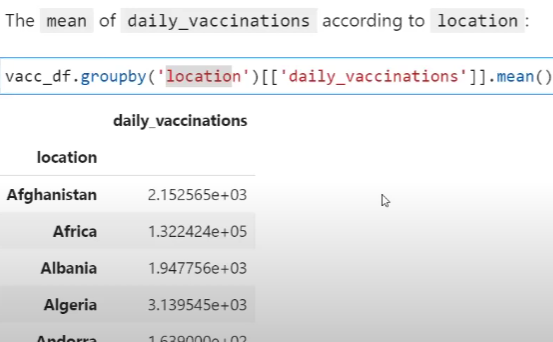
**שאלות אפשריות בניתוח**

מה הממוצע של החיסונים היומיים לפי מיקום?

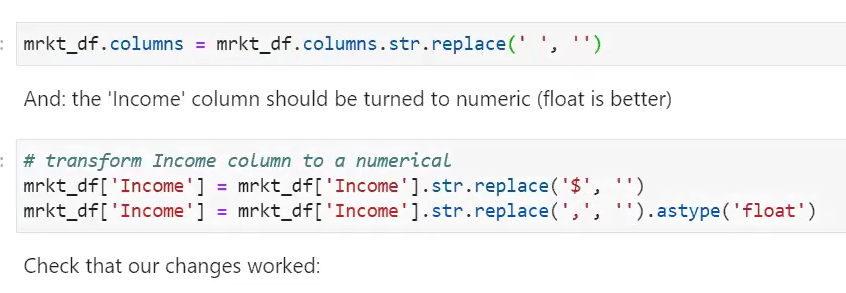
vacc\_df.groupby('location')[['total\_vaccinations']].mean()



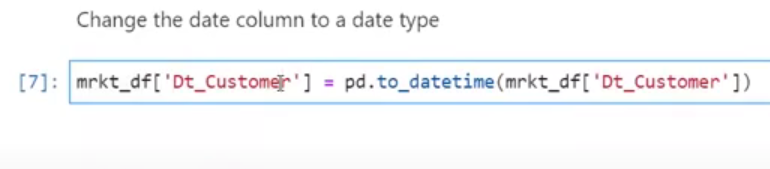
במידה ויש לי עמודה שעם פסיקים או דולר ונרצה לטפל בה נעשה את זה באופן הבא:

mrkt\_df['Income'] = mrkt\_df['Income'].str.replace('$', '')

mrkt\_df['Income'] = mrkt\_df['Income'].str.replace(',', '').astype('float')



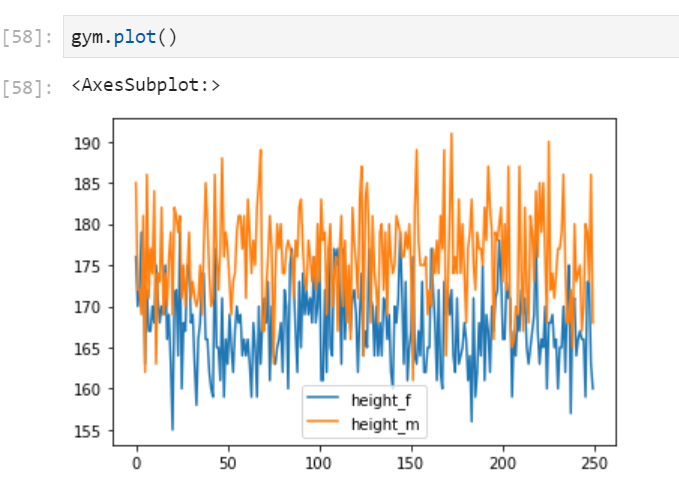
כדי לשנות מתאריך שהוא אובייקט לתאריך ממשי נעשה את זה ככה

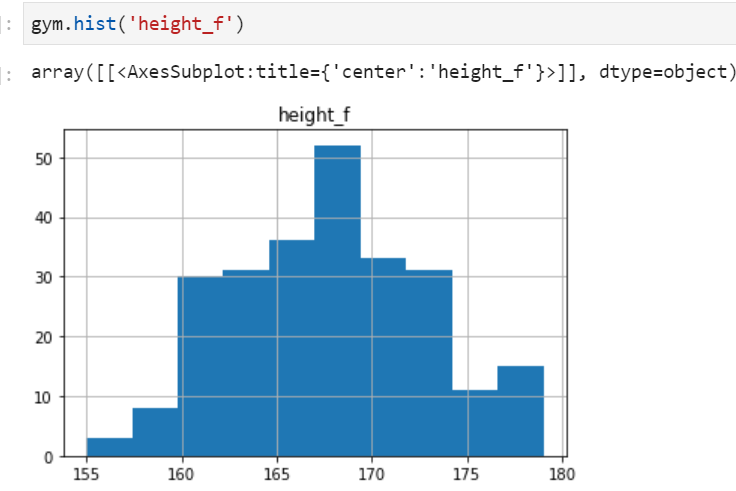
mrkt\_df['Dt\_Customer'] = pd.to\_datetime(mrkt\_df['Dt\_Customer'])

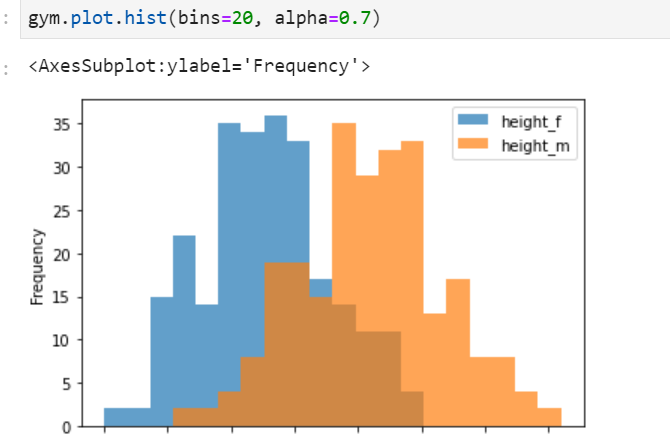
**היסטוגרמה**

נשתמש בהיסטוגרמה כדי להשוות בין שני פרמטרים .

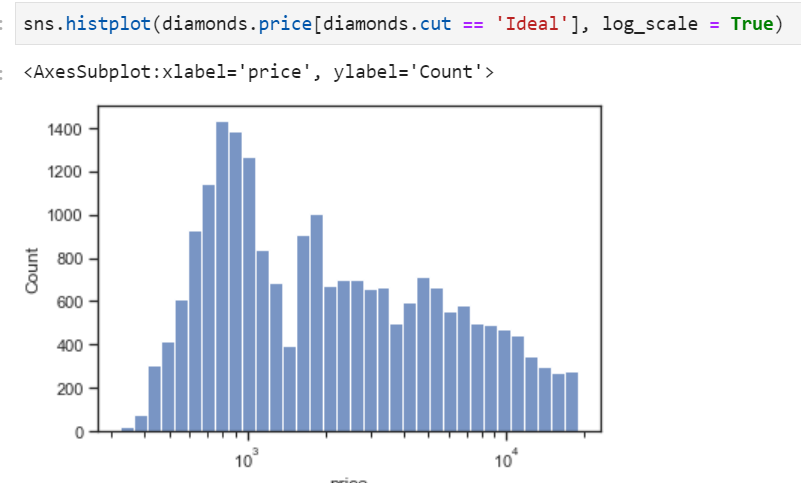
סוגי היסטוגרמה:

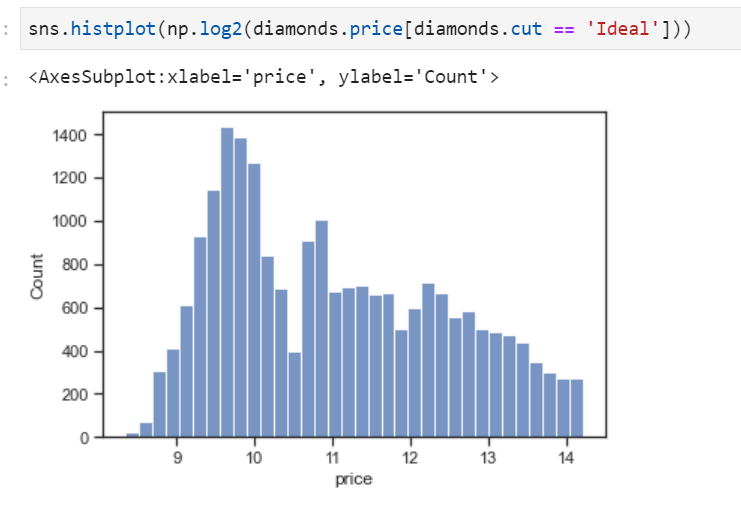






נקודות על היסטוגרמה:  
bins=טווח\מדרגות  
אלפא=שקיפות   
נשתמש בplot.hist שיציג לנו שני פרמטרים בתמונה אחת  
נשתמש ב.hist שנרצה להציג בשני תמונות נפרדות  
במידה ונרצה לשנות את טווח ציר האיקס ללוג 10 נשתמש בבפקודה הזו:

