Utilizing Statistical Learning Algorithm to Solve Problems in Aerospace Engineering

Hafizh Renanto Akhmad*, Abisatya Hadyan Dhananjaya†, Isna Nur Firdausi‡, and Rizky Amalis Zhuraida§
Institut Teknologi Bandung, Bandung, Jawa Barat, 40132

I. Introduction

Aerospace engineering encompasses the design, development, and operation of aircraft and spacecraft systems. Recently, statistical learning algorithms have emerged as powerful tools for solving complex problems in aerospace engineering. Statistical learning is a field of study that focuses on developing algorithms and techniques to extract meaningful insights and patterns from data. It allows aerospace engineers to leverage the wealth of data collected from flight tests, simulations, wind tunnel experiments, and operational databases to gain deeper insights and make informed decisions. Statistical learning algorithms can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning algorithms use labelled data with known outcomes to learn patterns and relationships between input variables and the desired output. Unsupervised learning algorithms use labelled data with unknown outcomes to predict and classify new data points accurately. This report discusses the potential of statistical learning algorithms in aerospace engineering. It discusses various algorithms, their applications, and challenges specific to the aerospace domain. It examines case studies and real-world examples to highlight the effectiveness of statistical learning algorithms in improving performance, optimizing designs, enhancing safety, and enabling data-driven decision-making. By leveraging these algorithms' capabilities, the aerospace industry can push the boundaries of innovation, improve operational efficiency, and drive advancements in aircraft design, operation, and safety.

II. Statistical Learning

A. Definition

Statistical learning referes to the set approaches or methods to estimate a function relating a set of inputs and a set of outputs based on a set of known datas. Suppose we have a set of inputs (or features) X and a set of outputs (or responses) Y. There exists a fixed, unknown function f representing the relationship between X and Y such that

$$Y = f(X) + \epsilon \tag{1}$$

where ϵ is a random error term, which is independent of X and has mean zero. f represents the systematic information that X provides about Y [1]. Statistical learning is a set of approaches to estimate f based on some given data.

Regression vs Classification There are two types of problems in statistical learning: regression and classification. In regression problems, the response Y is quantitative, while in classification problems, the response Y is qualitative or categorical.

B. Linear Regression

1. Definition

Linear regression is a very simple approach in statistical learning. It is classified as a parametric method, which means that it makes an assumption about the functional form, or shape, of f, and a supervised learning, which means that it uses a set of labeled datas, and is used for regression problems. Linear regression assumes that there exists a

*13621060, Undergraduate Student, Faculty of Mechanical and Aerospace Engineering
†13621046, Undergraduate Student, Faculty of Mechanical and Aerospace Engineering
‡13621078, Undergraduate Student, Faculty of Mechanical and Aerospace Engineering
§13620071, Undergraduate Student, Faculty of Mechanical and Aerospace Engineering

linear relationship between $X = (X_1, X_2, \ldots, X_p)$ and Y, which means that f can be expressed as

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p \tag{2}$$

where X_j represents the jth feature of X and β_j represents the jth coefficient of f. β_j quantifies the relation between X_j and Y, which can be interpreted as average effect on Y of a one unit increase in X_j, holding other predictors fixed. The values of β_j are unknown and must be estimated based on the training data. Suppose $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ are the estimates of $\beta_0, \beta_1, \ldots, \beta_p$, respectively, we may have our estimate of Y as

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + ... + \hat{\beta}_p X_p \tag{3}$$

When $p = 1$, the linear regression model is called a simple linear regression model, otherwise, it is called a multiple linear regression model.

2. Estimating the Coefficients

The most common approach to estimate the coefficients is Ordinary Least Squares (OLS). OLS chooses $\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p$ to minimize the residual sum of squares (RSS), which is defined as

$$\text{RSS}\left(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p\right) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \hat{\beta}_2 x_{i2} - ... - \hat{\beta}_p x_{ip}\right)^2 \tag{4}$$

where n is the number of training data. Let

$$\mathbf{F} = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1p} \\ 1 & x_{21} & x_{22} & \ldots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \ldots & x_{np} \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad \hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_p \end{bmatrix}$$

and thus (4) can be rewritten as

$$\text{RSS}\left(\hat{\boldsymbol{\beta}}\right) = \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^T \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \tag{5}$$

The OLS estimates of $\beta_0, \beta_1, \ldots, \beta_p$ are the values that minimize RSS, or the values that satisfy

$$\frac{\partial \text{RSS}}{\partial \hat{\boldsymbol{\beta}}} = -2\mathbf{F}^T \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) = 0 \tag{6}$$

which gives

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{F}^T\mathbf{F}\right)^{-1} \mathbf{F}^T\mathbf{y} \tag{7}$$

3. Evaluating the Accuracy of the Coefficient Estimates

For each β_j, we have to assess its accuracy using a hypothesis test. Before that, we need to calculate each $\text{Var}\left(\hat{\beta}_j\right)$, the variance of $\hat{\beta}_j$. As each β_j in $\boldsymbol{\beta}$ is a random variable, using the definition of variance and (7), we have the variance of $\hat{\boldsymbol{\beta}}$ as

$$\text{Var}\left(\hat{\boldsymbol{\beta}}\right) = \sigma^2 \left(\mathbf{F}^T\mathbf{F}\right)^{-1} \tag{8}$$

where σ^2 is the variance of the error terms ϵ, which can be estimated using

$$\hat{\sigma}^2 = \frac{1}{n_{\text{train}} - p - 1} \text{RSS}\left(\hat{\boldsymbol{\beta}}\right) \tag{9}$$

where n_{train} is the number of training data and p is the number of features. The variance of $\hat{\beta}_j$ is the jth diagonal element of (8). After calculating variance for each $\hat{\beta}_j$, they can be tested using the following procedure:

1) Calculate the standard error of $\hat{\beta}_j$ using

$$\mathrm{SE}\left(\hat{\beta}_j\right) = \sqrt{\mathrm{Var}\left(\hat{\beta}_j\right)} \tag{10}$$

2) Determine the significant level α (usually 0.05)
3) Determine H_0 and H_1. As we're assessing relation between β_j and y, H_0 is $\beta_j = 0$ and H_1 is $\beta_j \neq 0$.
4) Calculate the t-statistic

$$t_j = \frac{\hat{\beta}_j - 0}{\mathrm{SE}\left(\hat{\beta}_j\right)} \tag{11}$$

5) Calculate the p-value using the t-statistic and the t-distribution with $n_{\text{train}} - p - 1$ degrees of freedom
6) Reject H_0 if p-value $< \alpha$, otherwise fail to reject H_0

4. Evaluating the Model Fit

After evaluating each β_j, we may evaluate the overall model performance. To evaluate how fit is the model with the training data, we may use the R^2 score and the RSE score. For linear regression, the R^2 score is defined as

$$R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} \tag{12}$$

where TSS is the total sum of squares. TSS is defined as

$$\mathrm{TSS} = \sum_{i=1}^{n_{\text{train}}} (y_i - \bar{y})^2 \tag{13}$$

where $\bar{y}$ is the mean of y. The RSE score is defined as

$$\mathrm{RSE} = \sqrt{\frac{1}{n_{\text{train}} - p - 1}\mathrm{RSS}} \tag{14}$$

A good model fit should have a high R^2 score and a low RSE score.

