PREDICTING FUTURE PRODUCT PRICES USING FACEBOOK PROPHET

IMPORT LIBRARIES AND DATASET

```
#conda install -c conda-forge fbprophet
```

```
Note: you may need to restart the kernel to use updated packages.
usage: conda-script.py [-h] [-V] command ...
conda-script.py: error: unrecognized arguments: fbprophet
```

```
# import libraries
import pandas as pd # Import Pandas for data manipulation using dataframes
import numpy as np # Import Numpy for data statistical analysis
import matplotlib.pyplot as plt # Import matplotlib for data visualisation
import random
import seaborn as sns
from fbprophet import Prophet
```

```
# dataframes creation for both training and testing datasets
avocado_df = pd.read_csv("avocado.csv")
```

- Date: The date of the observation
- AveragePrice: the average price of a single avocado
- type: conventional or organic
- year: the year
- Region: the city or region of the observation
- 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

```
# Let's view the head of the training dataset
avocado_df.head()
```

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Sm B
0	0	12/27/2015	1.33	64236.62	1036.74	54454.85	48.16	8696.87	8603
1	1	12/20/2015	1.35	54876.98	674.28	44638.81	58.33	9505.56	9408
2	2	12/13/2015	0.93	118220.22	794.70	109149.67	130.50	8145.35	8042
3	3	12/6/2015	1.08	78992.15	1132.00	71976.41	72.58	5811.16	5677

```
# Let's view the last elements in the training dataset
avocado_df.tail()
```

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	
18244	7	2/4/2018	1.63	17074.83	2046.96	1529.20	0.00	13498.67	130
18245	8	1/28/2018	1.71	13888.04	1191.70	3431.50	0.00	9264.84	89
18246	9	1/21/2018	1.87	13766.76	1191.92	2452.79	727.94	9394.11	93
18247	10	1/14/2018	1.93	16205.22	1527.63	2981.04	727.01	10969.54	109
18248	11	1/7/2018	1.62	17489.58	2894.77	2356.13	224.53	12014.15	119

```
avocado_df.describe()
```

	Unnamed: 0	AveragePrice	Total Volume	4046	4225	4770
count	18249.000000	18249.000000	1.824900e+04	1.824900e+04	1.824900e+04	1.824900e+04
mean	24.232232	1.405978	8.506440e+05	2.930084e+05	2.951546e+05	2.283974e+04
std	15.481045	0.402677	3.453545e+06	1.264989e+06	1.204120e+06	1.074641e+05
min	0.000000	0.440000	8.456000e+01	0.000000e+00	0.000000e+00	0.000000e+00
25%	10.000000	1.100000	1.083858e+04	8.540700e+02	3.008780e+03	0.000000e+00
50%	24.000000	1.370000	1.073768e+05	8.645300e+03	2.906102e+04	1.849900e+02
75%	38.000000	1.660000	4.329623e+05	1.110202e+05	1.502069e+05	6.243420e+03
max	52.000000	3.250000	6.250565e+07	2.274362e+07	2.047057e+07	2.546439e+06

```
avocado_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18249 entries, 0 to 18248
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Unnamed: 0    18249 non-null  int64  
 1   Date          18249 non-null  object 
 2   AveragePrice  18249 non-null  float64
```

```
 3   Total Volume  18249 non-null  float64
 4   4046          18249 non-null  float64
 5   4225          18249 non-null  float64
 6   4770          18249 non-null  float64
 7   Total Bags    18249 non-null  float64
 8   Small Bags    18249 non-null  float64
 9   Large Bags    18249 non-null  float64
 10  XLarge Bags   18249 non-null  float64
 11  type          18249 non-null  object
 12  year          18249 non-null  int64
 13  region        18249 non-null  object
dtypes: float64(9), int64(2), object(3)
memory usage: 1.9+ MB
```

```
avacado_df.isnull().sum()
```

