final\_project

Student 1 & Student 2

2024-12-01

# 0. Contribution Statement

## Student 1

Student 1 mostly worked on questions …

## Student 2

Student 2 mostly worked on questions …

# Introduction

Cities provide a glimpse into the local population of an area and are reflective of the culture and lifestyles in the region in which they are located. However, not all cities are created equal, as different socieoeconomic factors within the population and the surrounding region impact the quality of life of residents.

### Data

Our primary dataset consists of a collection of various studies done by the CDC in 2018 that detail various characteristics of a particular city and its demographics, ranging from the percentage of people that are unemployed to the percentage of civilian noninstitutionalized population with a disability. There are 72836 entries.

We include a secondary dataset in our **Advanced Analysis** section of our report that consists of the climate action surveys of major international corporations and if they responded to the survey or not. There are two surveys collected (Water and Climate Change) between 2018 and 2020, inclusive.

### Objective

Our objective is to analyze and document any causal relationships between features in the dataset, with an emphasis on per capita income (EP\_PCI) and unemployment rate (EP\_UNEMP).

# Basic Analysis

## **Question 1: SIMPLE REGRESSION** - Fit a regression line of the data predicting a city’s EP\_PCI (estimated per capita income)\* based on the estimated proportion of unemployment EP\_UNEMP. How well does the regression line fit this relationship?

### Methods

**Data Cleaning & Preparation**

| 0% | 25% | 50% | 75% | 100% |
| --- | --- | --- | --- | --- |
| -999 | 3.3 | 5.2 | 8 | 100 |

## [1] 546

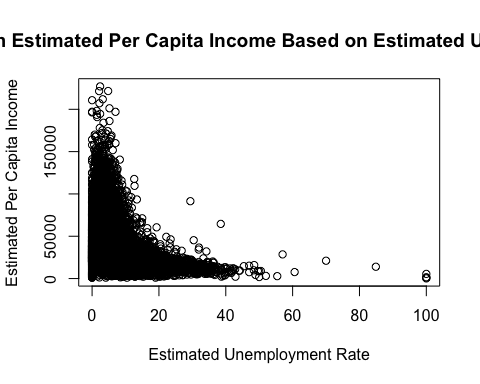
| 0% | 25% | 50% | 75% | 100% |
| --- | --- | --- | --- | --- |
| -999 | 21571 | 28460 | 38120 | 227064 |

## [1] 481

We want to create a regression model for EP\_PCI, the estimated per capita income, based on the estimated proportion of unemployment, EP\_UNEMP. We are predicting EP\_PCI because it describes the *rate* of unemployment and is normalized against population, unlike E\_PCI, which describes the estimate *count* the unemployed.

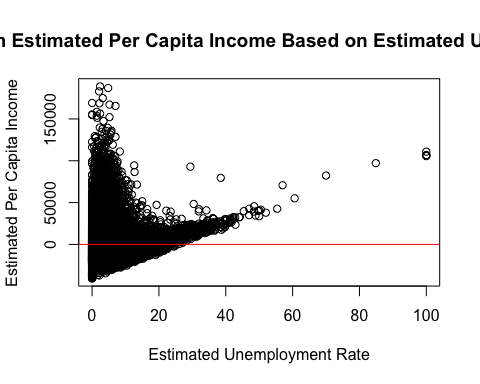
We will check the conditions for fitting a simple linear regression: linearity, homoscedasticity, independence, and normality.

1. **Linearity** - Is there a linear relationship between our explanatory variable, EP\_UNEMP and our response variable, EP\_PCI?



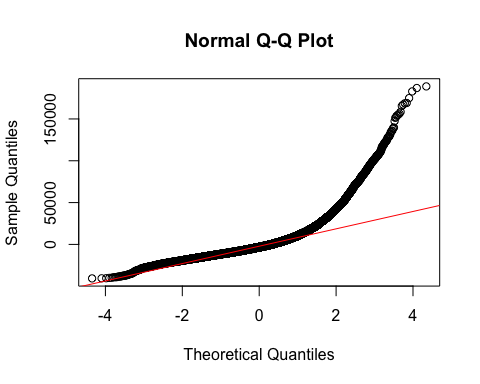
From our scatterplot, there appears to be a nonlinear negative trend between unemployment rate and estimated per capita income. As the estimated unemployment rate increases, the estimated per capita income exponentially decreases.

1. **Homoscedasticity**: Would the variance for the estimated unemployment rate be constant if we were to use a linear model?

