



USED CARS

ALEX PASHAYEV

OHAD Yael


DOR NAGAOKER

הנושא שלנו: כלי רכב משומשים מוטיבציה לפרויקט:

שוק רכבי היד שנייה גדל מיום ליום בגלל הקורונה ומחסור
בשבבים רשימת ההמתנה לרכבים חדשים רק גדלה וכתוצאה מכך מחירי
כלי הרכבים המשומשים עלו בעשרות אחוזים.
קשה לחזות מה המחיר בשוק היום ואין זמן מתאים יותר לעשות למידת
מכונה לחיזוי מחיר כלי רכב משומשים



Find your match



Make Body style **Advanced search** **Shop by what matters most**

New/used
Used cars

Make
All makes

Model
All models

Price
No max price

Distance
All miles from

ZIP


Search

From cargo space to tech, tell us what you like and we'll find you cars to love.

Body style
Car

Price
No max price

The only flexible workflow platform built for every business needs



Workflow

Learn More

<https://www.cars.com/> מקור המידע:

BeautifulSoup,pandas ספריות:

היקף נתונים: כ-70000 נתונים



CRAWLING

כמות סוגי הרכב גדולה מאוד ולכן
בחרנו מספר סוגים מובילים

```
for i in range(1,2):
    website="https://www.cars.com/shopping/results/?page="+str(i)+"&page_size=100"

    response = requests.get(website)
    soup=bs(response.content,'html.parser')
    cars=soup.find_all('div',attrs={'class':'vehicle-card'})
    for car in cars:
        try:
            Name.append(car.find("h2"))
        except:
            Name.append("NaN")

        try:
            Mileage.append(car.find("div",{ "class":"mileage" }).get_text())
        except:
            Mileage.append("NaN")

        try:
            CountRaiting.append(car.find("span",{ "class":"sds-rating__link sds-b
        except:
            CountRaiting.append("NaN")

        try:
            Rating.append(car.find("span",{ "class":"sds-rating__count" }).get_text())
        except:
            Rating.append("NaN")

        try:
            Price.append(car.find("span",{ "class":"primary-price" }).get_text())
        except:
            Price.append("NaN")

        try:
            link1.append(car.find("a")["href"])
        except:
            link1.append("NaN")
```



ועוד..

CRAWLING

כמות סוגי הרכב גדולה מאוד ולכן
בחרנו מספר סוגים מובילים

- בהתחלה יצרנו מערך לכל אחת מהעמודות
- מכל רכב שמרנו קישור שמתכו משכנו נתונים נוספים שלא הופיעו בעמוד הראשי כמו צבע רכב, סוג דלק ועוד ..



DATA TABLE

	Name	Year	Mileage	Count	Raiting	Rating	Fuel type	Interior color	Exterior_color	Price
0	Toyota Corolla LE	2019	47,644 mi.		1632.0	4.8	Gasoline	Black	Slate Metallic	\$19,991
1	Toyota Corolla LE	2018	38,971 mi.		272.0	4.6	Gasoline	Ash	Red	\$18,700
2	Toyota Corolla LE	2018	73,147 mi.		245.0	3.4	Gasoline	Black	Tan	\$14,800
3	Toyota Corolla LE	2020	52,955 mi.		1369.0	4.3	Gasoline	—	Silver	\$19,489
4	Toyota Corolla LE	2015	88,010 mi.		929.0	4.7	Gasoline	Ash	Classic Silver Metallic	\$15,985
...
8848	Tesla Model 3 Long Range	2020	11,122 mi.		29.0	4.6	Electric	Black	Gray	\$60,900
8849	Tesla Model 3 Performance	2019	34,396 mi.		11.0	2.2	Electric	Red	White	\$58,999
8850	Tesla Model 3 Long Range	2020	46,921 mi.		NaN	NaN	Electric	Black	Blue	\$53,999
8851	Tesla Model 3 Long Range	2019	22,720 mi.		130.0	4.2	Electric	Black	Pearl White Multi	\$53,997
8852	Tesla Model 3 Long Range	2018	35,762 mi.		23.0	2.8	Electric	Black	White	\$44,950

