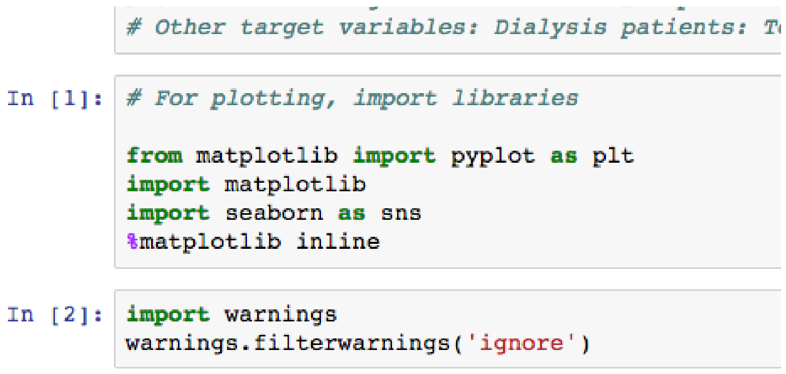
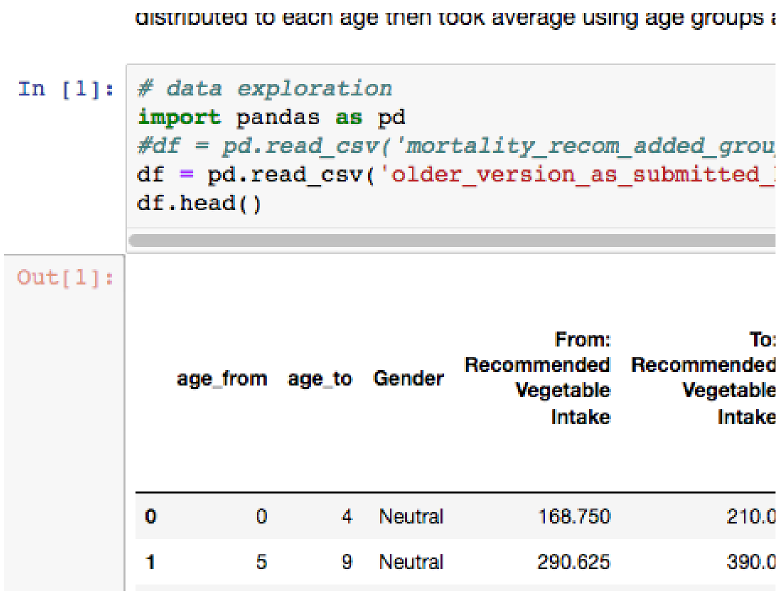
Just to give you a sense of the code (One note, in research people usually do not write production ready code, at least for the first iteration)

Exploratory Analysis:



```
# Other target variables: Dialysis patients: T

In [1]: # For plotting, import libraries

        from matplotlib import pyplot as plt
        import matplotlib
        import seaborn as sns
        %matplotlib inline

In [2]: import warnings
        warnings.filterwarnings('ignore')
```



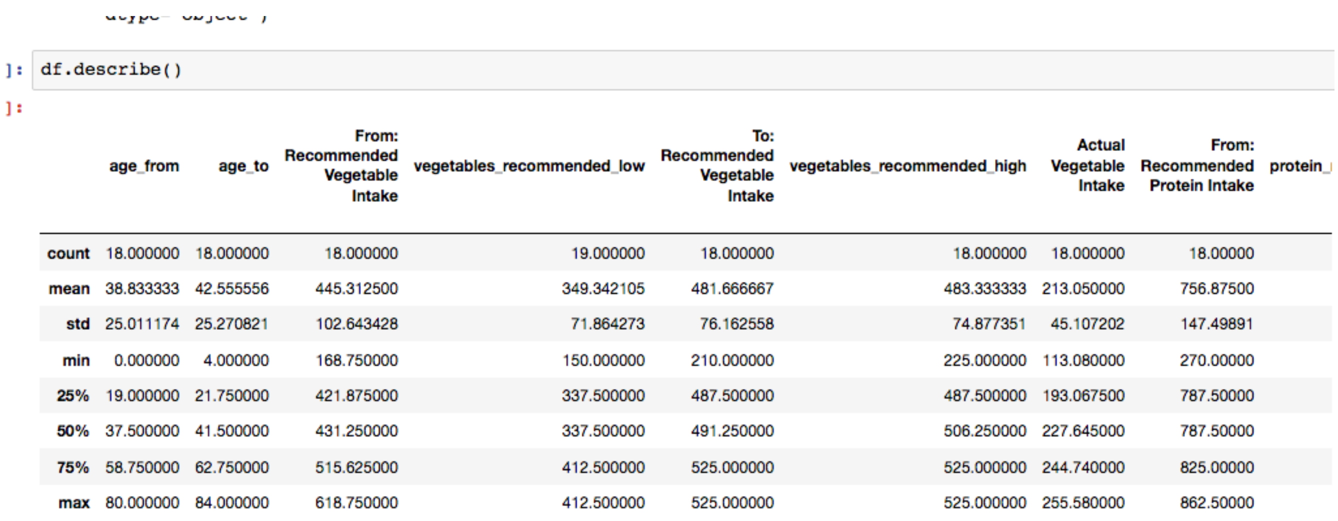
distributed to each age then took average using age groups a

```
In [1]: # data exploration
        import pandas as pd
        #df = pd.read_csv('mortality_recom_added_grou
        df = pd.read_csv('older_version_as_submitted_
        df.head()

Out[1]:
```

	age_from	age_to	Gender	From: Recommended Vegetable Intake	To: Recommended Vegetable Intake
0	0	4	Neutral	168.750	210.0
1	5	9	Neutral	290.625	390.0

Quantitative

just one example.
You could check interval, distribution, Null/Alternative Hypothesis, Chi-Square, p value based tests



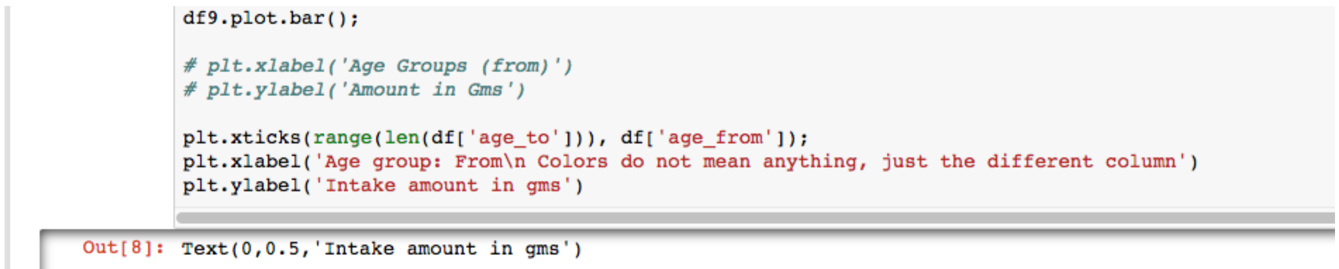
```
dtype='object')
```

```
]: df.describe()
```

```
]:
```

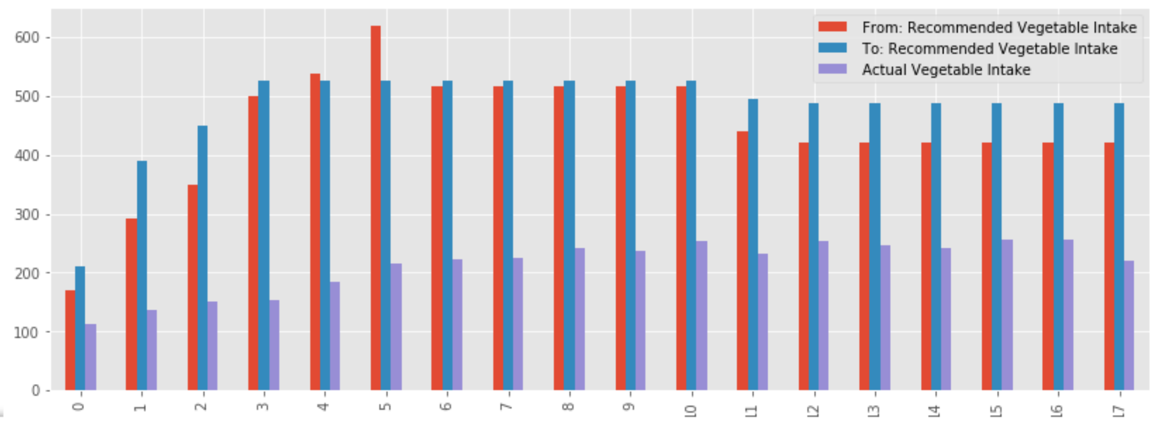
	age_from	age_to	From: Recommended Vegetable Intake	vegetables_recommended_low	To: Recommended Vegetable Intake	vegetables_recommended_high	Actual Vegetable Intake	From: Recommended Protein Intake	protein_
count	18.000000	18.000000	18.000000	19.000000	18.000000	18.000000	18.000000	18.00000	
mean	38.833333	42.555556	445.312500	349.342105	481.666667	483.333333	213.050000	756.87500	
std	25.011174	25.270821	102.643428	71.864273	76.162558	74.877351	45.107202	147.49891	
min	0.000000	4.000000	168.750000	150.000000	210.000000	225.000000	113.080000	270.00000	
25%	19.000000	21.750000	421.875000	337.500000	487.500000	487.500000	193.067500	787.50000	
50%	37.500000	41.500000	431.250000	337.500000	491.250000	506.250000	227.645000	787.50000	
75%	58.750000	62.750000	515.625000	412.500000	525.000000	525.000000	244.740000	825.00000	
max	80.000000	84.000000	618.750000	412.500000	525.000000	525.000000	255.580000	862.50000	

Univariate



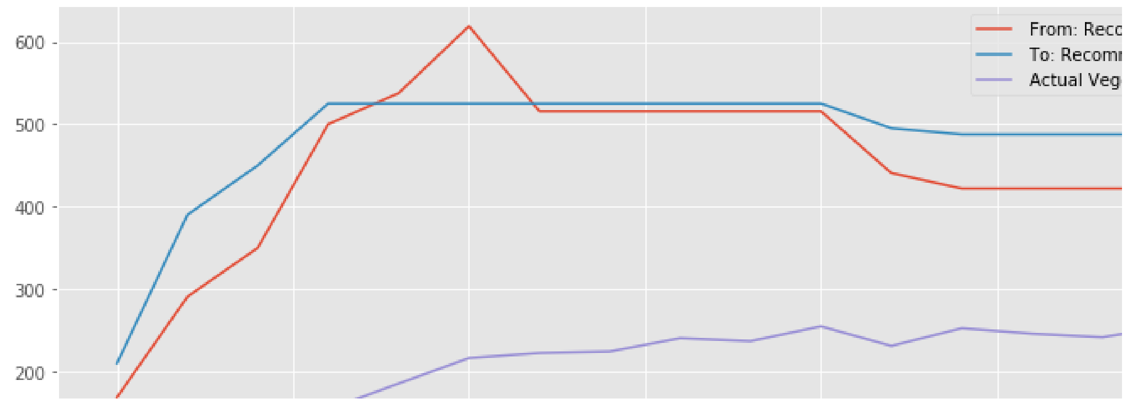
```
df9.plot.bar();

# plt.xlabel('Age Groups (from)')
# plt.ylabel('Amount in Gms')

plt.xticks(range(len(df['age_to'])), df['age_from']);
plt.xlabel('Age group: From\n Colors do not mean anything, just the different column')
plt.ylabel('Intake amount in gms')
```

```
Out[8]: Text(0,0.5,'Intake amount in gms')
```



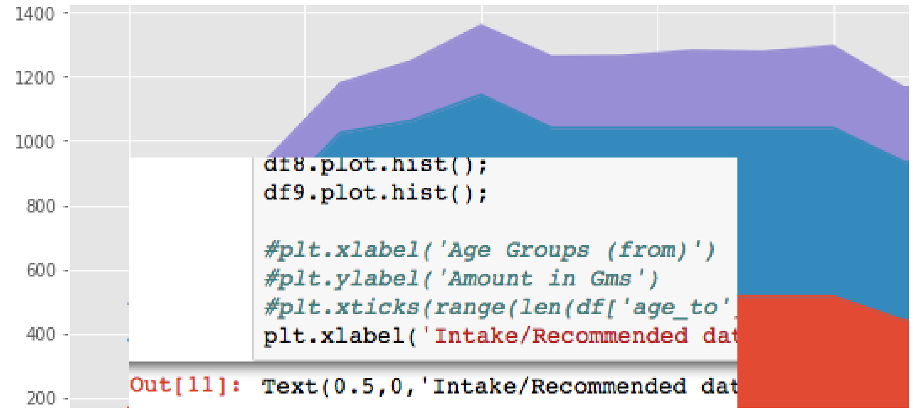
```
df8.plot.line();
df9.plot.line();

# plt.xlabel('Age Groups (from)')
# plt.ylabel('Amount in Gms')

plt.xticks(range(len(df['age_to'])), df['age_from']);
plt.xlabel('Age group: From')
plt.ylabel('Intake amount in gms')
```

```
9]: Text(0,0.5,'Intake amount in gms')
```



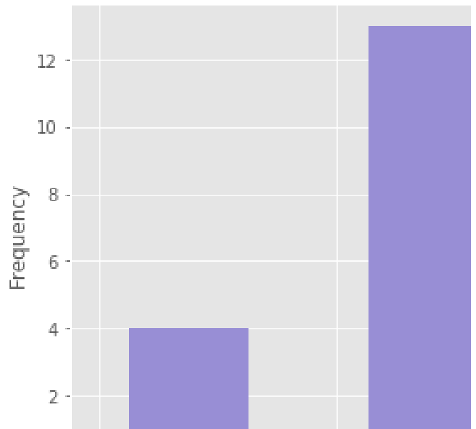
```
df8.plot.area();
df9.plot.area();


plt.xticks(range(len(df['age_to'])), df['age_from']);
plt.xlabel('Age group: From')
plt.ylabel('Intake amount in gms')
```

```
]: Text(0,0.5,'Intake amount in gms')
```



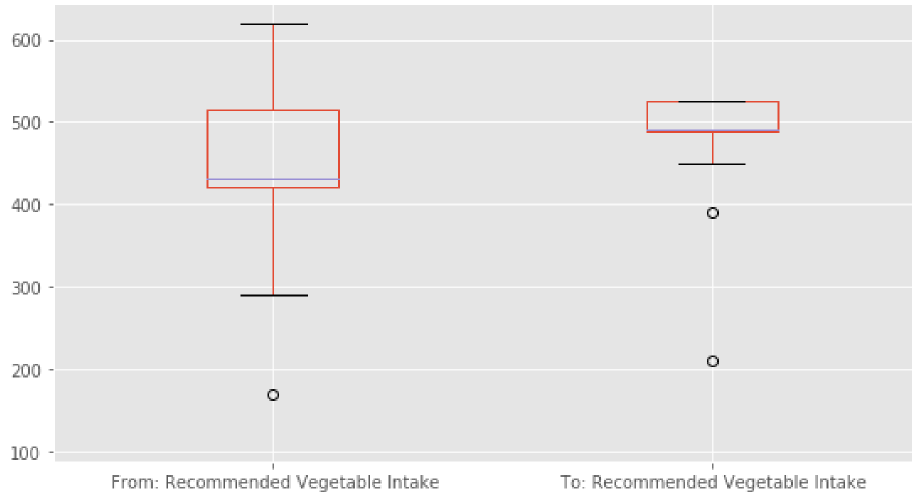
```
df8.plot.hist();
df9.plot.hist();

#plt.xlabel('Age Groups (from)')
#plt.ylabel('Amount in Gms')
#plt.xticks(range(len(df['age_to'
plt.xlabel('Intake/Recommended dat
```

```
Out[11]: Text(0.5,0,'Intake/Recommended dat
```



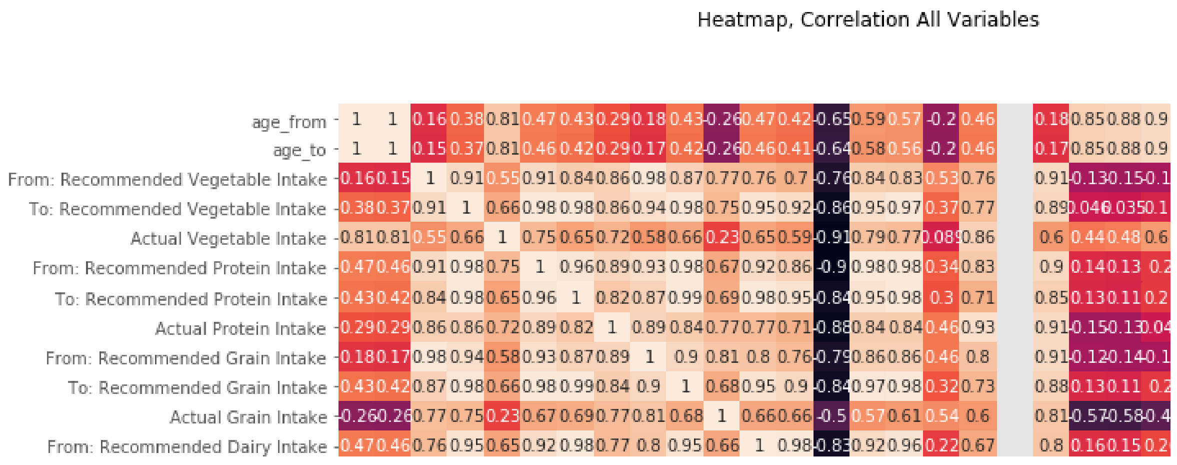
```
df88.plot.box();
#df99.plot.box();
plt.xticks(range(1, len(df88.columns)+1),df88.columns,rotation=9

#plt.xlabel('Age Groups (from)')
plt.ylabel('Amount in Gms')

#plt.xticks(range(len(df['age_to'])), df['age_from']);
```

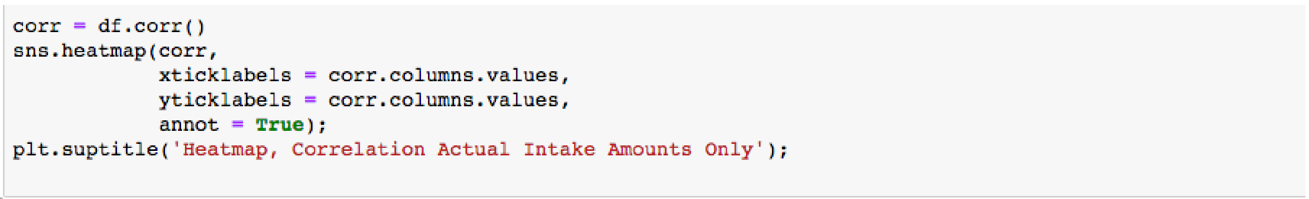
```
]: Text(0,0.5,'Amount in Gms')
```



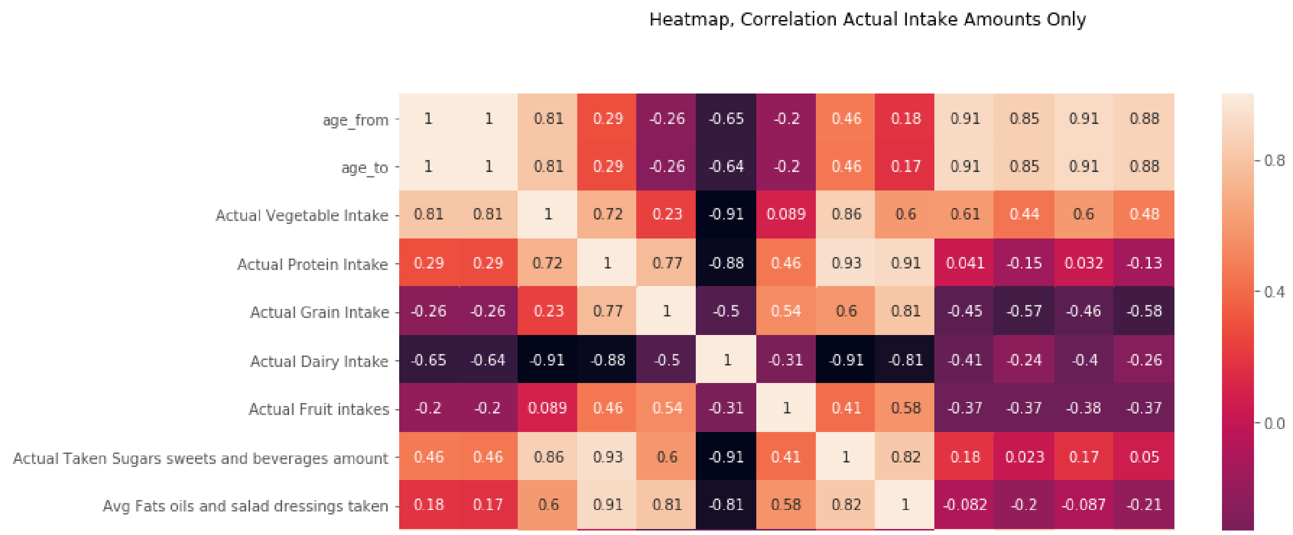
Regression with all Variables/Features

```
corr = df.corr()
sns.heatmap(corr,
            xticklabels = corr.columns.values,
            yticklabels = corr.columns.values,
            annot = True);
plt.suptitle('Heatmap, Correlation All Variables');
```

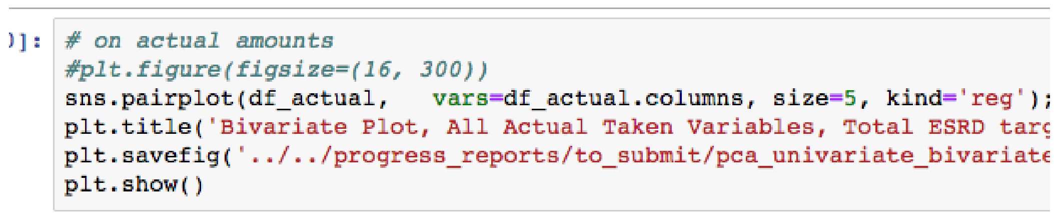




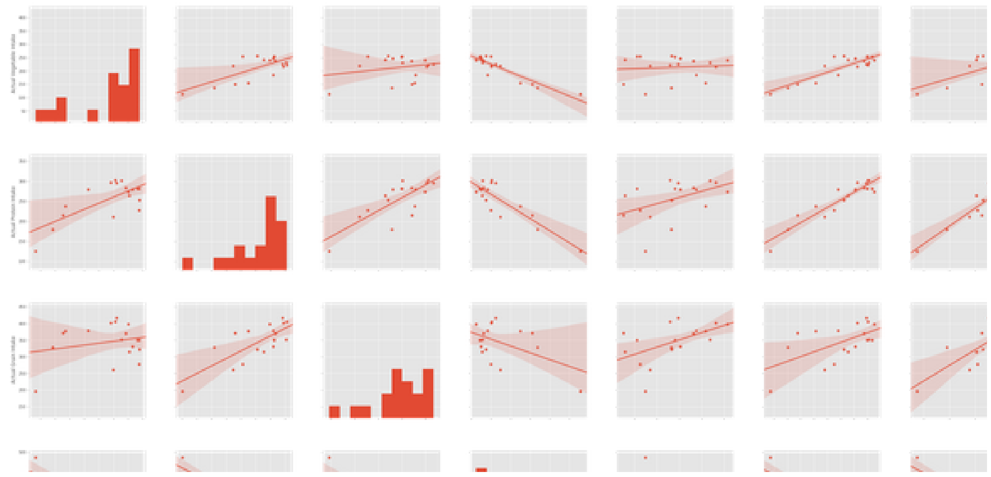
```
corr = df.corr()
sns.heatmap(corr,
            xticklabels = corr.columns.values,
            yticklabels = corr.columns.values,
            annot = True);
plt.suptitle('Heatmap, Correlation Actual Intake Amounts Only');
```



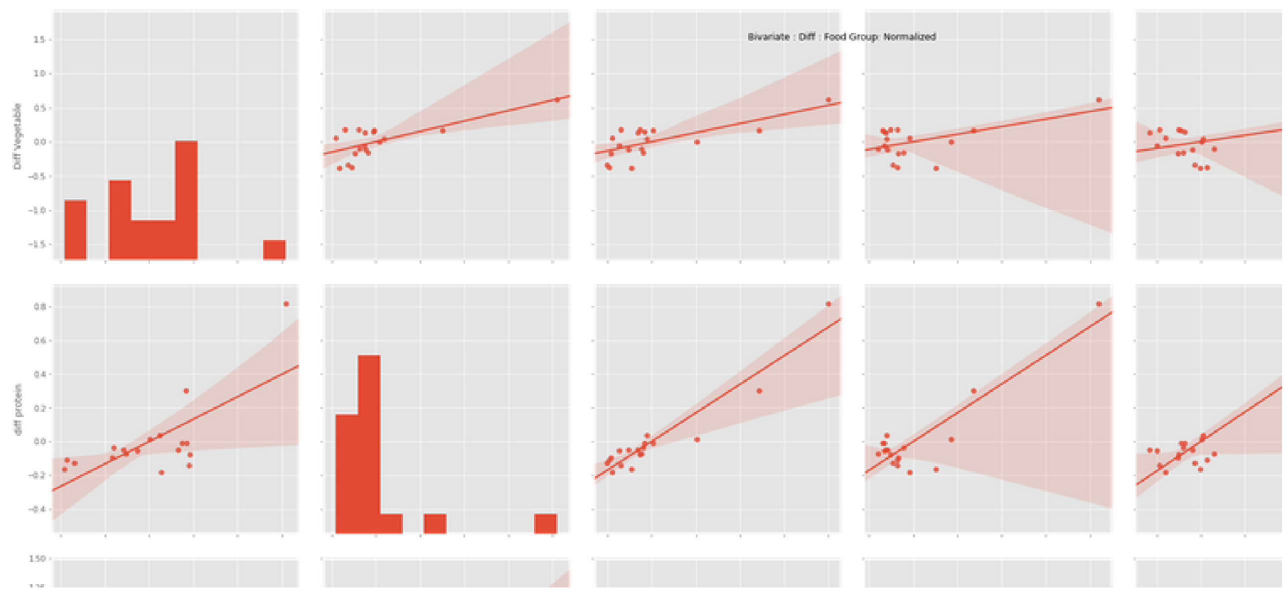
Bivariate Exploratory



```
)]: # on actual amounts
    #plt.figure(figsize=(16, 300))
    sns.pairplot(df_actual,   vars=df_actual.columns, size=5, kind='reg');
    plt.title('Bivariate Plot, All Actual Taken Variables, Total ESRD targ
    plt.savefig('../../progress_reports/to_submit/pca_univariate_bivariate
    plt.show()
```



```
sns.pairplot(df_normalized_diff,   vars=df_normalized_diff.columns, size=5, kind='reg', as
plt.suptitle('Bivariate : Diff : Food Group: Normalized\n')
plt.ylabel('Difference in Intake amount from Recommended : Normalized')
plt.xlabel('Intakes')
plt.savefig('../../progress_reports/to_submit/pca_univariate_bivariate/bivariate_diff_norm
plt.show()
```


Bivariate : Diff : Food Group: Normalized
Diff Vegetable
diff protein