חיזוי טמפרטורה בהינתן היסטוריית מדידות קודמת

פרוייקט בבינה מלאכותית
236502

הפקולטה למדעי המחשב, טכניון

מגישים:

טל פרצ׳וק, 203969522

בן מלצר, 204095038

תוכן עניינים

מבוא

מזג האוויר משפיע על חיינו באופן יום יומי, החל מהשלב המוקדם בבוקר של "מה אלבש היום?" ועד השפעה על "מה אעשה היום?" או "היכן נטייל בחופש?" ואף החלטות הרות גורל, כמו למשל "האם מזג האוויר יפגע ביתרוננו במלחמה?" – כלומר, השפעתו כלל אינה זניחה, וניכר שיש למזג האוויר חלק נרחב ביותר בקבלת ההחלטות של האדם החל מהפשוטות ביותר ועד לכבדות ביותר.

מהו בעצם **מזג האוויר**? *כלל המאפיינים המתארים את מצב האטמוספירה, כאשר רוב התופעות הנבדקות הינן בסמיכות לפני הקרקע*[1].

לכן, נוכל להגדיר: **חיזוי מזג אוויר** *הוא תחום במדעים העוסק בפעילות חיזוי מצב מאפייני האטמוספירה בהינתן מיקום וזמן.*

בני האדם החלו בניסיונות לחזות את מזג האוויר עוד בעת העתיקה, כאשר העדויות הראשונות החלו כבר ב650 לפנה"ס. בתקופה זו הופיעו בתרבויות שונות ניסיונות החיזוי, כאשר כל תרבות ביצעה זאת בצורה שונה, לדוגמא:

- תבניות עננים בשילוב עם אסטרולוגיה.
- צבעי השמיים בערב שלפני מחר.
- ככל שהשקיעה אדומה יותר, היום שלמחרת יהיה נעים יותר.

במרוצת השנים, ככל שהתפתח המדע, ככה שוכללו השיטות. בעת הנוכחית, השיטה המודרנית הראשונה הופיע רק לאחר המצאת הטלגרף[2], וזאת על ידי העברת המידע מאיזור לאיזור באופן מהיר ומיידי, דבר שאפשר שקלול מידע מאיזור נרחב יותר, תוך הבנת התנאים המתפתחים באיזורים רחוקים יותר.

כיום, קיימות מגוון שיטות חיזוי, כאשר השתיים העיקריות הן:

1. <u>חיזוי נומרי</u>: תהליך המאחד תוצאות ממגוון רחב של מודלים מתמטיים בהתבסס על תנאי מזג אוויר עכשוויים. לרוב, כחלק מתהליכים אלה, מחולק העולם למעין סריג תלת ממדי, כאשר בכל תא בסריג מחושב כיצד ישתנו המאפיינים בהינתן פרק זמן כלשהו שנקרא "צעד זמן".
   על מנת לחשב את צעד הזמן הראשון נדרש איסוף מידע מקדים, שעל בסיסו יחושב הצעד וכך הלאה שאר הצעדים.
   מודל מרכזי המשתמש בעיקר בחישובים נומריים לשם חישוב מאפייני מזג אוויר הוא Global Forecast System (GFS) - מודל זה מייצר מאגרי מידע המשמשים מודלים אחרים לחיזוי של עד 16 ימים קדימה, והוא מורכב מארבעה תתי מודלים המחולקים למשימות שונות[3].
2. <u>מודלים מטאורולוגיים</u>: תהליך המבוסס על ניתוח חיזויים נומריים רבים תוך שימוש בנתונים עדכניים של שכבות האטמוספירה וכן הוספת סימולציות רבות שמתבססות על תוצאות המודלים הנומריים.
   שני המודלים המרכזיים היום הינם:
   - המודל האמריקאי: National Weather Service's Global Forecast System (GFS)
   - המודל האירופאי: European Center for Medium-Range Weather Forecast (ECMWF)

נכון לתקופה זו, המודל הנחשב יותר מבין השניים הינו המודל האירופאי, שבאופן ממוצע מציע חיזוי מדויק יותר, ואף היה הראשון לחזות את פגיעת ההוריקן Sandy על אדמת ארצות הברית, כאשר המודל האמריקאי חזה שההוריקן ימשיך לים ולא יפגע בחוף במלוא עוצמתו[4] .

למרות זאת, בעקבות השקעה תקציבית חסרת תקדים לאחר חוסר ההצלחה עם הוריקן סנדי, עדיין קיימים מקרים

---

[1] https://en.wikipedia.org/wiki/Weather

[2] David Hochfelder (1998). "Joseph Henry: Inventor of the Telegraph?". Smithsonian Institution

[3] https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs

[4] https://www.washingtonpost.com/news/capital-weather-gang/wp/2018/05/18/

וסיטואציות בהם המודל האמריקאי עדיף, כמו למשל זיהוי סופות שלגים.
הסיבה לכך שכל אחד מהמודלים הנ״ל נותן תוצאת חיזוי שונה נעוצה בעובדה שכל אחד מהם אוגר את המידע בצורה שונה (ולרוב ע״י לוויינים ותחנות מדידה בעלי טכנולוגיות שונות לחלוטין פרי פיתוח עצמאי של הגוף אליו המודל שייך) וכן בכך שהנוסחאות המתמטיות שבשימוש שונות, ובפרט ההנחות המתמטיות והפיזיות שנעשות על מנת להשתמש בנוסחאות (כמו למשל רמת דיוק עשרוני).

דוגמא מרכזית לחשיבות הדיוק בחיזוי היא השפעתו על הכלכלה. מוערך כי 20% מכלכלת ארה״ב מושפעת באופן ישיר ממזג האוויר, החל בחקלאות ואנרגיה, ועד לתחומי הבנייה, תיירות ובידור.
בשל כך, בארה״ב ובשאר העולם – ובפרט בישראל בה אנו נתמקד, יש ניסיון לאגור כמה שיותר מידע בנושא, ובפרט תיעוד מירבי של מדידות התנאים והמדדים.
עם זאת, רוב המודלים העכשוויים, ובפרט הנ״ל, יודעים לתת תחזית טובה עד כשבוע קדימה, ובאופן די מדויק עד כשניים-שלושה ימים קדימה, וגם זאת תוך עדכון מתמיד של התחזית לפי הימים הקודמים.

מתוך כך, הבנתנו היא שעלינו לבנות מודל שייתן רמת דיוק מירבית, תוך שאיפה שיוכל לעשות זאת על כמה שיותר מאפייני מזג אוויר.
בנוסף, בשל ההבנה כי מדידות היום הבא משפיעות מאד על הימים שלאחר מכן, ריכזנו את מירב מאמצינו על חיזוי היום הבא עבור תאריך מסוים ונתוני העבר.

לסיכום, הבעיה אותה ניסינו לפתור היא:

<u>בהינתן יום מסויים, מה תהיה הטמפרטורה באיזור תחנת מדידה ספציפית כלשהי, בהינתן היסטוריית מדידות</u>

*במאמר מוסגר, כדי למקד את הפתרון שלנו, ניתן את איזור תחנת הטכניון כדוגמא, ועליה נבסס את תוצאות הניסויים בחלק המתודולוגיה הניסויית.

תיאור פתרון הבעיה

השיטה בה ננסה לפתור בעיה זו, תוך ניסיון מרבי להגיע לתוצאות מדויקות, תהיה למידת מכונה.
הסיבה בה בחרנו בשיטה זו נובעת בעיקר מהעובדה שנתוני מאפייני מזג האוויר הם מספריים, ולכן בעזרת תהליך למידה ניתן יהיה ללמוד בעזרת נתונים היסטוריים מה ניתן לצפות שיקרה ביום נתון בהינתן הימים שלפניו – ולסווג את הטמפרטורה של היום הבא בהתאם. שיטה הסיווג (או הרגרסיה) הינה קלאסית לבעיה זו. על מנת להגיע לפתרון מיוחל, חילקנו את הפרויקט למספר שלבים:

1. קבלת רקע כללי על חיזוי מזג אוויר

בתור התחלה, כיוון שהידע שהיה לנו בכל הנוגע למזג אוויר היה כללי ביותר היינו צריכים להעשיר את הידע בתחום זה.
השלב הראשוני היה להבין מהן ההשפעות של המאפיינים השונים אחד על השני. נהוג להתייחס למאפיינים הבאים כמאפיינים העיקריים: טמפרטורה ממוצעת, מקסימלית ומינימלית, טמפרטורה בקרבת הקרקע, לחות, מהירות רוח וקרינה.
כפי שהוסבר במבוא, למדנו שיש בשימוש שתי שיטות מרכזיות לחיזוי – מודלים נומריים ומודלים מטאורולוגיים. לאחר מחקר שעשינו עבור כל אחת מהשיטות, שבסיסו היה לאפיין מהן הדרישות לכל צורת חיזוי ומה נדרש לעשות כדי להגיע לפתרון בעזרת השיטה הנבדקת, הגענו למספר תובנות ראשוניות הקשורות לדרישות הבסיסיות ביותר לכל אחת מצורות המידול:

i. מידול מטאורולוגי:

- מידול דינאמי ובזמן אמת של שכבות האטמוספירה, ובפרט: תנועתה ומידע תרמודינאמי. כלומר, הוצאת מידע חזותי רב ממאגרי המידע ושימוש בטכניקות עיבוד תמונה מתקדמות[5].
- ניתוח פיזיולוגיה בזמן אמת של כלל הנתונים שהתקבלו על ידי פיתוח נוסחאות מתמטיות מסובכות[6].
- עדכון התחזית בזמן אמת בהתבסס על מידע שפוענח, תוך עדכונה מדי מספר שעות.
- כוח עיבוד גדול ביותר (סדרי גודל של מחשבי-על) על מנת להצליח לעבד את כלל המידע באופן כזה שאפשר יהיה להשתמש בו ולבדוק את המודל בזמן סביר.

ii. מידול נומרי:

- משתמש בנתוני עבר רבים ככל הניתן, תוך מתן דגש על שימוש בנתונים מאומתים.
- חלוקת תא השטח לאזורים (סוג של סריג).
- הגדרת צעד זמן.
- חישוב מאפייני מזג האוויר באופן מדויק ככל הניתן.

---

[5] http://weather.ou.edu/~scavallo/classes/metr_5004/f2013/lectures/NWP_LecturesFall2013.pdf

[6] https://www.ecmwf.int/en/research/modelling-and-prediction

דוגמא למורכבות הנוסחאות הנ"ל[4]:

Equations of motion (ECWMF model)

$$\frac{\partial U}{\partial t} + \frac{1}{a\cos^2\theta}\left\{U\frac{\partial U}{\partial \lambda} + v\cos\theta\frac{\partial U}{\partial \theta}\right\} + \dot{\eta}\frac{\partial U}{\partial \eta}$$

$$(-fv) + \frac{1}{a}\left\{\frac{\partial \phi}{\partial \lambda} + R_{dry}T_v\frac{\partial}{\partial \lambda}(\ln p)\right\} = P_U + K_U$$

East-west wind

$$\frac{\partial V}{\partial t} + \frac{1}{a\cos^2\theta}\left\{U\frac{\partial V}{\partial \lambda} + V\cos\theta\frac{\partial V}{\partial \theta} + \sin\theta(U^2+V^2)\right\} + \dot{\eta}\frac{\partial V}{\partial \eta}$$

$$+ fU + \frac{\cos\theta}{a}\left\{\frac{\partial \phi}{\partial \theta} + R_{dry}T_v\frac{\partial}{\partial \theta}(\ln p)\right\} = P_V + K_V$$

North-south wind

$$\frac{\partial T}{\partial t} + \frac{1}{a\cos^2\theta}\left\{U\frac{\partial T}{\partial \theta} + V\cos\theta\frac{\partial T}{\partial \theta}\right\} + \dot{\eta}\frac{\partial T}{\partial \eta} - \frac{\kappa T_v\omega}{(1+(\delta-1)q)p} = P_T + K_T$$

Temperature

$$\frac{\partial q}{\partial t} = \frac{1}{a\cos^2\theta}\left\{U\frac{\partial q}{\partial \lambda} + V\cos\theta\frac{\partial q}{\partial \theta}\right\} = \eta\frac{\partial q}{\partial \eta} = P_q + K_q$$

Humidity

$$\frac{\partial}{\partial t}\left(\frac{\partial p}{\partial \eta}\right) + \nabla.\left(\mathbf{v}_H\frac{\partial p}{\partial \eta}\right) + \frac{\partial}{\partial \eta}\left(\dot{\eta}\frac{\partial p}{\partial \eta}\right) = 0$$

Continuity of mass

$$\frac{\partial p_{surf}}{\partial t} = -\int_0^1 \nabla.\left(\mathbf{v}_H\frac{\partial p}{\partial \eta}\right)d\eta$$

Surface pressure

2. <u>סוג המסווג</u>

כבר לפני איסוף נתונים וקבלת החלטות על כיווני מחקר שונים, הבנו שעלינו להחליט באיזו שיטת סיווג להשתמש, וזאת על מנת שכאשר נאסוף מידע ונרצה לסדרו לקראת עיבוד, נרצה לדעת מהם הערכים עליהם נרצה לעבוד ומהן סוג התשובות אותן נרצה לתת. ההתלבטות הייתה בין שתי אפשרויות:

i. Classifier: מסווג בינארי העונה ב"כן" או "לא". במקרה של הבעיה שלפתחנו יעבוד כך שעבור כל מאפיין שנבדק יענה לדוגמא על שאלת הטמפרטורה "יהיה חם" / "יהיה קר" או על שאלת הגשם ב"יהיה גשם" / "לא יהיה גשם".

ii. Linear Regressor: משמש לשערוך ערך מספרי רציף.

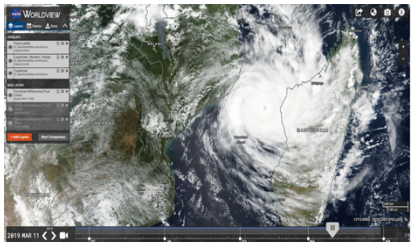
מתוך הבנה שנתוני מזג אוויר הינם ערכים מספריים ממשיים, כך שערכם יכול להשתנות באופן דרסטי מתקופה לתקופה, וכן הם ערכים רציפים, ברור היה לנו שהדרך הנכונה תהיה להשתמש ב- Regressor.

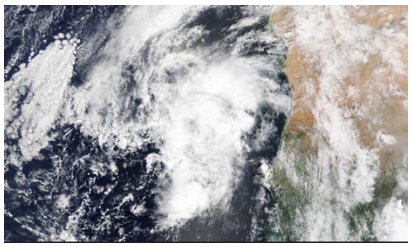
3. <u>איסוף נתונים</u>

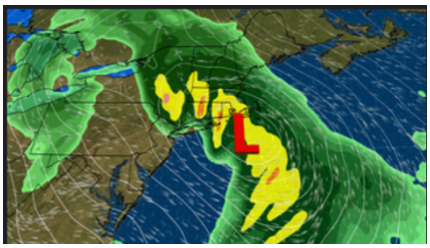
לאחר שהבנו שנרצה להשתמש ב- Regressor, התקבלה החלטה לחפש מאגרי נתונים המכילים כמה שיותר מידע רציף. עם זאת, עד כמה שהדבר נשמע פשוט – מסתבר שיש סוגים רבים של מידע שבו משתמשים לצורך חיזוי מזג אוויר.

בשלב הראשוני אספנו מספר מאגרי מידע המכילים נתונים שעלולים לתרום לנו, מתוך ההבנה שיש קשר בין תופעות טבע למדידות:

- מדידות רעידות אדמה שנערכו על ידי הגיאופיסי לישראל.
- מדידות נתוני מזג אוויר שנערכו על ידי השירות המטאורולוגי.
  https://ims.data.gov.il/he/node/46
- נתוני לוויין ומפות עננות גלובאליים ובפרט נתונים ויזואליים על שכבות האטמוספירה ותנועתה (NASA GIBS)
- אתרים שונים העוסקים בתחום מזג האוויר כמו לדוגמא https://www.wunderground.com







זוהי בעצם הנקודה השנייה שבה להחלטות שנעשה תהיה השפעה על המשך הפרויקט. לאחר התעמקות בכלל מאגרי המידע שאספנו, חקירת היתרונות והחסרונות בשימוש בכל מאגר, ניסיונות איחוד ושימוש במספר מאגרים שונים, ולמרות הרצון הגדול להימשך לתחומי ה - Vision Neural Networks, החלטנו להיצמד לנתונים שמצאנו בעזרת השירות המטאורולוגי, היות והיו מקיפים (מאגרי מידע אחרים הניבו בעיקר כפילויות), ומספריים.
לשמחתנו השירות המטאורולוגי אסף מידע רב החל מקום המדינה ועד היום. ישנם מספר סוגי של מאגרי מידע – דקותי, עשר דקותי והיסטורי.
המידע מחולק לפי תחנות מדידה, כאשר לכל תחנת מדידה 16 פיצ׳רים.
החלטנו שמידע דקותי עלול להיות עמוס מדי בצורה שלא תוסיף הרבה למידע העשר דקותי, ובנוסף המידע ההיסטורי הכיל ״חורים״ וחוסרי מידע לא קבועים, ולכן בחרנו במידע העשר דקותי.
לאחר מעבר על התחנות ובדיקה איזה פיצ׳רים בהן רלוונטיים ונתוניהם מהימנים, החלנו לצבור מספר דוגמאות מדגמי לפי תחנות נקודתיות, בתחילה עבור מספר ימים בודד, ולאחר מכן חודשים ואף שנים.
בהמשך אף אספנו את כלל הנתונים מכל התחנות בישראל למשך כמעט 4 שנים .
בכך בעצם יצרנו מאגר מידע גדול של נתוני אמת של מדידות פיצ׳רים שונים מכל רחבי הארץ – דבר שבהמשך יאפשר לנו לחקור כיוונים שונים והשפעות שונות על התוצאות.

4. עיבוד הדוגמאות

לאחר איסוף הדוגמאות החל שלב הסידור שלהן על מנת שנוכל לעבוד בצורה נוחה ויעילה.
את כלל המידע שקיבלנו בעזרת הAPI הייעודי של השירות המטאורולוגי קיבלנו בפורמט Json, ובאופן כזה שלא מאפשר עבודה נוחה ויעילה למול מאגד הנתונים.
כאשר התחלנו את התהליך המידע נראה כך:


```
{"stationId": 43, "data": [{"datetime": "2019-08-03T00:10:00+03:00", "channels": [{"id": 1, "name": "WSmax", "alias": null, "value": 3.2, "status": 2,
"valid": false, "description": null}, {"id": 2, "name": "WDmax", "alias": null, "value": 264.0, "status": 2, "valid": false, "description": null},
{"id": 3, "name": "WS", "alias": null, "value": 1.6, "status": 2, "valid": false, "description": null}, {"id": 4, "name": "WD", "alias": null, "value":
267.0, "status": 2, "valid": false, "description": null}, {"id": 5, "name": "STDwd", "alias": null, "value": 13.7, "status": 2, "valid": false,
"description": null}, {"id": 6, "name": "TD", "alias": null, "value": 24.8, "status": 1, "valid": true, "description": null}, {"id": 7, "name": "RH",
"alias": null, "value": 80.0, "status": 1, "valid": true, "description": null}, {"id": 8, "name": "TDmax", "alias": null, "value": 24.9, "status": 1,
"valid": true, "description": null}, {"id": 9, "name": "TDmin", "alias": null, "value": 24.8, "status": 1, "valid": true, "description": null}, {"id":
10, "name": "Grad", "alias": null, "value": 0.0, "status": 1, "valid": true, "description": null}, {"id": 11, "name": "NIP", "alias": null, "value":
2.0, "status": 1, "valid": true, "description": null}, {"id": 12, "name": "DiffR", "alias": null, "value": 0.0, "status": 1, "valid": true,
"description": null}, {"id": 13, "name": "WS1mm", "alias": null, "value": 1.9, "status": 2, "valid": false, "description": null}, {"id": 15, "name":
"Ws10mm", "alias": null, "value": 1.6, "status": 2, "valid": false, "description": null}, {"id": 16, "name": "Time", "alias": null, "value": 8.0,
"status": 2, "valid": false, "description": null}, {"id": 20, "name": "Rain", "alias": null, "value": 0.0, "status": 1, "valid": true, "description":
null}]}, {"datetime": "2019-08-03T00:20:00+03:00", "channels": [{"id": 1, "name": "WSmax", "alias": null, "value": 2.7, "status": 2, "valid": false,
```


כפי שניתן לראות הרבה מאד מהנתונים כלל לא היו רלוונטיים – ולכן עיבדנו את קבצי המקור כך שמדידה עבור תחנה כלשהי תראה כך:

	DiffR	Grad	NIP	RH	Rain	STDwd	TD	TDmax	TDmin	TG	Time	WD	WDmax	WS	WS1mm	WSmax	Ws10mm
2019-07-01T00:10:00+03:00	[0.0]	[0.0]	[nan]	[44.0]	[0.0]	[10.2]	[30.6]	[30.6]	[30.6]	[30.1]	[10.0]	[349.0]	[355.0]	[2.6]	[3.7]	[4.4]	[2.6]

כפי שניתן לראות כבר מהשורה הראשונה, ישנן מספר החלטות שהיינו צריכים לקבל:

i. האם נרצה לעבד מידע עבור יום שלם או לפי שעות?
ii. כיצד נתמודד עם מידע חסר, כמו למשל בפיצ׳ר NIP שערכו nan, דבר שנוצר בגלל חוסר מידע של תחנת המדידה או מידע לא תקין.

החלטנו שמכיוון שמעניין אותנו להגיע לתוצאות עבור יום שלם, נמצע את הנתונים שהתקבלו לאורך כל היום – כלומר מנתונים עבור כל 10 דקות, נמצע את היום כולו ונייצג כל יום על ידי שורה אחת.
בנוסף, החלטנו שבמקרה של מידע חסר עבור פיצ׳ר מסוים נבצע ממוצע על המידע הקרוב ביותר מבחינת זמנים ונשתמש בו (בהמשך נסביר כיצד התמודדנו עם חוסרי מידע)
כעת לאחר קבלת ההחלטה הנ״ל, כל שנותר הוא להציג מדידה עבור יום שלם באופן הבא:

	WSmax	WDmax	WS	WD	STDwd	TD	RH	TDmax	TDmin	Grad	NIP	DiffR	WS1mm	Ws10mm	Time	Rain
2019-07-01	4.855944055944050	145.26573426573400	2.691608391608390	149.5034965034970	17.87622377622380	32.76643356643360	33.47552447552450	32.94055944055940	32.60559440559440	346.56643356643400		59.50349650349650	3.7384615384615400	2.9209790209790200	1174.2517482517500	0.0

השלב הבא היה ליישם עיקרון שנתקלנו בו בשלב למידת נושא מזג האוויר והוא שיש מקרים בהם כיוון הרוח במעלות אינו משנה – ועדיף לפצל ערך זה לשמונה כיווני רוח במקום.
השיטה בה עשינו את זה נקראת One Hot Encoder (כפי שתוסבר בהמשך) כך שבעצם לא נשתמש עוד

בפיצ׳ר WD, אלא נשתמש בערך בינארי עבור כל אחד מהפיצ׳רים הבאים:
צפון, דרום, מזרח, מערב, צפון-מזרח, צפון-מערב, דרום-מזרח, דרום-מערב – כך שערכם יהיה 1 אם כיוון הרוח זהה להם, ואחרת 0. כלומר, עמודת כיוון הרוח נמחקה, ובמקומה הופיעו העמודות הבאות:

North-West	South-East	South-West	South	West	North-East	North	East
0	0	1	0	0	0	0	0

5. בניית רגרסורים ולמידה

לאחר איסוף כל המידע משלל התחנות ברחבי הארץ, הבנו כי ניתן להתמודד עם בעיית החיזוי הנ"ל בשתי דרכים שונות. בהינתן ניסיון חיזוי טמפרטורה באיזור תחנת הטכניון:

i. נשתמש במידע הנאסף מתחנת הטכניון בלבד לצורך בניית סט המידע בעזרתו נאמן ונבחן את המודל שלנו.
ii. נשתמש במידע הנאסף מכל תחנות המדידה בארץ לצורך בניית סט המידע בעזרתו נאמן ונבחן את המודל שלנו.

בשלב זה בנינו רגרסור עבור סט המידע כפי שסידרנו אותו בשלב הקודם, וזאת במטרה לתת עבור תאריך מסוים תחזית בהינתן הימים שקדמו לו.
חילקנו את בניית הרגרסור לשתי אפשרויות:

i. <u>רגרסור שונה לכל תחנה</u>: מטרתו לבדוק את איכות הלמידה והחיזוי בהינתן מידע השייך לתחנה ספציפית, כך שתהליך הלמידה אותו נבדוק יתבצע על מידע הבנוי ממאפיינים הספציפיים לתחנה זו בתקופת הזמן הנבדקת.
ii. <u>רגרסור מאוחד לכלל תחנות המדידה</u>: מטרתו לבדוק את איכות הלמידה בהינתן מידע השייך לכלל התחנות, והשפעתו של מידע x על תחנה y, כך שתהליך הלמידה אותו נבדוק יתבצע על מידע הבנוי ממאפייני כלל התחנות, והבדיקה תהיה עבור תחנה ספציפית כחלק מסט המידע הכולל.

תהליך הבנייה של הרגרסור עצמו היה זהה מבחינת אלגוריתם עבור כל אחת מהאפשרויות, כאשר אנו בודקים מספר רגרסורים מוכרים שיפורטו בהמשך הדו״ח.
כל מסווג נבדק על פרמטרים שונים ובשיטות שונות שכללו:

i. הוספת הפיצ׳רים של n הימים הקודמים כפיצ׳ר עבור יום כלשהו עבור n משתנה, וזאת במטרה לבדוק השפעה נוספת של ימים קודמים על חיזוי יום כלשהו.
ii. השמטת פיצ׳רים שרמת הקורלציה הלינארית שלהם לפיצ׳ר מרכזי שנבדק נמוכה מערך c משתנה.
iii. בחירת k הפיצ׳רים הטובים ביותר לפי שיטת Select K Best.
iv. חילוק המידע לסט אימון וסט מבחן לפי שיטת K Cross Validation עבור K=10, וזאת במטרה לבדוק את רמת השגיאה עבור ימים שאינם מתויגים של התהליך עבור מספר משתנה של דוגמאות אימון.

את השיטות הנ״ל שילבנו על מנת לקבל מסווג אידאלי, מתוך מטרה לקבל כמה שיותר מידע על רלוונטיות הפיצ׳רים ביחס אחד לשני, וכן השפעתם על פיצ׳ר מרכזי (כאשר פעולה זו נבדקת לכל פיצ׳ר שהוא כפי שהוגדרו במאפייני מזג אוויר), הטמפרטורה, לשם מציאת הערכים האופטימליים שיניבו רגרסור הנותן שגיאת פרדיקציה נמוכה ביותר – והוא ישמש כרגרסור האידאלי עבור התחנה במידה ונבנה לפי מידע של תחנה יחידה, או במידה ובודקים את כלל התחנות יחדיו.
בסופו של דבר, המסווג ה״כולל״ ישתמש במסווג האידאלי לתחנה הספציפית שעבורה בודקים את מזג האוויר.

תיאור המערכת

על מנת לחזות טמפרטורה, בנינו בשפת Python מערכת הבנויה ממספר סקריפטים וכן מספר API's, כאשר לכל אחד מהם תפקיד אחר במערכת, ומטרתם היא ביצוע השלבים כפי שתיארנו בחלק הקודם.
עיקר פעולת המערכת הינו יצירת קבצי מידע ואימון הרגרסורים עליהם, כאשר הקבצים שונים זה מזה במספר התכונות אותם הם מכילים, סוג התכונות והקשר ביניהן.

1. איסוף נתונים

על מנת לאסוף את הנתונים השונים ממאגרי המידע של השירות המטאורולוגי יצרנו ממשק נוח לגישה שהתבסס על ספריית requests של פייתון, כך שנוכל למשוך מידע בהתאם לצרכים הבאים:

1. מס׳ תחנה.
2. פיצ׳ר ספציפי.
3. טווח תאריכים משתנה (לפי: טווח בין ימים ספציפיים, טווח בין שני חודשים נתונים, חודש נוכחי, יום נוכחי).
4. שמירה בקובץ לצורך גיבוי ושמירת סדר בנתונים.

2. בניית סט מידע נוח לעבודה

לאחר איסוף המידע רצינו להגדיר ממשק שבעזרתו נוכל לבצע מס׳ פעולות שיחזור על עצמו לאורך הפרוייקט בצורה פשוטה וקבועה:

1. יצירת Dataset עבור קובץ נתונים כלשהו שנאסף בשלב הקודם.
ביצענו פעולה זו ע״י פירוק קובץ המידע לפי פיצ׳רים, כאשר האינדקס הוא השעה שבה נאספה שורת המידע. כאשר היו ברשותנו כלל השורות העדכניות לפי אינדקס שעת מדידה, מיצענו ואיחדנו את כלל ה - Datasets המייצגים שעת מדידה לשורה שהאינדקס שלה הוא התאריך הנבדק. לאחר מכן ביצענו המרה של פיצ׳ר כיוון הרוח (WD).
את התהליך הנ״ל ביצענו לכל יום שנמצא בטווח התאריכים שבקובץ הנתונים, ולבסוף יצרנו Dataset שהוא איחוד כל העיבודים הנ״ל, כך שהאינדקסים שלו הם התאריכים.
לדוגמא, עבור התאריכים 1.4.2019-7.4.2019 קיבלנו את התוצאה הבאה:

	WSmax	WDmax	WS	STDwd	TD	RH	TDmax	TDmin	Grad	NIP	DiffR	WS1mm	Ws10mm	Time	Rain	North-West	South-East	South-West	South	West	North-East	North	East
2019-04-01	7.90	258.99	4.48	17.67	7.88	97.15	7.97	7.81	68.46	8.38	52.85	6.28	4.87	1182.76	0.03	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
2019-04-02	5.16	244.01	3.17	15.26	10.09	77.36	10.21	9.98	145.86	33.95	105.97	4.24	3.44	1165.05	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
2019-04-03	2.96	186.21	1.63	15.93	12.67	57.07	12.91	12.45	297.13	136.06	75.67	2.39	1.87	1182.60	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
2019-04-04	3.44	221.10	2.05	18.98	13.78	66.74	14.02	13.55	204.67	0.00	103.26	2.82	2.31	1182.29	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00
2019-04-05	4.94	297.80	3.01	15.14	11.45	86.51	11.65	11.29	248.26	79.63	113.39	4.01	3.29	1181.72	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
2019-04-06	3.40	145.62	1.94	16.58	14.56	62.11	14.78	14.39	179.21	76.87	104.27	2.78	2.17	1182.61	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2019-04-07	4.20	227.86	2.52	12.65	18.65	46.87	18.83	18.47	292.42	309.69	69.31	3.43	2.77	1180.73	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00

*לשם הצגה פשוטה של הטבלה שינינו את ייצוג המספרים לשתי ספרות עשרוניות, בעוד במקור המספר הוא מדויק יותר

3. עיבוד המידע והרחבת הפיצ׳רים

שלב זה התרכז ביחסים בין הפיצ׳רים השונים ובניסיונות הרחבה שלהם, וכולל את הפעולות הבאות:

1. מציאת קורלציה בין פיצ׳רים.
תודות לפונקציות מובנות בספריית pandas, על ה-Dataset שיצרנו בדקנו מהי הקורלציה בין הפיצ׳רים השונים לבין תיוג הטמפרטורה מחר. במידה והקורלציה הייתה רחוקה מדי התעלמנו מפיצ׳ר זה מתוך הבנה שהוא לאו דווקא בעל התאמה לינארית, לכן עלול לפגוע בתהליך הלמידה.
הסיבה בגללה החלטנו להתמקד בתוצאות לינאריות נעוצה בעובדה שכאשר התחלנו לחקור את הנושא ומתוך ניסיון להבין מהי הקורלציה בין המאפיינים השונים ביחס לטמפרטורה השתמשנו ב-Corr מספריית Pandas.
2. שימוש בערכי הפיצ׳רים של ימים קודמים כפיצ׳רים עבור הימים הבאים לשם בדיקת שיפור התוצאות בהינתן תלות בין ימים (יוסבר בהמשך בחלק המתודולוגיה הניסויית).

4. שלב הלמידה

שלב זה התחלק למספר תתי שלבים וזאת על מנת למצוא מהם ההגדרות האופטימליות לכל רגרסור, ולאחר מכן מי מהרגרסורים שנבדקו עדיף.
את כלל הרגרסורים בהם השתמשנו לקחנו מספריית scikit-learn. לכל אחד מהרגרסורים ביצענו את הפעולות הבאות:

1. מציאת תכונות אידאליות עבור הרגרסיות השונות:
פעולה זו בוצעה ע״י האלגוריתם הבא עבור Dataset כלשהו:
1.1. עבור הוספת x ימים אחורה כפיצ׳רים כאשר x נבחר מתוך [0,5,10,20,25,30]:
1.1.1. לכל מתאם קורלציה corr בתחום [0,0.5] בהפרשים של 0.1:
1.1.1.1. הוסף עמודה עבור TD_tommorow שתייצג את החיזוי ליום המחר.
1.1.1.2. מצא מהם המאפיינים (כלומר אילו עמודות) מקיימים שהקורלציה בינם לבין TD_tommorow גדולה בערך מוחלט מcorr – והשאר רק אותם כעמודות הDataset.
1.1.1.3. לכל k חשב את k הפיצ׳רים הטובים ביותר לפי שיטת select_k_best כאשר k נבחר מתוך [5,20,40,60,80,100]
1.1.1.3.1. לכל reg מתוך [Ridge,Lasso,ElasticNet,SVR,MLP,RandomForest]:
1.1.1.3.1.1. בצע חלוקה של הDataset לסט אימון וסט מבחן לפי שיטת k_fold כאשר k=10.
1.1.1.3.1.2. לכל חלוקה שהתקבלה בצע אימון של reg על סט האימון ולאחר מכן בחן את טיב ההצלחה על סט המבחן וחשב את השגיאה הממוצעת האבסולוטית במעלות.

