AIT203 Optimization Project – Question 1

Regression Modelling of Bike Sharing Demand

Team Members:

BT2024066, BT2024144, BT2024166

1 Introduction

This question focuses on building regression models to predict hourly bike rental demand using the Kaggle *Bike Sharing Demand* dataset. We work with the following model classes:

- Linear Regression
- Polynomial Regression of degrees 2, 3, and 4 (without interaction terms)
- Quadratic Regression with interaction terms (degree 2 only)

All models are implemented using the normal equation with NumPy, with the full implementation contained in the file `question1.py`. Preprocessing is performed carefully to avoid train/test leakage. The aim is to train on an 80% subset of the data, evaluate on the remaining 20%, and identify the best generalizing model using MSE and R^2.

2 Preprocessing and Leakage Prevention

2.1 Feature Engineering

The `datetime` column is decomposed into structured components:

$$\text{year},\ \text{month},\ \text{day},\ \text{hour},\ \text{dayofweek},$$

which help the model represent daily and seasonal demand patterns. The variables `casual` and `registered` are removed because:

$$\texttt{count} = \texttt{casual} + \texttt{registered},$$

making them direct sources of target leakage.

2.2 Train–Test Split

To prevent temporal leakage, the dataset is ordered by timestamp and partitioned into an 80% training set and a 20% testing set. This ensures that the model is trained only on past data and evaluated on future data, which aligns with standard forecasting practice.

Although random splitting can sometimes yield lower error metrics, the dataset is time-indexed and represents a forecasting problem. We therefore use a chronological 80/20 split so that training uses only past data and evaluation is performed on future observations. This avoids temporal leakage and provides a more realistic estimate of generalization performance.

2.3 Categorical Encoding

The categorical variables (`season`, `holiday`, `workingday`, `weather`) are one-hot encoded using category sets derived exclusively from the training data. Test data is encoded using the same basis, ensuring consistent dimensionality and no leakage.

2.4 Standardization of Numeric Features

Numeric features are standardized using training-set statistics:

$$x_{\text{std}} = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}},$$

so that the test data never influences preprocessing.

3 Model Construction

All regression coefficients are computed using the closed-form Normal Equation:

$$\hat{\beta} = (X^\top X)^{-1} X^\top y,$$

implemented in NumPy using a pseudoinverse for numerical stability.

3.1 Linear Regression

The baseline model uses standardized numeric features and one-hot encoded categorical variables.

3.2 Polynomial Regression (Degrees 2, 3, 4)

Polynomial models expand only the numeric features:

$$[X,\ X^2,\ \ldots,\ X^d],$$

without interactions. These expanded numeric features are combined with the categorical encodings to form the full design matrix.

3.3 Quadratic Model with Interaction Terms

For the degree-2 interaction model, we include:

- first-order numeric features,
- second-order (squared) features,
- pairwise products $x_i x_j$ for $i \leq j$,

in addition to the categorical one-hot variables. This substantially increases model flexibility but also variance.

4 Evaluation Metrics

4.1 Mean Squared Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

4.2 Coefficient of Determination (R^2)

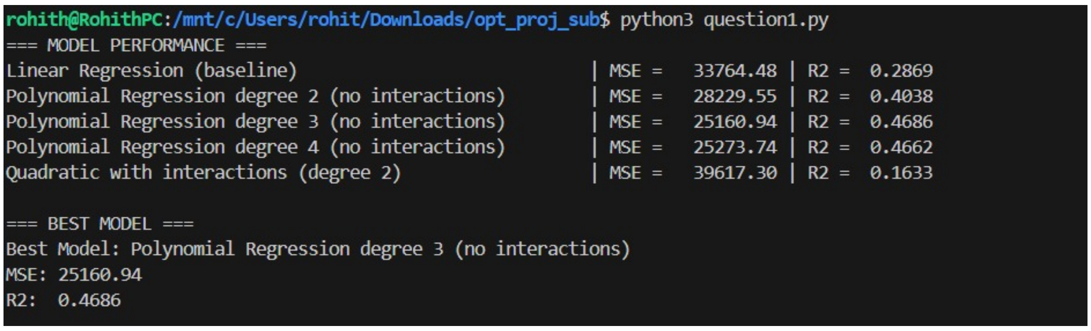
$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}.$$

Both metrics are computed using NumPy on predictions from each model.

5 Results

Model	Test MSE	Test R^2
Linear Regression (baseline)	33764.48	0.2869
Polynomial Regression (degree 2)	28229.55	0.4038
Polynomial Regression (degree 3)	**25160.94**	**0.4686**
Polynomial Regression (degree 4)	25273.74	0.4662
Quadratic w/ interactions (degree 2)	39617.30	0.1633

Terminal Output (from `question1.py`)



```
rohith@RohithPC:/mnt/c/Users/rohit/Downloads/opt_proj_sub$ python3 question1.py
=== MODEL PERFORMANCE ===
Linear Regression (baseline)                     | MSE =   33764.48 | R2 =  0.2869
Polynomial Regression degree 2 (no interactions) | MSE =   28229.55 | R2 =  0.4038
Polynomial Regression degree 3 (no interactions) | MSE =   25160.94 | R2 =  0.4686
Polynomial Regression degree 4 (no interactions) | MSE =   25273.74 | R2 =  0.4662
Quadratic with interactions (degree 2)           | MSE =   39617.30 | R2 =  0.1633

=== BEST MODEL ===
Best Model: Polynomial Regression degree 3 (no interactions)
MSE: 25160.94
R2:  0.4686
```

Figure 1: Summary of model performance printed by `question1.py`.

6 Discussion

The relative performance of the models can be explained using the bias–variance tradeoff and the ability of each model to capture curvature in the relationships.

- **Linear Regression (High Bias):** The model is too rigid to capture nonlinear temporal patterns such as morning/evening peaks and weather effects. High bias leads to underfitting and lower performance.
- **Polynomial Regression (Degree 2): Captures Curvature** Squared terms allow modelling of curved relationships (e.g., demand rising with temperature until a point and falling afterward). This reduces bias and improves MSE and R^2.

- **Polynomial Regression (Degree 3): Best Bias–Variance Tradeoff** Degree 3 introduces enough flexibility to model more complex curvature while keeping variance manageable. This balance results in the best generalization performance.
- **Polynomial Regression (Degree 4): Mild Overfitting** Increasing complexity further increases variance. Although the model fits the training set better, the test-set performance slightly declines.
- **Quadratic Model with Interactions: Very High Variance** The large number of interaction terms leads to multicollinearity and unstable coefficient estimates. Variance becomes too high, resulting in the poorest test performance.

7 Conclusion

We implemented all allowed regression models using NumPy and evaluated them after preprocessing the data carefully to avoid any train/test leakage. Among the permitted models, the **polynomial degree-3 regression (without interactions)** achieves the best test-set generalization, supported by the lowest MSE and highest R^2.

The results align with expected bias–variance and curvature behavior: moderately flexible non-linear models capture essential demand patterns effectively, while both less flexible (linear) and overly complex (interaction-heavy) models show weaker generalization performance.

8 Prediction Plots

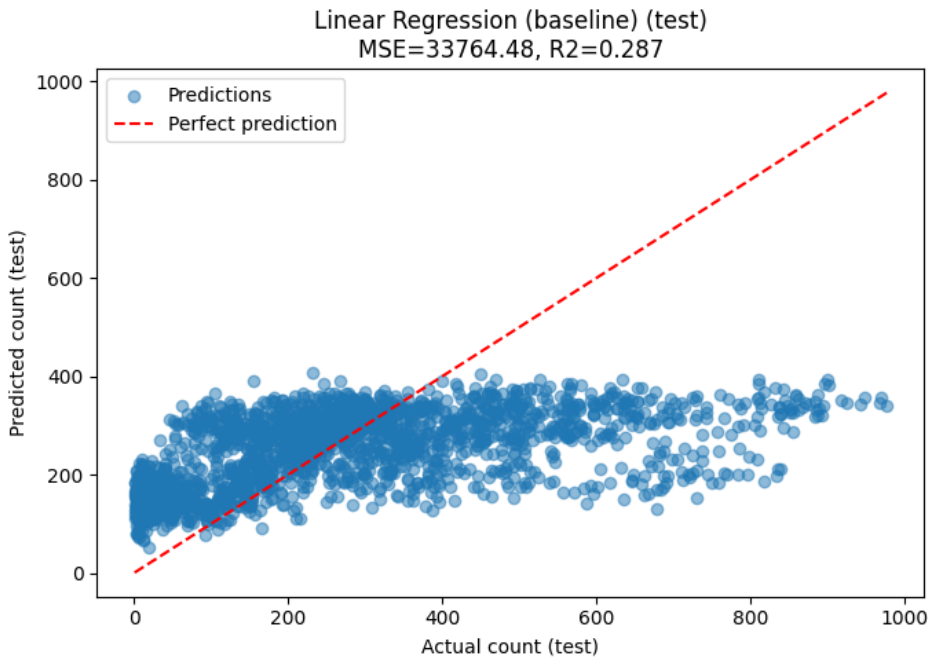


Figure 2: Linear Regression – Predicted vs. Actual

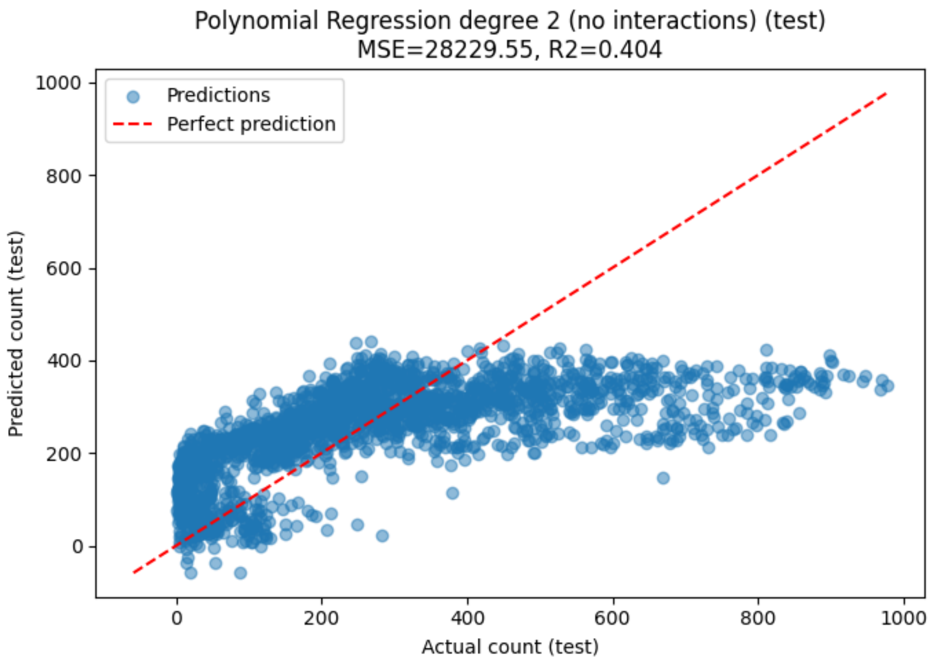


Figure 3: Polynomial Regression (Degree 2) – Predicted vs. Actual

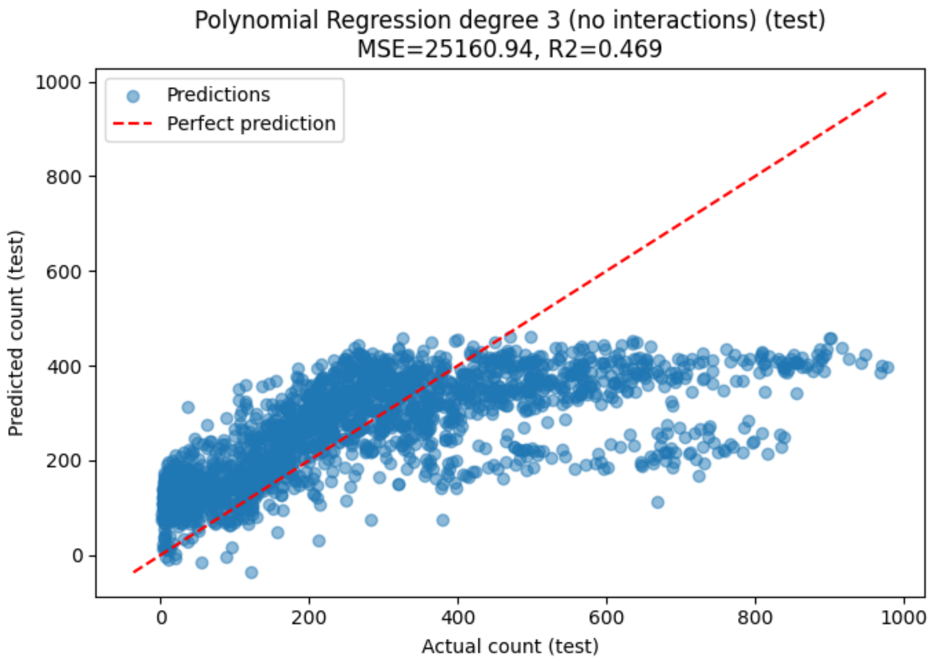


Figure 4: Polynomial Regression (Degree 3) – Predicted vs. Actual

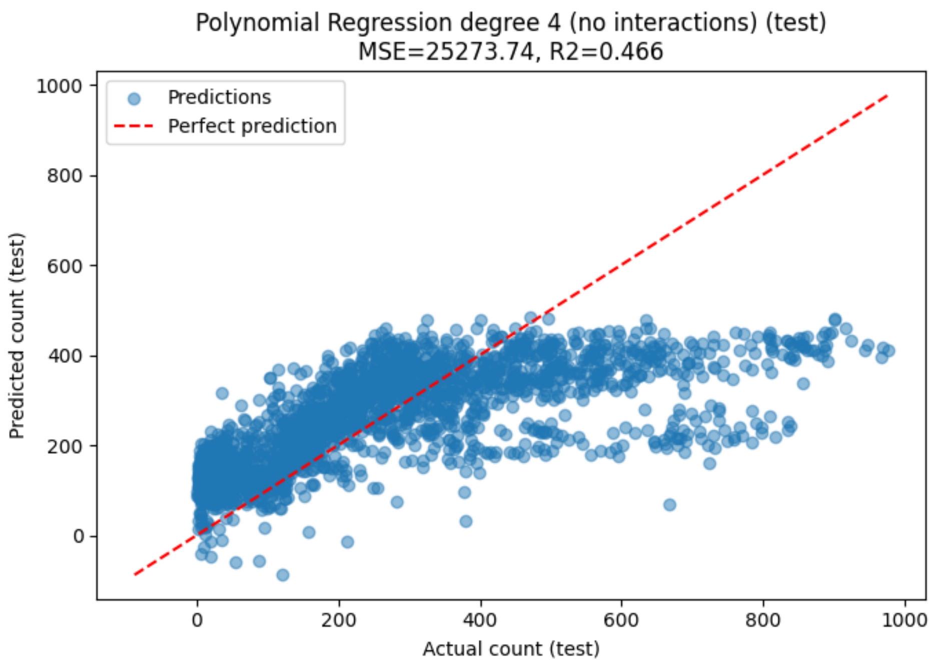


Figure 5: Polynomial Regression (Degree 4) – Predicted vs. Actual

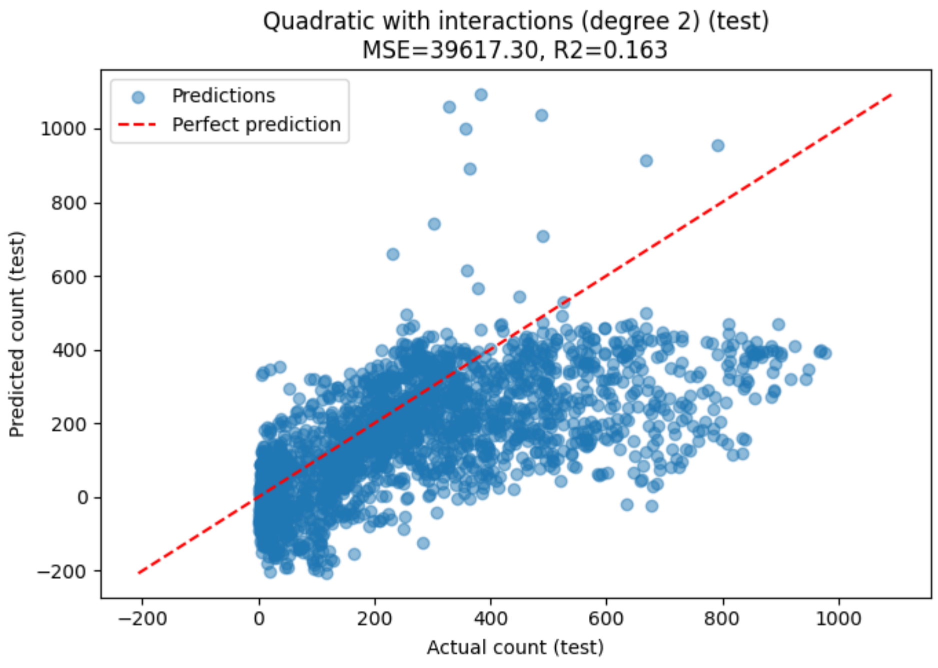


Figure 6: Quadratic Regression with Interactions – Predicted vs. Actual