EXPERIMENT REPORT

Student Name	Hemang Sharma
Project Name	**Assignment 1 - Regression Models Part C**
Date	30th March 2023
Deliverables	< PartC_HemangSharma_24695785.ipynb> < reg> < poly>

---

1. EXPERIMENT BACKGROUND	
Provide information about the problem/project such as the scope, the overall objective, expectations. Lay down the goal of this experiment and what are the insights, answers you want to gain or level of performance you are expecting to reach.	
1.a. Business Objective	The goal of this project is to build a predictive model for the target variable, "TARGET_deathRate", which represents the cancer death rate per 100,000 population. The model will be used to identify the most important factors that contribute to cancer death rates and to predict the death rate for new, unseen data, such as for different counties or regions. The results of the project can be used by businesses, organizations, and government agencies to identify areas with higher cancer death rates and to develop targeted interventions to reduce cancer deaths. For example, businesses in the healthcare industry can use the model to identify areas with higher cancer death rates and develop targeted marketing campaigns for cancer screenings or treatments. Government agencies can use the model to identify areas with higher cancer death rates and allocate resources for cancer prevention and treatment programs. The model can also be used to identify the most important factors that contribute to cancer death rates, such as socioeconomic status, environmental factors, and healthcare access. This information can be used to inform public health policies and initiatives aimed at reducing cancer death rates. In summary, the goal of this project is to build a predictive model for cancer death rates and to use the model to inform targeted interventions and public health policies to reduce cancer deaths.

1.b. Hypothesis	**Hypothesis:** The hypothesis of this experiment is that applying polynomial feature engineering to the training and validation datasets can improve the accuracy of the linear regression model for predicting the target variable 'TARGET_deathRate' in cancer patients. **Question:** Can a multivariate linear regression model with polynomial feature engineering outperform multivariate linear regression model without polynomial feature engineering? **Insight:** The code first loads the training and testing datasets, then splits the training dataset into training and validation datasets. It then defines the features and target variable and extracts the features and target variable as numpy arrays for training and validation. It then applies polynomial feature engineering to the training and validation data with degrees ranging from 1 to 5, and trains a multivariate linear regression model using the training data with polynomial features. It makes predictions using the validation data with polynomial features, and calculates the mean squared error and root mean squared error for each degree. Finally, it plots the predicted values against the actual values for the validation and testing data, and calculates the mean squared error and root mean squared error for the testing data. **Reasons for considering it:** Polynomial feature engineering is a useful technique for improving the accuracy of linear regression models by creating new features as polynomial combinations of the original features. This can capture nonlinear relationships between the features and the target variable that a linear model cannot capture. Therefore, it is worthwhile to consider applying polynomial feature engineering to the training and validation data to improve the accuracy of the linear regression model for predicting the target variable 'TARGET_deathRate' in cancer patients. The mean squared error and root mean squared error calculated for each degree can help in selecting the degree that provides the best balance between bias and variance, and the plots can provide insights into the relationship between the actual and predicted values.
1.c. Experiment Objective	The goal of the experiment is to improve the performance of the multivariate linear regression model using feature engineering and polynomial regression. The expected outcome of the experiment is to obtain a model with lower mean squared error (MSE) and root mean squared error (RMSE) on both the validation and testing data compared to the original model, indicating that the model can make more accurate predictions. The learning curves should also show that the model is not underfitting or overfitting and that the performance on the validation set can no longer be improved significantly by adding more training examples.

2. EXPERIMENT DETAILS	
Elaborate on the approach taken for this experiment. List the different steps/techniques used and explain the rationale for choosing them.	
2.a. Data Preparation	The data preparation steps taken in this code include: 1. Loading the training and testing datasets: This step involves reading the data files into Pandas dataframes using the read_csv function from Pandas. This step is necessary to access the data and perform any necessary preprocessing steps. 2. Splitting the training dataset into training and validation datasets: This step involves dividing the training dataset into two subsets, one for training the model and the other for validating the model. This step is necessary to prevent overfitting of the model to the training data and to evaluate the performance of the model on unseen data. 3. Defining the features and target variable: This step involves selecting the features that will be used to train the model and the target variable that the model will predict. This step is necessary to identify the inputs and outputs of the model. 4. Extracting the features and target variable as numpy arrays: This step involves converting the Pandas dataframes to numpy arrays, which are the preferred data format for most machine learning algorithms. This step is necessary to ensure compatibility with Scikit-Learn functions that require numpy arrays as inputs.
2.b. Feature Engineering	In the first part of the code, no feature engineering is performed. The features used for the regression model are directly taken from the dataset. In the second part of the code, polynomial feature engineering is performed using the PolynomialFeatures class from scikit-learn. This generates additional features by taking combinations of the existing features up to a certain degree. The rationale for performing polynomial feature engineering is to capture non-linear relationships between the features and the target variable. By adding polynomial features, the model can better capture non-linear relationships between the features and the target variable, which can lead to improved performance. The degree of polynomial features is a hyperparameter that needs to be tuned to balance between overfitting and underfitting.

2.c. Modelling	The model trained for this experiment is a multivariate linear regression model with polynomial feature engineering using scikit-learn library. The polynomial features are added to capture any non-linear relationships between the features and the target variable. The model is trained on the training set and evaluated on the validation set using mean squared error (MSE) and root mean squared error (RMSE) metrics. The hyperparameters tuned in this experiment are the degrees of the polynomial features used in the model, ranging from 1 to 5. The reason for choosing this model is that it is a simple yet powerful technique for predicting a continuous target variable based on multiple features. It can capture the linear and non-linear relationships between the features and the target variable, and it is easy to interpret and explain. Additionally, the use of polynomial features can increase the model's performance by capturing any non-linear relationships between the features and the target variable. The hyperparameters tuned in this experiment are the degrees of the polynomial features used in the model. The degrees range from 1 to 5, and the performance of the model is evaluated on the validation set using mean squared error (MSE) and root mean squared error (RMSE) metrics. The hyperparameters are tuned to find the optimal degree of polynomial features that balances model complexity and performance.

3. EXPERIMENT RESULTS

Analyse in detail the results achieved from this experiment from a technical and business perspective. Not only report performance metrics results but also any interpretation on model features, incorrect results, risks identified.

3.a. Technical Performance

The performance metrics for the model are:

- Mean squared error (MSE) on validation data: 417.43
- Root mean squared error (RMSE) on validation data: 20.43
- Mean squared error (MSE) on testing data: 488.62
- Root mean squared error (RMSE) on testing data: 22.10

The validation RMSE is lower than the testing RMSE, which suggests that the model may be overfitting to the training data. This is confirmed by the fact that the training set RMSE is higher than the validation set RMSE, indicating underfitting.

To address underfitting, the code performs polynomial feature engineering and evaluates the performance of a linear regression model with different degrees of polynomial features. The results show that the model with degree 2 polynomial features performs best on the validation set, with an RMSE of 20.32. However, the model with degree 5 polynomial features has a very high RMSE on the validation set, suggesting that it is overfitting to the training data.

The learning curves show that the model is overfitting when trained on a small amount of data, but the performance improves as more data is used for training. However, the validation score plateaus at around 0.70, indicating that the model's performance can no longer be improved significantly by adding more training examples.

The main underperforming cases/observations are not specified in the provided information. However, it is clear that the model is underfitting and that the degree of polynomial features should be increased to improve its performance. On the other hand, too high degree of polynomial features leads to overfitting, and this can be seen in the performance of the model with degree 5 polynomial features. Another potential root cause of underperformance could be the presence of outliers or non-linear relationships that are not captured by the linear regression model.

3.b. Business Impact

The business objective set earlier was to predict the death rate due to cancer for different counties in the USA. The experiments involved using multivariate linear regression with feature engineering using the scikit-learn library.

The initial model was found to be underfitting, which means that it was not capturing the complex relationship between the target variable and the features. To address this issue, polynomial feature engineering was applied to generate more complex features. The performance of the model was evaluated using mean squared error (MSE) and root mean squared error (RMSE) on the validation and testing datasets. The degree of polynomial features that resulted in the lowest MSE and RMSE on the validation dataset was chosen as the optimal degree for the model. The performance of the model was also visualized using scatter plots and learning curves.

Based on the results, the model with polynomial features of degree 2 resulted in the lowest MSE and RMSE on the validation dataset. However, the performance of the model decreased significantly for higher degrees, indicating that overfitting occurred. Therefore, the model with polynomial features of degree 2 should be selected for making predictions on new, unseen data.

	The impact of incorrect results for this business objective could be severe, as accurate predictions of death rates due to cancer are important for healthcare planning and resource allocation. Inaccurate predictions could lead to misallocation of resources and inadequate provision of healthcare services in some areas, which could result in lower survival rates and higher death rates due to cancer. Therefore, it is important to use accurate models for predicting death rates due to cancer to ensure that healthcare services are provided efficiently and effectively.
3.c. Encountered Issues	Here are some issues that have arisen during the experiments: 1. Underfitting: The initial model was underfitting, which means it was not able to capture the complexity of the data. To solve this, you applied polynomial feature engineering to increase the complexity of the model. 2. Overfitting: Polynomial feature engineering can lead to overfitting, where the model becomes too complex and starts to fit the noise in the data. To diagnose overfitting, you used learning curves, which showed that the model was overfitting when trained on a small amount of data. To solve this, you increased the size of the training dataset. 3. Diminishing returns: The first graph of the performance of the linear regression model with polynomial features of different degrees on the validation dataset showed that the improvement in the model's performance diminishes as the degree increases beyond 3. This is a common problem with polynomial regression models, and one solution is to use other types of regression models, such as regularization. 4. High error: Even after applying polynomial feature engineering, the model's error on the testing dataset was still high. This could be due to a number of factors, such as missing or noisy data, incorrect assumptions about the relationship between the features and the target variable, or using an inappropriate model. To solve this, you can try different types of regression models or improve the quality of the data by cleaning, imputing, or removing missing or noisy data. You can also try feature selection or engineering to identify the most relevant features for the model.

---

4. FUTURE EXPERIMENT	
Reflect on the experiment and highlight the key information/insights you gained from it that are valuable for the overall project objectives from a technical and business perspective.	
4.a. Key Learning	From the experiment, we can see that the initial linear regression model is underfitting, as the MSE and RMSE are high on both the validation and testing data. This indicates that the model is not able to capture the underlying relationship between the features and the target variable. To address this issue, we performed feature engineering using polynomial features of different degrees. The results show that the performance of the model improves as the degree of polynomial features increases up to 3. However, beyond degree 3, the model's performance starts to deteriorate, indicating overfitting. We also generated learning curves to visualize the performance of the model on the

	training and validation sets as a function of the number of training examples. The learning curves show that the model is overfitting when trained on a small amount of data, but the performance improves as more data is used for training. However, the validation score plateaus at around 0.70, indicating that the model's performance can no longer be improved significantly by adding more training examples. Based on these results, we can conclude that the initial linear regression model was too simple to capture the underlying relationship between the features and the target variable, and feature engineering using polynomial features of degree up to 3 improved the model's performance. However, further experimentation is needed to determine if other models, such as decision trees or neural networks, can outperform this linear regression model on this dataset. Additionally, other feature engineering techniques, such as feature selection or transformation, could also be explored to further improve the model's performance.
4.b. Suggestions / Recommendations	Potential next steps and experiments could include: 1. Feature selection: Analyze the features used in the model and eliminate any that may not be relevant or redundant. This could improve the model's performance and reduce overfitting. 2. Regularization: Add regularization to the linear regression model to prevent overfitting and improve generalization to new data. Ridge or Lasso regression could be used for this purpose. 3. Non-linear models: Explore non-linear models such as decision trees, random forests, or support vector machines (SVMs). These models may be able to capture non-linear relationships between features and the target variable and improve the model's performance. 4. Ensemble methods: Use ensemble methods such as bagging, boosting, or stacking to combine multiple models and improve the overall performance of the model. 5. Hyperparameter tuning: Tune the hyperparameters of the models to improve their performance. Grid search or random search could be used to find the best hyperparameters for the models. 6. Cross-validation: Use cross-validation to evaluate the performance of the models on different subsets of the data and ensure that the results are reliable and reproducible. 7. Data preprocessing: Explore different data preprocessing techniques such as normalization, scaling, or imputation to improve the quality of the data and the performance of the models. 8. Outlier detection: Analyze the data for outliers that may be affecting the model's performance and eliminate them if necessary. 9. Data augmentation: Use data augmentation techniques such as oversampling or undersampling to balance the data and improve the performance of the models. 10. Domain-specific feature engineering: Investigate domain-specific features that may be relevant to the target variable and add them to the model. For example, if the target variable is related to healthcare, adding features such as hospital density or healthcare spending could improve the model's performance.