Column Name	Description
Unnamed	SNo.
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	Total Number of Bags sold
Small Bags	Total Number of Small Bags sold
Large Bags	Total Number of Large Bags sold
XLarge Bags	Total Number of XLarge Bags sold
type	Organic or Conventional
year	The year of observation
region	the city or region of the observation

In [1]:

```
import numpy as np
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:

```
data = pd.read_csv(r"D:\Data science studies\NARESH It\date wise notes\extracted files\22nd august\RESUME PROJECT -- PRICE PREDICTION\avocado.csv", na_values=["?","#"]) #,header=None)
```

In [3]:

```
data.head(3)
```

Out[3]:

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	ye
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	20
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	20
2	2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	conventional	20

In [4]:

```
print('the dataset having ',data.shape[0],' no. of rows and ',data.shape[1],' no. of columns')
```

```
the dataset having  18249  no. of rows and  14  no. of columns
```

In [5]:

```
data.shape
```

Out[5]:

```
(18249, 14)
```

In [6]:

```
data.columns
```

Out[6]:

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

In [7]:

```
len(data.columns)
```

Out[7]:

```
14
```

In [8]:

```
data.rename(columns={'Unnamed: 0':'Sl.No'},inplace=True)
```

In [9]:

```
data.head(2)
```

Out[9]:

	Sl.No	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	regio
0	0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	conventional	2015	Alban
1	1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	conventional	2015	Alban

In [10]:

```
data.isnull().sum()
```

Out[10]:

```
Sl.No           0
Date            0
AveragePrice    0
Total Volume    0
4046            0
4225            0
4770            0
Total Bags      0
Small Bags      0
Large Bags      0
XLarge Bags     0
type            0
year            0
region          0
dtype: int64
```

In [11]:

```
data.dtypes
```

Out[11]:

```
Sl.No             int64
Date             object
AveragePrice    float64
Total Volume    float64
4046            float64
4225            float64
4770            float64
Total Bags      float64
Small Bags      float64
Large Bags      float64
XLarge Bags     float64
type             object
year              int64
region           object
dtype: object
```

In [12]:

```
data.describe()
```

Out[12]:

	Sl.No	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	
count	18249.000000	18249.000000	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04	1.8
mean	24.232232	1.405978	8.506440e+05	2.930084e+05	2.951546e+05	2.283974e+04	2.396392e+05	1.821947e+05	5.4
std	15.481045	0.402677	3.453545e+06	1.264989e+06	1.204120e+06	1.074641e+05	9.862424e+05	7.461785e+05	2.4
min	0.000000	0.440000	8.456000e+01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.0
25%	10.000000	1.100000	1.083858e+04	8.540700e+02	3.008780e+03	0.000000e+00	5.088640e+03	2.849420e+03	1.2

	Sl.No	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	
50%	24.000000	1.370000	1.073768e+05	8.645300e+03	2.906102e+04	1.849900e+02	3.974383e+04	2.636282e+04	2.6
75%	38.000000	1.660000	4.329623e+05	1.110202e+05	1.502069e+05	6.243420e+03	1.107834e+05	8.333767e+04	2.2
max	52.000000	3.250000	6.250565e+07	2.274362e+07	2.047057e+07	2.546439e+06	1.937313e+07	1.338459e+07	5.7

In [13]:

```
data.drop(['Sl.No'],axis = 1 , inplace=True)
```

In [14]:

```
data.head(2)
```

Out[14]:

	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

In [15]:

```
len(data.columns)
```

Out[15]:

```
13
```

In [16]:

```
data.nunique()
```

Out[16]:

```
Date              169
AveragePrice      259
Total Volume    18237
4046            17702
4225            18103
4770            12071
Total Bags      18097
Small Bags      17321
Large Bags      15082
XLarge Bags      5588
type                2
year                4
region             54
dtype: int64
```

In [17]:

```
data.type.unique()
```

Out[17]:

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

In [18]:

```
data.region.unique()
```

Out[18]:

```
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 [19]:

```
data=pd.get_dummies(data,columns=['type'],drop_first=True)
```

In [20]:

```
data.head(3)
```

Out[20]:

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	year	region	type_organic
0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	2015	Albany	0
1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	2015	Albany	0
2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	2015	Albany	0

In [21]:

```
data.type_organic.value_counts()
```

Out[21]:

```
0    9126
1    9123
Name: type_organic, dtype: int64
```

In [22]:

```
from sklearn.preprocessing import LabelEncoder
```

In [23]:

```
le= LabelEncoder()
```

In [24]:

```
data.region = le.fit_transform(data['region'])
```

data['region'].value_counts()

In [25]:

```
data.head(3)
```

Out[25]:

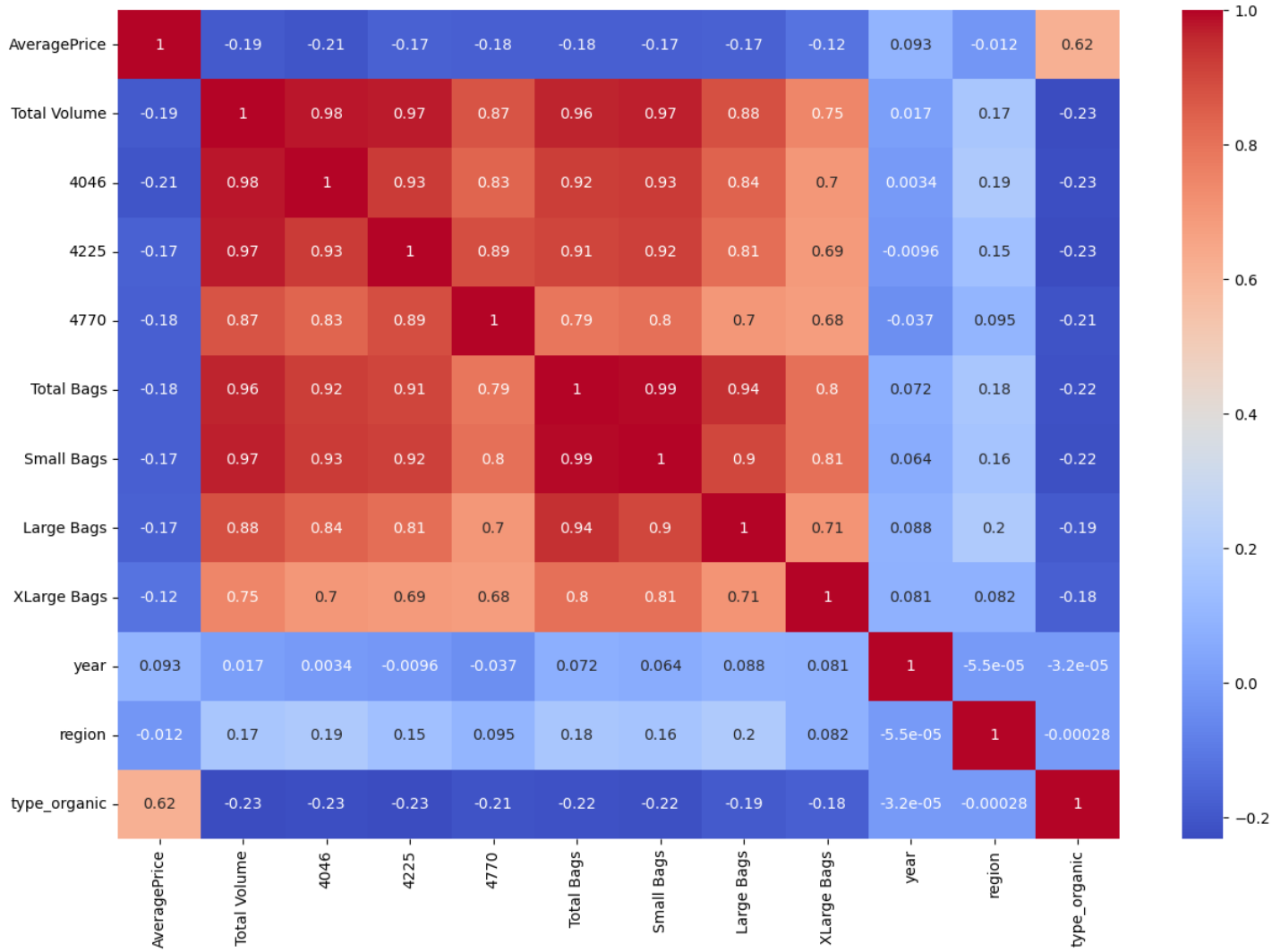
	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	year	region	type_organic
0	2015-12-27	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603.62	93.25	0.0	2015	0	0
1	2015-12-20	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408.07	97.49	0.0	2015	0	0
2	2015-12-13	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042.21	103.14	0.0	2015	0	0

In [26]:

```
cor = data.corr()
```

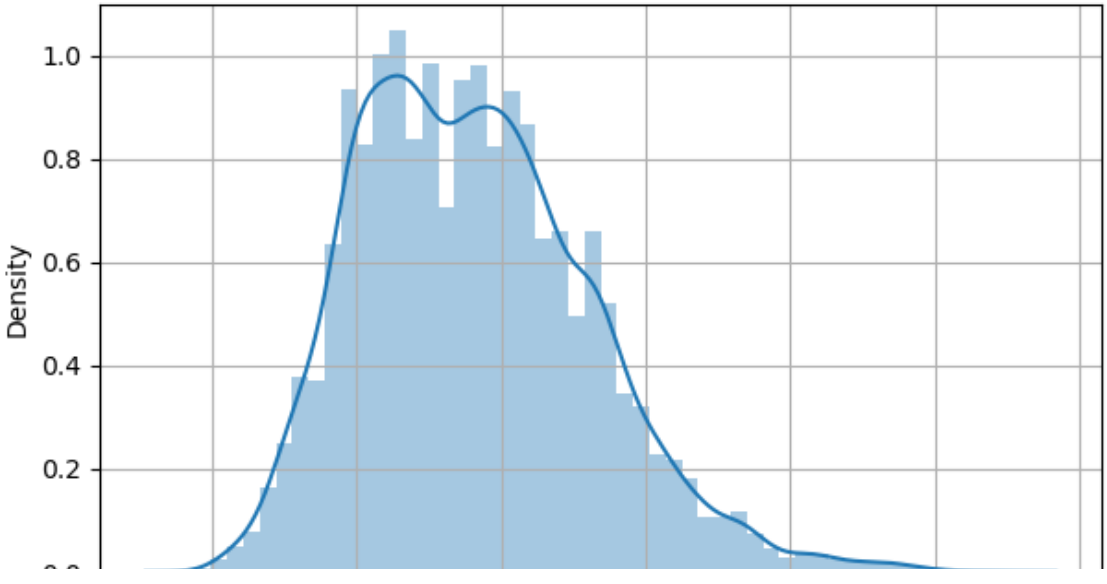
In [27]:

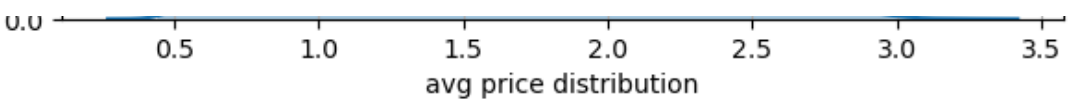
```
plt.figure(figsize=(15,10))
sns.heatmap(cor,annot=True,cmap='coolwarm')
plt.show()
```



In [28]:

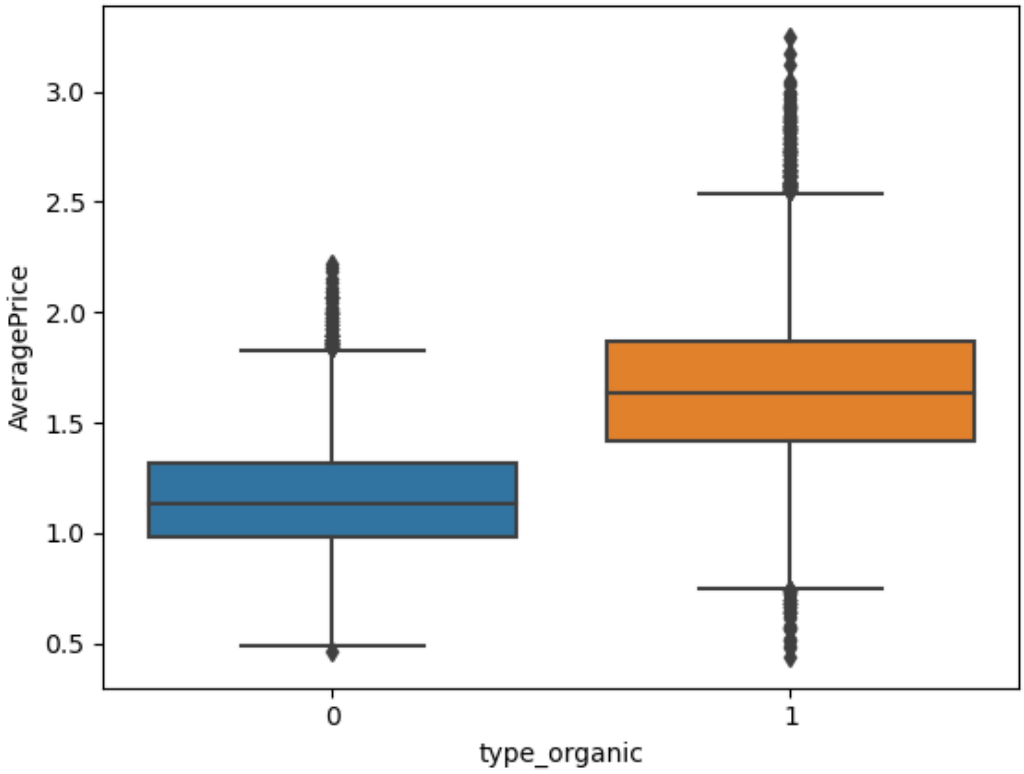
```
plt.figure(figsize=(7,4))
sns.distplot(data['AveragePrice'],axlabel='avg price distribution')
plt.grid(True)
plt.show()
```





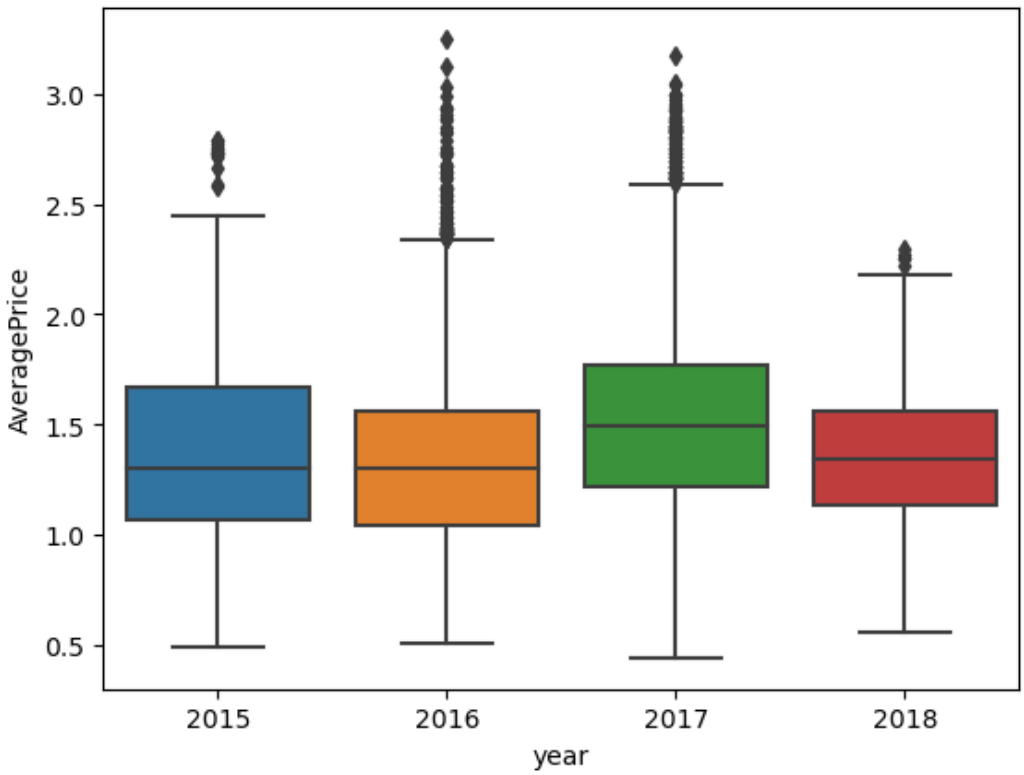
In [29]:

```
sns.boxplot(x='type_organic',y='AveragePrice',data=data)
plt.show()
```



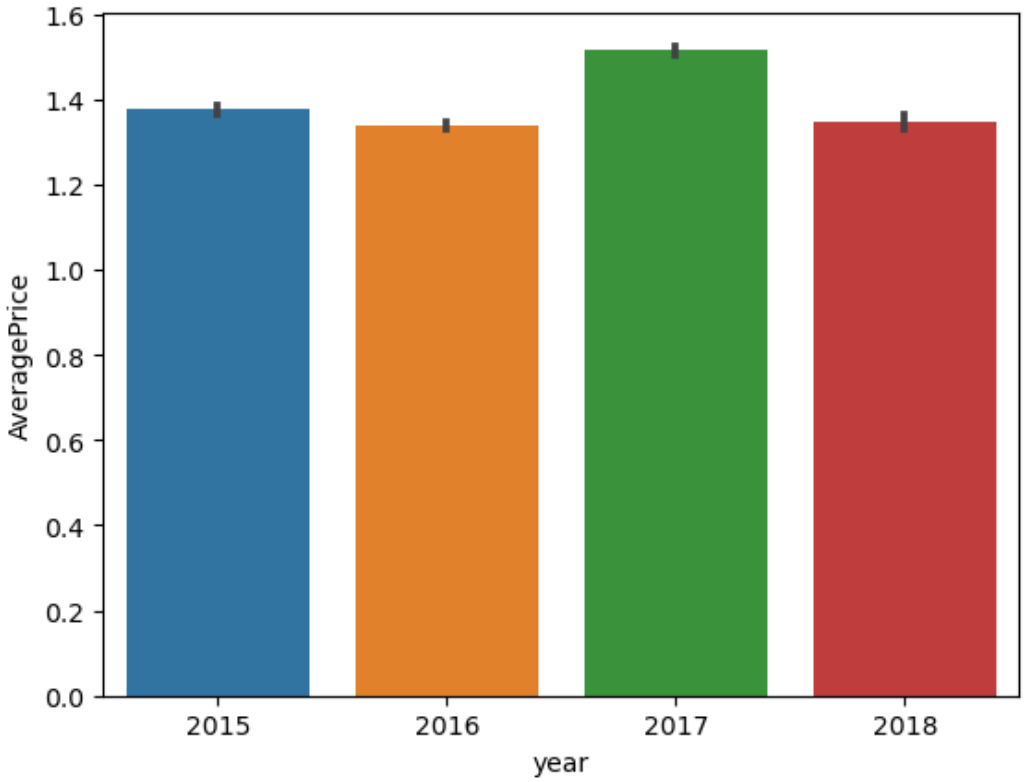
In [30]:

```
sns.boxplot(x='year',y='AveragePrice',data=data)
plt.show()
```



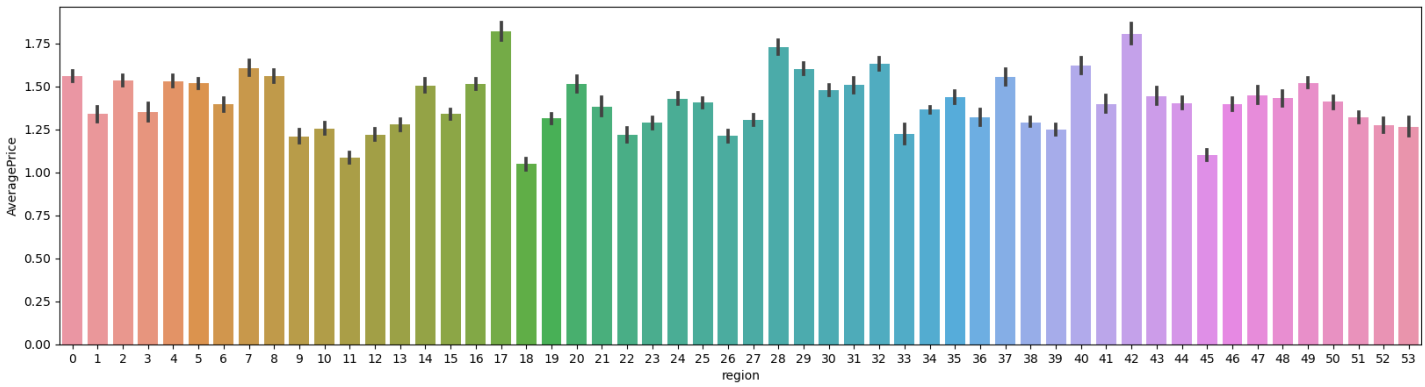
In [31]:

```
sns.barplot(x='year',y='AveragePrice',data=data)
plt.show()
```



In [32]:

```
plt.figure(figsize=(20,5))
sns.barplot(x=data['region'],y=data['AveragePrice'])
plt.show()
```



In [33]:

```
data1=data.copy()
```

In [34]:

```
X=data1[['Total Volume','Total Bags','year','region','type_organic']]
```

In [35]:

```
X
```

Out[35]:

	Total Volume	Total Bags	year	region	type_organic
0	64236.62	8696.87	2015	0	0
1	54876.98	9505.56	2015	0	0
2	118220.22	8145.35	2015	0	0
3	78992.15	5811.16	2015	0	0
4	51039.60	6183.95	2015	0	0

...	Total Volume	Total Bags	year	region	type_organic
18244	17074.83	13498.67	2018	53	1
18245	13888.04	9264.84	2018	53	1
18246	13766.76	9394.11	2018	53	1
18247	16205.22	10969.54	2018	53	1
18248	17489.58	12014.15	2018	53	1

18249 rows × 5 columns

In [36]:

```
y=data1['AveragePrice']
y
```

Out[36]:

```
0        1.33
1        1.35
2        0.93
3        1.08
4        1.28
         ...
18244    1.63
18245    1.71
18246    1.87
18247    1.93
18248    1.62
Name: AveragePrice, Length: 18249, dtype: float64
```

In [37]:

```
from sklearn.model_selection import train_test_split
```

In [38]:

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=123)
```

In [39]:

```
print(X_train.shape)
print('------------------')
print(X_test.shape)
print('------------------')
print(y_train.shape)
print('------------------')
print(y_test.shape)
```

```
(14599, 5)
------------------
(3650, 5)
------------------
(14599,)
------------------
(3650,)
```

In [40]:

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
sc.fit_transform(X_train)
sc.transform(X_test)
```

Out[40]:

```
array([[ 0.00512421,  0.25194327, -0.15924268,  1.06664235, -1.00116514],
       [-0.24535472, -0.23551225, -1.22419178,  0.23286343,  0.99883621],
       [-0.20552714, -0.17536478,  0.90570642, -0.72918917,  0.99883621],
       ...,
       [-0.21713739, -0.1705986 ,  0.90570642,  1.19491603,  0.99883621],
```

```
       [-0.21392592, -0.17024136, -0.15924268, -0.28023129,  0.99883621],
       [-0.09806036, -0.17608241, -1.22419178,  0.61768447, -1.00116514]])
```

Linear regression

In [41]:

```
from sklearn.linear_model import LinearRegression

lr = LinearRegression()
lr.fit(X_train,y_train)

y_pred = lr.predict(X_test)


bias = round(lr.score(X_train,y_train),2)
print('bias is = ',bias * 100 , '%')

variance = round(lr.score(X_test,y_test),2)
print('variance is = ', variance* 100 , '%')
```

```
bias is =  39.0 %
variance is =  40.0 %
```

In [ ]:

```

```

In [42]:

```
from sklearn.metrics import mean_squared_error,r2_score

mse = mean_squared_error(y_pred,y_test)

r2 = r2_score(y_pred,y_test)


print(mse)
print(r2)
```

```
0.0974761744764927
-0.537007012144475
```

Knn regression

In [43]:

```
from sklearn.neighbors import KNeighborsRegressor

knn=KNeighborsRegressor(n_neighbors=10)
knn.fit(X_train,y_train)

y_pred_knn = knn.predict(X_test)


bias = round(knn.score(X_train,y_train),2)
print('bias is = ',bias * 100 , '%')

variance = round(knn.score(X_test,y_test),2)
print('variance is = ',variance* 100 , '%')
```

```
bias is =  53.0 %
variance is =  44.0 %
```

In [44]:

```
from sklearn.metrics import mean_squared_error,r2_score
```

```
mse = mean_squared_error(y_pred_knn,y_test)

r2 = r2_score(y_pred_knn,y_test)


print(mse)
print(r2)
```

```
0.09099112712328768
-0.10518678832339123
```

In [ ]:

```

```

SVR

In [45]:

```
from sklearn.svm import SVR
svr = SVR()
svr.fit(X_train, y_train)

y_pred_svr = svr.predict(X_test)

bias = round(svr.score(X_train,y_train),2)
print('bias is = ',bias * 100 , '%')

variance = round(svr.score(X_test,y_test),2)
print( 'variance is = ',variance* 100 , '%')
```

```
bias is =  32.0 %
variance is =  32.0 %
```

In [ ]:

```

```

In [46]:

```
from sklearn.metrics import mean_squared_error,r2_score

mse = mean_squared_error(y_pred_svr,y_test)

r2 = r2_score(y_pred_svr,y_test)


print(mse)
print(r2)
```

```
0.11031844727769904
-1.280415566129399
```

DT Regressor

In [47]:

```
from sklearn.tree import DecisionTreeRegressor

dtr=DecisionTreeRegressor()
dtr.fit(X_train,y_train)

y_pred_dt = dtr.predict(X_test)


bias = round(dtr.score(X_train,y_train),2)
print('bias is = ',bias * 100 , '%')
```

```
variance = round(dtr.score(X_test,y_test),2)
print( 'variance is = ',variance* 100 , '%')
```

```
bias is =  100.0 %
variance is =  63.0 %
```

In [48]:

```
from sklearn.metrics import mean_squared_error,r2_score

mse = mean_squared_error(y_pred_dt,y_test)

r2 = r2_score(y_pred_dt,y_test)


print(mse)
print(r2)
```

```
0.05995438356164383
0.6329623695033476
```

Random forest

In [49]:

```
from sklearn.ensemble import RandomForestRegressor

rf=DecisionTreeRegressor()
rf.fit(X_train,y_train)

y_pred_rf = rf.predict(X_test)


bias = round(rf.score(X_train,y_train),2)
print('bias is = ',bias * 100 , '%')

variance = round(rf.score(X_test,y_test),2)
print( 'variance is = ',variance* 100 , '%')
```

```
bias is =  100.0 %
variance is =  63.0 %
```

In [50]:

```
from sklearn.metrics import mean_squared_error,r2_score

mse = mean_squared_error(y_pred_rf,y_test)

r2 = r2_score(y_pred_rf,y_test)


print(mse)
print(r2)
```

```
0.06059673972602739
0.6277387091444329
```

In [ ]:

```

```