In [1]:
```
# The first data set (avocado) was download from :Link to dataset: https://www.kaggle.com/neuromusic/avocado-prices.
# The second data set link :https://www.downloadexcelfiles.com/us_en/download-excel-file-list-state-capitals-united-states#.Xc4z61dKjIV
#         Columns
# index
# Date           The date of the observation
# AveragePrice   the average price of a single avocado
# Total Volume   Total number of avocados sold
# 4046           Total number of avocados with PLU 4046 sold
# 4225           Total number of avocados with PLU 4225 sold
# 4770           Total number of avocados with PLU 4770 sold
# Total Bags
# Small Bags
# Large Bags
# XLarge Bags
# type           conventional or organic
# year           the year
# region         the city or region of the observation
```

In [ ]:

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

In [2]:
```
# load a file  and print the head of our data
df1 = pd.read_csv('avocado-prices/state_capitals_us.csv',index_col=0)
df1.head()
```

Out[2]:

SNo	State	Abr.	Date of statehood	Capital	Capital since	Land area (mi²)	Most populous city?	Municipal population	Metropolitan population	Notes
1	Alabama	AL	1819.0	Montgomery	1846.0	155.4	No	205764.0	374536.0	Birmingham is the state's largest city
2	Alaska	AK	1959.0	Juneau	1906.0	2716.7	No	31275.0	NaN	Juneau is the largest capital by land area. An...
3	Arizona	AZ	1912.0	Phoenix	1889.0	474.9	Yes	1445632.0	4192887.0	Phoenix is the most populous U.S. state capita...
4	Arkansas	AR	1836.0	Little Rock	1821.0	116.2	Yes	193524.0	877091.0	NaN
5	California	CA	1850.0	Sacramento	1854.0	97.2	No	466488.0	2527123.0	The Supreme Court of California is headquarter...

In [3]:
```
# load a file  and print the head of our data
df2 = pd.read_csv('avocado-prices/avocado.csv',index_col=0)
df2.head()
```

Out[3]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Albany
1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Albany
2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	2015	Albany
3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	conventional	2015	Albany
4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	conventional	2015	Albany

In [4]:
```
# load both df1 and df2 merge using sql and print the head of our data
df = pd.read_csv('avocado-prices/merge_df.csv',index_col=0)
df.head()
```

Out[4]:

field1	Date	AveragePrice	TotalVolume	4046	4225	4770	TotalBags	SmallBags	LargeBags	XLargeBags	...	State	Abr.	I
0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	...	New York	NY	
1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	...	New York	NY	
2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	...	New York	NY	
3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	...	New York	NY	
4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	...	New York	NY	

5 rows × 24 columns

In [5]:
```
# Print the shape of our df
print(df.shape)
```

```
(18249, 24)
```

In [6]:
```
#drop columns( Metropolitan, population Notes and Capital since)
df = df.drop(columns=['Metropolitanpopulation', 'Notes', 'Capitalsince'])
df.head()
```

Out[6]:

field1	Date	AveragePrice	TotalVolume	4046	4225	4770	TotalBags	SmallBags	LargeBags	XLargeBags	...	year	region
0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	...	2015	Albany
1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	...	2015	Albany
2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	...	2015	Albany
3	2015-12-06	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677.40	133.76	0.0	...	2015	Albany
4	2015-11-29	1.28	51039.60	941.48	43838.39	75.78	6183.95	5986.26	197.69	0.0	...	2015	Albany

5 rows × 21 columns

In [7]:
```
# convert date to datetime
df['Date']=pd.to_datetime(df.Date)
```

In [8]:
```
# check the table to see if any nna and the check all data types
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18249 entries, 0 to 11
Data columns (total 21 columns):
Date                   18249 non-null datetime64[ns]
AveragePrice           18249 non-null float64
TotalVolume            18249 non-null float64
4046                   18249 non-null float64
4225                   18249 non-null float64
4770                   18249 non-null float64
TotalBags              18249 non-null float64
SmallBags              18249 non-null float64
LargeBags              18249 non-null float64
XLargeBags             18249 non-null float64
type                   18249 non-null object
year                   18249 non-null int64
region                 18249 non-null object
SNo                    3042 non-null float64
State                  3042 non-null object
Abr.                   3042 non-null object
Dateofstatehood        3042 non-null float64
Capital                3042 non-null object
Landarea(mi²)          3042 non-null float64
Mostpopulouscity?      3042 non-null object
Municipalpopulation    3042 non-null float64
dtypes: datetime64[ns](1), float64(13), int64(1), object(6)
memory usage: 3.1+ MB
```