8853 rows × 9 columns



זאת הטבלה לפני טיפול
בנתונים ויש בה
כ- **80000** ערכים

טיפול בנתונים

- הפכנו את משתני סוגי הדלק ל **0**: בנזין ו **1**: חשמלי
- יצרנו עמודה נוספת שמחלק את מחיר הרכב לרמות של מ **1**- עד **9**
- את העמודות שכוללות מחרוזות של מספרים שינינו ל משתנים מסוג **int** | **float** בעזרת **pd.to_numeric**
- ביצענו השלמה בכל משתנה סוג דלק שחסר בעזרת **fillna**
- נבדק האם יש שורות כפולות
- היו עמודות שהיינו צריכים לשנות תוכן שלהם לצורך למידת המכונה
- מחקנו עמודות עם נתונים חסרים



אחרי טיפול בנתונים

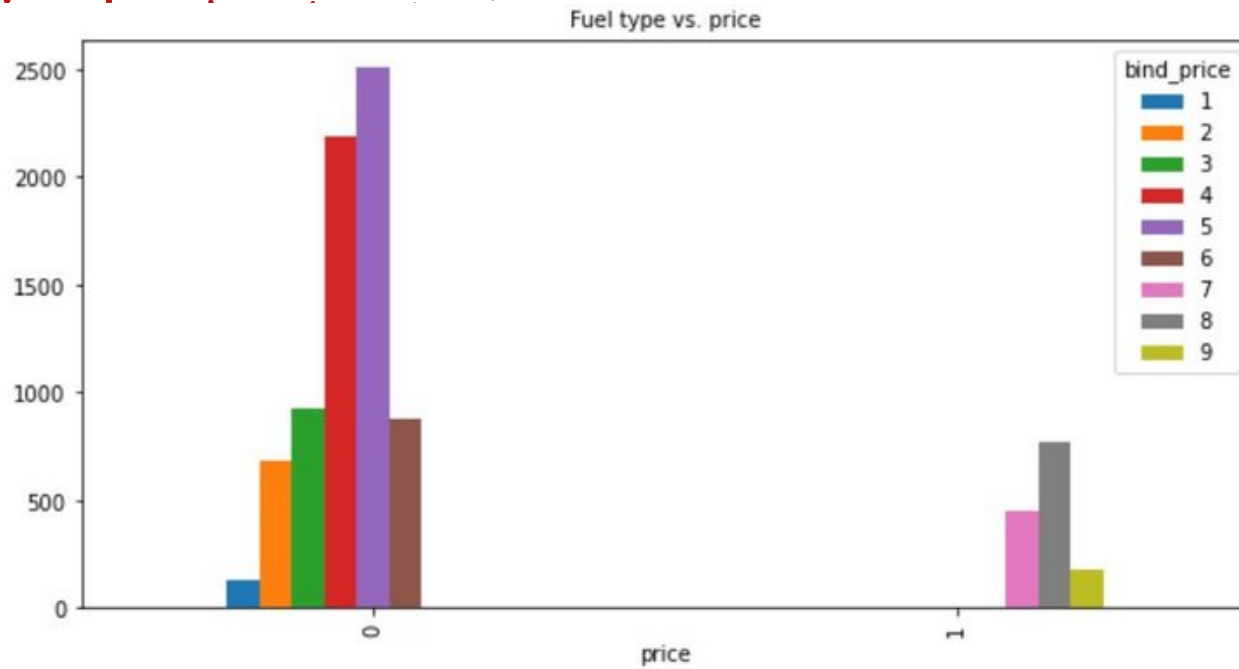
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7475 entries, 0 to 8852
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  -
 0   Name                 7475 non-null   object  
 1   Year                 7475 non-null   int64   
 2   Mileage              7475 non-null   float64  
 3   Count Raiting        7475 non-null   float64  
 4   Rating               7475 non-null   float64  
 5   Fuel type            7475 non-null   int64   
 6   Interior_color        7475 non-null   object  
 7   Exterior_color        7475 non-null   object  
 8   Price                7475 non-null   float64  
 9   bind_price           7475 non-null   int64   
dtypes: float64(4), int64(3), object(3)
memory usage: 642.4+ KB
```

אחרי טיפול בנתונים נשארנו עם
כ-**70000** ערכים לא כולל
עמודות נוספות שיצרנו



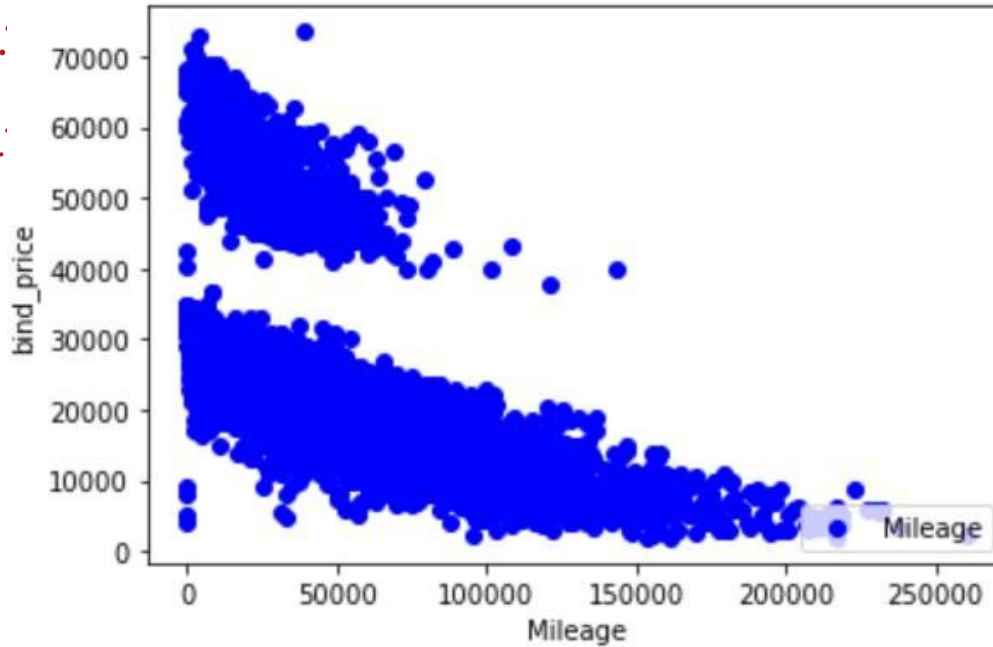
EDA



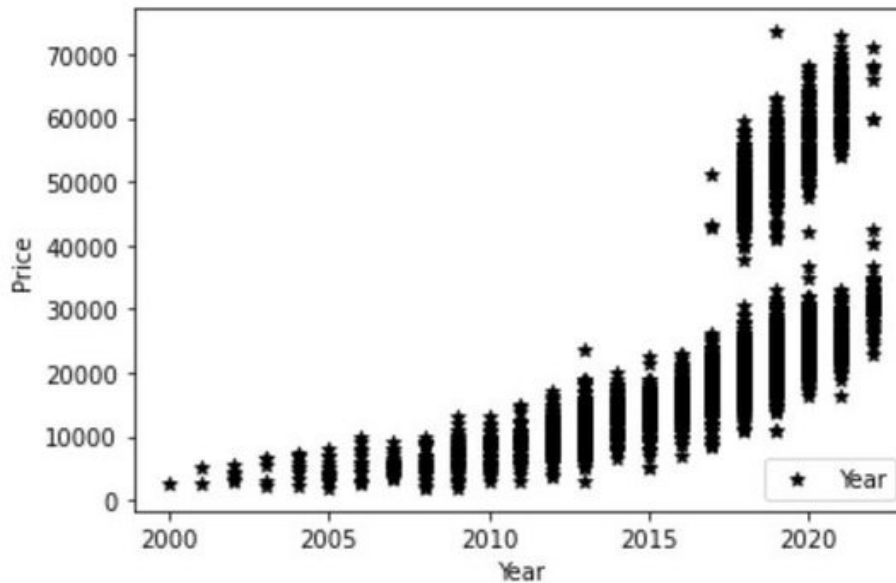
כמו שניתן לראות בגרף כל הרכבים
המונעים בעזרת דלק
נעים בטווח מחירים של **1** עד **6**
ורכבים חשמליים נעים בטווח **7** עד **9**



EDA



גרף של מחיר מול מיל
גם פה ניתן לראות שככל שהמיל גבוה ככה גם
המחיר יורד.
ההפרדה שנצורה בגרף היא הפער בין רכבים
חשמלים לבין רכבים המונעים בדלק



גרף של מחיר מול שנה
וגם פה יש את אותה הפרדה



MACHINE LEARNING

- במידת מכונה השתמשנו בספריית **sklearn** ובאלגוריתמים של רגרסיה ליניארית וברגרסיה לוגיסטית כדי לחזות:
 1. מחיר על סמך כל הנתונים (רגרסיה רב משתנית).
 2. קילומטראז' על סמך כל הנתונים (רגרסיה רב משתנית).
 3. האם לרכב יש בביקורת יותר מדירוג שלושה כוכבים וחצי או לא.
- בשלב הראשוני של למידת המכונה הפכנו משתנים שמים למשתנים קטגוריים לדוגמא: סוג כלי הרכב, צבע ריפוד, וצבע הרכב
בעזרת **LabelEncoder().fit_transform**
- בשלב השני של למידת המכונה השתמשנו בפונקצית **scale** לצורך נירמול הנתונים



MACHINE LEARNING CONCLUSIONS

בשלב השלישי הפרדנו לעמודות מטרה והפעלנו אלגוריתם של

רגרסיה ליניארית וגילינו כי:

ניתן לחזות מחיר על סמך הנתונים בסבירות גבוהה.

1

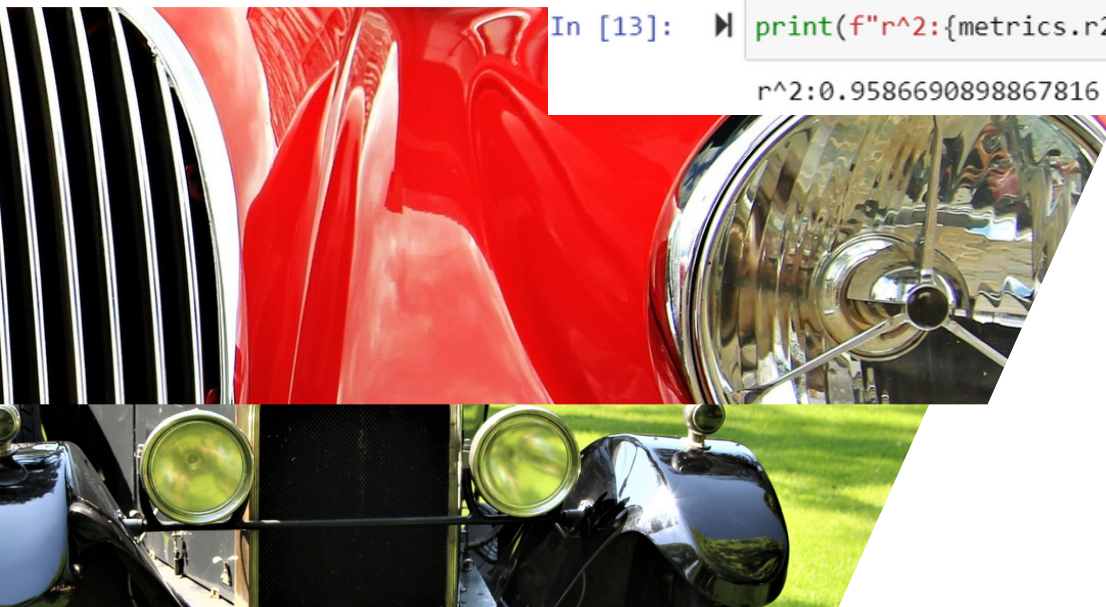
```
In [7]: X = cars_numeric.copy()
y = X['Price']
X = X.drop('Price',axis=1)
X = X.drop('bind_price',axis=1)
```

