Introduction to Data Science

GROUP 14

Name	UH ID
Md. Moshiur Rahman	**2400991**
Divya Sai Sree Pilli	**2404251**
Jayaprakash Yadav Guntumani	**2391724**
Tanushree Mukherjee	**2413888**

Step 1: Overview of the Dataset:

Dataset Rationale: The dataset chosen for this project is focused on the placement of Wave Energy Converters (WECs) near Adelaide, Australia. It includes detailed information on the positions of 16 WECs and their corresponding absorbed power, as well as the total energy output of the system. The primary reason for selecting this dataset is its relevance to the optimization of renewable energy systems, specifically in wave energy. By applying data science techniques, we can better understand how different WEC configurations influence energy production. This kind of analysis can help drive improvements in energy efficiency and sustainability, making it highly relevant for real-world applications in renewable energy.

Target Variable: The key variable we aim to predict in this dataset is the total power output (Powerall). This variable reflects the sum of the energy absorbed by all 16 WECs in the array, and it is expressed as a continuous numerical value. The goal of this project is to build models that can accurately predict the total power output based on the spatial configuration of the WECs, making this a typical regression problem.

Problem Type: Since the total power output is a continuous variable, this is a regression problem. Our task is to develop models that can forecast the total energy output of the WEC array based on various input features, such as WEC positions and absorbed power. Understanding and optimizing these configurations can provide valuable insights into improving the efficiency of renewable energy systems.

Potential Use Case: The findings from this dataset could be applied to real-world wave farm designs, helping optimize WEC placement to increase overall energy output. The insights could also be used to

guide decision-making in the deployment of wave energy systems in similar coastal environments worldwide.

Step 2: Data Preprocessing and Cleaning

Handling Missing Values: We ensure data completeness by analyzing the dataset for missing values across all features. Missing values in the WEC position coordinates and absorbed power columns were handled using mean imputation. This approach was selected as it maintains the dataset's overall distribution and consistency, avoiding bias. Mean imputation provided a reliable way to fill missing data while preserving dataset integrity.

Analyzing and Handling Zero Values: We conducted a detailed analysis of zero values in the WEC position coordinates and absorbed power columns. These zeros were evaluated to determine if they represented inactive WECs or indicated missing or erroneous data. After analysis, we chose to treat these zeros as missing values and imputed them using KNN imputation. This method uses spatial patterns from neighboring data points, ensuring that imputed values align with the data's geometric structure.

Outlier Detection and Handling: Outliers were identified in the WEC position and absorbed power columns using the Interquartile Range (IQR) method. Handling of outliers was based on their proportion within the dataset:

If outliers comprised less than 5% of the dataset, they were removed.

If they exceeded 5%, we replaced them with the mean of the first (Q1) and third quartiles (Q3), preserving the range and minimizing the effect of extreme values.

Feature Scaling: Feature scaling was implemented to standardize the WEC positions and absorbed power columns. Standardization was used to achieve zero mean and unit variance, making the data more suitable for regression models. This step ensures that models treat all features equally, preventing features with larger ranges from dominating model performance.

Additional Transformations: We analyzed skewness in the power columns. Only features with skewness beyond ±0.5 were transformed using the Yeo-Johnson transformation, which accommodates both positive and negative values. This transformation helped stabilize variances, ensuring the data distribution is more normal and supporting model performance.

Final Preprocessing Check: After completing all preprocessing and cleaning steps, the final cleaned and scaled dataset was saved for further analysis. This refined dataset is now ready for use as a reliable foundation for model training and future analysis steps.

Step 3: Model Building

Model Selection: In this project, we have selected 5 different models and compared their performance based on training and test data using each of the following models. Here, we also used the Evaluation Metrics for each model such as MSE and R-square score.

Model 1: Linear Regression

In this model, we have split our data into 80% Training and 20% Test sets. The goal of this model is to fit a linear relationship between the features and the target variables. Using this model, we could achieve a significantly good result of Mean Squared Error (MSE) of around 45,824,671.83 and quite a desired and higher R-square score of 0.985, indicating a high level of accuracy in the prediction of the output.

Model 2: KNN

We implemented this model using 10 neighbors, which is using the average of the 10 nearest data points to predict the target output. This model, however, results in a bit higher MSE of around 409,645,796.89 with an optimal R-square score of 0.868, showing that it was less accurate compared to other models.

Model 3: Random Forest Regressor

This ensemble method utilized almost 100 decision trees to predict the target variable by taking the average of their individual outputs. This Random Forest model achieves a better MSE value of around 409,645,796.89 with an averagely optimum R-square score of around 0.868, displaying similar performance as that of KNN model but with the reduced overfitting issues due to its ensemble nature.

Model 4: SVM Regressor

In this model, we used both the linear and non-linear kernels to train and test the dataset.

Linear Kernel:

This variant of SVM achieves significantly lower MSE of around 87,269,463.53 and much better Rsquare score of 0.972, which makes it one of the better-performing models.

RBF Kernel (Non-Linear):

On the other hand, this variant of SVM produced a significantly higher MSE of around

3,099,615,920.30 with an unexpectedly lower R-square score of almost 0.003, indicating its poor performance on this specific dataset, which might be because of the high dimensionality and complexity of the data.

Model 5: XGBoost

The XGBoost model, is mostly known for its gradient boosting approach. This model is being trained with around 100 estimators. It gave a very balanced performance with much lower MSE of around 89,857,382.95 and quite higher R-square score of around 0.971, making it equally efficient as the Linear Kernel SVM; however, with faster computation speed and better handling of overfitting issues.

Step 4.1: Model Performance Comparison:

Model	Mean Squared Error (MSE)	R2 Score
Linear Regression	45,824,671.83	0.9853
K-Nearest Neighbors	409,645,796.89	0.8683
SVM RBF Kernel	3,099,615,920.30	0.0032
SVM (Linear)	87,269,463.53	0.972
XGBoost	89,857,382.95	0.9711

Conclusion based on Model Comparison: Among all the models, we tested so far, Linear Regression achieved the best performance with the lowest MSE value of around 45,824,671.83 and highest Rsquare score of 0.985, showing its strong predictive accuracy for this specific dataset.

Concurrently, XGBoost and SVM with Linear Kernel also showed pretty well and almost parallel performance, giving the higher R-square score of 0.971 and 0.972 respectively, while keeping the MSE values subsequently low.

However, on the other side, KNN and Random Forest showed moderate performance with optimal Rsquare scores but higher MSE values, proving to be the less desirable models compared to the other two mentioned above.

While SVM with RBF Kernel, performed unexpectedly poor with extremely high MSE and almost near to zero R-square score, showing that it struggled with capturing the underlying data patterns, proving it to be the most undesired model for this specific dataset.

Overall, we can conclude that Linear Regression is the most suitable model for this dataset, balancing both simplicity and accuracy of the data.

Step 4.2: Feature Selection

Lasso Regression for Feature Selection

Lasso Regression (Least Absolute Shrinkage and Selection Operator) is a widely used technique for feature selection and dimensionality reduction. Its effectiveness stems from the following advantages:

1. Automatic Feature Selection

- Lasso incorporates L1 regularization, which penalizes the absolute magnitudes of feature coefficients. This penalty forces some coefficients to shrink exactly to zero.
- Features with zero coefficients are effectively removed from the model, enabling Lasso to automatically select the most important predictors while excluding irrelevant ones.

2. Handles Multicollinearity

- In cases where features are highly correlated (multicollinearity), Lasso can select one representative feature from the group, thereby reducing redundancy and ensuring a more stable model.

3. Efficiency and Simplicity

- Unlike iterative wrapper methods, which require building multiple models to evaluate feature importance, Lasso achieves feature selection in a single optimization step. This makes it computationally efficient and well-suited for high-dimensional datasets.

4. Enhanced Interpretability

- By reducing the number of predictors, Lasso simplifies the model. The retained features and their coefficients provide valuable insights into their influence on the target variable, improving interpretability.

Why Lasso Regression Was Chosen

- The dataset used for this project includes many features, such as positions and power outputs for multiple Wave Energy Converters (WECs).
- Not all features contribute equally to predicting the target variable, Powerall.
- Lasso helps identify and retain only the most relevant features, improving model performance and interpretability while minimizing overfitting.

Key Findings After Feature Selection

The analysis following feature selection reveals the following insights into model performance:

1. Linear Regression and SVM (Linear Kernel)

- These models emerged as the top performers, achieving near-perfect R^2 scores (0.9999) and exceptionally low RMSE values (370.01 on the training set and 230.19 on the test set).
- Their outstanding performance highlights their suitability for this dataset, likely due to the strong linear relationships present in the features.

2. K-Nearest Neighbors (KNN)

- Delivered moderate performance with a test R^2 of 0.9143 and significantly higher RMSE values (16,353 on the test set).
- While it shows promise, its accuracy and error rates lag behind the top models.

3. Random Forest and Gradient Boosting

- These models provided reasonable results but were less accurate compared to the best performers.
- The test R^2 was relatively lower at 0.8309, accompanied by higher RMSE values, indicating room for improvement.

4. SVM (RBF Kernel)

- Showed decent predictive performance with a test R^2 of 0.9038 but fell short compared to the linear models.
- While effective, it was outperformed by models that leveraged the dataset's linear nature.
- Linear Regression and SVM with a linear kernel were identified as the most effective models for this dataset, delivering exceptional accuracy and low error rates. Future efforts should focus on further optimizing these models to enhance their predictive power and robustness.

Step 4.3: Hyperparameter Tuning:

Randomized Search Cross-Validation was performed to optimize the hyperparameters of the models used in this project. The process involved sampling 10 random hyperparameter combinations for each model and evaluating them using 3-fold cross-validation with negative mean squared error as the scoring metric. Linear Regression was excluded as it has no tunable hyperparameters.

Best Parameters:

1. **KNN**: weights='distance', n_neighbors=10
2. **Random Forest**: n_estimators=200, max_depth=None
3. **SVM (Linear Kernel):** C=10
4. **SVM (RBF Kernel)**: gamma='scale', C=10
5. **Gradient Boosting**: n_estimators=200, learning_rate=0.2

These optimized hyperparameters are expected to enhance the models' predictive performance for the regression task.

Once Hyperparameter Tuning is done, we are trying to refine the prediction of total power output (**Powerall**) of Wave Energy Converters (WECs) by:

1. **Selecting significant features** using bi-directional elimination.
2. **Building and evaluating multiple regression models** on the reduced dataset.
3. Identifying the **best-performing model** based on training and testing performance metrics.
4. Validating the model on a **new dataset** to ensure generalization.

Step 4.4: Feature Selection Using Bi-Directional Elimination

- **Purpose:** To identify the most significant features for predicting Powerall, reducing noise and improving model performance.
- **Method Used:** A **wrapper method**, Bi-Directional Elimination, was implemented using SequentialFeatureSelector from sklearn.feature_selection.

This method iteratively adds or removes features based on their contribution to the model's performance, evaluated using R^2 score.

Key Steps in the Code:

1. **Initialize the Sequential Feature Selector:**
   - LinearRegression was used as the estimator.
   - Parameters: n_features_to_select = "auto": Automatically selects the optimal number of features.
   - direction = "forward": Features are added sequentially.
   - scoring = "r2": R^2 score was used as the evaluation metric.
   - cv = 3: Cross-validation with 3 folds ensured robustness.
2. **Fit the selector on scaled training data** to identify significant features.
3. **Output the selected features:**
   - The selected features included Y3, Y4, X7, X12, Y12, …, P16.

Outcome:

A reduced dataset was created, containing only the selected features for training and testing.

Step 4.5: Building and Evaluating Regression Models

- **Purpose:** To train multiple regression models on the reduced dataset and evaluate their performance based on training and testing metrics.
- **Models Evaluated:**
- **Linear Regression**
- **K-Nearest Neighbors (KNN)**
- **Random Forest**
- **Support Vector Machine (SVM)** with linear and RBF kernels
- **Gradient Boosting**

Key Steps in the Code:

1. **Initialize each model** with specific parameters (e.g., n_neighbors=5 for KNN, n_estimators=50 for Random Forest).
2. **Train the models** using the reduced training dataset.

3. **Make predictions** for both training and testing datasets.

4. **Evaluate each model** using:

- **Root Mean Squared Error (RMSE):** Measures the average prediction error in the same units as the target variable.
- **R^2 Score:** Indicates the proportion of variance explained by the model.

Model Results:

1. **Linear Regression:**

- **Train RMSE:** 369.96, **Test RMSE:** 230.30
- **Train R^2:** 0.999955, **Test R^2:** 0.999983
- **Insights:** Near-perfect performance, confirming the dataset's strong linear structure.

2. **K-Nearest Neighbors (KNN):**

- **Parameters:** n_neighbors=5
- **Train RMSE:** 14,996.46, **Test RMSE:** 18,327.09
- **Train R^2:** 0.926, **Test R^2:** 0.892
- **Insights:** Moderate performance due to its reliance on local relationships.

3. **Random Forest:**

- **Parameters:** n_estimators=50, max_depth=10
- **Train RMSE:** 20,847.30, **Test RMSE:** 23,107.04
- **Train R^2:** 0.857, **Test R^2:** 0.829
- **Insights:** Reasonable performance but exhibited slight overfitting.

4. **Support Vector Machine (SVM):**

- **Linear Kernel:**
- **Train RMSE:** 370.07, **Test RMSE:** 229.97
- **Train R^2:** 0.999955, **Test R^2:** 0.999983
- **Insights:** Matched Linear Regression in accuracy.
- **RBF Kernel:**
- **Parameters:** C=100, gamma=0.1
- **Train RMSE:** 25,315.52, **Test RMSE:** 25,562.87

- **Train R^2:** 0.790, **Test R^2:** 0.791
- **Insights:** Underperformed due to its sensitivity to hyperparameters.

5. **Gradient Boosting:**

- **Parameters:** n_estimators=50, max_depth=3
- **Train RMSE:** 18,310.37, **Test RMSE:** 18,843.29
- **Train R^2:** 0.890, **Test R^2:** 0.886
- **Insights:** Strong performance but less effective than simpler models like Linear Regression.

We have also implemented some additional Models to validate the accuracy and alignment of our dataset.

1. XGBoost

- Implemented XGBRegressor with 100 estimators and evaluated it.
- **Performance:**
- **Train RMSE:** 8,587.59, **Test RMSE:** 9,687.17
- **Train R^2:** 0.976, **Test R^2:** 0.970
- **Insights:** Efficiently captured complex patterns, performing better than Gradient Boosting.

2. Extreme Learning Machine (ELM)

- Used a single-layer feed-forward neural network with 50 neurons.
- **Performance:**
- **Train RMSE:** 90,037.30, **Test RMSE:** 90,963.08
- **Train R^2:** -1.658, **Test R^2:** -1.651
- **Insights:** Failed to capture relationships, resulting in poor performance.

3. Neural Network

- Trained a neural network with 64 hidden units and ReLU activation.
- **Performance:**
- **Train RMSE:** 189,440.63, **Test RMSE:** 190,418.75

- **Train R^2:** -10.77, **Test R^2:** -10.62
- **Insights:** Underperformed due to architecture or training issues.

4. Ensemble Model

- Combined predictions from Linear Regression, SVM (Linear Kernel), and XGBoost using VotingRegressor.
- **Performance:**
- **Train RMSE:** 2,882.75, **Test RMSE:** 3,238.25
- **Train R^2:** 0.997, **Test R^2:** 0.997
- **Insights:** Outperformed all other models, demonstrating the power of combining strengths.

Step 4.6: Deployment and New Data Evaluation

- The best-performing model (**SVM Linear Kernel**) was saved and tested on a new dataset from Tasmania.
- **Results:**
- **RMSE:** 19,177.67, **R^2:** 1.000
- **Insights:** The model generalized perfectly to the new data, explaining all variance in Powerall.

Conclusion

The steps taken in this document demonstrate:

1. **Effective feature selection** using Bi-Directional Elimination to focus on the most impactful predictors.
2. **Rigorous model evaluation** with multiple regression techniques.
3. Deployment of the **SVM Linear Kernel**, achieving robust and generalizable predictions on unseen data.

The project involved collaborative efforts from all team members, each contributing their expertise to ensure the successful completion of various phases of the machine learning workflow. Below is a detailed breakdown of each member's responsibilities:

Team Member	Responsibilities
Md Moshiur Rahman	• **Data Cleaning**: Organized and preprocessed raw datasets to prepare them for modeling. • **Modeling**: Built and trained six different machine learning models. • **Model Evaluation**: Assessed model performance using various metrics to determine their effectiveness. • **Project Documentation:** Prepared a comprehensive report summarizing of the project till project proposal to Model Evaluation.
Jayaprakash Yadav Guntumani	• **Hyperparameter Tuning**: Optimized model parameters to improve performance. • **Running All Models**: Managed the execution of all models to ensure consistency and accuracy. • **Feature Selection**: Identified the most relevant features to enhance model efficiency. • **Model Building**: Constructed models based on selected features and parameters.
Tanushree Mukherjee	• **Additional Model Ensemble**: Developed ensemble techniques to improve model performance. • **XGBoost**: Built and fine-tuned XGBoost models for complex problemsolving. • **Neural Network Model Building:** Designed and trained neural network models for advanced prediction tasks**.** • **Documentation:** Compiled project details, methodologies, and results for final reporting.
Divya Sai Sree Pilli	• **Gradient Boosting**: Implemented gradient boosting models to capture non-linear patterns in the data. • **Linear Regression Model Comparison**: Analyzed and compared the performance of linear regression models. • **Final Insights**: Derived actionable insights from the analysis and model outputs. • **Model Prediction on New Data**: Tested the best-performing models on unseen data to validate their predictive capabilities. • **Project Documentation**: Prepared a comprehensive report summarizing all phases of the project.