Practical Part – Answers Yonatan Sheffer

3. Data Exploration & Preprocessing Decisions

While exploring the data, I checked the types and value ranges of each column to look for invalid or illogical entries. Based on this analysis, I made several decisions about which rows and features to keep or remove. For example:

-Bedrooms: I noticed that some rows had bedrooms = 0, which doesn't make sense for a living space. I removed those rows.

-Square Footage (sqft_living, sqft_lot, sqft_lot15): These values represent area and must be positive. I made sure to remove any rows with negative values.

-Years (yr_built, yr_renovated): A house can’t be renovated before it was built, so I removed rows where yr_renovated < yr_built.

Features Removed

To clean the data and improve the model’s accuracy, I removed features that were either not useful for prediction, had invalid or illogical values, or were duplicates or missing. For example:

-id: This is just an identifier and doesn't give useful information for predicting house price, so I dropped it.

-date: It was hard to extract useful numeric features from the date, so I also removed it.

Features Kept

I kept all numeric and categorical features that had logical values and looked relevant to the price.

Additional Features Created

After reviewing the existing features, I thought about creating new ones that might give the model more helpful information. These were based on basic logic and what usually affects house prices:

-house_age = 2015 - yr_built: Older houses might be cheaper or need more maintenance.

-was_renovated = 1 if yr_renovated > 0, else 0: Renovated houses often sell for higher prices.

-total_rooms = bedrooms + bathrooms: Gives a simple measure of how many rooms the house has in total.

-living_to_lot_ratio = sqft_living / sqft_lot: Shows how much of the space is actually used for living space.

Handling Invalid or Missing Values

I removed rows with missing values (there were very few),I removed duplicate rows to avoid repeated data affecting the results. I also dropped rows that had illogical values, like those mentioned above.

4. Feature Evaluation

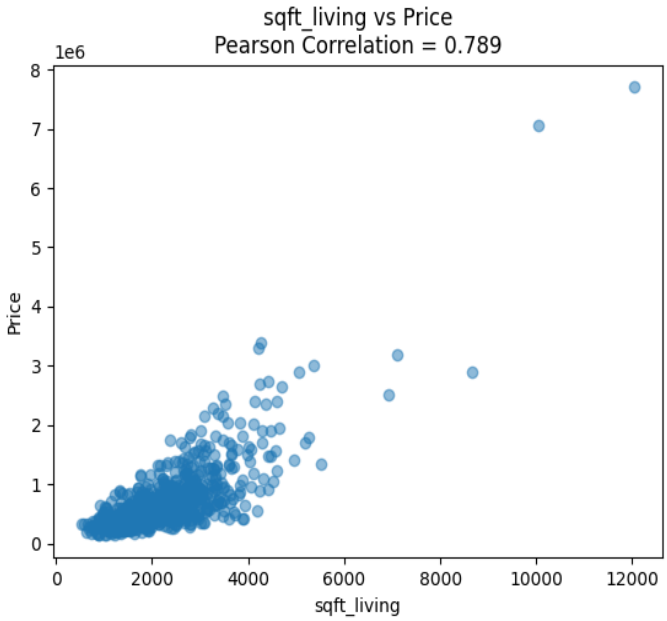
Beneficial Feature: sqft_living

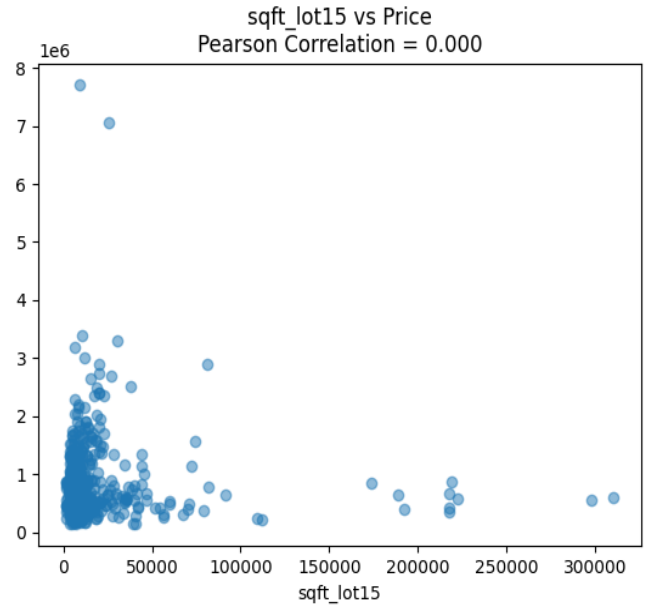
Correlation: The Pearson correlation between sqft_living and price is 0.789, which indicates a strong positive relationship. This means that as the living area of the house increases, the price tends to increase as well. Because of that i can conclude that, sqft_living is highly beneficial for predicting house prices. It makes intuitive sense that larger living spaces would result in higher prices, and the correlation confirms this relationship.

Non-Beneficial Feature: sqft_lot15

Correlation: The Pearson correlation between sqft_lot15 and price is 0.0, indicating no linear relationship. This suggests that the lot size of the house (in the context of the 15 nearest neighbors) does not have any meaningful impact on the price of the house. So i can conclude that although lot size might intuitively seem relevant, this feature does not correlate with price and may not improve the model's predictions.

Plots of these features vs house price:





6. In the plot you can see:

Trend in Loss (Dark Blue Line)

General Behavior: The mean test loss decreases as the percentage of training data increases from 10% to 100%. However, it does not decrease linearly or smoothly.

Early Phase (10%–30%): There's a steep drop in loss initially. This is expected because small training sets result in underfitting, leading to high errors. Adding more data improves the model quickly.

Middle Range (30%–70%): The loss fluctuates but shows signs of stabilization. The fluctuations may be due to the variance in the random samples drawn at each step.

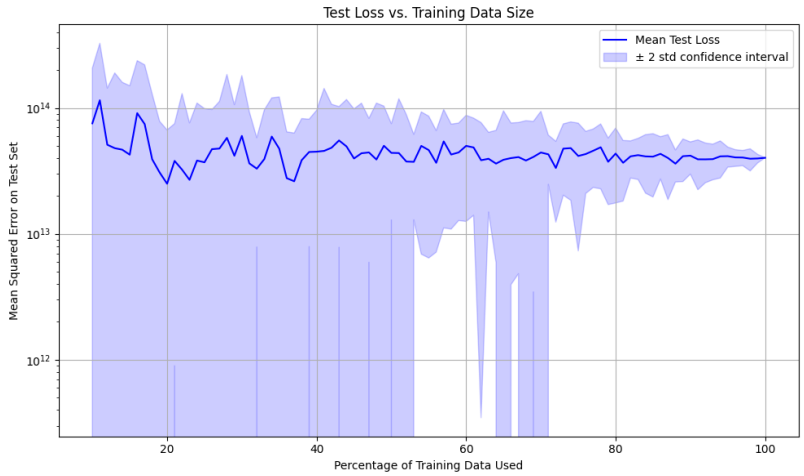
Late Phase (70%–100%): The mean test loss becomes more stable, indicating that the model has seen enough data to generalize well, and additional data doesn't significantly change performance.

Trend in Confidence Interval (Shaded Area)

Early Phase (10%–30%): The confidence interval is very wide, indicating high variability in model performance. Small training sets lead to instability in the fitted model due to random sampling.

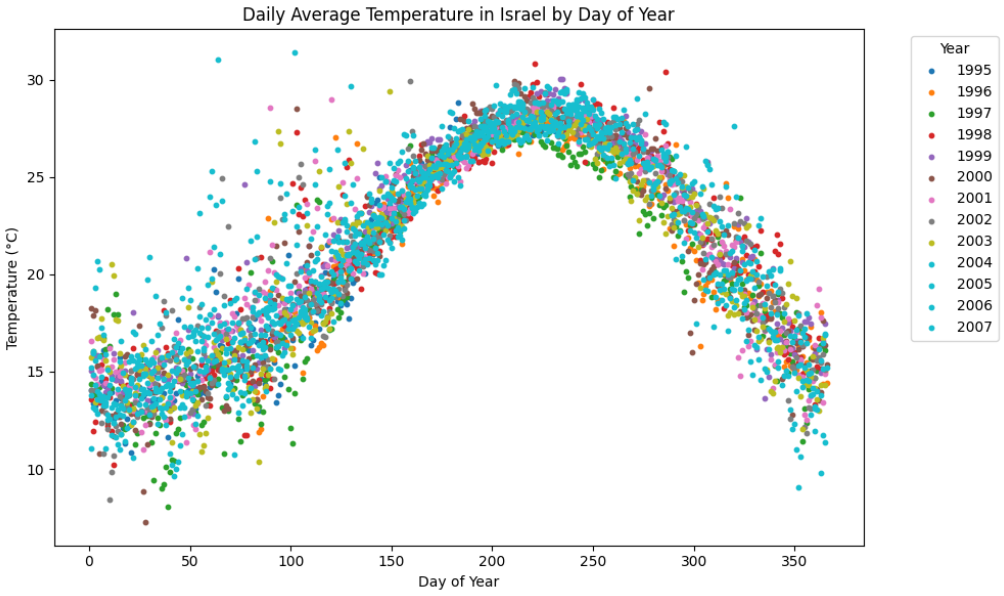
Middle Range (30%–70%): The interval narrows slightly, but there are still notable spikes (especially around 60–70%), suggesting occasional unstable behavior due to unlucky sampling or noisy data.

