סיכום מדע הנתונים (Data Science) (מבוסס על קורס מדע הנתונים, Campus IL)

שלבי עבודה:

- **השערת המחקר:** הגדרת הבעיה, תיאום ציפיות והבנה של הבעיה באופן כללי.
- **הכנות מקדימות לפתרון:** הרכשה, אחסון וטיפול בנתונים.
- **ניתוח נתונים מתקדם:** הכוונה לפתרון (EDA) והפתרון (למידת מכונה).

הרכשה ואחסון נתונים

יש כל מיני פורמטים של נתונים. המרכזיים הם CSV ו־JSON, המשמשים כל אחד לשימושים שונים. פורמט CSV הוא לרוב פורמט שיוצא לאחר שהנתונים סודרו בתוך טבלה, לעומת זאת פורמט JSON הוא פורמט שמתקבל באמצעות הרכשת הנתונים מ־API מסוים.

באמצעות API

לכל חברה/אתר שפותח בפני הציבור גישה לנתונים שלו, יש תבנית וחוקים משלו. יש לעקוב אחר החוקים וההוראות. את התשובה נקבל בפורמט JSON, איתה נשחק קצת ונמיר ל־DataFrame.

read_csv doc | to_csv doc | JSON doc | JSON to DataFrame doc | JSON Guide | JSON Notebook

באמצעות Crawling

1. הגדרת הבעיה והנתונים הנחוצים.
2. לימוד והבנה של מבנה האתר ושל מבנה העמודים הפנימיים שלו.
3. הרכשת דפי אינטרנט שיש בהם את ההפניות לדפי התוכן הרלוונטיים ומעבר עליהם.
4. חילוץ ההפניות מדפי התוכן.
5. חילוץ התוכן מדפי התוכן.

כדי להרכיש נתונים בדרך זו צריך ידע לפחות בסיסי בה־HTML.

HTML Tutorial by W3 Schools

ב־Python יש ספרייה שנקראת BeautifulSoup, והיא מחקה את הדרך שבה ניגשים לאלמנטים באמצעות JavaScript (כדוגמת getElementById).

פונקציות מרכזיות בספרייה:

find: Finds an html element or tag, and returns the first match.
find_all: Returns a list (iterable) of all html elements that match the find criteria.
get_text: Returns the readable text inside the html elements contained in the BS object.
string: Convenience property of a tag to get the single string within this tag.
prettify: Will turn a BeautifulSoup object into a nicely formatted Unicode string, with a separate line for each tag and each string.
BeautifulSoup Notebook | Building a Scrapper Guide

טיפול בסיסי בנתונים

סוגי משתנים:

סולם מדידה	תכונות	דוגמא	תצוגה גרפית קלאסית
שמי	זהות	שפת אם	Bar plot / Pie chart
סדר	זהות, סדר	לא טוב, טוב, טוב מאוד	Bar plot
רווח	זהות, סדר, הפרש	טמפרטורה	Histogram
מנה	זהות, סדר, הפרש, יחס	משקל	Histogram

נהוג לחלק גם למשתנים בדידים ומשתנים רציפים. משתנים בדידים יהיו בעלי טווח מסוים. ואילו משתנים רציפים בעלי טווח אינסופי.

במצב של משתנים בדידים, נבנה טבלת שכיחויות. במצב של משתנים רציפים נבדוק סטיית התקן (Standard Deviation) ואת השונות (Variance) – סטיית התקן בריבוע. נוכל להשתמש ב־()df.describe שיניב את הממוצע (mean) ואת סטיית התקן (std).

קשר בין משתנים:

נבדוק האם יש קשר בין משתנים מסוימים (בין X ל־Y). מהי עוצמת הקשר? האם המשתנים מלמדים אחד על השני? וכו'...

כדי לבדוק קשר בין משתנים משתמשים בדיאגרמת פיזור (scatter plot), כאשר נשים את המשתנים שלנו כ־X וכ־Y. יש לראות האם הגרף יוצר צורה כלשהי.

All Commands Notebook

טיפול מתקדם בנתונים

לפני הטיפול המתקדם, חשוב מאוד ליצור טבלה/רשימה הכוללת את שם המשתנים ב־df ואת משמעותם.