2. מציאת הפרמטרים העיקריים עבור הרגרסיות השונות
2.1. עבור כל רגרסיה:
2.1.1. בנה אובייקט מסוג Grid Search CV.
2.1.2. אמן את האובייקט בעזרת סט המידע והפרמטרים המתאימים וחזה תוצאות בהתאם.
2.1.3. שמור את התוצאות בקובץ המתאים.

לאחר ביצוע שני השלבים הנ״ל, ולאחר בדיקה מיהו המסווג הטוב ביותר ותחת אילו פרמטרים, עבור כל תחנה נשמור את המסווג הטוב ביותר שהתקבל, ובאיזה סט מידע להשתמש – כלומר זה שהחזיר שגיאה אבסולוטית ממוצעת מינימלית.

5. חיזוי טמפרטורה

שלב זה הינו יחסית פשוט, וכולל הרצת המסווג עבור היום המבוקש והחזרת ערך הטמפרטורה ביום זה.
כדי להריץ את המסווג ביצענו הכנה מראש:

1. בעזרת התוצאות מהניסויים בנינו את סט המידע המתאים ואת המסווג המתאים, לחיזוי מיטבי באיזור תחנת הטכניון.
2. את המסווג שמרנו בקובץ joblib לאחר שלימדנו אותו 80% מסט המידע המתאים (ע"י שימוש ב – train_test_split כאשר test_size = 0.2). את סט המבחן ואת התוצאות האמיתיות לסט המבחן, שמרנו בקבצי csv במטרה להדפיס את התוצאות.

כדי לבחון את המסווג יש להריץ את קובץ הבדיקה באופן הבא: python PredictTempTechnion.py וזאת לאחר שיותקנו כל הספריות הנדרשות מהקובץ libs.txt התוצאות יודפסו על המסך.

מתודולוגיה ניסויית

כחלק מפרק זה נתמקד במספר גורמים:

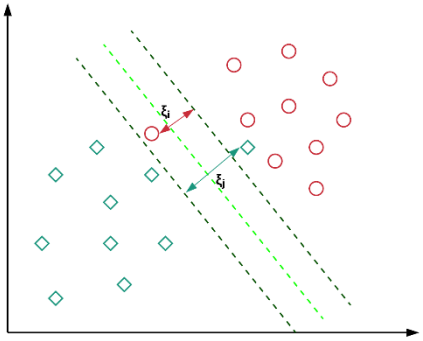
1. המסווגים אותם נבדוק בניסויים:

במסגרת פרק זה החלטנו לבחור מספר רגרסורים שונים הפועלים בשיטות שונות, וזאת על מנת לבחון האם שיטת חישוב כלשהי עדיפה על האחרת – או שמא בהינתן הנתונים גם הרגרסור הפשוטה ביותר תספיק.

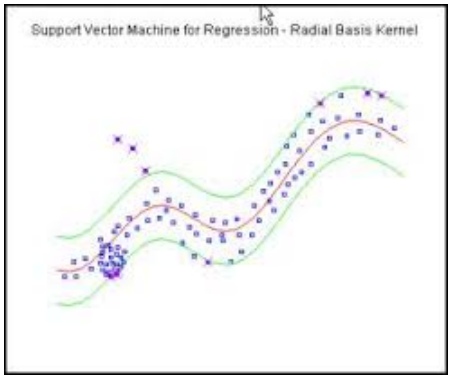
SVR- Support Vector Regression:

מסווג מסוג זה מפריד את דוגמאות, אך בשונה ממסווגים אחרים, מנסה למקסם את המרחק בינו לבין הדוגמאות בעלות תיוגים שונים, הקרובות ביותר אליו, ובכך ממקסם את הפעם בין שתי הדוגמאות הקרובות ביותר בעלות תיוג שונה. למסווג זה מספר פרמטרים אותם ניתן לעדכן אך אנחנו נעסוק בשניים:

- **kernel** – משתנה זה מקבל מחרוזת המייצגת את סוג ההפרדה שהאלגוריתם יבצע. ישנן 4 אופציות:
  - **Linear** – האלגוריתם מפריד בין הדוגמאות בעזרת קו לינארי.



  - **Polynomial** – המפריד משתמש בפונקציה פולינומית להפרדה הדוגמאות

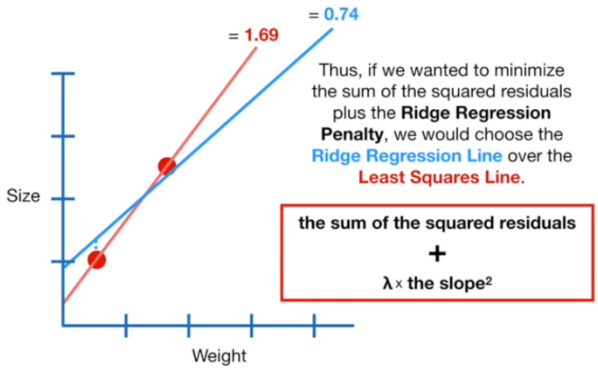


  - **RBF** – המפריד "מייצר" תכונות חדשות עבור ה- dataframe, ע"י שימוש בפונקציית מרחק בין דוגמא ספציפית לשאר הדוגמאות האחרות. בניסוי זה נשתמש בפונקציית גאוס (הפופולארית ביותר בהקשר של RBF).

  - **Sigmoid** – הפרדה ע"י שימוש בפונקציית tanh.
- **C** – משתנה זה מגדיר את מידת הסבילות, tolerance, כאשר המסווג שלנו טועה בסיווג עבור דוגמא מסויימת. ערך C גדול יותר מגדיל את הקנס עבור תיוג לא נכון של דוגמא מסויימת.

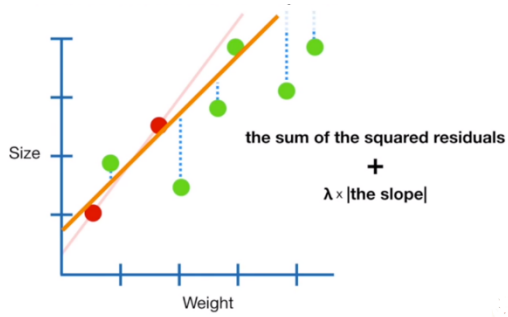
Ridge Regression:

מפריד את הדוגמאות ע"י פוניקציה לינארית, אך בשונה ממפריד לינארי רגיל, המפריד הנ"ל ממזער פונקצייה שונה מהפונקצייה אותה ממזער מפריד לינארי פשוט – בעוד מפריד פשוט מנסה למצוא מינימום לבעיית הריבועים הפחותים בניסיון מציאת הפונקציה המפרידה, מפריד Ridge ממזער אותה בעייה עם תוספת שהיא מכפלת סכום הריבועים של מקדמי הפונקציה במשתנה alpha – hyper parameter שניתן לשנות. נשתמש ב- cross-validation כדי להכריע מהו ערך ה- alpha המיטבי.



Lasso Regression:

מפריד את הדוגמאות ע"י פונקציה לינארית אך בשונה ממפריד לינארי רגיל, ובדומה למפריד Ridge, הפונקצייה אותה מנסה למזער היא בעיית הריבועים הפחותים בתוספת מכפלת סכום הערכים המוחלטים של מקדמי הפונקציה במשתנה alpha – hyper parameter אותו ניתן לשנות. נשתמש ב- cross-validation כדי להכריע מהו ערך ה- alpha המיטבי.



ElasticNet Regression:

מפריד את הדוגמאות ע"י פונקציה לינארית אך בשונה ממפריד לינארי רגיל, הפונקצייה אותה מנסה למזער היא בעיית הריבועים הפחותים בתוספת פנקציות הקנס המוזכרות ברגרסיות מסוג Ridge ו- Lasso. נבדוק שני פרמטרים שונים עבור מסווג זה:

1. alpha – פרמטר המכפיל את פונקציית הקנס המחושבת.
2. l1_ratio – פרמטר המגידר את המשקל שניתן לכל אחת משתי פונקציות הקנס המחושבות. כאשר l1_ratio = 0, ElasticNet עובד באופן דומה ל- Ridge, וכאשר l1_ratio = 1, ElasticNet עובד באופן דומה ל- Lasso.

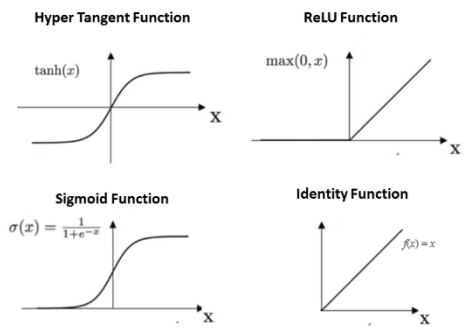
את הפרמטרים האידאליים נמצא ע"י cross-validation.

Multi-layer Perceptron Regression:

רשת ניורונים המורכבת ממספר שכבות, כאשר בכל שכבה נמצאים ניורונים – יחידות חישוב שמבצעות פעולות מתמטיות. השכבה הראשונה של הרשת קולטת את המידע שמגיע כקלט, והשכבה האחרונה היא שכבת הפלט. כל שכבה ברשת מחוברת לשכבה הבאה ע"י קשתות שמסמלות משקלים. מטרת העבודה של הרשת היא לבצע אופטימיזציה למשקלים אלה, כלומר לעדכן עד כמה ניורון מסויים בעל חשיבות לקבלת פלט מדוייק יותר בשכבת הפלט. MLPR בונה רשת כמתוארת למעלה.

לרשת ניורונים מסוג זה מספר פרמטרים שאותם נחקור:

- **Activation** – פונקציית האקטיבציה שמחשב כל ניורון. נבחן את הפונקציות הבאות:
  - Identity – פונקציית הזהות. המידע מועבר מניורון בשכבה A לניורון בשכבה B בלי כל שינוי.
  - Logistic – עבור כל מידע X שמועבר לניורון בשכבה A – מועבר $f(x) = \frac{1}{1+e^{-x}}$ לניורון בשכבה B.
  - Tanh – עבור כל מידע X שמועבר לניורון בשכבה A – מועבר $f(x) = \tanh(x)$ לניורון בשכבה B.
  - Relu – עבור כל מידע X שמועבר לניורון בשכבה A – מועבר $f(x) = \max(0, x)$ לניורון בשכבה B.



- **Solver** – קובע כיצד משתנים המשקלים ברשת. נבחן את האופטימיזטורים הבאים:
  - lbfgs – אופטימייזר hill-climbing ממשפחת quasi-Newton methods. מומלץ להרצה כאשר סט הדוגמאות איתו עובדים קטן (פחות מכמה אלפים).
  - Adam – גם כן מבוסס על stochastic gradient descent, אך מומלץ יותר להרצה על סט דוגמאות גדול.

- **Alpha** – מגביל את גודל המשקולות. הגדלת אלפא עשויה לתקן שונות (variance) גבוהה (סימן ל-overfitting), והקטנתו עשויה לתקן הטיה (bias) גבוהה (סימן ל- underfitting).

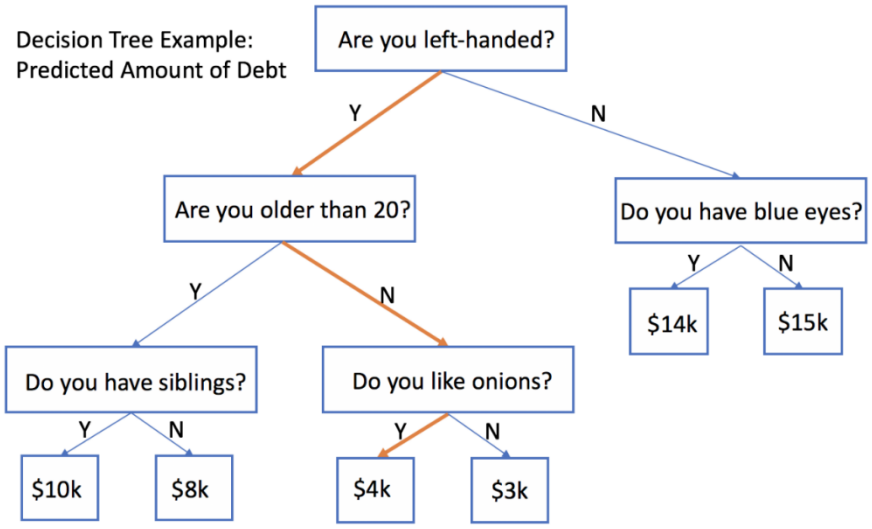
Random Forest Regression:

אלגוריתם מבוסס עצי החלטה. עץ החלטה הוא אלגוריתם שיתן לתאר כעץ, כאשר כל חוליה פנימית היא החלטה, או בעצם ערך של פיצ'ר מסויים, לפיה העץ מפצל את סט הדוגמאות שלנו, וכל עלה בעץ הוא בעצם הסיווג עבור אותה דוגמא.

Random forest הוא אלגוריתם המכיל מספר עצי החלטה, כאשר הסיווג עבור דוגמא מסויימת נבחר לפי דעת רוב עצי ההחלטה בוועדה.

לאלגוריתם מסוג זה מספר פרמטרים שאותם נחקור:

- **Criterion** – פונקציה המחשבת את איכות החלוקה של חוליה בעץ. הערכים אותם חקרנו הם:
  - Mean Squared Error – MSE
  - Mean Absolute Error – MAE
- **Min_samples_split** – מספר הדוגמאות המינימלי הנדרש בצומת כדי לפצל אותה.
- **Min_samples_leaf** – מספר הדוגמאות המינימלי הנדרש בצומת מסויימת כדי שתחשב עלה.
- **n_estimators** – כמות עצי ההחלטה ביער.



2. דרכי דילול ועיבוי מידע בהן נשתמש:

כנזכר בתחילת הדו"ח, את המידע קיבלנו מתחנות מדידה קיימות בכל רחבי הארץ, כל תחנה מספקת פיצ'רים שונים עבור כל יום מדידה. כדי להתמודד עם כמות המידע עבור שתי הגישות לפתרון שהגדרנו, היינו צריכים לעבד את סט הדוגמאות עבור המשימות השונות. הדרכים בהן השתמשנו הן:

- **One Hot Encoder** – עבור הגישה הראשונה לפתרון. בעוד כל הפיצ'רים בעלי ערכים לא חסומים (טמפרטורה, מהירות רוח, לחות וכדומה), אחד הפיצ'רים שיכולנו להגדיר חסם לערכו הוא כיוון הרוח. פיצ'ר זה יכול לקבל כל ערך בין $0^o - 360^o$ המציין את כיוונה של הרוח. החלטנו להגדיר את הכיוונים באופן הבא:
  - עבור ערך 0/360 – נגדיר כיוון מזרח.
  - עבור ערך 1-89 – נגדור כיוון צפון מזרח.
  - עבור ערך 90 – נגדיר כיוון צפון
  - עבור ערך 91-179 – נגדיר כיוון צפון מערב.
  - עבור ערך 180 – נגדיר כיוון מערב.
  - עבור ערך 181-269 – נגדיר כיוון דרום מערב.
  - עבור ערך 270 – נגדיר כיוון דרום.
  - עבור ערך 271-359 – נגדיר כיוון דרום מזרח.

לאחר המרת הערכים לשמות הכיוונים, השמתנו את הפיצ'ר של כיוון הרוח והוספנו במקומו שמונה פיצ'רים בוליאניים חדשים, כשמות הכיוונים, כאשר עבור יום מדידה מסויים, הערך 1 ניתן לכיוון הרוח שנמדד באותו יום, ולשאר הכיוונים ניתן הערך 0.

Date	WS	TD	RH	NE	NW	SE	SW	North	South	West	East
2016-03-02	3.08	19.46	28.47	0	1	0	0	0	0	0	0
2016-03-03	5.23	12.09	86.63	0	0	1	0	0	0	0	0

*דוגמא של מידע אחרי OHE (השמטנו כמה פיצ'רים כדי שהתמונה תיכנס במסמך)

- **השלמת ערכים חסרים** – כדי להשלים ערכים לא תקינים עבור פיצ'רים מסויימים, עבדנו בשיטה הבאה: אם כמות הערכים החסרים עבור פיצ'ר מסויים גדולה מ-25% מכמות הדוגמאות הכוללת, נוותר על הפיצ'ר הנ"ל, אחרת – נשתמש בפונקציה mean כדי לחשב את ממוצע כל הערכים של אותו פיצ'ר שכן קיימים, וערך זה יהיה ערך הפיצ'ר עבור כל הדוגמאות שלא קיימת מדידה של פיצ'ר זה.