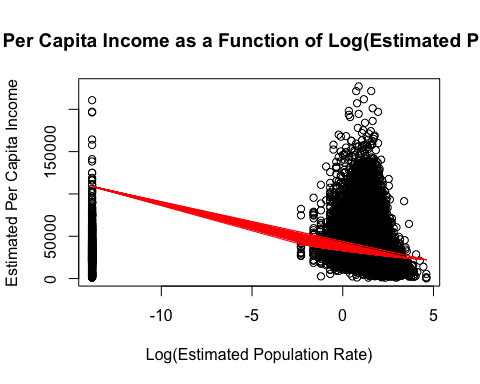


The residuals do *not* have constant variance and so, the data is not homoscedastic.

1. **Independence**: Because the residuals exhibit a clear pattern, independence may be violated.
2. **Normality**: A QQ plot of our residuals shows our data deviates from the y = x line. Its slight curve upward suggests the distribution of our residuals are right-skewed.



### Analysis

To transform the data, we will take the log of EP\_UNEMP and EP\_PCI because log transformations are often used to stabilize variance and help make data more suitable for regression. 

With our log-transformed model, we have an MSE of about 0.201, which represents the averages squared difference between the actual values of the estimated per capita income and the predicted values. This number is pretty low, which indicates our model performed well. However, the MSE of our original model is 301,671,723, which means our model performed very poorly without the log transformation.

### Conclusion

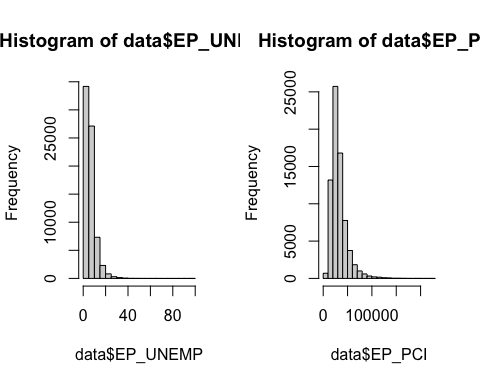
Based on these findings, the relationship between the estimated unemployment rate (EP\_UNEMP) and the estimated per capita income (EP\_PCI) do not meet the assumptions needed to fit a simple linear regression model without a transformation:

1. **Linearity:** The scatterplot shows a nonlinear, exponential relationship between EP\_UNEMP and EP\_PCI, which suggests a simple linear model is not the most effective.
2. **Homoscedasticity:** The residuals from the untransformed model show clear patterns and non-constant variance, which reaffirms the idea that our model does not perform well without a log transformation.
3. **Independence:** Similar to the point addressed in #2, the residual plots show clear patterns, which indicate a basic linear model is not suitable for this distribution.
4. **Normality**: The QQ Plot demonstrates a slight right skew, which means that the residuals from a simple linear model are not normal, and so a transformation is necessary.

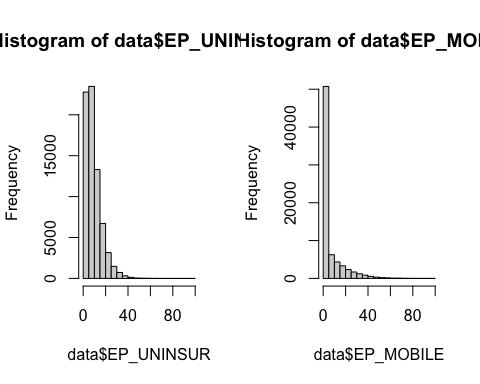
## **Question 2: Distribution Analysis** - How do the distributions of EP\_PCI and EP\_UNEMP compare to one another and among other features?

### Methods

To see the distributions of the EP\_UNEMP and EP\_PCI, we graphed each feature as a histogram.



Additionally, we compared these distributions to other features that we believe would be related to this linear model, such as the percentage of people who are uninsured (EP\_UNINSUR) and the percentage of mobile homes in the city (EP\_MOBILE).



### Analysis

We will conduct a **KS Analysis** to test whether the distribution of these features in our dataset are statistically similar.

* **H0:** The distribution of both features are the same.
* **H1:** The distribution of the features differ.

We meet the conditions to perform this test because these features are independent from one another and the data is continuous.

Exact two-sample Kolmogorov-Smirnov test: data$EP\_UNEMP and data$EP\_PCI

| Test statistic | P value | Alternative hypothesis |
| --- | --- | --- |
| 1 | NA NA | two-sided |

Exact two-sample Kolmogorov-Smirnov test: data$EP\_UNEMP and data$EP\_UNINSUR

| Test statistic | P value | Alternative hypothesis |
| --- | --- | --- |
| 0.2257 | 0 \* \* \* | two-sided |

Exact two-sample Kolmogorov-Smirnov test: data$EP\_UNEMP and data$EP\_MOBILE

| Test statistic | P value | Alternative hypothesis |
| --- | --- | --- |
| 0.5283 | 0 \* \* \* | two-sided |

Exact two-sample Kolmogorov-Smirnov test: data$EP\_PCI and data$EP\_UNINSUR

| Test statistic | P value | Alternative hypothesis |
| --- | --- | --- |
| 1 | NA NA | two-sided |

Exact two-sample Kolmogorov-Smirnov test: data$EP\_PCI and data$EP\_MOBILE

| Test statistic | P value | Alternative hypothesis |
| --- | --- | --- |
| 1 | NA NA | two-sided |

### Conclusion

When looking at our visualizations for the features, it appears that are is no significant difference between their distributions, as they are all right-skewed. However, our results from the KS test reflect the opposite conclusion, as the test statistics for each test are less than the test statistic at alpha = 0.05. As a result, we cannot conclude if the distributions of these features are statistically similar, and therefore we cannot conclude that they affect the value of EP\_UNEMP.

## **Question 3: Hypothesis Testing** - Do cities in states with higher percentages of EP\_PCI (estimated per capita income) have significantly fewer climate action responses compared to those with lower percentages?

### Methods

H0: There is no significant difference between the number of climate action responses in states with higher percentages of EP\_PCI compared to states with lower percentages. H1: States with higher percentages of EP\_PCI have significantly *fewer* climate action responses compared to states with lower percentages of EP\_PCI.

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

We have just created a DataFrame that shows the state, whether they took a climate or water action, and the EP\_PCI (estimated per capita income) for each city recorded. NOTE: Cities are not included in our merged DataFrame since we are just interested in the cities’ states.

### Analysis

The summary\_df keeps track of the number of climate change-related actions taken for various cities grouped by state and PCI status (either higher or lower than the median value).

##   
## Shapiro-Wilk normality test  
##   
## data: higher\_counts  
## W = 0.41224, p-value = 1.8e-11

##   
## Shapiro-Wilk normality test  
##   
## data: lower\_counts  
## W = 0.45169, p-value = 4.79e-11

Neither of these groups are normally distributed, so we need to use a Mann-Whitney U Test for comparing the mean counts of climate action for the groups with EP\_PCI lower vs. higher than the median EP\_PCI.

##   
## Wilcoxon rank sum exact test  
##   
## data: count by pci\_status  
## W = 769, p-value = 0.3852  
## alternative hypothesis: true location shift is less than 0

Because (p = 0.3852) > (alpha = 0.05), we fail to reject the null hypothesis. Cities in states with higher percentages of EP\_PCI *do not* have significantly fewer climate action responses compared to those with lower percentages.

### Conclusion

We merged the SVI Data with the city data and performed data cleaning to keep track of the number of cities with an EP\_PCI status higher or lower than the median EP\_PCI status. Since Shapiro-Wilk tests showed that our data for each group - Higher and Lower - were both not normally distributed, we had to use a Mann Whitney U Test (aka Wilcoxon Rank Sum Test) to compare the two groups since it is a non-parametric statistical test. Since we ended with a p-value of 0.3852 > 0.05, we fail to reject the null hypothesis. This means that cities in states with higher percentages of EP\_PCI *do not* have significantly fewer climate action responses compared to those with lower percentages.

## **Question 4: Correlation Analysis** - Is there a significant correlation between the EP\_PCI and EP\_UNEMP? How strong is the relationship, and what does it suggest about the role of unemployment in estimated per capita income?

### Methods

To understand the correlation between these two features, we calculate the covariance for the features.

## [1] -0.4062332

### Analysis

Next, we test the significance of their correlation with the following hypotheses:

* **H0:** The correlation between EP\_PCI and EP\_UNEMP is equal to zero.
* **H1:** The correlation between EP\_PCI and EP\_UNEMP is not equal to zero.

Pearson’s product-moment correlation: data$EP\_PCI and data$EP\_UNEMP

| Test statistic | df | P value | Alternative hypothesis | cor |
| --- | --- | --- | --- | --- |
| -119.4 | 72171 | 0 \* \* \* | two.sided | -0.4062 |

The correlation coefficient between EP\_PCI and EP\_UNEMP is **-0.4062332**, which signifies a somewhat strong negative correlation between the two features. After performing a significance test on this coefficient, it is clear that it is statistically significant, as the test statistic is **-119.4** and the p-value is **0**. The negative sign in the test statistic shows that the correlation between these two features is negative, with its large magnitude reinforcing this relationship. Additionally, the p-value tells us to reject the null hypothesis, providing evidence that the correlation between the two feature is not equal to zero.