```
In [12]: reg=LinearRegression()
reg.fit(X_Train,y_Train)
y_preds=reg.predict(X_Test)
y_preds
```

```
Out[12]: array([53102.87083458, 55949.75900482, 21074.81732089, .
19269.19251332, 16462.83981876, 56589.16454528])
```

```
In [13]: print(f"r^2:{metrics.r2_score(y_Test,y_preds)}")
```

```
r^2:0.9586690898867816
```



MACHINE LEARNING CONCLUSIONS

2. ניתן לחזות קילומטראז' על סמך הנתונים בסבירות נמוכה יחסית.

```
In [15]: ▶ X2 = cars_numeric.copy()
          y2 = X2['Mileage']
          X2 = X2.drop('Mileage',axis=1)
```

```
In [19]: ▶ print(f"r^2:{metrics.r2_score(y2_Test,y2_preds)}")
          r^2:0.6744384985261485
```



MACHINE LEARNING CONCLUSSIONS

בשלב הרביעי הפרדנו לעמודות מטרה והפעלנו אלגוריתם של רגרסיה לוגיסטית וגילינו שאכן ניתן לחזות האם לרכב יש ביקורת גבוהה יותר מ**3.5** כוכבים בסבירות גבוהה

```
[14]: ▶ isatleast3andahalfRate = df.drop({"Rating"},axis=1)  
      isatleast3andahalfRate
```

```
In [18]: ▶ clf = LogisticRegression()  
          clf.fit(xtrain,ytrain)  
          accuracy = clf.score(xtest,ytest)  
          print(f"accuracy is {accuracy}")
```

```
accuracy is 0.918673087212413
```

