**NutriClick Exploratory Data Analysis**

Smart personalized nutrition system, providing timely nutritional advice to guide diet patterns.

Where non-communicable diseases (NCDs) are estimated to be responsible for 71% of all deaths (41 million people per annum) globally. There is a strong association between NCDs and diet; properly designed nutrition has the potential to significantly reduce the effects and occurrence of NCDs, and nutrition could play a significant role in achieving this reduction. (WHO,2019)

and lack of personalized systems tailored to the middle east region that can monitor what food is eaten or should be eaten and can advise on portion size and nutrition based on measurement of recent physical activity, all of this encouraged this idea.

\*Read Dataset file and convert to DataFrame and choice best columns and replace null value to (0) and display data

#read csv  
nutritionData = pd.read\_csv('dataset/nutrition.csv')  
  
dfnutritionData=pd.DataFrame(nutritionData)  
#fill zero in null value  
dfnutritionData.fillna(0)  
#print(dfnutritionData.head())  
  
finalData=dfnutritionData.drop(['saturated\_fat','tocopherol\_alpha',  
 'vitamin\_k', 'calcium', 'copper', 'irom', 'magnesium', 'manganese',  
 'phosphorous', 'potassium', 'selenium', 'zink', 'protein', 'alanine',  
 'arginine', 'aspartic\_acid', 'cystine', 'glutamic\_acid', 'glycine',  
 'histidine', 'hydroxyproline', 'isoleucine', 'leucine', 'lysine',  
 'methionine', 'phenylalanine', 'proline', 'serine', 'threonine',  
 'tryptophan', 'tyrosine', 'valine', 'fiber',  
 'fructose', 'galactose', 'glucose', 'lactose', 'maltose', 'sucrose',  
 'fat', 'saturated\_fatty\_acids', 'monounsaturated\_fatty\_acids',  
 'polyunsaturated\_fatty\_acids', 'fatty\_acids\_total\_trans', 'alcohol',  
 'ash', 'caffeine', 'theobromine'],axis = 1)  
  
print(finalData)

\*Choice name and calories column and print First 100 foods have maximum calories Quantity

###get claroies and food name  
cal\_food=finalData[['name','calories']]  
sortDataCalories =cal\_food.sort\_values(by=['calories'],ascending=False)  
##print First 100 foods have maximum calories Quantity  
print(sortDataCalories.head(100))

\*Get total fat and print first 100 food have maximum total fat

###get total\_fat and food name  
#remove (g) from fat  
finalData['total\_fat'] = finalData['total\_fat'].map(lambda x: x.rstrip('g'))  
#convert to int  
finalData.total\_fat = pd.to\_numeric(finalData.total\_fat, errors='coerce')  
fat\_food=finalData[['name','total\_fat','calories']]  
sortDataFat =fat\_food.sort\_values(by=['total\_fat'],ascending=False)  
##check top calories  
DataCal=dfnutritionData[['name','calories']]  
sortDataCal =DataCal.sort\_values(by=['calories'],ascending=False)  
##print First 100 foods have maximum total\_fat  
#print(sortDataFat.head(100))  
TopFat=sortDataFat.head(100)  
#print(TopFat)  
##query Ex:calories > 100 & total\_fat < 10  
cap = finalData.query('calories > 100 & total\_fat < 10')

\*Get top protein and low calories

### check top protein and low calories  
TopProtLowCal=dfnutritionData[['name','protein','calories']]  
SortTopProtLowCal=TopProtLowCal.head(10);  
#print(SortTopProtLowCal)

\* Bring the highest sugar-containing foods

## check max food-sugars :  
TopSug=dfnutritionData[['name','sugars']]   
TopSug['sugars'] = TopSug['sugars'].map(lambda x: x.rstrip('g'))  
TopSug.sugars = pd.to\_numeric(TopSug.sugars, errors='coerce')  
TopSug['sugars']=TopSug['sugars'].fillna(0).astype(np.int64)  
TopSug=TopSug.sort\_values(by=['sugars'],ascending=False)  
TopTenSug=TopSug.head(10)  
print(TopTenSug)

