Software Project Cost Estimation using Linear Regression Model on COCOMO81 Dataset

Google Colab Link:
https://colab.research.google.com/drive/15uYum9DkriGiH1sJrKrVPZR2xQFzcapF?usp=sharing

Presented by:

Saman Qayyum

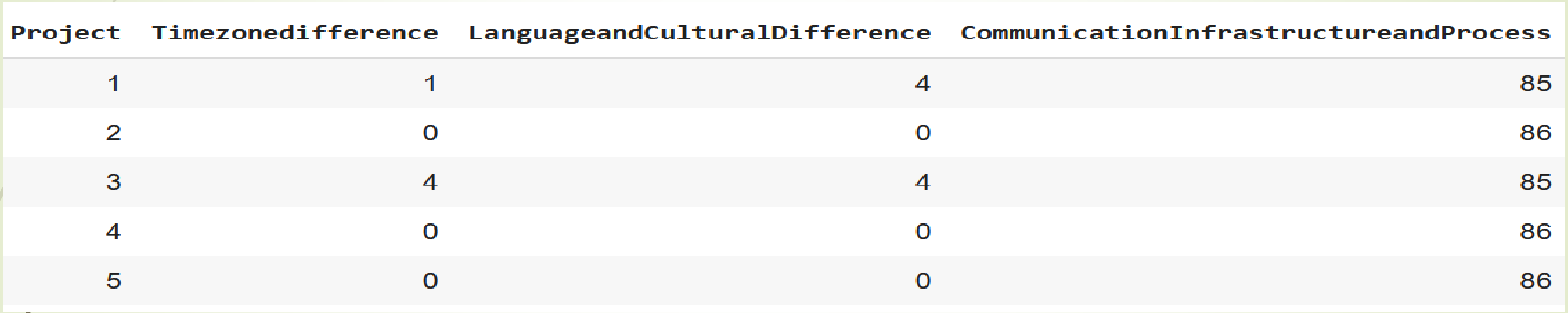
Student ID: 23010599

Problem Statement:

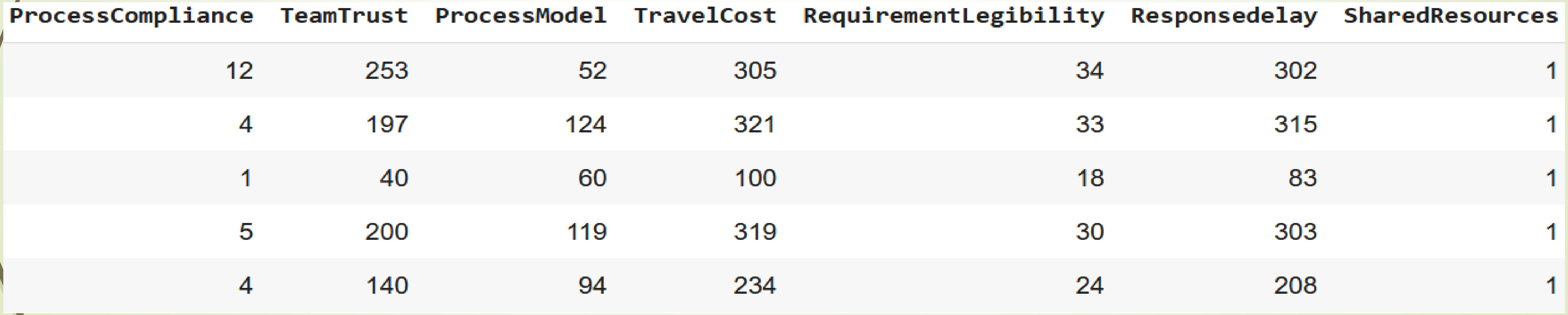
The COCOMO dataset offers valuable insights into software project estimation. This study aims to evaluate the accuracy of the COCOMO model in estimating effort, cost, and duration. By analyzing the dataset, we'll assess the model's performance in modern software development scenarios. Additionally, we'll identify factors that impact estimation accuracy, such as project size and complexity. Our goal is to provide actionable recommendations for improving estimation practices based on empirical evidence. This research will contribute to enhancing the reliability of software project planning and management. Through data-driven analysis, we'll uncover patterns and trends that influence estimation outcomes. Ultimately, this study seeks to bridge the gap between traditional estimation models and contemporary software development practices.

Dataset Description:

- We have downloaded a dataset from https://zenodo.org/records/268424
- Cocomo (Constructive Cost Model) is a recognized model for estimating software development effort, cost and duration. This dataset contains information about 81 projects data related to software engineering.



Project	Timezonedifference	LanguageandCulturalDifference	CommunicationInfrastructureandProcess
1	1	4	85
2	0	0	86
3	4	4	85
4	0	0	86
5	0	0	86



ProcessCompliance	TeamTrust	ProcessModel	TravelCost	RequirementLegibility	Responsedelay	SharedResources
12	253	52	305	34	302	1
4	197	124	321	33	315	1
1	40	60	100	18	83	1
5	200	119	319	30	303	1
4	140	94	234	24	208	1

Data Preprocessing:

- **Fitting and Transforming the Training Data**:

X_train_scaled = scaler.fit_transform(X_train) computes the mean and standard deviation of each feature in the training dataset X_train. Then, it uses these statistics to standardize the training data, resulting in X_train_scaled, where each feature now has a mean of 0 and a standard deviation of 1. The fit_transform method is a combination of fit and transform applied together.

- **Rescaling the Features:**

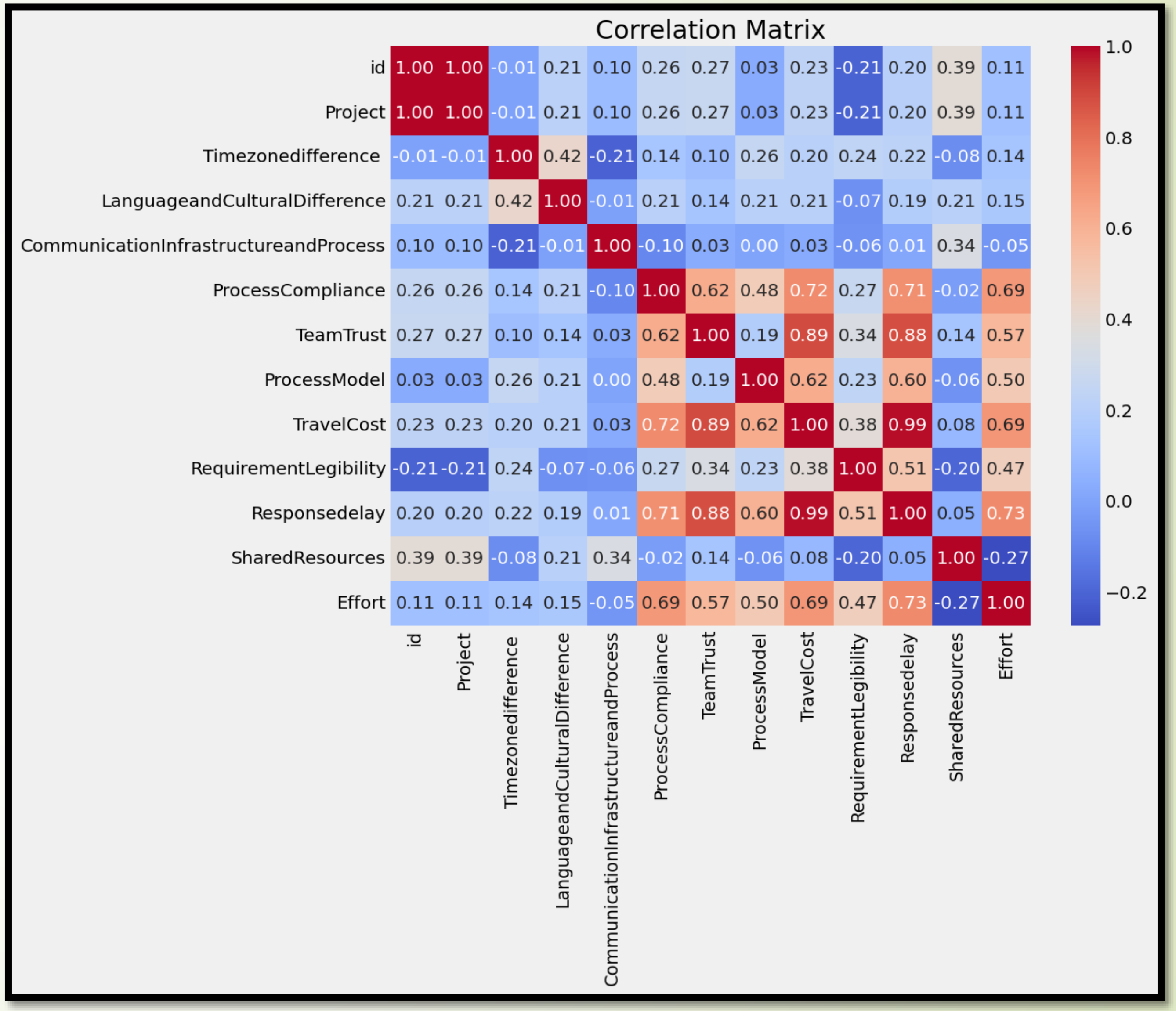
This scaler is used for standardization, a common preprocessing step in machine learning. Standardization involves rescaling the features of your data so that they have a mean of 0 and a standard deviation of 1.

- **Correlation Matrix:**

The correlation matrix contains a range of variables that seem to be from a dataset possibly related to project management or team performance metrics.

Correlation Matrix:

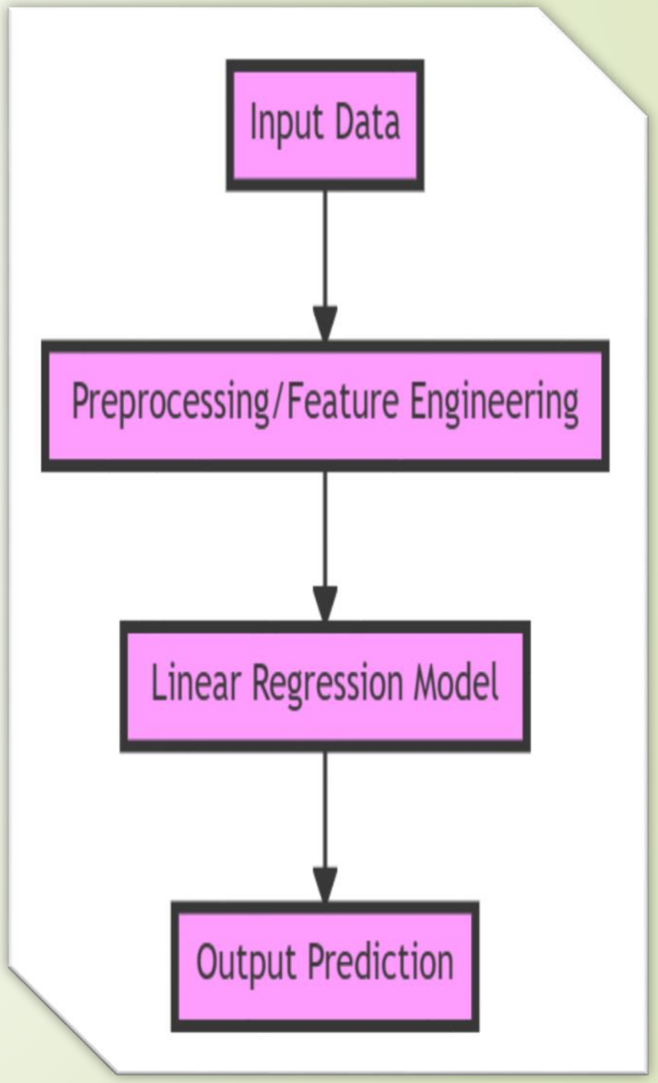
Each cell within the matrix represents the correlation coefficient between two variables. These coefficients range from -1 to 1, with 1 being a perfect positive correlation, -1 a perfect negative correlation, and 0 indicating no correlation.
The colors range from blue to red, with blue indicating a negative correlation, red a positive correlation, and white no correlation.



	id	Project	Timezonedifference	LanguageandCulturalDifference	CommunicationInfrastructureandProcess	ProcessCompliance	TeamTrust	ProcessModel	TravelCost	RequirementLegibility	Responsedelay	SharedResources	Effort
id	1.00	1.00	-0.01	0.21	0.10	0.26	0.27	0.03	0.23	-0.21	0.20	0.39	0.11
Project	1.00	1.00	-0.01	0.21	0.10	0.26	0.27	0.03	0.23	-0.21	0.20	0.39	0.11
Timezonedifference	-0.01	-0.01	1.00	0.42	-0.21	0.14	0.10	0.26	0.20	0.24	0.22	-0.08	0.14
LanguageandCulturalDifference	0.21	0.21	0.42	1.00	-0.01	0.21	0.14	0.21	0.21	-0.07	0.19	0.21	0.15
CommunicationInfrastructureandProcess	0.10	0.10	-0.21	-0.01	1.00	-0.10	0.03	0.00	0.03	-0.06	0.01	0.34	-0.05
ProcessCompliance	0.26	0.26	0.14	0.21	-0.10	1.00	0.62	0.48	0.72	0.27	0.71	-0.02	0.69
TeamTrust	0.27	0.27	0.10	0.14	0.03	0.62	1.00	0.19	0.89	0.34	0.88	0.14	0.57
ProcessModel	0.03	0.03	0.26	0.21	0.00	0.48	0.19	1.00	0.62	0.23	0.60	-0.06	0.50
TravelCost	0.23	0.23	0.20	0.21	0.03	0.72	0.89	0.62	1.00	0.38	0.99	0.08	0.69
RequirementLegibility	-0.21	-0.21	0.24	-0.07	-0.06	0.27	0.34	0.23	0.38	1.00	0.51	-0.20	0.47
Responsedelay	0.20	0.20	0.22	0.19	0.01	0.71	0.88	0.60	0.99	0.51	1.00	0.05	0.73
SharedResources	0.39	0.39	-0.08	0.21	0.34	-0.02	0.14	-0.06	0.08	-0.20	0.05	1.00	-0.27
Effort	0.11	0.11	0.14	0.15	-0.05	0.69	0.57	0.50	0.69	0.47	0.73	-0.27	1.00

Architecture of Linear Regression Model:

- **Input Data**: In the context of a Linear Regression model, this data often consists of one or more independent variables (predictors) and a dependent variable. We selected
- **Independent Variables:** Time zone difference, Language and Cultural Difference, Communication Infrastructure and Process, Process Compliance, Team Trust, Process Model, Travel Cost, Requirement Legibility, Response delay.
- **Dependent Variables:** Team Trust, Process Model, Travel Cost.
- **Preprocessing/Feature Engineering**: Before the data can be used to train a model, it usually needs to be cleaned and transformed. This step may involve handling missing values, encoding categorical variables, normalizing or standardizing numerical values, and potentially creating new features from the existing ones to better capture the underlying patterns in the data.
- **Linear Regression:** Linear Regression is a statistical method that models the relationship between a dependent variable and one or more independent variables by fitting a linear equation to observed data.
- **Output:** The output is the predicted value of the dependent variable based on the linear relationship learned by the model.

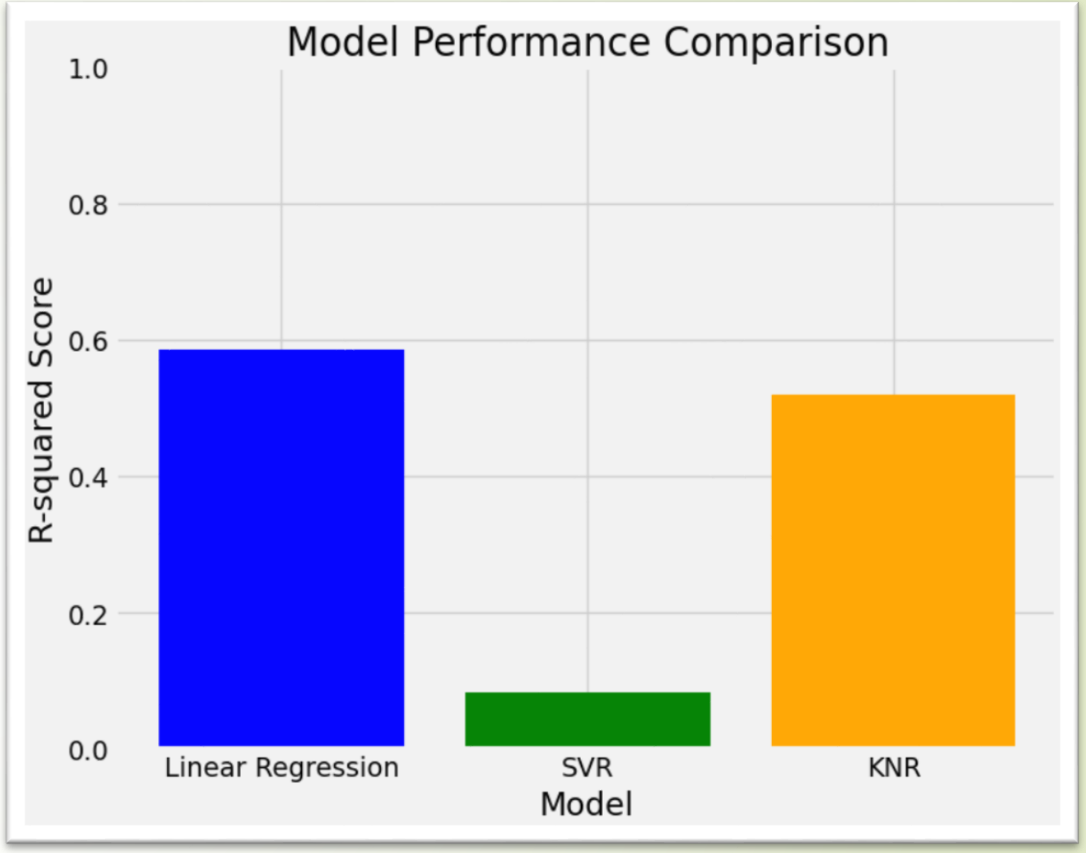


Parameters Used:

- Linear regression is a simple model that fits a linear relationship between the input features and the target variable without any tuning parameters.
- **kernel**: Specifies the kernel type to be used in the algorithm. It can be 'linear', 'poly', 'rbf', 'sigmoid', or other custom kernels. In this case, the code tests both 'linear' and 'rbf' kernels.
- **C**: Regularization parameter. The strength of the regularization is inversely proportional to C. Must be positive. Smaller values specify stronger regularization. The code tests various values of C from 1 to 10.
- neighbors=3 that is **k**=3

Results:
Model Performance Comparison:

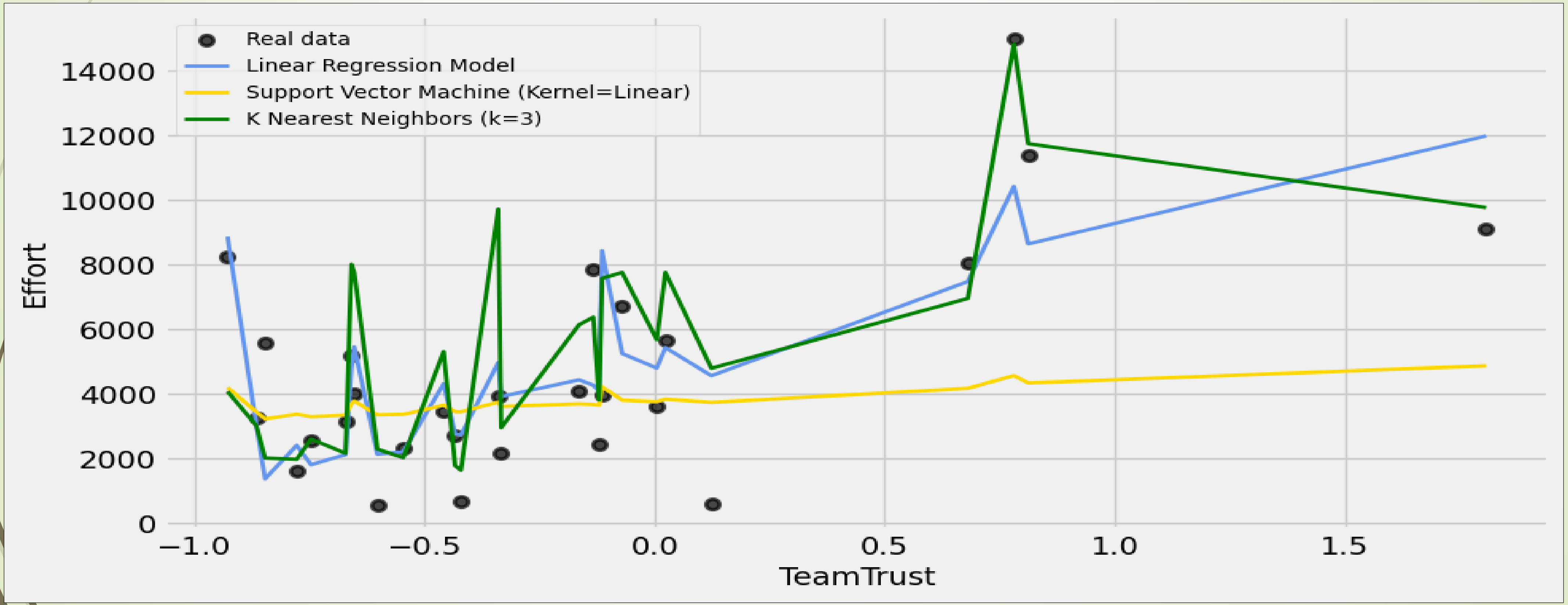
- **Linear Regression**: This model has an R-squared score of approximately 0.5, which is the highest among the three models. This indicates that about 50% of the variance in the dependent variable can be explained by the independent variables in the model. It is performing better than the other two models.
- **SVR (Support Vector Regression)**: The SVR model has an R-squared score close to 0, which is significantly lower than the Linear Regression model. This suggests that the SVR model does not explain the variance in the dependent variable well for the given data.
- **KNR (K-Nearest Neighbors Regression)**: This model has an R-squared score of approximately 0.75, which is higher than Linear Regression, indicating that the KNR model is able to explain 75% of the variance in the dependent variable.



(An R^2 score of 1 means that the model perfectly predicts the target variable, an R^2 score of 0 means that the model does no better than a model that simply predicts the mean of the target variable for all observations)

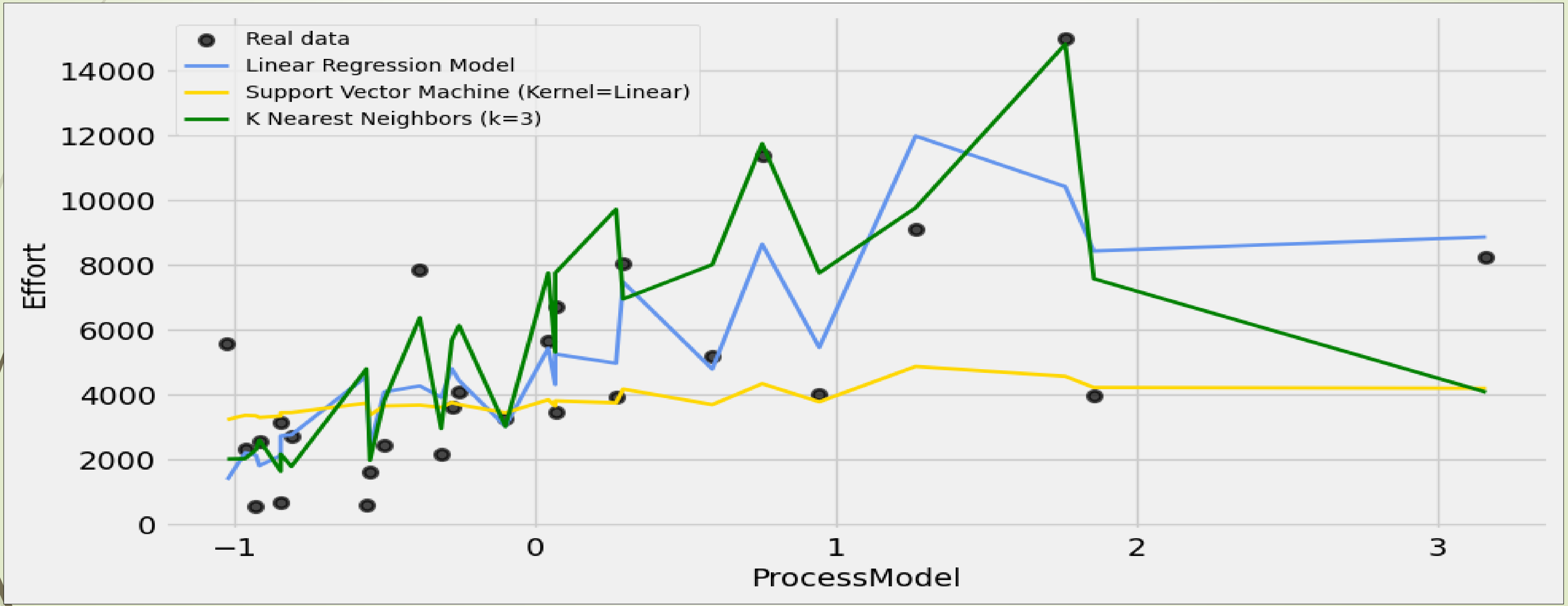
Results: Cost (Effort) Prediction: Effort Vs Team Trust

- The graph compares the performance of three models: Linear Regression, SVM with a linear kernel, and K Nearest Neighbors (k=3) against real data on "Team Trust" vs. "Effort". Linear Regression shows a general increasing trend. SVM's linear predictions are stable but fail to reflect data complexity, seemingly underfitting. KNN tracks actual data fluctuations more closely, indicating a better fit but with potential overfitting. Overall, We have observed that Linear Regression performed better than support vector regression and KNN in terms of finding cost (effort) of project when TeamTrust is a dependent variable.



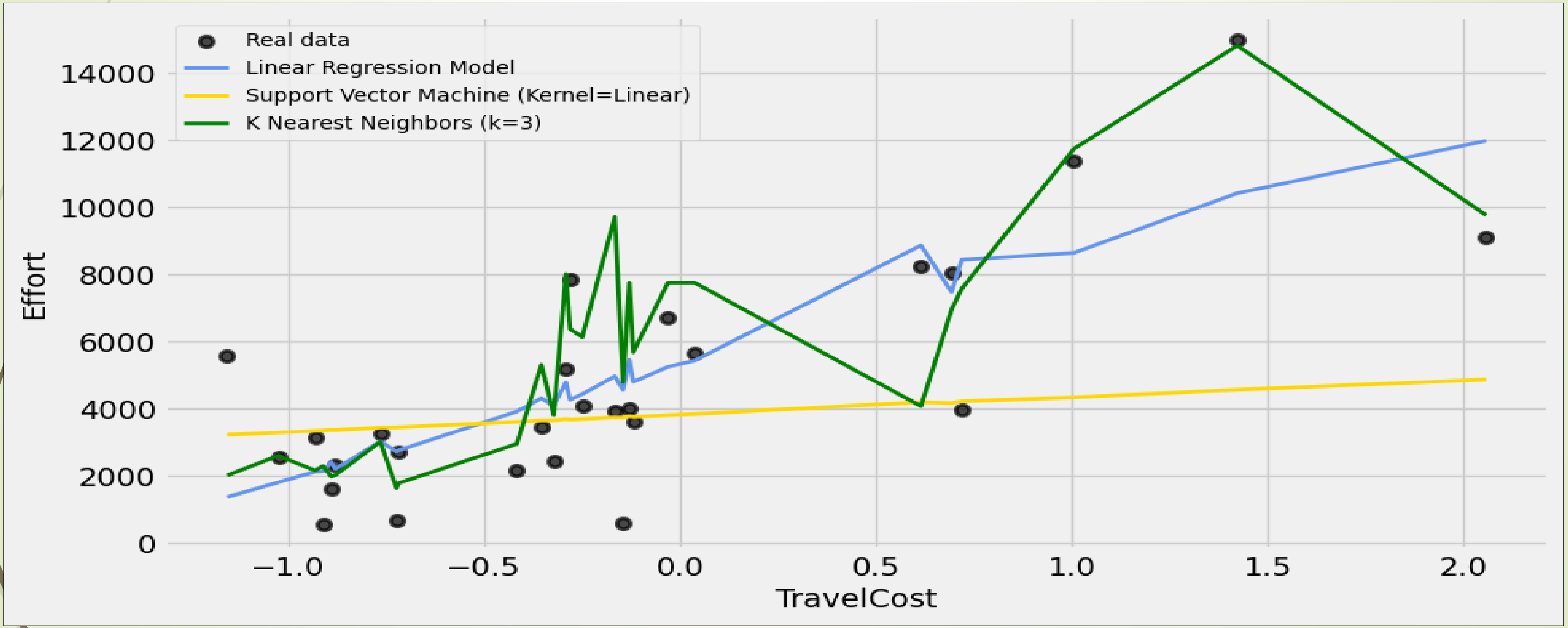
Results: Cost (Effort) Prediction: Effort Vs Process Model

- We have observed that Linear Regression performed better than support vector regression and KNN in terms of finding cost (effort) of project when ProcessModel is a dependent variable. Each model exhibits distinct performance characteristics, with LR showing the closest alignment with the real data points, followed by KNN, while SVM demonstrates the least alignment.



Results: Cost (Effort) Prediction: Effort Vs Travel Cost

- The line graph illustrates predictions for "Effort" as a function of "TravelCost" using three different predictive models compared to actual data. Linear Regression underestimates effort, showing a linear relationship. The SVM model's predictions are almost constant, indicating a poor fit. KNN closely follows the actual data, suggesting higher accuracy. So, KNN performed better than support vector regression and Linear Regression in terms of finding cost (effort) of project when TravelCost is a dependent variable.



Limitations & Improvements:

Limitations:

- **Linearity**: Linear regression assumes a linear relationship between the independent and dependent variables. This assumption does not hold for datasets where the relationship is nonlinear, leading to poor model performance.
- **Outliers**: Linear regression models are sensitive to outliers in the data. A few outliers can significantly affect the regression line and consequently the overall predictions.
- **Feature Selection**: The performance of a linear regression model is highly dependent on the choice of independent variables. Irrelevant or redundant features can degrade the model's performance.

Improvements:

- **Remove Irrelevant Features**: We can use domain knowledge or statistical tests to identify and remove features that do not significantly contribute to the prediction.
- **Dimensionality Reduction**: Techniques like Principal Component Analysis (PCA) can reduce the feature space to a smaller set of uncorrelated components, which might improve model performance.

References:

- Cristianini, N., & Shawe-Taylor, J. (2000). An introduction to support vector machines and other kernel-based learning methods.
- Kumari, K., & Yadav, S. (2018). Linear regression analysis study. Journal of the practice of Cardiovascular Sciences, 4(1), 33-36.
- Desharnais, J. M. (1988). Statistical analysis on the productivity of data processing with development projects using the function point technique. Université du Québec à Montréal.. Cambridge university press.