- **הוספת ימים כפיצ'רים** – עבור הגישה הראשונה לפתרון. כנזכר בתחילת הדוח, כמות הפיצ'רים עבור תחנת מדידה אחת היא 16 פיצ'רים במקרה הטוב (במקרה וכל הפיצ'רים אכן נמדדים האותה תחנת מדידה). בנסיון להגדיל את כמות הפיצ'רים, עבור כל דוגמא הוספנו כמות מסויימת של מדידות מימים קודמים לאותה דוגמא.

Date	TD	RH	TDmax	TDmin	Grad	NIP	DiffR	Rain	TD_1	RH_1	TDmax_1
2016-01-01	8.255	75.160	8.387	8.112	69.368	26.479	53.465	0.035	8.933	72.535	9.105
2016-02-01	10.045	90.063	10.117	9.969	48.285	12.611	41.903	0.106	8.255	75.160	8.387

*דוגמא להוספת יום אחורה כפיצ'רים לדוגמא מסויימת (לא הבנתי למה סימנתי את זה בתכלת)

- **<u>השמטת פיצ'רים לפי פונקציית Corr()</u>** – למבנה הנתונים DataFrame קיימת פונקציה בשם Corr().
הפונקציה מחשבת את PCC - Pearson's correlation coefficient בין משתנה X למשתנה Y. מקדם המתאם של פירסון הוא חלוקת השונות של שני משתנים בתוצר סטיות התקן שלהם. ככל שערך זה גבוה יותר, כך הקשר הלינארי בין X ל- Y גדול יותר. בחלק זה, בדקנו ערכים שונים של סף עבור מקדם המתאם (ערכים בין 0 ל- 0.5) והשארנו רק פיצ'רים שמקדם המתאם שלהם גדול מערך הסף באטירציה הנוכחית.

- **<u>SelectKBest</u>** – פונקציה מהספריה scikit-learn, שבוחרת את K הפיצ'רים הטובים ביותר, לפי פונקציות score מסויימת. הפרמטרים לפונקציה זו בהם השתמשנו הם:
  - Score_func – הפונקציה שעל פיה ניתנים ציונים לכל פיצ'ר, ביחס ל- label הנבחר. אנחנו השתמשנו בפונקציה f_regression, המתאימה לבעיות מסוג זה.
  - K – כמות הפיצ'רים הכי טובים שיבחרו בסוף האלגוריתם. כאן ננסה כמות שונה של פי'צרים להשאיר בסוף פעולת האלגוריתם, ונתייחס לכמות זו בניסויים שלנו.

תיאור הניסויים:

1. שלב הלמידה:

בשלב זה נבדוק את איכות המסווגים שבחרנו לפתרון הבעיה. לאחר בניית ה- dataset הראשוני, השתמשנו בדרכי עיבוי ודילול מידע שונות (הנזכרות למעלה) והרצנו כל אלגוריתם למידה בצורתו הדיפולטיבית, זאת כדי לקבל את הפיצ'רים הטובים ביותר איתם עובד כל מסווג:

1. השמטנו את כל הפיצ'רים שקיים חוסר גדול במדידה שלהם, והשלמנו בעזרת פונקציית mean את אלה שהחוסר שלהם היה נמוך.
2. השתמשנו ב- one hot encoder לפריסת הפיצ'ר WD – Wind Direction, ל-8 פיצ'רים בינאריים חדשים המציינים את 8 כיווני הרוח.
3. הוספנו "ערכי ימים קודמים" כפיצ'רים לכל הדוגמאות שלנו (בדקנו ערכים שונים של הוספת ימים, כולל נסיון לא להוסיף ימים כלל).

בסוף שלב זה, התחלנו למפות בצורה אלגוריתמית את הפיצ'רים ב- 2 שיטות שונות:

1. השמטנו פיצ'רים שאינם קורלטיביים דיים לתיוג שלנו בעזרת הפונצקיה corr של dataframe (גם כאן, בדקנו ערכים שונים עבור ההגדרה "כמה פיצ'ר מסויים קורלטיבי לתיוג", כולל נסיון לא למפות כלל את המידע לפי מדד זה).
2. השתמשנו באלגוריתם select_k_best שנמצא תחת ספריית sklearn (שוב, בדקנו ערכי k שונים לפונקציה זו).

לבסוף, השתמשנו ב- cross-validation (כאשר כמות ה-folds היא 10) כדי ללמד כל אלגוריתם לפי הפיצ'רים הנבחרים מהתהליך הנ"ל בצורה הכי הוגנת ולא מוטת.

לאחר שלב זה ניקח את הפיצ'רים שעבדו בצורה הכי טובה עם כל מסווג, ונריץ את כל המסווגים פעם נוספת, כדי לברר אילו hyper-parameters יתנו שיפור לפעולתו הדיפולטיבית של כל מסווג.

כדי לבחון זאת בצורה מסודרת, השתמשנו במבנה מעטפת שנקרא GridSearchCV – מבנה זה מקבל מסווג מסויים, ומילון פרמטרים, ומריץ את המסווג עם כל וריאציה של הפרמטרים, כאשר הוא מבצע cross-validation עם n_folds = 10, ופונקציית מבחן mean_absolute_error, ומחזיר תוצאה עבור כל וריאציה כזאת.
ככל שהשגיאה האבסולוטית קטנה יותר, כך נגיד כי המסווג אותו בדקנו עובד בצורה טובה יותר. בחלק זה נראה את התוצאות שהתקבלו לאחר הרצת כל המסווגים עם הערכים אותם החלטנו לבדוק.

תוצאות מלאות כולל נתונים מספריים נראה עבור תחנת הטכניון בלבד, אך בסוף קטע זה נראה השוואה בין התוצאות הכי טובות עבור כל איזור מדידה.

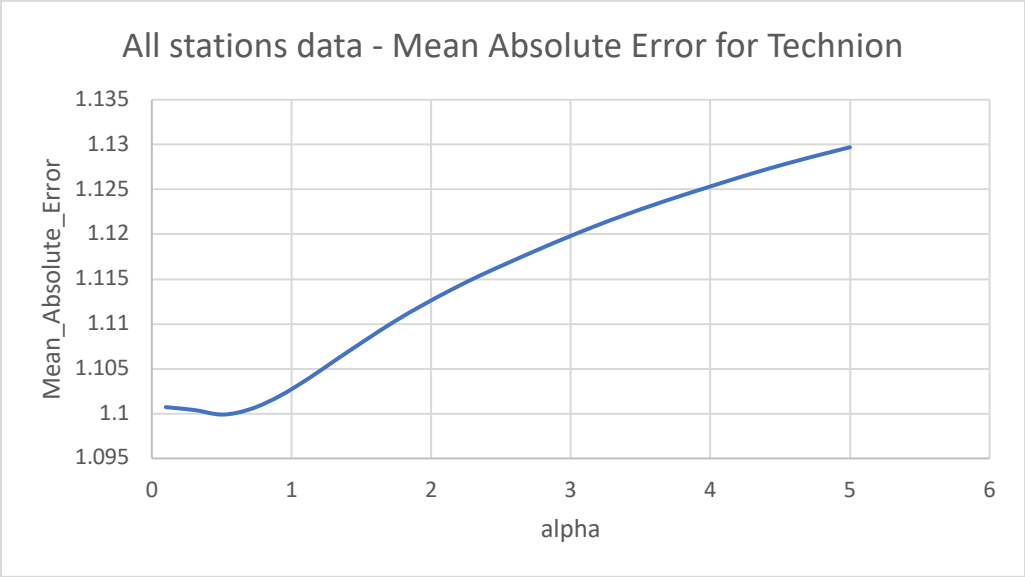
בנוסף, נראה השוואה בין המסווג שעובד עם מידע רק מתחנת המדידה בסביבת הבדיקה, לבין מסווג שמשתמש במידע מכל תחנות המדידה בארץ, וזאת נעשה עבור 2 תחנות המבדק שלנו, וננסה לקבוע היכן קיבלנו את הסיווג הטוב ביותר.

:Ridge

בדקנו את השגיאה האבסולוטית כפונקציה של משתנה alpha שמגדיר את גודל הקנס שמתקבל בחישוב (כנזכר למעלה).

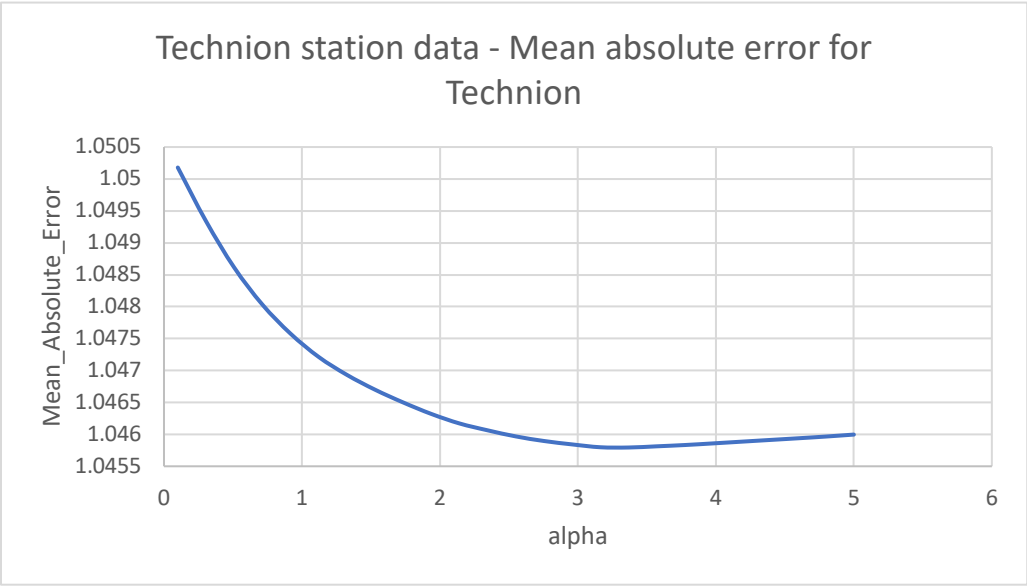
עבור מידע מכל תחנות המדידה:

alpha	mean_absolute_error
0.1	1.100718702
0.304167	1.100397598
0.508333	1.099891099
0.7125	1.100532287
0.916667	1.101967393
1.120833	1.10389803
1.325	1.106061944
1.529167	1.108182365
1.733333	1.110231383
1.9375	1.112078233
2.141667	1.113783831
2.345833	1.115359477
2.55	1.116797158
2.754167	1.118199384
2.958333	1.119545162
3.1625	1.120830441
3.366667	1.122029149
3.570833	1.123152547
3.775	1.124214657
3.979167	1.125224944
4.183333	1.126222565
4.3875	1.127177584
4.591667	1.12806804
4.795833	1.128901773
5	1.129701579

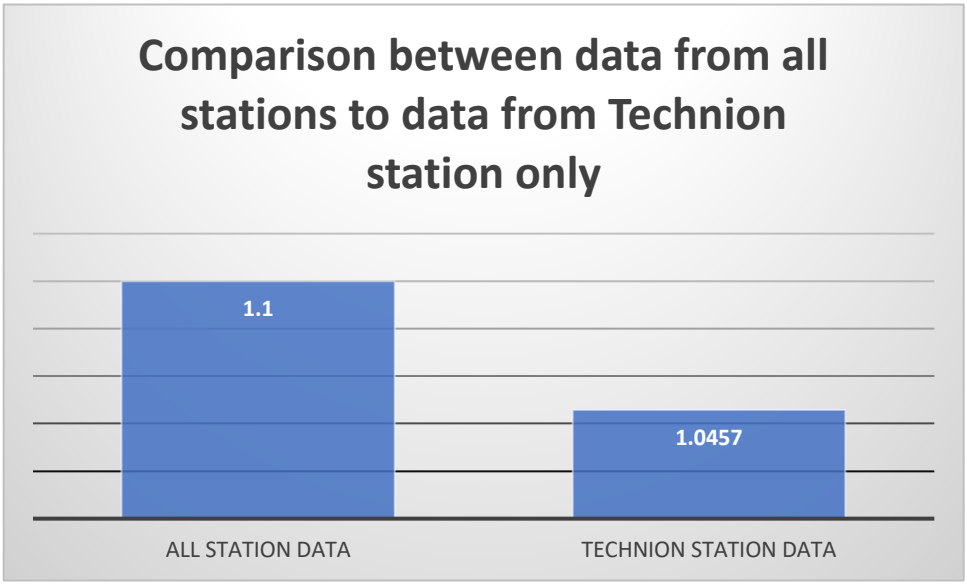


עבור מידע מתחנת המידע של הטכניון בלבד:

alpha	mean_absolute_error
0.1	1.050179365
0.304167	1.049340359
0.508333	1.048614562
0.7125	1.04803398
0.916667	1.047578022
1.120833	1.047212982
1.325	1.046932603
1.529167	1.046700022
1.733333	1.046499791
1.9375	1.046319895
2.141667	1.046170778
2.345833	1.046063097
2.55	1.045967985
2.754167	1.045895988
2.958333	1.045842518
3.1625	1.04579815
3.366667	1.045794083
3.570833	1.045812703
3.775	1.045832942
3.979167	1.045858523
4.183333	1.045885209
4.3875	1.045911753
4.591667	1.045938101
4.795833	1.045965675
5	1.045996632



השוואה בין טיב החיזוי כאשר משתמשים במידע מכל תחנות המדידה מול שימוש במידע מתחנת הטכניון בלבד:



מהניסויים אותם ביצענו ניתן לראות מספר דברים:

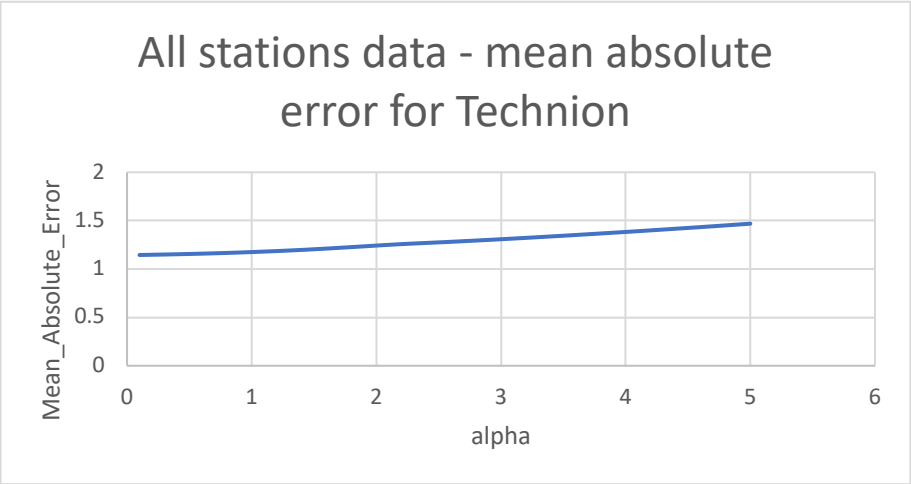
1. ניתן לראות שבאופן כללי, ככול שהפרמטר alpha גדל, השגיאה האבסולוטית בחיזוי גדלה כאשר נתחשב במידע שמכיל את כל תחנות המדידה בארץ – ניתן אולי ליחס זאת לכמות ומורכבות הפיצ'רים אותם אנו מנסים לסנן בעזרת אלגוריתמי feature selection עליהם דיברנו בחלק הקודם.
2. ניתן לראות שכאשר מתייחסים למידע המתקבל מתחנת הטכניון בלבד, השגיאה האבסולוטית בחיזוי קטנה כאשר מתקרבים לערך alpha = 3.
3. ניתן לקבוע בוודאות שעבור מסווג זה, עדיף להשתמש במידע מתחנת המדידה של הטכניון בלבד – השגיאה האבסולוטית הקטנה ביותר כאשר מתייחסים למידע שנאסף מכל התחנות גדולה יותר מהשגיאה הנ"ל כאשר מתייחסים למידע מתחנת המדידה בטכניון בלבד. כמובן ניתן לראות שהמסווג הנ"ל עובד טוב יותר כאשר מתייחסים למידע מתחנת הטכניון.

Lasso:

גם כאן, בדקנו את השגיאה האבסולוטית כפונקציה של משתנה alpha שמגדיר את גודל הקנס שמתקבל בחישוב (כנזכר למעלה).

