PROJECT REPORT

בלה ליפובה 305857666
יפעת חיים 304828379
מור גלברג 302844519

בסדנה זו מימשנו סוכן Ad Network המתחרה מול סוכנים אחרים על קמפיינים ושטחי פרסום לקמפיינים בהם זכה. ההחלטות שמקבל הסוכן מושפעות מהמצב בסימולציה הנוכחית. בנוסף, כחלק מתהליך בניית הסוכן אספנו נתונים מהתחרויות, ובעזרתם שיפרנו את ביצועיו הן באמצעות אלגוריתם של למידה חישובית והן באמצעות מודלים פשוטים יותר.

נתאר את האופן שבו בנינו את האסטרטגיות עבור שלוש המטרות שסוכן ה-Ad Network שלנו צריך להשיג:

1. זכייה בקמפיינים.
2. זכייה ב-impressions.
3. השגת UCS level מסוים.

כחלק מבניית האסטרטגיות, חיפשנו ספרייה של למידה חישובית שתאפשר לנו להריץ אלגוריתמים באמצעות קוד. מצאנו ש-WEKA הוא כלי המאפשר להריץ אלגוריתמים של למידה חישובית הן באמצעות קוד java והן באמצעות GUI, מה שהקל עלינו להתנסות באלגוריתמים שונים ובבחירת הנתונים שישמשו ללמידה.

יש לציין שללא קשר לשיטת ה-testing הנבחרת, WEKA מאפשר לשמור רק את המודל שנבנה על כל ה-data. לכן, בחרנו את שיטת ה-testing של split של 85% ל-training, כדי שנקבל חיווי קרוב ככל האפשר למודל שיישמר לבסוף.

אסטרטגיה לזכייה בקמפיינים

תחילה החלטנו ליצור כלי שמחשב bid שמבטא את ההערכה שלנו לתקציב הנחוץ למימוש הקמפיין. לשם כך בנינו את המחלקה CampaignBidCalculator שתחשב את ה-bid לפי הפרמטרים הבאים שמפורטים לעומק ב-design specification:

1. אורך הקמפיין.
2. כמות ה-Impressions הממוצעת ליום.
3. Quality score.
4. Targeted segment ספציפי (כמות segments גדולה יותר).
5. חפיפה בימים וב-segments של קמפיינים.

בנוסף, כאשר השרת שלח הודעות מסוג CampaignAuctionReport, השתמשנו במידע זה כדי לשפר את ה-bid שלנו תוך כדי משחק ע"י זיהוי ה-bids המנצחים עד כה באותה סימולציה.

לאחר התנסות במספר תחרויות, הבנו שלעיתים כדאי לוותר על קמפיינים בהינתן מצבים מסוימים, ולכן הוספנו לוגיקה המכריעה האם לנסות לזכות בקמפיין או לא. הלוגיקה מתבססת על הקריטריונים הבאים שמפורטים לעומק ב-design specification:

1. הימים הראשונים לסימולציה.
2. התאמה של ה-targeted segments.
3. אורך הקמפיין.

במידה והחלטנו שאיננו מעוניינים בקמפיין זה, ניתן את ה-bid המקסימלי שעומד בדרישות המתוארות ב-spec של השרת. המטרה היא לאפשר לנו לזכות בקמפיין רק בצורה רנדומלית. כמו כן, ניתן את ה-Bid המקסימלי כאשר יש לנו quality score נמוך, מכיוון שהסיכוי שלנו לזכות בקמפיין בצורה לא רנדומלית נמוך מאוד. אם אכן אנו זוכים בקמפיין בצורה רנדומלית, אנו למעשה נקבל תקציב גדול למימוש הקמפיין ואף נוכל לשמור את חלקו כדי להגדיל את המאזן שלנו.

הצעד הבא הוא לשפר את ה-bid באמצעות למידה חישובית.

ראשית נציין את ה-attributes שייאספו לצורך הלמידה שמתוכם נבחר בשלב ה-training:

- win – משתנה בוליאני המעיד על זכייה או הפסד של Bid נתון בקמפיין המוצע.
- bid – משתנה נומרי המייצג את הצעת הסוכן שלנו עבור הקמפיין.
- bidFor1000Imps - משתנה נומרי המייצג את הצעת הסוכן שלנו עבור הקמפיין ל-1000 impressions. מכיוון שה-bid תלוי באופן משמעותי בכמות ה-impressions, אנו מנטרלים את ההבדל בין קמפיין גדול לקטן על ידי שמירת ה-bid ל-1000 impressions.
- dayOfCampaignOppotunity – משתנה נומרי המייצג את היום בו הוצע הקמפיין למכרז. משתנה זה אמור לתפוס מגמות כמו bids גבוהים יותר בתחילת הקמפיין לעומת סופו, או להיפך.
- campaignLength - מספר ימי הקמפיין. משתנה זה אמור לתפוס מגמות כמו העדפת קמפיינים ארוכים.
- totalReachImps – מספר ה-impressions שיש להשיג עבור הקמפיין. משתנה זה אמור לתפוס מגמות כמו העדפת קמפיינים עם מעט impressions.
- dailyAvgReachImps - מספר ה-impressions הממוצע שיש להשיג ביום עבור הקמפיין. משתנה זה אמור לתפוס מגמות כמו העדפת קמפיינים עם מעט impressions ליום. בשונה מ-totalReachImps , כאן אנו ממצעים את כמות ה-impressions שעלינו להשיג ביחס לאורך הקמפיין, מכיוון שלעיתים קשה להכריע איזה קמפיין עדיף - קמפיין ארוך עם reachImps גבוה או קמפיין קצר עם reachImps נמוך. ממוצע ה-impressions ליום מאפשר השוואה קלה יותר.
- qualityScore – משתנה נומרי המייצג את המוניטין של הסוכן שלנו. ה-quality score של הסוכן שלנו משפיע על סיכויי הזכייה שלנו בקמפיין.
- targetedSegmentAmount - מספר ה-Segments ב-Targeted segment של הקמפיין המוצע. ככל שה-targeted segment יותר ספציפי (targetedSegmentAmount יותר גדול), פחות סוגי impressions מתאימים לו. משתנה זה אמור לתפוס מגמות כמו העדפת קמפיינים עם targeted segment פחות ספציפי.

לכל סוכן x (כולל הסוכן שלנו):

- campaignsNumForAgent[x] - מספר הקמפיינים שבהם זכה הסוכן. משתנה זה אמור לתפוס מגמות כמו: ככל שהסוכן השיג יותר קמפיינים, הרצון שלו לזכות בקמפיינים נוספים נמוך יותר.

- activeCampaignsNumForAgent[x] - מספר הקמפיינים הפעילים של הסוכן (אשר התחילו אך טרם הסתיימו). משתנה זה אמור לתפוס מגמות כמו: ככל שלסוכן יש יותר קמפיינים פעילים, הרצון שלו לזכות בקמפיינים נוספים נמוך יותר.
- maxSegmentSuitabilityForAgent[x] - משתנה נומרי המייצג את רמת ההתאמה המקסימלית בין ה-Targeted segment של הקמפיינים הפעילים של הסוכן לבין ה- Targeted segment של הקמפיין המוצע. משתנה זה אמור לתפוס מגמות כמו: אם לסוכן יש כבר קמפיין עם Targeted segment דומה לקמפיין המוצע, ייתכן שהוא יעדיף לא לקחת על עצמו גם את קמפיין זה.
- mostSuitableActiveCampaignsNumForAgent[x] - מספר הקמפיינים הפעילים שה-Targeted segment שלהם הוא בהתאמה מקסימלית עם ה-Targeted segment של הקמפיין המוצע. משתנה זה אמור לתפוס מגמות כמו: ככל שלסוכן יש יותר קמפיינים פעילים עם Targeted segment דומה לקמפיין המוצע, ייתכן שהרצון שלו לזכות בקמפיין זה נמוך יותר.
- activeCampaignsOverlapForAgent[x] – משתנה נומרי המייצג את גודל החפיפה בימים של הקמפיינים הפעילים של הסוכן לימי הקמפיין המוצע. משתנה זה אמור לתפוס מגמות כמו: ככל שלסוכן יש יותר קמפיינים שחופפים בימים לקמפיין המוצע, ייתכן שהרצון שלו לזכות בקמפיין זה נמוך יותר מחשש שלא יוכל למלא את כולם.

נתונים נוספים שנאספו מתוך ה-server logs:

- bidOfSupposedToBeWinner – משתנה נומרי המייצג את ההצעה שאמורה לזכות במכרז על הקמפיין. ייתכן שהצעה זו לא זכתה במקרה שהקצאת הקמפיין היתה רנדומלית. אנו אוספים את ה-AdNetBidMessage ששולחים הסוכנים ולאחר מכן מבצעים חישוב כפי שמתואר ב-spec של השרת כדי לזהות מהי ההצעה הטובה ביותר.
- supposedToBeWinnerBidFor1000Imps – משתנה נומרי המייצג את ההצעה שאמורה לזכות במכרז על הקמפיין ל-1000 impressions. מכיוון שה-bid תלוי באופן משמעותי בכמות ה-impressions, אנו מנטרלים את ההבדל בין קמפיין גדול לקטן על ידי שמירת ה-bid ל-1000 impressions.
- supposedToBeWinner – שם הסוכן בעל ההצעה שאמורה לזכות.
- chosenAsWinner – שם הסוכן שזכה בקמפיין. לא יהיה זהה ל-supposedToBeWinner רק כאשר הקצאת הקמפיין היא רנדומלית.
- supposedToBeOurWin – משתנה בוליאני שערכו 1 אם ההצעה שלנו היא ההצעה שהיתה אמורה לזכות בקמפיין. שונה מ-win כאשר ההצעה שלנו היתה ההצעה הטובה ביותר, אך הקמפיין הוקצה רנדומלית לסוכן אחר.

איסוף נתונים:

שמרנו את הנתונים מתחרויות שהתנהלו מול הקבוצות האחרות בסדנה, והתקבלו 6001 רשומות של מידע תקין.

Experimentation:

לאחר איסוף הנתונים, הניסיון הראשון שלנו היה לחזות את ה-bid המנצח לקמפיין. מקורס בכלכלה אנו יודעים שלשם חיזוי משתנה נומרי, נחוצה רגרסיה. מחיפוש ב-WEKA החלטנו לנסות את האלגוריתמים linear regression ו-SMO regression (עם kernels שונים).

משתנה המטרה שלנו היה supposedToBeWinnerBidFor1000Imps, מכיוון שרצינו להימנע מהתחשבות בהקצאות הרנדומליות.

הרצנו את אלגוריתמים אלו עם הכנה שונה של הנתונים ועם שילובים שונים של features:

- עם ה-attributes שאספנו על סוכנים אחרים ובלעדיהם – הסיבה שהחלטנו לנסות להשמיט את ה-attributes של הסוכנים האחרים, היא שלא היתה לנו אפשרות לזהות את הסוכנים מסימולציה לסמולציה. כלומר, בתחילת הסדנה חשבנו שהתחרויות תמיד יתבצעו מול אותם סוכנים, וכן שהסוכן שלנו יידע לזהות אותם בתחילת המשחק לפי שמם שגם יישמר ממשחק למשחק. אך, נוכחנו לדעת שאין זה אפשרי, וכי בכל משחק מוקצה שם שונה (adv1, adv2 וכו'). משמעות הדבר מבחינת איסוף הנתונים במהלך המשחק הוא שאנו לא יכולים לייחד לכל סוכן attributes משלו שיאספו מסימולציה לסימולציה. לכן, המידע שנאסף על הסוכנים עלול לא לסייע בגילוי התנהגויות קבועות ואולי אף ייפגע.
- שימוש ב-totalReachImps מול שימוש ב-dailyAvgReachImps - לא נשתמש בשני ה-attributes האלה ביחד, שכן בצירוף campaign length הם יוצרים משוואה לינארית, ולכן אחד מהם מתייתר.
- בחירת attributes בעזרת feature selection – עבור הצירופים השונים של attributes שיצרנו בצורה אינטואיטיבית, החלטנו להיעזר גם ב-feature selection כדי שיכוון אותנו ל-attributes המשפיעים ביותר. השתמשנו גם ב-feature selection מסוג wrapper method וגם ב-PCA.

נציג את התוצאות של כמה מהניסיונות:

1. **ה-features:** dayOfCampaignOppotunity, campaignLength, dailyAvgReachImps, qualityScore, targetedSegmentAmount וכל ה-attributes של הסוכנים.
האלגוריתם: linear regression.

```
=== Evaluation on test split ===
=== Summary ===

Correlation coefficient                  0.4051
Mean absolute error                     21.9667
Root mean squared error                 50.239
Relative absolute error                 98.5793 %
Root relative squared error             91.4989 %
Total Number of Instances              900
```

2. הפעלת אלגוריתם PCA ל-feature selection על דוגמא 1.
ה-features שנבחרו: 27 ה-attributes הראשונים של דוגמא 1.
האלגוריתם: linear regression.

```
=== Evaluation on test split ===
=== Summary ===

Correlation coefficient                  0.4115
Mean absolute error                     21.9107
Root mean squared error                 50.0375
Relative absolute error                 98.3281 %
Root relative squared error             91.1318 %
Total Number of Instances              900
```

3. **ה-features:** dayOfCampaignOppotunity, campaignLength, totalReachImps, qualityScore, targetedSegmentAmount.
האלגוריתם: SMO regression עם Polynomial kernel ועם נרמול אוטומטי של ה- training

.data

```
=== Evaluation on test split ===
=== Summary ===

Correlation coefficient                  0.3734
Mean absolute error                     20.1507
Root mean squared error                 52.4482
Relative absolute error                 90.4296 %
Root relative squared error             95.5225 %
Total Number of Instances              900
```

כפי שניתן לראות, תוצאות הרגרסיה אינן מוצלחות. ה-root relative squared error קרוב מאוד ל-100%, מה שאומר שהחיזוי אינו טוב מהממוצע. לאור התוצאות, החלטנו לנסות להשתמש ב-classifier כדי לחזות זכייה בקמפיין או הפסד.

כאשר הסתכלנו רק על ה-bid שלנו עם תוצאת המכרז לקמפיין, ראינו כי יש הבדל משמעותי מאוד בין מספר הזכיות לבין מספר ההפסדים. מכיוון שאספנו גם את ה-bid שאמור לנצח בכל מכרז, החלטנו להשתמש ב-Bids אלו כדי לאזן בין מספר הזכיות להפסדים ובכך למנוע הטייה של ה-classifier לטובת תיוג הפסד. הכנת ה-data עבור ה-classifier התנהלה כדלקמן:

יצרנו שני attributes חדשים שנקראים TheBidFor1000Imps ו-supposedToBeWinOrLoss שיכילו את ה-bid ל-1000 impressions ואת תוצאת המכרז שהיא זכייה או הפסד. יצרנו שני עותקים של כל ה-data שאספנו לפי ה-attributes שתוארו לעיל.

עותק מספר 1: עבור כל הרשומות העתקנו לתוך TheBidFor1000Imps את הערך supposedToBeWinnerBidFor1000Imps (ה-bid הטוב ביותר ל-1000 impressions שהיה אמור לנצח לולא היו הקצאות אקראיות), וכן הצבנו 1 ב-supposedToBeWinOrLoss שכן אלו תמיד ה-bids המנצחים.

עותק מספר 2: השארנו רק את הרשומות שבהן הפסדנו במכרז לקמפיין. עבור כל הרשומות שנותרו העתקנו לתוך TheBidFor1000Imps את הערך bidFor1000Imps (ה-bid שנתנו עבור הקמפיין ל-1000 impressions), וכן הצבנו 0 ב-supposedToBeWinOrLoss שכן אלו ה-bids המפסידים.

איחדנו את שני העותקים האלו לקובץ אחד, ועשינו randomize על סדר הרשומות כדי שלא יהיו קודם ה-bids המנצחים ואז ה-bids המפסידים במקבצים. קיבלנו 11,271 רשומות מתוכם 6001 הפסדים ו-5270 זכיות.

לסיום הסרנו את ה-attributes הבאים שהפכו למיותרים:

win, bid, bidFor1000Imps , BidOfSupposedToBeWinner, supposedToBeWinnerBidFor1000Imps, ChosenAsWinner, SupposedToBeWinner, supposedToBeOurWin

משתנה המטרה שלנו כעת הוא supposedToBeWinOrLoss. מחיפוש ב-WEKA ובאינטרנט החלטנו לנסות את האלגוריתמים הבאים ל-classification: alternating decision tree, J48, naïve bayes, random forest, random tree, SMO ו-logistic regression (שמחזיר סיכוי לקבל תיוג 1).

הרצנו את אלגוריתמים אלו עם הכנה שונה של הנתונים ועם שילובים שונים של features:

- עם ה-attributes שאספנו על סוכנים אחרים ובלעדיהם.

- שימוש ב-totalReachImps מול שימוש ב-dailyAvgReachImps.
- בחירת attributes בעזרת feature selection.
- עם Normalization וללא – נרמול הנתונים אמור למנוע הטייה של ה-classifier ל-attributes בעלי טווח גדול של ערכים.

נציג את התוצאות של כמה מהניסיונות עבור האלגוריתמים שהניבו תוצאות טובות יותר:

1. **ה-features:** dayOfCampaignOppotunity, campaignLength, dailyAvgReachImps, qualityScore, targetedSegmentAmount, TheBidFor1000Imps וכל ה-attributes של הסוכנים.

האלגוריתם: random forest.

```
=== Summary ===

Correctly Classified Instances        1423               84.1514 %
Incorrectly Classified Instances       268               15.8486 %
Kappa statistic                          0.6831
Mean absolute error                      0.2234
Root mean squared error                  0.3398
Relative absolute error                 44.8345 %
Root relative squared error             68.0482 %
Total Number of Instances             1691

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                   0.864     0.179      0.813     0.864     0.838      0.912     0
                   0.821     0.136      0.87      0.821     0.845      0.912     1
Weighted Avg.      0.842     0.156      0.843     0.842     0.842      0.912

=== Confusion Matrix ===

   a   b   <-- classified as
 692 109 |   a = 0
 159 731 |   b = 1
```

2. **ה-features:** זהים לדוגמא 1, אבל מנורמלים בין 0 ל-1 באמצעות ה-Normalize של WEKA.
האלגוריתם: random forest.

```
=== Summary ===

Correctly Classified Instances        1434               84.8019 %
Incorrectly Classified Instances       257               15.1981 %
Kappa statistic                          0.6962
Mean absolute error                      0.2145
Root mean squared error                  0.3309
Relative absolute error                 43.0584 %
Root relative squared error             66.2613 %
Total Number of Instances             1691

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                   0.873     0.174      0.819     0.873     0.845      0.919     0
                   0.826     0.127      0.878     0.826     0.851      0.919     1
Weighted Avg.      0.848     0.15       0.85      0.848     0.848      0.919

=== Confusion Matrix ===

   a   b   <-- classified as
 699 102 |   a = 0
 155 735 |   b = 1
```

3. הפעלת feature selection מסוג wrapper method על ה-features של דוגמא 2.
ה-features שנבחרו: campaignLength, dailyAvgReachImps, qualityScore, activeCampaignsOverlapForAgent_0, activeCampaignsNumForAgent_1, campaignsNumForAgent_2, activeCampaignsOverlapForAgent_2, TheBidFor1000Imps.
האלגוריתם: random forest.

```
=== Summary ===

Correctly Classified Instances        1547               91.4843 %
Incorrectly Classified Instances       144                8.5157 %
Kappa statistic                          0.8297
Mean absolute error                      0.1202
Root mean squared error                  0.2525
Relative absolute error                 24.1153 %
Root relative squared error             50.556  %
Total Number of Instances             1691

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                   0.938     0.106      0.889     0.938     0.913      0.967     0
                   0.894     0.062      0.941     0.894     0.917      0.967     1
Weighted Avg.      0.915     0.083      0.916     0.915     0.915      0.967

=== Confusion Matrix ===

   a   b   <-- classified as
 751  50 |   a = 0
  94 796 |   b = 1
```

4. **ה-features:** campaignLength, dailyAvgReachImps, qualityScore, TheBidFor1000Imps.

האלגוריתם: J48.

```
=== Summary ===

Correctly Classified Instances        1571               92.9036 %
Incorrectly Classified Instances       120                7.0964 %
Kappa statistic                          0.8579
Mean absolute error                      0.1142
Root mean squared error                  0.2461
Relative absolute error                 22.918  %
Root relative squared error             49.2817 %
Total Number of Instances             1691

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.94      0.081      0.913     0.94      0.926      0.958     0
                 0.919     0.06       0.945     0.919     0.932      0.958     1
Weighted Avg.    0.929     0.07       0.929     0.929     0.929      0.958

=== Confusion Matrix ===

   a   b   <-- classified as
 753  48 |   a = 0
  72 818 |   b = 1
```

5. **ה-features:** campaignLength, totalReachImps, qualityScore, TheBidFor1000Imps.

האלגוריתם: J48.

```
=== Summary ===

Correctly Classified Instances        1574               93.081  %
Incorrectly Classified Instances       117                6.919  %
Kappa statistic                          0.8614
Mean absolute error                      0.1097
Root mean squared error                  0.2439
Relative absolute error                 22.0159 %
Root relative squared error             48.8477 %
Total Number of Instances             1691

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.939     0.076      0.917     0.939     0.928      0.958     0
                 0.924     0.061      0.944     0.924     0.934      0.958     1
Weighted Avg.    0.931     0.068      0.931     0.931     0.931      0.958

=== Confusion Matrix ===

   a   b   <-- classified as
 752  49 |   a = 0
  68 822 |   b = 1
```

התוצאות הטובות ביותר התקבלו מהאלגוריתם J48 על ה-features המתוארים בדוגמא 5. לכן, נשתמש במודל שנוצר על מנת לשפר את ה-bid שמחזיר ה-CampaignBidCalculator. נשים לב שכפי ששיערנו, קיבלנו תוצאות טובות יותר כאשר לא השתמשנו ב-attributes של הסוכנים בשל בעיית איסוף הנתונים.

נשתמש ב-classifier באופן הבא: ניקח את ה-bid שמחזיר ה-CampaignBidCalculator כנקודת מוצא. נגדיר גבול תחתון וגבול עליון עבור ה-bid בהתאם לנקודת המוצא כדי להגדיר את הטווח שמוסכם עלינו במטרה למנוע שינויים קיצוניים ב-bid. על הטווח שנוצר נבצע חיפוש בינארי על מנת למצוא את הנקודה האופטימלית (ה-bid הכי גבוה) שבה אנו מקבלים תיוג win מה-classifier. במקרה שאנו לא מקבלים תיוג win, נבחר להשתמש בגבול התחתון כ-bid.

לבסוף, אנו בודקים שה-bid עומד במגבלות המתוארות ב-spec. אם ה-bid נמוך מדי, נעלה אותו כדי לעמוד בדרישת המינימום.

אסטרטגיה לזכייה ב-Impressions

תחילה החלטנו ליצור כלי שמחשב bid שמבטא את חשיבות ה-impression עבורנו. לשם כך בנינו את המחלקה ImpressionBidAndWeightCalculator שתחשב את ה-bid.

ה-bid ההתחלתי הינו ממוצע ה-bids עבור קמפיין זה עד כה. במקרה שזהו היום הראשון של הקמפיין, נאתחל את ה-bid להיות אחוז מסוים מהתקציב המקסימלי ל-impression בממוצע.

ImpressionBidAndWeightCalculator מתאים את ה-bid לפי הפרמטרים הבאים שמפורטים לעומק ב-design specification:

- התקציב המקסימלי ל-impression בממוצע - שווה ליתרת התקציב חלקי מספר ה-impressions שנותר להשיג. ערך זה מהווה גבול עליון עבור ה-bid.
- אחוזי הצלחה קודמים.
- התחשבות בקמפיינים של סוכנים אחרים.
- Ad type.
- Device.
- עוצמת ה-market segment.

לאחר התנסות במספר תחרויות, ראינו מקרים שבהם quality score נמוך הקשה עלינו לזכות בקמפיינים ואף הוציא אותנו מהמשחק. הבנו שחשוב מאוד להשיג אחוז מסוים מה- reach impressions אף במחיר של פגיעה ב-bank status כדי להימנע מ-quality score שיוציא אותנו מהמשחק. לכן, במקרים שבהם אנו לקראת סוף הקמפיין ונותרו יותר מ-60% מכמות ה-impressions שיש להשיג, נאפשר לחרוג מהתקציב המקסימלי ל-impression בממוצע.

הצעד הבא הוא למצוא אלגוריתם למידה חישובית שיסייע לשיפור ה-bid.

ראשית נציין את ה-attributes שייאספו לצורך הלמידה שמתוכם נבחר בשלב ה-training:

פרמטרים המתארים את סוג ה-impression:

- isMobile – משתנה בוליאני המקבל את הערך 1 כאשר ה-Device של ה-impression הוא Mobile, ואת הערך 0 כשהוא PC.
- isVideo - משתנה בוליאני המקבל את הערך 1 כאשר ה-AdType של ה-impression הוא Video, ואת הערך 0 כשהוא Text.

- isFemale - משתנה בוליאני המקבל את הערך 1 כאשר ה-Gender של ה-impression הוא female, ואת הערך 0 כשהוא male. [1]
- isMale - משתנה בוליאני המקבל את הערך 1 כאשר ה-Gender של ה-impression הוא male, ואת הערך 0 כשהוא female. [1]
- isHighIncome - משתנה בוליאני המקבל את הערך 1 כאשר ה-Income של ה-impression הוא high, ואת הערך 0 כשהוא low.[2]
- isLowIncome - משתנה בוליאני המקבל את הערך 1 כאשר ה-Income של ה-impression הוא low, ואת הערך 0 כשהוא high. [2]
- isYoung - משתנה בוליאני המקבל את הערך 1 כאשר ה-Age של ה-impression הוא young, ואת הערך 0 כשהוא old. [3]
- isOld - משתנה בוליאני המקבל את הערך 1 כאשר ה-Age של ה-impression הוא old, ואת הערך 0 כשהוא young. [3]
- publisher – שם המפרסם.

נתונים נוספים:

- bid – משתנה נומרי המייצג את הצעת הסוכן עבור ה-impression.
- winCountBidCountRatio – כמות ה-impressions התואמים לפרמטרים לעיל שבהם זכינו היום ביחס לאלו שהצענו עליהם במכרז.
- publisherPopularity – מספר הכניסות ל-publisher לפי ה-AdType כפי שניתן ב-PublishersReport.
- reachImps - מספר ה-impressions שיש להשיג עבור הקמפיין.
- impsToGo - מספר ה-impressions שנותר להשיג עבור הקמפיין.
- achievedImpsPrecentageAccordingToAgentLimit – כמות ה-impressions שהשגנו היום ביחס לכמות ה-impressions שביקשנו עבור הקמפיין (מגבלת ה-Impressions ב- bid bundle).
- achievedImpsPrecentageAccordingToCampDemand - כמות ה-impressions שהשגנו עד היום ביחס לכמות ה-impressions שיש להשיג עבור הקמפיין.
- budgetLeft – התקציב שנותר לקמפיין.
- budgetLeftPerImp – התקציב הממוצע שניתן להקצות עבור impression אחד בהתחשב בתקציב שנותר לקמפיין.

לכל סוכן x (כולל הסוכן שלנו):

- activeCampaignsNumForAgent[x] - מספר הקמפיינים הפעילים של הסוכן (אשר התחילו אך טרם הסתיימו). משתנה זה אמור לתפוס מגמות כגון ביקוש רב מצד סוכנים שיש להם קמפיינים רבים.
- maxSegmentSuitabilityForAgent[x] - משתנה נומרי המייצג את רמת ההתאמה המקסימלית בין ה-Targeted segment של הקמפיינים הפעילים של הסוכן לבין ה- market segment של ה-impression. משתנה זה אמור לתפוס מגמות כגון ביקוש רב מצד סוכנים שיש להם קמפיין בעלי התאמה גבוהה ל-Impression הזה.
- mostSuitableActiveCampaignsNumForAgent[x] - מספר הקמפיינים הפעילים שה-Targeted segment שלהם הוא בהתאמה מקסימלית עם ה-Targeted segment של

---

[1] אם isMale וגם isFemale שניהם 0, ניתן להסיק שה-Gender הוא unknown.
[2] אם isHighIncome וגם isLowIncome שניהם 0, ניתן להסיק שה-Income הוא unknown.
[3] אם isYoung וגם isOld שניהם 0, ניתן להסיק שה-Age הוא unknown.

הקמפיין המוצע. משתנה זה אמור לתפוס מגמות כגון ביקוש רב מצד סוכנים שיש להם קמפיינים רבים בעלי התאמה גבוהה ל-Impression הזה.

- daysToGoForAgentWeightedAccordingToSuitability[x] – הימים שנותרו לקמפיינים של הסוכן משוקללים לפי ההתאמה בין ה-Targeted segment של הקמפיינים הפעילים של הסוכן לבין ה-Segment של ה-Impression המוצע. משתנה זה אמור לתפוס מגמות כגון ביקוש רב מצד סוכנים שיש להם קמפיינים בעלי התאמה גבוהה ל-Impression הזה אך מעט זמן לממשם.
- avgDaysToGoWithMaxSuitabilityForAgent[x] – ממוצע הימים שנותרו לקמפיינים של הסוכן שהם בעלי ההתאמה המקסימלית עם ה-Segment של ה-impression המוצע. משתנה זה אמור לתפוס מגמות כגון ביקוש רב מצד סוכנים שיש להם קמפיינים בעלי התאמה מקסימלית ל-Impression הזה אך מעט זמן לממשם.

איסוף נתונים:

שמרנו את הנתונים מתחרויות שהתנהלו מול הקבוצות האחרות בסדנה, והתקבלו 41,943 רשומות של מידע תקין.

Experimentation:

לאחר איסוף הנתונים, הניסיון הראשון שלנו היה לחזות את ה-bid הדרוש לזכייה ב-impression. ניסינו את האלגוריתמים linear regression ו-SMO regression.

משתנה המטרה שלנו היה bid.

הרצנו את אלגוריתמים אלו עם הכנה שונה של הנתונים ועם שילובים שונים של features:

- עם ה-attributes שאספנו על סוכנים אחרים ובלעדיהם.
- בחירת attributes בעזרת feature selection.

התוצאות לא היו מוצלחות באף אחד מהניסויים. לדוגמא:

<u>ה-features:</u> isMobile, isVideo, isFemale, isMale, isHighIncome, isLowIncome, isYoung, isOld, publisherPopularity, impsToGo, achievedImpsPrecentageAccordingToAgentLimit, achievedImpsPrecentageAccordingToCampDemand, winCountBidCountRatio.

<u>האלגוריתם:</u> linear regression.

```
=== Summary ===

Correlation coefficient                  0.2618
Mean absolute error                      0.6826
Root mean squared error                  3.5787
Relative absolute error                120.2741 %
Root relative squared error             96.518  %
Total Number of Instances             6291
```

נציין שהיה חשוב לנו לראות השפעה חזקה של winCountBidCountRation על bid, מכיוון שהוא מעיד על טיב ה-bid. בשל העובדה שלא ראינו השפעה משמעותית, החלטנו להריץ את הרגרסיה רק על רשומות עם winCountBidCountRation גבוה, מה שמבטיח את טיב ה-bid. נותרו 3456 רשומות.

גם כאן התוצאות לא היו מוצלחות באף אחד מהניסויים. לדוגמא:

ה-features: isMobile, isVideo, isFemale, isHighIncome, isYoung, publisherPopularity.
האלגוריתם: SMO regression.

```
=== Summary ===

Correlation coefficient                  0.0955
Mean absolute error                      1.2299
Root mean squared error                 10.3369
Relative absolute error                 56.1116 %
Root relative squared error            100.2984 %
Total Number of Instances              518
```

החלטנו לשנות את משתנה המטרה ל-winCountBidCountRation כדי לראות האם הרגרסיה מצליחה לזהות מתי bid הוא מוצלח.

לצערנו, גם הפעם התוצאות לא היו מספקות. לדוגמא:

ה-features: bid, isMobile, isVideo, isFemale, isMale, isHighIncome, isLowIncome, isYoung, isOld, publisherPopularity, impsToGo, achievedImpsPrecentageAccordingToAgentLimit, achievedImpsPrecentageAccordingToCampDemand וכל ה-attributes של הסוכנים.
האלגוריתם: linear regression.

```
=== Summary ===

Correlation coefficient                  0.3231
Mean absolute error                      0.2218
Root mean squared error                  0.282
Relative absolute error                 90.2141 %
Root relative squared error             94.655  %
Total Number of Instances             6291
```

אנו סבורים שההשפעה של מצב הסוכנים האחרים והקמפיינים שברשותם היא משמעותית מאוד בחיזוי bid טוב, ולכן אנו חוששים שהעובדה שאנו לא יכולים לייחד לכל סוכן attributes משלו (שיאספו מסימולציה לסימולציה) מקשה על בניית רגרסיה מוצלחת. לכן, החלטנו לנסות להריץ רגרסיה רק על נתונים שנאספים מתחילת הסימולציה, כך שנוכל לייחד attributes לכל סוכן. מכיוון שיש מאות רשומות לסימולציה אחת, חשבנו שכמות כזו עשויה להספיק כדי לבנות מודל. בחנו את התוצאות של האלגוריתם LWL שמאפשר לבצע למידה רשומה אחר רשומה, כך שנוכל לעדכן את המודל תוך כדי הסימולציה.

הכנת הנתונים היתה כדלקמן: חילקנו את הקובץ לשניים. בקובץ הראשון היו נתונים מהימים הראשונים של הסימולציה, והם שימשו לבניית המודל. הקובץ השני מכיל את שאר הנתונים ושימש לבדיקת המודל. כך עשינו עבור מספר סימולציות כדי לבדוק האם התוצאות עקביות. אך, רק עבור חלק מהסימולציות המודל נתן חיזוי טוב ל-bid. לדוגמא:

```
=== Summary ===

Correlation coefficient                  0.9073
Mean absolute error                      0.1076
Root mean squared error                  0.1214
Relative absolute error                 16.3634 %
Root relative squared error             18.4555 %
Total Number of Instances             259
```

```
=== Summary ===

Correlation coefficient                  0.1632
Mean absolute error                      0.1781
Root mean squared error                  0.3233
Relative absolute error                 87.4799 %
Root relative squared error            119.8522 %
Total Number of Instances              505
```

בעקבות תוצאות אלו ניסינו להגדיל את מספר הרשומות שנמצאות בקובץ המשמש ללמידה, אך התוצאות עדיין לא היו עקביות בין הסימולציות.

כפי שניתן לראות, אלגוריתמי הלמידה החישובית לא היו מועילים. לכן, החלטנו להסתכל על הבעיה מהפן הכלכלי שלה. חקרנו את הנושא של מכרז מחיר שני, ולמדנו כי לא משנה מהן ההצעות האחרות תמיד יהיה כדאי לכל מציע להציע את השווי האמיתי של המוצר. מבחינתנו, משמעות הדבר היא שתמיד כדאי לנו לתת את ה-bid שמחזיר ה-ImpressionBidAndWeightCalaculator, כי הוא מייצג את חשיבות ה-impression עבורנו. נזכיר שה-ImpressionBidAndWeightCalaculator מחזיר בדרך כלל bid הקרוב ל-bid המקסימלי מבחינתנו (שהוא התקציב המקסימלי ל-impression בממוצע). מכיוון שבפועל אנו משלמים את המחיר השני (שבדרך כלל נמוך מההצעה שלנו), מובטח שיישאר לנו רווח כלשהו מתקציב הקמפיין.

אסטרטגיה להשגת UCS level

כדי לפתור בעיה זו, נשתמש בנתונים הן מסימולציות קודמות והן מהסימולציה הנוכחית.

ראשית נציין את ה-attributes שנאסוף:

- Bid
- UCS Level

איסוף נתונים:

שמרנו את הנתונים מתחרויות שהתנהלו מול הקבוצות האחרות בסדנה, והתקבלו 41,082 רשומות של מידע תקין.

כאמור, הסוכן מחזיק גם את הנתונים שמתקבלים במהלך הסימולציה הנוכחית.

בניית כלי העזר:

בנינו שני כלים להחלטה על UCS bid:

1. UCSBidModel – כפי שמפורט ב-design specification.
2. שימוש באוסף וקטורים מהסימולציה הנוכחית - כפי שמפורט ב-design specification.

כדי להשתמש בכלים אלו, קודם כל יש להחליט על רמת ה-UCS שבה אנו מעוניינים.

האסטרטגיה הראשונה שניסינו היתה להשיג תמיד רמת UCS גבוהה. הסיבה לכך היתה שגם אם אין לנו קמפיינים פעילים כרגע, ייתכן שנזכה ב-campaign opportunity שמוצע באותו יום ונזדקק לשירות הקלאסיפיקציה בעוד יומיים.

במהלך התחרויות שמנו לב שהוצאות ה-UCS שלנו גבוהות מאוד ביחס לאחרים, ופעמים רבות אין שימוש בשירות הקלאסיפיקציה כתוצאה מכך שאין לנו אף קמפיין שדורש impressions.

החלטנו לשנות אסטרטגיה, כך שההחלטה על רמת ה-UCS הרצויה תהיה תלויה בכמות ה-impressions שעלינו להשיג עבור הקמפיינים הקיימים שלנו. ככל שיש כמות גדולה יותר להשיג, נרצה רמת UCS גבוהה יותר. הסיבה לכך היא שככל שיש לנו יותר impressions להשיג, נרצה לדעת בדיוק מה ה-segment של כל Impression מוצע ולא "לבזבז" על impression עם unknown segment. נשים לב שכאשר אין לנו impressions להשיג, נבחר לתת bid 0, למרות שייתכן שנשיג קמפיין שיתחיל ביום שבו ייכנס לתוקף ה-UCS level שיתקבל כתוצאה מה-UCS bid. כדי לפצות על הסיכוי ל-UCS נמוך ביום הראשון, אנו בוחרים קמפיינים שאינם קצרים, אלא אם כן הם בעלי reach impressions נמוך יחסית.