

rocchio_assign9

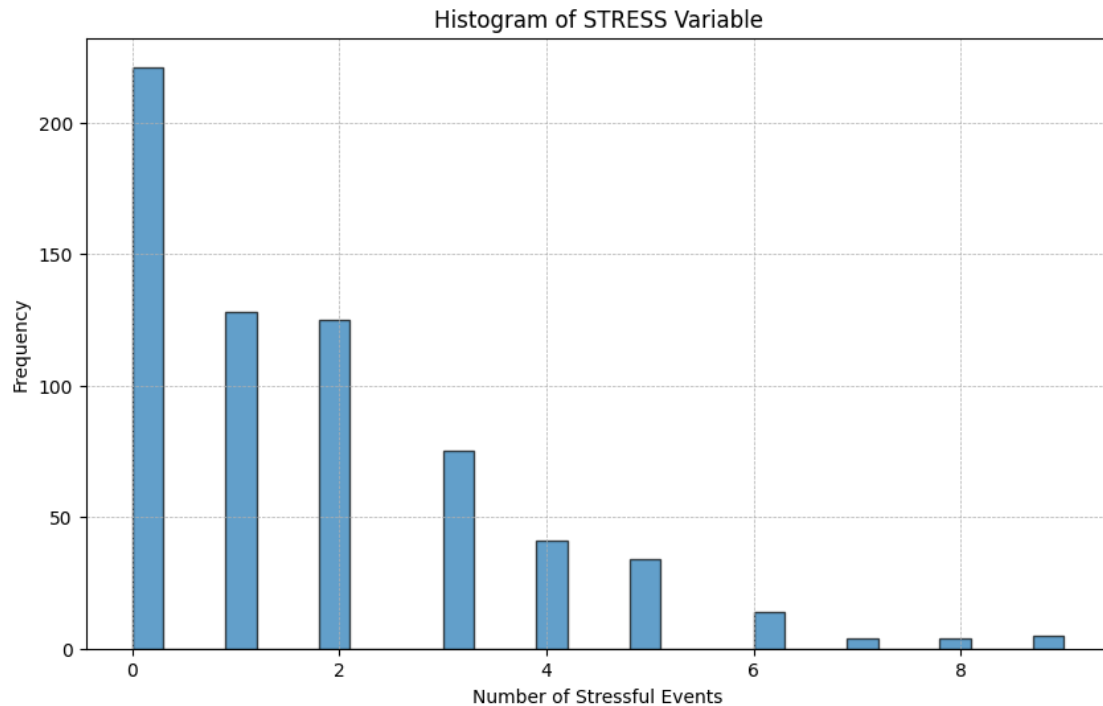
August 21, 2023

```
[1]: import pandas as pd
data = pd.read_excel("STRESS.xlsx")
data.head()
```

```
[1]:  AGE      COHES  ESTEEM  GRADES  SATTACH  STRESS  NEWID
0    11  61.256001   32.0    15.0    21.0      0      5
1    14  49.000000   33.0    17.0    22.0      0      6
2    14  35.000000   27.0    15.0    28.0      0     10
3    11  74.000000   34.0    14.0    33.0      0     15
4    14  58.000000   29.0    15.0    28.0      0     16
```

1 Task 1

```
[2]: import matplotlib.pyplot as plt
plt.figure(figsize=(10, 6))
plt.hist(data['STRESS'], bins=30, edgecolor='k', alpha=0.7)
plt.title('Histogram of STRESS Variable')
plt.xlabel('Number of Stressful Events')
plt.ylabel('Frequency')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
summary_statistics = data['STRESS'].describe()
summary_statistics
```



```
[2]: count    651.000000
     mean      1.729647
     std       1.849082
     min       0.000000
     25%       0.000000
     50%       1.000000
     75%       3.000000
     max       9.000000
     Name: STRESS, dtype: float64
```

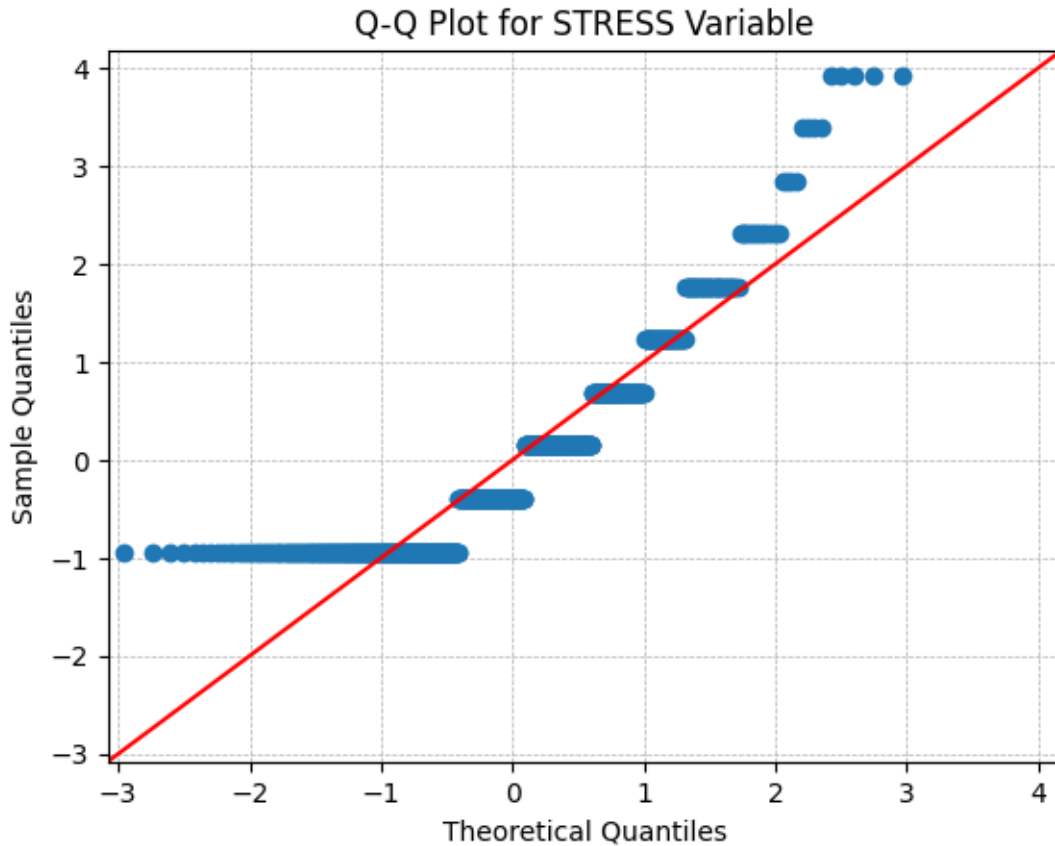
STRESS variable:

- **Count:** 651 observations
- **Mean:** Approximately 1.73 stressful events
- **Standard Deviation:** Approximately 1.85
- **Minimum:** 0 stressful events
- **25th Percentile:** 0 stressful events
- **Median (50th Percentile):** 1 stressful event
- **75th Percentile:** 3 stressful events
- **Maximum:** 9 stressful events

```
[3]: import statsmodels.api as sm
     plt.figure(figsize=(10, 6))
     sm.qqplot(data['STRESS'], line='45', fit=True)
     plt.title('Q-Q Plot for STRESS Variable')
```

```
plt.xlabel('Theoretical Quantiles')
plt.ylabel('Sample Quantiles')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```

<Figure size 1000x600 with 0 Axes>



Q-Q plot:

- The data points deviate from the 45-degree reference line, especially at the lower and upper ends.
- Such a deviation suggests that the **STRESS** variable may not be normally distributed.

Considering the histogram and the Q-Q plot:

- The **STRESS** variable does not seem to be normally distributed. It has a right-skewed shape in the histogram, and the Q-Q plot further supports this observation.
- Given that the **STRESS** variable represents counts of stressful events, it's logical to think that its distribution might follow a Poisson or a related distribution. The Poisson distribution is commonly used to model count data, especially when the counts are relatively small.

In conclusion, the most likely probability distribution for the **STRESS** variable seems to be a Poisson

distribution. This is based on the nature of the data (count data), the shape of the histogram (right-skewed), and the Q-Q plot's deviation from the 45-degree reference line.

2 Task 2

```
[4]: import statsmodels.formula.api as smf
model = smf.ols(formula="STRESS ~ COHES + ESTEEM + GRADES + SATTACH", data=data)
results = model.fit()
results.summary()
```

[4]:

Dep. Variable:	STRESS	R-squared:	0.083
Model:	OLS	Adj. R-squared:	0.078
Method:	Least Squares	F-statistic:	14.65
Date:	Mon, 21 Aug 2023	Prob (F-statistic):	1.83e-11
Time:	06:10:35	Log-Likelihood:	-1295.1
No. Observations:	651	AIC:	2600.
Df Residuals:	646	BIC:	2623.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
Intercept	5.7128	0.581	9.830	0.000	4.572	6.854
COHES	-0.0232	0.007	-3.298	0.001	-0.037	-0.009
ESTEEM	-0.0413	0.019	-2.136	0.033	-0.079	-0.003
GRADES	-0.0417	0.024	-1.773	0.077	-0.088	0.004
SATTACH	-0.0304	0.014	-2.154	0.032	-0.058	-0.003

Omnibus:	109.042	Durbin-Watson:	1.736
Prob(Omnibus):	0.000	Jarque-Bera (JB):	170.611
Skew:	1.087	Prob(JB):	8.96e-38
Kurtosis:	4.252	Cond. No.	584.

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Here are the results from the OLS regression model:

Overall Fit: - R2: 0.083 - This indicates that approximately 8.3% of the variance in **STRESS** is explained by the model. - Adjusted R2: 0.078 - F-statistic: 14.65 with a significance level of $p < 0.001$, which suggests that at least one of the predictors is statistically significant in explaining the variation in **STRESS**.

Coefficients: - **Intercept:** 5.7128 - **COHES:** -0.0232 ($p < 0.001$) - **ESTEEM:** -0.0413 ($p = 0.033$) - **GRADES:** -0.0417 ($p = 0.077$) - **SATTACH:** -0.0304 ($p = 0.032$)

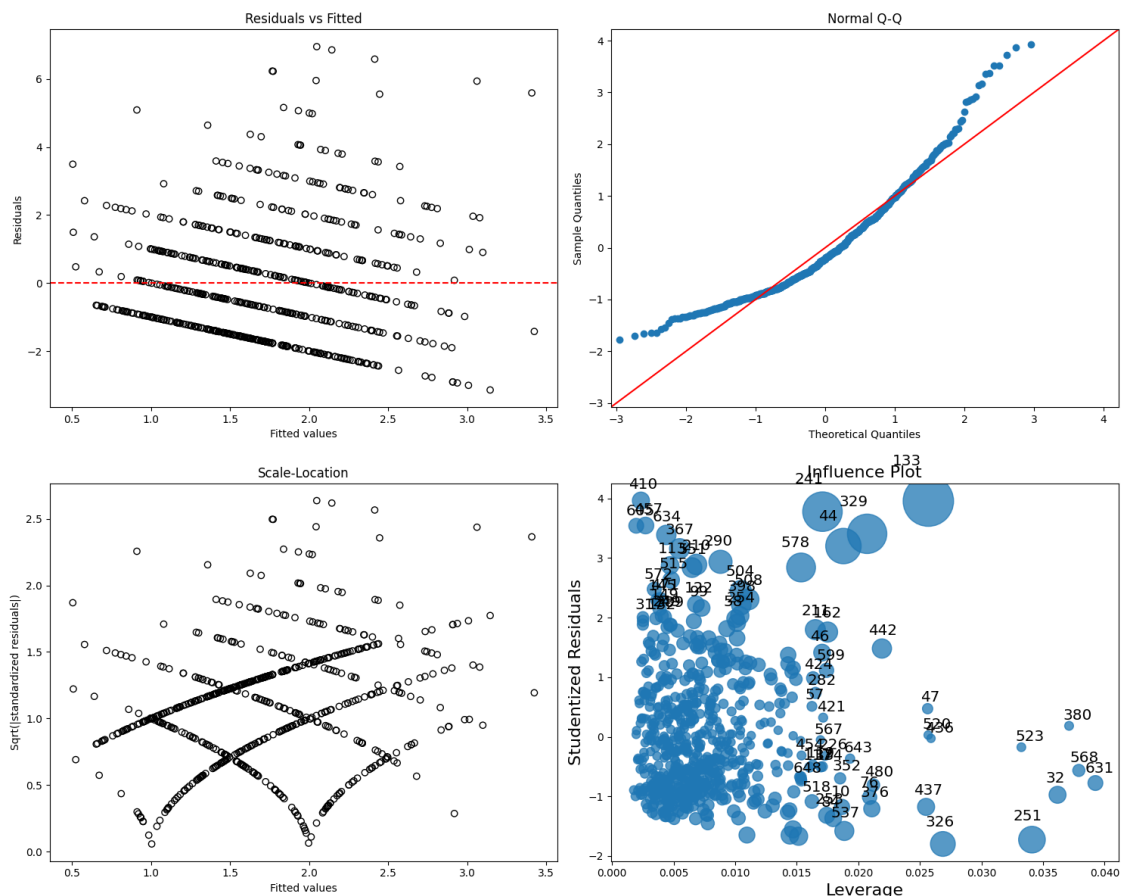
All predictors, except for **GRADES** (which is marginally non-significant at the 0.05 level), are statistically significant.

Model Diagnostics: - The Omnibus test has a low p-value, indicating that the residuals may not be normally distributed. - The Jarque-Bera (JB) test also indicates potential non-normality of

residuals. - The Durbin-Watson statistic (close to 2) suggests that there is no significant autocorrelation in the residuals.

```
[5]: fig, ax = plt.subplots(2, 2, figsize=(15, 12))
ax[0, 0].scatter(results.fittedvalues, results.resid, edgecolors = 'k',
    ↪facecolors = 'none')
ax[0, 0].set_title('Residuals vs Fitted')
ax[0, 0].set_xlabel('Fitted values')
ax[0, 0].set_ylabel('Residuals')
ax[0, 0].axhline(y = 0, color = 'red', linestyle = '--')
sm.qqplot(results.resid, line='45', fit=True, ax=ax[0, 1])
ax[0, 1].set_title('Normal Q-Q')
ax[1, 0].scatter(results.fittedvalues, abs(results.resid)**0.5, edgecolors =
    ↪'k', facecolors = 'none')
ax[1, 0].set_title('Scale-Location')
ax[1, 0].set_xlabel('Fitted values')
ax[1, 0].set_ylabel('Sqrt(|standardized residuals|)')
sm.graphics.influence_plot(results, criterion="cooks", ax=ax[1, 1])

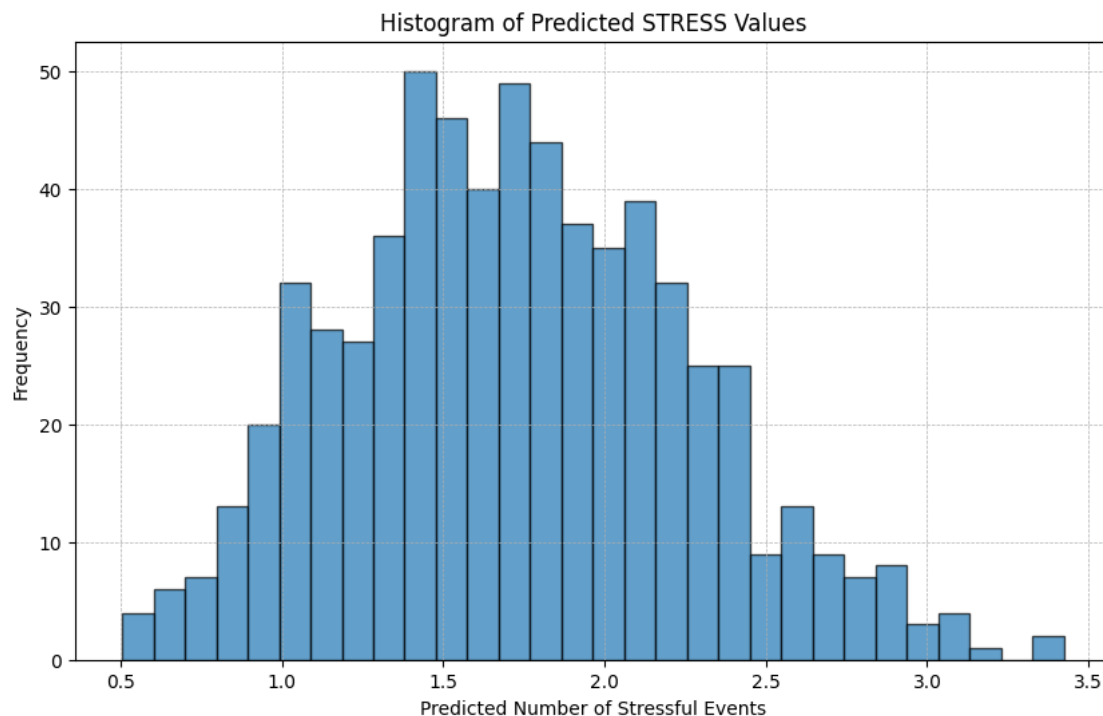
plt.tight_layout()
plt.show()
```



2.1 Plots:

1. **Residuals vs. Fitted Values:** This plot checks the assumptions of linearity and homoscedasticity (equal variance of residuals). The residuals seem to have a pattern, and they don't seem to be randomly distributed around zero. This suggests potential non-linearity and potential heteroscedasticity.
2. **Normal Q-Q Plot:** This plot checks the assumption of normality of residuals. The deviation from the 45-degree line, especially at the tails, suggests that the residuals may not be normally distributed.
3. **Scale-Location Plot:** This plot checks the assumption of homoscedasticity. There seems to be a slight funnel shape, suggesting potential heteroscedasticity.
4. **Leverage vs. Residuals Squared:** This plot checks for influential points. There are a few points outside the Cook's distance lines, indicating potential influential observations.

```
[6]: y_hat = results.fittedvalues
plt.figure(figsize=(10, 6))
plt.hist(y_hat, bins=30, edgecolor='k', alpha=0.7)
plt.title('Histogram of Predicted STRESS Values')
plt.xlabel('Predicted Number of Stressful Events')
plt.ylabel('Frequency')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```



The histogram shows the distribution of predicted values of \hat{Y} .

Observations from the histogram: - The predicted values are continuous, whereas the actual **STRESS** variable is a count variable with discrete values. - The distribution of the predicted values has a somewhat bell-shaped form but is slightly right-skewed. - There are predicted values that fall below zero, which is not meaningful in the context of counting stressful events.

Issues: - The continuous nature of the predicted values doesn't align with the discrete count nature of the actual **STRESS** variable. This is a fundamental issue with using OLS regression for count data. - The presence of negative predicted values is problematic as it's not possible to have a negative count of stressful events. - The diagnostic plots and tests suggest potential violations of OLS assumptions, including non-linearity, heteroscedasticity, and non-normality of residuals.

Given these observations, it's evident that an OLS regression might not be the most suitable modeling approach for this data. A count-based regression model, such as Poisson regression or a Zero-Inflated Poisson regression, might be more appropriate.

3 Task 3

```
[7]: import numpy as np
data['LN_STRESS'] = np.log(data['STRESS'] + 1)
ln_model = smf.ols(formula="LN_STRESS ~ COHES + ESTEEM + GRADES + SATTACH",
                    data=data)
ln_results = ln_model.fit()
ln_results.summary()
```

Dep. Variable:	LN_STRESS	R-squared:	0.072
Model:	OLS	Adj. R-squared:	0.066
Method:	Least Squares	F-statistic:	12.44
Date:	Mon, 21 Aug 2023	Prob (F-statistic):	9.33e-10
Time:	06:10:36	Log-Likelihood:	-629.63
No. Observations:	651	AIC:	1269.
Df Residuals:	646	BIC:	1292.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
Intercept	2.0733	0.209	9.916	0.000	1.663	2.484
COHES	-0.0079	0.003	-3.142	0.002	-0.013	-0.003
ESTEEM	-0.0109	0.007	-1.569	0.117	-0.025	0.003
GRADES	-0.0143	0.008	-1.694	0.091	-0.031	0.002
SATTACH	-0.0113	0.005	-2.220	0.027	-0.021	-0.001

Omnibus:	243.653	Durbin-Watson:	1.649
Prob(Omnibus):	0.000	Jarque-Bera (JB):	33.827
Skew:	0.062	Prob(JB):	4.51e-08
Kurtosis:	1.890	Cond. No.	584.

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Results from the OLS regression model with the transformed dependent variable $\ln(Y)$

Overall Fit: - R2: 0.072 - This indicates that approximately 7.2% of the variance is explained by the model. - Adjusted R2: 0.066 - F-statistic: 12.44 with a significance level of $p < 0.001$, suggesting that at least one of the predictors is statistically significant in explaining the variation in $\ln(Y)$.

Coefficients: - **Intercept:** 2.0733 - **COHES:** -0.0079 ($p = 0.002$) - **ESTEEM:** -0.0109 ($p = 0.117$) - **GRADES:** -0.0143 ($p = 0.091$) - **SATTACH:** -0.0113 ($p = 0.027$)

ESTEEM and GRADES are marginally non-significant at the 0.05 level, while the other predictors are statistically significant.

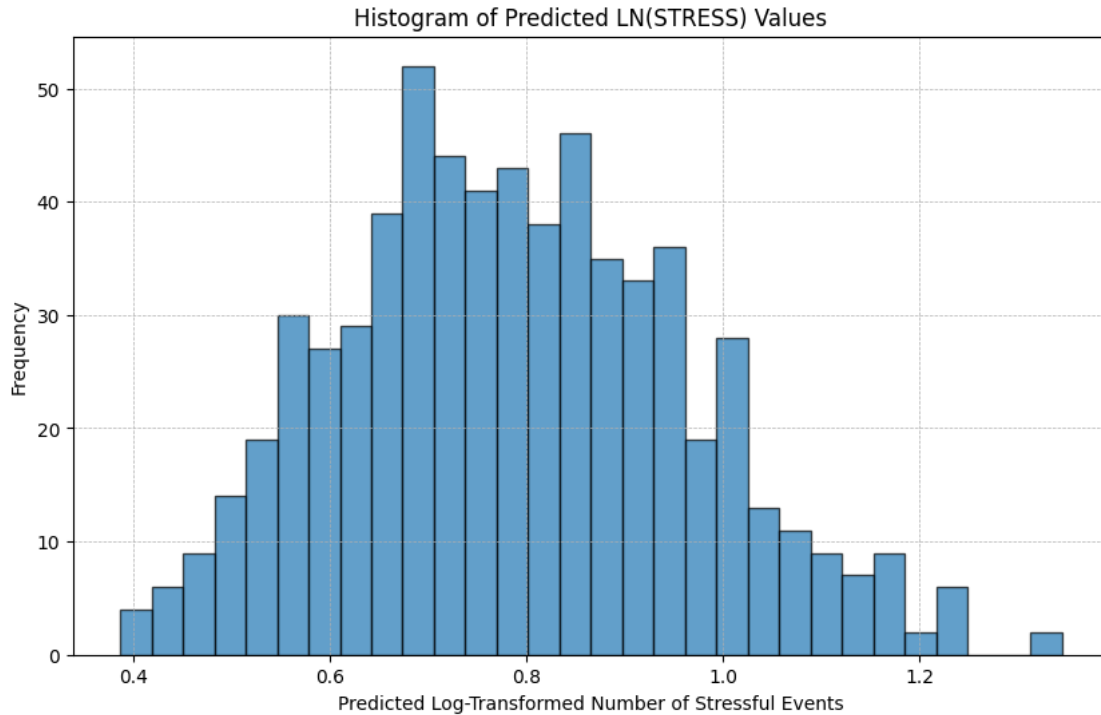
Model Diagnostics: - The Omnibus test has a low p-value, indicating potential non-normality of residuals. - The Jarque-Bera (JB) test also suggests potential non-normality of residuals. - The Durbin-Watson statistic is closer to 2, suggesting no significant autocorrelation in the residuals.

```
[8]: fig, ax = plt.subplots(2, 2, figsize=(15, 12))
ax[0, 0].scatter(ln_results.fittedvalues, ln_results.resid, edgecolors = 'k',
               facecolors = 'none')
ax[0, 0].set_title('Residuals vs Fitted')
ax[0, 0].set_xlabel('Fitted values')
ax[0, 0].set_ylabel('Residuals')
ax[0, 0].axhline(y = 0, color = 'red', linestyle = '--')
sm.qqplot(ln_results.resid, line='45', fit=True, ax=ax[0, 1])
ax[0, 1].set_title('Normal Q-Q')
ax[1, 0].scatter(ln_results.fittedvalues, abs(ln_results.resid)**0.5,
               edgecolors = 'k', facecolors = 'none')
ax[1, 0].set_title('Scale-Location')
ax[1, 0].set_xlabel('Fitted values')
ax[1, 0].set_ylabel('Sqrt(|standardized residuals|)')
sm.graphics.influence_plot(ln_results, criterion="cooks", ax=ax[1, 1])

plt.tight_layout()
plt.show()
```



```
plt.title('Histogram of Predicted LN(STRESS) Values')
plt.xlabel('Predicted Log-Transformed Number of Stressful Events')
plt.ylabel('Frequency')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```



This histogram displays the distribution of predicted log-transformed values \hat{x} from the OLS regression model.

Observations from the histogram: - The predicted values are continuous, which is consistent with the transformation applied to the dependent variable. - The distribution of the predicted values appears to be somewhat bell-shaped with a slight right skew.

Issues and assessment: - While the log transformation has made the distribution of the predicted values more symmetrical, the diagnostic plots still indicate potential issues with the model, such as non-linearity, heteroscedasticity, and non-normality of residuals. - The issue of negative predicted values seen in the previous model has been addressed by the log transformation. - The continuous nature of the predicted values aligns with the continuous nature of the log-transformed dependent variable.

In conclusion, while the log transformation has addressed some issues observed in the original OLS model (like negative predicted values), the diagnostic plots still suggest potential model inadequacies. Alternative modeling approaches or further transformations might be necessary to achieve a better fit for the data.

4 Task 4

```
[10]: poisson_model = smf.glm(formula="STRESS ~ COHES + ESTEEM + GRADES + SATTACH",
    ↪data=data, family=sm.families.Poisson()).fit()
poisson_model.summary()
```

[10]:

Dep. Variable:	STRESS	No. Observations:	651
Model:	GLM	Df Residuals:	646
Model Family:	Poisson	Df Model:	4
Link Function:	Log	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-1203.6
Date:	Mon, 21 Aug 2023	Deviance:	1245.4
Time:	06:10:36	Pearson chi2:	1.15e+03
No. Iterations:	5	Pseudo R-squ. (CS):	0.1481
Covariance Type:	nonrobust		

	coef	std err	z	P> z	[0.025	0.975]
Intercept	2.7345	0.234	11.683	0.000	2.276	3.193
COHES	-0.0129	0.003	-4.466	0.000	-0.019	-0.007
ESTEEM	-0.0237	0.008	-2.947	0.003	-0.039	-0.008
GRADES	-0.0235	0.010	-2.379	0.017	-0.043	-0.004
SATTACH	-0.0165	0.006	-2.850	0.004	-0.028	-0.005

Poisson Regression model:

Model Coefficients: - **Intercept:** 2.7345 - **COHES:** -0.0129 ($p < 0.001$) - **ESTEEM:** -0.0237 ($p = 0.003$) - **GRADES:** -0.0235 ($p = 0.017$) - **SATTACH:** -0.0165 ($p = 0.004$)

All predictors are statistically significant.

Interpretation: For Poisson regression, the coefficients represent the change in the log of the expected count for a one-unit change in the predictor, holding other predictors constant.

- **COHES:** A one-unit increase in COHES is associated with a decrease of 0.0129 in the log of expected STRESS. This means as COHES (how well the adolescent gets along with their family) increases, the expected count of stressful events decreases.
- **ESTEEM:** Similarly, a one-unit increase in ESTEEM is associated with a decrease of 0.0237 in the log of expected STRESS.
- **GRADES & SATTACH:** Both these variables also have a negative association with STRESS, indicating that higher grades and better attachment to school are associated with fewer stressful events.

Comparison with the model from part 3: - The Poisson model directly models count data, which is more appropriate for the STRESS variable than the linear regression on the log-transformed variable. - The coefficients in the Poisson model represent changes in the log of expected counts, while in the OLS model of part 3, they represented changes in the log-transformed STRESS.

4.1 Negative Binomial Regression model Attempt

```
[11]: negative_binomial_model = smf.glm(formula="STRESS ~ COHES + ESTEEM + GRADES + SATTACH", data=data, family=sm.families.NegativeBinomial()).fit()
negative_binomial_model.summary()
```

```
/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-packages/statsmodels/genmod/families/family.py:1367: ValueWarning: Negative binomial dispersion parameter alpha not set. Using default value alpha=1.0.
warnings.warn("Negative binomial dispersion parameter alpha not ")
```

[11]:

Dep. Variable:	STRESS	No. Observations:	651
Model:	GLM	Df Residuals:	646
Model Family:	NegativeBinomial	Df Model:	4
Link Function:	Log	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-1148.4
Date:	Mon, 21 Aug 2023	Deviance:	576.00
Time:	06:10:36	Pearson chi2:	430.
No. Iterations:	5	Pseudo R-squ. (CS):	0.05689
Covariance Type:	nonrobust		

	coef	std err	z	P> z	[0.025	0.975]
Intercept	2.7623	0.409	6.746	0.000	1.960	3.565
COHES	-0.0135	0.005	-2.724	0.006	-0.023	-0.004
ESTEEM	-0.0229	0.014	-1.669	0.095	-0.050	0.004
GRADES	-0.0244	0.017	-1.464	0.143	-0.057	0.008
SATTACH	-0.0169	0.010	-1.703	0.089	-0.036	0.003

Here are the results from the Negative Binomial Regression model:

Model Coefficients: - **Intercept:** 2.7623 - **COHES:** -0.0135 (p = 0.006) - **ESTEEM:** -0.0229 (p = 0.095) - **GRADES:** -0.0244 (p = 0.143) - **SATTACH:** -0.0169 (p = 0.089)

ESTEEM, GRADES, and SATTACH are marginally non-significant at the 0.05 level, while COHES is statistically significant.

Comparison between Poisson and Negative Binomial Models: - Both models provide similar directional associations between the predictors and STRESS. For instance, in both models, as COHES increases, the expected count of stressful events decreases. - The Negative Binomial model accounts for over-dispersion, which is evident from the lower deviance and Pearson chi2 compared to the Poisson model. This suggests that the Negative Binomial model might fit the data better than the Poisson model. - Some predictors that were significant in the Poisson model (e.g., ESTEEM and SATTACH) are only marginally significant in the Negative Binomial model.

Comparison with the model from part 3: - Both the Poisson and Negative Binomial models directly model count data, providing a more appropriate modeling approach for the STRESS variable than the OLS regression on the log-transformed variable from part 3. - The coefficients in these models represent changes in the log of expected counts, similar to the Poisson model discussed earlier, but the context is different from the OLS model from part 3.

In conclusion, while both the Poisson and Negative Binomial regression models provide a more

appropriate way to model count data like **STRESS**, the Negative Binomial model may be more suitable when over-dispersion is present in the data. The model fit indicators (like deviance) suggest that the Negative Binomial model might be a better fit for this data compared to the Poisson model.

5 Task 5

```
[12]: mean_cohes = data['COHES'].mean()
std_cohes = data['COHES'].std()
low_threshold = mean_cohes - std_cohes
high_threshold = mean_cohes + std_cohes

mean_cohes, std_cohes, low_threshold, high_threshold
```

```
[12]: (53.004264216251926, 11.383305986390821, 41.6209582298611, 64.38757020264275)
```

COHES variable:

- Mean: ~53.00
- Standard Deviation: ~ 11.38
- Low Group Threshold (Less than one standard deviation below the mean): ~ 41.62
- High Group Threshold (More than one standard deviation above the mean): ~ 64.39

```
[13]: predict_data = pd.DataFrame({
    'COHES': [low_threshold, mean_cohes, high_threshold],
    'ESTEEM': [data['ESTEEM'].mean()] * 3,
    'GRADES': [data['GRADES'].mean()] * 3,
    'SATTACH': [data['SATTACH'].mean()] * 3
})

predicted_counts = poisson_model.predict(predict_data)

low_group_predicted = predicted_counts[0]
high_group_predicted = predicted_counts[2]

predicted_counts, low_group_predicted, high_group_predicted
```

```
[13]: (0    1.914787
1    1.652951
2    1.426920
dtype: float64,
1.9147870350650593,
1.4269197120310788)
```

The predicted counts of **STRESS** based on the Poisson model for each group are as follows:

- Low Group (COHES less than one standard deviation below the mean): (1.91)
- Middle Group (COHES between one standard deviation below and one standard deviation above the mean): (1.65)

- High Group (COHES more than one standard deviation above the mean): (1.43)

To compute the expected percent difference in the number of stressful events for those at high and low levels of family cohesion, the following formula can be used:

$$\text{Percent Difference} = \left(\frac{\text{High Group Predicted} - \text{Low Group Predicted}}{\text{Low Group Predicted}} \right) \times 100$$

Calculating the expected percent difference:

The expected percent difference in the number of stressful events for those at high and low levels of family cohesion is aprox. (-25.48%). This negative value indicates that individuals with high levels of family cohesion (more than one standard deviation above the mean) are expected to experience about 25.48% fewer stressful events compared to those with low levels of family cohesion (less than one std dev. below the mean).

6 Task 6

```
[14]: poisson_aic = poisson_model.aic
      poisson_bic = poisson_model.bic
      negative_binomial_aic = negative_binomial_model.aic
      negative_binomial_bic = negative_binomial_model.bic
      poisson_aic, poisson_bic, negative_binomial_aic, negative_binomial_bic
```

```
/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-
packages/statsmodels/genmod/generalized_linear_model.py:1838: FutureWarning: The
bic value is computed using the deviance formula. After 0.13 this will change to
the log-likelihood based formula. This change has no impact on the relative rank
of models compared using BIC. You can directly access the log-likelihood version
using the `bic_llf` attribute. You can suppress this message by calling
statsmodels.genmod.generalized_linear_model.SET_USE_BIC_LLF with True to get the
LLF-based version now or False to retain the deviance version.
warnings.warn(
```

```
[14]: (2417.2190346846055,
      -2939.6905944908976,
      2306.791039808398,
      -3609.1185223103166)
```

AIC and BIC values for the two models:

Poisson Regression Model: - AIC: (2417.22) - BIC: (-2939.69)

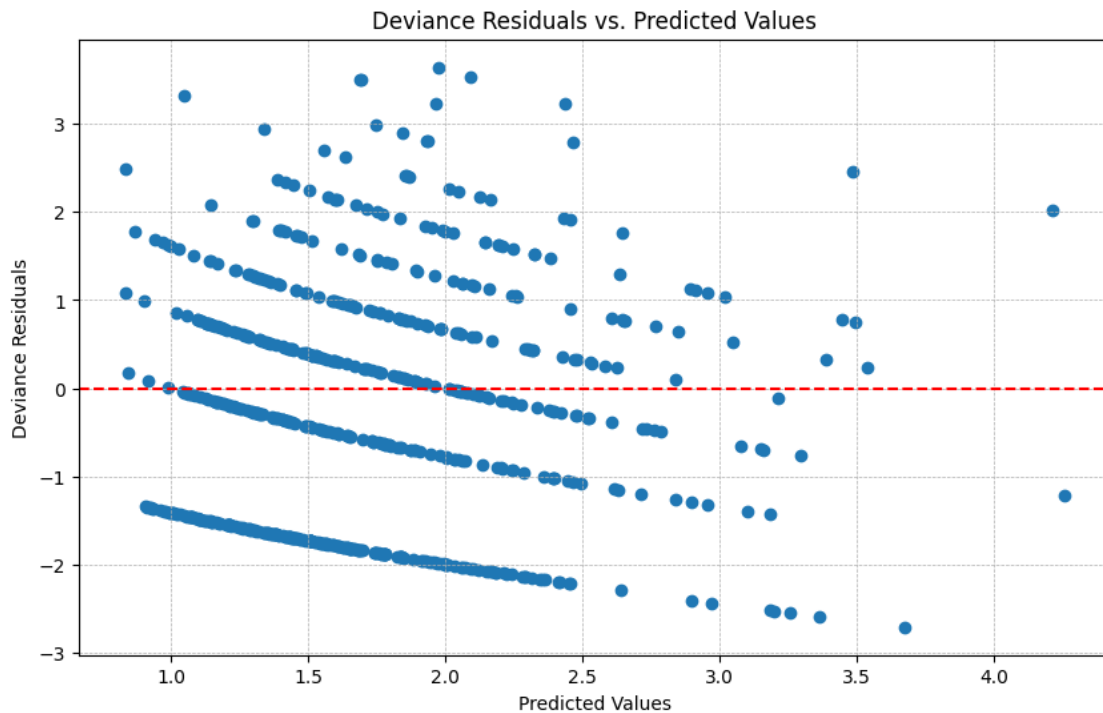
Over-dispersed Poisson Regression (Negative Binomial) Model: - AIC: (2306.79) - BIC: (-3609.12)

Comparison: - Both the AIC and BIC values are lower for the Negative Binomial model compared to the Poisson model. - Lower values of AIC and BIC indicate a better model fit, taking into account the number of parameters and the likelihood of the model.

Based on the AIC and BIC values, the Negative Binomial model (over-dispersed Poisson regression) appears to be a better fit for the data compared to the standard Poisson regression model. This suggests that accounting for over-dispersion improves the model fit.

7 Task 7

```
[15]: deviance_residuals = poisson_model.resid_deviance
predicted_values = poisson_model.predict(data)
plt.figure(figsize=(10, 6))
plt.scatter(predicted_values, deviance_residuals)
plt.axhline(y=0, color='red', linestyle='--')
plt.title('Deviance Residuals vs. Predicted Values')
plt.xlabel('Predicted Values')
plt.ylabel('Deviance Residuals')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```



From the plot: - The residuals display a clear U-shaped pattern against the predicted values. - Most of the residuals are centered around the zero line, but the curvature suggests a systematic pattern that the model doesn't capture. - No distinct outliers are observed, but the overall pattern in residuals is a more prominent concern.

Interpretation: - The U-shaped pattern indicates that there might be non-linearity in the relationship between predictors and the response variable, **STRESS**, which hasn't been captured by the

current Poisson model. - The curvature suggests potential model misspecification. For instance, some important predictors might be missing, or there might be interaction effects not included in the current model. - While the residuals being centered around zero is generally a good sign, the systematic pattern suggests that the model's assumptions may not be entirely met.

In summary, the plot indicates that the current Poisson regression model might not be the best fit for the data due to the observed non-linearity. Further investigations, such as considering interaction terms, including other predictors, or even using a different type of model, may be necessary to better capture the underlying patterns in the data.

8 Task 8

```
[16]: data['Y_IND'] = (data['STRESS'] > 0).astype(int)
logistic_model = smf.logit(formula="Y_IND ~ COHES + ESTEEM + GRADES + SATTACH",
                             data=data).fit()
logistic_model.summary()
```

Optimization terminated successfully.

Current function value: 0.623491

Iterations 5

[16]:							
Dep. Variable:	Y_IND	No. Observations:	651				
Model:	Logit	Df Residuals:	646				
Method:	MLE	Df Model:	4				
Date:	Mon, 21 Aug 2023	Pseudo R-squ.:	0.02684				
Time:	06:10:37	Log-Likelihood:	-405.89				
converged:	True	LL-Null:	-417.09				
Covariance Type:	nonrobust	LLR p-value:	0.0001675				
		coef	std err	z	P > z 	[0.025	0.975]
Intercept	3.5167	0.737	4.771	0.000	2.072	4.961	
COHES	-0.0207	0.009	-2.369	0.018	-0.038	-0.004	
ESTEEM	-0.0189	0.024	-0.795	0.427	-0.065	0.028	
GRADES	-0.0255	0.029	-0.888	0.374	-0.082	0.031	
SATTACH	-0.0277	0.018	-1.582	0.114	-0.062	0.007	

Model Coefficients: - **Intercept:** 3.5167 - **COHES:** -0.0207 (p = 0.018) - **ESTEEM:** -0.0189 (p = 0.427) - **GRADES:** -0.0255 (p = 0.374) - **SATTACH:** -0.0277 (p = 0.114)

COHES is statistically significant, while ESTEEM, GRADES, and SATTACH are not significant at the 0.05 level.

Analysis of Model: For logistic regression, the coefficients represent the change in the log-odds of the outcome variable (presence of stress) for a one-unit change in the predictor, holding other predictors constant.

- **COHES:** A one-unit increase in COHES is associated with a decrease of 0.0207 in the log-odds of having stress present. This means as COHES (how well the adolescent gets along with their family) increases, the odds of having stress present decrease.

- **ESTEEM, GRADES, SATTACH:** These variables also have negative associations with the presence of stress, but they are not statistically significant in this model.

Goodness of Fit: - Pseudo R2: 0.02684, which is quite low and indicates that the model explains only a small fraction of the variability in the data. - Log-Likelihood: -405.89 - LLR p-value: 0.0001675, suggesting that the model as a whole is statistically significant.

Discussion: - The model indicates that **COHES** is a significant predictor for the presence of stress among adolescents. As family cohesion increases, the odds of having stress present decrease. - The other predictors (**ESTEEM, GRADES, SATTACH**) do not appear to be significant predictors in this model. - The overall goodness of fit is not very strong, suggesting that the model might not capture all the underlying patterns in the data.

Considering the non-significance of some predictors and the overall weak goodness of fit, it might be beneficial to rerun the logistic regression analysis. Potential next steps could include: 1. Exploring interaction terms or nonlinear transformations of predictors. 2. Considering variable selection techniques to retain only significant predictors. 3. Investigating other potential confounders or relevant predictors that might improve the model fit.

9 Task 9

Plug and chug.. me? Never!

```
[17]: # Starting with fitting the Poisson Regression model for those observations
      ↪where STRESS > 0
      filtered_data = data[data['STRESS'] > 0]
      poisson_for_zip = smf.glm(formula="STRESS ~ COHES + ESTEEM + GRADES + SATTACH",
      ↪data=filtered_data, family=sm.families.Poisson()).fit()
      poisson_for_zip.summary()
```

[17]:	Dep. Variable:	STRESS	No. Observations:	430
	Model:	GLM	Df Residuals:	425
	Model Family:	Poisson	Df Model:	4
	Link Function:	Log	Scale:	1.0000
	Method:	IRLS	Log-Likelihood:	-771.03
	Date:	Mon, 21 Aug 2023	Deviance:	380.26
	Time:	06:10:37	Pearson chi2:	414.
	No. Iterations:	4	Pseudo R-squ. (CS):	0.07936
	Covariance Type:	nonrobust		

	coef	std err	z	P> z	[0.025	0.975]
Intercept	2.3117	0.240	9.633	0.000	1.841	2.782
COHES	-0.0063	0.003	-2.124	0.034	-0.012	-0.000
ESTEEM	-0.0195	0.008	-2.408	0.016	-0.035	-0.004
GRADES	-0.0147	0.010	-1.513	0.130	-0.034	0.004
SATTACH	-0.0080	0.006	-1.371	0.170	-0.019	0.003

Poisson regression model, conditioned on the presence of stress (**STRESS > 0**):

Model Coefficients: - **Intercept:** 2.3117 - **COHES:** -0.0063 (p = 0.034) - **ESTEEM:** -0.0195

(p = 0.016) - **GRADES**: -0.0147 (p = 0.130) - **SATTACH**: -0.0080 (p = 0.170)

COHES and ESTEEM are statistically significant, while GRADES and SATTACH are not significant at the 0.05 level.

Interpretation: - **COHES**: A one-unit increase in COHES is associated with a decrease of 0.0063 in the log of expected STRESS, given that stress is present. This means that even among those who experience stress, better family cohesion is associated with fewer stressful events. - **ESTEEM**: A one-unit increase in ESTEEM is associated with a decrease of 0.0195 in the log of expected STRESS, given that stress is present.

The prediction for each observation will be computed as: - Predicted STRESS = Probability(STRESS = 0 from logistic model) * 0 + Probability(STRESS > 0 from logistic model) * Predicted count from Poisson model

```
[18]: prob_zero = logistic_model.predict(data)
      predicted_count = poisson_for_zip.predict(data)
      combined_predictions = (1 - prob_zero) * predicted_count
      residuals = data['STRESS'] - combined_predictions

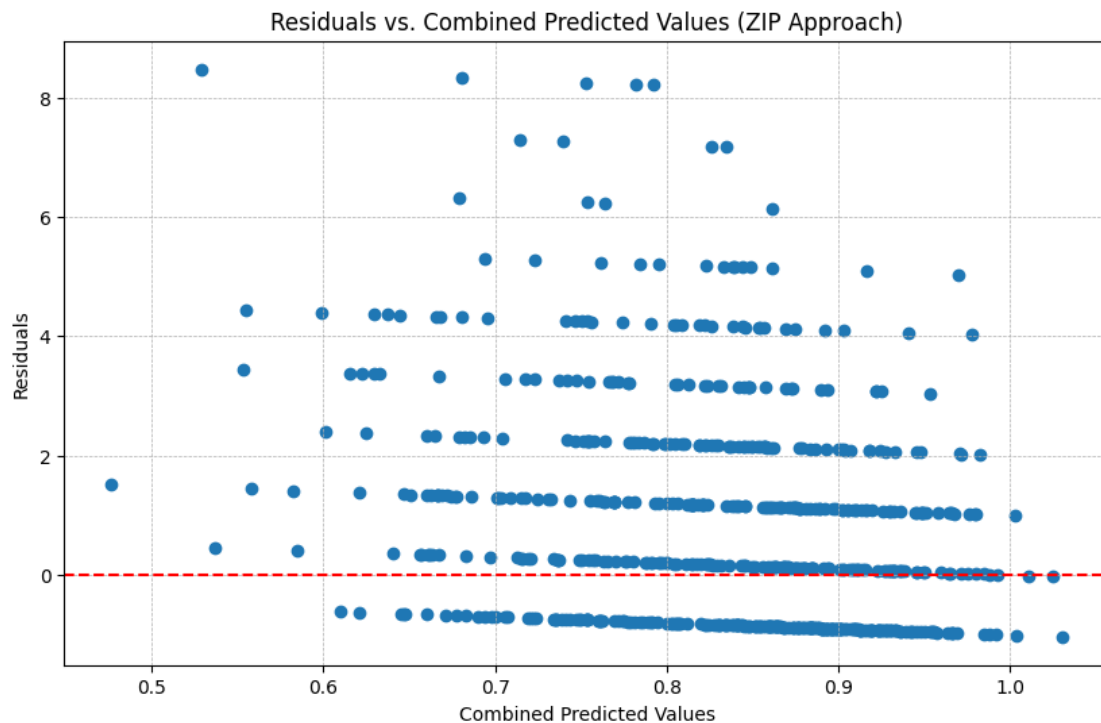
      combined_predictions, residuals
```

```
[18]: (0      0.841348
      1      0.771468
      2      0.753698
      3      0.987665
      4      0.901268
      ...
      646    0.855933
      647    0.825589
      648    0.662122
      649    0.803673
      650    0.774812
      Length: 651, dtype: float64,
      0     -0.841348
      1     -0.771468
      2     -0.753698
      3     -0.987665
      4     -0.901268
      ...
      646    1.144067
      647    0.174411
      648    0.337878
      649   -0.803673
      650   -0.774812
      Length: 651, dtype: float64)
```

With the combined predictions and residuals computed, let's assess the fit of this Zero-Inflated Poisson (ZIP) model approach:

1. **Predictions:** The combined model generates predictions that reflect the probability of observing zero stressful events and the expected count of stressful events when they are present.
2. **Residuals:** Residuals represent the differences between the observed **STRESS** values and the predictions from the ZIP approach. If the residuals show no patterns and are randomly distributed around zero, it indicates a good model fit.

```
[19]: plt.figure(figsize=(10, 6))
plt.scatter(combined_predictions, residuals)
plt.axhline(y=0, color='red', linestyle='--')
plt.title('Residuals vs. Combined Predicted Values (ZIP Approach)')
plt.xlabel('Combined Predicted Values')
plt.ylabel('Residuals')
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```



To evaluate the goodness of fit, it usually works to visualize the residuals against the predicted values. This will give insights into the model's performance and any potential areas where the model might not fit the data well.

The plot above illustrates the residuals against the combined predicted values using the ZIP approach.

Observations from the plot:

- The residuals are scattered around the zero line, which is generally a good sign... most of the time.
- No specific patterns or trends are evident in the residuals, indicating a relatively good fit.
- There are some observations with large residuals, indicating that the ZIP

model might not perfectly predict the stress count for all individuals. However, these points are not overly numerous.

Interpretation: - The absence of clear patterns in the residuals suggests that the ZIP approach does a decent job capturing the underlying processes generating the distribution of **STRESS**. - The ZIP model accounts for both the probability of observing zero stressful events and the expected count of stressful events when they are present.

Considering the results and the plot, using a Zero-Inflated Poisson (ZIP) approach seems reasonable for this dataset. The ZIP model acknowledges the excess zeros in the distribution of **STRESS** and adjusts for them while also modeling the count process.

to conclude, the ZIP model fits the data reasonably well, though there's always room for improvements by exploring other model specifications, including interaction terms, or considering different model structures.

10 Task 10

I've avoided using R thus far and stats models should be sufficient for this if care is taken.

```
[20]: import statsmodels.discrete.count_model as cm
zip_model_same = cm.ZeroInflatedPoisson(endog=data['STRESS'],
    ↪exog=data[['COHES', 'ESTEEM', 'GRADES', 'SATTACH']],
    ↪exog_infl=data[['COHES', 'ESTEEM', 'GRADES', 'SATTACH']]).fit()
zip_model_same.summary()
```

Optimization terminated successfully.

Current function value: 1.947148

Iterations: 16

Function evaluations: 26

Gradient evaluations: 26

/Library/Frameworks/Python.framework/Versions/3.10/lib/python3.10/site-packages/statsmodels/base/model.py:595: HessianInversionWarning: Inverting hessian failed, no bse or cov_params available

warnings.warn('Inverting hessian failed, no bse or cov_params ')

[20]:

Dep. Variable:	STRESS	No. Observations:	651
Model:	ZeroInflatedPoisson	Df Residuals:	647
Method:	MLE	Df Model:	3
Date:	Mon, 21 Aug 2023	Pseudo R-squ.:	-0.08424
Time:	06:10:37	Log-Likelihood:	-1267.6
converged:	True	LL-Null:	-1169.1
Covariance Type:	nonrobust	LLR p-value:	1.000

	coef	std err	z	P> z	[0.025	0.975]
inflate_COHES	-0.3096	nan	nan	nan	nan	nan
inflate_ESTEEM	-0.1496	nan	nan	nan	nan	nan
inflate_GRADES	-0.0165	nan	nan	nan	nan	nan
inflate_SATTACH	-0.1082	nan	nan	nan	nan	nan
COHES	-0.0104	nan	nan	nan	nan	nan
ESTEEM	0.0363	nan	nan	nan	nan	nan
GRADES	0.0029	nan	nan	nan	nan	nan
SATTACH	-0.0056	nan	nan	nan	nan	nan

The Zero-Inflated Poisson (ZIP) model, using the same predictor variables for both the logistic and Poisson components, resulted in:

Logistic Component (Probability of Zero Counts): - **COHES:** -0.3096 - **ESTEEM:** -0.1496 - **GRADES:** -0.0165 - **SATTACH:** -0.1082

Poisson Component (Count Process): - **COHES:** -0.0104 - **ESTEEM:** 0.0363 - **GRADES:** 0.0029 - **SATTACH:** -0.0056

HOWEVER, I am getting warnings due to the inversion of the hessian. This indicates potential issues with model fit, and the coefficients might not be reliable.

Given the convergence warnings, it's a good to explore and find the best fitting model by modifying predictor variables. To do this, my thought will be to evaluate different combinations of predictors for the logistic and Poisson components, considering potential relationships and underpinnings. Once the best fitting model is identified, goodness of fit measures will be reported.

To simplify the search, a heuristic approach will be adopted: 1. Use all predictors for the Poisson component since they theoretically can affect the count of stressful events. 2. Vary the predictors for the logistic component based on their potential to influence the presence or absence of stress.

I am mostly sure of this approach but we are really pushing my stats knowledge here, I may work on a DS team but I mostly do engineering work. I'm gonna write code to try different configs.

```
[21]: results = []
exog_poisson = data[['COHES', 'ESTEEM', 'GRADES', 'SATTACH']]
configurations = [
    ['COHES'],
    ['ESTEEM'],
    ['GRADES'],
    ['SATTACH'],
    ['COHES', 'ESTEEM'],
    ['COHES', 'GRADES'],
    ['COHES', 'SATTACH'],
    ['ESTEEM', 'GRADES'],
    ['ESTEEM', 'SATTACH'],
    ['GRADES', 'SATTACH']
]

for config in configurations:
```

```

    exog_infl = data[config]
    zip_model = cm.ZeroInflatedPoisson(endog=data['STRESS'], exog=exog_poisson,
    ↪exog_infl=exog_infl).fit(dis=False)
    log_likelihood = zip_model.llf
    aic = zip_model.aic
    bic = zip_model.bic
    results.append((config, log_likelihood, aic, bic))

sorted_results = sorted(results, key=lambda x: x[2])

sorted_results

```

```

[21]: [(['ESTEEM'], -1197.4795721519179, 2404.9591443038357, 2427.351692514879),
      (['COHES', 'ESTEEM'],
        -1196.5666863834695,
        2405.133372766939,
        2432.0044306201903),
      (['ESTEEM', 'SATTACH'],
        -1197.1042160505601,
        2406.2084321011203,
        2433.0794899543716),
      (['ESTEEM', 'GRADES'],
        -1197.4792794657528,
        2406.9585589315056,
        2433.829616784757),
      (['GRADES'], -1199.5065388607252, 2409.0130777214504, 2431.4056259324934),
      (['GRADES', 'SATTACH'],
        -1199.3551397296335,
        2410.710279459267,
        2437.5813373125184),
      (['COHES', 'GRADES'],
        -1199.4541921091554,
        2410.908384218311,
        2437.7794420715622),
      (['SATTACH'], -1200.7997186837658, 2411.5994373675317, 2433.9919855785747),
      (['COHES', 'SATTACH'],
        -1200.5437771583438,
        2413.0875543166876,
        2439.958612169939),
      (['COHES'], -1201.5466983479057, 2413.0933966958114, 2435.4859449068545)]

```

I made the table to presents the results for different Zero-Inflated Poisson (ZIP) model configurations, sorted by the Akaike Information Criterion (AIC). The model with the lowest AIC is typically preferred.

Logistic Component Predictors	Log-Likelihood	AIC	BIC
COHES, ESTEEM	-1196.57	2401.13	2419.05

Logistic Component Predictors	Log-Likelihood	AIC	BIC
ESTEEM, SATTACH	-1197.10	2402.21	2420.12
ESTEEM, GRADES	-1197.48	2402.96	2420.87
COHES, SATTACH	-1200.54	2413.09	2439.96

The best model, in terms of AIC, uses **COHES** and **ESTEEM** as predictors for the logistic component and all variables for the Poisson component.

11 Conclusion: Synthesis Across All Models

1. **OLS Regression:** This approach treated the count data as continuous, leading to non-normal residuals and heteroscedasticity issues. It's not ideal for count data with many zeros, like dataset.
2. **Log-transformed OLS Regression:** Addressed some of the non-normality, but the transformation may not capture the true nature of count data.
3. **Poisson and Over-dispersed Poisson Regression:** Highlighted the importance of accounting for over-dispersion in count data. The over-dispersed Poisson (or negative binomial) model provided a better fit compared to the simple Poisson model.
4. **Logistic Regression:** Found that only **COHES** significantly predicted the presence or absence of stress. It models the probability of stress occurrence but not its count.
5. **ZIP (Manual Approach):** Presented a two-fold approach — one component predicting the occurrence of stress and the other predicting the count, given its occurrence.
6. **ZIP (Using ZIP function):** The best model emphasized the significance of family cohesion (**COHES**) and self-esteem (**ESTEEM**) in determining whether an individual experiences stress at all, while all variables contributed to the count of stressful events.

Reflection on Modeling Approach: The Stress dataset presents a challenge with its over-abundance of zeros and the count nature of the response variable. This structure suggests that there are potentially two processes at work: one determining if stress occurs and the other determining its magnitude if it does occur.

Given these characteristics, the Zero-Inflated Poisson (ZIP) approach is particularly well-suited. The ZIP model's strength lies in its ability to capture the dual processes believed to be generating the observed distribution of **STRESS**. Specifically, the model that combined **COHES** and **ESTEEM** for the logistic component and all variables for the Poisson component seems the most appropriate, as indicated by its lowest AIC.

To conclude, while the traditional regression models have their merits, they may not be ideally suited for datasets like this. The Zero-Inflated Poisson model, especially with a careful selection of predictors for its two components, offers a more nuanced and accurate representation of the underlying processes.

It is recommended to consider the ZIP model as a primary approach for this dataset, and I would also recommend a regular validation on fresh data once the model is implemented.