tobacco_consumption

April 10, 2022

1 Tobacco Consumption

Tobacco consumption is one of the primary causes of lung cancer in the World. Tobacco in the form of cigars and cigarettes is usually available to adult population in many supermarkets and grocery stores. The data obtained for this analysis describes Tobacco Consumption in USA from 2000 to 2020. From behavior of the data in those 21 years, the aim of the project is to predict total tobacco consumption in 2021 and 2022.

At first, the libraries used for this project are imported.

[ ]:
```
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import numpy as np
import seaborn as sns
import random
import math
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.api import Holt
from statsmodels.tsa.arima.model import ARIMA
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_percentage_error
```

An additional import is included in order to ignore some warnings while processing the data.

[ ]:
```
import warnings
warnings.simplefilter(action="ignore", category=FutureWarning)
```

1.1 Extraction

The data for this project is stored in a *.csv* file. The path to the file is defined in the variable *DATA_PATH*.

[ ]:
```
DATA_PATH = "../data/Tobacco_Consumption.csv"
```

The file is read and a sample of the data is shown.

[ ]:
```
tobacco_data_raw = pd.read_csv(DATA_PATH)
tobacco_data_raw.sample(10)
```

[ ]:

```
     Year LocationAbbrev LocationDesc  Population                  Topic  \
151  2011             US     National   237657645  Noncombustible Tobacco
69   2005             US     National   222003984  Noncombustible Tobacco
207  2015             US     National   247773709     Combustible Tobacco
134  2010             US     National   235153929     Combustible Tobacco
148  2011             US     National   237657645     Combustible Tobacco
177  2013             US     National   242542967     Combustible Tobacco
209  2016             US     National   249485228     Combustible Tobacco
59   2004             US     National   219552929     Combustible Tobacco
229  2017             US     National   252063800     Combustible Tobacco
96   2007             US     National   227239768     Combustible Tobacco

               Measure             Submeasure        Data Value Unit  \
151  Smokeless Tobacco                  Snuff                 Pounds
69   Smokeless Tobacco        Chewing Tobacco                 Pounds
207             Cigars           Small Cigars                 Cigars
134      Loose Tobacco           Pipe Tobacco                 Pounds
148      Loose Tobacco  Roll-Your-Own Tobacco  Cigarette Equivalents
177      Loose Tobacco    Total Loose Tobacco                 Pounds
209             Cigars           Large Cigars                 Cigars
59       Loose Tobacco           Pipe Tobacco                 Pounds
229      Loose Tobacco  Roll-Your-Own Tobacco  Cigarette Equivalents
96       Loose Tobacco  Roll-Your-Own Tobacco  Cigarette Equivalents

       Domestic     Imports        Total  Domestic Per Capita  \
151   103097456      464519    103561975                0.434
69     38883276      315916     39199192                0.175
207   530680751    23505000    554185751                2.000
134    22266231     2822447     25088678                0.000
148  2412067938   209522215   2621590154               10.000
177    42645920     3456679     46102599                0.000
209  5056761893  6974839000  12031600893               20.000
59      3904810      794487      4699297                0.000
229  1212400738    99499323   1311900062                5.000
96   8508941785   816734031   9325675815               37.000

     Imports Per Capita  Total Per Capita
151               0.002             0.436
69                0.001             0.177
207               0.000             2.000
134               0.000             0.000
148               1.000            11.000
177               0.000             0.000
209              28.000            48.000
59                0.000             0.000
229               0.000             5.000
96                4.000            41.000
```

1.2 Exploratory Data Analysis

Describe data table

[ ]:
```
tobacco_data_raw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 273 entries, 0 to 272
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Year                 273 non-null    int64
 1   LocationAbbrev       273 non-null    object
 2   LocationDesc         273 non-null    object
 3   Population           273 non-null    int64
 4   Topic                273 non-null    object
 5   Measure              273 non-null    object
 6   Submeasure           273 non-null    object
 7   Data Value Unit      273 non-null    object
 8   Domestic             273 non-null    int64
 9   Imports              273 non-null    int64
 10  Total                273 non-null    int64
 11  Domestic Per Capita  273 non-null    float64
 12  Imports Per Capita   273 non-null    float64
 13  Total Per Capita     273 non-null    float64
dtypes: float64(3), int64(5), object(6)
memory usage: 30.0+ KB
```

In this table, there are categorial and numerical variables.

The exploration will initially focus on categorical variables and later on the numerical ones.

1.2.1 Categorical Data Exploration

The categorical data columns are filtered from the original dataframe.

[ ]:
```
# Filter categorical variables from data
tobacco_categorical_data = tobacco_data_raw.select_dtypes(exclude=['int',␣
↪'float'])
# Show head of tables
tobacco_categorical_data.head(10)
```

[ ]:
```
  LocationAbbrev LocationDesc                  Topic           Measure  \
0             US     National  Noncombustible Tobacco  Smokeless Tobacco
1             US     National     Combustible Tobacco        Cigarettes
2             US     National     Combustible Tobacco            Cigars
3             US     National     Combustible Tobacco     Loose Tobacco
4             US     National     Combustible Tobacco     Loose Tobacco
5             US     National     Combustible Tobacco            Cigars
6             US     National     Combustible Tobacco     Loose Tobacco
```

```
7              US     National     Combustible Tobacco      Loose Tobacco
8              US     National     Combustible Tobacco      Loose Tobacco
9              US     National     Combustible Tobacco             Cigars

              Submeasure        Data Value Unit
0        Chewing Tobacco                 Pounds
1     Cigarette Removals             Cigarettes
2           Total Cigars                 Cigars
3     Total Loose Tobacco  Cigarette Equivalents
4     Total Loose Tobacco                Pounds
5           Small Cigars                 Cigars
6           Pipe Tobacco                 Pounds
7  Roll-Your-Own Tobacco  Cigarette Equivalents
8  Roll-Your-Own Tobacco                 Pounds
9           Large Cigars                 Cigars
```

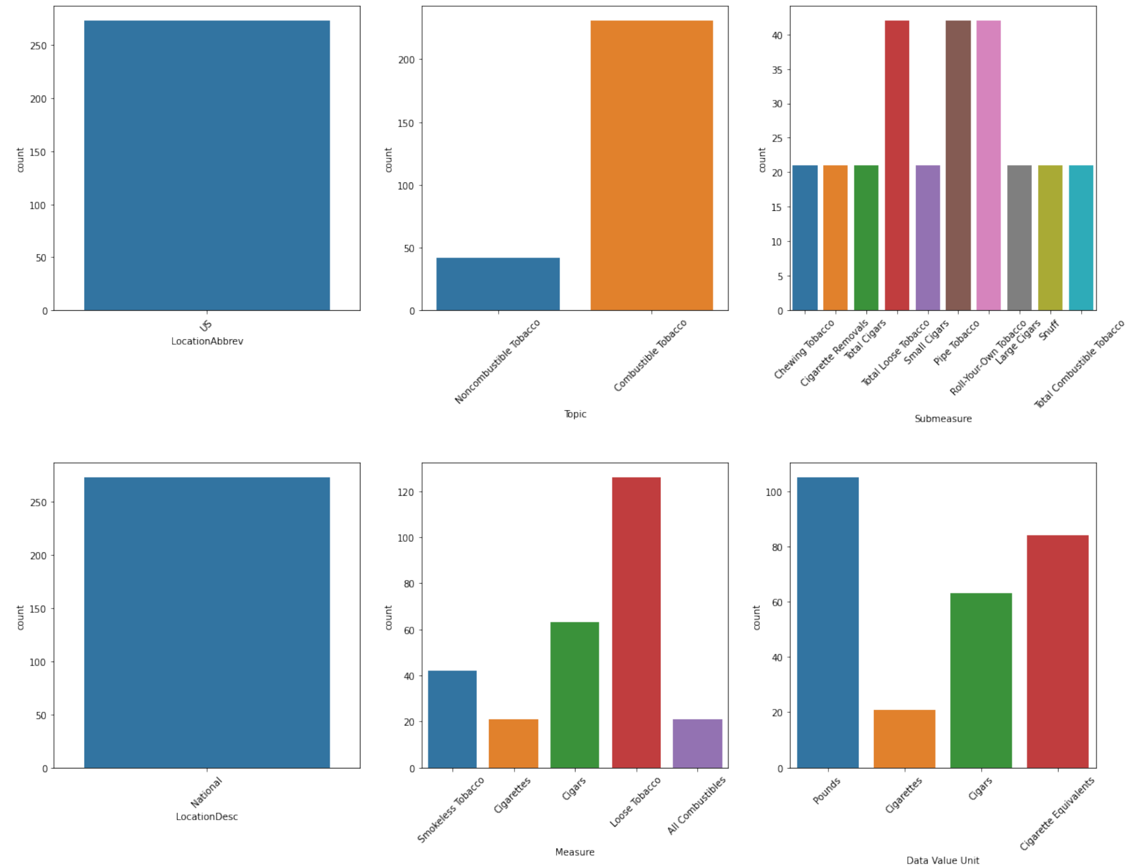
Categorical data columns are identified.

[ ]:
```
# Show numbers of columns
print(f"There is a total  of {len(tobacco_categorical_data.columns)}␣
↪categorical data columns")
# Show name of the columns
print(f"The columns are: {tobacco_categorical_data.columns}")
```

```
There is a total  of 6 categorical data columns
The columns are: Index(['LocationAbbrev', 'LocationDesc', 'Topic', 'Measure',
'Submeasure',
       'Data Value Unit'],
      dtype='object')
```

To explore the frecuency of elements for each column, frecuency is ploted in a bar chart, where x axis is the name of the elements in the column, and yaxis is the number of times the element is in the column.