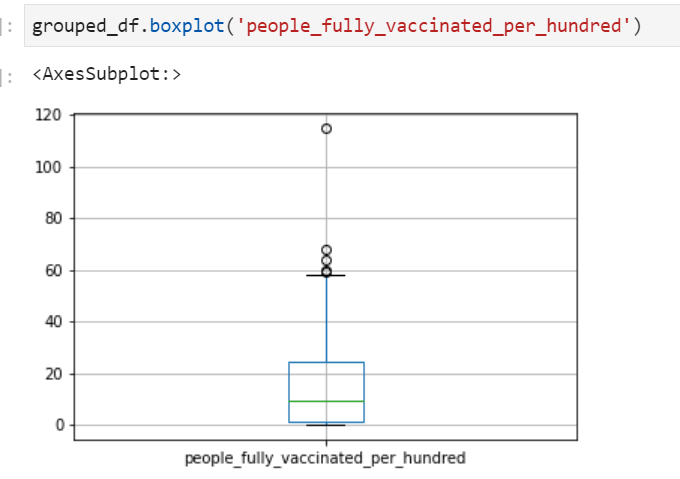


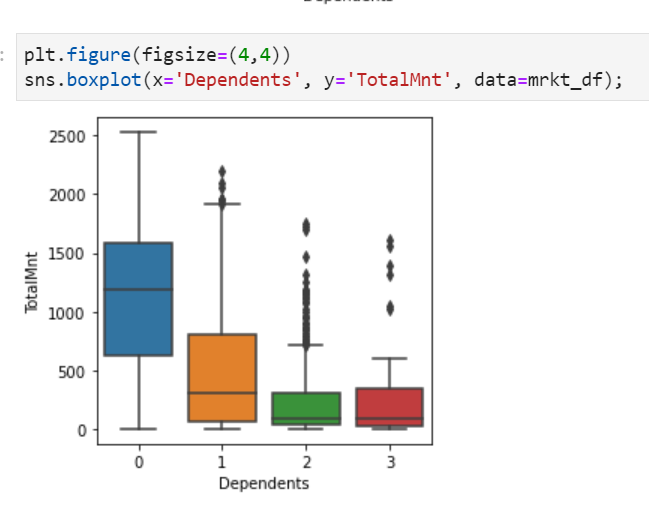


**בוקס פלוט**

נשתמש בבוקס פלוט שנרצה לראות פיזור של הנתונים .הקו בתוך הבוקס פלוט יציג לנו את החציון, והנקודות יציגו את סטיית התקן.

grouped\_df.boxplot('people\_fully\_vaccinated\_per\_hundred')

plt.figure(figsize=(4,4))

sns.boxplot(x='Dependents', y='TotalMnt', data=mrkt\_df);

plt.figure(figsize=(15,10))

plt.subplot(2,2,1)

sns.boxplot(x = 'class', y = 'sepal\_length', data = iris\_df, palette="gist\_ncar\_r")

plt.subplot(2,2,2)

sns.boxplot(x = 'class', y = 'sepal\_width', data = iris\_df, palette="gist\_ncar\_r")

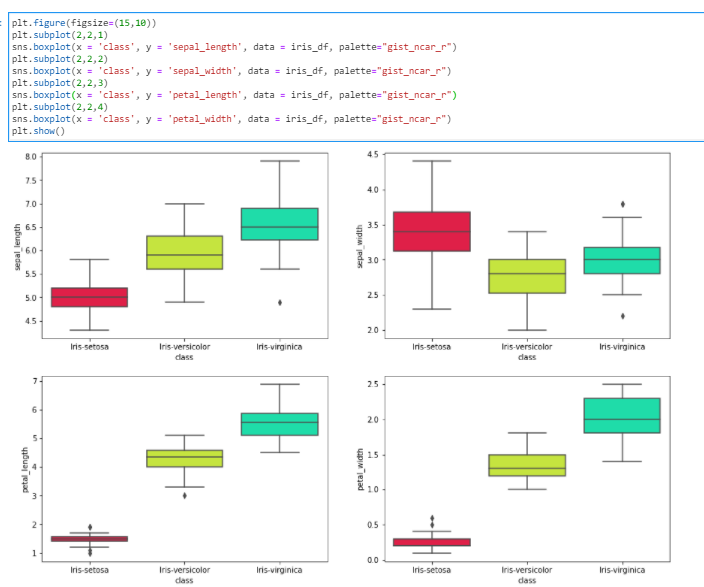
plt.subplot(2,2,3)

sns.boxplot(x = 'class', y = 'petal\_length', data = iris\_df, palette="gist\_ncar\_r")

plt.subplot(2,2,4)

sns.boxplot(x = 'class', y = 'petal\_width', data = iris\_df, palette="gist\_ncar\_r")

plt.show()



**קורולציה**

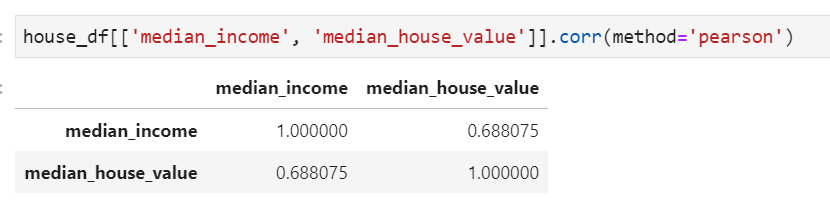
נבדוק בין שני משתנים והם ינועו בין מינוס 1 ל1.

נרשום את זה כציר איקס שווה וציר וואי שווה.

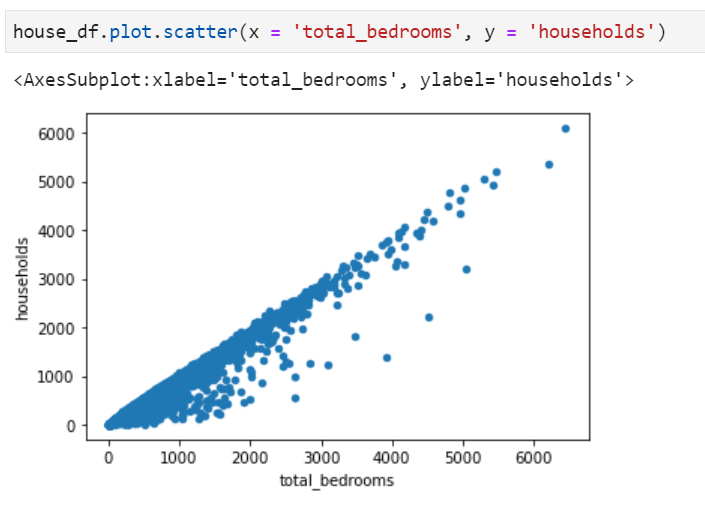
יש מספר סוגי קורלציה :פירסון ,ספירמן וקנדל.

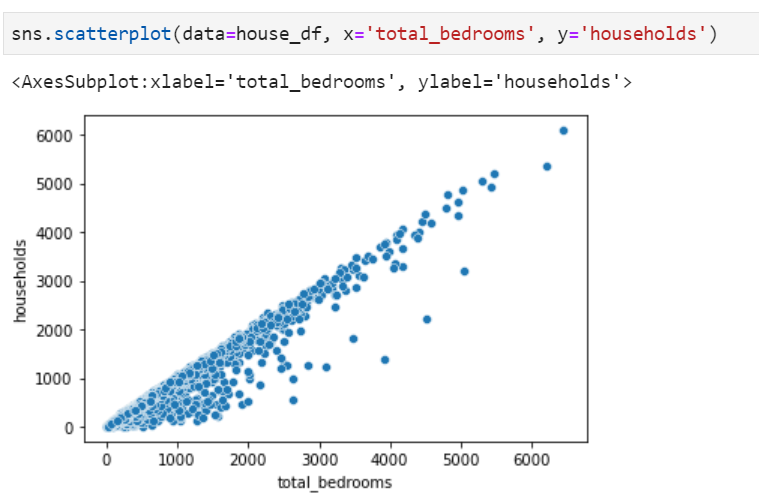
הברירת מחדל היא פירסון

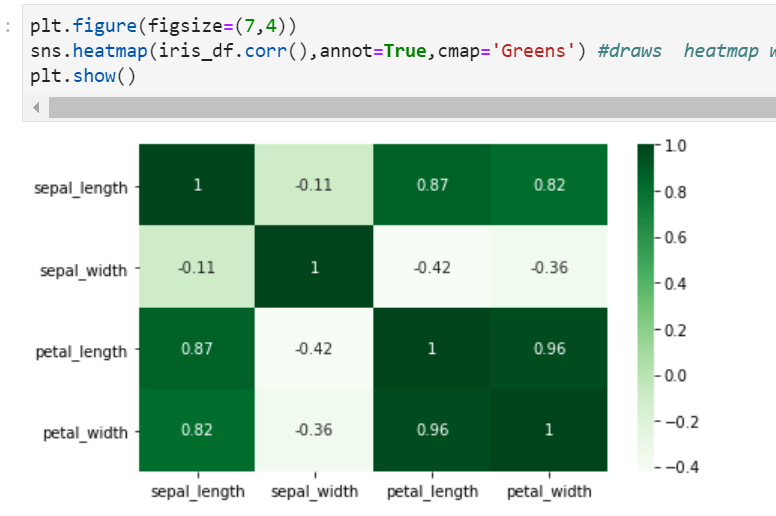
house\_df[['median\_income', 'median\_house\_value']].corr(method='pearson')



house\_df.plot.scatter(x = 'total\_bedrooms', y = 'households')



sns.scatterplot(data=house\_df, x='total\_bedrooms', y='households')

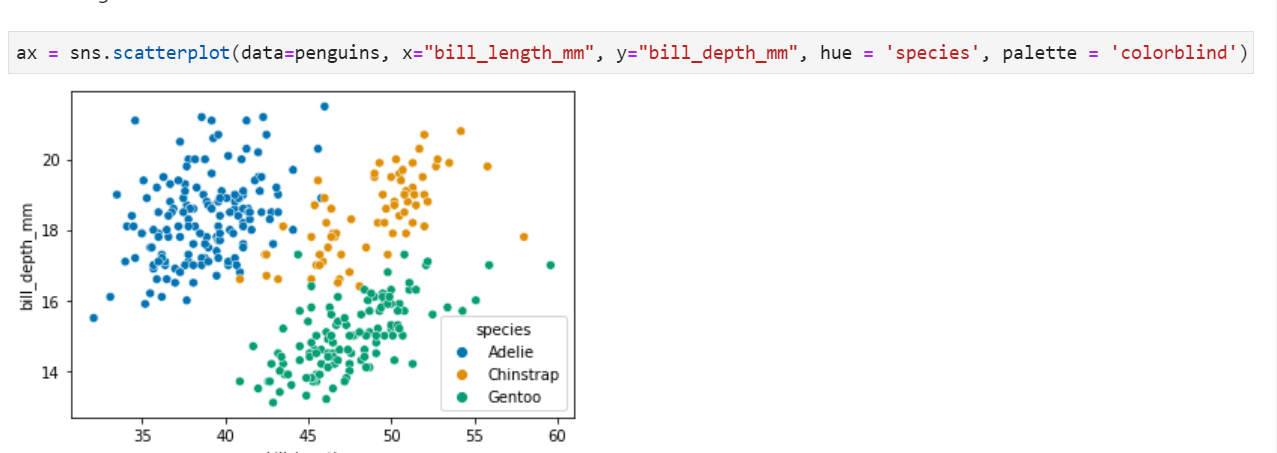


plt.figure(figsize=(7,4))