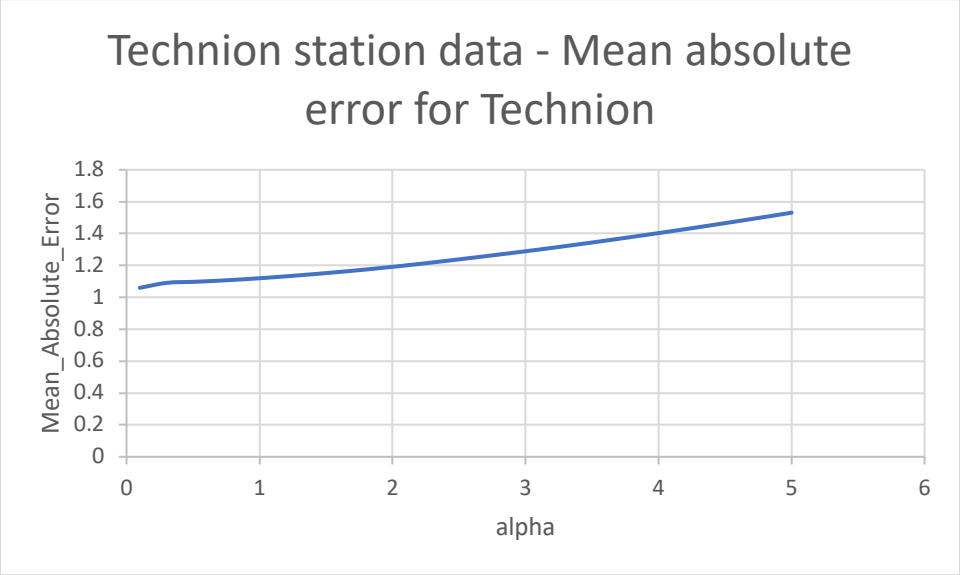
עבור מידע מכל תחנות המדידה:

alpha	mean_absolute_error
0.1	1.144909554
0.304167	1.149871183
0.508333	1.155834909
0.7125	1.162773699
0.916667	1.171296489
1.120833	1.181274283
1.325	1.192747089
1.529167	1.206137541
1.733333	1.221273517
1.9375	1.23729045
2.141667	1.253058807
2.345833	1.266273253
2.55	1.278170641
2.754167	1.29129181
2.958333	1.30496315
3.1625	1.31933714
3.366667	1.334202175
3.570833	1.34952852
3.775	1.365375729
3.979167	1.381498172
4.183333	1.398157981
4.3875	1.415179345
4.591667	1.432538681
4.795833	1.450296076
5	1.468633791

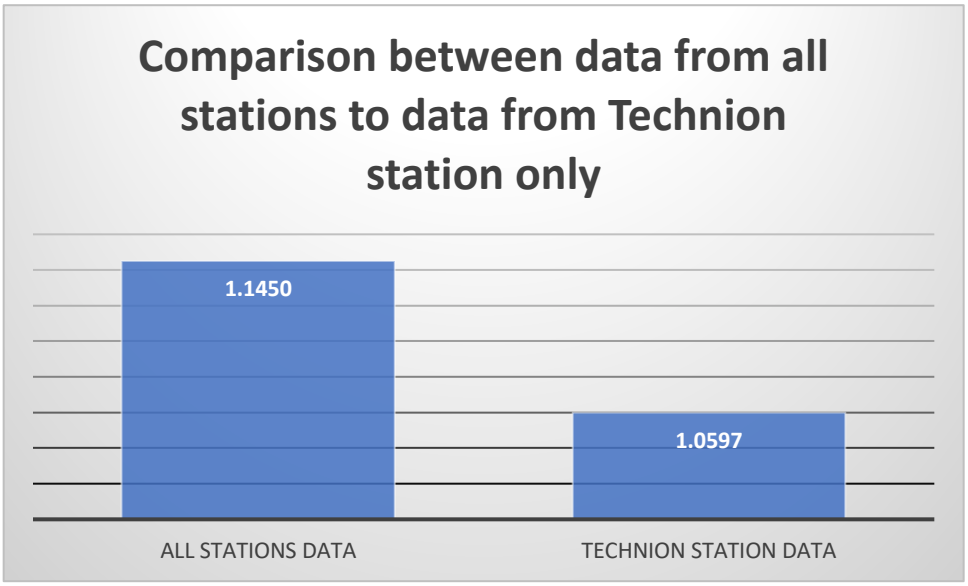


עבור מידע מתחנת המידע של הטכניון בלבד:

alpha	mean_absolute_error
0.1	1.059794648
0.304167	1.090991523
0.508333	1.097309629
0.7125	1.105299881
0.916667	1.115276802
1.120833	1.126883209
1.325	1.139946434
1.529167	1.154142353
1.733333	1.169281657
1.9375	1.1860248
2.141667	1.203875252
2.345833	1.223140295
2.55	1.243195817
2.754167	1.263563821
2.958333	1.284548393
3.1625	1.306202362
3.366667	1.328836895
3.570833	1.351963857
3.775	1.37586917
3.979167	1.400382798
4.183333	1.425385665
4.3875	1.45104822
4.591667	1.477077147
4.795833	1.503664127
5	1.530779835



השוואה בין טיב החיזוי כאשר משתמשים במידע מכל תחנות המדידה מול שימוש במידע מתחנת הטכניון בלבד:



מהניסויים אותם ביצענו ניתן לראות מספר דברים:

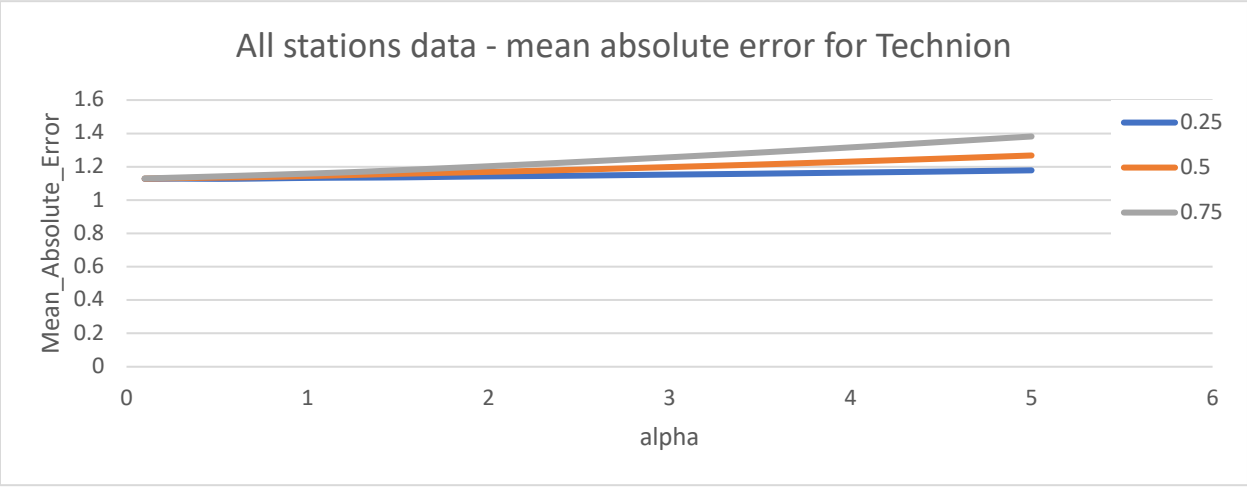
1. ניתן לראות שבאופן כללי, ובשתי הגישות שבדקנו, ככול שהפרמטר alpha גדל, השגיאה האבסולוטית בחיזוי גדלה. לפי פעולת האלגוריתם, ניתן לומר כי הגדלת alpha מגבילה את יכולות המודל ומפשטת אותו. היות שהגדלת ערך alpha משפיעה לרעה על התוצאה ומגדילה את השגיאה, אנו משערים כי המודל בו אנו עוסקים מורכב יותר, והגבלה שלו פוגעת בחיזוי.
2. מהגרף האחרון ניתן לקבוע בוודאות שעבור מסווג זה, עדיף להשתמש במידע מתחנת המדידה של הטכניון בלבד – השגיאה האבסולוטית הקטנה ביותר כאשר מתייחסים למידע שנאסף מכל התחנות גדולה יותר מהשגיאה הנ"ל כאשר מתייחסים למידע מתחנת המדידה בטכניון בלבד (שיפור של 1% בשגיאה).

ElasticNet:

בדקנו את השגיאה האבסולוטית כפונקציה של משתנה alpha שמגדיר את גודל הקנס שמתקבל בחישוב (מוסבר בחלק של אלגוריתמי הלמידה) ושלה המשתנה l1_ratio שמגדיר את היחס בהתחשבות בכל פונקציית קנס.

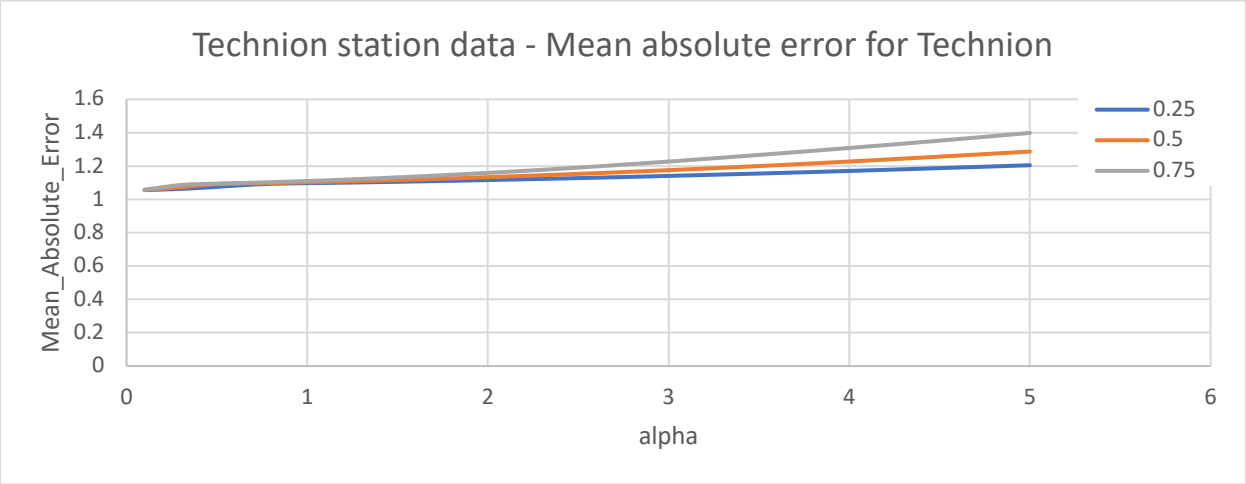
עבור מידע מכל תחנות המדידה:

alpha\l1_ratio	0.25	0.5	0.75
0.1	1.12770762	1.12861149	1.12970755
0.304166667	1.12860608	1.13190529	1.13576684
0.508333333	1.12733658	1.13470749	1.14214938
0.7125	1.12901954	1.13726907	1.14879814
0.916666667	1.13205666	1.14180587	1.15587395
1.120833333	1.13402703	1.14717261	1.16299462
1.325	1.13490224	1.15265859	1.171158
1.529166667	1.13630176	1.15809858	1.18042505
1.733333333	1.13892823	1.16299473	1.19004563
1.9375	1.14158697	1.16771369	1.1999646
2.141666667	1.1433898	1.17268399	1.21028432
2.345833333	1.14499667	1.17815178	1.22083703
2.55	1.14722763	1.18418502	1.23193926
2.754166667	1.14990812	1.19052998	1.24317144
2.958333333	1.15236306	1.19678482	1.25439624
3.1625	1.15456628	1.2030214	1.26579873
3.366666667	1.15685843	1.20934359	1.27745253
3.570833333	1.15940007	1.21594723	1.28949906
3.775	1.16202809	1.22297213	1.30205686
3.979166667	1.16458333	1.23011827	1.3148173
4.183333333	1.16712342	1.23729592	1.3277949
4.3875	1.16978989	1.24461635	1.34114068
4.591666667	1.17258864	1.25209434	1.35467315
4.795833333	1.17550501	1.25978003	1.36822953
5	1.17848353	1.26763737	1.38159129

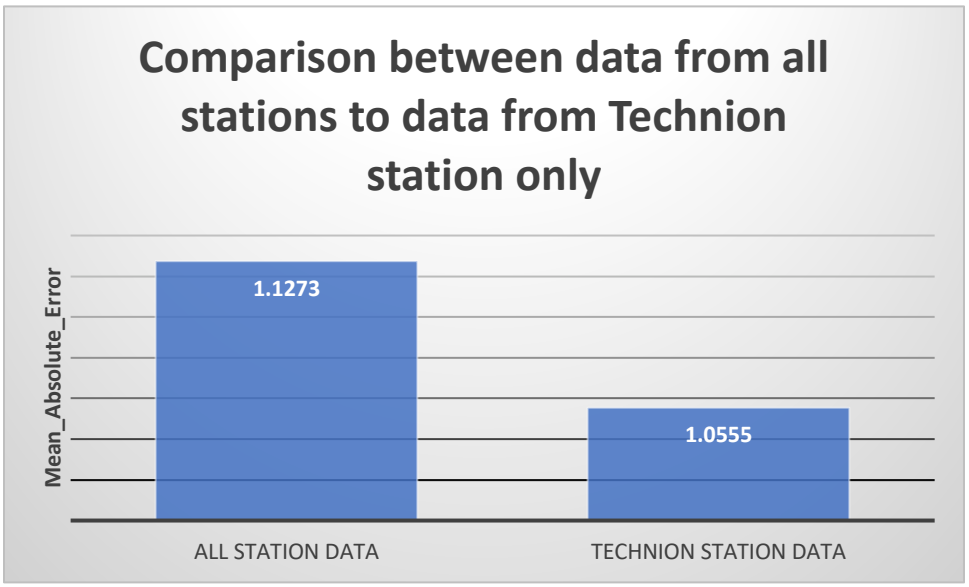


עבור מידע מתחנת המדידה של הטכניון בלבד:

alpha\l1_ratio	0.25	0.5	0.75
0.1	1.05553038	1.05649178	1.05796793
0.304166667	1.06317926	1.07339155	1.08663305
0.508333333	1.07535536	1.09247874	1.09463745
0.7125	1.08958548	1.096113	1.10027719
0.916666667	1.09595996	1.10052877	1.10701338
1.120833333	1.09890801	1.10553149	1.11478757
1.325	1.10230447	1.11100499	1.1236737
1.529166667	1.10608823	1.11706932	1.13351068
1.733333333	1.11011833	1.12377559	1.14408065
1.9375	1.11443549	1.13102888	1.15531899
2.141666667	1.11909884	1.13870195	1.16729577
2.345833333	1.12390364	1.14671376	1.18014711
2.55	1.12885504	1.15522307	1.1939899
2.754166667	1.13405738	1.16402142	1.20857986
2.958333333	1.13958224	1.17326718	1.22375264
3.1625	1.14529223	1.18290455	1.23949847
3.366666667	1.15113284	1.19300119	1.25582635
3.570833333	1.15711476	1.20354811	1.27236848
3.775	1.16338613	1.21446355	1.28928447
3.979166667	1.16987679	1.22582727	1.30653409
4.183333333	1.17659298	1.23760881	1.32429156
4.3875	1.18351369	1.24966062	1.34230842
4.591666667	1.19056977	1.26193421	1.36064437
4.795833333	1.19769439	1.27433799	1.37947578
5	1.20487604	1.28694882	1.39871127



השוואה בין טיב החיזוי כאשר משתמשים במידע מכל תחנות המדידה מול שימוש במידע מתחנת הטכניון בלבד (לקחנו את התוצאה הטובה ביותר משתי הדרכים להשוואה זו):



מהניסויים אותם ביצענו ניתן לראות מספר דברים:

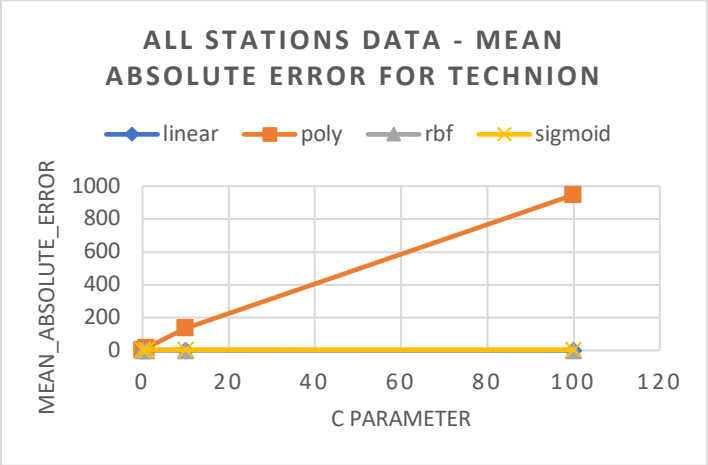
1. ניתן לראות שבאופן כללי, ובשתי הגישות שבדקנו, ככול שהפרמטר alpha גדל, השגיאה האבסולוטית בחיזוי גדלה.
2. אלגוריתם ElasticNet הוא אלגוריתם המשלב את אופן הסיווג בו משתמשים אלגוריתמי Ridge ו-Lasso. לפי התוצאות הנ"ל, ניתן לראות שכאשר ElasticNet נוטה יותר לסווג לפי Ridge (כלומר ערך l1_ratio < 0.5), השגיאה האבסולוטית בחיזוי קטנה יותר עבור רוב ערכי alpha שנבדקו, בשתי הגישות השונות שנקטו, בעוד שכאשר ElasicNet נוטה יותר לסווג לפי Lasso (ערך l1_ratio > 0.5), השגיאה האבסולוטית בחיזוי גדולה יותר לרוב ערכי הalpha שנבדקו, גם כן בשתי הגישות השונות שנקטו. תוצאה זו מאשרת את מה שראינו בשני הניסויים הקודמים, כאשר אלגוריתם Ridge החזיר תוצאות טובות יותר מאלגוריתם Lasso.
3. בהסתכלות כללית על התוצאות של שתי הגישות, ניתן לראות שכאשר מתייחסים למידע מתחנת הטכניון בלבד, מקבלים שגיאה אבסולוטית קטנה יותר, לעומת השגיאה כאשר מסתכלים על מידע מכל תחנות המדידה בארץ.

Support Vector Regression:

בניסוי זה בדקנו את השגיאה האבסולוטית כפונקציה של ערכים שונים של הפרמטר C, וכפונקציה של הפרמטר kernel.

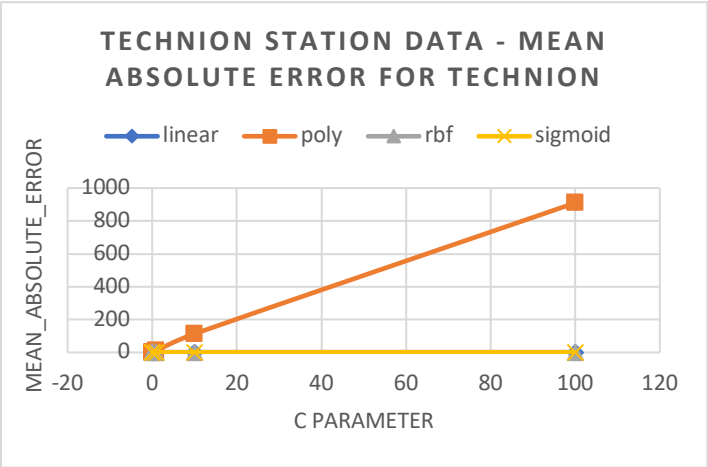
עבור מידע מכל תחנות המדידה:

C\kernel	linear	poly	rbf	sigmoid
0.01	1.128986	1.542548	4.61187	5.3946801
0.1	1.129207	2.42029	1.783738	5.3946801
1	1.130042	16.94826	1.239848	5.3946801
10	1.134449	136.3809	1.180653	5.3946801
100	1.13902	947.1096	1.249943	5.3946801

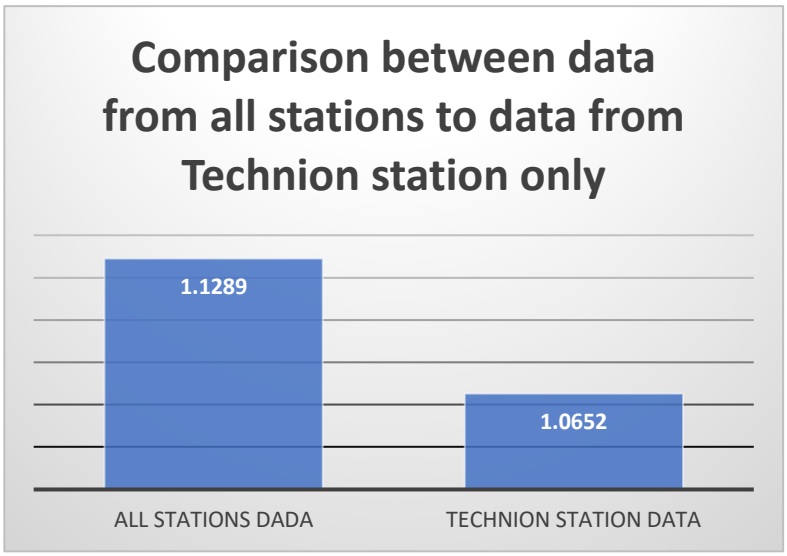


עבור מידע מתחנת הטכניון בלבד:

C\ kernel	linear	poly	rbf	sigmoid
0.01	1.084646	1.515008	4.617666	1.0652333
0.1	1.085272	2.177846	1.964787	1.0652333
1	1.086309	11.48014	1.294947	1.0652333
10	1.087844	113.4826	1.220669	1.0652333
100	1.088677	909.8934	1.317211	1.0652333



השוואה בין טיב החיזוי כאשר משתמשים במידע מכל תחנות המדידה מול שימוש במידע מתחנת הטכניון בלבד (לקחנו את התוצאה הטובה ביותר משתי הדרכים להשוואה זו):



מהניסויים אותם ביצענו ניתן לראות מספר דברים:

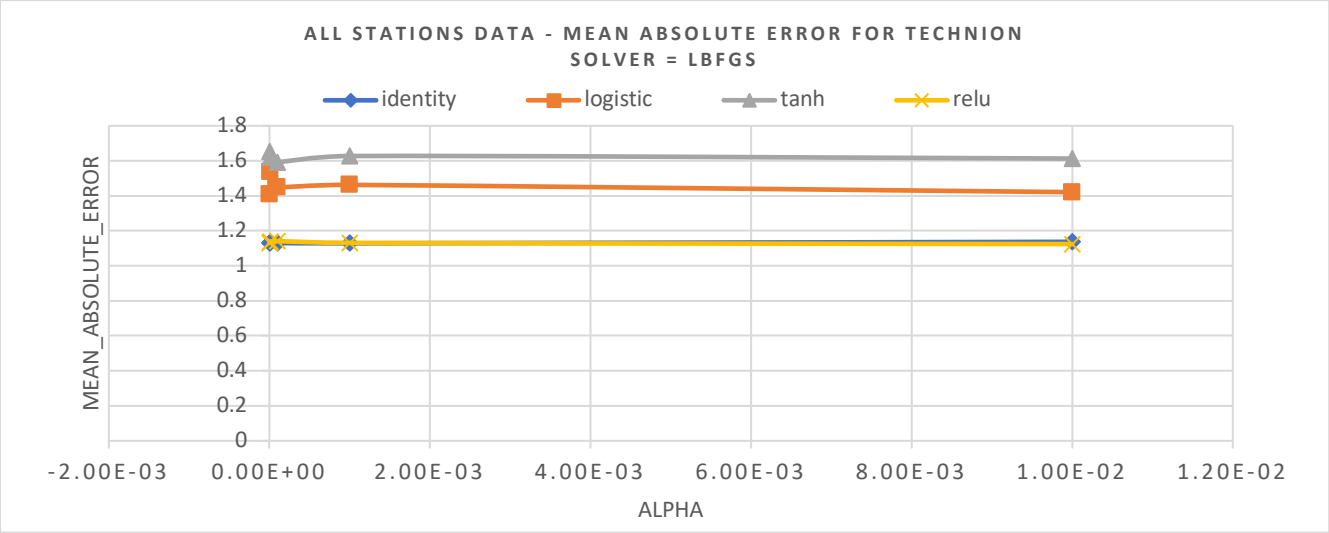
1. השינוי בפרמטר C אינו משפיע על התוצאה בשגיאה האבסולוטית, בשתי הגישות לפתרון, כאשר משתמשים ב- sigmoid כפונקציה המפרידה. ניתן אולי להסיק מכך שלא היו הרבה טעויות בסיווג, ולכן גודל הקנס לא השפיע על הדיוק.
2. פונקצית ההפרדה rbf מחזירה שגיאה אבסולוטית קטנה יותר כאשר מתחשבים במידע מכל תחנות המדידה בארץ.
3. עבור רוב סוגי הפונקציות המפרידות, פרמטר C קטן יותר נתן שגיאה קטנה יותר. הסיבה לכך יכולה להיות שהמודל אינו מספק מענה מדוייק לבעיה הנ"ל, והיות והגדלת פרמטר זה דורשת דיוק גבוה יותר מהמסווג, הגדלתו יכולה להשפיע לרעה על הסיווג הסופי.
4. ניתן לראות שפונקציה פולינומית כמפריד אינה נותנת מענה טוב לפתרון הבעיה.
5. גם כאן, שימוש במידע מתחנת הטכניון בלבד הניב שגיאה קטנה יותר, מאשר שימוש במידע מכל תחנות המדידה בארץ.

MLPRegression:

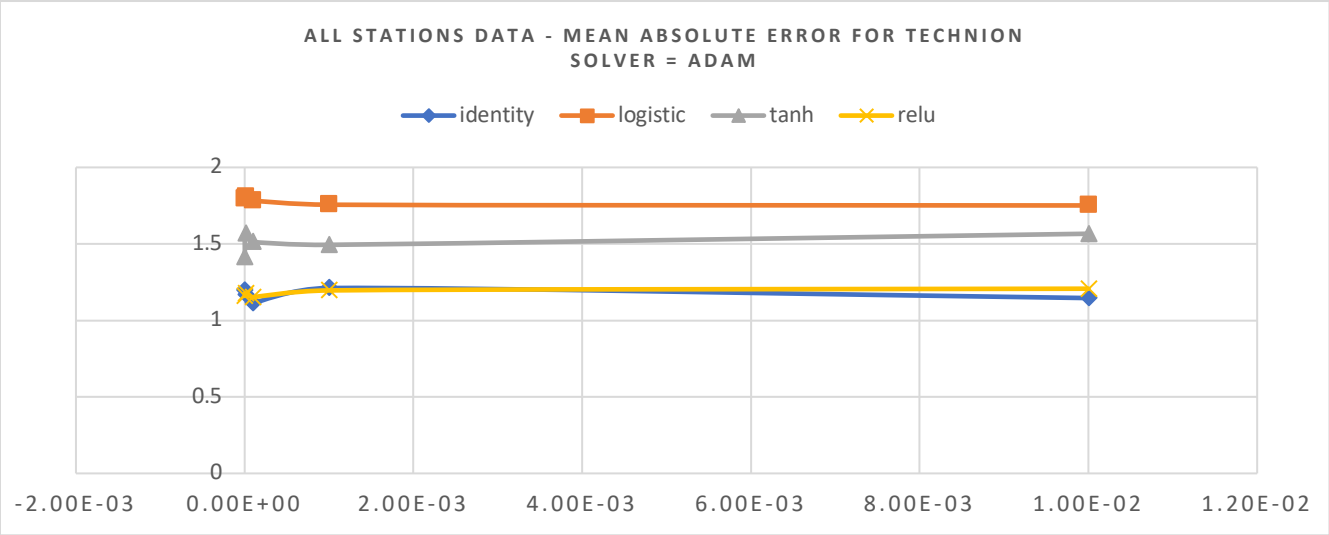
בדקנו את השגיאה האבסולוטית כפונקציה של הפרמטרים activation – פונקציית האקטיבציה בה משתמש האלגוריתם בכל שכבה, solver – קובע כיצד המשקלים ברשת משתנים, ו- alpha הקובע את הרף אליו יכולים להגיע המשקולות.

עבור מידע מכל תחנות המדידה:

alpha\activation	identity	logistic	tanh	relu
1.00E-06	1.1302534	1.407012	1.653896	1.130221
1.00E-05	1.1348183	1.534518	1.621862	1.138283
0.0001	1.1303045	1.449256	1.591333	1.141194
0.001	1.1285619	1.462522	1.627322	1.131032
0.01	1.1370638	1.420858	1.612866	1.124345

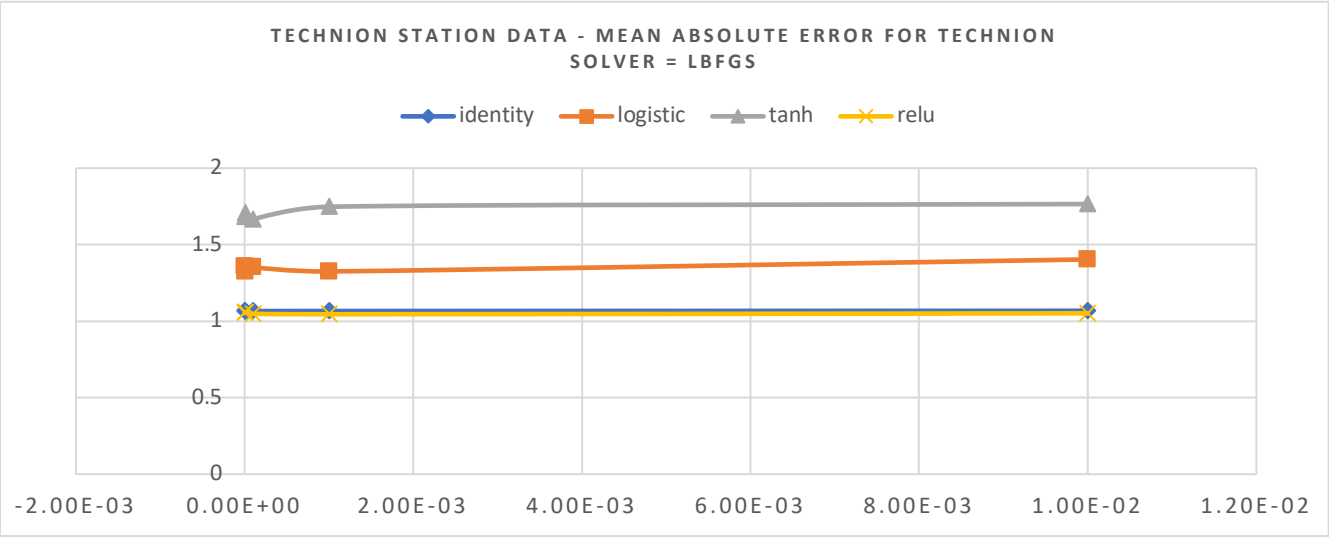


alpha\activation	identity	logistic	tanh	relu
1.00E-06	1.1978898	1.797144	1.413576	1.160459
1.00E-05	1.1684623	1.80844	1.570861	1.176996
0.0001	1.1124221	1.783366	1.513296	1.152016
0.001	1.2117623	1.756171	1.49394	1.196055
0.01	1.145999	1.751351	1.566471	1.207259

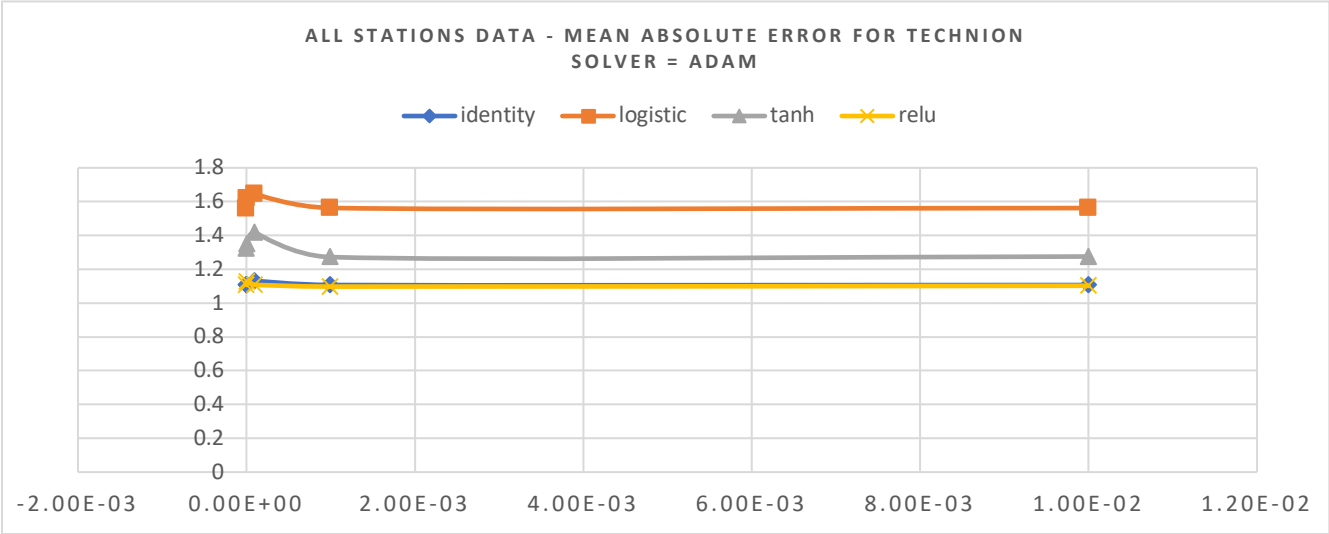


עבור מידע מתחנת הטכניון בלבד:

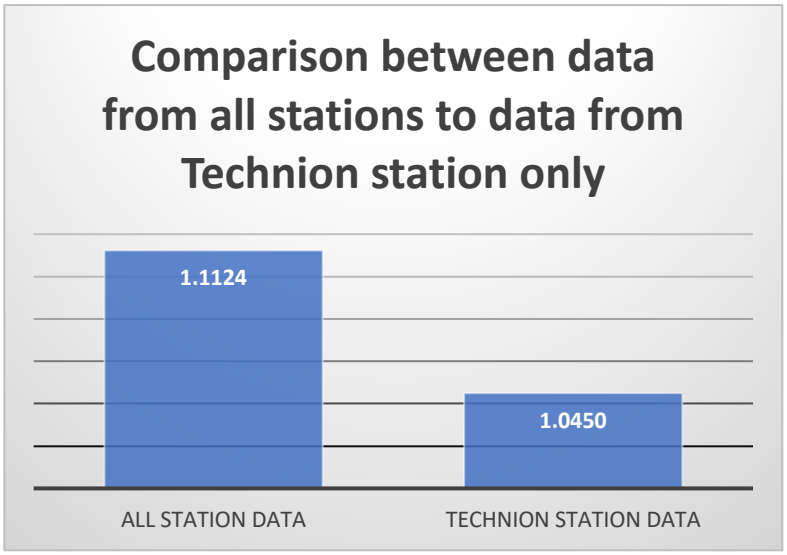
alpha\activation	identity	logistic	tanh	relu
1.00E-06	1.0685079	1.356813	1.681767	1.053414
1.00E-05	1.0656038	1.324518	1.708241	1.058879
0.0001	1.0658882	1.351049	1.667111	1.048119
0.001	1.0664273	1.325047	1.747947	1.045002
0.01	1.0678567	1.40371	1.765536	1.051263



alpha\activation	identity	logistic	tanh	relu
1.00E-06	1.1087639	1.558465	1.323147	1.107354
1.00E-05	1.1082421	1.617811	1.348651	1.127948
0.0001	1.1313513	1.644928	1.41614	1.108413
0.001	1.1070558	1.562824	1.272102	1.097529
0.01	1.1068599	1.562118	1.275442	1.102431



השוואה בין טיב החיזוי כאשר משתמשים במידע מכל תחנות המדידה מול שימוש במידע מתחנת הטכניון בלבד (לקחנו את התוצאה הטובה ביותר משתי הדרכים להשוואה זו):



מהניסויים אותם ביצענו ניתן לראות מספר דברים:

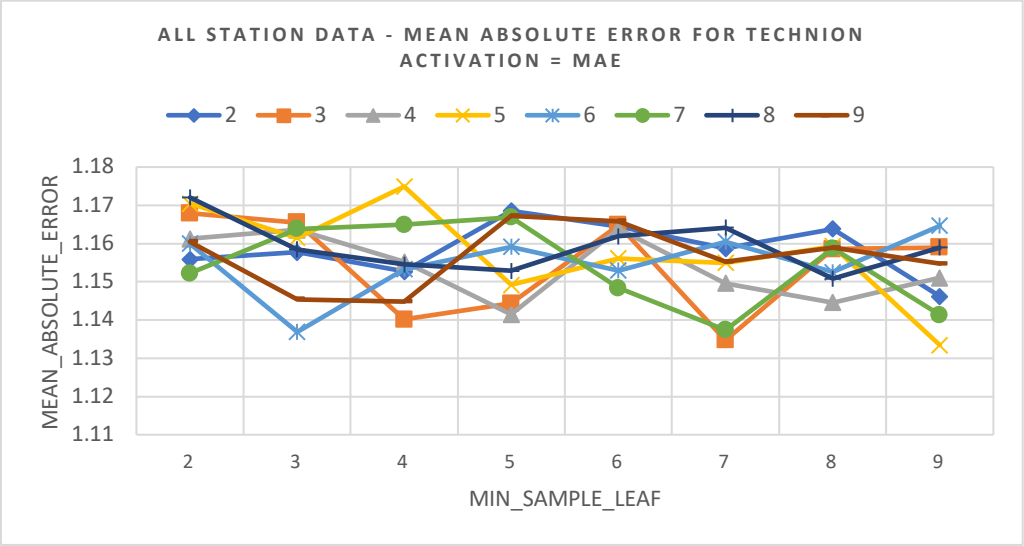
1. באופן מגמתי, נראה שהשימוש ב- solver מסוג lbfgs נותן שגיאה קטנה יותר, כמצופה, היות ולפי תיאור המסווגים הנזכר למעלה, אכן lbfgs הוא solver שעובד טוב יותר כאשר מספר הדוגמאות הכללי קטן (פחות מכמה אלפים, כמו במקרה שלנו), לעומת adam שעובד טוב יותר כאשר מספר הדגומאות גדול. המקרה היוצא דופן הוא פונקציית האקטיבציה tanh, שנראה שעובדת טוב יותר כאשר משתמשים ב- adam כ- solver.
2. עבור 2 הגישות שננקטו – התייחסות למידע מכל תחנות המדידה בארץ או למידע מתחנת הטכניון בלבד, נראה שפונקציות האקטיבציה identity, relu מניבות שגיאות אבסולוטיות קטנות יותר.
3. לבסוף, ניתן לראות מהגרף האחרון ששימוש במידע מתחנת הטכניון עדיף על שימוש במידע מכל תחנות המדידה בארץ באופן די משמשעותי (שיפור של 1% בשגיאה).

Random Forest Regression:

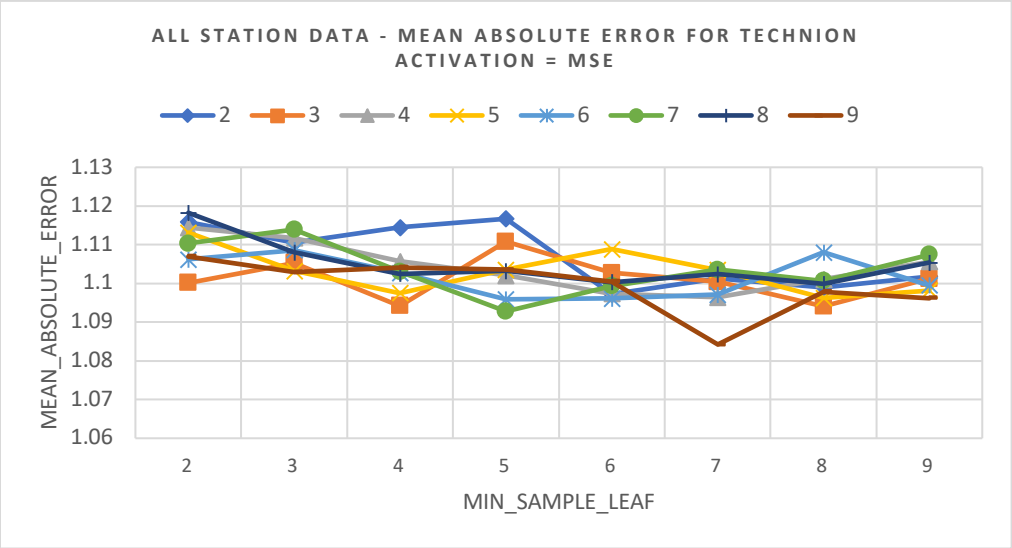
בדקנו את השגיאה האבסולוטית כפונקציה של n_estimators – כמות עצי ההחלטה, min_sample_split – כמות הדוגמאות המינימאלית הנדרשת לפיצול צומת, min_sample_leaf – כמות הדוגמרות המינימאלית הנדרשת לצומת להיקרא עלה, ו-critarion – הפונקציה המחשבת את איכות החלוקה בעץ.
קיבענו את n_estimator להיות 40 (עבור 2 הגישות, עבור פרמטר זה התקבלו התוצאות הכי טובות ולכן הצגנו מידע לפי ערך זה) ונראה את התוצאות שהתקבלו:

עבור מידע מכל תחנות המדידה:

leaf\split	2	3	4	5	6	7	8	9
2	1.155907	1.167935	1.161264	1.170469	1.159956	1.152156	1.172037	1.160479
3	1.157739	1.16552	1.163543	1.161506	1.136875	1.16386	1.158454	1.14539
4	1.152672	1.140136	1.155234	1.174911	1.15327	1.164953	1.154567	1.144813
5	1.168524	1.144282	1.14147	1.149148	1.159162	1.166854	1.152848	1.167235
6	1.164535	1.164796	1.163853	1.156096	1.152983	1.148368	1.161902	1.165872
7	1.158733	1.134717	1.149621	1.154952	1.16045	1.137371	1.164118	1.155244
8	1.163814	1.158561	1.14461	1.15927	1.152546	1.158698	1.15082	1.158886
9	1.146206	1.158947	1.151055	1.133446	1.164624	1.141342	1.158986	1.154665

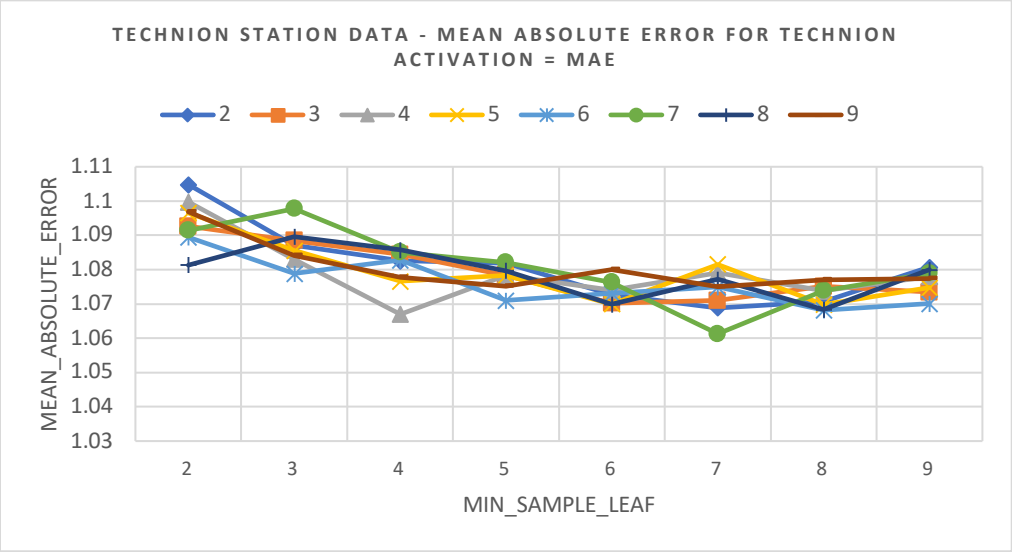


leaf\split	2	3	4	5	6	7	8	9
2	1.115846	1.100145	1.114428	1.113134	1.106228	1.110361	1.118251	1.106937
3	1.110608	1.105255	1.111703	1.103192	1.108555	1.113921	1.108001	1.102908
4	1.114499	1.094225	1.105729	1.097495	1.10276	1.103093	1.102442	1.104106
5	1.116702	1.110808	1.102029	1.103542	1.095894	1.092783	1.103198	1.103543
6	1.097238	1.102699	1.097367	1.108883	1.096153	1.099339	1.100331	1.100493
7	1.1013	1.100442	1.096438	1.103437	1.097132	1.103591	1.102357	1.084232
8	1.099033	1.094088	1.1011	1.096248	1.108055	1.100652	1.09989	1.097824
9	1.101857	1.101206	1.105619	1.09814	1.099531	1.107416	1.105361	1.09617

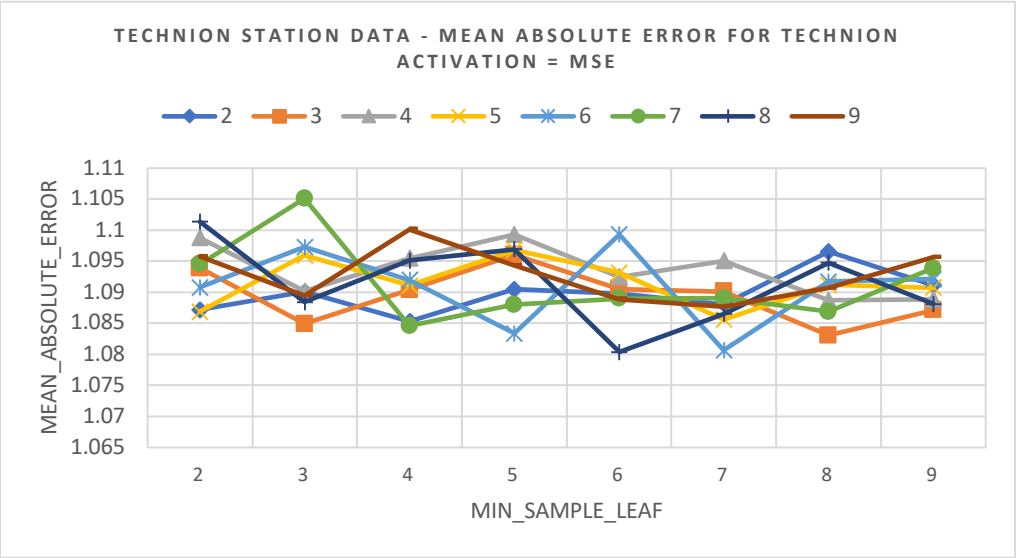


עבור מידע מתחנת הטכניון בלבד:

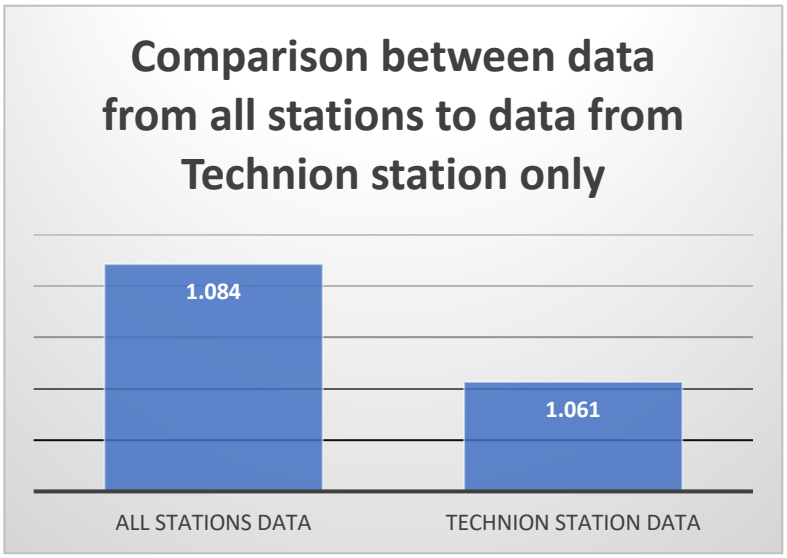
leaf\split	2	3	4	5	6	7	8	9
2	1.104815	1.092667	1.099742	1.096541	1.089456	1.091454	1.081334	1.096864
3	1.087131	1.088421	1.083074	1.08559	1.078802	1.097761	1.089603	1.084191
4	1.08257	1.084472	1.06698	1.07657	1.082711	1.085054	1.085733	1.077808
5	1.081889	1.078334	1.078338	1.078278	1.070988	1.0821	1.079709	1.075184
6	1.072657	1.070182	1.074076	1.070012	1.073044	1.076343	1.069917	1.080026
7	1.068804	1.071036	1.079148	1.081506	1.07493	1.061278	1.077167	1.075015
8	1.070731	1.075063	1.073796	1.069819	1.068173	1.073921	1.06836	1.077032
9	1.0807	1.073455	1.07798	1.074699	1.070032	1.07889	1.079949	1.077397



leaf\split	2	3	4	5	6	7	8	9
2	1.087193	1.093895	1.098773	1.086863	1.090782	1.094571	1.101372	1.09574
3	1.090073	1.084916	1.090178	1.095935	1.097277	1.105117	1.088437	1.089423
4	1.085314	1.090355	1.095467	1.091088	1.091971	1.084602	1.095137	1.100114
5	1.090504	1.096036	1.099312	1.096803	1.083356	1.088001	1.096937	1.094297
6	1.089753	1.090525	1.092499	1.093181	1.099298	1.088996	1.080385	1.088903
7	1.088064	1.090144	1.09508	1.085604	1.080661	1.089052	1.086506	1.087674
8	1.096571	1.083073	1.088717	1.091212	1.09171	1.086872	1.094738	1.090671
9	1.091135	1.087135	1.088884	1.090741	1.092154	1.093875	1.088132	1.095614



השוואה בין טיב החיזוי כאשר משתמשים במידע מכל תחנות המדידה מול שימוש במידע מתחנת הטכניון בלבד (לקחנו את התוצאה הטובה ביותר משתי הדרכים להשוואה זו):

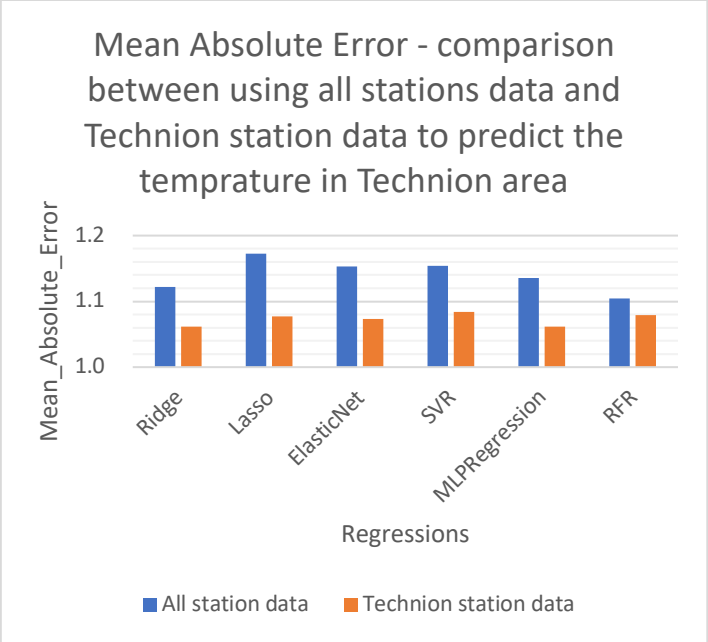


מהניסויים אותם ביצענו ניתן לראות מספר דברים:

1. כפי שנכתב בתחילת הניסוי, ניסינו ערכים שונים עבור כמות עצי ההחלטה ביער (בין 10 ל-50), והתוצאות הטובות ביותר עבור שתי הגישות התקבלו כאשר כמות עצי ההחלטה הייתה 40. השיפור בדיוק נבע כנראה משפע העצים שלקחו חלק בועדה, והם מייצגים בחירה מציאותית בצורה טובה יותר. הירידה בתוצאות כאשר נבדקו ערכים גדולים מ- 40 נבעה כנראה מהתאמת יתר לנתונים, ובכך גרמה לחיזוי לא נכון.
2. עבור מידע מתחנת הטכניון בלבד, ניתן לראות שכאשר activation = mae, קיבלנו מצב בו הגדלת מספר הדוגמאות בצומת להיחשבות צומת זה כעלה, נותנת שגיאה קטנה יותר בחיזוי (התוצאות בניסוי זה מונוטוניות יותר), ונראה שהשגיאה הקטנה ביותר מתקבלת כאשר min_sample_leaf = 7. אנו משערים כי ערך זה מונע התאמת יתר (overfitting) ולכן מתקבלות תוצאות טובות יותר.
3. עבור שתי הגישות, נראה ששימוש בפונקצית אקטיבציה mae מניב שגיאות קטנות יותר בחיזוי.
4. כפי שניתן לראות מהגרף האחרון, שימוש במידע מתחנת הטכניון בלבד מוביל לשגיאה קטנה יותר בחיזוי הטמפרטורה.

2. סיכום התוצאות עבור חיזוי הטמפרטורה באיזור תחנת הטכניון:

Regression\data	All station data	Technion station data
Ridge	1.1	1.045
Lasso	1.144	1.059
ElasticNet	1.127	1.055
SVR	1.129	1.065
MLPRegression	1.112	1.045
RFR	1.084	1.061

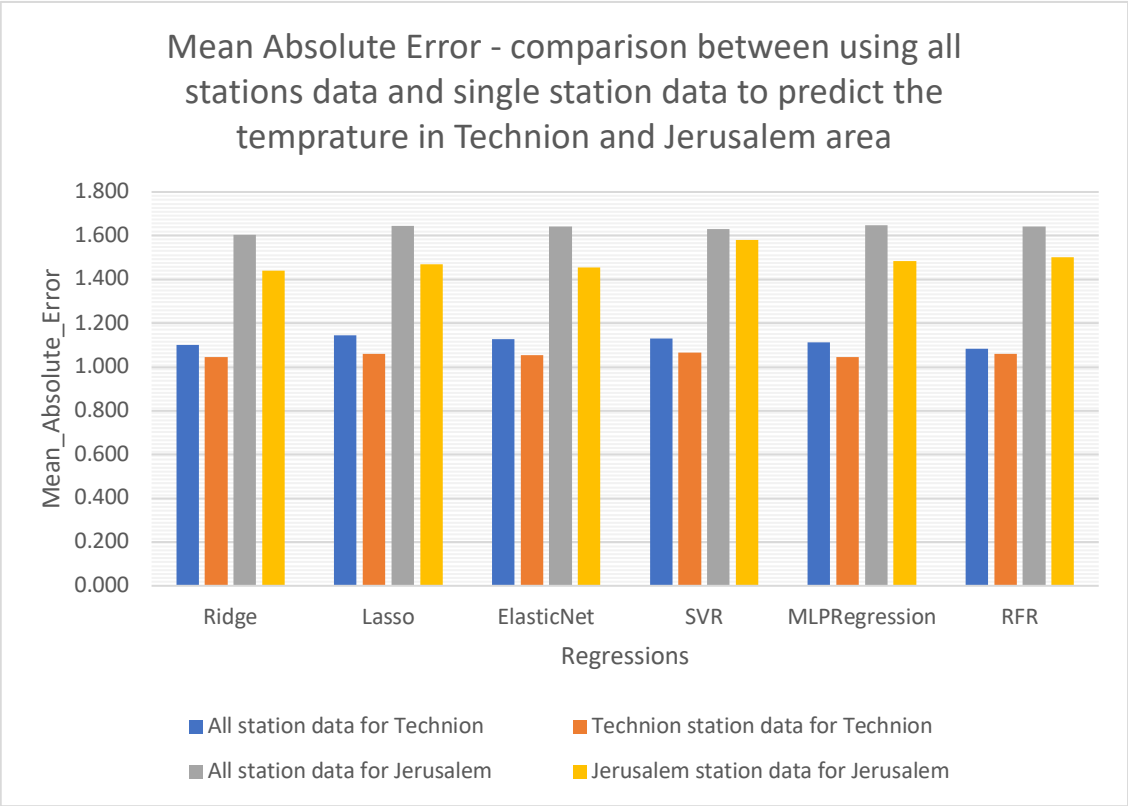


מסקנות:

1. ניתן לראות, שכאשר מסתכלים על הדיוק בסיווג כאשר מסתכלים על מידע מכל תחנות המדידה, או על מידע מתחנת הטכניון בלבד, ההבדל בין המסווג שעובד בצורה הטובה ביותר למסווג שעובד בצורה הגרועה מגיע כמעט לעשירית המעלה, שזהו שיפור של 1.05% בשגיאה.
2. במבט כללי, ההבדלים בין התחשבות במידע מכל תחנות המדידה בארץ, לעומת התחשבות במידע מתחנת הטכניון בלבד דיי משמעותיים וניתן לקבוע שכאשר נרצה לחזות את הטמפרטורה, נשתמש במידע שמגיע מתחנת המדידה בלבד, ולא מכל תחנות המדידה, כאשר השיפור גדול ביותר מעשירית המעלה, שיפור של 1.1% בשגיאה.
3. להפתעתנו, כדי לחזות את הטמפרטורה בטכניון, נשתמש ב- MLPR או ב- Ridge, שההבדל ביניהם זניח לחלוטין מבחינת השגיאה האבסולוטית הסופית, היות ורגרסור מסוג Ridge הוא מסווג מאוד פשוט בפעולתו, בעוד ש- MLPR היא בעצם רשת ניורונים הנחשבת מתוחכמת יותר.

3. השוואת תוצאות סופיות של חיזוי טמפרטורה באיזור הטכניון לחיזוי טמפרטורה באיזור ירושלים

Regression\data	All station data for Technion	Technion station data for Technion	All station data for Jerusalem	Jerusalem station data for Jerusalem
Ridge	1.1	1.045	1.605	1.441
Lasso	1.145	1.059	1.646	1.469
ElasticNet	1.127	1.055	1.643	1.454
SVR	1.129	1.065	1.631	1.582
MLPRegression	1.112	1.045	1.648	1.483
RFR	1.084	1.061	1.643	1.501



מסקנות:

1. נראה כי המודלים עובדים בצורה דומה:
   a. גם באיזור ירושלים, התוצאות הטובות ביותר התקבלו כאשר מתייחסים למידע מתחנת ירושלים בלבד, ולא למידע מכל תחנות המדידה בארץ כדי לחזות את הטמפרטורה. אם נשליך תוצאות אלה על הכלל, ניתן לומר כי החיזוי הטוב יותר יתקבל כאשר מתייחסים לנתונים מקומיים של התחנה ללא התייחסות לתחנות נוספות, ששימוש במידע שלהן מוריד את רמת הדיוק.
   b. הרגרסור שהחזיר את התוצאה הטובה ביותר הוא Ridge, בדומה לתוצאות הניסוי עבור חיזוי באיזור תחנת הטכניון.

2. נראה שיש שוני בטיב התוצאות בין 2 איזורי המבחן השונים. הסיבה לכך היא ככל הנראה המדידות שהתקבלו מאתר הרשות המטיאורולוגית - תחנת ירושלים מדדה יותר תכונות (פיצ'רים) מתחנת הטכניון, מה שהגדיל את מגוון הפיצ'רים עבור נסיון חיזוי באיזור תחנת ירושלים, ואולי בכך גרם למודל להיות מסובך יותר ולכן להחזיר תוצאות פחות טובות.

סיכום

מטרתנו בפרוייקט זה הייתה למצוא מה תהיה השיטה הטובה ביותר לחיזוי טמפרטורה בהינתן מידע על מדידות היסטוריות של מזג אוויר.
כדי להגיע לפתרון עברנו מספר שלבים שלהערכתנו היו הכרחיים:
הראשון - הבנת התחום אליו אנחנו ניגשים. כלומר, ממצב בו הידע שלנו על מזג אוויר, וחיזוי מזג אוויר וטמפרטורה בפרט - כללי ביותר, למצב שנבין מספיק על מנת לדעת על אילו פרמטרים להסתכל, ומה הדרישות הבסיסיות מבחינת תלות בין מאפיין למאפיין.

השלב השני - תכנון הפתרון ומציאת המידע. שלב זה כלל חיפוש מעמיק ביותר שלמידע וכן חיפוש מאמרים שונים העוסקים בתחום החיזוי עצמו, מתוך הבנה שתחום זה הינו נחקר ביותר, וסביר כי רבים עוסקים בו, ובפרט מנסים לשלב שיטות למידה שונות.

השלב השלישי - עיבוד המידע. בשלב זה ניתחנו את כלל התוצאות שקיבלנו לגבי הרגרסורים השונים, מתוך מטרה להבין מהי השפעת הבחירות ואיסוף המידע על תוצאות החיזוי. עיקר המאמץ היה פירוק המידע שנאסף לתתי נושאים שונים ולגרפים שונים כך שנוכל להגיע למסקנות בצורה הבהירה ביותר - וזאת מתוך הבנה שפיזור כל המידע למספר גרפים בודדים ללא חשיבה מראש מה ההשוואות שכדאי לעשות - עלול לגרום לחוסר בהירות בתמונה הכללית, ויכול להקשות מאד על הניתוחים השונים שביצענו לגבי כל מאפיין, ובפרט עלול היה להוביל למסקנות שגויות.
בנוסף, חשוב לציין כי גם בפן האישי למדנו הרבה, ומעבר לתחום מזג האוויר.
למדנו כיצד לעבוד עם מידע גולמי, ולעבד אותו, ולהפוך אותו לסדרת נתונים נוחה לעבודה, שבעזרתה ניתן לקדם את הניסויים שברצוננו לעשות, ולבנות תשתית מספקת לצורך עבודה עם מידע רב.
שיפרנו במספר מונים את היכולת שלנו לתכנן קדימה ניסויים בהינתן המידע אותו נרצה לנתח, ובפרט את היכולת לסנן ולהבדיל אילו מהפעולות שאנו עושים באפשרותן לתרום לנו, ואילו מהן עלולות להיראות לנו מעניינות לבדיקה אך בסופו של דבר לא יקדמו אותנו הלאה בדרך לפתרון. בהחלט ניתן לומר שבעתיד, תכנון ניסויים והסתכלות קדימה בצורה יעילה כבר בשלב המקדים יוכלו להקל עלינו את הדרך להגעה לפתרון יותר טוב.

הערות

בניסיון להבין מהן החולשות העיקריות בפרוייקט ניסינו לחשוב על כיוונים שונים בהם השימוש בו יכול להיות בעייתי, אך כאלה שלאו דווקא ניתנים לשליטה מיידית, או כאלה שבהם שגיאת החיזוי תהיה גדולה מדי עד כדי הפיכה של המודל ללא רלוונטי:

1. בניסוי זה ניסינו לחזות את הטמפרטורה באיזור מסויים מחר, בהינתן מידע על היום הנוכחי. לכן, כדי שהמודל שלנו יפעל כמצופה, נצטרך לתת לו מידע שנמדד מתחנת המדידה המתאימה. האלגוריתם שלנו ידע לסדר את המידע כך שאלגוריתם הלמידה יפעל בצורה המיטבית, כפי שפעלנו בניסויים.
2. כדי לחזות ימים נוספים מעבר ליום האחרון בו נרשמה מדידה בתחנה מסויימת, נצטרך לחזות את כל התכונות עבור כל יום מעבר ליום האחרון המתועד בסט המידע. כלומר כדי לחזות ימים נוספים נצטרך להסתמך על מידע שחזינו בעצמינו - דבר העלול לגרום להגדלת השגיאה, וככל שמספר הימים המתבססים על חיזוי גדל, התוצאה עלולה להיות פחות ופחות מהימנה.
3. המודל שלנו יוכל לחזות בהצלחה בתקופת הזמן הנוכחית. השינויים באקלים בין תקופות זמן ארוכות יגרום למודל שלנו להתיישן, היות והוא סטטי, וסט המידע עליו אנו מסתמכים אינו מתעדכן.
4. אנו מסתמכים על מידע שמגיע מתחנות מדידה המפוזרות ברחבי הארץ, המפעילות מכשירים שונים שלפעמים אינם מודדים את כלל הפיצ'רים. בהינתן סט מידע שמכיל דוגמאות להן 10 פיצ'רים שנמדדו בצורה תקינה, ישנו הסיכוי שנרצה לחזות מזג אויר עבור דוגמא לה נמדדו רק 9 פיצ'רים, ועם זאת המודל לא ידע להתמודד.

כיווני מחקר עתידיים

מתוך הבנה שתחום החיזוי של מזג האוויר הוא דינאמי ביותר, וכן העובדה שהוא בנוי על מודלים כבדים ביותר שמתוחזקים ונבנים על ידי מדינות, ברור שקיימים כיוונים רבים אליהם ניתן להתרחב:

1. תחילה, העובדה שבפרוייקט התעמקנו בעיקר בפיצ׳ר הטמפרטורה אינה מגבילה התיאוריה עליה התבססנו. ניתן לבדוק האם באמצעות אותן שיטות יהיה אפשר לחזות כל פיצ׳ר שהוא שנמצא בסט הנתונים איתו עבדנו ועליו נבנה המסווג.
2. ניתוח מבוסס מיקום - היינו רוצים שבהינתן נ״צ ספציפי נוכל לומר מה תהיה המדידה בו. אנחנו מניחים שצורת עבודה זו תדרוש איסוף מידע עבור כל נ"צ בארץ (בהנחה והמודל מתעסק רק בישראל), ועיבוד המידע בצורות שונות לחיזוי מיטבי.
3. עיבוד תמונה - כפי שתיארנו במבוא, רוב המודלים המרכזיים היום שפועלים בעולם מתבססים על ניתוח זמן אמת של נתונים המגיעים מלוויינים שונים ומכינים תיאור ויזואלי של שכבות האטמוספירה השונות - תחום שכלל לא נגענו בו בפרוייקט, ויכול להיות מעניין כיצד שילוב של ניתוחים שכאלו בצירוף המודל שבנינו ישנה את התוצאות.
4. טכניקות למידה עמוקה - במודל זה הנגיעה היחידה שלנו בתחום זה היה שילוב של הרגרסור MLP. כפי שהראנו, כאשר נבדק על נתונים מתחנה בודדת תוצאותיו היו טובות ביחס למודלים אחרים - משמע, קיים יתרון כלשהו לתחום זה בפעולות החיזוי. יהיה מעניין לראות כיצד הגדלה דרסטית של תקופת המדידות בה משתמשים (אולי אף עשרות שנים) בשימוש עם מודלי למידה עמוקה תוכל לשפר את תוצאת החיזוי, ובפרט לראות האם תצליח לשבור את מחסום מספר הימים הבודד שלהם ניתן לחזות בצורה טובה ביחס לאוסף הנתונים.