STA141A Final Project

William Shih, Chi Chen

November 24, 2019

Tasks:

William Shih - Parts 2-4 + Functions, Exploratory Data Analysis, Regression, List of Functions

Chi Chen - Part 1, Part 5 Introduction and Classification

I. Introduction

1.1 - Description of the Dataset

This project uses the American Community Survey 2018 Public Use Microdata Sample which is a survey by the U.S. Census Bureau that collects detailed demographic data from approximately 1% of the US population each year. his is a recent dataset which was collected throughout 2018 and released on November 14, 2019. We will use just the person records. We will filter the dataset to those age 2 5-64. An important detail about this dataset that it is a weighted sample, so sample weights have to be used for calculating any mean, median, or regression coefficient.

For our project, we will not use all of the variables, but just a subset of 20 variables. We chose this list of 20 variables before doing any analysis and we did not do any analysis on any variable that is not in this list of 20 variables. We will mainly focus on demographic and income related variables such as education, race, sex, occupation, industry, income, hours worked, and area.

Here are the list of variables from the survey used in this project. An additional variable called PWGTP will be the weights. The 80 replicate weights are called PWGTP1, PWGTP2, ... to PWGTP80.

Table 1: List of Variables used in this Project

Variables	Definitions
PERNP	Earnings
NAICSP	Industry
OCCP	Occupation
JWAP	Time Arrival to Work
WKW	Weeks Worked Past Year
WRK	Worked Last Week
MAR	Marital Status
MARHT	Times Married
ENG	English Fluency
JWTR	Mode of Transport to Work
QTRBIR	Quarter of Birth
RAC3P	Race
SEX	Sex
WKHP	Usual Hours Worked per Week
SCHL	Educational Attainment
HISP	Hispanic origin
JWMNP	Travel Time to Work
AGEP	Age
PUMA	Area
FOD1P	Field of Degree

Note that we modified the dataset before doing any analysis on it. See "Appendix 3 - Getting the Code" to see how to replicate the exact dataset used in this project.

1.2 - Statistical Methods

We will use weighted least squares (WLS) to predict the level of earnings from all the other variables. The dataset is a weighted sample and the lm() function allows us to use the sample weights in the model. We will focus only on earned income from work for the regression part and we will look at two different measures of income from work: earnings per year and earnings per hour. Earnings per hour can be calculated by multiplying hours worked per week in past 12 months and weeks worked in past 12 months, which I then divide by the total earnings.

Classification will also be used to predict earnings labor force partipication and marital status. We will use linear disriminant analysis, logistic regression, and k-nearest neighbor regression to classify labor force partipication and marital status. In the project proposal, we stated we would use AIC, BIC, Mallow's Cp, and adjusted R-squared to select the best weighted least squares model. We will prefer using BIC over the others to reduce the number of parameters selected in the model. We will calculate and list AIC, BIC, and adjusted R-squared of all the considered models in tables. BIC has a penalty of log(n) per parameter added, while AIC has a penalty of 2 per paramater added. There are over 1,200,000 data points in our dataset. That means BIC has a penalty of 14 per parameter added and AIC has a panelty of 2. The lm() function uses a large amount of memory, so we try not to use too many parameters. We use the 'biglm' package for all of our calculations except for the final model. We prefer to use lm() for the final model since we can use anova() and diagnostic tests on a lm() object, but not on a 'speedlm' or 'biglm' object. That is why we don't want too many paramters or there won't be enough memory to use lm().

A more accurate method of calculating standard errors involve using the 80 replicate weights to obtain replicate standard errors. Because using these replicate weights use a lot of computation time, only the final model will use the replicate weights. We will calculate confidence intervals for the parameters and R-squared using the standard errors calculated from the replicate weights. These replicate weights were calculated using the successive difference replication method. We will ignore the standard errors calculated in the lm() model and only use the replicate standard errors. These replicate standard errors are generally larger than the standard errors from the lm() model. This involves running the regression 81 times and the code for that can be found in "Appendix 4 - Getting the Standard Errors".

Information about the ACS replicate weights and formula: https://usa.ipums.org/usa/repwt.shtml

1.3 - Rest of the Report

Part 2 is about with the summary statistics to explain each of the variables.

Parts 3-4 are about the weighted least squares model on earnings per hour and earnings per year.

Part 5 is about classification on marital status and labor force partipication using k-NN, logistic regression, and LDA. This section is relatively short and not as detailed, because computation is very long with complex models with glm and k-NN.

II. Summary Statistics and Plots

This section describes summary statistics for the regression model only. The big diference is that the classification models won't be filtered, but the regression model will be. This is because of the transformation to the dependent variable earnings per year and earnings per hour.

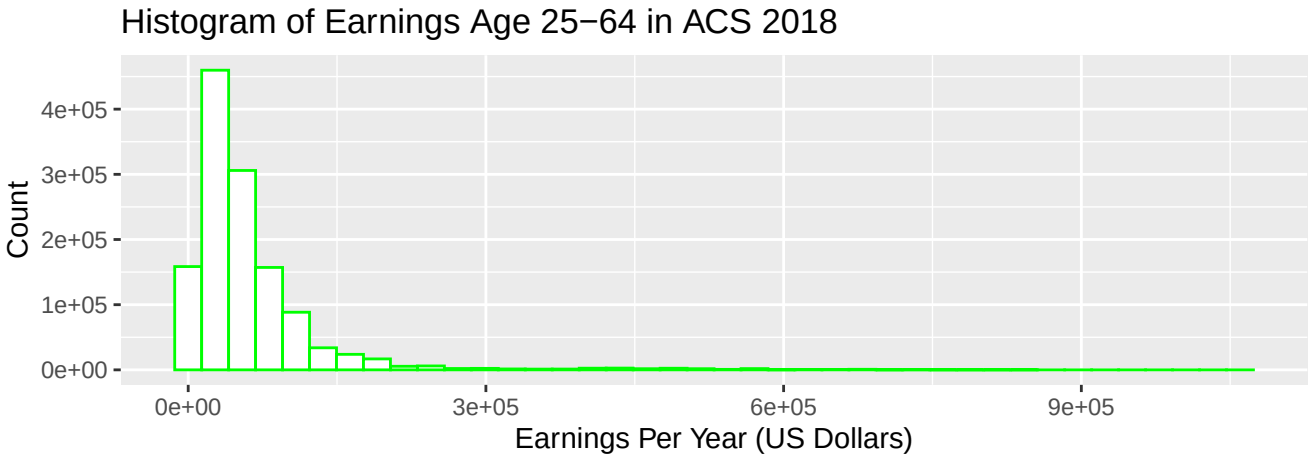
2.1 - PERNP - Earnings

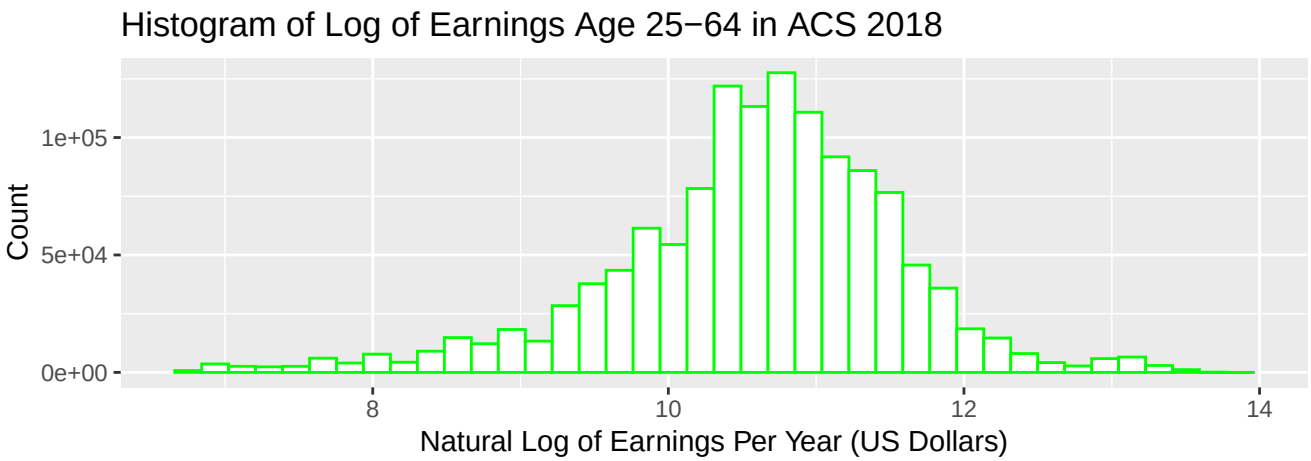
We will remove all values 0 or below.

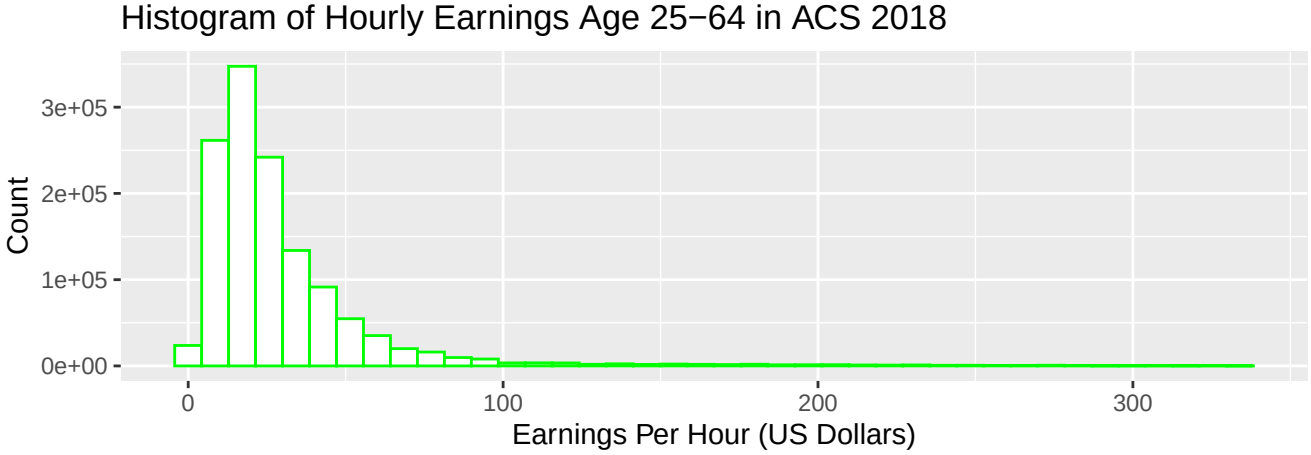
Two reasons for this: 1) We will do a log transformation on PERNP and we can't take the logarithm of values of 0 or lower. 2) It is impossible to predict the "potential income" of people who are not working, so we want to only include people are working.

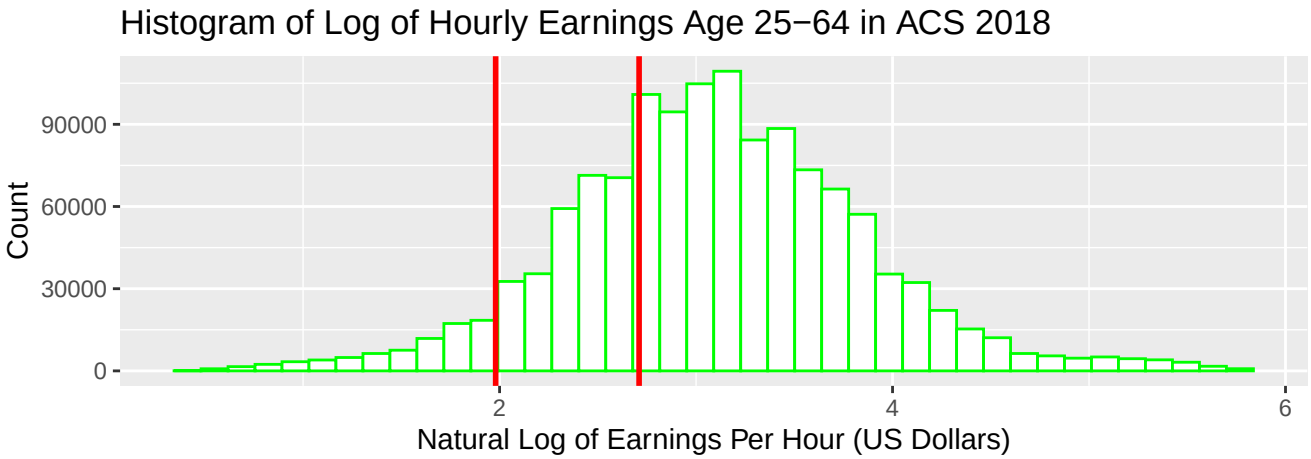
We will also remove all values in the bottom 1% of those making at least $1, because these people worked very few hours so their "potential income" is unreliable. The bottom 1% earned 900 dollars or less in the past 12 months, which is less than 10% as much as a minimum wage worker. Once calculating earnings per hour, we will further remove the bottom 0.2% and top 0.2%, because of outliers of those reporting high income and very few hours or vice versa. Many of these people misreported their hours worked or total income.

This final dataset will used for all regression analysis of earnings and earnings per hour. This dataset has 1,279,767 observations.









The logarithm of earnings as the distribution of earnings is closer to a log-normal distribution than a normal distribution. The distribution of hourly earnings is not skewed, while the distribution of annual earnings is slightly skewed. We believe that this is because of part time workers and people who do not work the whole year.

The red lines represent the $7.25 per hour and $15 per hour. There are workers who earn less than minimum wage, but there appears to be a break in the distribution approximately where the minimum wage is.

We often think of changes in income and prices as proportional changes, such as stock prices and inflation and wage growth. Empirically, it has been found that income is log-normal with a power law upper tail. Thus, we expect a heavy upper tail in the Q-Q plot. It makes sense to do the linear model with the dependent variable as logarithm of earnings and the logarithm of earnings per hour, beause we expect most variables such as education to have a proportional impact on income, not a linear one. So we will apply a log transformation on the dependent variable and use the natural log of earnings and natural log of earnings per hour.

For the rest of the summary tables, we will convert the average of the log transformed earnings back to actual numbers to make them easier to interpret. The average of the log tranformed earnings is equal to the geometric mean of the earnings. If the earnings are not skewed, the log-average should be equal to the median. For earnings per year, the log-average is lower than the median, indicating it is skewed. But for earnings per hour, the log-average is usually very close to the median, indicating that is likely not skewed.

2.2 - NAICSP - Industry

Table 2: Converting 2018 ACS Industry Code to 2017 NAICS Sectors

Code	2017 NAICS Sector
None	None
AGR	Agriculture, Forestry, Fishing and Hunting
EXT	Mining, Quarrying, and Oil and Gas Extraction
UTL	Utilities
CON	Construction
MFG	Manufacturing
WHL	Wholesale Trade
RET	Retail Trade
TRN	Transportation and Warehousing
INF	Information
FIN	Finance, Insurance, Real Estate, Rental, Leasing
PRF	Professional, Scientific, and Technical Services+Management of Companies
EDU	Educational Services
MED	Health Care
SCA	Social Assistance
ENT	Arts, Entertainment, Recreation, Accommodation, Food Services
SRV	Other Services
ADM	Public Administration
MIL	Military

Table 3: Geometric Mean and Median Earnings per Year/Hour split by Industry

	Earnings Per Year		Earnings Per Hour		
Industry	Log-Average	Median	Log-Average	Median	# of Workers
UTL	65,910	70,917	31.58	33.50	1,171,477
EXT	65,393	70,917	27.88	27.35	725,396
FIN	56,130	55,720	28.54	26.87	9,055,269
INF	54,996	60,786	28.77	29.31	2,668,872
ADM	53,453	57,747	26.59	26.87	6,713,887
MIL	52,372	55,720	22.43	23.45	625,805
MFG	46,875	48,629	22.93	21.98	14,316,143
WHL	46,170	48,629	22.85	21.98	3,469,590
PRF	45,359	50,655	25.57	25.40	16,572,394
MED	41,284	42,044	23.07	21.98	16,194,116
CON	38,307	40,524	20.51	20.10	9,694,686
TRN	38,124	41,537	19.81	19.54	6,669,895
EDU	36,285	44,272	21.56	22.08	12,502,542
RET	27,627	30,393	16.43	15.63	12,929,479
AGR	27,078	30,393	14.43	13.78	1,664,111
SRV	25,990	29,886	16.41	16.33	6,355,439
ENT	22,210	24,314	14.28	13.84	10,176,357
SCA	22,132	25,327	14.40	14.65	2,851,447

Table 4: Geometric Mean and Median Earnings per Year/Hour split by Industry

	Earnings Per Year		Earnings Per Hour		
Industry	Log-Average	Median	Log-Average	Median	# of Workers
95%-100%	80,390	86,113	38.87	39.43	7,202,600
90%-95%	65,864	70,917	32.17	31.91	6,656,751
85%-90%	61,048	63,825	30.36	29.53	6,721,657
80%-85%	56,321	59,773	27.64	27.35	7,534,281
75%-80%	51,574	53,694	26.84	26.87	8,224,729
70%-75%	50,288	49,642	24.84	23.06	4,045,279
65%-70%	46,477	50,655	23.96	23.76	6,788,566
60%-65%	43,447	48,629	23.87	23.88	7,692,906
55%-60%	41,737	43,563	21.44	20.52	5,818,509
50%-55%	38,743	40,524	20.60	20.03	15,264,069
40%-45%	36,750	39,004	20.36	19.54	4,721,598
35%-40%	34,764	42,550	20.45	21.44	8,559,956
30%-35%	33,674	36,471	18.96	18.91	5,044,952
25%-30%	30,534	32,419	16.92	16.28	6,715,595
20%-25%	26,710	30,393	16.62	16.28	7,750,957
15%-20%	24,807	27,151	16.20	15.54	5,527,293
10%-15%	22,378	25,327	14.20	13.88	7,520,881
5%-10%	20,185	21,782	13.01	12.59	7,245,654
0%-5%	17,858	20,262	12.85	12.25	5,320,672

This variable classifies the industry the person works in. There are two ways to split up NAICSP (standing for North American Industrial Classification System), which classifies businesses by type of economic activity. We can split it up by sector (combined sector 54/55 and sector 71/72) or by quantiles of equal size groups. We found the geometric mean earnings of every single NAICS code and sorted them. Then we calculated 20 equal size groups for the NAICSP variable.

So the 95%-100% category indicates the top 5% paying industries and 0%-5% category indicates the lowest 5% paying industries. The industry groups are not the same size since some industries are larger thn other industries and the industries do not evenly divide into the 20 groups.

Splitting up NAICSP by 20 equal size groups creates a larger range of group means and medians than splitting up NAICSP by 18 sectors, so it will be a better predictor of earnings. We believe that splitting into these equal size groups is always going be at least as good of a predictor as splitting up by sector. This is because while splitting up by sector can result in low and high paying industries being in the same sector; this can never be the case when we are sorting by the geometric mean earnings and forming equal size groups.

2.3 - OCCP - Occupation

Table 5: Converting 2018 ACS Occupation Code to SOC Major Occupational Groups

Code	Major Occupational Group
None	None
MGR	Management
BUS	Business
FIN	Financial Operations
CMM	Computer and Mathematical
ENG	Architecture and Engineering
SCI	Life, Physical, and Social Science
CMS	Community and Social Service
LGL	Legal
EDU	Education, Training, and Library
ENT	Arts, Design, Entertainment, Sports,and Media
MED	Healthcare Practitioners and Technical
HLS	Healthcare Support
PRT	Protective Service
EAT	Food Preparation and Serving Related
CLN	Building and Grounds Cleaning and Maintenance
PRS	Personal Care and Service
SAL	Sales and Related
OFF	Office and Administrative Support
FFF	Farming, Fishing,and Forestry
CON	Construction and Extraction
RPR	Installation, Maintenance, and Repair
PRD	Production
TRN	Transportation and Material Moving
MIL	Military

Table 6: Geometric Mean and Median Earnings per Year/Hour split by Occupation

	Earnings Per Year		Earnings Per Hour		
Occupation	Log-Average	Median	Log-Average	Median	# of Workers
LGL	81,436	81,048	40.00	38.10	1,531,184
ENG	74,504	81,048	35.72	36.91	2,654,374
CMM	73,910	81,048	36.85	39.08	4,664,586
MGR	68,642	70,917	31.92	31.75	14,710,697
FIN	65,344	65,851	32.83	31.75	3,136,780
BUS	59,785	64,838	30.48	30.29	4,704,364
SCI	59,479	62,812	30.58	30.44	1,306,157
MED	56,869	59,773	31.02	30.39	8,788,490
MIL	52,445	57,747	23.00	23.45	312,429
PRT	45,295	50,655	22.61	22.96	2,856,205
RPR	42,113	45,589	20.79	21.49	4,209,322
CMS	38,594	42,550	21.13	21.49	2,366,149
ENT	36,621	45,083	24.01	24.42	2,708,982
SAL	35,797	39,511	20.15	19.54	12,047,460
CON	34,810	38,545	19.01	19.54	7,473,235
EDU	34,412	43,563	21.18	21.98	8,362,450
PRD	32,994	35,965	17.20	17.19	7,915,203
OFF	31,395	35,458	18.07	17.91	14,947,706
TRN	28,472	30,393	15.94	15.63	10,111,052
FFF	21,448	24,314	12.48	12.21	926,058
HLS	21,280	25,327	13.85	13.87	4,255,724
CLN	19,595	22,288	13.24	13.03	5,352,926
PRS	18,973	21,275	13.36	13.19	3,281,355
EAT	17,795	20,262	11.98	11.88	5,734,017

Table 7: Geometric Mean and Median Earnings per Year/Hour split by Occupation

	Earnings Per Year		Earnings Per Hour		
Occupation	Log-Average	Median	Log-Average	Median	# of Workers
95%-100%	112,717	111,441	51.73	51.17	6,902,020
90%-95%	80,193	86,113	38.13	39.08	6,590,167
85%-90%	71,098	75,982	33.67	34.19	6,991,195
80%-85%	64,232	68,891	30.42	30.94	6,538,887
75%-80%	60,797	64,838	30.01	29.51	6,624,902
70%-75%	54,916	60,786	28.51	29.31	6,993,923
65%-70%	50,525	52,681	25.47	25.40	6,385,419
60%-65%	46,288	50,655	24.73	24.42	6,779,278
55%-60%	41,743	48,629	22.44	23.01	9,155,325
50%-55%	39,715	42,246	19.70	19.54	4,314,318
45%-50%	37,029	40,524	18.72	19.54	8,012,654
40%-45%	33,982	38,498	18.58	18.92	5,635,317
35%-40%	31,082	35,256	17.21	17.19	6,564,042
30%-35%	28,686	32,216	16.85	16.91	6,819,882
25%-30%	27,122	30,393	16.31	16.12	7,270,018
20%-25%	24,519	28,367	15.38	14.94	7,591,124
15%-20%	21,540	25,327	14.05	14.07	5,575,008
10%-15%	20,117	23,301	13.18	13.03	6,207,561
5%-10%	17,293	20,262	12.17	11.95	7,074,159
0%-5%	14,320	16,210	11.47	11.23	6,331,706

This variable asks about the occupation of the worker. There are two ways to split up the occupation variable. We can split it up by major occupational group or by quantiles of equal size groups sorted by the geometric mean of subgroups of the major occupational groups.

2.4 - JWAP - Time Arrival to Work

Table 8: Geometric Mean and Median Earnings per Year/Hour split by Time Arrival to Work

	Earnings Per Year		Earnings Per Hour		
Time Arrival to Work	Log-Average	Median	Log-Average	Median	# of Workers
8:30AM-8:45AM	46,654	50,655	24.40	24.42	5,621,424
8:15AM-8:30AM	46,135	48,629	24.02	23.45	6,280,775
8AM-8:15AM	45,774	48,629	23.56	23.45	8,655,531
8:45AM-9AM	45,716	48,629	24.28	24.01	4,416,026
7:45AM-8AM	44,857	45,589	22.85	21.98	9,243,098
7:15AM-7:30AM	44,743	45,589	22.35	21.98	7,407,880
7:30AM-7:45AM	44,707	45,589	22.62	21.98	8,515,089
7AM-7:15AM	44,508	46,096	21.98	21.71	7,154,138
9AM-9:15AM	44,117	48,629	23.72	23.45	4,322,348
6:45AM-7AM	43,488	45,589	21.58	21.30	6,378,087
6:30AM-6:45AM	43,298	45,589	21.35	21.00	5,294,369
6:15AM-6:30AM	42,523	44,576	20.91	20.52	3,948,620
6:00AM-6:15AM	42,383	44,576	20.62	20.03	3,465,837
9:15AM-9:30AM	41,790	44,576	22.92	22.12	2,589,462
9:30AM-9:45AM	41,156	42,550	22.60	21.98	2,033,361
5:30AM-6AM	40,739	42,550	19.83	19.54	5,323,138
5AM-5:30AM	39,496	40,524	19.09	19.15	2,821,346
4AM-5AM	38,293	40,524	18.45	18.57	2,660,228
9:45AM-10AM	36,260	38,498	21.02	19.54	1,405,527
12AM-4AM	35,743	38,498	17.77	17.76	1,865,465
6PM-11:59PM	34,802	37,485	18.92	18.79	4,093,896
10AM-10:15AM	34,247	35,762	20.14	19.54	1,519,769
10:15AM-10:30AM	32,843	33,726	19.59	18.76	1,035,407
10:30AM-11AM	31,309	32,419	18.95	17.93	1,317,025
11AM-12PM	29,191	30,393	17.96	17.14	1,525,507
1PM-2PM	28,712	30,393	16.74	16.12	1,385,451
2PM-6PM	28,073	30,393	16.71	16.28	6,695,324
12PM-1PM	27,839	30,393	17.10	16.47	1,165,768
Home	22,808	25,327	20.06	19.54	16,217,009

This variable classifies the time of day the worker normally arrives to work. This variable was arbitarily split based on how many people are in each group. People who arrive to work at 8:30AM-8:45AM get paid the most, which is 45 minutes later than the mode of 7:45-8:00AM, while people who arrive earlier or later get paid less.

2.5 - WKW - Weeks Worked Past Year

Table 9: Geometric Mean and Median Earnings per Year/Hour split by Weeks Worked Past Year

	Earnings Per Year		Earnings Per Hour		
Weeks Worked Past Year	Log-Average	Median	Log-Average	Median	# of Workers
50-52wks	46,223	46,602	21.93	21.49	111,725,674
48-49wks	33,555	33,432	19.78	18.92	2,632,737
40-47wks	26,726	27,354	18.75	17.96	6,842,306
27-39wks	16,774	16,716	16.60	15.93	5,833,829
14-26wks	9,424	9,523	15.75	14.85	3,817,442
01-13wks	4,302	3,911	22.07	20.29	3,504,917

This variable asks about the number of weeks worked in the past 12 months and must be a categorical variable since that is how it is divided in the survey. Generally, working more weeks per year increase both earnings per year and earnings per hour.

2.6 - WRK - Worked Last Week

Table 10: Geometric Mean and Median Earnings per Year/Hour split by Worked Last Week

	Earnings Per Year		Earnings Per Hour		
Worked Last Week	Log-Average	Median	Log-Average	Median	# of Workers
Worked last week	41,734	44,272	21.72	21.21	117,134,377
Not reported	32,613	35,458	19.06	18.76	9,758,546
Did not work last week	12,897	14,183	17.43	16.61	7,463,982

This variable asks whether the person worked last week. Many people did not respond to this question, so that may prove to be problematic. We may need to exclude this variable since many people did not respond to it.

