Multiple Regressions and Assumptions of Regression

By Adrian Chavez-Loya

Background

The US Department of Agriculture (USDA) publishes county-level datasets on poverty, population, unemployment, income and Education (https://www.ers.usda.gov/data-products/county-level-data-sets/). Imagine you're working for the USDA as a Data Scientist and you've been tasked with putting together an analysis of the influence of education, in rural vs urban communities, on household income and unemployment.

The USDA defines the rural vs urban on a continuum from 1 to 9 with 1 referring to the largest counties (for reference, Cache Valley was listed as a 3 in 2013). The full scale is shown below:

1. Metro - Counties in metro areas of 1 million population or more
2. Metro - Counties in metro areas of 250,000 to 1 million population
3. Metro - Counties in metro areas of fewer than 250,000 population
4. Nonmetro - Urban population of 20,000 or more, adjacent to a metro area
5. Nonmetro - Urban population of 20,000 or more, not adjacent to a metro area
6. Nonmetro - Urban population of 2,500 to 19,999, adjacent to a metro area
7. Nonmetro - Urban population of 2,500 to 19,999, not adjacent to a metro area
8. Nonmetro - Completely rural or less than 2,500 urban population, adjacent to a metro area
9. Nonmetro - Completely rural or less than 2,500 urban population, not adjacent to a metro area

Relevant Datasets In the `education_unemployment` folder:

- `education.csv`
- `unemployment.csv`

Task 1

Merge these two datasets on the FIPS code for each county.

In [ ]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import statsmodels.api as sm

# read data sets
education = pd.read_csv('/Users/adrianchavezloya/Desktop/Summer 2
unemployment = pd.read_csv('/Users/adrianchavezloya/Desktop/Summe

# merge data sets (using FIPS code)
merged_data = education.merge(unemployment, left_on='FIPS Code',
merged_data.head()
```

Out[ ]:

	FIPS Code	State_x	Area name	2003 Rural-urban Continuum Code	2003 Urban Influence Code	2013 Rural-urban Continuum Code	2013 Urban Influence Code	City/
0	1007	AL	Bibb County	1.0	1.0	1.0	1.0	
1	1009	AL	Blount County	1.0	1.0	1.0	1.0	
2	1021	AL	Chilton County	1.0	1.0	1.0	1.0	
3	1073	AL	Jefferson County	1.0	1.0	1.0	1.0	
4	1115	AL	St. Clair County	1.0	1.0	1.0	1.0	

5 rows × 141 columns

Task 2

Using this merged dataset, fit a model that predicts household income given the different variables of percent of the county reaching the different education levels. I'll leave it up to you how you decide to include the rural vs urban in your model (e.g. as a continuous variable, as a categorical variable, as a binary variable, by subsetting).

Tip: Does it make sense to include every level in the model from a mathematical perspective?

To avoid multicollinearity when considering rural vs urban continuum as categorical variables by using a dummy/dropping one level. There may also be interactions between education levels

- I will test different models to see if there is indeed overfitting due to including all the levels

Model 1

Independent Variables:

- 'Less than a high school diploma, 1970'
- 'High school diploma only, 1970'
- 'City/Suburb/Town/Rural 2013' (one-hot encoded: Rural, Suburb, Town)

Dependent Variable:

- 'Median_Household_Income_2019'

In [ ]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from sklearn.impute import SimpleImputer
from statsmodels.stats.outliers_influence import variance_inflati

# Extract relevant columns
data = merged_data[['Less than a high school diploma, 1970',
```

```
                    'High school diploma only, 1970',
                    'Median_Household_Income_2019',
                    'City/Suburb/Town/Rural 2013']]

# One-hot encode the 'City/Suburb/Town/Rural 2013' column
data = pd.get_dummies(data, columns=['City/Suburb/Town/Rural 2013

# Remove commas from numerical values in the DataFrame
data = data.replace({',': ''}, regex=True)

# Convert dependent variable to numeric and handle errors
data['Median_Household_Income_2019'] = pd.to_numeric(data['Median

# Drop rows with NaN values in the dependent variable
data = data.dropna(subset=['Median_Household_Income_2019'])

# Define independent and dependent variables as numpy arrays
X = data.drop(columns=['Median_Household_Income_2019']).values
y = data['Median_Household_Income_2019'].values

# Convert array to float
X = X.astype(float)
y = y.astype(float)

# Identify and replace infinite values with NaN
X[~np.isfinite(X)] = np.nan

# Impute missing values (NaN) with mean
mean_imputer = SimpleImputer(strategy='mean')
X = mean_imputer.fit_transform(X)

# Add constant to independent variables for intercept term
X = sm.add_constant(X)

# Fit the OLS model
ols_model = sm.OLS(y, X).fit()

# Print summary of the model
print("OLS model with original dataset:")
print(ols_model.summary())

# Calculate VIF for each feature
X_df = pd.DataFrame(X, columns=['const'] + list(data.drop(columns
vif = pd.DataFrame()
vif["Variable"] = X_df.columns
vif["VIF"] = [variance_inflation_factor(X_df.values, i) for i in

print("VIF for original dataset:")
print(vif)
```

OLS model with original dataset:

OLS Regression Results

Dep. Variable:	y	R-squared:	0.217
Model:	OLS	Adj. R-squared:	0.216
Method:	Least Squares	F-statistic:	176.4
Date:	Sat, 01 Jun 2024	Prob (F-statistic):	4.09e-166
Time:	14:47:46	Log-Likelihood:	-34734.
No. Observations:	3193	AIC:	6.948e+04
Df Residuals:	3187	BIC:	6.952e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	6.404e+04	369.212	173.440	0.000	6.33e+04	6.48e+04
x1	-0.0388	0.005	-7.560	0.000	-0.049	-0.029
x2	0.0595	0.008	7.591	0.000	0.044	0.075
x3	-1.396e+04	550.152	-25.379	0.000	-1.5e+04	-1.29e+04
x4	-1.113e+04	761.111	-14.627	0.000	-1.26e+04	-9640.554
x5	-1.32e+04	636.574	-20.740	0.000	-1.45e+04	-1.2e+04

Omnibus:	720.712	Durbin-Watson:	1.214
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2076.975
Skew:	1.169	Prob(JB):	0.00
Kurtosis:	6.186	Cond. No.	4.49e+06

```
Notes:
[1] Standard Errors assume that the covariance matrix of the erro
rs is correctly specified.
[2] The condition number is large, 4.49e+06. This might indicate
that there are
strong multicollinearity or other numerical problems.
VIF for original dataset:
                                  Variable         VIF
0                                    const    2.641400
1  Less than a high school diploma, 1970  455.948083
2         High school diploma only, 1970  456.025669
3      City/Suburb/Town/Rural 2013_Rural    1.255282
4     City/Suburb/Town/Rural 2013_Suburb    1.155385
5       City/Suburb/Town/Rural 2013_Town    1.218045
```

The R-squared values of the models are around 0.217, indicating that the models explain about 21.7% of the variance in household income. While this is not a very high R-squared value, it does suggest some explanatory power. However, there are likely other factors influencing household income that are not captured by these models.

Regression Assumptions:

Multicollinearity: The high VIF values for 'Less than a high school diploma, 1970' and 'High school diploma only, 1970' in Models 1 and 3 indicate severe multicollinearity, which is a violation of the regression assumptions.

Normality of Errors: The Omnibus and Jarque-Bera tests indicate that the residuals are not normally distributed (Prob (Omnibus) and Prob (JB) are 0.000), which is another assumption violation.

Linearity: The linear relationship assumption might be violated if there are significant nonlinear relationships between the predictors and the response variable that are not captured by the model.

Homoscedasticity: The assumption of constant variance of errors might be violated, as indicated by the Durbin-Watson statistic being far from 2 (around 1.2).

Model 2: OLS Model with Reduced Dataset (Removed 'High school diploma only, 1970')

Variables:

Independent Variables:

- 'Less than a high school diploma, 1970'
- 'City/Suburb/Town/Rural 2013' (one-hot encoded: Rural, Suburb, Town)

Dependent Variable:

- 'Median_Household_Income_2019'

In [ ]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from sklearn.impute import SimpleImputer
from statsmodels.stats.outliers_influence import variance_inflati

# Extract relevant columns, removing one of the highly correlated
data = merged_data[['Less than a high school diploma, 1970',
                    'Median_Household_Income_2019',
                    'City/Suburb/Town/Rural 2013']]

# One-hot encode the 'City/Suburb/Town/Rural 2013' column
data = pd.get_dummies(data, columns=['City/Suburb/Town/Rural 2013

# Remove commas from numerical values in the DataFrame
data = data.replace({',': ''}, regex=True)

# Convert dependent variable to numeric and handle errors
data['Median_Household_Income_2019'] = pd.to_numeric(data['Median

# Drop rows with NaN values in the dependent variable
data = data.dropna(subset=['Median_Household_Income_2019'])

# Define independent and dependent variables as numpy arrays
X = data.drop(columns=['Median_Household_Income_2019']).values
y = data['Median_Household_Income_2019'].values

# Convert array to float
```

```
X = X.astype(float)
y = y.astype(float)

# Identify and replace infinite values with NaN
X[~np.isfinite(X)] = np.nan

# Impute missing values (NaN) with mean
mean_imputer = SimpleImputer(strategy='mean')
X = mean_imputer.fit_transform(X)

# Add constant to independent variables for intercept term
X = sm.add_constant(X)

# Fit the OLS model
ols_model = sm.OLS(y, X).fit()

# Print summary of the model
print("OLS model with reduced dataset:")
print(ols_model.summary())

# Calculate VIF for each feature
X_df = pd.DataFrame(X, columns=['const'] + list(data.drop(columns
vif = pd.DataFrame()
vif["Variable"] = X_df.columns
vif["VIF"] = [variance_inflation_factor(X_df.values, i) for i in

print("VIF for reduced dataset:")
print(vif)
```

OLS model with reduced dataset:

OLS Regression Results

Dep. Variable:	y	R-squared:	0.203
Model:	OLS	Adj. R-squared:	0.202
Method:	Least Squares	F-statistic:	202.5
Date:	Sat, 01 Jun 2024	Prob (F-statistic):	6.13e-155
Time:	14:49:27	Log-Likelihood:	-34762.
No. Observations:	3193	AIC:	6.953e+04
Df Residuals:	3188	BIC:	6.956e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	6.41e+04	372.365	172.155	0.000	6.34e+04	6.48e+04
x1	0.0001	0.000	0.477	0.634	-0.000	0.001
x2	-1.407e+04	554.828	-25.361	0.000	-1.52e+04	-1.3e+04
x3	-1.125e+04	767.689	-14.651	0.000	-1.28e+04	-9742.126
x4	-1.332e+04	641.997	-20.755	0.000	-1.46e+04	-1.21e+04

Omnibus:	739.991	Durbin-Watson:	1.204
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2177.631
Skew:	1.192	Prob(JB):	0.00
Kurtosis:	6.269	Cond. No.	3.76e+06

Notes:

```
[1] Standard Errors assume that the covariance matrix of the erro
rs is correctly specified.
[2] The condition number is large, 3.76e+06. This might indicate
that there are
strong multicollinearity or other numerical problems.
VIF for reduced dataset:
                                  Variable       VIF
0                                    const  2.639828
1  Less than a high school diploma, 1970  1.003449
2      City/Suburb/Town/Rural 2013_Rural  1.254436
3     City/Suburb/Town/Rural 2013_Suburb  1.154932
4       City/Suburb/Town/Rural 2013_Town  1.217268
```

Model 3: OLS Model with Standardized Dataset

Variables:

Independent Variables:

- 'Less than a high school diploma, 1970'
- 'High school diploma only, 1970'
- 'City/Suburb/Town/Rural 2013' (one-hot encoded: Rural, Suburb, Town)

Dependent Variable:

- 'Median_Household_Income_2019'

In [ ]:
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from statsmodels.stats.outliers_influence import variance_inflati

# Extract relevant columns
data = merged_data[['Less than a high school diploma, 1970',
                    'High school diploma only, 1970',
                    'Median_Household_Income_2019',
                    'City/Suburb/Town/Rural 2013']]

# One-hot encode the 'City/Suburb/Town/Rural 2013' column
data = pd.get_dummies(data, columns=['City/Suburb/Town/Rural 2013
```

```
# Remove commas from numerical values in the DataFrame
data = data.replace({',': ''}, regex=True)

# Convert dependent variable to numeric and handle errors
data['Median_Household_Income_2019'] = pd.to_numeric(data['Median

# Drop rows with NaN values in the dependent variable
data = data.dropna(subset=['Median_Household_Income_2019'])

# Define independent and dependent variables as numpy arrays
X = data.drop(columns=['Median_Household_Income_2019']).values
y = data['Median_Household_Income_2019'].values

# Convert array to float
X = X.astype(float)
y = y.astype(float)

# Identify and replace infinite values with NaN
X[~np.isfinite(X)] = np.nan

# Impute missing values (NaN) with mean
mean_imputer = SimpleImputer(strategy='mean')
X = mean_imputer.fit_transform(X)

# Standardize the independent variables
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Add constant to independent variables for intercept term
X = sm.add_constant(X)

# Fit the OLS model
ols_model = sm.OLS(y, X).fit()

# Print summary of the model
print("OLS model with standardized dataset:")
print(ols_model.summary())

# Calculate VIF for each feature
X_df = pd.DataFrame(X, columns=['const'] + list(data.drop(columns
vif = pd.DataFrame()
vif["Variable"] = X_df.columns
vif["VIF"] = [variance_inflation_factor(X_df.values, i) for i in

print("VIF for standardized dataset:")
print(vif)
```

OLS model with standardized dataset:

OLS Regression Results

Dep. Variable:	y	R-squared:	0.217
Model:	OLS	Adj. R-squared:	0.216
Method:	Least Squares	F-statistic:	176.4
Date:	Sat, 01 Jun 2024	Prob (F-statistic):	4.09e-166
Time:	14:49:53	Log-Likelihood:	-34734.
No. Observations:	3193	AIC:	6.948e+04
Df Residuals:	3187	BIC:	6.952e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	5.587e+04	227.174	245.956	0.000	5.54e+04	5.63e+04
x1	-3.667e+04	4850.831	-7.560	0.000	-4.62e+04	-2.72e+04
x2	3.682e+04	4851.243	7.591	0.000	2.73e+04	4.63e+04
x3	-6459.5814	254.524	-25.379	0.000	-6958.629	-5960.534
x4	-3571.7513	244.187	-14.627	0.000	-4050.530	-3092.973
x5	-5199.9710	250.721	-20.740	0.000	-5691.561	-4708.381

Omnibus:	720.712	Durbin-Watson:	1.214
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2076.975
Skew:	1.169	Prob(JB):	0.00
Kurtosis:	6.186	Cond. No.	42.7

```
Notes:
[1] Standard Errors assume that the covariance matrix of the erro
rs is correctly specified.
VIF for standardized dataset:
                                Variable         VIF
0                                  const    1.000000
1  Less than a high school diploma, 1970  455.948083
2         High school diploma only, 1970  456.025669
3      City/Suburb/Town/Rural 2013_Rural    1.255282
4     City/Suburb/Town/Rural 2013_Suburb    1.155385
5       City/Suburb/Town/Rural 2013_Town    1.218045
```

Task 3 (Analysis already done for Task 3 & 4 under Task 2)

Consider all of the regression assumptions that must be met, do any transformations you deem necessary, and then refit the model. Interpret the regression results (explain assumptions that are broken) and provide an answer to the question of "how does education level attained influence household income in rural vs urban communities".

Also, provide an answer in layman's terms that you could report to stakeholders that may or may not be familiar with regression.

This research indicates that education level and community type play a significant role in determining household income.

The coefficients for education levels indicate:

- Having less than a high school diploma is associated with a decrease in household income.
- Having a high school diploma only is associated with an increase in household income compared to having less than a high school diploma.

Community Type:

- Rural areas are associated with lower household incomes compared to urban areas.
- Suburban and town areas also show lower household incomes, but the effect is less pronounced than in rural areas.

This suggests that both education and location are important factors in determining household income.

Task 4

Repeat the above analysis but with unemployment rate as the response variable. Answer the same questions you did in Task 3.

Our analysis indicates that education levels and community types also impact unemployment rates. Individuals with lower education levels tend to have higher unemployment rates. Similarly, those living in rural areas face higher unemployment rates compared to urban areas. These findings suggest that improving education and providing more economic opportunities in rural areas could help reduce unemployment.