WGU D208 TASK 1 REV 10 - MATTINSON

Part III(a): Data Preparation

C. Summarize data preparation for multiple regression by doing the following:

In [1]:
```
# import and configure packages
from imports import *
%matplotlib inline
warnings.filterwarnings('ignore')
```

```
w:\code\wgu\py39\Scripts\python.exe
python version: 3.9.7
pandas version: 1.4.2
numpy version: 1.22.3
scipy version: 1.8.0
sklearn version: 1.0.2
matplotlib version: 3.5.2
seaborn version: 0.11.2
```

In [2]:
```
from helpers import *
```

```
get_course_filename_str version: 1.5
save_course_table_csv version: 1.4
describe_dataframe_type version: 1.1
create_scatter_plot_fig version: 1.1
create_barplot_num_vs_cat_fig version: 1.9
create_distribution_plot_from_feature_fig version: 1.9
get_unique_values_list version: 1.2
create_correlation_matrix version: 1.2
get_redundant_pairs version: 1.0
get_top_n_correlations version: 1.1
create_simple_histogram_numerical_feature_fig version: 1.10
create_stacked_barplot_cat_or_bool_feature_fig version: 1.7
create_stacked_histogram_num_feature_fig version: 1.6
```

In [3]:
```
#constants
random_state = 42
plotColor = ['b','g','r','m','c', 'y']
markers = ['+','o','*','^','v','>','<']
target='MonthlyCharge'
```

In [5]:
```
# install statsmodels
!pip install statsmodels
```

```
Collecting statsmodels
  Using cached statsmodels-0.13.2-cp39-cp39-win_amd64.whl (9.1 MB)
Requirement already satisfied: scipy>=1.3 in w:\code\wgu\py39\lib\site-packages (from statsmodel
s) (1.8.0)
Requirement already satisfied: packaging>=21.3 in w:\code\wgu\py39\lib\site-packages (from stats
models) (21.3)
Requirement already satisfied: numpy>=1.17 in w:\code\wgu\py39\lib\site-packages (from statsmode
ls) (1.22.3)
Collecting patsy>=0.5.2
  Using cached patsy-0.5.2-py2.py3-none-any.whl (233 kB)
Requirement already satisfied: pandas>=0.25 in w:\code\wgu\py39\lib\site-packages (from statsmod
els) (1.4.2)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in w:\code\wgu\py39\lib\site-packages (f
rom packaging>=21.3->statsmodels) (3.0.8)
Requirement already satisfied: pytz>=2020.1 in w:\code\wgu\py39\lib\site-packages (from pandas>=
0.25->statsmodels) (2022.1)
Requirement already satisfied: python-dateutil>=2.8.1 in w:\code\wgu\py39\lib\site-packages (fro
m pandas>=0.25->statsmodels) (2.8.2)
Requirement already satisfied: six in w:\code\wgu\py39\lib\site-packages (from patsy>=0.5.2->sta
tsmodels) (1.16.0)
Installing collected packages: patsy, statsmodels
Successfully installed patsy-0.5.2 statsmodels-0.13.2
```

In [6]:
```
import statsmodels.api as sm
import statsmodels.formula.api as smf
```

Import Raw Data. Import churn_clean.csv.

In [7]:
```
# load data from .csv
df_raw = pd.read_csv('data/churn_clean.csv')
```

TABLE 1.RAW

Initial state of data before any manipulation.

In [8]:
```
save_course_table_csv(data=df_raw, title='RAW', title_only=True)
```

	0	1	
CaseOrder	1	2	
Customer_id	K409198	S120509	
Interaction	aa90260b-4141-4a24-8e36-b04ce1f4f77b	fb76459f-c047-4a9d-8af9-e0f7d4ac2524	344d114c-3736-c7
UID	e885b299883d4f9fb18e39c75155d990	f2de8bef964785f41a2959829830fb8a	f1784cfa9f6d92ae816197
City	Point Baker	West Branch	
State	AK	MI	
County	Prince of Wales-Hyder	Ogemaw	
Zip	99927	48661	
Lat	56.251	44.32893	
Lng	-133.37571	-84.2408	
Population	38	10446	
Area	Urban	Urban	
TimeZone	America/Sitka	America/Detroit	America/
Job	Environmental health practitioner	Programmer, multimedia	Chief Fina
Children	0	1	
Age	68	27	
Income	28561.99	21704.77	
Marital	Widowed	Married	
Gender	Male	Female	
Churn	No	Yes	
Outage_sec_perweek	7.978323	11.69908	
Email	10	12	
Contacts	0	0	
Yearly_equip_failure	1	1	
Techie	No	Yes	
Contract	One year	Month-to-month	
Port_modem	Yes	No	
Tablet	Yes	Yes	
InternetService	Fiber Optic	Fiber Optic	
Phone	Yes	Yes	
Multiple	No	Yes	
OnlineSecurity	Yes	Yes	
OnlineBackup	Yes	No	
DeviceProtection	No	No	

	0	1	
TechSupport	No	No	
StreamingTV	No	Yes	
StreamingMovies	Yes	Yes	
PaperlessBilling	Yes	Yes	
PaymentMethod	Credit Card (automatic)	Bank Transfer(automatic)	Credit Card
Tenure	6.795513	1.156681	
MonthlyCharge	172.455519	242.632554	
Bandwidth_GB_Year	904.53611	800.982766	2
Item1	5	3	
Item2	5	4	
Item3	5	3	
Item4	3	3	
Item5	4	4	
Item6	4	3	
Item7	3	4	
Item8	4	4	

```
shape: (10000, 50)
Table saved to: TABLES/RAW.CSV
```

In [9]:
```
df_clean = df_raw.copy() # create a clean dataframe
```

Drop Unwanted Data. Drop unwanted data.

In [10]:
```
# remove unwanted data
cols_to_drop = ['City','County','Zip','Job','TimeZone', 'State', 'Churn',
            'Lat', 'Lng', 'UID', 'Customer_id','Interaction', 'CaseOrder',
            'Item1','Item2','Item3','Item4','Item5','Item6','Item7','Item8']
df_clean = df_clean.drop(columns=cols_to_drop)
```

Convert Yes/No to Bool. Convert Yes/No columns to True/False and then retype to Boolean.

#define categorical data with yes/no values cat_yes_no_cols = ['Churn', 'Techie', 'Port_modem', 'Tablet', 'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup','DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'PaperlessBilling'] # for each yes_no column, replace and retype for c in cat_yes_no_cols: df_clean[c] = df_clean[c].replace({"No":False, "Yes":True}) df_clean[c] = df_clean[c].astype('bool')

In [11]:
```
# describe data by datatype
describe_dataframe_type(df_clean)
```

1. Population is numerical (CONTINUOUS) - type: int64.
  Unique: [0, 2, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25,
26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 32801, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 4
9, 50, 51, 52, 53, 54, 55, 56, 57, 58, 32825, 59, 61, 62, 63, 64, 60, 66, 67, 68, 69, 70, 71, 7
2, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 9
6, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 1
16, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 13
5, 136, 32901, 138, 137, 140, 141, 142, 143, 139, 145, 146, 144, 148, 149, 147, 151, 152, 153, 1
54, 155, 156, 157, 158, 159, 160, 161, 32929, 163, 162, 165, 166, 167, 168, 169, 170, 171, 172,
173, 174, 175, 32944, 177, 178, 179, 180, 176, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191,
192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 2
11, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 230, 23
1, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 33014, 247, 248, 249, 2
50, 246, 252, 253, 254, 255, 256, 257, 251, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 26
9, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 33049, 282, 281, 284, 285, 286, 287, 2
88, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 307, 308, 309, 31
0, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 65853, 327, 328, 329, 3
30, 331, 332, 333, 33103, 335, 336, 338, 337, 340, 341, 342, 339, 344, 345, 346, 343, 348, 349,
350, 351, 352, 353, 354, 347, 98660, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368,
369, 370, 371, 372, 373, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 388, 389, 3
90, 391, 392, 393, 394, 395, 396, 397, 398, 399, 33168, 401, 402, 403, 404, 405, 406, 407, 408,
400, 410, 411, 412, 413, 414, 415, 417, 418, 419, 420, 421, 422, 423, 424, 426, 427, 428, 429, 4
30, 431, 432, 433, 434, 437, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 450, 451, 452, 45
3, 454, 456, 457, 33225, 459, 460, 458, 463, 464, 465, 466, 467, 468, 469, 470, 471, 33240, 473,
474, 475, 476, 472, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 4
94, 496, 497, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 51
5, 516, 517, 66053, 33287, 520, 521, 519, 523, 524, 518, 66056, 527, 522, 529, 530, 528, 531, 53
3, 534, 535, 532, 537, 539, 540, 541, 543, 544, 545, 547, 548, 549, 550, 551, 553, 554, 555, 55
6, 558, 559, 560, 33328, 562, 563, 564, 561, 566, 567, 568, 569, 570, 571, 573, 574, 575, 577, 5
78, 579, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 597, 598, 59
9, 601, 602, 603, 33372, 604, 605, 607, 608, 609, 610, 611, 33380, 613, 612, 615, 617, 618, 619,
620, 621, 622, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 6
40, 641, 642, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 33420, 655, 654, 657, 658, 659,
660, 661, 656, 664, 665, 666, 667, 668, 33435, 670, 671, 672, 673, 674, 33442, 676, 677, 678, 67
9, 680, 681, 682, 683, 684, 686, 687, 688, 689, 690, 692, 693, 694, 695, 696, 697, 698, 699, 70
0, 701, 702, 703, 705, 706, 707, 708, 710, 711, 712, 713, 714, 715, 33484, 718, 719, 722, 723, 7
24, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 74
3, 744, 745, 746, 748, 749, 751, 752, 753, 754, 756, 757, 758, 759, 760, 761, 762, 763, 764, 76
5, 766, 767, 768, 770, 771, 772, 774, 776, 777, 778, 779, 780, 781, 782, 33551, 784, 785, 786, 7
87, 788, 790, 791, 793, 794, 796, 797, 798, 799, 800, 801, 802, 803, 805, 807, 808, 809, 810, 81
1, 813, 814, 815, 816, 817, 819, 820, 821, 822, 823, 824, 825, 164, 827, 828, 826, 830, 829, 83
2, 831, 834, 836, 837, 838, 839, 840, 841, 843, 33612, 845, 846, 844, 848, 851, 852, 853, 855, 8
56, 33626, 859, 860, 861, 863, 865, 867, 868, 870, 871, 872, 873, 875, 876, 877, 878, 879, 880,
33649, 882, 883, 884, 885, 886, 887, 889, 890, 892, 33661, 893, 895, 896, 897, 898, 899, 900, 90
1, 902, 903, 904, 905, 907, 910, 911, 36, 913, 181, 915, 916, 33685, 918, 917, 920, 921, 919, 92
3, 924, 925, 926, 927, 929, 931, 932, 933, 934, 936, 937, 938, 939, 940, 941, 942, 943, 944, 94
5, 946, 947, 948, 949, 950, 952, 953, 954, 955, 956, 33725, 958, 959, 960, 961, 33723, 963, 964,
965, 966, 967, 968, 969, 970, 33735, 972, 973, 974, 975, 976, 978, 979, 980, 982, 983, 984, 985,
986, 987, 988, 989, 990, 991, 33760, 993, 994, 66531, 995, 996, 998, 999, 1000, 1001, 1002, 100
3, 1004, 1005, 1006, 33769, 1010, 1011, 1012, 1013, 1015, 1016, 1018, 1019, 1020, 33786, 1024, 1
025, 1026, 1027, 1028, 1029, 1030, 1032, 1033, 1034, 1038, 1039, 1040, 1041, 1042, 1043, 1044, 1
045, 1046, 66583, 1049, 1051, 1052, 1056, 1057, 1058, 1059, 1060, 1061, 1062, 1063, 1065, 1066,
1067, 1068, 1069, 1070, 1071, 1073, 1074, 1075, 1076, 1077, 1078, 1079, 1080, 1081, 1082, 1083,
1084, 1085, 1087, 1088, 1089, 1090, 1092, 1095, 1096, 1098, 1099, 1101, 1103, 1104, 1105, 1106,
33875, 1108, 1107, 1110, 1111, 1112, 1113, 1114, 1116, 1117, 1118, 1119, 1120, 1121, 1122, 1123,
1124, 1125, 1126, 1128, 1129, 1130, 1131, 1132, 33901, 1134, 1135, 1136, 1139, 1141, 1142, 1143,
1144, 1145, 1148, 1149, 1150, 1151, 1152, 1153, 1154, 1155, 1156, 1157, 1158, 1159, 1160, 1162,
1163, 1164, 1165, 1166, 1167, 1168, 1169, 1170, 1171, 1173, 1174, 1175, 1176, 1177, 1178, 1179,
1181, 1182, 1185, 1186, 1187, 33953, 1191, 33960, 1193, 1194, 1195, 1197, 33965, 1199, 1198, 120
1, 1202, 1203, 1204, 1206, 1207, 1208, 1209, 1210, 1211, 1212, 1213, 1215, 1219, 1220, 1223, 122
4, 1225, 1227, 1231, 1232, 1233, 1234, 1236, 1238, 1239, 1240, 1241, 1242, 1243, 1245, 1246, 328
12, 1249, 1251, 1253, 1256, 1257, 1259, 1260, 1261, 1262, 1263, 1264, 1265, 1267, 1268, 1269, 12

70, 1271, 1272, 1273, 1274, 1276, 1277, 1278, 1280, 1282, 1283, 1284, 1285, 1286, 1288, 34056, 1
289, 1292, 1295, 1296, 1298, 258, 1300, 1301, 1302, 1303, 1305, 1306, 1307, 1310, 1312, 34082, 1
315, 1316, 1317, 1319, 1321, 1324, 1325, 1326, 1327, 1329, 1330, 1331, 1332, 1333, 34100, 1336,
1337, 1338, 1339, 1341, 1343, 1344, 1345, 1346, 1347, 1348, 1349, 1350, 34119, 1352, 1353, 1351,
1355, 1357, 1358, 1359, 1360, 1363, 1364, 1367, 34135, 1369, 1370, 1371, 1372, 1368, 1373, 1375,
1376, 1378, 1380, 1383, 1385, 1387, 1388, 1389, 1393, 1394, 1397, 1398, 1399, 1400, 1401, 1402,
1404, 1405, 1406, 1407, 1408, 1409, 1410, 1411, 1412, 1413, 1414, 1415, 1416, 1418, 1419, 1420,
1421, 1423, 34191, 1425, 1426, 1427, 1428, 1429, 1430, 1431, 1432, 1433, 1434, 1436, 1437, 1439,
1442, 1443, 1444, 1445, 1446, 1448, 1450, 1451, 1452, 1454, 1455, 1457, 1458, 1460, 1463, 1464,
1468, 1469, 34237, 1471, 1470, 1474, 1475, 1476, 1477, 1478, 1480, 1482, 1483, 1484, 1485, 1488,
1489, 1490, 1491, 1493, 1496, 1499, 1501, 1503, 34271, 1505, 1506, 1507, 1508, 1509, 34276, 151
2, 1516, 1517, 67053, 1519, 1518, 1521, 1520, 1523, 1524, 1526, 1528, 1530, 1531, 1532, 1533, 15
35, 1536, 34305, 1538, 1539, 34308, 1541, 1537, 1543, 34312, 1544, 34313, 1547, 1548, 34315, 155
0, 1551, 1554, 1557, 1559, 1560, 1562, 1564, 1565, 1566, 1567, 1569, 1570, 1571, 1572, 1574, 157
6, 1577, 1578, 1579, 1581, 1583, 1584, 1585, 34355, 1588, 1589, 1590, 34359, 1592, 1593, 1594, 1
595, 1596, 34365, 1598, 1597, 1600, 1601, 1602, 1599, 1604, 1605, 1611, 1612, 34381, 1614, 1616,
1618, 1619, 34390, 1622, 1623, 1624, 1626, 1627, 1629, 1630, 1634, 1635, 1639, 1640, 1641, 1643,
1645, 1649, 1651, 1652, 1655, 1656, 1659, 1660, 1662, 1663, 1665, 1666, 1668, 1669, 1672, 1673,
1674, 1676, 1677, 1678, 1679, 1680, 1684, 1685, 1686, 1687, 1690, 1691, 1692, 34460, 1693, 1694,
1696, 1697, 1700, 1702, 34470, 1704, 1705, 1706, 1703, 1708, 1709, 1711, 1712, 1714, 1715, 1716,
1718, 1719, 34488, 1720, 1722, 1723, 1724, 1725, 34496, 1730, 1733, 1735, 1736, 34506, 1740, 174
1, 1747, 1748, 1749, 1750, 34519, 1751, 1753, 1755, 1756, 1757, 1759, 1760, 1761, 1765, 1766, 17
67, 1769, 1770, 1771, 1772, 1773, 1775, 1776, 34545, 1780, 1782, 1783, 1784, 1785, 1786, 1788, 1
789, 1791, 1792, 356, 1796, 1797, 1799, 1800, 1803, 1808, 1811, 1812, 1814, 1815, 1817, 1819, 18
22, 1823, 1826, 1827, 1829, 1830, 1831, 1833, 1835, 1837, 1838, 1839, 1840, 1841, 1842, 1843, 18
45, 1848, 1851, 1853, 1854, 1855, 1856, 34624, 1860, 1861, 1864, 1865, 1868, 1869, 1870, 34639,
1871, 1874, 1875, 34642, 1878, 1879, 1883, 1886, 1888, 1889, 1892, 1893, 1894, 1897, 1898, 1899,
34669, 1902, 1903, 1906, 1907, 34675, 1909, 1910, 1911, 1912, 1913, 1916, 1917, 1918, 1919, 192
0, 1921, 1922, 1924, 1925, 1926, 1927, 1928, 1932, 1935, 1936, 1938, 1939, 1940, 1947, 1948, 194
9, 1955, 1956, 1959, 1961, 1962, 1966, 1968, 1970, 1972, 1973, 34744, 1976, 1978, 1980, 1983, 32
959, 1986, 1988, 1990, 1991, 1992, 1994, 1995, 1997, 1998, 2000, 2001, 2003, 2004, 2006, 2007, 2
008, 2011, 2012, 2013, 34781, 34784, 2017, 2018, 2019, 2016, 2023, 34791, 2025, 34794, 2027, 202
8, 2029, 2031, 2032, 2033, 2034, 2035, 2041, 2043, 2047, 2049, 2050, 2051, 2052, 2053, 2055, 205
6, 2057, 2058, 2060, 2064, 2065, 2067, 2068, 2072, 2073, 2074, 2076, 2077, 2079, 2080, 2081, 208
2, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2093, 2095, 2097, 2098, 2099, 2101, 2103, 2104, 210
5, 2106, 2107, 2108, 2109, 2112, 2114, 2115, 2119, 2121, 2122, 2123, 2124, 2127, 2128, 2129, 213
0, 2131, 2132, 2135, 2136, 2137, 2138, 2139, 2141, 2142, 2144, 2145, 2146, 2148, 34918, 2153, 21
56, 2157, 2158, 2159, 34928, 2161, 2164, 2166, 2168, 2171, 2172, 34941, 2174, 2175, 34945, 2178,
2179, 2180, 2181, 34950, 2186, 2190, 2191, 2192, 2193, 33000, 34962, 2196, 2194, 2201, 2205, 220
9, 2215, 2216, 2217, 2220, 2221, 2223, 34993, 2226, 2227, 2225, 2229, 2228, 2231, 2230, 2233, 22
38, 2240, 2242, 2243, 2245, 2246, 2247, 2249, 2252, 2258, 35028, 2263, 2267, 33015, 2270, 2271,
2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2282, 2285, 35054, 2287, 2288, 2292, 35061, 229
3, 2295, 2297, 2299, 2301, 2302, 2304, 2305, 2306, 2308, 2309, 2310, 2311, 2315, 2317, 2318, 350
87, 2319, 2325, 2326, 2328, 2331, 35100, 2333, 35102, 2335, 2336, 2337, 2338, 2334, 35107, 3510
1, 2345, 2348, 2349, 2350, 33031, 35118, 2353, 2352, 35124, 2357, 2358, 2363, 2368, 2370, 2371,
35142, 2377, 2379, 2381, 2382, 35151, 2383, 2385, 2387, 2388, 35157, 2390, 2389, 2393, 2394, 239
5, 2396, 2397, 35161, 2400, 2401, 2402, 2405, 2408, 2409, 2413, 2416, 2419, 2421, 2423, 2424, 24
30, 2431, 2432, 2433, 2435, 2439, 2440, 2442, 2451, 2453, 2454, 2455, 2459, 35230, 2464, 2465, 2
467, 2468, 2469, 2472, 2473, 2474, 2475, 2476, 35240, 2479, 2481, 2482, 2484, 2487, 2489, 2491,
2494, 2498, 2499, 2500, 2502, 35270, 2510, 35279, 2514, 2515, 2520, 2521, 2525, 2526, 2529, 253
2, 2534, 2535, 2536, 2539, 2540, 2544, 35317, 2550, 2551, 2553, 2557, 2558, 2562, 2564, 2565, 25
68, 2571, 2576, 35345, 2578, 2579, 2580, 35349, 2581, 2577, 2589, 68128, 2593, 2592, 2594, 2596,
2604, 35373, 2605, 35376, 2608, 2610, 2609, 2613, 2616, 2617, 2618, 2621, 35389, 2623, 2624, 262
5, 2634, 2638, 2639, 2640, 2643, 35414, 526, 2649, 2653, 2656, 2660, 2662, 2663, 2667, 35436, 26
69, 2668, 2673, 2674, 2677, 2678, 2680, 2681, 2682, 2683, 2684, 2688, 35457, 2690, 2689, 2692, 2
694, 35462, 2696, 35465, 65665, 2698, 2701, 2703, 2705, 2706, 2707, 2709, 2710, 2712, 2713, 271
5, 2716, 2717, 2718, 2719, 35489, 2722, 2723, 2725, 2732, 2733, 2737, 2738, 2739, 2744, 2745, 27
46, 2747, 2750, 2753, 35523, 2755, 2757, 2759, 2762, 2763, 68300, 2765, 2771, 2775, 2778, 2779,
2780, 2781, 2782, 2783, 2788, 2789, 2792, 2800, 2801, 2802, 35573, 2806, 35575, 2808, 2810, 281
3, 2815, 2816, 2817, 2820, 2821, 35591, 2824, 2830, 2831, 2834, 35603, 2835, 2838, 2840, 2841, 2
844, 2845, 2846, 2848, 2850, 2854, 2856, 2857, 2860, 2862, 2863, 2864, 2865, 2869, 2871, 2872, 2
874, 2875, 2876, 35643, 2878, 2879, 2883, 2885, 2886, 2889, 2892, 2898, 2901, 2905, 2907, 2909,

2912, 35681, 2914, 2913, 2916, 2917, 2918, 2919, 2920, 2922, 2924, 2926, 2930, 2931, 2932, 2934, 35702, 2935, 2939, 2942, 2945, 2946, 2948, 2950, 2951, 2952, 2953, 2956, 2958, 2961, 2962, 2963, 2964, 2965, 2966, 2967, 2968, 2973, 35743, 2976, 2978, 2981, 2983, 2984, 2986, 2988, 2989, 2990, 2992, 35763, 2997, 2998, 2999, 3001, 35771, 3003, 3006, 3011, 3014, 3015, 3027, 3028, 3029, 3579 8, 35799, 3032, 3033, 3034, 3039, 3050, 3052, 35824, 3058, 3060, 3064, 3066, 3067, 3068, 35840, 3072, 3074, 3076, 3078, 3079, 3080, 3081, 3085, 3088, 3089, 3091, 3093, 3094, 3096, 3097, 3098, 3099, 3101, 3102, 3104, 3107, 3108, 3109, 3111, 3112, 3114, 35885, 3118, 3119, 3122, 3123, 3124, 3128, 3133, 3134, 3137, 3139, 68676, 3141, 3142, 3144, 35913, 3149, 3150, 3151, 35923, 3155, 315 7, 3160, 3161, 3163, 3167, 3170, 3171, 3172, 3177, 3182, 35950, 35952, 3185, 3191, 3192, 3195, 3 196, 3199, 3200, 3208, 3209, 3211, 3213, 3215, 3216, 3219, 3220, 3223, 3224, 3225, 3226, 3227, 3 230, 3231, 3232, 35999, 3239, 3241, 3242, 3245, 3248, 3250, 3251, 3255, 3256, 3263, 3264, 3268, 3269, 3271, 3274, 3275, 3278, 36047, 3281, 3282, 3284, 3291, 36062, 3294, 3298, 3299, 3300, 330 2, 3306, 3308, 3313, 3316, 3317, 3318, 3322, 3324, 3326, 3327, 3334, 3337, 3340, 3345, 3347, 335 1, 3353, 3357, 3365, 3368, 3370, 36138, 3373, 3374, 33235, 3378, 3381, 3383, 675, 3400, 3403, 36 172, 3404, 3406, 3408, 3410, 3412, 3414, 36184, 3417, 3422, 3423, 3424, 3431, 3433, 36202, 3437, 3441, 3443, 3445, 3448, 3454, 36224, 3456, 3461, 3463, 3473, 36245, 3479, 3480, 3481, 36260, 349 2, 3494, 3496, 36265, 3498, 3499, 3502, 3503, 3504, 36282, 3515, 3518, 3519, 3521, 3522, 3525, 3 6293, 3527, 3526, 3531, 3535, 3543, 36312, 3546, 3547, 3551, 3558, 3560, 3564, 36335, 3572, 357 4, 3576, 3579, 3583, 3586, 3587, 3590, 3591, 3592, 3594, 3599, 69142, 3607, 3609, 3612, 3613, 36 16, 3620, 3622, 3628, 3632, 3634, 3636, 3639, 36409, 3643, 3645, 36414, 3649, 3650, 3651, 36423, 3658, 3662, 3667, 3671, 3672, 36441, 3678, 3681, 3682, 3685, 36455, 3690, 3691, 3696, 3697, 370 0, 3703, 3707, 3708, 3709, 3716, 3718, 36489, 3730, 3734, 3735, 3736, 3738, 3739, 3744, 3745, 37 47, 3748, 3752, 3755, 3756, 3757, 3758, 3759, 3761, 36531, 3764, 3766, 3767, 3768, 3769, 36539, 3776, 3777, 36550, 3783, 3787, 3796, 3801, 3802, 3803, 3809, 3810, 3812, 3814, 3817, 3818, 3821, 3826, 3831, 3834, 3837, 3841, 3842, 3843, 3845, 36613, 3852, 3853, 3854, 3860, 3866, 3867, 3873, 3874, 3875, 3876, 69413, 3878, 3879, 3880, 3881, 3883, 3884, 3891, 3892, 3895, 3898, 3899, 3903, 3908, 36681, 3916, 3929, 3934, 3937, 36711, 3945, 3949, 3951, 3956, 3957, 36724, 36731, 3966, 39 68, 3975, 36753, 3986, 3988, 3989, 3991, 3993, 4001, 4003, 4004, 4005, 4006, 4007, 4010, 4013, 4 022, 36792, 4024, 4027, 4029, 4030, 4035, 4037, 4038, 4046, 4048, 36817, 4049, 4052, 69589, 405 9, 4066, 4072, 4075, 4082, 4083, 4085, 4088, 4089, 4101, 4102, 4108, 4112, 4125, 4127, 4128, 102 433, 4132, 4134, 4139, 4141, 4143, 4152, 4154, 36923, 4155, 4158, 4160, 4164, 4173, 4175, 4177, 4179, 4180, 4183, 36959, 4193, 36962, 4201, 4204, 4206, 4207, 4210, 4212, 4216, 36986, 4223, 422 5, 4228, 4231, 4235, 4242, 4243, 4245, 4246, 4250, 4251, 4260, 4261, 37035, 4269, 4270, 37042, 4 277, 4278, 4283, 4286, 37056, 4288, 4292, 4295, 4298, 37067, 4300, 4301, 4308, 37080, 4316, 432 6, 4334, 4335, 4339, 4343, 4350, 4351, 4352, 37122, 4354, 4356, 4357, 4358, 4363, 4365, 4366, 43 70, 4381, 4382, 4384, 4388, 4389, 4390, 4395, 4401, 4402, 4403, 4406, 4407, 4408, 4409, 37183, 4 420, 4425, 37201, 4434, 4435, 4436, 37204, 37207, 4442, 4444, 4447, 4455, 4465, 4466, 37234, 447 0, 4476, 4477, 4478, 4479, 4480, 4483, 4487, 4489, 4495, 4498, 4500, 4501, 4504, 4508, 4513, 451 8, 4522, 4528, 4535, 4543, 4546, 4552, 4553, 4555, 4566, 4572, 4575, 4584, 4585, 4586, 912, 3735 8, 4591, 37360, 4595, 4596, 4597, 37375, 37377, 4611, 4612, 4613, 4621, 4622, 4625, 4631, 4633, 4634, 4637, 4641, 4642, 4644, 4646, 4649, 4651, 4654, 4655, 70199, 4667, 4685, 4689, 4695, 3746 4, 4696, 4698, 4704, 4705, 37476, 4709, 4710, 4711, 4712, 4715, 4719, 4721, 37492, 4731, 4734, 4 739, 4745, 4747, 4755, 4757, 4759, 37527, 4761, 37533, 4767, 37540, 4776, 4777, 4779, 4787, 479 3, 4799, 4800, 4802, 4807, 4813, 4818, 4821, 4824, 4825, 4827, 4832, 4833, 4837, 4839, 4841, 484 3, 4846, 4852, 4858, 4868, 4869, 4870, 37639, 4872, 37641, 37646, 4879, 4880, 4884, 4885, 4889, 4892, 4894, 4900, 4901, 4904, 4906, 4907, 4909, 4912, 4913, 4917, 4918, 4924, 4926, 4928, 4929, 4938, 4941, 37711, 4946, 4949, 4950, 4959, 37727, 4967, 4968, 4969, 4970, 4971, 4976, 4979, 498 1, 4982, 4983, 4986, 4988, 4990, 4991, 4993, 4994, 4996, 4998, 5005, 37774, 5009, 5012, 5015, 50 16, 5019, 5020, 5025, 5028, 5036, 5038, 5041, 5044, 5055, 5058, 5059, 5060, 5061, 5062, 5068, 50 76, 70614, 5081, 5082, 5085, 5089, 5090, 5092, 5098, 5100, 5105, 5107, 37878, 5111, 5114, 5115, 5123, 5124, 37906, 5143, 5144, 5148, 5157, 5161, 5168, 5176, 5179, 5182, 5184, 5185, 5188, 5190, 5192, 5196, 5199, 37976, 5210, 5211, 5221, 5230, 5234, 5237, 5239, 5241, 38013, 38016, 5256, 525 9, 38028, 5264, 5273, 5275, 5278, 5282, 5283, 5284, 5287, 5288, 5292, 5295, 5300, 5302, 5303, 38 072, 38073, 5307, 5308, 5310, 5316, 5325, 5332, 5334, 5337, 5342, 5344, 5346, 5350, 5360, 5363, 5364, 5366, 5367, 5370, 5375, 5381, 5383, 5385, 5394, 5401, 5402, 5403, 5407, 38177, 5409, 5424, 5428, 5429, 5430, 103732, 5438, 5440, 5442, 5449, 5450, 5455, 5456, 5465, 5467, 5480, 5490, 549 1, 5492, 5493, 5494, 5497, 38267, 5500, 38272, 5515, 5516, 5518, 5523, 5524, 5526, 5527, 5529, 5 542, 5546, 5547, 5548, 5549, 5553, 38321, 5555, 5557, 5563, 38334, 38335, 5570, 5582, 5584, 558 5, 5586, 5587, 5590, 38363, 5596, 5600, 5601, 5602, 38376, 5615, 38383, 38388, 5621, 5622, 5633, 5638, 38406, 5639, 5642, 5643, 5646, 5651, 5652, 38424, 5659, 38436, 5670, 38443, 5682, 5684, 56 88, 5693, 5694, 5698, 5705, 5706, 38476, 5709, 5714, 5723, 5729, 5733, 5736, 5748, 38517, 38524, 5760, 5762, 5779, 38547, 38553, 5791, 5796, 38568, 5803, 38579, 5817, 5819, 5820, 5823, 71360, 3

