# hw3

October 25, 2018

## 1 STA 141B: Homework 3

Fall 2018

### 1.1 Information

After the colons (in the same line) please write just your first name, last name, and the 9 digit student ID number below.

First Name:Ruochen
Last Name:Zhong
Student ID:912888970

### 1.2 Instructions

#### 1.2.1 New item: Please print your answer notebook to pdf (make sure that it is not too many pages, > 10, due to long output) and submit as the homework solution with your zip file.

We use a script that extracts your answers by looking for cells in between the cells containing the exercise statements. So you

- MUST add cells in between the exercise statements and add answers within them and
- MUST NOT modify the existing cells, particularly not the problem statement

To make markdown, please switch the cell type to markdown (from code) - you can hit 'm' when you are in command mode - and use the markdown language. For a brief tutorial see: https://daringfireball.net/projects/markdown/syntax

#### 1.2.2 Introduction

The US Department of Agriculture publishes price estimates for fruits and vegetables online. The most recent estimates are based on a 2013 survey of US retail stores.

The estimates are provided as a collection of MS Excel files, with one file per fruit or vegetable. The `hw3_data.zip` file contains the fruit and vegetable files in the directories `fruit` and `vegetables`, respectively.

**Exercise 1.** Use pandas to extract the "Fresh" row(s) from the fruit Excel files. Combine the data into a single data frame. Your data frame should look something like this:

| type | food | form | price_per_lb | yield | lb_per_cup | price_per_cup |
|---|---|---|---|---|---|---|
| fruit | watermelon | Fresh1 | 0.333412 | 0.52 | 0.330693 | 0.212033 |
| fruit | cantaloupe | Fresh1 | 0.535874 | 0.51 | 0.374786 | 0.3938 |
| vegetables | onions | Fresh1 | 1.03811 | 0.9 | 0.35274 | 0.406868 |
| ... | | | | | | |

It's okay if the rows and columns of your data frame are in a different order. These modules are especially relevant:

- `str` methods
- `os`
- `os.path`
- pandas: `read_excel()`, `concat()`, `.fillna()`, `.str`, plotting methods

Ask questions and search the documentation/web to find the functions you need.

```
In [1]: import os
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        # check the directory
        os.getcwd()
```

```
Out[1]: '/Users/ruochenzhong/Desktop/STA141B'
```

```
In [2]: # read the list of files in fruit and vegetables
        ls_fruit_file = os.listdir('hw3_data/fruit')
        ls_veg_file = os.listdir('hw3_data/vegetables')
```

```
In [3]: def read_file(file_name,list_name):
            """ read the excels to be a dataframe, extract and combine useful rows and columns
            Args:
                file_name: the name of the file in the directory
                list_name: the list of indivudual excel names in the file

            Returns:
                data: a dataframe contains extracted rows from the excel files
            """
            # create an empty dataframe
            data = pd.DataFrame()

            for i in range(0,len(list_name)):
               # read the rows in every file in the fruit and store them into a dataframe
               row = pd.read_excel(os.path.join(file_name,list_name[i]), header = None)
               # add a column to show the food type of those rows
               row['food'] = list_name[i].split('.')[0]
               # combine dataframes together
```

```
            data = data.append(row,ignore_index = True, sort = True)

        # extract the rows contain "fresh"
        data = data[data[0].str.contains("Fresh") == True]
        # drop useless columns
        data = data.dropna(thresh=1, axis=1)
        del data[2],data[5]
        # rename the columns and add their types
        data.columns = ['form', 'price_per_lb', 'yield',
                        'lb_per_cup', 'price_per_cup','food']
        data['type'] = os.path.split(file_name)[1]
        #reset the index and the order
        data = data.reset_index(drop = True)
        data = data[['type','food','form', 'price_per_lb',
                     'yield', 'lb_per_cup', 'price_per_cup']]
        return(data)

    fruit = read_file('hw3_data/fruit', ls_fruit_file)
    print("number of rows =", fruit.shape[0])
    fruit.head()
```

```
number of rows = 24
```

Out[3]:

| | type | food | form | price_per_lb | yield | lb_per_cup | price_per_cup |
|---|---|---|---|---|---|---|---|
| 0 | fruit | cherries | Fresh1 | 3.59299 | 0.92 | 0.341717 | 1.33455 |
| 1 | fruit | tangerines | Fresh1 | 1.37796 | 0.74 | 0.407855 | 0.759471 |
| 2 | fruit | oranges | Fresh1 | 1.03517 | 0.73 | 0.407855 | 0.578357 |
| 3 | fruit | blackberries | Fresh1 | 5.77471 | 0.96 | 0.31967 | 1.92292 |
| 4 | fruit | apricots | Fresh1 | 3.04007 | 0.93 | 0.363763 | 1.1891 |

**Exercise 2.** Reuse your code from exercise 1.1 to extract the "Fresh" row(s) from the vegetable Excel files.

Does your code produce the correct prices for tomatoes? If not, why not? Do any other files have the same problem as the tomatoes file?

You don't need to extract the prices for these problem files. However, make sure the prices are extracted for files like asparagus that don't have this problem.

In [4]:
```
veg = read_file('hw3_data/vegetables', ls_veg_file)
print("number of rows =", veg.shape[0])
veg.head()
```

```
number of rows = 33
```

Out[4]:

| | type | food | form | price_per_lb | \ |
|---|---|---|---|---|---|
| 0 | vegetables | turnip_greens | Fresh1 | 2.47175 | |
| 1 | vegetables | artichoke | Fresh1 | 2.21305 | |
| 2 | vegetables | acorn_squash | Fresh1 | 1.17225 | |

```
3  vegetables         celery                        Fresh1          NaN
4  vegetables      cucumbers  Fresh, consumed with peel1      1.29593

      yield lb_per_cup price_per_cup
0      0.75    0.31967       1.05353
1  0.375309   0.385809       2.27497
2  0.458554   0.451948       1.15536
3       NaN        NaN           NaN
4      0.97   0.264555      0.353448
```

It doesn't produce the correct price for tomatoes' price becuase it is missing values. There are another 7 files which have the same problems with the tomatoes.

```
In [5]: # subset rows with missing prices and print them out
        veg_miss = veg[pd.isnull(veg['price_per_cup'])]
        veg_miss['food']
```

```
Out[5]: 3              celery
        8           mushrooms
        11           broccoli
        16            carrots
        20        cauliflower
        21           tomatoes
        23    lettuce_romaine
        30            spinach
        Name: food, dtype: object
```

**Exercise 3.** Remove rows without a price from the vegetable data frame and then combine the fruit and vegetable data frames. Make sure all columns of numbers are numeric (not strings).

```
In [6]: # subset vegetable's dataframe by excluding rows with missing prices
        veg2 = veg[pd.notnull(veg['price_per_cup'])]
        # combining fruit's and vegetable's dataframe
        total = fruit.append(veg2, ignore_index = True)
        # making columns of numbers to numeric
        for i in range(3,7):
            total.iloc[:,i] = pd.to_numeric(total.iloc[:,i])

        print("number of rows =", total.shape[0])
        total.head()
```

```
number of rows = 49
```

```
Out[6]:     type          food     form  price_per_lb  yield  lb_per_cup  \
        0  fruit      cherries   Fresh1      3.592990   0.92    0.341717
        1  fruit    tangerines   Fresh1      1.377962   0.74    0.407855
        2  fruit       oranges   Fresh1      1.035173   0.73    0.407855
        3  fruit  blackberries  Fresh1      5.774708   0.96    0.319670
```

```
4  fruit      apricots   Fresh1      3.040072   0.93    0.363763

   price_per_cup
0       1.334548
1       0.759471
2       0.578357
3       1.922919
4       1.189102
```

```
In [7]: # check the class
        print(total.dtypes)
```

```
type              object
food              object
form              object
price_per_lb     float64
yield            float64
lb_per_cup       float64
price_per_cup    float64
dtype: object
```

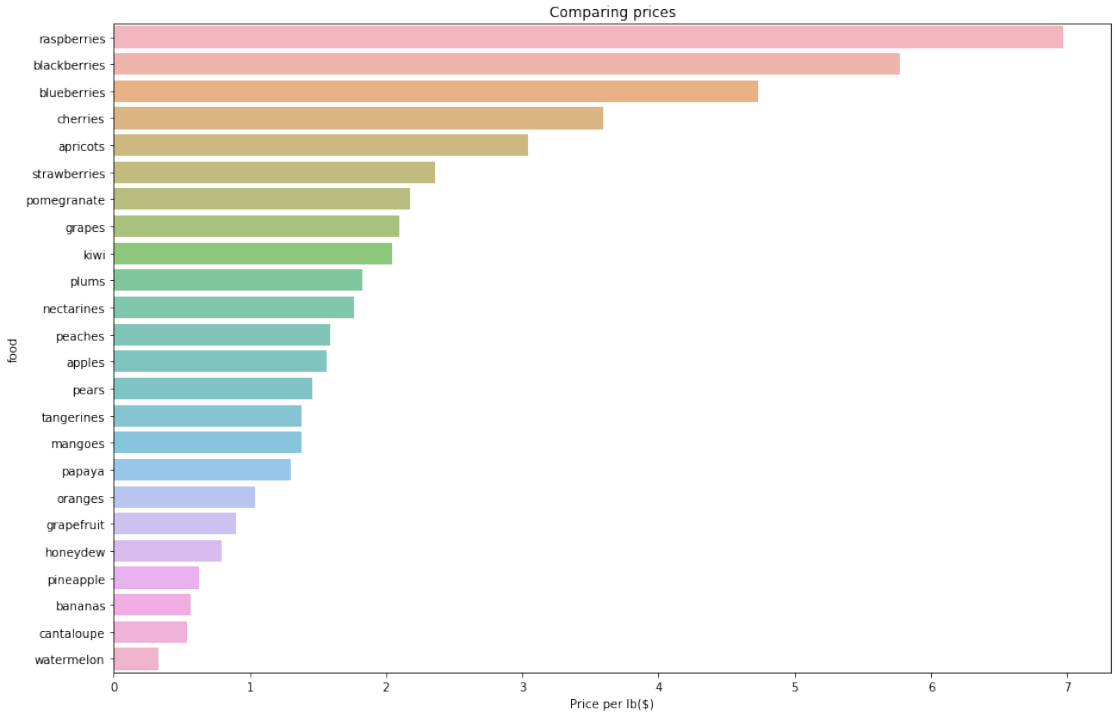
**Exercise 4.** Discuss the questions below (a paragraph each is sufficient). Use plots to support your ideas.

- What kinds of fruits are the most expensive (per pound)? What kinds are the least expensive?
- How do the price distributions compare for fruit and vegetables?
- Which foods are the best value for the price?
- What's something surprising about this data set?
- Which foods do you expect to provide the best combination of price, yield, and nutrition? A future assignment may combine this data set with another so you can check your hypothesis.

```
In [8]: # sort the dataframe by price to form a new dataframe
        fruit_sort = fruit.sort_values('price_per_lb', ascending = False)
        # change string to numeric
        for i in range(3,7):
            fruit_sort.iloc[:,i] = pd.to_numeric(fruit_sort.iloc[:,i])

        plt.figure(figsize=(15,10))
        sns.barplot(fruit_sort['price_per_lb'], fruit_sort['food'], alpha = 0.7)
        plt.xlabel('Price per lb($)')
        plt.title('Comparing prices')
```

```
Out[8]: Text(0.5,1,'Comparing prices')
```



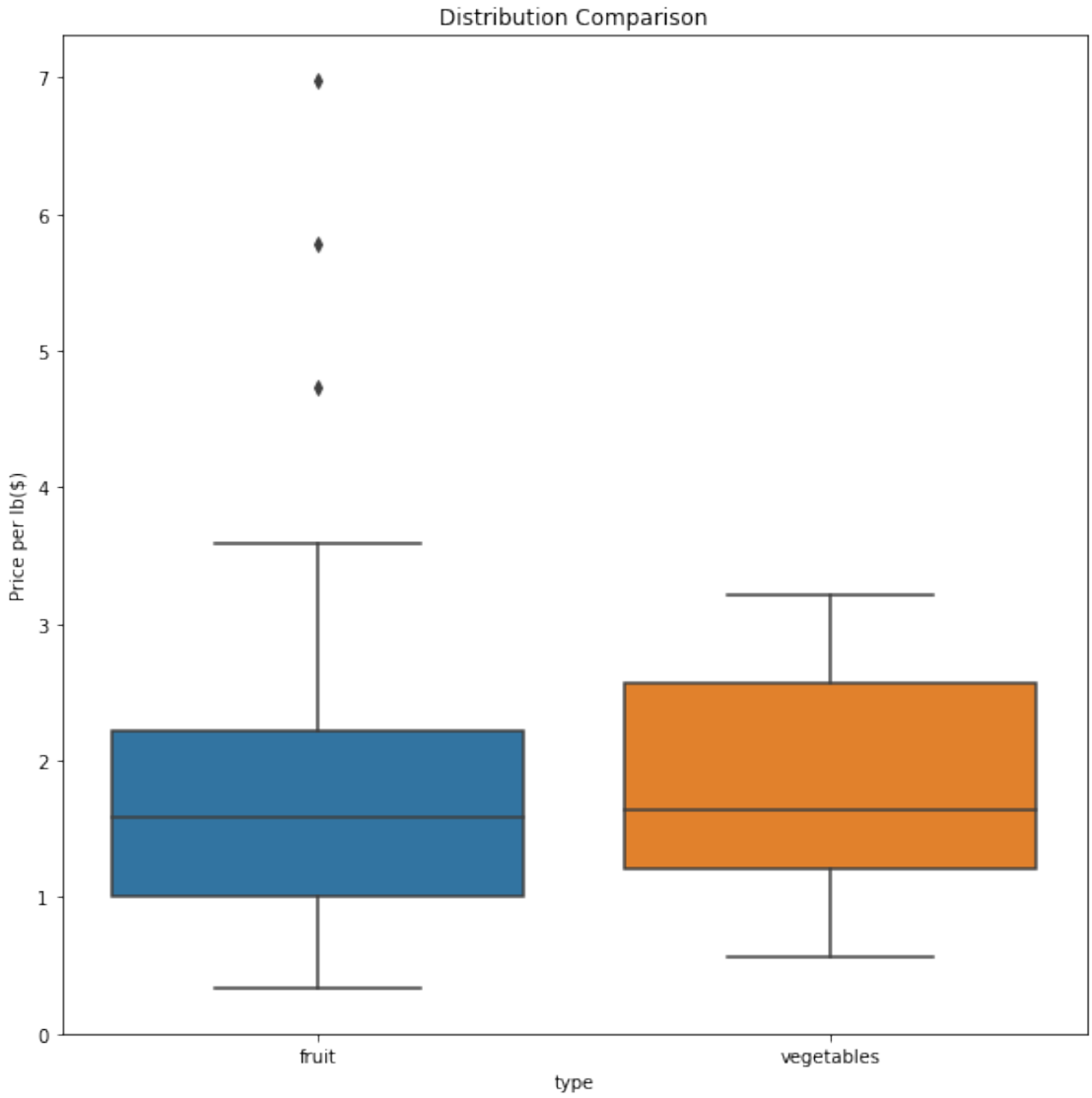

Question1:

From this graph, it shows that raspberries have the highest price per pound and watermelon has the lowest price per pound.

```
In [9]: plt.figure(figsize=(10,10))
        plot1 = sns.boxplot(x="type", y="price_per_lb", data=total)
        plot1.set_ylabel('Price per lb($)')
        plot1.set_title('Distribution Comparison')
        # show a summery of fruit and vegetables distribution
        total.groupby('type')['price_per_lb'].describe()
```

```
Out[9]:             count      mean       std       min       25%       50%       75%  \
        type
        fruit        24.0  2.076877  1.675687  0.333412  1.000830  1.579351  2.219895
        vegetables   25.0  1.838701  0.817285  0.564320  1.213039  1.639477  2.569235

                         max
        type
        fruit       6.975811
        vegetables  3.213552
```



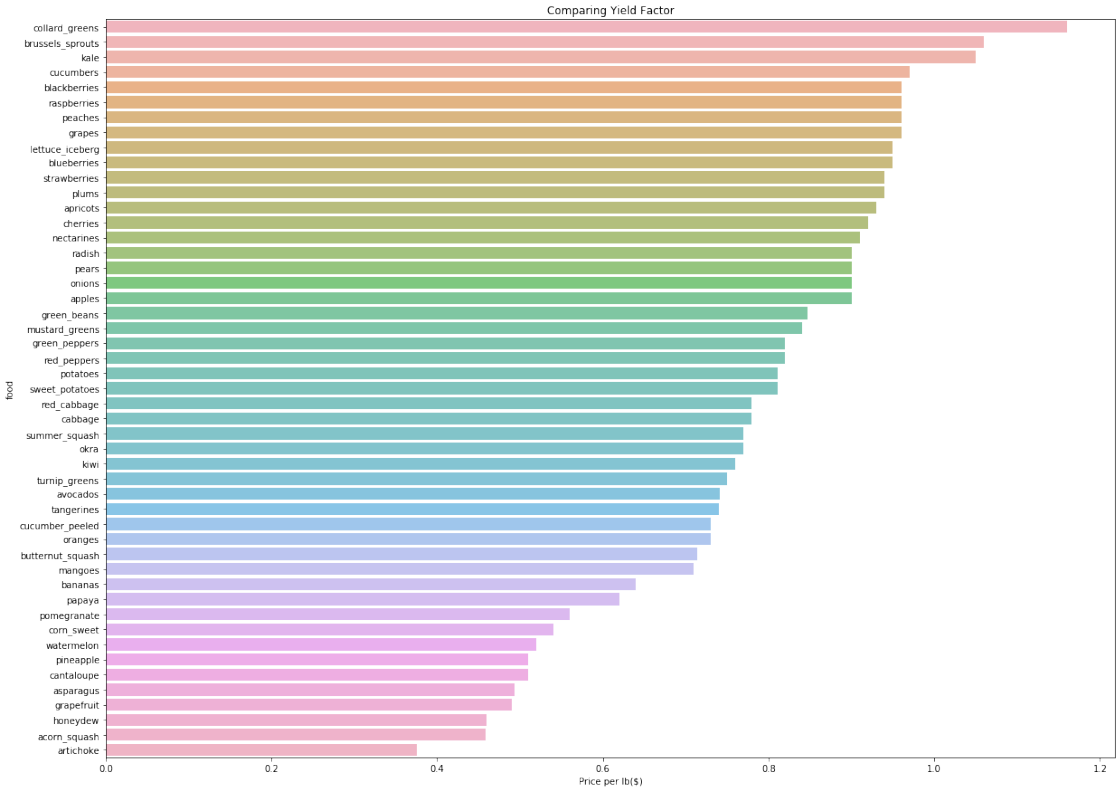

Question 2:

From the distribution comparison of fruits and vegetables, it shows that the price range of vegetables tends to be narrower. However, the distribution of fruits shows a wide range of prices and there are some outliers whose prices are extremely high.Fruit's price percentile between 75% to 100% has a larger range while vegetable's percentile between 50% to 75% has a larger range.Both distributions tends to be right-skewed, which means more of fruits and vegetables tends to be in the lower price range but several of fruits and vegetables have a very high prices and make the average of the whole catagories exceed the median.

```
In [10]: # distinguish the same rows of cucumver and cabbage
         total.loc[28,'food'] = 'cucumber_peeled'
         total.loc[36,'food'] = 'red_cabbage'
         total_sort = total.sort_values('yield', ascending = False)
```

```
        plt.figure(figsize=(20,15))
        sns.barplot(total_sort['yield'], total_sort['food'], alpha = 0.7)
        plt.xlabel('Price per lb($)')
        plt.title('Comparing Yield Factor')
```

```
Out[10]: Text(0.5,1,'Comparing Yield Factor')
```




Question 3:

From the above graph, according to the yield, collard greens hace the best value for the price. The yield means the what percent of the raw food can be edible. For the collard greens, more than 100% percents of it can be eated, this means it not only doesn't have any unedible but its weight even increasing when in the process of cooking. Therefore, it has the best value for the its price because all the orginal part of it and its increasing weight can be eaten.

Question 4:

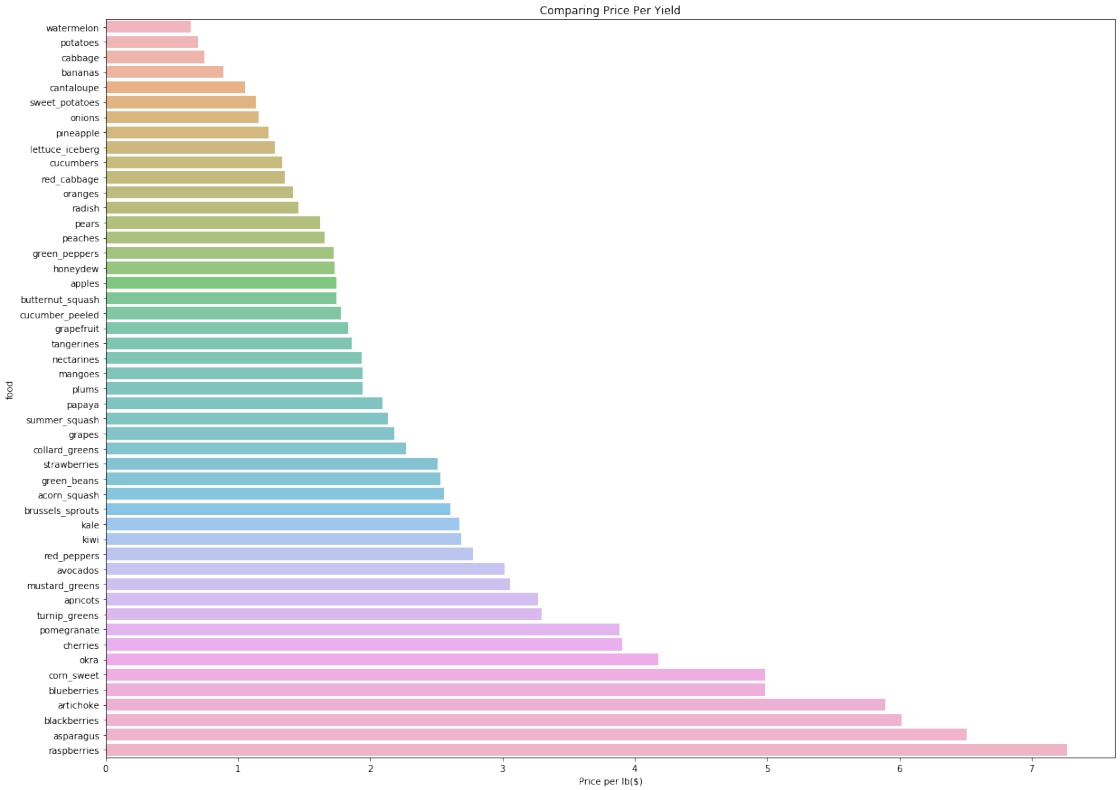
I think the surprising thing is that three kinds of food have yield factors which exceed 1. At first, this value seems impossible because in common sense, the highest value of yield factor can only be 1. When I open these 3 foods' excel, I found that all of them are cooked before selling.This means these food possess a high percentage to be eligible and in the process of cooking, their weights increase by some factors such as absorb water during boiling or steaming. Therefore, this can explain why the yield of these foods can exceed 1.

```
In [11]: # create a new variable in dataframe
         total_sort['price/yield'] = total_sort['price_per_lb'] / total_sort['yield']
```

```
total_sort2 = total_sort.sort_values('price/yield')
plt.figure(figsize=(20,15))
sns.barplot(total_sort2['price/yield'], total_sort2['food'], alpha = 0.7)
plt.xlabel('Price per lb($)')
plt.title('Comparing Price Per Yield')
total_sort2.head(3)
```

Out[11]:

| | type | food | form | price_per_lb | yield | lb_per_cup | price_per_cup | price/yield |
|---|---|---|---|---|---|---|---|---|
| 13 | fruit | watermelon | Fresh1 | 0.333412 | 0.520000 | 0.330693 | 0.212033 | 0.641177 |
| 32 | vegetables | potatoes | Fresh1 | 0.564320 | 0.811301 | 0.264555 | 0.184017 | 0.695574 |
| 35 | vegetables | cabbage | Fresh green cabbage1 | 0.579208 | 0.778797 | 0.330693 | 0.245944 | 0.743722 |




Question 5:

I use the "price per pound" to divide the "yield" for this question. It helps me get which foods has the best combiantion of price and yield. These three foods has the best combination of price and yield. After checking their nutrition proportions, watermelon tends to have less nutritions compared to others. For potatoes and cabbage, I expect that potatoes can provide the best combination of price, yield, and nutrition becuase after checking at

http://foodstruct.com/compare/potato-vs-cabbage, they almost have same overall nutrition values, but potatoes have a better price per yield.