Late Phase (70%–100%): The confidence interval narrows considerably, meaning the model's performance becomes more consistent across different random samples. This is a good sign—it shows that the model is reliable when trained on more data.

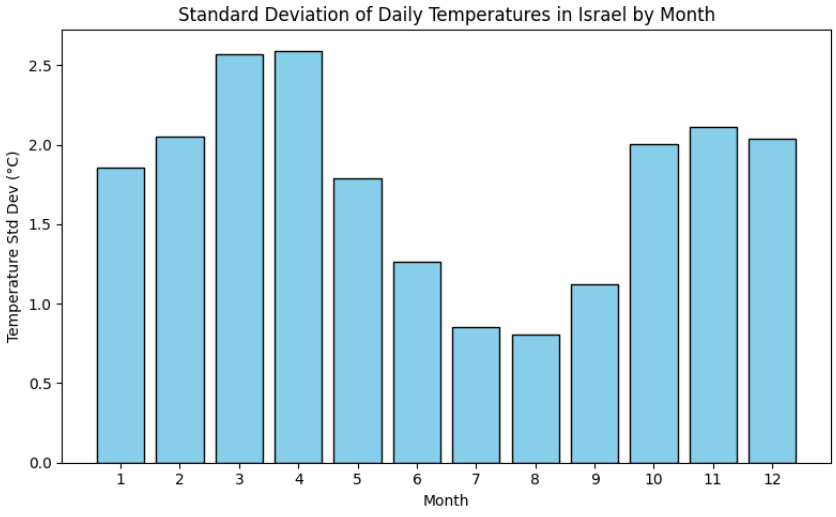


Polynomial Fitting

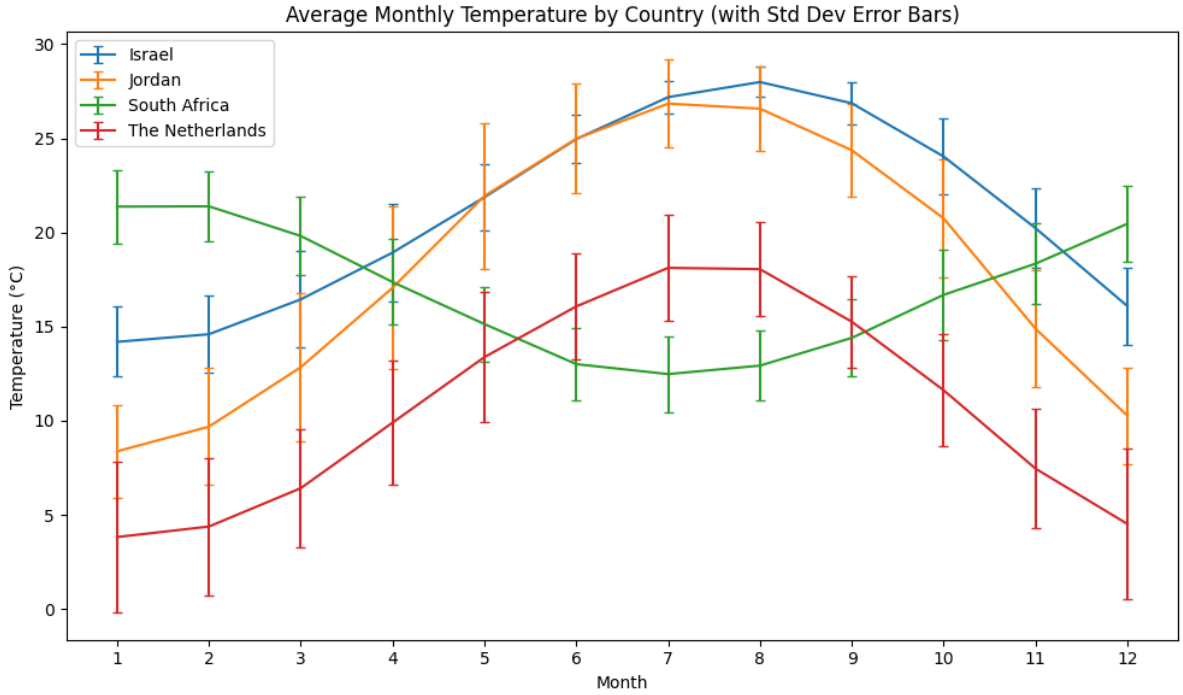
3. For the scatter plot: A suitable polynomial degree for this data would likely be **3 to 5**. The temperature curve across the year is smooth and seasonal, so a low-degree polynomial can capture the overall trend without overfitting the noise. Higher-degree polynomials might fit better on the training data but will likely overfit and perform worse on unseen data.



Based on bar graph, it's clear that: Summer months (July, August) have very low variance- temperatures are stable, And Winter to Spring (January–April) and Autumn (October–December) show higher variance, especially March and April.
Because of that, a model will not perform equally well across all months. The standard deviation of temperatures is much lower in the summer, meaning the temperature is more predictable. In contrast, spring and late winter months (like March and April) show high variability, making accurate predictions harder during those periods.

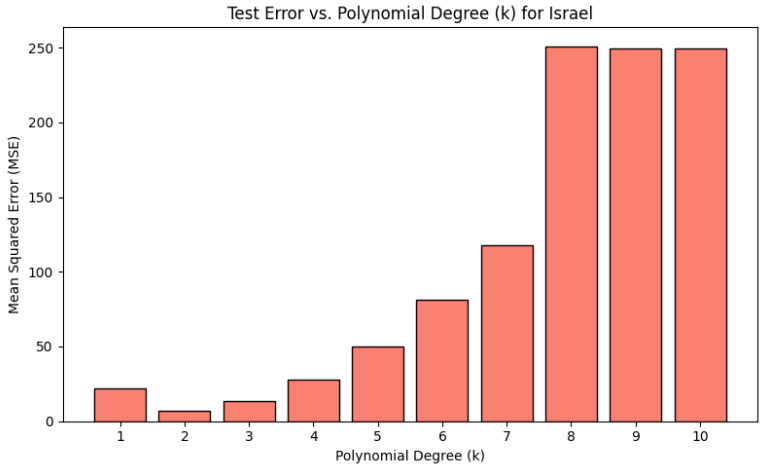


4. Based on the full dataset graph, it is not accurate to say all countries share a similar pattern. While most countries (Israel, Jordan, the Netherlands) follow a seasonal pattern (warmer in summer, colder in winter), other countries in a different climate can have a totally different pattern (South Africa for example). In addition, the amplitude and shape of the temperature curve pattern differ between countries due to geographic and climatic differences.



5. The best fitting model is the polynomial of degree 2, which achieves the lowest test error of 6.86. Since no other degree achieves a comparable or lower error, and degree 2 is relatively simple, it is the clear choice.

While degree 1 could be considered as a simpler alternative with a slightly higher error (22.19), higher degrees ($k \geq 4$) show rapidly increasing test error, suggesting overfitting. Therefore, the optimal choice is $k = 2$.



6. The plot shows how well a model trained on Israel's temperature data with a polynomial degree of k=2 performs on other countries. We see that: Jordan has the lowest error (MSE ≈ 26), which makes sense because it has a very similar seasonal pattern to Israel- geographically and climatically, the two countries are quite close. South Africa shows moderate error (MSE ≈ 79) while The Netherlands has the highest error (MSE ≈ 122).

This aligns with the insight from Question 4: while some countries (like Jordan) share a similar temperature pattern and are better predicted by a model trained on Israel, geographic and climatic differences lead to poor generalization in others (like South Africa and the Netherlands).