[Write-up, visuals, in-code comments & explanations, and citations available on Jupyter Notebook (ipynb) document]

SUMMARY

Quadratic approximation is the most effective method for filling in missing data, as it uses three points for estimation, unlike linear approximation which uses only two. However, when projecting future values, linear approximation performs better. This is evident when comparing the scatterplots for the final week, where the quadratic approximation skews the estimated points, while the linear approximation continues smoothly based on the two closest points.

Hot deck approximation, while useful for categorical data with repeatable patterns, like surveys, is unsuitable for gas prices. Since gas prices follow a time-dependent trend, randomly selecting past prices does not provide an accurate estimate.

Gas prices generally follow a somewhat linear trend, though they are not perfectly linear. Adding noise can make the estimations more realistic, but as shown in the histograms, prices tend to cluster around specific points, with variances of 0.05 and 0.2, so a standard deviation of 1.5 is too large, as we can see, the scatterplots of that noise don't produce a good approximation of the missing data. Using a standard deviation of 0.1, as shown in the alternative approach in each case, produces a much better and realistic approximation of the missing data.

Finally, the linear regression line offers a useful high-bias model, highlighting the overall trend rather than focusing on individual data points. In this project, it reveals a gradual decrease in regular gas prices and a sharper decline in diesel prices over time, allowing us to draw meaningful conclusions about the long-term trends in gas prices.

CITATIONS

- https://pandas.pydata.org/docs/getting_started/intro_tutorials/09_timeseries.html (to tidy my data)
- https://pandas.pydata.org/docs/user_guide/timeseries.html (to add the extra week)
- https://scikitlearn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html#sklearn.linear_model.LinearRegression (to learn how to calculate the linear regression)
- *Ilana Berlin* (Helped me clarify stuff about the math)