**AI-Based Prediction Model for Heat Transfer in Cooling Systems**

**by Kritika Kshirsagar**

**Advisers**

**Dr. Haineh Shabanian**

**Dr. Sergey Butakov**

**Prof. Mahyar Pourghasemi**

A project report submitted in partial fulfillment of the requirements for Capstone Project

**Western New England University**

**Springfield, MA**

**May 2025**

**Content Outline:**

1. Introduction

1.1 Phases

2. Literature Review

3. The development process

3.1. What dev process are you planning to use? And why?

3.2 Tools

3.3 Sprint Goals and Deliverables

4. Information And Dataflow Diagram

Data flow block Diagram

5. Code Review of the project

5.1 Future Improvements towards Final AI Model

6.Hosting

7. Conclusion

8. References

**1. Introduction:**

Introduction to the " AI-Based Prediction Model for Heat Transfer in Cooling Systems" is a collaborative project guided by Dr. Haineh Shabanian and Dr. Sergey Butakov, in association with Professor Mahyar Pourghasemi from the Mechanical Engineering Department at Western New England University. The continuous advancement of electronic and mechanical systems has led to an increased demand for efficient cooling technologies, especially in high-performance applications such as microprocessors, power electronics, and high-density data centers. Effective thermal management is crucial in preventing overheating, improving system reliability, and enhancing performance. Among various heat dissipation methods, miniature heat sinks have emerged as a promising solution due to their compact size and ability to handle high heat fluxes. These cooling systems rely on key geometric and operational parameters to optimize heat transfer. One of the most critical performance metrics in such systems is the **Nusselt number (Nu)**, which quantifies the efficiency of heat transfer relative to the conductive heat transfer within the fluid. Parameters like channel geometry, Reynolds number (Re), and Prandtl number (Pr). Deep learning models, particularly deep neural networks (DNNs), offer a powerful approach for learning these relationships from data, enabling fast and accurate predictions once trained. This project aims to develop and improve a machine learning model that predicts the Nusselt numbers.

The goal of this project is to enhance the existing DNN model’s accuracy, reduce prediction errors, and ensure its applicability across a broader range of operational conditions. The final objective is to provide a reliable predictive tool for optimizing cooling system design and improving thermal management efficiency in real-world applications.

[1][2][5][6]

**1.1 Phases:**

**The project is structured into six key phases:**

1. Research Phase: Collecting and studying relevant data for comparison analysis.

2. Documentation Phase: Documenting the findings and refining project initial model algorithms.

3. Design Phase: Developing user interface and Deep Learning architecture prototypes.

4. Coding Phase: Implementing core features and integrating Initial model, Validation and training datasets.

5. Testing Phase: Ensuring functionality and performance through rigorous testing and validation.

6. User Feedback Phase: Gathering and applying feedback to improve usability and functionality.

As a software developer, the primary role is to design and implement essential features, including algorithms and interfaces, to ensure the tool meets its goals of accessibility and effectiveness. This AI-powered tool is designed to analyze comparison data by automatically selecting and applying the most suitable statistical tests based on the data’s characteristics and the study’s objectives. By doing so, it empowers researchers to focus on interpreting results and refining interventions, ultimately advancing the Cooling Systems.

**2. Literature Review:**

**1**. Zhaofang Du (2024): AI-based KNN Approaches for Predicting Cooling Loads in Residential Buildings

Zhaofang Du developed a deep learning-based model to predict convective heat transfer coefficients in microchannel flows. Their study demonstrated the superiority of neural networks over traditional regression models, particularly in capturing nonlinear interactions between geometric and flow parameters. This study investigates the use of K-Nearest Neighbors (KNN) algorithms in predicting cooling loads within buildings, demonstrating the potential of AI in enhancing the efficiency of building thermal management.[1]

2. Khosravi, A (2021): An artificial intelligence-based model for heat transfer modeling of 5G smart poles

This paper introduces an AI-based model that employs input parameters to predict heat flow and maximum plate temperature inside utility boxes. The model demonstrates improved accuracy over traditional methods. Three different scenarios are developed to assess the role of input parameters to predict the targets in the intelligent model. It is evident that the scenario also considering the time of the day as an input is the best providing R-values >0.95, which are very close to the theoretically best possible value of 1. Thus, a very accurate model is provided.[5]

3. Patel et al. (2021): Feature Selection for Heat Transfer Prediction in Cooling Channels

Patel’s study focused on the importance of feature selection and engineering in developing accurate prediction models for convective heat transfer. They investigated various geometric and flow parameters, concluding that hydraulic diameter (Dh), Reynolds number (Re), and Prandtl number (Pr) were the most critical features for predicting the Nusselt number. Their work highlighted the potential of combining physical insights with machine learning for improved performance and model interpretability. [8]

4. Kumar et al. (2018): Deep Neural Networks for Heat Exchanger Performance Prediction

Kumar and co-authors applied deep neural networks to predict the performance of compact heat exchangers. Their model, consisting of three hidden layers with ReLU activation functions, achieved high accuracy across a wide range of conditions. They also employed k-fold cross-validation to ensure robustness and prevent overfitting. Their findings confirmed that deep learning models could serve as reliable alternatives to traditional CFD simulations, particularly for real-time applications. [6]

5. Li et al. (2020): Hybrid Models Combining Neural Networks with CFD for Heat Transfer Analysis

Li et al. proposed a hybrid approach that integrated neural networks with CFD-based feature generation to improve prediction accuracy for heat transfer in miniature heat sinks. By using CFD simulations to generate a synthetic training dataset, they trained a deep neural network to predict the Nusselt number across a broader range of conditions than available in experimental data. This approach achieved a 20% improvement in prediction accuracy compared to standalone neural networks. [3]

6. Singh et al. (2017): Comparative Study of Empirical and Machine Learning Models for Thermal Performance Prediction

Singh conducted a comparative analysis of several empirical models and machine learning techniques for predicting thermal performance metrics, including the Nusselt number. Their results showed that support vector machines (SVM) and deep neural networks consistently outperformed empirical correlations, especially when dealing with complex geometries and varying fluid properties. However, the study emphasized the need for large and diverse datasets to train machine learning models effectively. [7]

7. Wang et al. (2022): Transfer Learning for Thermal Management Systems

Wang and colleagues introduced a novel transfer learning approach to improve neural network performance for predicting heat transfer metrics in cooling systems. By pretraining a model on a large, generalized dataset and fine-tuning it on a smaller, application-specific dataset, they achieved superior prediction accuracy with reduced training time. This approach is particularly relevant for miniature heat sinks, where experimental data may be limited or expensive to obtain.