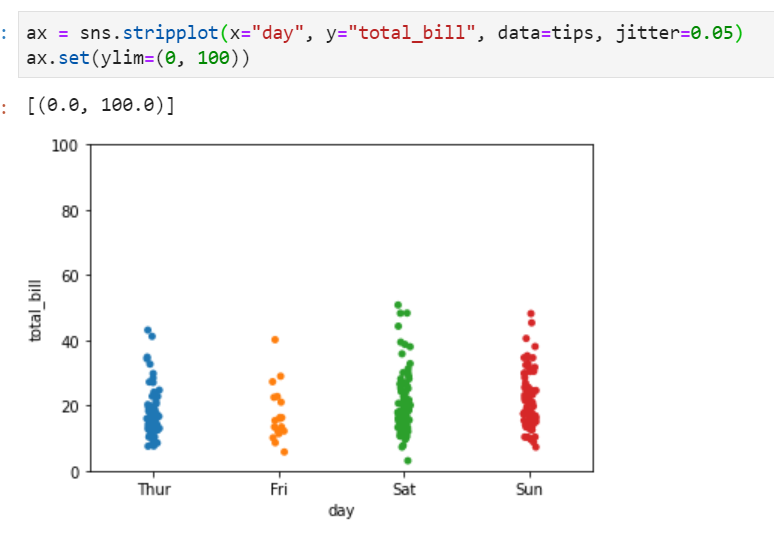
sns.heatmap(iris\_df.corr(),annot=True,cmap='Greens') #draws heatmap with input as the correlation matrix calculted by(iris.corr())

plt.show()

נשתמש בהיטמפ כאשר נרצה להציג את הקורולוציה בצורה ויזואלית ולא במלל

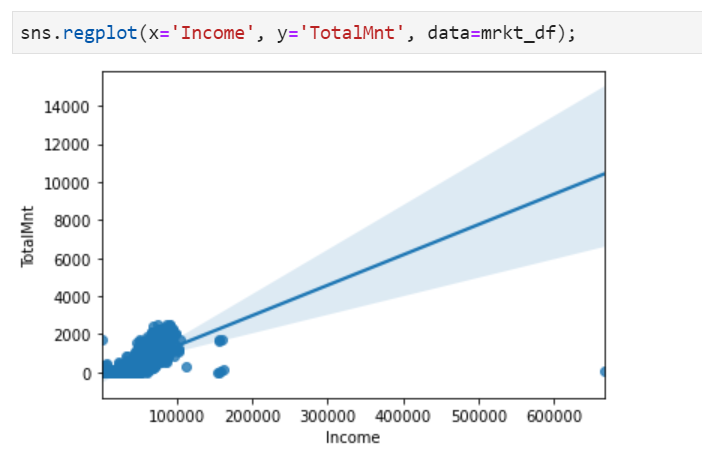


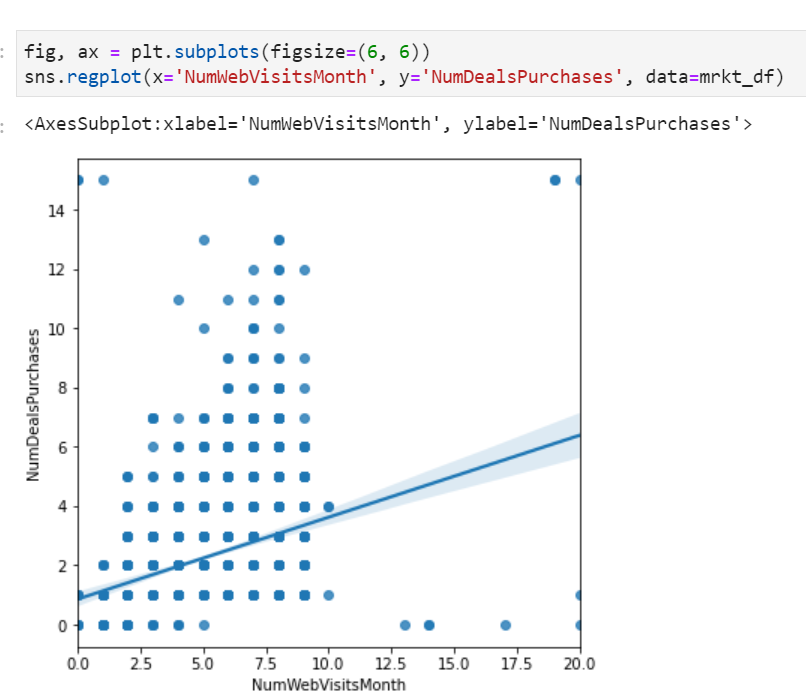
ax = sns.scatterplot(data=penguins, x="bill\_length\_mm", y="bill\_depth\_mm", hue = 'species', palette = 'colorblind')

 ax = sns.stripplot(x="day", y="total\_bill", data=tips, jitter=0.05)

ax.set(ylim=(0, 100))

נשתמש בסטריפלוט להראות תצוגה אנכית נוכל לשלוט בצירים איקס וואי ולהגדיר את הטווח שלהם

sns.regplot(x='Income', y='TotalMnt', data=mrkt\_df);

fig, ax = plt.subplots(figsize=(6, 6))

sns.regplot(x='NumWebVisitsMonth', y='NumDealsPurchases', data=mrkt\_df)