[ ]:
```
# Create plot object
fig, ax = plt.subplots(2,3, figsize=(20, 15))
fig.subplots_adjust(hspace=.5)
i = 0
# Add subplot of frecuency of elements per column of categociall data
for col in tobacco_categorical_data.columns:
    sns.countplot(tobacco_categorical_data[col], ax=ax[i%2, math.floor(i/2)])
    i+=1
# Rotate axis of each subplot
for ax in fig.axes:
    plt.sca(ax)
    plt.xticks(rotation=45)
```



For *LocationDesc* and *LocationAbbrev* columns there is only one unique value each. Therefore, these columns are constants.

Most values in submeasure have a 21 apperances in the table.

The combinations of values in the columns "Measure", "Submeasure" and "Units" is further explored, to identify how many time each different combinations is shown in the table.

Categorical data combinations Unique combinations of categories are obtained.

[ ]:
```
# Get unique combinations by dropping duplicated categorical columns
tobacco_categorical_data.drop_duplicates()
```

[ ]:
```
  LocationAbbrev LocationDesc                   Topic            Measure  \
0             US     National  Noncombustible Tobacco  Smokeless Tobacco
1             US     National     Combustible Tobacco         Cigarettes
2             US     National     Combustible Tobacco             Cigars
3             US     National     Combustible Tobacco      Loose Tobacco
4             US     National     Combustible Tobacco      Loose Tobacco
5             US     National     Combustible Tobacco             Cigars
6             US     National     Combustible Tobacco      Loose Tobacco
```

```
7               US      National      Combustible Tobacco      Loose Tobacco
8               US      National      Combustible Tobacco      Loose Tobacco
9               US      National      Combustible Tobacco             Cigars
10              US      National      Combustible Tobacco      Loose Tobacco
11              US      National   Noncombustible Tobacco   Smokeless Tobacco
12              US      National      Combustible Tobacco    All Combustibles

                   Submeasure        Data Value Unit
0              Chewing Tobacco                 Pounds
1           Cigarette Removals             Cigarettes
2                 Total Cigars                 Cigars
3           Total Loose Tobacco  Cigarette Equivalents
4           Total Loose Tobacco                Pounds
5                 Small Cigars                 Cigars
6                 Pipe Tobacco                 Pounds
7         Roll-Your-Own Tobacco  Cigarette Equivalents
8         Roll-Your-Own Tobacco                Pounds
9                 Large Cigars                 Cigars
10                Pipe Tobacco   Cigarette Equivalents
11                       Snuff                 Pounds
12    Total Combustible Tobacco  Cigarette Equivalents
```

Describe combinations and unique combinations.

```
[ ]: # Get number of unique combinations and total combinations in the table
total_categories_combinations = len(tobacco_categorical_data)
unique_categories_combinations = len(tobacco_categorical_data.drop_duplicates())
# Print summary
print(f"Total combinations of categories (rows):␣
 ↪{total_categories_combinations}")
print(f"Find {unique_categories_combinations} unique category combinations")
print(f"Relation: {total_categories_combinations/
 ↪unique_categories_combinations}")
```

```
Total combinations of categories (rows): 273
Find 13 unique category combinations
Relation: 21.0
```

13 combinations are repeated 21 times in the table.

This number match the number of years in the data. The dataset included 13 different values per year.

1.2.2 Numerical Data Exploration

The numerical data columns are filtered from the original dataframe.

```
[ ]:
```

```
# Filter numerical variables from data
tobacco_numerical_data = tobacco_data_raw.select_dtypes(include=['int',␣
↪'float'])
# Show head of tables
tobacco_numerical_data.head(10)
```

```
[ ]:    Year  Population      Domestic      Imports         Total  \
0  2000   209786736      45502156        91965      45594121
1  2000   209786736  423250355675  12319663000  435570018675
2  2000   209786736    5612867329    548243000    6161110329
3  2000   209786736    8291276800    702741662    8994018462
4  2000   209786736      16841656      1427444      18269100
5  2000   209786736    2243135044     36049000    2279184044
6  2000   209786736       5352683       739887       6092570
7  2000   209786736    5656109785    338489600    5994599385
8  2000   209786736      11488973       687557      12176530
9  2000   209786736    3369732285    512194000    3881926285

   Domestic Per Capita  Imports Per Capita  Total Per Capita
0                0.217                 0.0             0.217
1             2018.000                59.0          2076.000
2               27.000                 3.0            29.000
3               40.000                 3.0            43.000
4                0.000                 0.0             0.000
5               11.000                 0.0            11.000
6                0.000                 0.0             0.000
7               27.000                 2.0            29.000
8                0.000                 0.0             0.000
9               16.000                 2.0            19.000
```

Numerical data columns are identified.

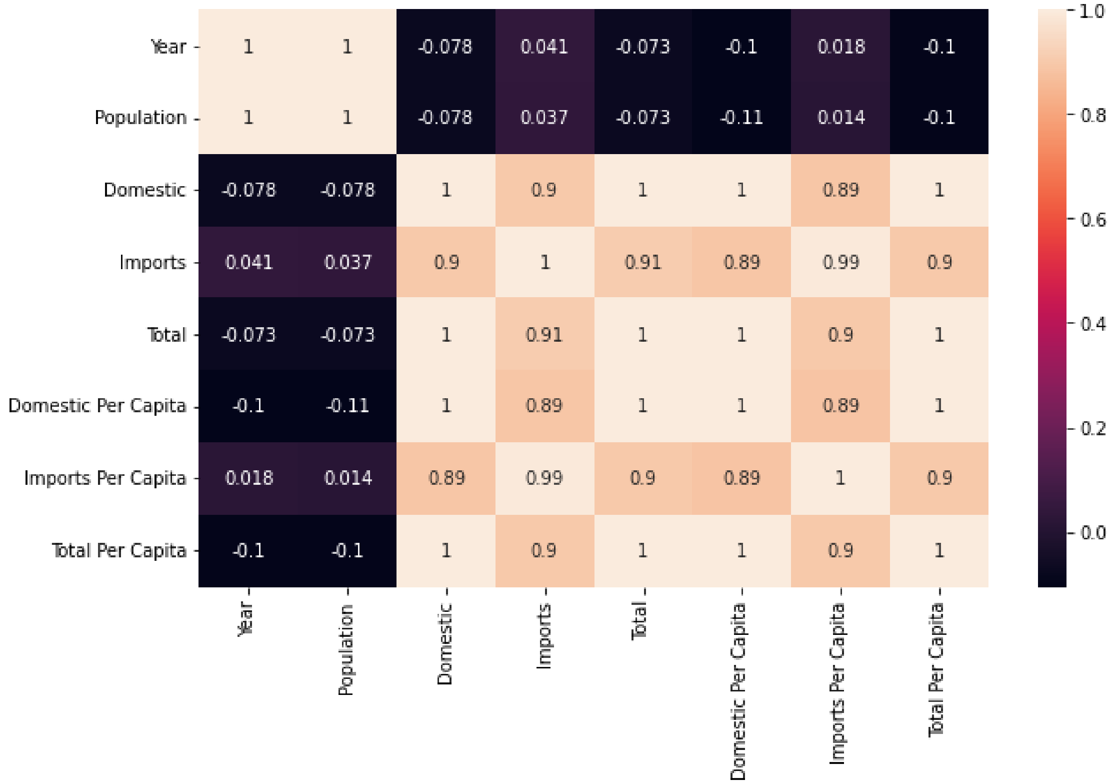
```
[ ]: # Show numbers of columns
print(f"There is a total  of {len(tobacco_numerical_data.columns)} numerical␣
↪data columns")
# Show name of the columns
print(f"The columns are: {tobacco_numerical_data.columns}")
```

```
There is a total  of 8 numerical data columns
The columns are: Index(['Year', 'Population', 'Domestic', 'Imports', 'Total',
       'Domestic Per Capita', 'Imports Per Capita', 'Total Per Capita'],
      dtype='object')
```

To understand how each variable is related to each other, correlations are obtained and plotted.

```
[ ]: # Explore correlations
correlations = tobacco_numerical_data.corr()
# Plot correlations
```

```
sns.heatmap(correlations, annot=True)
plt.show()
```



Year and *Population* have a strong correation with each other, but a low correation to tobacco values.

Per capita values have a strong correlation with normal values.

A test is applied to verify if per capita values are obtained from total values and population.

[ ]:
```
# Obtain difference between per capital columns and normal columns divided by␣
↪population
relation_per_capita = round(tobacco_numerical_data["Total"]/
↪tobacco_numerical_data["Population"], 1) - tobacco_numerical_data["Total Per␣
↪Capita"]
round(relation_per_capita.median(), 3)
```

[ ]: 0.003

The difference is close to 0. Therefore, the next expressions can be established from the data:

$$Domestic\_per\_capita = \frac{Domestic}{Population}$$

$$Imports\_per\_capita = \frac{Imports}{Population}$$
$$Total\_per\_capita = \frac{Total}{Population}$$

For further analysis, per capita columns are excluded.

Domestic and *Imports* have a strong correlation to *Total* column.

[ ]:
```
# Difference between total and imports + domestic is obtained
difference_total = tobacco_numerical_data["Total"]-␣
↪tobacco_numerical_data["Domestic"] - tobacco_numerical_data["Imports"]
difference_total.median()
```

[ ]:
```
0.0
```

The difference is 0, so
$$Total = Imports + Domestic$$

To have a better understading of this variables, it is needed to combine numerical exploration with the unique categories exploration. After that analysis, the relation between submeasures is expected to be identified.