8597, 5831, 5835, 5846, 5849, 5851, 38620, 5858, 5863, 5864, 5869, 5870, 5882, 5885, 5886, 3865
9, 5907, 5910, 5912, 5920, 5924, 5929, 5931, 5933, 5935, 38703, 5943, 5950, 5951, 5955, 5960, 59
64, 38733, 5967, 5973, 5978, 5979, 38748, 5981, 5983, 38753, 5985, 5993, 6002, 6009, 6017, 3878
7, 6019, 6020, 6022, 6023, 6030, 6032, 6037, 6039, 6042, 6047, 6049, 6050, 6055, 6060, 38833, 38
835, 6067, 6071, 38839, 6079, 6082, 38865, 6100, 6108, 38885, 6119, 6123, 6137, 6138, 6139, 614
0, 38909, 6153, 6161, 38932, 6165, 6172, 6188, 38957, 6190, 6197, 6206, 6210, 6213, 6225, 71763,
6233, 6241, 39012, 6250, 6252, 6254, 6257, 39035, 6274, 6275, 39043, 6277, 6279, 6282, 6285, 628
9, 6292, 6293, 39062, 6295, 6294, 39065, 6297, 6298, 6302, 6304, 6307, 39078, 39088, 39090, 632
8, 6335, 71878, 6343, 6349, 6354, 6359, 6366, 6376, 6380, 6384, 6393, 6394, 6399, 6402, 6403, 64
07, 6408, 6409, 6412, 6416, 6428, 6432, 6435, 6436, 6438, 6443, 6447, 6450, 39221, 6454, 6457, 3
9227, 6460, 6463, 6464, 39234, 6470, 6473, 6487, 6502, 39273, 6506, 6507, 6508, 6513, 6519, 652
1, 6524, 72061, 6533, 6538, 39309, 6543, 6544, 6551, 6553, 6569, 6570, 6571, 6572, 39344, 6581,
6585, 6592, 6604, 6608, 39376, 6610, 6612, 39381, 39384, 6623, 39391, 6627, 6633, 6635, 6637, 66
38, 6658, 6672, 6673, 39440, 6686, 6694, 6699, 6705, 6709, 39478, 6737, 6740, 6749, 6750, 6754,
39523, 6756, 6762, 6772, 72332, 39568, 6803, 6815, 6818, 6823, 6826, 39600, 6833, 6835, 6846, 39
616, 6849, 6852, 6853, 6854, 39624, 6871, 39641, 6874, 39642, 6878, 6880, 39649, 6888, 39658, 68
91, 6900, 6902, 6904, 6910, 6911, 6912, 6915, 6917, 6921, 6922, 6923, 6924, 39694, 39698, 6932,
6935, 72479, 6947, 6956, 6962, 39731, 6965, 6975, 6976, 6982, 6986, 6988, 6989, 39757, 6990, 700
4, 7008, 7011, 7015, 7016, 7017, 39786, 7020, 7023, 7033, 7050, 72592, 7058, 7064, 7070, 39842,
7079, 7080, 39850, 7088, 39859, 7093, 7104, 7109, 7114, 7117, 7122, 7134, 7139, 7147, 7154, 715
5, 7156, 7157, 7169, 7174, 7175, 7186, 7190, 7194, 39962, 7197, 7215, 7216, 7218, 7221, 7222, 39
994, 7232, 7233, 7234, 40005, 7237, 7244, 7246, 7247, 40014, 7249, 40019, 7252, 7256, 7257, 726
0, 7263, 7265, 7272, 40041, 7278, 7281, 7284, 7285, 7286, 7289, 7290, 7291, 7292, 7294, 40063, 7
302, 7329, 7330, 7334, 7346, 7349, 7350, 7351, 7354, 7360, 7362, 40131, 7368, 7371, 7379, 7380,
7390, 7397, 7398, 40170, 7404, 7405, 7410, 7413, 7415, 7430, 7436, 7439, 7453, 7456, 7458, 7462,
7470, 7477, 40248, 7482, 7485, 7488, 7496, 7511, 7514, 7519, 40290, 7527, 7530, 7531, 7532, 753
6, 40305, 7552, 7562, 7564, 7565, 40335, 40336, 40337, 7570, 40340, 7575, 7584, 7594, 7602, 761
1, 7617, 40385, 7624, 7626, 7633, 7637, 7638, 7655, 7656, 40425, 7665, 40434, 40439, 40441, 767
6, 40447, 7691, 7699, 7710, 7711, 7714, 7715, 7722, 32870, 7731, 7738, 7742, 1540, 7750, 7751, 7
749, 7753, 7762, 7772, 40543, 7780, 7782, 7785, 7786, 7789, 7790, 7797, 7799, 7801, 7802, 7811,
7825, 7836, 40604, 7842, 7847, 40622, 7862, 7863, 7870, 7875, 7878, 40649, 7883, 7884, 7885, 789
0, 7900, 7901, 7906, 7912, 40684, 40685, 7917, 7934, 7941, 7945, 7947, 7955, 7958, 7972, 7979, 4
0752, 7993, 7999, 8002, 40770, 8006, 8008, 40778, 8013, 8028, 8038, 40820, 8052, 40825, 40828, 8
060, 40830, 40837, 8071, 8075, 8081, 40857, 40860, 8093, 40866, 8100, 8103, 8109, 8112, 8116, 81
20, 8134, 73671, 8139, 8140, 8142, 8144, 8146, 8148, 8149, 8156, 8165, 8171, 8188, 8189, 8192, 8
199, 8205, 8208, 8217, 8219, 8223, 8225, 8226, 8230, 41000, 8236, 8239, 73778, 41014, 8248, 4101
8, 41026, 8261, 8272, 8284, 41053, 8294, 8295, 8303, 8304, 8307, 8324, 8332, 8335, 8339, 41107,
8345, 8354, 73891, 8368, 8376, 8377, 8379, 8381, 41149, 41151, 8385, 41155, 8397, 8402, 8407, 84
09, 8412, 8419, 8428, 8431, 41204, 8437, 8447, 41235, 8469, 8470, 8483, 8488, 8489, 8501, 41276,
8510, 8525, 8538, 8542, 8543, 8549, 8564, 8565, 8577, 8582, 8589, 8596, 8601, 8611, 8618, 41387,
8621, 8624, 41392, 8629, 8631, 74172, 41407, 8652, 8658, 8660, 8683, 8684, 8685, 41454, 8688, 86
89, 8691, 41468, 8701, 8704, 8706, 41480, 8720, 8725, 8738, 8741, 41510, 8742, 8750, 8751, 4152
3, 8759, 8761, 8769, 8770, 8774, 8777, 8778, 8784, 41563, 8800, 8810, 8811, 8819, 8823, 8826, 88
31, 41600, 8834, 8851, 8852, 8854, 8859, 8867, 8875, 74413, 8878, 8880, 8882, 41665, 8913, 8920,
8959, 41733, 8965, 41743, 8979, 8981, 8986, 8990, 9005, 9022, 9025, 9026, 9045, 9049, 41818, 905
4, 9064, 74601, 41839, 9071, 9073, 9076, 9077, 9092, 41861, 41863, 9100, 9102, 9103, 9110, 9111,
9124, 9131, 41901, 9144, 9145, 9152, 9155, 9162, 9165, 9177, 9178, 9179, 9191, 41961, 9201, 920
5, 9227, 9230, 9231, 9236, 9238, 42011, 9246, 9248, 42023, 9262, 42041, 42042, 9282, 9286, 4205
9, 9295, 9298, 9301, 9325, 9338, 9339, 42116, 9353, 9354, 9357, 9361, 9379, 9391, 42161, 9399, 9
400, 9417, 9422, 9427, 9434, 74971, 9437, 42227, 9463, 9469, 9470, 9475, 9481, 9489, 9491, 9493,
9496, 9499, 42272, 9505, 9510, 9523, 9542, 9560, 9562, 9572, 9575, 9583, 9585, 9593, 9611, 9612,
9624, 9627, 9635, 9647, 9663, 42442, 9679, 9692, 42463, 9696, 9709, 9718, 9725, 42522, 9755, 425
24, 9762, 42532, 9768, 75306, 9771, 9774, 9777, 9791, 42582, 9826, 9827, 9836, 42614, 9852, 986
9, 9871, 9876, 42650, 9899, 9906, 9908, 9909, 9910, 9916, 9922, 9924, 9930, 9932, 9936, 9948, 99
52, 9956, 9957, 42741, 9974, 9975, 9982, 9986, 9996, 9998, 10011, 10012, 10019, 10025, 10026, 10
030, 10034, 42812, 10045, 10052, 10056, 10057, 75635, 42868, 10109, 10115, 10119, 10122, 10134,
42910, 10148, 10152, 10184, 10185, 10189, 10211, 42979, 10218, 42991, 10228, 10232, 10235, 1023
7, 10238, 10243, 10246, 10268, 10274, 10294, 43078, 10314, 10321, 43112, 10347, 43116, 43123, 43
149, 10382, 10389, 10395, 10398, 10423, 10424, 10429, 10434, 10440, 43208, 43209, 10445, 10446,
10450, 10451, 10459, 43232, 10480, 10481, 10482, 10489, 10494, 10499, 10501, 10509, 10511, 1051
2, 10516, 10531, 10536, 10546, 10550, 10552, 10577, 10586, 10587, 10605, 10614, 10621, 10622, 10
626, 76177, 10652, 10653, 10667, 10692, 43461, 10695, 10712, 10720, 10727, 10728, 10729, 10741,

10777, 10780, 43551, 10790, 10805, 10810, 10815, 10818, 10821, 10826, 10827, 43597, 10836, 1084
2, 10856, 43633, 10873, 10874, 10885, 10891, 10895, 10896, 10898, 10911, 10927, 10928, 43714, 10
949, 10951, 10963, 43736, 10976, 10977, 10980, 10987, 10989, 11022, 43795, 43801, 11054, 11092,
11097, 11100, 11106, 43881, 11116, 11118, 11123, 11142, 11147, 11158, 11161, 11173, 11174, 1118
8, 11195, 11196, 11197, 11202, 11217, 11237, 11239, 11245, 11254, 11268, 11272, 11275, 76811, 76
819, 11289, 11294, 11295, 11300, 11317, 11323, 11333, 44103, 11335, 11342, 11346, 11352, 11357,
11364, 11371, 44139, 44145, 11384, 11393, 11396, 11400, 11435, 76973, 11446, 11455, 11461, 4423
3, 11466, 44239, 11472, 44249, 11484, 11499, 11528, 11530, 44300, 11537, 11540, 11543, 11557, 44
328, 11561, 11573, 11582, 44352, 11590, 11595, 11598, 11601, 11611, 11627, 11629, 11631, 77168,
11634, 11647, 11656, 11658, 11677, 44451, 44455, 44460, 44468, 11702, 44474, 11722, 11725, 2332,
11734, 11740, 11741, 11761, 11762, 11785, 77342, 11816, 11817, 11846, 11848, 11850, 11854, 1186
5, 11866, 44641, 44647, 11880, 11885, 11895, 11897, 11908, 11910, 11923, 11937, 11941, 11949, 44
740, 11974, 44750, 11988, 44759, 44760, 11996, 12047, 44826, 12058, 12063, 12067, 12078, 12089,
12092, 12096, 12099, 12135, 12141, 12153, 12158, 12162, 12181, 44968, 12230, 12237, 12275, 1227
9, 12300, 12301, 12326, 12327, 12338, 12349, 12374, 12389, 45158, 12390, 12396, 12404, 12407, 12
414, 12415, 12418, 12424, 45193, 12439, 12447, 12449, 12453, 12478, 12492, 12530, 45298, 12531,
12541, 12562, 12570, 12577, 12590, 12592, 12628, 12645, 12658, 12659, 12672, 12683, 12695, 1270
5, 12709, 45487, 12722, 12726, 12727, 12737, 12742, 12749, 12760, 12768, 12789, 12791, 12802, 12
808, 12823, 45596, 12830, 12846, 12853, 12864, 12874, 45652, 12886, 12895, 12908, 12911, 12914,
12918, 12919, 12920, 45688, 12923, 45694, 12926, 12954, 12968, 12991, 45763, 13000, 13001, 1300
5, 13015, 13017, 13019, 13020, 13055, 13061, 13077, 13088, 13111, 13112, 45885, 13120, 13124, 13
128, 13132, 13133, 45910, 13156, 13166, 13168, 13178, 13181, 13190, 13202, 13216, 13221, 13222,
45995, 13229, 13243, 13245, 13247, 46024, 13259, 13265, 13266, 13268, 13269, 46041, 13287, 1328
8, 46064, 46087, 13351, 78900, 13365, 13384, 13410, 13443, 46215, 13448, 13474, 79010, 13475, 13
484, 13487, 13489, 13490, 13493, 46276, 46286, 46299, 13535, 13538, 13546, 13547, 111850, 13549,
13553, 46327, 13564, 46333, 13568, 13569, 46344, 13587, 13601, 13607, 13622, 13644, 13649, 4641
8, 13659, 13678, 13680, 13712, 13715, 13719, 13720, 13734, 13736, 79276, 13744, 13745, 46513, 13
757, 13762, 13768, 46545, 13788, 13789, 13805, 13809, 46581, 46589, 46623, 13863, 13864, 13865,
46647, 13892, 13895, 13905, 46675, 13915, 13920, 13932, 13937, 13958, 13961, 13970, 13977, 1398
1, 13991, 46775, 14018, 14019, 14025, 14026, 14037, 14055, 46869, 14103, 46875, 14112, 14119, 46
890, 46904, 46920, 14182, 14191, 14194, 14208, 14210, 14225, 46995, 14242, 14243, 14260, 14267,
14269, 14279, 47067, 14305, 14312, 14318, 14323, 14328, 14334, 14357, 14381, 14386, 14393, 1441
3, 14429, 14433, 14438, 14440, 47227, 79996, 14464, 14472, 14479, 14523, 14524, 14534, 47309, 14
550, 14555, 14557, 14559, 14569, 47346, 14580, 47367, 14601, 14606, 80143, 14619, 14630, 14632,
14634, 14644, 14645, 14657, 14659, 14673, 14676, 14684, 14687, 47459, 14693, 14696, 14700, 1471
0, 14719, 14723, 47493, 80264, 14744, 14790, 14795, 14796, 14801, 14802, 14823, 14839, 14850, 14
852, 14863, 47645, 14892, 14918, 14928, 14931, 14942, 47710, 14945, 80489, 47732, 14970, 14980,
14985, 47757, 14990, 14989, 80555, 47791, 15033, 15041, 15059, 15062, 15074, 15075, 15099, 1511
3, 15119, 15129, 15133, 15142, 15156, 15171, 15183, 15184, 15201, 47974, 15210, 15226, 15232, 48
008, 15247, 15251, 15260, 15261, 15288, 15289, 15299, 15322, 15326, 48099, 15364, 48133, 15379,
15383, 15391, 15400, 15415, 15420, 15436, 15477, 15515, 15517, 15521, 15529, 15535, 15560, 1556
9, 48351, 15586, 15589, 15591, 15599, 15603, 15604, 48382, 48386, 15623, 15644, 15650, 15652, 15
657, 15663, 15666, 48438, 15671, 15684, 15695, 15696, 15702, 15703, 15713, 15715, 15721, 15722,
15724, 15725, 15728, 15738, 15741, 15762, 15763, 15786, 15787, 15788, 15806, 15812, 15816, 1582
2, 15836, 15848, 15882, 15900, 15929, 48710, 15944, 48713, 15956, 15964, 15979, 15988, 48770, 48
775, 16019, 16022, 16062, 48833, 16079, 16080, 48854, 16088, 48863, 16101, 16110, 48905, 16160,
16163, 16169, 16171, 16177, 16182, 16192, 16194, 16200, 48969, 48990, 16228, 16243, 16246, 1624
9, 49019, 81789, 16254, 16299, 16305, 16350, 16352, 16357, 16371, 16373, 16374, 16379, 16382, 49
158, 16394, 16395, 16398, 16405, 16413, 16425, 16427, 16434, 16455, 16459, 16479, 16487, 16488,
16501, 16503, 49271, 16514, 16515, 16518, 16522, 16525, 16540, 16545, 16546, 16548, 49344, 1657
8, 49350, 16591, 16592, 49397, 16630, 16629, 16638, 16644, 16649, 16659, 16665, 16681, 16690, 16
715, 49487, 82257, 16760, 16783, 16785, 16791, 16812, 16819, 49612, 16845, 16850, 16853, 49630,
16882, 16888, 49657, 16894, 49664, 16897, 16902, 16906, 49694, 16943, 16955, 16968, 16972, 1697
4, 17007, 49787, 49790, 17042, 17052, 17055, 17058, 49836, 17086, 17092, 17094, 49870, 17107, 49
877, 17119, 49894, 49907, 17141, 17145, 17148, 17191, 17208, 17218, 17242, 17243, 17246, 17248,
17274, 17287, 17307, 50079, 17334, 17348, 17349, 17351, 17356, 50137, 17370, 17389, 17398, 1744
8, 17452, 17473, 17479, 17484, 17486, 17493, 17502, 17516, 50288, 17536, 17543, 50328, 17563, 17
576, 17585, 17587, 17600, 17616, 50386, 17620, 17639, 17642, 83179, 17644, 17653, 17654, 17666,
17694, 17696, 17701, 17713, 17732, 17733, 17766, 17768, 17783, 17801, 17806, 17808, 17823, 1787
2, 50640, 17879, 17881, 17907, 17909, 17920, 17923, 17989, 17990, 17999, 18015, 18029, 18042, 18
043, 18065, 18074, 18075, 83615, 18090, 50869, 18103, 18110, 18141, 18170, 18181, 18184, 18185,
18188, 18192, 18198, 50972, 18208, 18210, 18213, 18216, 50992, 18229, 51003, 83782, 18249, 1825

2, 18272, 18274, 51045, 18281, 51069, 18301, 18302, 18304, 51082, 18321, 18322, 51097, 18346, 51
117, 18353, 18372, 18377, 18389, 18391, 18402, 18422, 18427, 18448, 18465, 51275, 18518, 18529,
18532, 18533, 18540, 18547, 18575, 18578, 18579, 18580, 18586, 51369, 18609, 18617, 18621, 1862
5, 51426, 18674, 51447, 18691, 18720, 18722, 18731, 18745, 18756, 18778, 18813, 51582, 18820, 18
828, 18829, 18856, 18869, 18875, 18889, 18894, 18899, 51667, 18914, 18937, 51706, 18945, 18951,
18975, 18977, 18978, 18998, 51767, 19001, 19039, 19073, 19092, 19097, 19100, 19135, 19141, 1914
6, 19151, 19162, 19165, 19168, 19201, 19215, 19217, 19237, 19255, 19270, 19283, 19301, 52078, 19
324, 19345, 19350, 19358, 19361, 19389, 19390, 19393, 19401, 19403, 19414, 19415, 19419, 19432,
19443, 19461, 19463, 52254, 19522, 19527, 52299, 19535, 19540, 19546, 19551, 52322, 19556, 1955
9, 19569, 19578, 19582, 19586, 19592, 19593, 52371, 52394, 19637, 19649, 19650, 19659, 19669, 52
454, 19704, 19706, 19713, 52484, 19717, 19721, 19735, 19746, 19747, 19773, 19806, 19843, 19855,
19858, 19866, 19869, 52669, 19904, 19912, 19913, 19919, 52688, 19936, 19942, 19943, 19951, 1998
0, 20006, 20007, 20008, 20021, 52802, 20044, 20049, 20077, 20136, 20150, 52922, 20163, 20167, 52
939, 20176, 20180, 20193, 20195, 52967, 20215, 20234, 20235, 20258, 20269, 20297, 20300, 20306,
20312, 20322, 20327, 53098, 20338, 53111, 53140, 53142, 20375, 20435, 20449, 20450, 53266, 2050
1, 20537, 20543, 20558, 20586, 53364, 20602, 20604, 20606, 20609, 20631, 53431, 20676, 20681, 20
695, 20696, 20715, 20731, 20759, 20760, 20765, 20766, 53552, 53582, 20818, 20825, 20872, 20884,
86421, 20897, 53686, 20920, 20925, 20926, 20929, 20935, 20940, 20943, 20952, 20958, 21013, 2103
8, 21044, 21066, 53841, 21095, 53866, 21105, 21111, 53879, 53886, 21129, 21130, 21131, 21137, 21
141, 21156, 86703, 21177, 21184, 21202, 21204, 21212, 21218, 21226, 54023, 86811, 21276, 21282,
21301, 54081, 21314, 21319, 21331, 54101, 21343, 21353, 54150, 86926, 21394, 21408, 21409, 5419
0, 21434, 21448, 21462, 21480, 21519, 21529, 21556, 21572, 21578, 21589, 21605, 21607, 21609, 21
628, 54413, 21681, 54464, 87240, 21732, 21734, 21738, 21739, 54507, 21743, 21750, 21763, 21765,
21767, 54540, 21792, 21801, 21804, 21825, 21829, 21833, 54601, 21847, 54621, 21854, 21869, 2188
8, 21895, 21900, 21939, 21943, 21952, 21962, 21970, 87509, 21981, 21997, 54780, 22014, 22028, 22
052, 22059, 54829, 22073, 22096, 22103, 22115, 22127, 22152, 22187, 22201, 22202, 22225, 22254,
22255, 22258, 22273, 22274, 22275, 22294, 55122, 22356, 22380, 22385, 22388, 22405, 22425, 5521
0, 22447, 55215, 55222, 22492, 22499, 22529, 22544, 22547, 22548, 22553, 22569, 22574, 55352, 22
602, 22609, 22640, 22684, 22704, 22733, 22734, 22738, 22744, 55519, 22766, 22795, 88344, 22811,
88349, 22825, 22836, 22840, 22849, 55623, 22858, 55652, 22892, 22924, 22925, 22929, 22935, 2293
9, 22947, 55737, 23021, 23029, 23052, 23064, 23078, 55857, 23116, 23117, 23140, 23144, 23155, 23
160, 55937, 23191, 23200, 23230, 23247, 23256, 23285, 23298, 23329, 23354, 23361, 23370, 23373,
23381, 23402, 23425, 23438, 23458, 56233, 23474, 23512, 56284, 23517, 23526, 23530, 89075, 5630
8, 23590, 23612, 23618, 56406, 23666, 56446, 23746, 23761, 23770, 23787, 23790, 23808, 23835, 23
858, 23867, 56641, 56642, 23875, 23878, 23894, 23980, 23989, 23990, 56765, 24013, 24022, 56790,
56811, 24066, 56838, 24089, 24092, 24112, 24118, 24147, 24163, 24181, 56959, 24198, 24202, 2423
1, 24257, 24264, 24278, 24286, 24296, 24304, 24325, 24364, 24385, 57181, 24416, 24438, 24447, 24
453, 57226, 24463, 24473, 24475, 24494, 24514, 24522, 24529, 57306, 24542, 24570, 57344, 24580,
24613, 24615, 24622, 24628, 24634, 24653, 24666, 24674, 24682, 24687, 24693, 24694, 57511, 2474
3, 24744, 24774, 24779, 24780, 24785, 24813, 24824, 24838, 24842, 24861, 24871, 57642, 24878, 57
658, 24936, 24965, 24974, 90517, 24983, 57775, 25030, 57800, 25044, 25048, 25052, 25059, 25077,
25086, 25091, 57886, 25134, 90675, 25150, 25178, 57955, 25220, 25240, 25244, 25248, 25256, 2525
8, 25266, 25270, 25271, 25328, 25335, 25339, 58114, 58123, 25360, 58129, 58160, 58167, 25420, 25
445, 25454, 25463, 25475, 25480, 25487, 25503, 25537, 25576, 25577, 25578, 25597, 25607, 25619,
58431, 58439, 25710, 25723, 25735, 25744, 25752, 25764, 25792, 25794, 25815, 25823, 25827, 2585
4, 25901, 25903, 58673, 25907, 25916, 25928, 58705, 25948, 25979, 25998, 58797, 26032, 26038, 26
044, 26054, 26057, 26058, 26061, 26086, 26094, 26117, 26131, 26146, 26147, 26177, 26182, 26188,
26208, 26216, 26235, 26238, 59009, 26291, 26320, 26325, 26340, 26358, 26381, 26401, 26404, 5919
1, 26448, 26473, 26539, 26545, 26556, 59324, 26563, 26565, 26588, 26601, 26607, 26613, 26616, 26
634, 26640, 26656, 26659, 26671, 26679, 26683, 26695, 59482, 26719, 26729, 26736, 26744, 26749,
26759, 59556, 26797, 26809, 59586, 26838, 59607, 59609, 26863, 59637, 26893, 26897, 26902, 2693
5, 26943, 26999, 27011, 27017, 27044, 27059, 27073, 27089, 27093, 27101, 27186, 27202, 27211, 27
224, 27226, 60033, 27314, 27317, 27329, 27345, 27361, 27367, 27419, 60191, 27428, 27454, 27457,
27476, 27478, 27479, 60254, 60270, 27515, 27522, 27525, 27532, 27541, 27543, 27551, 27566, 2756
8, 27590, 27606, 27612, 27613, 27618, 27630, 60408, 27649, 27661, 27676, 60461, 27698, 60476, 27
726, 27728, 27734, 27736, 27766, 27773, 27790, 27808, 27811, 27818, 27819, 27842, 27848, 27851,
27856, 27872, 27897, 27899, 60699, 27938, 27946, 27981, 27998, 28006, 60781, 28016, 28047, 2804
9, 28085, 28121, 28147, 28165, 60944, 28176, 28221, 28234, 28254, 28275, 61045, 28287, 28319, 28
382, 28465, 28487, 61258, 28496, 28508, 28558, 28576, 28577, 28596, 28633, 28666, 28671, 28709,
28717, 28727, 61495, 61509, 28744, 28755, 28759, 61572, 28804, 28846, 28849, 94395, 28866, 2887
7, 28924, 28929, 94512, 29007, 29015, 29018, 29020, 61818, 29061, 61850, 61860, 29105, 29114, 29
133, 29138, 29143, 29154, 29210, 29213, 29219, 29223, 29231, 29232, 29243, 29268, 62043, 29279,

29287, 29294, 29321, 29360, 62131, 62135, 29402, 29404, 29414, 29426, 29440, 29460, 29490, 2949
9, 62274, 29516, 29522, 29539, 29568, 29586, 29599, 29601, 29608, 62382, 29617, 62429, 62430, 29
693, 62491, 29737, 29743, 29747, 29807, 29809, 29825, 62625, 29896, 29903, 29911, 29921, 29942,
29976, 29986, 29997, 30003, 30010, 30035, 30050, 30068, 30076, 30097, 30125, 30141, 62916, 3016
6, 30178, 30196, 30211, 30225, 30226, 30228, 30231, 30253, 30274, 30305, 30309, 30329, 30350, 30
373, 30375, 30376, 30377, 30378, 30385, 30390, 30424, 63218, 30478, 30522, 63318, 30556, 30576,
30589, 30609, 63393, 63425, 30662, 30672, 30688, 30689, 63465, 30697, 30733, 30734, 30746, 3076
2, 30764, 30768, 30781, 30784, 30785, 30795, 30804, 30807, 30809, 30815, 30873, 63659, 96436, 30
910, 30922, 30927, 30933, 30935, 30955, 30970, 30975, 30996, 96575, 31056, 31085, 31101, 31102,
63884, 63894, 63911, 31148, 31178, 31194, 31196, 31251, 31318, 31331, 31371, 31474, 31498, 3156
0, 31607, 31632, 64417, 31725, 31736, 31739, 31749, 31758, 31790, 31791, 31795, 31816, 31819, 31
845, 31846, 31848, 31859, 31860, 31883, 31927, 31990, 32000, 32005, 32013, 32019, 32029, 32031,
32035, 32066, 32069, 32080, 32084, 32096, 32117, 32135, 32195, 32203, 32204, 32211, 32257, 3225
9, 32261, 32264, 32269, 32275, 32315, 32351, 65123, 32363, 32388, 32425, 32471, 32477, 32489, 32
525, 32526, 32539, 32564, 65335, 32568, 32572, 65347, 65355, 32603, 32653, 32681, 32688, 32737,
32739]

2. Area is categorical (CATEGORICAL): ['Urban' 'Suburban' 'Rural'].

3. Children is numerical (CONTINUOUS) - type: int64.
  Unique: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

4. Age is numerical (CONTINUOUS) - type: int64.
  Unique: [18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 3
9, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 6
3, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 8
7, 88, 89]

5. Income is numerical (CONTINUOUS) - type: float64.
  Min: 348.670 Max: 258900.700 Std: 28199.917

6. Marital is categorical (CATEGORICAL): ['Widowed' 'Married' 'Separated' 'Never Married' 'Divor
ced'].

7. Gender is categorical (CATEGORICAL): ['Male' 'Female' 'Nonbinary'].

8. Outage_sec_perweek is numerical (CONTINUOUS) - type: float64.
  Min: 0.100 Max: 21.207 Std: 2.976

9. Email is numerical (CONTINUOUS) - type: int64.
  Unique: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23]

10. Contacts is numerical (CONTINUOUS) - type: int64.
  Unique: [0, 1, 2, 3, 4, 5, 6, 7]

11. Yearly_equip_failure is numerical (CONTINUOUS) - type: int64.
  Unique: [0, 1, 2, 3, 4, 6]

12. Techie is categorical (CATEGORICAL): ['No' 'Yes'].

13. Contract is categorical (CATEGORICAL): ['One year' 'Month-to-month' 'Two Year'].

14. Port_modem is categorical (CATEGORICAL): ['Yes' 'No'].

15. Tablet is categorical (CATEGORICAL): ['Yes' 'No'].

16. InternetService is categorical (CATEGORICAL): ['Fiber Optic' 'DSL' 'None'].

17. Phone is categorical (CATEGORICAL): ['Yes' 'No'].

18. Multiple is categorical (CATEGORICAL): ['No' 'Yes'].

```
19. OnlineSecurity is categorical (CATEGORICAL): ['Yes' 'No'].

20. OnlineBackup is categorical (CATEGORICAL): ['Yes' 'No'].

21. DeviceProtection is categorical (CATEGORICAL): ['No' 'Yes'].

22. TechSupport is categorical (CATEGORICAL): ['No' 'Yes'].

23. StreamingTV is categorical (CATEGORICAL): ['No' 'Yes'].

24. StreamingMovies is categorical (CATEGORICAL): ['Yes' 'No'].

25. PaperlessBilling is categorical (CATEGORICAL): ['Yes' 'No'].

26. PaymentMethod is categorical (CATEGORICAL): ['Credit Card (automatic)' 'Bank Transfer(automa
tic)' 'Mailed Check'
 'Electronic Check'].

27. Tenure is numerical (CONTINUOUS) - type: float64.
  Min: 1.000  Max: 71.999  Std: 26.443

28. MonthlyCharge is numerical (CONTINUOUS) - type: float64.
  Min: 79.979  Max: 290.160  Std: 42.943

29. Bandwidth_GB_Year is numerical (CONTINUOUS) - type: float64.
  Min: 155.507  Max: 7158.982  Std: 2185.295
```

In [12]:
```
# explore missing data
missing = df_clean[df_clean.columns[df_clean.isna().any()]].columns
df_missing = df_clean[missing]
print(df_missing.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Empty DataFrameNone
```

In [13]:
```
# fill null values for missing numerical data
df_clean['Children'].fillna(0, inplace=True)
df_clean['Age'].fillna(df_clean['Age'].mean(), inplace=True)
df_clean['Income'].fillna(df_clean['Income'].mean(), inplace=True)
df_clean['Bandwidth_GB_Year'].fillna(df_clean['Bandwidth_GB_Year'].mean(), inplace=True)
df_clean['Tenure'].fillna(df_clean['Tenure'].mean(), inplace=True)
```

In [14]:
```
# fill null values for missing categorical data
df_clean['Techie'].fillna('No', inplace=True)
df_clean['Phone'].fillna('No', inplace=True)
df_clean['TechSupport'].fillna('No', inplace=True)
```

In [15]:
```
# explore missing data
missing = df_clean[df_clean.columns[df_clean.isna().any()]].columns
df_missing = df_clean[missing]
print(df_missing.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Empty DataFrameNone
```

In [16]:
```
# look for duplicate data - looking for zero rows
df_clean[df_clean.duplicated()]
```

Out[16]:

Population	Area	Children	Age	Income	Marital	Gender	Outage_sec_perweek	Email	Contacts	...	OnlineBa

0 rows × 29 columns

In [17]:
```
# check if any cols are duplicated - Looking for False
df_clean.columns.duplicated().any()
```

Out[17]:
```
False
```

In [18]:
```
# check if any rows are duplicated - looing for False
df_clean.duplicated().any()
```

Out[18]:
```
False
```

In [ ]:
```

```

C5. Provide a copy of the cleaned dataset.

TABLE 2.CLEAN

Cleaned data.

In [19]:
```
save_course_table_csv(data=df_clean, title='CLEAN', title_only=True)
```

	0	1	2	3
Population	38	10446	3735	13863
Area	Urban	Urban	Urban	Suburban
Children	0	1	4	1
Age	68	27	50	48
Income	28561.99	21704.77	9609.57	18925.23
Marital	Widowed	Married	Widowed	Married
Gender	Male	Female	Female	Male
Outage_sec_perweek	7.978323	11.69908	10.7528	14.91354
Email	10	12	9	15
Contacts	0	0	0	2
Yearly_equip_failure	1	1	1	0
Techie	No	Yes	Yes	Yes
Contract	One year	Month-to-month	Two Year	Two Year
Port_modem	Yes	No	Yes	No
Tablet	Yes	Yes	No	No
InternetService	Fiber Optic	Fiber Optic	DSL	DSL
Phone	Yes	Yes	Yes	Yes
Multiple	No	Yes	Yes	No
OnlineSecurity	Yes	Yes	No	Yes
OnlineBackup	Yes	No	No	No
DeviceProtection	No	No	No	No
TechSupport	No	No	No	No
StreamingTV	No	Yes	No	Yes
StreamingMovies	Yes	Yes	Yes	No
PaperlessBilling	Yes	Yes	Yes	Yes
PaymentMethod	Credit Card (automatic)	Bank Transfer(automatic)	Credit Card (automatic)	Mailed Check
Tenure	6.795513	1.156681	15.754144	17.087227
MonthlyCharge	172.455519	242.632554	159.947583	119.95684
Bandwidth_GB_Year	904.53611	800.982766	2054.706961	2164.579412

```
shape: (10000, 29)
Table saved to: TABLES/CLEAN.CSV
```

In [ ]:

Part III(b): Data Exploration

C. Explore data for multiple regression by doing the following:

C6. Explore Categorical Data.

In [20]:
```
# print out input variables
for c in df_clean.loc[:, df_clean.columns != target]:
    if df_clean.dtypes[c] == "object":
        print('\n{} is categorical: {}.'.format(c,df_clean[c].unique()))
    else:
        print('\n{} is numerical:'.format(c ))
        print('\trange = {} - {}'.format(df_clean[c].min(),df_clean[c].max()))
        print('\tmean = {:.2f} +/- {:.2f}'.format(df_clean[c].mean(), df_clean[c].std()))
```

```
Population is numerical:
        range = 0 - 111850
        mean = 9756.56 +/- 14432.70

Area is categorical: ['Urban' 'Suburban' 'Rural'].

Children is numerical:
        range = 0 - 10
        mean = 2.09 +/- 2.15

Age is numerical:
        range = 18 - 89
        mean = 53.08 +/- 20.70

Income is numerical:
        range = 348.67 - 258900.7
        mean = 39806.93 +/- 28199.92

Marital is categorical: ['Widowed' 'Married' 'Separated' 'Never Married' 'Divorced'].

Gender is categorical: ['Male' 'Female' 'Nonbinary'].

Outage_sec_perweek is numerical:
        range = 0.09974694 - 21.20723
        mean = 10.00 +/- 2.98

Email is numerical:
        range = 1 - 23
        mean = 12.02 +/- 3.03

Contacts is numerical:
        range = 0 - 7
        mean = 0.99 +/- 0.99

Yearly_equip_failure is numerical:
        range = 0 - 6
        mean = 0.40 +/- 0.64

Techie is categorical: ['No' 'Yes'].

Contract is categorical: ['One year' 'Month-to-month' 'Two Year'].

Port_modem is categorical: ['Yes' 'No'].

Tablet is categorical: ['Yes' 'No'].

InternetService is categorical: ['Fiber Optic' 'DSL' 'None'].

Phone is categorical: ['Yes' 'No'].

Multiple is categorical: ['No' 'Yes'].

OnlineSecurity is categorical: ['Yes' 'No'].

OnlineBackup is categorical: ['Yes' 'No'].

DeviceProtection is categorical: ['No' 'Yes'].

TechSupport is categorical: ['No' 'Yes'].

StreamingTV is categorical: ['No' 'Yes'].
```

```
StreamingMovies is categorical: ['Yes' 'No'].

PaperlessBilling is categorical: ['Yes' 'No'].

PaymentMethod is categorical: ['Credit Card (automatic)' 'Bank Transfer(automatic)' 'Mailed Chec
k'
 'Electronic Check'].

Tenure is numerical:
        range = 1.00025934 - 71.99928
        mean = 34.53 +/- 26.44

Bandwidth_GB_Year is numerical:
        range = 155.5067148 - 7158.98153
        mean = 3392.34 +/- 2185.29
```

In [21]:
```
# define categorical data
categorical_features = df_clean.select_dtypes(include="object").columns
print(categorical_features)
```

```
Index(['Area', 'Marital', 'Gender', 'Techie', 'Contract', 'Port_modem',
       'Tablet', 'InternetService', 'Phone', 'Multiple', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'PaperlessBilling', 'PaymentMethod'],
      dtype='object')
```

In [22]:
```
# define bool features
bool_features = df_clean.select_dtypes(include="bool").columns
print(bool_features)
```

```
Index([], dtype='object')
```

In [23]:
```
# convert array to list then append lists
cat_or_bool_features = categorical_features.tolist() + bool_features.tolist()
print(cat_or_bool_features)
```

```
['Area', 'Marital', 'Gender', 'Techie', 'Contract', 'Port_modem', 'Tablet', 'InternetService',
'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'Strea
mingTV', 'StreamingMovies', 'PaperlessBilling', 'PaymentMethod']
```

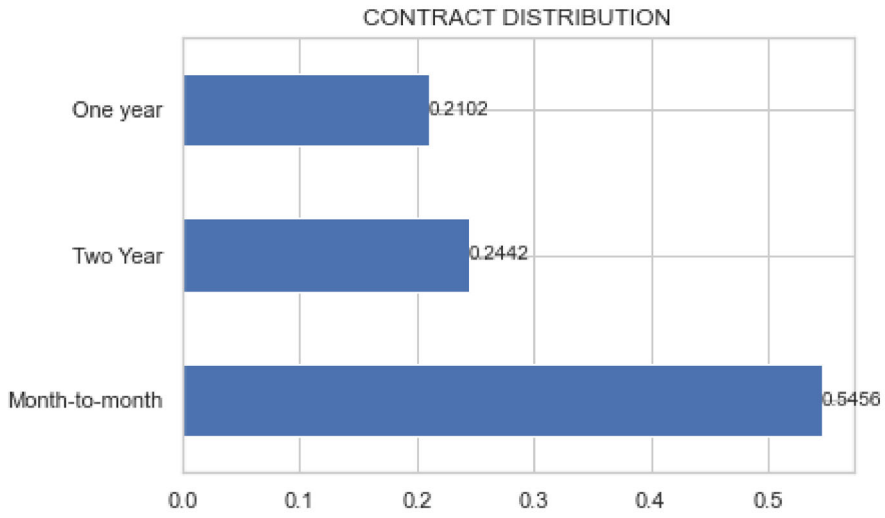
In [24]:
```
# describe cat and bool features by datatype
describe_dataframe_type(df_clean[cat_or_bool_features])
```

```
1. Area is categorical (CATEGORICAL): ['Urban' 'Suburban' 'Rural'].

2. Marital is categorical (CATEGORICAL): ['Widowed' 'Married' 'Separated' 'Never Married' 'Divor
ced'].

3. Gender is categorical (CATEGORICAL): ['Male' 'Female' 'Nonbinary'].

4. Techie is categorical (CATEGORICAL): ['No' 'Yes'].

5. Contract is categorical (CATEGORICAL): ['One year' 'Month-to-month' 'Two Year'].

6. Port_modem is categorical (CATEGORICAL): ['Yes' 'No'].

7. Tablet is categorical (CATEGORICAL): ['Yes' 'No'].

8. InternetService is categorical (CATEGORICAL): ['Fiber Optic' 'DSL' 'None'].

9. Phone is categorical (CATEGORICAL): ['Yes' 'No'].

10. Multiple is categorical (CATEGORICAL): ['No' 'Yes'].

11. OnlineSecurity is categorical (CATEGORICAL): ['Yes' 'No'].

12. OnlineBackup is categorical (CATEGORICAL): ['Yes' 'No'].

13. DeviceProtection is categorical (CATEGORICAL): ['No' 'Yes'].

14. TechSupport is categorical (CATEGORICAL): ['No' 'Yes'].

15. StreamingTV is categorical (CATEGORICAL): ['No' 'Yes'].

16. StreamingMovies is categorical (CATEGORICAL): ['Yes' 'No'].

17. PaperlessBilling is categorical (CATEGORICAL): ['Yes' 'No'].

18. PaymentMethod is categorical (CATEGORICAL): ['Credit Card (automatic)' 'Bank Transfer(automa
tic)' 'Mailed Check'
 'Electronic Check'].
```

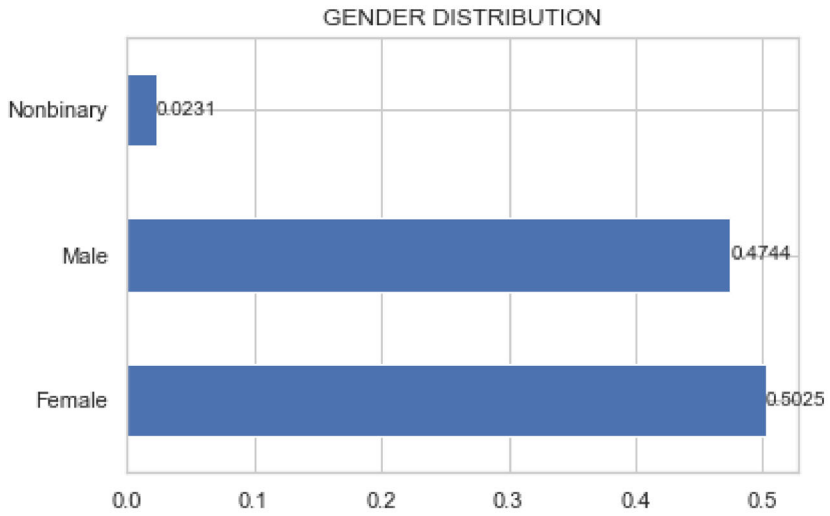
C9. Explore Categorical Data.

In [25]:
```
# create distribution plots for a selected list of categorical or boolean features
cat_or_bool_features = ['Contract', 'Gender']
for idx,f in enumerate(cat_or_bool_features):
    create_distribution_plot_from_feature_fig(
        data=df_clean, cat_or_bool_feature=f,
        sect='3B_C9', caption=str(idx+1))
```

```
Figure saved to: FIGURES/__FIG_3B_C9_1_CONTRACT_DISTRIBUTION.PNG
Figure saved to: FIGURES/__FIG_3B_C9_2_GENDER_DISTRIBUTION.PNG
```



FIGURES/__FIG_3B_C9_1_CONTRACT_DISTRIBUTION.PNG

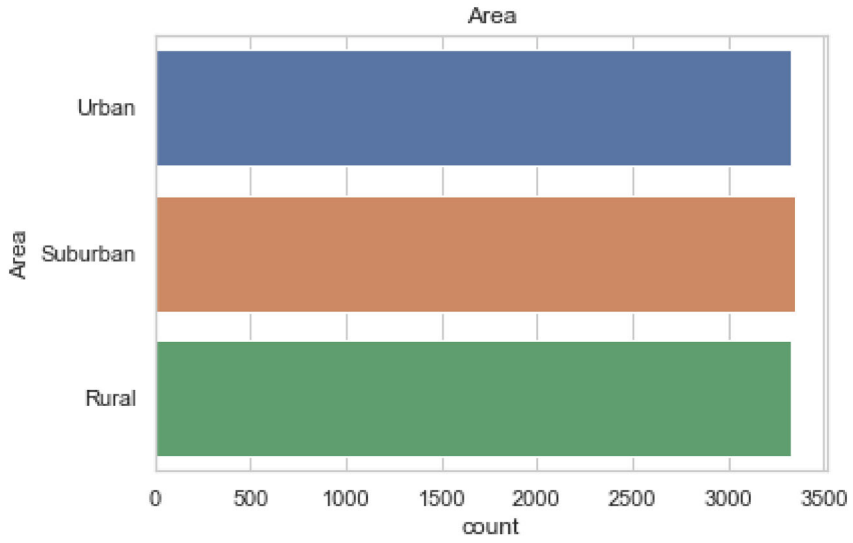


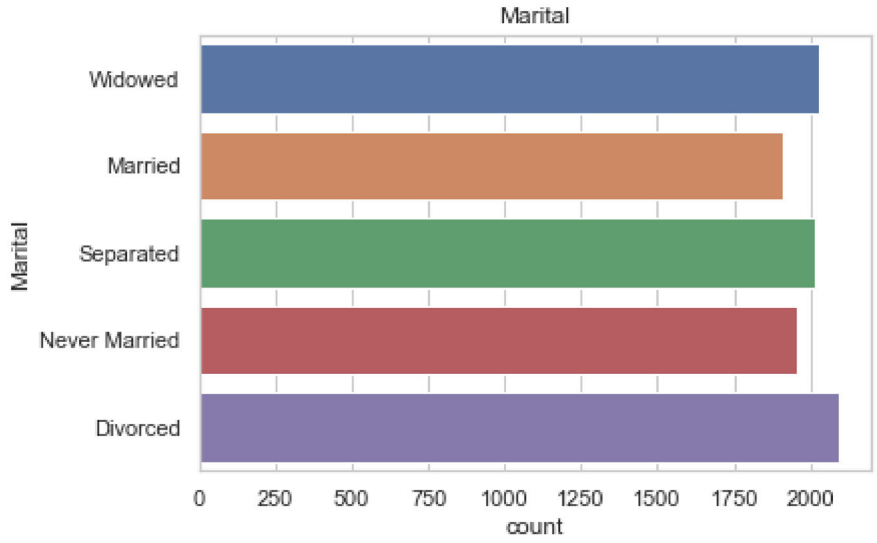
FIGURES/__FIG_3B_C9_2_GENDER_DISTRIBUTION.PNG

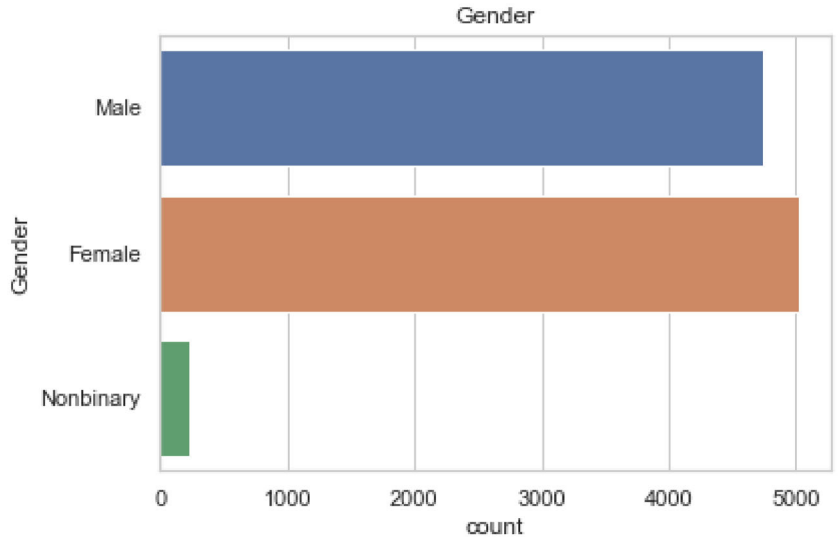
C11. Explore Categorical Data

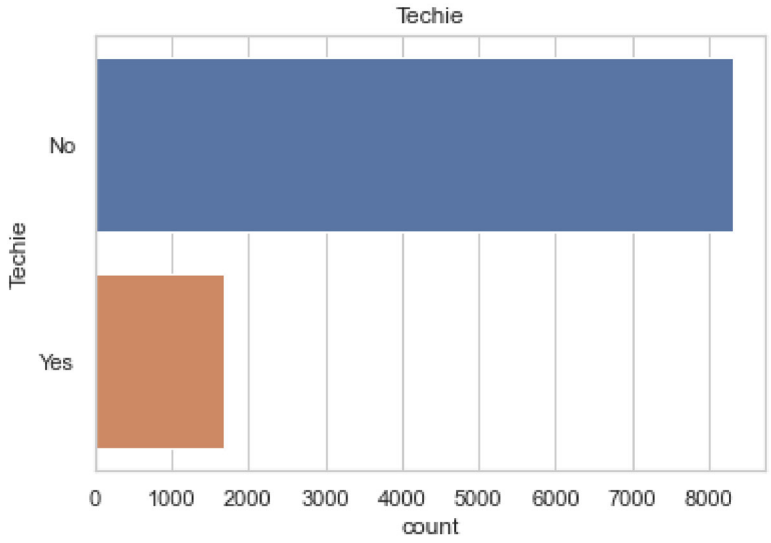
Prior to converting the categorical data for use in the model, as part of exploratory data analysis, I will visualize the original categorical data using a countplot. In a moment, the categorical data will be converted to dummy data and I will lose the original data.

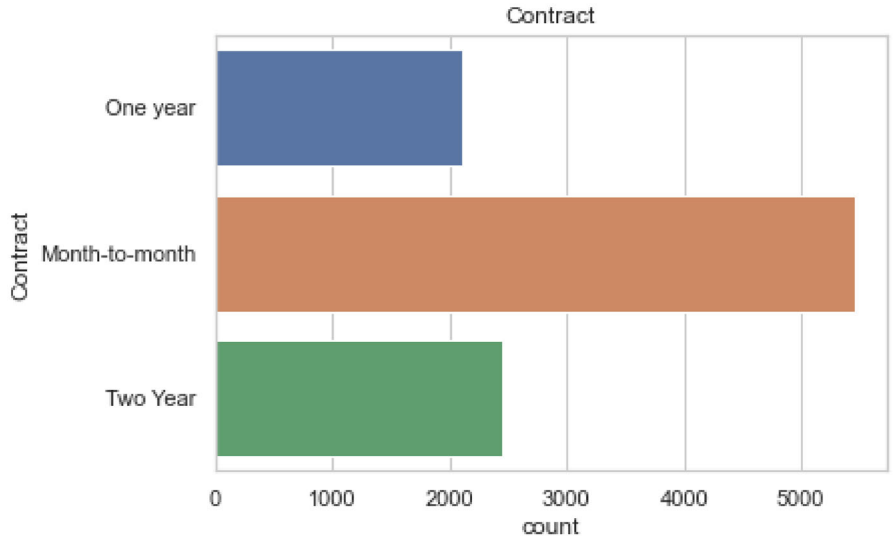
In [26]:
```
# plot categorical data - before it gets converted
for i, c in enumerate(categorical_features):
    ax = sns.countplot(y=c, data=df_clean)
    plt.title(c)
    plt.savefig('figures/' + str(c) + '.png')
    plt.show()
```

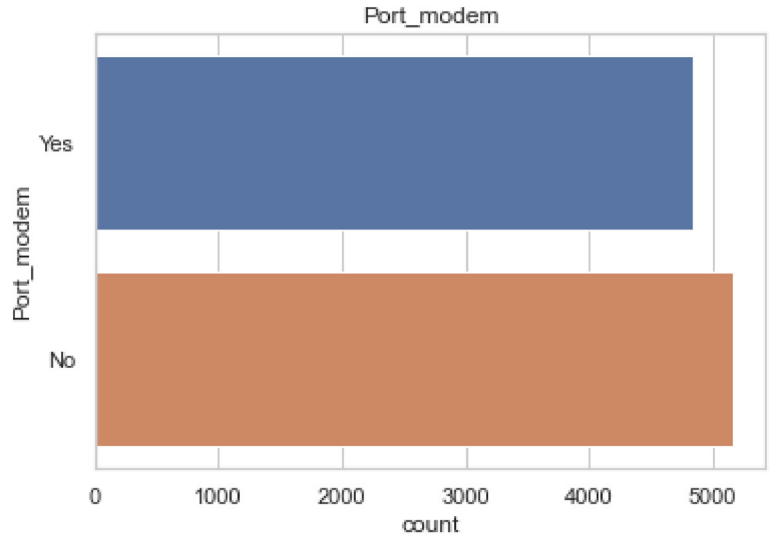

Area
Urban
Suburban
Rural
Area
0
500
1000
1500
2000
2500
3000
3500
count

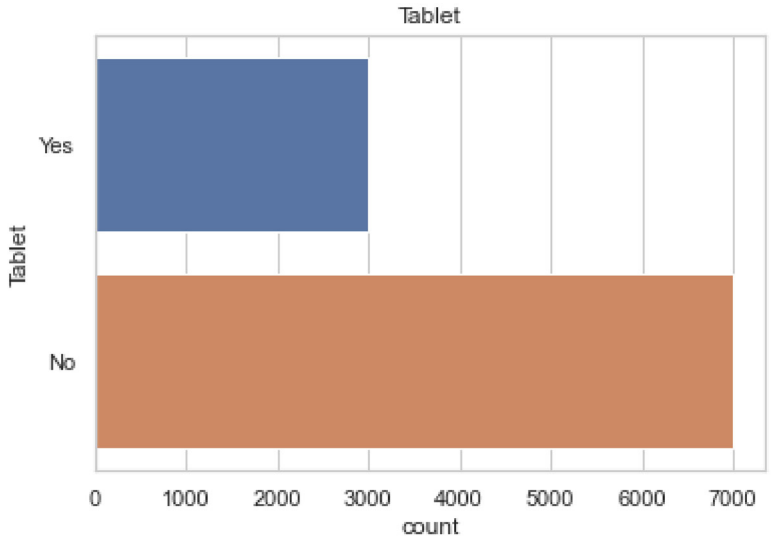

Marital
Widowed
Married
Separated
Never Married
Divorced
Marital
0
250
500
750
1000
1250
1500
1750
2000
count

