Impact of Climate Indicators on Visibility

Andrew Fennimore, Birendra Khimding, Muzhgan Rustaqui

Department of Engineering, University of San Diego

AAI-500 Probability and Statistics for Artificial Intelligence

Dallin Munger, M.S.

February 23, 2025

Summary

This study explores the relationship between climate indicators and visibility, a critical factor in aviation, transportation, and environmental safety. Using a dataset from Ocean County Airport (MJX) with 11,423 samples, key variables such as temperature, humidity, wind speed, precipitation, and atmospheric pressure were analyzed. Data preprocessing involved handling missing values, addressing multicollinearity, and transforming skewed data to improve model accuracy. Initial analysis using an Ordinary Least Squares (OLS) model showed that some climate indicators significantly impact visibility. A Generalized Linear Model (GLM) with log transformation was then applied, but challenges such as data skewness and missing values affected predictive performance. Despite these limitations, findings confirm a significant association between climate indicators and visibility, highlighting the need for more complete datasets and advanced modeling techniques in future research.

Impact of Climate Indicators on Visibility

Predicting visibility is crucial for ensuring safety across various industries, including aviation, transportation, and climate safety. According to Zhang et al. (2022), visibility, simply put, is a measure of atmospheric transparency. Measuring it is vital for safety in industries such as aviation, transportation, and environmental monitoring (Kadam et al., 2023). According to Ortega et al. (2022), poor visibility conditions are linked to approximately 31,500 traffic accidents yearly in the United States, leading to 11,500 injuries and 500 fatalities. In addition, measuring visibility can be an important indicator of air quality and pollutants like aerosols (Liang et al., 2023). This is especially vital in poorer regions where measuring instrumentation and data are unavailable (Liang et al., 2023). By examining the relationships between climate indicators such as weather and humidity, our project aims to help stakeholders implement effective safety measures proactively.

The most common method for measuring visibility is by using scattered visiometers to measure the distance that can be seen (Liang et al., 2023). Airports measure visibility through a system known as RVR, or runway visual range, which provides a consistent method to determine the distance a pilot can expect to see (U.S. Department of Transportation, 2024). Measuring equipment can be seen along runways and is used to determine if pilots can fly safely during possibly unsafe weather conditions (U.S. Department of Transportation, 2024).

This study aims to use easily accessible variables to ensure that our model can be universally applied regardless of location or wealth. The dataset used to build the model was obtained from Ocean County Airport (MJX) and comprises 11,423 samples. This dataset includes various climate indicators, including temperature, humidity, wind speed, precipitation, and atmospheric pressure. We not only aim to determine if there is an association between visibility and the climate indicators but attempt to develop a model to predict future visibility.

Null Hypothesis (H_0): Climate indicators do not have any significant impact on visibility.
Alternative Hypothesis (H_1): At least one of the climate indicators has a significant impact on visibility.

Table 1

Columns and description of MJX dataset

	Variable Name	Description
1	station	Three or four character site identifier
2	valid	Timestamp of the observation
3	tmpf	Air Temperature in Fahrenheit, typically @ 2 meters
4	dwpf	Dew Point Temperature in Fahrenheit, typically @ 2 meters
5	relh	Relative Humidity in %
6	drct	Wind Direction in degrees from *true* north
7	sknt	Wind Speed in knots
8	p01i	One hour precipitation for the period from the observation time to the time of the previous hourly precipitation reset.
9	alti	Pressure altimeter in inches
10	mslp	Sea Level Pressure in millibar
11	vsby	Visibility in miles
12	gust	Wind Gust in knots
13	skyc1	Sky Level 1 Coverage
14	skyc2	Sky Level 2 Coverage
15	skyc3	Sky Level 3 Coverage
16	skyc4	Sky Level 4 Coverage
17	skyl1	Sky Level 1 Altitude in feet
18	skyl2	Sky Level 2 Altitude in feet
19	skyl3	Sky Level 3 Altitude in feet
20	skyl4	Sky Level 4 Altitude in feet
21	wxcodes	Present Weather Codes (space separated)
22	feel	Apparent Temperature (Wind Chill or Heat Index) in Fahrenheit
23	ice_accretion_1hr	Ice Accretion over 1 Hour (inches)
24	ice_accretion_3hr	Ice Accretion over 3 Hours (inches)
25	ice_accretion_6hr	Ice Accretion over 6 Hours (inches)
26	peak_wind_gust	Peak Wind Gust (from PK WND METAR remark) (knots)
27	peak_wind_drct	Peak Wind Gust Direction (from PK WND METAR remark) (deg)
28	peak_wind_time	Peak Wind Gust Time (from PK WND METAR remark)
29	metar	Unprocessed reported observation in METAR format

Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Cleaning and Preparation

Of the total variables found in the dataset, 6 of the variables had no datapoints, including skyc4, skyl4, 3 stages of ice_accretion and snowdepth. These variables can be immediately removed. Other variables that were of no obvious use included "valid" and "metar" which were dates with only unique values and "station" which had only 1 unique value.

There were several areas of difficulty that were encountered when analyzing the data. This includes dealing with missing variables, dealing with a heavy left skew, and addressing the large quantity of variables used for analysis. Not properly dealing with one of these issues could lead to erroneous conclusions. To address these issues, the team needed to use several methods and statistical tools even before analysis.

When dealing with missing values, it is imperative that they are addressed correctly. Not doing so can lead to biased estimates or reduce statistical power, leading to invalid conclusions from the data (Kang, 2013). The data set uses "M" for all missing values and treated as NA. This model assumes MCAR for all variables with a threshold above 15%. MCAR, or missing completely at random, is the ideal assumption and used when missing values are missing only due to random issues (Kang, 2013). This could include issues like instrument failure for a day or power being lost (Kang, 2013). The assumption of MCAR permitted us to omit cases that were null in our data. This is the most common approach when data is missing but may lead to false power of variables, even if the MCAR assumption is true (Kang, 2013). However, the sheer number of values missing makes it impossible to use this method of imputation, and so instead we imposed the 15% threshold. The data set uses “M” for all missing values and treated as NA within the dataset.

We noticed a negative skew with the dependent variable “vsby”. There are no outliers that would impact the distribution of the variables, so thoughts of using a generalized linear model were entertained even before any analysis. Scatterplots would later confirm the need for further data transformation.

Our dataset only consisted of 11,423 samples; however, this is still far larger than others discussed and utilized in class. According to Fan et al. (2014), larger datasets can be useful for determining patterns and providing better oversight than smaller ones. However, some challenges with bigger datasets include scalability, spurious correlations, ad measurement errors (Fan et al., 2014). It also proved to be somewhat taxing on our computational efforts and required careful adjustments to code.

Exploratory Data Analysis

Table 2 provides an overview of the descriptive statistics for the dataset, as well as other pertinent information. Results with greater than 15% null value were removed from analysis, as well as the other variables previously mentioned.

Looking at the dependent variable “vsby” we see a median of 10 miles. This is also equivalent to the highest value. This further demonstrates the need to transform some of our variables when we start to begin analysis.

Table 2

Descriptive statistics and pertinent variable information of variables in MJX data set

		Data Type	Count	Mean	Median	Std	Mode	Null Percentage	Unique Values	Most Frequent Count
1	tmpf	Quantitative	11418	54.56	54.0	18.48	[59.0]	0.04%		
2	dwpf	Quantitative	11416	45.75	48.0	18.53	[59.0]	0.06%		
3	relh	Quantitative	11416	75.4	80.83	20.79	[100.0]	0.06%		
4	drct	Quantitative	11141	162.4	180.0	122.03	[0.0]	2.47%		
5	sknt	Quantitative	11415	6.03	6.0	4.7	[0.0]	0.07%		
6	alti	Quantitative	11312	30.02	30.01	0.23	[30.03]	0.97%		
7	mslp	Quantitative	8928	1017.29	1017.0	7.89	[1017.3]	21.84%		
8	vsby	Quantitative	11422	8.91	10.0	2.59	[10.0]	0.01%		
9	gust	Quantitative	1855	20.27	19.0	4.62	[18.0]	83.76%		
10	skyc4	Quantitative	0					100.0%		
11	skyl1	Quantitative	6484	3022.92	1900.0	3015.17	[400.0]	43.24%		
12	skyl2	Quantitative	2177	4528.2	3700.0	2911.25	[6000.0]	80.94%		
13	skyl3	Quantitative	805	5730.93	5000.0	2714.98	[6000.0]	92.95%		
14	skyl4	Quantitative	0					100.0%		
15	ice_accretion_1hr	Quantitative	0					100.0%		
16	ice_accretion_3hr	Quantitative	0					100.0%		
17	ice_accretion_6hr	Quantitative	0					100.0%		
18	peak_wind_gust	Quantitative	452	29.56	29.0	3.47	[26.0]	96.04%		
19	peak_wind_drct	Quantitative	452	267.57	290.0	54.72	[290.0]	96.04%		
20	feel	Quantitative	11414	53.09	54.0	21.33	[59.0]	0.08%		
21	snowdepth	Quantitative	0					100.0%		
22	station	Qualitative	11423					0.0%	1	
23	valid	Qualitative	11423				)6 12:55', '2024-12-29 04:55']	0.0%	11420	2
24	p01i	Qualitative	11420				['0.00']	0.03%	47	9365
25	skyc1	Qualitative	11393				['CLR']	0.26%	6	4909
26	skyc2	Qualitative	2177				['OVC']	80.94%	3	964
27	skyc3	Qualitative	805				['OVC']	92.95%	3	656
28	wxcodes	Qualitative	2351				['-RA', 'BR']	79.42%	29	576
29	peak_wind_time	Qualitative	452				20 15:20', '2024-07-10 20:22']	96.04%	421	4
30	metar	Qualitative	11423					0.0%	11423	

Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Figure 1

Histogram of the dependent variable visibility

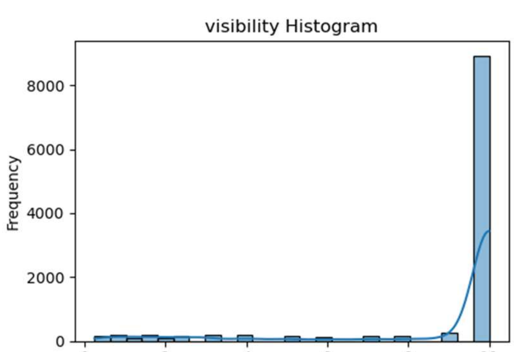
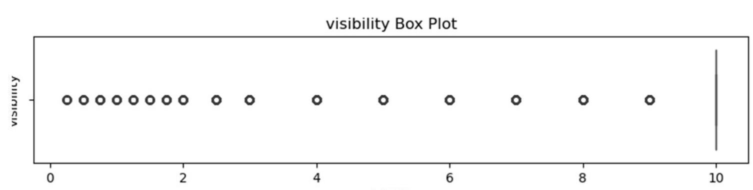


Figure 2

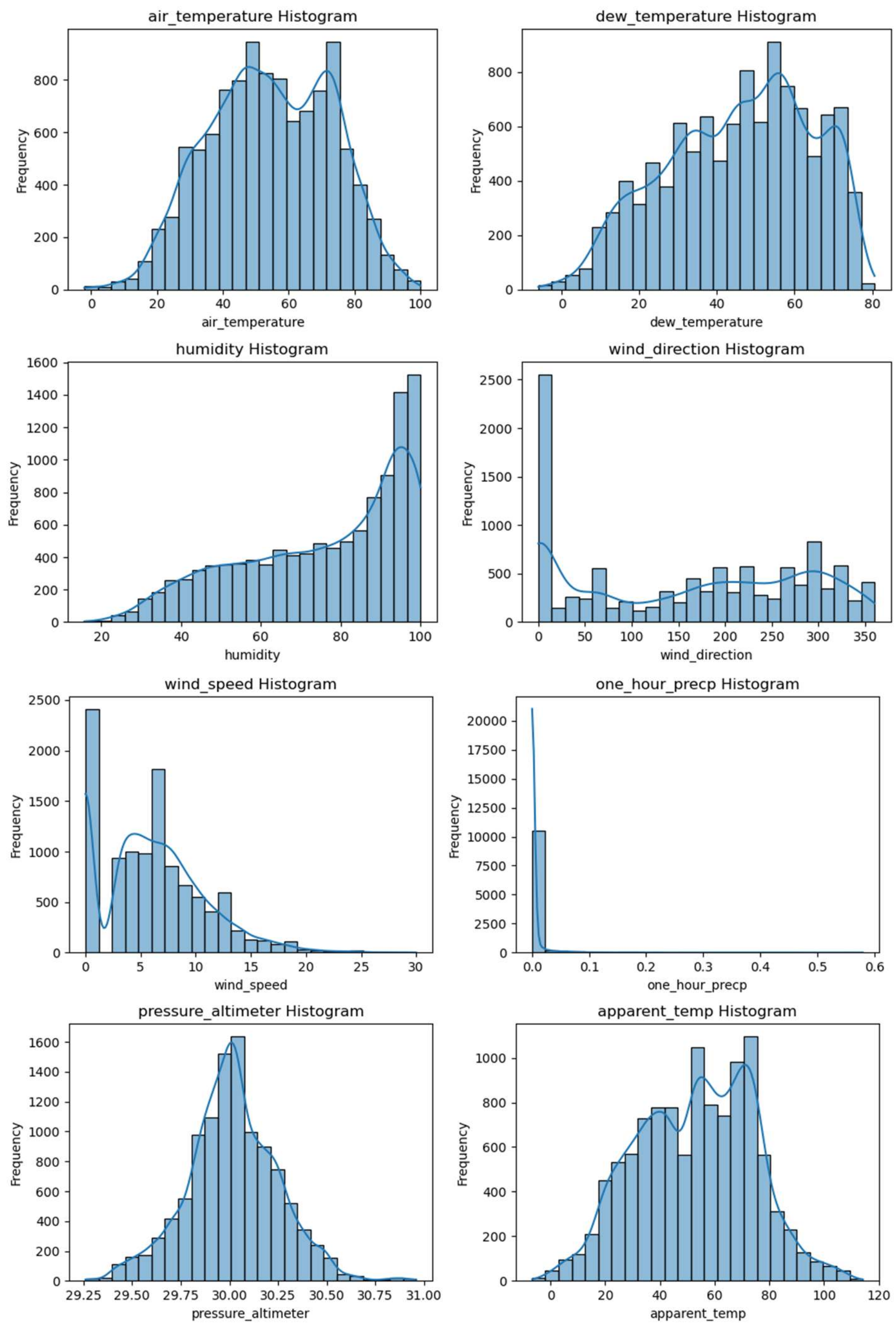
Box plot of the dependent variable visibility



Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Figure 2

Distribution plot of pertinent variables



Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Visualization

We can see from graph 3 and graph 4 verification of the negative skew. This will likely make it difficult to create a good fit for a model. According to Dugan and Greyserman (2019), a negative skew may initially give a false impression of an improved model, while a positive skew has the opposite effect. Data transformation will be necessary for a best fit model.

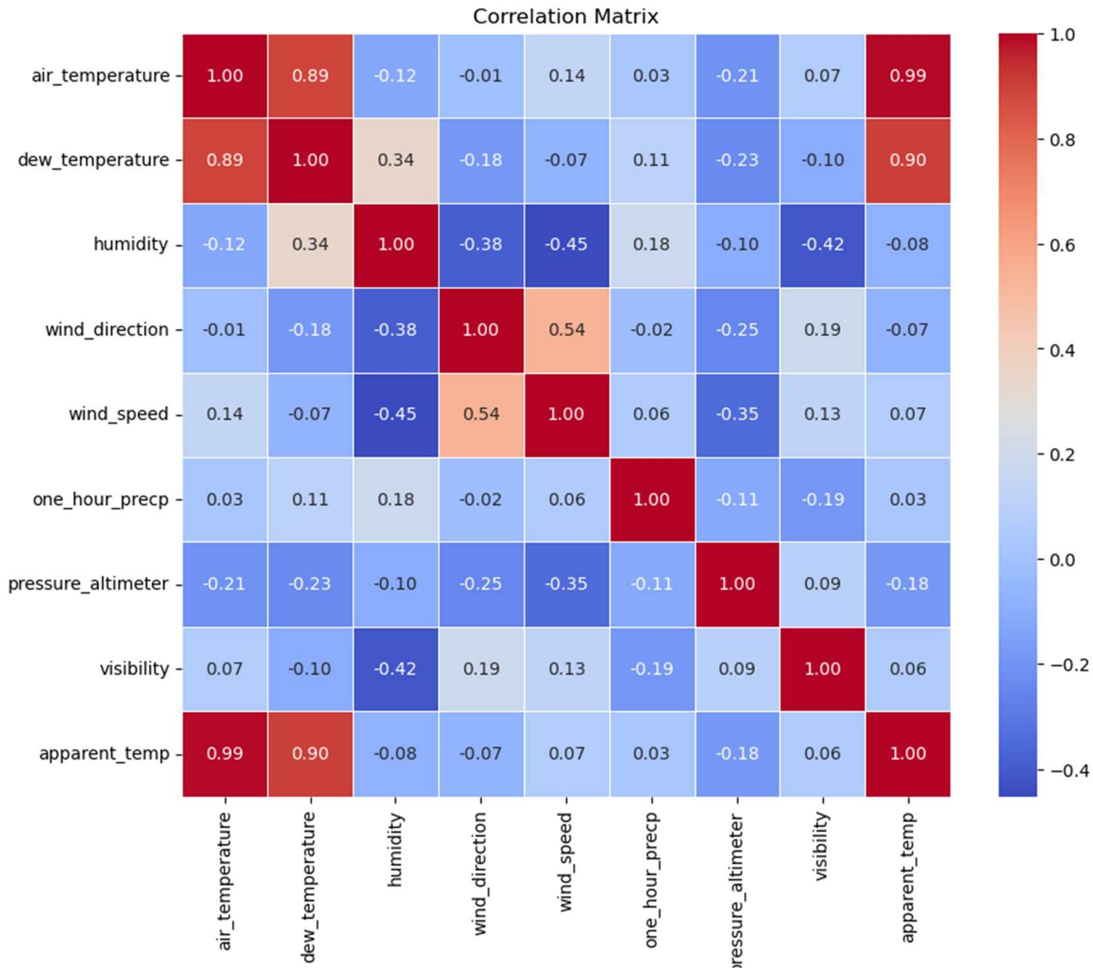
The histograms for the predictor variables show a wide range of distributions. While pressure, apparent temperature, air temperature and wind direction show a fairly bell-shaped normal distribution, wind speed shows a strong positive skew. Humidity demonstrates the opposite distribution and shows a positive skew, and one hour precipitation shows values all clustered at 0. Fortunately, no assumptions need to be made of the distributions of explanatory variables when dealing with linear models and least squares (Agresti & Kateri, 2021).

In the correlation matrix for Table 5, it's evident that certain variables, such as dew temperature, air temperature, and apparent temperature, exhibit multicollinearity. Multicollinearity refers to when explanatory variables showing some overlap and demonstrating redundant values (Agresti & Greyserman, 2019). Its effects can be seen when two predictors that are highly correlated are assessed at the same time in a regression model and ignoring this can lead to misleading interpretations of results (Vatcheva & Lee, 2016).

Depending on what the research is looking for would depend on how to deal with the issue. According to Vatcheva and Lee (2016), Multicollinearity will not impact the fit of the model. If however, we are looking to investigate associations, multicollinearity can obscure effects of an independent variable on the outcome variable (Vatcheva & Lee, 2016). Given their high correlations, it would be prudent to eliminate two of these variables and retain only one. This approach will provide a more accurate assessment of statistical power.

Figure 3

Correlation matrix of data set MJX



Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Creating and Analyzing Models

Ordinary Least Squares Model

$$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x$$

Ordinary Least Square Models attempt to find a linear relationship between a dependent variable (vsby) and the independent variables (Agresti & Katari, 2021). Several assumptions are necessary for an ordinary least squares model. According to Williams et al (2013), these include being unbiased, consistent and efficient. Being unbiased according to Williams et al. (2013), is the mean of a sample is the same as the true parameter of the mean of the population. It is basically stating that the samples that are obtained must represent the population as a whole. Consistent means that as a sample size increases, so does its accuracy (Williams et al., 2013). Efficiency then is accuracy of the samples. Normality of the residuals, or the difference between the observed and predicted values, is also a necessity in ordinary least squares modeling (William et al., 2013).

Our model does not fit these assumptions well. We do not see normality of residuals, making it difficult to accurately fit a regression model. However, we can use it to compare it to the generalized linear model and determine what transformations are necessary. These results, however, should be used with caution.

After removing variables that had less than a 15% threshold, values that showed multicollinearity, and ones that could not have an impact on the outcome, the variables left include tmpf (air temperature), relh (relative humidity), sknt (wind speed in knots), alti (pressure altimeter) and skyc1 (sky level 1 coverage). Skyc1 is the only categorical variable, and dummy variables can be used to incorporate them into a linear model. Using these variables in an ordinary least squares model were sknt is the only one that is not significant. We can tell this by looking at our P values, in which if it is greater than .05, we can be sure that it doesn't significantly impact the model.

Results indicate that about 25% of the variability of visibility can be explained with the model. We can tell that by looking at the R squared which is .252. The F statistic indicates to us that the model is significant at 421 where we see a P(F) value at 0.00. This means that there is a low probability that the F statistic occurred by chance. Since we know that we know and can prove significance, we can reject our null hypothesis claiming that there are no association between climate indicators and visibility.

Figure 4

Ordinary Least Squares Model for vsby = skyc1 + tmpf +vrelh + sknt + altic

OLS Regression Results

Dep. Variable:	vsby	R-squared:	0.252
Model:	OLS	Adj. R-squared:	0.251
Method:	Least Squares	F-statistic:	421.0
Date:	Sun, 23 Feb 2025	Prob (F-statistic):	0.00
Time:	10:37:15	Log-Likelihood:	-25046.
No. Observations:	11267	AIC:	5.011e+04
Df Residuals:	11257	BIC:	5.019e+04
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-1.1464	3.153	-0.364	0.716	-7.326	5.034
C(skyc1)[T.CLR]	0.4979	0.068	7.278	0.000	0.364	0.632
C(skyc1)[T.FEW]	0.3827	0.085	4.496	0.000	0.216	0.550
C(skyc1)[T.OVC]	-1.3470	0.072	-18.604	0.000	-1.489	-1.205
C(skyc1)[T.SCT]	0.2867	0.088	3.273	0.001	0.115	0.458
C(skyc1)[T.VV]	-7.2896	0.648	-11.241	0.000	-8.561	-6.018
tmpf	0.0050	0.001	4.146	0.000	0.003	0.007
relh	-0.0369	0.001	-27.301	0.000	-0.040	-0.034
sknt	0.0031	0.006	0.531	0.595	-0.008	0.014
alti	0.4188	0.103	4.054	0.000	0.216	0.621

Omnibus:	3092.933	Durbin-Watson:	0.354
Prob(Omnibus):	0.000	Jarque-Bera (JB):	7343.041
Skew:	-1.543	Prob(JB):	0.00
Kurtosis:	5.475	Cond. No.	1.49e+04

Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

To try and adjust for the negative skew and create a better fit, we must transform the y variable. To do this, we reflected the y so that there was a positive skew, and used the log transformation to try and reduce the skew. You can see slightly improved results. We know this due to the increased R- squared value and F statistic. We also notice that sknt is now significant, which is a change from the previous model.

Figure 5

Ordinary Least Squares Model for log of vsby(reflected) = skyc1 + tmpf + vrelh + sknt + altic

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        reflect_logvsby   R-squared:                       0.276
Model:                            OLS   Adj. R-squared:                  0.275
Method:                 Least Squares   F-statistic:                     475.6
Date:                Sun, 23 Feb 2025   Prob (F-statistic):               0.00
Time:                        10:35:51   Log-Likelihood:                -10746.
No. Observations:               11267   AIC:                         2.151e+04
Df Residuals:                   11257   BIC:                         2.159e+04
Df Model:                           9
Covariance Type:            nonrobust
==================================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
Intercept          3.9165      0.886      4.420      0.000       2.180       5.653
C(skyc1)[T.CLR]   -0.1412      0.019     -7.344      0.000      -0.179      -0.104
C(skyc1)[T.FEW]   -0.1054      0.024     -4.406      0.000      -0.152      -0.059
C(skyc1)[T.OVC]    0.3688      0.020     18.122      0.000       0.329       0.409
C(skyc1)[T.SCT]   -0.0668      0.025     -2.715      0.007      -0.115      -0.019
C(skyc1)[T.VV ]    1.7418      0.182      9.557      0.000       1.385       2.099
tmpf              -0.0013      0.000     -3.948      0.000      -0.002      -0.001
relh               0.0122      0.000     31.975      0.000       0.011       0.013
sknt               0.0036      0.002      2.235      0.025       0.000       0.007
alti              -0.1483      0.029     -5.106      0.000      -0.205      -0.091
==============================================================================
Omnibus:                     1885.269   Durbin-Watson:                   0.401
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3040.956
Skew:                           1.150   Prob(JB):                         0.00
Kurtosis:                       4.089   Cond. No.                     1.49e+04
==============================================================================
```

Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

General Linear Model with Log

$$g(\mu) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

General linear models (GLM) are similar to the ordinary least squares (OLS) model but offer more flexibility when dealing with skewed or unordinary data. They do not require the assumptions that ordinary least squares does. Another benefit of the GLM model is the use of the link function, which allows the linear model to be related to the response variable (Kumar, 2023). GLM can also fit other distribution types; in this case, we can see a gamma distribution if xx is reversed.

In our model, we observe a non-normal distribution, which logically suggests that a GLM model would be in our best interest. To further ensure that there is no multicollinearity, we also tested the Variance Inflation Factor (VIF). The VIF measures correlation by quantifying the increase due to the correlation (Agresti & Kateri, 2021). A high VIF indicates high correlation with other predictor variables, and it is generally best practice to keep it below 10.

Figure 6

VIF of remaining variables

	Feature	VIF
0	air_temperature	5.426510
1	humidity	6.728688
2	one_hour_precp	1.057841
3	cloud_ordinal	2.300363

Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Figure 7

GLM model of remaining variables

Dep. Variable:	visibility	No. Observations:	8787
Model:	GLM	Df Residuals:	8782
Model Family:	Gamma	Df Model:	4
Link Function:	Log	Scale:	0.041134
Method:	IRLS	Log-Likelihood:	-8496.1
Date:	Sat, 22 Feb 2025	Deviance:	637.44
Time:	18:29:06	Pearson chi2:	361.
No. Iterations:	13	Pseudo R-squ. (CS):	0.1961
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	1.0173	0.011	89.707	0.000	0.995	1.039
air_temperature	0.0007	0.000	6.091	0.000	0.000	0.001
humidity	-0.0027	0.000	-23.235	0.000	-0.003	-0.002
one_hour_precp	-0.6214	0.082	-7.614	0.000	-0.781	-0.461
cloud_ordinal	-0.0328	0.001	-22.633	0.000	-0.036	-0.030

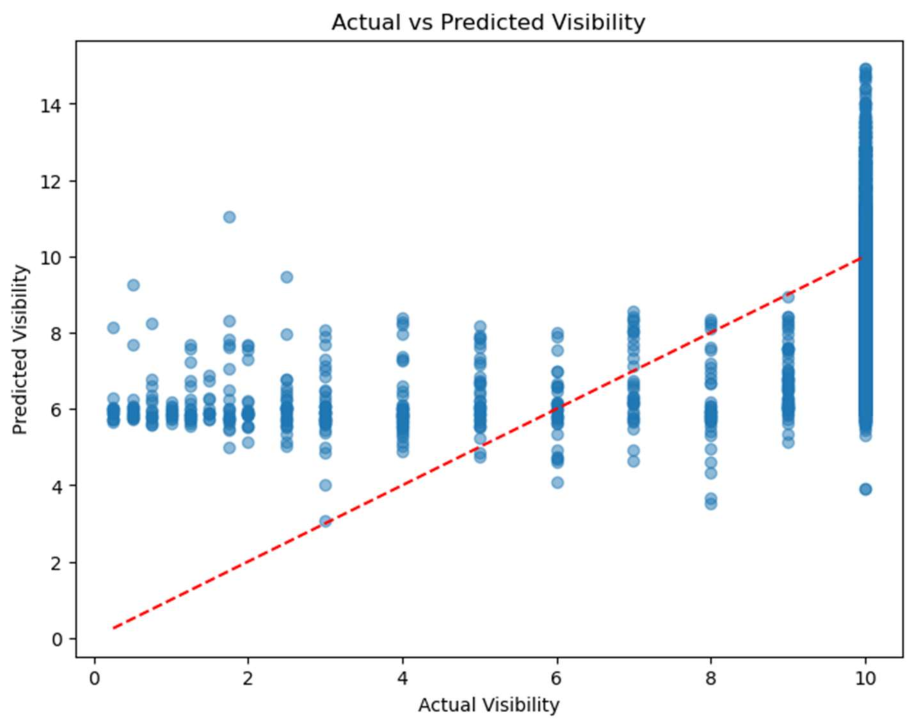
Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

When dealing with glm models, there are several important indicators to look at. Like the ordinary least squares model, we see the coefficients and how each variable impacts the model. In this example, we see a less than .05 p value and z values far from 0 for all of the predicting variables, indicating that all of the variables are significant. Deviance shows how much a model improves when predictors are added and generally the lower the score the better. An important indicator not shown is the mean absolute error which is used to measure the absolute differences between actual and predicted values. A lower MAE usually indicates better model performance. In our case, our MAE was 2.11.

After we created a model, we can better predict how it will perform by dividing the data into test and predict. You can see the model significantly predicts higher than the actual for most data points. Unfortunately, our model did not predict visibility well using our current variables and transformations.

Figure 7

GLM model of remaining variables



Note. Data from Iowa State University College of Ag. (2025). ASOS-AWOS-METAR Data Download. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Discussion

One of the biggest issues with dealing with this data is the large number of null data. This makes it difficult to use and we don't know what the reason for the missing data. The skew in the data set also makes it very difficult to work with, and negative skews are often difficult to deal with. A better model would be likely if we could account for these two issues effectively.

Conclusions

Based on our models and the comparison of p-values for each climate indicator, we can reject the null hypothesis that there is no association between climate indicators and visibility. However, fitting our model proved challenging due to missing variables and a negative skew in the distribution, which complicated the analysis. For future use, it would be beneficial to find more complete datasets that can provide accurate and reliable information. Additionally, it may be useful to explore machine learning techniques in the future.

References

Agresti, A., & Kateri, M. (2021). *Foundations of Statistics for Data Scientists*. CRC Press.

Dugan, Z., & Greyserman, A. (2019). The impact of skew on performance and bias: How skew distorts short term performance, triggers bias, and changes drawdowns. *Journal of Behavioral and Experimental Finance*, *22*, 232–238. https://doi.org/10.1016/j.jbef.2019.03.008

Fan, J., Han, F., & Liu, H. (2014). Challenges of Big Data analysis. *National Science Review*, *1*(2), 293–314. https://doi.org/10.1093/nsr/nwt032

Iowa State University College of Ag. (2025). *ASOS-AWOS-METAR Data Download*. Iastate.edu. https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS

Kadam, G., Tobaria, A., Arya, S., Prof, A., & Kaushik, J. (2023). Climate Visibility Prediction Using Machine Learning. In *International Research Journal of Engineering and Technology*. https://www.irjet.net/archives/V10/i5/IRJET-V10I5281.pdf

Kang, H. (2013). The prevention and handling of the missing data. *Korean Journal of Anesthesiology*, *64*(5), 402–406. https://doi.org/10.4097/kjae.2013.64.5.402

Kumar, A. (2023, December 7). *GLM vs Linear Regression: Difference, Examples*. Analytics Yogi. https://vitalflux.com/glm-vs-linear-regression-difference-examples/

Liang, C.-W., Chang, C.-C., Hsiao, C.-Y., & Liang, C.-J. (2023). Prediction and analysis of atmospheric visibility in five terrain types with artificial intelligence. *Heliyon*, *9*(8), e19281. https://doi.org/10.1016/j.heliyon.2023.e19281

Ortega, L. C., Otero, L. D., Solomon, M., Otero, C. E., & Fabregas, A. (2022). Deep learning models for visibility forecasting using climatological data. *International Journal of Forecasting*, *39*(2). https://doi.org/10.1016/j.ijforecast.2022.03.009

U.S. Department of Transportation. (2024). *Evaluating the Performance of Runway Visual Range Sensors for FAA | Volpe National Transportation Systems Center*. Dot.gov. https://www.volpe.dot.gov/news/evaluating-performance-runway-visual-range-sensors-faa

Vatcheva, K. P., & Lee, M. (2016). Multicollinearity in Regression Analyses Conducted in Epidemiologic Studies. *Epidemiology: Open Access*, *06*(02). https://doi.org/10.4172/2161-1165.1000227

Williams, M., Alberto, C., Grajales, G., & Kurkiewicz, D. (2013). Assumptions of Multiple Regression: Correcting Two Misconceptions. *Practical Assessment, Research & Evaluation*, *18*(11). https://files.eric.ed.gov/fulltext/EJ1015680.pdf

Zhang, Y., Wang, Y., Zhu, Y., Yang, L., Ge, L., & Luo, C. (2022). Visibility Prediction Based on Machine Learning Algorithms. *Atmosphere*, *13*(7), 1125–1125. https://doi.org/10.3390/atmos13071125

Appendix

Final Project Code

Code written by Birendra Khimding and Andrew Fennimore

Github: https://github.com/Fenn3963/Weather-Impact-on-Air-Traffic-Management

In [86]:

```
import pandas as pd
import numpy as np
from IPython.display import display, HTML
import statsmodels.formula.api as smf
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

Retreive data from MJX

In [64]:

```
# pull from the MJX
url = "https://raw.githubusercontent.com/Fenn3963/Weather-Impact-on-Air-Traffic-Management/refs/heads/main/MJX.csv"

#All values with na are labeled as M
weather = pd.read_csv("MJX.csv" , na_values= "M")
```

Create dictionary for descriptive stats and other pertinant information

In [66]:

```
# Column descriptions dictionary, retreived directly from https://mesonet.agron.iastate.edu/request/download.phtml?network=NJ_ASOS
column_descriptions = {
    "station": "Three or four character site identifier",
    "valid": "Timestamp of the observation",
    "tmpf": "Air Temperature in Fahrenheit, typically @ 2 meters",
    "dwpf": "Dew Point Temperature in Fahrenheit, typically @ 2 meters",
    "relh": "Relative Humidity in %",
    "drct": "Wind Direction in degrees from *true* north",
    "sknt": "Wind Speed in knots",
    "p01i": "One hour precipitation for the period from the observation time to the time of the previous hourly precipitation reset. This varies slightly by site. Values are in inches. This value may or may not contain frozen precipitation melted by some device on the sensor or estimated by some other means. Unfortunately, we do not know of an authoritative database denoting which station has which sensor.",
```

```
    "alti": "Pressure altimeter in inches",
    "mslp": "Sea Level Pressure in millibar",
    "vsby": "Visibility in miles",
    "gust": "Wind Gust in knots",
    "skyc1": "Sky Level 1 Coverage",
    "skyc2": "Sky Level 2 Coverage",
    "skyc3": "Sky Level 3 Coverage",
    "skyc4": "Sky Level 4 Coverage",
    "skyl1": "Sky Level 1 Altitude in feet",
    "skyl2": "Sky Level 2 Altitude in feet",
    "skyl3": "Sky Level 3 Altitude in feet",
    "skyl4": "Sky Level 4 Altitude in feet",
    "wxcodes": "Present Weather Codes (space separated)",
    "feel": "Apparent Temperature (Wind Chill or Heat Index) in Fahrenheit",
    "ice_accretion_1hr": "Ice Accretion over 1 Hour (inches)",
    "ice_accretion_3hr": "Ice Accretion over 3 Hours (inches)",
    "ice_accretion_6hr": "Ice Accretion over 6 Hours (inches)",
    "peak_wind_gust": "Peak Wind Gust (from PK WND METAR remark) (knots)",
    "peak_wind_drct": "Peak Wind Gust Direction (from PK WND METAR remark) (deg)",
    "peak_wind_time": "Peak Wind Gust Time (from PK WND METAR remark)",
    "metar": "Unprocessed reported observation in METAR format"
}

################################################################################
###################
# Split up the quantitative and qualitative data
quant = weather.select_dtypes(include=["number"])
qual = weather.select_dtypes(exclude=["number"])

# create dictionary of the statsistical information and descriptions
stats_dict = {}

################################################################################
###################

# Quantitative stats
for col in quant.columns:
    mode_values = quant[col].mode().dropna().tolist()
    if mode_values:
        mode = mode_values
    else:
        mode = None

    # Calculate stats and give description
    count = quant[col].count()
```

```
    mean = round(quant[col].mean(), 2)
    median = round(quant[col].median(), 2)
    std = round(quant[col].std(), 2)
    data_type = "Quantitative"
    description = column_descriptions.get(col)

    # Find the percentage of null values
    null_percentage = round((quant[col].isnull().sum() / len(quant[col])) * 100, 2) #find
percentage of values with "none"

    # Create stats dictionary
    stats = {
        "Description": description,
        "Data Type": data_type,
        "Count": count,
        "Mean": mean,
        "Median": median,
        "Std": std,
        "Mode": mode,
        "Null Percentage": f"{null_percentage}%"  #% that doesn't have values
    }

    # Filter out None values to then store in the dictionary, used to calculate percentage
    stats_filtered = {}
    for k, v in stats.items():
        if v is not None:
            stats_filtered[k] = v

    stats_dict[col] = stats_filtered

##############################################################################
##################

# Qualitative stats
for col in qual.columns:
    mode_values = qual[col].mode().dropna().tolist()

    # If every value is unique, set mode to None
    if len(mode_values) == len(qual[col].dropna().unique()):
        mode_output = None
    else:
        if mode_values:
            mode_output = mode_values
        else:
            mode_output = None
```

```
    # Get the count
    if mode_output is not None:
        most_frequent_count = qual[col].value_counts().iloc[0]
    else:
        most_frequent_count = None

    # Calculate all the stats for qualitative portion
    count = qual[col].count()
    unique_values = qual[col].nunique()
    data_type = "Qualitative"
    description = column_descriptions.get(col, "No description available")

    # Calculate the percentage of null values
    null_percentage = round((qual[col].isnull().sum() / len(qual[col])) * 100, 2) #find percentage
of values with the none value

    # Create stats dictionary
    stats = {
        "Description": description,
        "Data Type": data_type,
        "Count": count,
        "Mode": mode_output,
        "Unique Values": unique_values,
        "Most Frequent Count": most_frequent_count,
        "Null Percentage": f"{null_percentage}%"
    }

    # Filter out None values and store in stats_dict
    stats_filtered = {}
    for k, v in stats.items():
        if v is not None:
            stats_filtered[k] = v
    stats_dict[col] = stats_filtered

################################################################################
###################

# Print in green
html_code = '<p style="font-size:20px; color:green;">Description of columns:</p>'
display(HTML(html_code)) #makes it look nicer

"""
#print all of the variables and statistics associated
for col, stats in stats_dict.items():
```

```
    print(f"\nStatistics for '{col}':")
    for key, value in stats.items():
        print(f"  {key}: {value}")
"""
```

Description of columns:

Out[66]:

'\n#print all of the variables and statistics associated\nfor col, stats in stats_dict.items():\n print(f"\nStatistics for \'{col}\':")\n for key, value in stats.items():\n print(f" {key}: {value}")\n'

Creating seperate charts of the stats so it is easier to view

In [68]:

```
#This will create seperate external files based on the data information


#Create a seperate csv file of dictionary so it is easier to view
des_chart = pd.DataFrame(stats_dict).T  # transpose to have variables as rows

# Drop the description since I am putting it in another seperate csv
if "Description" in des_chart.columns:
    des_chart = des_chart.drop(columns=["Description"])

# filename used, can easily change if need be
filename = "weather_variables.csv"

# Save as csv to a whole new file
des_chart.to_csv(filename, index=True)
```

In [69]:

```
#Creates a separate csv to show variable's descriptions
descriptions = pd.DataFrame(list(column_descriptions.items()), columns=["Variable",
"Description"]) #single out the descriptions from the dictionary

# Define the CSV filename
filename = "variable_descriptions.csv"
descriptions.to_csv(filename, index=False)
```

Dealing with missing data

In [71]:

```
# Number of Missing vlaues in the dataframe
weather.isna().sum()
```

Out[71]:

```
station             0
```

```
valid                  0
tmpf                   5
dwpf                   7
relh                   7
drct                 282
sknt                   8
p01i                   3
alti                 111
mslp                2495
vsby                   1
gust                9568
skyc1                 30
skyc2               9246
skyc3              10618
skyc4              11423
skyl1               4939
skyl2               9246
skyl3              10618
skyl4              11423
wxcodes             9072
ice_accretion_1hr  11423
ice_accretion_3hr  11423
ice_accretion_6hr  11423
peak_wind_gust     10971
peak_wind_drct     10971
peak_wind_time     10971
feel                   9
metar                  0
snowdepth          11423
dtype: int64
```

In [72]:

```
# Setting a threshold to remove any column with more then 15% missing value
threshold = len(weather)*.15
cols_drop_nan = weather.columns[weather.isna().sum() <= threshold]

# Drop row with missing values
weather.dropna(subset=cols_drop_nan, inplace=True)

# Droping columns with more then 15% missing values

cols_to_drop = weather.columns[weather.isna().sum() > 0]
print(cols_to_drop)
weather.drop(columns=cols_to_drop , inplace=True)
```

```
Index(['mslp', 'gust', 'skyc2', 'skyc3', 'skyc4', 'skyl1', 'skyl2', 'skyl3',
       'skyl4', 'wxcodes', 'ice_accretion_1hr', 'ice_accretion_3hr',
```

```
       'ice_accretion_6hr', 'peak_wind_gust', 'peak_wind_drct',
       'peak_wind_time', 'snowdepth'],
      dtype='object')
```

OLS Model

In [74]:

```
#put here since it uses old variables
# Model 1: Predict visibility using various weather variables
model = smf.ols(formula="vsby ~ tmpf + relh + sknt + alti + C(skyc1)", data=weather).fit()

# Print summary of the model
print(model.summary())

#Inversing and transforming to deal with negative skew
import numpy as np
import statsmodels.formula.api as smf

# tyring an inverse log tranformation
K = weather["vsby"].max() + 1
weather["reflect_log_vsby"] = np.log(K - weather["vsby"])

# Fit the ols model using the new variable
model = smf.ols(formula="reflect_log_vsby ~ tmpf + relh + sknt + alti + C(skyc1)",
data=weather).fit()

# View the summary of the model
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
=========
Dep. Variable:                   vsby   R-squared:                       0.252
Model:                            OLS   Adj. R-squared:                  0.252
Method:                 Least Squares   F-statistic:                     412.2
Date:                Sun, 23 Feb 2025   Prob (F-statistic):               0.00
Time:                        17:40:25   Log-Likelihood:                -24505.
No. Observations:               10996   AIC:                         4.903e+04
Df Residuals:                   10986   BIC:                         4.910e+04
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
==============
                      coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept          -0.5139      3.209     -0.160      0.873      -6.805       5.777
C(skyc1)[T.CLR]     0.5108      0.070      7.327      0.000       0.374       0.647
```

```
C(skyc1)[T.FEW]     0.3932      0.087      4.526      0.000       0.223       0.564
C(skyc1)[T.OVC]    -1.3546      0.074    -18.396      0.000      -1.499      -1.210
C(skyc1)[T.SCT]     0.2936      0.090      3.280      0.001       0.118       0.469
C(skyc1)[T.VV ]    -7.2806      0.652    -11.163      0.000      -8.559      -6.002
tmpf                0.0052      0.001      4.268      0.000       0.003       0.008
relh               -0.0375      0.001    -26.888      0.000      -0.040      -0.035
sknt                0.0015      0.006      0.257      0.797      -0.010       0.013
alti                0.3989      0.105      3.793      0.000       0.193       0.605
==============================================================================
=========
Omnibus:                     2981.840   Durbin-Watson:                   0.354
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             6985.271
Skew:                          -1.531   Prob(JB):                         0.00
Kurtosis:                       5.424   Cond. No.                     1.49e+04
==============================================================================
=========
```

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.49e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

```
                            OLS Regression Results
==============================================================================
=========
Dep. Variable:      reflect_log_vsby   R-squared:                       0.276
Model:                           OLS   Adj. R-squared:                  0.276
Method:                Least Squares   F-statistic:                     466.1
Date:               Sun, 23 Feb 2025   Prob (F-statistic):               0.00
Time:                       17:40:25   Log-Likelihood:                 -10534.
No. Observations:              10996   AIC:                         2.109e+04
Df Residuals:                  10986   BIC:                         2.116e+04
Df Model:                          9
Covariance Type:           nonrobust
==============================================================================
==============
                     coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------
Intercept           3.7340      0.901      4.145      0.000       1.968       5.500
C(skyc1)[T.CLR]    -0.1439      0.020     -7.355      0.000      -0.182      -0.106
C(skyc1)[T.FEW]    -0.1074      0.024     -4.402      0.000      -0.155      -0.060
C(skyc1)[T.OVC]     0.3710      0.021     17.951      0.000       0.330       0.412
C(skyc1)[T.SCT]    -0.0687      0.025     -2.734      0.006      -0.118      -0.019
C(skyc1)[T.VV ]     1.7404      0.183      9.507      0.000       1.382       2.099
tmpf               -0.0014      0.000     -4.147      0.000      -0.002      -0.001
relh                0.0123      0.000     31.574      0.000       0.012       0.013
```

```
sknt            0.0042      0.002      2.559      0.011       0.001       0.007
alti           -0.1426      0.030     -4.832      0.000      -0.201      -0.085
==============================================================================
Omnibus:                     1815.734   Durbin-Watson:                   0.402
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2907.645
Skew:                           1.142   Prob(JB):                         0.00
Kurtosis:                       4.064   Cond. No.                     1.49e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.49e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Improving data readability and accessibility

In [76]:

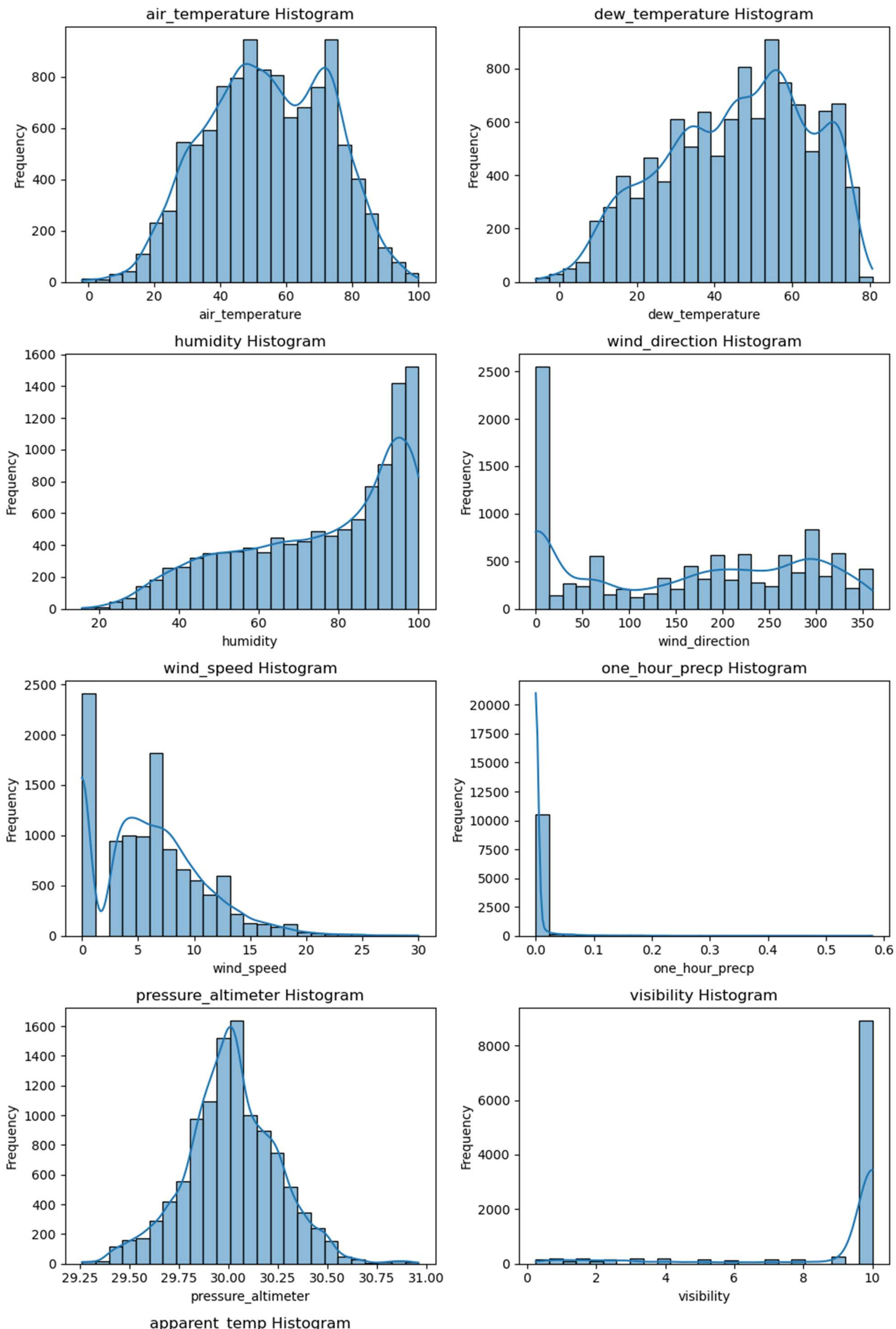
```
weather.rename(columns={'valid': 'timestamp', 'tmpf': 'air_temperature','dwpf':'dew_temperature', 'relh':'humidity', 'drct':'wind_direction', 'sknt':'wind_speed', 'p01i':'one_hour_precp' , 'alti':'pressure_altimeter', 'vsby':'visibility', 'skyc1':'cloud_coverage', 'feel':'apparent_temp', 'metar':'unprocessed_observation'  } , inplace=True)
weather.columns

#Replacing Char value to float
weather['one_hour_precp'] = weather['one_hour_precp'].replace('T', '0.001')

#Changing the one_hour_precp column data type to float
weather['one_hour_precp'] = weather['one_hour_precp'].astype(float)

# Checking for number of 0.0 vlueas in the dataframe
col_with_zeor = (weather == 0.0).sum()
print(col_with_zeor)
station                    0
timestamp                  0
air_temperature            1
dew_temperature           17
humidity                   0
wind_direction          2413
wind_speed              2413
one_hour_precp          9015
pressure_altimeter         0
visibility                 0
cloud_coverage             0
apparent_temp              1
```

```
unprocessed_observation       0
reflect_log_vsby          8928
dtype: int64
```

Univariate EDA (Single Variable Analysis)

In [78]:

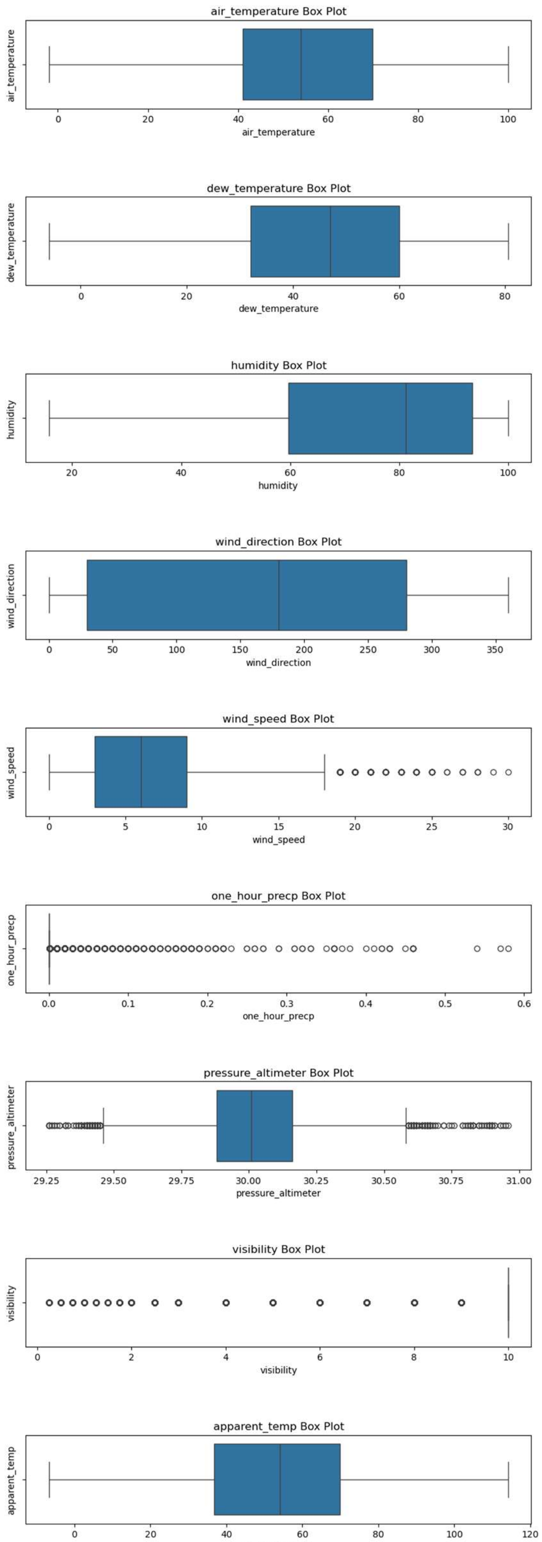
```
# Univariate EDA (Single Variable Analysis)

# Histogram for all the Numerical Column
import seaborn as sns
import matplotlib.pyplot as plt

columns_to_plot = ['air_temperature', 'dew_temperature', 'humidity','wind_direction'
,'wind_speed', 'one_hour_precp', 'pressure_altimeter', 'visibility', 'apparent_temp']

fig, axes = plt.subplots(5, 2, figsize=(10, 18))
axes = axes.flatten()
for i, col in enumerate(columns_to_plot):
    sns.histplot(weather[col], kde=True, ax=axes[i], bins=25)  # kde=True adds a density curve
    axes[i].set_title(f'{col} Histogram')
    axes[i].set_xlabel(col)
    axes[i].set_ylabel('Frequency')

if len(columns_to_plot) < len(axes):
    axes[-1].set_visible(False)

# Adjust layout
plt.tight_layout()
plt.show()
```


air_temperature Histogram
Frequency
air_temperature
dew_temperature Histogram
Frequency
dew_temperature
humidity Histogram
Frequency
humidity
wind_direction Histogram
Frequency
wind_direction
wind_speed Histogram
Frequency
wind_speed
one_hour_precp Histogram
Frequency
one_hour_precp
pressure_altimeter Histogram
Frequency
pressure_altimeter
visibility Histogram
Frequency
visibility
apparent_temp Histogram

Box plot for all the numerical columns

In [80]:

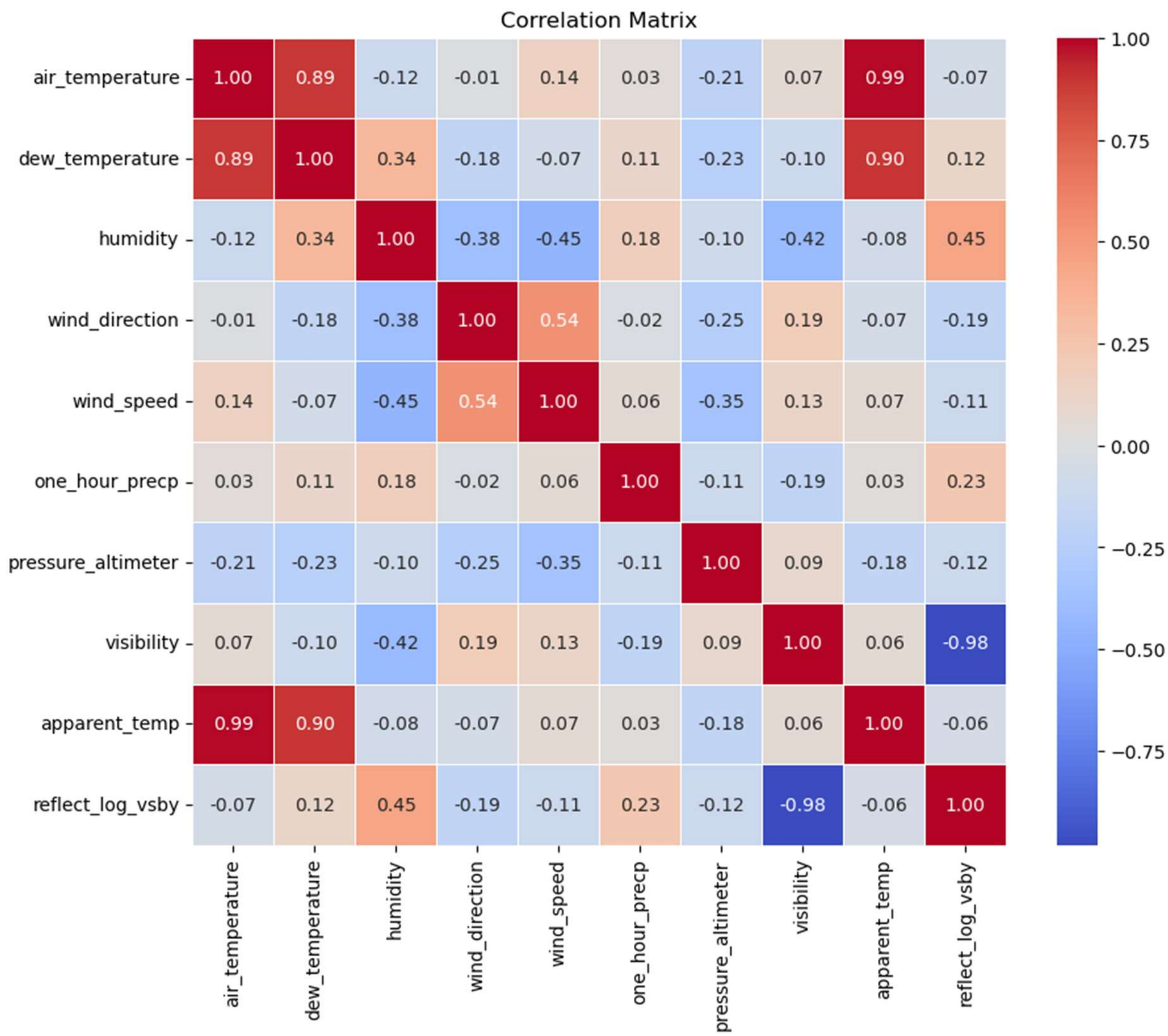
```
fig, axes = plt.subplots(9, 1, figsize=(10, 30))
axes = axes.flatten()

for i, col in enumerate(columns_to_plot):
    if col in weather.columns:
        sns.boxplot(x=weather[col], ax=axes[i])  # Box plot for each variable
        axes[i].set_title(f'{col} Box Plot')
        axes[i].set_ylabel(col)
    else:
        print(f"Warning: Column '{col}' not found in DataFrame.")

# Hide the last unused subplot if necessary
if len(columns_to_plot) < len(axes):
    axes[-1].set_visible(False)

plt.subplots_adjust(hspace=1)
plt.show()
```



Bivariate EDA (Two Variable Analysis)

In [82]:

```
#Bivariate EDA (Two Variable Analysis)

# Drop non-numeric columns
weather_numeric = weather.select_dtypes(include=['number'])

# The correlation matrix
correlation_matrix = weather_numeric.corr()

# Cloud_coverage and Visibility
visibility_by_cloud = weather.groupby('cloud_coverage')['visibility'].mean()

# Visibility with all other Numerical columns
pairs_to_plot = [
    ('air_temperature', 'visibility'),
    ('humidity', 'visibility'),
    ('wind_speed', 'visibility'),
    ('dew_temperature', 'visibility'),
    ('one_hour_precp', 'visibility'),
    ('pressure_altimeter', 'visibility')
]

# Multivariate EDA (Multiple Variables Analysis)

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```



In [83]:

```
#Change the Cloud catagorical data to ordinal numerical data
# Create a dictionary to map each category
ordinal_mapping = {
    'CLR': 0,
    'FEW': 1,
    'SCT': 2,
    'BKN': 3,
    'OVC': 4,
    'VV': 5
}

weather['cloud_ordinal']= weather['cloud_coverage'].map(ordinal_mapping)
weather['cloud_ordinal'].value_counts()
weather = weather.dropna(subset=['cloud_ordinal'])
```

```
# Slpit the data into Dependent and Independent variable
x = weather.drop(columns=['visibility', 'cloud_coverage', 'station', 'timestamp',
'unprocessed_observation', 'dew_temperature' , 'apparent_temp', 'wind_direction',
'pressure_altimeter', 'wind_speed'])
y = weather['visibility']

# Taing the data using 80% and predicting the 20% x data with the y
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1)
```

GLM model

In [84]:

```
#Using GLM for Model Selectin becuase our dependent data is not normal and is skewed.
# GLM with out interection
import statsmodels.api as sm
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Add intercept
x_const = sm.add_constant(x_train)

# Apply log to Y
y_log_transformed = np.log1p(y_train)

# Gamma GLM
glm_no_interaction = sm.GLM(y_log_transformed, x_const,
family=sm.families.Gamma(link=sm.families.links.Log())).fit()
print(glm_no_interaction.summary())
```

```
                 Generalized Linear Model Regression Results
==============================================================================
=========
Dep. Variable:           visibility   No. Observations:              8787
Model:                          GLM   Df Residuals:                  8781
Model Family:                 Gamma   Df Model:                         5
Link Function:                  Log   Scale:                     0.020553
Method:                        IRLS   Log-Likelihood:             -3607.0
Date:              Sun, 23 Feb 2025   Deviance:                    243.46
Time:                      17:40:27   Pearson chi2:                  180.
No. Iterations:                  16   Pseudo R-squ. (CS):          0.9271
Covariance Type:          nonrobust
==============================================================================
==============
                  coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
```

```
const              0.8398      0.008    103.241      0.000       0.824       0.856
air_temperature   9.147e-05   8.42e-05     1.087      0.277   -7.35e-05      0.000
humidity           0.0006   8.71e-05      6.894      0.000       0.000       0.001
one_hour_precp      0.9863      0.058     16.879      0.000       0.872       1.101
reflect_log_vsby   -0.3635      0.002   -150.363      0.000      -0.368      -0.359
cloud_ordinal      -0.0007      0.001     -0.705      0.481      -0.003       0.001
==============================================================================
```

In [88]:

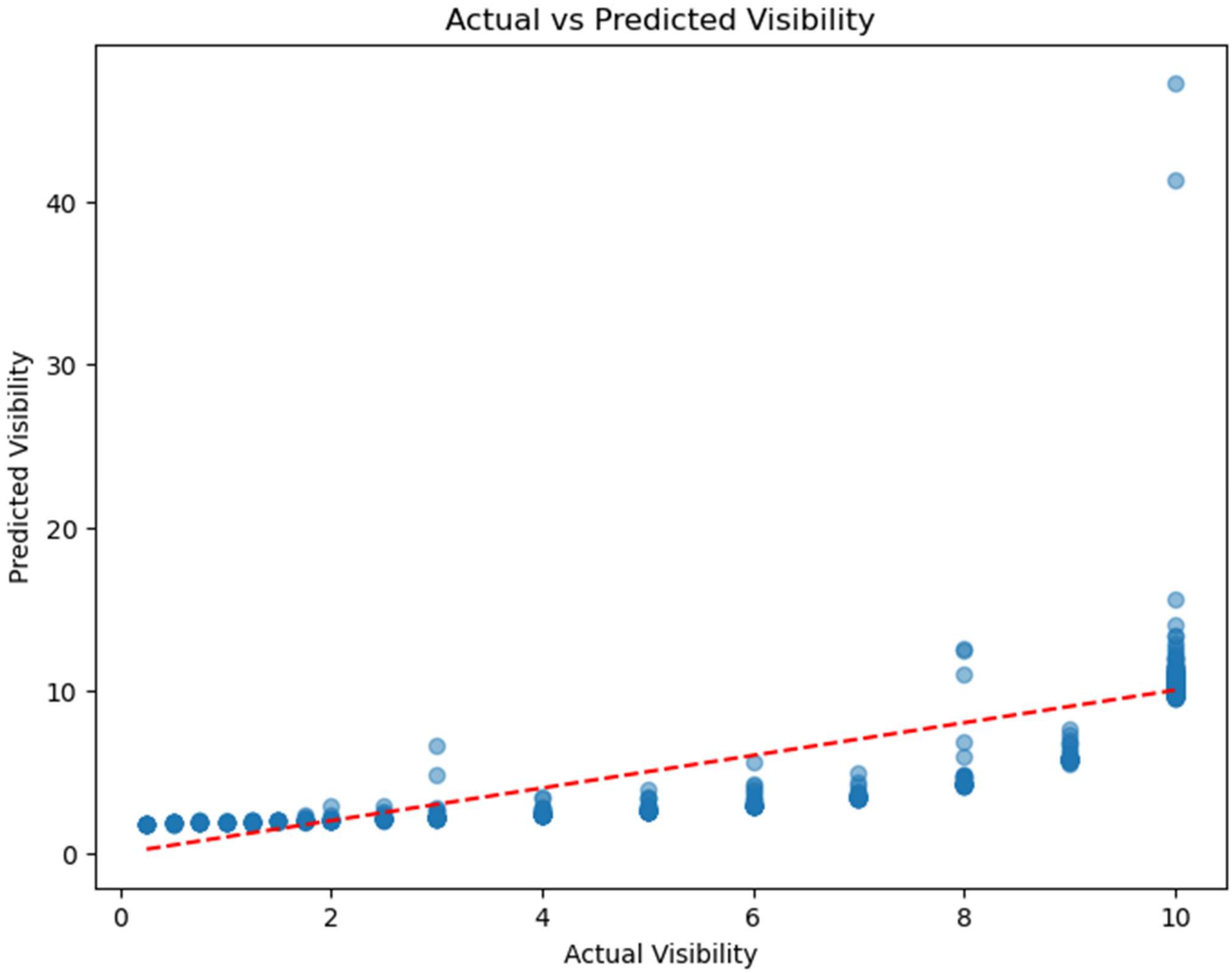
```
# VIF for each Predictor
vif_data = pd.DataFrame()
vif_data["Feature"] = x_train.columns
vif_data["VIF"] = [variance_inflation_factor(x_train.values, i) for i in
range(len(x_train.columns))]


print(vif_data)
           Feature       VIF
0   air_temperature  5.742731
1          humidity  7.306205
2    one_hour_precp  1.092184
3  reflect_log_vsby  1.599495
4     cloud_ordinal  2.510221
```

In [90]:

```
# Test the Gamma Model
x_test_const = sm.add_constant(x_test)
x_predict = glm_no_interaction.predict(x_test_const)
x_predict_exp = np.expm1(x_predict)

# Check the performace of the Model

mae = mean_absolute_error(y_test, x_predict_exp)
r2 = r2_score(y_test, x_predict_exp)
Mean Absolute Error (MAE): 0.7248092889421853
r2_score (MAE): 0.6452686254306526
```

In [92]:

```
plt.figure(figsize=(8, 6))
plt.scatter(y_test, x_predict_exp, alpha=0.5)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--')
plt.xlabel('Actual Visibility')
plt.ylabel('Predicted Visibility')
plt.title('Actual vs Predicted Visibility')
plt.show()
```


Actual vs Predicted Visibility
Predicted Visibility
Actual Visibility
0
10
20
30
40
0
2
4
6
8
10

In [ ]: