In [ ]:
```
"""
Name - Jordan Helton
Student ID - 801012514
Homework 0
Github Repo - https://github.com/jhelto12/ECGR_Homework_0

"""
```

In [ ]:
```
"""

Problem 1
    Part 1 - See Below
    Part 2 - See Below
    Part 3 - [Column] had the lowest loss or cost because it had the highest slope
    Part 4 - A higher learning rate means a higher slope of the graph and a lower cost

"""
```

In [80]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [81]:
```
dataset = pd.read_csv('https://raw.githubusercontent.com/jhelto12/ECGR_Homework_0/mair
dataset.head()
M = len(dataset)
```

In [82]:
```
x1Column = dataset.values[:, 0]
x2Column = dataset.values[:, 1]
x3Column = dataset.values[:, 2]
yColumn = dataset.values[:, 3]
exampleCnt = len(yColumn)

x1 = x1Column
x2 = x2Column
x3 = x3Column

print('X1 = ', x1Column[: 5])
print('X2 = ', x2Column[: 5])
print('X3 = ', x3Column[: 5])
print('Y = ', yColumn[: 5])
print('Number of Training Examples = ', exampleCnt)
```

```
X1 =  [0.         0.04040404 0.08080808 0.12121212 0.16161616]
X2 =  [3.44       0.1349495  0.82989899 1.52484848 2.21979798]
X3 =  [0.44       0.88848485 1.3369697  1.78545454 2.23393939]
Y =  [4.38754501 2.6796499  2.96848981 3.25406475 3.53637472]
Number of Training Examples =  100
```

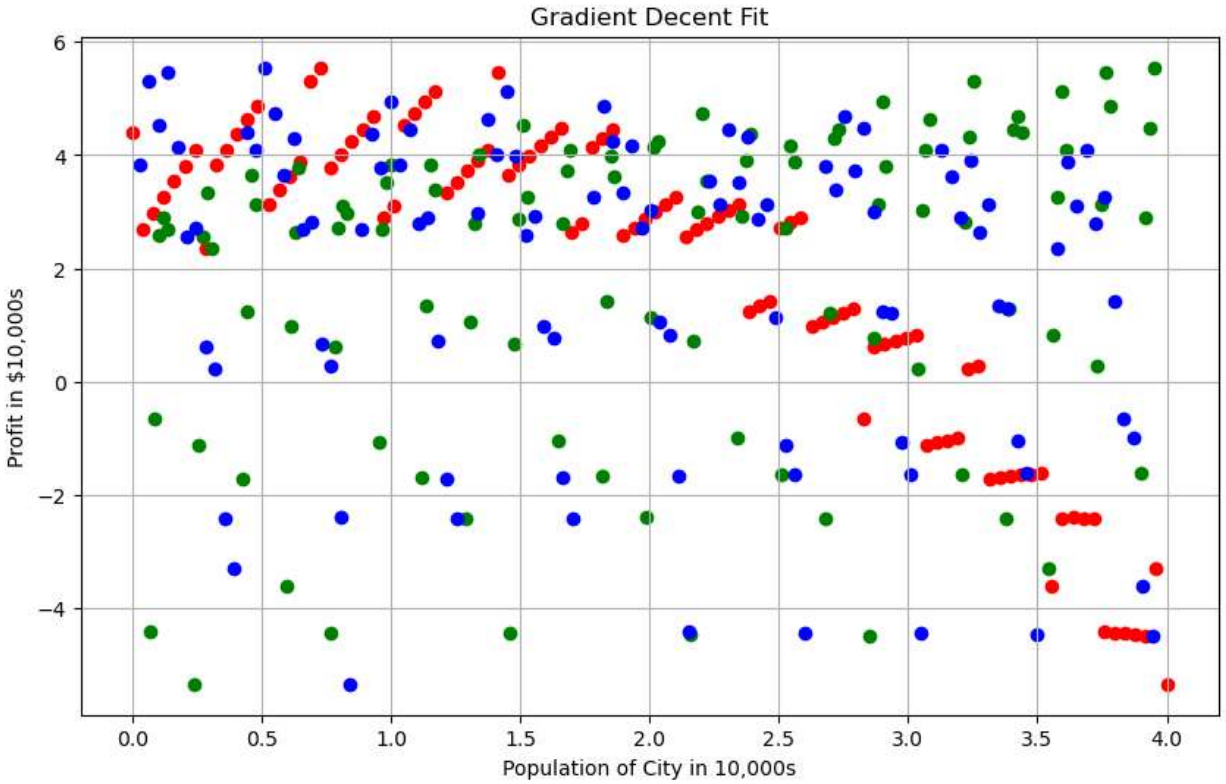
In [83]:
```
#function for calculating the Cost
def computeCost(inColumn, outColumn, theta):

    prediction = inColumn.dot(theta)
    error = np.subtract(prediction, outColumn)
    squareError = np.square(error)

    J = 1/(2*exampleCount) * np.sum(squareError)
```

```
    return J
```

In [84]:
```
#function for calculating Gradient Decent
def gradientDecent(inColumn, outColumn, theta, alpha, iterations):
    costHist = np.zeros(iterations)

    for i in range(iterations):
        prediction = inColumn.dot(theta)
        error = np.subtract(prediction, outColumn)
        deltaSum = (alpha/exampleCount) * inColumn.transpose().dot(error)
        theta = theta - deltaSum
        costHist[i] = computeCost(inColumn, outColumn, theta)

    return theta, costHist
```

In [106…
```
#function for setting up the column graph displays
def graphDisplay(inColumn, outColumn, inputColor):
    plt.scatter(inColumn, outColumn, color = inputColor)
    plt.rcParams["figure.figsize"] = (10,6)
    plt.grid()
    plt.xlabel('Population of City in 10,000s')
    plt.ylabel('Profit in $10,000s')
    plt.title('Gradient Decent Fit')
```

In [107…
```
#Column Graphs
graphDisplay(x1Column, yColumn, 'red')
graphDisplay(x2Column, yColumn, 'green')
graphDisplay(x3Column, yColumn, 'blue')
```

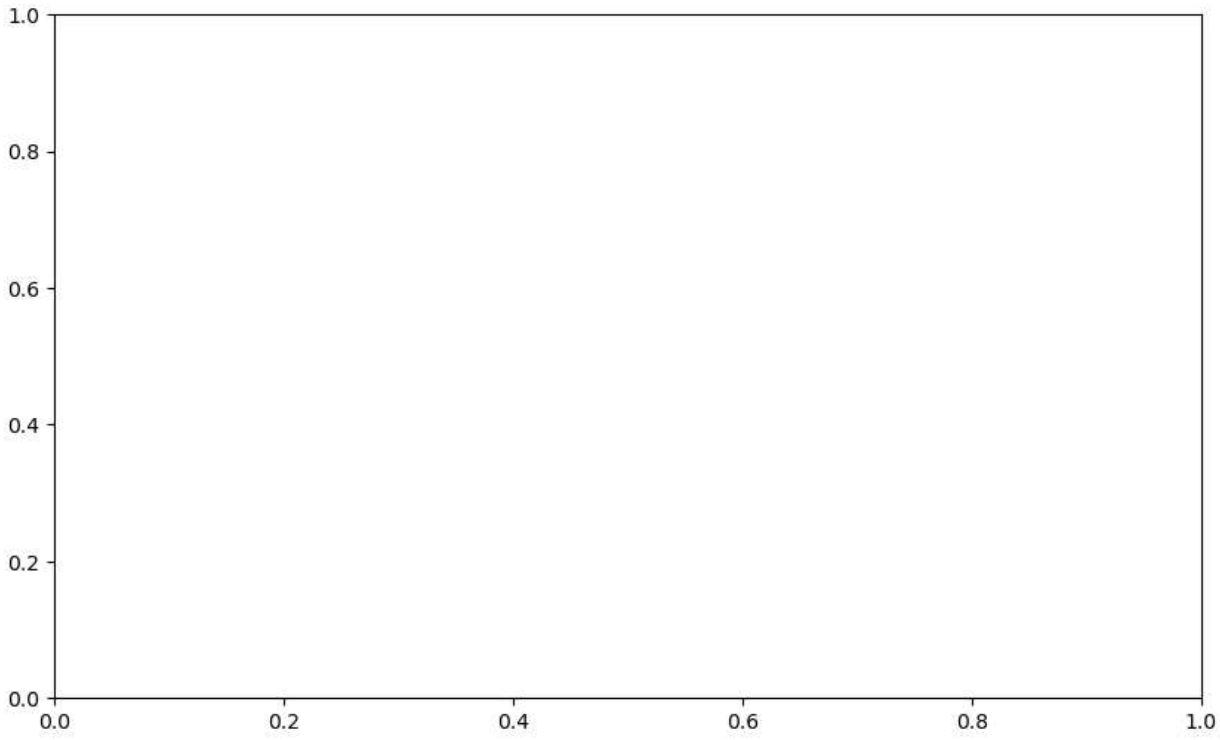


In [105…
```
Col1 = np.ones((exampleCnt, 1))
Col2 = x1.reshape(exampleCnt, 1)
x2 = np.hstack((Col1, Col2))
```

```
alphaVal = 0.001 #0.01
iterationCnt = 1500
thetaVal = np.zeros(2)
costVal = computeCost(x2, yColumn, thetaVal)
thetaVal, costHist = gradientDecent(x2, yColumn, thetaVal, alphaVal, iterationCnt)

graphDisplay(x1Column, yColumn, thetaVal)
plt.plot(x1Column, x2.dot(thetaVal), color = 'red', label = 'X2 Linear Regression')
plt.title('X1 Scatter Plot')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
D:\anaconda\lib\site-packages\matplotlib\axes\_axes.py in _parse_scatter_color_args
(c, edgecolors, kwargs, xsize, get_next_color_func)
   4152             try:
-> 4153                 mcolors.to_rgba_array(kwcolor)
   4154             except ValueError as err:

D:\anaconda\lib\site-packages\matplotlib\colors.py in to_rgba_array(c, alpha)
    376     else:
--> 377         rgba = np.array([to_rgba(cc) for cc in c])
    378

D:\anaconda\lib\site-packages\matplotlib\colors.py in <listcomp>(.0)
    376     else:
--> 377         rgba = np.array([to_rgba(cc) for cc in c])
    378

D:\anaconda\lib\site-packages\matplotlib\colors.py in to_rgba(c, alpha)
    186     if rgba is None:  # Suppress exception chaining of cache lookup failure.
--> 187         rgba = _to_rgba_no_colorcycle(c, alpha)
    188         try:

D:\anaconda\lib\site-packages\matplotlib\colors.py in _to_rgba_no_colorcycle(c, alph
a)
    268     if not np.iterable(c):
--> 269         raise ValueError(f"Invalid RGBA argument: {orig_c!r}")
    270     if len(c) not in [3, 4]:

ValueError: Invalid RGBA argument: 1.7416775291008961

The above exception was the direct cause of the following exception:

ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_15528\3599340980.py in <module>
      8 thetaVal, costHist = gradientDecent(x2, yColumn, thetaVal, alphaVal, iteratio
nCnt)
      9
---> 10 graphDisplay(x1Column, yColumn, thetaVal)
     11 plt.plot(x1Column, x2.dot(thetaVal), color = 'red', label = 'X2 Linear Regres
sion')
     12 plt.title('X1 Scatter Plot')

~\AppData\Local\Temp\ipykernel_15528\3572213113.py in graphDisplay(inColumn, outColum
n, inputColor)
      1 #function for setting up the column graph displays
      2 def graphDisplay(inColumn, outColumn, inputColor):
----> 3     plt.scatter(inColumn, outColumn, color = inputColor)
      4     plt.rcParams["figure.figsize"] = (10,6)
      5     plt.grid()

D:\anaconda\lib\site-packages\matplotlib\pyplot.py in scatter(x, y, s, c, marker, cma
p, norm, vmin, vmax, alpha, linewidths, edgecolors, plotnonfinite, data, **kwargs)
   2817         vmin=None, vmax=None, alpha=None, linewidths=None, *,
   2818         edgecolors=None, plotnonfinite=False, data=None, **kwargs):
-> 2819     __ret = gca().scatter(
   2820         x, y, s=s, c=c, marker=marker, cmap=cmap, norm=norm,
   2821         vmin=vmin, vmax=vmax, alpha=alpha, linewidths=linewidths,

D:\anaconda\lib\site-packages\matplotlib\__init__.py in inner(ax, data, *args, **kwar
```

```
gs)
   1410     def inner(ax, *args, data=None, **kwargs):
   1411         if data is None:
-> 1412             return func(ax, *map(sanitize_sequence, args), **kwargs)
   1413
   1414         bound = new_sig.bind(ax, *args, **kwargs)

D:\anaconda\lib\site-packages\matplotlib\axes\_axes.py in scatter(self, x, y, s, c, m
arker, cmap, norm, vmin, vmax, alpha, linewidths, edgecolors, plotnonfinite, **kwarg
s)
   4378             orig_edgecolor = kwargs.get('edgecolor', None)
   4379         c, colors, edgecolors = \
-> 4380             self._parse_scatter_color_args(
   4381                 c, edgecolors, kwargs, x.size,
   4382                 get_next_color_func=self._get_patches_for_fill.get_next_colo
r)

D:\anaconda\lib\site-packages\matplotlib\axes\_axes.py in _parse_scatter_color_args
(c, edgecolors, kwargs, xsize, get_next_color_func)
   4153                 mcolors.to_rgba_array(kwcolor)
   4154             except ValueError as err:
-> 4155                 raise ValueError(
   4156                     "'color' kwarg must be a color or sequence of color "
   4157                     "specs.  For a sequence of values to be color-mapped, use
"

ValueError: 'color' kwarg must be a color or sequence of color specs.  For a sequence
of values to be color-mapped, use the 'c' argument instead.
```

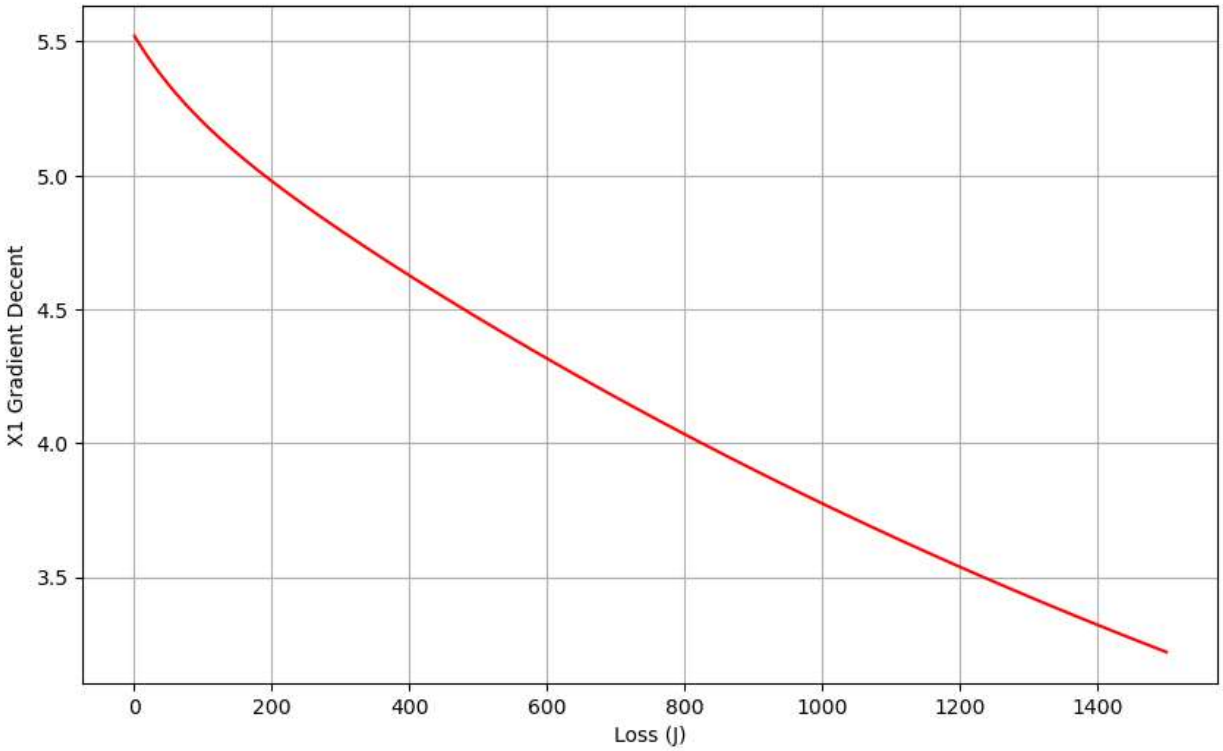


In [102…

```
plt.plot(range(1, iterationCnt+1), costHist, color = 'red')
plt.rcParams["figure.figsize"] = (10,6)
plt.grid()
plt.xlabel('Loss (J)')
plt.ylabel('X1 Gradient Decent')
```

Out[102]:

```
Text(0, 0.5, 'X1 Gradient Decent')
```


5.5
5.0
4.5
4.0
3.5
X1 Gradient Decent
0
200
400
600
800
1000
1200
1400
Loss (J)

In [ ]: