Assignment 2 – Regression using Scikit-learn

Student Name: Chukwuma Anayo-Ezikeoha **Student ID**: 22358526 **Programme**: 4BCT

Git Hub Link. : https://github.com/G-cae78/Regression-Model-Training

Algorithm 1 – Gaussian Process Regression

Detailed Description of Algorithm 1

The Gaussian Processes are nonparametric learning methods which can be used to solve both classification and regression problems. Unlike traditional parametric models which assume a fixed form with a set number of parameters, they can adapt their complexity to the amount of data available to them (GeeksForGeeks, 2025). The Gaussian process regressor is a supervised learning model described as "Gaussian" because it scales up from finite to an infinite number of dimensions. For any Gaussian process, if any subset of points is picked, those points will follow a multivariate Gaussian distribution which looks like figure 1. The goal of applying these to a dataset is to learn about the underlying distribution. GPs make predictions based on a mean function and a covariance (kernel) function, allowing them to estimate not only the predicted value but also the level of uncertainty associated with each prediction (SciKit-learn, 2024).

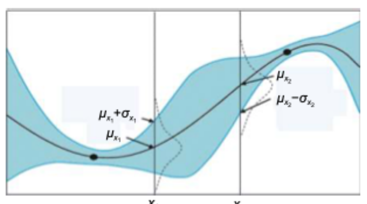


Fig. 1. One-dimensional Gaussian process.

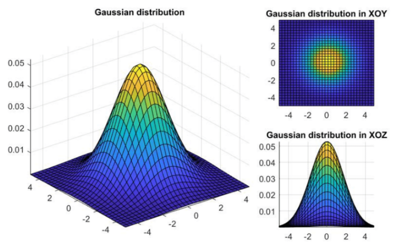


Figure 1 & 2: Shape of a Gaussian Distribution as per ResearchGate and Scikit-Learn (2024)

Why I Chose this Algorithm

GP provides a confidence interval along with its predictions, this would be an invaluable statistic to take into account when finetuning the model, especially in cases where the predictions might not change by much. It's another way to verify the direction each training step is moving in. The size of the dataset also makes it suitable for the GP model, as it is medium-sized and contains 9 features. GP is known to struggle computationally with large datasets but performs really well with smaller to medium-sized ones. Additionally, from looking at the dataset, the relationships between temperature, chemical composition, and tensile strength are most likely smooth and continuous, which aligns with GP's known strength in modelling gradual changes rather than big jumps. Considering the complexity of the chemical combination that determines the final tensile strength, the Gaussian Process would be a perfect fit.

Hyperparameter Details for Tuning.

Kernel : This parameter picks the covariance function of the Gaussian Process. This function defines how data points will influence each other, setting the overall shape smoothness and variability of the learned function. The kernel takes in two points and returns a similarity measure between the points in the form of a scalar. Each kernel is a functional method of its own, taking in its own hyperparameters that control the characteristics and impact the model fit. For example, the WhiteKernel is often combined with other kernels to model observation noise which represents random fluctuations or measurement errors that should not be explained by the underlying function itself. Higher noise levels lead to smoother, more conservative predictions as the GP trats data variations as random noise, while lower noise levels make the model trust the data more closely, potentially leading to overfitting. If no kernel is specified the default is ConstantKernel(1.0, constant_value_bounds="fixed") * RBF(1.0, length_scale_bounds="fixed) (Scikit-learn, 2024), which a starting point.

Alpha: The alpha parameter controls stability and tolerance by adding a tiny value to the kernel matrix during training to prevent mathematical errors (Scikit-learn, 2024). Also referred to as nugget, a higher alpha value makes the model ignore small variations and produce smoother predictions; a lower value results in the model following the data more tightly and may increase the risk of overfitting (Distill, 2019). When used in combination with a kernel such as the WhiteKernel, which already deals with noise, alpha should be kept very small(less than 1e^{-10}) to avoid double-counting the noise in the data.

Algorithm 2 – KNearest Neighbors (KNN) Regression

Detailed Description of Algorithm 2.

KNN is one of many supervised machine learning algorithms. KNN is mostly used for classification tasks, but it also has many benefits when used in regression problems. KNN is often referred to as a *Lazy* Learner because it does not explicitly learn a model during the training phase (GeeksForGeeks, 2025). All of the models' computations, such as measuring the distances between neighbours, are calculated during the prediction. A number of closest data samples, dictated by the k-value, is then selected, and their target values are averaged out to give the final prediction (Scikit-learn, 2024).

KNN is a non-parametric learning method, unlike parametric models that assume a specific form, KNN makes no prior assumptions about the data's distribution before training. It then predicts outcomes based on the similarity between the input data and its closest examples in the training set (Sayed, 2024).

$$\text{distance}(x, X_i) = \sqrt{\sum_{j=1}^{d}(x_j - X_{i_j})^2]}$$

$$d\,(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

$$d\,(x, y) = \left(\sum_{i=1}^{n} (x_i - y_i)^p\right)^{\frac{1}{p}}$$

Figure 3: Mathematical Expressions for Euclidean, Manhattan and Minkowski Distance as per GeeksForGeeks (2025)

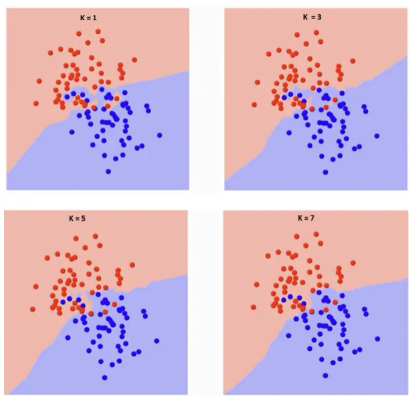


Figure 4: Effects of different K values on a dataset as per (Ebad Sayed on Medium, 2024).

Why I Chose this Algorithm.

KNN's non-parametric feature makes it suitable for the prediction of tensile strength based on the complex and non-linear relationship between temperature and chemical composition. It's simplicity and interpretability will make it easier to understand and visualise the relationships between the parameters and how they interact to determine the final tensile strength of steel. This is particularly useful for identifying patterns and trends within the dataset, providing a good baseline for comparison with more complex regression models like Gaussian Process.

Hyperparameter Details for Tuning.

Metric*:* The metric parameter defines the metric that will be used to compute how close neighbours are within the model. The default distance metric used is the Minkowski, other distance metrics include Euclidean Distance and Manhattan Distance. The Minkowski distance metric generalises to standard Euclidean distance when p=2 (Scikit-learn, 2024). KNN relies on distance metrics to determine which neighbour to use for regression tasks (GeeksForGeeks, 2025). The Euclidean Distance measure the straight line distance from point A to B, while

Manhattan distance is a grid based distance, Minkowski is generally preferred because it serves as a family which encompasses the two other distance metrics (p=2) for Euclidean and (p=1) for Manhattan (GeeksForGeeks, 2025).

N_neighbors: This hyperparameter specifies the number of nearest neighbours to consider when the model is making predictions. The default value for n_neighbors is 5 (Scikit-learn, 2024). For instance, if n_neighbour is set to 5, the algorithm will identify the 5 closest training samples and bases its prediction on the values from those 5 samples. For regression tasks the model typically uses averaging across the data samples to make predict the final value. Smaller values make the model more sensitive to noise and can lead to overfitting, while larger values create smoother boundaries but may oversimplify leading to underfitting (Sayed, 2024). Some approaches used to find the optimal value for this parameter include cross validation which test different values across multiple data splits to find the optimal value that yields the highest accuracy, other methods include elbow method (GeeksForGeeks, 2025).

Algorithm 1 – Gaussian Process Regression - Model Training and Evaluation

[Todo]

Training and Evaluation Details

[Todo]

Hyperparameter Tuning

[Todo]

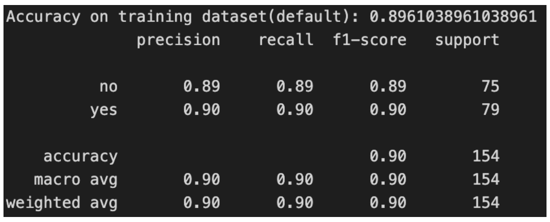
Discussion of results

[Todo]

Algorithm 2 – Nearest Neighbors Regression - Model Training and Evaluation

Training and Evaluation Details

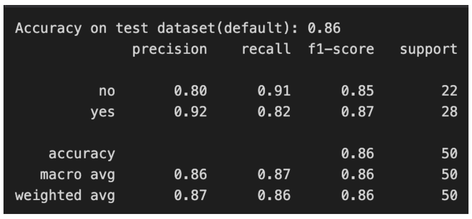
[Todo]



```
Accuracy on training dataset(default): 0.8961038961038961
              precision    recall  f1-score   support

          no       0.89      0.89      0.89        75
         yes       0.90      0.90      0.90        79

    accuracy                           0.90       154
   macro avg       0.90      0.90      0.90       154
weighted avg       0.90      0.90      0.90       154
```



```
Accuracy on test dataset(default): 0.86
              precision    recall  f1-score   support

          no       0.80      0.91      0.85        22
         yes       0.92      0.82      0.87        28

    accuracy                           0.86        50
   macro avg       0.86      0.87      0.86        50
weighted avg       0.87      0.86      0.86        50
```

Figure 7-8: LR Model Performance with Default Hyperparameters

Hyperparameter Tuning and Discussion of results

[Todo]

Comparative Analysis of Algorithm Performances

[Todo]

Recommended Hyperparameter Valued based on Results

Gaussian Process Regression (learning_rate): [todo: value]

Gaussian Process Regression (alpha): [todo: value]

Nearest Neighbors Regressor (weights) : [todo: value]

Nearest Neighbors Regressor (n_jobs): [todo: value]

Key Findings

[Todo]

Concluding Remarks

[Todo]

References

Scikit-learn (2024) *Ensemble methods — Random forest parameters*. Available at: https://scikit-learn.org/stable/modules/ensemble.html#random-forest-parameters (Accessed: 5 November 2025).

Wikipedia (nd.) Linear Regression. Available at: https://en.wikipedia.org/wiki/Linear_regression (Accessed 5 November 2025)

Wikipedia () Regression Analysis- *Isotonic Regression*. Available at: https://en.wikipedia.org/wiki/Isotonic_regression (Accessed: 5 November 2025)

Scikit-learn (2024) Gaussian Process — *Gaussian process Regression— Random forest parameters*. Available at: https://scikit-learn.org/stable/modules/gaussian_process.html#gaussian-process-regression-gpr (Accessed: 5 November 2025).

Optimized Faster-RCNN in Real-time Facial Expression Classification — *Scientific Figure on ResearchGate*. Available at: https://www.researchgate.net/figure/Gaussian-Distribution-It-can-be-seen-from-the-figure-that-the-closer-to-the-desired_fig2_34049043 (Accessed: 5 November 2025)

How Priors of Initial Hyperparameters Affect Gaussian Process Regression Models. Available at: https://arxiv.org/pdf/1605.07906 (Accessed: 6 Nov 2025)

Medium (2024) Voting Regressor: Intuition and Implementation. Available athttps://scikit-learn.org/stable/auto_examples/miscellaneous/plot_isotonic_regression.html#sphx-glr-auto-examples-miscellaneous-plot-isotonic-regression-py (Accessed 6 November 2025).

Distill (2019) A Visual Exploration of Gaussian Processes. Available at: https://distill.pub/2019/visual-exploration-gaussian-processes/ (Accessed 6 Nov 2025)

Hyperparameter Optimisation for Machine Learning Models based on Bayesian Optimisation. Available at: https://www.sciencedirect.com/science/article/pii/S1674862X19300047 (Accessed: 6 November 2025)

Scikit-learn (2024) Ensemble Methods —*K Nearest Neighbour Regressor*. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html#sklearn.neighbors.KNeighborsRegressor (Accessed: 8 November 2025).

Scikit-learn (2024) Gaussian Process —*Kernels for Gaussian Process*. Available at: https://scikit-learn.org/stable/modules/gaussian_process.html#gp-kernels (Accessed: 8 November 2025).

GeeksforGeeks (2025) K Nearest Neigbor. Available at https://www.geeksforgeeks.org/machine-learning/k-nearest-neighbours/ (Accessed: 8 November 2025).

Chen, S. and Luc, N.M. (2022) 'RRMSE voting regressor: A weighting function based improvement to ensemble regression', Mathematics for Engineers, Institute for Technologies of Metals, University of Duisburg-Essen. Available at: https://arxiv.org/pdf/2207.04837 (Accessed 8 November 2025).

Joblib Joblib.parallel_backend (2023).Available at: https://joblib.readthedocs.io/en/latest/generated/joblib.parallel_backend.html#joblib.parallel_backend (Accessed 9 November 2025).

Medium (2024) Mastering K-Nearest Neighbors: A Comprehensive Guide with Detailed Code. Available at: https://medium.com/@sayedebad.777/mastering-k-nearest-neighbors-knn-aa4b2ffca68b (Accessed 9 November 2025).

Log Of Work

Date	**Updates**
4/11/25	• Research Regression Model
5/11/25	• Start Write up for Gaussian Regression
6/11/25	• Discuss Hyperparameter selection • Finish Write up for Gaussian Process Regression
8/11/25	• Research write up fort Voting Regressor
9/11/25	• Write about hyperparameters for Voting Regressor • Submit half-way point
11/11/25	• Redo second algorithm to use KNN instead of voting regressor. • Resubmit Half Way point with KNN. • Write code and split dataset.