1.2.3 Integrated Exploration (Categories & Numerical Data)

As each year has the same category combinations, one year (2000) is used as a sample. As this analysis focuses in tobacco consumption, only units related to products are taken in to account. Therefore, unit "Pounds" is excluded.

[ ]:
```
# Get filtered df
products_df = tobacco_data_raw[(tobacco_data_raw["Data Value Unit"] !=␣
↪"Pounds") & (tobacco_data_raw["Year"] == 2000)]
products_df
```

[ ]:
```
    Year LocationAbbrev LocationDesc  Population                Topic  \
1   2000             US     National   209786736  Combustible Tobacco
2   2000             US     National   209786736  Combustible Tobacco
3   2000             US     National   209786736  Combustible Tobacco
5   2000             US     National   209786736  Combustible Tobacco
7   2000             US     National   209786736  Combustible Tobacco
9   2000             US     National   209786736  Combustible Tobacco
10  2000             US     National   209786736  Combustible Tobacco
12  2000             US     National   209786736  Combustible Tobacco

          Measure            Submeasure        Data Value Unit  \
1      Cigarettes    Cigarette Removals             Cigarettes
2          Cigars          Total Cigars                 Cigars
3   Loose Tobacco   Total Loose Tobacco  Cigarette Equivalents
5          Cigars          Small Cigars                 Cigars
7   Loose Tobacco  Roll-Your-Own Tobacco Cigarette Equivalents
```

```
9             Cigars              Large Cigars                   Cigars
10     Loose Tobacco              Pipe Tobacco  Cigarette Equivalents
12  All Combustibles  Total Combustible Tobacco  Cigarette Equivalents

        Domestic      Imports         Total  Domestic Per Capita  \
1   423250355675  12319663000  435570018675               2018.0
2     5612867329    548243000    6161110329                 27.0
3     8291276800    702741662    8994018462                 40.0
5     2243135044     36049000    2279184044                 11.0
7     5656109785    338489600    5994599385                 27.0
9     3369732285    512194000    3881926285                 16.0
10    2635167015    364252062    2999419077                 13.0
12  437154499804  13570647662  450725147466               2084.0

    Imports Per Capita  Total Per Capita
1                 59.0            2076.0
2                  3.0              29.0
3                  3.0              43.0
5                  0.0              11.0
7                  2.0              29.0
9                  2.0              19.0
10                 2.0              14.0
12                65.0            2148.0
```

Total Loose Tobacco is compared to Pipe Tobaco and Roll-Your-Own Tobacco

[ ]:
```
# Compare diff between Total Loose Tobacco and Pipe Tobacco
products_df["Domestic"][3] - products_df["Domestic"][7] -␣
 ↪products_df["Domestic"][10]
```

[ ]:
```
0
```

Therefore,

$$Total\_Loose\_Tobacco = Pipe\_Tobacco + Roll\_Your\_Own\_Tobacco$$

The Loose Tobacco values are in the table twice (as pounds and as cigarette equivalents), that's the reason the frecuency was the double than other cases in categorical data analysis.

Total Cigars are compared to Small and Large Cigars…

[ ]:
```
products_df["Domestic"][2] - products_df["Domestic"][5] -␣
 ↪products_df["Domestic"][9]
```

[ ]:
```
0
```

For Cigars:

$$Total\_Cigars = Small\_Cigars + Large\_Cigars$$

```
[ ]: # Get sum of non-total submeasures
sum_cigarettes = products_df["Domestic"][~products_df["Submeasure"].str.
 ↪contains("Total")].sum()
# Compare sum to Total Combustible Tobacco variable
products_df["Domestic"][products_df["Submeasure"]=="Total Combustible Tobacco"]␣
 ↪- sum_cigarettes
```

```
[ ]: 12    0
Name: Domestic, dtype: int64
```

$$Total\_Combustible\_Tobacco = Total\_Cigars + Total\_Loose\_Tobaccoo + Cigarette\_Removals$$

Cigarette, Cigarette Equivalents, and Cigars units have a 1:1:1 relationship.

Total Combustible Tobacco contains information of all types of tobacco products submeasures. This value will be the target variable that is going to be predicted in the analysis.

1.3 Data Wrangling

The original dataframe is filtered and transfromed to get a useful table focused in the target variable. Unnecesary columns are drop and year is set as index of the table.

```
[ ]: total_combustible_tobacco_df =␣
 ↪tobacco_data_raw[tobacco_data_raw["Submeasure"]=="Total Combustible Tobacco"]
# Drop columns with constant information
total_combustible_tobacco_df.drop(columns=["LocationAbbrev", "LocationDesc",␣
 ↪"Topic", "Measure",
    "Submeasure", "Data Value Unit"], inplace=True)
# To reduce data with similar behavior, per capita values will be also ignored␣
 ↪in the transformation
total_combustible_tobacco_df.drop(columns=["Domestic Per Capita", "Imports Per␣
 ↪Capita", "Total Per Capita"], inplace=True)
# Year to index and datetime object
total_combustible_tobacco_df.set_index("Year", inplace = True)
total_combustible_tobacco_df.index = pd.
 ↪to_datetime(total_combustible_tobacco_df.index, format = "%Y")
# Show time series
total_combustible_tobacco_df
```

```
C:\Users\edson\AppData\Local\Temp\ipykernel_10308\332722847.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  total_combustible_tobacco_df.drop(columns=["LocationAbbrev", "LocationDesc",
"Topic", "Measure",
```

```
C:\Users\edson\AppData\Local\Temp\ipykernel_10308\332722847.py:6:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  total_combustible_tobacco_df.drop(columns=["Domestic Per Capita", "Imports Per
Capita", "Total Per Capita"], inplace=True)
```

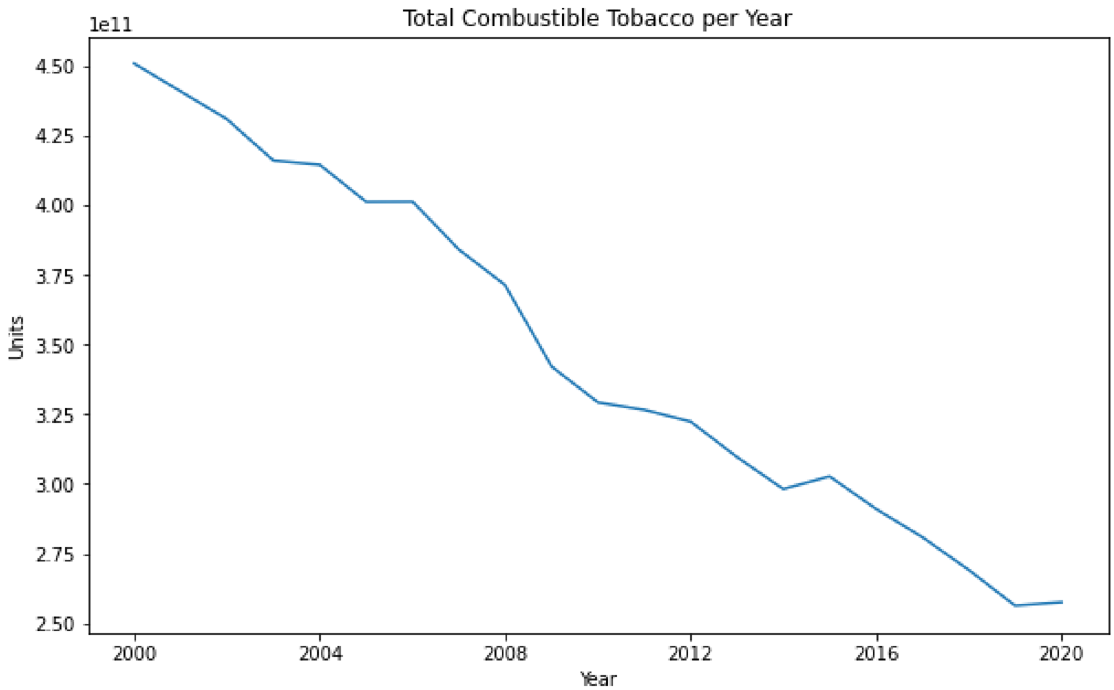
[ ]:

Year	Population	Domestic	Imports	Total
2000-01-01	209786736	437154499804	13570647662	450725147466
2001-01-01	212345162	424447310483	16245487123	440692797606
2002-01-01	214754648	408303465673	22459975923	430763441596
2003-01-01	217068101	391196952717	24733051892	415930004609
2004-01-01	219552929	390166439041	24254733538	414421172579
2005-01-01	222003984	380712422031	20474781600	401187203631
2006-01-01	224583123	382748665603	18493161400	401241827003
2007-01-01	227239768	367880476823	16206169554	384086646377
2008-01-01	229945137	357080696204	14183410908	371264347112
2009-01-01	232458335	329491123539	12632961400	342124084939
2010-01-01	235153929	317475052695	11764157615	329239210311
2011-01-01	237657645	315683514293	10893776123	326577290416
2012-01-01	240185952	310009439421	12386990662	322396430083
2013-01-01	242542967	295476582903	14164450662	309641033564
2014-01-01	245273438	282020020624	16175868538	298195889163
2015-01-01	247773709	285537129046	17214155385	302751284431
2016-01-01	249485228	273881993213	17154801323	291036794536
2017-01-01	252063800	263359518483	17581121431	280940639914
2018-01-01	253768092	249351495626	19823602569	269167785869
2019-01-01	255200373	234601291857	21813751754	256415043611
2020-01-01	256662010	235888722742	21726898662	257615621404

Plot total over the years

[ ]:

```
sns.lineplot(x=total_combustible_tobacco_df.index,␣
  ↪y=total_combustible_tobacco_df["Total"])
plt.title("Total Combustible Tobacco per Year")
plt.ylabel("Units")
plt.show()
```



Store new table as csv.

```
[ ]: # Export ts to df
OUTPUT_PATH = "../data/Transformed_Tobacco_Consumption.csv"
total_combustible_tobacco_df.to_csv(OUTPUT_PATH, index=False)
```

1.4 Exploration of Transformed Data

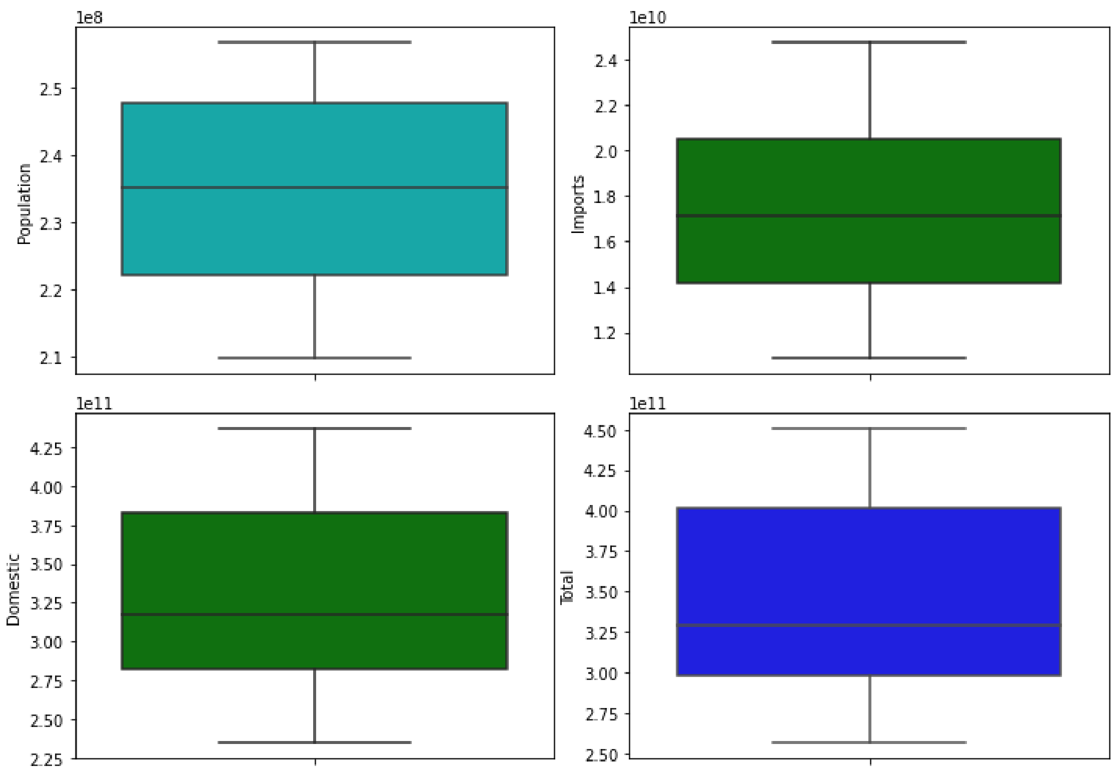
```
[ ]: # Explore variables distribution
total_combustible_tobacco_df.describe().convert_dtypes()
```

```
[ ]:          Population            Domestic             Imports  \
count              21.0                21.0                21.0
mean   234547860.285714  330117467277.190491  17331140748.761906
std     15123590.490099   63129184488.71344   4140066674.594648
min        209786736.0      234601291857.0       10893776123.0
25%        222003984.0      282020020624.0       14164450662.0
50%        235153929.0      317475052695.0       17154801323.0
75%        247773709.0      382748665603.0       20474781600.0
max        256662010.0      437154499804.0       24733051892.0

                     Total
count                 21.0
mean   347448271248.571411
std     63456602238.587479
```

```
min         256415043611.0
25%         298195889163.0
50%         329239210311.0
75%         401241827003.0
max         450725147466.0
```

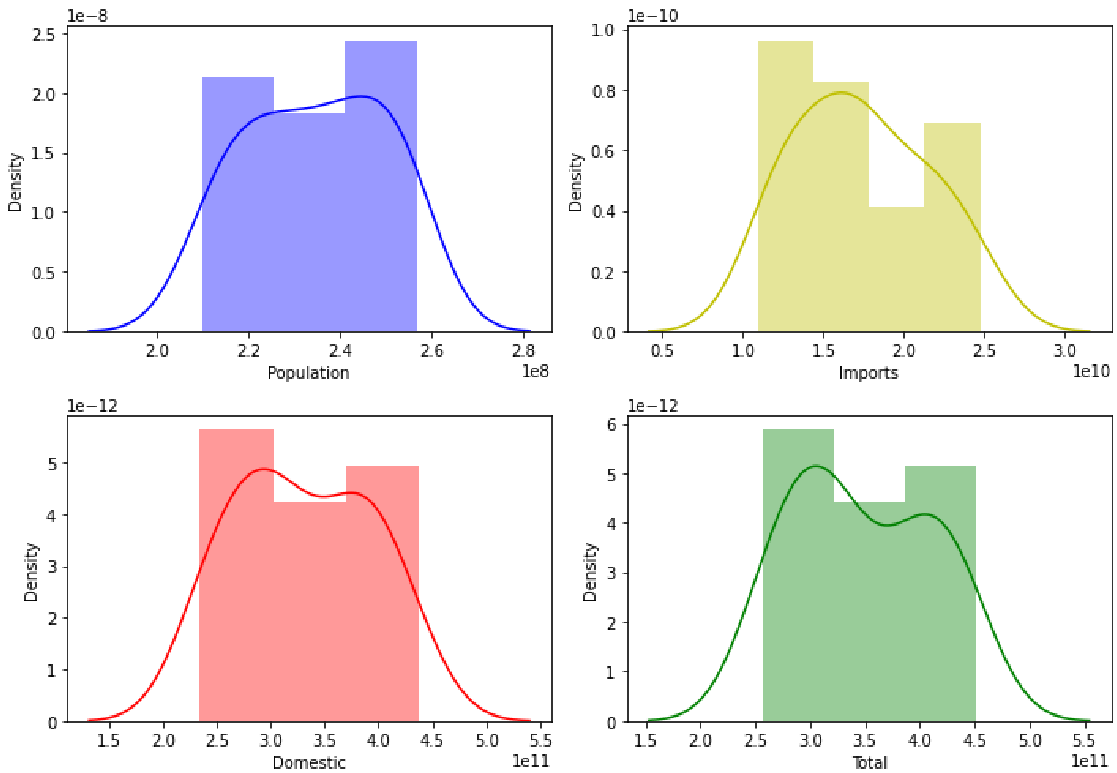
Show boxplots and histograms

```
[ ]: # Create boxplots
COLORS = ["b","g", "r", "c", "m", "y"]
fig, ax = plt.subplots(2,2, figsize=(10,7))
i = 0
for col in total_combustible_tobacco_df.columns:
    sns.boxplot(y=col, data=total_combustible_tobacco_df, color = random.
 ↪choice(COLORS), ax=ax[i%2, math.floor(i/2)])
    i+=1

plt.tight_layout()
```



For population and total, data seems to be symmetric. However, Domestic and Total are a little right-skewed.

```
[ ]: # Create histogram
COLORS = ["b","g", "r", "c", "m", "y"]
fig, ax = plt.subplots(2,2, figsize=(10,7))
i = 0
for col in total_combustible_tobacco_df.columns:
    sns.distplot(total_combustible_tobacco_df[col], color = random.
 ↪choice(COLORS), ax=ax[i%2, math.floor(i/2)])
    i+=1

plt.tight_layout()
```

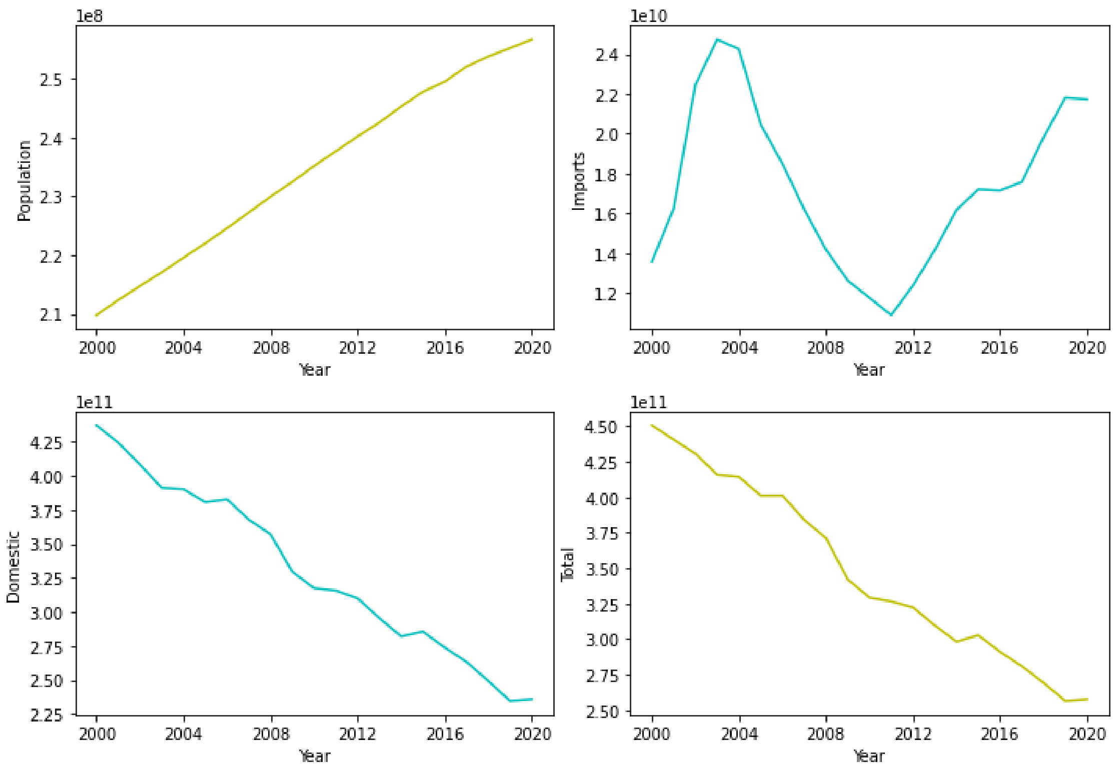


All variables seem close to be symetric. The previously identified as skewed variables are also close to the center of the data.

Plots all trends by year.

```
[ ]: COLORS = ["b","g", "r", "c", "m", "y"]
fig, ax = plt.subplots(2,2, figsize=(10,7))
i = 0
for col in total_combustible_tobacco_df.columns:
    sns.lineplot(x=total_combustible_tobacco_df.index,␣
 ↪y=total_combustible_tobacco_df[col],
        color = random.choice(COLORS), ax=ax[i%2, math.floor(i/2)])
```

```
    i+=1
plt.tight_layout()
```



The percentage of change of variables over time is explored.

1.4.1 % Change over the years

Get % of change of each variable and plot.

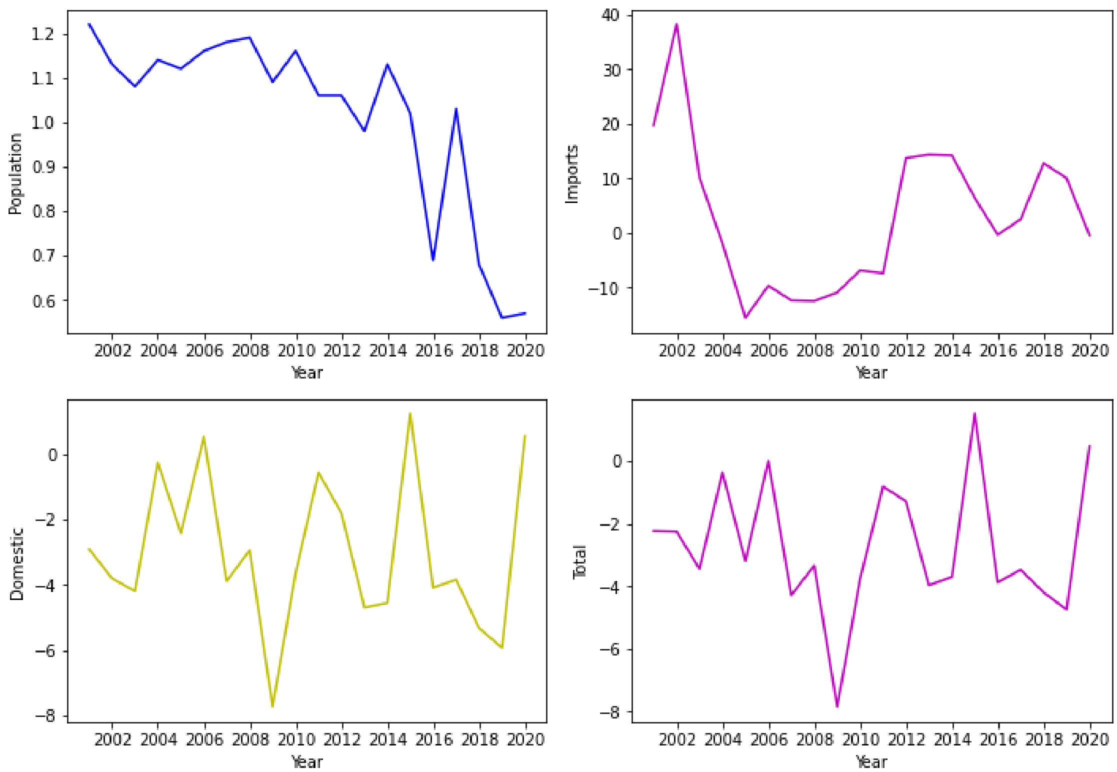
[ ]:
```
ts_change_df = total_combustible_tobacco_df.pct_change().dropna()
ts_change_df = round(ts_change_df *100,2)
ts_change_df.head(5)
```

[ ]:

Year	Population	Domestic	Imports	Total
2001-01-01	1.22	-2.91	19.71	-2.23
2002-01-01	1.13	-3.80	38.25	-2.25
2003-01-01	1.08	-4.19	10.12	-3.44
2004-01-01	1.14	-0.26	-1.93	-0.36
2005-01-01	1.12	-2.42	-15.58	-3.19

[ ]:
```
# Plot % of change of varibles
COLORS = ["b","g", "r", "c", "m", "y"]
```

```
fig, ax = plt.subplots(2,2, figsize=(10,7))
i = 0
for col in ts_change_df.columns:
    sns.lineplot(x=ts_change_df.index, y=ts_change_df[col], color = random.
↪choice(COLORS), ax=ax[i%2, math.floor(i/2)])
    i+=1
plt.tight_layout()
```



There is no clear behavior related to how much does each vairables changes per year.

1.4.2 Stationarity

Augmented Dickey-Fuller test is applied to verify if the data is stationary.

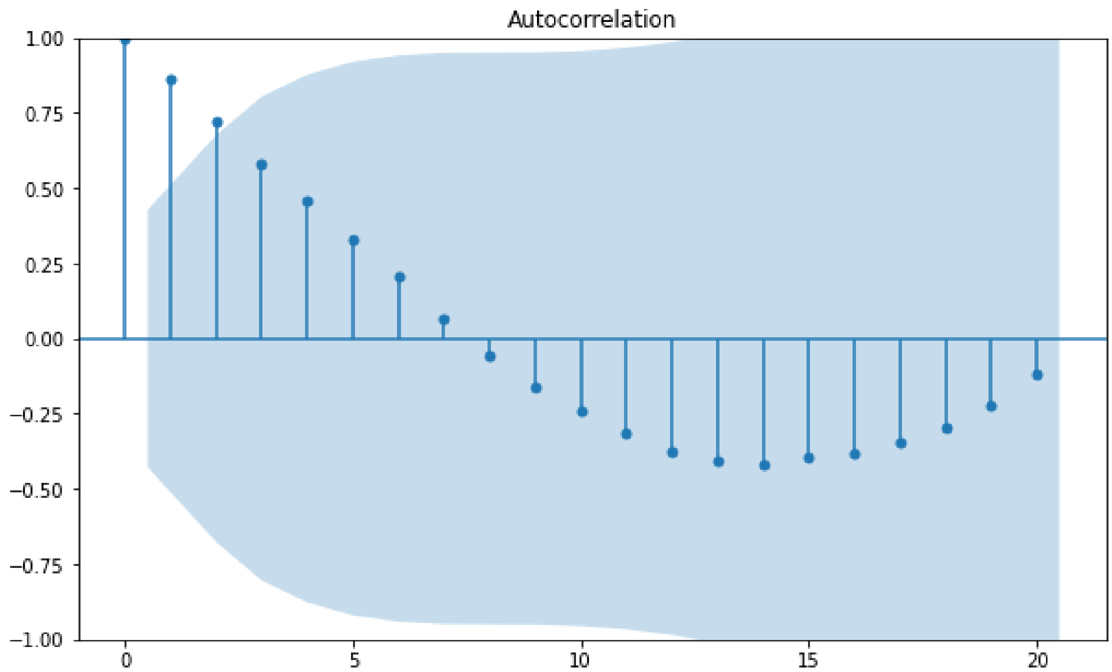
[ ]:
```
X = total_combustible_tobacco_df["Total"].values

result = sm.tsa.stattools.adfuller(X)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
        print('\t%s: %.3f' % (key, value))
```

```
ADF Statistic: -4.365042
p-value: 0.000342
Critical Values:
        1%: -4.138
        5%: -3.155
        10%: -2.714
```

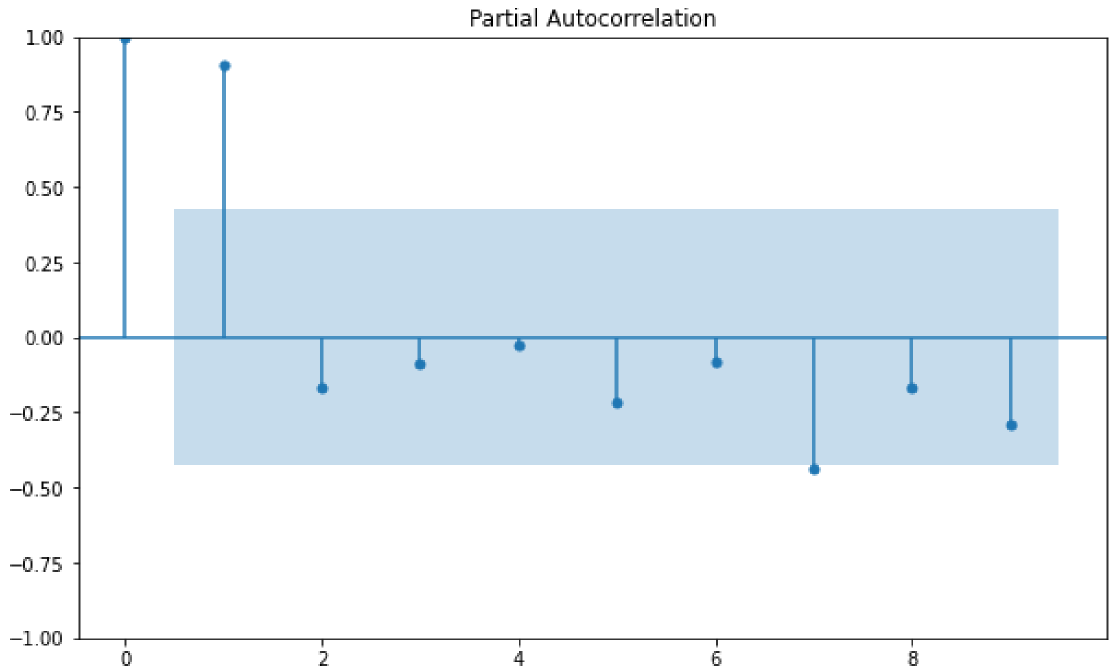
The p-value is really small (less than 5% threshold), so it is confirmed the total column is a stationary time series.

1.4.3 ACF and PACF

[ ]:
```
plt.rc("figure", figsize=(10,6))
sm.graphics.tsa.plot_acf(total_combustible_tobacco_df["Total"], lags=20)
plt.show()
```

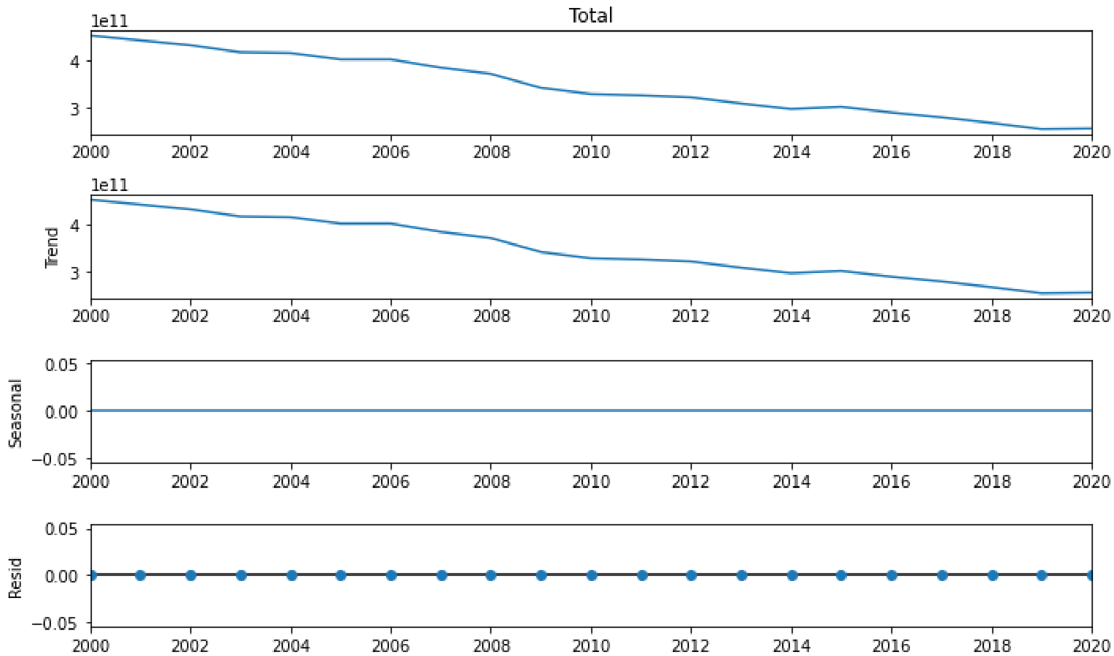


[ ]:
```
plt.rc("figure", figsize=(10,6))
sm.graphics.tsa.plot_pacf(total_combustible_tobacco_df["Total"], lags=9)
plt.show()
```



1.4.4 Time Series Decomposition

[ ]:
```
ts_decompose = seasonal_decompose(total_combustible_tobacco_df["Total"],␣
    ↪model="additive")
ts_decompose.plot()
plt.show()
```



There is neither seasonal component or resid.

1.5 Modeling

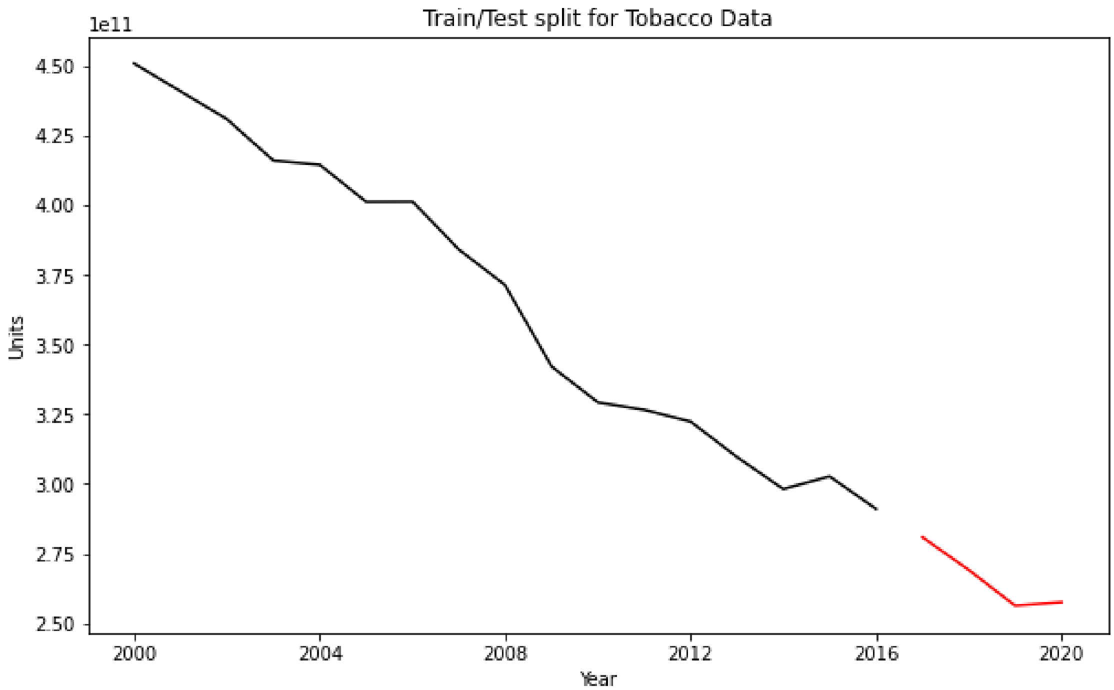
In this project, three models are compared: - Linear Regression - AutoRegresive Integrated Moving Average (ARIMA) - Holt's Exponential Smoothing

Two metrics are used for comparing and selecting a model: - Relative Root Mean Square Error (rRMSE) - Mean Absolute Percentage Error (MAPE)

One of the models will be selected to predict the total tobacco consumption over the next years.

At first, data is splitted in test and train datasets

[ ]:
```
train = total_combustible_tobacco_df["Total"][total_combustible_tobacco_df.
 ↪index < "12-12-2016"]
test = total_combustible_tobacco_df["Total"][total_combustible_tobacco_df.index␣
 ↪> "12-12-2016"]

plt.plot(train, color="black")
plt.plot(test, color = "red" )
plt.ylabel("Units")
plt.xlabel("Year")
plt.title("Train/Test split for Tobacco Data")
plt.show()
```



1.5.1 Linear Regression Model

In this model, indepedent variable is Year and dependent variable is the total tobacco consumption. Based on that, X and Y are defined.

[ ]:
```
# Get X and Y from training data
X = train.index.year.values.reshape(-1, 1)
Y = train.values.reshape(-1, 1)
```

The model is created and trained.

[ ]:
```
LR_model = LinearRegression()
LR_model.fit(X, Y)
```

[ ]:
```
LinearRegression()
```

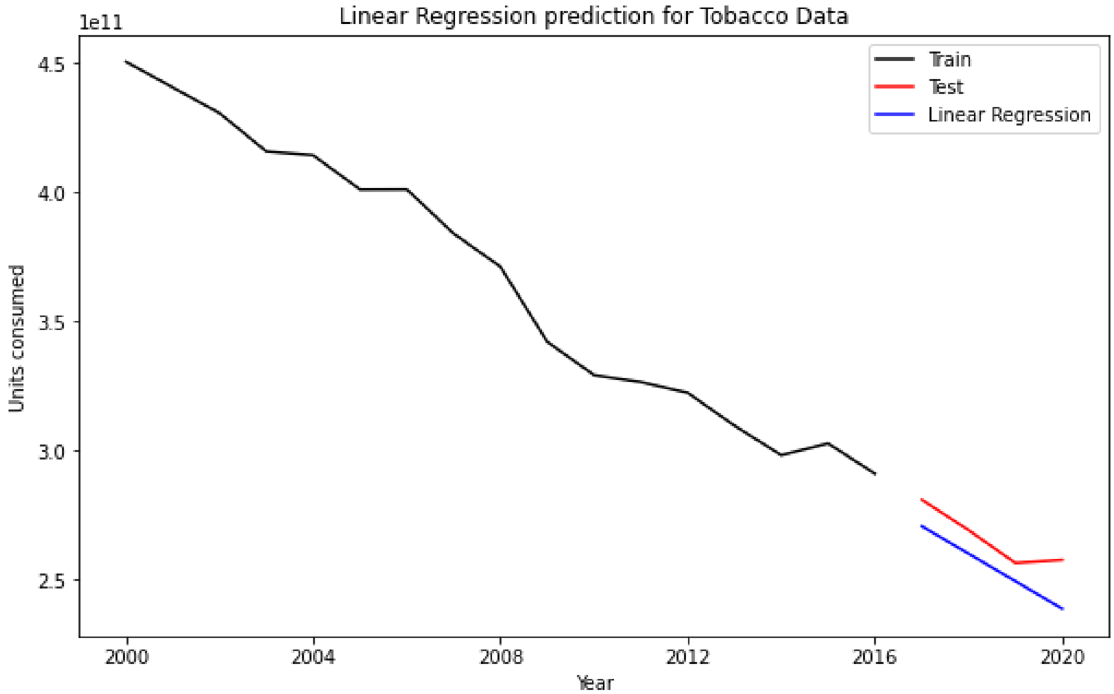
To test the model, predictions of the years in the test set (2017-2020) are made.

[ ]:
```
y_pred_lr = pd.Series(LR_model.predict(test.index.year.values.reshape(-1, 1)).
  ↪flatten(), index=test.index)
y_pred_lr
```

[ ]:
```
Year
2017-01-01    2.706961e+11
2018-01-01    2.600396e+11
2019-01-01    2.493831e+11
```

```
2020-01-01    2.387266e+11
dtype: float64
```

[ ]:
```
plt.plot(train, color="black", label = "Train")
plt.plot(test, color = "red", label = "Test")
plt.plot(y_pred_lr, color = "blue", label= "Linear Regression")
plt.ylabel("Units consumed")
plt.xlabel("Year")
plt.title("Linear Regression prediction for Tobacco Data")
plt.legend(loc="upper right")
plt.show()
```



1.5.2 ARIMA Model

Train model and tune hyperparameters.

[ ]:
```
ARIMA_model = ARIMA(train, order=(3,3,2))
ARIMA_model = ARIMA_model.fit()
print(ARIMA_model.summary())
```

```
c:\Users\edson\Documents\Repositorios\Data-Science\Tobacco-Consumption-
Prediction\code\jenv\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: No frequency information was provided, so inferred frequency AS-
JAN will be used.
  self._init_dates(dates, freq)
```

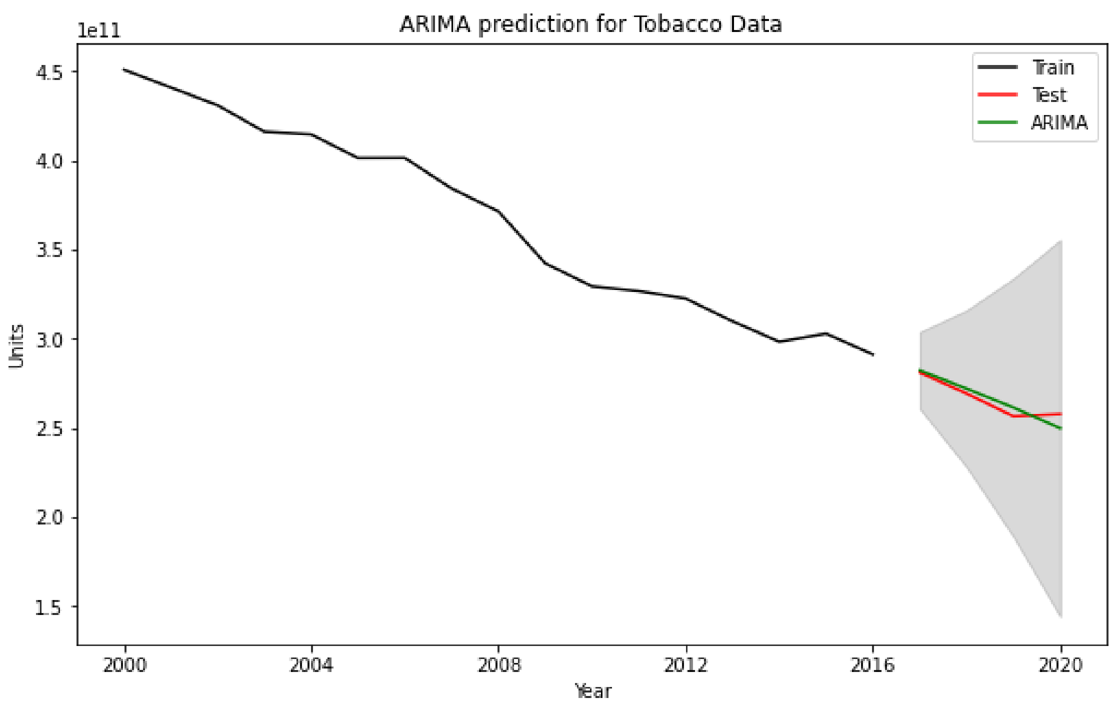
```
c:\Users\edson\Documents\Repositorios\Data-Science\Tobacco-Consumption-
Prediction\code\jenv\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: No frequency information was provided, so inferred frequency AS-
JAN will be used.
  self._init_dates(dates, freq)
c:\Users\edson\Documents\Repositorios\Data-Science\Tobacco-Consumption-
Prediction\code\jenv\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: No frequency information was provided, so inferred frequency AS-
JAN will be used.
  self._init_dates(dates, freq)
c:\Users\edson\Documents\Repositorios\Data-Science\Tobacco-Consumption-
Prediction\code\jenv\lib\site-
packages\statsmodels\tsa\statespace\sarimax.py:978: UserWarning: Non-invertible
starting MA parameters found. Using zeros as starting parameters.
  warn('Non-invertible starting MA parameters found.'

                               SARIMAX Results
==============================================================================
Dep. Variable:                  Total   No. Observations:                   17
Model:                 ARIMA(3, 3, 2)   Log Likelihood                -341.950
Date:                Sun, 10 Apr 2022   AIC                            695.901
Time:                        12:34:09   BIC                            699.735
Sample:                    01-01-2000   HQIC                           695.546
                         - 01-01-2016
Covariance Type:                  opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1         -1.0102      0.479     -2.108      0.035      -1.950      -0.071
ar.L2         -0.2133      0.257     -0.830      0.407      -0.717       0.291
ar.L3         -0.0263      0.088     -0.301      0.763      -0.198       0.145
ma.L1         -0.2394      0.612     -0.391      0.696      -1.439       0.960
ma.L2         -0.7161      0.851     -0.842      0.400      -2.383       0.951
sigma2      1.172e+20   7.68e-21   1.53e+40      0.000    1.17e+20    1.17e+20
================================================================================
===
Ljung-Box (L1) (Q):                   0.25   Jarque-Bera (JB):
0.83
Prob(Q):                              0.62   Prob(JB):
0.66
Heteroskedasticity (H):               6.17   Skew:
0.53
Prob(H) (two-sided):                  0.07   Kurtosis:
2.44
================================================================================
===

Warnings:
```

```
[1] Covariance matrix calculated using the outer product of gradients (complex-
step).
[2] Covariance matrix is singular or near-singular, with condition number
2.06e+56. Standard errors may be unstable.
```

To test the model, predictions of the years in the test set (2017-2020) are made.

```
[ ]: arima_pred = ARIMA_model.get_forecast(len(test))
     # Get confidence interval
     y_conf_int_df = arima_pred.conf_int(alpha=0.05)
     y_conf_int_df
     # Get predictions for test set years
     y_pred_arima = ARIMA_model.predict(start=test.index[0], end=test.index[-1])
```

```
[ ]: plt.plot(train, color="black", label = "Train")
     plt.plot(test, color = "red", label = "Test")
     plt.plot(y_pred_arima, color = "green", label= "ARIMA")
     plt.fill_between(y_conf_int_df.index, y_conf_int_df["lower Total"],␣
      ↪y_conf_int_df["upper Total"], color="k", alpha=0.15)
     plt.ylabel("Units")
     plt.xlabel("Year")
     plt.title("ARIMA prediction for Tobacco Data")
     plt.legend(loc="upper right")
     plt.show()
```



1.5.3 Holt's Exponential Smoothing Model (ETS)

Train model

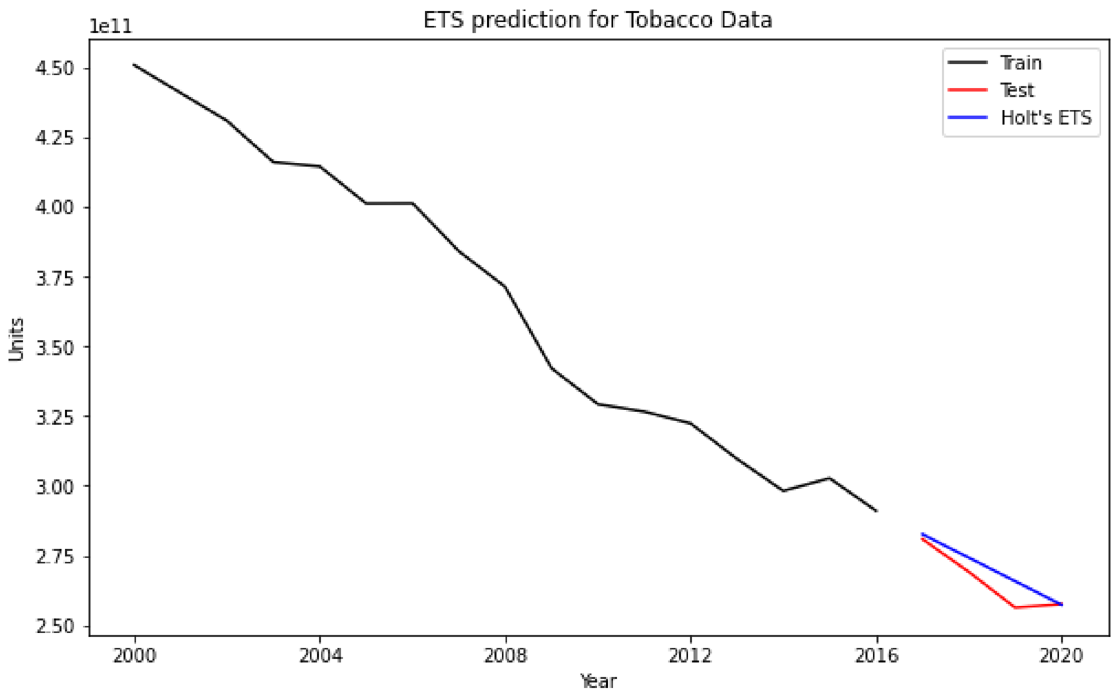
```
[ ]: holt_model = Holt(train, initialization_method="estimated").
     ↪fit(smoothing_level=0.8, smoothing_trend=0.2, optimized=False)
```

```
c:\Users\edson\Documents\Repositorios\Data-Science\Tobacco-Consumption-
Prediction\code\jenv\lib\site-packages\statsmodels\tsa\base\tsa_model.py:471:
ValueWarning: No frequency information was provided, so inferred frequency AS-
JAN will be used.
  self._init_dates(dates, freq)
```

To test the model, predictions of the years in the test set (2017-2020) are made.

```
[ ]: y_pred_ets = holt_model.forecast(len(test))
```

```
[ ]: plt.plot(train, color="black", label = "Train")
     plt.plot(test, color = "red", label = "Test")
     plt.plot(y_pred_ets, color = "blue", label= "Holt's ETS")
     plt.ylabel("Units")
     plt.xlabel("Year")
     plt.title("ETS prediction for Tobacco Data")
     plt.legend(loc="upper right")
     plt.show()
```



1.5.4 Evaluation and Selection

Define function to obtain relative root mean square error of predicted values.

```
[ ]: def rRMSE(actual_values, predicted_values, mean_value):
         rmse_value = np.sqrt(mean_squared_error(actual_values, predicted_values))
         rrmse_value = rmse_value/mean_value
         return rrmse_value
```

Create table that summarizes results of evaluation metrics.

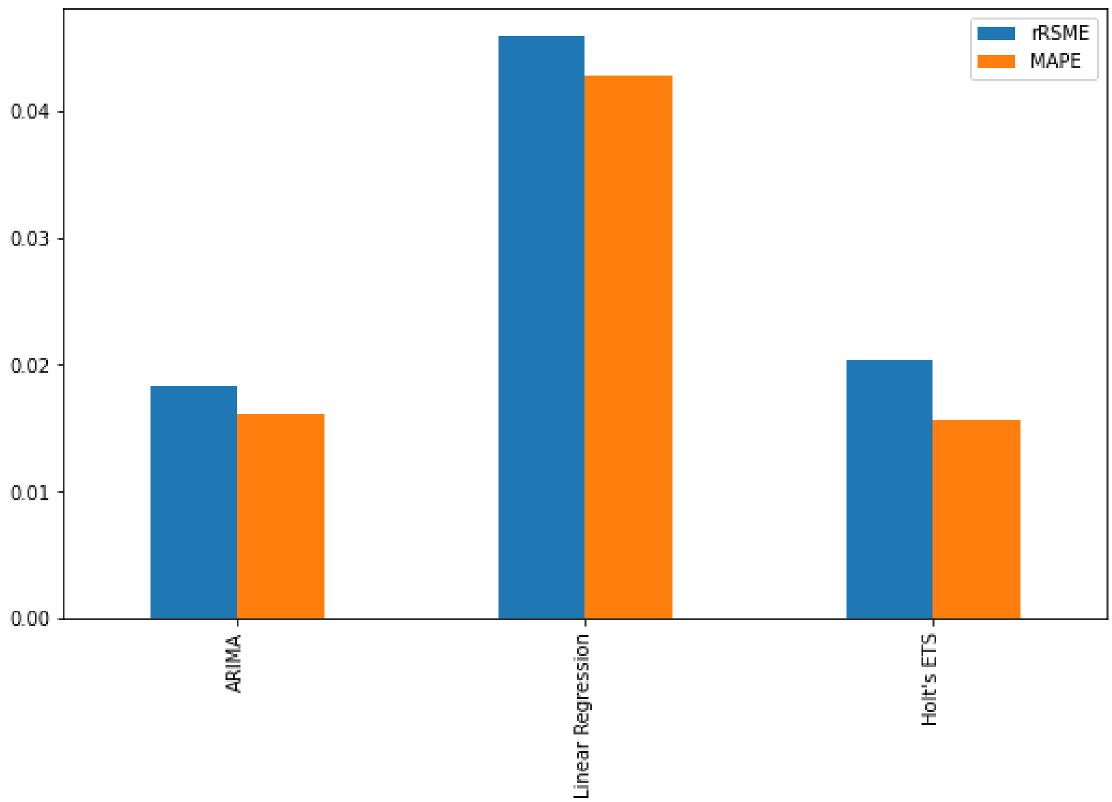
```
[ ]: # Get evaluation metrics in the form of a dict
     data_results = {"rRSME": [rRMSE(test.values, y_pred_arima.values, test.mean()),
                               rRMSE(test.values, y_pred_lr.values, test.mean()),
                               rRMSE(test.values, y_pred_ets.values, test.mean())],
                     "MAPE": [mean_absolute_percentage_error(test.values,␣
      ↪y_pred_arima.values),
                               mean_absolute_percentage_error(test.values, y_pred_lr.
      ↪values),
                               mean_absolute_percentage_error(test.values, y_pred_ets.
      ↪values)]}
     # Dict to df
     evaluation_df = pd.DataFrame(data_results, index= ["ARIMA", "Linear␣
      ↪Regression", "Holt's ETS"])
     evaluation_df
```

[ ]:

	rRSME	MAPE
ARIMA	0.018342	0.016066
Linear Regression	0.045826	0.042781
Holt's ETS	0.020407	0.015655

```
[ ]: evaluation_df.plot.bar()
```

```
[ ]: <AxesSubplot:>
```



Linear Regression model provided a worse performance than the other two models. ARIMA and ETS got similar results in MAPE metric, but performance of ARIMA was better than ETS in rRMSE metric. For ARIMA, both metric results are under 2%.

ARIMA model is selected.

1.6 Results

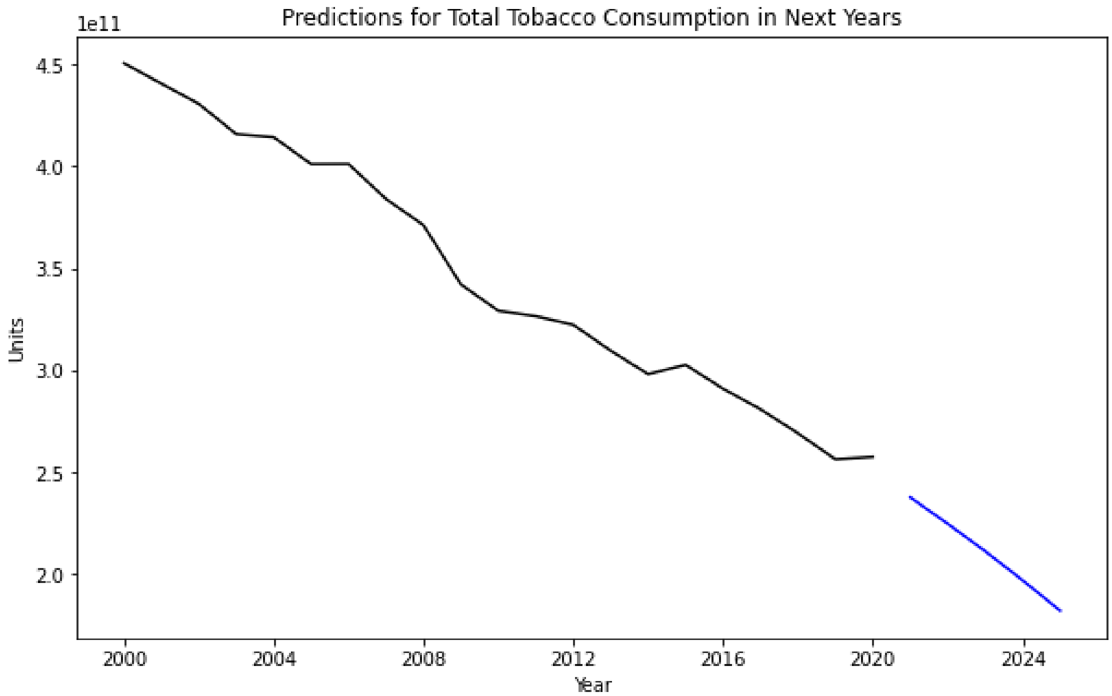
Get tobacco consumption for 2021, 2022 and 2023.

[ ]:
```
y_predictions_future = ARIMA_model.predict(start="2021", end="2025")
y_predictions_future
```

[ ]:
```
2021-01-01    2.376935e+11
2022-01-01    2.247738e+11
2023-01-01    2.113127e+11
2024-01-01    1.970416e+11
2025-01-01    1.821700e+11
Freq: AS-JAN, Name: predicted_mean, dtype: float64
```

[ ]:
```
plt.plot(total_combustible_tobacco_df["Total"], color="black", label = "Train")
plt.plot(y_predictions_future, color = "blue", label= "Predictions")
plt.ylabel("Units")
```

```
plt.xlabel("Year")
plt.title("Predictions for Total Tobacco Consumption in Next Years")
plt.show()
```



1.7 Conclusions

Total Tobacco Consumption is decreasing through the years in the United States.

For the next years, the total consumption is expected to keep decreasing.