\* Bring the highest protein-containing foods

## check max food-protein :  
TopProt=dfnutritionData[['name','protein']]   
TopProt['protein'] = TopProt['protein'].map(lambda x: x.rstrip('g'))  
TopProt.protein = pd.to\_numeric(TopProt.protein, errors='coerce')  
TopProt['protein']=TopProt['protein'].fillna(0).astype(np.int64)  
TopProt=TopProt.sort\_values(by=['protein'],ascending=False)  
TopTenProt=TopProt.head(10)  
print(TopTenProt)

\* Bring the highest carbohydrate-containing foods

TopCarb=dfnutritionData[['name','carbohydrate']]   
TopCarb['carbohydrate'] = TopCarb['carbohydrate'].map(lambda x: x.rstrip('g'))  
TopCarb.carbohydrate = pd.to\_numeric(TopCarb.carbohydrate, errors='coerce')  
TopCarb['carbohydrate']=TopCarb['carbohydrate'].fillna(0).astype(np.int64)  
TopCarb=TopCarb.sort\_values(by=['carbohydrate'],ascending=False)  
TopTenCarb=TopCarb.head(10)  
print(TopTenCarb)

\* Bring the highest total-fat-containing foods

## check max food-total\_fat :  
TopFat=dfnutritionData[['name','total\_fat']]   
TopFat['total\_fat'] = TopFat['total\_fat'].map(lambda x: x.rstrip('g'))  
TopFat.total\_fat = pd.to\_numeric(TopFat.total\_fat, errors='coerce')  
TopFat['total\_fat']=TopFat['total\_fat'].fillna(0).astype(np.int64)  
TopFat=TopFat.sort\_values(by=['total\_fat'],ascending=False)  
TopTenFat=TopFat.head(10)  
print(TopTenFat)

//try compare between multiple food containing like below

\* Compare between max fat and max calories

##compare max fat and max calories  
  
MaxCalMaxFat=dfnutritionData[['name','total\_fat','calories']]  
MaxCalMaxFat['total\_fat'] = MaxCalMaxFat['total\_fat'].map(lambda x: x.rstrip('g'))  
MaxCalMaxFat.total\_fat = pd.to\_numeric(MaxCalMaxFat.total\_fat, errors='coerce')  
MaxCalMaxFat['total\_fat']=MaxCalMaxFat['total\_fat'].fillna(0).astype(np.int64)  
MaxCalMaxFat=MaxCalMaxFat.sort\_values(by=['total\_fat'],ascending=False)  
MaxCalMaxFat=MaxCalMaxFat.sort\_values(by=['calories'],ascending=False)  
MaxCalMaxFat=MaxCalMaxFat.head(10)  
print(MaxCalMaxFat)

##compare max fat and Low protein?????  
  
MaxFatMinProt=dfnutritionData[['name','total\_fat','protein']]  
MaxFatMinProt['total\_fat'] = MaxFatMinProt['total\_fat'].map(lambda x: x.rstrip('g'))  
MaxFatMinProt['protein'] = MaxFatMinProt['protein'].map(lambda x: x.rstrip('g'))  
MaxFatMinProt.total\_fat = pd.to\_numeric(MaxFatMinProt.total\_fat, errors='coerce')  
MaxFatMinProt.protein = pd.to\_numeric(MaxFatMinProt.protein, errors='coerce')  
MaxFatMinProt['total\_fat']=MaxFatMinProt['total\_fat'].fillna(0).astype(np.int64)  
MaxFatMinProt['protein']=MaxFatMinProt['protein'].fillna(0).astype(np.int64)  
MaxFatMinProt=MaxFatMinProt.sort\_values(['total\_fat'], ascending=[False])  
MaxFatMinProt=MaxFatMinProt.head(10)  
print(MaxFatMinProt)  
  
  
  
##compare Min fat and Max protein?????  
  
MainFatMaxProt=dfnutritionData[['name','total\_fat','protein']]  
MainFatMaxProt['total\_fat'] = MainFatMaxProt['total\_fat'].map(lambda x: x.rstrip('g'))  
MainFatMaxProt['protein'] = MainFatMaxProt['protein'].map(lambda x: x.rstrip('g'))  
MainFatMaxProt.total\_fat = pd.to\_numeric(MainFatMaxProt.total\_fat, errors='coerce')  
MainFatMaxProt.protein = pd.to\_numeric(MainFatMaxProt.protein, errors='coerce')  
MainFatMaxProt['total\_fat']=MainFatMaxProt['total\_fat'].fillna(0).astype(np.int64)  
MainFatMaxProt['protein']=MainFatMaxProt['protein'].fillna(0).astype(np.int64)  
MainFatMaxProt=MainFatMaxProt.sort\_values(['protein'], ascending=[False])  
MainFatMaxProt=MainFatMaxProt.head(10)  
print(MainFatMaxProt)  
  
  
  
##compare max fat and Max calories and min Protein  
  
MaxFatMaxCalMinProt=dfnutritionData[['name','total\_fat','protein','calories']]  
MaxFatMaxCalMinProt['total\_fat'] = MaxFatMaxCalMinProt['total\_fat'].map(lambda x: x.rstrip('g'))  
MaxFatMaxCalMinProt['protein'] = MaxFatMaxCalMinProt['protein'].map(lambda x: x.rstrip('g'))  
MaxFatMaxCalMinProt.total\_fat = pd.to\_numeric(MaxFatMaxCalMinProt.total\_fat, errors='coerce')  
MaxFatMaxCalMinProt.protein = pd.to\_numeric(MaxFatMaxCalMinProt.protein, errors='coerce')  
MaxFatMaxCalMinProt['total\_fat']=MaxFatMaxCalMinProt['total\_fat'].fillna(0).astype(np.int64)  
MaxFatMaxCalMinProt['protein']=MaxFatMaxCalMinProt['protein'].fillna(0).astype(np.int64)  
MaxFatMaxCalMinProt=MaxFatMaxCalMinProt.sort\_values(['total\_fat','calories','protein'],  
 ascending=[False,False,True])  
MaxFatMaxCalMinProt=MaxFatMaxCalMinProt.head(30)  
print(MaxFatMaxCalMinProt)

\*Bring the food which contains milk and bring food contains and any item like below

##Check Food that contain milk  
  
SortByNAmeMilk=dfnutritionData[['name','total\_fat','protein']]   
SortByNAmeMilk['total\_fat'] = SortByNAmeMilk['total\_fat'].map(lambda x: x.rstrip('g'))  
SortByNAmeMilk['protein'] = SortByNAmeMilk['protein'].map(lambda x: x.rstrip('g'))  
SortByNAmeMilk.total\_fat = pd.to\_numeric(SortByNAmeMilk.total\_fat, errors='coerce')  
SortByNAmeMilk.protein = pd.to\_numeric(SortByNAmeMilk.protein, errors='coerce')  
SortByNAmeMilk['total\_fat']=SortByNAmeMilk['total\_fat'].fillna(0).astype(np.int64)  
SortByNAmeMilk['protein']=SortByNAmeMilk['protein'].fillna(0).astype(np.int64)  
  
SortByNAmeMilk=SortByNAmeMilk[SortByNAmeMilk['name'].str.contains('milk')]  
SortByNAmeMilk=SortByNAmeMilk.sort\_values(['total\_fat'], ascending=[False])  
SortByNAmeMilk=SortByNAmeMilk.head(10)  
print(SortByNAmeMilk)  
  
  
  
##Check Food that contain oil  
  
SortByNAmeOil=dfnutritionData[['name','total\_fat','protein']]  
SortByNAmeOil['total\_fat'] = SortByNAmeOil['total\_fat'].map(lambda x: x.rstrip('g'))  
SortByNAmeOil['protein'] = SortByNAmeOil['protein'].map(lambda x: x.rstrip('g'))  
SortByNAmeOil.total\_fat = pd.to\_numeric(SortByNAmeOil.total\_fat, errors='coerce')  
SortByNAmeOil.protein = pd.to\_numeric(SortByNAmeOil.protein, errors='coerce')  
SortByNAmeOil['total\_fat']=SortByNAmeOil['total\_fat'].fillna(0).astype(np.int64)  
SortByNAmeOil['protein']=SortByNAmeOil['protein'].fillna(0).astype(np.int64)  
  
SortByNAmeOil=SortByNAmeOil[SortByNAmeOil['name'].str.contains('oil')]  
SortByNAmeOil=SortByNAmeOil.sort\_values(['total\_fat'], ascending=[False])  
SortByNAmeOil=SortByNAmeOil.head(30)  
print(SortByNAmeOil)  
  
##Check Food that contain Fruit  
  
SortByNAmeFruit=dfnutritionData[['name','total\_fat','protein']]  
SortByNAmeFruit['total\_fat'] = SortByNAmeFruit['total\_fat'].map(lambda x: x.rstrip('g'))  
SortByNAmeFruit['protein'] = SortByNAmeFruit['protein'].map(lambda x: x.rstrip('g'))  
SortByNAmeFruit.total\_fat = pd.to\_numeric(SortByNAmeFruit.total\_fat, errors='coerce')  
SortByNAmeFruit.protein = pd.to\_numeric(SortByNAmeFruit.protein, errors='coerce')  
SortByNAmeFruit['total\_fat']=SortByNAmeFruit['total\_fat'].fillna(0).astype(np.int64)  
SortByNAmeFruit['protein']=SortByNAmeFruit['protein'].fillna(0).astype(np.int64)  
  
SortByNAmeFruit=SortByNAmeFruit[SortByNAmeFruit['name'].str.contains('fruit')]  
SortByNAmeFruit=SortByNAmeFruit.sort\_values(['total\_fat'], ascending=[False])  
SortByNAmeFruit=SortByNAmeFruit.head(10)  
print(SortByNAmeFruit)

**Plotting the data**

\*Plot top 10 foods that have maximum fat

##Plot Top 10 Fat Food in bar graph   
  
name = list(TopTenFat.name)  
fat = list(TopTenFat.total\_fat)  
plt.bar(name, fat, color ='green',  
 width = 0.6)  
plt.xlabel("name")  
plt.ylabel("No.Fat")  
plt.title("Top 10 food-max Fat")  
plt.xticks(rotation=90)  
plt.show()

\*Plot top 10 foods that have maximum protein

##Plot Top 10 Protein Food in Bar graph   
  
name = list(TopTenProt.name)  
fat = list(TopTenProt.protein)  
plt.bar(name, fat, color ='red',  
 width = 0.6)  
plt.xlabel("name")  
plt.ylabel("No.Protein")  
plt.title("Top 10 food-max Protein")  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have maximum carbohydrate

##Plot Top 10 Carbs in Bar graph   
  
name = list(TopTenCarb.name)  
fat = list(TopTenCarb.carbohydrate)  
plt.bar(name, fat, color ='blue',  
 width = 0.6)  
plt.xlabel("name")  
plt.ylabel("No.Carbs")  
plt.title("Top 10 food-max Carbs")  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have maximum calories and fat in stack graph

##Plot Top 10 Cal and Fat in Stack graph   
  
name =list(MaxCalMaxFat.name)  
YFat =list(MaxCalMaxFat.total\_fat)  
ZCal = list(MaxCalMaxFat.calories)  
X\_axis = np.arange(len(name))  
plt.bar(X\_axis - 0.2, YFat, 0.4, label = 'Fat')  
plt.bar(X\_axis + 0.2, ZCal, 0.4, label = 'Calories')  
   
plt.xticks(X\_axis, name)  
plt.xlabel("Names")  
plt.ylabel("Number of Fat and Cal")  
plt.title("Foods with max Fat and max Calories")  
plt.legend()  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have maximum fat and low protein in stack graph

##Plot Top 10 Fat and Low Protein in Stack graph   
  
name =list(MaxFatMinProt.name)  
YFat =list(MaxFatMinProt.total\_fat)  
ZPro = list(MaxFatMinProt.protein)  
X\_axis = np.arange(len(name))  
plt.bar(X\_axis - 0.2, YFat, 0.4, label = 'Fat')  
plt.bar(X\_axis + 0.2, ZPro, 0.4, label = 'Protein')  
   
plt.xticks(X\_axis, name)  
plt.xlabel("Names")  
plt.ylabel("Number of Fat and Protein")  
plt.title("Foods with max Fat and Low Protein")  
plt.legend()  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have low fat and height protein in stack graph

##Plot Top 10 Min Fat and Max Protein in Bar graph   
  
name =list(MainFatMaxProt.name)  
YFat =list(MainFatMaxProt.total\_fat)  
ZPro = list(MainFatMaxProt.protein)  
X\_axis = np.arange(len(name))  
plt.bar(X\_axis - 0.2, YFat, 0.4, label = 'Fat')  
plt.bar(X\_axis + 0.2, ZPro, 0.4, label = 'Protein')  
   
plt.xticks(X\_axis, name)  
plt.xlabel("Names")  
plt.ylabel("Number of Fat and Protein")  
plt.title("Foods with min Fat and max Protein")  
plt.legend(["Fat","Protein"])  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have fat and low protein for foods milk group in stack graph

##Plot Top 10 Fat and Low Protein for foods(Milk group) in Stack graph   
  
name =list(SortByNAmeMilk.name)  
YFat =list(SortByNAmeMilk.total\_fat)  
ZPro = list(SortByNAmeMilk.protein)  
X\_axis = np.arange(len(name))  
plt.bar(X\_axis - 0.2, YFat, 0.4, label = 'Fat')  
plt.bar(X\_axis + 0.2, ZPro, 0.4, label = 'Protein')  
   
plt.xticks(X\_axis, name)  
plt.xlabel("Names")  
plt.ylabel("Number of Fat and Protein")  
plt.title("Foods with max Fat and Low Protein with milks foods")  
plt.legend(["Fat","Protein"])  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have fat and low protein for foods oil group in stack graph

##Plot Top 10 Fat and Low Protein for foods(oil group) in Stack graph   
  
name =list(SortByNAmeOil.name)  
YFat =list(SortByNAmeOil.total\_fat)  
ZPro = list(SortByNAmeOil.protein)  
X\_axis = np.arange(len(name))  
plt.bar(X\_axis - 0.2, YFat, 0.4, label = 'Fat')  
plt.bar(X\_axis + 0.2, ZPro, 0.4, label = 'Protein')  
   
plt.xticks(X\_axis, name)  
plt.xlabel("Names")  
plt.ylabel("Number of Fat and Protein")  
plt.title("Foods with max Fat and Low Protein with Oil foods")  
plt.legend(["Fat","Protein"])  
plt.xticks(rotation=90)  
plt.show()

\* Plot top 10 foods that have fat and low protein for foods fruit group in stack graph

##Plot Top 10 Fat and Low Protein for foods(fruit group) in Stack graph   
  
name =list(SortByNAmeFruit.name)  
YFat =list(SortByNAmeFruit.total\_fat)  
ZPro = list(SortByNAmeFruit.protein)  
X\_axis = np.arange(len(name))  
plt.bar(X\_axis - 0.2, YFat, 0.4, label = 'Fat')  
plt.bar(X\_axis + 0.2, ZPro, 0.4, label = 'Protein')  
   
plt.xticks(X\_axis, name)  
plt.xlabel("Names")  
plt.ylabel("Number of Fat and Protein")  
plt.title("Foods with max Fat and Low Protein with Fruit foods")  
plt.legend(["Fat","Protein"])  
plt.xticks(rotation=90)  
plt.show()

\*Plot top fats in food using liner graph

##plot linear graph for Top Fats in food  
  
name = list(TopTenFat.name)  
fat = list(TopTenFat.total\_fat)  
plt.plot(name, fat, 'ro')  
plt.axis([1,10,20,140])  
plt.xticks(rotation=90)  
plt.legend(["Fat"])  
plt.show()

\*Plot top 10 protein in food using liner graph

##Plot Top 10 Protein Food in linear graph   
  
name = list(TopTenProt.name)  
fat = list(TopTenProt.protein)  
plt.plot(name, fat, 'ro')  
plt.axis([1,10,20,140])  
plt.xticks(rotation=90)  
plt.legend(["Protein"])  
plt.show()

\*Plot top 10 carbohydrate in food using liner graph

##Plot Top 10 Carbs in Bar graph   
  
name = list(TopTenCarb.name)  
fat = list(TopTenCarb.carbohydrate)  
plt.plot(name, fat, 'ro')  
plt.axis([1,10,20,140])  
plt.xticks(rotation=90)  
plt.legend(["Carbs"])  
plt.show()

\*Plot top foods with carbohydrate using scatter plot

#Scatter Plot for Top foods with carbs  
name = list(TopTenCarb.name)  
fat = list(TopTenCarb.carbohydrate)  
plt.scatter(name, fat)  
plt.xticks(rotation=90)  
plt.legend(["Carbs"])  
plt.show()

\*Plot top foods with protein using scatter plot and the same on fat

#Scatter Plot for Top foods with Protein  
name = list(TopTenProt.name)  
fat = list(TopTenProt.protein)  
plt.scatter(name, fat)  
plt.xticks(rotation=90)  
plt.legend(["Protein"])  
plt.show()

#Scatter Plot for Top foods with Fat  
name = list(TopTenFat.name)  
fat = list(TopTenFat.total\_fat)  
plt.scatter(name, fat)  
plt.xticks(rotation=90)  
plt.legend(["Fat"])  
plt.show()

\*Plot top fat and low protein using scatter plot

##Sctter Plot Top 10 Fat and Low Protein   
name =list(MaxFatMinProt.name)  
YFat =list(MaxFatMinProt.total\_fat)  
ZPro = list(MaxFatMinProt.protein)  
plt.scatter(name, YFat)  
plt.scatter(name, ZPro)  
plt.xticks(rotation=90)  
plt.legend(["Fat", "Protein"])  
plt.show()

\*Plot top 10 low fat and height protein using scatter plot

##Sctter Plot Top 10 Min Fat and Max Protein   
name =list(MainFatMaxProt.name)  
YFat =list(MainFatMaxProt.total\_fat)  
ZPro = list(MainFatMaxProt.protein)  
plt.scatter(name, YFat)  
plt.scatter(name, ZPro)  
plt.xticks(rotation=90)  
plt.legend(["Fat", "Protein"])  
plt.show()

\*Plot top 10 fat and top protein using scatter plot

##Scatter Plot Top 10 Fat and Top Calories   
name =list(MaxCalMaxFat.name)  
YFat =list(MaxCalMaxFat.total\_fat)  
ZCal = list(MaxCalMaxFat.calories)  
plt.scatter(name, YFat)  
plt.scatter(name, ZCal)  
plt.xticks(rotation=90)  
plt.legend(["Fat", "Calories"])  
plt.show()

\*Plot top 10 fat and calories and low protein using scatter plot

##Scatter Plot Top 10 Fat and calories and low Protein   
name =list(MaxCalMaxFat.name)  
YFat =list(MaxCalMaxFat.total\_fat)  
ZCal = list(MaxCalMaxFat.calories)  
ZPro = list(MaxFatMinProt.protein)  
plt.scatter(name, YFat)  
plt.scatter(name, ZCal)  
plt.scatter(name, ZPro)  
plt.xticks(rotation=90)  
plt.legend(["Fat", "Calories","Protein"])  
plt.show()

\*Plot top 10 fat and calories and max (protein and fat and calories) and min protein using scatter plot

##Scatter Plot Top 10 Fat and calories and max Protein Max Fat Max Cal Min Prot  
name =list(MaxCalMaxFat.name)  
YFat =list(MaxCalMaxFat.total\_fat)  
ZCal = list(MaxCalMaxFat.calories)  
ZPro = list(MainFatMaxProt.protein)  
plt.scatter(name, YFat)  
plt.scatter(name, ZCal)  
plt.scatter(name, ZPro)  
plt.xticks(rotation=90)  
plt.legend(["Fat", "Calories","Protein"])  
plt.show()

\*Plot top 10 height fat and height calories and low protein using scatter plot

##Scatter Plot Top 10 maxFat and max calories and min Protein   
name =list(MaxFatMaxCalMinProt.name)  
YFat =list(MaxFatMaxCalMinProt.total\_fat)  
ZCal = list(MaxFatMaxCalMinProt.calories)  
ZPro = list(MaxFatMaxCalMinProt.protein)  
plt.scatter(name, YFat)  
plt.scatter(name, ZCal)  
plt.scatter(name, ZPro)  
plt.xticks(rotation=90)  
plt.legend(["Fat", "Calories","Protein"])  
plt.show()

\*Plot top 10 height fat and height calories and low protein using scatter plot

##Scatter Plot Top 10 maxFat and max calories and min Protein   
name =list(MaxFatMaxCalMinProt.name)  
YFat =list(MaxFatMaxCalMinProt.total\_fat)  
ZCal = list(MaxFatMaxCalMinProt.calories)  
ZPro = list(MaxFatMaxCalMinProt.protein)  
plt.scatter(ZCal,YFat, c='r', marker='+')  
# Label Axes  
plt.xlabel('Calories')  
plt.ylabel('Fat')  
plt.title('Fat-Calories Occurrences')  
plt.legend(["Fat", "Calories"])  
plt.show()  
plt.clf()

\*seaborn distplot for max calories and max fat

##seaborn distplot for max cal and max fat  
sns.distplot([YFat,ZCal])  
plt.show()