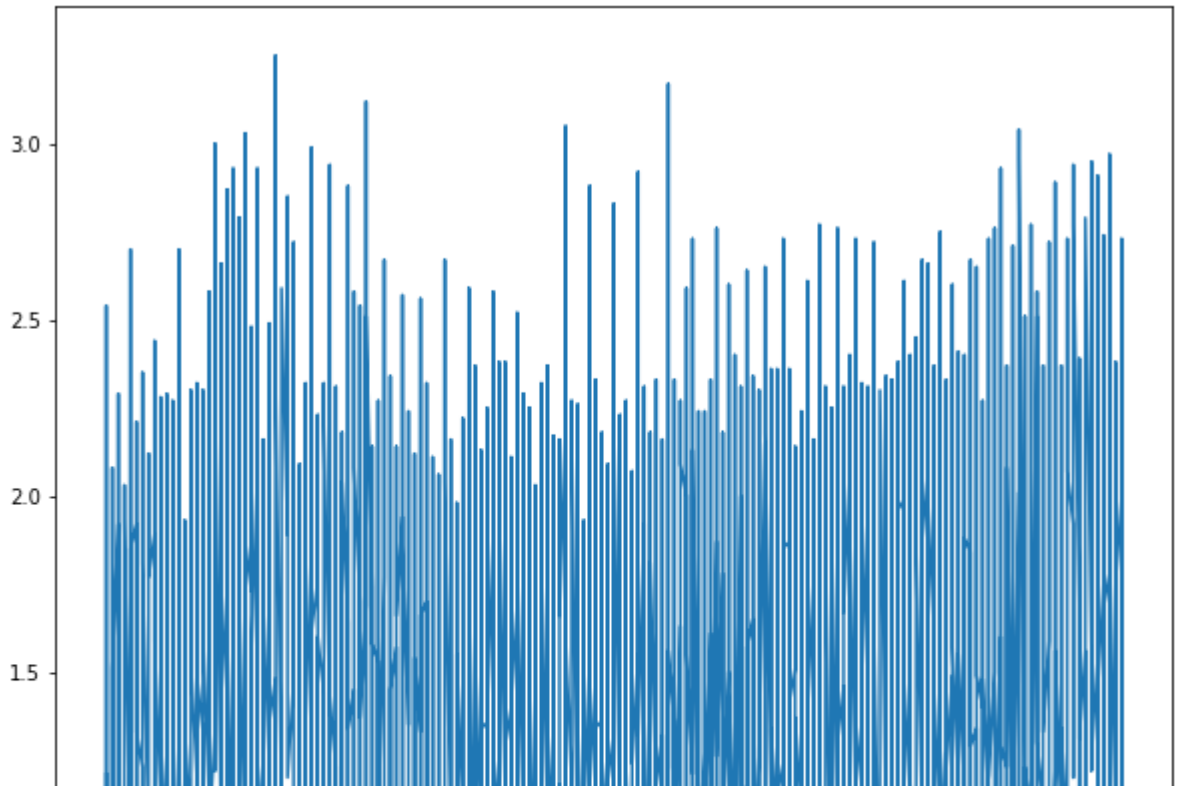
```
Unnamed: 0      0
Date            0
AveragePrice    0
Total Volume    0
4046            0
4225            0
4770            0
Total Bags      0
Small Bags      0
Large Bags      0
XLarge Bags     0
type            0
year            0
region          0
dtype: int64
```

EXPLORE DATASET

```
avocado_df = avocado_df.sort_values('Date')
```

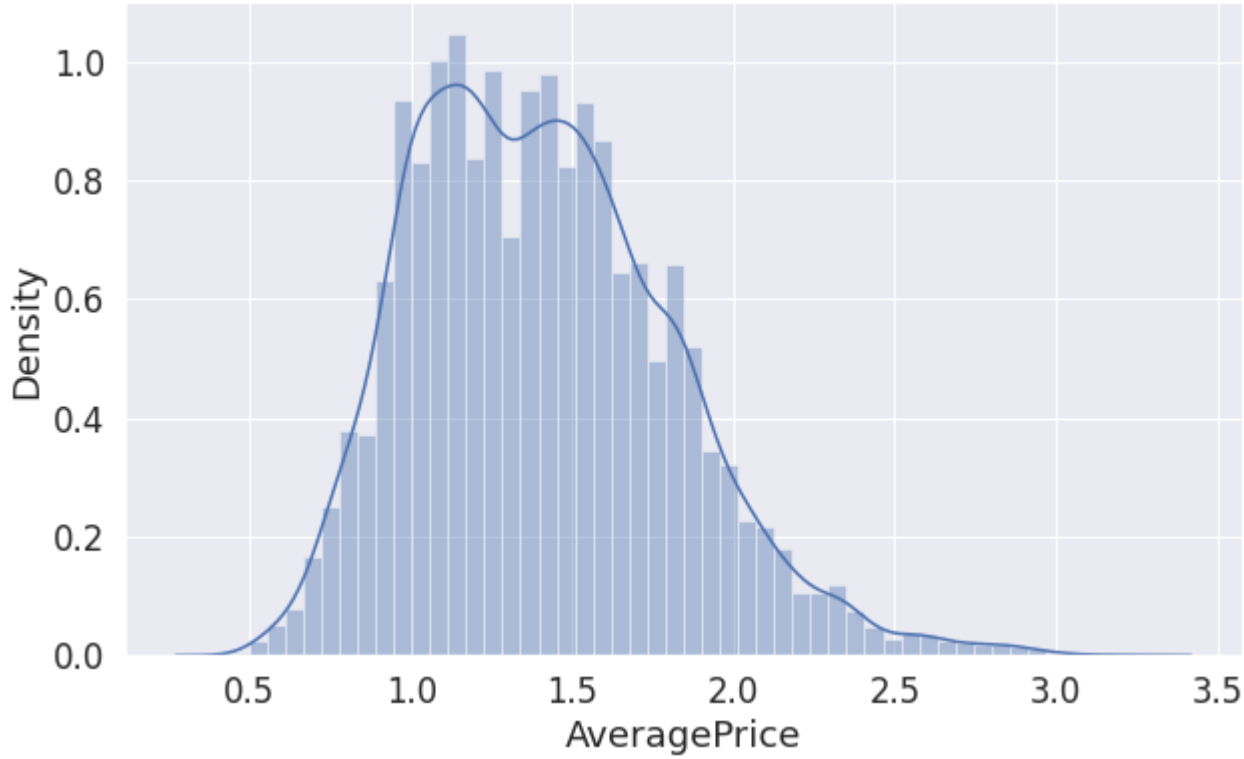
```
# Plot date and average price
plt.figure(figsize = (10,10))
plt.plot(avocado_df['Date'], avocado_df['AveragePrice'])
```

```
[<matplotlib.lines.Line2D at 0x7f57cb08d410>]
```



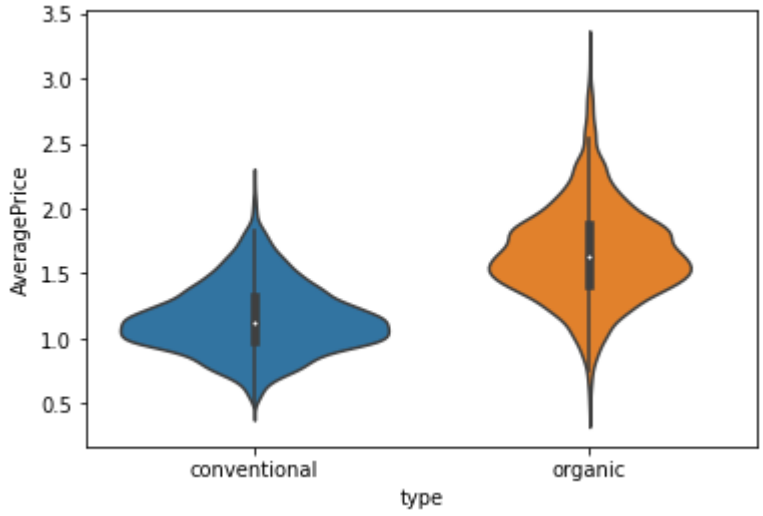

```
# Plot distribution of the average price
plt.figure(figsize = (10,6))
sns.distplot(avocado_df['AveragePrice'], color = 'b')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `di
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f57bf64c810>
```



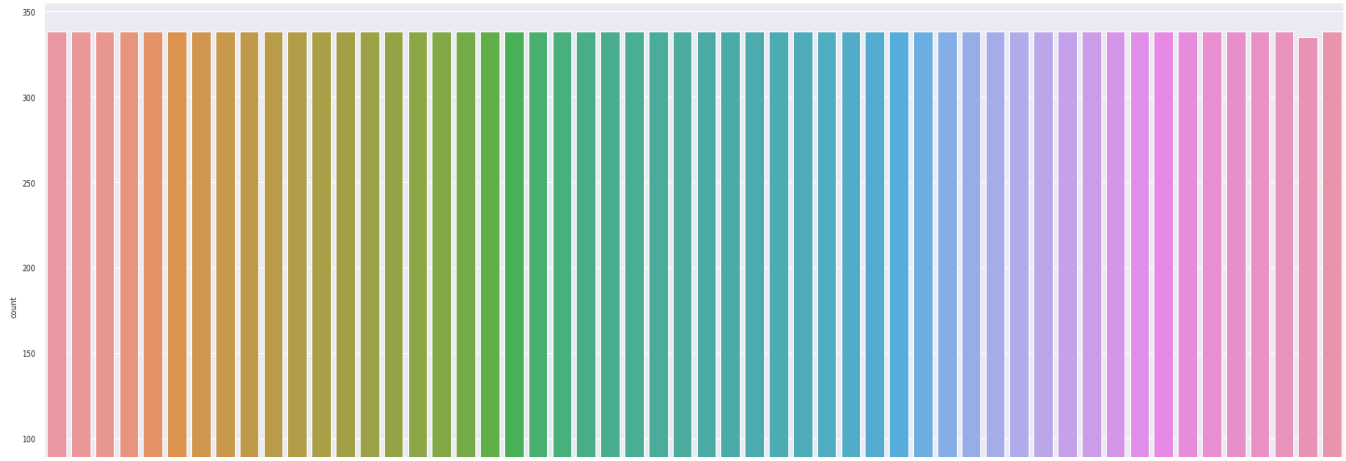
```
# Plot a violin plot of the average price vs. avocado type
sns.violinplot(y = 'AveragePrice', x = 'type', data = avocado_df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f57c3cc2110>
```



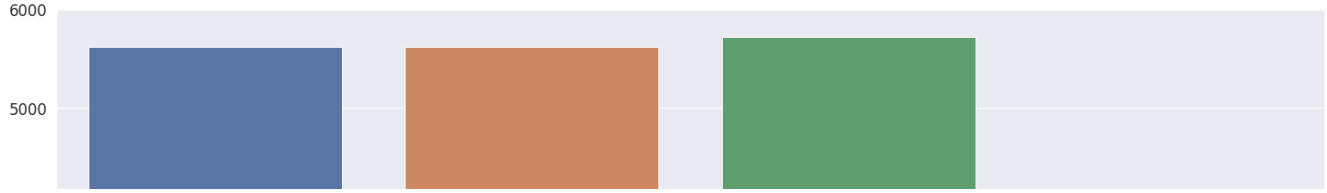

```
# Bar Chart to indicate the number of regions

sns.set(font_scale=0.7)
plt.figure(figsize=[25,12])
sns.countplot(x = 'region', data = avocado_df)
plt.xticks(rotation = 45)
```

```
(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 major ticklabel objects>)
```



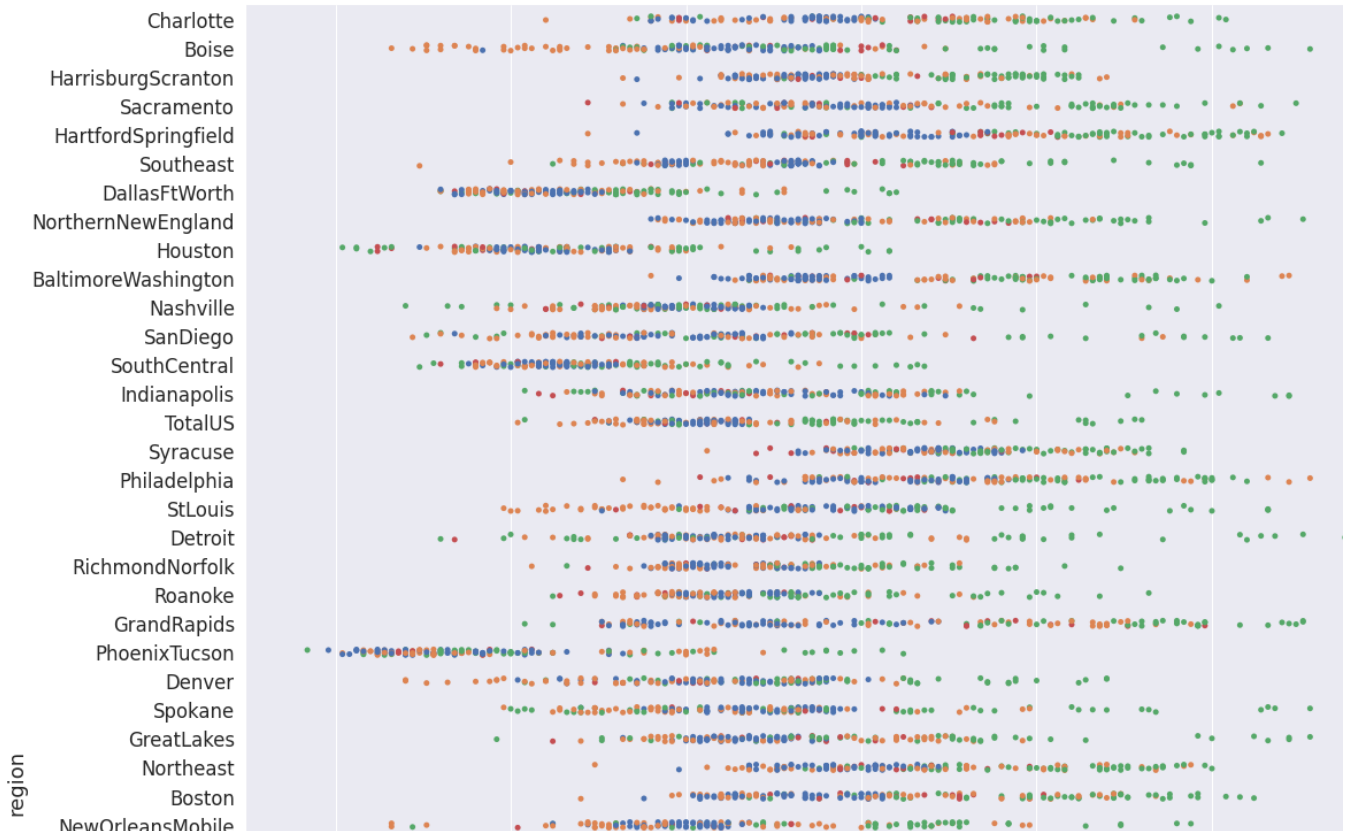

```
# Bar Chart to indicate the count in every year
sns.set(font_scale=1.5)
plt.figure(figsize=[25,12])
sns.countplot(x = 'year', data = avocado_df)
plt.xticks(rotation = 45)
```

```
(array([0, 1, 2, 3]), <a list of 4 Text major ticklabel objects>)
```




```
 # plot the avocado prices vs. regions for conventional avocados
conventional = sns.catplot('AveragePrice', 'region', data = avocado_df[avocado_df['type'] == 'conventional'],
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```




