Research report on Data Transformation with Python using Pandas:

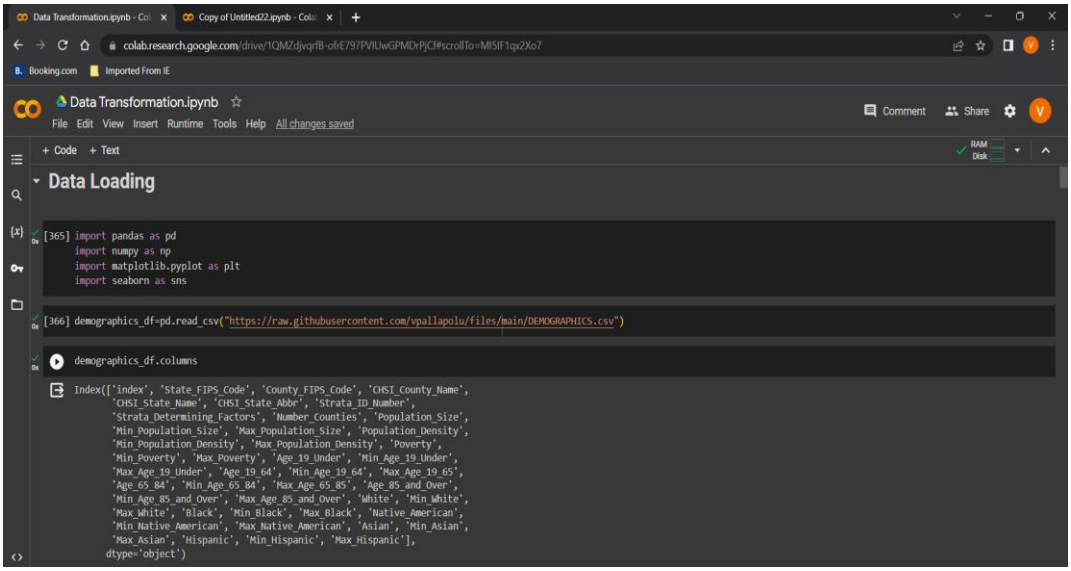
The process of transforming, cleaning, and organizing data into a format that can be used for analysis and support decision-making processes to advance an organization's expansion is known as data transformation [(1).] The process of data transformation can be constructive, destructive, aesthetic, or structural [(1)]. The data transformation procedure cleans up duplicate data, changes data types, removes unnecessary columns, deals with null values and outliers, renames the columns to easily understandable names and enhances the dataset to transform raw data into a format that can be used.

Introduction: The Demographics dataset chosen for the data transformation is from the Kaggle website [(2)] that indicates the population characteristics related to each state in the USA like county, strata determining factors, counties per strata, population size, population density, poverty, different age group population, different races population.

Demographic features like racial and population characteristics contribute to the enrichment and profiling of county-level observations. The steps involved in cleaning a dataset are as follows:

Step 1: Data loading

- The demographics dataset is downloaded on local computer as a csv (comma-separated value) file and then loaded into the google colab platform by using the pd.read_csv() function and giving the dataset URL as a parameter. Then, to perform the Data Transformation on the loaded dataset pandas, NumPy, Matplotlib, and Seaborn libraries are imported in the code. The code is as follows:

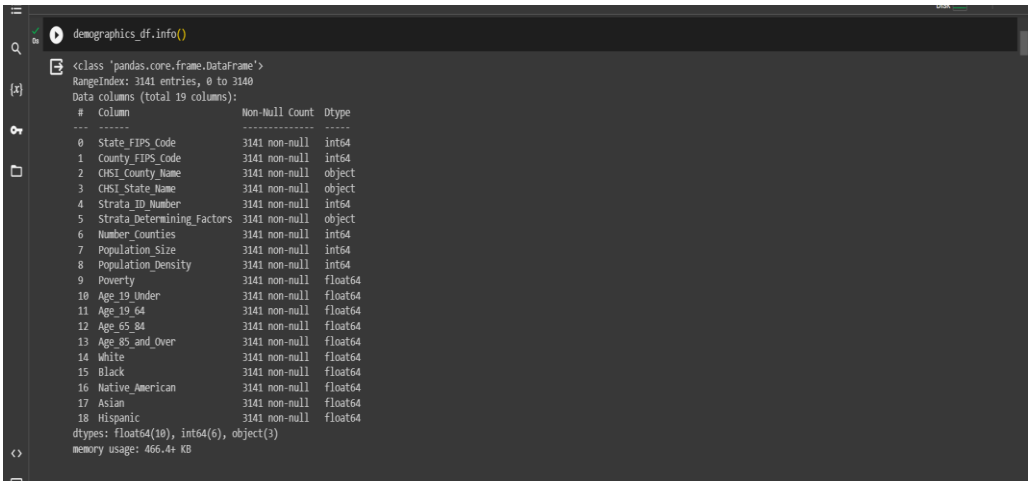


- The above picture also contains a code to view all the columns from the dataset by using columns attribute.

Step 2: Dropping unnecessary columns

- The columns which are not useful are dropped by using drop function and specifying the not useful columns in a string format and mentioning the axis which is 1 for columns in this case.

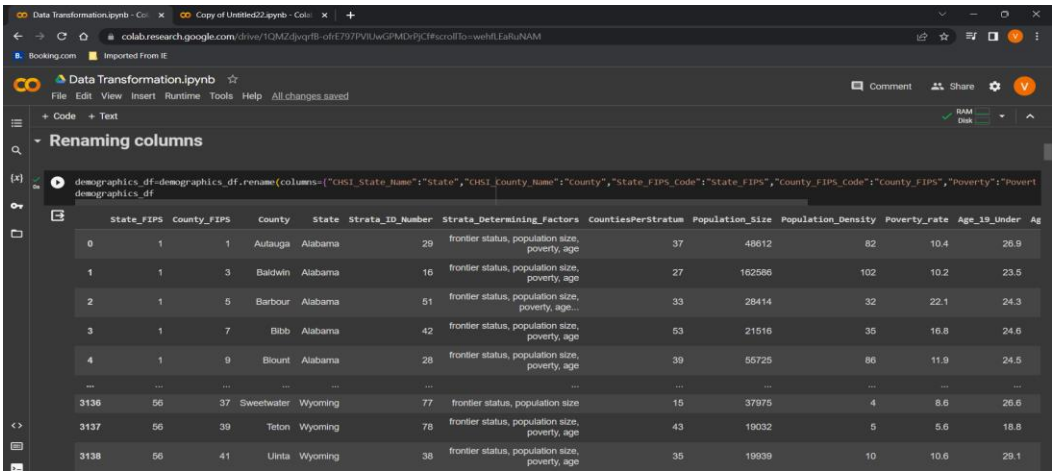
- The output for the above code shows the column names after removing unnecessary columns along with their datatypes which is as follows:



- This output shows the column names, datatype of all the columns and their non-null count after the removal of columns namely index, state abbreviation, min population size, max population size, min population density, max population density, min poverty, max poverty, min age 19 under, max age 19 under, min age 19-64, max age 19-64, min age 65-84, max age 65-84, min age 85 and over, max age 85 and over, min white, max white, min black, max black, min native American, max native American, min Asian, max Asian, min Hispanic, max Hispanic.

Step 3: Renaming the columns

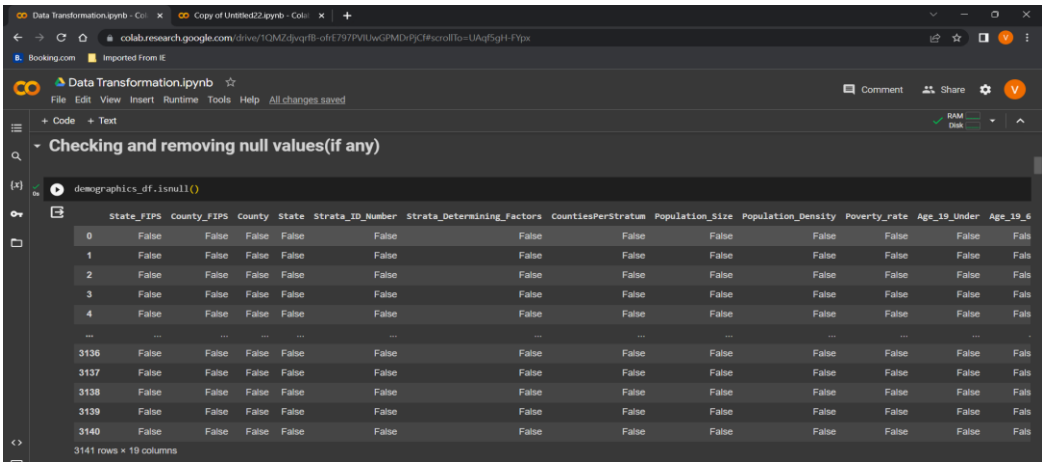
- The column names are renamed as a part of data transformation [5] to useful names by using the below function which uses rename () function and mentioning the column names along with the new names inside the dictionary as strings in key value pairs. The names changed for the columns are poverty-> poverty_rate, CHSI_State_Name-> state, CHSI_County_Name->County, State_FIPS_Code->State_FIPS, County_FIPS_Code->County_FIPS, Number_Counties-> CountiesPerStratum.



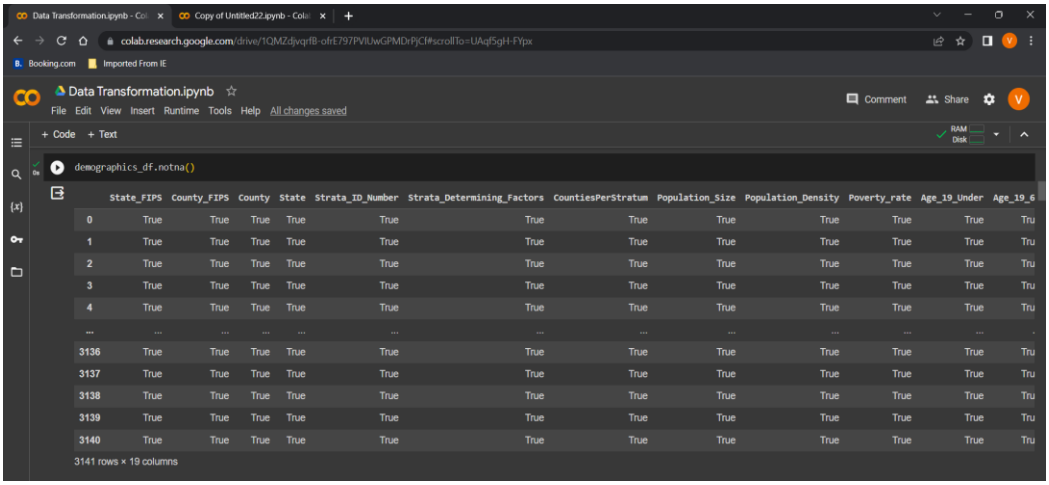
	State_FIPS	County_FIPS	County	State	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under	Ag
0	1	1	Autauga	Alabama	29	frontier status, population size, poverty, age	37	48612	82	10.4	26.9	
1	1	3	Baldwin	Alabama	16	frontier status, population size, poverty, age	27	162586	102	10.2	23.5	
2	1	5	Barbour	Alabama	51	frontier status, population size, poverty, age...	33	28414	32	22.1	24.3	
3	1	7	Bibb	Alabama	42	frontier status, population size, poverty, age	53	21516	35	16.8	24.6	
4	1	9	Blount	Alabama	28	frontier status, population size, poverty, age	39	55725	86	11.9	24.5	
...	...	...	...	...	...	...	...	...	...	...	...	
3136	56	37	Sweetwater	Wyoming	77	frontier status, population size	15	37975	4	8.6	26.6	
3137	56	39	Teton	Wyoming	78	frontier status, population size, poverty, age	43	19032	5	5.6	18.8	
3138	56	41	Uinta	Wyoming	38	frontier status, population size, poverty, age	35	19939	10	10.6	29.1	

Step 4: Checking and removing null values (if any)

- If the dataset contains any null value in any column those must be removed as a part of the data transformation. The loaded data frame is checked for null values by using isna () and notna () functions, represented in the below figures. The loaded dataset does not contain null value if isna function shows false and if notna function shows true. This dataset satisfies both the condition because it does not contain any null or missing value.



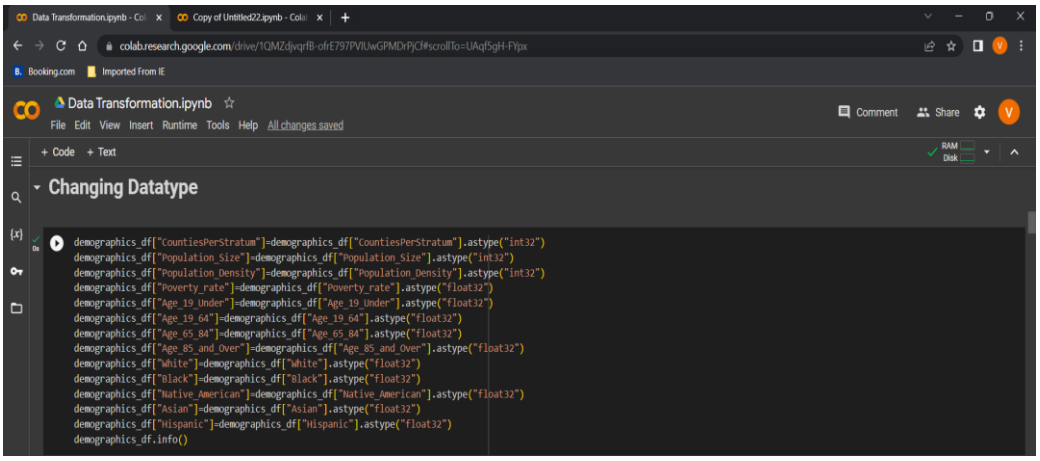
	State_FIPS	County_FIPS	County	State	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under	Age_19_6
0	False	False	False	False	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	...	...	
3136	False	False	False	False	False	False	False	False	False	False	False	Fals
3137	False	False	False	False	False	False	False	False	False	False	False	Fals
3138	False	False	False	False	False	False	False	False	False	False	False	Fals
3139	False	False	False	False	False	False	False	False	False	False	False	Fals
3140	False	False	False	False	False	False	False	False	False	False	False	Fals



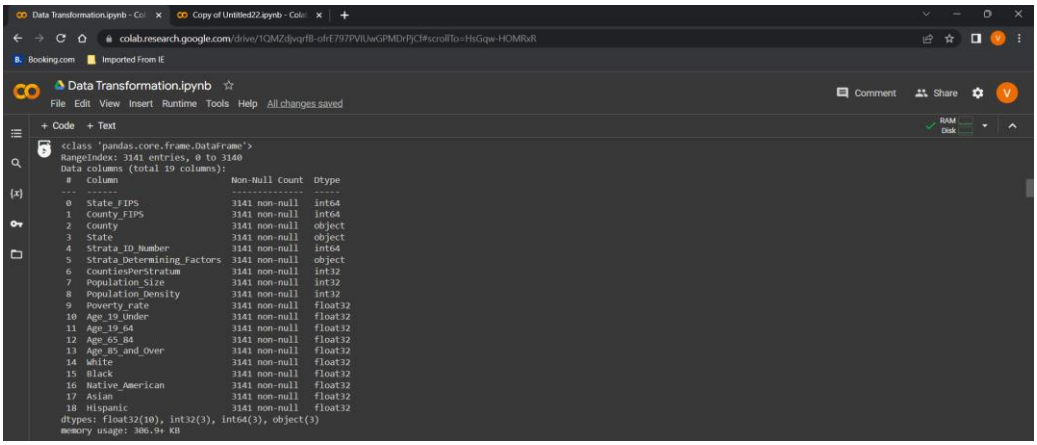
	State_FIPS	County_FIPS	County	State	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under	Age_19_6
0	True	True	True	True	True	True	True	True	True	True	True	Tru
1	True	True	True	True	True	True	True	True	True	True	True	Tru
2	True	True	True	True	True	True	True	True	True	True	True	Tru
3	True	True	True	True	True	True	True	True	True	True	True	Tru
4	True	True	True	True	True	True	True	True	True	True	True	Tru
...	...	...	...	...	...	...	...	...	...	...	...	
3136	True	True	True	True	True	True	True	True	True	True	True	Tru
3137	True	True	True	True	True	True	True	True	True	True	True	Tru
3138	True	True	True	True	True	True	True	True	True	True	True	Tru
3139	True	True	True	True	True	True	True	True	True	True	True	Tru
3140	True	True	True	True	True	True	True	True	True	True	True	Tru

Step 5: Changing datatypes

- The data type of the columns is changed as a part of the data transformation. In this dataset, Counties per stratum, population size, population density columns are down casted from int64 to int32, poverty rate, age 19 under, age 19-64, age 65-84, age 85 and over, White, Black, Native American, Asian, Hispanic columns are also down casted from float 64 to float 32. There are many data type conversions like int to str, str to int, int to float, int to datetime and many more by using two different functions astype () and pd.to_numeric() functions [(3)]. Operations between int32 and floats can sometimes upcast the int32s to int64 unnecessarily. Explicitly converting to int32 prevents this.



- The output of the above code shows the changed data types of the columns which can be viewed by using the info () function as follows:

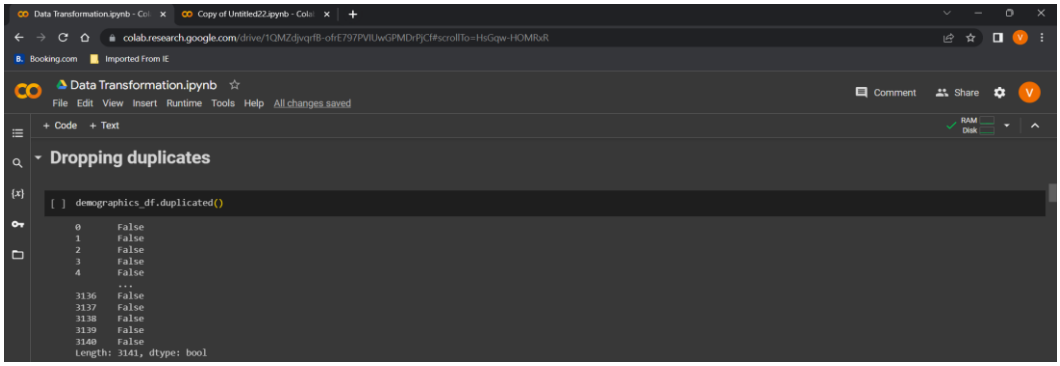


- Before changing the datatypes, the memory is 466kb. After changing down casting the memory dropped to 306kb. In such cases, where the memory problems arise the down casting is a good option.

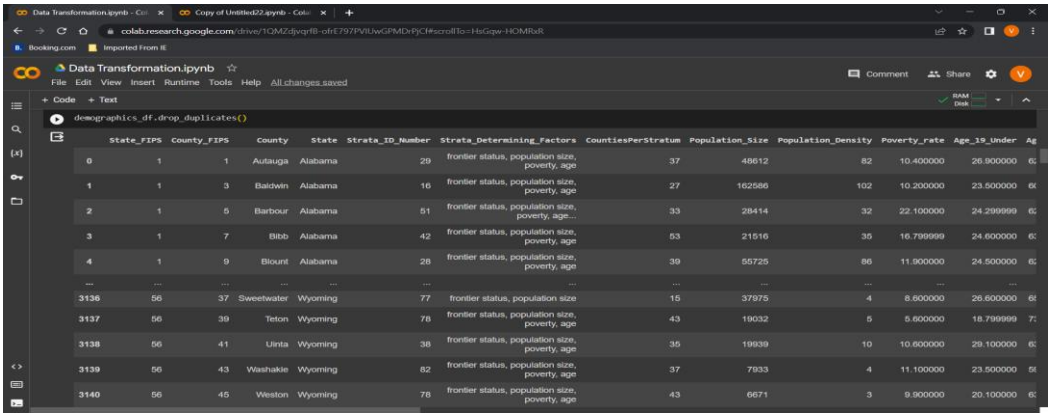
Step 6: Removing duplicates

- Prediction results may be skewed by duplicate data. To produce a more accurate prediction, it is crucial to look for and remove any duplicate rows from columns that should only contain unique values [(4)].

- The data frame is checked for any duplicate values by using duplicated () function. If there are any duplicate values, it returns false. This loaded dataset has no duplicate values so, there is no necessity to use functions to remove the duplicates.

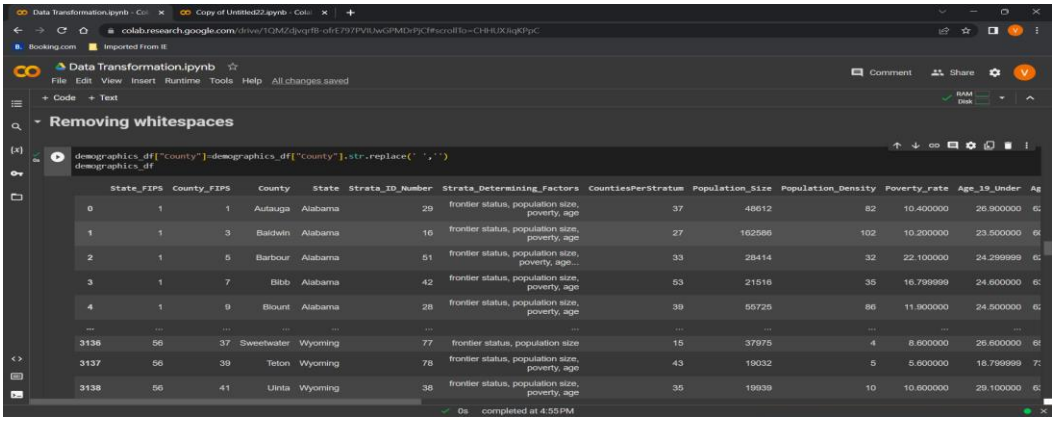


- If there are any duplicates in the data frame, they can be removed by using the drop_duplicates () function.



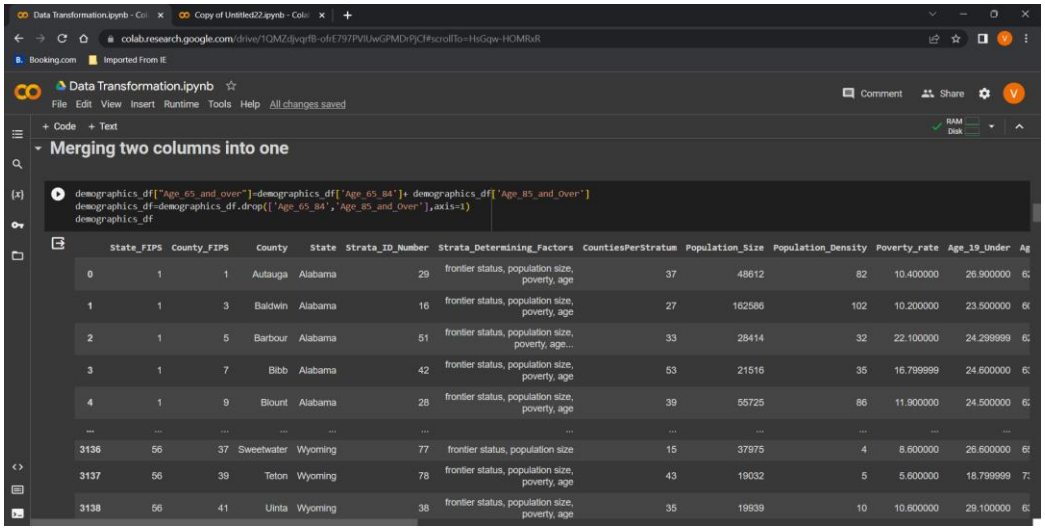
Step 7: Removing whitespaces

- The whitespaces are a form of redundancies in the dataset to be removed in the data cleaning process. If a dataset contains whitespaces or any other unnecessary punctuation marks that causes the dataset to look confusing so, in order to remove redundant data, the below function str.replace( ) [6] is used to replace them. In this data frame to remove the whitespace in the county name column this function is used.



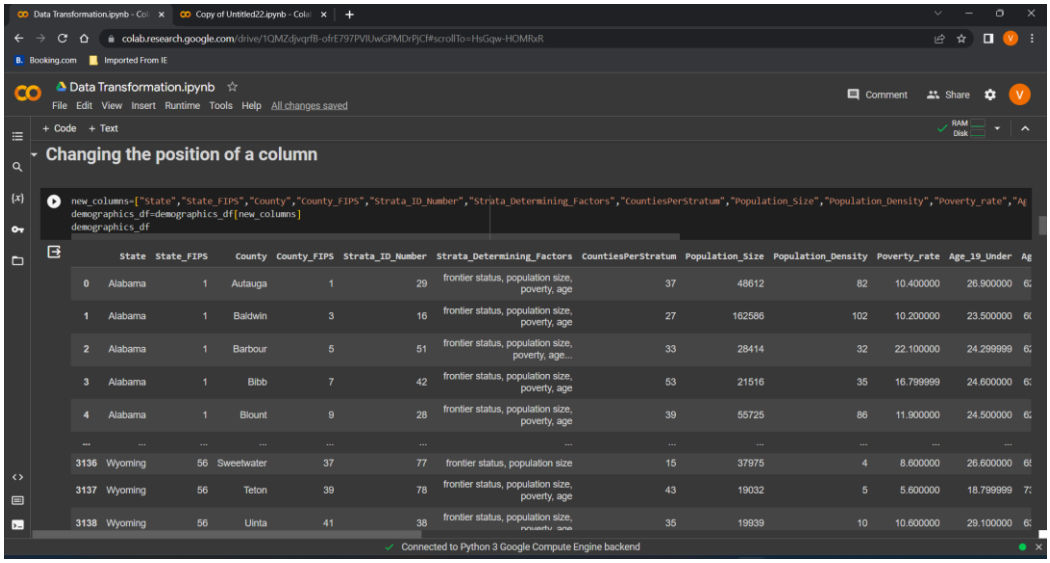
Step 8: Merging two columns into one field

- By using the data cleaning process, two columns which are of the same datatype can be combined into a single column [7]. Then, the previously combined columns must be removed to avoid redundancies. They can be removed by the already used drop() function and mentioning axis=1 (for columns). One must ensure that the two/ multiple columns which are to be combined should be of the same data type. Otherwise, errors are created. In the code below age 65-84 and age 85 and over columns are joined together to form new column: age 65 and over.



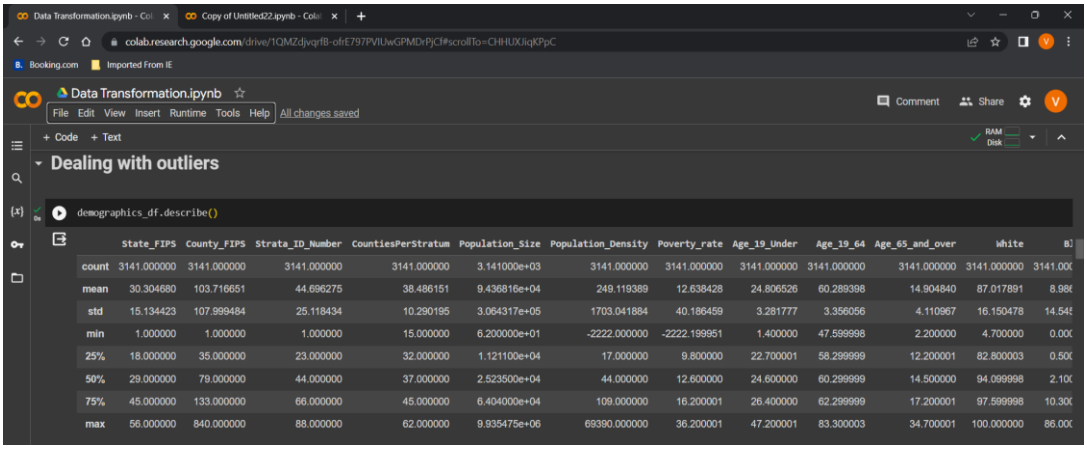
Step 9: Changing the positions of columns

- The position of columns should be changed after the above step to avoid confusion [8]. In the above step, the created new column is by default at the end of the data frame. That should be placed after age group columns so, the position is reindexed by using new columns variable and specifying the order of the columns. In this step, one can arrange the columns as needed.



Step 10: Identifying and Handling outliers

- Outliers are important data points which can sometimes be useful and sometimes useless. They can be capped, that is, replaced with minimum and maximum values. Whether the outliers are to be trimmed or capped depends upon the size of the dataset. Here, the decision is made to remove the outliers. Outliers can be removed by numerous methods and among them, z-score method using data visualization is used for the loaded dataset. It is a statistical measurement that helps us understand how close or far we are from the data set's arithmetic mean [9]. Methods like interquartile range, percentile method, etc. Can also be used.
- The describe() function is used to display the mean, standard deviation, minimum, maximum, 25%, 50%, 75% values for each column.

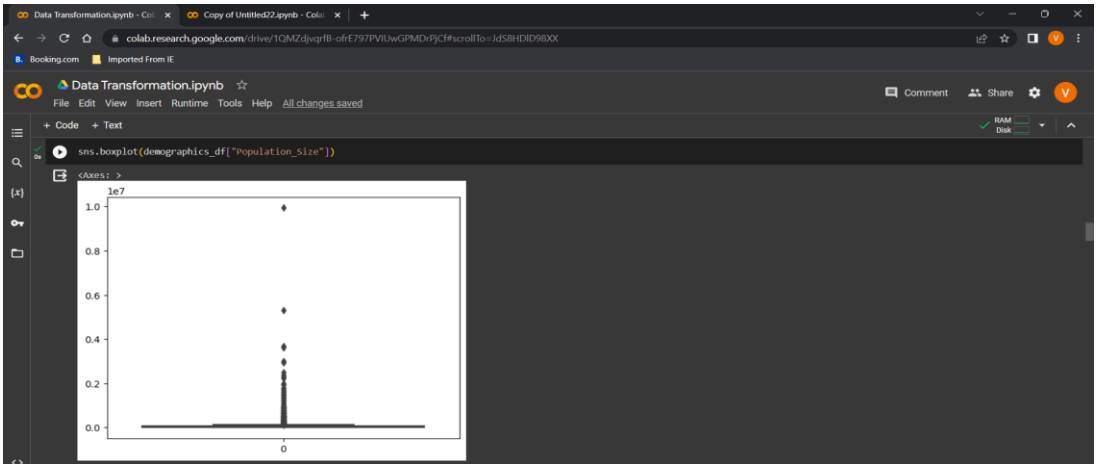


	State_FIPS	County_FIPS	Strata_ID_Number	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under	Age_19_64	Age_65_and_over	White	Bl
count	3141.000000	3141.000000	3141.000000	3141.000000	3.141000e+03	3141.000000	3141.000000	3141.000000	3141.000000	3141.000000	3141.000000	3141.00(
mean	30.304680	103.716651	44.696275	38.486151	9.436816e+04	249.119389	12.638428	24.806526	60.289398	14.904840	87.017891	8.98(
std	15.134423	107.999484	25.118434	10.290195	3.064317e+05	1703.041884	40.186459	3.281777	3.356056	4.110967	16.150478	14.54(
min	1.000000	1.000000	1.000000	15.000000	6.200000e+01	-2222.000000	-2222.199951	1.400000	47.599998	2.200000	4.700000	0.00(
25%	18.000000	35.000000	23.000000	32.000000	1.121100e+04	17.000000	9.800000	22.700001	58.299999	12.200001	82.800003	0.50(
50%	29.000000	79.000000	44.000000	37.000000	2.523500e+04	44.000000	12.600000	24.600000	60.299999	14.500000	94.099998	2.10(
75%	45.000000	133.000000	66.000000	45.000000	6.404000e+04	109.000000	16.200001	26.400000	62.299999	17.200001	97.599998	10.30(
max	56.000000	840.000000	88.000000	62.000000	9.935475e+06	69390.000000	36.200001	47.200001	83.300003	34.700001	100.000000	86.00(

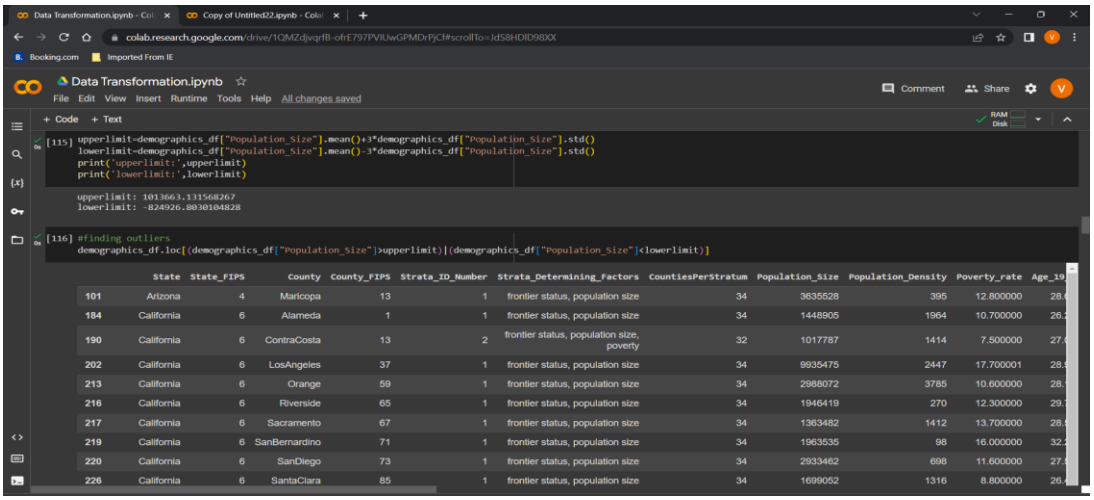
- The outliers are identified for each separate column by using z-score and visualization methods in the following way:

o **Identifying and removing outliers from the 'Population size' column:**

- Box plots are the best way of representing the outliers lying in a dataset, that is the reason box plots are chosen to represent the outliers out of so many visualization methods.
- A box plot is plotted for the population size column using seaborn library. This plot indicates the 4 data points outside the whiskers as outliers.

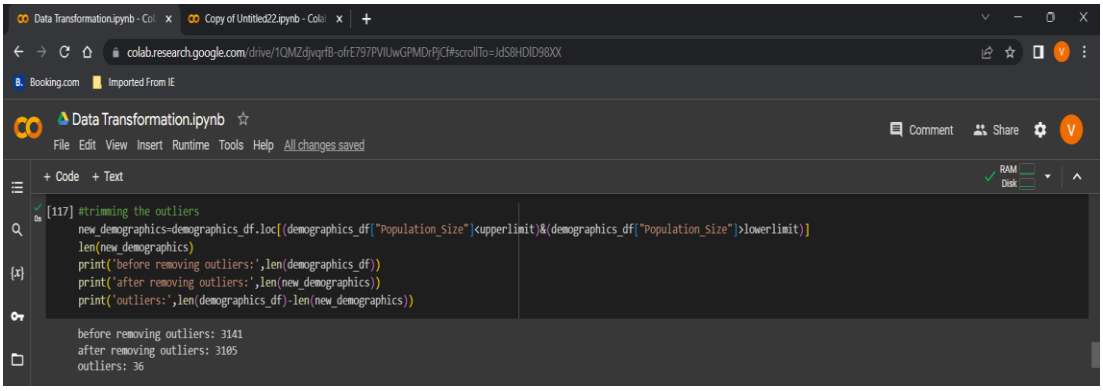


- In the z-score method, "any value that is greater than **+3** and less than **-3** standard deviation is an outlier value" principle is followed [9]. To detect outliers in pandas using Z-score, set a limit for any values outside of +3/-3 standard deviations from the mean to be considered outliers.
- The upper limit is the addition of mean and 3 times the standard deviation of the specific column and the lower limit is the subtraction of 3 times the standard deviation of the column from the mean [10]. The values falling outside the upper and lower limit range are considered as outliers. Below is another function loc to find the outliers where the condition satisfies the population size column greater than upper limit or the population size column less than the lower limit to get the outliers from that column.

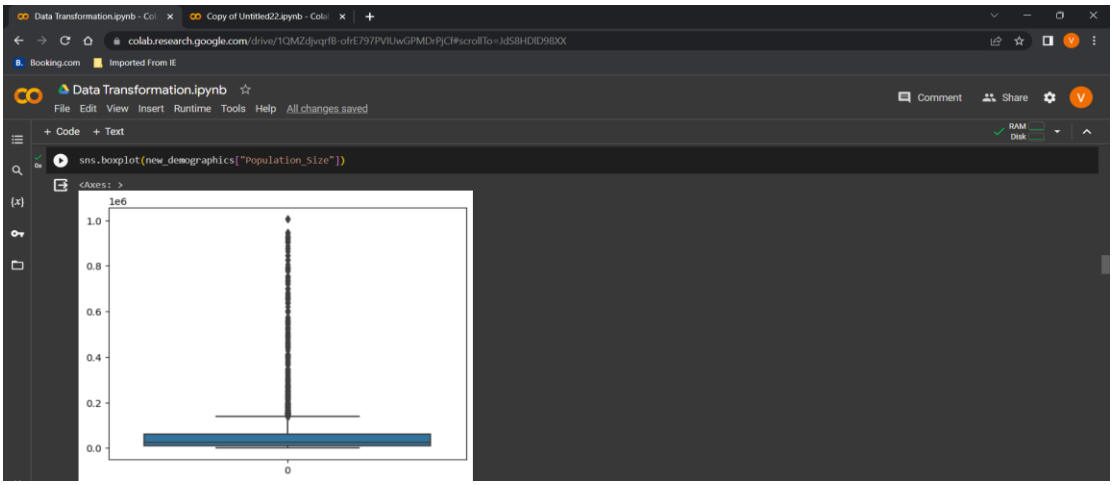


	State	State_FIPS	County	County_FIPS	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19
101	Arizona	4	Maricopa	13	1	frontier status, population size	34	3635528	395	12.800000	28.4
184	California	6	Alameda	1	1	frontier status, population size	34	1448905	1964	10.700000	26.2
190	California	6	ContraCosta	13	2	frontier status, population size, poverty	32	1017787	1414	7.500000	27.4
202	California	6	LosAngeles	37	1	frontier status, population size	34	9935475	2447	17.700001	28.5
213	California	6	Orange	59	1	frontier status, population size	34	2988072	3785	10.600000	28.1
216	California	6	Riverside	65	1	frontier status, population size	34	1946419	270	12.300000	29.1
217	California	6	Sacramento	67	1	frontier status, population size	34	1363482	1412	13.700000	28.5
219	California	6	SanBernardino	71	1	frontier status, population size	34	1963535	98	16.000000	32.2
220	California	6	SanDiego	73	1	frontier status, population size	34	2933462	698	11.600000	27.5
226	California	6	SantaClara	85	1	frontier status, population size	34	1699052	1316	8.800000	26.4

- The output for the above code gives the outliers in the specific rows for that column. Below is another code to represent the number of outliers in the population size column by modifying the above used function that is replacing the or condition with and condition and assigning it to a new variable new demographics. This represents the number of rows before removing the outliers and the number of rows after removing the outliers for this column.



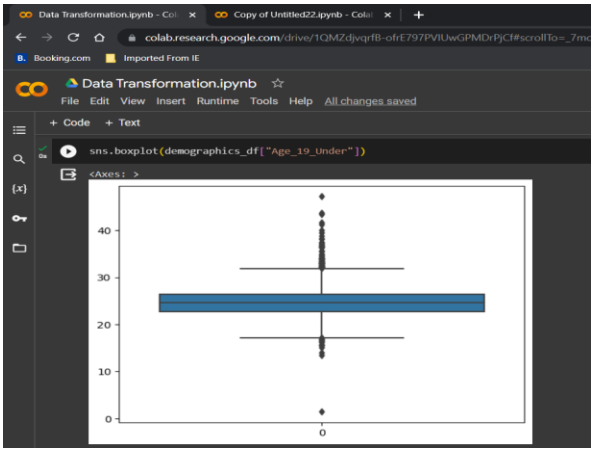
- Once again, the bar plot is plotted to check whether any data points are lying outside the whiskers. Comparing both the bar plots, that is the loaded data frame and the new data frame, one can tell that the outliers have been removed.

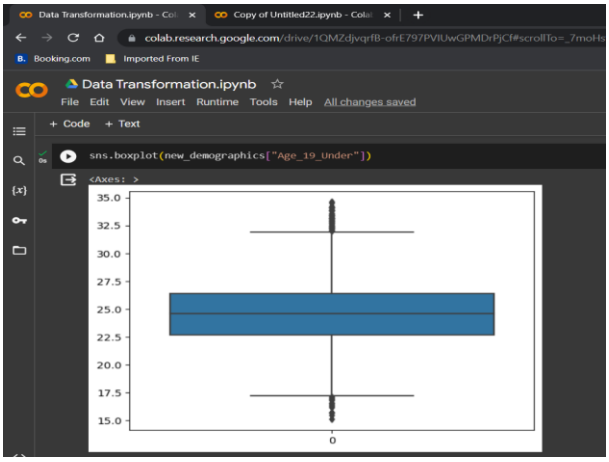


- The same procedure is followed for all the useful numeric columns to remove the outliers from the data frame by using the same z-score method which is mentioned in the appendix below.

Comparing the boxplots before and after removal of outliers:

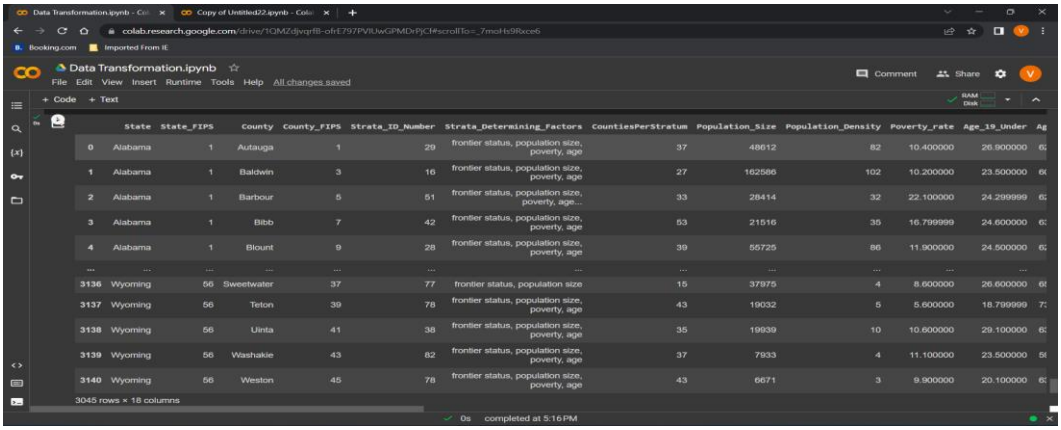
- **Box plots for 'Age 19 under' column before and after removal of outliers**





- Before removing outliers, the plot indicated the data points outside the whiskers which are potential outliers and after removing them, the plot is clear with no data points outside the whiskers and has a median value at 24.5, first quartile at 23 and third quartile at 26.
- The same procedure is followed for all the columns and the plots are compared to verify that the outliers are completely removed.

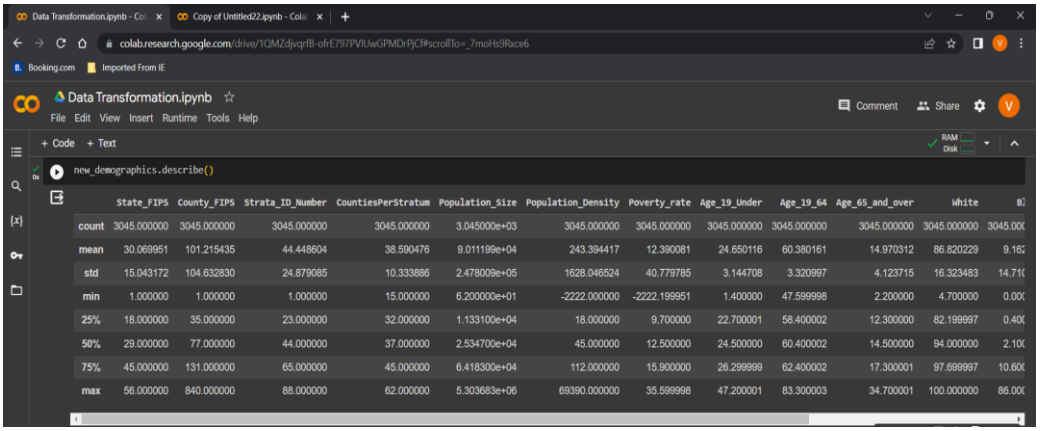
After removing all the outliers, by following the above-mentioned path to identify and remove outliers for all the columns the cleaned dataset without outliers is produced. The new data frame produced contains 3045 rows which when compared with the loaded data frame are free of outliers.



	State	State_FIPS	County	County_FIPS	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under	Ag
0	Alabama	1	Autauga	1	29	frontier status, population size, poverty, age	37	48612	82	10.400000	26.900000	6
1	Alabama	1	Baldwin	3	16	frontier status, population size, poverty, age	27	162586	102	10.200000	23.500000	6
2	Alabama	1	Barbour	5	51	frontier status, population size, poverty, age...	33	28414	32	22.100000	24.299999	6
3	Alabama	1	Bibb	7	42	frontier status, population size, poverty, age	53	21516	35	16.799999	24.600000	6
4	Alabama	1	Blount	9	28	frontier status, population size, poverty, age	39	55725	86	11.900000	24.500000	6
...	...	...	...	...	...	...	...	...	...	...	...	...
3136	Wyoming	56	Sweetwater	37	77	frontier status, population size	15	37975	4	8.600000	26.600000	6
3137	Wyoming	56	Teton	39	78	frontier status, population size, poverty, age	43	19032	5	5.600000	18.799999	7
3138	Wyoming	56	Uinta	41	38	frontier status, population size, poverty, age	35	19939	10	10.600000	29.100000	6
3139	Wyoming	56	Washakie	43	82	frontier status, population size, poverty, age	37	7933	4	11.100000	23.500000	5
3140	Wyoming	56	Weston	45	78	frontier status, population size, poverty, age	43	6671	3	9.900000	20.100000	6

3045 rows × 18 columns

The describe () function is used once again to check the variations in the mean, standard deviation, minimum, maximum, 25%, 50%, 75% values for the new data frame from that of the previously loaded data frame.



	State_FIPS	County_FIPS	Strata_ID_Number	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under	Age_19_64	Age_65_and_over	White	Bl
count	3045.000000	3045.000000	3045.000000	3045.000000	3.045000e+03	3045.000000	3045.000000	3045.000000	3045.000000	3045.000000	3045.000000	3045.00
mean	30.069951	101.215435	44.448604	38.590476	9.011199e+04	243.394417	12.390081	24.650116	60.380161	14.970312	86.820229	9.16
std	15.043172	104.632830	24.879085	10.333886	2.478009e+05	1628.046524	40.779785	3.144708	3.320997	4.123715	16.323483	14.71
min	1.000000	1.000000	1.000000	15.000000	6.200000e+01	-2222.000000	-2222.199951	1.400000	47.599998	2.200000	4.700000	0.00
25%	18.000000	35.000000	23.000000	32.000000	1.133100e+04	18.000000	9.700000	22.700001	58.400002	12.300000	82.199997	0.40
50%	29.000000	77.000000	44.000000	37.000000	2.534700e+04	45.000000	12.500000	24.500000	60.400002	14.500000	94.000000	2.10
75%	45.000000	131.000000	65.000000	45.000000	6.418300e+04	112.000000	15.900000	26.299999	62.400002	17.300001	97.699997	10.60
max	56.000000	840.000000	88.000000	62.000000	5.303683e+06	69390.000000	35.599998	47.200001	83.300003	34.700001	100.000000	86.00

Repository link to access the cleaned dataset: the GitHub repository publicly accessible link to the new cleaned dataset (after applying all the above-mentioned data cleaning tools) is mentioned below in the references [(11)].

Conclusion:

The demographics dataset was cleaned and made ready for analysis through extensive data transformation techniques. The steps included loading the data into a Pandas data frame, examining the columns and data types, removing redundant columns, renaming columns for clarity, checking for and handling missing values, down casting data types to reduce memory usage, merging relevant columns, rearranging columns, removing duplicates, standardizing text data, and identifying and removing outliers. Using the z-score method, visualizations such as boxplots were utilized to systematically identify outliers across several numerical columns. The dataset was reduced from 3141 to 3045 rows after the data transformations.

The data transformation process used here is an example of the essential preprocessing needed to draw valuable insights from raw datasets. Clean, high-quality data serves as the basis for further analysis and modeling, which can improve accuracy and have greater impact on data-driven business strategy and decision making.

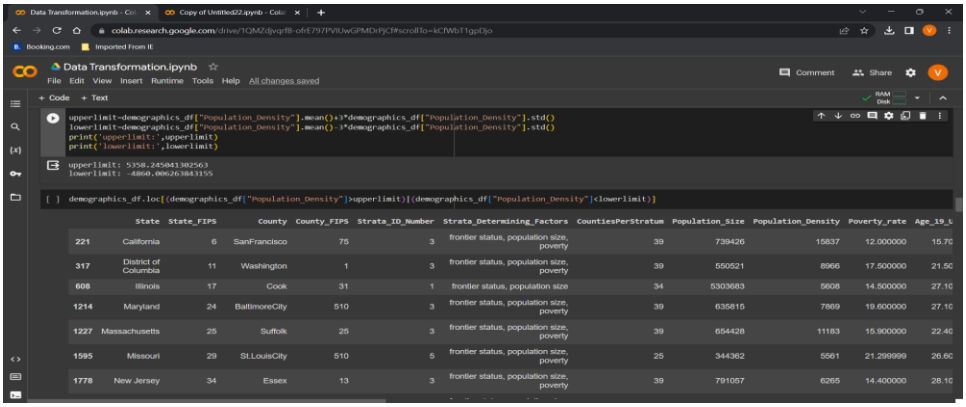
References:

1. What is data transformation?. TIBCO Software. (n.d.). https://www.tibco.com/reference-center/what-is-data-transformation#:~:text=Data%20transformation%20is%20the%20process,that%20of%20the%20destination%20system.
2. Demographic trends and Health Outcomes in the U.S. Kaggle. Accessed October 19, 2023. https://www.kaggle.com/datasets/thedevastator/demographic-trends-and-health-outcomes-in-the-u .
3. Sekan, F. (2023, February 3). 3 ways how to update data type of columns in Pandas. Medium. https://medium.com/@filip.sekan/3-ways-how-to-update-data-type-of-columns-in-pandas-97ddb5f32ae4
4. Data Cleaning - remove duplicates. Mage. (n.d.). https://www.mage.ai/blog/data-cleaning-remove-duplicates
5. Pandas.dataframe.rename#. pandas.DataFrame.rename - pandas 2.1.2 documentation. (n.d.). https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rename.html
6. How to remove space from columns in pandas a data scientists guide. Saturn Cloud Blog. (2023, October 25). https://saturncloud.io/blog/how-to-remove-space-from-columns-in-pandas-a-data-scientists-guide/#:~:text=strip()%20method-,The%20str.,column%20names%20or%20column%20values.
7. Concatenate two columns into a single column in Pandas dataframe. Concatenate two columns into a single column in pandas dataframe. (n.d.). https://net-informations.com/ds/pd/comb.htm
8. (NNK), N. (2023, October 5). Pandas - how to change position of a column. Spark By {Examples}. https://sparkbyexamples.com/pandas/pandas-change-position-of-a-column/
9. Zafar, A. (2022, February 25). Handling outliers in Pandas. Medium. https://medium.com/@arsalan_zafar/handling-outliers-in-pandas-5cd872eef508
10. YouTube. (2022). YouTube. Retrieved November 5, 2023, from https://www.youtube.com/watch?v=Cw2IvmWRcXs.
11. https://github.com/vpallapolu/ORES/blob/main/cleaned_demographics.csv

Appendix:

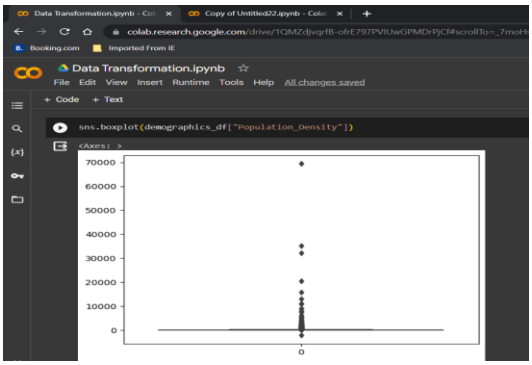
- **Handling Outliers for population density column**

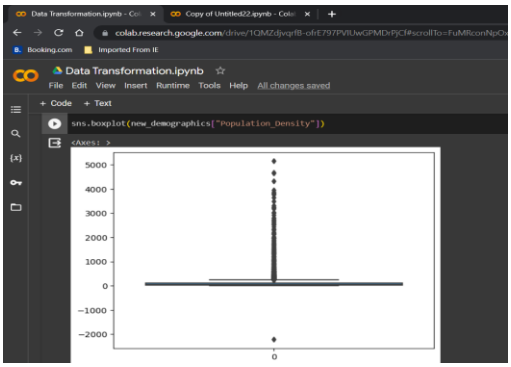
The code below in the picture calculates the upper and lower limits for “Population density” column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Population density' column values are outside the limits. These are outlier rows.



The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Population density column to create a new dataframe without outliers.

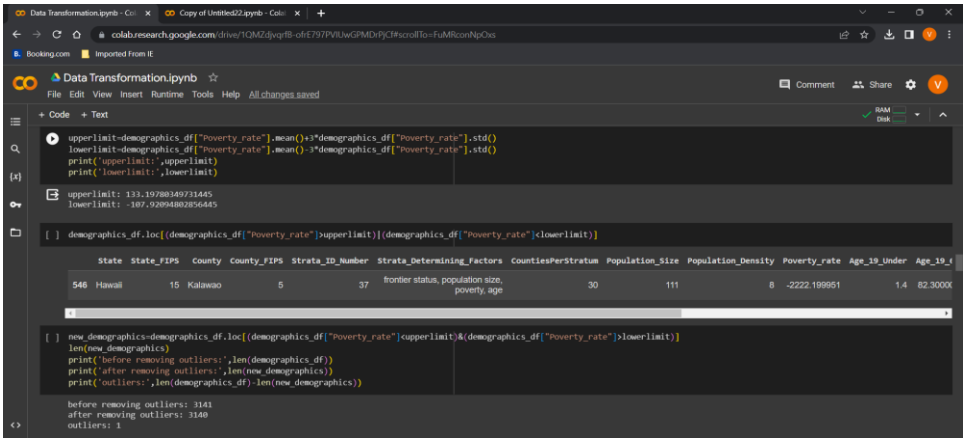
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Age 19 under' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Age 19 under' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.



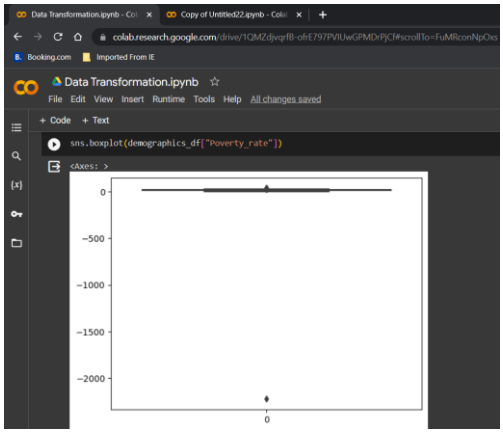


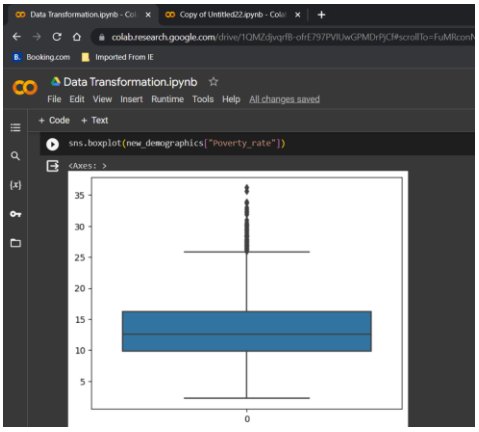
- **Handling outliers for poverty rate column**

The code below in the picture calculates the upper and lower limits for “Poverty rate” column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Poverty rate' column values are outside the limits. These are outlier rows. The code in the same picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Age 19 under column to create a new dataframe without outliers.



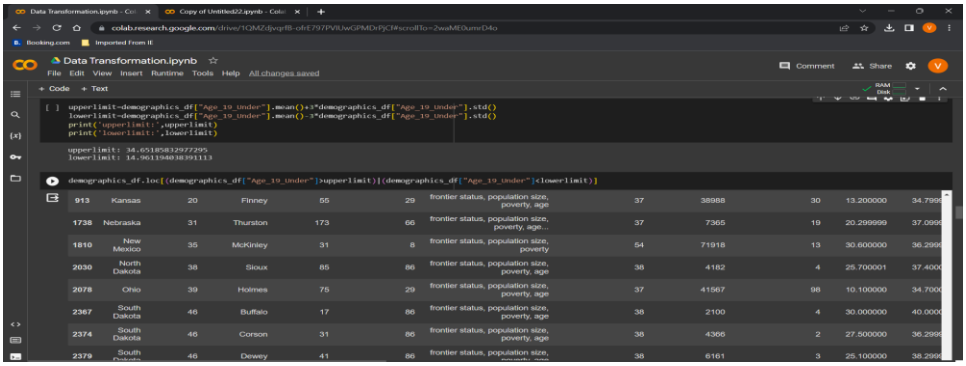
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Age 19 under' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Age 19 under' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.





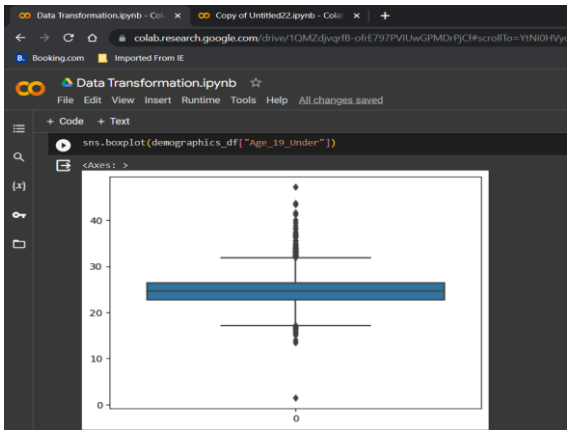
- **Handling outliers for age 19 under column**

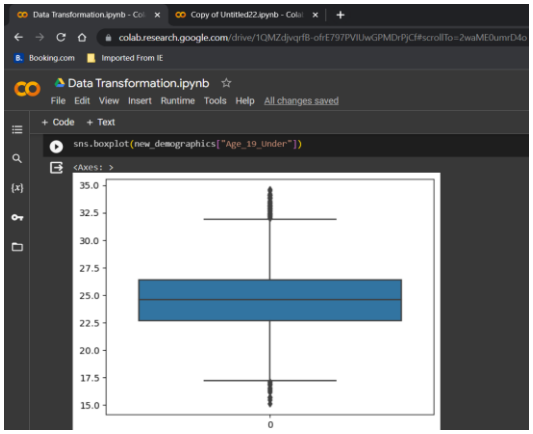
The code below in the picture calculates the upper and lower limits for “Age 19 under” column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Age 19 under' column values are outside the limits. These are outlier rows.



The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Age 19 under column to create a new dataframe without outliers.

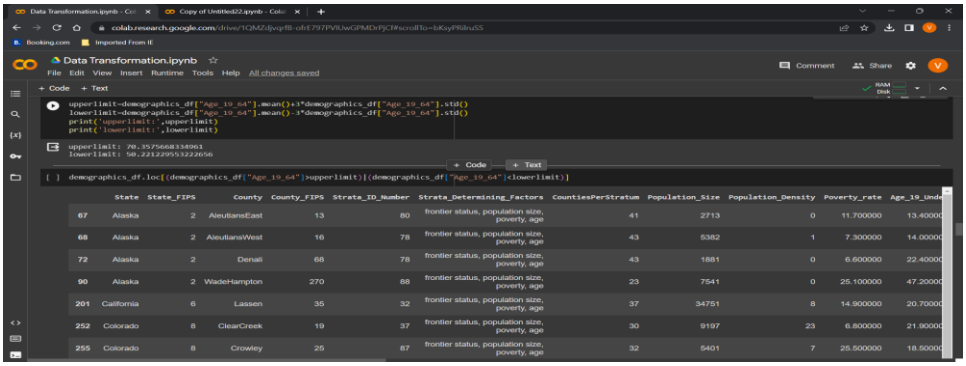
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Age 19 under' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Age 19 under' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.



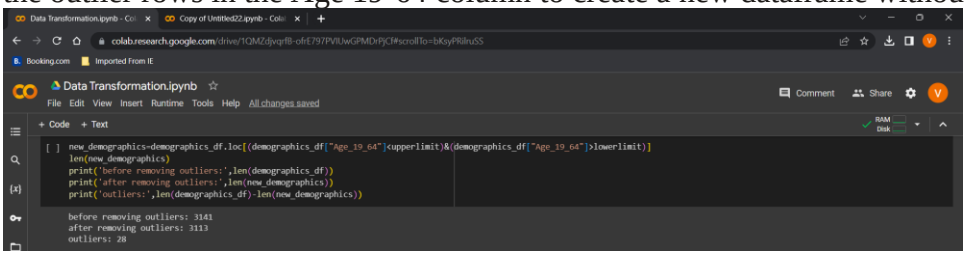


- **Handling outliers for age 19-64 column**

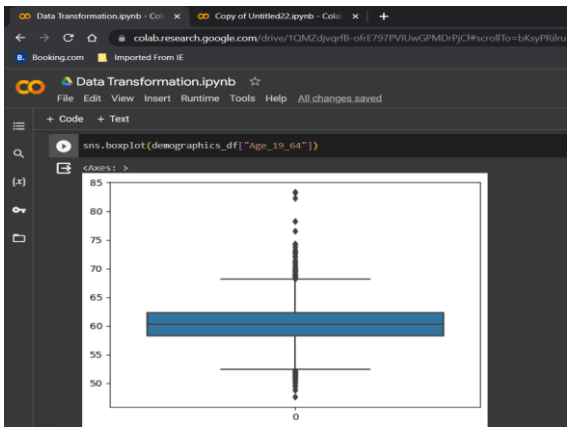
The code below in the picture calculates the upper and lower limits for “Age 19-64” column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Age 19-64' column values are outside the limits. These are outlier rows.

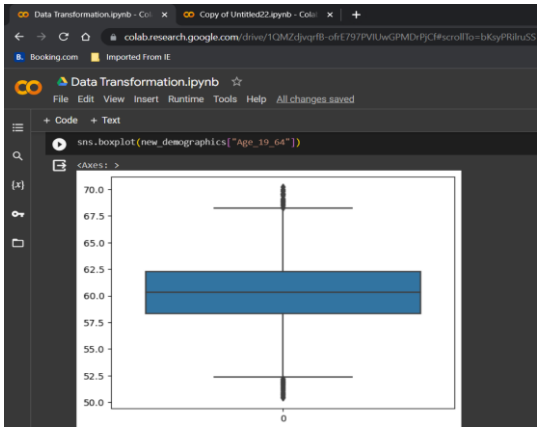


The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Age 19-64 column to create a new dataframe without outliers.



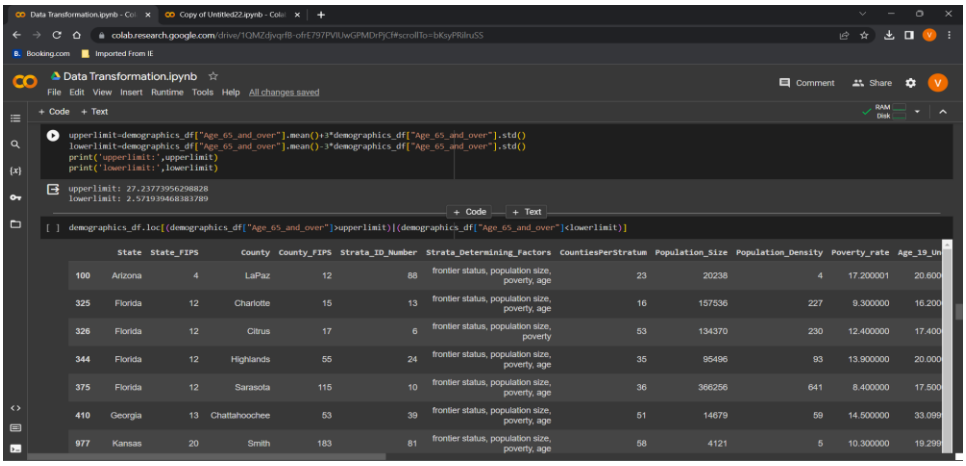
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Age 19-64' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Age 19-64' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.





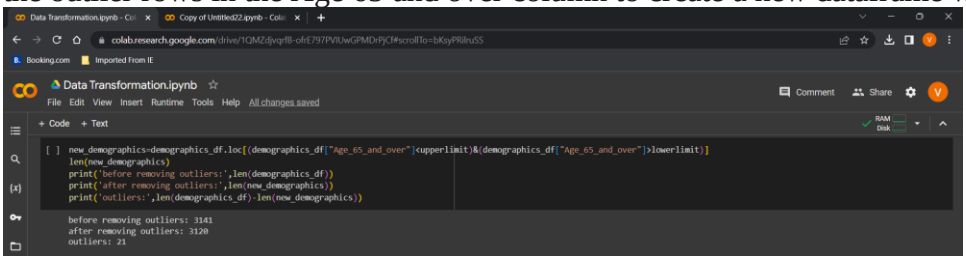
- **Handling outliers for age 65 and over column**

The code below in the picture calculates the upper and lower limits for “Age 65 and over” column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Age 65 and over' column values are outside the limits. These are outlier rows.

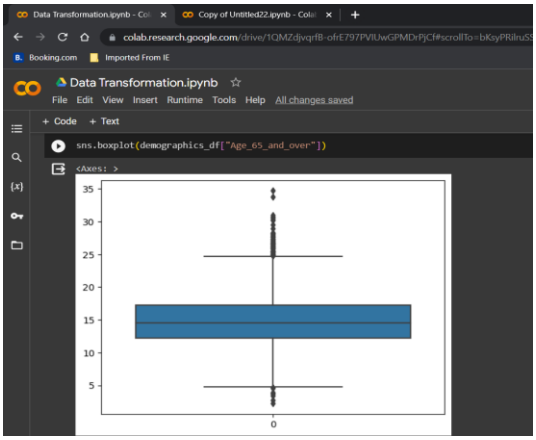


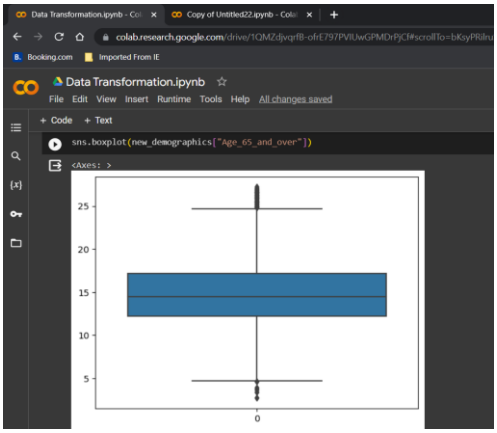
	State	State_FIPS	County	County_FIPS	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Un
100	Arizona	4	LaPaz	12	88	frontier status, population size, poverty, age	23	20238	4	17.200001	20.600
325	Florida	12	Charlotte	15	13	frontier status, population size, poverty, age	16	157536	227	9.300000	16.200
326	Florida	12	Citrus	17	6	frontier status, population size, poverty	53	134370	230	12.400000	17.400
344	Florida	12	Highlands	55	24	frontier status, population size, poverty, age	35	95496	93	13.900000	20.000
375	Florida	12	Sarasota	115	10	frontier status, population size, poverty, age	36	366256	641	8.400000	17.500
410	Georgia	13	Chattahoochee	53	39	frontier status, population size, poverty, age	51	14679	59	14.500000	33.099
977	Kansas	20	Smith	183	81	frontier status, population size, poverty, age	58	4121	5	10.300000	19.299

The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Age 65 and over column to create a new dataframe without outliers.



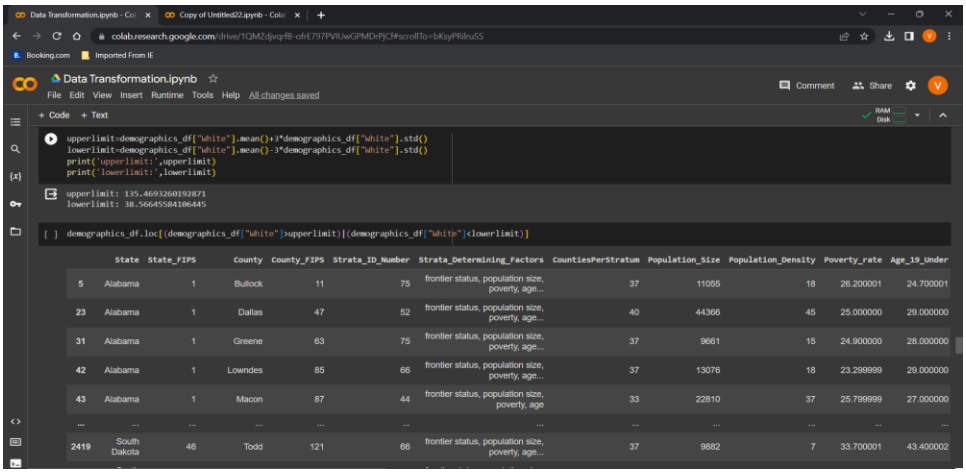
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Age 65 and over' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Age 65 and over' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.





- **Handling outliers for White column**

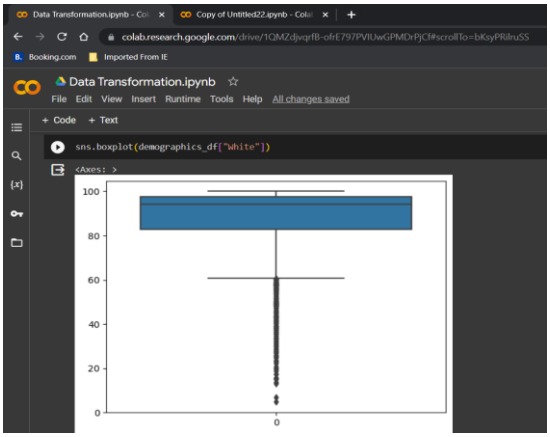
The code below in the picture calculates the upper and lower limits for “White” column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'White' column values are outside the limits. These are outlier rows.

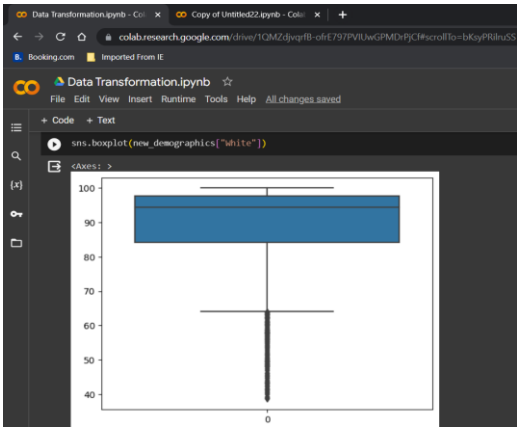


	State	State_FIPS	County	County_FIPS	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under
5	Alabama	1	Bullock	11	75	frontier status, population size, poverty, age...	37	11055	18	26.200001	24.700001
23	Alabama	1	Dallas	47	52	frontier status, population size, poverty, age...	40	44366	45	25.000000	29.000000
31	Alabama	1	Greene	63	75	frontier status, population size, poverty, age...	37	9661	15	24.900000	28.000000
42	Alabama	1	Lowndes	85	66	frontier status, population size, poverty, age...	37	13076	18	23.299999	29.000000
43	Alabama	1	Macon	87	44	frontier status, population size, poverty, age	33	22810	37	25.799999	27.000000
...	...	...	...	...	...	...	...	...	...	...	...
2419	South Dakota	46	Todd	121	66	frontier status, population size, poverty, age...	37	9882	7	33.700001	43.400002

The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the White column to create a new dataframe without outliers.

The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'White' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'White' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.

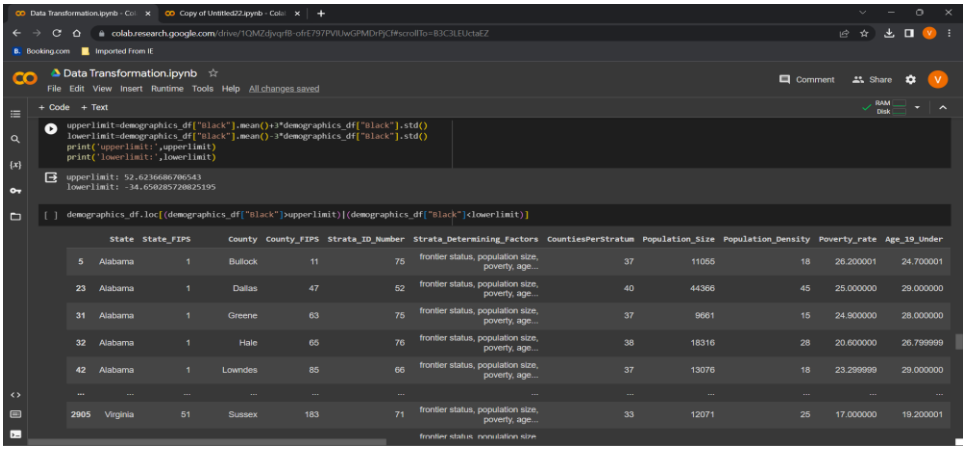




- **Handling outliers for Black column**

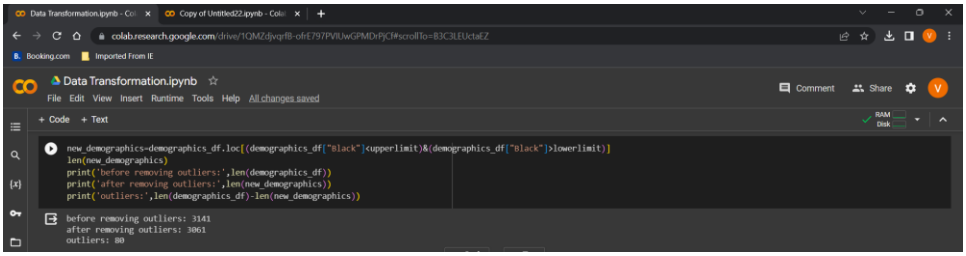
The code below in the picture calculates the upper and lower limits for Black column using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers.

demographics_df.loc[] function Identifies rows where 'Black' column values are outside the limits. These are outlier rows.

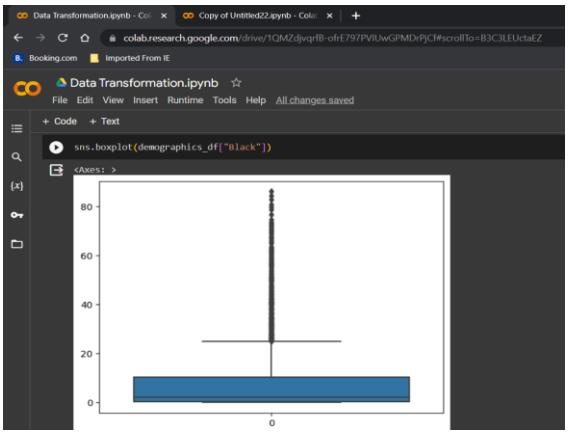


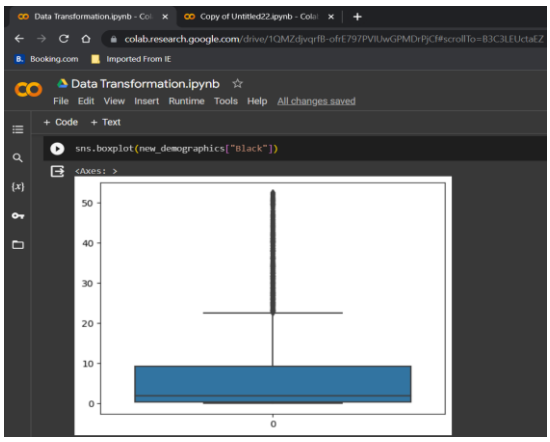
	State	State_FIPS	County	County_FIPS	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under
5	Alabama	1	Bullock	11	75	frontier status, population size, poverty, age...	37	11055	18	26.200001	24.700001
23	Alabama	1	Dallas	47	52	frontier status, population size, poverty, age...	40	44366	45	25.000000	29.000000
31	Alabama	1	Greene	63	75	frontier status, population size, poverty, age...	37	9661	15	24.900000	28.000000
32	Alabama	1	Hale	65	76	frontier status, population size, poverty, age...	38	18316	28	20.600000	26.799999
42	Alabama	1	Lowndes	85	66	frontier status, population size, poverty, age...	37	13076	18	23.299999	29.000000
...	...	...	...	...	...	...	...	...	...	...	...
2905	Virginia	51	Sussex	183	71	frontier status, population size, poverty, age...	33	12071	25	17.000000	19.200001

The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Black column to create a new dataframe without outliers.



The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Black' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Black' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.

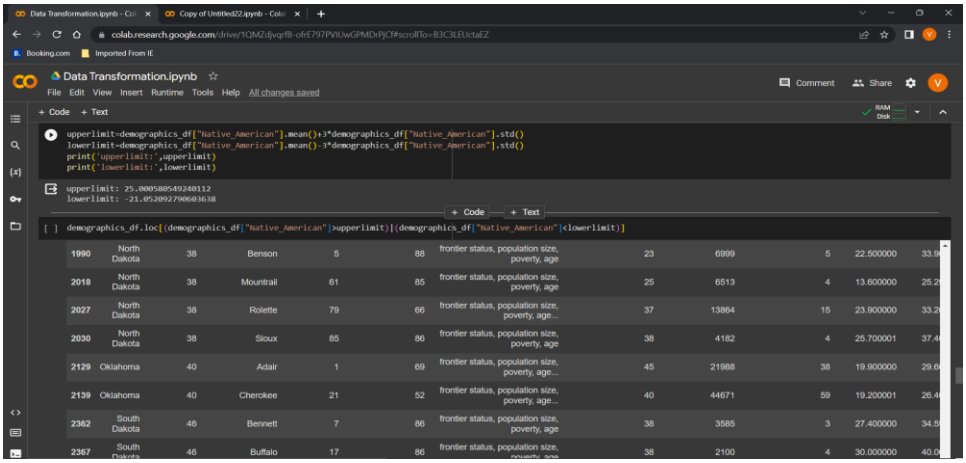




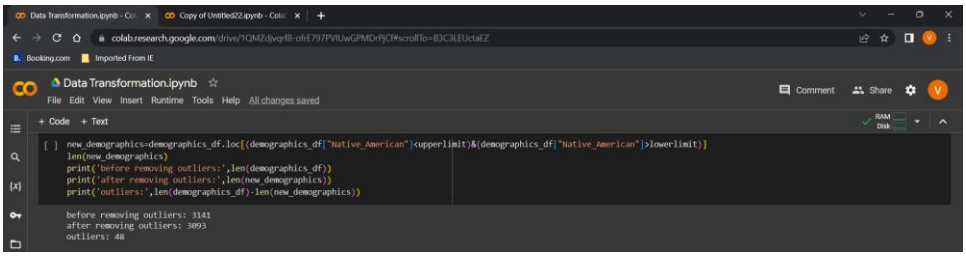
- **Handling outliers for Native American column**

The code below in the picture calculates the upper and lower limits using mean +/- 3 standard deviations by using z-score method. This gives a threshold range to define outliers.

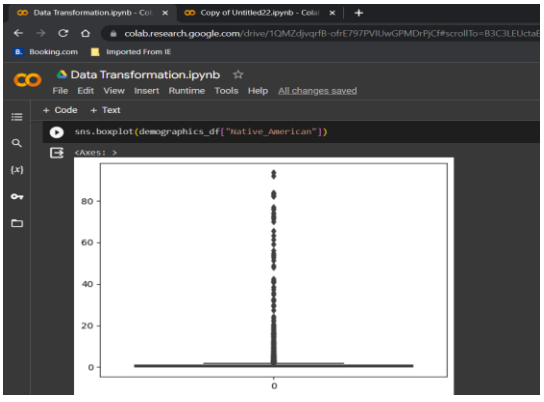
demographics_df.loc[] function Identifies rows where 'Native American' column values are outside the limits. These are outlier rows.

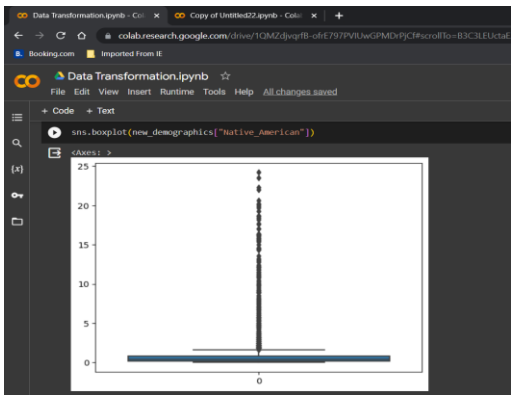


The code in the below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows in the Native American column to create a new dataframe without outliers.



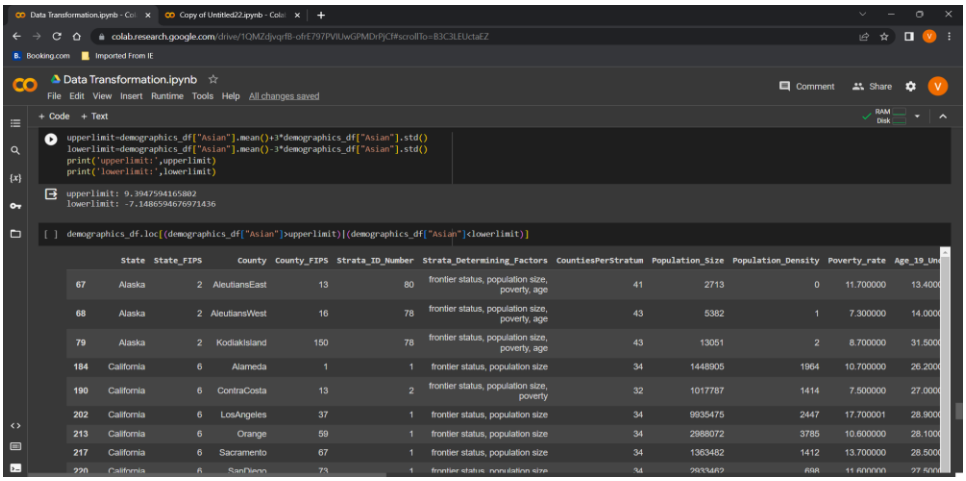
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Native American' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Native American' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.





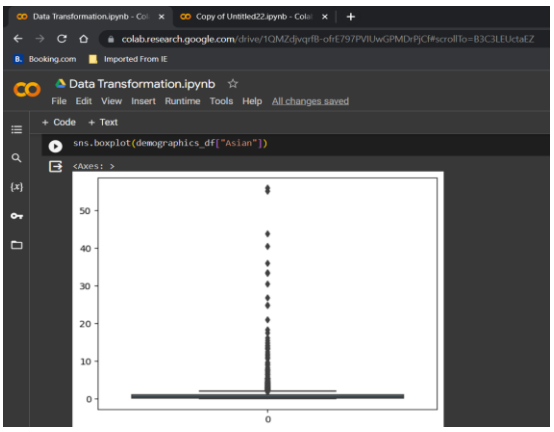
- **Handling outliers for Asian column**

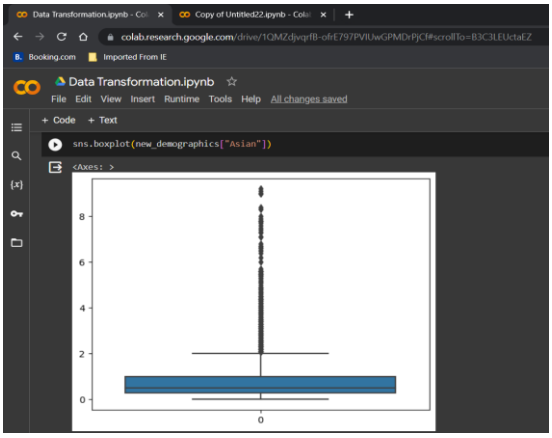
The code below calculates the upper and lower limits using mean +/- 3 standard deviations. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Asian' column values are outside the limits. These are outlier rows.



new_demographics = demographics_df.drop() function Drops the outlier rows to create a new dataframe without outliers.

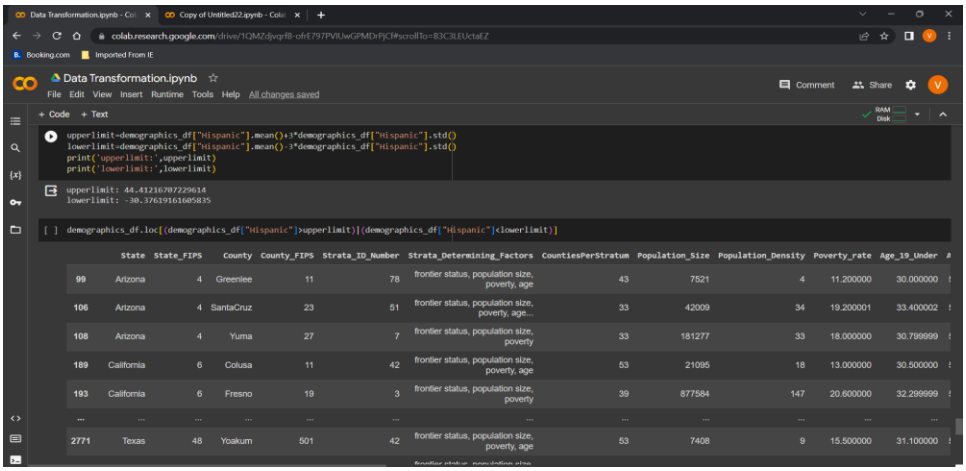
The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Asian' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Asian' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.





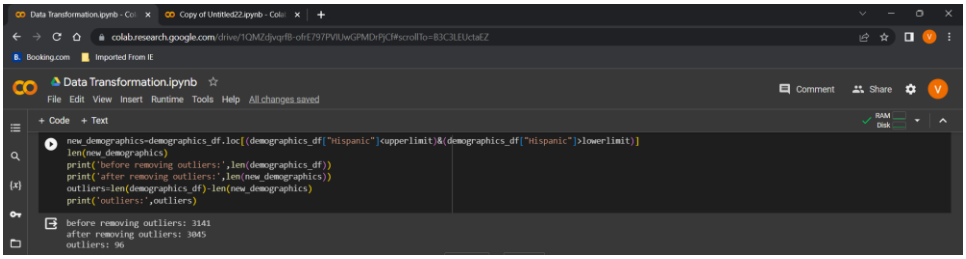
- **Handling outliers for Hispanic column**

The below picture indicates the code to Calculate upper and lower limits of ‘Hispanic’ column using mean +/- 3 standard deviations. This gives a threshold range to define outliers. demographics_df.loc[] function Identifies rows where 'Hispanic' column values are outside the limits. These are outlier rows.



	State	State_FIPS	County	County_FIPS	Strata_ID_Number	Strata_Determining_Factors	CountiesPerStratum	Population_Size	Population_Density	Poverty_rate	Age_19_Under
99	Arizona	4	Greenlee	11	78	frontier status, population size, poverty, age	43	7521	4	11.200000	30.000000
106	Arizona	4	SantaCruz	23	51	frontier status, population size, poverty, age...	33	42009	34	19.200001	33.400002
108	Arizona	4	Yuma	27	7	frontier status, population size, poverty	33	181277	33	18.000000	30.799999
189	California	6	Colusa	11	42	frontier status, population size, poverty, age	53	21095	18	13.000000	30.500000
193	California	6	Fresno	19	3	frontier status, population size, poverty	39	877584	147	20.600000	32.299999
...	...	...	...	...	...	...	...	...	...	...	...
2771	Texas	48	Yoakum	501	42	frontier status, population size, poverty, age	53	7408	9	15.500000	31.100000

The below picture uses new_demographics = demographics_df.drop() function to drop the outlier rows to create a new dataframe without outliers.



The left side picture depicts a code to Generate a boxplot to visually identify potential outliers in the 'Hispanic' column while the right-side picture depicts Boxplot on new dataframe to verify outliers have been removed. This code uses the z-score method to systematically identify outlier values outside 3 standard deviations from the mean in the 'Hispanic' column. It drops these outliers to clean the data for further analysis. The boxplots before and after validate the outliers were removed.