In [287]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split
import plotly.express as px
from sklearn.preprocessing import LabelEncoder
```

In [203]:
```
df = pd.read_csv("avocado.csv")
df
```

Out[203]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Sr B
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042
3	3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677
4	4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986
...	...	...	...	...	...	...	...	...	
18244	7	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	13066
18245	8	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940
18246	9	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351
18247	10	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919
18248	11	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988

18249 rows × 14 columns

In [204]:
```
df.shape
```

Out[204]:
```
(18249, 14)
```

In [205]:
```
df.columns.values
```

Out[205]:
```
array(['Unnamed: 0', 'Date', 'AveragePrice', 'Total Volume', '4046',
       '4225', '4770', 'Total Bags', 'Small Bags', 'Large Bags',
       'XLarge Bags', 'type', 'year', 'region'], dtype=object)
```

In [207]:
```
df = df.drop('Unnamed: 0', axis=1)
```

In [286]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18249 entries, 0 to 18248
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Date          18249 non-null  object 
 1   AveragePrice  18249 non-null  float64
 2   Total Volume  18249 non-null  float64
 3   4046          18249 non-null  float64
 4   4225          18249 non-null  float64
 5   4770          18249 non-null  float64
 6   Total Bags    18249 non-null  float64
 7   Small Bags    18249 non-null  float64
 8   Large Bags    18249 non-null  float64
 9   XLarge Bags   18249 non-null  float64
 10  type          18249 non-null  int32  
 11  year          18249 non-null  int64  
 12  region        18249 non-null  object 
dtypes: float64(9), int32(1), int64(1), object(2)
memory usage: 1.7+ MB
```

In [276]:
```
pd.set_option('float_format', '{:f}'.format)
df.describe()
```

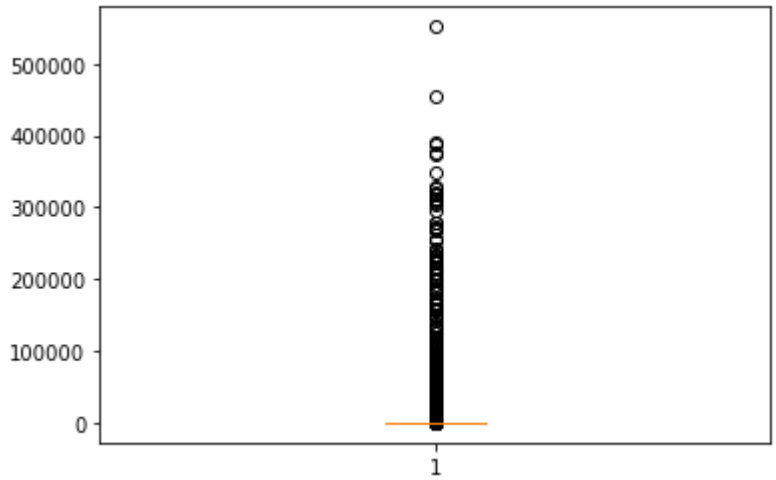
Out[276]:

	AveragePrice	Total Volume	4046	4225	4770	
count	18249.000000	18249.000000	18249.000000	18249.000000	18249.000000	
mean	1.405978	850644.013009	293008.424531	295154.568356	22839.735993	2
std	0.402677	3453545.355399	1264989.081763	1204120.401135	107464.068435	9
min	0.440000	84.560000	0.000000	0.000000	0.000000	
25%	1.100000	10838.580000	854.070000	3008.780000	0.000000	
50%	1.370000	107376.760000	8645.300000	29061.020000	184.990000	
75%	1.660000	432962.290000	111020.200000	150206.860000	6243.420000	1
max	3.250000	62505646.520000	22743616.170000	20470572.610000	2546439.110000	193

75% and max values of XLarge Bags has vast difference hence outliers. In XLarge Bags we may find outliers