```
# plot the avocado prices vs. regions for organic avocados
conventional = sns.catplot('AveragePrice', 'region', data = avocado_df[avocado_df['type'] == 'organic'], hue
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```

PREPARE THE DATA BEFORE APPLYING FACEBOOK PROPHET TOOL

```
avocado_df
```

	Unnamed: 0	Date	AveragePrice	Total Volume	4046	4225	4770	T
6039	52	1/1/2017	1.21	217051.50	47765.32	94571.69	15036.44	5967
5827	52	1/1/2017	0.92	104510.11	27845.16	9408.92	11341.75	5591
6516	52	1/1/2017	1.35	235430.29	41800.77	136109.66	420.83	5709
15323	52	1/1/2017	1.58	5948.66	772.98	2724.28	0.00	245
15853	52	1/1/2017	1.24	3707.67	245.43	38.31	2.39	342
...	...	...	...	...	...	...	...	
10130	16	9/6/2015	1.62	2794.65	321.17	1731.98	0.00	74
2200	16	9/6/2015	1.54	646810.20	154276.07	384033.22	30732.28	7776
10182	16	9/6/2015	2.07	2016.06	1096.25	55.10	8.04	85
900	16	9/6/2015	1.27	281256.19	3976.35	221997.70	497.89	5478
1836	16	9/6/2015	1.12	1714650.22	857114.48	547236.44	15397.86	29490

18249 rows × 14 columns

```
avocado_prophet_df = avocado_df[['Date', 'AveragePrice']]

avocado_prophet_df = avocado_prophet_df.rename(columns = {'Date': 'ds', 'AveragePrice': 'y'} )

avocado_prophet_df
```

ds y

DEVELOP MODEL AND MAKE PREDICTIONS - PART A

```
m = Prophet()
m.fit(avocado_prophet_df)
```

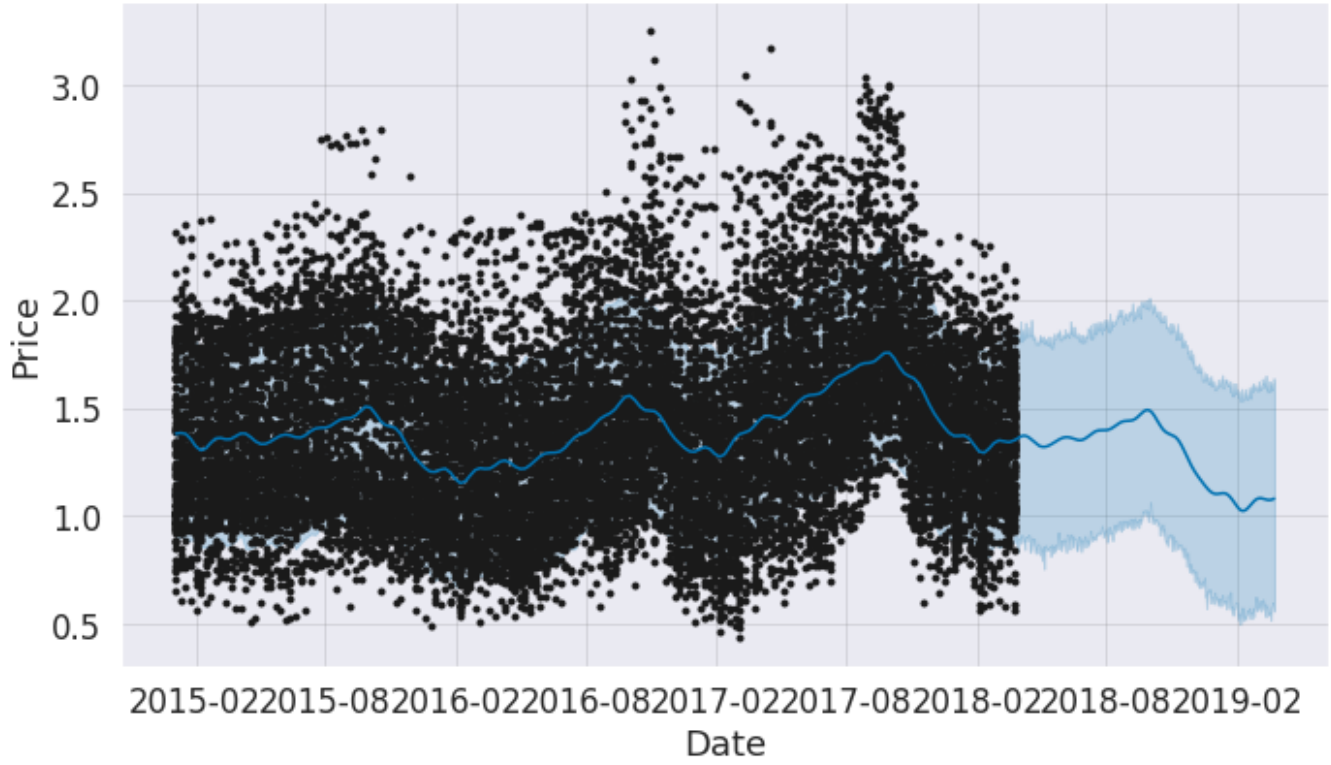
```
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to c
<fbprophet.forecaster.Prophet at 0x7f57c1e23a10>
```

```
# Forcasting into the future
future = m.make_future_dataframe(periods = 365)
forecast = m.predict(future)
```

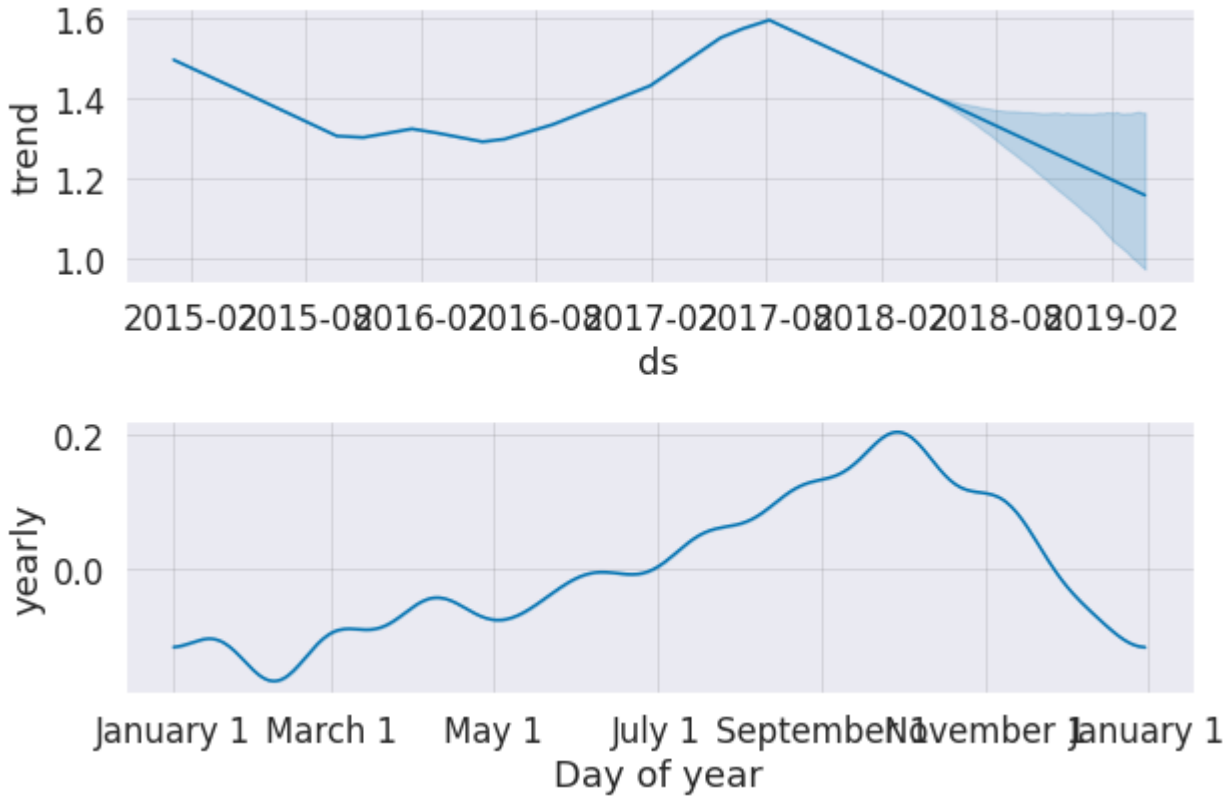
```
forecast
```

```
          ds     trend  yhat_lower  yhat_upper  trend_lower  trend_upper  additive_terms  a
```

```
figure = m.plot(forecast, xlabel = 'Date', ylabel = 'Price')
```




```
figure1 = m.plot_components(forecast)
```



DEVELOP MODEL AND MAKE PREDICTIONS (REGION SPECIFIC)

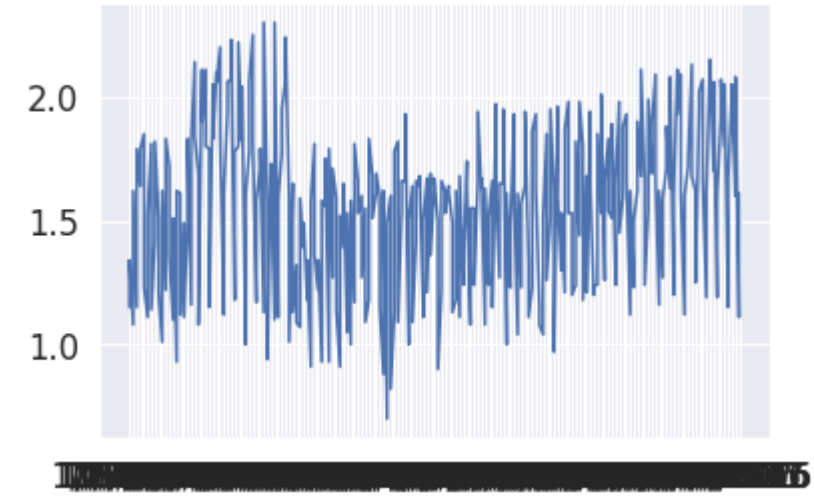
```
# dataframes creation for both training and testing datasets
avocado_df = pd.read_csv('avocado.csv')
```

```
# Select specific region
avocado_df_sample = avocado_df[avocado_df['region'] == 'Chicago']
```

```
avocado_df_sample = avocado_df_sample.sort_values('Date')
```

```
plt.plot(avocado_df_sample['Date'], avocado_df_sample['AveragePrice'])
```

```
INFO:matplotlib.category:Using categorical units to plot a list of strings that are all
INFO:matplotlib.category:Using categorical units to plot a list of strings that are all
[<matplotlib.lines.Line2D at 0x7f57bf810b90>]
```



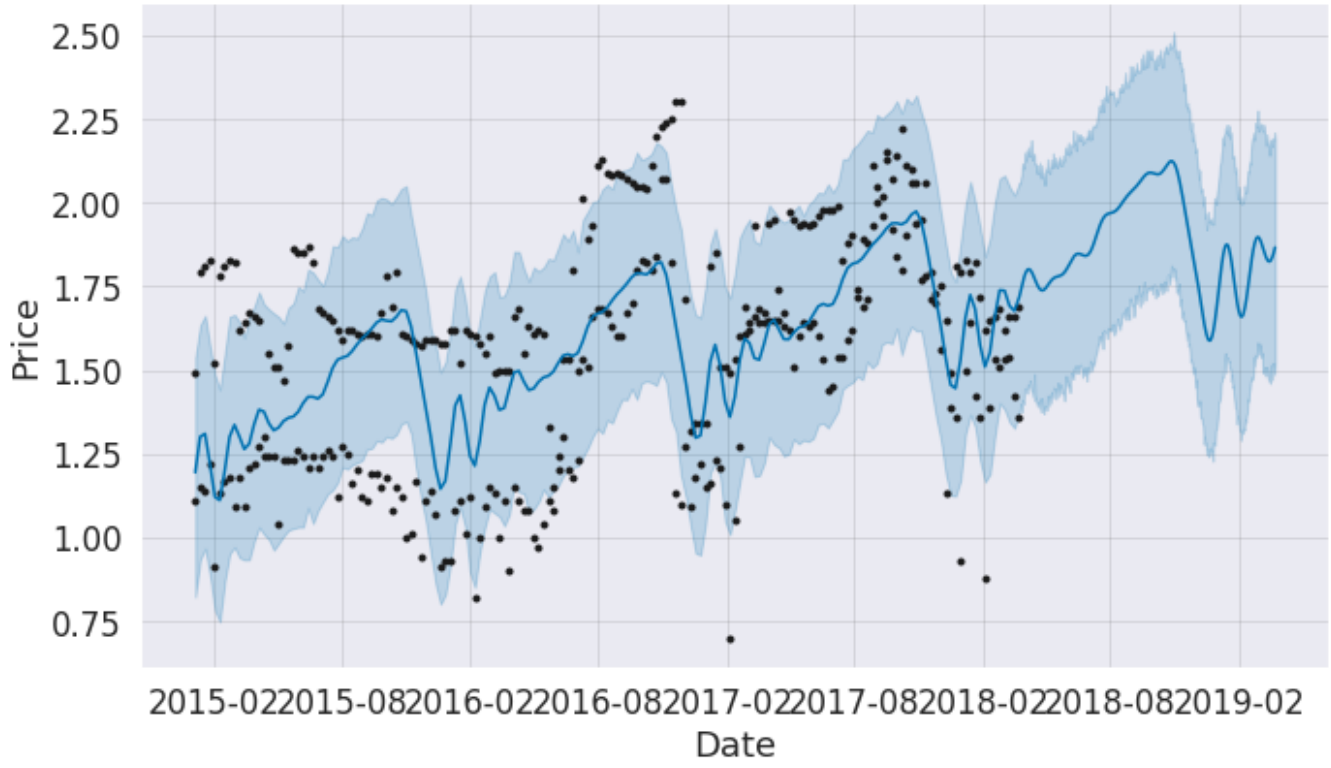

```
avocado_df_sample = avocado_df_sample.rename(columns = {'Date': 'ds', 'AveragePrice': 'y'})
```

```
m = Prophet()
m.fit(avocado_df_sample)
```

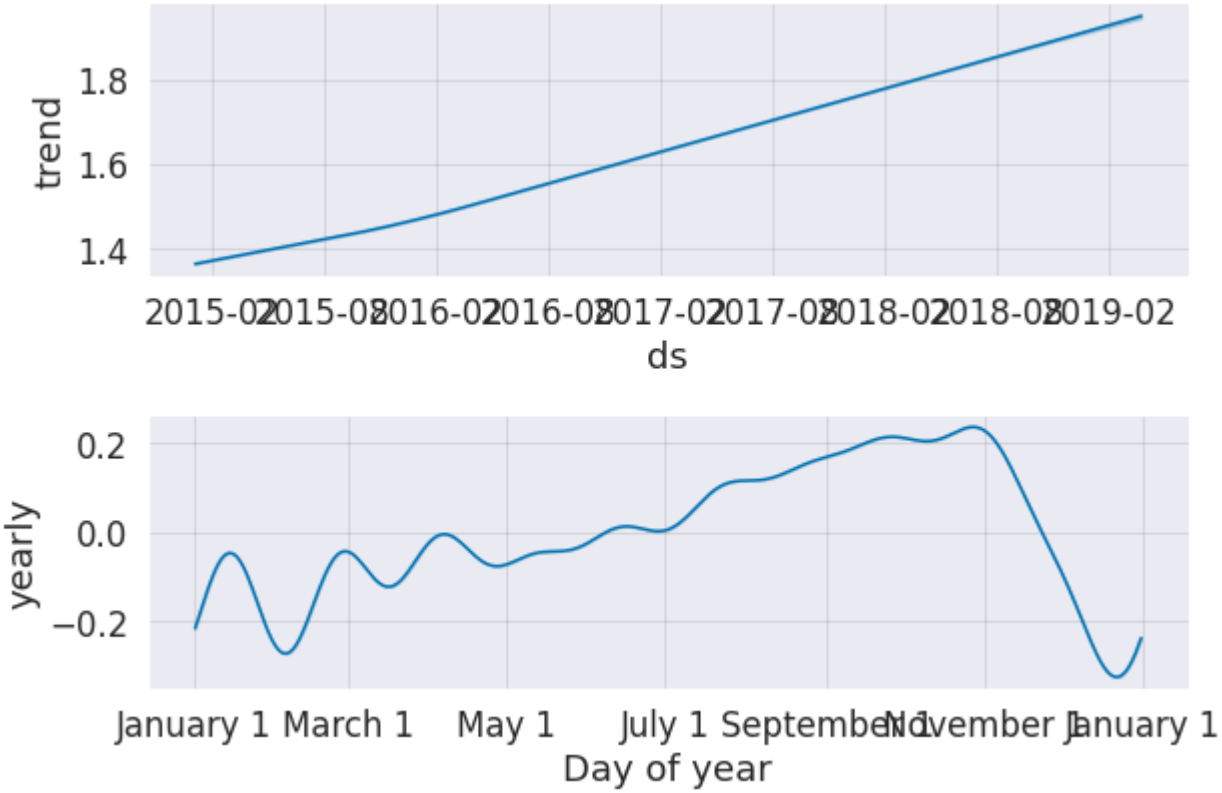
```
# Forcasting into the future
future = m.make_future_dataframe(periods=365)
forecast = m.predict(future)
```

```
INFO:fbprophet:Disabling weekly seasonality. Run prophet with weekly_seasonality=True to
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to c
```

```
figure = m.plot(forecast, xlabel='Date', ylabel='Price')
```




```
figure3 = m.plot_components(forecast)
```


trend
1.8
1.6
1.4
2015-02
2015-08
2016-02
2016-08
2017-02
2017-08
2018-02
2018-08
2019-02
ds
yearly
0.2
0.0
−0.2
January 1
March 1
May 1
July 1
September 1
November 1
January 1
Day of year