

Data Transformation with Python or R

To my peer Data Scientist,

In this, I have done data transformations with Python with my data i.e. us_county.

Introduction:

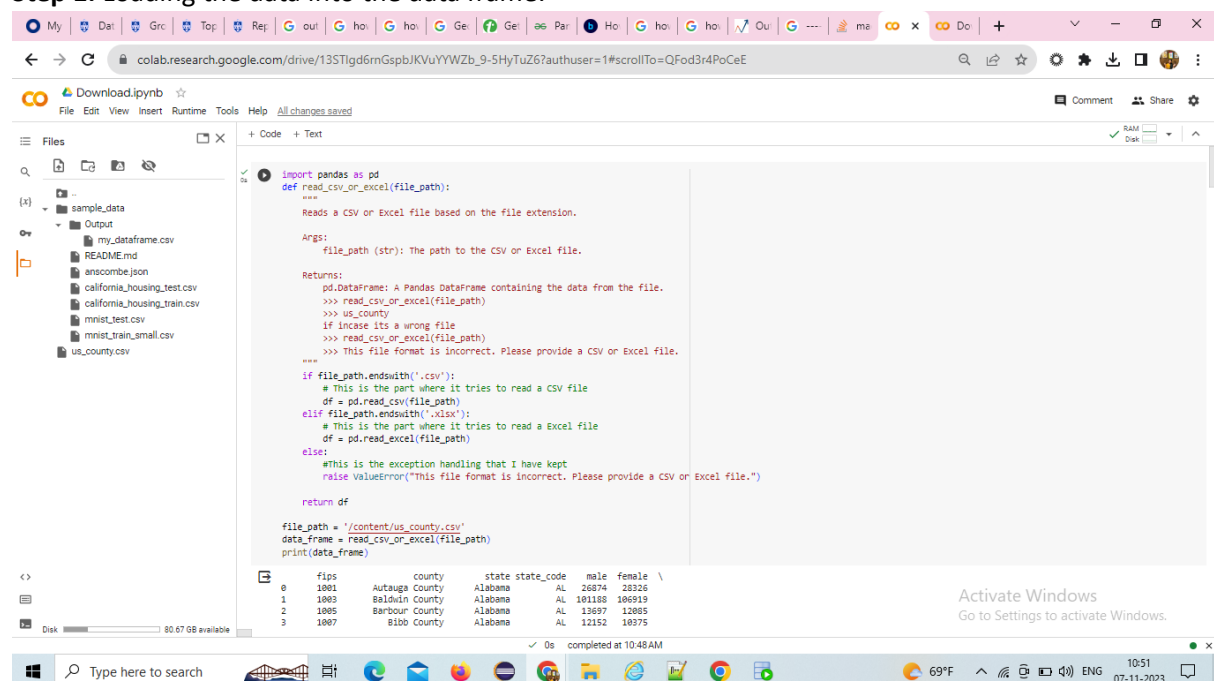
The dataset that I have taken gives detailed information about the country, state, male, female, age group, and demographics information such as latitude and longitude. The datasets clearly mentioned that the United States of America had the most reported COVID-19 cases.

The main objective of this analysis is to find out the patterns within the dataset to get a further understanding of the data. I also wanted to leverage it to choose a machine algorithm for predicting the survival rate of patients during the period of COVID-19. The dataset consists of demographic information population information (Such as male and female rates) and age information. In my previous dataset, I used a raw dataset so in this I am trying to clean the dataset by deleting unwanted rows and columns and by doing certain transformations. **US_COUNTY.CSV**

DATASET LINK: <https://drive.google.com/drive/folders/1RfLhJVOK45x9oGBmOKyZEpBAaHuITyYaw>

Data attributes: Fips, County, State, State code, male, female, median age, population, female percentage, Lat, long. So, in total my dataset has 3220 rows * 11 columns. The columns have a title/heading, which makes them readable.

Step 1: Loading the data into the data frame.



The screenshot shows a Google Colab notebook interface. The left sidebar displays a file explorer with a folder named 'sample_data' containing several CSV files, including 'us_county.csv'. The main area shows a Python code cell with the following content:

```
import pandas as pd
def read_csv_or_excel(file_path):
    """
    Reads a CSV or Excel file based on the file extension.

    Args:
        file_path (str): The path to the CSV or Excel file.

    Returns:
        pd.DataFrame: A Pandas DataFrame containing the data from the file.
    """
    >>> read_csv_or_excel(file_path)
    >>> us_county
    If incase its a wrong file
    >>> read_csv_or_excel(file_path)
    >>> This file format is incorrect. Please provide a CSV or Excel file.

    if file_path.endswith('.csv'):
        # This is the part where it tries to read a CSV file
        df = pd.read_csv(file_path)
    elif file_path.endswith('.xlsx'):
        # This is the part where it tries to read a Excel file
        df = pd.read_excel(file_path)
    else:
        # This is the exception handling that I have kept
        raise ValueError("This file format is incorrect. Please provide a CSV or Excel file.")

    return df

file_path = '/content/us_county.csv'
data_frame = read_csv_or_excel(file_path)
print(data_frame)
```

The output of the code is displayed below the cell, showing the first few rows of the 'us_county.csv' file:

	fips	county	state	state_code	male	female
0	1001	Autauga County	Alabama	AL	26874	28326
1	1002	Baldwin County	Alabama	AL	101186	106919
2	1005	Barbour County	Alabama	AL	13697	12085
3	1007	Bibb County	Alabama	AL	12152	10375

The bottom of the screenshot shows the Windows taskbar with the system clock indicating 10:51 on 07-11-2023.

Step 2: After running the code it will display the data that is there in CSV file.

	fips	county	state	state_code	male	female	\
0	1001	Autauga County	Alabama	AL	26874	28326	
1	1003	Baldwin County	Alabama	AL	101188	106919	
2	1005	Barbour County	Alabama	AL	13697	12085	
3	1007	Bibb County	Alabama	AL	12152	10375	
4	1009	Blount County	Alabama	AL	28434	29211	
...	
3215	72145	Vega Baja Municipio	Puerto Rico	NaN	25580	27791	
3216	72147	Vieques Municipio	Puerto Rico	NaN	4332	4439	
3217	72149	Villalba Municipio	Puerto Rico	NaN	11169	11824	
3218	72151	Yabucoa Municipio	Puerto Rico	NaN	16541	17608	
3219	72153	Yauco Municipio	Puerto Rico	NaN	17475	18964	
...	
	median_age	population	female_percentage	lat	long		
0	37.8	55200	51.315217	32.534923	-86.642730		
1	42.8	208107	51.376936	30.727479	-87.722564		
2	39.9	25782	46.873788	31.869581	-85.393210		
3	39.9	22527	46.055844	32.998628	-87.126475		
4	40.8	57645	50.673953	33.980869	-86.567380		
...		
3215	40.7	53371	52.071350	18.428461	-66.397926		
3216	43.6	8771	50.609965	18.122662	-65.439095		
3217	38.8	22993	51.424347	18.128155	-66.472816		
3218	42.5	34149	51.562271	18.070468	-65.896311		
3219	43.0	36439	52.043141	18.079728	-66.858276		
...		

[3220 rows x 11 columns]

Step 3: Next I am adding a new column financial data.

The screenshot shows a Google Colab notebook interface. At the top, there's a browser address bar showing the Colab URL. Below it, the notebook's file explorer is visible on the left, showing a folder named 'sample_data' and a file named 'my_dataframe.csv'. The main area of the notebook displays a code cell with the following content:

```
# adding a new column named financials with some values
new_column_values = [1001, 1003, 1005, 1007, 1009, 1011, 1013, 1015, 1017, 1019, 1021, 1023, 1025, 1027, 1029, 1031, 1033, 1035, 1037, 1039, 1041, 1043, 1045, 1047, 1049, 1051, 1053, 1055, 1057, 1059, 1061, 1063, 1065, 1067, 1069, 1071, 1073, 1075, 1077, 1079, 1081, 1083, 1085, 1087, 1089, 1091, 1093, 1095, 1097, 1099, 1101, 1103, 1105, 1107, 1109, 1111, 1113, 1115, 1117, 1119, 1121, 1123, 1125, 1127, 1129, 1131, 1133, 1135, 1137, 1139, 1141, 1143, 1145, 1147, 1149, 1151, 1153, 1155, 1157, 1159, 1161, 1163, 1165, 1167, 1169, 1171, 1173, 1175, 1177, 1179, 1181, 1183, 1185, 1187, 1189, 1191, 1193, 1195, 1197, 1199, 1201, 1203, 1205, 1207, 1209, 1211, 1213, 1215, 1217, 1219, 1221, 1223, 1225, 1227, 1229, 1231, 1233, 1235, 1237, 1239, 1241, 1243, 1245, 1247, 1249, 1251, 1253, 1255, 1257, 1259, 1261, 1263, 1265, 1267, 1269, 1271, 1273, 1275, 1277, 1279, 1281, 1283, 1285, 1287, 1289, 1291, 1293, 1295, 1297, 1299, 1301, 1303, 1305, 1307, 1309, 1311, 1313, 1315, 1317, 1319, 1321, 1323, 1325, 1327, 1329, 1331, 1333, 1335, 1337, 1339, 1341, 1343, 1345, 1347, 1349, 1351, 1353, 1355, 1357, 1359, 1361, 1363, 1365, 1367, 1369, 1371, 1373, 1375, 1377, 1379, 1381, 1383, 1385, 1387, 1389, 1391, 1393, 1395, 1397, 1399, 1401, 1403, 1405, 1407, 1409, 1411, 1413, 1415, 1417, 1419, 1421, 1423, 1425, 1427, 1429, 1431, 1433, 1435, 1437, 1439, 1441, 1443, 1445, 1447, 1449, 1451, 1453, 1455, 1457, 1459, 1461, 1463, 1465, 1467, 1469, 1471, 1473, 1475, 1477, 1479, 1481, 1483, 1485, 1487, 1489, 1491, 1493, 1495, 1497, 1499, 1501, 1503, 1505, 1507, 1509, 1511, 1513, 1515, 1517, 1519, 1521, 1523, 1525, 1527, 1529, 1531, 1533, 1535, 1537, 1539, 1541, 1543, 1545, 1547, 1549, 1551, 1553, 1555, 1557, 1559, 1561, 1563, 1565, 1567, 1569, 1571, 1573, 1575, 1577, 1579, 1581, 1583, 1585, 1587, 1589, 1591, 1593, 1595, 1597, 1599, 1601, 1603, 1605, 1607, 1609, 1611, 1613, 1615, 1617, 1619, 1621, 1623, 1625, 1627, 1629, 1631, 1633, 1635, 1637, 1639, 1641, 1643, 1645, 1647, 1649, 1651, 1653, 1655, 1657, 1659, 1661, 1663, 1665, 1667, 1669, 1671, 1673, 1675, 1677, 1679, 1681, 1683, 1685, 1687, 1689, 1691, 1693, 1695, 1697, 1699, 1701, 1703, 1705, 1707, 1709, 1711, 1713, 1715, 1717, 1719, 1721, 1723, 1725, 1727, 1729, 1731, 1733, 1735, 1737, 1739, 1741, 1743, 1745, 1747, 1749, 1751, 1753, 1755, 1757, 1759, 1761, 1763, 1765, 1767, 1769, 1771, 1773, 1775, 1777, 1779, 1781, 1783, 1785, 1787, 1789, 1791, 1793, 1795, 1797, 1799, 1801, 1803, 1805, 1807, 1809, 1811, 1813, 1815, 1817, 1819, 1821, 1823, 1825, 1827, 1829, 1831, 1833, 1835, 1837, 1839, 1841, 1843, 1845, 1847, 1849, 1851, 1853, 1855, 1857, 1859, 1861, 1863, 1865, 1867, 1869, 1871, 1873, 1875, 1877, 1879, 1881, 1883, 1885, 1887, 1889, 1891, 1893, 1895, 1897, 1899, 1901, 1903, 1905, 1907, 1909, 1911, 1913, 1915, 1917, 1919, 1921, 1923, 1925, 1927, 1929, 1931, 1933, 1935, 1937, 1939, 1941, 1943, 1945, 1947, 1949, 1951, 1953, 1955, 1957, 1959, 1961, 1963, 1965, 1967, 1969, 1971, 1973, 1975, 1977, 1979, 1981, 1983, 1985, 1987, 1989, 1991, 1993, 1995, 1997, 1999, 2001, 2003, 2005, 2007, 2009, 2011, 2013, 2015, 2017, 2019, 2021, 2023, 2025, 2027, 2029, 2031, 2033, 2035, 2037, 2039, 2041, 2043, 2045, 2047, 2049, 2051, 2053, 2055, 2057, 2059, 2061, 2063, 2065, 2067, 2069, 2071, 2073, 2075, 2077, 2079, 2081, 2083, 2085, 2087, 2089, 2091, 2093, 2095, 2097, 2099, 2101, 2103, 2105, 2107, 2109, 2111, 2113, 2115, 2117, 2119, 2121, 2123, 2125, 2127, 2129, 2131, 2133, 2135, 2137, 2139, 2141, 2143, 2145, 2147, 2149, 2151, 2153, 2155, 2157, 2159, 2161, 2163, 2165, 2167, 2169, 2171, 2173, 2175, 2177, 2179, 2181, 2183, 2185, 2187, 2189, 2191, 2193, 2195, 2197, 2199, 2201, 2203, 2205, 2207, 2209, 2211, 2213, 2215, 2217, 2219, 2221, 2223, 2225, 2227, 2229, 2231, 2233, 2235, 2237, 2239, 2241, 2243, 2245, 2247, 2249, 2251, 2253, 2255, 2257, 2259, 2261, 2263, 2265, 2267, 2269, 2271, 2273, 2275, 2277, 2279, 2281, 2283, 2285, 2287, 2289, 2291, 2293, 2295, 2297, 2299, 2301, 2303, 2305, 2307, 2309, 2311, 2313, 2315, 2317, 2319, 2321, 2323, 2325, 2327, 2329, 2331, 2333, 2335, 2337, 2339, 2341, 2343, 2345, 2347, 2349, 2351, 2353, 2355, 2357, 2359, 2361, 2363, 2365, 2367, 2369, 2371, 2373, 2375, 2377, 2379, 2381, 2383, 2385, 2387, 2389, 2391, 2393, 2395, 2397, 2399, 2401, 2403, 2405, 2407, 2409, 2411, 2413, 2415, 2417, 2419, 2421, 2423, 2425, 2427, 2429, 2431, 2433, 2435, 2437, 2439, 2441, 2443, 2445, 2447, 2449, 2451, 2453, 2455, 2457, 2459, 2461, 2463, 2465, 2467, 2469, 2471, 2473, 2475, 2477, 2479, 2481, 2483, 2485, 2487, 2489, 2491, 2493, 2495, 2497, 2499, 2501, 2503, 2505, 2507, 2509, 2511, 2513, 2515, 2517, 2519, 2521, 2523, 2525, 2527, 2529, 2531, 2533, 2535, 2537, 2539, 2541, 2543, 2545, 2547, 2549, 2551, 2553, 2555, 2557, 2559, 2561, 2563, 2565, 2567, 2569, 2571, 2573, 2575, 2577, 2579, 2581, 2583, 2585, 2587, 2589, 2591, 2593, 2595, 2597, 2599, 2601, 2603, 2605, 2607, 2609, 2611, 2613, 2615, 2617, 2619, 2621, 2623, 2625, 2627, 2629, 2631, 2633, 2635, 2637, 2639, 2641, 2643, 2645, 2647, 2649, 2651, 2653, 2655, 2657, 2659, 2661, 2663, 2665, 2667, 2669, 2671, 2673, 2675, 2677, 2679, 2681, 2683, 2685, 2687, 2689, 2691, 2693, 2695, 2697, 2699, 2701, 2703, 2705, 2707, 2709, 2711, 2713, 2715, 2717, 2719, 2721, 2723, 2725, 2727, 2729, 2731, 2733, 2735, 2737, 2739, 2741, 2743, 2745, 2747, 2749, 2751, 2753, 2755, 2757, 2759, 2761, 2763, 2765, 2767, 2769, 2771, 2773, 2775, 2777, 2779, 2781, 2783, 2785, 2787, 2789, 2791, 2793, 2795, 2797, 2799, 2801, 2803, 2805, 2807, 2809, 2811, 2813, 2815, 2817, 2819, 2821, 2823, 2825, 2827, 2829, 2831, 2833, 2835, 2837, 2839, 2841, 2843, 2845, 2847, 2849, 2851, 2853, 2855, 2857, 2859, 2861, 2863, 2865, 2867, 2869, 2871, 2873, 2875, 2877, 2879, 2881, 2883, 2885, 2887, 2889, 2891, 2893, 2895, 2897, 2899, 2901, 2903, 2905, 2907, 2909, 2911, 2913, 2915, 2917, 2919, 2921, 2923, 2925, 2927, 2929, 2931, 2933, 2935, 2937, 2939, 2941, 2943, 2945, 2947, 2949, 2951, 2953, 2955, 2957, 2959, 2961, 2963, 2965, 2967, 2969, 2971, 2973, 2975, 2977, 2979, 2981, 2983, 2985, 2987, 2989, 2991, 2993, 2995, 2997, 2999, 3001, 3003, 3005, 3007, 3009, 3011, 3013, 3015, 3017, 3019, 3021, 3023, 3025, 3027, 3029, 3031, 3033, 3035, 3037, 3039, 3041, 3043, 3045, 3047, 3049, 3051, 3053, 3055, 3057, 3059, 3061, 3063, 3065, 3067, 3069, 3071, 3073, 3075, 3077, 3079, 3081, 3083, 3085, 3087, 3089, 3091, 3093, 3095, 3097, 3099, 3101, 3103, 3105, 3107, 3109, 3111, 3113, 3115, 3117, 3119, 3121, 3123, 3125, 3127, 3129, 3131, 3133, 3135, 3137, 3139, 3141, 3143, 3145, 3147, 3149, 3151, 3153, 3155, 3157, 3159, 3161, 3163, 3165, 3167, 3169, 3171, 3173, 3175, 3177, 3179, 3181, 3183, 3185, 3187, 3189, 3191, 3193, 3195, 3197, 3199, 3201, 3203, 3205, 3207, 3209, 3211, 3213, 3215, 3217, 3219, 3221, 3223, 3225, 3227, 3229, 3231, 3233, 3235, 3237, 3239, 3241, 3243, 3245, 3247, 3249, 3251, 3253, 3255, 3257, 3259, 3261, 3263, 3265, 3267, 3269, 3271, 3273, 3275, 3277, 3279, 3281, 3283, 3285, 3287, 3289, 3291, 3293, 3295, 3297, 3299, 3301, 3303, 3305, 3307, 3309, 3311, 3313, 3315, 3317, 3319, 3321, 3323, 3325, 3327, 3329, 3331, 3333, 3335, 3337, 3339, 3341, 3343, 3345, 3347, 3349, 3351, 3353, 3355, 3357, 3359, 3361, 3363, 3365, 3367, 3369, 3371, 3373, 3375, 3377, 3379, 3381, 3383, 3385, 3387, 3389, 3391, 3393, 3395, 3397, 3399, 3401, 3403, 3405, 3407, 3409, 3411, 3413, 3415, 3417, 3419, 3421, 3423, 3425, 3427, 3429, 3431, 3433, 3435, 3437, 3439, 3441, 3443, 3445, 3447, 3449, 3451, 3453, 3455, 3457, 3459, 3461, 3463, 3465, 3467, 3469, 3471, 3473, 3475, 3477, 3479, 3481, 3483, 3485, 3487, 3489, 3491, 3493, 3495, 3497, 3499, 3501, 3503, 3505, 3507, 3509, 3511, 3513, 3515, 3517, 3519, 3521, 3523, 3525, 3527, 3529, 3531, 3533, 3535, 3537, 3539, 3541, 3543, 3545, 3547, 3549, 3551, 3553, 3555, 3557, 3559, 3561, 3563, 3565, 3567, 3569, 3571, 3573, 3575, 3577, 3579, 3581, 3583, 3585, 3587, 3589, 3591, 3593, 3595, 3597, 3599, 3601, 3603, 3605, 3607, 3609, 3611, 3613, 3615, 3617, 3619, 3621, 3623, 3625, 3627, 3629, 3631, 3633, 3635, 3637, 3639, 3641, 3643, 3645, 3647, 3649, 3651, 3653, 3655, 3657, 3659, 3661, 3663, 3665, 3667, 3669, 3671, 3673, 3675, 3677, 3679, 3681, 3683, 3685, 3687, 3689, 3691, 3693, 3695, 3697, 3699, 3701, 3703, 3705, 3707, 3709, 3711, 3713, 3715, 3717, 3719, 3721, 3723, 3725, 3727, 3729, 3731, 3733, 3735, 3737, 3739, 3741, 3743, 3745, 3747, 3749, 3751, 3753, 3755, 3757, 3759, 3761, 3763, 3765, 3767, 3769, 3771, 3773, 3775, 3777, 3779, 3781, 3783, 3785, 3787, 3789, 3791, 3793, 3795, 3797, 3799, 3801, 3803, 3805, 3807, 3809, 3811, 3813, 3815, 3817, 3819, 3821, 3823, 3825, 3827, 3829, 3831, 3833, 3835, 3837, 3839, 3841, 3843, 3845, 3847, 3849, 3851, 3853, 3855, 3857, 3859, 3861, 3863, 3865, 3867, 3869, 3871, 3873, 3875, 3877, 3879, 3881, 3883, 3885, 3887, 3889, 3891, 3893, 3895, 3897, 3899, 3901, 3903, 3905, 3907, 3909, 3911, 3913, 3915, 3917, 3919, 3921, 3923, 3925, 3927, 3929, 3931, 3933, 3935, 3937, 3939, 3941, 3943, 3945, 3947, 3949, 3951, 3953, 3955, 3957, 3959, 3961, 3963, 3965, 3967, 3969, 3971, 3973, 3975, 3977, 3979, 3981, 3983, 3985, 3987, 3989, 3991, 3993, 3995, 3997, 3999, 4001, 4003, 4005, 4007, 4009, 4011, 4013, 4015, 4017, 4019, 4021, 4023, 4025, 4027, 4029, 4031, 4033, 4035, 4037, 4039, 4041, 4043, 4045, 4047, 4049, 4051, 4053, 4055, 4057, 4059, 4061, 4063, 4065, 4067, 4069, 4071, 4073, 4075, 4077, 4079, 4081, 4083, 4085, 4087, 4089, 4091, 4093, 4095, 4097, 4099, 4101, 4103, 4105, 4107, 4109, 4111, 4113, 4115, 4117, 4119, 4121, 4123, 4125, 4127, 4129, 4131, 4133, 4135, 4137, 4139, 4141, 4143, 4145, 4147, 4149, 4151, 4153, 4155, 4157, 4159, 4161, 4163, 4165, 4167, 4169, 4171, 4173, 4175, 4177, 4179, 4181, 4183, 4185, 4187, 4189, 4191, 4193, 4195, 4197, 4199, 4201, 4203, 4205, 4207, 4209, 4211, 4213, 4215, 4217, 4219, 4221, 4223, 4225, 4227, 4229, 4231, 4233, 4235, 4237, 4239, 4241, 4243, 4245, 4247, 4249, 4251, 4253, 4255, 4257, 4259, 4261, 4263, 4265, 4267, 4269, 4271, 4273, 4275, 4277, 4279, 4281, 4283, 4285, 4287, 4289, 4291, 4293, 4295, 4297, 4299, 4301, 4303, 4305, 4307, 4309, 4311, 4313, 4315, 4317, 4319, 4321, 4323, 4325, 4327, 4329, 4331, 4333, 4335, 4337, 4339, 4341, 4343, 4345, 4347, 4349, 4351, 4353, 4355, 4357, 4359, 4361, 4363, 4365, 4367, 4369, 4371, 4373, 4375, 4377, 4379, 4381, 4383, 4385, 4387, 4389, 4391, 4393, 4395, 4397, 4399, 4401, 4403, 4405, 4407, 4409, 4411, 4413, 4415, 4417, 4419, 4421, 4423, 4425, 4427, 4429, 4431, 4433, 443
```

```

➡ New column added successfully!!!!
0      1001.0
1      1003.0
2      1005.0
3      1007.0
4      1009.0
...
3215    72145.0
3216    72147.0
3217    72149.0
3218         NaN
3219    72153.0
Name: financialdata, Length: 3220, dtype: float64

```

Step 5: Now I am trying to change the datatype from integer to float.

```

[6] # Changing the data type of column 'financialdata' from integer to float
data_frame['financialdata'] = data_frame['financialdata'].astype(float)

```

Step 6: After executing this code, we can clearly see that the datatype has been changed. Initially, it was an integer now it is changed to float.

```

▶ #this is the code to know the datatypes of all the columns
data_frame.dtypes

```

```

➡ fips                int64
   county            object
   state             object
   state_code        object
   male              int64
   female            int64
   median_age        float64
   population         int64
   female_percentage  float64
   lat               float64
   long              float64
   financialdata      float64
   dtype: object

```

Step 7: This is the output for the datatype that is been changed.

```

[8] #Output for the datatype that is changed from int64 to float64
print(data_frame.financialdata)

```

```

0      1001.0
1      1003.0
2      1005.0
3      1007.0
4      1009.0
...
3215    72145.0
3216    72147.0
3217    72149.0
3218         NaN
3219    72153.0
Name: financialdata, Length: 3220, dtype: float64

```

Step 8: Now I am trying to insert some values for the NaN values in my data.

```
# Fill missing values with a specific value i.e. with 0 and in the financial data column the only place that i inserted 0 is before the last value
#And I also added zeros in certain state_code column as it was NaN
data_frame.fillna({'financialdata': 0, 'state_code': 0})
```

	fips	county	state	state_code	male	female	median_age	population	female_percentage	lat	long	financialdata
0	1001	Autauga County	Alabama	AL	26874	28326	37.8	55200	51.315217	32.534923	-86.642730	1001.0
1	1003	Baldwin County	Alabama	AL	101188	106919	42.8	208107	51.376936	30.727479	-87.722564	1003.0
2	1005	Barbour County	Alabama	AL	13697	12085	39.9	25782	46.873788	31.869581	-85.393210	1005.0
3	1007	Bibb County	Alabama	AL	12152	10375	39.9	22527	46.055844	32.998628	-87.126475	1007.0
4	1009	Blount County	Alabama	AL	28434	29211	40.8	57645	50.673953	33.980869	-86.567380	1009.0
...
3215	72145	Vega Baja Municipio	Puerto Rico	0	25580	27791	40.7	53371	52.071350	18.428461	-66.397926	72145.0
3216	72147	Vieques Municipio	Puerto Rico	0	4332	4439	43.6	8771	50.609965	18.122662	-65.439095	72147.0
3217	72149	Villalba Municipio	Puerto Rico	0	11169	11824	38.8	22993	51.424347	18.128155	-66.472816	72149.0
3218	72151	Yabucoa Municipio	Puerto Rico	0	16541	17608	42.5	34149	51.562271	18.070468	-65.896311	0.0
3219	72153	Yauco Municipio	Puerto Rico	0	17475	18964	43.0	36439	52.043141	18.079728	-66.858276	72153.0

3220 rows x 12 columns

Step 9: Now I am trying to rename my columns male and female into male count and female count respectively.

```
[10] # Here I am trying to Rename the 'male' to 'male_count' and 'female' to 'female_count'
data_frame = data_frame.rename(columns={'male': 'male_count', 'female': 'female_count'})
```

Step 10: This is the output for changing the column name.

```
[11] #This is the output for rename columns
column_headings = data_frame.columns
print(column_headings)

Index(['fips', 'county', 'state', 'state_code', 'male_count', 'female_count',
       'median_age', 'population', 'female_percentage', 'lat', 'long',
       'financialdata'],
      dtype='object')
```

Step 11: Here I am trying to do some operations with two columns of my data

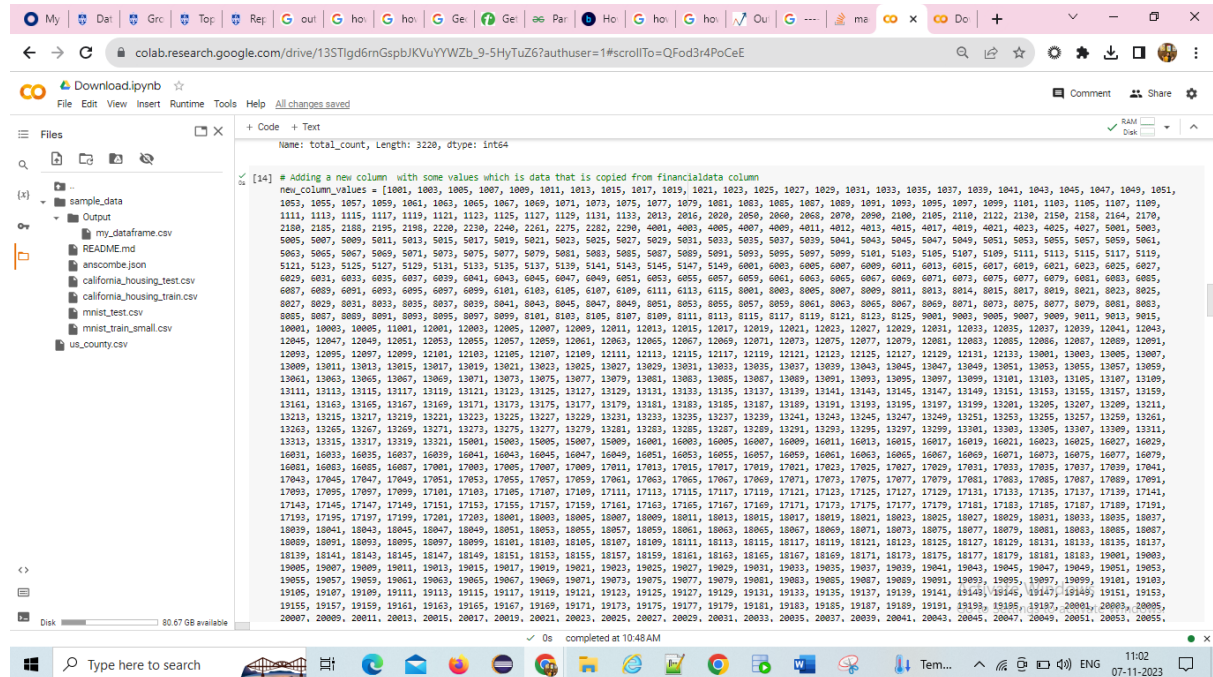
```
[12] # here i am trying to create a new column 'total_count' as the sum of columns 'male_count' and 'female_count'
data_frame['total_count'] = data_frame['male_count'] + data_frame['female_count']
```

Step 12: This is the output of creating a calculated field based on two other columns

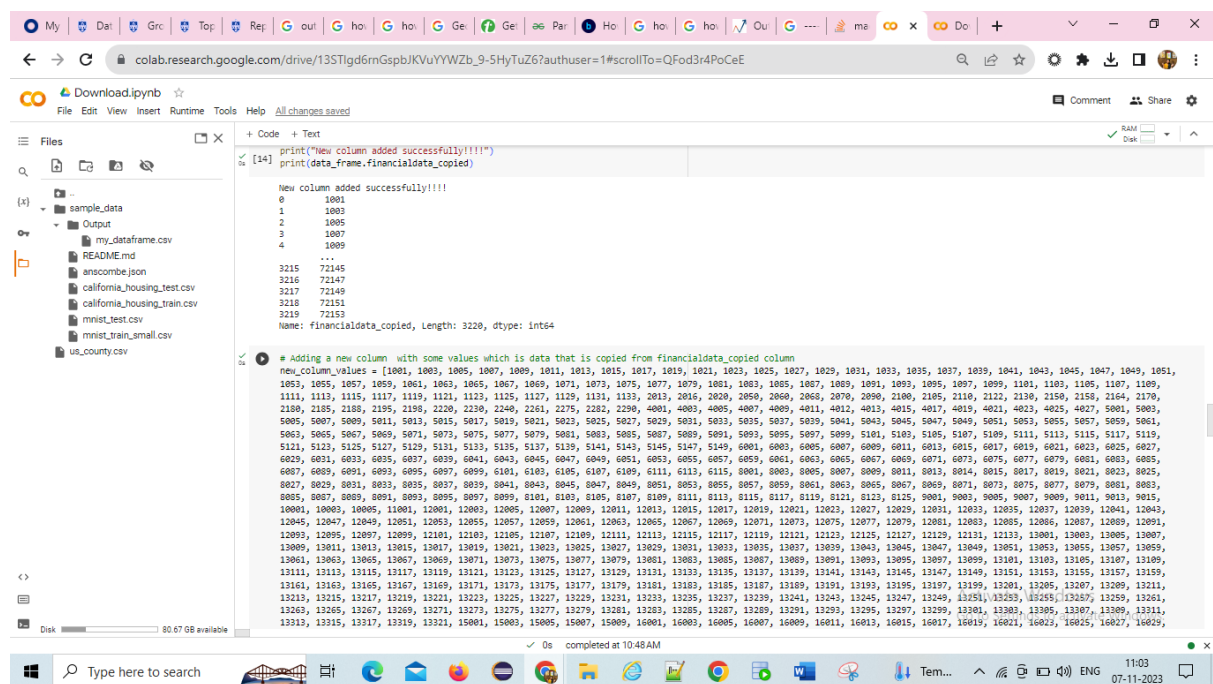
```
[13] #this is the output of creating a calculated field based on two other columns
print(data_frame.total_count)

0      55200
1     208107
2      25782
3      22527
4      57645
...
3215   53371
3216    8771
3217   22993
3218   34149
3219   36439
Name: total_count, Length: 3220, dtype: int64
```

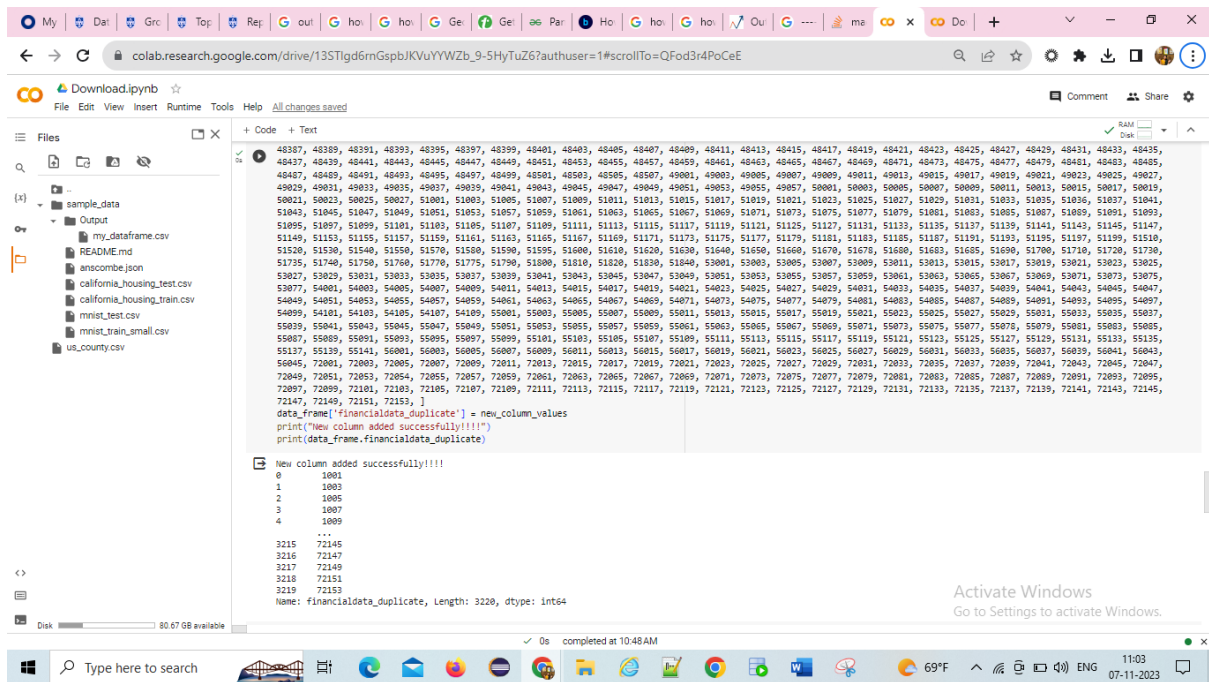
Step 13: Now I trying to duplicate a column that I have created



Step 14: For safety purposes, I am adding one more duplicate column so that I can delete it



Step 15: The reason why I do this is I don't want to delete my essential columns so I created a replica of the dummy column that I have created so that I could be able to drop the column without affecting my original data.



Step 16: Now two duplicate columns, has been inserted and I also performed delete function.

```
[16] column_headings = data_frame.columns
print(column_headings)

Index(['fips', 'county', 'state', 'state_code', 'male_count', 'female_count',
       'median_age', 'population', 'female_percentage', 'lat', 'long',
       'financialdata', 'total_count', 'financialdata_copied',
       'financialdata_duplicate'],
      dtype='object')

[17] #Here I am trying to drop the 'financialdata_duplicate' and 'financialdata_copied' column using the del statement
del data_frame['financialdata_duplicate']

[18] #Here I am trying to drop the 'financialdata_duplicate' and 'financialdata_copied' column using the del statement
del data_frame['financialdata_copied']

#Here we can clear know that the two unwanted columns such as financialdata_duplicate and financialdata_copied is deleted successfully!!!
column_headings = data_frame.columns
print(column_headings)

Index(['fips', 'county', 'state', 'state_code', 'male_count', 'female_count',
       'median_age', 'population', 'female_percentage', 'lat', 'long',
       'financialdata', 'total_count'],
      dtype='object')
```

Step 17: I am trying to save my file in my specified file path where I can have a copy of my data.

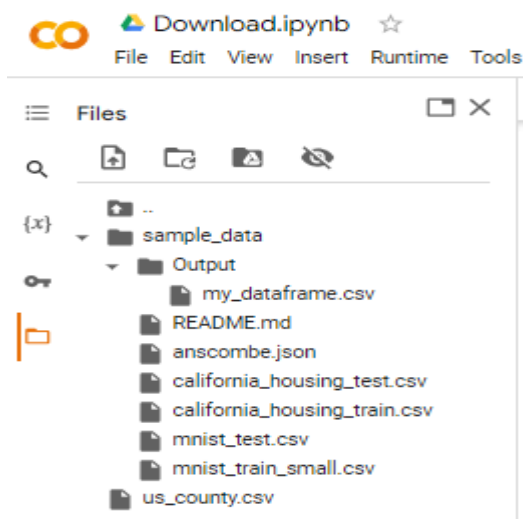
```
import pandas as pd

df = pd.DataFrame(data_frame)

# Specify the path where you want to save the CSV file
file_path = '/content/sample_data/Output/my_dataframe.csv'

# Save the DataFrame to the specified location
df.to_csv(file_path, index=False)
```

Step 18: This is the place where I am trying to save my data.



Step 19: Here I am trying to standardize my data.

```
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Here i am trying to standardize only two columns that is fips and population column
numeric_columns = data_frame[['fips', 'population']]

# here i am trying to initialize the StandardScaler
scaler = StandardScaler()
numeric_standardized = scaler.fit_transform(numeric_columns)
df_standardized = pd.DataFrame(numeric_standardized, columns=numeric_columns.columns)

print("Original DataFrame:")
print(data_frame)
print("\nStandardized DataFrame (numeric columns only):")
print(df_standardized)
```

Original DataFrame:

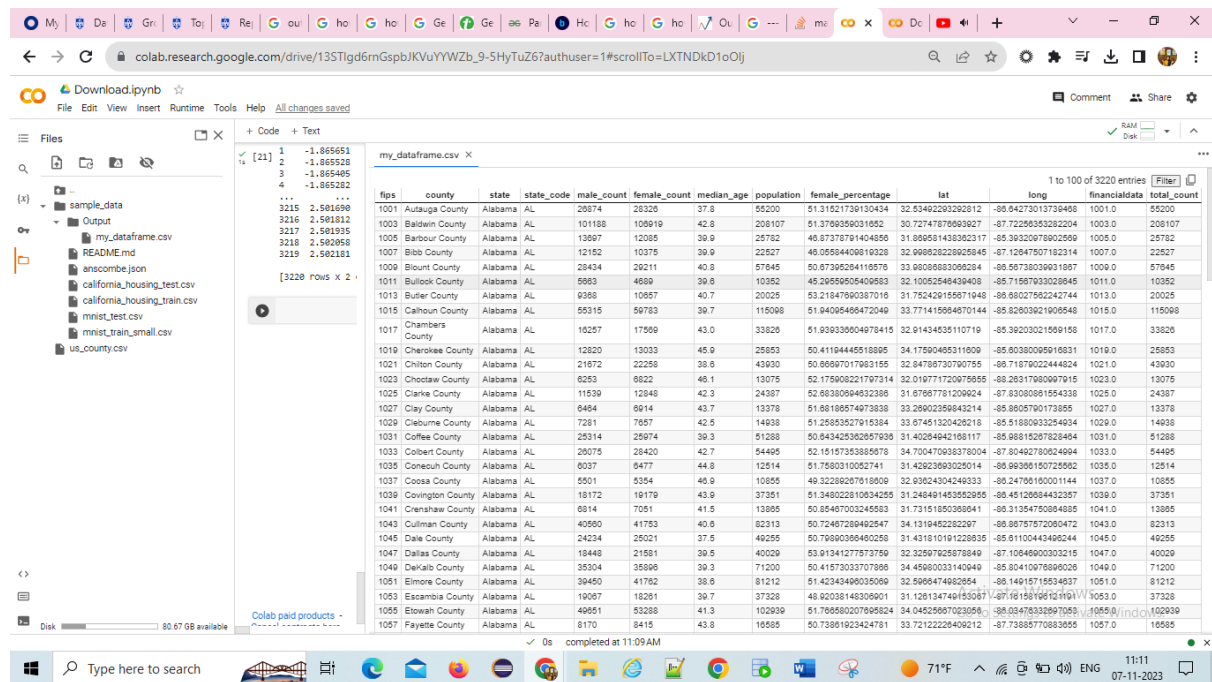
	fips	county	state	state_code	male_count	\
0	1001	Autauga County	Alabama	AL	26874	
1	1003	Baldwin County	Alabama	AL	101188	
2	1005	Barbour County	Alabama	AL	13697	
3	1007	Bibb County	Alabama	AL	12152	
4	1009	Blount County	Alabama	AL	28434	
...
3215	72145	Vega Baja Municipio	Puerto Rico	NaN	25580	
3216	72147	Vieques Municipio	Puerto Rico	NaN	4332	
3217	72149	Villalba Municipio	Puerto Rico	NaN	11169	
3218	72151	Yabucoa Municipio	Puerto Rico	NaN	16541	
3219	72153	Yauco Municipio	Puerto Rico	NaN	17475	

	female_count	median_age	population	female_percentage	lat	\
0	28326	37.8	55200	51.315217	32.534923	
1	106919	42.8	208107	51.376936	30.727479	
2	12085	39.9	25782	46.873788	31.869581	
3	10375	39.9	22527	46.055844	32.998628	
4	29211	40.8	57645	50.673953	33.980869	
...
3215	27791	40.7	53371	52.071350	18.428461	
3216	4439	43.6	8771	50.609965	18.122662	
3217	11824	38.8	22993	51.424347	18.128155	
3218	17608	42.5	34149	51.562271	18.070468	
3219	18964	43.0	36439	52.043141	18.079728	

	long	financialdata	total_count
0	-86.642730	1001.0	55200
1	-87.722564	1003.0	208107
2	-85.393210	1005.0	25782
3	-87.126475	1007.0	22527
4	-86.567380	1009.0	57645
...
3215	-66.397926	72145.0	53371
3216	-65.439095	72147.0	8771
3217	-66.472816	72149.0	22993
3218	-65.896311	NaN	34149
3219	-66.858276	72153.0	36439

[3220 rows x 13 columns]

Step 20: This is the dataset that I have received after all the transformation and clean-up of data.



fips	county	state	state_code	male_count	female_count	median_age	population	female_percentage	lat	long	financialdata	total_count
1001	Autauga County	Alabama	AL	28674	28328	37.8	85200	51.31921739103434	32.53462230230012	-88.54273013739468	1001.0	56520
1003	Baldwin County	Alabama	AL	101188	109919	42.8	208107	51.37869590310852	30.72747676966927	-87.72268332822204	1003.0	208107
1005	Barbour County	Alabama	AL	13597	12085	39.9	25782	48.87378791404896	31.89581433382317	-85.39320978902599	1005.0	25782
1007	Bibb County	Alabama	AL	12152	10375	39.9	22527	48.05584408919328	32.9889282228625945	-87.12647507182314	1007.0	22527
1009	Blount County	Alabama	AL	28434	20211	40.8	57645	50.67395284110570	33.980089833066284	-85.56738039931887	1009.0	57645
1011	Bullock County	Alabama	AL	5983	4689	39.8	10352	45.29559505409583	32.10052548439408	-85.71587933028645	1011.0	10352
1013	Butler County	Alabama	AL	9385	10057	40.7	20025	53.218476903387016	31.752429155671948	-86.08027562242744	1013.0	20025
1015	Calhoun County	Alabama	AL	55315	59783	39.7	115088	51.94095488472049	33.771415954670144	-85.82603921908548	1015.0	115088
1017	Chambers County	Alabama	AL	18257	17569	43.0	33826	51.939338604978415	32.81434355110719	-85.39203021569158	1017.0	33826
1019	Cherokee County	Alabama	AL	12820	13033	45.9	25853	50.41194445518895	34.17590495311809	-85.03302009918831	1019.0	25853
1021	Chilton County	Alabama	AL	21672	22258	39.8	43930	50.8697017983155	32.84789730790755	-86.718719022444824	1021.0	43930
1023	Choctaw County	Alabama	AL	6253	6822	48.1	13075	52.175908221797314	32.01977172097595	-88.2817080907915	1023.0	13075
1025	Clarke County	Alabama	AL	11539	12848	42.3	24387	52.68380904532386	31.87667781209924	-87.83080861554338	1025.0	24387
1027	Clay County	Alabama	AL	6464	6914	43.7	13378	51.68186574973838	33.28902259843214	-85.8605700173855	1027.0	13378
1029	Cleburne County	Alabama	AL	7281	7657	42.5	14938	51.25853527915384	33.87451320426218	-85.51880933254934	1029.0	14938
1031	Coffee County	Alabama	AL	25314	25974	39.3	51288	50.643425362657936	31.40294942108117	-85.98815267828494	1031.0	51288
1033	Colbert County	Alabama	AL	29075	28420	42.7	54495	52.15173538959578	34.700478038378004	-87.804927300246994	1033.0	54495
1035	Conecuh County	Alabama	AL	6037	6477	44.8	12514	51.1560310052741	31.42923863025014	-86.98396160725562	1035.0	12514
1037	Coosa County	Alabama	AL	5501	5354	45.9	10855	49.32289287018809	32.89824304246933	-86.247891802001144	1037.0	10855
1039	Covington County	Alabama	AL	18172	19179	43.9	37351	51.348022810634255	31.248494145359295	-86.45120984432357	1039.0	37351
1041	Crenshaw County	Alabama	AL	6814	7051	41.5	13885	50.85467003245593	31.3181850386841	-86.313547578084835	1041.0	13885
1043	Cullman County	Alabama	AL	40560	41753	40.8	82313	50.72467289492547	34.1319452282297	-86.86757572086472	1043.0	82313
1045	Dale County	Alabama	AL	24234	25021	37.5	49255	50.76980386480258	31.431810191228935	-85.91100443486244	1045.0	49255
1047	Dallas County	Alabama	AL	18448	21581	39.5	40029	53.91341277573789	32.32597625878849	-87.108469003303215	1047.0	40029
1049	DeKalb County	Alabama	AL	35304	35898	39.3	71200	50.41573033707896	34.45800033140949	-85.80410078898026	1049.0	71200
1051	Elmore County	Alabama	AL	39450	41762	38.6	81212	51.42343496035096	32.5968474862854	-86.14915715534837	1051.0	81212
1053	Escambia County	Alabama	AL	19087	18281	39.7	37328	48.92038148309601	31.126134746155067	-87.181861961219191	1053.0	37328
1055	Etowah County	Alabama	AL	49651	53268	41.3	102939	51.7658680207696824	34.04526667026399	-86.03478333697058	1055.0	102939
1057	Fayette County	Alabama	AL	7170	9415	43.8	16585	50.73881923424781	33.72122228409212	-87.73885776883955	1057.0	16585

Important Links:

GitHub Link

<https://github.com/santhiya-hds5210/ORES--Data-Transformation-with-Python-or-R#ores--data-transformation-with-python-or-r>

Drive Link:

<https://drive.google.com/drive/u/1/folders/1LKybVuUJ01DsA5uYHnUCKJlhQXOQDP>

Dataset Link: [US_COUNTY.CSV]

<https://drive.google.com/drive/folders/1RfLhJVOK45x9oGBmOKyZEpBAaHuITyYaw>

Appendix:

1. https://www.google.com/search?q=how+to+add+comma+to+3000+values+in+notepad%2B%2B&oq=how+to+add+comma+to+3000+values+in+no&gs_lcrp=EgZjaHJvbWUqBwgBECEYoAEyBggAEEUYOTIHCAEQIRigATIHCAIQIRigATIHCAMQIRigAdIBCTE4MTU0ajBqN6gCALACAA&sourceid=chrome&ie=UTF-8
2. https://www.google.com/search?q=how+to+make+3000+line+of+data+in+a+single+horizonta+line+in+notepad%2B%2B&oq=how+to+make+3000+line+of+data+in+a+single+horizonta+line+in+notepad%2B%2B&gs_lcrp=EgZjaHJvbWUyBggAEEUYOdIBCTM0NzAwajBqN6gCALACAA&sourceid=chrome&ie=UTF-8
3. https://www.google.com/search?sca_esv=579833118&sxsrf=AM9HkKnY5z6zssSxCf2HJgZ2L_M4cT03fg:1699287887755&q=Geographic+data+examples&uds=H4sIAAAAAAAAAA-MY5-JwrUjMLchJLRbiT0_NTy9KLMjITFZISSxJNGAskNBHFVJIhSo2QlcMAIcVU5tKAAAA&sa=X&ved=2ahUKEwiVta7X5K-CAXWbMDQIHYYiuDPMQxKsJegQIDRAB&ictx=0&biw=1366&bih=651&dpr=1
4. <https://www.tutorialspoint.com/get-the-data-type-of-column-in-pandas-python>

5. <https://www.geeksforgeeks.org/how-to-create-an-empty-dataframe-and-append-rows-columns-to-it-in-pandas/>
6. <https://builtin.com/data-science/rename-columns-pandas>
7. https://www.google.com/search?q=how+to+create+a+calculated+field+based+on+two+other+columns+dataframe&sca_esv=579833118&sxsrf=AM9HkKnatMkR4mMkAJ-HHGCVRSZuFP_e-g%3A1699292407777&ei=9yRJZdSRL4qIptQPvc-qsA0&ved=0ahUKEwiU6dbC9a-CAxUKhIkEHb2nCtYQ4dUDCBA&uact=5&oq=how+to+create+a+calculated+field+based+on+two+other+columns+dataframe&gs_lcp=Egxnd3Mtd2l6LXNlcnAiRWWhvdyB0byBjcmVhdGUgYSBjYWxjdWxhdGVkIGZpZWxkIGJhc2VkIG9uIHR3byBvdGhlcjBjb2x1bW5zIGRh dGFmcmFtZUjiSFD-BViPRXAEeAGQAQCYAbQCoAHIDKoBBzAuOS4xLjG4AQPIAQD4AQHC A goQABhH GNYEGLADwglIEECMYJ8ICBxAjGLACGCfiAwQYACBBiAYBkAYI&sclient=gws-wiz-serp
8. https://www.google.com/search?q=how+to+drop+a+column+in+dataframe+in+python&oq=how+to+drop+a+column+in+dataframe+in+pyhto&gs_lcrp=EgZjaHJvbWUqCQgBEAAYDRiABDIGCAAQRRg5MgkIARAAGA0YgAQyCAgCEAAYFhgeMggIAx AAGBYYYHjIICAQ QABgWGB4yCAgFEAAYFhgeMggIBhAAGBYYYHjIICAcQABgWGB4yCAgIEAAYFhge MgoICRAAGIYDGIoF0gEJMTQwOTdqMGo3qAIA sAIA&sourceid=chrome&ie=UTF-8
9. <https://www.analyticsvidhya.com/blog/2021/03/zooming-out-a-look-at-outlier-and-how-to-deal-with-them-in-data-science/>
10. [https://www.google.com/search?q=-----+ValueError+Traceback+\(most+recent+call+last\)+%3Cipython-input-64-6c056e54992a%3E+in+%3Ccell+line%3A+8%3E\(\)++6+7+%23+Fit+and+transform+the+DataFrame+to+standardize+the+data+----+%3E+8+df_standardized+%3D+scaler.fit_transform\(data_frame\)+9+10+%23+Convert+the+standardized+data+back+to+a+DataFrame+7+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fpandas%2Fcore%2Fgeneric.py+in+_array_\(self%2C+dtype\)+2068+2069+def+_array_\(self%2C+dtype%3A+npt.DTypeLike+%7C+None+%3D+None\)+-+%3E+np.ndarray%3A+-+%3E+2070+return+np.asarray\(self._values%2C+dtype%3Ddtype\)+2071+2072+def+_array_wrap_\(+ValueError%3A+could+not+convert+string+to+float%3A+%27Autauga+County%27&oq=-----+ValueError++++++Traceback+\(most+recent+call+last\)+%3Cipython-input-64-6c056e54992a%3E+in+%3Ccell+line%3A+8%3E\(\)++6+7+%23+Fit+and+transform+the+DataFrame+to+standardize+the+data+----+%3E+8+df_standardized+%3D+scaler.fit_transform\(data_frame\)+9+10+%23+Convert+the+standardized+data+back+to+a+DataFrame+7+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fpandas%2Fcore%2Fgeneric.py+in+_array_\(self%2C+dtype\)+2068+2069+++def+_array_\(self%2C+dtype%3A+npt.DTypeLike+%7C+None+%3D+None\)+-+%3E+np.ndarray%3A+-+%3E+2070++++++return+np.asarray\(self._values%2C+dtype%3Ddtype\)+2071+2072+++def+_array_wrap_\(+ValueError%3A+could+not+convert+string+to+float%3A+%27Autauga+County%27&gs_lcrp=EgZjaHJvbWUyBggAEUYOdIBBzk2N2owajeoAgCwAgA&sourceid=chrome&ie=UTF-8](https://www.google.com/search?q=-----+ValueError+Traceback+(most+recent+call+last)+%3Cipython-input-64-6c056e54992a%3E+in+%3Ccell+line%3A+8%3E()++6+7+%23+Fit+and+transform+the+DataFrame+to+standardize+the+data+----+%3E+8+df_standardized+%3D+scaler.fit_transform(data_frame)+9+10+%23+Convert+the+standardized+data+back+to+a+DataFrame+7+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fpandas%2Fcore%2Fgeneric.py+in+_array_(self%2C+dtype)+2068+2069+def+_array_(self%2C+dtype%3A+npt.DTypeLike+%7C+None+%3D+None)+-+%3E+np.ndarray%3A+-+%3E+2070+return+np.asarray(self._values%2C+dtype%3Ddtype)+2071+2072+def+_array_wrap_(+ValueError%3A+could+not+convert+string+to+float%3A+%27Autauga+County%27&oq=-----+ValueError++++++Traceback+(most+recent+call+last)+%3Cipython-input-64-6c056e54992a%3E+in+%3Ccell+line%3A+8%3E()++6+7+%23+Fit+and+transform+the+DataFrame+to+standardize+the+data+----+%3E+8+df_standardized+%3D+scaler.fit_transform(data_frame)+9+10+%23+Convert+the+standardized+data+back+to+a+DataFrame+7+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fpandas%2Fcore%2Fgeneric.py+in+_array_(self%2C+dtype)+2068+2069+++def+_array_(self%2C+dtype%3A+npt.DTypeLike+%7C+None+%3D+None)+-+%3E+np.ndarray%3A+-+%3E+2070++++++return+np.asarray(self._values%2C+dtype%3Ddtype)+2071+2072+++def+_array_wrap_(+ValueError%3A+could+not+convert+string+to+float%3A+%27Autauga+County%27&gs_lcrp=EgZjaHJvbWUyBggAEUYOdIBBzk2N2owajeoAgCwAgA&sourceid=chrome&ie=UTF-8)
11. <https://stackoverflow.com/questions/59807430/google-colab-can-we-restore-all-the-data-even-after-the-runtime-disconnects>
12. <https://chat.openai.com/c/d583192e-1edc-46f2-831e-8e336adefe07>

13. <https://www.forefront.ai/>
14. <https://colab.research.google.com/drive/1VorJR5TFT-58iqxp-eTMcx7h5SAAxCjX#scrollTo=Fj86ZXs1WIJM>
15. <https://canvas.slu.edu/courses/45377/assignments/343229>
16. https://www.google.com/search?q=how+to+standardise+my+data+in+dataframe&oq=how+to+standardise+my+data+in+dataframe&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIHCAEQIRigATIHCARigAdIBCDExODlqMGo3qAIAAsAIA&sourceid=chrome&ie=UTF-8
17. [https://www.google.com/search?q=-----+ValueError+Traceback+\(most+recent+call+last\)+%3Cipython-input-62-029504c47d07%3E+in+%3Ccell+line%3A+7%3E\(\)+5+df_uscounty+%3D+pd.read_csv\(%27%2Fcontent%2Fus_county.csv%27\)+6+plt.figure\(figsize%3D\(5%2C5\)\)+----%3E+7+sns.boxplot\(y%3D%27male_count%27%2Cdata%3Ddf_uscounty\)+8+plt.show\(\)+2+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fseaborn%2Fcategorical.py+in+establish_variables\(self%2C+x%2C+y%2C+hue%2C+data%2C+orient%2C+order%2C+hue_order%2C+units\)+539+if+isinstance\(var%2C+str\)%3A+540+err+%3D+f%22Could+not+interpret+input+%27%7Bvar%7D%27%22+--%3E+541+raise+ValueError\(err\)+542+543+%23+Figure+out+the+plotting+orientation+ValueError%3A+Could+not+interpret+input+%27male_count%27+%3Cfigure+size+500x500+with+0+Axes%3E&oq=-----+ValueError+++++++Traceback+\(most+recent+call+last\)+%3Cipython-input-62-029504c47d07%3E+in+%3Ccell+line%3A+7%3E\(\)+5+df_uscounty+%3D+pd.read_csv\(%27%2Fcontent%2Fus_county.csv%27\)+6+plt.figure\(figsize%3D\(5%2C5\)\)+----%3E+7+sns.boxplot\(y%3D%27male_count%27%2Cdata%3Ddf_uscounty\)+8+plt.show\(\)+2+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fseaborn%2Fcategorical.py+in+establish_variables\(self%2C+x%2C+y%2C+hue%2C+data%2C+orient%2C+order%2C+hue_order%2C+units\)+539+++++++if+isinstance\(var%2C+str\)%3A+540+++++++err+%3D+f%22Could+not+interpret+input+%27%7Bvar%7D%27%22+--%3E+541+++++++raise+ValueError\(err\)+542+543+++++++%23+Figure+out+the+plotting+orientation+ValueError%3A+Could+not+interpret+input+%27male_count%27+%3Cfigure+size+500x500+with+0+Axes%3E&gs_lcrp=EgZjaHJvbWUyBggAEEUYOdIBCDEwNTRqMGo3qAIAAsAIA&sourceid=chrome&ie=UTF-8](https://www.google.com/search?q=-----+ValueError+Traceback+(most+recent+call+last)+%3Cipython-input-62-029504c47d07%3E+in+%3Ccell+line%3A+7%3E()+5+df_uscounty+%3D+pd.read_csv(%27%2Fcontent%2Fus_county.csv%27)+6+plt.figure(figsize%3D(5%2C5))+----%3E+7+sns.boxplot(y%3D%27male_count%27%2Cdata%3Ddf_uscounty)+8+plt.show()+2+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fseaborn%2Fcategorical.py+in+establish_variables(self%2C+x%2C+y%2C+hue%2C+data%2C+orient%2C+order%2C+hue_order%2C+units)+539+if+isinstance(var%2C+str)%3A+540+err+%3D+f%22Could+not+interpret+input+%27%7Bvar%7D%27%22+--%3E+541+raise+ValueError(err)+542+543+%23+Figure+out+the+plotting+orientation+ValueError%3A+Could+not+interpret+input+%27male_count%27+%3Cfigure+size+500x500+with+0+Axes%3E&oq=-----+ValueError+++++++Traceback+(most+recent+call+last)+%3Cipython-input-62-029504c47d07%3E+in+%3Ccell+line%3A+7%3E()+5+df_uscounty+%3D+pd.read_csv(%27%2Fcontent%2Fus_county.csv%27)+6+plt.figure(figsize%3D(5%2C5))+----%3E+7+sns.boxplot(y%3D%27male_count%27%2Cdata%3Ddf_uscounty)+8+plt.show()+2+frames+%2Fusr%2Flocal%2Flib%2Fpython3.10%2Fdist-packages%2Fseaborn%2Fcategorical.py+in+establish_variables(self%2C+x%2C+y%2C+hue%2C+data%2C+orient%2C+order%2C+hue_order%2C+units)+539+++++++if+isinstance(var%2C+str)%3A+540+++++++err+%3D+f%22Could+not+interpret+input+%27%7Bvar%7D%27%22+--%3E+541+++++++raise+ValueError(err)+542+543+++++++%23+Figure+out+the+plotting+orientation+ValueError%3A+Could+not+interpret+input+%27male_count%27+%3Cfigure+size+500x500+with+0+Axes%3E&gs_lcrp=EgZjaHJvbWUyBggAEEUYOdIBCDEwNTRqMGo3qAIAAsAIA&sourceid=chrome&ie=UTF-8)
18. https://www.google.com/search?q=how+to+drop+a+column+in+dataframe+in+python&oq=how+to+drop+a+column+in+dataframe+in+python&gs_lcrp=EgZjaHJvbWUyCwgAEEUYJxg5GIoFMggIARAAGBYHjIICAIQABgWGB4yCAGDEAAYFhgeMggIBBAAGBYHjIICAUQABgWGB4yCAGGEAAYFhgeMggIBxAAGBYHjIICAgQABgWGB4yCAGJEAAyFhge0gEIMTEyMWowajeoAgCwAgA&sourceid=chrome&ie=UTF-8
19. https://www.google.com/search?q=what+is+the+code+to+see+the+only+headings+of+the+table+in+the+dataframe+in+python&oq=what+is+the+code+to+see+the+only+headings+of+the+table+in+the+dataframe+in+python&gs_lcrp=EgZjaHJvbWUyBggAEEUYOdIBCdyxMTlqMGo3qAIAAsAIA&sourceid=chrome&ie=UTF-8
20. https://www.google.com/search?q=how+to+insert+value+for+the+null+in+dataframe+python&oq=how+to+insert+value+for+the+null+in+dataframe+python&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIHCAEQABgWGB4yCAGCEAAYFhgeMgoIAxAGIYDGIoFMgoIBBAAGIYDGIoFMgoIBRAAGIYDGIoF0gEHODkxajBqN6gCALACAA&sourceid=chrome&ie=UTF-8