נתונים חסרים:

כדי לאתר נתונים חסרים נשתמש ב־()df.info, או אם df.describe(include='all').

dtypes doc | describe doc | info doc | isnull doc

ניתן לטפל בנתונים באמצעות מחיקת כל המופעים (השורות) בעלי NaN, באמצעות df.dropna(inplace=True). דרך נוספת זה הכנסת ערך אחר במקום NaN (0 או מחרוזת דיפולטית, או ערך משוער באמצעות ממדי מרכוז – ממוצע, חציון, הערך הכי שכיח (נפוץ) וכד'...)

dropna doc | fillna doc | Missing Data Notebook

רשומות כפולות (דופליקציות):

כדי לאתר דופליקציות, נשתמש ב־()df.dupliacted, אפשר להעביר רשימה של מאפיינים כדי לברר אם יש דופליקציות בהתאם למאפיינים מסוימים, ולא דופליקציות 100% (שכל מאפיינן זהים לחלוטין).

duplicated doc | Duplicates Notebook

כדי להסיר דופליקציות נשתמש ב־()df.drop_duplicates. אפשר להעביר למתודה subset=[feature] כדי למחוק דופליקציות לא 100%.

drop_duplicates doc

נתונים חריגים (קיצוניים) (Outliers):

כדי לאתר נתונים חריגים יש 3 דרכים:

1. **איתור ערכים חריגים בגרפים:** נצייר דיאגרמת פיזור/היסטוגרמה ונראה איפה יש נתונים ש"מתבודדים" מהאחרים.
2. **מרחק מחציון (IQR):** האחוזון ה־25% מסומנן ע"י Q1, 50% (חציון) מסומן ע"י Q2 ו־75% מסומן ע"י Q3. ההפרש בין Q3 ל־Q1 נקרא טווח בין־רבעוני (IQR – Interquartile Range). החריגים יהיו אלו שקטנים מ־Q1 – 1.5IQR או Q3 + 1.5IQR. נוכל להמחיש זאת באמצעות boxplot (מה שמעבר לקווים השחורים).
3. **מרחק סטיית תקן מהממוצע:** נראה כמה משתנה רחוק מהממוצע.

Scan for Outliers Notebook

אם דווקא הגיוני שהנתונים החריגים מופיעים, נשאיר ככה. אם זה לא הגיוני, נטפל בהם – אפשר להפוך אותם ל־NaN, ואז להתייחס ולטפל בהם כמו לנתונים חסרים.

hist doc | boxplot doc | percentile doc | lists doc

המרת נתונים מסוגים שונים:

אפשרויות לקידוד משתני בדידים:

1. **קידוד מספרי פשוט:** לתת לכל מאפיין מספר (לדוגמה, גבר – 1, אישה – 2).
2. **משתני דמה (dummy variables):** להוסיף עמודות כמספר המאפיינים ולתת 0 או 1 לכל עמודה. מימוש עם פייתון באמצעות pd.get_dummies(df, columns=[feature], prefix=[something]).

מין	מספר דגימה	X1	X2
זכר	1	1	0
זכר	2	1	0
נקבה	3	0	1
נקבה	4	0	1
זכר	5	1	0
נקבה	6	0	1

3. **קידוד בינארי:** מאוד דומה ל־dummy variables, רק בייצוג בינארי.

ערך קטגוריאלי	ערך טבעי	X1	X2	X3
גרוע מאוד	0	0	0	0
גרוע	1	0	0	1
בסדר	2	0	1	0
טוב	3	0	1	1
טוב מאוד	4	1	0	0

אפשרויות להמרת משתנה רציף למשתנה בדיד:

יש לחלק את המשתנה לכמה חלקים (לדוגמה 1 יכול לייצג גיל 1–10, 2 לגיל 10–20 וכו'...). לאחר מכן ליצור 2 רשימות: bins, שתייצג את החלוקה הנ"ל, ו־labels שתציב את המספרים המייצגים. אחר־כך משתמשים ב־pd.cut(df[feature], bins, labels=labels).

astype doc | replace doc | cut doc | Data Type Casting Notebook

ניתוח נתונים מתקדם

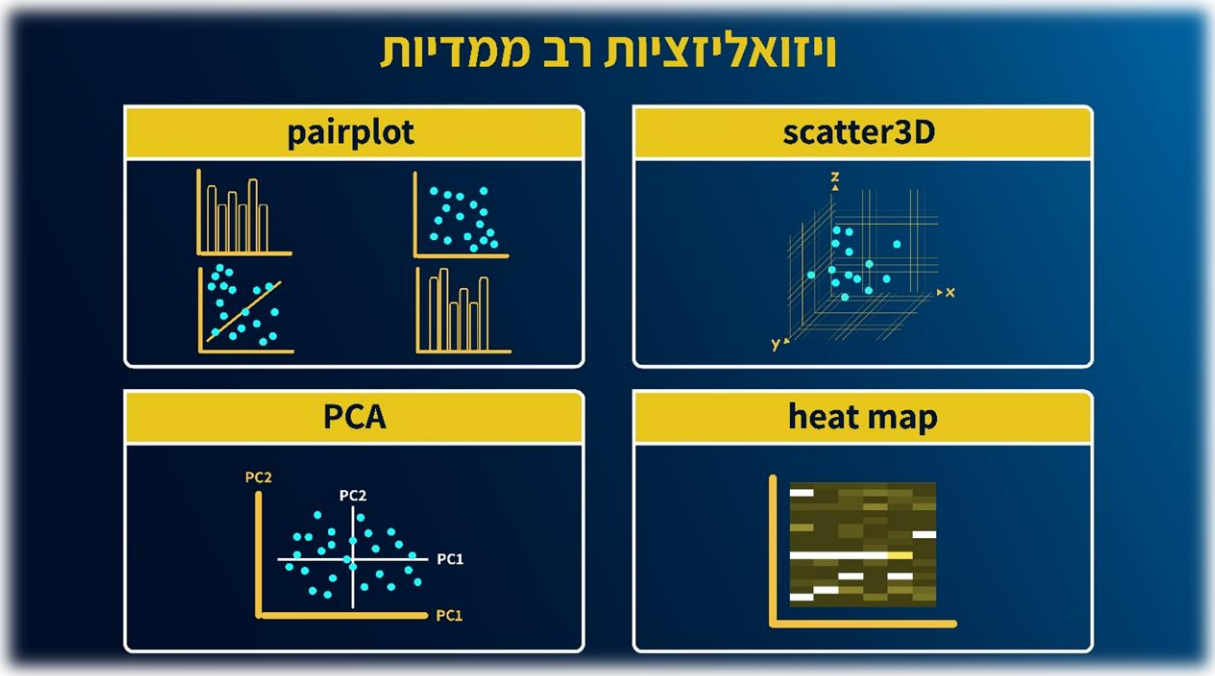
ניתוח חקרני של נתונים (או EDA – Exploratory Data Analysis, אקספלורציה) הוא השלב האחרון בטיפול בנתונים, והוא חשוב מאוד. EDA כולל התבוננות ויזואליות בנתונים (באופן אובייקטיבי), מבחנים סטטיסטיים וגיבוש תובנות לגבי הנתונים. למה ויזואליזציה? כדי להסתכל על כמות גדולה של נתונים. הוויזואליזציה עוזרת לזיהוי תבניות ומגמות, וזיהוי תבניות חריגות ויוצאות דופן.

סוגי ויזואליזציות:

- חד־ממדיות: הצגה גרפית של משתנה/מאפיין אחד.
- דו־ממדיות: הצגה גרפית של שני משתנים/מאפיינים.
- רב־ממדיות: הצגה של מספר רב של משתנים/מאפיינים.

כדי להציג cross tabulation, טבלה שמסכמת שני מאפיינים, נשתמש ב־pd.crosstab(df[f1], df[f2]). אפשר לנרמל את המאפיינים בטבלה באמצעות normalize='index' (לפי שורות).

חד־ממדיות

דו־ממדיות	מבחנים סטטיסטיים לבדיקת אי־תלות
רב־ממדיות	

אובייקט figure:

אובייקט זה מאפשר לנו לקבוע את גודל הגרף, הרזולוציה, צבע הרקע, התנהגות הצירים, להגדיר את ה־legend ולחלק ל־subplots.

יצירת figure	fig = plt.figure(figsize=(x_size, y_size))
הוספת subplot	new_sub = fig.add_subplot(nrows=1, ncols=2, index=1)
הגדרת הוויזואליזציה שתוצג ב־subplot	new_sub.hist(df.age, bins=20) (דוגמה)
הגדרת הכותרת	new_sub.set_title(title)
הגדרת ה־x וה־y	new_sub.set_xlabel(feature) / new_sub.set_ylabel(feature)
הגדרת צבע רקע	new_sub.set_facecolor(color)

מבחנים סטטיסטיים לבדיקת אי־תלות (Chi-Square Test of Independence):

מבחני חי בריבוע עוזרים לנו לבדוק האם השערה (H_0) היא נכונה או לא. אם נקבל בתוצאות המבחנים מספר קטן /שווה ל־0.05 (מוסכמה), ההשערה אינה נכונה ונצטרך לאמץ את ההשערה האלטרנטיבית (H_1). (הערה: H_0 יהיה השערה כי אין תלות, ואילו H_1 יהיה השערה כי יש תלות).

מימוש עם פייתון:

```
from scipy.stats import chi2_contingency
chi2_contingency(ctab)  # ctab is a cross tabulation between the two variables.
```

כדי להציג גרפים רב־ממדיים, נשתמש ב־pairplot ו־heatmap. לתלת־ממד נשתמש ב־scatter3D.

מימוש עם פייתון (3D):

```
From mpl_toolkits.mplot3d import Axes3D
ax = plt.axes(projection='3d')  # I want a 3D graph!
plt.x/ylabel(label)  # If you want.
ax.scatter3D(xdata, ydata, c=zdata, depthshade=False)
```

מימוש עם פייתון (רב־ממד):

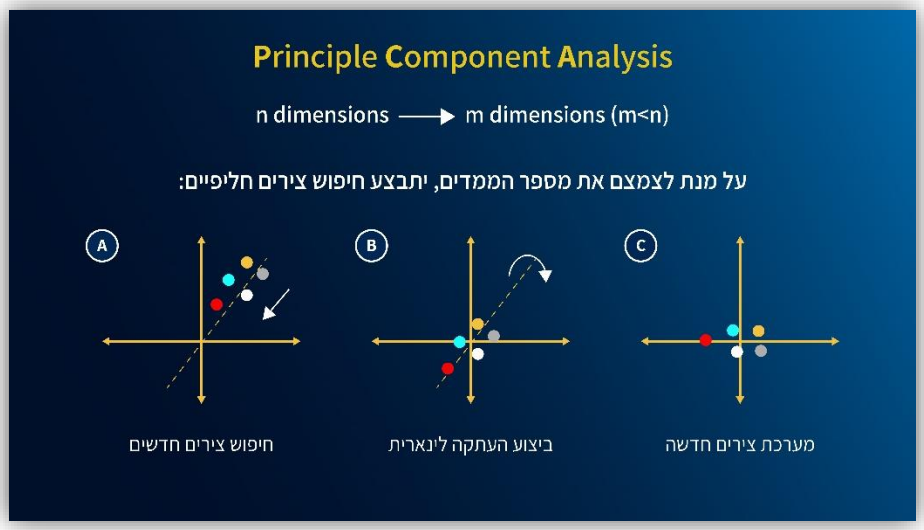
```
import seaborn as sns
sns.pairplot(df[[feature1, feature2, feature3, feature4, ...]]
```

יש לנו מלא רכיבים? כדי לצמצם אותם לשניים או שלושה ממדים נשתמש ב־PCA (Principal Component Analysis).

מספר הממדים הראשוני = n.
מספר המשתנים בדטה (לאחר צמצום) = m.

יש לשים לב:

- המשתנים נמדדו באותן יחידות מדידה.
- סוג המשתנים צריכים להיות משתני רווח/מנה.



Discrete Variables Notebook | Continuums Variables Notebook | Heat maps Notebook | PCA Notebook |