In [9]:
```
#inspect our df by printing the head() and the tail()
print(df.head())
print(df.tail())
```

```
              Date  AveragePrice  TotalVolume     4046       4225    4770  \
field1
0       2015-12-27          1.33     64236.62  1036.74   54454.85   48.16
1       2015-12-20          1.35     54876.98   674.28   44638.81   58.33
2       2015-12-13          0.93    118220.22   794.70  109149.67  130.50
3       2015-12-06          1.08     78992.15  1132.00   71976.41   72.58
4       2015-11-29          1.28     51039.60   941.48   43838.39   75.78

        TotalBags  SmallBags  LargeBags  XLargeBags         ...          year  \
field1                                                      ...
0         8696.87    8603.62      93.25         0.0         ...          2015
1         9505.56    9408.07      97.49         0.0         ...          2015
2         8145.35    8042.21     103.14         0.0         ...          2015
3         5811.16    5677.40     133.76         0.0         ...          2015
4         6183.95    5986.26     197.69         0.0         ...          2015

        region   SNo     State Abr. Dateofstatehood  Capital Landarea(mi²)  \
field1
0       Albany  32.0  New York   NY          1788.0   Albany          21.4
1       Albany  32.0  New York   NY          1788.0   Albany          21.4
2       Albany  32.0  New York   NY          1788.0   Albany          21.4
3       Albany  32.0  New York   NY          1788.0   Albany          21.4
4       Albany  32.0  New York   NY          1788.0   Albany          21.4

        Mostpopulouscity? Municipalpopulation
field1
0                      No             97856.0
1                      No             97856.0
2                      No             97856.0
3                      No             97856.0
4                      No             97856.0

[5 rows x 21 columns]
              Date  AveragePrice  TotalVolume     4046     4225    4770  \
field1
7       2018-02-04          1.63     17074.83  2046.96  1529.20    0.00
8       2018-01-28          1.71     13888.04  1191.70  3431.50    0.00
9       2018-01-21          1.87     13766.76  1191.92  2452.79  727.94
10      2018-01-14          1.93     16205.22  1527.63  2981.04  727.01
11      2018-01-07          1.62     17489.58  2894.77  2356.13  224.53

        TotalBags  SmallBags  LargeBags  XLargeBags         ...          year  \
field1                                                      ...
```

```
7         13498.67   13066.82     431.85        0.0        ...         2018
8          9264.84    8940.04     324.80        0.0        ...         2018
9          9394.11    9351.80      42.31        0.0        ...         2018
10        10969.54   10919.54      50.00        0.0        ...         2018
11        12014.15   11988.14      26.01        0.0        ...         2018

                  region SNo  State Abr. Dateofstatehood  Capital  \
field1
7       WestTexNewMexico NaN    NaN  NaN             NaN      NaN
8       WestTexNewMexico NaN    NaN  NaN             NaN      NaN
9       WestTexNewMexico NaN    NaN  NaN             NaN      NaN
10      WestTexNewMexico NaN    NaN  NaN             NaN      NaN
11      WestTexNewMexico NaN    NaN  NaN             NaN      NaN

       Landarea(mi²)  Mostpopulouscity? Municipalpopulation
field1
7                NaN                NaN                 NaN
8                NaN                NaN                 NaN
9                NaN                NaN                 NaN
10               NaN                NaN                 NaN
11               NaN                NaN                 NaN

[5 rows x 21 columns]
```

In [10]:

```
# use the describe() method to calculate summary statistics of your data
df.describe()
```

Out[10]:

	AveragePrice	TotalVolume	4046	4225	4770	TotalBags	SmallBags	LargeBags	XLargeBag
count	18249.000000	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	18249.0000(
mean	1.405978	8.506440e+05	2.930084e+05	2.951546e+05	2.283974e+04	2.396392e+05	1.821947e+05	5.433809e+04	3106.4265(
std	0.402677	3.453545e+06	1.264989e+06	1.204120e+06	1.074641e+05	9.862424e+05	7.461785e+05	2.439660e+05	17692.8946
min	0.440000	8.456000e+01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.0000(
25%	1.100000	1.083858e+04	8.540700e+02	3.008780e+03	0.000000e+00	5.088640e+03	2.849420e+03	1.274700e+02	0.0000(
50%	1.370000	1.073768e+05	8.645300e+03	2.906102e+04	1.849900e+02	3.974383e+04	2.636282e+04	2.647710e+03	0.0000(
75%	1.660000	4.329623e+05	1.110202e+05	1.502069e+05	6.243420e+03	1.107834e+05	8.333767e+04	2.202925e+04	132.5000(
max	3.250000	6.250565e+07	2.274362e+07	2.047057e+07	2.546439e+06	1.937313e+07	1.338459e+07	5.719097e+06	551693.6500(

In [11]:
```
# There are two types of avocado, firstly we analyse the conventional ones:
df_conventional = df[df.type == 'conventional']
```

In [12]:
```
# count type of conventional avocado by region
print (len(df.groupby('region')))
df_conventional['region'].value_counts()
```

54

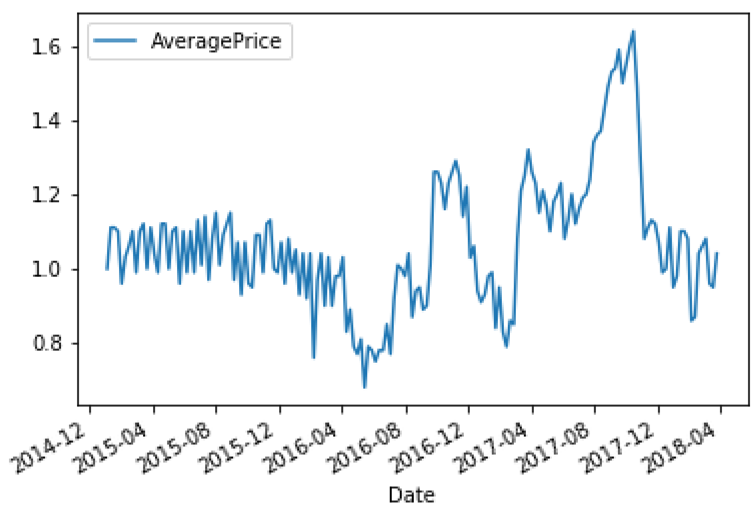
Out[12]:
```
GrandRapids            169
Detroit                169
Spokane                169
Houston                169
Boise                  169
Albany                 169
HartfordSpringfield    169
GreatLakes             169
Pittsburgh             169
LosAngeles             169
NorthernNewEngland     169
Tampa                  169
NewYork                169
Southeast              169
Northeast              169
SouthCarolina          169
Chicago                169
Roanoke                169
WestTexNewMexico       169
SanDiego               169
HarrisburgScranton     169
Portland               169
NewOrleansMobile       169
Boston                 169
Syracuse               169
Columbus               169
Seattle                169
StLouis                169
TotalUS                169
RaleighGreensboro      169
Philadelphia           169
PhoenixTucson          169
Plains                 169
BuffaloRochester       169
SouthCentral           169
Jacksonville           169
Louisville             169
RichmondNorfolk        169
SanFrancisco           169
Charlotte              169
DallasFtWorth          169
West                   169
Denver                 169
```

```
Midsouth               169
LasVegas               169
Sacramento             169
MiamiFtLauderdale      169
California             169
Indianapolis           169
BaltimoreWashington    169
Atlanta                169
Orlando                169
CincinnatiDayton       169
Nashville              169
Name: region, dtype: int64
```

In [13]:
```
# plot the AveragePrice of conventional avocado
regions_conventional = df_conventional.groupby(df_conventional.region)
date_conventional = regions_conventional.get_group('Atlanta')[['Date', 'AveragePrice']].reset_index(drop=True
)
```

In [14]:
```
date_conventional.plot(x='Date', y='AveragePrice', kind="line")
```

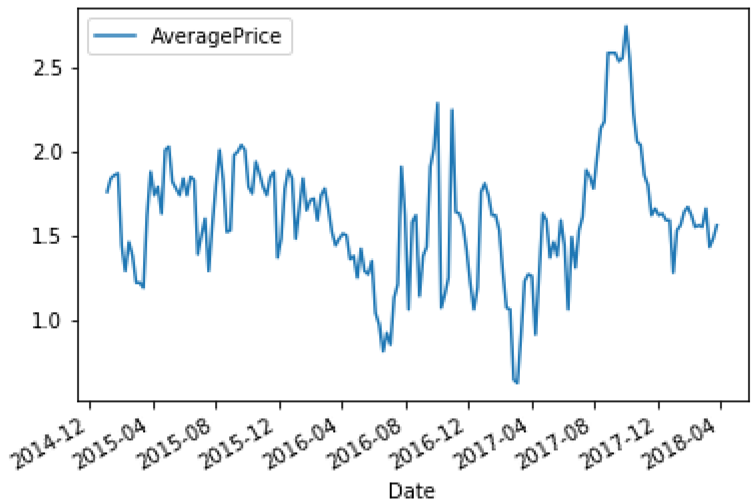
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x19a87b9e2e8>



In [15]:

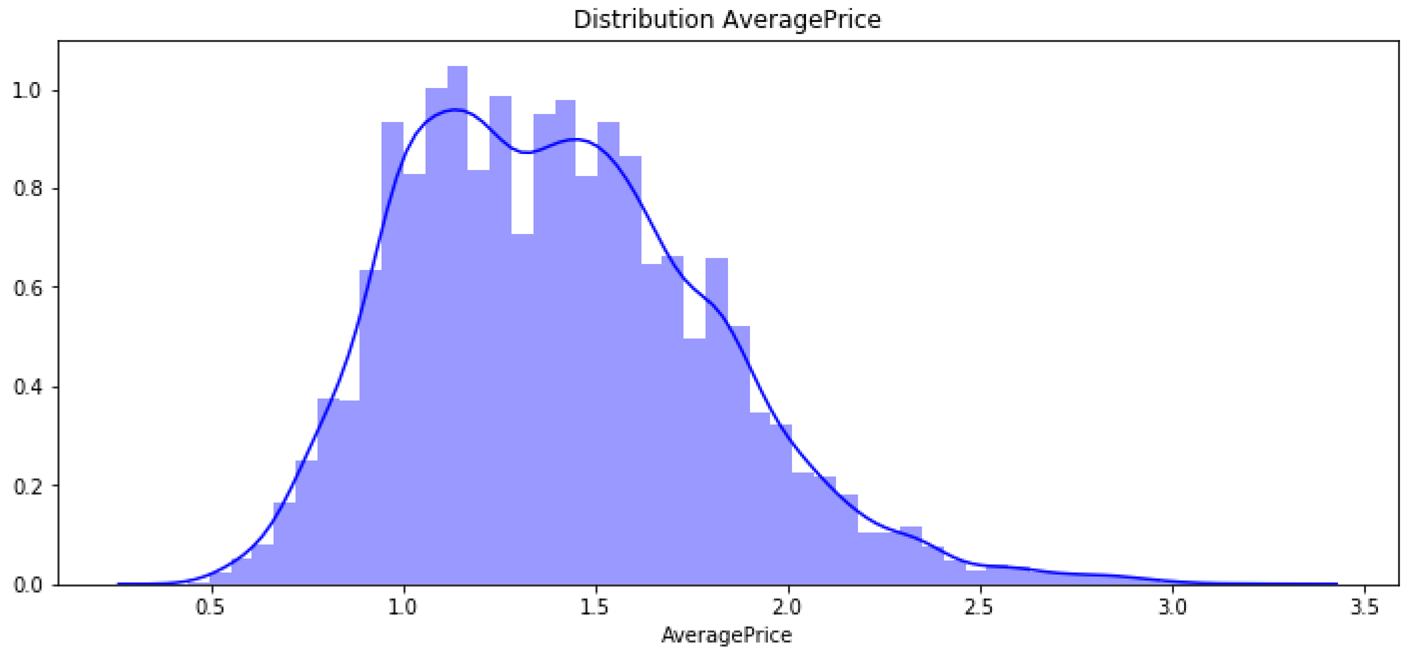
```python
# # plot the AveragePrice of organic avocado
df_organic = df[df.type == 'organic']
regions_organic = df_organic.groupby(df_organic.region)
date_organic = regions_organic.get_group('Atlanta')[['Date', 'AveragePrice']].reset_index(drop=True)
date_organic.plot(x='Date', y='AveragePrice', kind="line")
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x19a8824ae10>

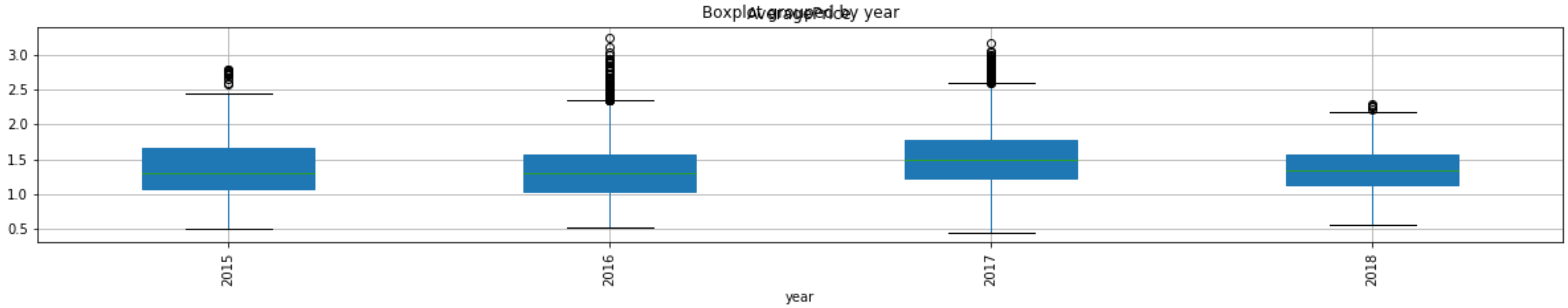


In [16]:
```
# plot the Distribution of AveragePrice
plt.figure(figsize=(12,5))
plt.title("Distribution AveragePrice")
ax = sns.distplot(df["AveragePrice"], color = 'b')
```

```
C:\Users\fcheb\Anaconda3\lib\site-packages\scipy\stats\stats.py:1713: FutureWarning: Using a non-tuple sequen
ce for multidimensional indexing is deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future th
is will be interpreted as an array index, `arr[np.array(seq)]`, which will result either in an error or a dif
ferent result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

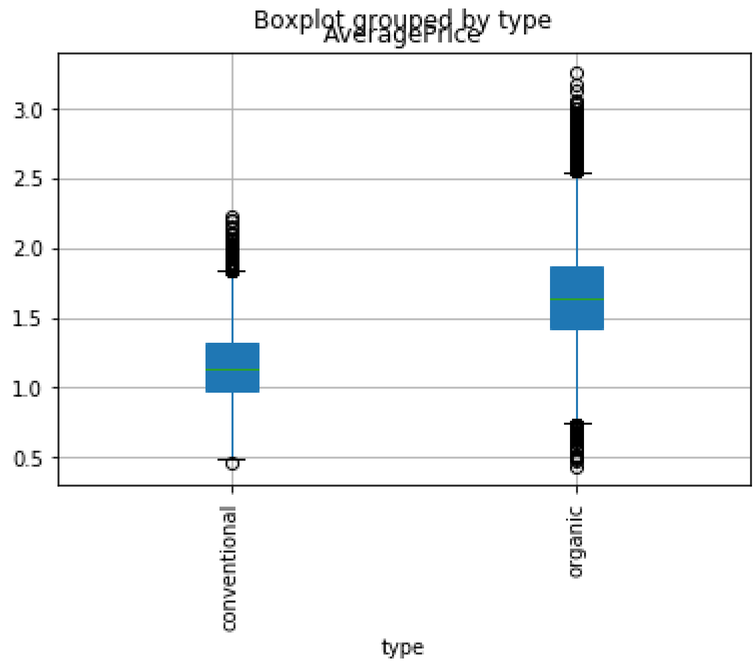


In [17]:
```
# Create the boxplot to show average price by year
df.boxplot(column='AveragePrice', by='year', figsize=(20,3), rot=90, patch_artist = True)
# Display the plot
plt.show()
```



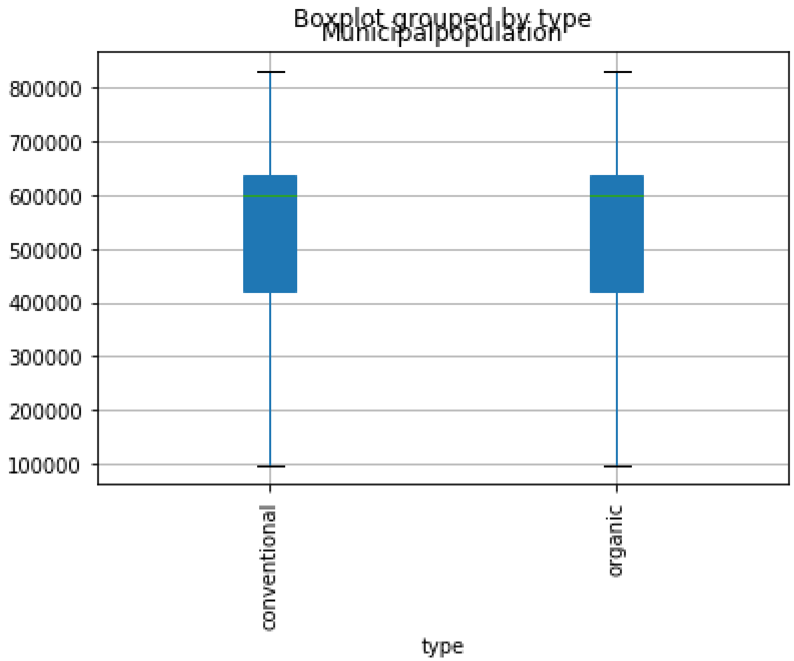
In [18]:
```
# Create the boxplot to show average price by type
df.boxplot(column='AveragePrice', by='type',  rot=90, patch_artist = True)

# Display the plot
plt.show()
```



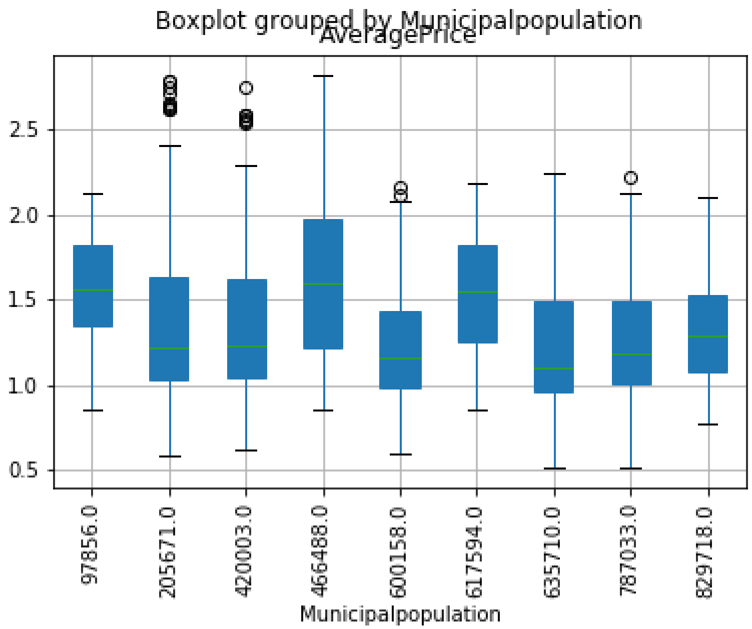
In [19]:
```
# Create the boxplot to show population type
df.boxplot(column='Municipalpopulation', by='type',  rot=90, patch_artist = True)

# Display the plot
plt.show()
```



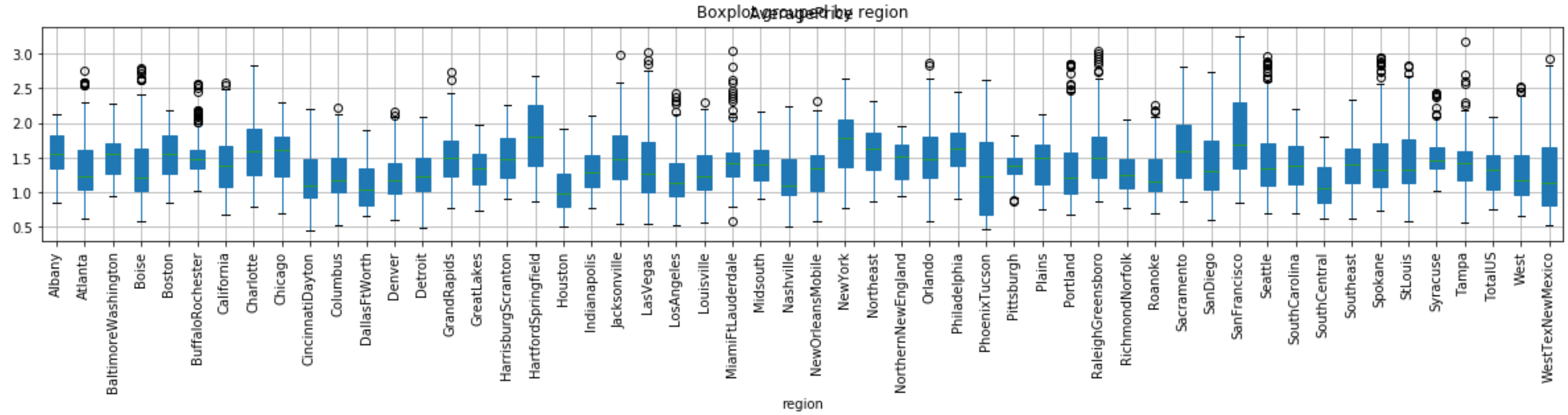
In [20]:
```
# Create the boxplot to compare average price and  Municipal population
df.boxplot(column='AveragePrice', by='Municipalpopulation',  rot=90, patch_artist = True)

# Display the plot
plt.show()
```



In [21]:
```
# Create the boxplot to show average price by region
df.boxplot(column='AveragePrice', by='region', figsize=(20,3), rot=90, patch_artist = True)

# Display the plot
plt.show()
```



In [22]:
```
# print the 10 1st region where the AveragePrice <= min(AveragePrice)
# top_place_to_leave = df[df.AveragePrice < 1.40]
top_place_to_leave = df.sort_values(["AveragePrice"], ascending=True)
top_place_to_leave.head()
```

Out[22]:

field1	Date	AveragePrice	TotalVolume	4046	4225	4770	TotalBags	SmallBags	LargeBags	XLargeBags	...	year	
43	2017-03-05	0.44	64057.04	223.84	4748.88	0.00	59084.32	638.68	58445.64	0.00	...	2017	C
47	2017-02-05	0.46	2200550.27	1200632.86	531226.65	18324.93	450365.83	113752.17	330583.10	6030.56	...	2017	
43	2017-03-05	0.48	50890.73	717.57	4138.84	0.00	46034.32	1385.06	44649.26	0.00	...	2017	
44	2017-02-26	0.49	44024.03	252.79	4472.68	0.00	39298.56	600.00	38698.56	0.00	...	2017	C
0	2015-12-27	0.49	1137707.43	738314.80	286858.37	11642.46	100891.80	70749.02	30142.78	0.00	...	2015	

5 rows × 21 columns

In [23]:
```
# Best 10 place to leave
print(top_place_to_leave['region'].unique())
```

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

Best top place to leave and continue have enough avocados base on our analysis in 2017 was:

CincinnatiDayton

In [24]:
```
# Filter the DataFrame down only to those columns to chart
top_place_to_leave = top_place_to_leave[["region","AveragePrice"]]

# Set the index to be "State" so they will be used as labels
top_place_to_leave = top_place_to_leave.set_index("region").head(30)

top_place_to_leave
```

Out[24]:

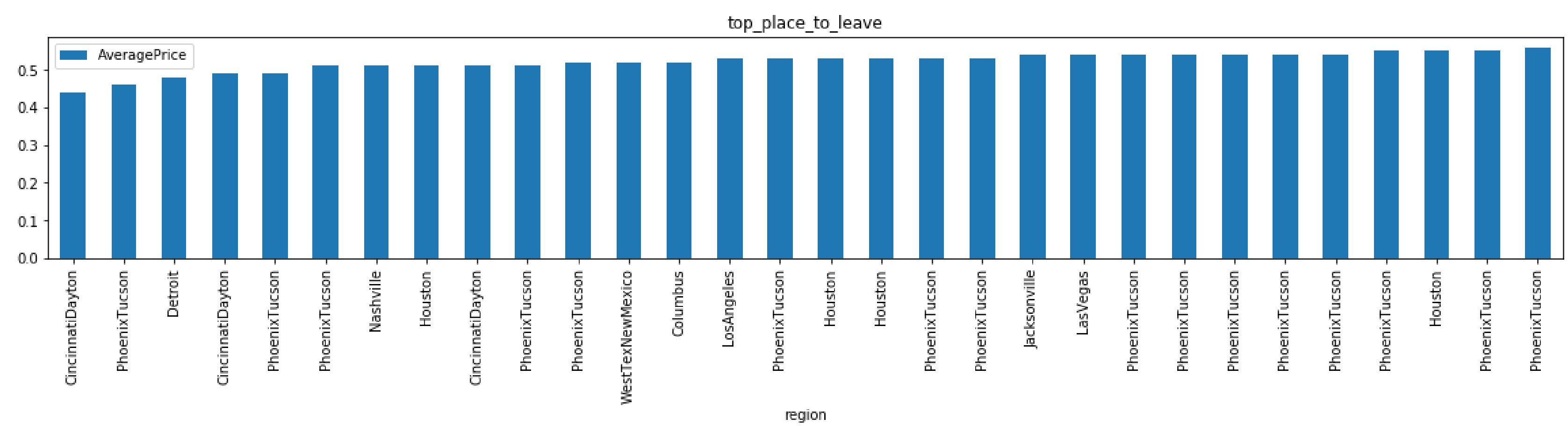
region	AveragePrice
CincinnatiDayton	0.44
PhoenixTucson	0.46
Detroit	0.48
CincinnatiDayton	0.49
PhoenixTucson	0.49
PhoenixTucson	0.51
Nashville	0.51
Houston	0.51
CincinnatiDayton	0.51
PhoenixTucson	0.51
PhoenixTucson	0.52
WestTexNewMexico	0.52
Columbus	0.52
LosAngeles	0.53
PhoenixTucson	0.53
Houston	0.53
Houston	0.53
PhoenixTucson	0.53
PhoenixTucson	0.53
Jacksonville	0.54
LasVegas	0.54
PhoenixTucson	0.54
PhoenixTucson	0.54
PhoenixTucson	0.54
PhoenixTucson	0.54

	AveragePrice
region	
PhoenixTucson	0.54
PhoenixTucson	0.55
Houston	0.55
PhoenixTucson	0.55
PhoenixTucson	0.56

In [25]:
```python
# Use DataFrame.plot() in order to create a bar chart of the data
top_place_to_leave.plot(kind="bar", figsize=(20,3))

# Set a title for the chart
plt.title("top_place_to_leave")

plt.show()
plt.tight_layout()
```




```
<Figure size 432x288 with 0 Axes>
```

In [ ]:
```python
```