2.7 - MAR - Marital Status

Table 11: Geometric Mean and Median Earnings per Year/Hour split by Marital Status

	Earnings Per Year		Earnings Per Hour		
Marital Status	Log-Average	Median	Log-Average	Median	# of Workers
Married	44,505	48,831	24.00	23.76	77,031,320
Divorced	36,150	40,524	20.00	19.54	16,261,084
Never Married	29,979	32,419	17.34	17.10	36,074,010
Widowed	29,715	32,419	18.25	17.59	1,886,123
Separated	28,228	30,393	16.71	16.12	3,104,368

This variable asks about the marital status of the person. Married workers earn the highest on average.

2.8 - MARHT - Number of Times Married

Table 12: Geometric Mean and Median Earnings per Year/Hour split by Times Married

	Earnings Per Year		Earnings Per Hour		
Times Married	Log-Average	Median	Log-Average	Median	# of Workers
Married Once	42,604	45,589	23.17	22.80	77,599,340
Married Twice	40,705	45,589	22.19	21.98	16,959,230
Married 3+ Times	37,330	41,537	20.80	20.52	3,724,325
Never Married	29,979	32,419	17.34	17.10	36,074,010

This variable asks about the number of times the person has married. Being married more than once correlates with lower earnings.

2.9 - ENG - English Fluency

Table 13: Geometric Mean and Median Earnings per Year/Hour split by English Fluency

	Earnings Per Year		Earnings Per Hour		
English Fluency	Log-Average	Median	Log-Average	Median	# of Workers
Only English	40,041	44,576	22.03	21.82	103,065,447
Very well	40,009	41,537	22.04	21.28	18,518,348
Well	29,955	30,393	17.13	16.28	6,573,345
Not well	23,047	25,327	13.59	13.03	4,629,958
Not at all	19,523	20,262	12.01	11.72	1,569,807

The variable asks about the ability of the person to speak English. Higher English literacy is correlated with higher earnings. However, it appears that the distribution for multilingual workers are right skewed, with the log-average being higher than the median, while the distribution is less skewed for English only speakers.

2.10 - JWTR - Mode of Transport to Work

Table 14: Geometric Mean and Median Earnings per Year/Hour split by Mode of Transport to Work

	Earnings Per Year		Earnings Per Hour		
Mode of Transport to Work	Log-Average	Median	Log-Average	Median	# of Workers
Ferryboat	85,992	91,179	39.87	40.69	60,877
Railroad	83,217	86,113	39.84	40.54	816,869
Subway	53,060	56,733	27.34	27.35	2,543,213
Streetcar	48,797	50,655	26.05	26.87	79,559
Motorcycle	45,947	48,629	22.70	22.30	190,451
Worked at home/Other	42,505	48,629	24.00	24.42	7,735,409
Car/truck/van	41,407	43,563	21.38	20.84	107,268,049
Bicycle	40,923	42,926	22.18	21.88	626,653
Taxi	32,551	30,393	18.56	16.86	258,282
Walked	32,024	32,078	18.30	17.10	2,469,130
Bus	31,041	30,393	18.34	17.10	2,734,676
Not at work	14,414	16,210	17.25	16.63	9,573,737

This variable asks about the principal type of transportation the person uses to get to work. Workers who primarily use trains earn more than workers who primarily drive to work. Workers who primarily use buses earn less than workers who primarily drive to work.

2.11 - QTRBIR - Quarter of Birth

Table 15: Geometric Mean and Median Earnings per Year/Hour split by Quarter of Birth

	Earnings Per Year		Earnings Per Hour		
Quarter of Birth	Log-Average	Median	Log-Average	Median	# of Workers
Apr-June	38,605	40,524	21.33	20.71	32,810,652
July-Sept	38,491	40,524	21.30	20.76	35,016,308
Oct-Dec	38,385	40,524	21.25	20.52	33,496,544
Jan-Mar	38,133	40,524	21.14	20.52	33,033,401

This variable asks about which quarter of the year (Q1,Q2,Q3, or Q4) the person was born in.

2.12 - RAC3P - Race

Table 16: Geometric Mean and Median Earnings per Year/Hour split by Race

	Earnings Per Year		Earnings Per Hour		
Race	Log-Average	Median	Log-Average	Median	# of Workers
Indian alone	62,278	78,008	33.98	39.08	2,048,614
Chinese alone	46,883	51,668	27.10	27.84	2,022,879
Filipino alone	40,903	45,589	22.94	23.45	1,455,794
White Alone	40,459	44,576	22.14	21.88	98,159,936
Other Asian or Pacific Islander	37,808	40,524	21.64	20.84	3,209,473
Mixed Race	35,707	40,524	20.41	19.54	3,454,415
Black Alone	30,330	33,432	17.40	17.10	16,267,282
Some Other Race alone	27,971	30,393	15.79	15.24	6,724,641
Native American Alone	27,737	31,203	16.64	16.63	1,013,871

This variable asks about the race of the person. We arbitrarily combined all of the mixed races together, which all individually each have a very small sample size. We grouped together all Asians which are not Chinese alone, Filipino alone, or Indian alone. For Indians alone, the median is much higher than the log-average, indicating that there is a long left tail.

2.13 - SEX - Sex

Table 17: Geometric Mean and Median Earnings per Year/Hour split by Sex

	Earnings Per Year		Earnings Per Hour		
Sex	Log-Average	Median	Log-Average	Median	# of Workers
Male	45,605	48,933	23.24	22.67	70,801,419
Female	31,713	35,458	19.24	19.29	63,555,486

This variable asks whether the person is a male or female. The gender wage gap appears to be 15% based on the median hourly wage. Also noteworthy is that male hourly wages are skewed (indicating a long right tail), but female hourly wages are not skewed. The log-average of the male hourly earnings is higher than the median of the male hourly earnings.

2.14 - WKHP - Usual Hours Worked Per Week

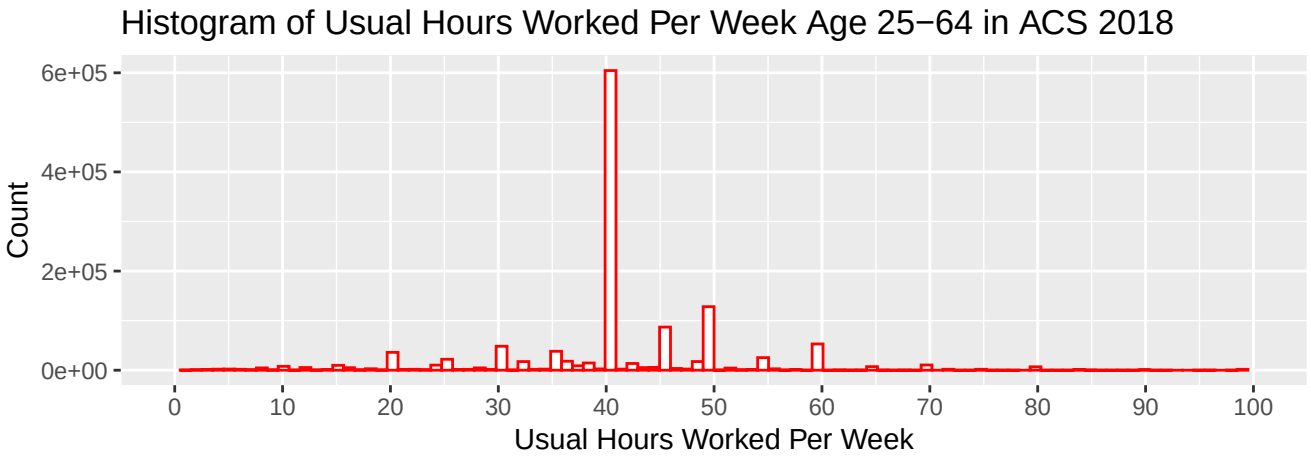


Table 18: Geometric Mean and Median Earnings per Year/Hour split by Usual Hours Worked per Week

	Earnings Per Year		Earnings Per Hour		
Usual Hours Worked per Week	Log-Average	Median	Log-Average	Median	# of Workers
61-65hrs	69,325	65,851	21.51	20.44	855,865
51-55hrs	68,185	65,851	24.96	23.91	3,305,493
56-60hrs	66,744	65,851	22.67	21.49	5,860,582
66-70hrs	63,279	60,786	18.52	17.59	1,205,208
46-50hrs	63,113	60,786	25.78	25.01	15,082,967
>70hrs	58,975	60,786	15.23	14.65	1,703,322
41-45hrs	57,016	56,733	25.78	25.62	11,213,040
40hrs	41,658	41,537	21.65	21.02	65,989,151
36-39hrs	33,587	34,445	19.18	19.00	4,398,507
31-35hrs	25,771	25,327	16.90	15.88	6,292,651
26-30hrs	18,266	18,641	14.90	13.96	6,114,150
21-25hrs	14,576	15,196	15.36	14.34	3,785,947
16-20hrs	11,198	11,651	16.19	14.81	4,657,728
11-15hrs	8,132	8,105	17.99	16.23	1,702,746
06-10hrs	5,434	5,167	20.66	20.29	1,526,047
01-5hrs	3,516	3,039	40.19	38.01	663,501

This variable asks for the usual hours worked per week during weeks which the person worked during in the past 12 months. The distribution for hours worked per week shows an extremely high peak at 40 hours per week. Also, people report hours worked per week in intervals of 5, so it makes sense to divide the variable into groups of 5 hours per week. Noteworthy is that earnings peak at 41-55 hours per week and decline with more hours worked per week.

2.15 - SCHL - Educational Attainment

Table 19: Geometric Mean and Median Earnings per Year/Hour split by Educational Attainment

	Earnings Per Year		Earnings Per Hour		
Educational Attainment	Log-Average	Median	Log-Average	Median	# of Workers
Professional	96,699	101,310	47.04	47.22	3,276,119
Doctorate	82,293	91,179	40.92	42.01	2,156,395
Master's	63,124	70,917	33.25	34.06	13,880,071
Bachelor's	50,492	55,720	27.16	27.35	31,546,030
Associate's	36,188	40,524	20.11	20.29	13,009,335
Some college	32,664	36,471	18.40	18.51	27,483,844
HS or equivalent	28,837	31,406	16.33	16.28	31,911,371
Less than HS diploma	22,699	25,327	13.71	13.40	11,093,740

For this variable, we grouped together High School Diploma and GED. We also grouped together all educational attainment less than high school completion.

2.16 - HISP - Hispanic Origin

Table 20: Geometric Mean and Median Earnings per Year/Hour split by Hispanic origin

	Earnings Per Year		Earnings Per Hour		
Hispanic origin	Log-Average	Median	Log-Average	Median	# of Workers
Not Hispanic	40,489	45,589	22.35	21.98	111,059,186
Hispanic,not Mexican	31,167	32,419	17.67	17.10	9,435,290
Mexican	28,986	30,393	16.13	15.63	13,862,429

For this varaible, we grouped together all people with Hispanic origin not from Mexico, as the size of each individual group is small. Hispanic origin refers to people that originate from Spain or Spanish-speaking countries in Latin America.

2.17 - JWMNP - Travel Time to Work

Table 21: Geometric Mean and Median Earnings per Year/Hour split by Travel Time to Work

	Earnings Per Year		Earnings Per Hour		
Travel Time to Work	Log-Average	Median	Log-Average	Median	# of Workers
61-80min	59,100	61,799	28.74	29.31	1,911,157
51-55min	56,581	60,786	27.67	27.68	592,016
46-50min	52,440	55,720	26.08	25.89	2,083,280
81-100min	52,360	55,720	26.18	26.05	1,929,830
31-35min	48,812	50,655	24.39	24.37	3,877,988
>100min	48,213	50,655	23.88	23.76	1,835,127
36-40min	47,234	50,655	23.86	23.45	5,168,736
56-60min	47,005	50,655	23.74	23.84	6,257,914
41-45min	46,903	50,655	23.56	23.45	7,990,734
21-25min	44,509	46,501	22.73	21.98	8,274,958
26-30min	41,620	43,563	21.48	21.00	16,978,515
16-20min	40,314	41,537	21.01	20.32	17,293,115
11-15min	38,203	40,524	20.19	19.54	18,354,916
06-10min	35,803	38,498	19.25	19.00	15,778,930
02-5min	33,289	35,458	18.21	17.59	9,141,624
00-01min	23,223	26,341	20.00	19.54	16,888,065

This variable asks for time to get from home to work during the reference week. This refers to a one-way commute time. Since people generally report travel time to work in intervals of 5 minutes, as seen in the histogram, we divided the variable into groups of 5 minutes. Worth noting is that once the one-way travel time to work surpasses 70 minutes, earnings decline on averge.

2.18 - PUMA - Area

Table 22: Geometric Mean and Median Earnings per Year/Hour split by Area

	Earnings Per Year		Earnings Per Hour		
Area	Log-Average	Median	Log-Average	Median	# of Workers
95%-100%	69,364	79,022	36.72	38.01	7,595,362
90%-95%	55,643	60,786	29.88	30.39	7,274,621
85%-90%	50,765	55,720	27.45	27.35	7,400,589
80%-85%	47,152	51,668	25.73	25.79	7,114,708
75%-80%	44,649	50,655	24.29	24.42	7,620,223
70%-75%	42,551	48,629	23.28	23.45	7,324,310
65%-70%	40,753	45,589	22.37	21.98	7,467,986
60%-65%	39,186	43,057	21.57	21.51	6,887,406
55%-60%	37,832	40,524	20.73	20.52	7,041,478
50%-55%	36,650	40,524	20.05	19.54	6,898,751
45%-50%	35,612	40,524	19.80	19.54	6,718,496
40%-45%	34,676	38,498	19.18	19.39	6,797,613
35%-40%	33,767	37,485	18.70	18.81	6,538,748
30%-35%	32,922	36,471	18.43	18.26	6,457,135
25%-30%	31,991	35,458	17.89	17.76	6,099,800
20%-25%	31,090	34,445	17.63	17.37	6,284,027
15%-20%	30,186	33,432	17.05	17.10	5,891,122
10%-15%	29,181	32,419	16.69	16.57	5,894,512
5%-10%	27,631	30,393	15.99	15.63	5,664,087
0%-5%	24,669	27,354	14.72	14.65	5,385,931

Public Use Microdata Areas are geographic units that contain at least 100,000 people. There are 2,378 PUMAs for the 2018 ACS, based on the 2010 US Census. This is the only area data provided in the survey that is more specific than the state the person lives in. We sorted the PUMAs by log-average earnings and divided into 20 groups of equal number of PUMAs. That means that each 5% group contains about 119 groups.

2.19 - AGEP - Age

Table 23: Geometric Mean and Median Earnings per Year/Hour split by Age

	Earnings Per Year		Earnings Per Hour		
Age	Log-Average	Median	Log-Average	Median	# of Workers
45-54	44,006	48,021	23.40	23.20	33,681,825
55-59	42,595	45,589	23.40	22.96	15,789,447
40-44	42,070	45,589	22.57	21.98	16,612,742
38-39	40,348	44,576	21.99	21.71	6,980,727
36-37	39,657	42,854	21.65	21.49	7,190,774
60-64	38,500	42,550	23.29	22.67	12,384,618
34-35	37,576	40,524	20.61	20.35	7,338,266
32-33	35,437	40,524	19.56	19.54	7,310,124
31	33,997	38,498	18.84	19.54	3,601,001
30	32,518	36,066	18.18	18.37	3,956,204
29	31,252	35,458	17.65	17.59	3,776,957
28	29,358	32,926	16.78	17.10	3,951,336
27	27,763	30,393	16.17	16.27	3,939,932
26	25,869	30,393	15.45	15.63	3,928,752
25	24,077	28,367	14.83	14.72	3,914,200

For the age variable, we arbitrarily divided so that there are more groups among young adults. From age 25-35, the log-average earnings per hour quickly grow. Therefore, it may better to have more predictors for this range of ages. For age 60-64, the median hourly earnings is the same as age 55-59, but the median annual earnings decline. This simply indicates that past the age of 60, workers on average start to work less.

2.20 - FOD1P - Field of Degree

Table 24: Geometric Mean and Median Earnings per Year/Hour split by Field of Degree

	Earnings Per Year		Earnings Per Hour		
Field of Degree	Log-Average	Median	Log-Average	Median	# of Workers
Engineering	80,142	91,179	39.38	41.52	5,012,472
Computer	75,109	87,126	37.82	40.90	2,026,368
Physical Sciences	68,526	75,982	35.21	35.66	1,468,815
Math	68,492	75,982	35.94	37.12	667,366
Biology	68,341	70,917	35.23	34.19	2,550,599
Business	62,597	67,877	31.54	31.67	10,953,367
Social Science	61,976	65,851	32.54	31.75	3,651,504
Health	56,999	65,851	32.10	33.70	4,043,226
Agriculture	52,771	58,760	26.18	26.87	508,959
Environment	50,354	55,720	26.58	26.66	357,001
Other	49,030	53,694	26.33	26.05	7,941,330
Language	47,530	53,694	27.19	27.35	1,905,274
Psychology	47,062	50,655	26.50	26.05	2,512,105
Multiple	46,939	52,681	26.24	26.05	425,857
Education	42,248	50,655	23.76	24.23	4,654,112
Art	40,312	46,096	23.87	24.23	2,180,260
NoDegree	30,152	33,432	17.14	17.10	83,498,290

Table 25: Geometric Mean and Median Earnings per Year/Hour split by Field of Degree

	Earnings Per Year		Earnings Per Hour		
Field of Degree	Log-Average	Median	Log-Average	Median	# of Workers
95%-100%	82,856	93,205	40.89	43.38	7,475,566
90%-95%	71,139	78,008	36.04	36.64	6,307,131
85%-90%	61,937	67,877	31.39	31.75	9,549,762
80%-85%	58,545	65,851	31.78	33.22	3,661,379
75%-80%	52,756	58,760	28.00	28.33	6,696,451
70%-75%	47,737	52,681	26.63	26.64	6,714,735
65%-70%	43,612	50,655	24.40	24.42	6,953,196
60%-65%	30,441	34,243	17.33	17.15	86,998,685

For the field of degree, there are two ways to divide this variable. We can divide into arbitrary major groups or by 5% quantiles. Only about 40% of workers get a degree, so that is why there are only 8 5% groups. The 60-65% group includes all worker without a degree plus the lowest earning degrees (this is due to the way we wrote the function).

III. Building the Regression Model

For WKHP,AGEP,JWMNP,JWAP,OCCP,NAICSP,PUMA, and FOD1P, there are multiple options to consider. We first add the other variables first then consider the variables with multiple options one by one. We generally choose the model with the lowest BIC, since it has the largest penalty per parameter.

The model with the lowest AIC, lowest BIC, and highest adjusted R-squared is listed in bold for each table.

3.1 - Adding WKW,WRK,MAR,ENG,HISP,JWTR,RAC3P,HISP,SEX, and SCHL to Earnings per Year Model

Table 26: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde

	AIC	BIC	Adjusted R-squared	df
WKW	**5466682**	**5466754**	**0.26538**	6
WRK	5754942	5754978	0.07979	3
MAR	5815795	5815856	0.03498	5
MARHT	5829073	5829121	0.02491	4
ENG	5835024	5835084	0.02037	5
JWTR	5744404	5744549	0.08734	12
QTRBIR	5861332	5861380	0.00002	4
RAC3P	5835050	5835159	0.02035	9
HISP	5842569	5842605	0.01457	3
SEX	5815207	5815231	0.03542	2
SCHL	5682367	5682463	0.13052	8

We add all of the variables in the list except for MARHT and QTRBIR. We overlooked this when choosing our initial variables, but MARHT is linearly dependent with MAR, since both of them contain the same category "Never Married". Since, MAR is a much better predictor of MARHT, we will not consider MARHT in the model at all. Meanwhile, the quarter of birth has no effect on predicting income at all so it will not be included in the regression model. All of the other variables are significant and will be included in the model.

3.2 - Adding WKHP to model with PERNP

Table 27: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL

	AIC	BIC	Adjusted R-squared	df
Null	5121121	5121664	0.43923	45
+WKHPF	4801236	4801971	0.56326	61
+factor(WKHP)	**4796434**	**4797822**	**0.56491**	115
+WKHP	4867817	4868372	0.53993	46
+WKHP+I(WKHP**2)	4807055	4807622	0.56126	47
+WKHP+I(WKHP**2)+I(WKHP**3)	4803866	4804445	0.56236	48
+sqrt(WKHP)+WKHP	4821226	4821793	0.55638	47
+sqrt(WKHP)+WKHP+I(WKHP**2)	4804960	4805539	0.56198	48

This is the only one where we do not choose the model with the lowest BIC. The summary plots showed that people generally choose to report the number of hours worked per week in intervals of 5. Few people report their hours worked per week that is not in intervals of 5 and are a biased sample. Therefore, we believe that a factor for every single hour per week will lead to overfitting. The real problem is that hours worked per week is not a smooth distribution since people estimate their hours worked per week rather than measuring it. So we choose the next best model instead, which are the same factors from the summary tables.

The other tables can be found in the section "Appendix 2 - Other Tables for Building the Regression Model".

3.3 - Adding AGEP to model with PERNP

We add "+sqrt(AGEP)+AGEP" to the model.

3.4 - Adding JWMNP to model with PERNP

We add "+JWMNPF" to the model, which are the same factors from summary table.

3.5 - Adding JWAP to model with PERNP

We add "+JWAPF" to the model, which are the same factors from summary table.

3.6 - Adding OCCP to model with PERNP

We add OCCP split into 40 equal groups to the model. The summary table had OCCP split into 20 equals groups. So we just split OCCP into more groups compared to the summary table. It is noteworthy that even splitting OCCP into 2 equal groups is nearly as good of a model as splitting OCCP into the 17 major employment groups.

3.7 - Adding NAICSP to model with PERNP

We add NAICSP split into 50 equal groups to the model. The summary table had NAICSP split into 20 equals groups. So we just split NAICSP into more groups compared to the summary table.

3.8 - Adding PUMA to model with PERNP

We add PUMA split into 40 equal groups to the model. The summary table had PUMA split into 20 equals groups. So we just split PUMA into more groups compared to the summary table.

3.9 - Adding FOD1P to model with PERNP

We add FOD1P split into 50 equal groups to the model. The summary table had FOD1P split into 20 equals groups. So we just split FOD1P into more groups compared to the summary table.

Moving on to Earnings per Hour

We'll just provide a summary of this section here, but it very similar to building the model with just earnings per year. We note that WKW, WRK, and WKHP lose most, but not all of the explanatory power when switching to earnings per hour. The model with WKW, WRK, MAR, ENG, JWTR, RAC3P, HISP, SEX, SCHL, WKHP drops from an adjusted R-squared of 56.49% to just 26.7%.

When we split up OCCP,NAICSP,PUMA, and FOD1P into equal groups, we are now sorting by average log transformed earnings per hour instead of average log transformed earnings per year. Therefore, these equal groups will be different from the ones in the earnings per year model. We refer to the variables split up by earnings per hour with an "H" after the variable name and the variables split up by earnings per year with a "Y" after the variable name.

3.10 - Adding WKW, WRK, MAR, ENG, JWTR, RAC3P, HISP, SEX, SCHL to model with PERNPHR

Table 28: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNPHR tilde

	AIC	BIC	Adjusted R-squared	df
WKW	5178542	5178614	0.01261	6
WRK	5186725	5186761	0.00628	3
MAR	5143490	5143550	0.03929	5
MARHT	5157981	5158030	0.02835	4
ENG	5161546	5161606	0.02564	5
JWTR	5174498	5174643	0.01573	12
QTRBIR	5194763	5194811	0.00002	4
RAC3P	5158801	5158910	0.02773	9
HISP	5165908	5165944	0.02231	3
SEX	5173936	5173960	0.01616	2
SCHL	**4955853**	**4955949**	**0.17031**	8

Adding WKHP to model with PERNPHR

Table 29: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL

	AIC	BIC	Adjusted R-squared	df
Null	4836279	4836822	0.24434	45
+WKHPF	4800455	4801191	0.26521	61
+factor(WKHP)	**4796553**	**4797940**	**0.26748**	115
+WKHP	4836280	4836835	0.24434	46
+WKHP+I(WKHP**2)	4833488	4834055	0.24599	47
+WKHP+I(WKHP**2)+I(WKHP**3)	4806367	4806946	0.26180	48
+sqrt(WKHP)+WKHP	4836210	4836777	0.24438	47
+sqrt(WKHP)+WKHP+I(WKHP**2)	4803830	4804409	0.26326	48

Again, we choose the model with "WKHPF" instead of "factor(WKHP)", even though factor(WKHP) has a lower BIC value.

Adding AGEP to model with PERNPHR

Adding JWMNP to model with PERNPHR

Adding OCCP to model with PERNPHR

Adding NAICSP to model with PERNPHR

Adding PUMA to model with PERNPHR

Adding FOD1P to model with PERNPHR

The corresponding tables listing the AIC,BIC, and adjusted R-squared of the considered models can be found in "Appendix 2 - Other Tables for Building the Regression Model" ##3.11 - Other Variations to PERNP

Table 30: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+ JWMNPF+JWAPF

	AIC	BIC	Adjusted R-squared	df
+OCCPY+NAICSPY+PUMAY+FOD1PY	4522757	4525688	0.64872	243
+OCCPH+NAICSPH+PUMAH+FOD1PH	4517096	4520100	0.65027	249
+OCCPY+NAICSPY+PUMAH+FOD1PY	4518470	4521401	0.64989	243
+OCCPH+NAICSPY+PUMAY+FOD1PY	4519640	4522656	0.64957	250
+OCCPH+NAICSPY+PUMAH+FOD1PY	**4515492**	**4518508**	**0.65071**	250

We also considered different variations to the PERNP model where OCCP,NAICSP,PUMA, and FOD1P are sorted by either year or hour. We found that OCCP and PUMA sorted by hour and NAICSP and FOD1P sorted by year produced the best model for the PERNP model.

3.12 - Other Variations to PERNPHR

Table 31: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+ JWMNPF+JWAPF

	AIC	BIC	Adjusted R-squared	df
+OCCPY+NAICSPY+PUMAY+FOD1PY	4521700	4524631	0.40911	243
+OCCPH+NAICSPH+PUMAH+FOD1PH	4516014	4519018	0.41173	249
+OCCPY+NAICSPY+PUMAH+FOD1PY	4517385	4520316	0.41110	243
+OCCPH+NAICSPY+PUMAY+FOD1PY	4518567	4521582	0.41056	250
+OCCPH+NAICSPY+PUMAH+FOD1PY	**4514391**	**4517407**	**0.41248**	250

What is surprising is that both models are actually the same. Sorting occupations and area of residence by earnings per hour is the best model regardless of whether we are looking earnings per hour and earnings per year. Sorting

industry and field of degree by earnings per year turns out to be the best model. This may not be surprising however, since we do adjust for hours worked and weeks worked in both models.

So in the end, we have the same variables included in both models. The AIC and BIC are nearly the same. The only real difference between the two models are the hours worked per week and weeks worked parameters.

IV - Final Regression Model for Earnings and Earnings per Year and Conclusions for Regression

4.1 - Final Model - Add SEX:log2(AGEP) and SCHL:log2(AGEP) and SEX:MAR

We consider three interaction effects in the model. All three are highly significant for both models. Since interaction terms increase the width of the confidence intervals drastically and lowers our level of confidence of the parameters, we won't consider any other interaction terms other than these three.

1) Interaction between Sex and Age
2) Interaction between Educational Attainment and Age
3) Interaction between Sex and Marital Status

We know that most women have children and then they have to take time off work and thus there is a gender wage gap. We don't have variable for how work experience commpared to age each worker has, so we can only use the interaction effect to mimic the fact that the gender wage gap gets larger with age. Also, we know the returns to education vary with education and age. Highly educated workers have a higher ceiling compared to lowly educated workers. And then marital status affects men and women differently, with single women likely more career oriented compared to single men.

We chose log2(AGEP) to be the interaction effect for age, because it turned out be a better fit than sqrt(AGEP). The problem with sqrt(AGEP) may be that the slope may be too low in the range age 25-64. We want a function that intersects the y-axis at approximately 25, since the age range of our dataset is age 25-64. The problem with the interaction terms is that the SCHL and SEX coefficients are not interpretable without using the interaction terms, since no one is age 0 in our dataset.

Table 32: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+ JWMNPF+JWAPF+OCCPH+NAICSPY+PUMAH+FOD1PY

	AIC	BIC	Adjusted R-squared	df
+SEX:log2(AGEP)+SCHL:log2(AGEP)+SEX:MAR	**4506884**	**4510056**	**0.41592**	263

Table 33: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+ JWMNPF+JWAPF+OCCPH+NAICSPY+PUMAH+FOD1PY

	AIC	BIC	Adjusted R-squared	df
+SEX:log2(AGEP)+SCHL:log2(AGEP)+SEX:MAR	**4507943**	**4511115**	**0.65276**	263

This is the adjusted R-squared and BIC for the final model.

We found it suprising that we could only explain 41.5% of the variation in earnings per hour with 263 variables. It turns out earnings per hour is extremely complex and there are many unobservable characteristics in people's

incomes. We predict at least 50% of variation in earnings per hour is due to unobservable characteristics. Another problem is that people's reported hours per week, commute time, and weeks worked per year are estimated rather than measured. These three variables, which are very important, may be inaccurate or unreliable, leading to a poorer model.

4.2 - Final Model - Description

Table 34: Final Model

Variable	Model for both Earnings and Earnings per Year	Df
WKW	Same Factors as Summary Statistics	5
WRK	Same Factors as Summary Statistics	2
MAR	Same Factors as Summary Statistics	4
ENG	Same Factors as Summary Statistics	4
JWTR	Same Factors as Summary Statistics	11
RAC3P	Same Factors as Summary Statistics	8
HISP	Same Factors as Summary Statistics	2
SEX	Same Factors as Summary Statistics	1
SCHL	Same Factors as Summary Statistics	7
WKHP	Same Factors as Summary Statistics	15
AGEP	Linear variable of age plus Square Root variable of age	2
JWMNP	Same as Summary Statistics	15
JWAP	Same as Summary Statistics	28
OCCP	50 equal groups sorted by Hourly Earnings	46
NAICSP	50 equal groups sorted by Yearly Earnings	41
PUMA	40 equal groups sorted by Hourly Earnings	39
FOD1P	50 equal groups sorted by Yearly Earnings	18
SEX:AGEP	Interaction between Sex and log base 2 of Age	2
SCHL:AGEP	Interaction between School and log base 2 of Age	7
SEX:MAR	Interaction between Sex and Marital Status	4
MARHT	Not included due to overlap with MAR	0
QTRBIR	Not included due to insignificance	0

This table describes all 20 variables and 3 interactions added to the model, as well as how many degrees of freedom each variable involves.

4.3 - ANOVA table

Table 35: ANOVA table for Hour model

Variable	Df	Sum Sq	Mean Sq	F value	Pr(>F)
WKW	5	935329	187066	5529.03	< 2.2e-16
WRK	2	168106	84053	2484.33	< 2.2e-16
MAR	4	2725639	681410	20140.17	< 2.2e-16
ENG	4	1957935	489484	14467.48	< 2.2e-16
JWTR	11	704907	64082	1894.06	< 2.2e-16
RAC3P	8	1240957	155120	4584.81	< 2.2e-16
HISP	2	299907	149953	4432.11	< 2.2e-16
SEX	1	1120846	1120846	33128.42	< 2.2e-16
SCHL	7	8961634	1280233	37839.36	< 2.2e-16
WKHPF	15	1541710	102781	3037.85	< 2.2e-16
sqrt(AGEP)	1	1107056	1107056	32720.82	< 2.2e-16
AGEP	1	240888	240888	7119.84	< 2.2e-16
JWMNPF	15	426295	28420	839.99	< 2.2e-16
JWAPF	28	146633	5237	154.78	< 2.2e-16
OCCPH	46	6800857	147845	4369.79	< 2.2e-16
NAICSPY	41	864784	21092	623.42	< 2.2e-16
PUMAH	39	1238646	31760	938.72	< 2.2e-16
FOD1PY	18	103683	5760	170.25	< 2.2e-16
SEX:log2(AGEP)	2	53288	26644	787.51	< 2.2e-16
SCHL:log2(AGEP)	7	117684	16812	496.91	< 2.2e-16
MAR:SEX	4	84590	21147	625.05	< 2.2e-16
Residuals	1279505	43289974	34	NA	NA

Table 36: ANOVA table for Year model

Variable	Df	Sum Sq	Mean Sq	F value	Pr(>F)
WKW	5	33118711	6623742	195613.22	< 2.2e-16 **
WRK	2	179922	89961	2656.74	< 2.2e-16 **
MAR	4	3053517	763379	22544.22	< 2.2e-16 **
ENG	4	2235318	558830	16503.43	< 2.2e-16 **
JWTR	11	764946	69541	2053.68	< 2.2e-16 **
RAC3P	8	1291719	161465	4768.40	< 2.2e-16 **
HISP	2	312137	156068	4609.03	< 2.2e-16 **
SEX	1	3209667	3209667	94788.30	< 2.2e-16 **
SCHL	7	10651444	1521635	44937.12	< 2.2e-16 **
WKHPF	15	15472293	1031486	30461.99	< 2.2e-16 **
sqrt(AGEP)	1	1096328	1096328	32376.89	< 2.2e-16 **
AGEP	1	244510	244510	7220.89	< 2.2e-16 **
JWMNPF	15	425647	28376	838.02	< 2.2e-16 **
JWAPF	28	150621	5379	158.86	< 2.2e-16 **
OCCPH	46	6799502	147815	4365.30	< 2.2e-16 **
NAICSPY	41	859423	20962	619.04	< 2.2e-16 **
PUMAH	39	1245548	31937	943.17	< 2.2e-16 **
FOD1PY	18	103200	5733	169.32	< 2.2e-16 **
SEX:log2(AGEP)	2	53755	26878	793.75	< 2.2e-16 **
SCHL:log2(AGEP)	7	118041	16863	498.00	< 2.2e-16 **
MAR:SEX	4	85426	21356	630.70	< 2.2e-16 **
Residuals	1279505	43325860	34	NA	NA

The table shows that the Sum of Squared Errors is nearly the same for the Hour model and the Year model except for WKHP and WKW. The MSE of Weeks Worked drops 97% in the hour model compared to the year model. The MSE of hours worked per week drops 90% in the hour model compared to the year model.

We found that sex has a high MSE, nearly as high as educational attainment. We found the gender earnings gap to be large in our model. Our adjusted gender wage gap is actually as large as the unadjusted gender gap found in the summary tables in Part 2.13. It may be that our occupational data is not specific enough to adjust the gender earnings gap properly.

4.4 - Constant Variance with Year Model vs Hour Model

```
#car::ncvTest(Year.model)
"Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 35525.12, Df = 1, p = < 2.22e-16"
#lmtest::bptest(Year.model)
"   studentized Breusch-Pagan test

data:  Year.model
BP = 75400, df = 261, p-value < 2.2e-16"
#car::ncvTest(Hour.model)
"Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 60.62692, Df = 1, p = 6.8984e-15"
#lmtest::bptest(Hour.model)
"   studentized Breusch-Pagan test

data:  Hour.model
BP = 76349, df = 261, p-value < 2.2e-16"
```

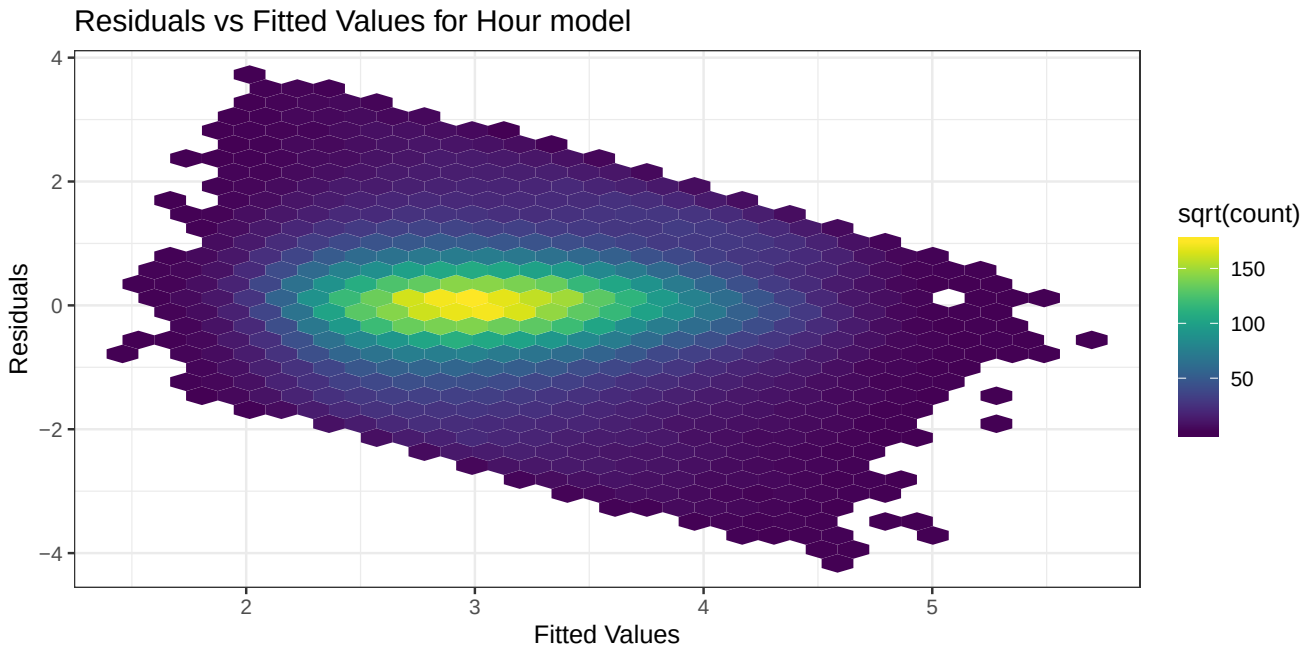
We believed from the summary plots that the Hour model would meet the assumption of homoskedascity more than the Year model. The NCV test looks at the fitted values and does find the Hour model to be closer to constant variance. However, the BP test, which looks at the explanatory variables, finds the two models to perform similarly. This is not exactly suprising since the explanatory variables of the two models are the same. Focusing on the fitted values, the Hour model does perform better with constant variance than the Year model.

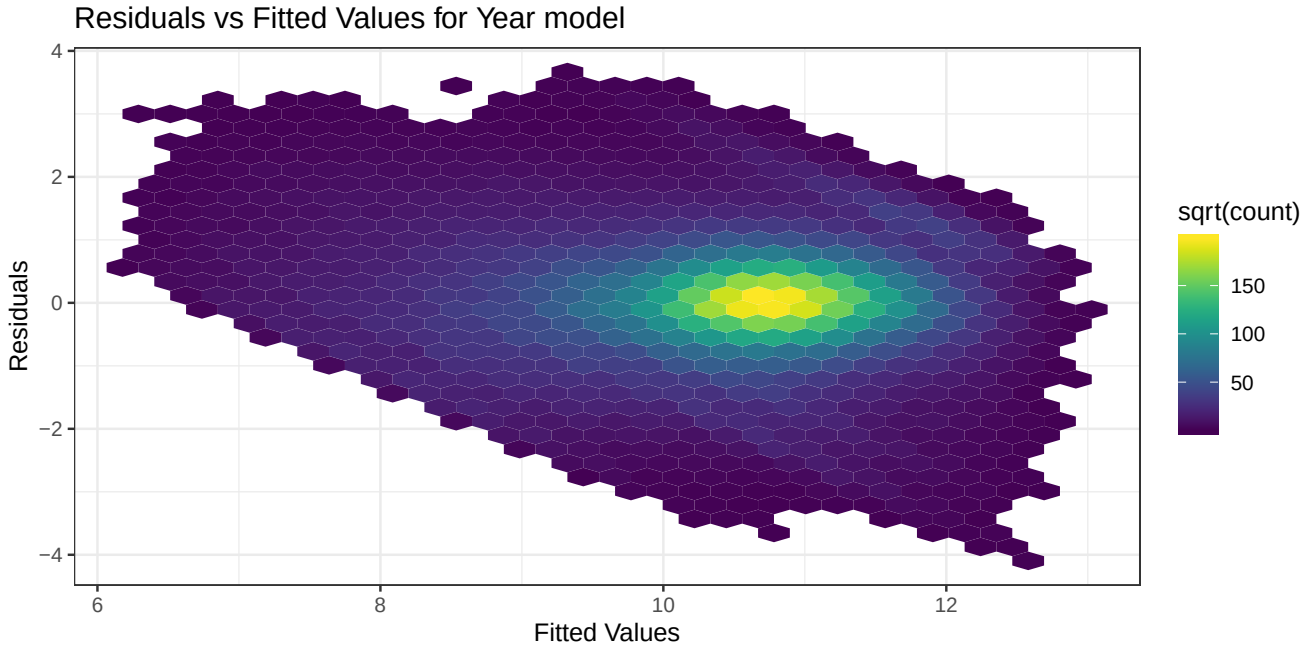
Confidence Intervals for R-squared

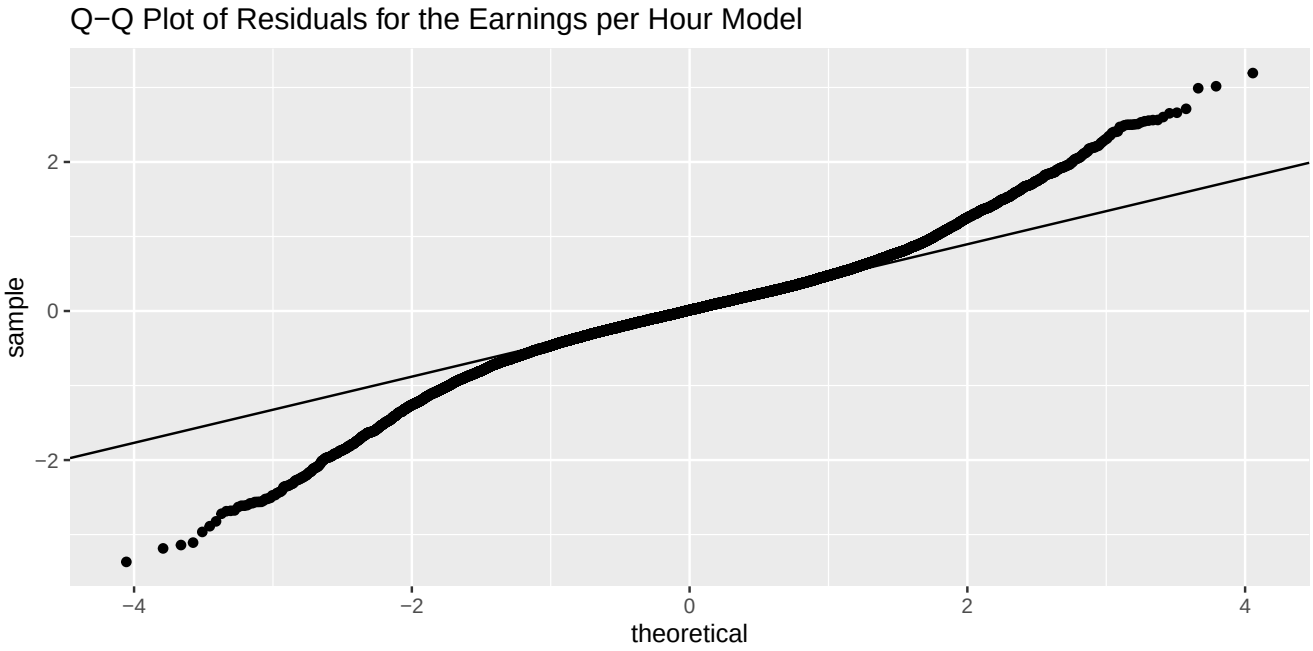
	Estimate	99% Confidence Interval Lower Bound	Upper Bound
Earnings per Year	0.6527600	0.6507531	0.6547668
Earnings per Hour	0.4159178	0.4135982	0.4182373

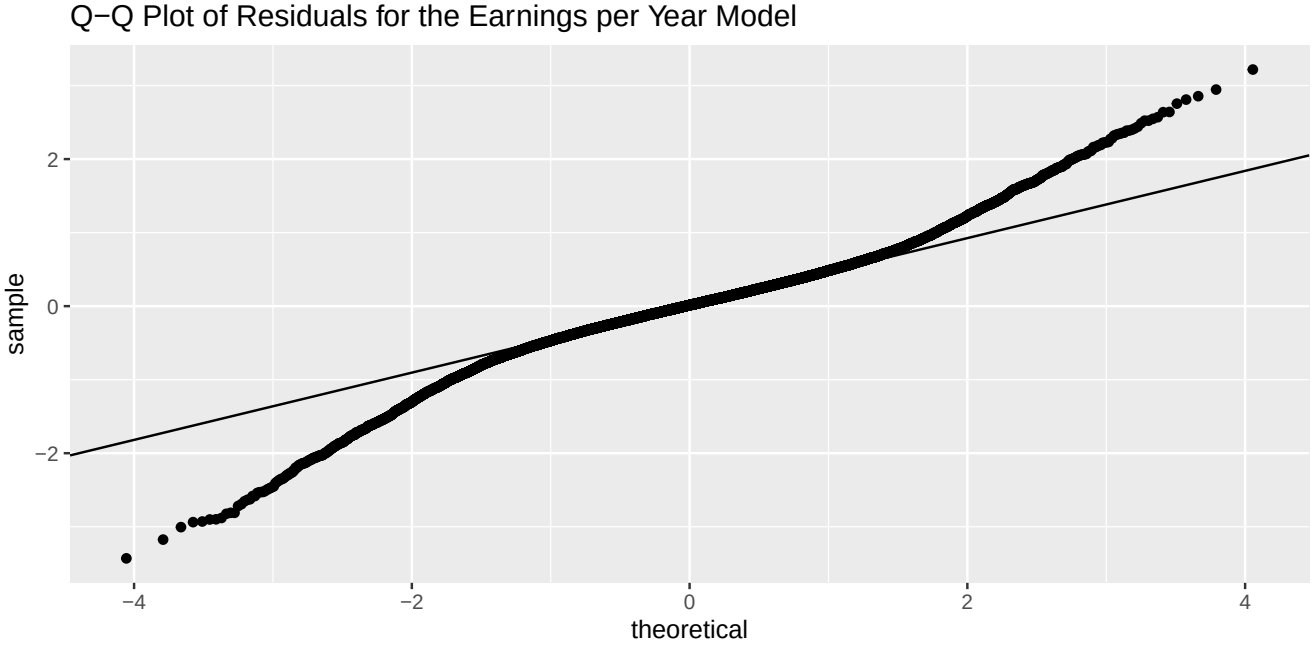
The 99% confidence interval for adjusted R-squared for the Hour model is (41.36% - 41.82%). The 99% confidence interval for adjusted R-squared for the Year model is (65.08% - 65.48%). The difference between adjusted R-squared and R-squared for both models is only about 0.01%, so they are almost the same.

4.5 - Residual Plot and QQ-Plots









The residual vs fitted values plot show a slanted rectangle since we excluded some of the higher and lower end outliers. There is not much to see in the residual vs fitted values plot, but it does give an idea of what the range of the error is based on the fitted value. There are many values with a residual of 1 or more, which is a huge amount. A residual of positive 1 means the individual earns 271% of what was predicted and a residual of -1 means the individual earns 37% of what was predicted. Despite that it appears there is a negative trend line in the residuals vs fitted values, further evaluation with lm() shows that there is actually a small positive trend line.

The QQ-plot for both the Hour model and Year model show a heavy tailed distribution on both sides. We only expected to see a heavy tail on the upper end, not on the lower end. Why do we see a heavy tail on the lower end too? We believe that this may be because we only counted earned income in our model. We did not take into account income transferred by the government. Many poorer people rely on the government for income, so if we counted that in our model, the heavy tail in the lower end may disappear.

4.6 - 99% Confidence Intervals for Hispanic origin, English Literacy, Commute Mode, Educational Attainment, and Marital Status

99% Confidence Intervals for HISP relative to Not Hispanic		
	Earnings per Hour	Earnings per Year
Not Hispanic	Baseline	Baseline
Mexican	(99.15%, 100.45%)	(99.13%, 100.45%)
Hispanic,not Mexican	(96.65%, 98.43%)	(96.66%, 98.42%)

Our model says that there is no difference in earnings between Mexican Americans and Non-Hispanic Americans once observable characteristics are taken in acccount. However, those from Spanish speaking countries other than Mexico earn less for some reason. A large percent of these are from Puerto Rico and other Carribean islands, which may have some unobservable disadvantage for some reason compared to the rest of Americans.

99% Confidence Intervals for ENG relative to Only English		
	Earnings per Hour	Earnings per Year
Only English	Baseline	Baseline
Very well	(97.3%, 98.6%)	(97.3%, 98.6%)
Well	(90.2%, 91.7%)	(90.2%, 91.7%)
Not well	(86.1%, 88.0%)	(86.0%, 88.0%)
Not at all	(84.1%, 87.0%)	(84.2%, 87.0%)

For some reason, multilingual Americans have an unobservable disadvantage compared to Americans that speak only English. Multilingualism is usually thought to be an advantage compared to monolingualism. One possible reason is that those who claim to speak English very well are still not completely fluent in English.

99% Confidence Intervals for JWTR relative to Not at work		
	Earnings per Hour	Earnings per Year
Not at work	Baseline	Baseline
Car/truck/van	(102.7%, 108.7%)	(103.1%, 109.0%)
Bus	(98.4%, 104.7%)	(98.8%, 104.9%)
Streetcar	(100.6%, 117.8%)	(100.7%, 118.2%)
Subway	(112.2%, 119.6%)	(112.8%, 120.0%)
Railroad	(115.1%, 122.6%)	(115.6%, 123.0%)
Ferryboat	(119.4%, 139.7%)	(119.8%, 140.1%)
Taxi	(99.7%, 113.7%)	(100.1%, 113.8%)
Motorcycle	(100.4%, 108.7%)	(100.7%, 108.9%)
Bicycle	(96.6%, 105.0%)	(97.0%, 105.3%)
Walked	(97.3%, 103.9%)	(97.7%, 104.2%)
Worked at home/Other	(105.0%, 108.6%)	(105.4%, 108.9%)

Those who take the subway, railroad, or ferryboat to work earn more than those who drive to work.

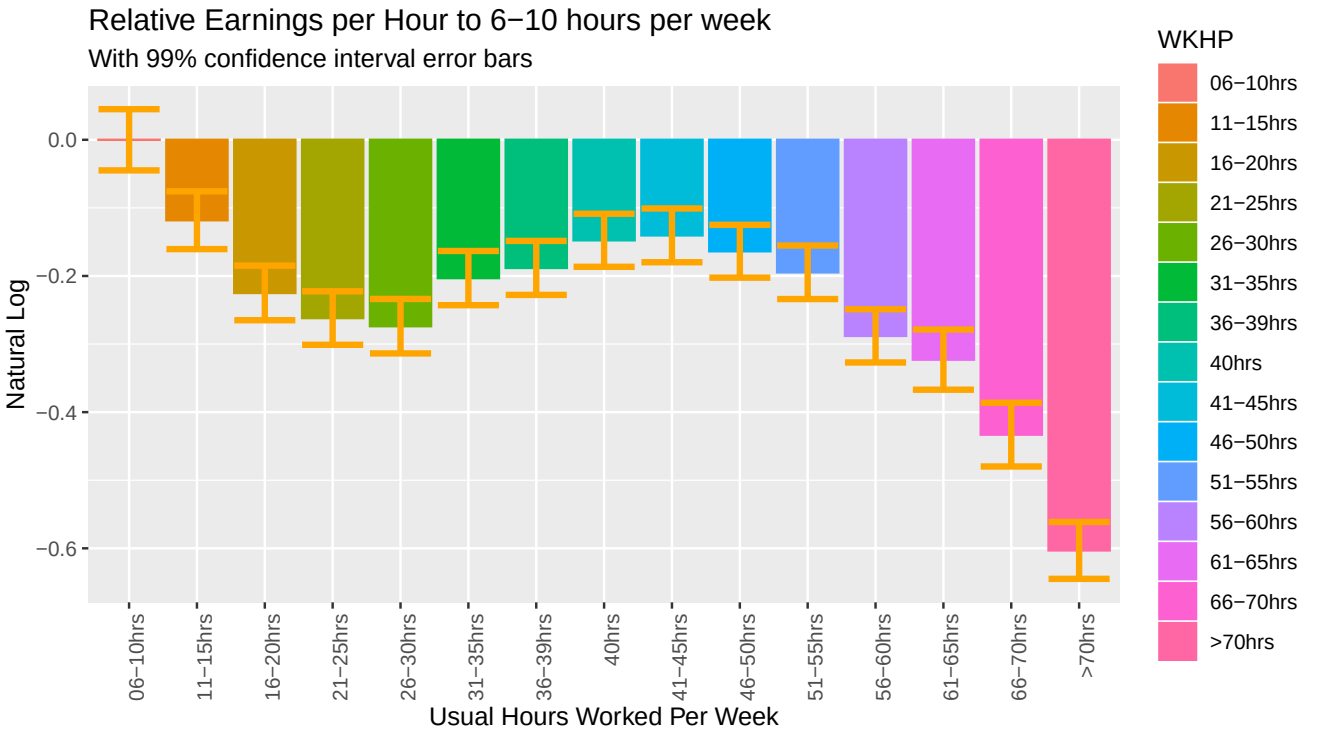
99% Confidence Intervals for Educational Attainment relative to less than HS for Hour model				
	Age 30	Age 40	Age 50	Age 60
Less than HS	Baseline	Baseline	Baseline	Baseline
HS or equivalent	(93.4%, 116.5%)	(94.2%, 117.7%)	(94.9%, 118.7%)	(95.5%, 119.5%)
Some college	(95.7%, 118.0%)	(98.3%, 121.3%)	(100.4%, 124.0%)	(102.1%, 126.2%)
Associate's	(97.1%, 119.6%)	(99.9%, 123.1%)	(102.1%, 126.0%)	(103.9%, 128.3%)
Bachelor's	(99.4%, 124.8%)	(104.8%, 131.8%)	(109.2%, 137.4%)	(112.9%, 142.3%)
Master's	(108.4%, 138.6%)	(116.8%, 149.5%)	(123.7%, 158.5%)	(129.7%, 166.3%)
Professional	(93.1%, 142.4%)	(110.1%, 168.9%)	(125.4%, 192.8%)	(139.6%, 214.9%)
Doctorate	(94.8%, 154.4%)	(108.1%, 176.6%)	(119.6%, 195.9%)	(130.0%, 213.3%)

Because of the interaction effect, the confidence intervals are very wide for educational attainment. But we can still get the general idea that those with high education get a growing advantage later in life compared to those of low education.

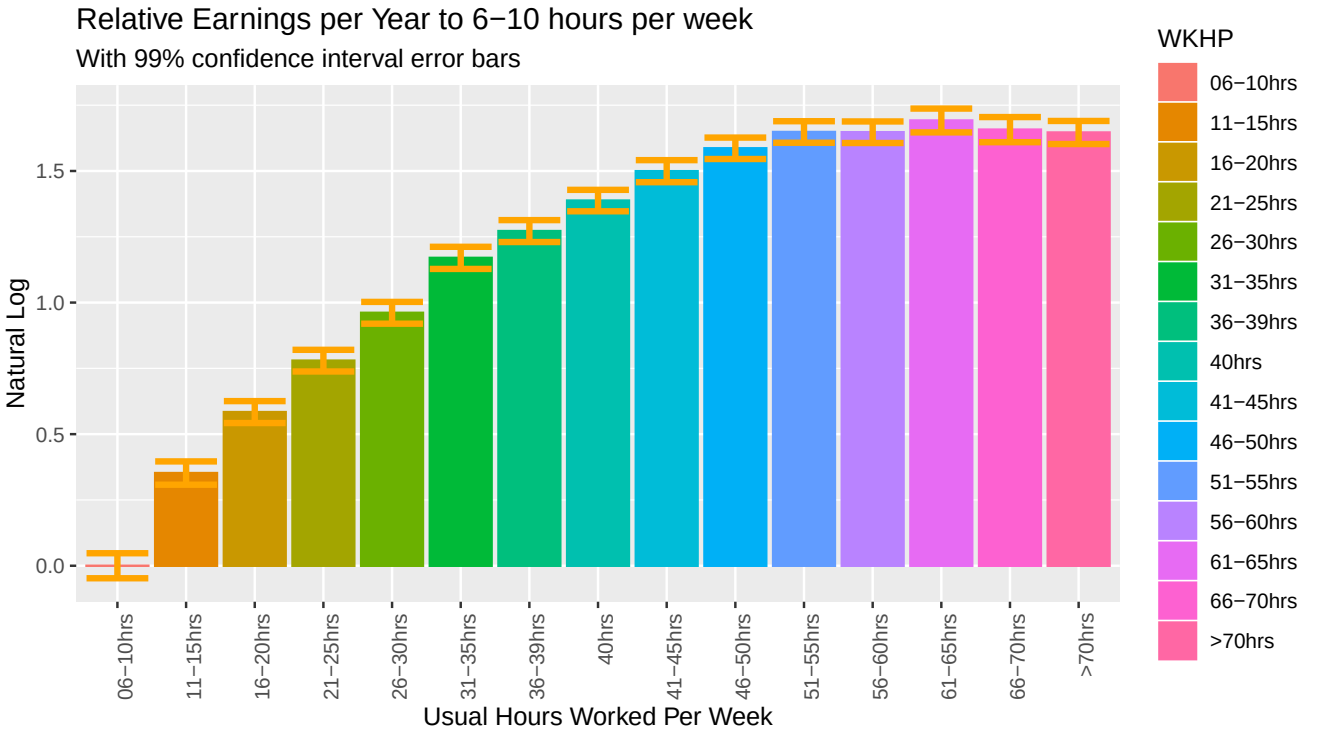
99% Confidence Intervals for Marital Status relative to Married Men for Hour model		
	Male	Female
Married	Baseline	(83.49%, 90.87%)
Widowed	(86.7%, 91.3%)	(81.42%, 88.07%)
Divorced	(89.5%, 91.1%)	(83.50%, 85.91%)
Separated	(88.3%, 91.6%)	(80.33%, 85.28%)
Never Married	(85.3%, 86.3%)	(83.31%, 85.72%)

Married women earn 83.5%-90.9% as much as married men based on the observable characteristics. It is hard to come up with explanations why this gap is so large. The only explanation is that we did not adjust for extremely detailed occupational data, because even with this massive sample size, it would not be appropriate to split into such small occupational groups. At least a large portion of this gap is probably caused by discrimination.The gender wage gap is non-existent for never married men and women. Divorced men still earn than never married women and divorced women. Married women earn about as much as never married women.

4.7 - Bar Charts and 99% Confidence Intervals for Usual Hours Worked Per Week, Commute Time, Race, and Means of Transport to Work

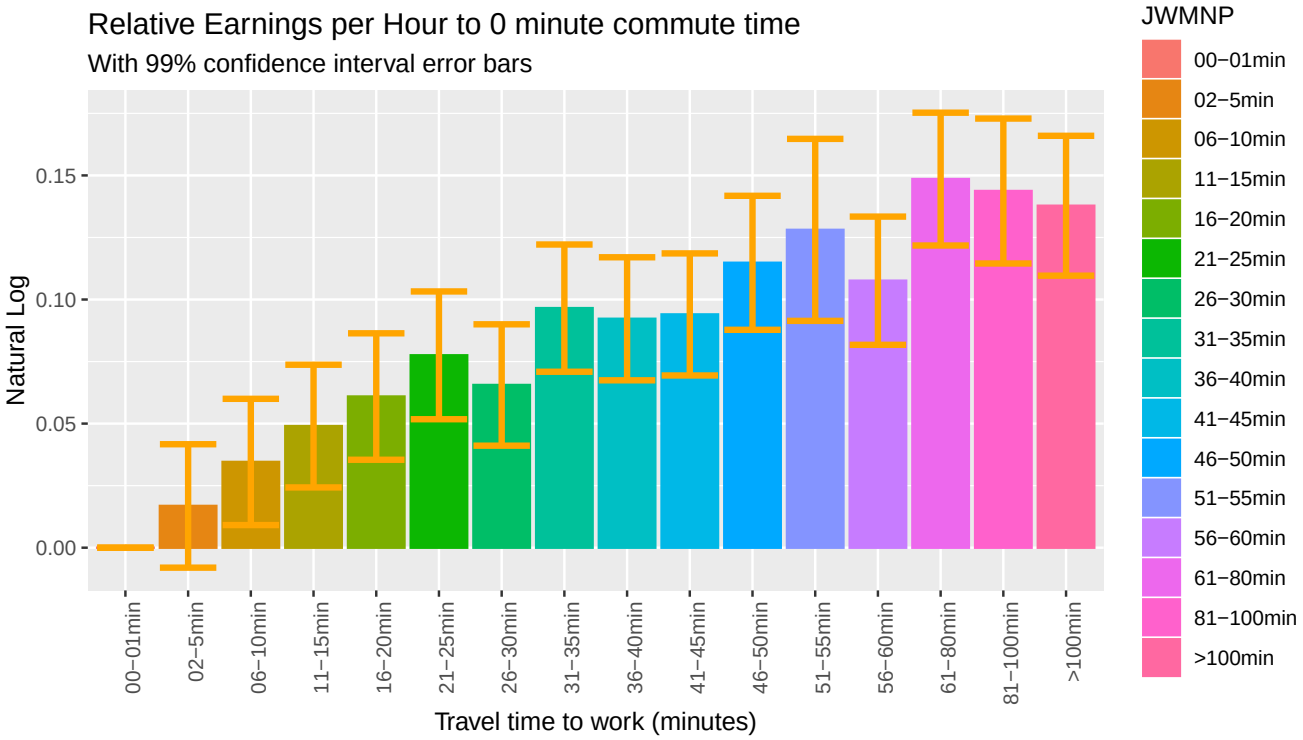


The relative earnings per hour chart shows that those who work more than 55 hours per week earn less per hour than those who work 36-55 hours per week. The entire confidence interval for earnings per hour at 56-60 hours per week is below 51-55 hours per weeek. Working 40-50 hours per week seems to be optimal for maximizing earnings per hour, excluding those who work less than 15 hours per week (maybe contract workers).

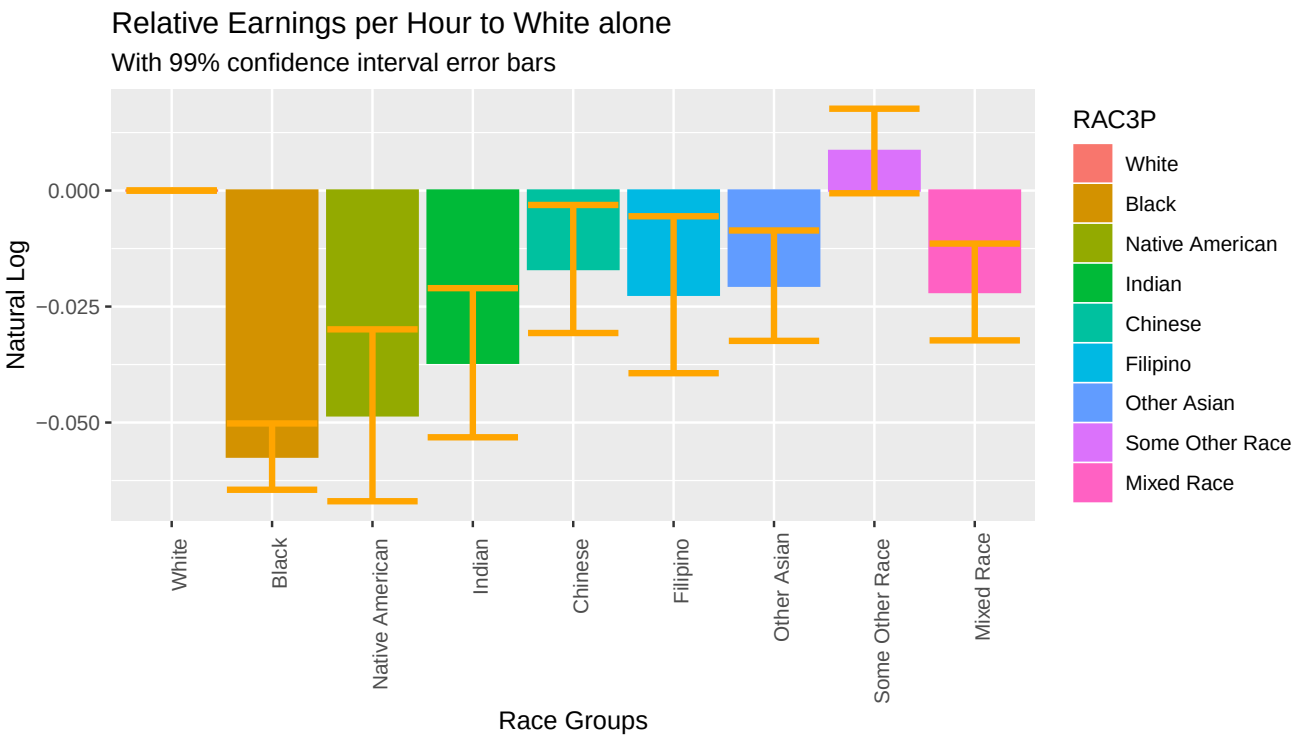


This chart shows relative earnings per year based on hours per week. The suggests that once you get past 50 hours

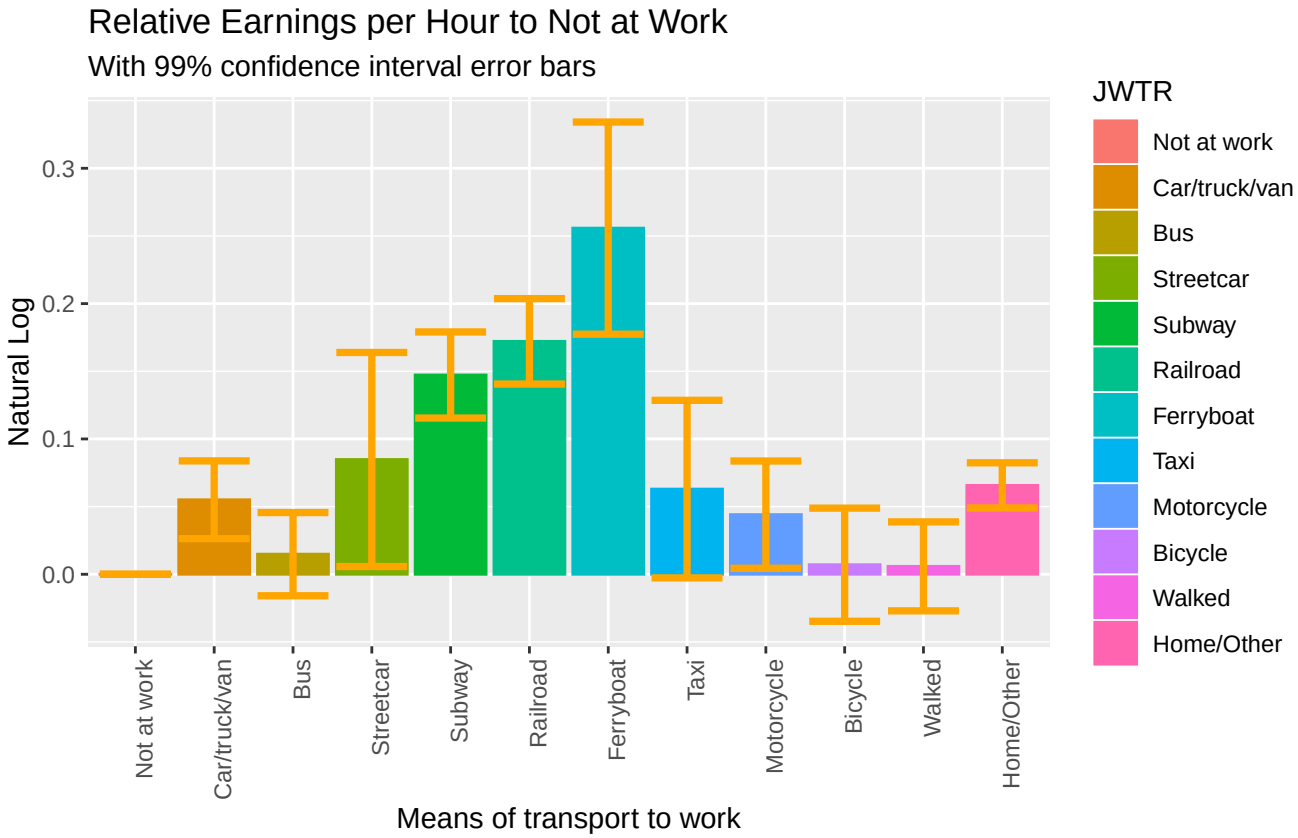
per week, your income does not go up any further. That is why from the earnings per hour chart, we can see earnings per hour go down past 50 hours per weeek. From the two charts, we conclude that it is not worth it for any one to work more than 50 hours per week. This may be why in modern industrialized countries, fewer people are working more than 50 hours per week. From an theoretical economics standpoint, the chart suggests that neither companies or workers produce any more output past 50 hours per week.



The chart shows relative earnings per hour relative to a 0 minute commute. From now on, we will only show relative earnings per hour, because the relative earnings per year chart is basically the same chart. People with longer commutes earn more money, but actually less than expected. People with a 30-50 minute commute earn 0.1 log points more than a 0 minute commute, which is about 10% more. This is despite that a 30-50 minute commute is 60-100 minutes round trip, which is a relative 12.5-20% increase in time spent from an 8 hour workday. We see that people do not get compensated for longer commutes as much as we expect them to for some reason. People with very long commutes must have some unobservable disadvantage for this to be the case.

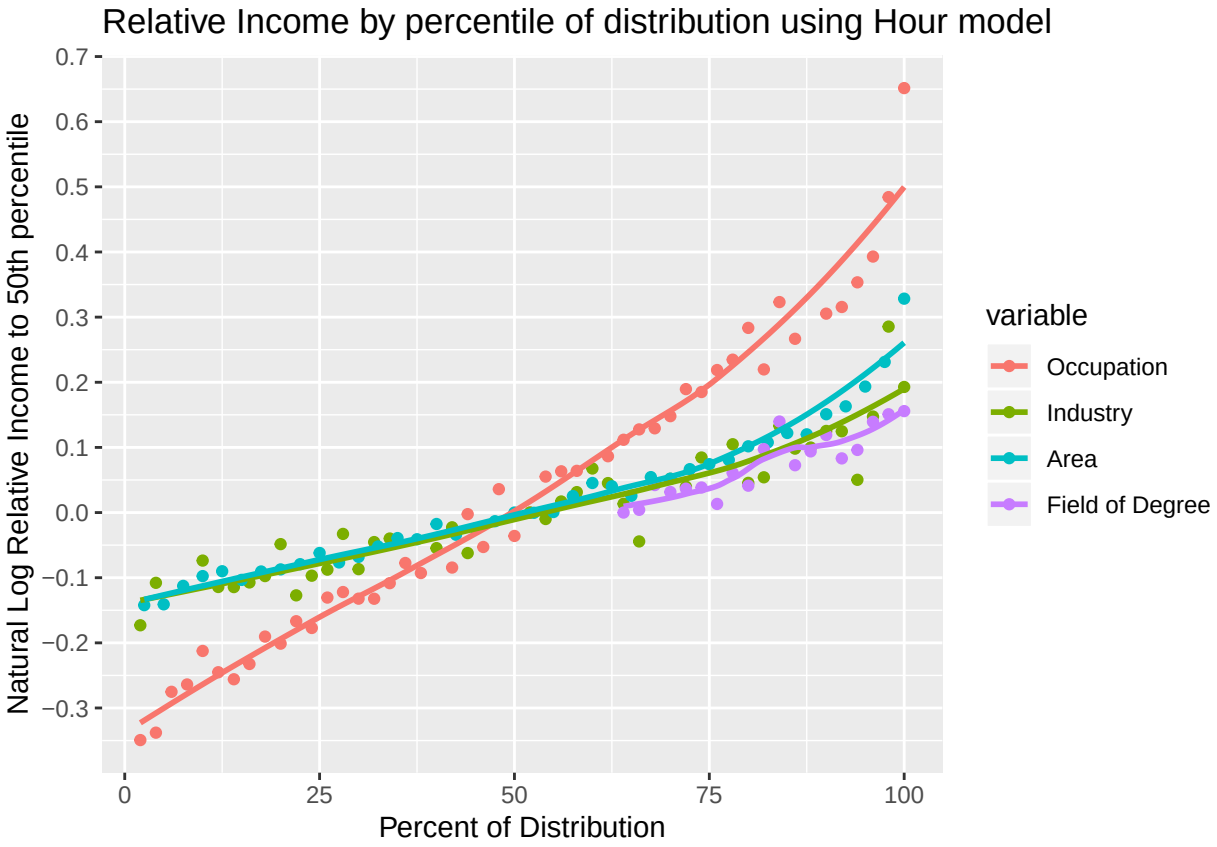


From this bar graph, we can see that all races have an income disadvantage compared to White Americans except for Some Other Race alone. In particular, Black Americans have a large disadvantage of about 6% compared to White Americans with a very high degree of confidence. Thus, there are some unobservable characteristics that lower the income of Black Americans signficantly compared to all other racial groups.

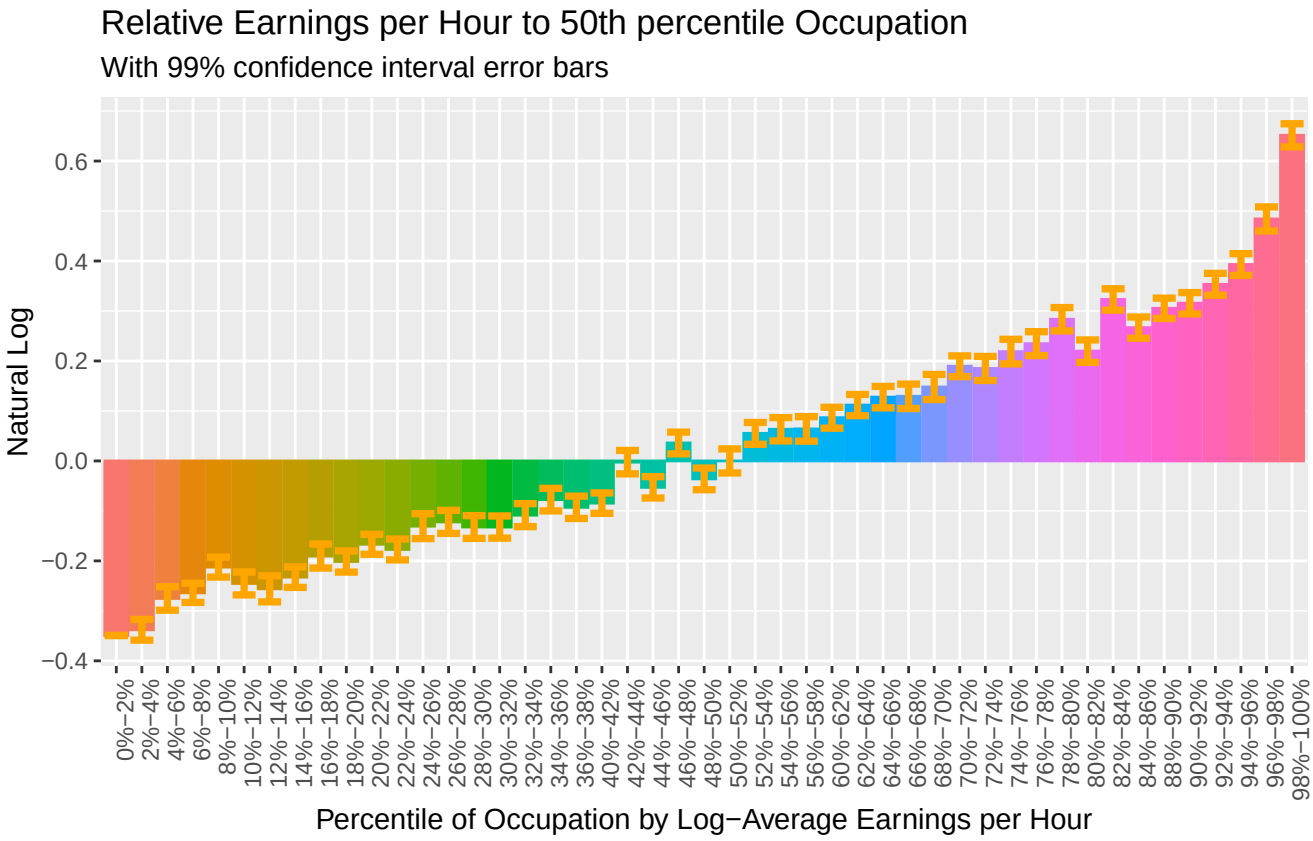


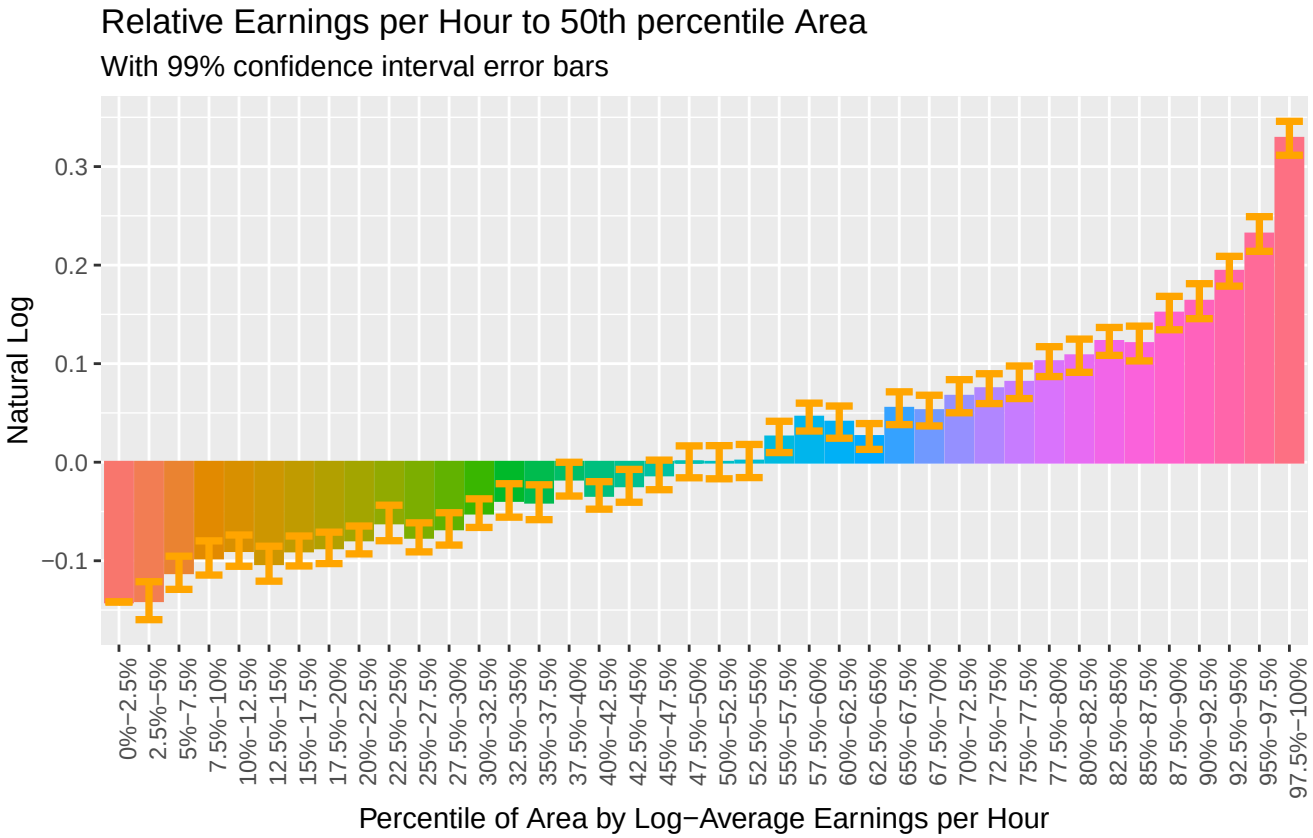
This are the same confidence intervals from the table from earlier regarding commute mode. Those using the subway, railroad, or ferry earn more than those who drive to work. Otherwise, the other confidence intervals overlap with those drive to work.

4.8- Effects of Occupation, Industry, and Area



In this chart, we overlayed occupation, industry, area, and field of degree with the percentile of distribution. We can see from this chart that the percentile of occupation affects income much more than percentile of industry or percentile or area. Also, the data for field of degree and industry is much noisier than that of occupation and area. We can predict from this alone that field of degree and industry will be worse predictors than area. We can see from the ANOVA table from earlier that this is in fact the case. OCCPH has a MSE of 147845, NAICSPY of 219092, PUMAH of 31760, an FOD1PY of 5760.





This bar graph shows that the impact of percentile of occupation on earnings is not log-linear. The curve slopes upward near the upper end of the percentile distribution as what we would expect from a heavy tailed distribution.

We can see the top 2% earns about 0.6 log points more than the 50th percentile. But the bottom 2% earns about 0.4 log points less than the 50th percentile.

The bar graph for area shows an even steeper slope for the upper tail. People who live in big, rich cities have a significant income a dvantage. The top 2.5% earn 0.3 log points more than the median while the bottom 2.5% earn 0.15 log points less than median.We know that the upper tail of income distributions follow a power-law probability distribution (Pareto distribution). This appears most in the occupation and area variables for this dataset.

The distributions of industry and field o f d egree d o n ot a ppear t o h ave a s teep u pper t ail a nd t hey l ook l ike a straight line on thelog-linear plot, so we did not include a bar graph for those two variables.

4.9 - Conclusion

We can now answer which variables are most important in predicting earnings per hour and earnings per year and how important each variable is explaining how much people earn in the United States in 2018. We successfuly created a regression model that can predict earnings per hour and earnings per year based on 263 parameters of 20 variables. In the future, we may try to use a generalized linear model with Gamma family with a log link function, which would take more computation time. We could use other transformations and include more variables to improe the model. This is definitely n ot t he b est m odel w e c ould h ave u sed, b ut i s p robably c lose t o t he b est w e c an do with only 263 parameters. The model fit would b e b etter i f we i ncluded m ore variables, b ut t hat would t ake more compuation time and we wouldn't be able to use lm() anymore. If we used biglm() or speedlm(), there is nearly no limit to how many variables we could add.

V. Classification u sing L DA, L ogistic R egression, a nd K-NN

5.1 - Introduction

In this section, we focus on solving the key question 2 and 3, that is, what is the expected marital status (married or not married) and labor force participation status (Full-Time or Part-Time/Unemployed) based on the selected predictors. We use LDA, logistic regression and k-NN classifier to classify each individual.
First, We randomly separated the data into a test data (100000 rows) and a training data (1547497 rows). We choose age, education level, sex, race, area, English literacy, occupation, wage per hour, and time arrival to work as predictors since we want to investigate the relationship between marital status or labor force participation status and these varaibles. At the end, we compare the accuracy of each model and find the best one to solve the questions we proposed before.

For labor force participation, we treat any one working at least 30 hours per week for 40 weeks out of the past 52 weeks to be full-time workers. Everyone else is unemployed or part time workers.

For marital status, we treat anyone married or widowed as part of the "Married" category. Those who are never married, separated, or divorced are part of the "Not Married" category.

For convenience, we will not use sample weights for LDA or k-NN, which will decrease the accuracy of the model.

5.2 - LDA model

5.2.1 Classification o n m arital status

First, we use the LDA model to classify marital status. Here is the confusion matrix.

The formula used for this model is lda(MAR~SCHL+SEX+AGEP+sqrt(AGEP)+JWAP+OCCPY+PUMAY+RAC3P+PERNP+
SCHL,SEX,JWAP,OCCPY,PUMAY, RAC3P, ENG have been converted to numeric variables.

```
             Actual
Predicted     Married Not Married
```

```
  Married       54478       25169
  Not Married    7354       12999
```

```
[1] "The accuracy is 67.48%."
```

As we can see, LDA model does not classify the test data well. It predicts a large number to be married when they are actually not. There are more false positives than true negatives. The true positive rate is high but the true negative rate is very low.

5.2.2 Classification on labor force participation

We also want to investigate the relationship between labor force participation status and the same six varibles. Below is the confusion matrix.

The formula used lda(WKHPF~SCHL+SEX+AGEP+sqrt(AGEP)+JWAP+OCCPY+PUMAY+RAC3P+ENG). SCHL,SEX,JWAP,OCCPY,PUMAY, RAC3P, ENG have been converted to numeric variables. We removed the PERNP variable since an income of zero would give away that the person is unemployed.

```
                      Actual
Predicted              Full-Time Unemployed/Part-Time
  Full-Time                57598                11821
  Unemployed/Part-Time      7398                23183
```

```
[1] "The accuracy is 80.78%."
```

The accuracy is not bad at 80.85%. Probably the biggest reason why the accuracy rate is high is that those who don't have an occupation in the past 5 years are definitely not e mployed. The OCCP variable defines "Unemployed for more than 5 years" to have no occupation. Those who have an occupation in the past 5 years are most likely fully employed.

However, the model appears to have a very hard time distinguishing part time workers from full time workers. This is because the accuracy rate for "Unemployed/part-time" is only about 66% despite that we can easily determine which workers are unemployed.

5.3. Logistic Regression

5.3.1 Classification on marital status

Since this is a logistic regression model, we use a generalized linear model with the binomial family. After training the model and fitting the test data, we get the confusion matrix as shown.

The formula used is glm(MAR ~ SCHL + AGEP + sqrt(AGEP) + SEX + ENG + RAC3P + JWAP + OCCPY + PUMAY + PERNP).

```
                                                    True Marital Status
                                  Predicted Marital Status Married Not Married
            Married       52247       22158
            Not Married    9585       16010
```

```
[1] "The accuracy is 68.26%."
```

This model has a similar accuracy rate to the LDA model. They both predict too many as married. However, the true negative rate is still much than of the LDA model. But the true positive rate is slightly lower compared to the LDA model.

5.3.2 Classification on labor force participation

For the labor force participation status, it is also a multiclass: full-time, part-time, and unemployed. We use the same method and here is the result.

The formula used is glm(WKHP ~ SCHL + AGEP + sqrt(AGEP) + SEX + RAC3P + JWAP + OCCPY + PUMAY + MAR + ENG, weights = PWGTP, family = binomial)

Again, we removed the PERNP variable, since an income of 0 would immediately give away that the person is unemployed.

```
                      True Labor Force
Predicted Labor Force  Full-Time Unemployed/Part-Time
  Full-Time                58929                12369
  Unemployed/Part-Time      6067                22635
```

```
[1] "The accuracy is 81.56%."
```

The accuracy is slightly higher than that of the LDA model. Both the true positive rate is higher and the true negative rate is higher. So this model is better than the LDA model for labor force participation. Of course, this model is much more complex than the LDA model since we used categorical variables for many of the variables instead of numeric variables, which are less precise. So the two models are not directly comparable, but since we really want the variables to be factors, LDA is less appropriate.

5.4 - K-NN Classification

5.4.1 Classification on marital status

The previous two models both have some problems on classifying marital status. We decide to use k-NN classifier and find out which model is the best. We chose to investigate how the model perform when k = 5, 9, and 19. Here are the three confusion matrices.

K = 5

```
             Actual
Predicted     Married Not Married
  Married       47715       21753
  Not Married   14117       16415
```

```
[1] "The accuracy for k = 5 is 64.13%."
```

K = 9

```
             Actual
Predicted     Married Not Married
  Married       50107       22343
  Not Married   11725       15825
```

```
[1] "The accuracy for k = 9 is 65.93%."
```

K = 19

```
             Actual
Predicted     Married Not Married
  Married       52210       23235
  Not Married    9622       14933
```

```
[1] "The accuracy for k = 19 is 67.14%."
```

The result shows highest accuracy using k = 19. The accuracy rate is lower than LDA and logistic regression model. When k = 19 or k = 9, the accuracy is significantly lower than that of k = 5, but still lower than logistic regression and LDA. We should test for higher values of k in the future to see if the accuracy rate can get higher.

5.4.2 Classification on labor force participation

We want to find out if there is an improvement for the labor force participation status classification using k-NN classifier. Here are the results.

K = 5

```
                      Actual
Predicted              Full-Time Unemployed/Part-Time
  Full-Time                59218                 9634
  Unemployed/Part-Time      5778                25370
```

```
[1] "The accuracy for k = 5 is 84.59%."
```

K = 9

```
                      Actual
Predicted              Full-Time Unemployed/Part-Time
  Full-Time                59920                 9555
  Unemployed/Part-Time      5076                25449
```

```
[1] "The accuracy for k = 9 is 85.37%."
```

K = 19

```
                      Actual
Predicted              Full-Time Unemployed/Part-Time
  Full-Time                60290                 9497
  Unemployed/Part-Time      4706                25507
```

```
[1] "The accuracy for k = 19 is 85.8%."
```

The accuracy when k = 19 is higher compared to k = 5 and k = 9. Furthermore, with any value of k, the accuracy of the k-NN classification is better than both LDA and logistic regression. The accuracy of the other two is still good. e should test for higher values of k in the future to see if the accuracy rate can get higher.

5.5 - Future Plan

For the next step, we may need to figure out why LDA, k-NN, and logistic regression perform poorly on classifying marital status. It is likely we need very complex models to get these models to accurately predict marital status. This would take even longer than the computation needed for the linear regression model. k-NN with a low k value gives us a complex model automatically. The advantage of k-NN for our data set is that it allows us to create a complex model without testing hundreds of models. It took us much longer to run the k-NN compared to the LDA or logistic regression. A low value of k may lead to overfitting to this specific dataset only, but any value of k that we tested in k-NN beats LDA and logistic regression. We believe it is unlikely to ever get a good model with LDA for marital status however, since LDA only allows numeric independent variables. Furthermore, we can implement k-fold cross-validation on the data set to evaluate the performance of these three models and find the best value of k for k-NN.

5.6 - Conclusion

Among the three classification models, when we classify labor force participation, the k-NN classifier performs the best of the three models. Adding more parameters to the logistic regression would increase accuracy. A small value of k will likely overfit the model, but the k values we tested all turned out to be more accurate than the logistic regression and LDA. But logistic regression was best for marital status. We do not believe LDA is a good method for measuring either marital status or labor force participation, because we can not use categorical variables to create a very complex model such as with logistic regression, which would be appropriate since this dataset has 1,600,000 data points.

Appendix 1 - Session Info

```
R version 3.6.1 (2019-07-05)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 18362)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.1252
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] MASS_7.3-51.4     class_7.3-15      nnet_7.3-12       gtools_3.8.1
 [5] scales_1.0.0      viridis_0.5.1     viridisLite_0.3.0 readr_1.3.1
 [9] biglm_0.9-1       DBI_1.0.0         kableExtra_1.1.0  e1071_1.7-2
[13] ggplot2_3.2.1     dplyr_0.8.3

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.2       plyr_1.8.4       pillar_1.4.2     compiler_3.6.1
 [5] tools_3.6.1      zeallot_0.1.0    digest_0.6.22    lattice_0.20-38
 [9] evaluate_0.14    tibble_2.1.3     gtable_0.3.0     pkgconfig_2.0.3
[13] rlang_0.4.1      rstudioapi_0.10  yaml_2.2.0       hexbin_1.28.0
[17] xfun_0.11        gridExtra_2.3    xml2_1.2.2       httr_1.4.1
[21] withr_2.1.2      stringr_1.4.0    knitr_1.26       vctrs_0.2.0
[25] hms_0.5.2        webshot_0.5.2    grid_3.6.1       tidyselect_0.2.5
```

```
[29] glue_1.3.1       R6_2.4.1         rmarkdown_1.17   reshape2_1.4.3
[33] purrr_0.3.3      magrittr_1.5     backports_1.1.5  htmltools_0.4.0
[37] rvest_0.3.5      assertthat_0.2.1 colorspace_1.4-1 labeling_0.3
[41] stringi_1.4.3    lazyeval_0.2.2   munsell_0.5.0    crayon_1.3.4
```

Appendix 2 - Other Tables for Building the Regression Model

Table 37: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF

	AIC	BIC	Adjusted R-squared	df
Null	4801236	4801971	0.56326	61
+AGEPF	4769493	4770397	0.57396	75
+factor(AGEP)	**4769208**	4770415	**0.57407**	100
+AGEP	4776950	4777698	0.57147	62
+AGEP+I(AGEP**2)	4769480	4770240	0.57396	63
+AGEP+I(AGEP**2)+I(AGEP**3)	4769359	**4770131**	0.57400	64
+sqrt(AGEP)+AGEP	4769371	**4770131**	0.57400	63
+sqrt(AGEP)+AGEP+I(AGEP**2)	4769373	4770145	0.57400	64

Table 38: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP

	AIC	BIC	Adjusted R-squared	df
Null	4769371	4770131	0.57400	63
+JWMNPF	**4759144**	**4760085**	**0.57740**	78
+JWMNP	4762662	4763434	0.57623	64
+JWMNP+I(JWMNP**2)	4759858	4760642	0.57716	65
+JWMNP+I(JWMNP**2)+I(JWMNP**3)	4759776	4760572	0.57718	66
+sqrt(JWMNP)+JWMNP	4760576	4761360	0.57692	65
+sqrt(JWMNP)+JWMNP+I(JWMNP**2)	4759860	4760656	0.57715	66

Table 39: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF

	AIC	BIC	Adjusted R-squared	df
Null	4759144	4760085	0.57740	78
+JWAPF	**4755544**	**4756823**	**0.57859**	106
+JWAP	4758442	4759395	0.57763	79
+JWAP+I(JWAP**2)	4758417	4759382	0.57764	80
+JWAP+I(JWAP**2)+I(JWAP**3)	4758147	4759124	0.57773	81
+sqrt(JWAP)+JWAP	4758436	4759401	0.57763	80
+sqrt(JWAP)+JWAP+I(JWAP**2)	4758412	4759389	0.57764	81

Table 40: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCP with Different Methods of Splitting Up OCCP

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4755544	4756823	0.57859	106	0
By Major Group	4659480	4661048	0.60907	130	23
2 equal groups	4664289	4665580	0.60759	107	1
5 equal groups	4607680	4609007	0.62457	110	4
10 equal groups	4594278	4595665	0.62849	115	9
20 equal groups	4587749	4589256	0.63038	125	19
30 equal groups	4585461	4587089	0.63104	135	29
40 equal groups	4583133	**4584882**	0.63172	145	39
50 equal groups	**4583051**	4584909	**0.63174**	154	48

Table 41: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCPY+NAICSP with Different Methods of Splitting Up NAICSP

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4583133	4584882	0.63172	145	0
By Major Group	4573287	4575241	0.63454	162	17
2 equal groups	4569780	4571541	0.63554	146	1
5 equal groups	4565548	4567346	0.63674	149	4
10 equal groups	4563389	4565247	0.63736	154	9
20 equal groups	4560617	4562583	0.63815	163	18
30 equal groups	4560040	4562115	0.63831	172	27
40 equal groups	4558799	4560958	0.63866	179	34
50 equal groups	**4557913**	**4560157**	**0.63892**	186	41

Table 42: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCPY+NAICSPY+PUMA with Different Methods of Splitting Up PUMA

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4557913	4560157	0.63892	186	0
By Major Group	4537199	4540492	0.64474	273	87
2 equal groups	4540123	4542379	0.64390	187	1
5 equal groups	4530076	4532368	0.64669	190	4
10 equal groups	4527528	4529880	0.64739	195	9
20 equal groups	4526149	4528621	0.64777	205	19
30 equal groups	4525756	4528350	0.64788	215	29
40 equal groups	4525470	**4528184**	0.64796	225	39
50 equal groups	**4525420**	4528255	**0.64798**	235	49

Table 43: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNP tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCPY+NAICSPY+PUMAY+FOD1P with Different Methods of Splitting Up FOD1P

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4525470	4528184	0.64796	225	0
By Major Group	4523747	4526654	0.64844	241	16
5 equal groups	4523662	4526388	0.64846	226	1
10 equal groups	4523443	4526193	0.64852	228	3
20 equal groups	4523019	4525817	0.64864	232	7
30 equal groups	4522914	4525760	0.64867	236	11
40 equal groups	4522899	4525794	0.64868	240	15
50 equal groups	**4522757**	**4525688**	**0.64872**	243	18

Table 44: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF

	AIC	BIC	Adjusted R-squared	df
Null	4800455	4801191	0.26521	61
+AGEPF	4768518	4769422	0.28333	75
+factor(AGEP)	**4768237**	4769443	**0.28350**	100
+AGEP	4775895	4776643	0.27918	62
+AGEP+I(AGEP**2)	4768521	4769281	0.28332	63
+AGEP+I(AGEP**2)+I(AGEP**3)	4768387	**4769159**	0.28339	64
+sqrt(AGEP)+AGEP	4768399	**4769159**	0.28339	63
+sqrt(AGEP)+AGEP+I(AGEP**2)	4768401	4769173	0.28339	64

Table 45: AIC, BIC, and Adjusted R-squared of Regression Models in addition to PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP

	AIC	BIC	Adjusted R-squared	df
Null	4768399	4769159	0.28339	63
+JWMNPF	**4758148**	**4759089**	**0.28911**	78
+JWMNP	4761711	4762483	0.28712	64
+JWMNP+I(JWMNP**2)	4758869	4759653	0.28871	65
+JWMNP+I(JWMNP**2)+I(JWMNP**3)	4758787	4759583	0.28875	66
+sqrt(JWMNP)+JWMNP	4759581	4760365	0.28831	65
+sqrt(JWMNP)+JWMNP+I(JWMNP**2)	4758871	4759667	0.28871	66

Table 46: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCP with Different Methods of Splitting Up OCCP

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4754642	4755921	0.29107	106	0
By Major Group	4658226	4659794	0.34253	130	23
2 equal groups	4659950	4661241	0.34164	107	1
5 equal groups	4603237	4604564	0.37018	110	4
10 equal groups	4588845	4590232	0.37722	115	9
20 equal groups	4581812	4583319	0.38064	125	19
30 equal groups	4579597	4581225	0.38171	135	29
40 equal groups	4578702	4580439	0.38215	144	38
50 equal groups	**4577438**	**4579271**	**0.38277**	152	46

Table 47: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCPH+NAICSP with Different Methods of Splitting Up NAICSP

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4577438	4579271	0.38277	152	0
By Major Group	4566696	4568735	0.38793	169	17
2 equal groups	4566669	4568515	0.38794	153	1
5 equal groups	4560907	4562789	0.39069	156	4
10 equal groups	4558812	4560754	0.39169	161	9
20 equal groups	4556853	4558904	0.39262	170	18
30 equal groups	4556199	4558334	0.39294	177	25
40 equal groups	4554800	4557031	0.39360	185	33
50 equal groups	**4554131**	**4556459**	**0.39392**	193	41

Table 48: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCPH+NAICSPH+PUMA with Different Methods of Splitting Up PUMA

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4554131	4556459	0.39392	193	0
By Major Group	4534540	4537918	0.40317	280	87
2 equal groups	4534269	4536609	0.40326	194	1
5 equal groups	4524580	4526956	0.40776	197	4
10 equal groups	4521488	4523925	0.40919	202	9
20 equal groups	4519911	4522468	0.40992	212	19
30 equal groups	4519485	4522163	0.41012	222	29
40 equal groups	4519172	**4521970**	0.41027	232	39
50 equal groups	**4519071**	4521990	**0.41032**	242	49

Table 49: AIC, BIC, and Adjusted R-squared of Linear Regression Model for PERNPHR tilde WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAPF+OCCPH+NAICSPH+PUMAH+FOD1P with Different Methods of Splitting Up FOD1P

	AIC	BIC	Adjusted R-squared	df	Additional df
Null	4519172	4521970	0.41027	232	0
By Major Group	4517024	4520015	0.41127	248	16
5 equal groups	4517325	4520135	0.41112	233	1
10 equal groups	4516923	4519758	0.41131	235	3
20 equal groups	4516573	4519456	0.41147	239	7
30 equal groups	4516117	4519048	0.41168	243	11
40 equal groups	4516245	4519224	0.41163	247	15
50 equal groups	**4516014**	**4519018**	**0.41173**	249	17

Appendix 3 - Getting the Code

This is the relevant code to replicate the two files in this project. Go to

census.gov/programs-surveys/acs/data/pums.html

Then click on "2018 ACS PUMS" and click on 1-Year. Scroll down until arriving at "csv_pus.zip", which should be 536M file size and modified on 31-Oct-2019 at 15:02. Download this file. Extract the folder. Inside this file should be "psam_pusa.csv" and "psam_pusb.csv". Then follow the code below to get the get the two files. Replace the file names for read.csv with the actual location of the file on the relevant computer and do the read.csv to get the data frames.

```
part1 = read.csv("./csv_pus/psam_pusa.csv", stringsAsFactors = FALSE)
part2 = read.csv("./csv_pus/psam_pusb.csv", stringsAsFactors = FALSE)
library(dplyr)
#Filter to age 25 to 64
part1 = filter(part1, AGEP %in% c(25:64))
part2 = filter(part2, AGEP %in% c(25:64))
#Combine the two files
acs2018 = rbind(part1, part2)
#Combine State to PUMA to get unique PUMAs
acs2018$PUMA = paste0(acs2018$ST, formatC(acs2018$PUMA,
                                          width = 5, format = "d", flag = "0"))
acs2018$PUMA = as.numeric(acs2018$PUMA)
#Use the income adjustment factor
acs2018$PERNP = acs2018$ADJINC/1000000 * acs2018$PERNP
#Select the variables
final = c("PERNP", "NAICSP","OCCP","JWAP","WKW","WRK","MAR","MARHT",
          "ENG","JWTR","QTRBIR","RAC3P","SEX","WKHP","SCHL",
          "HISP","JWMNP","AGEP","PUMA","FOD1P","PWGTP")
weight = paste0("PWGTP",1:80)
age2564final = select(acs2018, final)
age2564weight =  select(acs2018, weight)
#Set NAs to 0
age2564final[is.na(age2564final)] = 0
#Save as .rds files
save(age2564final, file = "age2564final.rds")
save(age2564weight, file = "age2564weight.rds")
```

After the doing above code, the age2564final.rds and age2564weight.rds files should be the same as the files used in the project. This dataset has 1,647,497 observations.

Appendix 4 - Getting the Standard Errors

Here is the relevant code to get the coefficients and standard errors for the final regression models of earnings per hour and earnings per year from the two files "age2564final.rds" and "age2564weights.rds". This takes a long time to run so it will not be run while knitting this document.

Time needed to run the following code: 2 hours and 25 minutes

```
load("age2564final.rds")
load("age2564weight.rds")
library(dplyr)
age2564final = cbind(age2564final, age2564weight)
rm(age2564weight)
age2564final = filter(age2564final, PERNP > 0)
age2564final = filter(age2564final, PERNP > quantile(age2564final$PERNP, 0.01))
age2564final$PERNPHR = WagePerHour(age2564final)
age2564final = filter(age2564final, PERNPHR > quantile(age2564final$PERNPHR, 0.002)
                      & PERNPHR < quantile(age2564final$PERNPHR, 0.998))
age2564final$PERNP = log(age2564final$PERNP)
age2564final$PERNPHR = log(age2564final$PERNPHR)
age2564final$FOD1PY = convertFOD1Pquantiles(age2564final, "PERNP", "PWGTP", seq(0, 1, length = 51))
age2564final$PUMAH = convertPUMAquantiles(age2564final, "PERNPHR", "PWGTP", seq(0, 1, length = 41))
age2564final$OCCPH = convertOCCPquantiles(age2564final, "PERNPHR", "PWGTP", seq(0, 1, length = 51))
age2564final$NAICSPY = convertNAICSPquantiles(age2564final, "PERNP", "PWGTP", seq(0, 1, length = 51))
convert = c("WKW","WRK","MAR","MARHT","ENG","HISP","JWTR","QTRBIR",
            "RAC3P","SEX", "SCHL","JWAP","JWMNP","WKHP")
age2564final = convertList(age2564final, convert)

Hour = formula("PERNPHR~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHP+sqrt(AGEP)+AGEP+JWMNP+
JWAP+OCCPH+NAICSPY+PUMAH+FOD1PY+SEX:I(AGEP^2)+SCHL:I(AGEP^2)+SEX:MAR")
Year = formula("PERNP~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHP+sqrt(AGEP)+AGEP+JWMNP+JWAP+
OCCPH+NAICSPY+PUMAH+FOD1PY+SEX:I(AGEP^2)+SCHL:I(AGEP^2)+SEX:MAR")

BICListHr = c(1:81)
RListHr = c(1:81)
BICListYr = c(1:81)
RListYr = c(1:81)
library(speedglm)
finalHr = GetSpeedLM(age2564final, Hour, "PWGTP")
finalHrcoef = finalHr$coefficients
BICListHr[1] = BIC(finalHr)
RListHr[1] = summary(finalHr)[[11]]
gc()
finalYr = GetSpeedLM(age2564final, Year, "PWGTP")
finalYrcoef = finalYr$coefficients
BICListYr[1] = BIC(finalYr)
RListYr[1] = summary(finalYr)[[11]]
gc()
regressionHour = c()
regressionYear = c()
replicates = paste0("PWGTP",1:80)
for (i in c(1:80)){
  age2564final[replicates[i]][age2564final[replicates[i]] < 0] = 1
```

```
  Hmodel = GetSpeedLM(age2564final, Hour, replicates[i])
  regressionHour = rbind(regressionHour, Hmodel$coefficients)
  BICListHr[i+1] = BIC(Hmodel)
  RListHr[i+1] = summary(Hmodel)[[11]]
  gc()
  Ymodel = GetSpeedLM(age2564final, Year, replicates[i])
  regressionYear = rbind(regressionYear, Ymodel$coefficients)
  BICListYr[i+1] = BIC(Ymodel)
  RListYr[i+1] = summary(Ymodel)[[11]]
  gc()
}
sHr = 0
sYr = 0
for (i in c(1:80)){
  sHr = sHr + (regressionHour[i,] - finalHrcoef)^2
  sYr = sYr + (regressionYear[i,] - finalYrcoef)^2
}
SErsquaredhour = (sum((RListHr[-1] - RListHr[1])^2) * 4/80)^0.5
SErsquaredyear = (sum((RListYr[-1] - RListYr[1])^2) * 4/80)^0.5
sHr = (sHr * 4/80)^0.5
sYr = (sYr * 4/80)^0.5
sHr = cbind(finalHrcoef, sHr, finalHrcoef/sHr, 2*pt(finalHrcoef/sHr, 1279505))
sYr = cbind(finalYrcoef, sYr, finalYrcoef/sYr, 2*pt(finalYrcoef/sYr, 1279505))
colnames(sHr) = c("Estimate","Standard Error","t value","Pr(>|t|)")
colnames(sYr) = c("Estimate","Standard Error","t value","Pr(>|t|)")
Replications = list(BICListHr, RListHr, BICListYr, RListYr, SErsquaredhour, SErsquaredyear)
save(Replications, file = "Replications.rds")
save(sHr, file = "SEHour.rds")
save(sYr, file = "SEYear.rds")
```

Appendix 5 - Functions

```
convertList <- function(x, y = c()){
  if ("PUMA" %in% y){
    x$PUMA = convertPUMA(x$PUMA)
    x$PUMA = factor(x$PUMA)
  }
  if ("OCCP" %in% y){
    x$OCCP = convertOCCP(x$OCCP)
    x$OCCP = factor(x$OCCP)
  }
  if ("FOD1P" %in% y){
    x$FOD1P = convertFOD1P(x$FOD1P)
    x$FOD1P = factor(x$FOD1P)
  }
  if ("NAICSP" %in% y){
    x$NAICSP = convertNAICSP(x$NAICSP)
  }
  if ("AGEP" %in% y){
    x$AGEP = convertAGEP(x$AGEP)
  }
  if ("ENG" %in% y){
    x$ENG = convertENG(x$ENG)
```

```
  }
  if ("HISP" %in% y){
    x$HISP = convertHISP(x$HISP)
  }
  if ("JWAP" %in% y){
    x$JWAP = convertJWAP(x$JWAP)
  }
  if ("JWMNP" %in% y){
    x$JWMNP = convertJWMNP(x$JWMNP)
  }
  if ("JWTR" %in% y){
    x$JWTR = convertJWTR(x$JWTR)
  }
  if ("MAR" %in% y){
    x$MAR = convertMAR(x$MAR)
  }
  if ("MARHT" %in% y){
    x$MARHT = convertMARHT(x$MARHT)
  }
  if ("QTRBIR" %in% y){
    x$QTRBIR = convertQTRBIR(x$QTRBIR)
  }
  if ("RAC3P" %in% y){
    x$RAC3P = convertRAC3P(x$RAC3P)
  }
  if ("SCHL" %in% y){
    x$SCHL = convertSCHL(x$SCHL)
  }
  if ("SEX" %in% y){
    x$SEX = convertSEX(x$SEX)
  }
  if ("WKHP" %in% y){
    x$WKHP = convertWKHP(x$WKHP)
    x$WKHP = factor(x$WKHP)
  }
  if ("WKW" %in% y){
    x$WKW = convertWKW(x$WKW)
    x$WKW = factor(x$WKW)
  }
  if ("WRK" %in% y){
    x$WRK = convertWRK(x$WRK)
  }
  return(x)
}


regression <- function(x, TheFormula, w = "PWGTP"){
  form = formula(TheFormula)
  gc()
  S = lm(form, weights = get(w), data = x)
  gc()
  AIC.model = AIC(S)
  BIC.model = BIC(S)
  Rsquared = summary(S)[10]
  degrees = summary(S)[[11]][2]
  L = as.data.frame(cbind(AIC.model, BIC.model, Rsquared, degrees))
```

```
  colnames(L) = c("AIC","BIC","AdjustedRsquared","df")
  return(L)
}

GetSpeedLM <- function(x, TheFormula, w= "PWGTP"){
    n = length(x[,1])
    one = x[1:400000,]
    two = x[400001:800000,]
    three = x[800001:n,]
    rm(x)
    gc()
    library(speedglm)
    form = formula(TheFormula)
    S = eval(bquote(speedlm(.(form), weights = one[[w]], data = one)))
    gc()
    S = update(S, sparse = FALSE, weights = two[[w]], data = two)
    gc()
    S = update(S, sparse = FALSE, weights = three[[w]], data = three)
    return(S)
}

regressionBig <- function(x, TheFormula, w = "PWGTP"){
    n = length(x[,1])
    one = x[1:400000,]
    two = x[400001:800000,]
    three = x[800001:n,]

    gc()
    library(biglm)
    form = formula(TheFormula)
    S = biglm(form, weights=~get(w), data = one)
    gc()
    S = update(S, two)
    gc()
    S = update(S, three)
    gc()

    AIC.model = n*log(deviance(S)/n) + 2 * length(coef(S))
    BIC.model = n*log(deviance(S)/n) + log(n) * length(coef(S))
    degrees = length(S[[6]])
    Rsquared = summary(S)[[4]]
    ARsquared = 1 - (1 - Rsquared) * ((n-1)/(n - degrees))
    L  = as.data.frame(cbind(AIC.model, BIC.model, ARsquared, degrees))
    colnames(L) = c("AIC","BIC","AdjustedRsquared","df")
    return(L)
}

regressionSpeed <- function(x, TheFormula, w = "PWGTP"){
    n = length(x[,1])
    one = x[1:400000,]
    two = x[400001:800000,]
    three = x[800001:n,]
    gc()
    library(speedglm)
    form = formula(TheFormula)
    S = eval(bquote(speedlm(.(form), weights = one[[w]], data = one)))
```

```
    gc()
    S = update(S, sparse = FALSE, weights = two[[w]], data = two)
    gc()
    S = update(S, sparse = FALSE, weights = three[[w]], data = three)
    AIC.model = AIC(S)
    BIC.model = BIC(S)
    degrees = summary(S)[[13]][2]
    Rsquared = summary(S)[[11]]
    L  = as.data.frame(cbind(AIC.model, BIC.model, Rsquared, degrees))
    colnames(L) = c("AIC","BIC","AdjustedRsquared","df")
    return(L)
}


regressionGroup <- function(x, Formula, groups = c(20), variable = "OCCP", type = "PERNP", w = "PWGTP"){
    TheFormula = paste0(Formula,"+","1")
    Frame = regressionBig(x, TheFormula, w)
    x[paste0(variable,0)] = get(paste0("convert",variable))(x[[variable]])
    TheFormula = paste0(Formula,"+",variable,"0")
    Frame[2,] = regressionBig(x, TheFormula, w)
    Number = c(0, length(unique(x[[paste0(variable,0)]]))-1)
    name = c("Null","By Major Group")
    for (i in groups){
        gc()
        name = c(name, paste0(i," equal groups"))
        x[paste0(variable,i+1)] =
            get(paste0("convert",variable,"quantiles"))(x, type, w, seq(0, 1, length = i+1))
        TheFormula = paste0(Formula,"+",variable,i+1)
        Frame[nrow(Frame)+1,] = regressionBig(x, TheFormula, w)
        Number = c(Number, length(unique(x[[paste0(variable,i+1)]]))-1)
    }
    Frame = cbind(Frame, Number)

    gc()
    Frame[,1] = formatC(as.numeric(Frame[,1]), format = "f", digits = 0)
    Frame[,2] = formatC(as.numeric(Frame[,2]), format = "f", digits = 0)
    Frame[,3] = formatC(as.numeric(Frame[,3]), format = "f", digits = 5)
    Frame =
        mutate(Frame,
            AIC = cell_spec(AIC, "latex", bold = (as.numeric(AIC) == min(as.numeric(Frame$AIC)))),
            BIC = cell_spec(BIC, "latex", bold = (as.numeric(BIC) == min(as.numeric(Frame$BIC)))),
            AdjustedRsquared = cell_spec(AdjustedRsquared, "latex",
            bold = (as.numeric(AdjustedRsquared) == max(as.numeric(Frame$AdjustedRsquared))))
        )
    Formula = gsub("~"," tilde ",Formula)
    rownames(Frame) = name
    colnames(Frame) = c("AIC", "BIC", "Adjusted R-squared","df","Additional df")
    capt = paste0("AIC, BIC, and Adjusted R-squared of Linear Regression Model for
    ",Formula,"+",variable," with Different Methods of Splitting Up ",variable)

    PrintTable = kable(Frame,
             caption = capt,
             booktabs = TRUE, escape = FALSE, linesep = "") %>%
             kable_styling(latex_options = c("HOLD_position"), full_width = FALSE)
    return(PrintTable)
}
```

```
regressionList <- function(x, Formulas, title, w = "PWGTP"){
    Frame = regressionBig(x, Formulas[1], w)
    if (length(Formulas) > 1){
        for (i in c(2:length(Formulas))){
            gc()
            Frame[nrow(Frame)+1,] = regressionBig(x, Formulas[i], w)
        }
    }

    Frame[,1] = formatC(as.numeric(Frame[,1]), format = "f", digits = 0)
    Frame[,2] = formatC(as.numeric(Frame[,2]), format = "f", digits = 0)
    Frame[,3] = formatC(as.numeric(Frame[,3]), format = "f", digits = 5)
    Frame =
        mutate(Frame,
            AIC = cell_spec(AIC, "latex", bold = (as.numeric(AIC) == min(as.numeric(Frame$AIC)))),
            BIC = cell_spec(BIC, "latex", bold = (as.numeric(BIC) == min(as.numeric(Frame$BIC)))),
            AdjustedRsquared = cell_spec(AdjustedRsquared, "latex",
            bold = (as.numeric(AdjustedRsquared) == max(as.numeric(Frame$AdjustedRsquared))))
        )
    colnames(Frame) = c("AIC", "BIC", "Adjusted R-squared","df")
    rownames(Frame) = sapply(Formulas, function(x) paste0(substr(x, nchar(title)+1, nchar(x))))
    rownames(Frame) = sapply(rownames(Frame), function(x) ifelse(x == "", "Null", x))
    title = gsub("~"," tilde ",title)
    capt = paste0("AIC, BIC, and Adjusted R-squared of Regression Models in addition to
    ", title)
    PrintTable = kable(Frame,
             caption = capt,
             booktabs = TRUE, escape = FALSE, linesep = "") %>%
             kable_styling(latex_options = c("HOLD_position"), full_width = FALSE)
    return(PrintTable)
}

GetFormulaVariations <- function(x, y){
    List = c(1:8)
    List[1] = x
    List[2] = paste0(x,"+",y,"F")
    List[3] = paste0(x,"+","factor(",y,")")
    List[4] = paste0(x,"+",y)
    List[5] = paste0(x,"+",y,"+I(",y,"**2)")
    List[6] = paste0(x,"+",y,"+I(",y,"**2)","+I(",y,"**3)")
    List[7] = paste0(x,"+","sqrt(",y,")+",y)
    List[8] = paste0(x,"+","sqrt(",y,")+",y,"+I(",y,"**2)")
    return(List)
}


GetVariables <- function(x = c()){
  Vars = c("PERNP","NAICSP","OCCP","JWAP","WKW","WRK","MAR","MARHT",
           "ENG","JWTR","QTRBIR","RAC3P","SEX","WKHP","SCHL",
           "HISP","JWMNP","AGEP","PUMA","FOD1P","PWGTP")
  Labels = c("Earnings","Industry","Occupation","Time Arrival to Work","Weeks Worked Past Year",
             "Worked Last Week","Marital Status","Times Married","English Fluency",
             "Mode of Transport to Work","Quarter of Birth","Race","Sex",
             "Usual Hours Worked per Week","Educational Attainment",
             "Hispanic origin","Travel Time to Work","Age","Area",
```

```
            "Field of Degree","Final Weights")
  A = data.frame(cbind(Vars, Labels))
  if (length(x) > 0){
    A = filter(A, Vars %in% x)
  }
  colnames(A) = c("Variables","Definitions")
  return(A)
}

GetWeights <- function(x = c()){
  Vars = paste0("PWGTP",1:80)
  return(Vars)
}

convertNAICSP <- function(x){
  x[x == ""] = "0"
  S = unique(x)
  None = S[startsWith(S, "0") | startsWith(S,"999920")]
  AGR = S[startsWith(S, "1")]
  EXT = S[startsWith(S, "21")]
  UTL = S[startsWith(S, "22")]
  CON = S[startsWith(S, "23")]
  MFG = S[startsWith(S, "3")]
  WHL = S[startsWith(S, "42")]
  RET = S[startsWith(S, "44") | startsWith(S,"45") | startsWith(S,"4MS")]
  TRN = S[startsWith(S, "48") | startsWith(S,"49")]
  INF = S[startsWith(S, "51")]
  FIN = S[startsWith(S, "52") | startsWith(S,"53")]
  PRF = S[startsWith(S, "54") | startsWith(S, "55") | startsWith(S, "56")]
  EDU = S[startsWith(S, "61")]
  MED = S[startsWith(S, "621") | startsWith(S, "622") | startsWith(S, "623")]
  SCA = S[startsWith(S, "624")]
  ENT = S[startsWith(S, "7")]
  SRV = S[startsWith(S, "8")]
  ADM = S[startsWith(S, "921") | startsWith(S, "923") | startsWith(S, "92M") |
            startsWith(S, "928P")]
  MIL = S[startsWith(S, "9281")]
  x[x %in% None] = "None"
  x[x %in% AGR] = "AGR"
  x[x %in% EXT] = "EXT"
  x[x %in% UTL] = "UTL"
  x[x %in% CON] = "CON"
  x[x %in% MFG] = "MFG"
  x[x %in% WHL] = "WHL"
  x[x %in% RET] = "RET"
  x[x %in% TRN] = "TRN"
  x[x %in% INF] = "INF"
  x[x %in% FIN] = "FIN"
  x[x %in% PRF] = "PRF"
  x[x %in% EDU] = "EDU"
  x[x %in% MED] = "MED"
  x[x %in% SCA] = "SCA"
  x[x %in% ENT] = "ENT"
  x[x %in% SRV] = "SRV"
  x[x %in% ADM] = "ADM"
  x[x %in% MIL] = "MIL"
```

```
  gc()
  x = as.factor(x)
  return(x)
}
```

```
PrintNAICSPnames <- function(x){
    industries = c("None","Agriculture, Forestry, Fishing and Hunting",
    "Mining, Quarrying, and Oil and Gas Extraction","Utilities","Construction","Manufacturing",
    "Wholesale Trade","Retail Trade","Transportation and Warehousing",
    "Information","Finance, Insurance, Real Estate, Rental, Leasing",
    "Professional, Scientific, and Technical Services+Management of Companies",
    "Educational Services","Health Care","Social Assistance",
    "Arts, Entertainment, Recreation, Accommodation, Food Services",
    "Other Services","Public Administration","Military")
    name = c("None","AGR","EXT","UTL","CON","MFG","WHL","RET","TRN","INF","FIN","PRF","EDU","MED","SCA","E
    "ADM","MIL")
    Frame = as.data.frame(cbind(name, industries))
    colnames(Frame) = c("Code","2017 NAICS Sector")
    PrintTable = kable(Frame,
             caption = "Converting 2018 ACS Industry Code to 2017 NAICS Sectors",
             booktabs = TRUE, linesep = "") %>%
             column_spec(1, width = "5em") %>%
             kable_styling(latex_options = c("HOLD_position"), full_width = FALSE)
    return(PrintTable)
}

PrintOCCPnames <- function(x){
    occupations = c("None","Management","Business","Financial Operations","Computer and Mathematical",
    "Architecture and Engineering","Life, Physical, and Social Science","Community and Social Service",
    "Legal","Education, Training, and Library","Arts, Design, Entertainment, Sports,and Media",
    "Healthcare Practitioners and Technical","Healthcare Support","Protective Service",
    "Food Preparation and Serving Related","Building and Grounds Cleaning and Maintenance",
    "Personal Care and Service","Sales and Related","Office and Administrative Support",
    "Farming, Fishing,and Forestry","Construction and Extraction","Installation, Maintenance, and Repair",
    "Production","Transportation and Material Moving","Military")
    name = c("None","MGR","BUS","FIN","CMM","ENG","SCI","CMS","LGL","EDU","ENT","MED","HLS",
           "PRT","EAT","CLN","PRS","SAL","OFF","FFF","CON","RPR","PRD","TRN","MIL")
    Frame = as.data.frame(cbind(name, occupations))
    colnames(Frame) = c("Code","Major Occupational Group")
    PrintTable = kable(Frame,
             caption = "Converting 2018 ACS Occupation Code to SOC Major Occupational Groups",
             booktabs = TRUE, linesep = "") %>%
             column_spec(1, width = "5em") %>%
             kable_styling(latex_options = c("HOLD_position"), full_width = FALSE)
    return(PrintTable)
}

convertOCCP <- function(x){
  x[x %in% 9920] = 0
  occupations = c(-1, 1, 440,750,960,1240,1560,1980,2060,2180,2555,2920,3550,3655,3960,4160,
                  4255,4655,4965,5940,6130,6950,7640,8990,9760,9830)
  name = c("None","MGR","BUS","FIN","CMM","ENG","SCI","CMS","LGL","EDU","ENT","MED","HLS",
           "PRT","EAT","CLN","PRS","SAL","OFF","FFF","CON","RPR","PRD","TRN","MIL")
  x = cut(x, breaks = occupations)
  levels(x) = name
  return(x)
```

```
}


convertNAICSPquantiles <- function(x, type = "PERNP", w = "PWGTP", quant = seq(0, 1, 0.05)){
  x = select(x, w, "NAICSP", type)
  y = group_split(x, NAICSP)
  k = group_keys(x, NAICSP)
  m = sapply(y, weightedmean, type, w)
  p = sapply(y, weight, w)
  combined = cbind(k, m, p)
  combined = arrange(combined, m)
  combined[,3] = cumsum(combined[,3])/sum(combined[,3])
  combined$p = cut(combined$p, breaks = quant)
  Split = group_split(combined, p)
  Groups = sapply(Split, function(x) x[,1])
  l = as.character(unique(combined$p))
  l = strsplit(substring(l, first = 2, last = nchar(l)-1), ",")
  l = sapply(l, function(x) paste0(as.numeric(x[1])*100,"%-",as.numeric(x[2])*100, "%"))
  x = x$NAICSP
  for (i in c(1:length(Groups))){
    x[x %in% Groups[[i]]] = l[i]
  }
  return(as.factor(x))
}

convertOCCPquantiles <- function(x, type = "PERNP", w = "PWGTP", quant = seq(0, 1, 0.05)){
  x = select(x, w, "OCCP", type)
  y = group_split(x, OCCP)
  k = group_keys(x, OCCP)
  m = sapply(y, weightedmean, type, w)
  p = sapply(y, weight, w)
  combined = cbind(k, m, p)
  combined = arrange(combined, m)
  combined[,3] = cumsum(combined[,3])/sum(combined[,3])
  combined$p = cut(combined$p, breaks = quant)
  Split = group_split(combined, p)
  Groups = sapply(Split, function(x) x[,1])
  l = as.character(unique(combined$p))
  l = strsplit(substring(l, first = 2, last = nchar(l)-1), ",")
  l = sapply(l, function(x) paste0(as.numeric(x[1])*100,"%-",as.numeric(x[2])*100, "%"))
  x = x$OCCP
  for (i in c(1:length(Groups))){
    x[x %in% Groups[[i]]] = l[i]
  }
  return(as.factor(x))
}


convertJWAP <- function(x){
  times = c(-1,0,45,57,63,69,72,75,78,81,84,87,90,93,96,99,102,105,108,111,114,117,
            120,123,129,141,153,165,213, 285)
  name = c("Home","12AM-4AM","4AM-5AM","5AM-5:30AM","5:30AM-6AM","6:00AM-6:15AM","6:15AM-6:30AM","6:30AM-6
           "6:45AM-7AM","7AM-7:15AM","7:15AM-7:30AM","7:30AM-7:45AM","7:45AM-8AM",
           "8AM-8:15AM","8:15AM-8:30AM","8:30AM-8:45AM","8:45AM-9AM","9AM-9:15AM",
           "9:15AM-9:30AM","9:30AM-9:45AM","9:45AM-10AM","10AM-10:15AM","10:15AM-10:30AM","10:30AM-11AM",
           "11AM-12PM","12PM-1PM","1PM-2PM","2PM-6PM","6PM-11:59PM")
```

```
  x = cut(x, breaks = times)
  levels(x) = name
  return(x)
}

convertWKW <- function(x){
    weeks = c(-1,0,1,2,3,4,5,6)
    name = c("0wks","50-52wks","48-49wks","40-47wks","27-39wks","14-26wks","01-13wks")
    x = cut(x, breaks = weeks)
    levels(x) = name
    return(x)
}

convertWRK <- function(x){
    work = c(-1,0,1,2)
    name = c("Not reported","Worked last week","Did not work last week")
    x = cut(x, breaks = work)
    levels(x) = name
    return(x)
}


convertMAR <- function(x){
    marriagestatus = c(0,1,2,3,4,5)
    name = c("Married","Widowed","Divorced","Separated","Never Married")
    x = cut(x, breaks = marriagestatus)
    levels(x) = name
    return(x)
}

convertMARHT <- function(x){
    marriagetimes = c(-1,0,1,2,3)
    name = c("Never Married","Married Once","Married Twice","Married 3+ Times")
    x = cut(x, breaks = marriagetimes)
    levels(x) = name
    return(x)
}

convertENG <- function(x){
    englishlit = c(-1,0,1,2,3,4)
    name = c("Only English","Very well","Well","Not well","Not at all")
    x = cut(x, breaks = englishlit)
    levels(x) = name
    return(x)
}

convertJWTR <- function(x){
    commutes = c(-1,0,1,2,3,4,5,6,7,8,9,10,12)
    name = c("Not at work","Car/truck/van","Bus","Streetcar","Subway","Railroad","Ferryboat","Taxi","Motor
        "Bicycle","Walked","Worked at home/Other")
    x = cut(x, breaks = commutes)
    levels(x) = name
    return(x)
}

convertQTRBIR <- function(x){
```

```
    quarter = c(0,1,2,3,4)
    name = c("Jan-Mar","Apr-June","July-Sept","Oct-Dec")
    x = cut(x, breaks = quarter)
    levels(x) = name
    return(x)
}




convertRAC3P <- function(x){
  races = c(0,1,2,3,4,5,6,14,15,100)
  name = c("White Alone","Black Alone","Native American Alone","Indian alone","Chinese alone",
           "Filipino alone","Other Asian or Pacific Islander","Some Other Race alone","Mixed Race")
  x = cut(x, breaks = races)
  levels(x) = name
  return(x)
}


convertWKHP <- function(x){
  hours = c(-1,0,5,10,15,20,25,30,35,39,40,45,50,55,60,65,70,99)
  name = c("0hrs","01-5hrs","06-10hrs","11-15hrs","16-20hrs","21-25hrs","26-30hrs",
           "31-35hrs","36-39hrs","40hrs","41-45hrs","46-50hrs","51-55hrs","56-60hrs","61-65hrs",
           "66-70hrs",">70hrs")
  x = cut(x, breaks = hours)
  levels(x) = name
  return(x)
}

convertSEX <- function(x){
    sexes = c(0,1,2)
    name = c("Male","Female")
    x = cut(x, breaks = sexes)
    levels(x) = name
    return(x)
}

#Convert educational attainment to factor
convertSCHL <- function(x){
  degrees = c(0,15,17,19,20,21,22,23,24)
  name = c("Less than HS diploma","HS or equivalent","Some college","Associate's","Bachelor's",
        "Master's","Professional","Doctorate")
  x = cut(x, breaks = degrees)
  levels(x) = name
  return(x)
}

#Convert Hispanic origin to factor
convertHISP <- function(x){
  hispanicorigin = c(0,1,2,24)
  name = c("Not Hispanic","Mexican","Hispanic,not Mexican")
  x = cut(x, breaks = hispanicorigin)
  levels(x) = name
  return(x)
}
```

```
#Convert Commute Time to factor
convertJWMNP <- function(x){
  commutes = c(-1,1,5,10,15,20,25,30,35,40,45,50,55,60,80,100,999)
  name = c("00-01min","02-5min","06-10min","11-15min","16-20min","21-25min","26-30min","31-35min",
           "36-40min","41-45min","46-50min","51-55min","56-60min","61-80min",
           "81-100min",">100min")
  x = cut(x, breaks = commutes)
  levels(x) = name
  return(x)
}

#Convert Age to factor
convertAGEP <- function(x){
  ages = c(24,25,26,27,28,29,30,31,33,35,37,39,44,54,59,64)
  name = c("25","26","27","28","29","30","31","32-33","34-35","36-37","38-39","40-44","45-54","55-59","60-
  x = cut(x, breaks = ages)
  levels(x) = name
  return(x)
}



#Convert PUMA to quantiles of mean income of PUMA
convertPUMAquantiles <- function(x, type = "PERNP", w = "PWGTP", quant = seq(0, 1, 0.05)){
  x = select(x, "PUMA", w, type)
  Temp = x$PUMA
  Split = group_split(x, PUMA)
  Med = cbind(group_keys(x, PUMA), sapply(Split, weightedmean, type))
  names(Med) = c("PUMA",type)
  Quants = quantile(Med[,2], quant)
  Quants = Quants[!duplicated(Quants)]
  Med[,2] = cut(Med[,2], breaks = Quants)
  Split2 = group_split(Med, get(type))
  PUMAsplit = sapply(Split2, function(x) return(x[,1]))
  QName = names(Quants)
  Name = c()
  for (i in c(1:length(Quants)-1)){
    Name[i] = paste0(QName[i],"-",QName[i+1])
  }
  if (length(PUMAsplit) > length(Name)){
    PUMAsplit[[1]] = c(PUMAsplit[[1]], PUMAsplit[[length(PUMAsplit)]])
  }
  for (i in c(1:length(PUMAsplit))){
    Temp[Temp %in% PUMAsplit[[i]]] = Name[i]
  }
  Temp = as.factor(Temp)
  return(Temp)
}
```

```
#Convert PUMA to arbitrarily decided regions where large city within each state are grouped
convertPUMA <- function(x){
  #Alabama 1

  JeffersonAL = c(101301:101305)
```

```
Mobile = c(102701:102703)
x[x %in% JeffersonAL] = "AL,Jefferson"
x[x %in% Mobile] = "AL,Mobile"
x[x %in% c(100000:102703)] = "Other"
#Alaska 2
x[x %in% c(200101:200400)] = "Other"
#Arizona 3
Maricopa = c(400100:400134)
Pima = c(400201:400209)
x[x %in% Maricopa] = "AZ,Maricopa"
x[x %in% Pima] = "AZ,Pima"
x[x %in% c(400135:400900)] = "Other"
gc()
#Arkansas 5
x[x %in% c(500100:502000)] = "Other"
#California 6
Alameda = c(600101:600110)
ContraCosta = c(601301:601309)
Fresno = c(601901:601907)
Kern = c(602901:602905)
LosAngeles = c(603701:603769)
OrangeCA = c(605901:605918)
Riverside = c(606501:606515)
Sacramento = c(606701:606712)
SanBernardino = c(607101:607115)
SanDiego = c(607301:607322)
SF = c(607501:607507)
SanMateo = c(608101:608106)
SantaClara = c(608501:608514)
Ventura = c(611101:611106)
x[x %in% Alameda] = "CA,Alameda"
x[x %in% ContraCosta] = "CA,ContraCosta"
x[x %in% Fresno] = "CA,Fresno"
x[x %in% Kern] = "CA,Kern"
x[x %in% LosAngeles] = "CA,LA"
x[x %in% OrangeCA] = "CA,Orange"
x[x %in% Riverside] = "CA,Riverside"
x[x %in% Sacramento] = "CA,Sacramento"
x[x %in% SanBernardino] = "CA,SanBernadino"
x[x %in% SanDiego] = "CA,SD"
x[x %in% SF] = "CA,SF"
x[x %in% SanMateo] = "CA,SM"
x[x %in% SantaClara] = "CA,SC"
x[x %in% Ventura] = "CA,Ventura"
x[x %in% c(600300:611300)] = "Other"
#Colorado 8
Denver = c(800812:800816)
ColoradoSprings = c(804101:804106)
DenverSuburb = c(800801:800811,800817:800824)
x[x %in% Denver] = "CO,Denver"
x[x %in% ColoradoSprings] = "CO,ElPaso"
x[x %in% DenverSuburb] = "CO,DenverS"
x[x %in% c(800100:804106)] = "Other"
#Connecticut 9
NewHaven = c(900901:900906)
Hartford = c(900300:900306)
```

```
x[x %in% NewHaven] = "CT,New"
x[x %in% Hartford] = "CT,Hart"
x[x %in% c(900100:901500)] = "Other"
#Delaware 10
x[x %in% c(1000101:1000300)] = "DE,Delaware"
#District of Columbia 11
x[x %in% c(1100101:1100105)] = "Urban"
#Florida 12
Broward = c(1201101:1201114)
Duval = c(1203101:1203107)
Hillsborough = c(1205701:1205708)
MiamiDade = c(1208601:1206824)
OrangeFL = c(1209501:1209510)
PalmBeach = c(1209901:1209911)
Pinellas = c(1210301:1210308)
x[x %in% Broward] = "FL,Broward"
x[x %in% Duval] = "FL,Duval"
x[x %in% Hillsborough] = "FL,Hillsbourough"
x[x %in% OrangeFL] = "FL,Orange"
x[x %in% PalmBeach] = "FL,PalmBeach"
x[x %in% Pinellas] = "FL,Pinellas"
x[x %in% c(1200101:1212704)] = "Other"
#Georgia 13
CAtlanta = c(1301001:1301008)
x[x %in% CAtlanta] = "GA,Atlanta"
x[x %in% c(1300100:1306002)] = "Other"
#Hawaii 15
x[x %in% c(1500100:1500308)] = "HI,Hawaii"
#Idaho 16
x[x %in% c(1600100:1601300)] = "Other"
#Illinois 17
Cook = c(1703401:1703422)
Chicago = c(1703501:1703532)
x[x %in% Cook] = "Urban"
x[x %in% Chicago] = "Urban"
x[x %in% c(1700104:1703700)] = "Other"
#Indiana 18
Marion = c(1802301:1802307)
x[x %in% Marion] = "IN,Marion"
x[x %in% c(1800101:1803600)] = "Other"
#Iowa 19
x[x %in% c(1900100:1902300)] = "IA,Iowa"
#Kansas 20
x[x %in% c(2000100:2001500)] = "KS,Kansas"
#Kentucky 21
Jefferson = c(2101701:2101706)
x[x %in% Jefferson] = "KY,Jefferson"
x[x %in% c(2100100:2102800)] = "Other"
#Louisiana 22
NewOrleans = c(2202300:2202302,2202400:2202402)
x[x %in% NewOrleans] = "LA,NewOrleans"
x[x %in% c(2200100:2202500)] = "Other"
#Maine 23
x[x %in% c(2300100:2301000)] = "ME,Maine"
#Maryland 24
BaltimoreCounty = c(2400501:2400507)
```

```
BaltimoreCity = c(2400801:2400805)
PrinceGeorge = c(2401101:2401107)
Montgomery = c(2401001:2401005)
x[x %in% BaltimoreCounty] = "Baltimore"
x[x %in% BaltimoreCity] = "BaltimoreCity"
x[x %in% PrinceGeorge] = "PrinceGeorge"
x[x %in% Montgomery] = "MD,Montgomery"
x[x %in% c(2400100:2401600)] = "Other"
#Massachusetts 25
Middlesex = c(2500501:2500508)
Boston = c(2503301:2503306)
x[x %in% Middlesex] = "Middlesex"
x[x %in% Boston] = "Boston"
x[x %in% c(2500100:2504903)] = "Other"
#Michigan 26
Oakland = c(2602901:2602908)
Wayne = c(2603201:2603213)
x[x %in% Oakland] = "MI,Oakland"
x[x %in% Wayne] = "MI,Wayne"
x[x %in% c(2600100:2603300)] = "Other"
#Minnesota 27
Hennepin = c(2701401:2701410)
x[x %in% Hennepin] = "MN,Hennepin"
x[x %in% c(2700100:2702600)] = "Other"
#Mississippi 28
x[x %in% c(2800100:2802100)] = "MS,Missisippi"
#Missouri 29
StLouis = c(2901801:2901808)
x[x %in% StLouis] = "MO,StLouis"
x[x %in% c(2900100:2902800)] = "Other"
#Montana 30
x[x %in% c(3000100,3000200,3000300,3000400,3000500,3000600,3000700)] = "MT,Montana"
#Nebraska 31
x[x %in% c(3100100,3100200,3100300,3100400,3100500,3100600,3100701:3100904)] = "NE,Nebraska"
#Nevada 32
Clark = c(3200401:3200413)
Other = c(3200101:3200300)
x[x %in% Clark] = "NV,Clark"
x[x %in% Other] = "Other"
#New Hampshire 33
x[x %in% c(3300100:3301000)] = "NH,NewHampshire"
#New Jersey 34
Bergen = c(3400301:3400308)
Middlesex = c(3400901:3400907)
x[x %in% Bergen] = "NJ,Bergen"
x[x %in% Middlesex] = "NJ,Middlesex"
x[x %in% c(3400101:3402600)] = "Other"
#New Mexico 35
Albuquerque = c(3500801:3500806)
x[x %in% Albuquerque] = "NM,Albuquerque"
x[x %in% c(3500100:3501200)] = "Other"
#New York 36
Erie = c(3601201:3601207)
Nassau = c(3603201:3603212)
Suffolk = c(3603301:3603313)
Bronx = c(3603701:3603710)
```

```
Manhattan = c(3603801:3603810)
Brooklyn = c(3604001:3604018)
Queens = c(3604101:3604114)
x[x %in% Erie] = "NY,Erie"
x[x %in% Nassau] = "NassauNY"
x[x %in% Suffolk] = "SuffolkNY"
x[x %in% Bronx] = "Bronx"
x[x %in% Manhattan] = "Manhattan"
x[x %in% Brooklyn] = "BrooklynNY"
x[x %in% Queens] = "QueensNY"
x[x %in% c(3600100:3604114)] = "Other"
#North Carolina 37
Wake = c(3701201:3701208)
Mecklenburg = c(3703101:3703108)
x[x %in% Wake] = "NC,Wake"
x[x %in% Mecklenburg] = "NC,Mecklenburg"
x[x %in% c(3700100:3705400)] = "Other"
#North Dakota 38
gc()
x[x %in% c(3800100,3800200,3800300,3800400,3800500)] = "Other"
#Ohio 39
Cuyahoga = c(3900901:3900910)
Columbus = c(3904101:3904111)
Hamilton = c(3905501:3905507)
x[x %in% Cuyahoga] = "OH,Cuya"
x[x %in% Columbus] = "OH,Colum"
x[x %in% Hamilton] = "OH,Hamilton"
x[x %in% c(3900100:3905700)] = "Other"
#Oklahoma 40
Oklahoma = c(4001001:4001006)
x[x %in% Oklahoma] = "OK,OklahomaCity"
x[x %in% c(4000100:4001601)] = "Other"
#Oregon 41
PortlandCity = c(4101301:4101316)
x[x %in% PortlandCity] = "Urban"
x[x %in% c(4100100:4101324)] = "Other"
#Pennsylvania 42
Allegheny = c(4201801:4201807)
Montgomery = c(4203101:4203106)
Philadelphia = c(4203201:4203211)
x[x %in% Allegheny] = "PA,Allegheny"
x[x %in% Montgomery] = "PA,Montgomery"
x[x %in% Philadelphia] = "PA,Philly"
x[x %in% c(4200101:4204002)] = "Other"
#Puerto Rico
x[x %in% c(7200101:7201102)] = "Puerto Rico"
#Rhode Island 44
x[x %in% c(4400101:4400400)] = "RI,Rhode Island"
#South Carolina 45
x[x %in% c(4500101:4501600)] = "SC,South Carolina"
#South Dakota 46
x[x %in% c(4600100,4600200,4600300,4600400,4600500,4600600)] = "Other"
#Tennessee 47
Shelby = c(4703201:4703208)
x[x %in% Shelby] = "TN,Shelby"
x[x %in% c(4700100:4703208)] = "Other"
```

```
  #Texas 48
  Dallas = c(4802301:4802322)
  Tarrant = c(4802501:4802516)
  Houston = c(4804601:4804638)
  Austin = c(4805301:4805309)
  SanAntonio = c(4805901:4805916)
  x[x %in% Dallas] = "TX,Dalas"
  x[x %in% Tarrant] = "TX,Tarrant"
  x[x %in% Houston] = "TX,HOuston"
  x[x %in% Austin] = "TX,AUstin"
  x[x %in% SanAntonio] = "TX,SA"
  x[x %in% c(4800100:4806900)] = "Other"
  #Utah 49
  SaltLake = c(4935001:4935009)
  x[x %in% SaltLake] = "UT,SaltLake"
  x[x %in% c(4903001:4957002)] = "Other"
  #Vermont 50
  x[x %in% c(5000100,5000200,5000300,5000400)] = "Other"
  #Virginia 51
  Fairfax = c(5159301:5159309)
  x[x %in% Fairfax] = "VA,Fairfax"
  x[x %in% c(5101301:5159309)] = "Other"
  #Washington 52
  Pierce = c(5311501:5311507)
  King = c(5311601:5311616)
  x[x %in% Pierce] = "WA,Pierce"
  x[x %in% King] = "WA,King"
  x[x %in% c(5310100:5311900)] = "Other"
  #West Virginia 54
  x[x %in% c(5400100:5401300)] = "WV,West Virginia"
  #Wisconsin 55
  Milwaukee = c(5540101:5541005)
  x[x %in% Milwaukee] = "WI,Milwaukee"
  x[x %in% c(5500100:5570301)] = "Other"
  #Wyoming 56
  x[x %in% c(5600100:5600500)] = "Other"
  x = as.factor(x)
  return(x)
}
```

```
#Convert field of degree to factors
convertFOD1P <- function(x){
  x[x %in% 0] = "NoDegree"
  x[x %in% c(1100:1199)] = "Agriculture"
  x[x %in% c(1301:1303)] = "Environment"
  x[x %in% c(2100:2107)] = "Computer"
  x[x %in% c(2300:2399)] = "Education"
  x[x %in% c(1401,2400:2599)] = "Engineering"
  x[x %in% c(2600:2603,3301:3302)] = "Language"
  x[x %in% c(3600:3699)] = "Biology"
  x[x %in% c(3700:3702)] = "Math"
  x[x %in% c(4000:4007)] = "Multiple"
  x[x %in% c(5000:5098)] = "Physical Sciences"
  x[x %in% c(5200:5299)] = "Psychology"
  x[x %in% c(5500:5599)] = "Social Science"
  x[x %in% c(6000:6099)] = "Art"
```

```
  x[x %in% c(6100:6199)] = "Health"
  x[x %in% c(6200:6299)] = "Business"
  x[x %in% c(1:9999)] = "Other"
  x = as.factor(x)
  return(x)
}


convertFOD1Pquantiles <- function(x, type = "PERNP", w = "PWGTP", quant = seq(0, 1, 0.05)){
  x = select(x, w, "FOD1P", type)
  y = group_split(x, FOD1P)
  k = group_keys(x, FOD1P)
  m = sapply(y, weightedmean, type, w)
  p = sapply(y, weight, w)
  combined = cbind(k, m, p)
  combined = arrange(combined, m)
  combined[,3] = cumsum(combined[,3])/sum(combined[,3])
  combined$p = cut(combined$p, breaks = quant)
  Split = group_split(combined, p)
  Groups = sapply(Split, function(x) x[,1])
  l = as.character(unique(combined$p))
  l = strsplit(substring(l, first = 2, last = nchar(l)-1), ",")
  l = sapply(l, function(x) paste0(as.numeric(x[1])*100,"%-",as.numeric(x[2])*100, "%"))
  x = x$FOD1P
  for (i in c(1:length(Groups))){
    x[x %in% Groups[[i]]] = l[i]
  }
  x = as.factor(x)
  return(x)
}


#Calculates the estimates and the standard errors of the regression coefficients
SEregression <- function(x, TheFormula){
  form = formula(TheFormula)
  replicates = paste0("PWGTP",1:80)
  regression = c()
  fin = summary(lm(form, weights = PWGTP, data = x))[5:11]
  final = fin[[1]][,1]
  l = vector(mode = "list", length = 81)
  l[[1]] = fin
  gc()
  df = summary(lm(form, weights = PWGTP, data = x))[[8]][2]
  for (i in c(1:80)){
    weigh = x[replicates[i]]
    weigh[weigh < 0] = 0
    intermed = summary(lm(form, weights = weigh[[1]], data = x))[5:11]
    gc()
    l[[i+1]] = intermed
    regression = rbind(regression, intermed[[1]][,1])
    gc()
  }
  s = 0
  for (i in c(1:80)){
    s = s + (regression[i,] - final)^2
  }
```

```
  s = (s * 4/80)^0.5
  s = cbind(final, s, final/s, 2*pt(final/s, df, lower = FALSE))
  colnames(s) = c("Estimate","Standard Error","t value","Pr(>|t|)")
  return(list(s,l))
}

#These means are gotten from ACS 2007, when the last time that detailed hours worked was available
#The weighted mean of each week range (from ACS 2018) was taken and used
#Gets wage per hour from weeks worked, hours worked per week, and total earnings
WagePerHour <- function(x){
  Weeks = x$WKW
  Weeks[Weeks == 1] = 51.85 #50-52
  Weeks[Weeks == 2] = 48.19 #48-49
  Weeks[Weeks == 3] = 42.38 #40-47
  Weeks[Weeks == 4] = 33.08 #27-39
  Weeks[Weeks == 5] = 21.32 #14-26
  Weeks[Weeks == 6] = 7.49 #1-13
  Hours = Weeks*x$WKHP
  Wage = x$PERNP/Hours
  return(Wage)
}


#Calculates the standard errors of the total population of the sample
SEpopulation <- function(x){
  replicates = paste0("PWGTP",1:80)
  final = sum(x[["PWGTP"]])
  eighty = c()
  for(i in c(1:80)){
    eighty = c(eighty, sum(x[replicates[i]]))
  }
  s = 0
  for(i in c(1:80)){
    s = s + (eighty[i] - final)^2
  }
  s = (s * 4/80)^0.5
  return(s)
}

#Calculates the standard errors of the mean of one of the variables in the sample
SEmean <- function(x, variable){
  factor = variable
  replicates = paste0("PWGTP",1:80)
  final = weightedmean(x, variable)
  eighty = c()
  for (i in c(1:80)){
    eighty = c(eighty, weightedmean(x, variable, replicates[i]))
  }
  s = 0
  for(i in c(1:80)){
    s = s + (eighty[i] - final)^2
  }
  s = (s * 4/80)^0.5
  return(s)
}
```

```
#The population estimate of this sample

weight <- function(x, w = "PWGTP"){
  return(sum(x[w]))
}

#The mean of one of the variables in this sample
weightedmean <- function(x, variable, weights = "PWGTP"){
  one = x[weights]
  two = x[variable]
  average = sum(one*two)/sum(one)
  return(average)
}

#The weighted median of the variable not assuming the variable is sorted low to high
weightedmedian <- function(x, variable, weights = "PWGTP"){
  one = x[weights]
  quantile = sum(one)/2
  two = x[variable]
  combined = as.data.frame(cbind(one, two))
  combined = arrange(combined, get(variable))
  fortypercent = 0.4*length(one[[1]])
  comp = sum(combined[1:fortypercent,1])
  comptemp = comp
  for (i in c(fortypercent:length(one[[1]]))){
    comptemp = comp
    comp = comp + combined[i,1]
    if (comp > quantile){
      break
    }
  }
  offsetpositive = quantile - comptemp
  offsetnegative = comp - quantile
  offsetpositive = offsetpositive/(comp-comptemp)
  offsetnegative = offsetnegative/(comp-comptemp)
  result = combined[i,2]*offsetpositive + combined[i-1,2]*offsetnegative
  return(result)
}
```

```
#Calculates the quantiles of the variable of this sample x. No assumption that the variable is sorted low
weightedquantiles <- function(x, variable, quantiles = seq(0.1,0.9,0.1), weights = "PWGTP"){
  one = x[weights]
  quantile = quantiles*sum(one)
  two = x[variable]
  combined = as.data.frame(cbind(one, two))
  combined = arrange(combined, get(variable))
  comp = 0
  count = 1
  ret = c()
  for (i in c(1:length(one[[1]]))){
    comptemp = comp
    comp = comp + combined[i,1]
    if (comp > quantile[count]){
      offsetpositive = quantiles[count] - comptemp
      offsetnegative = comp - quantiles[count]
      offsetpositive = offsetpositive/(comp-comptemp)
```

```
      offsetnegative = offsetnegative/(comp-comptemp)
      result = combined[i,2]*offsetpositive + combined[i-1,2]*offsetnegative
      ret = c(ret, result)
      count = count + 1
    }
    if (count > length(quantiles)){
      break
    }
  }
  return(ret)
}


GetSummaryStatistics <- function(x, variable, quantiles = FALSE){
    if (quantiles){
        x[variable] = get(paste0("convert",variable,"quantiles"))(x)
    }
    else{
        x = convertList(x, variable)
    }
    Split = group_split(x, get(variable))

    Salary = sapply(Split, weightedmean, "PERNP")
    SDSalary = sapply(Split, weightedmedian, "PERNP")
    Hour= sapply(Split, weightedmean, "PERNPHR")
    SDHour = sapply(Split, weightedmedian, "PERNPHR")
    Weight = sapply(Split, weight)

    Tab = cbind(group_keys(x, get(variable)), Salary, SDSalary, Hour, SDHour, Weight)
    Tab = arrange(Tab, desc(Salary))
    Tab[,c(2:5)] = exp(Tab[,c(2:5)])

    Tab[,6] = prettyNum(Tab[,6], big.mark = ",", digits = 0, scientific = FALSE)
    Tab[,5] = formatC(Tab[,5], big.mark = ",", format = "f", digits = 2)
    Tab[,4] = formatC(Tab[,4], big.mark = ",", format = "f", digits = 2)
    Tab[,3] = prettyNum(Tab[,3], big.mark = ",", digits = 0, scientific = FALSE)
    Tab[,2] = prettyNum(Tab[,2], big.mark = ",", digits = 0, scientific = FALSE)

    Name = as.character(GetVariables(variable)[,2])

    colnames(Tab) = c(Name,"Log-Average","Median","Log-Average","Median","# of Workers")

    capt = paste("Geometric Mean and Median Earnings per Year/Hour split by", Name)

    PrintTable = kable(Tab, "latex",
              caption = capt,
              booktabs = TRUE, linesep = "") %>%
              kable_styling(latex_options = c("HOLD_position"), full_width = FALSE) %>%
              column_spec(1, bold = T) %>%
              column_spec(2, border_left = T) %>%
              column_spec(4, border_left = T) %>%
              column_spec(6, border_left = T) %>%
              add_header_above(c(" " = 1, "Earnings Per Year" = 2, "Earnings Per Hour" = 2, " " = 1))
    return(PrintTable)
}
```

```
GetCoefficientPercent <- function(Model, Term, Term2){
    Model$coefficients = Model$coefficients[,1]
    N = Model$xlevels[[Term2]][1]
    One = data.frame(0.00)
    A = as.data.frame(Model$coefficients[grepl(Term, names(Model$coefficients))])
    rownames(One) = N
    colnames(One) = colnames(A)
    A = rbind(One, A)
    A = tibble::rownames_to_column(A)
    B = strsplit(A[,1], "-", fixed = TRUE)
    B = data.frame(matrix(unlist(B), nrow = length(B), byrow = TRUE))
    A = cbind(A,B)
    A[,4] = as.character(A[,4])
    A[,4] = as.numeric(gsub("%","",A[,4]))
    A = arrange(A, X2)
    B = cbind(A[,4], A[,2])
    colnames(B) = c("Percent",Term)
    B = data.frame(B)
    return(B)
}


ConfidenceInterval <- function(Model, Term, Term2){
    colnames(Model$coefficients) = c("One","Two","Three","Four")
    N = Model$xlevels[[Term2]][1]
    One = data.frame(cbind(0.00,0,Model$coefficients[1,3],
            Model$coefficients[1,4]))
    A = Model$coefficients[grepl(Term, rownames(Model$coefficients)),]
    rownames(One) = N
    colnames(One) = colnames(A)
    A = rbind(One, A)
    Keep = rownames(A)
    Upper = A[,1] + 2.576*A[,2]
    Lower = A[,1] - 2.576*A[,2]
    A = as.data.frame(cbind(A[,1], Lower, Upper))
    rownames(A) = gsub(Term, "", Keep)
    A = tibble::rownames_to_column(A)
    colnames(A) = c(Term,"Estimate","99% Lower","99% Upper")
    return(A)
}

PrintConfidenceIntervalTable <- function(H, Y, Term, Term2){
    x = cbind(ConfidenceInterval(H, Term, Term2), ConfidenceInterval(Y, Term, Term2))
    library(scales)
    Hour = x[,3:4]
    Year = x[,7:8]
    Name = x[,1]
    Hour = exp(Hour)
    Year = exp(Year)
    Hour[,1] = percent(Hour[,1])
    Hour[,2] = percent(Hour[,2])
    Year[,1] = percent(Year[,1])
    Year[,2] = percent(Year[,2])
    HourText = paste0("(",Hour[,1],", ",Hour[,2],")")
    YearText = paste0("(",Year[,1],", ",Year[,2],")")
```

```
    Tab = cbind(HourText, YearText)
    Tab[1,] = c(" Baseline ", " Baseline ")
    capt = paste0("99% Confidence Intervals for ",colnames(x)[1], " relative to ", Name[1])
    colnames(Tab) = c("Earnings per Hour","Earnings per Year")
    rownames(Tab) = Name
    myHeader = c(capt = 3)
    names(myHeader) = c(capt)
    A = kable(Tab, "latex", booktabs = TRUE, linesep = "") %>%
        kable_styling(Tab, latex_options = c("HOLD_position")) %>%
        column_spec(1, bold = TRUE) %>%
        column_spec(2:3, border_left = TRUE) %>%
        add_header_above(header = myHeader, bold = TRUE)
    return(A)
}

PrintSCHLTable <- function(Model){
    A = Model$coefficients[grepl("SCHL", rownames(Model$coefficients)),]
    Thirty = A[1:7,1] + log2(30)*A[8:14,1]
    Forty= A[1:7,1] + log2(40)*A[8:14,1]
    Fifty = A[1:7,1] + log2(50)*A[8:14,1]
    Sixty = A[1:7,1] + log2(60)*A[8:14,1]
    SEthirty = sqrt(A[1:7,2]^2 + log2(30)*A[8:14,2]^2)
    SEforty = sqrt(A[1:7,2]^2 + log2(40)*A[8:14,2]^2)
    SEfifty = sqrt(A[1:7,2]^2 + log2(50)*A[8:14,2]^2)
    SEsixty = sqrt(A[1:7,2]^2 + log2(60)*A[8:14,2]^2)
    Combined = cbind(Thirty,Forty,Fifty,Sixty)
    CombinedSE = cbind(SEthirty,SEforty,SEfifty,SEsixty)
    Lower = exp(Combined - 2.576*CombinedSE)
    Upper = exp(Combined + 2.576*CombinedSE)

    Lower = apply(Lower, 2, percent)
    Upper = apply(Upper, 2, percent)
    Text = matrix(paste0("(", Lower, ", ", Upper, ")"), nrow = 7, ncol = 4, byrow = F)
    Text = rbind("Baseline",Text)
    rownames(Text) = c("Less than HS","HS or equivalent","Some college","Associate's",
    "Bachelor's","Master's","Professional","Doctorate")
    colnames(Text) = c("Age 30","Age 40","Age 50","Age 60")
    capt = "99% Confidence Intervals for Educational Attainment relative to less than HS for Hour model"
    myHeader = c(capt = 5)
    names(myHeader) = c(capt)
    A = knitr::kable(Text, "latex", booktabs = TRUE, linesep = "") %>%
        kable_styling(latex_options = c("HOLD_position")) %>%
        column_spec(1, bold = TRUE) %>%
        column_spec(2:5, border_left = TRUE) %>%
        add_header_above(header = myHeader, bold = TRUE)
    return(A)
}

PrintMARTable <- function(Model){
    A = Model$coefficients[grepl("SEX|MAR", rownames(Model$coefficients)),]
    B = Model$cov[grepl("MAR",rownames(Model$coefficients)),
                  grepl("SEXFemale",rownames(Model$coefficients))][1:4]
    MaleAGEP = log2(25)*A[6,1]
    FemaleAGEP = log2(25)*A[7,1]
    Female = FemaleAGEP + A[5,1]
    Female = c(0,A[1:4,1]) + Female + c(0,A[8:11,1]) - MaleAGEP
```

```
    Male = c(0, A[1:4,1])
    Combined = cbind(Male,Female)

    SEmale = c(0, A[1:4,2])
    SEfemale = c(A[5,2],
                 sqrt(A[1:4,2]^2+A[8:11,2]^2+2*B[1:4]))
    SE = cbind(SEmale,SEfemale)
    Lower = exp(Combined - 2.576*SE)
    Upper = exp(Combined + 2.576*SE)
    Lower = apply(Lower, 2, percent)
    Upper = apply(Upper, 2, percent)
    Text = matrix(paste0("(", Lower, ", ", Upper, ")"), nrow = 5, ncol = 2, byrow = F)
    Text[1,1] = "Baseline"
    colnames(Text) = c("Male","Female")
    rownames(Text) = c("Married","Widowed","Divorced","Separated","Never Married")
    capt = "99% Confidence Intervals for Marital Status relative to Married Men for Hour model"
    myHeader = c(capt = 3)
    names(myHeader) = c(capt)
    A = knitr::kable(Text, "latex", booktabs = TRUE, linesep = "") %>%
        kable_styling(latex_options = c("HOLD_position")) %>%
        column_spec(1, bold = TRUE) %>%
        column_spec(1:3, border_left = TRUE) %>%
        add_header_above(header = myHeader, bold = TRUE)
    return(A)
}

PrintReplicationsTable <- function(Replications){
    UpperYear = Replications[[4]][1]+Replications[[6]]*2.576
    LowerYear = Replications[[4]][1]-Replications[[6]]*2.576
    UpperHour = Replications[[2]][1]+Replications[[5]]*2.576
    LowerHour = Replications[[2]][1]-Replications[[5]]*2.576

    Tab = matrix(c(Replications[[4]][1], Replications[[2]][1], LowerYear, LowerHour, UpperYear, UpperHour)
    colnames(Tab) = c("Estimate","99% Confidence Interval Lower Bound","Upper Bound")
    rownames(Tab) = c("Earnings per Year","Earnings per Hour")

    Tab = as.data.frame(Tab)
    capt = paste0("99% Confidence Interval for Adjusted R-squared")
    A = knitr::kable(Tab, "latex", booktabs = TRUE, linesep = "") %>%
      kable_styling(latex_options = c("HOLD_position")) %>%
      column_spec(2:4, border_left = TRUE)
    return(A)
}
```

Appendix 6 - RMarkdown Code

```
library(dplyr)
library(ggplot2)
library(e1071)
library(kableExtra)
#1. Introduction
kable(GetVariables()[1:20,], caption = "List of Variables used in this Project", linesep = "") %>%
  kable_styling(latex_options = c("HOLD_position"))
```

```r
#2. Summary Statistics and Plots
load("age2564final.rds")


age2564final = filter(age2564final, PERNP > 0)
age2564final = filter(age2564final, PERNP > quantile(age2564final$PERNP, 0.01))
age2564final$PERNPHR = WagePerHour(age2564final)
age2564final = filter(age2564final, PERNPHR > quantile(age2564final$PERNPHR, 0.002)
                      & PERNPHR < quantile(age2564final$PERNPHR, 0.998))

ggplot(age2564final, aes(x = PERNP), weights = PWGTP) +
  geom_histogram(color = "green", fill = "white", bins = 40) +
  xlab("Earnings Per Year (US Dollars)") + ylab("Count") +
  labs(title = "Histogram of Earnings Age 25-64 in ACS 2018")
ggplot(age2564final, aes(x = log(PERNP)), weights = PWGTP) +
  geom_histogram(color = "green", fill = "white", bins = 40) +
  xlab("Natural Log of Earnings Per Year (US Dollars)") + ylab("Count") +
  labs(title = "Histogram of Log of Earnings Age 25-64 in ACS 2018")



ggplot(age2564final, aes(x = PERNPHR), weights = PWGTP) +
  geom_histogram(color = "green", fill = "white", bins = 40) +
  xlab("Earnings Per Hour (US Dollars)") + ylab("Count") +
  labs(title = "Histogram of Hourly Earnings Age 25-64 in ACS 2018")
ggplot(age2564final, aes(x = log(PERNPHR)), weights = PWGTP) +
  geom_histogram(color = "green", fill = "white", bins = 40) +
  geom_vline(xintercept = c(1.98, 2.71), color = "red", size = 1.0) +
   xlab("Natural Log of Earnings Per Hour (US Dollars)") + ylab("Count") +
  labs(title = "Histogram of Log of Hourly Earnings Age 25-64 in ACS 2018")

age2564final$PERNP = log(age2564final$PERNP)
age2564final$PERNPHR = log(age2564final$PERNPHR)

PrintNAICSPnames(0)
GetSummaryStatistics(age2564final, "NAICSP")
GetSummaryStatistics(age2564final, "NAICSP", TRUE)
PrintOCCPnames(0)
GetSummaryStatistics(age2564final, "OCCP")
GetSummaryStatistics(age2564final, "OCCP", TRUE)
GetSummaryStatistics(age2564final, "JWAP")
GetSummaryStatistics(age2564final, "WKW")
GetSummaryStatistics(age2564final, "WRK")
GetSummaryStatistics(age2564final, "MAR")
GetSummaryStatistics(age2564final, "MARHT")
GetSummaryStatistics(age2564final, "ENG")
GetSummaryStatistics(age2564final, "JWTR")
GetSummaryStatistics(age2564final, "QTRBIR")
GetSummaryStatistics(age2564final, "RAC3P")
GetSummaryStatistics(age2564final, "SEX")
ggplot(age2564final, aes(x = WKHP), weights = PWGTP) +
  geom_histogram(color = "red", fill = "white", bins = 100) +
  scale_x_continuous(limits = c(0, 100), breaks = seq(0, 100, by = 10)) +
  xlab("Usual Hours Worked Per Week") + ylab("Count") +
  labs(title = "Histogram of Usual Hours Worked Per Week Age 25-64 in ACS 2018")
GetSummaryStatistics(age2564final, "WKHP")
```

```
GetSummaryStatistics(age2564final, "SCHL")
GetSummaryStatistics(age2564final, "HISP")
ggplot(age2564final, aes(x = JWMNP), weights = PWGTP) +
  geom_histogram(color = "red", fill = "white", bins = 100) +
  scale_x_continuous(limits = c(0, 100), breaks = seq(0, 100, by = 10)) +
  xlab("Travel Time to Work (minutes)") + ylab("Count") +
  labs(title = "Histogram of Travel Time to Work Age 25-64 in ACS 2018")
GetSummaryStatistics(age2564final, "JWMNP")
GetSummaryStatistics(age2564final, "PUMA", TRUE)
GetSummaryStatistics(age2564final, "AGEP")
GetSummaryStatistics(age2564final, "FOD1P")

GetSummaryStatistics(age2564final, "FOD1P", TRUE)

age2564final$WKHP[age2564final$WKHP > 70] = 71
age2564final$WKHPF = convertWKHP(age2564final$WKHP)
age2564final$AGEPF = convertAGEP(age2564final$AGEP)
age2564final$JWMNPF = convertJWMNP(age2564final$JWMNP)
age2564final$JWAPF = convertJWAP(age2564final$JWAP)
convert = c("WKW","WRK","MAR","MARHT","ENG","HISP","JWTR","QTRBIR","RAC3P","SEX", "SCHL")
age2564final = convertList(age2564final, convert)

Formula = c("PERNP~WKW","PERNP~WRK","PERNP~MAR","PERNP~MARHT","PERNP~ENG",
            "PERNP~JWTR","PERNP~QTRBIR","PERNP~RAC3P","PERNP~HISP","PERNP~SEX","PERNP~SCHL")
regressionList(age2564final, Formula, title = "PERNP~")

Second = "PERNP~WKW+WRK+MAR+ENG+JWTR+RAC3P+SEX+SCHL"


Second = "PERNP~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL"
FormulaWKHP = GetFormulaVariations(Second,"WKHP")
regressionList(age2564final, FormulaWKHP, title = Second)
Third = paste0(Second,"+WKHPF")
FormulaAGEP = GetFormulaVariations(Third,"AGEP")
Table3 = regressionList(age2564final, FormulaAGEP, title = Third)
Fourth = paste0(Third, "+sqrt(AGEP)+AGEP")
FormulaJWMNP = GetFormulaVariations(Fourth,"JWMNP")
Table4 = regressionList(age2564final, FormulaJWMNP[-3], title = Fourth)
Fifth = paste0(Fourth, "+JWMNPF")
FormulaJWAP = GetFormulaVariations(Fifth,"JWAP")
Table5 = regressionList(age2564final, FormulaJWAP[-3], title = Fifth)

Sixth = paste0(Fifth, "+
               JWAPF")
groups = c(2,5,10,20,30,40,50)
Table6 = regressionGroup(age2564final, Sixth, groups, "OCCP")


age2564final$OCCPY = convertOCCPquantiles(age2564final, "PERNP", "PWGTP",
                                          seq(0,1,length = 41))
Seventh= paste0(Sixth, "+OCCPY")
Table7 = regressionGroup(age2564final, Seventh, groups, "NAICSP")

age2564final$NAICSPY = convertNAICSPquantiles(age2564final, "PERNP", "PWGTP",
                                          seq(0,1,length =51))
Eighth = paste0(Seventh, "+NAICSPY")
```

```
Table8 = regressionGroup(age2564final,
                Eighth, groups, "PUMA")


groups = c(5,10,20,30,40,50)
age2564final$PUMAY = convertPUMAquantiles(age2564final, "PERNP", "PWGTP",
                                          seq(0,1,length = 41))
Nineth = paste0(Eighth,"+PUMAY")
Table9 = regressionGroup(age2564final, Nineth, groups, "FOD1P")




Formula = c("PERNPHR~WKW","PERNPHR~WRK","PERNPHR~MAR","PERNPHR~MARHT",
            "PERNPHR~ENG","PERNPHR~JWTR","PERNPHR~QTRBIR","PERNPHR~RAC3P",
            "PERNPHR~HISP","PERNPHR~SEX","PERNPHR~SCHL")
regressionList(age2564final, Formula, title = "PERNPHR~")



Second = "PERNPHR~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL"
FormulaWKHP = GetFormulaVariations(Second,"WKHP")
regressionList(age2564final, FormulaWKHP, title = Second)
Third = paste0(Second,"+WKHPF")
FormulaAGEP = GetFormulaVariations(Third,"AGEP")
Table12 = regressionList(age2564final, FormulaAGEP, title = Third)
Fourth = paste0(Third,"+sqrt(AGEP)+AGEP")
FormulaJWMNP = GetFormulaVariations(Fourth,"JWMNP")
Table13 = regressionList(age2564final, FormulaJWMNP[-3], title = Fourth)

Sixth =paste0(Fourth, "+JWMNPF+
              JWAPF")
groups = c(2,5,10,20,30,40,50)
Table14 = regressionGroup(age2564final, Sixth, groups, "OCCP","PERNPHR")



age2564final$OCCPH = convertOCCPquantiles(age2564final, "PERNPHR", "PWGTP",
                                          seq(0,1,length = 51))
Seventh= paste0(Sixth,"+OCCPH")
Table15 = regressionGroup(age2564final, Seventh, groups, "NAICSP","PERNPHR")

age2564final$NAICSPH = convertNAICSPquantiles(age2564final, "PERNPHR", "PWGTP",
                                          seq(0,1,length  = 51))
Eighth = paste0(Seventh, "+NAICSPH")
Table16 = regressionGroup(age2564final,
                Eighth, groups, "PUMA","PERNPHR")


groups = c(5,10,20,30,40,50)
age2564final$PUMAH = convertPUMAquantiles(age2564final, "PERNPHR", "PWGTP",
                                          seq(0,1,length = 41))
Nineth = paste0(Eighth,"+PUMAH")
Table17 = regressionGroup(age2564final, Nineth, groups, "FOD1P","PERNPHR")
```

```
age2564final$FOD1PY = convertFOD1Pquantiles(age2564final, "PERNP", "PWGTP",
                                            seq(0,1,length = 51))
age2564final$FOD1PH = convertFOD1Pquantiles(age2564final, "PERNPHR", "PWGTP",
                                            seq(0,1,length = 51))
BaseYear = "PERNP~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+
JWMNPF+JWAPF"
BaseHour = "PERNPHR~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+
JWMNPF+JWAPF"
NormYr = paste0(BaseYear,"+OCCPY+NAICSPY+PUMAY+FOD1PY")
SwitchYr = paste0(BaseYear,"+OCCPH+NAICSPH+PUMAH+FOD1PH")
SwitchYr2 = paste0(BaseYear, "+OCCPY+NAICSPY+PUMAH+FOD1PY")
SwitchYr3 = paste0(BaseYear, "+OCCPH+NAICSPY+PUMAY+FOD1PY")
SwitchYr4 = paste0(BaseYear, "+OCCPH+NAICSPY+PUMAH+FOD1PY")

#"PERNP~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHP+sqrt(AGEP)+AGEP+JWMNP+JWAP+OCCPH+NAICSPY+PUMAH+FOD1P
ItemYear = c(NormYr, SwitchYr, SwitchYr2, SwitchYr3, SwitchYr4)

regressionList(age2564final, ItemYear, title = BaseYear)
NormHr = paste0(BaseHour,"+OCCPY+NAICSPY+PUMAY+FOD1PY")
SwitchHr = paste0(BaseHour,"+OCCPH+NAICSPH+PUMAH+FOD1PH")
SwitchHr2 = paste0(BaseHour, "+OCCPY+NAICSPY+PUMAH+FOD1PY")
SwitchHr3 = paste0(BaseHour, "+OCCPH+NAICSPY+PUMAY+FOD1PY")
SwitchHr4 = paste0(BaseHour, "+OCCPH+NAICSPY+PUMAH+FOD1PY")

ItemHr = c(NormHr, SwitchHr, SwitchHr2, SwitchHr3, SwitchHr4)
regressionList(age2564final, ItemHr, title = BaseHour)
Hour = paste0(SwitchHr4, "+SEX:log2(AGEP)+SCHL:log2(AGEP)+SEX:MAR")
Year = paste0(SwitchYr4, "+SEX:log2(AGEP)+SCHL:log2(AGEP)+SEX:MAR")

regressionList(age2564final, Hour, title = SwitchHr4)
regressionList(age2564final, Year, title = SwitchYr4)
rows = c("Variable", "Model for both Earnings and Earnings per Year", "Df")
R1 = c("WKW","Same Factors as Summary Statistics" , "5")
R2 = c("WRK", "Same Factors as Summary Statistics", "2")
R3 = c("MAR", "Same Factors as Summary Statistics", "4")
R4 = c("ENG", "Same Factors as Summary Statistics", "4")
R5 = c("JWTR", "Same Factors as Summary Statistics", "11")
R6 = c("RAC3P", "Same Factors as Summary Statistics", "8")
R7 = c("HISP", "Same Factors as Summary Statistics", "2")
R8 = c("SEX", "Same Factors as Summary Statistics", "1")
R9 = c("SCHL", "Same Factors as Summary Statistics", "7")
R10 = c("WKHP", "Same Factors as Summary Statistics", "15")
R11 = c("AGEP", "Linear variable of age plus Square Root variable of age", "2")
R12 = c("JWMNP", "Same as Summary Statistics", "15")
R13 = c("JWAP", "Same as Summary Statistics", "28")
R14 = c("OCCP", "50 equal groups sorted by Hourly Earnings", "46")
R15 = c("NAICSP", "50 equal groups sorted by Yearly Earnings", "41")
R16 = c("PUMA", "40 equal groups sorted by Hourly Earnings", "39")
R17 = c("FOD1P", "50 equal groups sorted by Yearly Earnings", "18")
R18 = c("SEX:AGEP", "Interaction between Sex and log base 2 of Age", "2")
R19 = c("SCHL:AGEP", "Interaction between School and log base 2 of Age", "7")
R20 = c("SEX:MAR", "Interaction between Sex and Marital Status", "4")
R21 = c("MARHT", "Not included due to overlap with MAR", "0")
R22 = c("QTRBIR", "Not included due to insignificance", "0")

Tab = rbind(R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12,R13,R14,R15,R16,R17,R18,R19,R20,R21,R22)
```

```
rownames(Tab) = c(1:22)
colnames(Tab) = rows
capt = "Final Model"
knitr::kable(Tab, caption = capt, booktabs = TRUE, linesep = "") %>%
  kable_styling(latex_options = "HOLD_position") %>%
  column_spec(2, width = "32em", border_left = TRUE) %>%
  column_spec(3, border_left = TRUE)
library(readr)
#Hour.model = lm(formula("PERNPHR~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JW
#ANOVAhour = capture.output(anova(Hour.model))
#Year.model = lm(formula("PERNP~WKW+WRK+MAR+ENG+JWTR+RAC3P+HISP+SEX+SCHL+WKHPF+sqrt(AGEP)+AGEP+JWMNPF+JWAP
#ANOVAyear = capture.output(anova(Year.model))
#Hour.small = Hour.model[-8]
#Year.small = Hour.model[-8]
#Hour.small$cov = vcov(Hour.model)
#Year.small$cov = vcov(Year.model)
load("ANOVAhour.rds")
load("ANOVAyear.rds")
ANOVAhour = read_fwf(ANOVAhour[5:26], col_positions = fwf_widths(c(16, 7, 9,8,9,10)))
colnames(ANOVAhour) = c("Variable","Df","Sum Sq","Mean Sq", "F value","Pr(>F)")
knitr::kable(ANOVAhour, caption = "ANOVA table for Hour model", booktabs = TRUE, linesep = "") %>%
  kable_styling(latex_options = "HOLD_position") %>%
  column_spec(2:6, border_left = TRUE)

ANOVAyear = read_fwf(ANOVAyear[5:26], col_positions = fwf_widths(c(15, 8, 9,8,10,13)))
colnames(ANOVAyear) = c("Variable","Df","Sum Sq","Mean Sq", "F value","Pr(>F)")
knitr::kable(ANOVAyear, caption = "ANOVA table for Year model", booktabs = TRUE, linesep = "") %>%
  kable_styling(latex_options = "HOLD_position") %>%
  column_spec(2:6, border_left = TRUE)
#car::ncvTest(Year.model)
"Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 35525.12, Df = 1, p = < 2.22e-16"
#lmtest::bptest(Year.model)
"   studentized Breusch-Pagan test

data:  Year.model
BP = 75400, df = 261, p-value < 2.2e-16"
#car::ncvTest(Hour.model)
"Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 60.62692, Df = 1, p = 6.8984e-15"
#lmtest::bptest(Hour.model)
"   studentized Breusch-Pagan test

data:  Hour.model
BP = 76349, df = 261, p-value < 2.2e-16"

load("Replications.rds")
PrintReplicationsTable(Replications)
library(viridis)
load("HourSmall.rds")
G = as.data.frame(cbind(Hour.small$fitted.values, Hour.small$residuals))
ggplot(G, aes(x = V1, y = V2)) +
  geom_hex(aes(fill = stat(sqrt(count))), bins = 30) +
  scale_fill_viridis() + theme_bw() +
```

```
  xlab("Fitted Values") + ylab("Residuals") +
  labs(title = "Residuals vs Fitted Values for Hour model")
load("YearSmall.rds")
G = as.data.frame(cbind(Year.small$fitted.values, Year.small$residuals))
ggplot(G, aes(x = V1, y = V2)) +
  geom_hex(aes(fill = stat(sqrt(count))), bins = 30) +
  scale_fill_viridis() + theme_bw() +
  xlab("Fitted Values") + ylab("Residuals") +
  labs(title = "Residuals vs Fitted Values for Year model")

Residuals = sample(Hour.small$residuals, 20000)
Residuals = data.frame(Residuals)
y = quantile(Residuals[[1]], c(0.25,0.75))
x = qnorm(c(0.25,0.75))
slope = diff(y)/diff(x)
int = y[1L] - slope * x[1L]
ggplot(Residuals, aes(sample = Residuals)) + stat_qq() +
  geom_abline(slope = slope, intercept = int) +
  labs(title = "Q-Q Plot of Residuals for the Earnings per Hour Model")
load("YearSmall.rds")
Residuals = sample(Year.small$residuals, 20000)
Residuals = data.frame(Residuals)
y = quantile(Residuals[[1]], c(0.25,0.75))
x = qnorm(c(0.25,0.75))
slope = diff(y)/diff(x)
int = y[1L] - slope * x[1L]
ggplot(Residuals, aes(sample = Residuals)) + stat_qq() +
  geom_abline(slope = slope, intercept = int) +
  labs(title = "Q-Q Plot of Residuals for the Earnings per Year Model")
load("SEyear.rds")
load("SEhour.rds")
Hour.small$coefficients = sHr
Year.small$coefficients = sYr
PrintConfidenceIntervalTable(Hour.small,Year.small,"HISP","HISP")

PrintConfidenceIntervalTable(Hour.small,Year.small,"ENG","ENG")
PrintConfidenceIntervalTable(Hour.small,Year.small,"JWTR","JWTR")
PrintSCHLTable(Hour.small)
PrintMARTable(Hour.small)
B = ConfidenceInterval(Hour.small,"WKHP","WKHPF")
B[2:4] = B[2:4] - B[2,2]
B$WKHP = factor(B$WKHP , levels = B$WKHP)
ggplot(B[2:16,]) +
  geom_bar(aes(x = WKHP, y = Estimate, color = WKHP, fill = WKHP), stat = "identity")+
  geom_errorbar(aes(x = WKHP, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  xlab("Usual Hours Worked Per Week") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Hour to 6-10 hours per week",
       subtitle = "With 99% confidence interval error bars")
B = ConfidenceInterval(Year.small,"WKHP","WKHPF")
B[2:4] = B[2:4] - B[2,2]
B$WKHP = factor(B$WKHP , levels = B$WKHP )
ggplot(B[2:16,]) +
  geom_bar(aes(x = WKHP, y = Estimate, color = WKHP, fill = WKHP), stat = "identity")+
```

```
  geom_errorbar(aes(x = WKHP, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))  +
  xlab("Usual Hours Worked Per Week") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Year to 6-10 hours per week",
       subtitle = "With 99% confidence interval error bars")
B = ConfidenceInterval(Hour.small,"JWMNP","JWMNPF")
B$JWMNP = factor(B$JWMNP, levels = B$JWMNP)
ggplot(B) +
  geom_bar(aes(x = JWMNP, y = Estimate, color = JWMNP, fill = JWMNP), stat = "identity")+
  geom_errorbar(aes(x = JWMNP, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  xlab("Travel time to work (minutes)") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Hour to 0 minute commute time",
       subtitle = "With 99% confidence interval error bars")
B = ConfidenceInterval(Hour.small,"RAC3P","RAC3P")
B[,1] = gsub("Alone|alone","",B[,1])
B$RAC3P[7] = "Other Asian"
B$RAC3P = factor(B$RAC3P, levels = B$RAC3P)
ggplot(B) +
  geom_bar(aes(x = RAC3P, y = Estimate, color = RAC3P, fill = RAC3P), stat = "identity")+
  geom_errorbar(aes(x = RAC3P, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  xlab("Race Groups") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Hour to White alone",
       subtitle = "With 99% confidence interval error bars")
B = ConfidenceInterval(Hour.small,"JWTR","JWTR")
B[,1] = gsub("Worked at home","Home",B[,1])
B$JWTR= factor(B$JWTR, levels = B$JWTR)

ggplot(B) +
  geom_bar(aes(x = JWTR, y = Estimate, color = JWTR, fill = JWTR), stat = "identity")+
  geom_errorbar(aes(x = JWTR, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  xlab("Means of transport to work") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Hour to Not at Work",
       subtitle = "With 99% confidence interval error bars")

Occupation = GetCoefficientPercent(Hour.small, "OCCP","OCCPH")
Industry = GetCoefficientPercent(Hour.small, "NAICSP","NAICSPY")
Area = GetCoefficientPercent(Hour.small, "PUMA","PUMAH")
Field = GetCoefficientPercent(Hour.small,"FOD1P","FOD1PY")

A= merge(Occupation, Industry, all.x = TRUE, all.y=TRUE)
A= merge(A, Area, all.x=TRUE, all.y=TRUE)
A = merge(A, Field, all.x=TRUE, all.y=TRUE)
names(A) = c("Percent","Occupation","Industry","Area","Field of Degree")
Full = reshape2::melt(A, id.var = "Percent")
A[,2] = A[,2] - A[41,2]
```

```
A[,3] = A[,3] - A[41,3]
A[,4] = A[,4] - A[40,4]
Full = reshape2::melt(A, id.var = "Percent")
ggplot(Full, aes(x = Percent, y = value, col = variable)) +
  geom_point() +
  geom_smooth(method = 'loess', se = FALSE) +
  xlab("Percent of Distribution") +
  ylab("Natural Log Relative Income to 50th percentile") +
  labs(title = "Relative Income by percentile of distribution using Hour model") +
  scale_y_continuous(breaks = scales::pretty_breaks(10))
library(gtools)
B = ConfidenceInterval(Hour.small,"OCCPH","OCCPH")
B = B[mixedorder(B$OCCPH),]
B[,2:4] = B[,2:4] - B[25,2]
B$OCCPH = factor(B$OCCPH, levels = B$OCCPH)
ggplot(B) +
  geom_bar(aes(x = OCCPH, y = Estimate, color = OCCPH, fill = OCCPH),
           show.legend = FALSE, stat = "identity")+
  geom_errorbar(aes(x = OCCPH, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  scale_y_continuous(breaks = scales::pretty_breaks(5)) +
  xlab("Percentile of Occupation by Log-Average Earnings per Hour") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Hour to 50th percentile Occupation",
       subtitle = "With 99% confidence interval error bars")

B = ConfidenceInterval(Hour.small,"PUMAH","PUMAH")
B = B[mixedorder(B$PUMAH),]
B[,2:4] = B[,2:4] - B[21,2]
B$PUMAH = factor(B$PUMAH, levels = B$PUMAH)
ggplot(B) +
  geom_bar(aes(x = PUMAH, y = Estimate, color = PUMAH, fill = PUMAH),
           show.legend = FALSE, stat = "identity")+
  geom_errorbar(aes(x = PUMAH, ymin = get("99% Lower"), ymax = get("99% Upper")),
                color = "orange", size = 1.3) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  scale_y_continuous(breaks = scales::pretty_breaks(5)) +
  xlab("Percentile of Area by Log-Average Earnings per Hour") +
  ylab("Natural Log") +
  labs(title = "Relative Earnings per Hour to 50th percentile Area",
       subtitle = "With 99% confidence interval error bars")
knitr::opts_chunk$set(echo = TRUE, comment = NA)

library(dplyr)
set.seed(1)

load("age2564final.rds")
source("FunctionsForFinalProject.R")

data_final <- age2564final

data_final <- convertList(data_final,
                          c("SCHL","ENG","JWAP","SEX","RAC3P"))
data_final$WKHPF <- as.factor(ifelse(data_final$WKHP >= 30 & data_final$WKW %in% c(1:3), "Full-Time", "Une
data_final$SCHLF = data_final$SCHL
```

```r
data_final$RAC3PF = data_final$RAC3P
data_final$SEXF = data_final$SEX
data_final$ENGF = data_final$ENG

marriagestatus = c(0,2,5)
name = c("Married","Not Married")
data_final$MAR = cut(data_final$MAR, breaks = marriagestatus)
levels(data_final$MAR) = name



data_final$PUMA = as.numeric(convertPUMAquantiles(data_final, "PERNP", "PWGTP", seq(0,1,0.02)))
data_final$OCCP = as.numeric(convertOCCPquantiles(data_final, "PERNP","PWGTP", seq(0,1,0.02)))

data_final$PERNP[data_final$PERNP < 0] = 0
data_final$PERNP = sqrt(data_final$PERNP)
data_final$JWAP = as.numeric(data_final$JWAP)
library(nnet)
library(class)
library(MASS)

data_final$PUMA = jitter(data_final$PUMA)
data_final$SCHL <- as.numeric(data_final$SCHL)
data_final$SEX <- as.numeric(data_final$SEX)
data_final$RAC3P <- as.numeric(data_final$RAC3P)
data_final$ENG <- as.numeric(data_final$ENG)
Spl = sample(nrow(data_final),100000)

data_test <- data_final[Spl,]
data_train <- data_final[-Spl,]

lda_fit <- lda(MAR ~ SCHL + SEX + AGEP + sqrt(AGEP) + JWAP + OCCP + PUMA + RAC3P + PERNP + ENG, data = dat
lda_pred <- predict(lda_fit, data_test)
lda_cm <- table(Predicted = lda_pred$class, Actual = data_test$MAR)
lda_cm
lda_accu <- sum(diag(lda_cm)) / sum(lda_cm)
print(paste0("The accuracy is ", round(lda_accu, 4)*100, "%."))
lda_fit2 <- lda(WKHPF ~ SCHL + SEX + AGEP + sqrt(AGEP) + JWAP + OCCP + PUMA +  RAC3P + ENG, data = data_tr
lda_pred2 <- predict(lda_fit2, data_test)
lda_cm2 <- table(Predicted = lda_pred2$class, Actual = data_test$WKHPF)
lda_cm2
lda_accu2 <- sum(diag(lda_cm2)) / sum(lda_cm2)
print(paste0("The accuracy is ", round(lda_accu2, 4) * 100, "%."))
glm_fit <- glm(MAR ~ SCHLF + AGEP + sqrt(AGEP) + SEXF + ENGF + RAC3PF + JWAP + OCCP + PUMA + PERNP, weight
predicted = ifelse(predict(glm_fit, data_test, type = "response")<0.5,
                   "Married","Not Married")
glm_cm <- table(predicted, data_test$MAR, dnn = c("Predicted Marital Status","True Marital Status"))
glm_cm
glm_accu <- sum(diag(glm_cm)) / sum(glm_cm)
print(paste0("The accuracy is ", round(glm_accu, 4) * 100, "%."))
glm_fit2 <- glm(WKHPF ~ SCHLF + AGEP + sqrt(AGEP) + SEXF + RAC3PF + JWAP + OCCP + PUMA + MAR + ENGF, weigh
predicted = ifelse(predict(glm_fit2, data_test, type = "response")<0.5,
                   "Full-Time","Unemployed/Part-Time")
glm_cm2 <- table(predicted, data_test$WKHPF, dnn = c("Predicted Labor Force","True Labor Force"))
glm_cm2
glm_accu2 <- sum(diag(glm_cm2)) / sum(glm_cm2)
```

```r
print(paste0("The accuracy is ", round(glm_accu2, 4) * 100, "%."))
#PERNP, OCCPY,RAC3P,SEX,SCHL,AGEP,PUMAY
knn_10 <- knn(data_train[,c(1, 3, 12, 13, 15, 18, 19)], data_test[,c(1, 3, 12, 13, 15, 18, 19)], data_trai

knn_10_cm <- table(Predicted = knn_10, Actual = data_test$MAR)
knn_10_cm
knn_10_accu <- sum(diag(knn_10_cm)) / sum(knn_10_cm)
print(paste0("The accuracy for k = 5 is ", round(knn_10_accu, 4) * 100, "%."))
knn_20 <- knn(data_train[,c(1, 3, 12, 13, 15, 18, 19)], data_test[,c(1, 3, 12, 13, 15, 18, 19)], data_trai
knn_20_cm <- table(Predicted = knn_20, Actual = data_test$MAR)
knn_20_cm
knn_20_accu <- sum(diag(knn_20_cm)) / sum(knn_20_cm)
print(paste0("The accuracy for k = 9 is ", round(knn_20_accu, 4) * 100, "%."))
knn_30 <- knn(data_train[,c(1, 3, 12, 13, 15, 18, 19)], data_test[,c(1, 3, 12, 13, 15, 18, 19)], data_trai

knn_30_cm <- table(Predicted = knn_30, Actual = data_test$MAR)
knn_30_cm
knn_30_accu <- sum(diag(knn_30_cm)) / sum(knn_30_cm)
print(paste0("The accuracy for k = 19 is ", round(knn_30_accu, 4) * 100, "%."))
knn2_10 <- knn(data_train[,c(3, 4, 12, 13, 15, 18, 19)], data_test[,c(3, 4, 12, 13, 15, 18, 19)], data_tra

knn2_10_cm <- table(Predicted = knn2_10,Actual = data_test$WKHPF)
knn2_10_cm
knn2_10_accu <- sum(diag(knn2_10_cm)) / sum(knn2_10_cm)
print(paste0("The accuracy for k = 5 is ", round(knn2_10_accu, 4) * 100, "%."))
knn2_20 <- knn(data_train[,c(3, 4, 12, 13, 15, 18, 19)], data_test[,c(3, 4, 12, 13, 15, 18, 19)], data_tra

knn2_20_cm <- table( Predicted = knn2_20, Actual = data_test$WKHPF)
knn2_20_cm
knn2_20_accu <- sum(diag(knn2_20_cm)) / sum(knn2_20_cm)
print(paste0("The accuracy for k = 9 is ", round(knn2_20_accu, 4) * 100, "%."))
knn2_30 <- knn(data_train[,c(3, 4, 12, 13, 15, 18, 19)], data_test[,c(3, 4, 12, 13, 15, 18, 19)], data_tra

knn2_30_cm <- table(Predicted = knn2_30, Actual = data_test$WKHPF)
knn2_30_cm
knn2_30_accu <- sum(diag(knn2_30_cm)) / sum(knn2_30_cm)
print(paste0("The accuracy for k = 19 is ", round(knn2_30_accu, 4) * 100, "%."))
print(sessionInfo())
#Tables now listed here
Table3
Table4
Table5
Table6
Table7
Table8
Table9
Table12
Table13
Table14
Table15
Table16
Table17
```