In [277]:
```
plt.boxplot(x=df['XLarge Bags'])
```

Out[277]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1881e8740c8>,
  <matplotlib.lines.Line2D at 0x1881e87eb08>],
 'caps': [<matplotlib.lines.Line2D at 0x1881e887ac8>,
  <matplotlib.lines.Line2D at 0x1881e887fc8>],
 'boxes': [<matplotlib.lines.Line2D at 0x1881e87e908>],
 'medians': [<matplotlib.lines.Line2D at 0x1881e887c48>],
 'fliers': [<matplotlib.lines.Line2D at 0x1881e88eb08>],
 'means': []}
```



In [212]:
```
df.type.unique()
```

Out[212]:
```
array(['conventional', 'organic'], dtype=object)
```

In [282]:
```
df.region.unique()
```

Out[282]:
```
array(['Albany', 'Atlanta', 'BaltimoreWashington', 'Boise', 'Boston',
       'BuffaloRochester', 'California', 'Charlotte', 'Chicago',
       'CincinnatiDayton', 'Columbus', 'DallasFtWorth', 'Denver',
       'Detroit', 'GrandRapids', 'GreatLakes', 'HarrisburgScranton',
       'HartfordSpringfield', 'Houston', 'Indianapolis', 'Jacksonville',
       'LasVegas', 'LosAngeles', 'Louisville', 'MiamiFtLauderdale',
       'Midsouth', 'Nashville', 'NewOrleansMobile', 'NewYork',
       'Northeast', 'NorthernNewEngland', 'Orlando', 'Philadelphia',
       'PhoenixTucson', 'Pittsburgh', 'Plains', 'Portland',
       'RaleighGreensboro', 'RichmondNorfolk', 'Roanoke', 'Sacramento',
       'SanDiego', 'SanFrancisco', 'Seattle', 'SouthCarolina',
       'SouthCentral', 'Southeast', 'Spokane', 'StLouis', 'Syracuse',
       'Tampa', 'TotalUS', 'West', 'WestTexNewMexico'], dtype=object)
```

In [215]:
```
df.region.value_counts()
```

Out[215]:
```
Detroit                338
SanFrancisco           338
HarrisburgScranton     338
Denver                 338
GrandRapids            338
Nashville              338
Midsouth               338
Philadelphia           338
Charlotte              338
Plains                 338
HartfordSpringfield    338
Chicago                338
LasVegas               338
Jacksonville           338
Portland               338
Seattle                338
Pittsburgh             338
RichmondNorfolk        338
Spokane                338
West                   338
Boise                  338
Syracuse               338
Northeast              338
NewOrleansMobile       338
RaleighGreensboro      338
Southeast              338
StLouis                338
California             338
Houston                338
PhoenixTucson          338
Sacramento             338
Boston                 338
Louisville             338
MiamiFtLauderdale      338
SouthCentral           338
LosAngeles             338
SanDiego               338
BaltimoreWashington    338
Columbus               338
CincinnatiDayton       338
Albany                 338
NorthernNewEngland     338
NewYork                338
SouthCarolina          338
TotalUS                338
DallasFtWorth          338
Indianapolis           338
BuffaloRochester       338
Roanoke                338
Orlando                338
Tampa                  338
Atlanta                338
GreatLakes             338
```

```
WestTexNewMexico       335
Name: region, dtype: int64
```

In [214]:
```
df.type.value_counts()
```

Out[214]:
```
conventional    9126
organic         9123
Name: type, dtype: int64
```

There are almost similar number of types of avocado sold

In [216]:
```
df.year.unique()
```

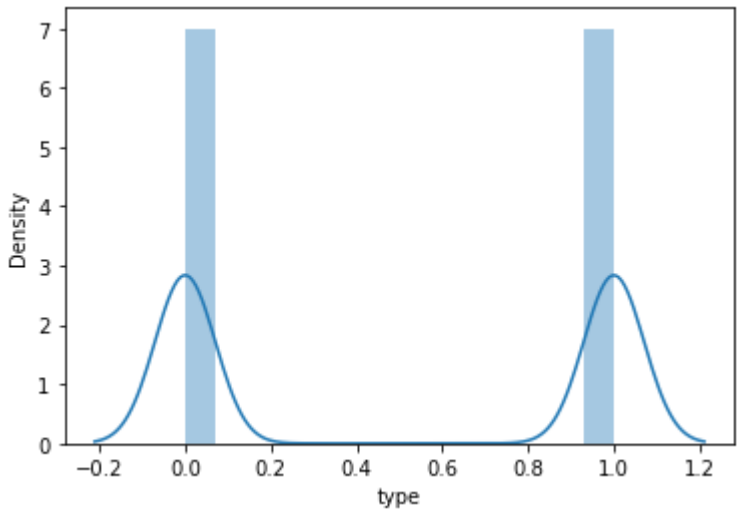
Out[216]:
```
array([2015, 2016, 2017, 2018], dtype=int64)
```

The data consists of weekly avacado retail prices from the year 2015 to 2018

In [256]:
```
df1= df.copy()
df1['type'] = df['type'].astype('category')
df1['type'] = df1['type'].cat.codes                    #encoding conventional av
df1
sns.distplot(df1.type)
```

```
C:\Users\Nikita\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning:

`distplot` is a deprecated function and will be removed in a future versio
n. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
```

Out[256]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x188113ed788>
```

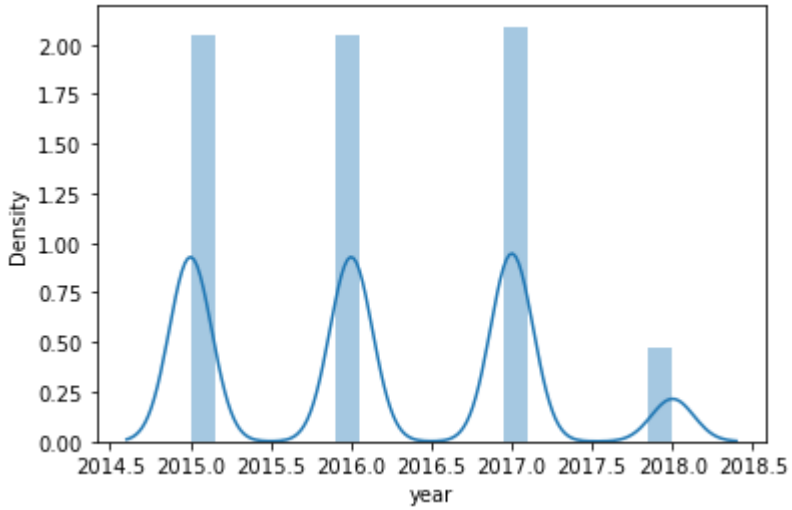


Here we are gettin 2 distribution plots for continous and catagorical data for 2 categories (0= conventional type , 1 = organic type)

In [218]:
```
sns.distplot(df.year)
```

```
C:\Users\Nikita\anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning:

`distplot` is a deprecated function and will be removed in a future versio
n. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function for histogr
ams).
```

Out[218]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1879ef850c8>
```



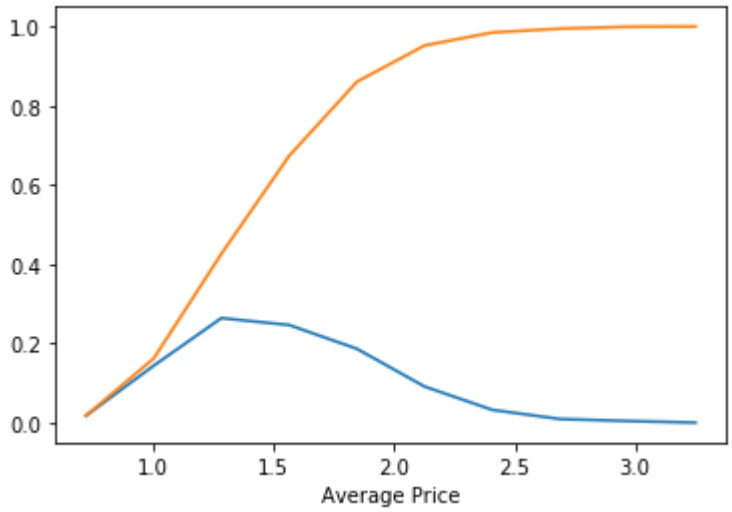
2018 has compariitively less data than that of other years

counts, bin_edges = np.histogram(df['Total Volume'], bins=10, density= True) plt.xlabel("Total Volume") pdf = counts/sum(counts) print("pdf",pdf) cdf = np.cumsum(pdf) print("cdf", cdf) plt.plot(bin_edges[1:], pdf); plt.plot(bin_edges[1:], cdf)

In [219]:
```
counts, bin_edges = np.histogram(df['AveragePrice'], bins=10, density= True)
plt.xlabel("Average Price")
pdf = counts/sum(counts)
print("pdf",pdf)
cdf = np.cumsum(pdf)
print("cdf", cdf)
plt.plot(bin_edges[1:], pdf);
plt.plot(bin_edges[1:], cdf)
```

```
pdf [0.01813798 0.14422708 0.26434325 0.24691764 0.18696915 0.09162146
 0.03276892 0.00969916 0.00471259 0.00060277]
cdf [0.01813798 0.16236506 0.42670831 0.67362595 0.8605951  0.95221656
 0.98498548 0.99468464 0.99939723 1.        ]
```

Out[219]:
```
[<matplotlib.lines.Line2D at 0x1879f045948>]
```



The average prices of avocados ranges from 0.44 to 3.25 for a single unit. According to the graph, majorly the price per avacado is between 1 to 1.5

In [220]:
```
sns.kdeplot(df.loc[(df['type']=='conventional'),'AveragePrice'], color='b', s
sns.kdeplot(df.loc[(df['type']=='organic'),'AveragePrice'], color='r', shade=
```

Out[220]: <matplotlib.axes._subplots.AxesSubplot at 0x187d1bea248>

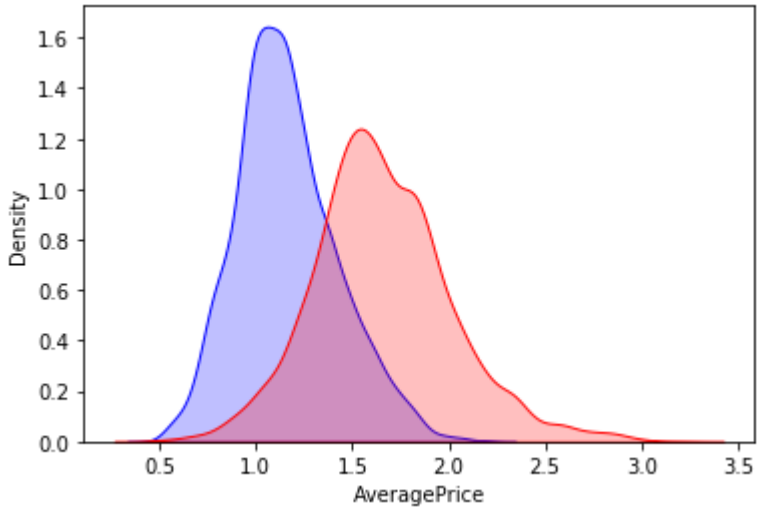


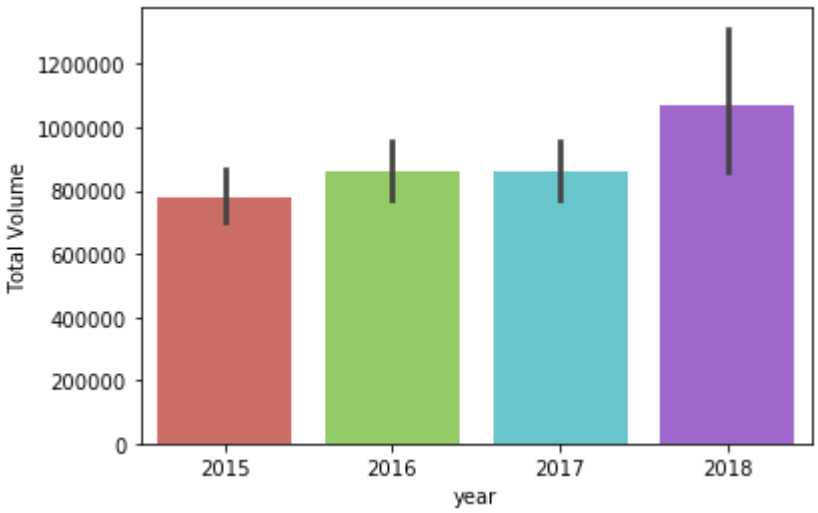
fig = px.histogram(df, x='AveragePrice', color='type', marginal='box', # or violin, rug
hover_data=df.columns)

fig.show()

In [257]:
```
sns.barplot(df['year'], df['Total Volume'], palette='hls')
```

```
C:\Users\Nikita\anaconda3\lib\site-packages\seaborn\_decorators.py:43: Futu
reWarning:

Pass the following variables as keyword args: x, y. From version 0.12, the
only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
```

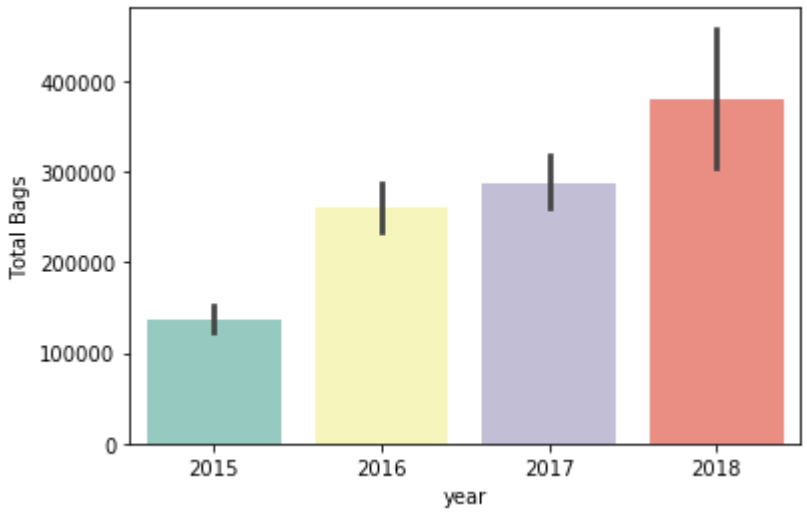
Out[257]: <matplotlib.axes._subplots.AxesSubplot at 0x1881deb0a08>



In [258]:
```
sns.barplot(df['year'], df['Total Bags'], palette='Set3')
```

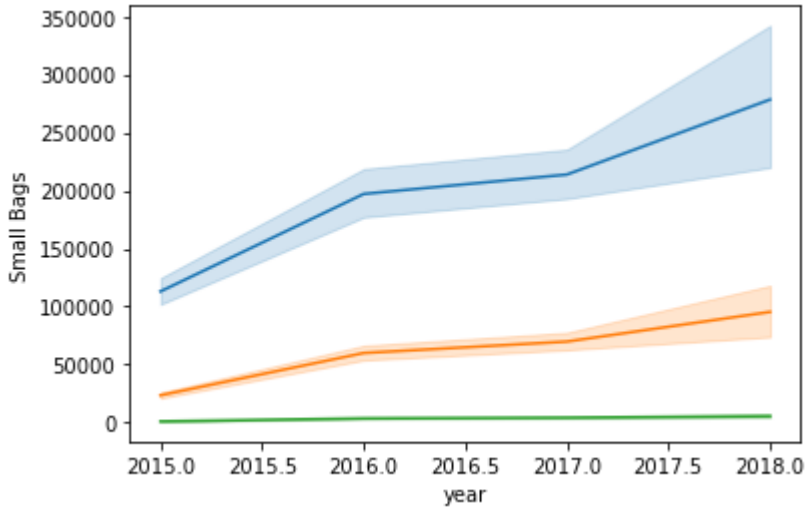
```
C:\Users\Nikita\anaconda3\lib\site-packages\seaborn\_decorators.py:43: Futu
reWarning:

Pass the following variables as keyword args: x, y. From version 0.12, the
only valid positional argument will be `data`, and passing other arguments
without an explicit keyword will result in an error or misinterpretation.
```

Out[258]: <matplotlib.axes._subplots.AxesSubplot at 0x1881df7d8c8>



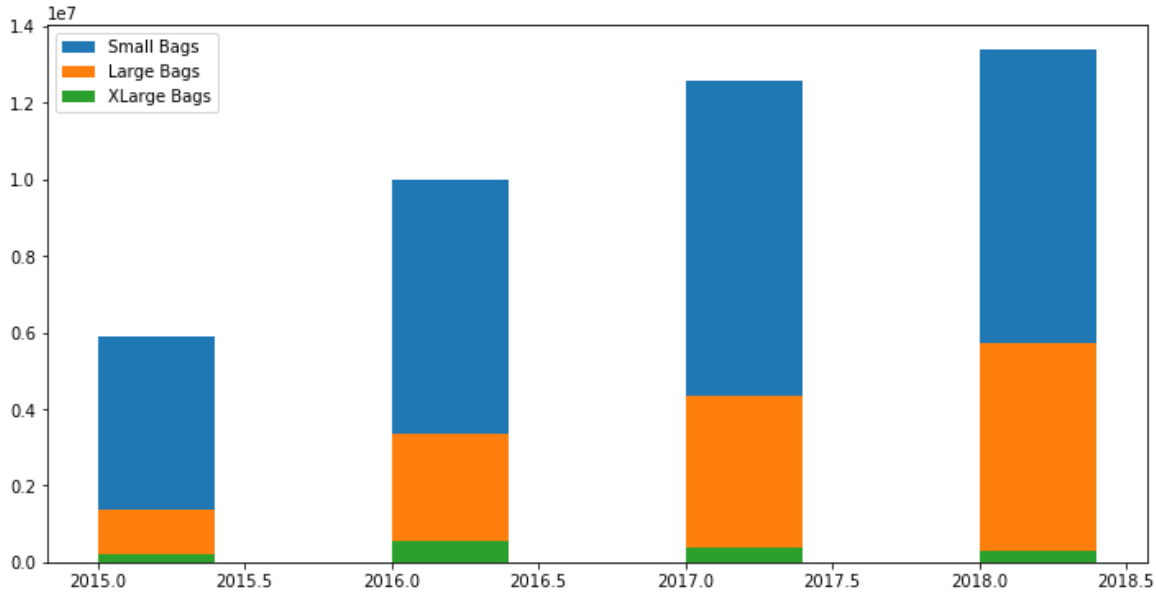
In [255]:
```
sns.lineplot(x=df['year'], y=df['Small Bags'])
sns.lineplot(x=df['year'], y=df['Large Bags'])
sns.lineplot(x=df['year'], y=df['XLarge Bags'])
```

Out[255]: <matplotlib.axes._subplots.AxesSubplot at 0x18811034a88>



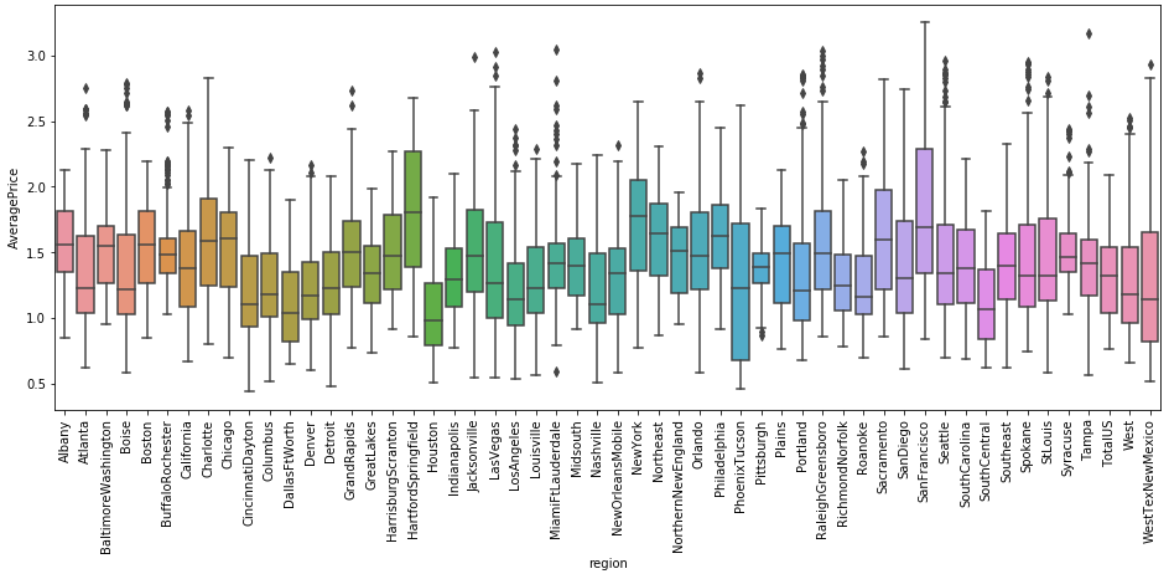
In [226]:
```
plt.figure(figsize=(12,6))
plt.bar(df['year'] + 0.2, df['Small Bags'], 0.4, label = 'Small Bags')
plt.bar(df['year'] + 0.2, df['Large Bags'], 0.4, label = 'Large Bags')
plt.bar(df['year'] + 0.2, df['XLarge Bags'], 0.4, label = 'XLarge Bags')
plt.legend()
```

Out[226]: <matplotlib.legend.Legend at 0x187dfacbe08>



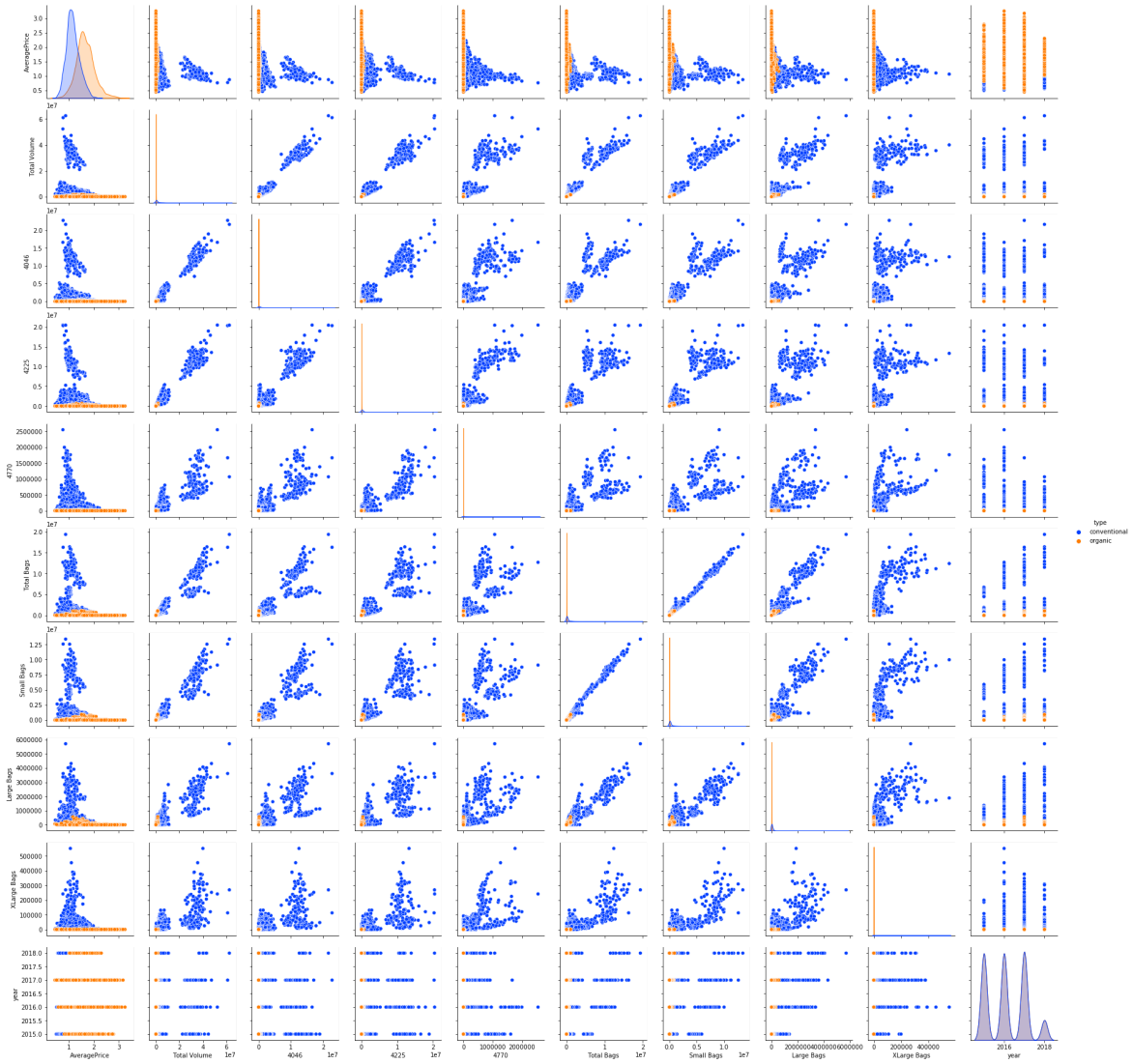
In [227]:
```
plt.figure(figsize=(16,6))
sns.boxplot(x=df['region'], y=df['AveragePrice'])
plt.xticks(rotation=90)
```

Out[227]:
```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
        34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
        51, 52, 53]),
 <a list of 54 Text xticklabel objects>)
```



In [228]:
```
sns.pairplot(df,hue='type',palette='bright')
```

Out[228]: <seaborn.axisgrid.PairGrid at 0x18804b01848>



In [229]:
```
le = LabelEncoder()
df['type'] = le.fit_transform(df['type'])
df
```

Out[229]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags
0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25
1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49
2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14
3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76
4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69
...	...	...	...	...	...	...	...	...	...
18244	2018-02-04	1.63	17074.83	2046.96	1529.20	0.00	13498.67	13066.82	431.85
18245	2018-01-28	1.71	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80
18246	2018-01-21	1.87	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31
18247	2018-01-14	1.93	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00
18248	2018-01-07	1.62	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01

18249 rows × 13 columns

In [230]:
```
X = df.iloc[:,2:-2]
X
```

Out[230]:

	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type
0	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	0
1	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	0
2	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	0
3	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	0
4	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	0
...	...	...	...	...	...	...	...	...	...
18244	17074.83	2046.96	1529.20	0.00	13498.67	13066.82	431.85	0.0	1
18245	13888.04	1191.70	3431.50	0.00	9264.84	8940.04	324.80	0.0	1
18246	13766.76	1191.92	2452.79	727.94	9394.11	9351.80	42.31	0.0	1
18247	16205.22	1527.63	2981.04	727.01	10969.54	10919.54	50.00	0.0	1
18248	17489.58	2894.77	2356.13	224.53	12014.15	11988.14	26.01	0.0	1

18249 rows × 9 columns

In [231]:
```
y=df.iloc[:,1].values
y
```

Out[231]:
```
array([1.33, 1.35, 0.93, ..., 1.87, 1.93, 1.62])
```

In [232]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, ran
```

In [233]:
```
X_train
```

Out[233]:

	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	t
3370	173825.82	49586.89	42501.52	35277.07	46460.34	23754.04	17567.37	5138.93	
2541	169118.65	88022.48	33530.37	41.17	47524.63	47493.52	31.11	0.00	
4988	564637.66	104831.87	157896.39	21607.39	280302.01	238110.78	42191.23	0.00	
14684	216330.45	26543.86	55055.64	81.03	134649.92	46628.66	88021.26	0.00	
8636	443295.30	82676.76	65334.67	47721.41	247562.46	178042.93	49942.51	19577.02	
...	...	...	...	...	...	...	...	...	
16304	170194.95	8951.42	43472.34	1596.01	116175.18	106878.45	9296.73	0.00	
79	554763.76	449311.47	30231.78	678.40	74542.11	55484.76	19010.81	46.54	
12119	2206.16	5.29	1132.48	0.00	1068.39	473.33	595.06	0.00	
14147	28707.18	1397.86	25119.52	0.00	2189.80	2186.47	3.33	0.00	
5640	102461.61	2468.78	86707.66	2546.08	10739.09	4950.13	4788.96	1000.00	

12226 rows × 9 columns

In [234]:
```
y_train
```

Out[234]:
```
array([0.95, 1.23, 0.85, ..., 1.38, 2.34, 1.39])
```

In [235]:
```
X_test
```

Out[235]:

	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags
17091	14071.97	225.39	3924.73	86.75	9835.10	9260.43	574.67
9441	91211.53	17839.96	47527.65	1.53	25842.39	25661.06	181.33
15443	21753.60	503.84	3596.40	0.00	17535.94	15555.28	1980.66
10982	29227.44	3334.06	15998.87	48.37	9846.14	8530.66	1315.48
2671	31936856.18	12680252.48	12998327.25	1143364.58	5114911.87	4342839.06	703542.10
...	...	...	...	...	...	...	...
13636	10596.57	465.92	2658.75	17.11	7454.79	6054.65	1400.14
7010	176082.45	72427.91	37438.94	93.89	66121.71	56903.06	7881.98
17054	56578.63	1762.76	16113.72	10.69	38691.46	497.52	38193.94
17615	23042.99	590.29	5224.55	0.00	17228.15	16438.54	789.61
10334	1155.28	23.52	1114.66	0.00	17.10	0.00	17.10

6023 rows × 9 columns

In [236]:
```
y_test
```

Out[236]:
```
array([2.03, 1.48, 1.75, ..., 1.51, 1.43, 1.87])
```

In [237]:
```
from sklearn.linear_model import LinearRegression
```

In [238]:
```
regressor = LinearRegression()
regressor.fit(X_train,y_train)
y_pred = regressor.predict(X_test)
df2 =  pd.DataFrame({'Actual':y_test, 'Predicted': y_pred})
df2
```

Out[238]:

	Actual	Predicted
0	2.03	1.652768
1	1.48	1.656110
2	1.75	1.653595
3	1.90	1.653645
4	1.05	0.934288
...	...	...
6018	1.95	1.652476
6019	1.42	1.168163
6020	1.51	1.649764
6021	1.43	1.653071
6022	1.87	1.652353

6023 rows × 2 columns

In [239]:
```
regressor.intercept_
```

Out[239]:
```
1.1684152801987813
```

In [240]:
```
regressor.coef_
```

Out[240]:
```
array([ 7.08836302e-06, -7.18483568e-06, -6.98433125e-06, -7.54008510e-06,
        2.07538227e-02, -2.07608852e-02, -2.07610156e-02, -2.07592414e-02,
        4.83825774e-01])
```

In [241]:
```
regressor.score(X, y)
```

Out[241]:
```
0.39797969843237535
```

In [250]:
```
rfr = RandomForestRegressor()
rfr.fit(X_train, y_train)
```

```
C:\Users\Nikita\anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245:
FutureWarning:

The default value of n_estimators will change from 10 in version 0.20 to 10
0 in 0.22.
```

Out[250]:
```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                      max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=10,
                      n_jobs=None, oob_score=False, random_state=None,
                      verbose=0, warm_start=False)
```

In [251]:
```
y_pred1 = rfr.predict(X_test)
```

In [252]:
```
df3 =  pd.DataFrame({'Actual':y_test, 'Predicted': y_pred})
df3
```

Out[252]:

	Actual	Predicted
0	2.03	1.652768
1	1.48	1.656110
2	1.75	1.653595
3	1.90	1.653645
4	1.05	0.934288
...	...	...
6018	1.95	1.652476
6019	1.42	1.168163
6020	1.51	1.649764
6021	1.43	1.653071
6022	1.87	1.652353

6023 rows × 2 columns

In [253]:
```
rfr.score(X,y)
```

Out[253]:
```
0.8798264337791556
```