US Medical Insurance Costs Project

Introduction and data structure

This project works to analyze trends in the US Medical Insurance Costs dataset. We will primarily investigate the effects of each variable on the cost of medical insurance.

The data contains 1338 observations on 7 variables: `age`, `sex`, `bmi`, `children`, `smoker`, `region`, and `charges`. Specifically:

- `age`, an integer variable, refers to the age of the patient in years.
- `sex`, a string variable, refers to the biological sex of the patient (male or female.)
- `bmi`, a float variable, refers to the Body Mass Index (BMI) of the patient, a common indicator of health.
- `children`, an integer variable, refers to the number of children the patient has.
- `smoker`, a string variable, refers to if the patient smokes or not (yes or no.)
- `region`, a string variable, refers to the region the patient lives in (southwest, southeast, northwest, or northeast.)
- `charges`, a float variable, refers to the cost of medical insurance, in dollars, for the patient.

Data integration

We will start by importing the data by reading in the `insurance.csv` file as lists.

In [ ]:
```
# import libraries
import csv
import statistics

#Create empty lists for the various attributes in insurance.csv
ages = []
sexes = []
bmis = []
num_children = []
smoker_statuses = []
regions = []
insurance_charges = []

# helper function to load csv data
def load_list_data(lst, csv_file, column_name):
    # open csv file
    with open(csv_file) as csv_info:
        # read the data from the csv file
        csv_dict = csv.DictReader(csv_info)
        # loop through the data in each row of the csv
        for row in csv_dict:
            # add the data from each row to a list
```

```
            lst.append(row[column_name])
        # return the list
        return lst

# look at the data in insurance_csv_dict
load_list_data(ages, "C:/Users/nicol/Downloads/python-portfolio-example-solution/py
load_list_data(sexes, "C:/Users/nicol/Downloads/python-portfolio-example-solution/p
load_list_data(bmis, "C:/Users/nicol/Downloads/python-portfolio-example-solution/py
load_list_data(num_children, "C:/Users/nicol/Downloads/python-portfolio-example-sol
load_list_data(smoker_statuses, "C:/Users/nicol/Downloads/python-portfolio-example-
load_list_data(regions, "C:/Users/nicol/Downloads/python-portfolio-example-solution
load_list_data(insurance_charges, "C:/Users/nicol/Downloads/python-portfolio-exampl
```

Out[ ]:
```
['16884.924',
 '1725.5523',
 '4449.462',
 '21984.47061',
 '3866.8552',
 '3756.6216',
 '8240.5896',
 '7281.5056',
 '6406.4107',
 '28923.13692',
 '2721.3208',
 '27808.7251',
 '1826.843',
 '11090.7178',
 '39611.7577',
 '1837.237',
 '10797.3362',
 '2395.17155',
 '10602.385',
 '36837.467',
 '13228.84695',
 '4149.736',
 '1137.011',
 '37701.8768',
 '6203.90175',
 '14001.1338',
 '14451.83515',
 '12268.63225',
 '2775.19215',
 '38711',
 '35585.576',
 '2198.18985',
 '4687.797',
 '13770.0979',
 '51194.55914',
 '1625.43375',
 '15612.19335',
 '2302.3',
 '39774.2763',
 '48173.361',
 '3046.062',
 '4949.7587',
 '6272.4772',
 '6313.759',
 '6079.6715',
 '20630.28351',
 '3393.35635',
 '3556.9223',
 '12629.8967',
 '38709.176',
 '2211.13075',
 '3579.8287',
 '23568.272',
 '37742.5757',
 '8059.6791',
 '47496.49445',
```

```
'13607.36875',
'34303.1672',
'23244.7902',
'5989.52365',
'8606.2174',
'4504.6624',
'30166.61817',
'4133.64165',
'14711.7438',
'1743.214',
'14235.072',
'6389.37785',
'5920.1041',
'17663.1442',
'16577.7795',
'6799.458',
'11741.726',
'11946.6259',
'7726.854',
'11356.6609',
'3947.4131',
'1532.4697',
'2755.02095',
'6571.02435',
'4441.21315',
'7935.29115',
'37165.1638',
'11033.6617',
'39836.519',
'21098.55405',
'43578.9394',
'11073.176',
'8026.6666',
'11082.5772',
'2026.9741',
'10942.13205',
'30184.9367',
'5729.0053',
'47291.055',
'3766.8838',
'12105.32',
'10226.2842',
'22412.6485',
'15820.699',
'6186.127',
'3645.0894',
'21344.8467',
'30942.1918',
'5003.853',
'17560.37975',
'2331.519',
'3877.30425',
'2867.1196',
'47055.5321',
'10825.2537',
'11881.358',
```

```
'4646.759',
'2404.7338',
'11488.31695',
'30259.99556',
'11381.3254',
'19107.7796',
'8601.3293',
'6686.4313',
'7740.337',
'1705.6245',
'2257.47525',
'39556.4945',
'10115.00885',
'3385.39915',
'17081.08',
'9634.538',
'32734.1863',
'6082.405',
'12815.44495',
'13616.3586',
'11163.568',
'1632.56445',
'2457.21115',
'2155.6815',
'1261.442',
'2045.68525',
'27322.73386',
'2166.732',
'27375.90478',
'3490.5491',
'18972.495',
'18157.876',
'20745.9891',
'5138.2567',
'40720.55105',
'9877.6077',
'10959.6947',
'1842.519',
'5125.2157',
'7789.635',
'6334.34355',
'19964.7463',
'7077.1894',
'6948.7008',
'21223.6758',
'15518.18025',
'36950.2567',
'19749.38338',
'21348.706',
'36149.4835',
'10450.552',
'5152.134',
'5028.1466',
'10407.08585',
'4830.63',
'6128.79745',
```

```
'2719.27975',
'4827.90495',
'13405.3903',
'8116.68',
'1694.7964',
'5246.047',
'2855.43755',
'48824.45',
'6455.86265',
'10436.096',
'8823.279',
'8538.28845',
'11735.87905',
'1631.8212',
'4005.4225',
'7419.4779',
'7731.4271',
'43753.33705',
'3981.9768',
'5325.651',
'6775.961',
'4922.9159',
'12557.6053',
'4883.866',
'2137.6536',
'12044.342',
'1137.4697',
'1639.5631',
'5649.715',
'8516.829',
'9644.2525',
'14901.5167',
'2130.6759',
'8871.1517',
'13012.20865',
'37133.8982',
'7147.105',
'4337.7352',
'11743.299',
'20984.0936',
'13880.949',
'6610.1097',
'1980.07',
'8162.71625',
'3537.703',
'5002.7827',
'8520.026',
'7371.772',
'10355.641',
'2483.736',
'3392.9768',
'25081.76784',
'5012.471',
'10564.8845',
'5253.524',
'34779.615',
```

```
'19515.5416',
'11987.1682',
'2689.4954',
'24227.33724',
'7358.17565',
'9225.2564',
'7443.64305',
'14001.2867',
'1727.785',
'12333.828',
'6710.1919',
'19444.2658',
'1615.7667',
'4463.2051',
'17352.6803',
'7152.6714',
'38511.6283',
'5354.07465',
'35160.13457',
'7196.867',
'29523.1656',
'24476.47851',
'12648.7034',
'1986.9334',
'1832.094',
'4040.55825',
'12829.4551',
'47305.305',
'44260.7499',
'4260.744',
'41097.16175',
'13047.33235',
'43921.1837',
'5400.9805',
'11520.09985',
'33750.2918',
'11837.16',
'17085.2676',
'24869.8368',
'36219.40545',
'20462.99766',
'46151.1245',
'17179.522',
'14590.63205',
'7441.053',
'9282.4806',
'1719.4363',
'42856.838',
'7265.7025',
'9617.66245',
'2523.1695',
'9715.841',
'2803.69785',
'2150.469',
'12928.7911',
'9855.1314',
```

```
'22331.5668',
'48549.17835',
'4237.12655',
'11879.10405',
'9625.92',
'7742.1098',
'9432.9253',
'14256.1928',
'47896.79135',
'25992.82104',
'3172.018',
'20277.80751',
'42112.2356',
'2156.7518',
'3906.127',
'1704.5681',
'16297.846',
'21978.6769',
'38746.3551',
'9249.4952',
'6746.7425',
'24873.3849',
'12265.5069',
'4349.462',
'12646.207',
'19442.3535',
'20177.67113',
'4151.0287',
'11944.59435',
'7749.1564',
'8444.474',
'1737.376',
'42124.5153',
'8124.4084',
'34838.873',
'9722.7695',
'8835.26495',
'10435.06525',
'7421.19455',
'4667.60765',
'4894.7533',
'24671.66334',
'35491.64',
'11566.30055',
'2866.091',
'6600.20595',
'3561.8889',
'42760.5022',
'47928.03',
'9144.565',
'48517.56315',
'24393.6224',
'13429.0354',
'11658.37915',
'19144.57652',
'13822.803',
```

```
'12142.5786',
'13937.6665',
'41919.097',
'8232.6388',
'18955.22017',
'13352.0998',
'13217.0945',
'13981.85035',
'10977.2063',
'6184.2994',
'4889.9995',
'8334.45755',
'5478.0368',
'1635.73365',
'11830.6072',
'8932.084',
'3554.203',
'12404.8791',
'14133.03775',
'24603.04837',
'8944.1151',
'9620.3307',
'1837.2819',
'1607.5101',
'10043.249',
'4751.07',
'13844.506',
'2597.779',
'3180.5101',
'9778.3472',
'13430.265',
'8017.06115',
'8116.26885',
'3481.868',
'13415.0381',
'12029.2867',
'7639.41745',
'36085.219',
'1391.5287',
'18033.9679',
'21659.9301',
'38126.2465',
'16455.70785',
'27000.98473',
'15006.57945',
'42303.69215',
'20781.48892',
'5846.9176',
'8302.53565',
'1261.859',
'11856.4115',
'30284.64294',
'3176.8159',
'4618.0799',
'10736.87075',
'2138.0707',
```

```
'8964.06055',
'9290.1395',
'9411.005',
'7526.70645',
'8522.003',
'16586.49771',
'14988.432',
'1631.6683',
'9264.797',
'8083.9198',
'14692.66935',
'10269.46',
'3260.199',
'11396.9002',
'4185.0979',
'8539.671',
'6652.5288',
'4074.4537',
'1621.3402',
'19594.80965',
'14455.64405',
'5080.096',
'2134.9015',
'7345.7266',
'9140.951',
'18608.262',
'14418.2804',
'28950.4692',
'46889.2612',
'46599.1084',
'39125.33225',
'2727.3951',
'8968.33',
'9788.8659',
'6555.07035',
'7323.734819',
'3167.45585',
'18804.7524',
'23082.95533',
'4906.40965',
'5969.723',
'12638.195',
'4243.59005',
'13919.8229',
'2254.7967',
'5926.846',
'12592.5345',
'2897.3235',
'4738.2682',
'37079.372',
'1149.3959',
'28287.89766',
'26109.32905',
'7345.084',
'12730.9996',
'11454.0215',
```

```
'5910.944',
'4762.329',
'7512.267',
'4032.2407',
'1969.614',
'1769.53165',
'4686.3887',
'21797.0004',
'11881.9696',
'11840.77505',
'10601.412',
'7682.67',
'10381.4787',
'22144.032',
'15230.32405',
'11165.41765',
'1632.03625',
'19521.9682',
'13224.693',
'12643.3778',
'23288.9284',
'2201.0971',
'2497.0383',
'2203.47185',
'1744.465',
'20878.78443',
'25382.297',
'28868.6639',
'35147.52848',
'2534.39375',
'1534.3045',
'1824.2854',
'15555.18875',
'9304.7019',
'1622.1885',
'9880.068',
'9563.029',
'4347.02335',
'12475.3513',
'1253.936',
'48885.13561',
'10461.9794',
'1748.774',
'24513.09126',
'2196.4732',
'12574.049',
'17942.106',
'1967.0227',
'4931.647',
'8027.968',
'8211.1002',
'13470.86',
'36197.699',
'6837.3687',
'22218.1149',
'32548.3405',
```

```
'5974.3847',
'6796.86325',
'2643.2685',
'3077.0955',
'3044.2133',
'11455.28',
'11763.0009',
'2498.4144',
'9361.3268',
'1256.299',
'21082.16',
'11362.755',
'27724.28875',
'8413.46305',
'5240.765',
'3857.75925',
'25656.57526',
'3994.1778',
'9866.30485',
'5397.6167',
'38245.59327',
'11482.63485',
'24059.68019',
'9861.025',
'8342.90875',
'1708.0014',
'48675.5177',
'14043.4767',
'12925.886',
'19214.70553',
'13831.1152',
'6067.12675',
'5972.378',
'8825.086',
'8233.0975',
'27346.04207',
'6196.448',
'3056.3881',
'13887.204',
'63770.42801',
'10231.4999',
'23807.2406',
'3268.84665',
'11538.421',
'3213.62205',
'45863.205',
'13390.559',
'3972.9247',
'12957.118',
'11187.6567',
'17878.90068',
'3847.674',
'8334.5896',
'3935.1799',
'39983.42595',
'1646.4297',
```

```
'9193.8385',
'10923.9332',
'2494.022',
'9058.7303',
'2801.2588',
'2128.43105',
'6373.55735',
'7256.7231',
'11552.904',
'45702.02235',
'3761.292',
'2219.4451',
'4753.6368',
'31620.00106',
'13224.05705',
'12222.8983',
'1664.9996',
'58571.07448',
'9724.53',
'3206.49135',
'12913.9924',
'1639.5631',
'6356.2707',
'17626.23951',
'1242.816',
'4779.6023',
'3861.20965',
'43943.8761',
'13635.6379',
'5976.8311',
'11842.442',
'8428.0693',
'2566.4707',
'15359.1045',
'5709.1644',
'8823.98575',
'7640.3092',
'5594.8455',
'7441.501',
'33471.97189',
'1633.0444',
'9174.13565',
'11070.535',
'16085.1275',
'17468.9839',
'9283.562',
'3558.62025',
'25678.77845',
'4435.0942',
'39241.442',
'8547.6913',
'6571.544',
'2207.69745',
'6753.038',
'1880.07',
'42969.8527',
```

```
'11658.11505',
'23306.547',
'34439.8559',
'10713.644',
'3659.346',
'40182.246',
'9182.17',
'34617.84065',
'12129.61415',
'3736.4647',
'6748.5912',
'11326.71487',
'11365.952',
'42983.4585',
'10085.846',
'1977.815',
'3366.6697',
'7173.35995',
'9391.346',
'14410.9321',
'2709.1119',
'24915.04626',
'20149.3229',
'12949.1554',
'6666.243',
'32787.45859',
'13143.86485',
'4466.6214',
'18806.14547',
'10141.1362',
'6123.5688',
'8252.2843',
'1712.227',
'12430.95335',
'9800.8882',
'10579.711',
'8280.6227',
'8527.532',
'12244.531',
'24667.419',
'3410.324',
'4058.71245',
'26392.26029',
'14394.39815',
'6435.6237',
'22192.43711',
'5148.5526',
'1136.3994',
'27037.9141',
'42560.4304',
'8703.456',
'40003.33225',
'45710.20785',
'6500.2359',
'4837.5823',
'3943.5954',
```

```
'4399.731',
'6185.3208',
'46200.9851',
'7222.78625',
'12485.8009',
'46130.5265',
'12363.547',
'10156.7832',
'2585.269',
'1242.26',
'40103.89',
'9863.4718',
'4766.022',
'11244.3769',
'7729.64575',
'5438.7491',
'26236.57997',
'34806.4677',
'2104.1134',
'8068.185',
'2362.22905',
'2352.96845',
'3577.999',
'3201.24515',
'29186.48236',
'40273.6455',
'10976.24575',
'3500.6123',
'2020.5523',
'9541.69555',
'9504.3103',
'5385.3379',
'8930.93455',
'5375.038',
'44400.4064',
'10264.4421',
'6113.23105',
'5469.0066',
'1727.54',
'10107.2206',
'8310.83915',
'1984.4533',
'2457.502',
'12146.971',
'9566.9909',
'13112.6048',
'10848.1343',
'12231.6136',
'9875.6804',
'11264.541',
'12979.358',
'1263.249',
'10106.13425',
'40932.4295',
'6664.68595',
'16657.71745',
```

```
'2217.6012',
'6781.3542',
'19361.9988',
'10065.413',
'4234.927',
'9447.25035',
'14007.222',
'9583.8933',
'40419.0191',
'3484.331',
'36189.1017',
'44585.45587',
'8604.48365',
'18246.4955',
'43254.41795',
'3757.8448',
'8827.2099',
'9910.35985',
'11737.84884',
'1627.28245',
'8556.907',
'3062.50825',
'19539.243',
'1906.35825',
'14210.53595',
'11833.7823',
'17128.42608',
'5031.26955',
'7985.815',
'23065.4207',
'5428.7277',
'36307.7983',
'3925.7582',
'2416.955',
'19040.876',
'3070.8087',
'9095.06825',
'11842.62375',
'8062.764',
'7050.642',
'14319.031',
'6933.24225',
'27941.28758',
'11150.78',
'12797.20962',
'17748.5062',
'7261.741',
'10560.4917',
'6986.697',
'7448.40395',
'5934.3798',
'9869.8102',
'18259.216',
'1146.7966',
'9386.1613',
'24520.264',
```

```
'4350.5144',
'6414.178',
'12741.16745',
'1917.3184',
'5209.57885',
'13457.9608',
'5662.225',
'1252.407',
'2731.9122',
'21195.818',
'7209.4918',
'18310.742',
'4266.1658',
'4719.52405',
'11848.141',
'17904.52705',
'7046.7222',
'14313.8463',
'2103.08',
'38792.6856',
'1815.8759',
'7731.85785',
'28476.73499',
'2136.88225',
'1131.5066',
'3309.7926',
'9414.92',
'6360.9936',
'11013.7119',
'4428.88785',
'5584.3057',
'1877.9294',
'2842.76075',
'3597.596',
'23401.30575',
'55135.40209',
'7445.918',
'2680.9493',
'1621.8827',
'8219.2039',
'12523.6048',
'16069.08475',
'43813.8661',
'20773.62775',
'39597.4072',
'6117.4945',
'13393.756',
'5266.3656',
'4719.73655',
'11743.9341',
'5377.4578',
'7160.3303',
'4402.233',
'11657.7189',
'6402.29135',
'12622.1795',
```

```
'1526.312',
'12323.936',
'36021.0112',
'27533.9129',
'10072.05505',
'45008.9555',
'9872.701',
'2438.0552',
'2974.126',
'10601.63225',
'37270.1512',
'14119.62',
'42111.6647',
'11729.6795',
'24106.91255',
'1875.344',
'40974.1649',
'15817.9857',
'18218.16139',
'10965.446',
'46113.511',
'7151.092',
'12269.68865',
'5458.04645',
'8782.469',
'6600.361',
'1141.4451',
'11576.13',
'13129.60345',
'4391.652',
'8457.818',
'3392.3652',
'5966.8874',
'6849.026',
'8891.1395',
'2690.1138',
'26140.3603',
'6653.7886',
'6282.235',
'6311.952',
'3443.064',
'2789.0574',
'2585.85065',
'46255.1125',
'4877.98105',
'19719.6947',
'27218.43725',
'5272.1758',
'1682.597',
'11945.1327',
'29330.98315',
'7243.8136',
'10422.91665',
'44202.6536',
'13555.0049',
'13063.883',
```

```
'19798.05455',
'2221.56445',
'1634.5734',
'2117.33885',
'8688.85885',
'48673.5588',
'4661.28635',
'8125.7845',
'12644.589',
'4564.19145',
'4846.92015',
'7633.7206',
'15170.069',
'17496.306',
'2639.0429',
'33732.6867',
'14382.70905',
'7626.993',
'5257.50795',
'2473.3341',
'21774.32215',
'35069.37452',
'13041.921',
'5245.2269',
'13451.122',
'13462.52',
'5488.262',
'4320.41085',
'6250.435',
'25333.33284',
'2913.569',
'12032.326',
'13470.8044',
'6289.7549',
'2927.0647',
'6238.298',
'10096.97',
'7348.142',
'4673.3922',
'12233.828',
'32108.66282',
'8965.79575',
'2304.0022',
'9487.6442',
'1121.8739',
'9549.5651',
'2217.46915',
'1628.4709',
'12982.8747',
'11674.13',
'7160.094',
'39047.285',
'6358.77645',
'19933.458',
'11534.87265',
'47462.894',
```

```
 '4527.18295',
 '38998.546',
 '20009.63365',
 '3875.7341',
 '41999.52',
 '12609.88702',
 '41034.2214',
 '28468.91901',
 '2730.10785',
 '3353.284',
 '14474.675',
 '9500.57305',
 '26467.09737',
 '4746.344',
 '23967.38305',
 '7518.02535',
 '3279.86855',
 '8596.8278',
 '10702.6424',
 '4992.3764',
 '2527.81865',
 '1759.338',
 '2322.6218',
 '16138.76205',
 '7804.1605',
 '2902.9065',
 '9704.66805',
 '4889.0368',
 '25517.11363',
 '4500.33925',
 '19199.944',
 '16796.41194',
 '4915.05985',
 '7624.63',
 '8410.04685',
 '28340.18885',
 '4518.82625',
 '14571.8908',
 '3378.91',
 '7144.86265',
 '10118.424',
 '5484.4673',
 '16420.49455',
 '7986.47525',
 '7418.522',
 '13887.9685',
 '6551.7501',
 '5267.81815',
 ...]
```

Exploratory data analysis

First, we will explore some overall trends of the data set by calculating some summary statistics of the costs in the data set.

In [ ]:
```
## average
insurance_charges = [float(x) for x in insurance_charges] #convert charges list fro
mean_costs = round(sum(insurance_charges) / len(insurance_charges), 2) #calculate m
print("The average cost for patients in the dataset is", mean_costs, "dollars.")

## standard deviation
std_dev_costs = round(statistics.stdev(insurance_charges),2)
print("The standard deviation of the patient's costs is", std_dev_costs, "dollars."

## minimum and maximum value
min_cost = min(insurance_charges)
max_cost = max(insurance_charges)
print("The cheapest cost in the dataset is", min_cost, "dollars, and the most expen
```

```
The average cost for patients in the dataset is 13270.42 dollars.
The standard deviation of the patient's costs is 12110.01 dollars.
The cheapest cost in the dataset is 1121.8739 dollars, and the most expensive cost i
n the data set is 63770.42801 dollars.
```

Now, we will explore each of the variables individually and their relationship with the cost variable.

Age

First, we will calculate some summary statistics of the ages of the patients in the dataset in order to gain a better picture of the data set.

In [ ]:
```
## average age
ages = [int(x) for x in ages] #convert ages list from string to integers
mean_age = round(sum(ages) / len(ages), 2) #calculate mean
print("The average age of the patients in the dataset is", mean_age, "years.")

## standard deviation
std_dev_age = round(statistics.stdev(ages),2)
print("The standard deviation of the patient's ages is", std_dev_age, "years.")

## minimum and maximum value
min_age = min(ages)
max_age = max(ages)
print("The youngest patient in the dataset is", min_age, "years old, and the oldest
```

```
The average age of the patients in the dataset is 39.21 years.
The standard deviation of the patient's ages is 14.05 years.
The youngest patient in the dataset is 18 years old, and the oldest patient in the d
ata set is 64 years old.
```

From the above analysis, we can see that the ages of the patients in the dataset are pretty widely distributed.

There may be a correlation between the age range of the patient and the average medical insurance cost. We will separate the patients into age categories and calculate the average cost for each age range. For our age ranges, we will use a standard age range distribution as follows:

- 18 to 24
- 25 to 34
- 35 to 44
- 45 to 54
- 55 to 64

In [ ]:
```
## average cost by age range

# iterate through each age, categorize by age range, calculate mean cost
tot_18 = 0
tot_25 = 0
tot_35 = 0
tot_45 = 0
tot_55 = 0

num_18 = 0
num_25 = 0
num_35 = 0
num_45 = 0
num_55 = 0

i = -1

for age in ages:
    i += 1
    cost = insurance_charges[i]
    if age < 25:
        tot_18 += cost
        num_18 += 1
    elif age < 35:
        tot_25 += cost
        num_25 += 1
    elif age < 45:
        tot_35 += cost
        num_35 += 1
    elif age < 55:
        tot_45 += cost
        num_45 += 1
    elif age < 65:
        tot_55 += cost
        num_55 += 1

# calculate mean, unless there are no observations in that age range
if num_18 != 0:
    avg_18 = round(tot_18 / num_18 , 2)
else:
    avg_18 = 0
if num_25 != 0:
    avg_25 = round(tot_25 / num_25 , 2)
else:
    avg_25 = 0
if num_35 != 0:
    avg_35 = round(tot_35 / num_35 , 2)
else:
```

```
    avg_35 = 0
if num_45 != 0:
    avg_45 = round(tot_45 / num_45 , 2)
else:
    avg_45 = 0
if num_55 != 0:
    avg_55 = round(tot_55 / num_55 , 2)
else:
    avg_55 = 0

print("The average cost for the", "18 to 24", "age range is", avg_18, "dollars, wit
print("The average cost for the", "25 to 34", "age range is", avg_25, "dollars, wit
print("The average cost for the", "35 to 44", "age range is", avg_35, "dollars, wit
print("The average cost for the", "45 to 54", "age range is", avg_45, "dollars, wit
print("The average cost for the", "55 to 64", "age range is", avg_55, "dollars, wit
```

```
The average cost for the 18 to 24 age range is 9011.34 dollars, with 278 total obser
vations.
The average cost for the 25 to 34 age range is 10352.39 dollars, with 271 total obse
rvations.
The average cost for the 35 to 44 age range is 13134.17 dollars, with 260 total obse
rvations.
The average cost for the 45 to 54 age range is 15853.93 dollars, with 287 total obse
rvations.
The average cost for the 55 to 64 age range is 18513.28 dollars, with 242 total obse
rvations.
```

From the above analysis, we can see that insurance cost seems to increase with age. We can also see that the sample size in each of the age ranges is approximately equal. To further investigate this correlation, we could run an ANOVA test for difference in group means or use regression analysis to find the correlation of the two variables.

Sex

First, we will look at the distribution of the sexes of the patients in the dataset in order to gain a better picture of the data set.

In [ ]:
```
## number of ppl in each sex

num_f = 0
num_m = 0

for sex in sexes:
    if sex == "female":
        num_f += 1
    if sex == "male":
        num_m += 1

print("There are", num_f, "females and", num_m, "males in the dataset.")
```

```
There are 662 females and 676 males in the dataset.
```

The number of females and males in the dataset is approximately equal. We can now look at the difference in costs between the sexes.

In [ ]:
```
## average cost by sex

tot_f = 0
tot_m = 0

i = -1

for sex in sexes:
    i += 1
    cost = insurance_charges[i]
    if sex == "female":
        tot_f += cost
    if sex == "male":
        tot_m += cost

avg_f = round(tot_f / num_f, 2)
avg_m = round(tot_m / num_m, 2)

print("The average cost for females is", avg_f, "dollars.")
print("The average cost for males is", avg_m, "dollars.")
```

```
The average cost for females is 12569.58 dollars.
The average cost for males is 13956.75 dollars.
```

It seems that the average cost of medical insurance for males is higher than the average cost of medical insurance for females. To further investigate this difference, we could run an ANOVA test for the difference between means.

BMI

First, summary statistics.

In [ ]:
```
## average bmi
bmis = [float(x) for x in bmis] #convert bmi list from string to float
mean_bmi = round(sum(bmis) / len(bmis), 2) #calculate mean
print("The average BMI of the patients in the dataset is", mean_bmi)

## standard deviation
std_dev_bmi = round(statistics.stdev(bmis),2)
print("The standard deviation of the patient's BMI is", std_dev_bmi)

## minimum and maximum value
min_bmi = min(bmis)
max_bmi = max(bmis)
print("The smallest BMI in the dataset is", min_bmi, "and the largest in the data s
```

```
The average BMI of the patients in the dataset is 30.66
The standard deviation of the patient's BMI is 6.1
The smallest BMI in the dataset is 15.96 and the largest in the data set is 53.13
```

From the above analysis, we can see that there is a large spread of BMIs in the data set. Unfortunately, it seems that on average, the patients BMI indicates that they are obese (over 30.) America!

Now, we will investigate the difference in mean costs between the different BMI ranges. The BMI ranges we will use will be as follows:

- Underweight: 18.5 or below
- Healthy weight: 18.5-25 (non-inclusive)
- Overweight: 25(inclusive)-30(non-inclusive)
- Obese: 30 or above

In [ ]:
```
## insurance cost by bmi range

# iterate through bmi list, categorize by range, calculate mean of each range

# init vars
tot_uw = 0
tot_hw = 0
tot_ow = 0
tot_ob = 0

num_uw = 0
num_hw = 0
num_ow = 0
num_ob = 0

i = -1

# iteration (categorization)
for bmi in bmis:
    i += 1
    cost = insurance_charges[i]
    if bmi <= 18.5:
        tot_uw += cost
        num_uw += 1
    elif bmi < 25:
        tot_hw += cost
        num_hw += 1
    elif bmi < 30:
        tot_ow += cost
        num_ow += 1
    elif bmi >= 30:
        tot_ob += cost
        num_ob += 1


# calculate means
avg_uw = round(tot_uw / num_uw, 2)
avg_hw = round(tot_hw / num_hw, 2)
avg_ow = round(tot_ow / num_ow, 2)
avg_ob = round(tot_ob / num_ob, 2)

print("The average medical insurance cost for underweight patients is", avg_uw, "do
print("The average medical insurance cost for healthy weight patients is", avg_hw,
print("The average medical insurance cost for overweight patients is", avg_ow, "dol
print("The average medical insurance cost for obese patients is", avg_ob, "dollars,
```

```
The average medical insurance cost for underweight patients is 8657.62 dollars, with
21 observations.
The average medical insurance cost for healthy weight patients is 10434.53 dollars,
with 224 observations.
The average medical insurance cost for overweight patients is 10987.51 dollars, with
386 observations.
The average medical insurance cost for obese patients is 15552.34 dollars, with 707
observations.
```

It seems that the average medical insurance cost increases with BMI, but it should be noted that there is a large difference in sample size for each of the groups. The obese patients group makes up over half the data set. Because of the large disparity in sample sizes, it may be worthwhile to categorize based on a different scale.

For further analysis, we can test for correlation between increase in BMI and increase in medical insurance cost using regression analysis. In this case, we won't have to deal with the differing sample sizes.

Children

First, summary statistics.

In [ ]:
```
## average children
num_children = [int(x) for x in num_children] #convert bmi list from string to floa
mean_children = round(sum(num_children) / len(num_children), 2) #calculate mean
print("The average number of children of the patients in the dataset is", mean_chil

## standard deviation
std_dev_children = round(statistics.stdev(num_children),2)
print("The standard deviation of the patient's number of children is", std_dev_chil

## minimum and maximum value
min_children = min(num_children)
max_children = max(num_children)
print("The smallest number of children in the dataset is", min_children, "children
```

```
The average number of children of the patients in the dataset is 1.09 children.
The standard deviation of the patient's number of children is 1.21 children.
The smallest number of children in the dataset is 0 children and the largest number
of children in the data set is 5 children.
```

Since the smallest number of children in the data set is 0, and the largest number is 5, we will categorize based on these integer values and calculate the group means.

In [ ]:
```
tot_0c = 0
tot_1c = 0
tot_2c = 0
tot_3c = 0
tot_4c = 0
tot_5c = 0

num_0c = 0
num_1c = 0
```

```
num_2c = 0
num_3c = 0
num_4c = 0
num_5c = 0

i = -1

for num in num_children:
    i += 1
    cost = insurance_charges[i]
    if num == 0:
        tot_0c += cost
        num_0c += 1
    if num == 1:
        tot_1c += cost
        num_1c += 1
    if num == 2:
        tot_2c += cost
        num_2c += 1
    if num == 3:
        tot_3c += cost
        num_3c += 1
    if num == 4:
        tot_4c += cost
        num_4c += 1
    if num == 5:
        tot_5c += cost
        num_5c += 1

avg_0c = round(tot_0c / num_0c, 2)
avg_1c = round(tot_1c / num_1c, 2)
avg_2c = round(tot_2c / num_2c, 2)
avg_3c = round(tot_3c / num_3c, 2)
avg_4c = round(tot_4c / num_4c, 2)
avg_5c = round(tot_5c / num_5c, 2)

print("The average insurance cost for patients with 0 children is", avg_0c, "dollar
print("The average insurance cost for patients with 1 child is", avg_1c, "dollars,
print("The average insurance cost for patients with 2 children is", avg_2c, "dollar
print("The average insurance cost for patients with 3 children is", avg_3c, "dollar
print("The average insurance cost for patients with 4 children is", avg_4c, "dollar
print("The average insurance cost for patients with 5 children is", avg_5c, "dollar
```

```
The average insurance cost for patients with 0 children is 12365.98 dollars, with 57
4 observations.
The average insurance cost for patients with 1 child is 12731.17 dollars, with 324 o
bservations.
The average insurance cost for patients with 2 children is 15073.56 dollars, with 24
0 observations.
The average insurance cost for patients with 3 children is 15355.32 dollars, with 15
7 observations.
The average insurance cost for patients with 4 children is 13850.66 dollars, with 25
observations.
The average insurance cost for patients with 5 children is 8786.04 dollars, with 18
observations.
```

It seems like the relationship between number of children and insurance cost is non linear, with the highest insurance costs being in the group with 2 and 3 children and lower insurance costs with 0, 1, 4, or 5 children. Further analysis is needed to investigate this relationship. In addition, it should be noted that there are significantly less observations for patients with 4 and 5 children, and significantly more observations for patients with 0 children.

Smoking status

First, summary statistics.

In [ ]:
```
## number of ppl in each smoking status

num_y = 0
num_n = 0

for patient in smoker_statuses:
    if patient == "yes":
        num_y += 1
    if patient == "no":
        num_n += 1

print("There are", num_y, "smokers and", num_n, "non-smokers in the dataset.")
```

```
There are 274 smokers and 1064 non-smokers in the dataset.
```

The number of smokers and non-smokers in the dataset is unequal. However, we will still calculate the group means of each group.

In [ ]:
```
tot_y = 0
tot_n = 0

i = -1

for patient in smoker_statuses:
    i += 1
    cost = insurance_charges[i]
    if patient == "yes":
        tot_y += cost
    if patient == "no":
        tot_n += cost

avg_y = round(tot_y / num_y, 2)
avg_n = round(tot_n / num_n, 2)

print("The average insurance cost for smokers is", avg_y, "dollars, and", avg_n, "d
```

```
The average insurance cost for smokers is 32050.23 dollars, and 8434.27 dollars for
non-smokers.
```

It seems like the average insurance cost is less for non-smokers. Further analysis is needed, we could run an ANOVA test for the difference between means.

Region

First, summary statistics.

In [ ]:
```
## number of ppl in each smoking status

num_sw = 0
num_se = 0
num_nw = 0
num_ne = 0

for region in regions:
    if region == "southwest":
        num_sw += 1
    if region == "southeast":
        num_se += 1
    if region == "northwest":
        num_nw += 1
    if region == "northeast":
        num_ne += 1

print("There are", num_sw, "patients from the Southwest,", num_se, "patients in the
```

```
There are 325 patients from the Southwest, 364 patients in the Southeast, 325 patien
ts in the Northwest, and 324 patients in the Northeast.
```

It seems like the number of patients from each region is approximately equal. We will now calculate the group means of each region.

In [ ]:
```
## number of ppl in each smoking status

tot_sw = 0
tot_se = 0
tot_nw = 0
tot_ne = 0

i = -1

for region in regions:
    i += 1
    cost = insurance_charges[i]
    if region == "southwest":
        tot_sw += cost
    if region == "southeast":
        tot_se += cost
    if region == "northwest":
        tot_nw += cost
    if region == "northeast":
        tot_ne += cost

avg_sw = round(tot_sw / num_sw, 2)
avg_se = round(tot_se / num_se, 2)
avg_nw = round(tot_nw / num_nw, 2)
avg_ne = round(tot_ne / num_ne, 2)
```

```
print("The average insurance cost for patients in the Southwest is", avg_sw, "dolla
print("The average insurance cost for patients in the Southeast is", avg_se, "dolla
print("The average insurance cost for patients in the Northwest is", avg_nw, "dolla
print("The average insurance cost for patients in the Northeast is", avg_ne, "dolla
```

```
The average insurance cost for patients in the Southwest is 12346.94 dollars.
The average insurance cost for patients in the Southeast is 14735.41 dollars.
The average insurance cost for patients in the Northwest is 12417.58 dollars.
The average insurance cost for patients in the Northeast is 13406.38 dollars.
```

It seems like the insurance cost for patients in the east is slightly higher than those in the west. Further analysis is needed; we could run an ANOVA test for difference between group means.

Conclusion

Overall, there are many interesting trends to be found in the medical insurance data set. Further analysis can be done through statistical testing such as ANOVA or regression analysis.