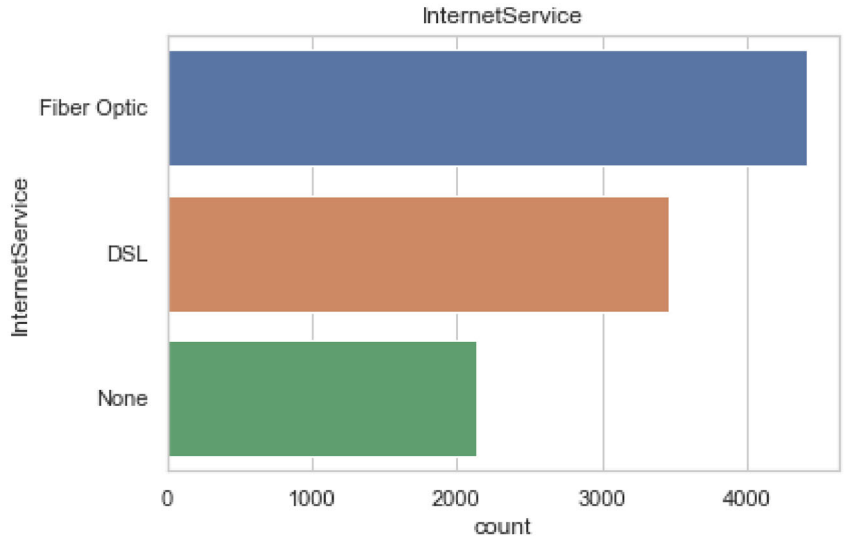

Gender
Male
Female
Nonbinary
Gender
0
1000
2000
3000
4000
5000
count

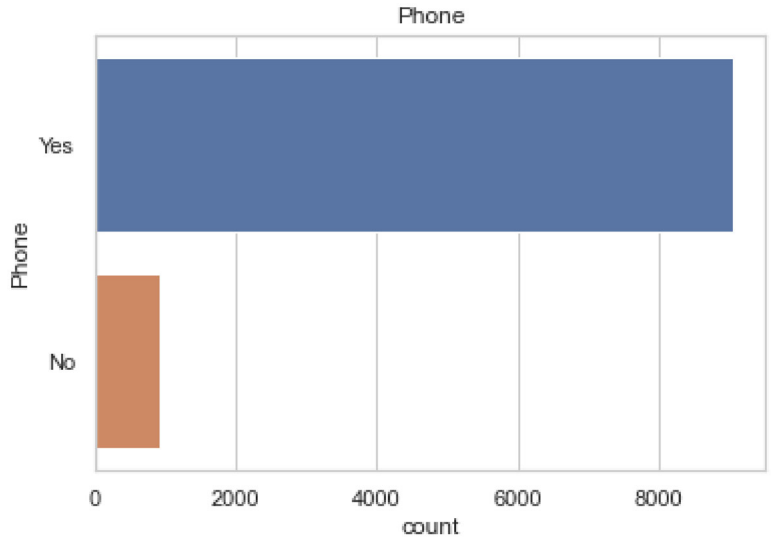

Techie
No
Yes
Techie
0
1000
2000
3000
4000
5000
6000
7000
8000
count

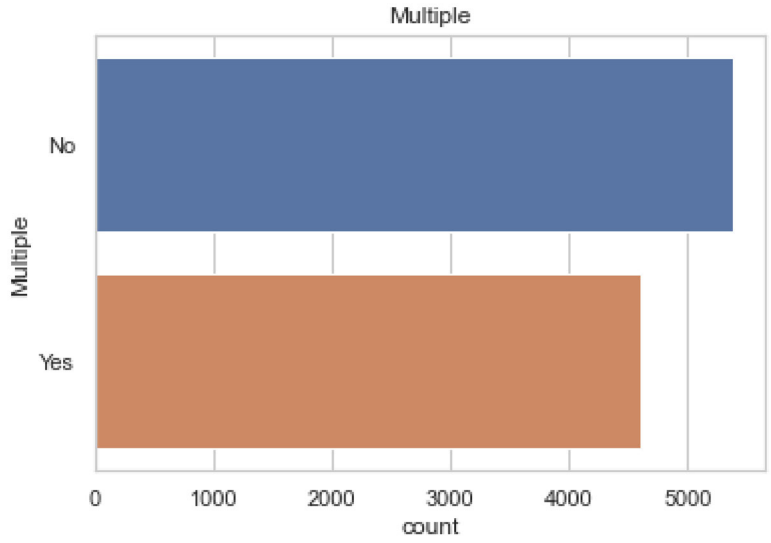

Contract
One year
Month-to-month
Two Year
Contract
0
1000
2000
3000
4000
5000
count


Port_modem
Yes
No
Port_modem
0
1000
2000
3000
4000
5000
count

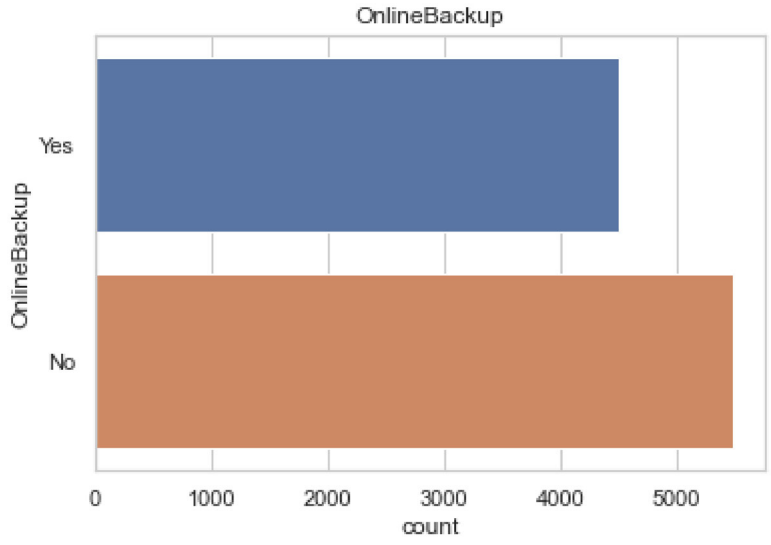

Tablet
Yes
No
Tablet
0
1000
2000
3000
4000
5000
6000
7000
count

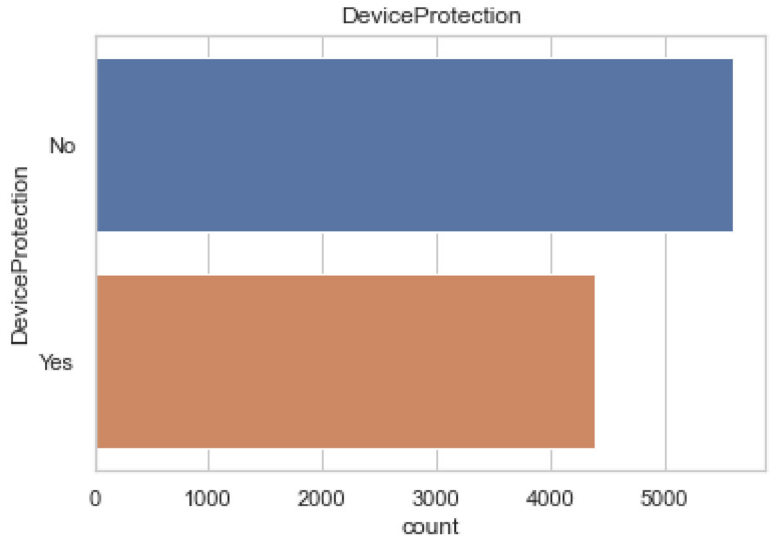

InternetService
Fiber Optic
DSL
None
InternetService
0
1000
2000
3000
4000
count

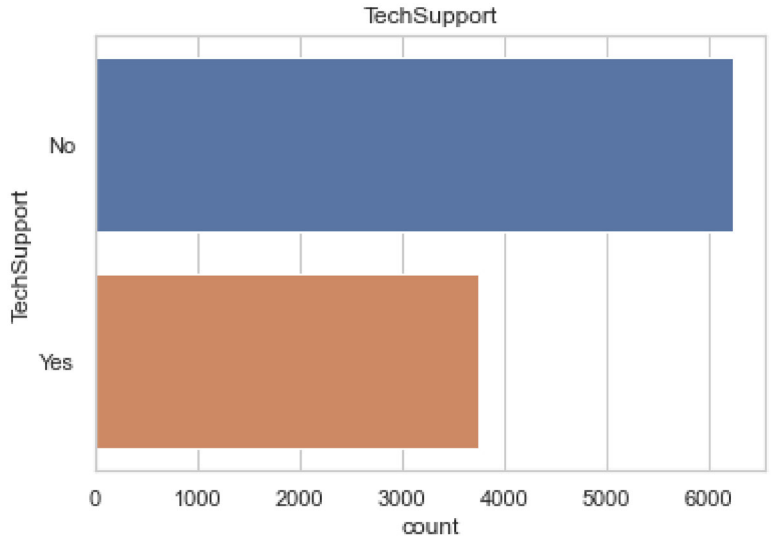

Phone
Yes
No
Phone
0
2000
4000
6000
8000
count

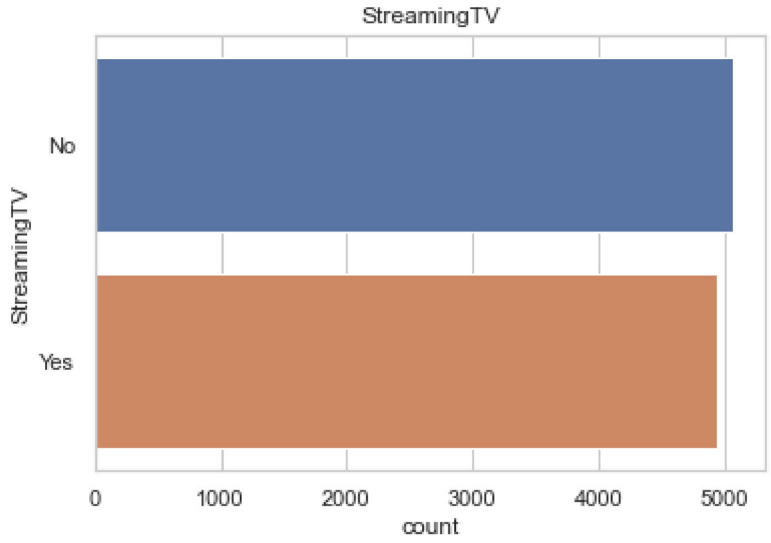

Multiple
No
Yes
Multiple
0
1000
2000
3000
4000
5000
count

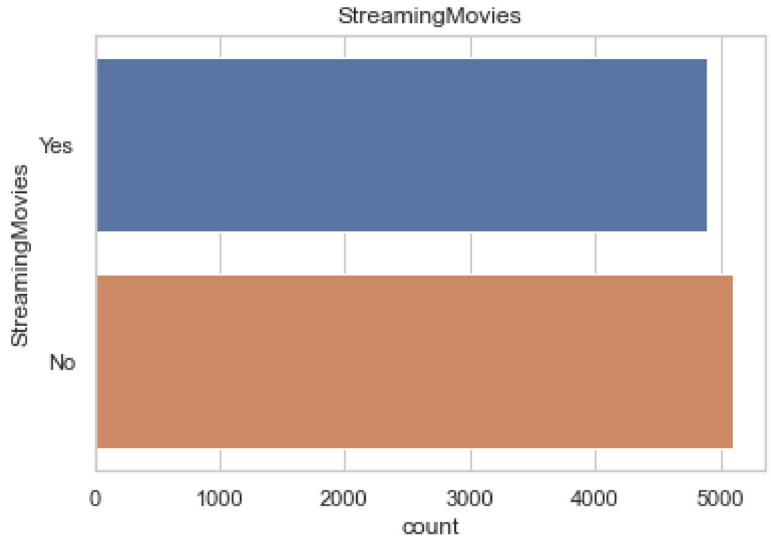

OnlineSecurity
Yes
No
OnlineSecurity
0
1000
2000
3000
4000
5000
6000
count

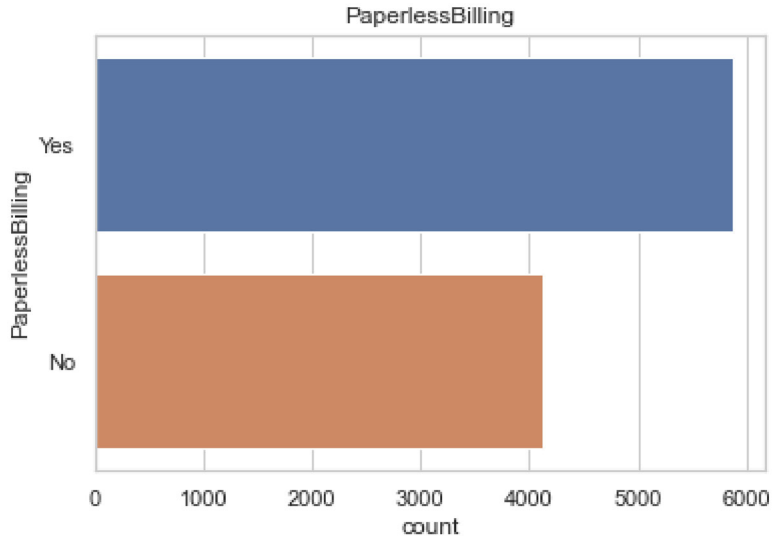

OnlineBackup
Yes
No
OnlineBackup
0
1000
2000
3000
4000
5000
count

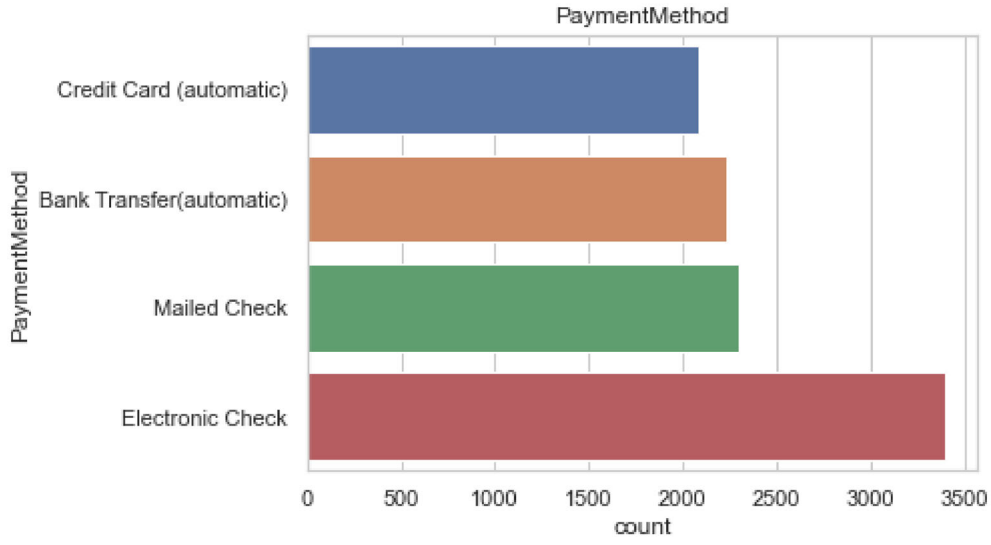

DeviceProtection
No
Yes
DeviceProtection
0
1000
2000
3000
4000
5000
count


TechSupport
No
Yes
TechSupport
0
1000
2000
3000
4000
5000
6000
count


StreamingTV
No
Yes
StreamingTV
0
1000
2000
3000
4000
5000
count







C10. Transform Categorical Data.

The regression model requires all of the independent variables to be numeric. Because there are many categorical data, each will have to be converted into numeric data. The data is converted into dummy numeric data using the pandas .get_dummies() method. After the conversion, the original data is removed.

The method uses the option 'drop_first=True'. Most of the categorical data has two or more unique values. When using this option, the .get_dummies() method will remove the first unique value, which is good, because of the multi-collinear nature of this operation. It can be a problem, however, if the data that is removed is data that is necessary. For the purpose of this analysis, I am using the 'drop_first' option, but future analysis may decide to use the other data. I am creating a variable called 'contract' with a snapshot of the contract data before the conversion in order to present an example of this potential problem.

In [27]:
```
# convert categorical data
for c in categorical_features:
    if c in df_clean.columns:
        df_clean = pd.get_dummies(df_clean, columns=categorical_features, drop_first=True)
        print(df_clean.select_dtypes(include="uint8").columns)
```

```
Index(['Area_Suburban', 'Area_Urban', 'Marital_Married',
       'Marital_Never Married', 'Marital_Separated', 'Marital_Widowed',
       'Gender_Male', 'Gender_Nonbinary', 'Techie_Yes', 'Contract_One year',
       'Contract_Two Year', 'Port_modem_Yes', 'Tablet_Yes',
       'InternetService_Fiber Optic', 'InternetService_None', 'Phone_Yes',
       'Multiple_Yes', 'OnlineSecurity_Yes', 'OnlineBackup_Yes',
       'DeviceProtection_Yes', 'TechSupport_Yes', 'StreamingTV_Yes',
       'StreamingMovies_Yes', 'PaperlessBilling_Yes',
       'PaymentMethod_Credit Card (automatic)',
       'PaymentMethod_Electronic Check', 'PaymentMethod_Mailed Check'],
      dtype='object')
```

In [28]:
```
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 38 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Population                              10000 non-null  int64
 1   Children                                10000 non-null  int64
 2   Age                                     10000 non-null  int64
 3   Income                                  10000 non-null  float64
 4   Outage_sec_perweek                      10000 non-null  float64
 5   Email                                   10000 non-null  int64
 6   Contacts                                10000 non-null  int64
 7   Yearly_equip_failure                    10000 non-null  int64
 8   Tenure                                  10000 non-null  float64
 9   MonthlyCharge                           10000 non-null  float64
 10  Bandwidth_GB_Year                       10000 non-null  float64
 11  Area_Suburban                           10000 non-null  uint8
 12  Area_Urban                              10000 non-null  uint8
 13  Marital_Married                         10000 non-null  uint8
 14  Marital_Never Married                   10000 non-null  uint8
 15  Marital_Separated                       10000 non-null  uint8
 16  Marital_Widowed                         10000 non-null  uint8
 17  Gender_Male                             10000 non-null  uint8
 18  Gender_Nonbinary                        10000 non-null  uint8
 19  Techie_Yes                              10000 non-null  uint8
 20  Contract_One year                       10000 non-null  uint8
 21  Contract_Two Year                       10000 non-null  uint8
 22  Port_modem_Yes                          10000 non-null  uint8
 23  Tablet_Yes                              10000 non-null  uint8
 24  InternetService_Fiber Optic             10000 non-null  uint8
 25  InternetService_None                    10000 non-null  uint8
 26  Phone_Yes                               10000 non-null  uint8
 27  Multiple_Yes                            10000 non-null  uint8
 28  OnlineSecurity_Yes                      10000 non-null  uint8
 29  OnlineBackup_Yes                        10000 non-null  uint8
 30  DeviceProtection_Yes                    10000 non-null  uint8
 31  TechSupport_Yes                         10000 non-null  uint8
 32  StreamingTV_Yes                         10000 non-null  uint8
 33  StreamingMovies_Yes                     10000 non-null  uint8
 34  PaperlessBilling_Yes                    10000 non-null  uint8
 35  PaymentMethod_Credit Card (automatic)   10000 non-null  uint8
 36  PaymentMethod_Electronic Check          10000 non-null  uint8
 37  PaymentMethod_Mailed Check              10000 non-null  uint8
dtypes: float64(5), int64(6), uint8(27)
memory usage: 1.1 MB
```

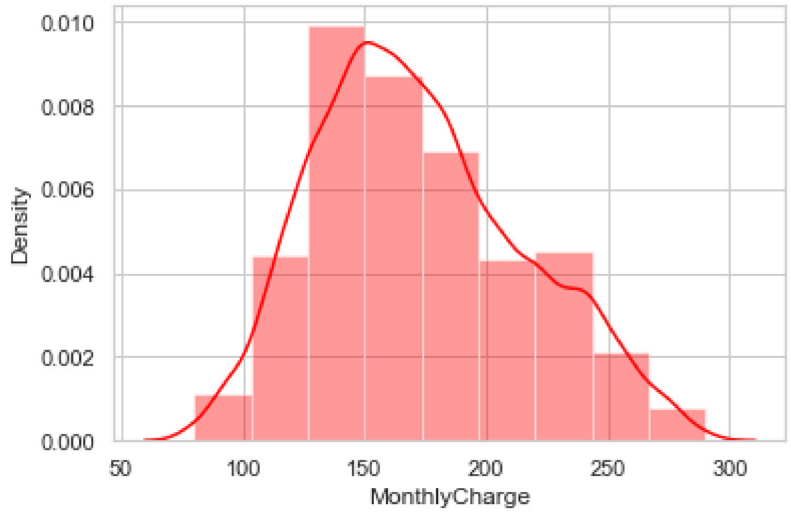
C7. Explore Numerical Data.

In [29]:
```
# variable for numeric data
numerical_features = df_clean.select_dtypes(include="number").columns
print(numerical_features)
```

```
Index(['Population', 'Children', 'Age', 'Income', 'Outage_sec_perweek',
       'Email', 'Contacts', 'Yearly_equip_failure', 'Tenure', 'MonthlyCharge',
       'Bandwidth_GB_Year', 'Area_Suburban', 'Area_Urban', 'Marital_Married',
       'Marital_Never Married', 'Marital_Separated', 'Marital_Widowed',
       'Gender_Male', 'Gender_Nonbinary', 'Techie_Yes', 'Contract_One year',
       'Contract_Two Year', 'Port_modem_Yes', 'Tablet_Yes',
       'InternetService_Fiber Optic', 'InternetService_None', 'Phone_Yes',
       'Multiple_Yes', 'OnlineSecurity_Yes', 'OnlineBackup_Yes',
       'DeviceProtection_Yes', 'TechSupport_Yes', 'StreamingTV_Yes',
       'StreamingMovies_Yes', 'PaperlessBilling_Yes',
       'PaymentMethod_Credit Card (automatic)',
       'PaymentMethod_Electronic Check', 'PaymentMethod_Mailed Check'],
      dtype='object')
```

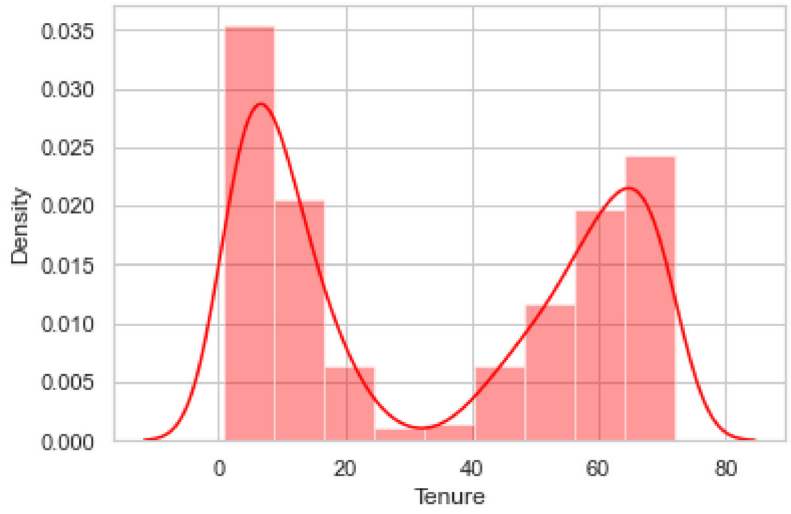
In [30]:
```
sns.distplot(df_clean['MonthlyCharge'], kde=True, color='red', bins=9)
```

Out[30]:
```
<AxesSubplot:xlabel='MonthlyCharge', ylabel='Density'>
```



In [31]:
```
sns.distplot(df_clean['Tenure'], kde=True, color='red', bins=9)
```

Out[31]:
```
<AxesSubplot:xlabel='Tenure', ylabel='Density'>
```

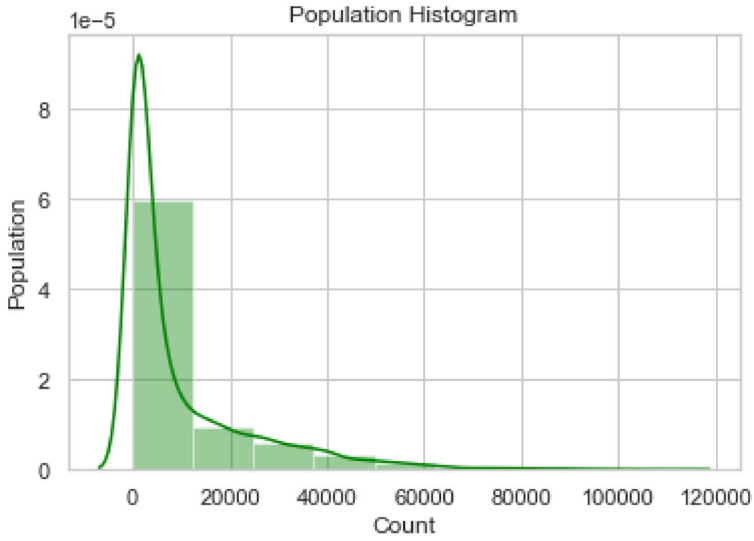


C8. Explore Numerical Data

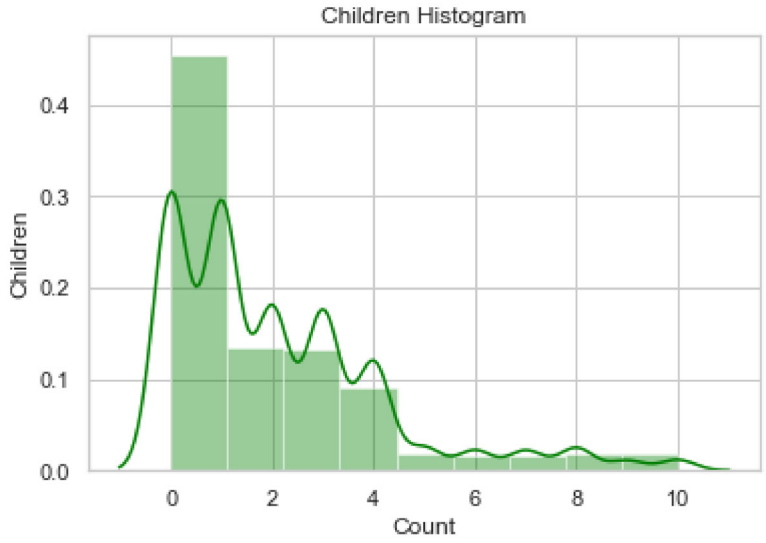
In [32]:
```
# simple histogram for numerical features
```

```
for idx,f in enumerate(numerical_features):
    #df_clean[[f]].hist()
    create_simple_histogram_numerical_feature_fig(
        data=df_clean, numerical_feature=f, bins=9,
        sect='3B_C8', caption=str(idx+1))
```

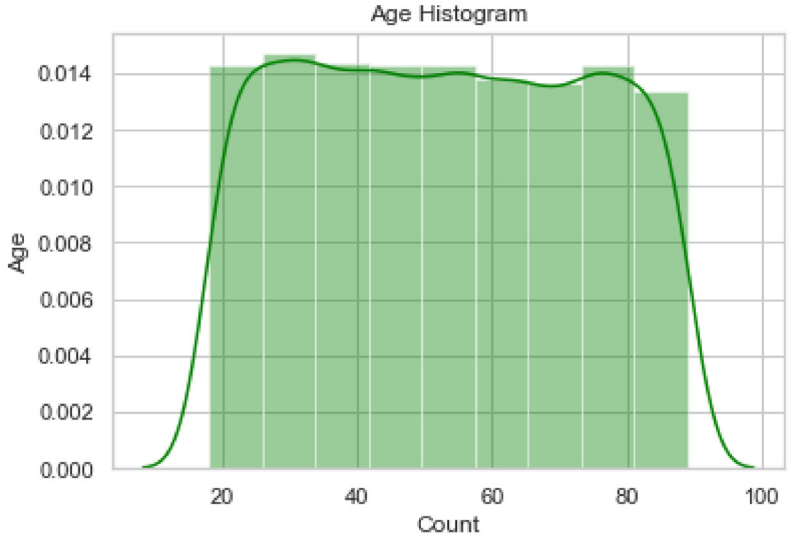
```
Figure saved to: FIGURES/__FIG_3B_C8_1_POPULATION_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_2_CHILDREN_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_3_AGE_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_4_INCOME_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_5_OUTAGE_SEC_PERWEEK_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_6_EMAIL_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_7_CONTACTS_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_8_YEARLY_EQUIP_FAILURE_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_9_TENURE_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_10_MONTHLYCHARGE_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_11_BANDWIDTH_GB_YEAR_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_12_AREA_SUBURBAN_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_13_AREA_URBAN_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_14_MARITAL_MARRIED_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_15_MARITAL_NEVER_MARRIED_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_16_MARITAL_SEPARATED_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_17_MARITAL_WIDOWED_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_18_GENDER_MALE_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_19_GENDER_NONBINARY_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_20_TECHIE_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_21_CONTRACT_ONE_YEAR_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_22_CONTRACT_TWO_YEAR_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_23_PORT_MODEM_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_24_TABLET_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_25_INTERNETSERVICE_FIBER_OPTIC_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_26_INTERNETSERVICE_NONE_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_27_PHONE_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_28_MULTIPLE_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_29_ONLINESECURITY_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_30_ONLINEBACKUP_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_31_DEVICEPROTECTION_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_32_TECHSUPPORT_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_33_STREAMINGTV_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_34_STREAMINGMOVIES_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_35_PAPERLESSBILLING_YES_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_36_PAYMENTMETHOD_CREDIT_CARD_(AUTOMATIC)_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_37_PAYMENTMETHOD_ELECTRONIC_CHECK_HISTOGRAM.PNG
Figure saved to: FIGURES/__FIG_3B_C8_38_PAYMENTMETHOD_MAILED_CHECK_HISTOGRAM.PNG
```



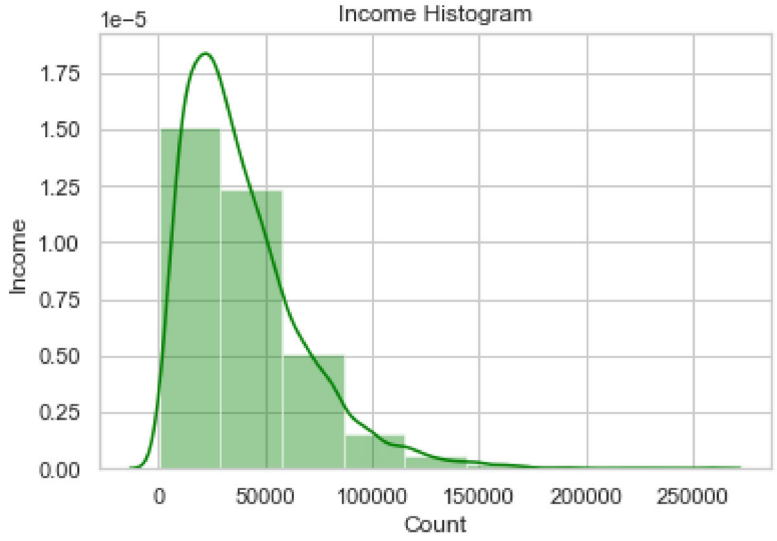
FIGURES/__FIG_3B_C8_1_POPULATION_HISTOGRAM.PNG



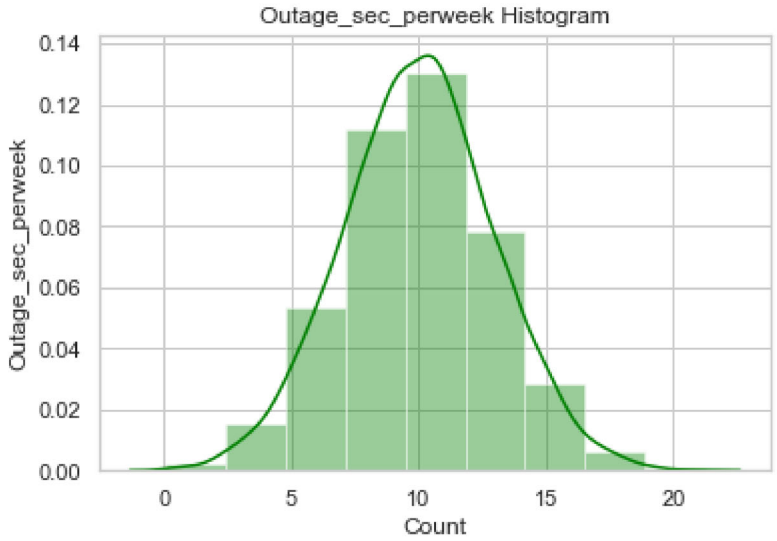
FIGURES/__FIG_3B_C8_2_CHILDREN_HISTOGRAM.PNG



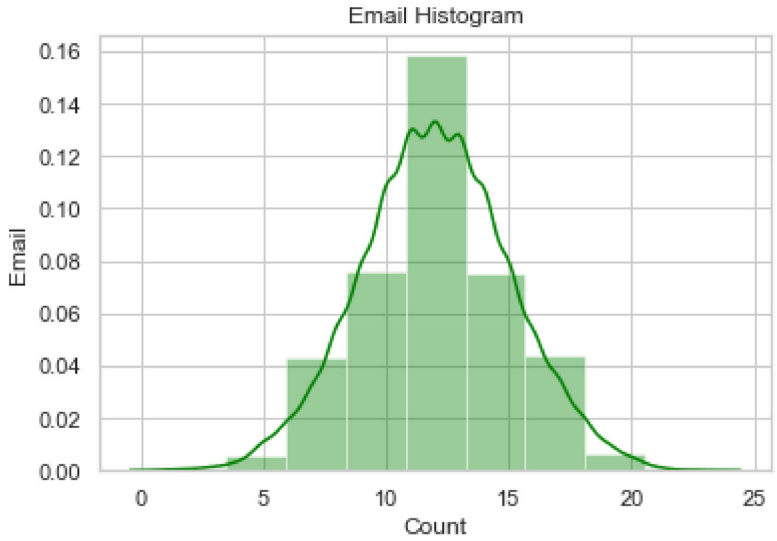
FIGURES/__FIG_3B_C8_3_AGE_HISTOGRAM.PNG



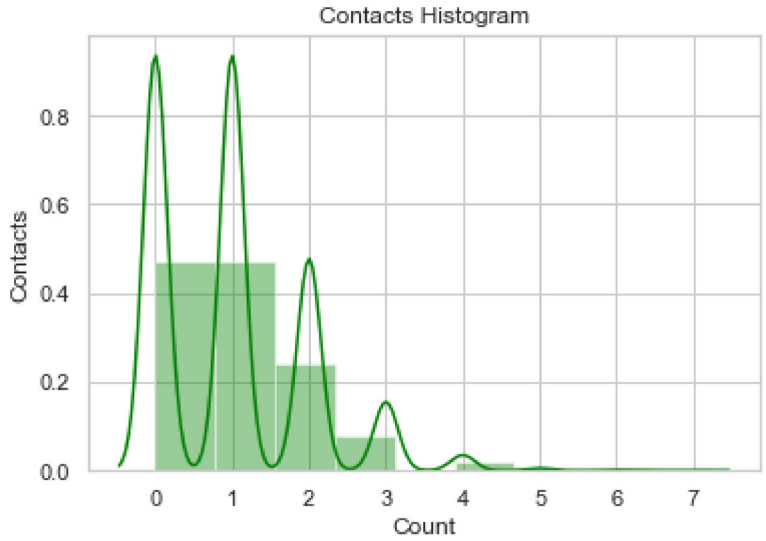
FIGURES/__FIG_3B_C8_4_INCOME_HISTOGRAM.PNG



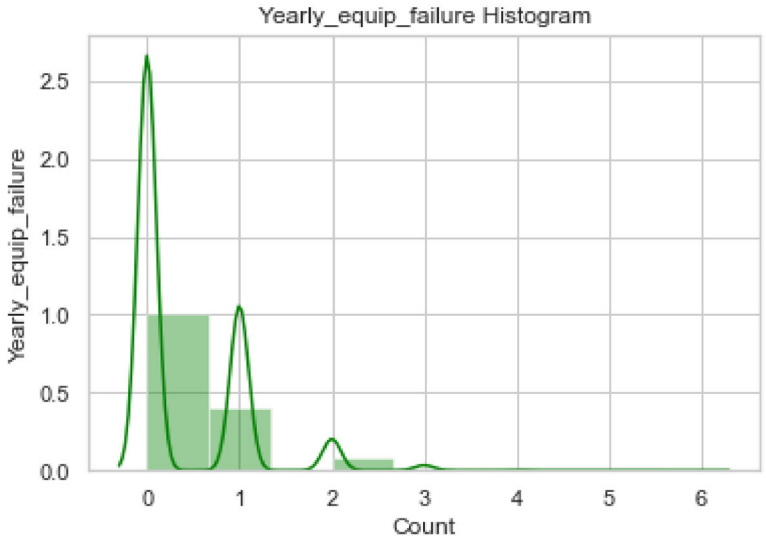
FIGURES/__FIG_3B_C8_5_OUTAGE_SEC_PERWEEK_HISTOGRAM.PNG



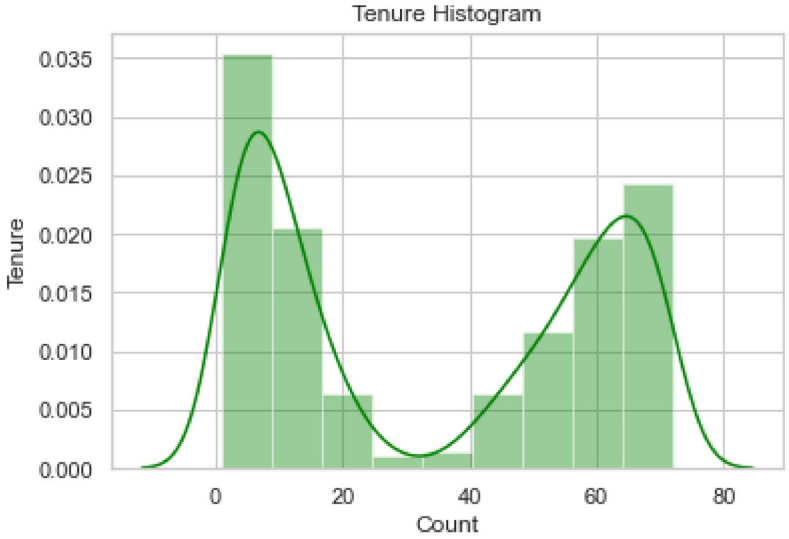
FIGURES/__FIG_3B_C8_6_EMAIL_HISTOGRAM.PNG



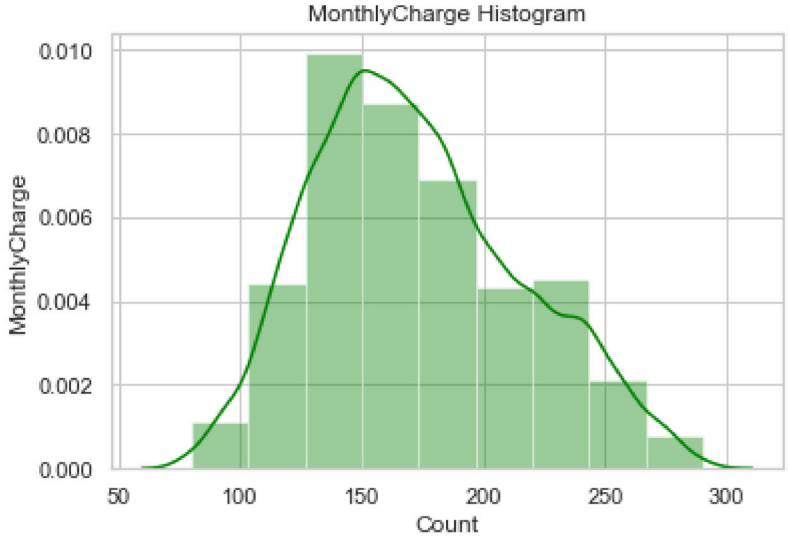
FIGURES/__FIG_3B_C8_7_CONTACTS_HISTOGRAM.PNG



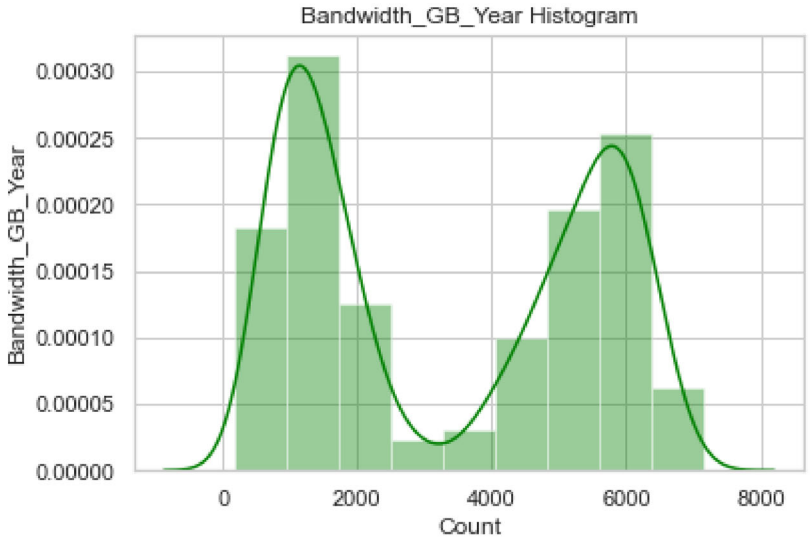
FIGURES/__FIG_3B_C8_8_YEARLY_EQUIP_FAILURE_HISTOGRAM.PNG



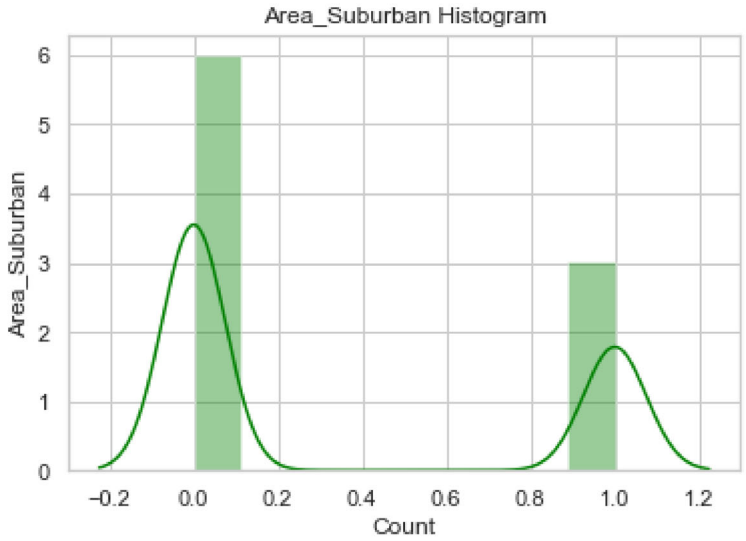
FIGURES/__FIG_3B_C8_9_TENURE_HISTOGRAM.PNG



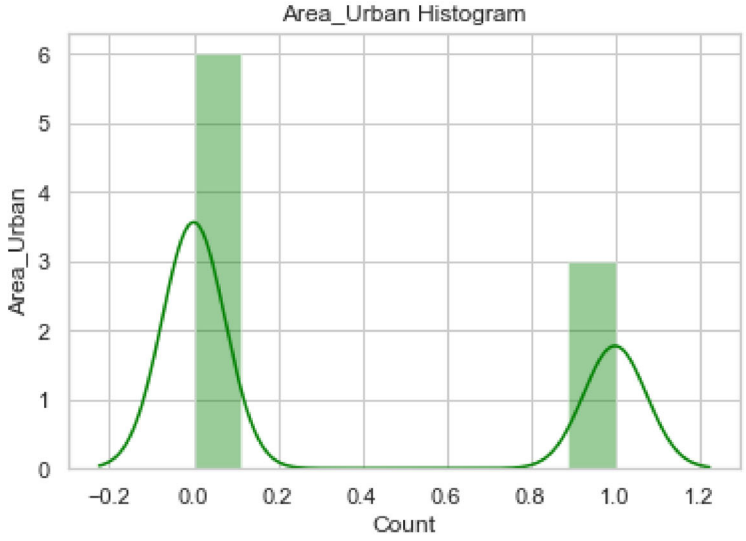
FIGURES/__FIG_3B_C8_10_MONTHLYCHARGE_HISTOGRAM.PNG



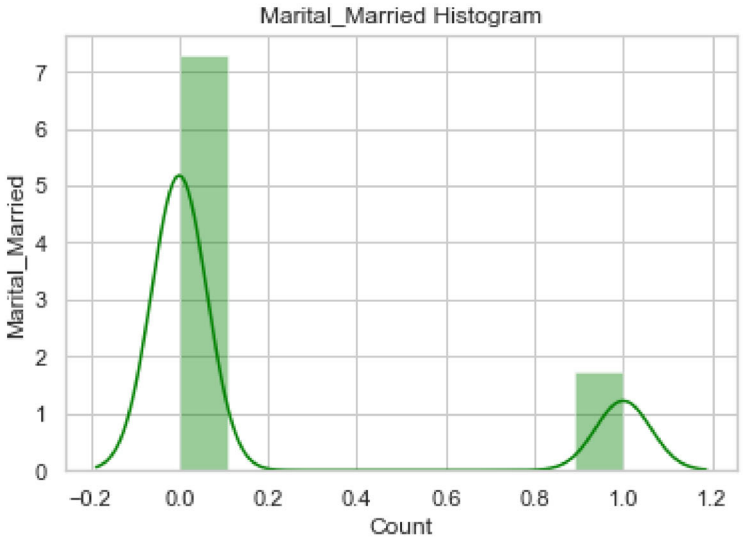
FIGURES/__FIG_3B_C8_11_BANDWIDTH_GB_YEAR_HISTOGRAM.PNG



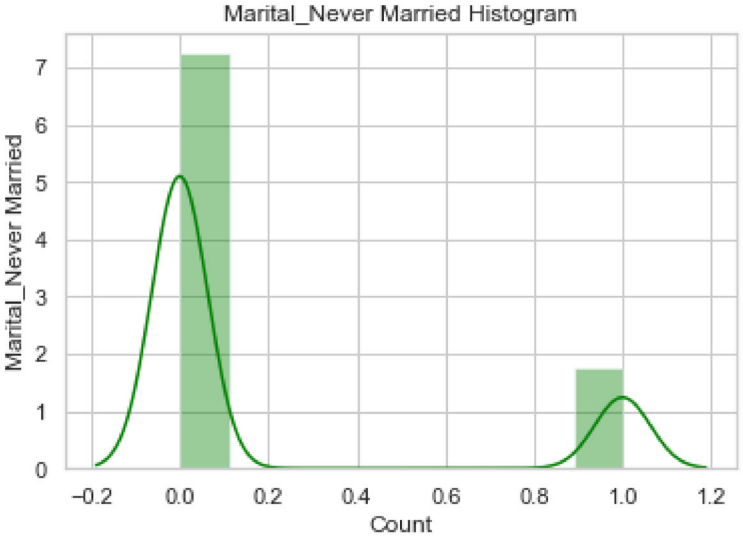
FIGURES/__FIG_3B_C8_12_AREA_SUBURBAN_HISTOGRAM.PNG



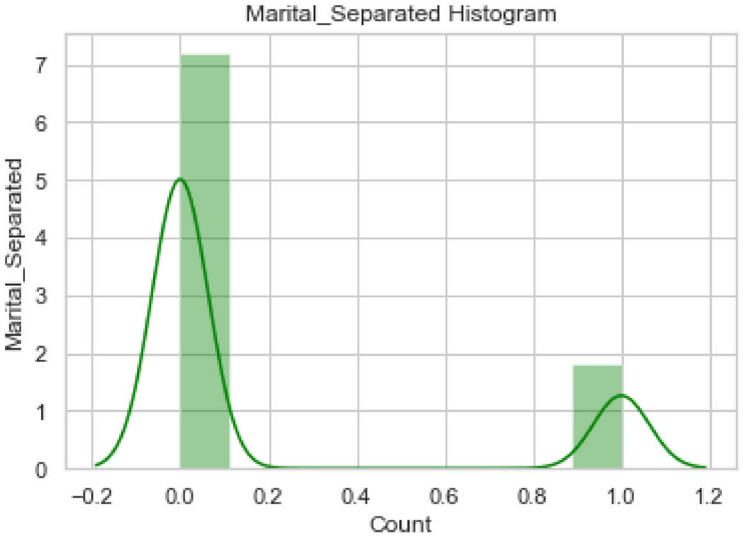
FIGURES/__FIG_3B_C8_13_AREA_URBAN_HISTOGRAM.PNG



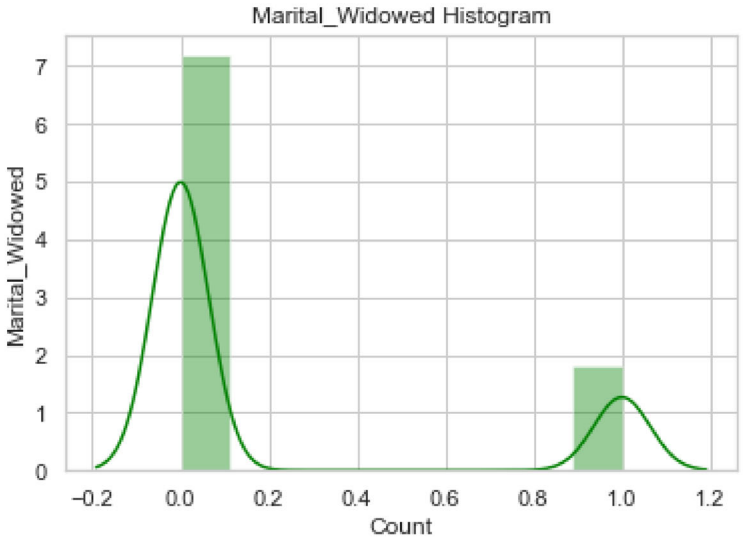
FIGURES/__FIG_3B_C8_14_MARITAL_MARRIED_HISTOGRAM.PNG



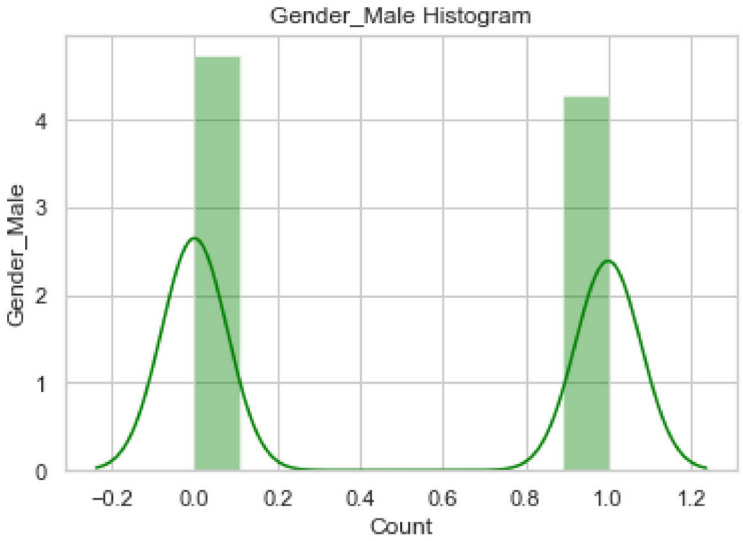
FIGURES/__FIG_3B_C8_15_MARITAL_NEVER_MARRIED_HISTOGRAM.PNG



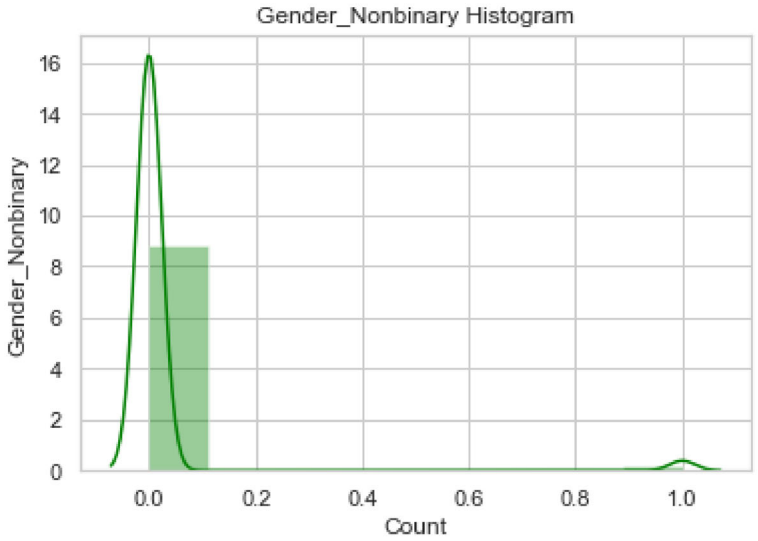
FIGURES/__FIG_3B_C8_16_MARITAL_SEPARATED_HISTOGRAM.PNG



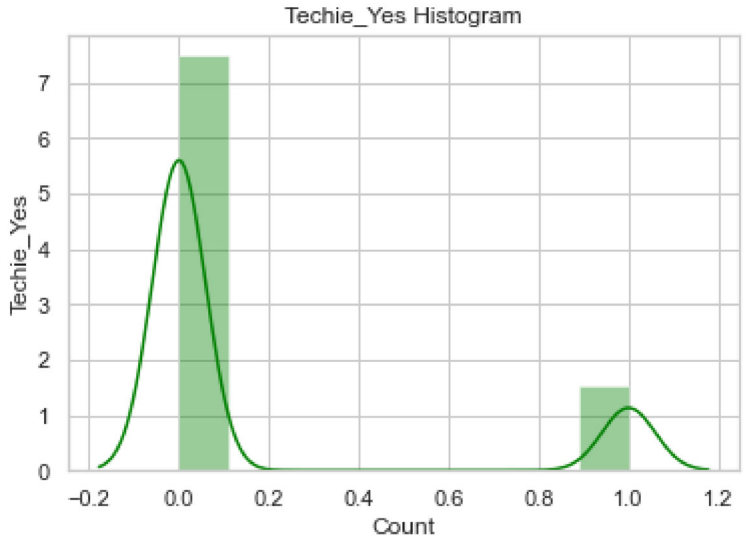
FIGURES/__FIG_3B_C8_17_MARITAL_WIDOWED_HISTOGRAM.PNG



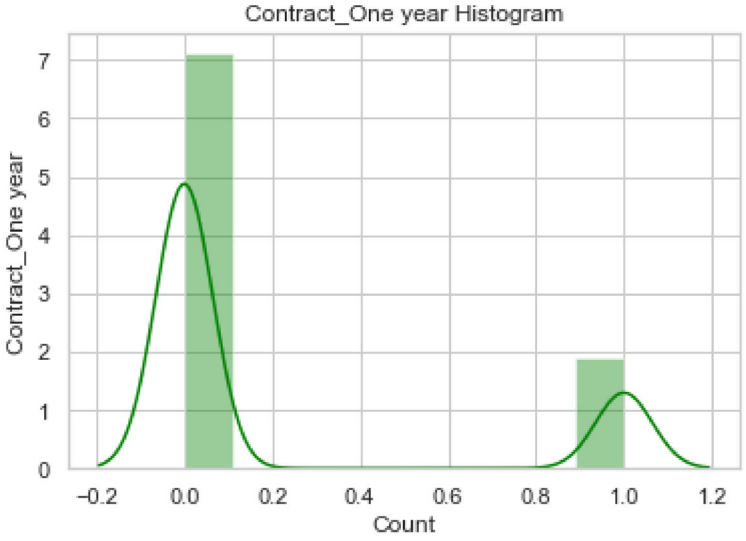
FIGURES/__FIG_3B_C8_18_GENDER_MALE_HISTOGRAM.PNG



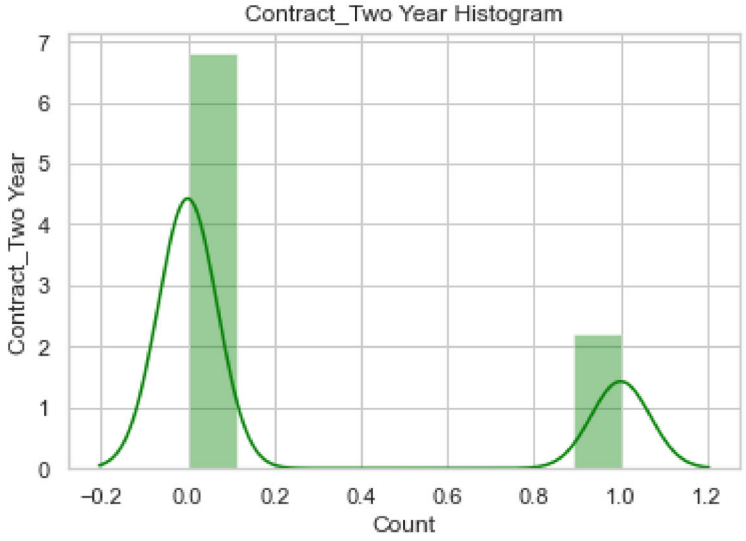
FIGURES/__FIG_3B_C8_19_GENDER_NONBINARY_HISTOGRAM.PNG



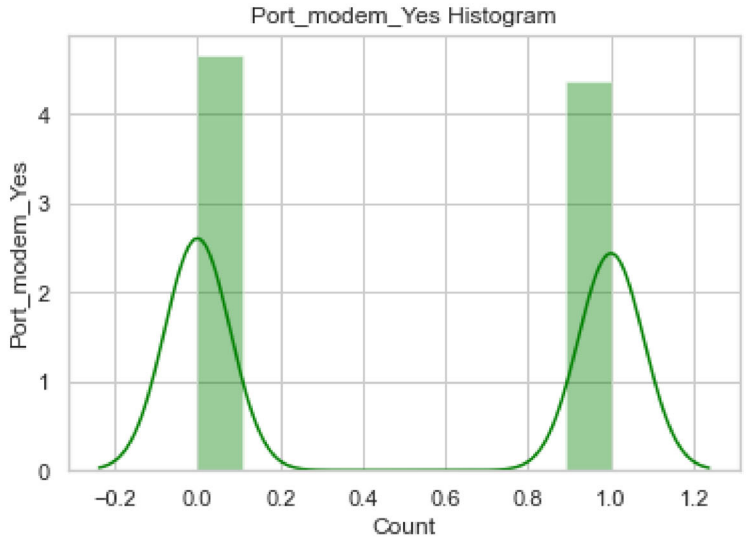
FIGURES/__FIG_3B_C8_20_TECHIE_YES_HISTOGRAM.PNG



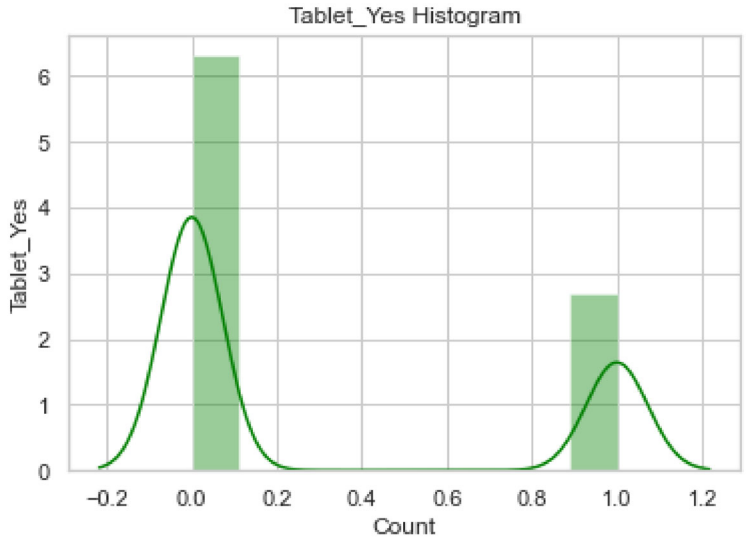
FIGURES/__FIG_3B_C8_21_CONTRACT_ONE_YEAR_HISTOGRAM.PNG



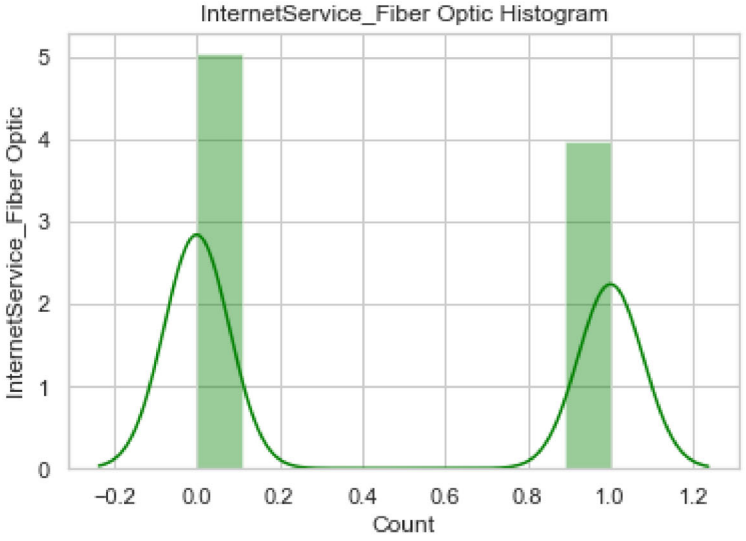
FIGURES/__FIG_3B_C8_22_CONTRACT_TWO_YEAR_HISTOGRAM.PNG



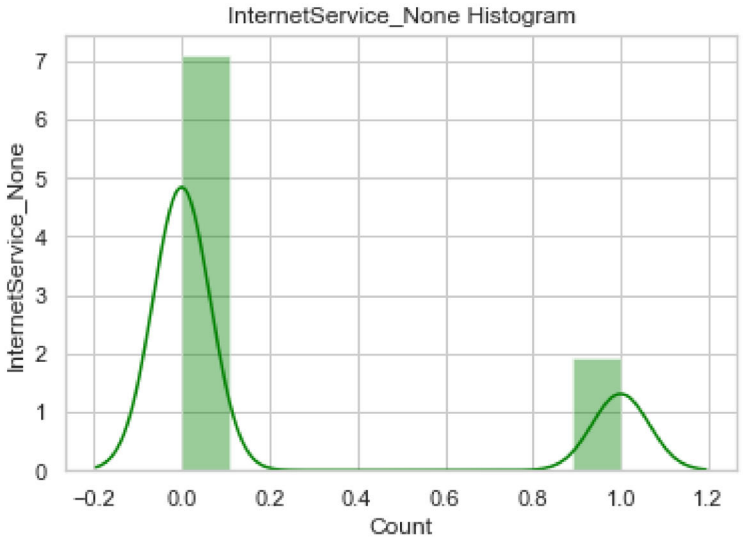
FIGURES/__FIG_3B_C8_23_PORT_MODEM_YES_HISTOGRAM.PNG



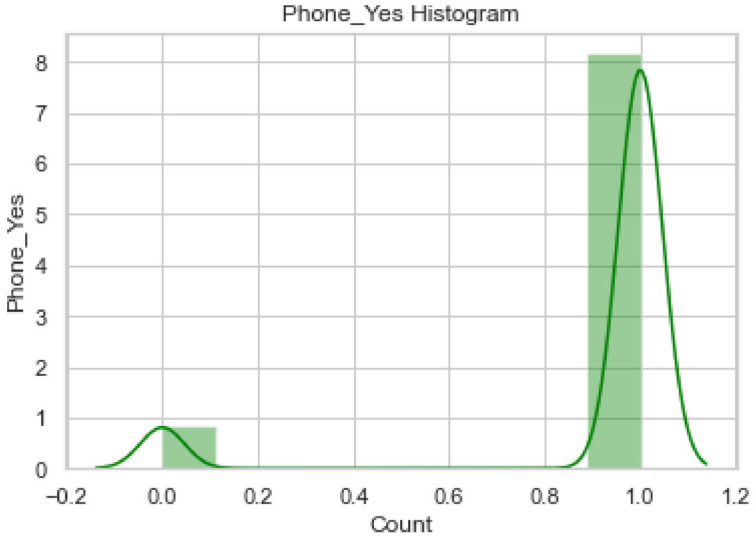
FIGURES/__FIG_3B_C8_24_TABLET_YES_HISTOGRAM.PNG



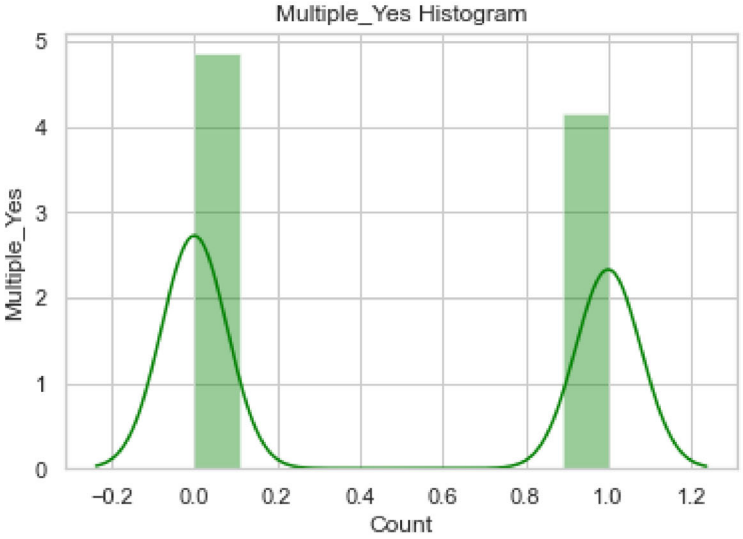
FIGURES/__FIG_3B_C8_25_INTERNETSERVICE_FIBER_OPTIC_HISTOGRAM.PNG



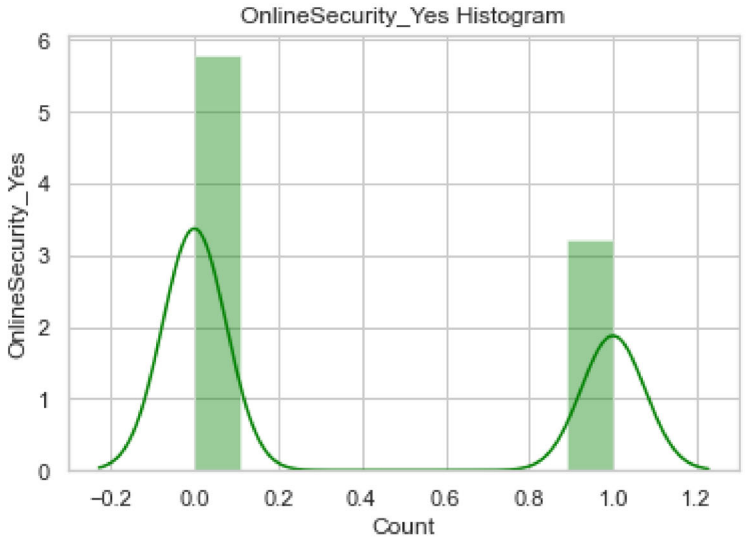
FIGURES/__FIG_3B_C8_26_INTERNETSERVICE_NONE_HISTOGRAM.PNG



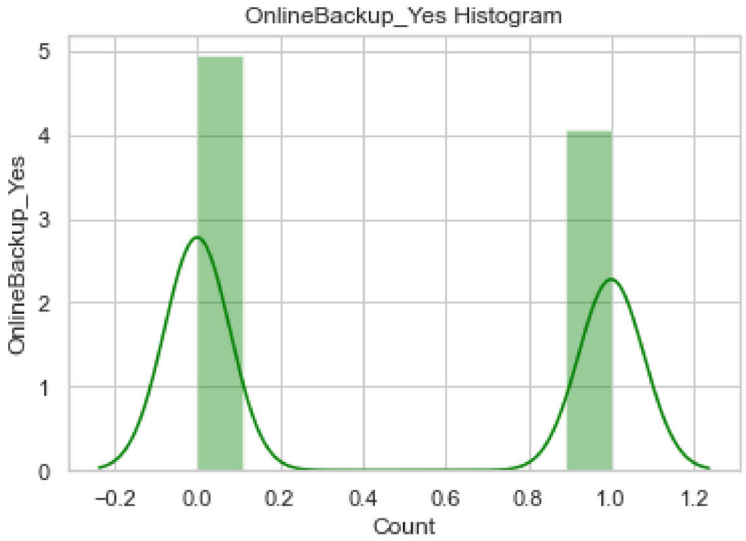
FIGURES/__FIG_3B_C8_27_PHONE_YES_HISTOGRAM.PNG



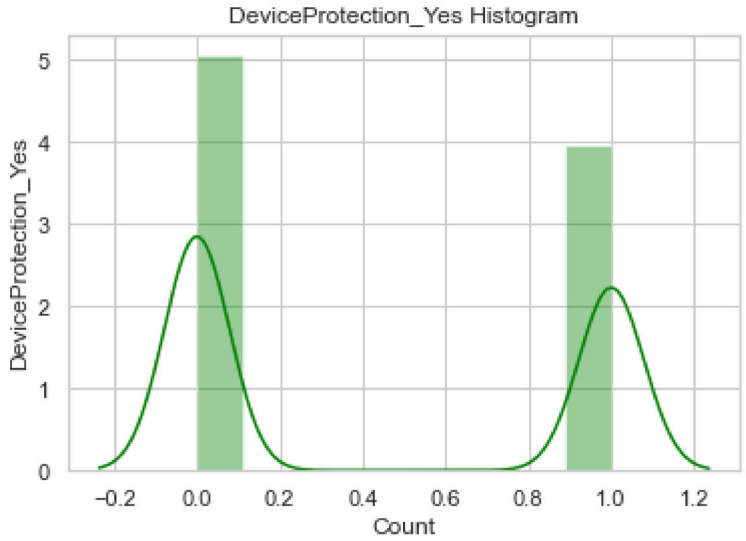
FIGURES/__FIG_3B_C8_28_MULTIPLE_YES_HISTOGRAM.PNG



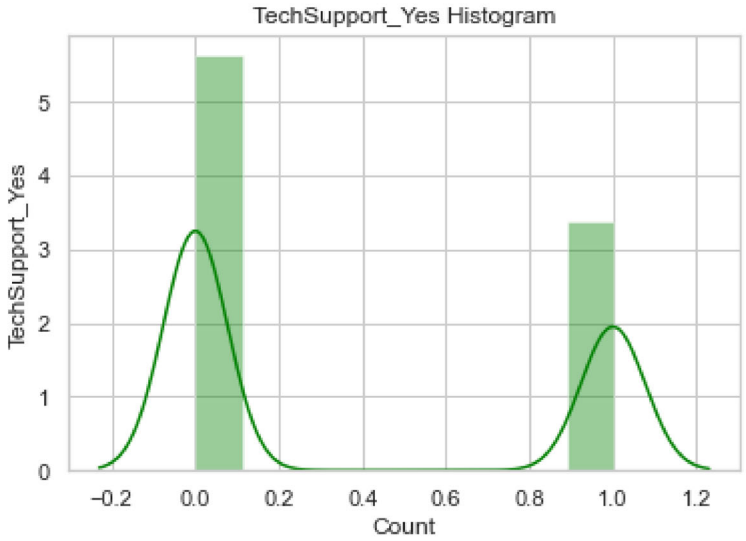
FIGURES/__FIG_3B_C8_29_ONLINESECURITY_YES_HISTOGRAM.PNG



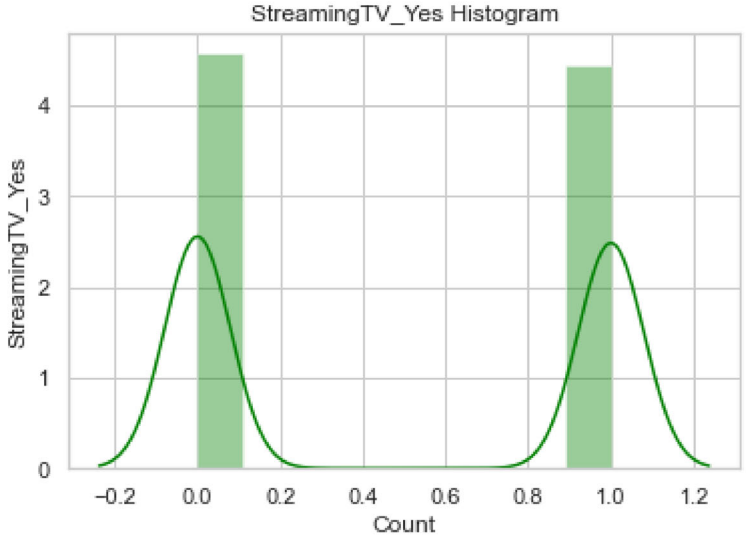
FIGURES/__FIG_3B_C8_30_ONLINEBACKUP_YES_HISTOGRAM.PNG



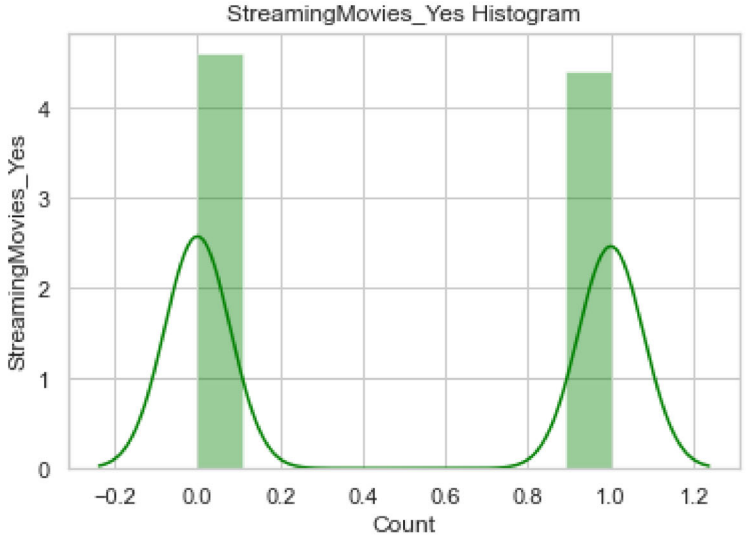
FIGURES/__FIG_3B_C8_31_DEVICEPROTECTION_YES_HISTOGRAM.PNG



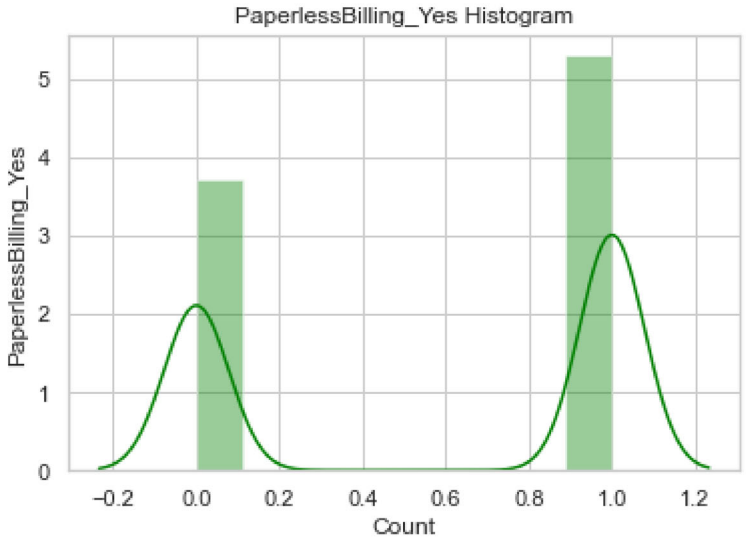
FIGURES/__FIG_3B_C8_32_TECHSUPPORT_YES_HISTOGRAM.PNG



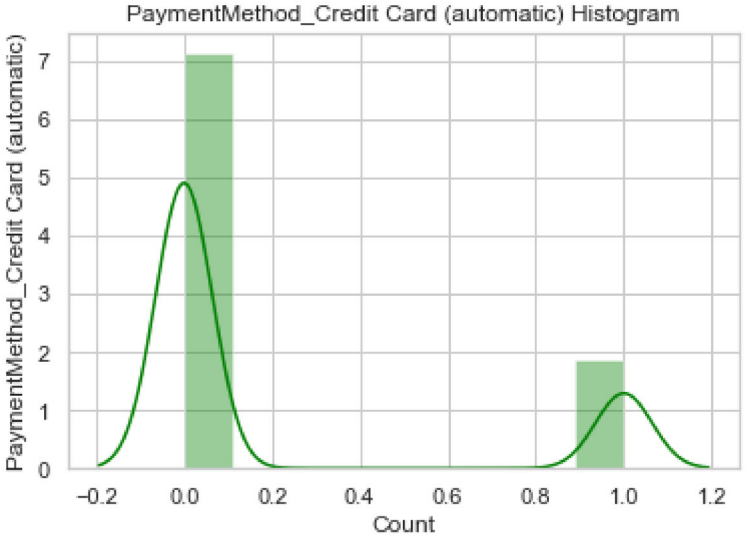
FIGURES/__FIG_3B_C8_33_STREAMINGTV_YES_HISTOGRAM.PNG



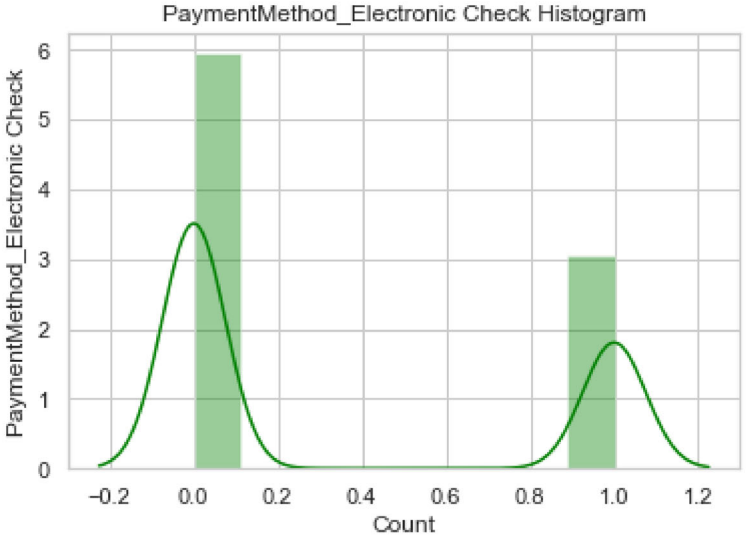
FIGURES/__FIG_3B_C8_34_STREAMINGMOVIES_YES_HISTOGRAM.PNG



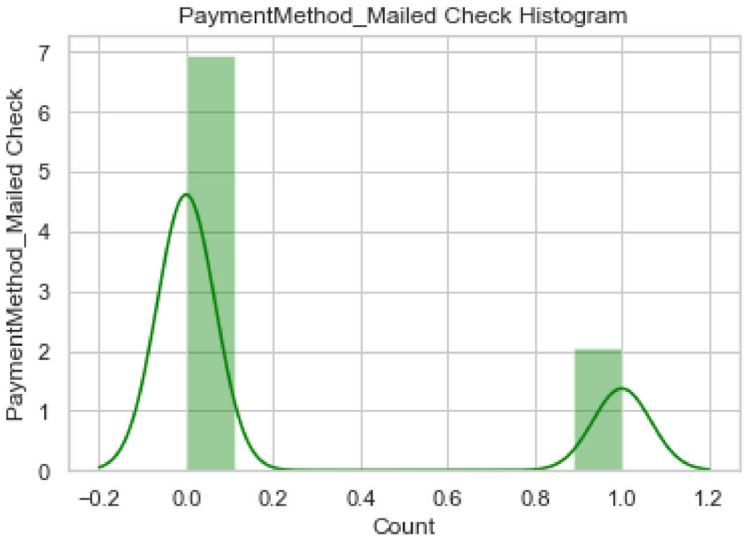
FIGURES/__FIG_3B_C8_35_PAPERLESSBILLING_YES_HISTOGRAM.PNG



FIGURES/__FIG_3B_C8_36_PAYMENTMETHOD_CREDIT_CARD_(AUTOMATIC)_HISTOGRAM.PNG



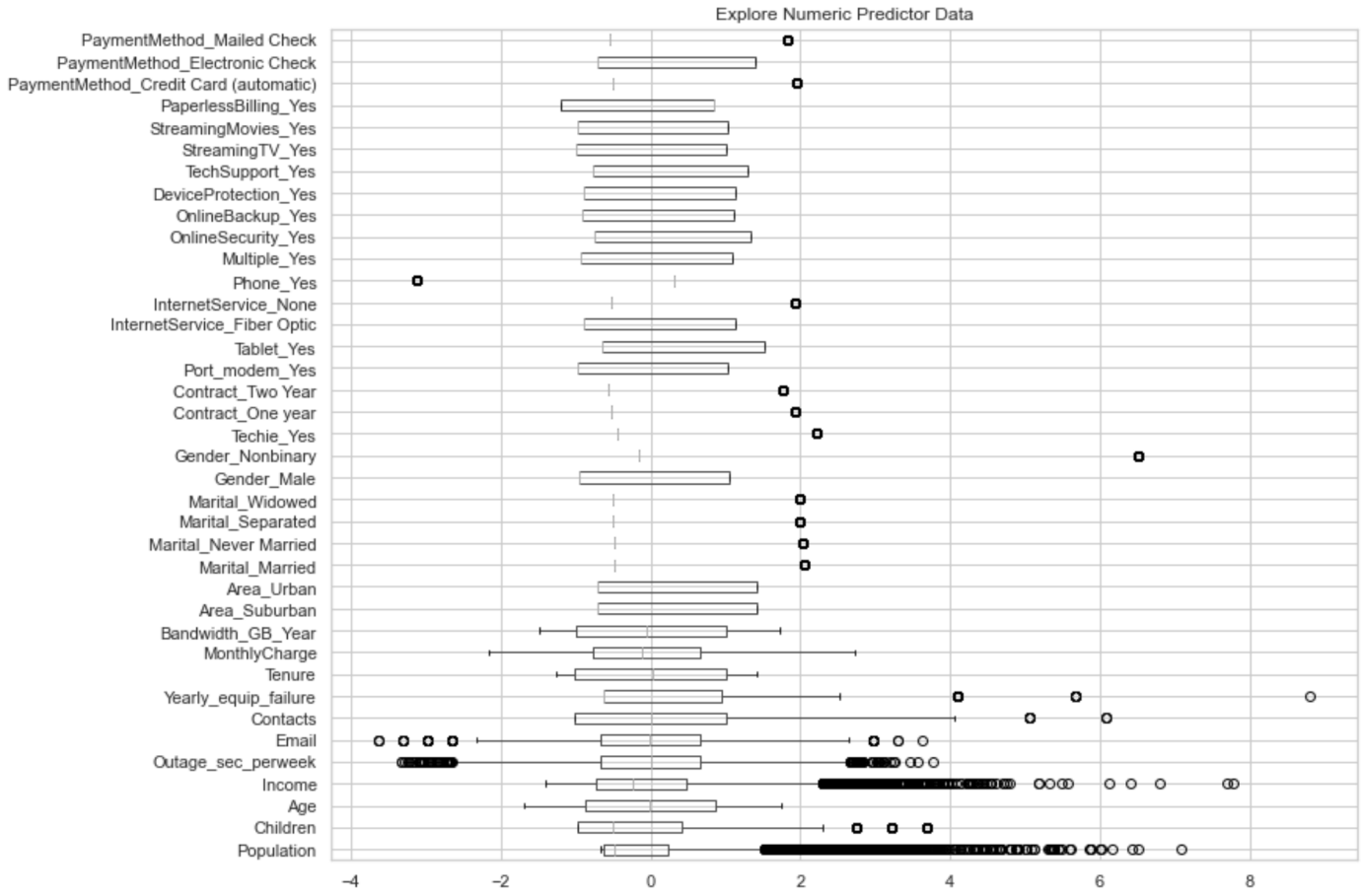
FIGURES/__FIG_3B_C8_37_PAYMENTMETHOD_ELECTRONIC_CHECK_HISTOGRAM.PNG



FIGURES/__FIG_3B_C8_38_PAYMENTMETHOD_MAILED_CHECK_HISTOGRAM.PNG

C12. Explore Numerical Data

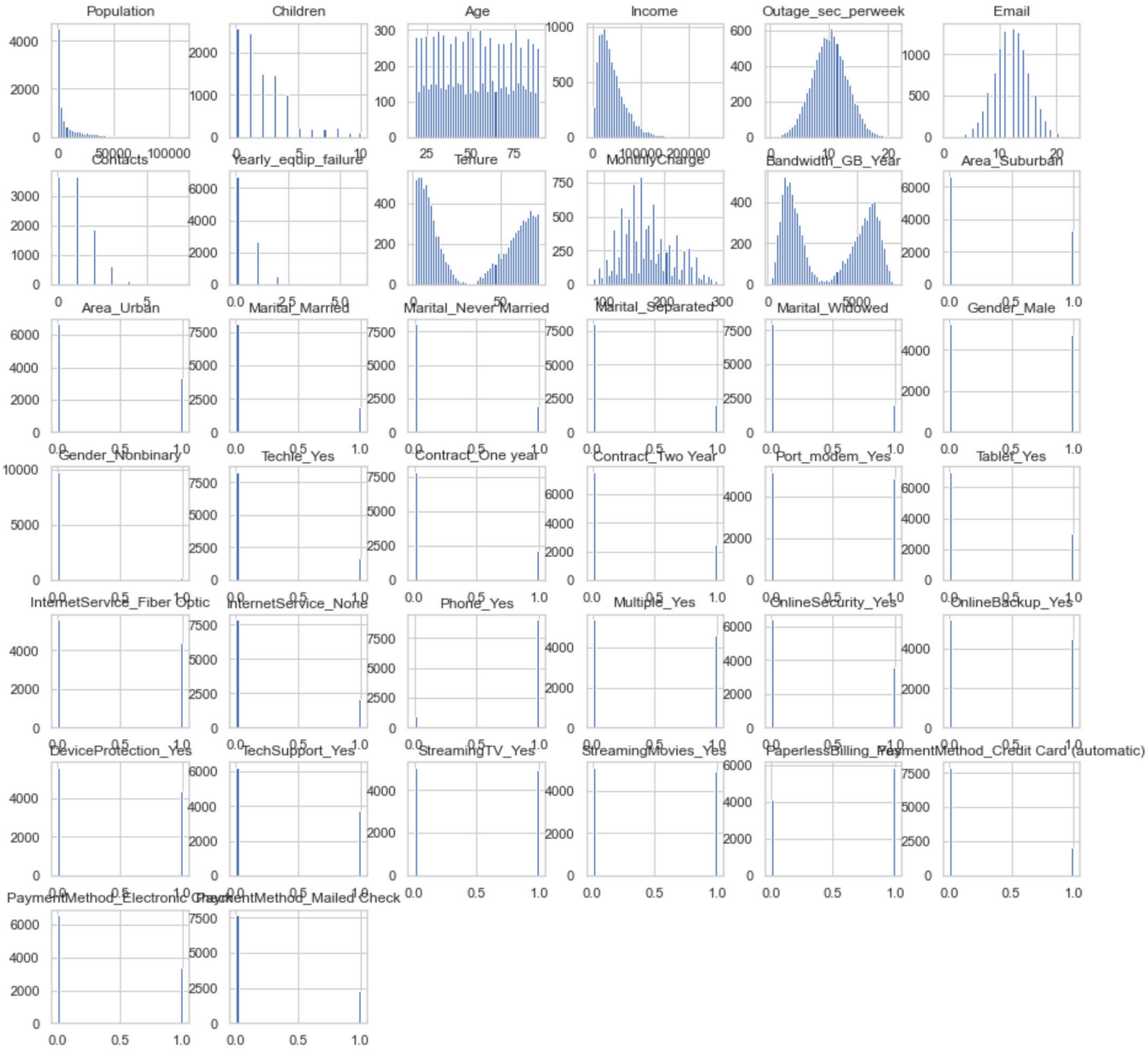
In [33]:
```
# explore numeric features
plt.figure(figsize=(12, 10))
std_numeric_data = (df_clean[numerical_features] -df_clean[numerical_features].mean()) / df_clea
ax = std_numeric_data.boxplot(vert=False)
plt.title('Explore Numeric Predictor Data')
plt.savefig('figures/' + 'FIG_C12_1_' + 'BOXPLOT.png', facecolor='w')
plt.show()
#print(std_numeric_data.describe(percentiles=None).round(3).T)
#print(df[num_cols].describe(percentiles=None).round(3).T)
```



C13. Explore Numerical Data.

In [34]:
```
# histogram plot numeric data
fig = plt.figure(figsize=(10, 20))
ax = df_clean[numerical_features].hist(bins = 50, figsize=(15,15))
plt.title('Numeric Data')
fig.tight_layout(h_pad=5, w_pad=5)
plt.savefig('figures/' + 'FIG_C13_1_' + 'HISTOGRAM.png', facecolor='w')
plt.show()
```

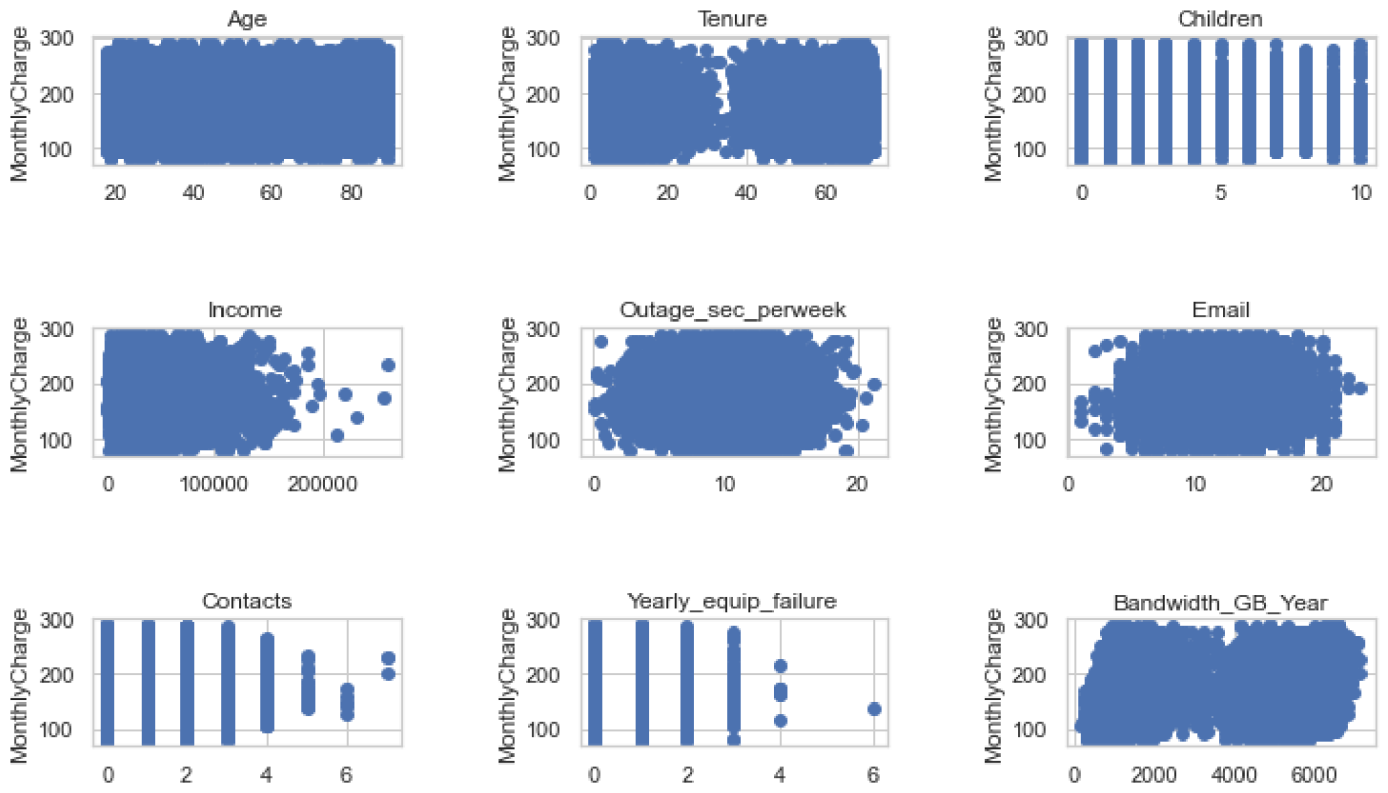
```
<Figure size 720x1440 with 0 Axes>
```



C14. Bivariate Scatter Plot of Numerical Features.

Here are the scatter plots of selected numeric data vs. the target variable of 'MonthlyCharge'. One of the assumptions is that independent and dependent variables are linear, so I am looking for linear relationships here.

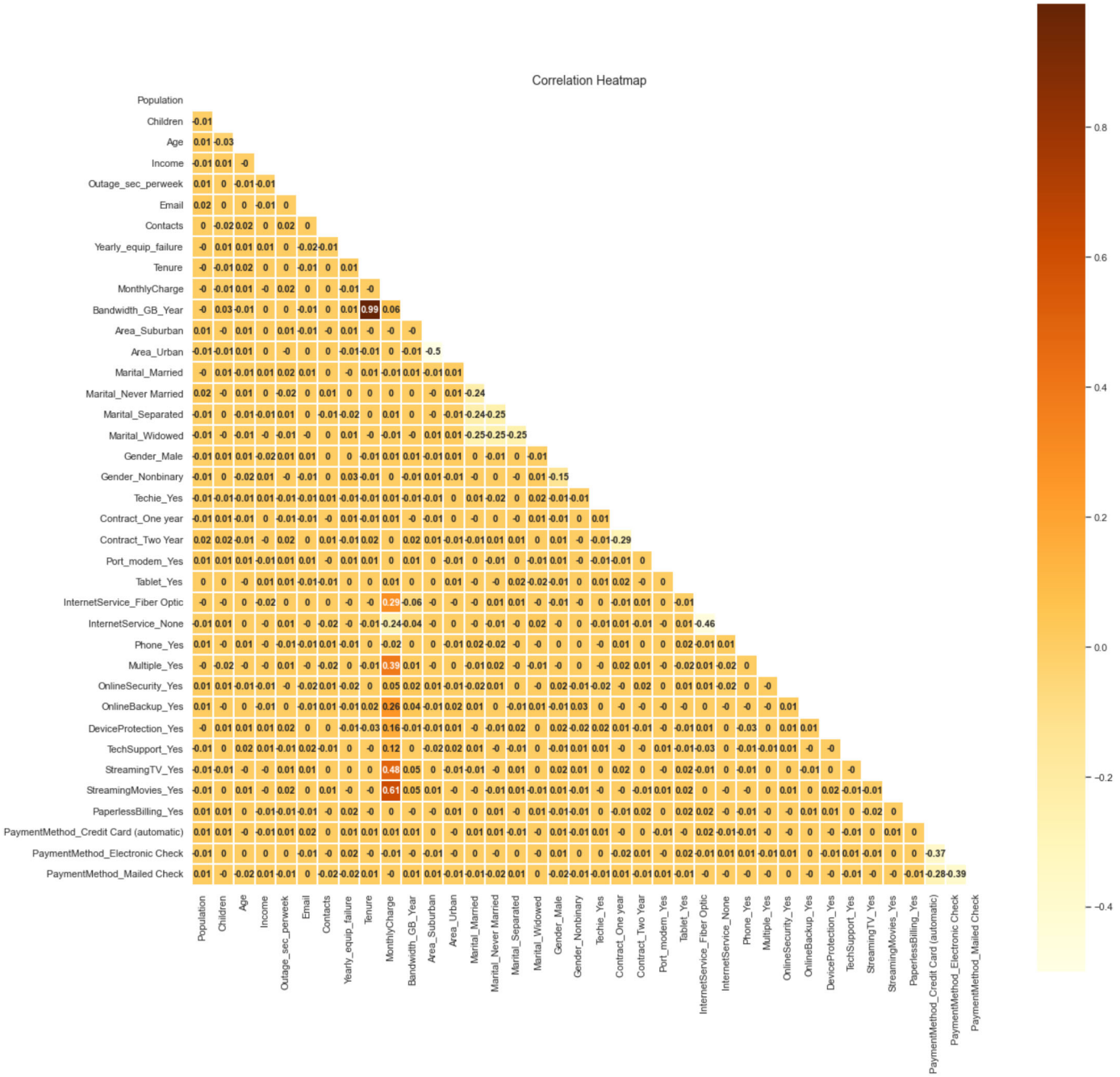
In [35]:
```
# scatter plot of selected features
fig = plt.figure(figsize=(10, 20))
features = ['Age','Tenure','Children','Income',
            'Outage_sec_perweek','Email','Contacts',
            'Yearly_equip_failure','Bandwidth_GB_Year']
target = df_clean['MonthlyCharge']

for i, c in enumerate(features):
    plt.subplot(10, 3, i+1)
    x = df_clean[c]
    y = target
    plt.scatter(x, y, marker='o')
    plt.title(c)
    plt.ylabel('MonthlyCharge')
    fig.tight_layout(h_pad=5, w_pad=5)

plt.savefig('figures/' + 'FIG_C14_1_' + 'BIVARIATE_SCATTER.png', facecolor='w')
```



In [ ]:

In [36]:
```
def Generate_heatmap_graph(corr, chart_title, mask_uppertri=False ):
    """ Based on features , generate correlation matrix """
    mask = np.zeros_like(corr)
    mask[np.triu_indices_from(mask)] = mask_uppertri
    fig,ax = plt.subplots(figsize=(20,20))
    sns.heatmap(corr
                , mask = mask
                , square = True
                , annot = True
                , annot_kws={'size': 10.5, 'weight' : 'bold'}
                , cmap=plt.get_cmap("YlOrBr")
                , linewidths=.1)
    plt.title(chart_title, fontsize=14)
    plt.show()
var_corr = round(df_clean.corr(),2)
```

```
Generate_heatmap_graph(var_corr
                       ,chart_title = 'Correlation Heatmap'
                       ,mask_uppertri = True)
```



sns.lmplot(x='Tenure', y='MonthlyCharge', data=df_clean, hue='Churn', fit_reg=False, markers=["o", "x"], palette= plotColor)
plt.show()

Part IV: Analysis

D. Perform data analysis and report on results by doing the following:

D1. Initial Model.

In [37]:
```
# convert array to list then append lists then remove target
model_features = numerical_features.tolist()
```

```
#model_features.remove('Churn')
```

In [38]:
```
target = 'MonthlyCharge'
model_features.remove(target)
```

In [39]:
```
y = df_clean[target]
X = df_clean[model_features]
```

In [40]:
```
# initial model
Xc = sm.add_constant(X)
model = sm.OLS(y, Xc).fit()
print(model.summary2()) # using alternate summary layout
```

```
                            Results: Ordinary least squares
==============================================================================================
Model:                        OLS                     Adj. R-squared:            0.995
Dependent Variable:           MonthlyCharge           AIC:                       49597.8789
Date:                         2022-05-28 18:02        BIC:                       49871.8719
No. Observations:             10000                   Log-Likelihood:            -24761.
Df Model:                     37                      F-statistic:               5.966e+04
Df Residuals:                 9962                    Prob (F-statistic):        0.00
R-squared:                    0.996                   Scale:                     8.3155
----------------------------------------------------------------------------------------------
                                    Coef.   Std.Err.     t     P>|t|   [0.025   0.975]
----------------------------------------------------------------------------------------------
const                             -88.3666   0.6488 -136.2005 0.0000 -89.6384 -87.0948
Population                         -0.0000   0.0000   -0.9277 0.3536  -0.0000   0.0000
Children                           -9.5210   0.0358 -266.2744 0.0000  -9.5910  -9.4509
Age                                 1.0142   0.0038  268.2800 0.0000   1.0068   1.0217
Income                              0.0000   0.0000    0.7061 0.4801  -0.0000   0.0000
Outage_sec_perweek                  0.0057   0.0097    0.5916 0.5541  -0.0133   0.0248
Email                              -0.0017   0.0095   -0.1781 0.8587  -0.0204   0.0170
Contacts                           -0.0199   0.0292   -0.6800 0.4965  -0.0772   0.0374
Yearly_equip_failure               -0.0078   0.0454   -0.1719 0.8635  -0.0969   0.0812
Tenure                            -25.3563   0.0881 -287.6991 0.0000 -25.5291 -25.1835
Bandwidth_GB_Year                   0.3095   0.0011  287.7098 0.0000   0.3074   0.3116
Area_Suburban                      -0.1677   0.0707   -2.3731 0.0177  -0.3062  -0.0292
Area_Urban                         -0.0841   0.0708   -1.1886 0.2346  -0.2229   0.0546
Marital_Married                    -0.0133   0.0914   -0.1453 0.8845  -0.1924   0.1658
Marital_Never Married              -0.0524   0.0909   -0.5763 0.5644  -0.2305   0.1257
Marital_Separated                   0.0672   0.0901    0.7458 0.4558  -0.1094   0.2438
Marital_Widowed                    -0.0146   0.0900   -0.1626 0.8709  -0.1911   0.1618
Gender_Male                       -20.1183   0.0905 -222.2582 0.0000 -20.2958 -19.9409
Gender_Nonbinary                    6.5516   0.1960   33.4241 0.0000   6.1674   6.9359
Techie_Yes                          0.0512   0.0773    0.6625 0.5077  -0.1003   0.2027
Contract_One year                   0.0415   0.0742    0.5594 0.5759  -0.1039   0.1868
Contract_Two Year                   0.0638   0.0703    0.9074 0.3642  -0.0740   0.2017
Port_modem_Yes                      0.0449   0.0578    0.7772 0.4371  -0.0683   0.1581
Tablet_Yes                         -0.0009   0.0631   -0.0138 0.9890  -0.1246   0.1229
InternetService_Fiber Optic       148.0308   0.4503  328.7031 0.0000 147.1480 148.9136
InternetService_None              115.2702   0.4524  254.7721 0.0000 114.3833 116.1571
Phone_Yes                          -0.0183   0.0993   -0.1841 0.8539  -0.2130   0.1764
Multiple_Yes                       10.3089   0.0967  106.5650 0.0000  10.1192  10.4985
OnlineSecurity_Yes                -20.7889   0.1014 -204.9362 0.0000 -20.9878 -20.5901
OnlineBackup_Yes                   -6.5860   0.1168  -56.4004 0.0000  -6.8149  -6.3571
DeviceProtection_Yes              -13.7972   0.1084 -127.2990 0.0000 -14.0096 -13.5847
TechSupport_Yes                    11.1126   0.0599  185.5809 0.0000  10.9953  11.2300
StreamingTV_Yes                   -28.3892   0.2520 -112.6494 0.0000 -28.8832 -27.8952
StreamingMovies_Yes               -12.7029   0.2333  -54.4516 0.0000 -13.1602 -12.2456
PaperlessBilling_Yes               -0.0733   0.0587   -1.2489 0.2117  -0.1884   0.0418
PaymentMethod_Credit Card (automatic)   0.0268   0.0880    0.3041 0.7610  -0.1457   0.1993
PaymentMethod_Electronic Check      0.0688   0.0787    0.8733 0.3825  -0.0856   0.2231
PaymentMethod_Mailed Check          0.1827   0.0860    2.1256 0.0336   0.0142   0.3512
----------------------------------------------------------------------------------------------
Omnibus:                    48314.337          Durbin-Watson:                 2.011
Prob(Omnibus):              0.000              Jarque-Bera (JB):              1091.294
Skew:                       -0.026             Prob(JB):                      0.000
Kurtosis:                   1.382              Condition No.:                 1670116
==============================================================================================
* The condition number is large (2e+06). This might indicate             strong
multicollinearity or other numerical problems.
```

D2. Feature Reduction.

In [41]:
```
# find predictor pairs with high coorelation
#custom_corr_matrix(X,'Model_2')
get_top_n_correlations(X, 20)
```

Out[41]:

Tenure	Bandwidth_GB_Year	0.991495
Area_Suburban	Area_Urban	0.500711
InternetService_Fiber Optic	InternetService_None	0.461753
PaymentMethod_Electronic Check	PaymentMethod_Mailed Check	0.390989
PaymentMethod_Credit Card (automatic)	PaymentMethod_Electronic Check	0.367992
Contract_One year	Contract_Two Year	0.293243
PaymentMethod_Credit Card (automatic)	PaymentMethod_Mailed Check	0.279547
Marital_Separated	Marital_Widowed	0.253210
Marital_Never Married	Marital_Widowed	0.248636
	Marital_Separated	0.247636
Marital_Married	Marital_Widowed	0.245075
	Marital_Separated	0.244089
	Marital_Never Married	0.239680
Gender_Male	Gender_Nonbinary	0.146092
Bandwidth_GB_Year	InternetService_Fiber Optic	0.061956
	StreamingTV_Yes	0.054314
	StreamingMovies_Yes	0.045600
	InternetService_None	0.044727
	OnlineBackup_Yes	0.041740
Phone_Yes	DeviceProtection_Yes	0.030339

dtype: float64

In [42]:
```
# drop all columns from model where p-value > 0.05 (see Geeks for Geeks (2021))
equation = model.summary2().tables[1]
temp_drop = []
for i in equation.itertuples():
    if i[4] > 0.05:
        temp_drop.append(i[0])
        print('Drop {} with p-value of {:.3f}.'.format(i[0],i[4]))
X = pd.DataFrame(X) # reset dataframe
X.drop(temp_drop, axis = 1, inplace=True) # drop
```

```
Drop Population with p-value of 0.354.
Drop Income with p-value of 0.480.
Drop Outage_sec_perweek with p-value of 0.554.
Drop Email with p-value of 0.859.
Drop Contacts with p-value of 0.497.
Drop Yearly_equip_failure with p-value of 0.863.
Drop Area_Urban with p-value of 0.235.
Drop Marital_Married with p-value of 0.885.
Drop Marital_Never Married with p-value of 0.564.
Drop Marital_Separated with p-value of 0.456.
Drop Marital_Widowed with p-value of 0.871.
Drop Techie_Yes with p-value of 0.508.
Drop Contract_One year with p-value of 0.576.
Drop Contract_Two Year with p-value of 0.364.
Drop Port_modem_Yes with p-value of 0.437.
Drop Tablet_Yes with p-value of 0.989.
Drop Phone_Yes with p-value of 0.854.
Drop PaperlessBilling_Yes with p-value of 0.212.
Drop PaymentMethod_Credit Card (automatic) with p-value of 0.761.
Drop PaymentMethod_Electronic Check with p-value of 0.383.
```

In [43]:
```
# drop other columns with high multi-collinearity (see Geeks for Geeks (2021))
temp_drop =['Bandwidth_GB_Year', 'Tenure','Children','InternetService_None','Age']
X = pd.DataFrame(X) # reset dataframe
X.drop(temp_drop, axis = 1, inplace=True) # drop
```

In [44]:
```
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 12 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Area_Suburban                 10000 non-null  uint8
 1   Gender_Male                   10000 non-null  uint8
 2   Gender_Nonbinary              10000 non-null  uint8
 3   InternetService_Fiber Optic   10000 non-null  uint8
 4   Multiple_Yes                  10000 non-null  uint8
 5   OnlineSecurity_Yes            10000 non-null  uint8
 6   OnlineBackup_Yes              10000 non-null  uint8
 7   DeviceProtection_Yes          10000 non-null  uint8
 8   TechSupport_Yes               10000 non-null  uint8
 9   StreamingTV_Yes               10000 non-null  uint8
 10  StreamingMovies_Yes           10000 non-null  uint8
 11  PaymentMethod_Mailed Check    10000 non-null  uint8
dtypes: uint8(12)
memory usage: 117.3 KB
```

D3. Updated Model.

In [45]:
```
# updated model
Xc = sm.add_constant(X)
model = sm.OLS(y, Xc).fit()
print(model.summary2()) # using alternate summary layout
```

```
                    Results: Ordinary least squares
==============================================================================
Model:                 OLS               Adj. R-squared:        0.946
Dependent Variable:    MonthlyCharge     AIC:                   74366.7372
Date:                  2022-05-28 18:02  BIC:                   74460.4716
No. Observations:      10000             Log-Likelihood:        -37170.
Df Model:              12                F-statistic:           1.465e+04
Df Residuals:          9987              Prob (F-statistic):    0.00
R-squared:             0.946             Scale:                 99.236
------------------------------------------------------------------------------
                            Coef.  Std.Err.    t     P>|t|   [0.025  0.975]
------------------------------------------------------------------------------
const                      78.8953   0.3004 262.6253 0.0000 78.3065 79.4842
Area_Suburban               0.0860   0.2112   0.4072 0.6839 -0.3280  0.5000
Gender_Male                -0.1950   0.2018  -0.9662 0.3340 -0.5906  0.2006
Gender_Nonbinary           -0.9131   0.6708  -1.3613 0.1735 -2.2279  0.4017
InternetService_Fiber Optic 24.7407  0.2008 123.2322 0.0000 24.3471 25.1342
Multiple_Yes               32.7987   0.1999 164.0971 0.0000 32.4069 33.1905
OnlineSecurity_Yes          2.8015   0.2080  13.4711 0.0000  2.3938  3.2091
OnlineBackup_Yes           22.5834   0.2004 112.7161 0.0000 22.1906 22.9761
DeviceProtection_Yes       12.4557   0.2009  62.0043 0.0000 12.0619 12.8494
TechSupport_Yes            12.6435   0.2059  61.3954 0.0000 12.2398 13.0472
StreamingTV_Yes            42.1841   0.1993 211.6510 0.0000 41.7934 42.5748
StreamingMovies_Yes        52.3389   0.1994 262.5063 0.0000 51.9481 52.7297
PaymentMethod_Mailed Check  0.1155   0.2372   0.4872 0.6261 -0.3493  0.5804
------------------------------------------------------------------------------
Omnibus:                29582.971        Durbin-Watson:           2.007
Prob(Omnibus):          0.000            Jarque-Bera (JB):        699.816
Skew:                   -0.052           Prob(JB):                0.000
Kurtosis:               1.708            Condition No.:           12
==============================================================================
```

D4. Updated Model.

In [46]:
```
# find predictor pairs with high coorelation
get_top_n_correlations(X, 10)
```

Out[46]:
```
Gender_Male                 Gender_Nonbinary             0.146092
Gender_Nonbinary            OnlineBackup_Yes             0.029316
InternetService_Fiber Optic TechSupport_Yes              0.026211
Gender_Male                 PaymentMethod_Mailed Check   0.022103
DeviceProtection_Yes        StreamingMovies_Yes          0.019450
Gender_Male                 DeviceProtection_Yes         0.018678
Gender_Nonbinary            DeviceProtection_Yes         0.016523
Gender_Male                 OnlineSecurity_Yes           0.016105
Area_Suburban               TechSupport_Yes              0.015650
Gender_Male                 StreamingTV_Yes              0.015094
dtype: float64
```

In [47]:
```
# drop all columns from model where p-value > 0.05 (see Geeks for Geeks (2021))
equation = model.summary2().tables[1]
temp_drop = []
for i in equation.itertuples():
    if i[4] > 0.05:
        temp_drop.append(i[0])
        print('Drop {} with p-value of {:.3f}.'.format(i[0],i[4]))
X = pd.DataFrame(X) # reset dataframe
X.drop(temp_drop, axis = 1, inplace=True) # drop
```

```
Drop Area_Suburban with p-value of 0.684.
Drop Gender_Male with p-value of 0.334.
Drop Gender_Nonbinary with p-value of 0.173.
Drop PaymentMethod_Mailed Check with p-value of 0.626.
```

D5. Final Model.

In [48]:
```
Xc = sm.add_constant(X) # reset
model = sm.OLS(y, Xc).fit()
print(model.summary2()) # using alternate summary layout
```

```
                    Results: Ordinary least squares
================================================================================
Model:                 OLS               Adj. R-squared:        0.946
Dependent Variable:    MonthlyCharge     AIC:                   74361.6344
Date:                  2022-05-28 18:02  BIC:                   74426.5275
No. Observations:      10000             Log-Likelihood:        -37172.
Df Model:              8                 F-statistic:           2.198e+04
Df Residuals:          9991              Prob (F-statistic):    0.00
R-squared:             0.946             Scale:                 99.225
--------------------------------------------------------------------------------
                            Coef.  Std.Err.    t     P>|t|   [0.025  0.975]
--------------------------------------------------------------------------------
const                      78.8429   0.2689 293.2430 0.0000 78.3159 79.3700
InternetService_Fiber Optic 24.7418   0.2007 123.2522 0.0000 24.3483 25.1353
Multiple_Yes               32.7986   0.1999 164.1060 0.0000 32.4068 33.1904
OnlineSecurity_Yes          2.8009   0.2079  13.4720 0.0000  2.3934  3.2084
OnlineBackup_Yes           22.5755   0.2002 112.7469 0.0000 22.1830 22.9680
DeviceProtection_Yes       12.4560   0.2008  62.0275 0.0000 12.0624 12.8496
TechSupport_Yes            12.6418   0.2059  61.4057 0.0000 12.2382 13.0454
StreamingTV_Yes            42.1795   0.1993 211.6736 0.0000 41.7889 42.5701
StreamingMovies_Yes        52.3391   0.1994 262.5468 0.0000 51.9484 52.7299
--------------------------------------------------------------------------------
Omnibus:                30980.596         Durbin-Watson:            2.008
Prob(Omnibus):          0.000             Jarque-Bera (JB):         702.004
Skew:                   -0.052            Prob(JB):                 0.000
Kurtosis:               1.706             Condition No.:            5
================================================================================
```

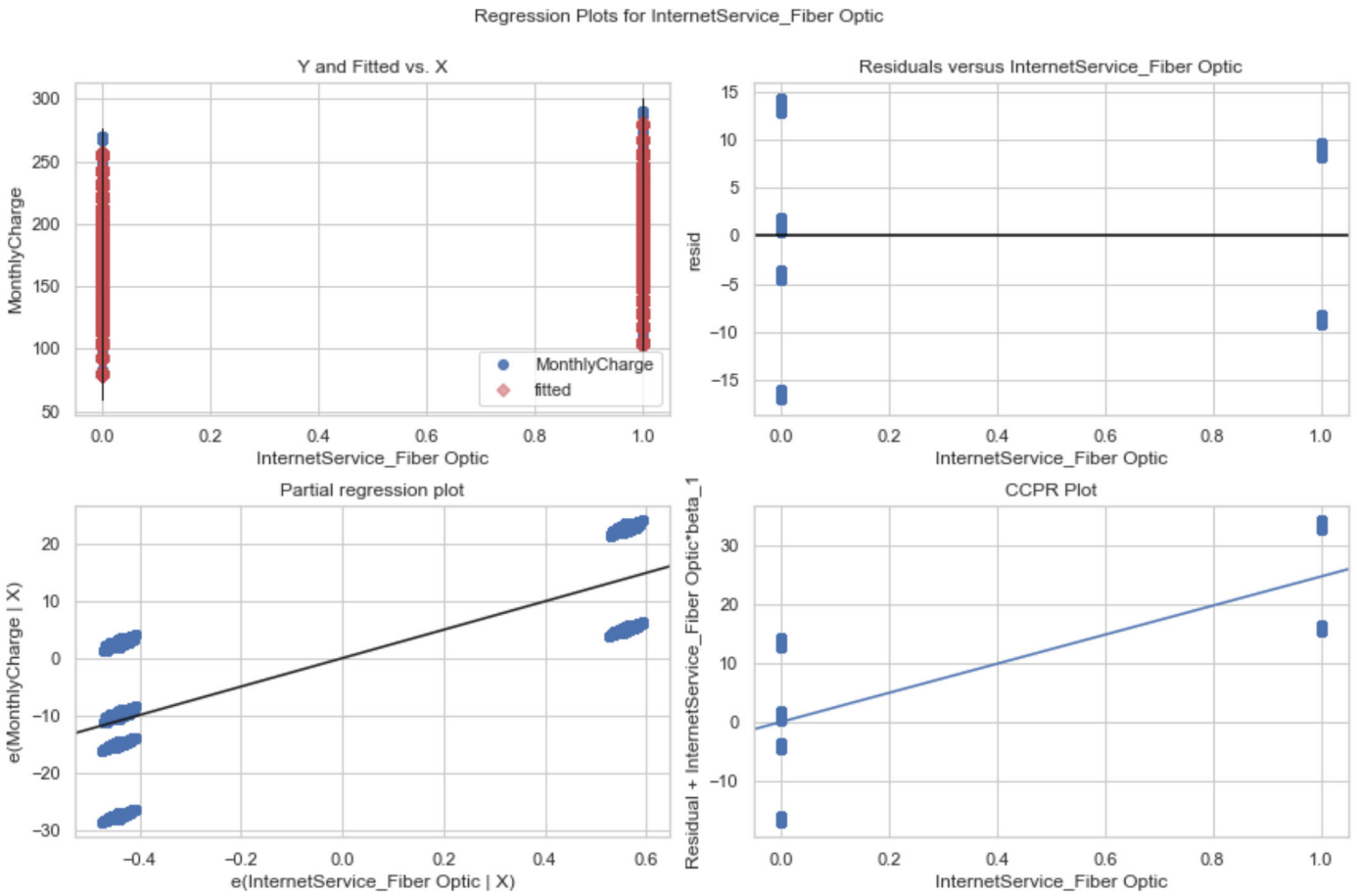
Part V: Compare and Results

E. Compare models and explain results.

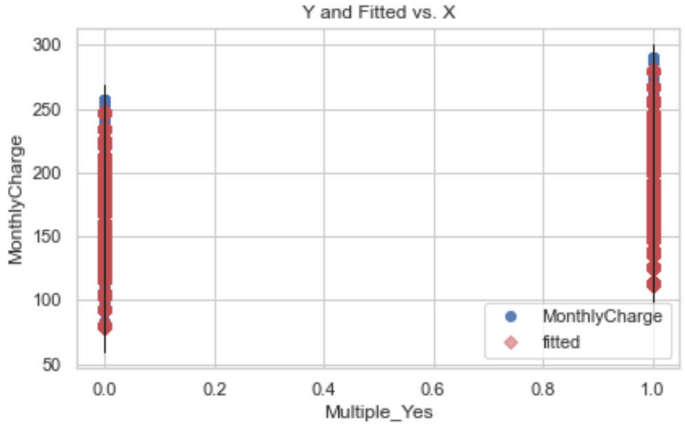
E1. Final Model.

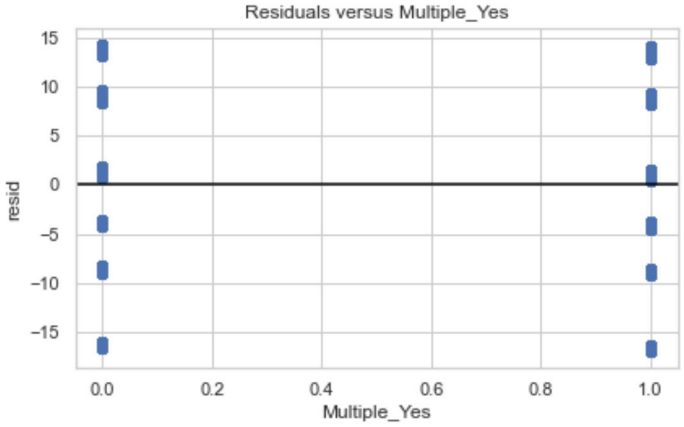
In [49]:
```
#create residual plots for all of the model's final predictor variables
for c in X.columns:
    fig = plt.figure(figsize=(12,8))
    fig = sm.graphics.plot_regress_exog(model, c, fig=fig)
```

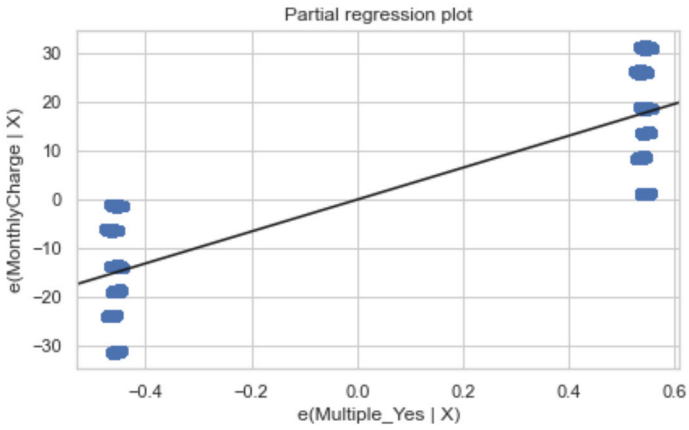
```
eval_env: 1
eval_env: 1
eval_env: 1
eval_env: 1
eval_env: 1
eval_env: 1
eval_env: 1
eval_env: 1
```

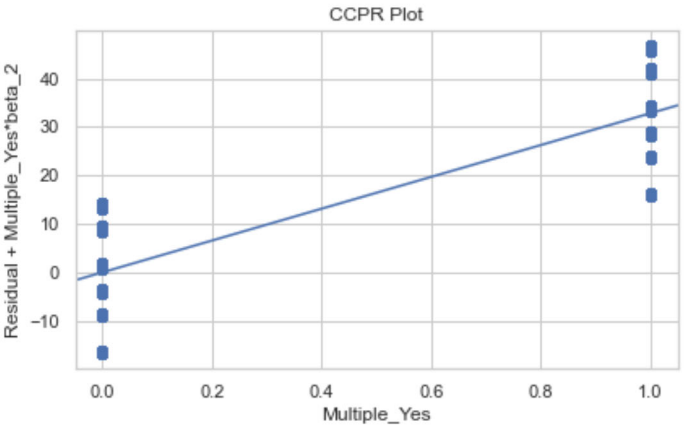


Regression Plots for Multiple_Yes

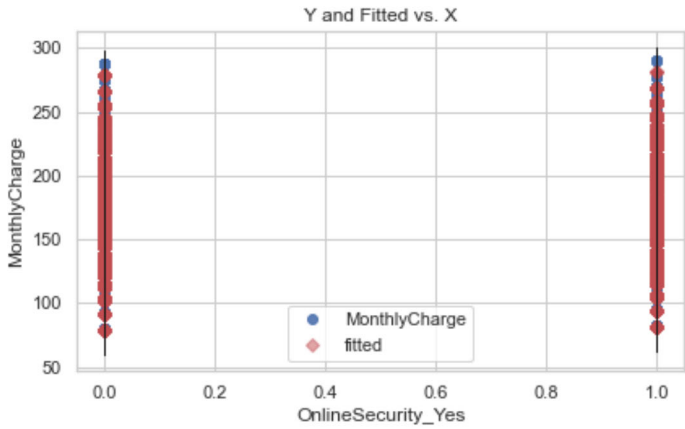


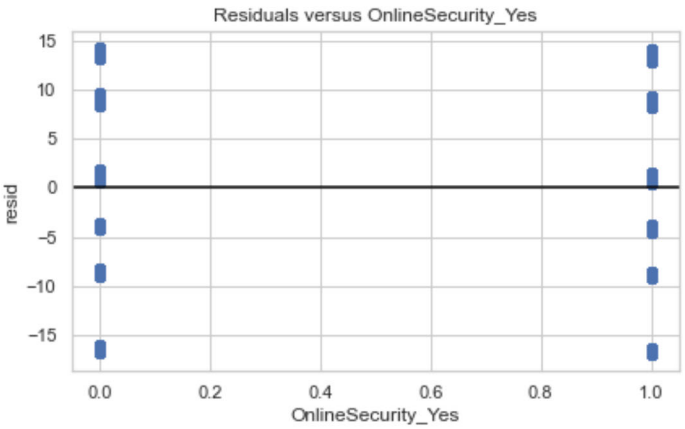


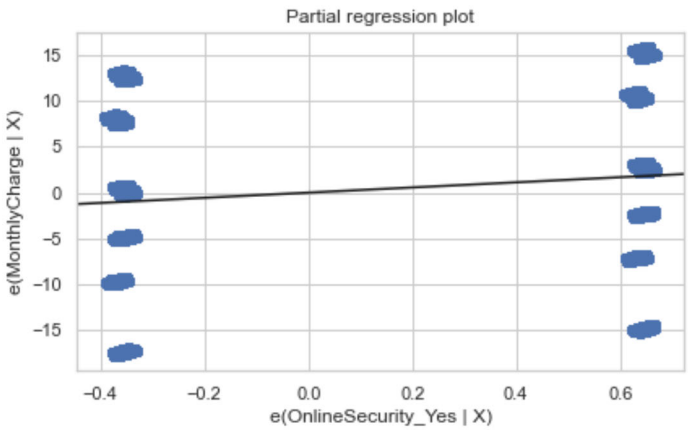


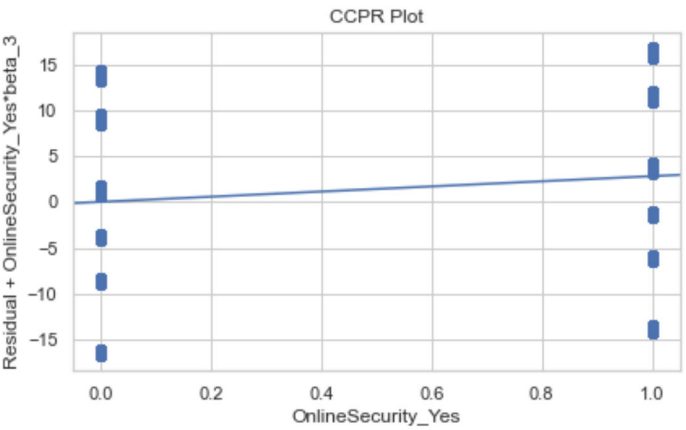


Regression Plots for OnlineSecurity_Yes

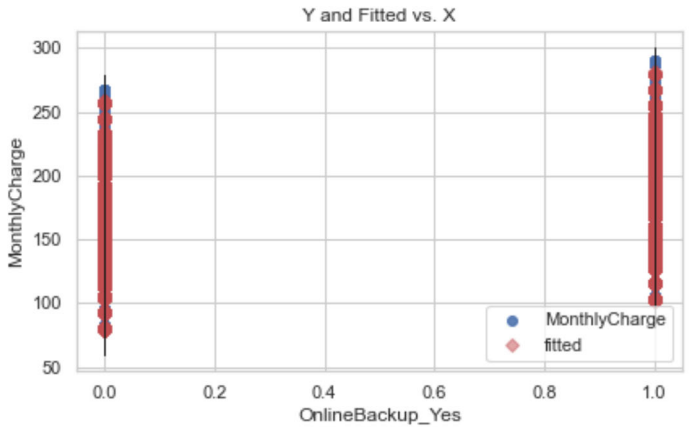


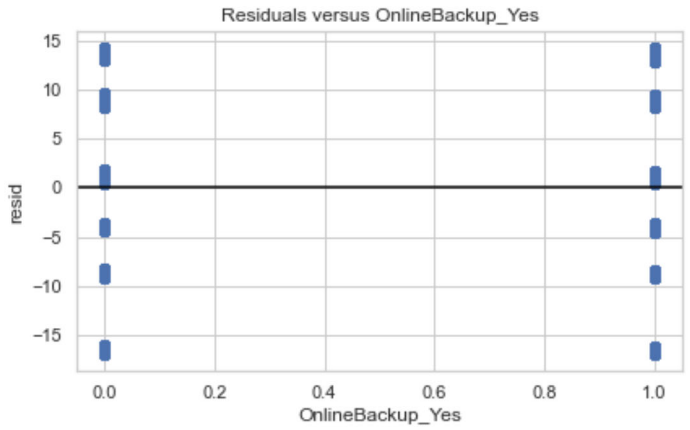


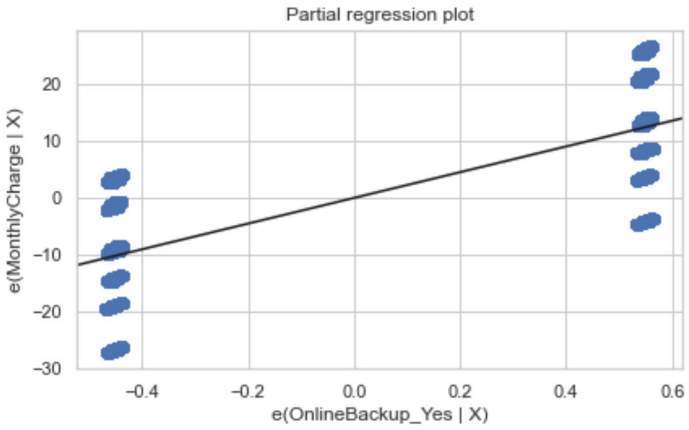


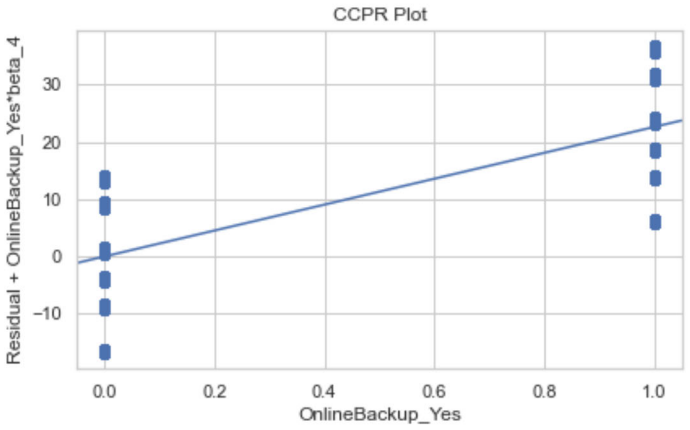


Regression Plots for OnlineBackup_Yes

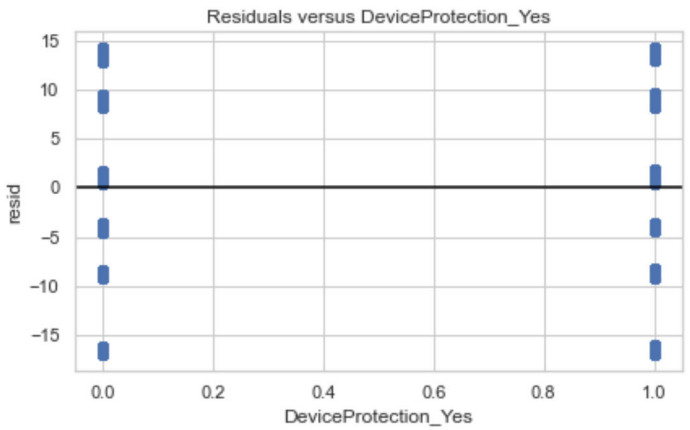


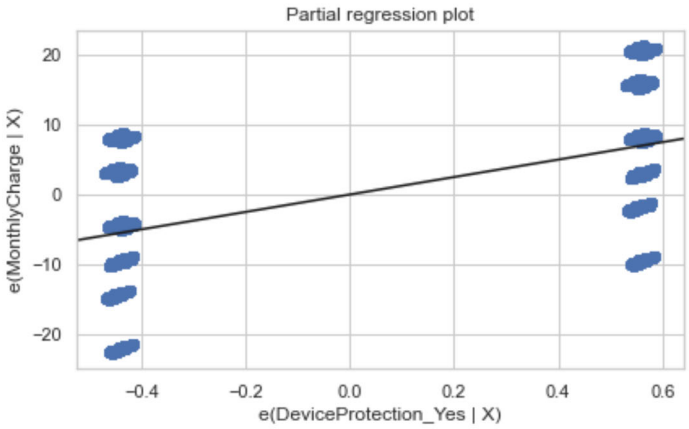


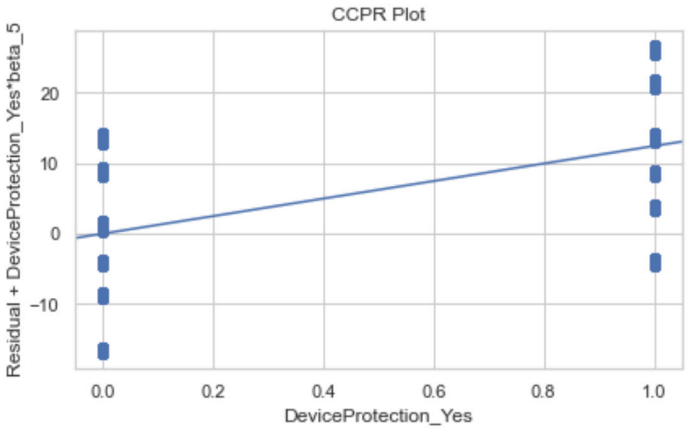




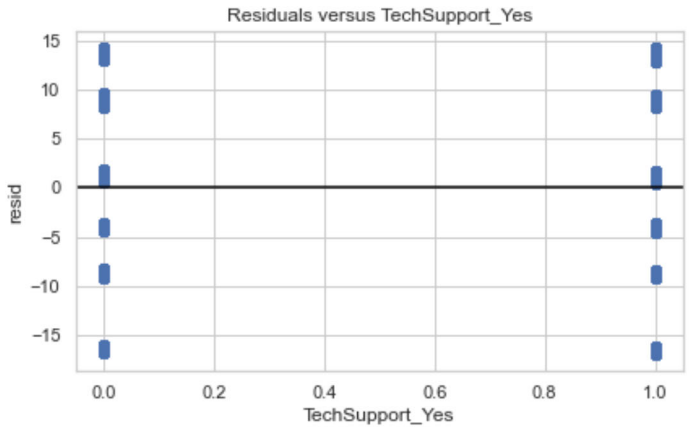
Regression Plots for DeviceProtection_Yes

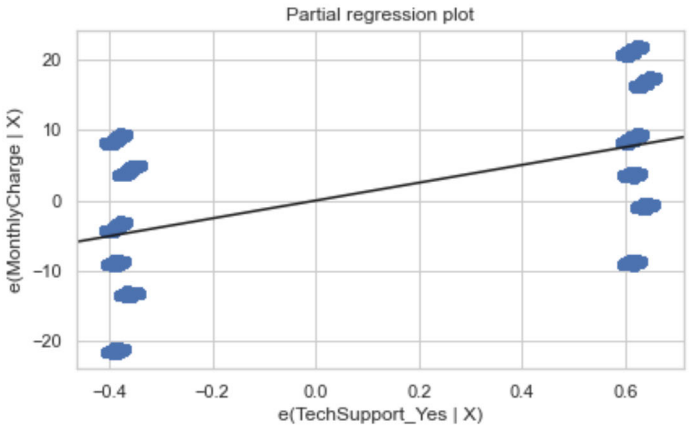


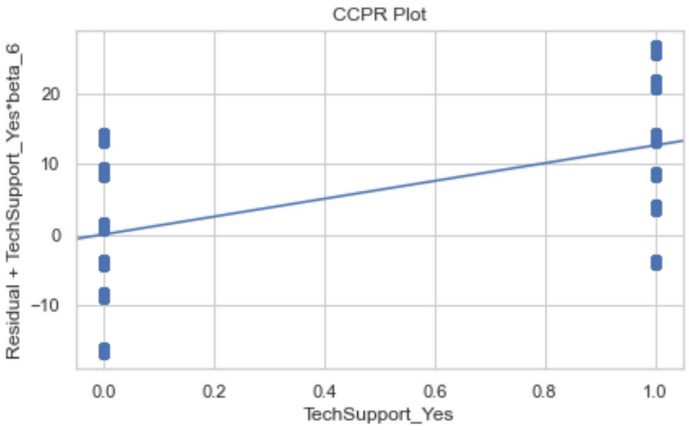




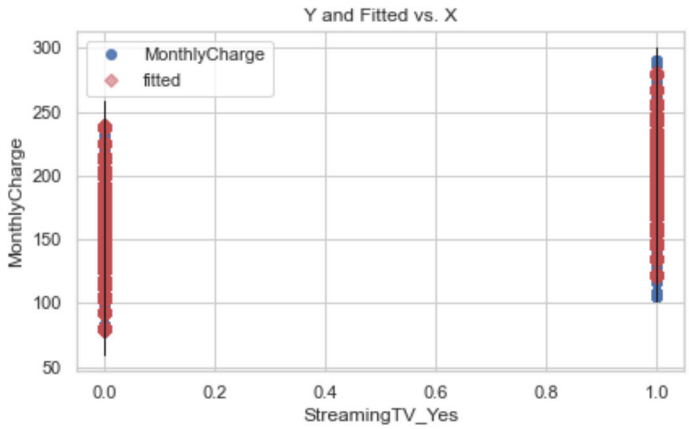
Regression Plots for TechSupport_Yes

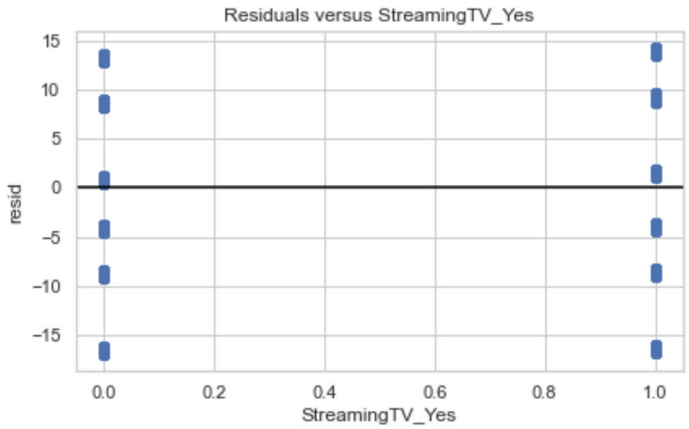


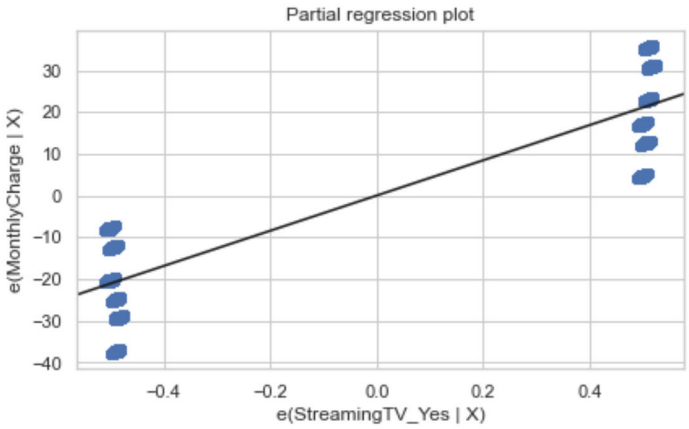


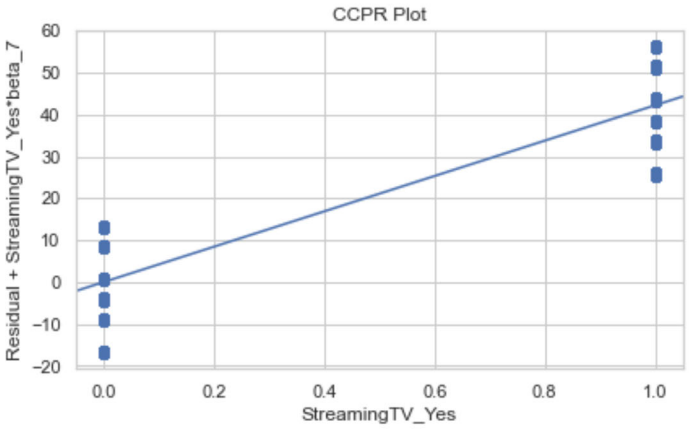


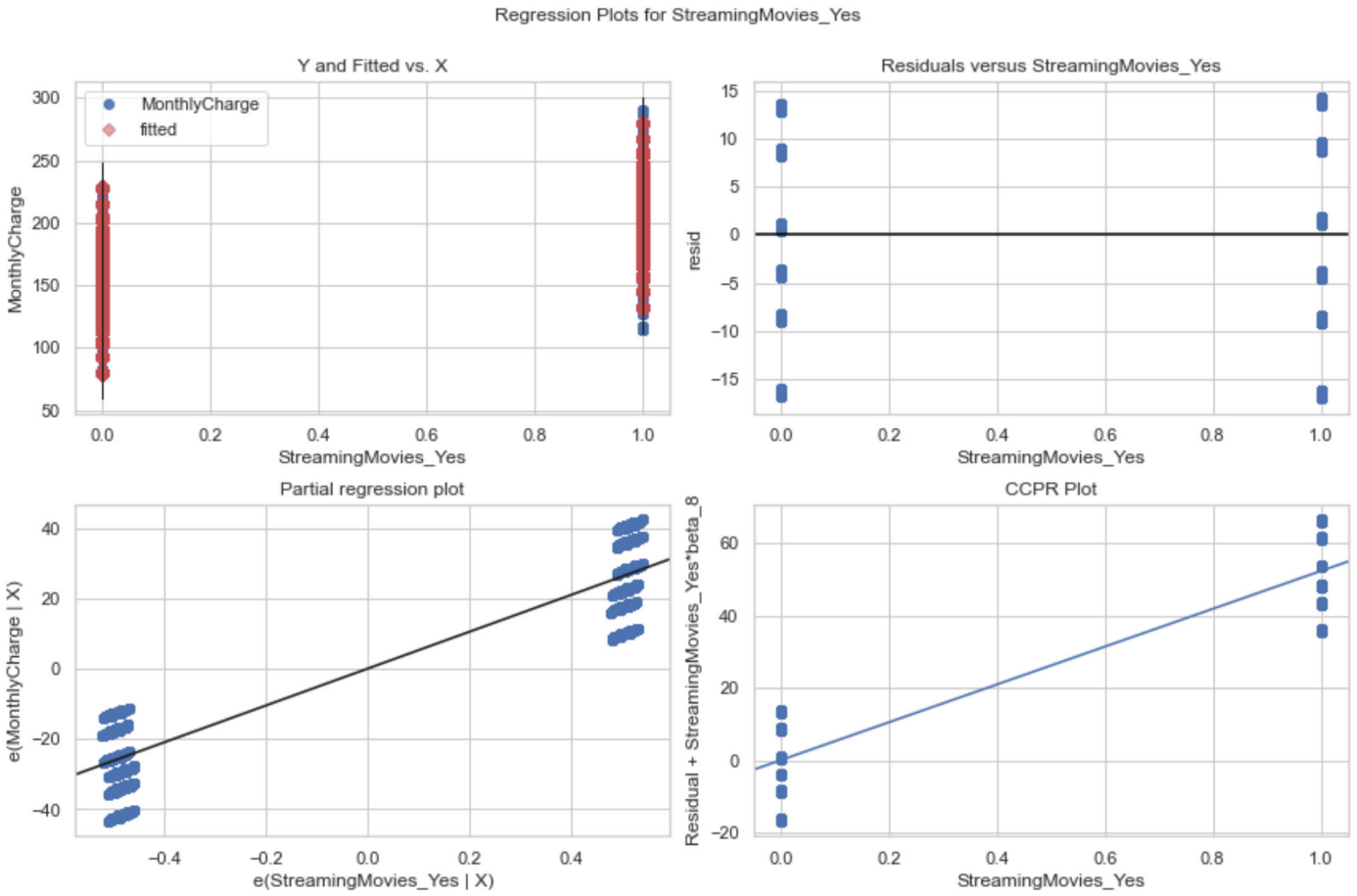
Regression Plots for StreamingTV_Yes











Part VI: Final Results

F. Final results.

F1. Final Results.

In [50]:
```
# equation of the regression line/plane
print('Adj. R-squared: {}'.format(model.summary2().tables[0][3][0]))
equation = model.summary2().tables[1]
print('Estimate [{}] as y = '.format(model.summary2().tables[0][1][1]))
for i in equation.itertuples():
    print('   {:+.3f} x ( {} ) '.format(i[1],i[0]))
```

```
Adj. R-squared: 0.946
Estimate [MonthlyCharge] as y =
   +78.843 x ( const )
   +24.742 x ( InternetService_Fiber Optic )
   +32.799 x ( Multiple_Yes )
   +2.801 x ( OnlineSecurity_Yes )
   +22.575 x ( OnlineBackup_Yes )
   +12.456 x ( DeviceProtection_Yes )
   +12.642 x ( TechSupport_Yes )
   +42.179 x ( StreamingTV_Yes )
   +52.339 x ( StreamingMovies_Yes )
```

The final model uses eight (8) predictor variables and has an R-squared value of 94.6% and a condition number of 5 which indicate that this is a pretty good model.

Because the final regression model is based on categorical data, yes and no values, then each of the cooefficients has the behaviour of adding a given value if yes, or adding zero (0) if no. For example, if the customer only has fiber optic service and nothing else, then you could acurately predict the monthly charge by adding the constant value of 78.84 to the cooefficient value of 24.74 which equals $103.58 in this case.

We can see that in each of the residual plots, the values are randomly distributed above and below the zero line. This is an indication of multivariate normality, which is to say that the residuals are normally distributed.

In [ ]: