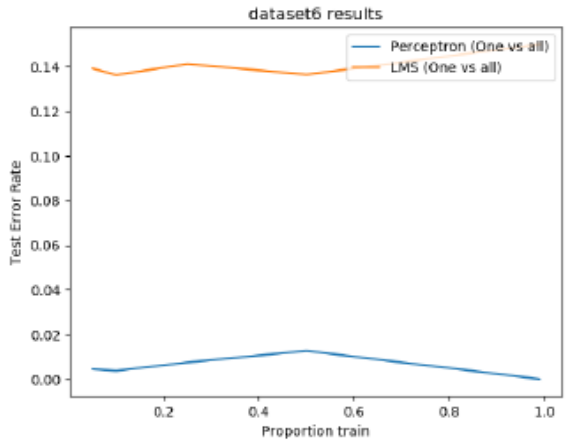


וכפי שניתן לראות LMS פעול טוב יותר על דאטא רועש.

LMS דורש סט אימון הרבה יותר קטן. הדאטא שבחרנו הוא דאטא לא לינארי בלתי ניתן להפרדה בגלל רעש רנדומלי. ההסבר הוא שדוגמאות רועשותלא משפיעות במידה משמעותית על גבול ההפרדה של אלגוריתם הLMS, מכיוון שהפרופורציה שלהם ביחס לכל הדאטא היא קטנה.

מצד שני, הם כן משפיעים עם הPerceptron, מכיוון שאם הדאטא לא ניתן להפרדה לינארית, אי אפשר להפתיח כי הוא יתכנס בכלל.





הפרספטרון פועל טוב יותר על דאטא מופרד היטב ללא רעש.

בדוגמה זו יש לנו דאטא שניתן להפרדה לינארית בצורה טובה ולכן כמות קטנה של דאטא מספיקה להתכנסות הפרספטרון.