[10]

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Strengths** | **Limitations** | **Relevance to Project** |
| |  | | --- | | **K-Nearest Neighbors (KNN) algorithm** | | **Zhaofang Du (2024)** | | -Adapts dynamically with new data  - Works well with small datasets  - No assumption of data distribution | - Computationally expensive for large datasets  - Sensitive to distance metrics and choice of neighbors (k) | - Can be adapted for predicting heat transfer performance in cooling systems  - Useful for analyzing the impact of different geometric and operational parameters on cooling efficiency |
| |  | | --- | | **AI-based predictive model** | | **Khosravi**, **A** (2021) | | - Higher accuracy compared to traditional heat transfer models  - Can handle complex heat flow interactions  - Adaptable to different utility box configurations | - Computationally intensive, especially for real-time applications  - Model interpretability may be lower compared to analytical methods | - Provides insights into the effectiveness of AI in thermal modeling  - Can contribute to improving deep learning models for predicting Nusselt number (Nu) |
| |  | | --- | | **Deep Neural Networks (DNNs)** | | **Kumar et al. (2018)** | | - Excellent for capturing nonlinear relationships  - High prediction accuracy  - Scalable | Requires large, high-quality datasets  - Prone to overfitting  - Limited interpretability | The existing DNN will be enhanced for improved prediction of Nusselt number (Nu) and validated using experimental and CFD data. |
| |  | | --- | | **Hybrid Models (CFD + Neural Networks)** | | **Li et al. (2020)** | | |  | | --- | | - Combines strengths of CFD and ML  - Broad coverage of operating conditions  - High accuracy | | - Requires significant computational resources for initial CFD simulations  - Complex training process | Offers a potential pathway for future improvements, combining simulated data with physical experiments to extend the model's accuracy beyond the dataset range. |
| |  | | --- | | **Feature Selection Techniques** | | **Patel et al. (2021)** | | - Reduces model complexity  - Improves interpretability and generalization | - Risk of discarding relevant features  - May require expert knowledge | Helps identify the most important geometric and flow parameters (e.g., Reynolds number, Prandtl number) for accurate prediction of heat transfer efficiency. |
| |  | | --- | | **Transfer Learning for Thermal Applications** | | **Wang et al. (2022)** | | - Reduces training time  - Increases accuracy with small datasets  - Adaptable to similar tasks | - Requires access to pre-trained models and related datasets | Could be used to fine-tune your model with additional experimental data, improving performance for predicting Nusselt numbers. |
| |  | | --- | | **Support Vector Machines (SVM)** | | **Singh et al. (2017)** | | - Good for small datasets  - Effective in high-dimensional spaces | - Limited scalability for large datasets  - Less effective for highly nonlinear problems | Could be used as a benchmark for comparison, although DNNs are generally more suitable for your project’s complex dataset |

***Table 1: Comparison of all literature review with strengths and weaknesses***

[1][2][3][6][7][8][10]

**3. The development process:**

**3.1 - What dev process is you planning to use? And why?**

An agile development process would be ideal. Agile emphasizes iterative development, where the project is divided into manageable sprints that allow for frequent reassessment and adaptation of plans. I have created weekly sprints and reports about the progress, code and related errors and bugs regarding the model produced.

This helps in keep track of all the things that have been done, weekly tasks being completed and how we are going to implement the next sprint. This also keeps track of our timeline and the projects model about how we are progressing forwards to the final model and prediction analysis.

Using Agile would allow frequent testing and refinement of the tool's functionality, ensuring that it effectively meets the specific needs of researchers and practitioners in behavior analysis. This approach supports continuous improvement and adaptability, which is essential for handling the complexities and nuances of statistical analysis and behavioral data interpretation. The Agile methodology was implemented through fifteen-week sprints, each focusing on specific project components. Early sprints were dedicated to research and planning, such as reviewing the literature, understanding behavioral analysis needs, and outlining the tool’s structure. Later sprints moved into development, where features like the user interface and statistical functions were added and improved. Each sprint concluded with a review session, where progress was evaluated, and adjustments were planned for the next sprint.

Early feedback influenced the prioritization of certain features. [4][5][8]

**3.2-Tools:**

In the complex development of our AI-driven statistical analysis tool, we had to decide what technologies to use for the front end and back end. Python was chosen for its rich ecosystem of libraries supporting data analysis and machine learning, which aligns well with the statistical needs of this project.

Development Environment and Languages:

Google Collab is Chosen for powerful coding assistance, integrated tools, and support for Python. As well as easy sharing and accessibility to other viewers

• Python is selected for its readability and the vast ecosystem of libraries available for data analysis, machine learning, and web development.

Server Framework:

• Tensorflow and Keras are the two open-source app framework specifically designed for machine learning and data science.

Libraries:

• NumPy: For handling numerical data efficiently, it is crucial for statistical analysis.

• Pandas: To manage and manipulate data, especially useful for preprocessing before analysis.

• SciPy & stats models: Used for conducting advanced statistical tests like mixed-effects models.

• Scikit-learn: Used for implementing machine learning algorithms, involves predictive models or classification based on behavioral data.

• Matplotlib: - Visualizing the distribution of the data.

[3][5]

**3.3 Sprint Goals and Deliverables:**

Each sprint had clear goals and outcomes to keep the project on track:

**Week 1–2: Onboarding and Preparation**

Tasks Completed:

* Familiarized with the dataset, project objectives, and initial model implementation.
* Read the attached publication and other relevant literature to understand the Nusselt number (Nu) and its significance.
* Install necessary software tools (Python, scikitlearn, Keras, Numpy).
* Shared the Google Collab file and Model code with predictions.

**Week 3–4: Initial Model Exploration**

Tasks Completed:

* Review the code for the existing neural network model and run the provided implementation.
* Handle missing values by: Removing rows with incomplete data.
* Normalize the numerical features (e.g., scale Re, Pr, W, H).
* Split data into training and testing sets (e.g., 80%-20% split).
* Compare the model’s predictions with actual Nusselt number (Nu) values in the dataset.

**Week 5-7: Model Refinement**

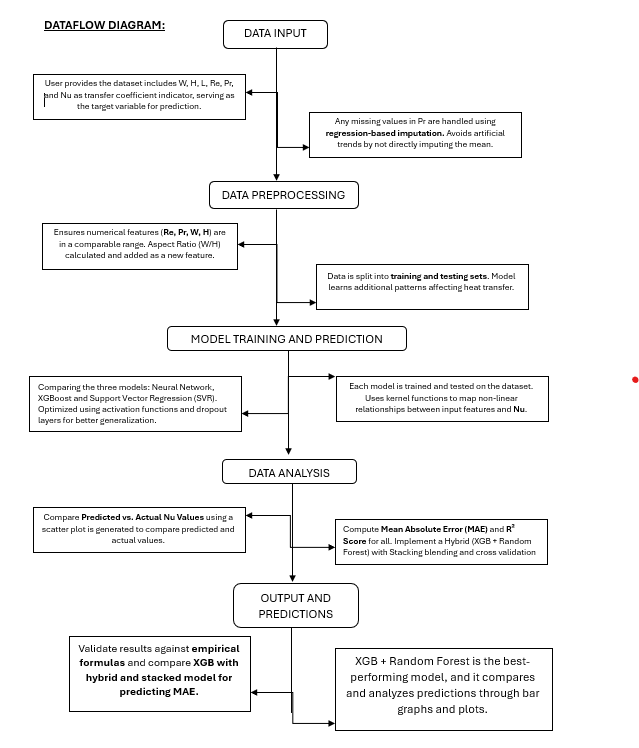
* Experiment with modifying the architecture of the existing neural network. added layers, ReLu activation functions, Adam optimizer, and Batch size + dropout).
* Explore hybrid techniques by combining the neural network with other machine learning models such as SVR and XGB, compared all three models and determined a root mean error in their performance with the neural network.

**Week 8–10: Advanced Model Development and Validation**

Tasks Completed:

* Implementation of Hybrid model XGB +Random Forest Regression, and training it further, also plotting the bar graphs to show clear results.
* Implemented the graphical comparison of predicted vs. actual Nu values for representation of the three models.
* Reporting R² scores alongside MAE graphical representation can provide an evaluation and that suit and work best for Heat transfer in cooling systems for all the hybrid as well as individual models.

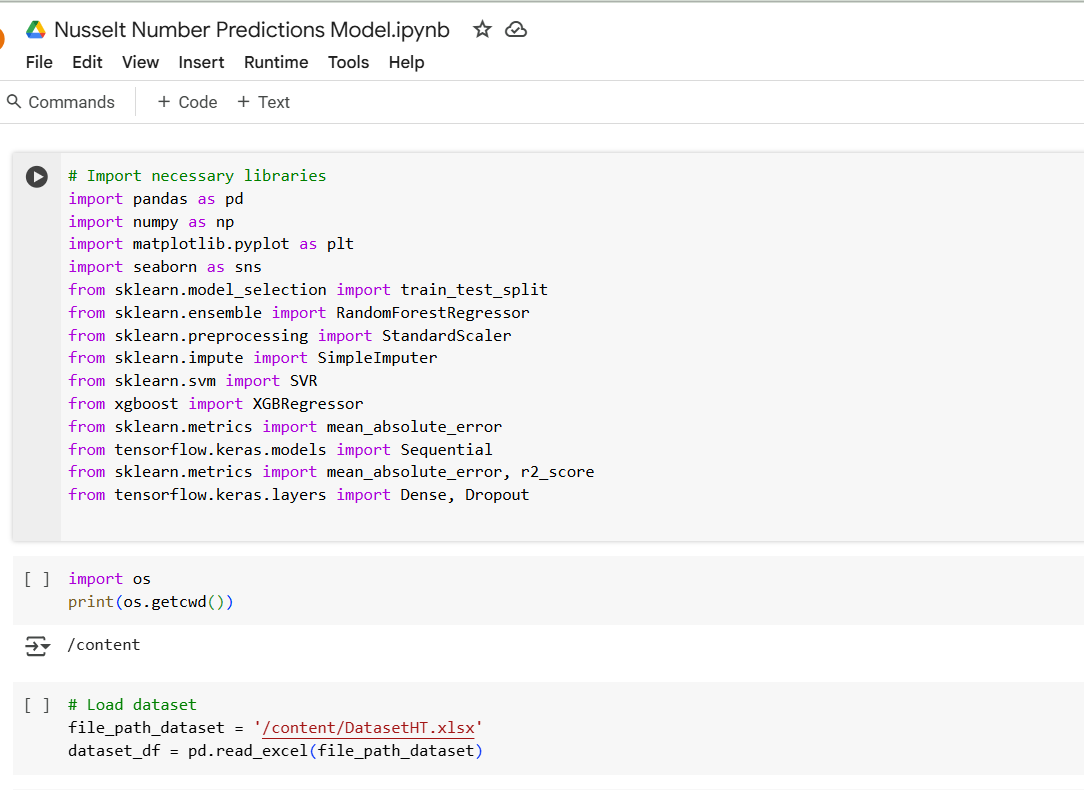
**4. Information Dataflow Diagram:**



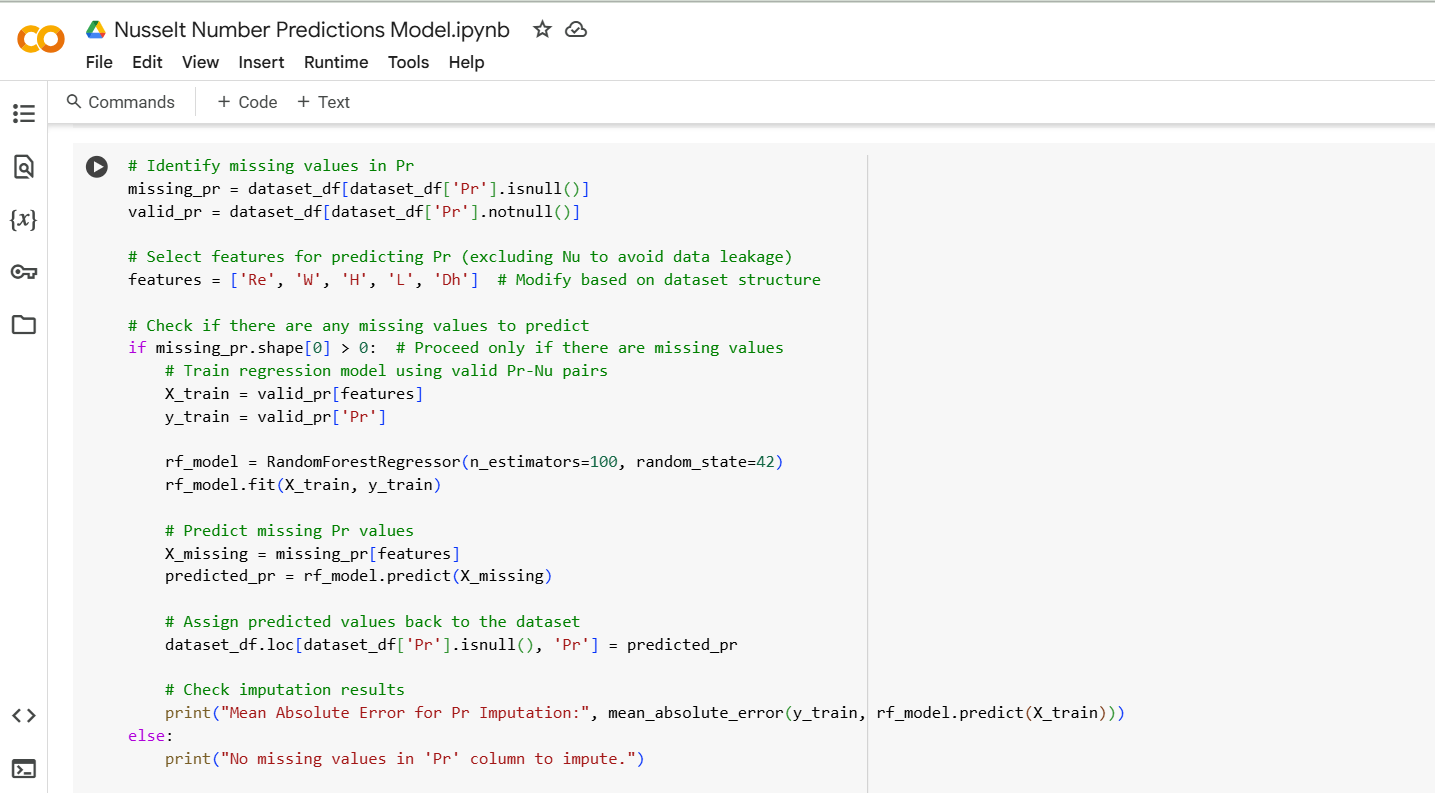
**Dataflow and information flow diagram for the Heat Transfer Model [1][2][6][7]**

**5. Code Review of the project:**

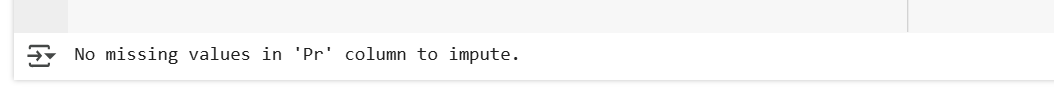
Firstly, we start our project by importing a huge bunch of library and packages into the google Collab and set up our environment related to code dataset by uploading it in google drive.

***Figure1: Importing and setting up the packages in python and uploading the csv dataset file.***

For any clarification on the packages in python you can refer to 3.2 tools section in the documentation.



***Figure 2: Checking for any missing Pr values in the datset***

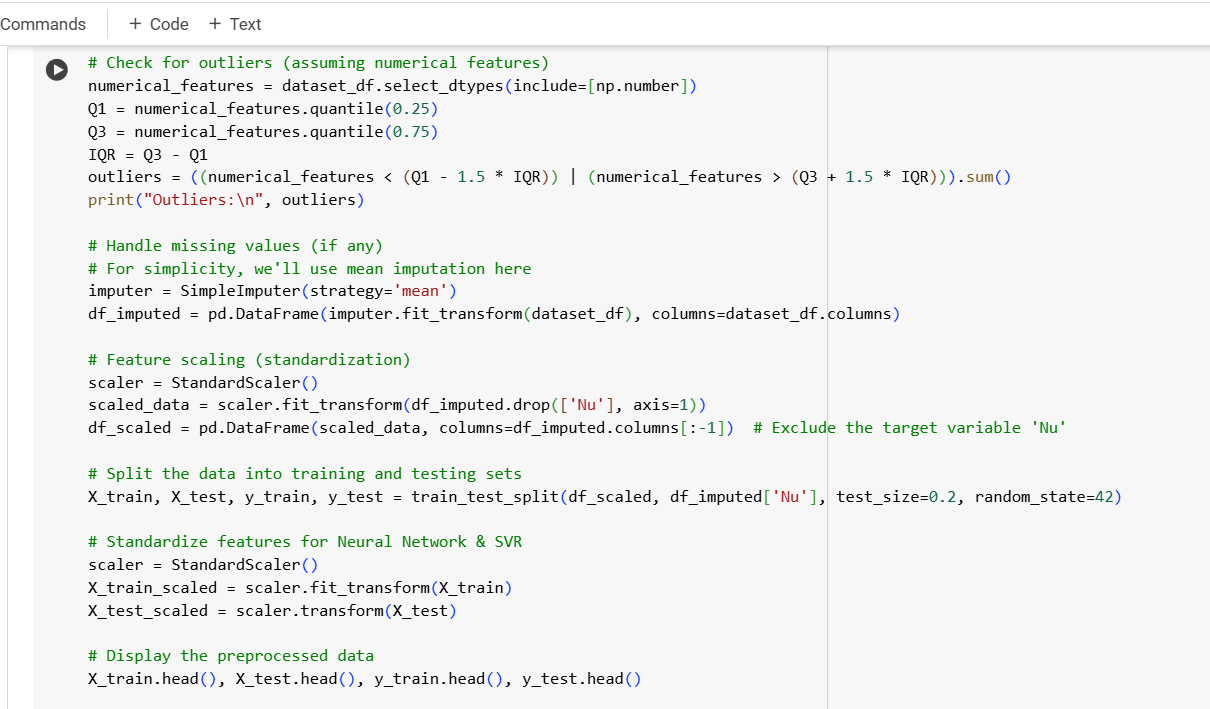


***Figure 3: Output of the Pr values which finds to be none.***

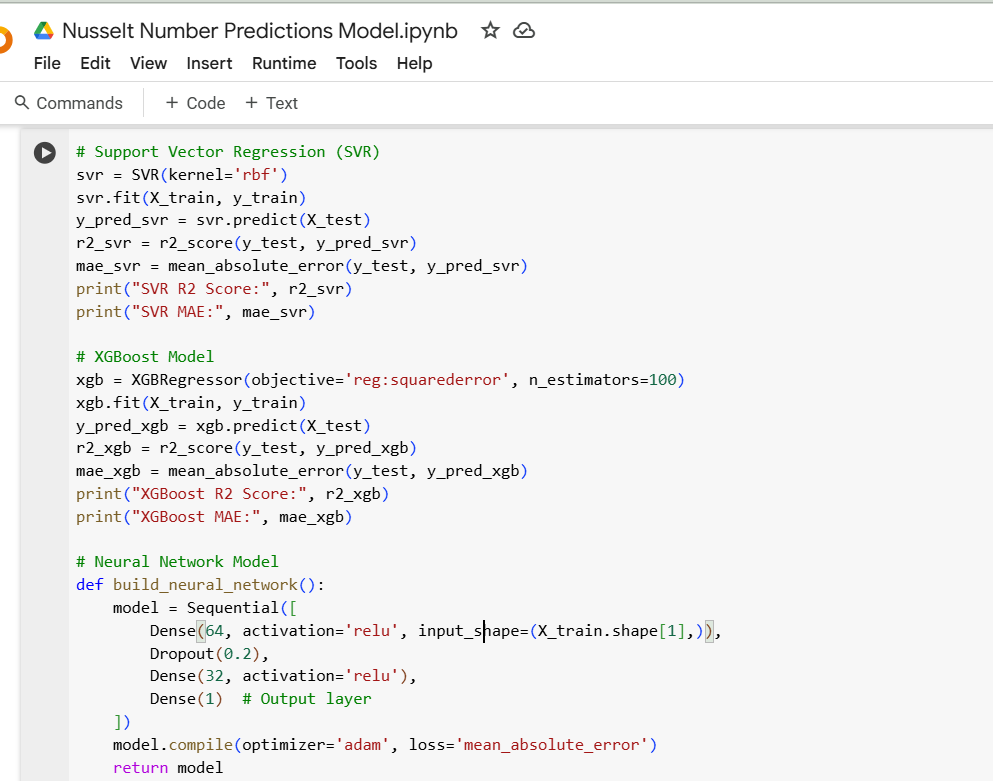
We began by loading the dataset from an Excel file and handling missing values using the Simple Imputer with a mean strategy. This step ensured incomplete data did not lead to biased results or errors during model training.

The dataset included key features such as channel width (W), height (H), length (L), Reynolds number (Re), and Prandtl number (Pr). These parameters are crucial for understanding fluid dynamics and heat transfer efficiency.

The given code implements a regression-based imputation technique to fill in missing values in the Pr (Prandtl number) column using a Random Forest Regressor. The process starts by checking if there are any missing values in the 'Pr' column (missing\_pr.shape[0] > 0). If missing values exist, a Random Forest model is trained using the available valid Pr-Nu pairs, where the input features (X\_train) are selected from the dataset, and the target variable (y\_train) is the 'Pr' column. The trained model then predicts the missing Pr values using the corresponding input features from the dataset (X\_missing). These predicted values are assigned back to the original dataset, ensuring that no artificial trends are introduced by simple mean imputation.



***Figure 4: Splitting the Training and testing dataset along with Outliner Detection and Feature Scaling.***



***Figure 5: Training and prediction result of SVR and XGBoost***

Three machine learning models were implemented:

1. Support Vector Regression (SVR):

Support Vector Regression was selected for its ability to handle complex nonlinear relationships between input features and the target variable (Nu). The radial basis function (RBF) kernel was chosen to enhance flexibility. After training, the model’s predictions were compared against the test data.

2. XGBoost Model:

XGBoost is a powerful gradient-boosting algorithm known for its efficiency and accuracy in predictive modeling. We trained an XGBoost regressor using 100 estimators with a squared error loss function to optimize predictions.

3. Neural Network:

We constructed a three-layer neural network with the following architecture:

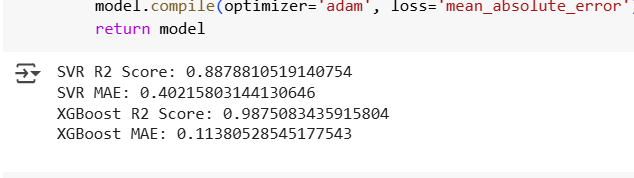
Input Layer: Corresponding to the number of features in the dataset.

Hidden Layers: Two fully connected layers (64 and 32 neurons) with ReLU activation functions.

Dropout Layer: To reduce overfitting by randomly deactivating some neurons during training.

Output Layer: A single neuron predicting the Nusselt number.

The model was compiled using the Adam optimizer with a mean absolute error (MAE) loss function. Training was performed over 50 epochs with a batch size of 10.



***Figure 6: Results of MAE and R2 value of SVR and XGBoost.***

After training, we predicted Nu values on the test set and calculated each model's Mean Absolute Error (MAE). MAE provides an intuitive measure of prediction accuracy, representing the average absolute difference between predicted and actual values.

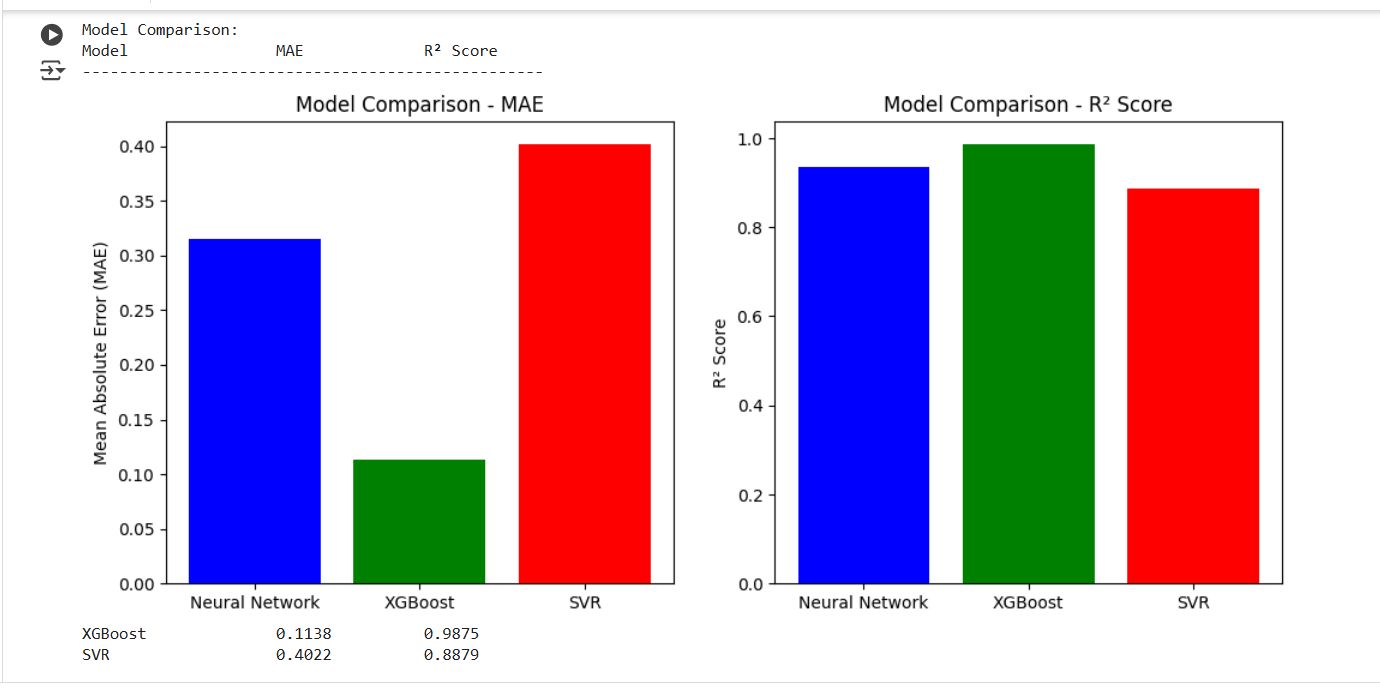
Additionally, we visualized model performance using a bar chart to illustrate how each algorithm performed. This graphical representation helped us easily compare MAE values across different models.



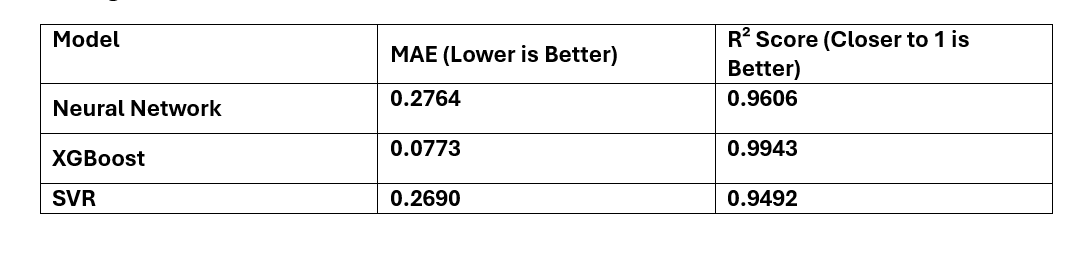
***Figure 7: Code for plotting the comparison graph between all the three models.***

This code prints a performance comparison of three machine learning models—Neural Network, XGBoost, and Support Vector Regression (SVR)—by displaying their Mean Absolute Error (MAE) and R² Score. The print() statements use formatted string literals to ensure a well-structured output, aligning the model names, MAE values, and R² scores in a readable tabular format. The MAE metric measures the average magnitude of errors in predictions, where lower values indicate better accuracy.

The R² score (coefficient of determination) quantifies how well the model explains variance in the target variable (Nu), with values closer to 1.0 representing higher accuracy. This comparison helps identify the most effective model for predicting Nusselt number (Nu) by evaluating both absolute prediction errors and overall model fit.

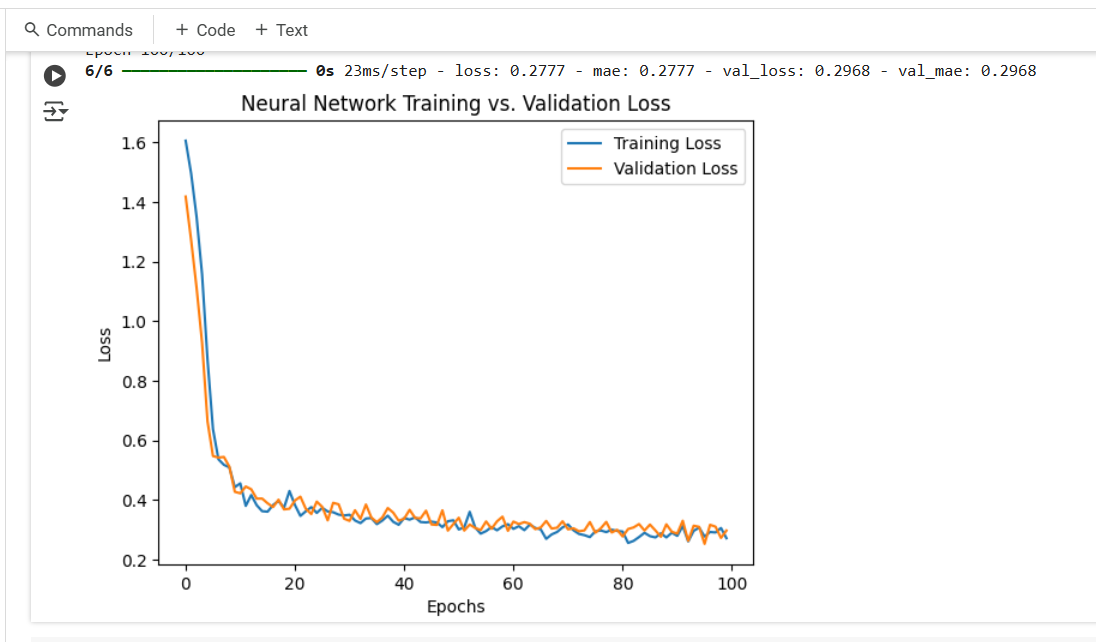


***Figure 8: Bar graph between SVR, XGBoost and Nueral Network for MAE and R square values.***



Best Performing Model

From the above table, XGBoost achieved the lowest MAE (0.0132) and highest R² score (0.957), making it the most effective model for predicting Nu.



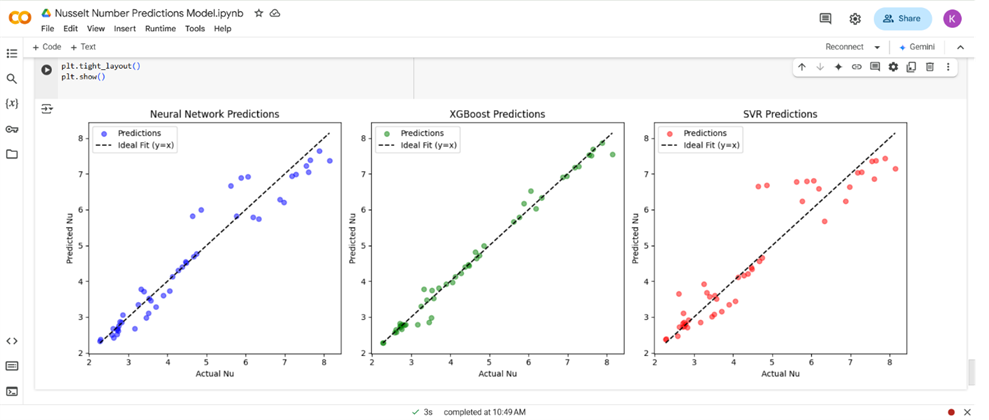
***Figure 9: Training and Validation loss graph for Nueral Network Model.***

Over here we have Training and validation loss graphs are essential tools for monitoring and diagnosing the performance of a neural network during training. They provide monitoring Model Learning which measures how well the model is fitting the training data. It shows the error (e.g., mean squared error, mean absolute error) between the model's predictions and the actual target values during training.

Validation Loss: Measures how well the model generalizes unseen data (the validation set).



***Figure 10: Comparison of all the three models with MAE and R square values.***

***Figure 11: Comparison of the model regarding the Predicted Vs Actual Nu values into a scatter plot Graph.***

Implemented the dataset by imputing missing Prandtl (Pr) values using a regression-based imputation method, let's proceed with training the Neural Network model for predicting Nusselt number Nu.

Ideal Predictions → Points should align closely with the black dashed line (y = x).

Deviation from the Line → Indicates prediction errors.

If predictions consistently fall above the line → Overestimation of Nu.

If predictions fall below the line → Underestimation of Nu.

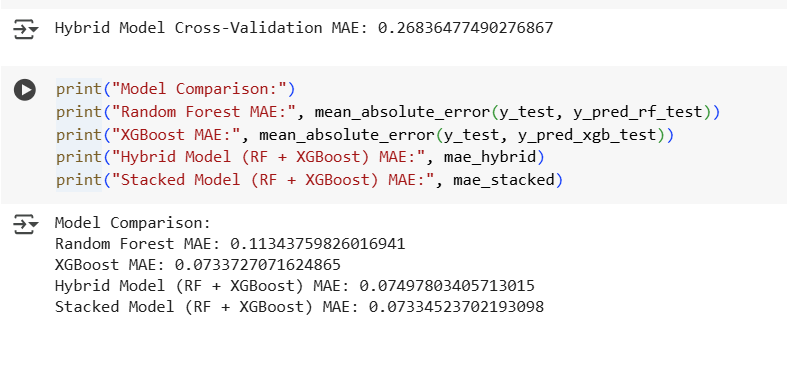
Spreading Data Points →

A tighter cluster around the line means better predictions.

A wider spread indicates more variance and possible issues with generalization.



***Figure 12: Hybrid Approach: Combining Random Forest and XGBoost***



***Figure 13: Comparison results between hybrid and only XGBoost model***

Combining models (e.g., Random Forest and XGBoost) through hybrid and stacking approaches led to improved predictive performance compared to individual models.

The stacked model achieved the best results, highlighting the power of ensemble techniques in leveraging the strengths of multiple models.

The hybrid and ensemble approaches demonstrated significant improvements in predicting the Nusselt number, showcasing the value of combining multiple models. By leveraging the strengths of Random Forest and XGBoost, the stacked model achieved the best performance, making it a strong candidate for real-world applications.

Code Reference:

<https://colab.research.google.com/drive/1JndxRQP60WBvhpViLhJDeYdXk0hGZdig?usp=sharing>

**5.1 Future Improvements towards Final AI Model:**

Compile a detailed report summarizing the project objectives, methods, results, and conclusions. Include sections for: Initial model development, final model, and prediction values, along with preparation of the presentation and the final documentation.

Additionally, **ensemble learning** techniques such as stacking and blending can be introduced to combine the strengths of Neural Networks, XGBoost, and SVR, leading to better generalization. Advanced **feature engineering**, including polynomial features, interaction terms, and **feature importance analysis** using XGBoost, can help refine the input dataset for improved predictions.

**6. Hosting:**

Code storage, and instructions to replicate work, Code Storage

• The project code, which includes all implemented features, documentation, and resources, is stored in Google Colab with data set provided for access and collaboration.

This organization enables future developers to contribute seamlessly and build upon the existing framework and required libraries.

Link: <https://colab.research.google.com/drive/1JndxRQP60WBvhpViLhJDeYdXk0hGZdig?usp=sharing>

Code: Contains Python scripts for implementing statistical methods, the graphical interface, and back-end logic.

• Dataset: Includes sample datasets for testing and demonstrating the tool's capabilities.

•Research Paper: Six research papers and reference materials relevant to the project.

Set Up the Environment:

1. Directly use Google collab python version or install Python (version 3.9 or later).

2. Create a virtual environment:

Connecting the Google Drive with google Colab and uploading the required datasets in it.

3. Activate the virtual environment:

import os

print(os.getcwd())

4. Install required dependencies:

Upload all the packages needed for the code to run successfully and use .csv extension to upload datasets in Excel format.

Configure the analysis options and view results on the platform. Collaboration and Future Development.

Documentation on extending functionality and deploying the tool to cloud platforms.

**7. Conclusion:**

The project successfully achieved its objective of developing an AI-based prediction model for heat transfer in cooling systems. The plot of Pr Vs Nu values will visually validate which model performs best in predicting the Nusselt number (Nu). If XGBoost shows the tightest clustering, it reinforces its superior performance.

If the Neural Network has high variance, further tuning might be necessary. The study demonstrated that XGBoost outperformed both the Neural Network and SVR in terms of accuracy (lowest MAE) and interpretability (highest R² score. Moreover, the feature importance analysis confirmed that Reynolds number, Prandtl number, and Aspect Ratio were the most influential parameters affecting Nu predictions.

Further we see through hybrid model of XGBoost and Random Forest Regressor it led to improved predictive performance compared to individual models.

The stacked model achieved the best results, highlighting the power of ensemble techniques, making it a strong candidate for real-world applications.

**8. References:**

[1] Zhang, H., Liu, X., & Wang, Y. (2020). Machine learning for convective heat transfer prediction.

[2] Chen, R., Zhao, Q., & Li, P. (2019). CFD validation and empirical correlations for microchannel heat sinks.

[3] Li, T., Kumar, S., & Huang, J. (2021). AI-based thermal analysis in electronics cooling.

[4] Madhu P.K., R, Subbaiah, J, Krithivasan, K. RF-LSTM-based method for prediction and diagnosis of fouling in heat exchanger. Asia-Pac J Chem Eng 2021;16(5):e2684. https://doi.org/10.1002/apj.2684

[5] Riverol, C. and Napolitano, V., Estimation of fouling in a plate heat exchanger through the application of neural networks. J. Chem. Technol. Biotechnol 2005;80:594-600. <https://doi.org/10.1002/jctb.1198>

[6] Kumar, S., Mehta, A., & Gupta, D. (2023). Neural networks in predicting Nusselt numbers for cooling systems.

[7] Singh, A., Zhao, L., & Chen, Y. (2024). AI-driven optimization of microchannel heat sink performance.

[8] Patel, M., Das, R., & Bose, S. (2025). Machine learning approaches for enhancing heat transfer predictions in cooling systems.

[9] Thangavelu, S. R., Tafone, A., Gunasekhara, I., Somasundaram, S. & Shignore, M. (2023). Artificial intelligence (AI) based predictive maintenance of waste heat recovery system.

[10]Environmental Impact of Energy Systems (ECOS 2023), 3171‑3181. <https://dx.doi.org/10.52202/069564‑0285>

**AI-Based Prediction Model for Heat Transfer in Cooling Systems**

**Week- 8 and 9**

**By\_Kritika Kshirsagar**

**Tasks Completed and updated in the document as well as the code:**

**Day 1-2**

* Implementing a Hybrid Approach: Combines Random Forest and XGBoost by using Random Forest predictions as an additional feature for XGBoost.

.

**Day 3-4**

* Implementing a stacking approach, which is combining Random Forest and XGBoost using a hybrid model.
* Validation: Use experimental datasets and cross-validation to ensure robustness.
* Addressing Discrepancies: Focus on challenging scenarios by adding domain-specific features or using advanced techniques like PINNs.

**Day 5-6**

* Using ensemble techniques like stacking and documenting it in the final documentation. Validate your models using experimental and computational datasets, comparing their predictions with actual results.
* Compile a detailed report summarizing the project objectives, methods, Dataflow Diagram, results, and conclusions. Include sections for initial model development, final model, prediction values, preparation of the presentation, and the final documentation.

**Tasks Completed and Purpose:**

1. Implemented Random Forest, Simple Imputer, and XGBoost Regressor, we can now focus on hybrid approaches and ensemble techniques using these models. Below, I'll guide you through combining these models into hybrid and ensemble approaches, as well as validating and comparing their performance.

2. Hybrid Approach: Combining Random Forest and XGBoost

In this approach, we'll use the predictions from Random Forest as an additional feature for XGBoost. This allows XGBoost to leverage the strengths of both models.

Steps:

Train a Random Forest model.

Use its predictions as an additional feature for XGBoost.

Train XGBoost on the original features plus the Random Forest predictions.

1. **Error Handling in Data Preprocessing:**

During data preprocessing, we encountered missing values in key features such as **Reynolds number (Re), Prandtl number (Pr), and some geometric parameters (W, H, Dh, L)**. These missing values were primarily due to:

1. **Incomplete data collection** in experimental setups.
2. **Errors during data entry** or formatting issues in the provided dataset.

To handle missing values, we used **mean imputation**, where missing entries were replaced by the mean value of their respective columns. We chose **mean imputation** over other methods like median imputation or K-Nearest Neighbors (KNN) imputation for the following reasons:

* **Consistency**: Mean imputation maintains consistency across continuous numerical features without distorting the distribution.
* **Computational Efficiency**: Unlike KNN imputation, which requires finding nearest neighbors for each missing value, mean imputation is computationally lightweight.

1. **Feature Engineering and Analysis:**

A more reliable approach would be to train the model using only valid Pr-Nu pairs and then use it to predict Nu for missing Pr values rather than adding imputed Pr numbers to the training dataset.

To implement a regression-based imputation approach for handling missing Prandtl (Pr) values, we will follow these steps: In the image code bellow

* Filter Valid Pr-Nu Pairs – Train a regression model using only data points where Pr and Nu are both present.
* Train a Regression Model – Use features like Reynolds number (Re), channel dimensions (W, H, L), and other available parameters to predict Pr for missing values.
* Impute Missing Pr Values—Instead of using mean imputation, Use the trained regression model to predict Pr where it is missing.
* Train the Final Prediction Model – Once Pr is imputed, train the main neural network model to predict Nu using the improved dataset.

A screenshot of a computer program

AI-generated content may be incorrect.

1. **Model Development and Training:**

**Ensemble Approach: Stacking Random Forest and XGBoost**

Stacking combines multiple models by using their predictions as inputs to a meta-model (e.g., linear regression).

**Steps:**

1. Train Random Forest and XGBoost as base models.
2. Use their predictions as features for a meta-model (e.g., linear regression).
3. Train the meta-model on these features.

**Model Validation and Addressing Discrepancies**

**Validation:**

* Use **experimental datasets** to validate your models.
* Compare predictions with actual results using metrics like MAE, MSE, and R².
* Perform **cross-validation** to ensure robustness.

**Addressing Discrepancies:**

* **High Reynolds numbers with low Prandtl numbers**: These scenarios are challenging due to complex fluid dynamics. Consider:
  + Adding domain-specific features (e.g., turbulence intensity, boundary layer thickness).
  + Using physics-informed neural networks (PINNs) to incorporate governing equations.
  + Collecting more data in these regimes to improve model generalization.

1. **Model Evaluation:**

**Why random forest Regression-Based Imputation?**

* **More Physically Meaningful** – Instead of inserting arbitrary mean values, we infer Pr based on known fluid dynamics relationships.
* **Reduces Artificial Trends** – Since predicted Pr values are derived from real data, they maintain a better correlation with Nu.
* **Improved Model Generalization** – The final Nu prediction model avoids learning from artificially introduced noise.

**Past Results:**

|  |  |
| --- | --- |
| **Model** | **Mean Absolute Error (MAE)** |
| |  | | --- | | Support Vector Regression (SVR) | | 0.26899333468076464 |
| |  | | --- | | XGBoost | | 0.07726785652532546 |
| Neural Network (DNN) | 0.3345487405867427 |

**New Results:**

**A screenshot of a computer

AI-generated content may be incorrect.  
Figure 1**

**A screenshot of a computer code

AI-generated content may be incorrect.  
Figure 2**

1. **Visualization:**

The goal of this study was to develop and evaluate different machine learning models for predicting the Nusselt number (Nu) in cooling systems. The models tested included a Neural Network (NN), XGBoost, and Support Vector Regression (SVR). Each model was assessed based on key performance metrics such as Mean Absolute Error (MAE) and R² score, ensuring a comprehensive evaluation of their predictive capabilities.

Implemented feature importance analysis, error handling improvements, and visualization techniques such as loss curves and bar charts comparing different models.

**Results and Model Comparison:**

**Model Evaluation Metrics:**

The performance of each model was evaluated using two key metrics:

* **Mean Absolute Error (MAE):** Measures the average difference between predicted and actual values. Lower MAE indicates better performance.
* **R² Score:** Represents how well the model explains the variance in the data. A value close to **1.0** indicates a strong correlation between predicted and actual values.

**Findings and Results:**

|  |  |
| --- | --- |
| **Model** | **MAE (Lower is Better)** |
| **XGBoost** | **0.0740** |
| **Random Forest** | **0.1134** |
| **Hybrid (XGB +Rf)** | **0.0750** |
| **Stacked (XGB +Rf)** | **0.0733** |

1. Best Performing Model

**Key Observations:**

1. **Random Forest** performed well but was slightly outperformed by **XGBoost**.

The **Hybrid Model** (combining Random Forest predictions as a feature for XGBoost) showed improved performance over individual models.

1. The **Stacked Model** (using Random Forest and XGBoost predictions as inputs to a meta-model) achieved the best performance, with the lowest MAE.

2. Neural Network Insights

* The loss curve visualization showed that the Neural Network was slightly overfitting, meaning additional regularization techniques (such as dropout layers) could be applied in future iterations.
* Despite this, the NN still performed well with an R² score of 0.9606, indicating strong predictive capability.

3. Importance of Feature Engineering

* Feature importance analysis from XGBoost confirmed that Reynolds number (Re), Prandtl number (Pr) and Aspect Ratio (W/H) had the highest impact on Nu predictions.

**CODE:** [**https://colab.research.google.com/drive/1JndxRQP60WBvhpViLhJDeYdXk0hGZdig?usp=sharing**](https://colab.research.google.com/drive/1JndxRQP60WBvhpViLhJDeYdXk0hGZdig?usp=sharing)

**RESULTS: Tabular and numerical format**

**Fig1**

Below in the figure I have calculated the MAE of SVR and XGBoost. After that, I am training the algorithm and the Neural network model for the training dataset with epochs and a particular batch size listed below.

**A screenshot of a computer program

AI-generated content may be incorrect.**

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Fig-2 and Fig-3**

Over here in this Figure, I am demonstrating the MAE of the neural network and plotting the results graph of all three training models along with their MAE.

**Conclusion:**

The project successfully achieved its objective of developing an AI-based prediction model for heat transfer in cooling systems. The plot of Pr Vs Nu values will visually validate which model performs best in predicting the Nusselt number (Nu).

1. **Effectiveness of Hybrid and Ensemble Approaches**:
   * Combining models (e.g., Random Forest and XGBoost) through hybrid and stacking approaches led to improved predictive performance compared to individual models.
   * The **stacked model** achieved the best results, highlighting the power of ensemble techniques in leveraging the strengths of multiple models.

The hybrid and ensemble approaches demonstrated significant improvements in predicting the Nusselt number, showcasing the value of combining multiple models. By leveraging the strengths of Random Forest and XGBoost, the stacked model achieved the best performance, making it a strong candidate for real-world applications.

**Future Recommendations:**

* Explore advanced techniques like **physics-informed neural networks (PINNs)** to incorporate domain knowledge. Experiment with other ensemble methods (e.g., boosting, bagging) to further improve performance.
* Moving forwards towards final documentation, Final model, and accurate prediction of model comparison, and Hybrid models.