**בר פלוט קאונט פלוט וקאט פלוט**

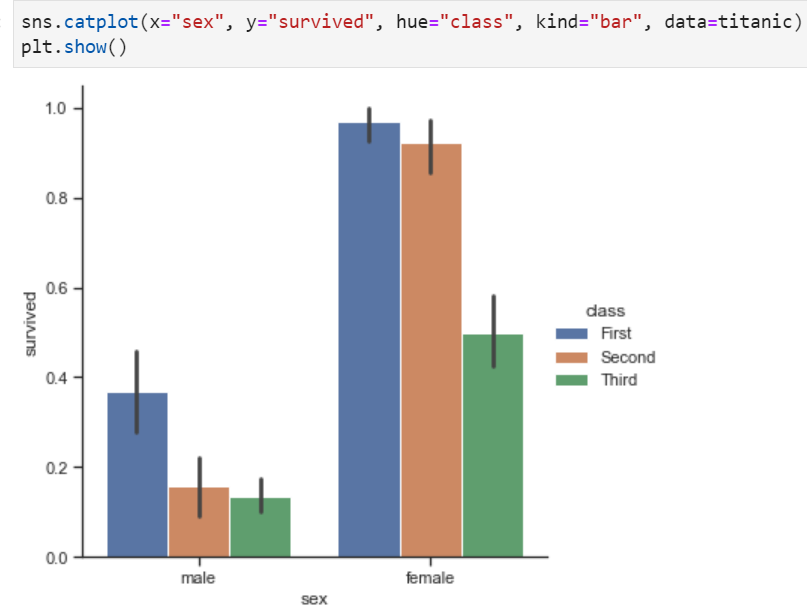
דיאגרמת עמודות הקו השחור יציג לנו סטיית תקן.

sns.barplot(x="sex", y="survived", hue="class", data=titanic)

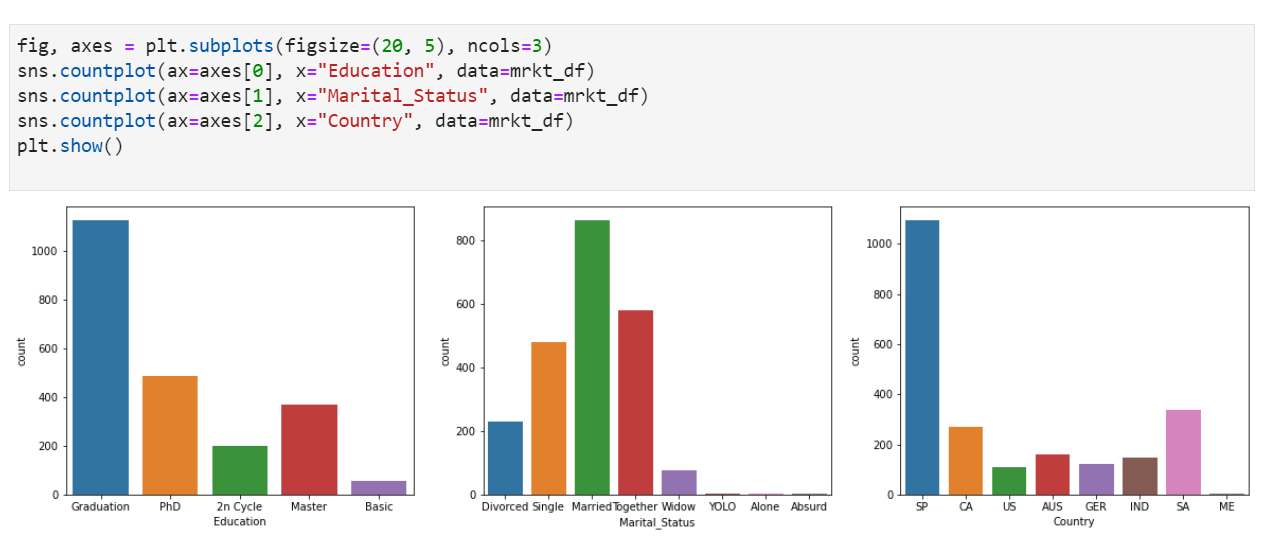
plt.show()

sns.countplot(x="sex", hue="class", data=titanic)

plt.show()

sns.catplot(x="sex", y="survived", hue="class", kind="bar", data=titanic)

plt.show()



fig, axes = plt.subplots(figsize=(20, 5), ncols=3)

sns.countplot(ax=axes[0], x="Education", data=mrkt\_df)

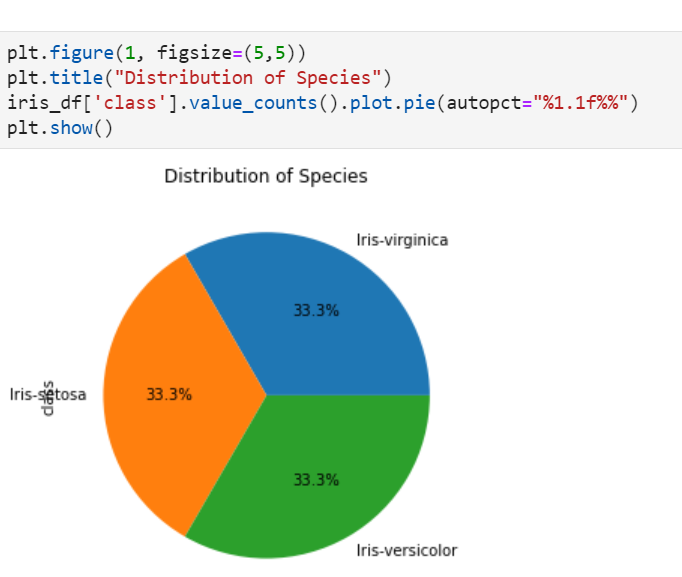
sns.countplot(ax=axes[1], x="Marital\_Status", data=mrkt\_df)

sns.countplot(ax=axes[2], x="Country", data=mrkt\_df)

plt.show()

**פאי**

נציג בצורת פאי מספר משתנים עדיפות עד 2 3 משתנים ,ההצגה בצורת עוגה וניתן לעשות עם אחוזים וגם ללא אחוזים רצוי עם.



plt.figure(1, figsize=(5,5))

plt.title("Distribution of Species")

iris\_df['class'].value\_counts().plot.pie(autopct="%1.1f%%")

plt.show()

**עצי החלטה**

ישנם מספר סוגי עצים:

(מלל)Classifier )ACC(-סוג זה מיועד לנתונים בדידים

(מספרי)Regressor-סוג זה מיועד לנתונים רציפים )MCE(

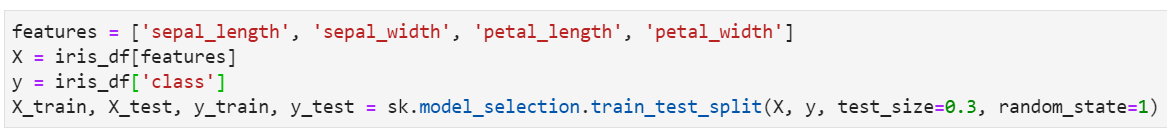
**Classifier**

תחילה נייבא את הספריה :

from sklearn.tree import DecisionTreeClassifier

from sklearn import tree

לאחר שייבאנו את הספריות נגדיר איקס ואי שאיקס מייצג את הפיצרים וואי מייצג את הלייבל.



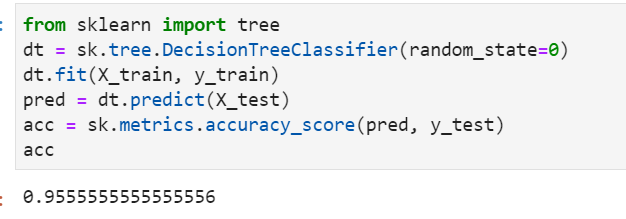
features = ['sepal\_length', 'sepal\_width', 'petal\_length', 'petal\_width']

X = iris\_df[features]

y = iris\_df['class']

X\_train, X\_test, y\_train, y\_test = sk.model\_selection.train\_test\_split(X, y, test\_size=0.3, random\_state=1)

בוחרים עץ לפי הנתונים שרשמנו למעלה, מאמנים את העץ ואז מנבאים לפי פרדיקט ואז נבדוק דיוק

 from sklearn import tree

dt = sk.tree.DecisionTreeClassifier(random\_state=0)

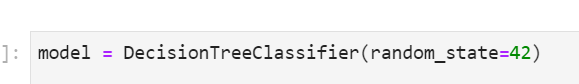
dt.fit(X\_train, y\_train)

pred = dt.predict(X\_test)

acc = sk.metrics.accuracy\_score(pred, y\_test)

acc

במידה ואני מתבקשת לעומק העץ לאחר ההגדרת מודל sk.tree.DecisionTreeClassifier נזין את העומק  
model = DecisionTreeClassifier(max\_depth=4,random\_state=42)



**Regressor**

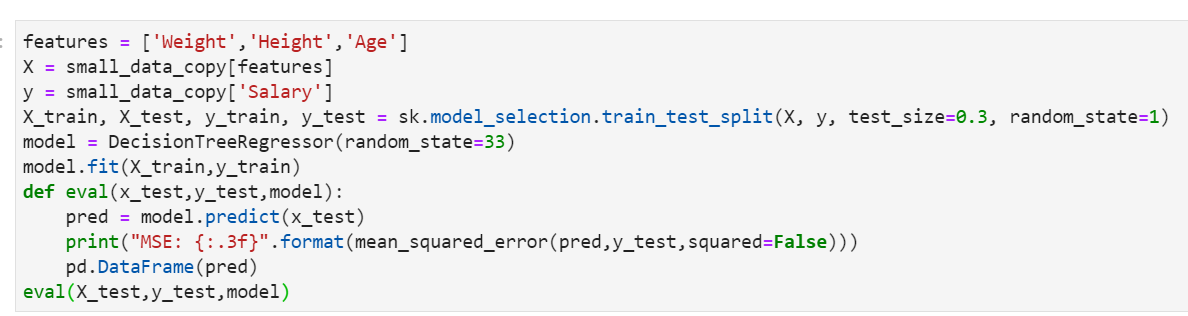
תחילה נייבא את הספריות:

from sklearn import tree

from sklearn.metrics import mean\_squared\_error

from sklearn.tree import DecisionTreeRegressor

from sklearn.metrics import mean\_squared\_errorלאחר שייבאנו את הספריות נגדיר איקס ואי שאיקס מייצג את הפיצרים וואי מייצג את הלייבל.



features = ['Weight','Height','Age']

X = small\_data\_copy[features]

y = small\_data\_copy['Salary']

X\_train, X\_test, y\_train, y\_test = sk.model\_selection.train\_test\_split(X, y, test\_size=0.3, random\_state=1)

model = DecisionTreeRegressor(random\_state=33)

model.fit(X\_train,y\_train)

def eval(x\_test,y\_test,model):

pred = model.predict(x\_test)

print("MSE: {:.3f}".format(mean\_squared\_error(pred,y\_test,squared=False)))

pd.DataFrame(pred)

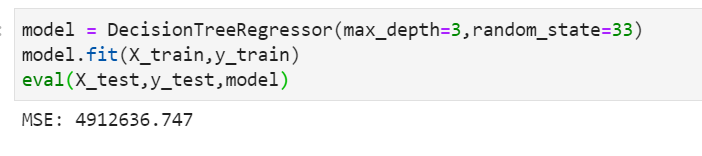
eval(X\_test,y\_test,model)

כדי להגדיר מקס דפט:

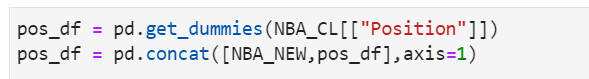
model = DecisionTreeRegressor(max\_depth=3,random\_state=33)

model.fit(X\_train,y\_train)

eval(X\_test,y\_test,model)



עוד דרך להפוך לקטגורי:



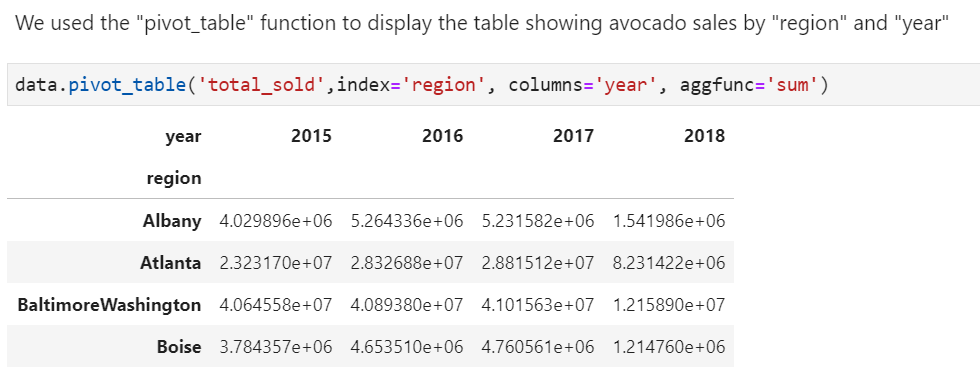
pos\_df = pd.get\_dummies(NBA\_CL[["Position"]])

pos\_df = pd.concat([NBA\_NEW,pos\_df],axis=1)

**בשביל להפוך מידע לקטגורי (ממלל למספרים)**

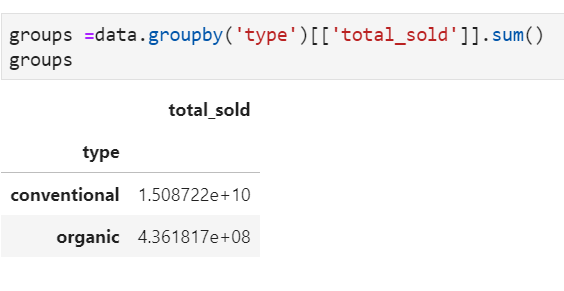
 data["type"]=data["type"].astype("category").cat.codes

**פיבוט טייבל**- טבלה לפי קריטריונים. תחילה נגדיר את העמודה עליה נרצה לעבוד באינדקס.



data.pivot\_table('total\_sold',index='region', columns='year', aggfunc='sum')

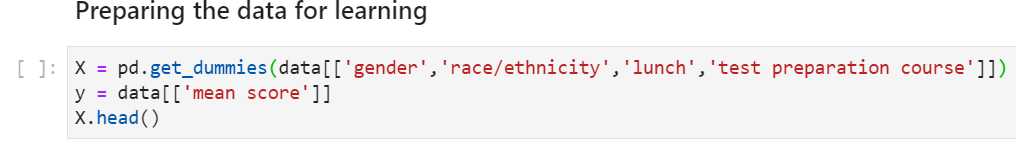
**גרופ ביי-** מקבץ את הרשומות לקבוצות (ע"פ מה לקבץ את הרשומות – ע"פ העמודה שאני בחרתי

****

groups =data.groupby('type')[['total\_sold']].sum()

groups

**get\_dummies**- זו פונקציה שמשנה ממילים למספרים כמו למשל גברים נשים ל-1,0



X = pd.get\_dummies(data[['gender','race/ethnicity','lunch','test preparation course']])

y = data[['mean score']]

X.head()