jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [2]:
```
#import necessary Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
#importing the dataset
data = pd.read_csv(r"C:\Users\Dell\Downloads\archive.zip")
```

In [4]:
```
#analyse the data
data.head()
```

Out[4]:

	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
0	100% Bran	N	C	70	4	1	130	10.0	5.0	6	280	25	3	1.0	0.33	68.402973
1	100% Natural Bran	Q	C	120	3	5	15	2.0	8.0	8	135	0	3	1.0	1.00	33.983679
2	All-Bran	K	C	70	4	1	260	9.0	7.0	5	320	25	3	1.0	0.33	59.425505
3	All-Bran with Extra Fiber	K	C	50	4	0	140	14.0	8.0	0	330	25	3	1.0	0.50	93.704912
4	Almond Delight	R	C	110	2	2	200	1.0	14.0	8	-1	25	3	1.0	0.75	34.384843

In [5]:
```
data.describe()
```

Out[5]:

	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating

jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [8]:
```
plt.figure(figsize = (14, 8))
sns.heatmap(data.corr(), annot=True)
```

Out[8]: <AxesSubplot:>

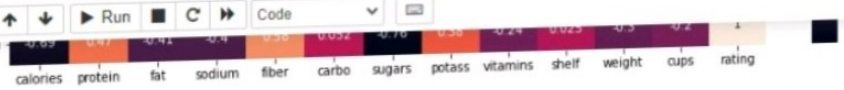
jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code



In [9]:
```
#data visualisation
data.corr()
```

Out[9]:

	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
calories	1.000000	0.019066	0.498610	0.300649	-0.293413	0.250681	0.562340	-0.066609	0.265356	0.097234	0.696091	0.087200	-0.689376
protein	0.019066	1.000000	0.208431	-0.054674	0.500330	-0.130864	-0.329142	0.549407	0.007335	0.133865	0.216158	-0.244469	0.470618
fat	0.498610	0.208431	1.000000	-0.005407	0.016719	-0.318043	0.270819	0.193279	-0.031156	0.263691	0.214625	-0.175892	-0.409284
sodium	0.300649	-0.054674	-0.005407	1.000000	-0.070675	0.355983	0.101451	-0.032603	0.361477	-0.069719	0.308576	0.119665	-0.401295
fiber	-0.293413	0.500330	0.016719	-0.070675	1.000000	-0.356083	-0.141205	0.903374	-0.032243	0.297539	0.247226	-0.513061	0.584160
carbo	0.250681	-0.130864	-0.318043	0.355983	-0.356083	1.000000	-0.331665	-0.349685	0.258148	-0.101790	0.135136	0.363932	0.052055
sugars	0.562340	-0.329142	0.270819	0.101451	-0.141205	-0.331665	1.000000	0.021696	0.125137	0.100438	0.450648	-0.032358	-0.759675
potass	-0.066609	0.549407	0.193279	-0.032603	0.903374	-0.349685	0.021696	1.000000	0.020699	0.360663	0.416303	-0.495195	0.380165
vitamins	0.265356	0.007335	-0.031156	0.361477	-0.032243	0.258148	0.125137	0.020699	1.000000	0.299262	0.320324	0.128405	-0.240544
shelf	0.097234	0.133865	0.263691	-0.069719	0.297539	-0.101790	0.100438	0.360663	0.299262	1.000000	0.190762	-0.335269	0.025159
weight	0.696091	0.216158	0.214625	0.308576	0.247226	0.135136	0.450648	0.416303	0.320324	0.190762	1.000000	-0.199583	-0.298124
cups	0.087200	-0.244469	-0.175892	0.119665	-0.513061	0.363932	-0.032358	-0.495195	0.128405	-0.335269	-0.199583	1.000000	-0.203160
rating	-0.689376	0.470618	-0.409284	-0.401295	0.584160	0.052055	-0.759675	0.380165	-0.240544	0.025159	-0.298124	-0.203160	1.000000

jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

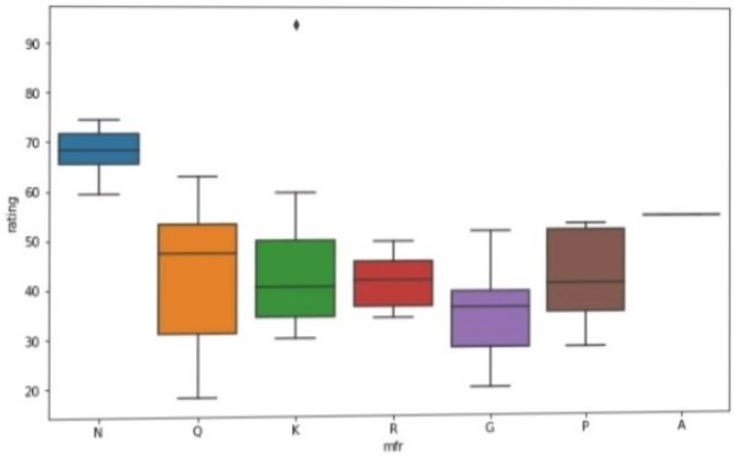
File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [10]:
```
plt.figure(figsize = (10, 6))
sns.boxplot(data = data, x = "mfr", y = "rating")
```

Out[10]: <AxesSubplot:xlabel='mfr', ylabel='rating'>



In [11]:
```
sns.pairplot(data=data, markers=["^","v"], palette="inferno")
```

jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

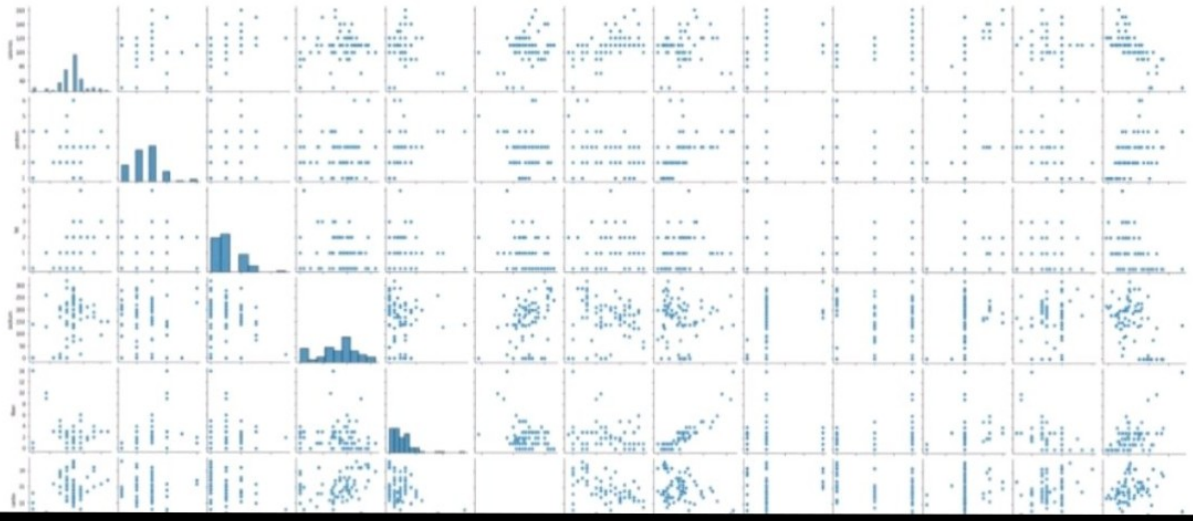
File Edit View Insert Cell Kernel Widgets Help

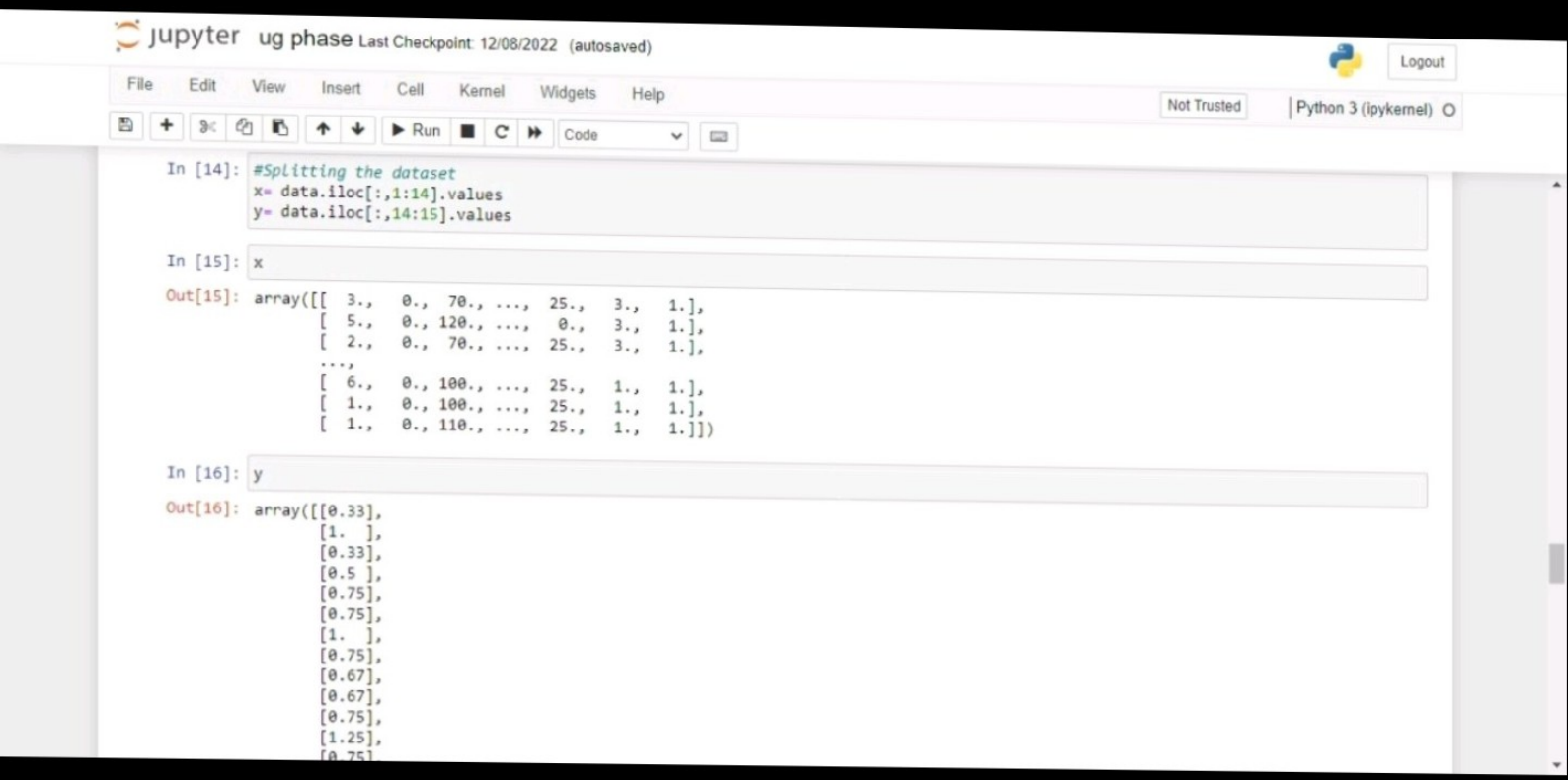
Not Trusted | Python 3 (ipykernel)

Run Code

```
In [11]: sns.pairplot(data=data, markers=["^","v"], palette="inferno")
```

```
Out[11]: <seaborn.axisgrid.PairGrid at 0x1eb6462ca60>
```





jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [14]:
```
#Splitting the dataset
x= data.iloc[:,1:14].values
y= data.iloc[:,14:15].values
```

In [15]:
```
x
```

Out[15]:
```
array([[  3.,   0.,  70., ...,  25.,   3.,   1.],
       [  5.,   0., 120., ...,   0.,   3.,   1.],
       [  2.,   0.,  70., ...,  25.,   3.,   1.],
       ...,
       [  6.,   0., 100., ...,  25.,   1.,   1.],
       [  1.,   0., 100., ...,  25.,   1.,   1.],
       [  1.,   0., 110., ...,  25.,   1.,   1.]])
```

In [16]:
```
y
```

Out[16]:
```
array([[0.33],
       [1.  ],
       [0.33],
       [0.5 ],
       [0.75],
       [0.75],
       [1.  ],
       [0.75],
       [0.67],
       [0.67],
       [0.75],
       [1.25],
       [0.75]
```

jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [12]:
```
#Lable Encoding
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data["mfr"] = le.fit_transform(data["mfr"])
data["type"] = le.fit_transform(data["type"])
```

In [13]:
```
data
```

Out[13]:

	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
0	100% Bran	3	0	70	4	1	130	10.0	5.0	6	280	25	3	1.0	0.33	68.402973
1	100% Natural Bran	5	0	120	3	5	15	2.0	8.0	8	135	0	3	1.0	1.00	33.983679
2	All-Bran	2	0	70	4	1	260	9.0	7.0	5	320	25	3	1.0	0.33	59.425505
3	All-Bran with Extra Fiber	2	0	50	4	0	140	14.0	8.0	0	330	25	3	1.0	0.50	93.704912
4	Almond Delight	6	0	110	2	2	200	1.0	14.0	8	-1	25	3	1.0	0.75	34.384843
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
72	Triples	1	0	110	2	1	250	0.0	21.0	3	60	25	3	1.0	0.75	39.106174
73	Trix	1	0	110	1	1	140	0.0	13.0	12	25	25	2	1.0	1.00	27.753301
74	Wheat Chex	6	0	100	3	1	230	3.0	17.0	3	115	25	1	1.0	0.67	49.787445
75	Wheaties	1	0	100	3	1	200	3.0	17.0	3	110	25	1	1.0	1.00	51.592193
76	Wheaties Honey Gold	1	0	110	2	1	200	1.0	16.0	8	60	25	1	1.0	0.75	36.187559

77 rows × 16 columns

jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [17]:
```
#one hot encoding
from sklearn.preprocessing import OneHotEncoder
one = OneHotEncoder()
a = one.fit_transform(x[:,0:1]).toarray()
x = np.delete(x,[0],axis=1)
x=np.concatenate((a,x),axis=1)
```

In [18]:
```
#splitting the data into train and test
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2, random_state = 0)
```

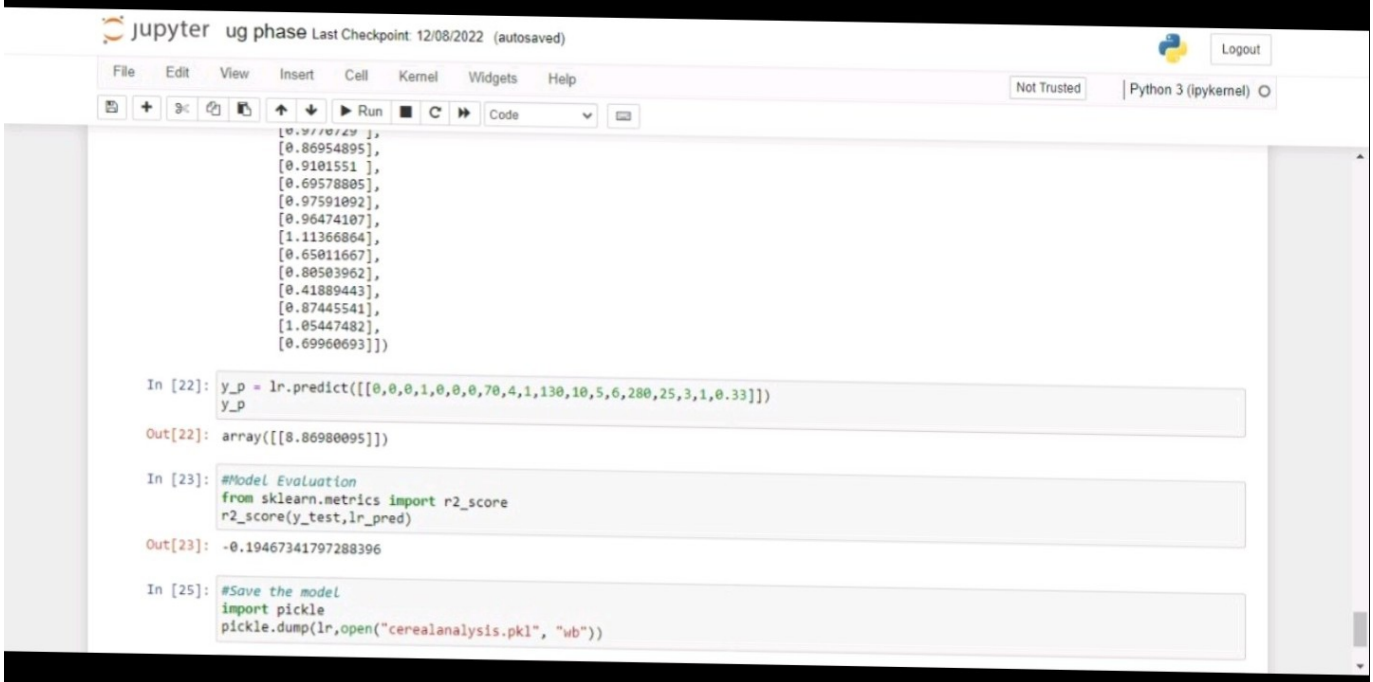
In [19]:
```
#Training and Testing the model
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
```

Out[19]:
```
LinearRegression()
```

In [20]:
```
lr_pred = lr.predict(x_test)
```

In [21]:
```
lr_pred
```

Out[21]:
```
array([[0.92942348],
       [0.84707312],
       [0.7007123 ],
       [0.9770729 ],
       [0.86954895],
       [0.9101551 ],
```



jupyter ug phase Last Checkpoint: 12/08/2022 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [5]:
```
data.describe()
```

Out[5]:

	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
count	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000
mean	106.883117	2.545455	1.012987	159.675325	2.151948	14.597403	6.922078	96.077922	28.246753	2.207792	1.029610	0.821039	42.665705
std	19.484119	1.094790	1.006473	83.832295	2.383364	4.278956	4.444885	71.286813	22.342523	0.832524	0.150477	0.232716	14.047289
min	50.000000	1.000000	0.000000	0.000000	0.000000	-1.000000	-1.000000	-1.000000	0.000000	1.000000	0.500000	0.250000	18.042851
25%	100.000000	2.000000	0.000000	130.000000	1.000000	12.000000	3.000000	40.000000	25.000000	1.000000	1.000000	0.670000	33.174094
50%	110.000000	3.000000	1.000000	180.000000	2.000000	14.000000	7.000000	90.000000	25.000000	2.000000	1.000000	0.750000	40.400208
75%	110.000000	3.000000	2.000000	210.000000	3.000000	17.000000	11.000000	120.000000	25.000000	3.000000	1.000000	1.000000	50.828392
max	160.000000	6.000000	5.000000	320.000000	14.000000	23.000000	15.000000	330.000000	100.000000	3.000000	1.500000	1.500000	93.704912

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 77 entries, 0 to 76
Data columns (total 16 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   name      77 non-null     object
 1   mfr       77 non-null     object
 2   type      77 non-null     object
 3   calories  77 non-null     int64
 4   protein   77 non-null     int64
 5   fat       77 non-null     int64
```