In [1]:
```
# Importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
# importing data
df=pd.read_csv('FAO.csv',encoding='latin-1')
df
```

Out[2]:

	Area Abbreviation	Area Code	Area	Item Code	Item	Element Code	Element	Unit	latitude
0	AFG	2	Afghanistan	2511	Wheat and products	5142	Food	1000 tonnes	33.94
1	AFG	2	Afghanistan	2805	Rice (Milled Equivalent)	5142	Food	1000 tonnes	33.94
2	AFG	2	Afghanistan	2513	Barley and products	5521	Feed	1000 tonnes	33.94
3	AFG	2	Afghanistan	2513	Barley and products	5142	Food	1000 tonnes	33.94
4	AFG	2	Afghanistan	2514	Maize and products	5521	Feed	1000 tonnes	33.94
...	...	...	...	...	...	...	...	...	...
21472	ZWE	181	Zimbabwe	2948	Milk - Excluding Butter	5142	Food	1000 tonnes	-19.02
21473	ZWE	181	Zimbabwe	2960	Fish, Seafood	5521	Feed	1000 tonnes	-19.02
21474	ZWE	181	Zimbabwe	2960	Fish, Seafood	5142	Food	1000 tonnes	-19.02

	Area Abbreviation	Area Code	Area	Item Code	Item	Element Code	Element	Unit	latitude
21475	ZWE	181	Zimbabwe	2961	Aquatic Products, Other	5142	Food	1000 tonnes	-19.02
21476	ZWE	181	Zimbabwe	2928	Miscellaneous	5142	Food	1000 tonnes	-19.02

21477 rows × 63 columns

In [3]:
```
df.head()
```

Out[3]:

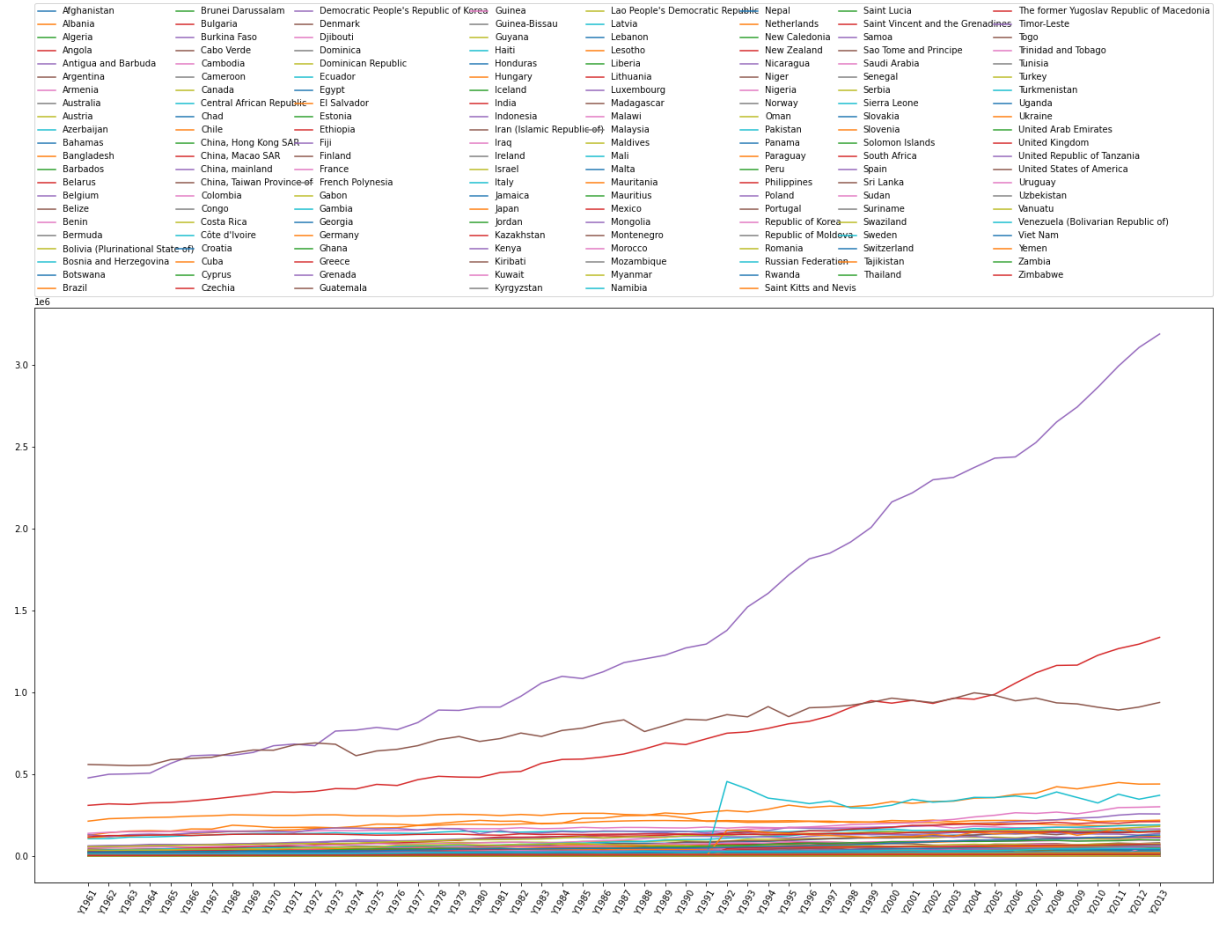
	Area Abbreviation	Area Code	Area	Item Code	Item	Element Code	Element	Unit	latitude	longitud
0	AFG	2	Afghanistan	2511	Wheat and products	5142	Food	1000 tonnes	33.94	67.7
1	AFG	2	Afghanistan	2805	Rice (Milled Equivalent)	5142	Food	1000 tonnes	33.94	67.7
2	AFG	2	Afghanistan	2513	Barley and products	5521	Feed	1000 tonnes	33.94	67.7
3	AFG	2	Afghanistan	2513	Barley and products	5142	Food	1000 tonnes	33.94	67.7
4	AFG	2	Afghanistan	2514	Maize and products	5521	Feed	1000 tonnes	33.94	67.7

5 rows × 63 columns

EDA

In [4]:
```
#Plot for annual produce of different countries with quantity versus ye
ars
```

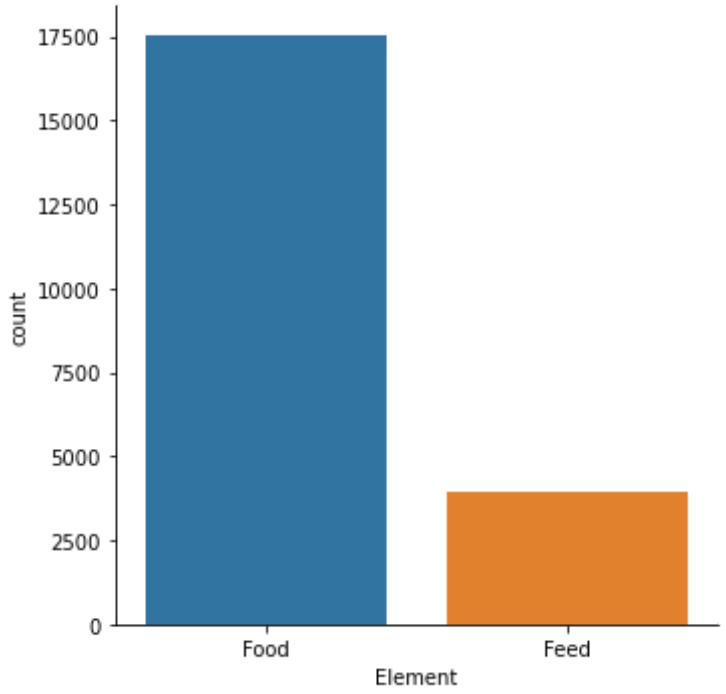
In [5]:
```
area_list = list(df['Area'].unique())
year_list = list(df.iloc[:,10:].columns)

plt.figure(figsize=(24,12))
for ar in area_list:
    yearly_produce = []
    for yr in year_list:
        yearly_produce.append(df[yr][df['Area'] == ar].sum())
    plt.plot(yearly_produce, label=ar)
plt.xticks(np.arange(53), tuple(year_list), rotation=60)
plt.legend(bbox_to_anchor=(0., 1.02, 1., .102), loc=3, ncol=8, mode="ex
pand", borderaxespad=0.)
plt.savefig('p.png')
plt.show()
```



In [6]:
```
#Food and feed plot
```

In [7]:
```
sns.factorplot("Element", data=df, kind="count")
plt.show()
```

```
C:\Users\parveen\anaconda3\lib\site-packages\seaborn\categorical.py:3666: UserWarning: The `factorplot` function has been renamed to `catplot`. The original name will be removed in a future release. Please update your code. Note that the default `kind` in `factorplot` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```

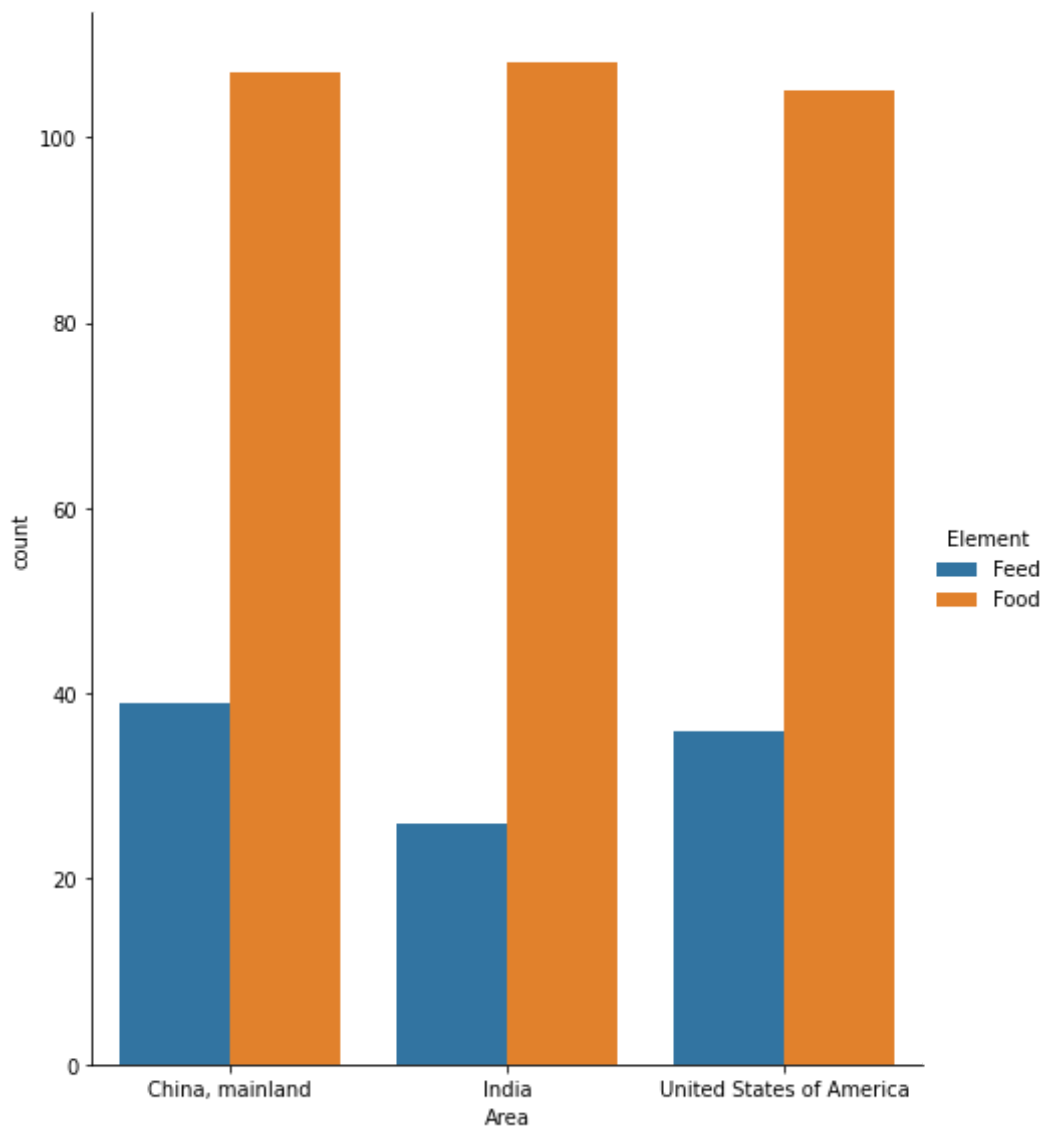


In [8]:
```
#Food and feed plot for the largest producers(India, USA, China)
```

In [9]:
```
sns.factorplot("Area", data=df[(df['Area'] == "India") | (df['Area'] ==
 "China, mainland") | (df['Area'] == "United States of America")], kind
="count", hue="Element", size=8, aspect=.8)
```

```
C:\Users\parveen\anaconda3\lib\site-packages\seaborn\categorical.py:367
2: UserWarning: The `size` parameter has been renamed to `height`; plea
se update your code.
  warnings.warn(msg, UserWarning)
```

Out[9]:
```
<seaborn.axisgrid.FacetGrid at 0x685fa308b0>
```



In [10]:
```
new_df_dict = {}
for ar in area_list:
    yearly_produce = []
    for yr in year_list:
```

```
        yearly_produce.append(df[yr][df['Area']==ar].sum())
    new_df_dict[ar] = yearly_produce
new_df = pd.DataFrame(new_df_dict)

new_df.head()
```

Out[10]:

	Afghanistan	Albania	Algeria	Angola	Antigua and Barbuda	Argentina	Armenia	Australia	Austria	Azerb
0	9481.0	1706.0	7488.0	4834.0	92.0	43402.0	0.0	25795.0	22542.0	
1	9414.0	1749.0	7235.0	4775.0	94.0	40784.0	0.0	27618.0	22627.0	
2	9194.0	1767.0	6861.0	5240.0	105.0	40219.0	0.0	28902.0	23637.0	
3	10170.0	1889.0	7255.0	5286.0	95.0	41638.0	0.0	29107.0	24099.0	
4	10473.0	1884.0	7509.0	5527.0	84.0	44936.0	0.0	28961.0	22664.0	

5 rows × 174 columns

In [11]:
```
new_df = pd.DataFrame.transpose(new_df)
new_df.columns = year_list

new_df.head()
```

Out[11]:

	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968	Y1969	Y1970
Afghanistan	9481.0	9414.0	9194.0	10170.0	10473.0	10169.0	11289.0	11508.0	11815.0	10454.0
Albania	1706.0	1749.0	1767.0	1889.0	1884.0	1995.0	2046.0	2169.0	2230.0	2395.0
Algeria	7488.0	7235.0	6861.0	7255.0	7509.0	7536.0	7986.0	8839.0	9003.0	9355.0
Angola	4834.0	4775.0	5240.0	5286.0	5527.0	5677.0	5833.0	5685.0	6219.0	6460.0
Antigua and Barbuda	92.0	94.0	105.0	95.0	84.0	73.0	64.0	59.0	68.0	77.0

5 rows × 53 columns

In [12]:
```
mean_produce = []
for i in range(174):
    mean_produce.append(new_df.iloc[i,:].values.mean())
new_df['Mean_Produce'] = mean_produce

new_df['Rank'] = new_df['Mean_Produce'].rank(ascending=False)

new_df.head()
```

Out[12]:

	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968	Y1969	Y1970
Afghanistan	9481.0	9414.0	9194.0	10170.0	10473.0	10169.0	11289.0	11508.0	11815.0	10454.0
Albania	1706.0	1749.0	1767.0	1889.0	1884.0	1995.0	2046.0	2169.0	2230.0	2395.0
Algeria	7488.0	7235.0	6861.0	7255.0	7509.0	7536.0	7986.0	8839.0	9003.0	9355.0
Angola	4834.0	4775.0	5240.0	5286.0	5527.0	5677.0	5833.0	5685.0	6219.0	6460.0
Antigua and Barbuda	92.0	94.0	105.0	95.0	84.0	73.0	64.0	59.0	68.0	77.0

5 rows × 55 columns

In [13]:
```
item_list = list(df['Item'].unique())

item_df = pd.DataFrame()
item_df['Item_Name'] = item_list

for yr in year_list:
    item_produce = []
    for it in item_list:
        item_produce.append(df[yr][df['Item']==it].sum())
    item_df[yr] = item_produce
```

In [14]:
```
item_df.head()
```

Out[14]:

	Item_Name	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968
0	Wheat and products	138829.0	144643.0	147325.0	156273.0	168822.0	169832.0	171469.0	179530.0
1	Rice (Milled Equivalent)	122700.0	131842.0	139507.0	148304.0	150056.0	155583.0	158587.0	164614.0
2	Barley and products	46180.0	48915.0	51642.0	54184.0	54945.0	55463.0	56424.0	60455.0
3	Maize and products	168039.0	168305.0	172905.0	175468.0	190304.0	200860.0	213050.0	215613.0
4	Millet and products	19075.0	19019.0	19740.0	20353.0	18377.0	20860.0	22997.0	21785.0

5 rows × 54 columns

In [15]:
```
sum_col = []
for i in range(115):
    sum_col.append(item_df.iloc[i,1:].values.sum())
item_df['Sum'] = sum_col
item_df['Production_Rank'] = item_df['Sum'].rank(ascending=False)

item_df.head()
```

Out[15]:

	Item_Name	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968
0	Wheat and products	138829.0	144643.0	147325.0	156273.0	168822.0	169832.0	171469.0	179530.0
1	Rice (Milled Equivalent)	122700.0	131842.0	139507.0	148304.0	150056.0	155583.0	158587.0	164614.0
2	Barley and products	46180.0	48915.0	51642.0	54184.0	54945.0	55463.0	56424.0	60455.0
3	Maize and products	168039.0	168305.0	172905.0	175468.0	190304.0	200860.0	213050.0	215613.0
4	Millet and products	19075.0	19019.0	19740.0	20353.0	18377.0	20860.0	22997.0	21785.0

5 rows × 56 columns

In [16]:
```
item_df
```

Out[16]:

	Item_Name	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968
0	Wheat and products	138829.0	144643.0	147325.0	156273.0	168822.0	169832.0	171469.0	179530.0
1	Rice (Milled Equivalent)	122700.0	131842.0	139507.0	148304.0	150056.0	155583.0	158587.0	164614.0
2	Barley and products	46180.0	48915.0	51642.0	54184.0	54945.0	55463.0	56424.0	60455.0
3	Maize and products	168039.0	168305.0	172905.0	175468.0	190304.0	200860.0	213050.0	215613.0
4	Millet and products	19075.0	19019.0	19740.0	20353.0	18377.0	20860.0	22997.0	21785.0
...	...	...	...	...	...	...	...	...	...
110	Sunflower seed	54.0	70.0	67.0	41.0	47.0	60.0	84.0	77.0
111	Cottonseed	1324.0	1816.0	2186.0	2238.0	2388.0	2454.0	2943.0	2645.0
112	Sugar non-centrifugal	9892.0	9274.0	9768.0	10348.0	11066.0	10966.0	10094.0	9927.0
113	Ricebran Oil	70.0	73.0	79.0	97.0	96.0	108.0	110.0	125.0
114	Meat, Aquatic Mammals	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

115 rows × 56 columns

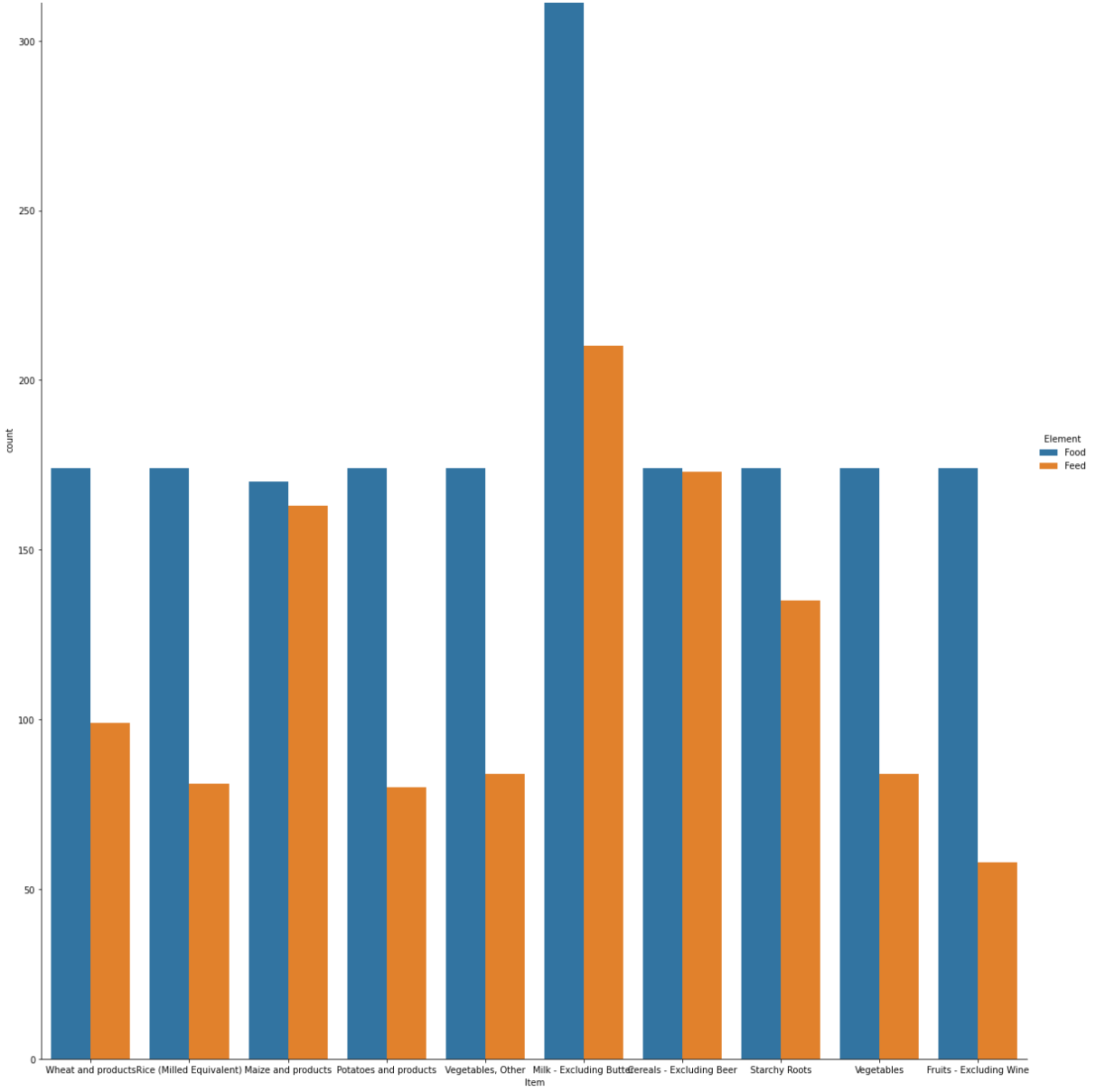
In [17]:
```
#Most produced food item
```

In [18]:
```
item_df['Item_Name'][item_df['Production_Rank'] < 11.0].sort_values()
```

Out[18]:
```
56    Cereals - Excluding Beer
65     Fruits - Excluding Wine
3           Maize and products
53     Milk - Excluding Butter
6        Potatoes and products
1     Rice (Milled Equivalent)
57               Starchy Roots
64                  Vegetables
27           Vegetables, Other
0           Wheat and products
Name: Item_Name, dtype: object
```

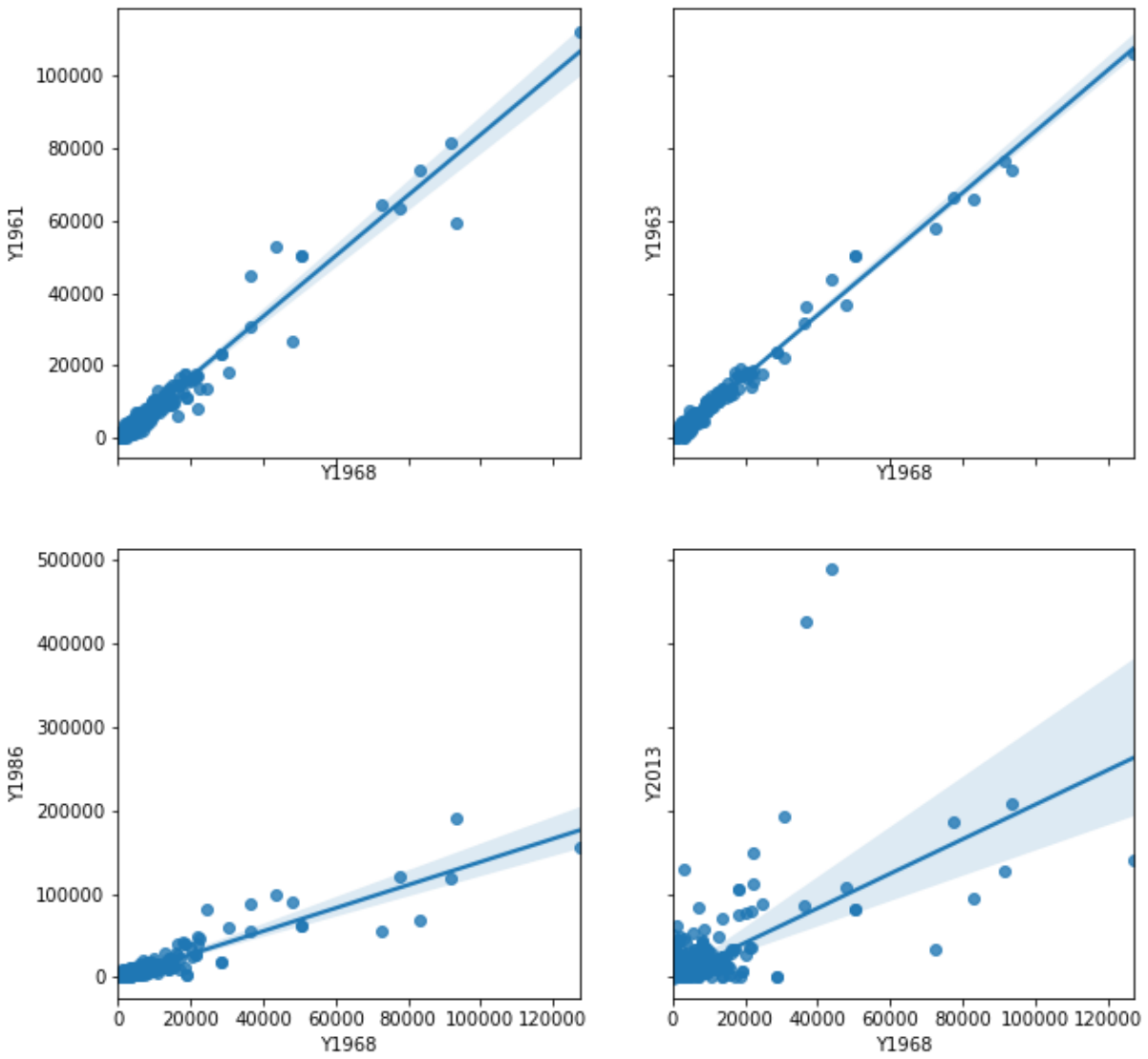
In [19]:
```
sns.factorplot("Item", data=df[(df['Item']=='Wheat and products') | (df['Item']=='Rice (Milled Equivalent)') | (df['Item']=='Maize and products') | (df['Item']=='Potatoes and products') | (df['Item']=='Vegetables, Other') | (df['Item']=='Milk - Excluding Butter') | (df['Item']=='Cereals - Excluding Beer') | (df['Item']=='Starchy Roots') | (df['Item']=='Vegetables') | (df['Item']=='Fruits - Excluding Wine')], kind="count", hue="Element", size=20, aspect=.8)
plt.show()
```

```
C:\Users\parveen\anaconda3\lib\site-packages\seaborn\categorical.py:3666: UserWarning: The `factorplot` function has been renamed to `catplot`. The original name will be removed in a future release. Please update your code. Note that the default `kind` in `factorplot` (`'point'`) has changed `'strip'` in `catplot`.
  warnings.warn(msg)
C:\Users\parveen\anaconda3\lib\site-packages\seaborn\categorical.py:3672: UserWarning: The `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```



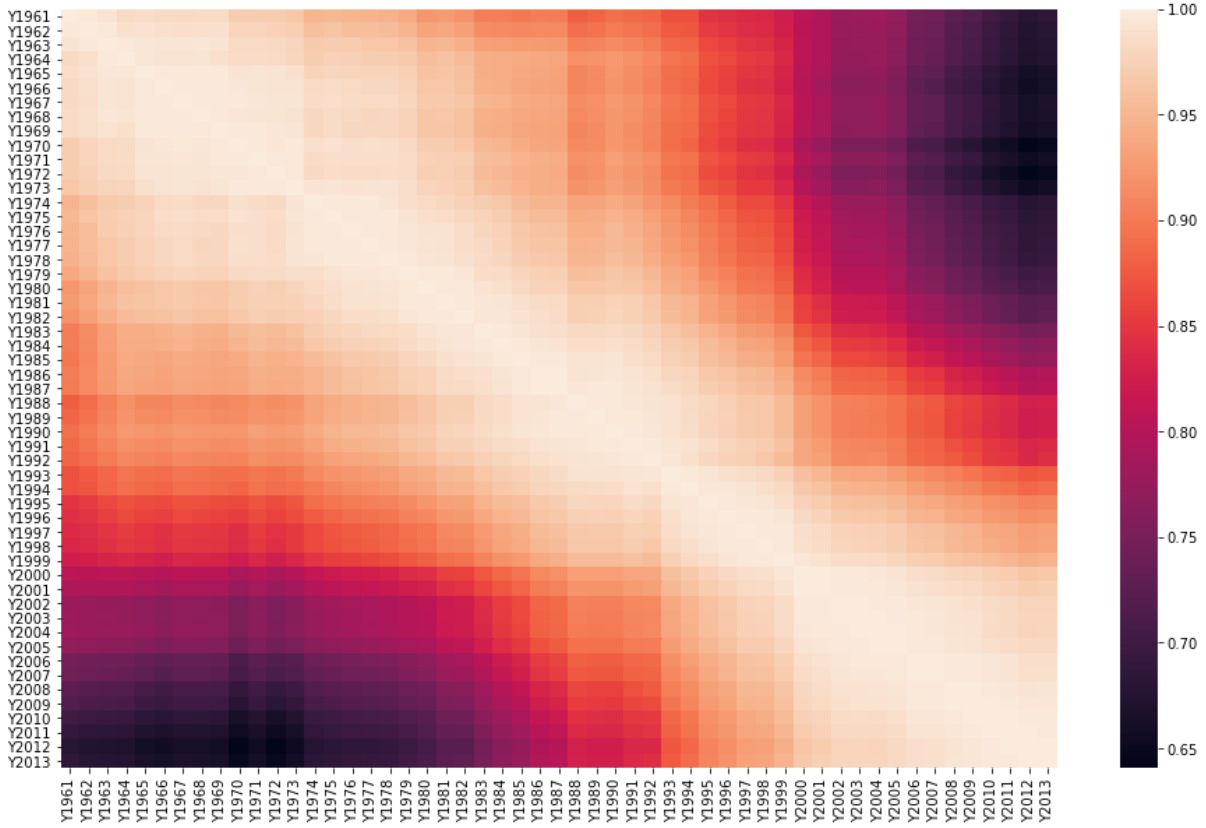
In [20]:
```
f, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2, sharex='col', sharey=
'row', figsize=(10,10))
ax1.set(xlabel='Y1968', ylabel='Y1961')
```

```
ax2.set(xlabel='Y1968', ylabel='Y1963')
ax3.set(xlabel='Y1968', ylabel='Y1986')
ax4.set(xlabel='Y1968', ylabel='Y2013')
sns.jointplot(x="Y1968", y="Y1961", data=df, kind="reg", ax=ax1)
sns.jointplot(x="Y1968", y="Y1963", data=df, kind="reg", ax=ax2)
sns.jointplot(x="Y1968", y="Y1986", data=df, kind="reg", ax=ax3)
sns.jointplot(x="Y1968", y="Y2013", data=df, kind="reg", ax=ax4)
plt.close(2)
plt.close(3)
plt.close(4)
plt.close(5)
```

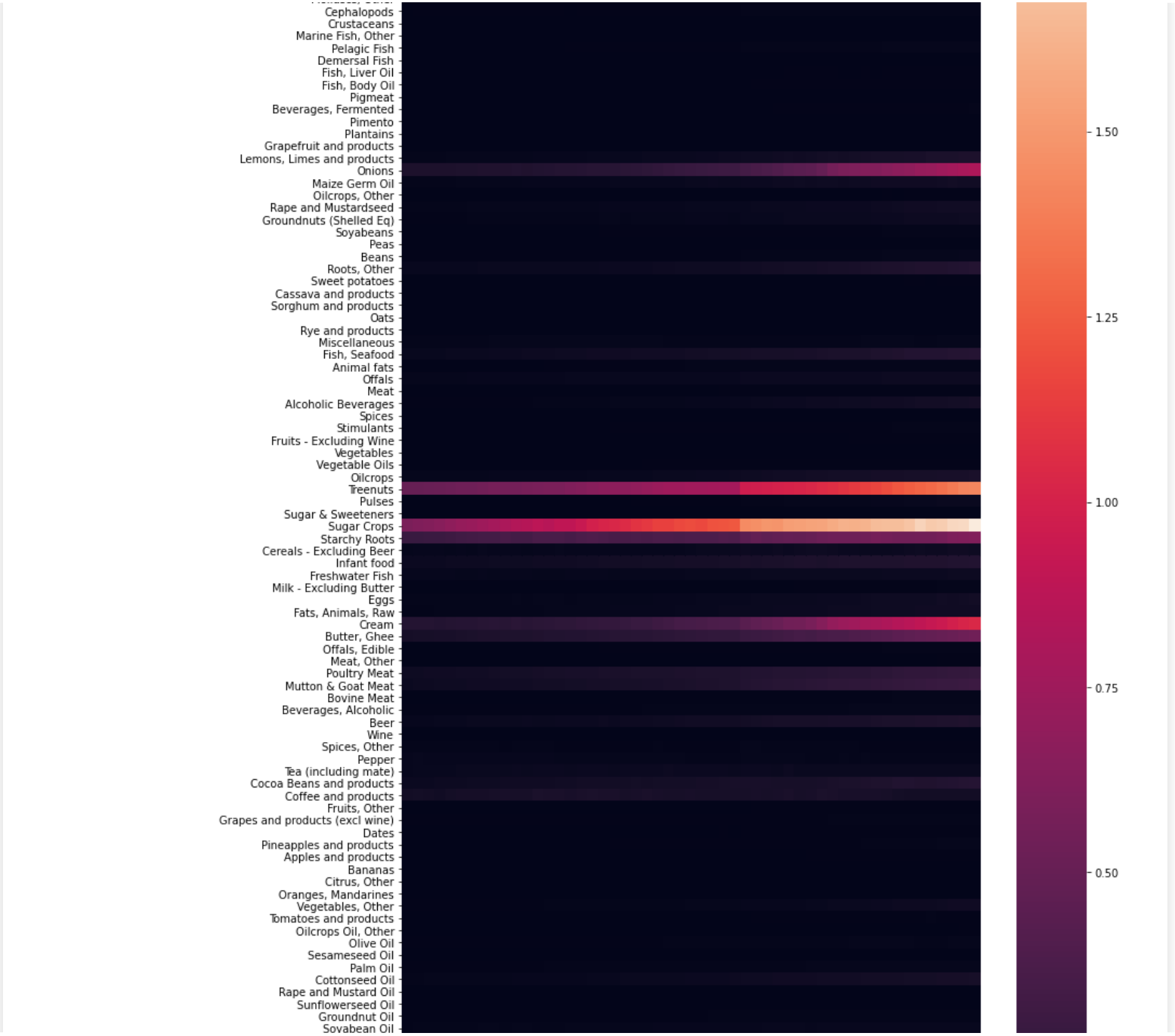


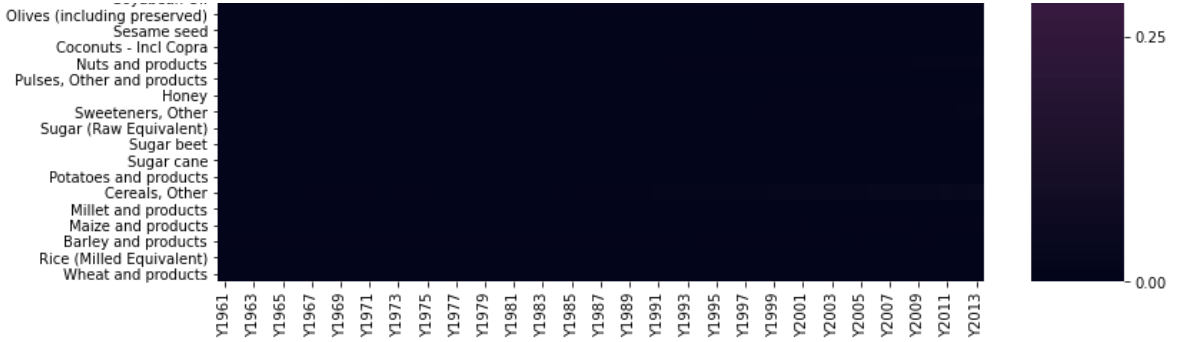
In [21]:
```
year_df = df.iloc[:,10:]
fig, ax = plt.subplots(figsize=(16,10))
sns.heatmap(year_df.corr(), ax=ax)
```

Out[21]: <matplotlib.axes._subplots.AxesSubplot at 0x685928ad30>



In [22]:
```
new_item_df = item_df.drop(["Item_Name","Sum","Production_Rank"], axis
= 1)
fig, ax = plt.subplots(figsize=(12,24))
sns.heatmap(new_item_df,ax=ax)
ax.set_yticklabels(item_df.Item_Name.values[::-1])
plt.show()
```





Clustering

In [23]:
```
new_df.head()
```

Out[23]:

	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968	Y1969	Y1970
Afghanistan	9481.0	9414.0	9194.0	10170.0	10473.0	10169.0	11289.0	11508.0	11815.0	10454.0
Albania	1706.0	1749.0	1767.0	1889.0	1884.0	1995.0	2046.0	2169.0	2230.0	2395.0
Algeria	7488.0	7235.0	6861.0	7255.0	7509.0	7536.0	7986.0	8839.0	9003.0	9355.0
Angola	4834.0	4775.0	5240.0	5286.0	5527.0	5677.0	5833.0	5685.0	6219.0	6460.0
Antigua and Barbuda	92.0	94.0	105.0	95.0	84.0	73.0	64.0	59.0	68.0	77.0

5 rows × 55 columns

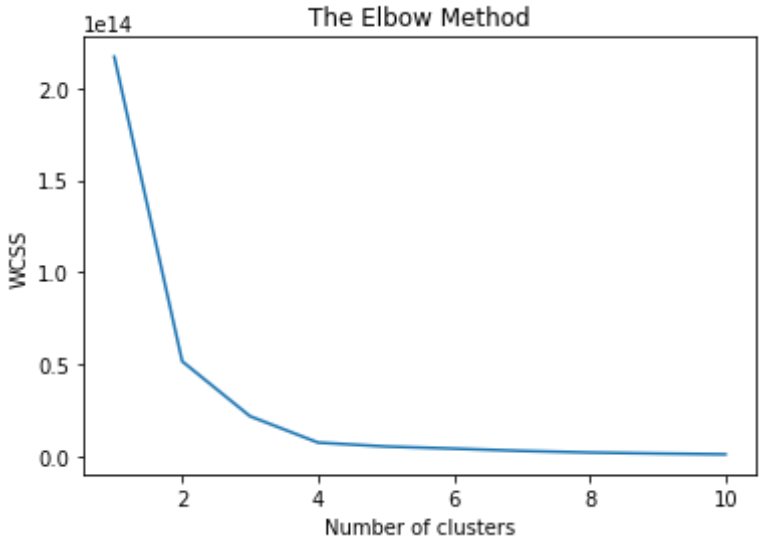
In [24]:
```
X=new_df.iloc[:,:-2].values
X=pd.DataFrame(X)
X=X.apply(pd.to_numeric,axis=0)
X.columns=year_list
X
```

Out[24]:

	Y1961	Y1962	Y1963	Y1964	Y1965	Y1966	Y1967	Y1968	Y1969	Y1970	...
0	9481.0	9414.0	9194.0	10170.0	10473.0	10169.0	11289.0	11508.0	11815.0	10454.0	...
1	1706.0	1749.0	1767.0	1889.0	1884.0	1995.0	2046.0	2169.0	2230.0	2395.0	...
2	7488.0	7235.0	6861.0	7255.0	7509.0	7536.0	7986.0	8839.0	9003.0	9355.0	...
3	4834.0	4775.0	5240.0	5286.0	5527.0	5677.0	5833.0	5685.0	6219.0	6460.0	...
4	92.0	94.0	105.0	95.0	84.0	73.0	64.0	59.0	68.0	77.0	...
...	...	...	...	...	...	...	...	...	...	...	...
169	9523.0	9369.0	9788.0	10539.0	10641.0	10772.0	11126.0	12014.0	12537.0	13503.0	...
170	23856.0	25220.0	26053.0	26377.0	26961.0	27355.0	27745.0	28698.0	29565.0	30841.0	...
171	2982.0	3038.0	3147.0	3224.0	3328.0	3358.0	3420.0	3411.0	3386.0	3348.0	...
172	2976.0	3057.0	3069.0	3121.0	3236.0	3523.0	3688.0	3791.0	3904.0	4006.0	...
173	3260.0	3503.0	3479.0	3738.0	3940.0	3991.0	4202.0	4443.0	4486.0	4714.0	...

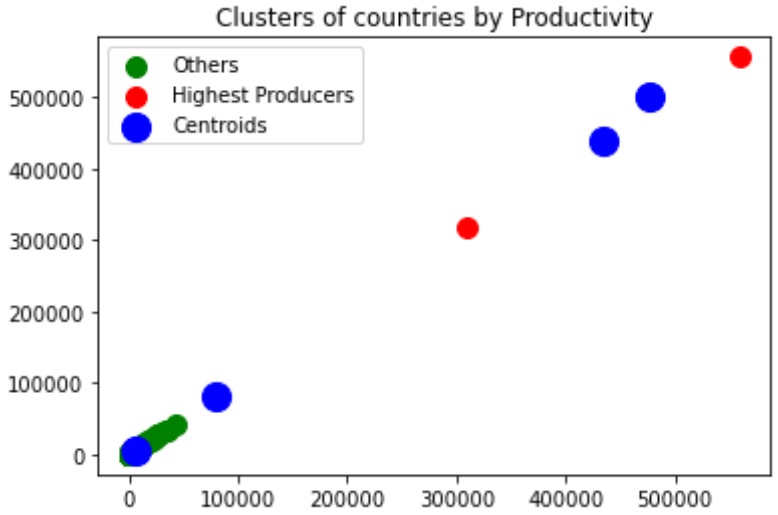
174 rows × 53 columns

In [25]:
```
from sklearn.cluster import KMeans
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i,init='k-means++',max_iter=300,n_init=1
0,random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1,11),wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



In [26]:
```
kmeans = KMeans(n_clusters=4,init='k-means++',max_iter=300,n_init=10,ra
ndom_state=0)
y_kmeans = kmeans.fit_predict(X)
X=X.values
```

In [27]:
```
plt.scatter(X[y_kmeans == 0, 0], X[y_kmeans == 0,1],s=100,c='green',lab
el='Others')
plt.scatter(X[y_kmeans == 1, 0], X[y_kmeans == 1,1],s=100,c='red',label
='Highest Producers')
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],s
=200,c='blue',label='Centroids')
plt.title('Clusters of countries by Productivity')
plt.legend()
plt.show()
```



In [28]:
```
import joblib
```

In [29]:
```
joblib.dump(kmeans,'Fao-kmeans.save')
```

Out[29]:
```
['Fao-kmeans.save']
```

In [ ]:
```

```