### Conclusion

We can conclude that there is a statistically significant negative correlation between EP\_PCI and EP\_UNEMP, which is supported by the significance test we performed on the calculated correlation coefficient. This shows us that as unemployment increases within a city’s population, the city’s per capita income is expected to decrease. Additionally, the relationship between these two features is rather strong, as the magnitude of the correlation coefficient is a signiificant magnitude below zero.

## **Advanced Analysis: Multiple Regression** - Fit a regression line of the data predicting a city’s EP\_PCI (estimated per capita income) based on the estimated proportion of unemployment EP\_UNEMP, city population, and city density. How well does the regression line fit this relationship?

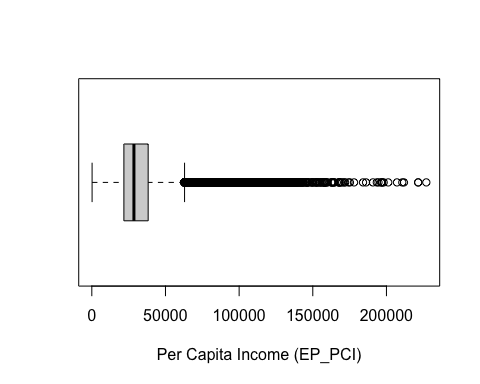
### Methods

We used XGBoost to optimize our prediction of EP\_PCI, the estimated per capita income.

### Data Preparation

To fit a regression line as specified above, we will need the following variables from the data dataset:

* Dependent Variable: EP\_PCI - estimated per capita income
* Independent Variables:
  + E\_NOHSDP - percentage of people without a high school diploma (age 25+)
  + EP\_DISAB - percentage of civilian non-institutionalized population who are disabled
  + EP\_MINRTY - percentage of minority people We choose these features because education is one of the strongest predictors of income, and we would expect people without a high school diploma to have fewer opportunities for job growth and thus, make a lower income than those with a higher degree. Disability status may also affect one’s ability to work full time, leading them to make lower incomes. Cities with a higher percentage of disabled individuals may have a lower per capita income. Thirdly, minority populations may deal with socioeconomic disparities and restricted opportunities to education, wealth, and jobs. We also expect there to be a relationship between the proportion of minorities and income. Perhaps cities with a higher proportion of minorities may also have a lower per capita income.

 As you can see, there are many outliers in our data, so we removed data points that were below the 25th percentile or above the 75th percentile, per the IQR Rule.

##   
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':  
##   
## slice

## Loading required package: ggplot2

## Loading required package: lattice

### Model Training

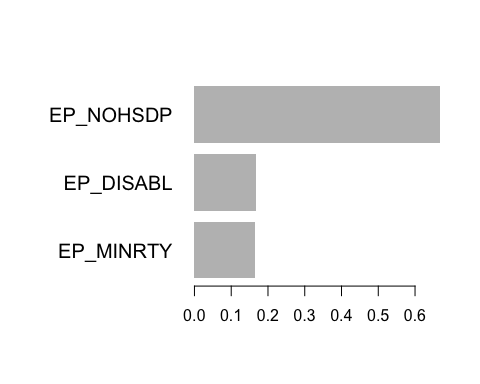
With the aid of the package Caret, we used 80% of the original dataset for training and the remaining 20% for testing. Our model used the objective function of regression with mean squared error and our evaluation metric was RMSE for better interpretability.

The hyperparameters we used were learning rate = 0.1, max\_depth = 6, subsample = 0.8 (80% of the samples used per boosting round to prevent overfitting), and colsample\_bytree = 0.8 (80% of the features used per tree).

## [1] train-rmse:28791.889314 test-rmse:28791.290757   
## Multiple eval metrics are present. Will use test\_rmse for early stopping.  
## Will train until test\_rmse hasn't improved in 10 rounds.  
##   
## [2] train-rmse:26113.645433 test-rmse:26110.284032   
## [3] train-rmse:23725.952867 test-rmse:23716.488420   
## [4] train-rmse:21597.012579 test-rmse:21586.526410   
## [5] train-rmse:19747.647454 test-rmse:19735.929866   
## [6] train-rmse:18062.854529 test-rmse:18046.039756   
## [7] train-rmse:16571.865686 test-rmse:16550.509275   
## [8] train-rmse:15289.802376 test-rmse:15267.135614   
## [9] train-rmse:14125.610842 test-rmse:14098.658025   
## [10] train-rmse:13140.374899 test-rmse:13113.405451   
## [11] train-rmse:12244.501491 test-rmse:12216.596353   
## [12] train-rmse:11469.269870 test-rmse:11438.264607   
## [13] train-rmse:10795.194067 test-rmse:10764.854267   
## [14] train-rmse:10221.984904 test-rmse:10189.997579   
## [15] train-rmse:9747.041050 test-rmse:9717.912875   
## [16] train-rmse:9343.638943 test-rmse:9316.172037   
## [17] train-rmse:8978.757091 test-rmse:8952.416938   
## [18] train-rmse:8672.547570 test-rmse:8647.662720   
## [19] train-rmse:8416.786088 test-rmse:8394.893787   
## [20] train-rmse:8216.274186 test-rmse:8196.676431   
## [21] train-rmse:8048.662284 test-rmse:8031.601790   
## [22] train-rmse:7895.768012 test-rmse:7879.960467   
## [23] train-rmse:7780.749299 test-rmse:7769.297153   
## [24] train-rmse:7674.226962 test-rmse:7666.638351   
## [25] train-rmse:7586.881889 test-rmse:7583.170886   
## [26] train-rmse:7515.148433 test-rmse:7515.254272   
## [27] train-rmse:7461.964848 test-rmse:7466.411989   
## [28] train-rmse:7410.063790 test-rmse:7419.095416   
## [29] train-rmse:7368.717652 test-rmse:7382.579669   
## [30] train-rmse:7337.802473 test-rmse:7355.816555   
## [31] train-rmse:7306.995466 test-rmse:7327.786283   
## [32] train-rmse:7280.671016 test-rmse:7304.823939   
## [33] train-rmse:7259.987793 test-rmse:7287.797740   
## [34] train-rmse:7242.052955 test-rmse:7272.615968   
## [35] train-rmse:7227.953827 test-rmse:7261.047284   
## [36] train-rmse:7216.110150 test-rmse:7252.915791   
## [37] train-rmse:7205.121052 test-rmse:7250.015559   
## [38] train-rmse:7195.669920 test-rmse:7245.308023   
## [39] train-rmse:7186.645944 test-rmse:7238.804029   
## [40] train-rmse:7180.604028 test-rmse:7236.983582   
## [41] train-rmse:7174.087270 test-rmse:7233.080776   
## [42] train-rmse:7167.186603 test-rmse:7230.699319   
## [43] train-rmse:7162.574994 test-rmse:7228.867275   
## [44] train-rmse:7157.985267 test-rmse:7228.474013   
## [45] train-rmse:7155.734816 test-rmse:7229.005543   
## [46] train-rmse:7152.254007 test-rmse:7228.588155   
## [47] train-rmse:7148.250335 test-rmse:7227.958251   
## [48] train-rmse:7145.046100 test-rmse:7227.826237   
## [49] train-rmse:7141.625770 test-rmse:7227.406961   
## [50] train-rmse:7138.952102 test-rmse:7225.932930   
## [51] train-rmse:7136.269516 test-rmse:7224.625251   
## [52] train-rmse:7131.963139 test-rmse:7226.376090   
## [53] train-rmse:7128.752480 test-rmse:7225.851226   
## [54] train-rmse:7126.688361 test-rmse:7226.549001   
## [55] train-rmse:7124.525367 test-rmse:7226.997982   
## [56] train-rmse:7122.080179 test-rmse:7227.271892   
## [57] train-rmse:7117.897586 test-rmse:7228.676427   
## [58] train-rmse:7114.981236 test-rmse:7228.971807   
## [59] train-rmse:7112.577202 test-rmse:7228.251129   
## [60] train-rmse:7109.476647 test-rmse:7227.758916   
## [61] train-rmse:7107.149941 test-rmse:7227.126328   
## Stopping. Best iteration:  
## [51] train-rmse:7136.269516 test-rmse:7224.625251

## RMSE: 7224.625

## MAE: 5243.471

 The plot importance graph in the XG Boost package shows the importance of each of the features we used: earning a high school degree - 0.667, disability status - 0.167, minority group - 0.166. It seems as though the feature of having a high school degree EP\_NOHSDP had the strongest impact on our model.

### Analysis

## [1] train-rmse:28767.467028+41.820326 test-rmse:28767.945013+99.320821   
## Multiple eval metrics are present. Will use test\_rmse for early stopping.  
## Will train until test\_rmse hasn't improved in 10 rounds.  
##   
## [2] train-rmse:26147.396567+52.217655 test-rmse:26149.493110+78.273927   
## [3] train-rmse:23766.938070+46.156461 test-rmse:23771.826127+79.926146   
## [4] train-rmse:21647.539507+63.884746 test-rmse:21653.587482+53.709223   
## [5] train-rmse:19756.401314+46.678633 test-rmse:19767.336302+68.676359   
## [6] train-rmse:18080.419880+45.044425 test-rmse:18094.140785+68.692599   
## [7] train-rmse:16605.960919+55.563501 test-rmse:16621.539753+53.046248   
## [8] train-rmse:15295.393182+35.080065 test-rmse:15316.063282+62.545391   
## [9] train-rmse:14129.086651+30.583420 test-rmse:14154.530399+61.027092   
## [10] train-rmse:13112.611180+20.817080 test-rmse:13142.681778+60.702228   
## [11] train-rmse:12231.092760+15.763892 test-rmse:12268.117632+69.432984   
## [12] train-rmse:11460.707010+13.240288 test-rmse:11501.674101+67.128031   
## [13] train-rmse:10794.344637+18.403720 test-rmse:10841.168647+65.696260   
## [14] train-rmse:10222.470963+26.233885 test-rmse:10274.321215+65.490006   
## [15] train-rmse:9730.074032+20.296670 test-rmse:9788.192997+57.948599   
## [16] train-rmse:9316.471581+26.489089 test-rmse:9380.307206+60.362415   
## [17] train-rmse:8964.398218+33.186481 test-rmse:9033.900666+59.140278   
## [18] train-rmse:8669.567815+39.454776 test-rmse:8744.752058+60.949494   
## [19] train-rmse:8415.610546+32.615053 test-rmse:8496.516711+51.527072   
## [20] train-rmse:8201.071097+27.123398 test-rmse:8287.081191+42.256729   
## [21] train-rmse:8027.415937+21.192031 test-rmse:8118.299443+35.513780   
## [22] train-rmse:7884.977850+24.157363 test-rmse:7979.819578+33.414939   
## [23] train-rmse:7763.645896+20.478481 test-rmse:7864.165525+29.294215   
## [24] train-rmse:7659.832389+16.863088 test-rmse:7766.015915+21.564263   
## [25] train-rmse:7575.163515+18.798549 test-rmse:7685.887992+18.739965   
## [26] train-rmse:7506.605770+20.781303 test-rmse:7621.846423+18.365381   
## [27] train-rmse:7447.281438+16.524947 test-rmse:7567.450667+13.245630   
## [28] train-rmse:7401.409871+17.993996 test-rmse:7525.375535+12.855159   
## [29] train-rmse:7363.368319+19.402895 test-rmse:7491.324930+13.365621   
## [30] train-rmse:7326.259276+17.455510 test-rmse:7458.499753+12.127357   
## [31] train-rmse:7297.055708+14.789630 test-rmse:7434.363318+10.431772   
## [32] train-rmse:7274.173654+14.818095 test-rmse:7414.481159+9.136084   
## [33] train-rmse:7254.084608+16.118766 test-rmse:7398.824090+8.186894   
## [34] train-rmse:7234.430866+14.481323 test-rmse:7383.890875+9.203537   
## [35] train-rmse:7217.297204+12.767733 test-rmse:7370.692315+10.005155   
## [36] train-rmse:7204.989389+13.464098 test-rmse:7362.081420+10.147842   
## [37] train-rmse:7193.098127+14.970901 test-rmse:7353.705390+9.210647   
## [38] train-rmse:7182.699452+13.699932 test-rmse:7347.282330+10.928568   
## [39] train-rmse:7174.427216+12.932416 test-rmse:7342.468620+11.705662   
## [40] train-rmse:7165.737019+11.348141 test-rmse:7337.874621+13.991556   
## [41] train-rmse:7157.453728+10.682054 test-rmse:7334.386521+13.681076   
## [42] train-rmse:7150.632806+9.720385 test-rmse:7331.966838+14.858379   
## [43] train-rmse:7144.889487+10.435446 test-rmse:7329.292559+15.467196   
## [44] train-rmse:7139.186631+10.535972 test-rmse:7327.559427+16.494010   
## [45] train-rmse:7134.623830+11.205591 test-rmse:7325.849647+16.786840   
## [46] train-rmse:7129.467813+10.739371 test-rmse:7325.022301+16.984454   
## [47] train-rmse:7123.697323+10.347692 test-rmse:7323.220135+17.760988   
## [48] train-rmse:7120.145093+11.352782 test-rmse:7322.253279+17.639103   
## [49] train-rmse:7115.086035+11.513814 test-rmse:7320.817229+18.320591   
## [50] train-rmse:7110.466901+11.308731 test-rmse:7319.642928+19.118471   
## [51] train-rmse:7106.118177+10.992375 test-rmse:7319.553534+19.972202   
## [52] train-rmse:7102.397155+10.839206 test-rmse:7319.151180+20.473606   
## [53] train-rmse:7098.411742+10.666530 test-rmse:7319.813800+21.157911   
## [54] train-rmse:7094.834251+10.089105 test-rmse:7318.951150+21.741089   
## [55] train-rmse:7091.424169+10.010231 test-rmse:7319.293514+21.551044   
## [56] train-rmse:7087.914088+10.062092 test-rmse:7318.998479+21.458105   
## [57] train-rmse:7084.526695+10.111767 test-rmse:7318.893066+21.445097   
## [58] train-rmse:7079.898621+10.176995 test-rmse:7319.123991+21.634411   
## [59] train-rmse:7077.001999+10.462055 test-rmse:7319.440640+22.171867   
## [60] train-rmse:7072.931056+10.074815 test-rmse:7320.085362+22.517537   
## [61] train-rmse:7069.270782+10.421702 test-rmse:7320.147758+23.044623   
## [62] train-rmse:7065.295422+10.242960 test-rmse:7320.460342+23.284751   
## [63] train-rmse:7062.306601+10.105623 test-rmse:7320.641078+23.548631   
## [64] train-rmse:7058.683678+9.527883 test-rmse:7320.957011+23.847115   
## [65] train-rmse:7055.161048+9.408577 test-rmse:7321.249534+23.118277   
## [66] train-rmse:7051.489296+9.244965 test-rmse:7321.204389+23.396609   
## [67] train-rmse:7048.171863+8.822082 test-rmse:7321.746800+23.897909   
## Stopping. Best iteration:  
## [57] train-rmse:7084.526695+10.111767 test-rmse:7318.893066+21.445097

## Best number of rounds: 57

The best number of rounds we can use in our model is about 56. We will train our model again with the new optimal best number of rounds.

## [1] train-rmse:28725.110641 test-rmse:28720.399887   
## Multiple eval metrics are present. Will use test\_rmse for early stopping.  
## Will train until test\_rmse hasn't improved in 10 rounds.  
##   
## [2] train-rmse:26109.597465 test-rmse:26103.601979   
## [3] train-rmse:23784.390240 test-rmse:23778.607635   
## [4] train-rmse:21647.897840 test-rmse:21639.401355   
## [5] train-rmse:19747.215138 test-rmse:19735.591844   
## [6] train-rmse:18107.743846 test-rmse:18096.894951   
## [7] train-rmse:16665.162443 test-rmse:16652.986650   
## [8] train-rmse:15332.585131 test-rmse:15316.044460   
## [9] train-rmse:14217.602303 test-rmse:14200.824423   
## [10] train-rmse:13185.993052 test-rmse:13167.681668   
## [11] train-rmse:12286.214965 test-rmse:12267.043943   
## [12] train-rmse:11549.750288 test-rmse:11531.669862   
## [13] train-rmse:10868.623160 test-rmse:10849.714925   
## [14] train-rmse:10276.805523 test-rmse:10256.831289   
## [15] train-rmse:9777.312919 test-rmse:9759.863225   
## [16] train-rmse:9345.277442 test-rmse:9327.415836   
## [17] train-rmse:8986.004104 test-rmse:8970.587327   
## [18] train-rmse:8702.621843 test-rmse:8690.587320   
## [19] train-rmse:8445.938974 test-rmse:8437.007574   
## [20] train-rmse:8229.702069 test-rmse:8223.248350   
## [21] train-rmse:8066.581539 test-rmse:8064.447969   
## [22] train-rmse:7915.959337 test-rmse:7917.211906   
## [23] train-rmse:7787.114394 test-rmse:7790.864048   
## [24] train-rmse:7693.322269 test-rmse:7699.633138   
## [25] train-rmse:7605.817899 test-rmse:7615.721368   
## [26] train-rmse:7529.760560 test-rmse:7543.909421   
## [27] train-rmse:7467.227225 test-rmse:7485.421807   
## [28] train-rmse:7415.845249 test-rmse:7437.628878   
## [29] train-rmse:7379.899255 test-rmse:7406.405200   
## [30] train-rmse:7351.684647 test-rmse:7381.239699   
## [31] train-rmse:7327.466906 test-rmse:7360.251329   
## [32] train-rmse:7306.156580 test-rmse:7342.484420   
## [33] train-rmse:7289.189236 test-rmse:7329.318868   
## [34] train-rmse:7269.387798 test-rmse:7312.495217   
## [35] train-rmse:7253.964747 test-rmse:7299.162389   
## [36] train-rmse:7237.693202 test-rmse:7286.609276   
## [37] train-rmse:7226.778928 test-rmse:7277.998255   
## [38] train-rmse:7218.847842 test-rmse:7273.719985   
## [39] train-rmse:7212.492757 test-rmse:7270.270200   
## [40] train-rmse:7206.974116 test-rmse:7267.156165   
## [41] train-rmse:7197.489601 test-rmse:7260.698697   
## [42] train-rmse:7189.097080 test-rmse:7255.309137   
## [43] train-rmse:7181.487241 test-rmse:7254.380645   
## [44] train-rmse:7173.827704 test-rmse:7249.794967   
## [45] train-rmse:7167.109449 test-rmse:7250.242370   
## [46] train-rmse:7160.057929 test-rmse:7246.174072   
## [47] train-rmse:7155.188729 test-rmse:7243.108229   
## [48] train-rmse:7151.299333 test-rmse:7243.125016   
## [49] train-rmse:7146.188612 test-rmse:7243.236046   
## [50] train-rmse:7143.208062 test-rmse:7243.155871   
## [51] train-rmse:7139.951725 test-rmse:7241.695396   
## [52] train-rmse:7138.567751 test-rmse:7241.531760   
## [53] train-rmse:7135.590838 test-rmse:7242.647642   
## [54] train-rmse:7132.182040 test-rmse:7240.876793   
## [55] train-rmse:7129.605443 test-rmse:7240.331249   
## [56] train-rmse:7125.390704 test-rmse:7239.634117   
## [57] train-rmse:7121.452555 test-rmse:7238.897583

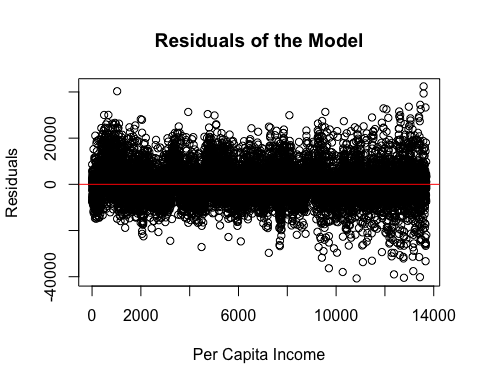
## RMSE: 7224.625

## MAE: 5243.471

After performing cross-validation, our model has an RMSE of about 7224.625 and an MAE of about 5243.471. Our **normalized RMSE** would be 7224.625 / mean(target) = **0.2439**, which means that the model’s predictions deviate from the actual values by about 24.39% on average. The **coefficient of variation (CV)** for this problem would be sd(target) / mean(target) = 0.3826. 1 - 0.2439/0.3826 = **0.3624**, which means our normalized RMSE is > 30% smaller than the coefficient of variation. This indicates that our model *does* explain a significant amount of the variance in the data.

### Conclusion

* At first glance, the model appears to have an extremely high RMSE, but when normalized against the mean(target), our model is actually reasonably accurate in predicting per capita income.
* From the feature importance, we saw that EP\_NOHSDP had the highest gain, which means it has the greatest contribution to reducing prediction errors. Contextually, this makes sense because low-income people may not be able to afford tuition at a university and with a lower degree, have a smaller skillset and thus, be paid less.
* Since EP\_NOHSDP was the most significant predictor, tutoring programs that help students in grades K-12 grasp a better understanding of their coursework may improve per capita income in the populaton since more students would be able to achieve high school diplomas.

 - Our model does a fairly good job at predicting! The residuals are roughly randomly scattered around 0. The model does appear to have some fanning present as income increases after $8,000, but is roughly still homoscedastic.

## Conclusion & Discussion

In conclusion, our analysis of the causal relationship between per capita income (EP\_PCI) and unemployment rate (EP\_UNEMP) in the socioeconomic vulnerability index dataset reveal that there is a strong negative correlation between them, though a linear regression may not be the best model to capture the nuances of their relationship. Additionally, the distributions of these features appear to be similar, but there is no statistical evidence to support this. In our advanced analysis, we attempt to fit a multiple linear regression with the addition of new features that describe the population distribution and density of a city.

One limitation that we noticed with our data was missing data in the socioeconomic vulnerability index dataset. Although we removed missing data, the resulting data could underestimate or overestimate certain features of the dataset, which inhibits our ability to make accurate calculations.