C. Logistic Regression

1. Definition

Logistic or logit regression is another simplest method in staistical learning. It is a classification model that predicts the probability of a data point belonging to a class using the logistic function. The logistic function is defined as

$$f(x) = \frac{e^x}{1 + e^x} = \frac{1}{1 + e^{-x}} \tag{15}$$

Using (15), we have the logistic regression model as

$$p\left(X\right) = \frac{1}{1 + e^{-\left(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p\right)}} \tag{16}$$

where $p\left(X\right)$ is the probability of the response of X belonging to a certain class. By manipulating the left-hand side, we may have

$$\frac{p\left(X\right)}{1 - p\left(X\right)} = e^{\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p} \tag{17}$$

which is called the odds of the data points having a response of the corresponding class. Taking the natural logarithm of both sides, we have

$$\ln\left(\frac{p\left(X\right)}{1 - p\left(X\right)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p \tag{18}$$

which is called the log-odss or logit.

When $p = 1$, the logistic regression model is called the simple logistic regression model. Otherwise, it is called the multiple logistic regression model.

2. Estimating the Regression Coefficients

The regression coefficients β_j can be estimated using the maximum likelihood method (MLE). Let

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 1 & x_1 & x_2 & \dots & x_p \end{bmatrix} \tag{19}$$

and thus, we may rewrite (16) as

$$p\left(\mathbf{F}\right) = \frac{1}{1 + e^{-\mathbf{F}\boldsymbol{\beta}}} \tag{20}$$

The likelihood function is defined as

$$\mathrm{L}\left(\boldsymbol{\beta}\right) = \prod_{i:y_i=1} p\left(\mathbf{F}_i\right) \prod_{i':y_{i'}=0} \left(1 - p\left(\mathbf{F}_{i'}\right)\right) = \prod_{i=1}^{n_{\text{train}}} p\left(\mathbf{F}_i\right)^{y_i} \left(1 - p\left(\mathbf{F}_i\right)\right)^{1-y_i} \tag{21}$$

The MLE method estimates $\beta_0, \beta_1, \cdots, \beta_p$ by maximizing (21). However, it is easier to maximize the log-likelihood function. The log likelihood function is defined as

$$\ln \mathrm{L}\left(\boldsymbol{\beta}\right) = \sum_{i=1}^{n_{\text{train}}} \left(y_i \mathbf{F_i} \boldsymbol{\beta} - \ln\left(1 + e^{\mathbf{F}_i \boldsymbol{\beta}}\right)\right) \tag{22}$$

and its derivative as

$$\frac{\partial \ln \mathrm{L}\left(\boldsymbol{\beta}\right)}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{n_{\text{train}}} \left(y_i - p\left(\mathbf{F}_i\right)\right) \mathbf{F}_i \tag{23}$$

(22) is maximized when (23) is equal to zero. However, there is no closed-form solution for $\boldsymbol{\beta}$, and thus, numerical methods are used to find the solution.

One of the most popular numerical solution for this problem is the Newton-Raphson method. The Newton-Raphson method is an iterative method that uses the first derivative of a function to find the root of the function. The general form of the Newton-Raphson method is:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{24}$$

where x_i is the i-th iteration, $f(x_i)$ is the function that we want to find the root of, and $f'(x_i)$ is the first derivative of $f(x_i)$. Therefore, we need first to compute the second derivative of the log-likelihood function

$$\frac{\partial^2 \ln \mathrm{L}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}\, \partial \boldsymbol{\beta}^T} = -\sum_{i=1}^{n_{\text{train}}} p\left(\mathbf{F}_i\right) \left(1 - p\left(\mathbf{F}_i\right)\right) \mathbf{F}_i \mathbf{F}_i^T \tag{25}$$

and thus the Newton-Raphson method for MLE for logistic regression is:

$$\hat{\boldsymbol{\beta}}_{i+1} = \hat{\boldsymbol{\beta}}_i - \left(\frac{\partial^2 \ln\left(\mathrm{L}(\hat{\boldsymbol{\beta}})\right)}{\partial \boldsymbol{\beta}\, \partial \boldsymbol{\beta}^T} \right)^{-1} \left(\frac{\partial \ln\left(\mathrm{L}(\hat{\boldsymbol{\beta}})\right)}{\partial \boldsymbol{\beta}} \right)^T \tag{26}$$

3. Evaluating the Accuracy of the Coefficient Estimates

Same as linear regression, we may evaluate each coefficient β_j accuracy using a hypothesis test. Before that, we also need to calculate the variance of β_j. The variance of β_j is the negative jth diagonal element of the inverse of the Hessian matrix (the second derivative of the log-likelihood function):

$$\mathrm{Var}(\beta_j) = -\left[\left(\frac{\partial^2 \ln\left(\mathrm{L}(\hat{\boldsymbol{\beta}})\right)}{\partial \boldsymbol{\beta}\, \partial \boldsymbol{\beta}^T} \right)^{-1} \right]_{jj} \tag{27}$$

After calculating the variance of β_j, we may conduct the following hypothesis test:

1) Calculate the standard error of β_j using

$$\mathrm{SE}(\beta_j) = \sqrt{\mathrm{Var}(\beta_j)} \tag{28}$$

2) Determine the significant level α (usually 0.05).
3) Determine H_0 and H_1. In this case, $H_0 : \beta_j = 0$ and $H_1 : \beta_j \neq 0$.
4) Calculate the t-statistic using

$$t_j = \frac{\beta_j - 0}{\mathrm{SE}(\beta_j)} \tag{29}$$

5) Calculate the p-value using the t-statistic and the degrees of freedom $n_{\text{train}} - p - 1$.
6) Reject H_0 if $p < \alpha$, otherwise fail to reject H_0.

D. K-Nearest Neighbors

1. Definition

K-nearest neighbors (KNN) is a supervised learning, non-parametric algorithm used for both regression and classification problems. It is a non-parametric method because it does not make any assumptions about the underlying data distribution. KNN is a lazy learning algorithm, which means that it does not have a training phase. Instead, it memorizes all the training data and when a new data point is given, it uses the training data to make a prediction.

KNN predicts the output of a new data point by looking at the k closest data points in the training set. The term "closest" is defined by a distance metric, which is usually the Euclidean distance. The Euclidean distance between two data points with inputs of $\mathbf{x}_a = (x_{a1}, x_{a2}, \cdots, x_{ap})$ and $\mathbf{x}_b = (x_{b1}, x_{b2}, \cdots, x_{bp})$ is defined as

$$d(\mathbf{x}_a, \mathbf{x}_b) = \sqrt{\sum_{i=1}^{p} (x_{ai} - x_{bi})^2} \tag{30}$$

where p is the number of features. As we take the "distance" that is measured from the value difference for each feature, it is important to scale the features so that they have the same scale. Otherwise, the feature with the largest scale will dominate the distance metric.

For this report, we will treat every k neighbors uniformly. For classification problems, the output of KNN is the mode of the k closest data points, that is,

$$\hat{y} = \mathrm{mode}(y_1, y_2, \cdots, y_k) \tag{31}$$

On the other hand, the output of KNN is the mean of the k closest data points for regression problems, that is,

$$\hat{y} = \frac{1}{k} \sum_{i=1}^{k} y_i \tag{32}$$

2. Finding the Optimal Number of Neighbors

The number of neighbors, k, is a hyperparameter of the KNN algorithm. The optimal value of k depends on the data. Therefore, to find the optimal value of k, we may iterate through different values of k and choose the value of k that gives the best performance. In this report, we will use K-fold cross-validation algorithm to find the optimal value of k.

The K-fold cross-validation works by dividing the training data into K folds. Then, for each value of k, we train the model using $K - 1$ folds and test the model using the remaining fold. We repeat this process K times, each time using a different fold as the test set. The average performance of the model is then computed. The value of k that gives the best performance is chosen as the optimal value of k. The metrics use for evaluating the performance of the model depends on the problem. All of the metrics used in this report will be discussed in the next section.

E. Correlation Coefficient

The correlation coefficient is a measure of the linear relationship between two variables. The most common correlation coefficient is the Pearson correlation coefficient, which is defined as

$$r_{XY} = \frac{\sum_{i=1}^{n} (X_i - \mu_X)(Y_i - \mu_Y)}{\sqrt{\sum_{i=1}^{n} (X_i - \mu_X)^2 \sum_{i=1}^{n} (Y_i - \mu_Y)^2}} \tag{33}$$

where X_i and Y_i are the i-th data points of X and Y, respectively, $\bar{X}$ and $\bar{Y}$ are the mean of X and Y, respectively, and n is the number of data points. The Pearson correlation coefficient is bounded between -1 and 1. A value of -1 indicates a perfect negative linear relationship, while a value of 1 indicates a perfect positive linear relationship. A value of 0 indicates no linear relationship.

F. Model Accuracy Metrics

1. Regression Problem

For regression problems, we will use the root mean squared error (RMSE) and mean absolute error (MAE) as the accuracy metrics. The RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (y_i - \hat{y}_i)^2} \tag{34}$$

where n_{test} is the number of data points in the test set, y_i is the actual value of the i-th data point, and $\hat{y}_i$ is the predicted value of the i-th data point. The MAE is defined as

$$\text{MAE} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} |y_i - \hat{y}_i| \tag{35}$$

2. Classification Problem

For classification problems, we will use the accuracy, precision, recall, and F1-score as the score metrics.

1) *Accuracy* is the most intuitive metric and it is simply a ratio of correctly predicted observation to the total observations. It is defined as

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}} \tag{36}$$

2) *Precision* is the ratio of correctly predicted positive observations to the total predicted positive observations. It is defined as

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{37}$$

Precision is a good metric to use when the cost of false positives is high.

3) *Recall* is the ratio of correctly predicted positive observations to the all observations in actual class. It is defined as

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{38}$$

Recall is a good metric to use when the cost of false negatives is high.

4) *F1-score* is the weighted average of precision and recall. Therefore, this metric takes both false positives and false negatives into account. It is defined as

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{39}$$

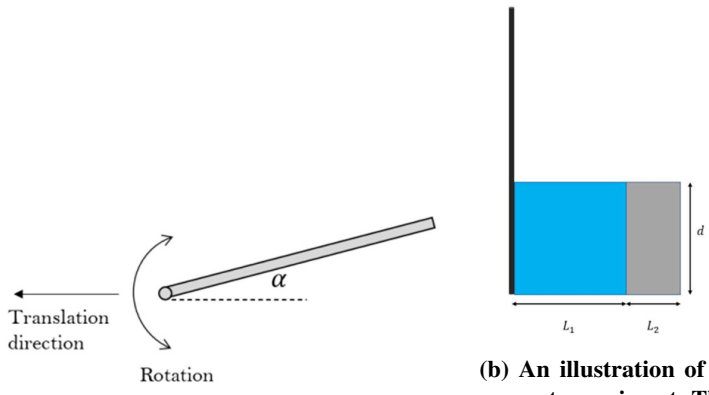
F1-score is a good metric to use when we need to seek a balance between precision and recall.

III. Problem Cases

A. Case 1: Flapping Plate Thrust

We're using the data from Fathurrohim et al. [2] about kinematic of a flapping plate to simulate the fin's movement of a fish on AUV (Autonomous Underwater Vehicle). In the study, the effect of the fin's stiffness and the flapping frequency on thrust production while the AUV is moving forward. The schematic is given in Figure 1, here the fin rotates with a maximum angle alpha with a certain flapping frequency. The plate is composed of Acrylic and silicone

rubber length attached in tandem. The plate with 0% and 100% percentage of acrylic represents the most flexible and the stiffest panel, respectively. The stiffness of the plate is varied by changing the proportion of the Acrylic and the silicone rubber length, in which the effective stiffness of the composite plate can then be calculated. For each proportion of material, the flapping frequency is varied and the thrust is measured using a set of experimental setup comprising a towing tank and a load cell.



(a) Caudal fin kinematic

(b) An illustration of the fin panel used in the current experiment. The length L of the fin panel is 100 mm. The length of the acrylic and silicone rubber part are denoted as L_1 and L_2, respectively

Fig. 1 Schematic of the problem

The dataset consists of 140 data points and 3 columns. The columns are the flapping frequency `flapfreq`, the acrylic percentage `percentage_ac` indicating the stiffness, and the thrust `Thrust_N`. For this problem, we are going to predict the thrust produced by the flapping plate given the flapping frequency and the acrylic percentage.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 140 entries, 0 to 139
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   percentage_ac  140 non-null    int64
 1   flapfreq       140 non-null    float64
 2   Thrust_N       140 non-null    float64
dtypes: float64(2), int64(1)
memory usage: 3.4 KB
```

Fig. 2 Information of the data set for case 1

Data Inspection and Preprocessing First, we check null values from the data set and make sure there is no null values, which is confirmed by 2. Next, we may calculate the Pearson correlation coefficient between each columns, as shown in 3. We can see that the magnitude of the correlation coefficient of flapping frequency on the matrix is 0.3 and the magnitude of the correlation coefficient of stiffness is 0.15. So, we know that flapping frequency is more correlated to the output rather than stiffness. However, both the magnitudes are closer to 0 which means that both variables are quite uncorrelated with the output.

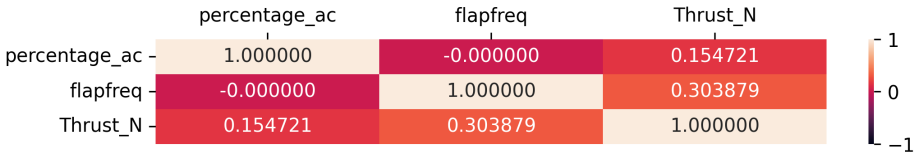


Fig. 3 Pearson correlation coefficient heatmap for case 1

We may also plot the data in a 3D scatter plot, as shown in Figure 4.

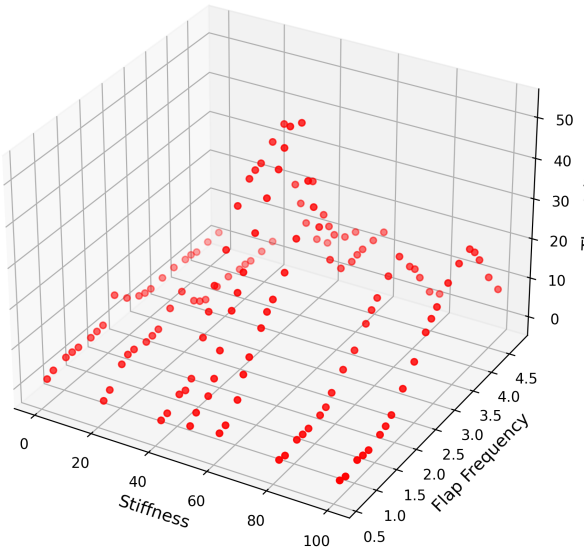


Fig. 4 3D scatter plot of the data in case 1

Train-Test Split Before we train the model, we need to split the data into training and testing set. We will use 80% of the data for training and the rest for testing. The testing set will be used to evaluate the accuracy of the model. We will use `train_test_split` from `sklearn.model_selection` to split the data.

Model Selection Here, we have to predict a continuous value, thus we will use regression models. We will use linear regression and KNN regression.

Model Training dan Evaluation

Linear Regression Using 7, we can calculate the value of βs for linear regression. Their values including the hypothesis test is shown in Table 1. We also obtained the value of R^2 and MSE, which are stored in Table 2.

Table 1 Coefficient for Linear Regression Model in Case 1

Beta	Value	Standard Error	t-statistic	p-value	Hypothesis Test Result
0	2.685464	3.355178	0.800394	0.425208	Do Not Reject H_0
1	0.048543	0.039070	1.242471	0.216705	Do Not Reject H_0
2	3.225045	0.977165	3.300410	0.001302	Reject H_0

Table 2 Coefficient of Determination and Residual Standard Error for Linear Regression Model in Case 1

Score Metric	Value
R^2	0.106227
RSE	12.910842

KNN Regression Before building the KNN model, we will scale the data based on the maximum and minimum values of the training set by using `MinMaxScaler` from `sklearn.preprocessing`, because the data distribution is not a normal but there is no outlier. The cross-validation for hyperparameter-tuning the k values will be conducted with `GridSearchCV` from `sklearn.model_selection`. Using 10-fold cross-validation, we obtained the best value of k to be 2. Figure 5 shows the cross-validation score for each k value.

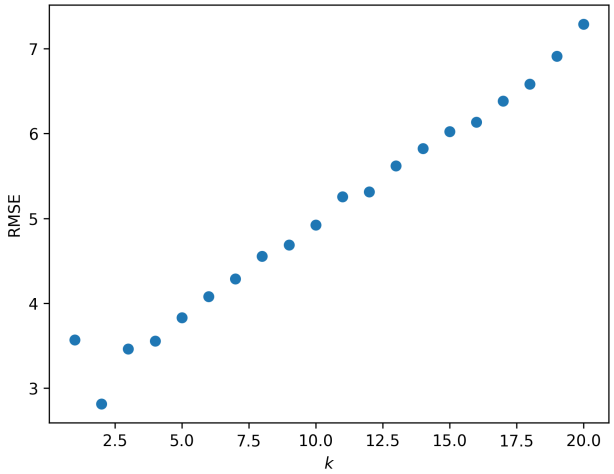


Fig. 5 Cross-validation score for each k value in case 1

Model Evaluation We will use the testing set to evaluate the accuracy of the model, using RSME and MAE value. The result is shown in Table 3.

Table 3 RMSE and MAE of Linear Regression and KNN Regression in Case 3

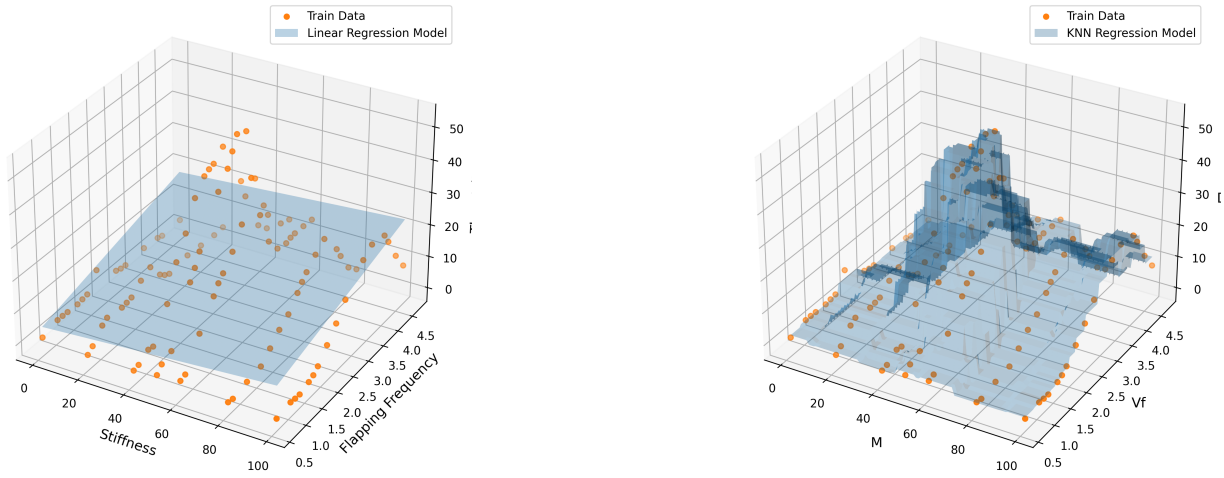
Model	RMSE	MAE
Linear Regression	11.102670	8.399192
KNN Regression	2.308239	1.182974

After comparing 2 models we created earlier, we know that RMSE and MAE values of the KNN regression model is lower than the linear regression model. From there, we can conclude that the KNN regression model better describes the approximation relationship between thrust as the output with the frequency and the stiffness as the input variables. Linear regression model does not fit to interpret the data set we process before can be seen from the RSE and R^2 score obtained, the R^2 score is closer to 0, it shows that data does not fit to the regression line, and the relation between the inputs and thrust is not linear.

Conclusion From the results of calculating the coefficient of the linear regression model and the hypothesis testing in Table 1, we can see that for β_0 and β_1, the hypothesis testing result is to do not reject H_0, we can interpret that the coefficients are not significant. While for β_2 the hypothesis testing result is to reject H_0, which means the coefficient is significant.

The results of hypothesis testing that do not reject H_0 are of β_0 and β_1, where β_0 is the intercept parameter and β_1 is the coefficient of the independent variable percentage of acrylic (stiffness). Because the H_0 of coefficient variable stiffness is not rejected, it can be concluded that this variable is most likely unrelated to output variable (Thrust). However, there is still a relationship between the independent variable flapping frequency and output because the hypothesis results for β_2 is to reject H_0.

We may plot the result of both linear regression and KNN regression in a form of surface over the range of the training data, which is shown in Figure 6.



(a) Linear Regression **(b) KNN Regression**

Fig. 6 Surface plot of the linear regression and KNN regression model in case 1

B. Case 2: Aerodynamic Coefficient

We're using data from a study investigating about the impact of transonic airfoil geometries on the drag and lift production. The airfoil is designed to operate at Mach number of 0.73 and Angle of Attack of 2 degrees. The aerodynamics coefficients are evaluated using a simplified aerodynamic solver, which neglects the viscosity of the air.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 12 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   rle       500 non-null    float64
 1   xup       500 non-null    float64
 2   yup       500 non-null    float64
 3   yxxup     500 non-null    float64
 4   xlo       500 non-null    float64
 5   ylo       500 non-null    float64
 6   yxxlo     500 non-null    float64
 7   alpha_te  500 non-null    float64
 8   beta_te   500 non-null    float64
 9   CL        500 non-null    float64
 10  CD        500 non-null    float64
 11  CM        500 non-null    float64
dtypes: float64(12)
memory usage: 47.0 KB
```

Fig. 7 Information of the data set for case 2

The dataset contains 12 columns and 500 rows. The description for each column are shown in Table 4. Figure 8

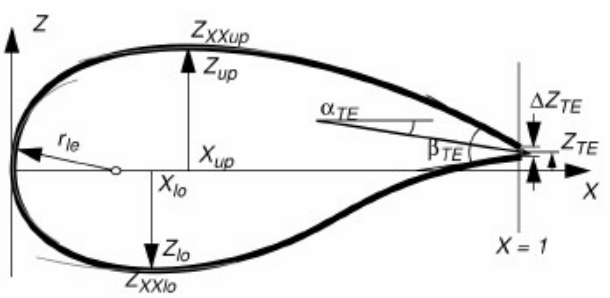


Fig. 8 Illustration of the parameters in case 2

also illustrate the parameters of the dataset. For this problem, we are going to predict the lift coefficient (CL) and drag coefficient (CD) based on the other parameters, and rank the importance of each parameter to the outputs.

Table 4 Description of Data in Case 2

Column Name	Definition
rle	Radius of leading edge
xup	Upper crest absissca
yup	Upper crest ordinate
yxxup	Upper crest curvature
xlo	Lower crest absissca
ylo	Lower crest ordinate
yxxlo	Lower crest curvature
alpha_te	Trailing edge direction
beta_te	Trailing edge wedge angle
CL	Lift coefficient
CD	Drag coefficient
CM	Moment coefficient

Data Inspection and Preprocessing Based on initial inspection, our dataset have a total of 500 entries, and none of them are null. Next, we may calculate the Pearson correlation parameter between each columns, as shown in Figure 9.

Fig. 9 Pearson correlation heatmap of the parameters in case 2

Based on values shown in 9, the input variables having the strongest correlation with CD, CL, and CM is demonstrated by yup, or the upper crest ordinate.

Train-Test Split Before we train the model, as we're going to rank the importance of each parameter, we may scale the data using `MinMaxScaler` from `sklearn.preprocessing`. We may also split the data into training and testing set, with the ratio of 80:20, using `train_test_split` from `sklearn.model_selection`.

Model Selection Here, we have to predict continuous values, thus we will use regression models. We will use linear regression and KNN regression.

Model Building and Evaluation

Linear Regression Using (7), we can calculate the value of βs of CD, CL, and CM for the linear regression model. The results including the hypothesis test is shown in Table 5. We also evaluate the R^2 and RSE score of the model as shown in Table 6.

Table 5 Coefficient for Linear Regression Model in Case 2

(a) Coefficient for Linear Regression of `CD`

Beta	Values	Standard Error	t-statistic	p-value	Hypothesis Test Result
0	0.003348	0.000917	3.650961	2.967747e-04	Reject H0
1	-0.000122	0.000574	-0.212497	8.318299e-01	Do Not Reject H0
2	-0.008557	0.000575	-14.881558	5.263659e-40	Reject H0
3	0.019872	0.000572	34.719043	1.666999e-121	Reject H0
4	-0.000251	0.000570	-0.440322	6.599476e-01	Do Not Reject H0
5	-0.001914	0.000577	-3.315638	9.998077e-04	Reject H0
6	0.006563	0.000580	11.320195	6.876674e-26	Reject H0
7	0.002742	0.000570	4.809075	2.167568e-06	Reject H0
8	-0.003241	0.000570	-5.684308	2.572116e-08	Reject H0
9	0.000050	0.000568	0.088217	9.297497e-01	Do Not Reject H0

(b) Coefficient for Linear Regression of `CL`

Beta	Values	Standard Error	t-statistic	p-value	Hypothesis Test Result
0	0.521795	0.005601	93.166063	4.622280e-269	Reject H0
1	0.015680	0.003507	4.471338	1.019951e-05	Reject H0
2	0.071101	0.003511	20.248937	7.215814e-63	Reject H0
3	0.263018	0.003495	75.250845	1.013006e-234	Reject H0
4	0.058669	0.003483	16.842160	2.998194e-48	Reject H0
5	-0.058992	0.003525	-16.734515	8.600908e-48	Reject H0
6	0.220595	0.003540	62.308767	3.810448e-205	Reject H0
7	0.075028	0.003482	21.547331	1.905243e-68	Reject H0
8	-0.104502	0.003482	-30.015382	1.611219e-103	Reject H0
9	0.000270	0.003466	0.077996	9.378711e-01	Do Not Reject H0

(c) Coefficient for Linear Regression of `CM`

	Beta	Standard Error	t-statistic	p-value	Hypothesis Test Result
0	0.045960	0.001605	28.634104	5.149599e-98	Reject H0
1	-0.004924	0.001005	-4.899170	1.411907e-06	Reject H0
2	0.026385	0.001006	26.219789	3.620143e-88	Reject H0
3	0.100599	0.001002	100.431480	2.655315e-281	Reject H0
4	0.010215	0.000998	10.232636	6.212311e-22	Reject H0
5	-0.026673	0.001010	-26.402278	6.382478e-89	Reject H0
6	0.052365	0.001015	51.611323	1.240393e-176	Reject H0
7	0.022564	0.000998	22.611134	5.382271e-73	Reject H0
8	-0.029863	0.000998	-29.929553	3.518305e-103	Reject H0
9	0.000521	0.000993	0.524790	6.000263e-01	Do Not Reject H0

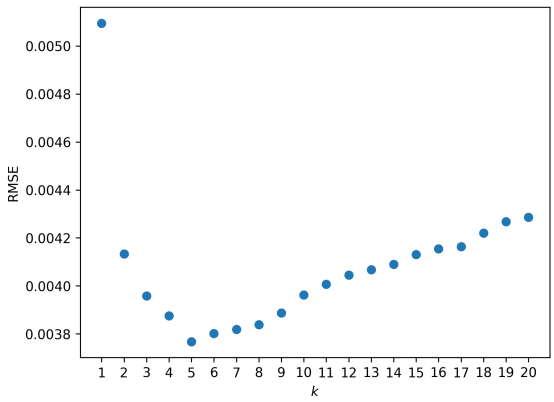
Table 6 Coefficient of Determination and Residual Standard Error for Linear Regression Model in Case 2

Score Metric	CD	CL	CM
R^2	0.808617	0.967612	0.975402
RSE	0.003305	0.020183	0.005784

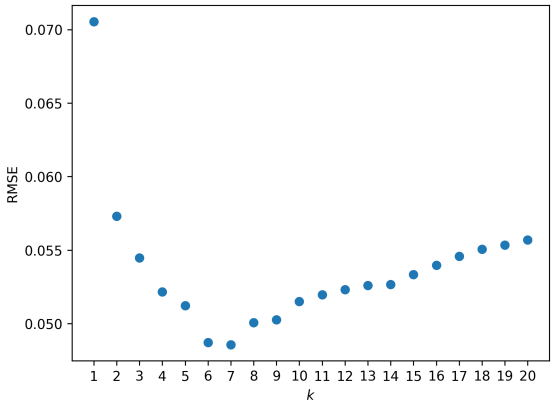
We may now evalute the linear regression model using R^2 and RSE score for each aerodynamic coefficients from 6.

1) CD
   From the table, we can see that the obtained value of R^2 is 0.808617. This translates to roughly 81% variance in the model is explained by the model. In engineering, preceding R^2 value is considered good and shows strong correlation between input and output. Therefore, the model is viable for making future predictions.
2) CL
   From the table, we can see that the obtained value of R^2 is 0.967612. This translates to roughly 97% variance in the model is explained by the model. This R^2 value is considered good and shows strong correlation between input and output and therefore, the model is viable for making future predictions.
3) CM
   From the table, we can see that the obtained value of R^2 is 0.975402. This translates to roughly 98% variance in the model is explained by the model. Same as previous coefficients, the R^2 value is considered good and shows strong correlation between input and output, making the model is viable for making future predictions.

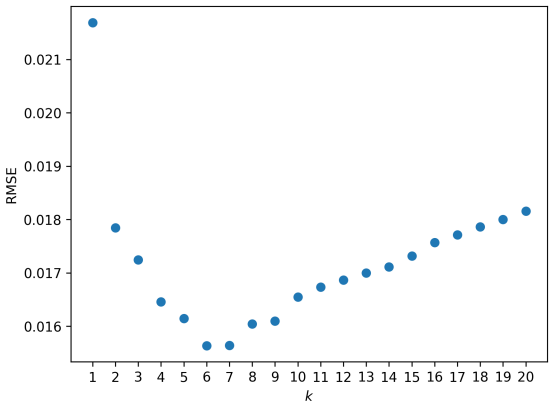
KNN Regression As the data has been scaled, we may continue directly to the model building. We will also conduct a grid search to find the k value for the KNN regression model, using `GridSearchCV` from `sklearn.model_selection`. With 5-fold cross validation, we obtain that the best k for CD, CL, and CM are 5, 7, and 6, respectively, as shown on Figure 10. We will then use these k values to build the KNN regression model.



(a) Grid Search for KNN Regression in Case 2 for CD



(b) Grid Search for KNN Regression in Case 2 for CL



(c) Grid Search for KNN Regression in Case 2 for CM

Fig. 10 Grid Search for KNN Regression in Case 2

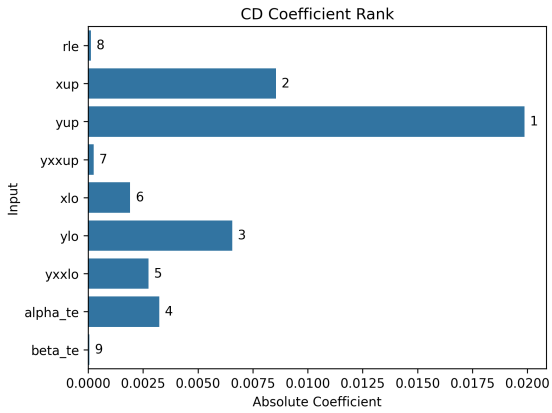
Model Evaluation We will use the testing set to evaluate the accuracy of the model, using RSME and MAE value. The result is shown in Table 7.

Table 7 Model Evaluation for KNN Regression in Case 2

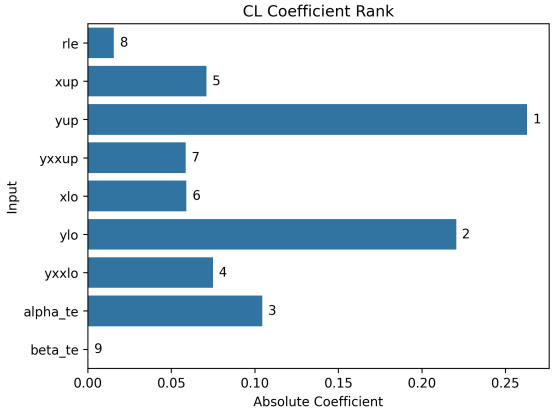
Coefficient	CD		CL		CM	
Metric	RSME	MAE	RSME	MAE	RSME	MAE
Linear Regression	0.003120	0.002393	0.018990	0.013370	0.014865	0.005039
KNN Regression	0.003514	0.003560	0.049321	0.039919	0.014645	0.011996

Conclusion Based on the result in Table 7, we can see that the linear regression model has less RMSE and MAE value for all aerodynamic coefficients. Therefore, we can conclude that the linear regression model is better than the KNN regression model for this case. This concludes that the relation between the aerodynamic coefficients and the input parameters is linear, which is also supported by the R^2 score. From here, we may rank the input parameters based on their importance to the aerodynamic coefficients, which is based on the magnitude of the coefficient of the linear regression model. The illustration of the ranking is shown in Figure 11. We can see that `yup` is the most important input parameters for all aerodynamic coefficients. This is also supported by the fact the correlation coefficient between `yup` and the aerodynamic coefficients are the highest among all input parameters. On the other hand, we can see that `beta_te` is the least important input parameters for all aerodynamic coefficients. This is also supported by the fact for

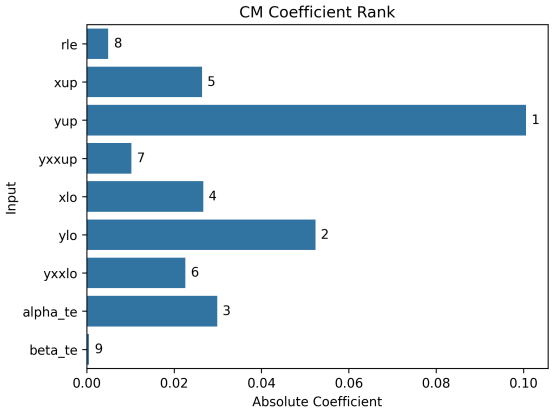
all hypothesis testing regarding the coefficient of `beta_te`, the result is always to not reject the null hypothesis, which means that the coefficient is not significant, and thus making it also insignificant in the linear regression model.



(a) Ranking of Input Parameters for `CD`



(b) Ranking of Input Parameters for `CL`



(c) Ranking of Input Parameters for `CM`

Fig. 11 Ranking of Input Parameters in Case 2

C. Case 3: Wing Flutter

We're using the data from Palar et al. [3] about flutter detection on NACA 64A10 airfoil. The data is retrieved from a numerical experiment which investigates the safety regime of a NACA 64A010 with two degrees of freedom: pitch and plunge motion. The study investigates the impact of two design conditions, namely, the Mach number and the flutter speed. A numerical aerodynamic solver was employed with the Mach number and the flutter speed as inputs, and the damping coefficient as the output. The illustration of the airfoil is shown in Figure 12.

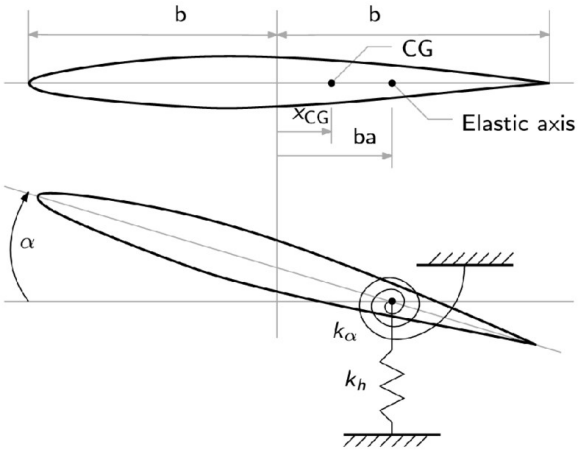


Fig. 12 NACA 64A010 Airfoil

The dataset consist of 300 data points and 3 columns. The columns are the Mach number `M`, the flutter speed `Vf`, and the damping coefficient `DC`. For this problem, we are going predict the occurence of flutter, that is, whether the damping coefficient is positive or negative, given the Mach number and the flutter speed. A negative damping coefficient indicates that the airfoil is unstable and will flutter, and vice versa.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   M       300 non-null    float64
 1   Vf      300 non-null    float64
 2   DC      300 non-null    float64
dtypes: float64(3)
memory usage: 7.2 KB
```

Fig. 13 Information of the data set for case 3

Data Inspection and Preprocessing Based on initial inspection, our dataset have a total of 300 entries, and none of them are null. Moreover, all of the columns are numerical, with general information stored in Table 8.

Table 8 General Statistics of Data in Case 3

	M	Vf	DC
count	300.000000	300.000000	300.000000
mean	0.768178	1.145337	-0.005795
std	0.092958	0.482497	0.039937
min	0.600000	0.400000	-0.104256
25%	0.687750	0.723500	-0.032059
50%	0.772000	1.100000	-0.000651
75%	0.858500	1.554500	0.023531
max	0.905000	2.000000	0.095338

and the Pearson correlation coefficient between each columns are shown in Figure 14.

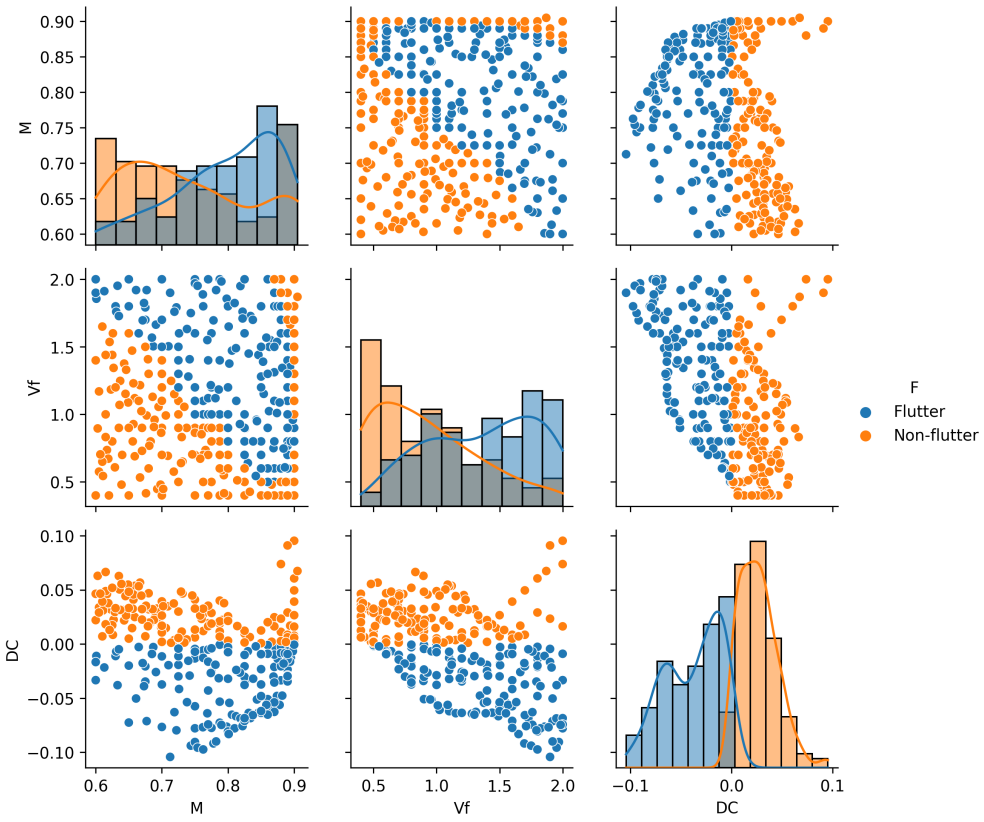


Fig. 15 Distribution of the data in case 3

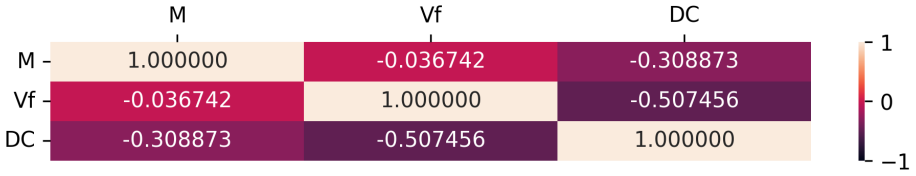


Fig. 14 Pearson correlation coefficient for case 3

We're given a set of numerical data, but we need to predict a categorical values. Therefore, we may add a new column, named f, which indicates whether the airfoil will flutter or not. We can see the distribution of the data in Figure 15. Moreover, as we have only two inputs and an output, we may plot the data in a 3D scatter plot, as shown in Figure 16. From Figure 16, we can see that high Mach number and high flutter speed is dominated by negative damping coefficient, which means that the airfoil will flutter. This is also supported by the correlation coefficient in Table 14, which shows that the damping coefficient is negatively correlated with both Mach number and flutter speed. From Table 14, we can see that the Mach number and flutter speed correlation coefficient is very small, which means that they are more likely to be independent with each other. We can also see the correlation between damping coefficient and other variables, which shows that the damping coefficient not quite correlated with the others, but it is more correlated with flutter speed than Mach number.

Train-Test Split Before we train the model, we need to split the data into training and testing set. We will use 80% of the data for training and the rest for testing. The testing set will be used to evaluate the accuracy of the model. We will use `train_test_split` from `sklearn.model_selection` to split the data.

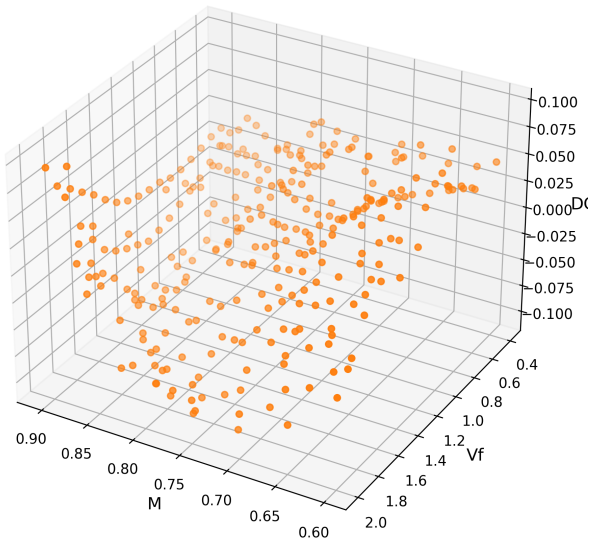


Fig. 16 3D scatter plot of the sample data in case 3

Model Selection Here, we have to predict a categorical value, so it's quite intuitive if we use a classification model. However, the classification of our target/response, `F`, is based on value of another column, `DC`, which can be predicted using a regression model. Therefore, we will use two approach in solving this problem:

1) Predict `DC` using a regression model, then obtain `F` by classifying the result manually.
2) Predict `F` using a classification model.

Model Training and Evaluation For both KNN algorithm, we will use `KNearestRegressor` and `KNearestClassifier` from `sklearn.neighbors`. Also, before building the KNN model, we will scale the data based on the maximum and minimum values of the training set by using `MinMaxScaler` from `sklearn.preprocessing`, because the data distribution is not a normal but there is no outlier. The cross-validation for hyperparameter-tuning the k values will be conducted with `GridSearchCV` from `sklearn.model_selection`.

Regression Model We will use linear regression and KNN regression to predict the damping coefficient. Using (7), the resulting coefficient for the linear regression model and the hypothesis testing is shown in Table 9. We can see that all H_0 is rejected and thus, all of the coefficients are significant.

Table 9 Coefficient for Linear Regression Model in Case 3

Beta	Value	Standard Error	t-statistic	p-value	Hypothesis Test Result
0	0.158557	0.017985	8.816034	2.555677e-16	Reject H_0
1	-0.145096	0.022156	-6.548710	3.543494e-10	Reject H_0
2	-0.046859	0.004335	-10.808856	2.045755e-22	Reject H_0

With this model, we obtained R^2 and RSE as stored in 10. R^2 value shown in the table indicates that the relation between `DC` and the inputs are not linear.

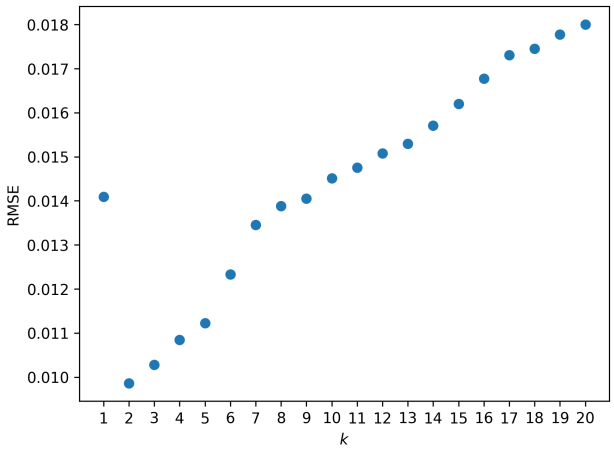
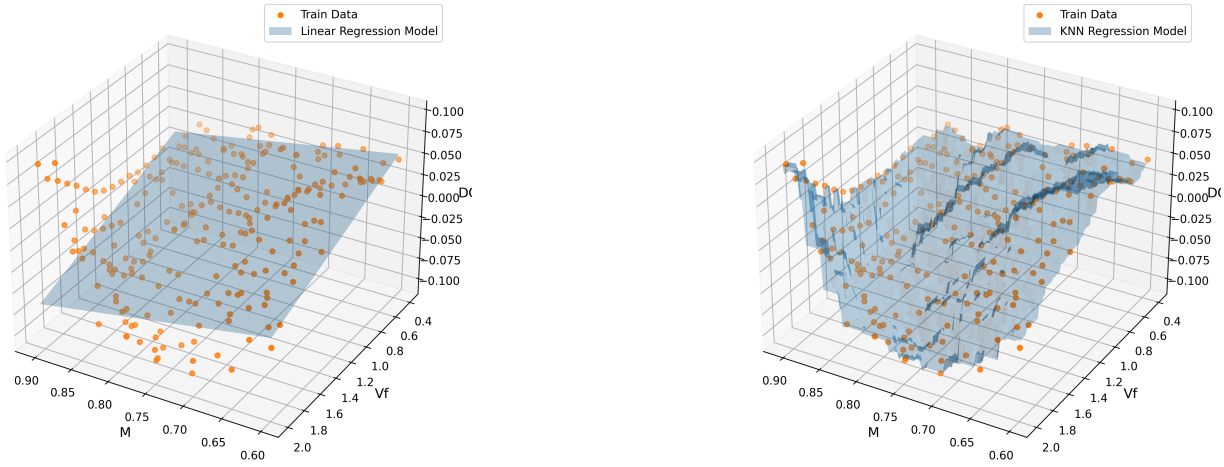


Fig. 17 k hyperparameter-tuning of KNN regression model using RMSE in case 3

Table 10 Coefficient of Determination and Residual Standard Error for Linear Regression Model in Case 3

Score Metric	Value
R^2	0.397773
RSE	0.032493

Now, we may train our KNN regression model. Using 10-folds cross-validation, we obtained the best k value of 2, by taking RMSE as the scoring method. Figure 17 shows the result of the hyperparameter-tuning, and shows that the RMSE is the lowest when $k = 2$. We may plot the result of both linear regression and KNN regression in a form of surface over the range of the training data, which is shown in Figure 18. We can see that the linear regression model is not able to capture the non-linear relation between the inputs and the output.



(a) Linear Regression **(b) KNN Regression**

Fig. 18 Prediction surface of the regression model on training data in case 3 using (a) linear regression and (b) KNN regression

After evaluating each model own parameters, we may now evaluate it with the testing data. The RMSE and MAE of the prediction is shown in Table 11, calculated using `mean_squared_error` and `mean_absolute_error` from

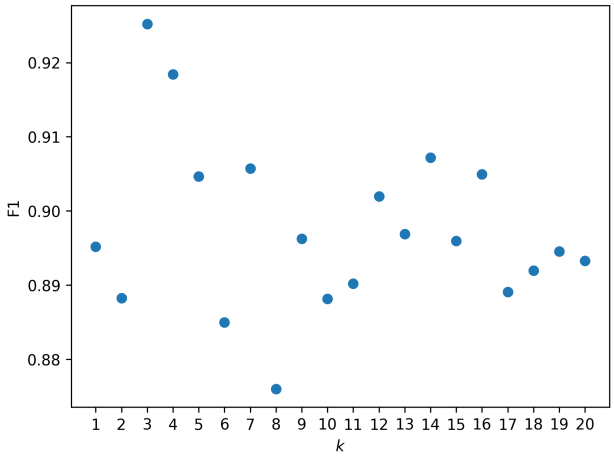


Fig. 19 k **hyperparameter-tuning of KNN classification model using F1-score in case 3**

`sklearn.metrics`. From there, we can see that the KNN regression model is able to predict the damping coefficient better than the linear regression model, as both the RMSE and MAE is lower than the linear regression model.

Table 11 RMSE and MAE of Linear Regression and KNN Regression in Case 3

Model	RMSE	MAE
Linear Regression	0.029969	0.019001
KNN Regression	0.007907	0.005573

Classification Model We will use logistic regression and KNN classification to predict the flutter occurence. Using the algorithm of (26), the resulting coefficient for the logistic regression model and the hypothesis testing is shown in Table 12. We can see that all H_0 is rejected and thus, all of the coefficients are significant.

Table 12 Coefficient for Logistic Regression Model in Case 3

Beta	Value	Standard Error	t-statistic	p-value	Hypothesis Test Result
0	0.158557	0.017985	8.816034	2.555677e-16	Reject H_0
1	-0.145096	0.022156	-6.548710	3.543494e-10	Reject H_0
2	-0.046859	0.004335	-10.808856	2.045755e-22	Reject H_0

Now, we may train our KNN classification model. Using 10-fold cross-validation, we obtained the best k value of 3, taking F1-score as the scoring method. This is because the occurence of flutter is very dangerous, a hazard to the wing design, and thus, we want to minimize the false negative rate. Figure 19 shows the result of the hyperparameter-tuning, and shows that the F1-score is the highest when $k = 3$.

Comparing All Models As we're dealing with a classification problem, we will use the scores of (36), (37), (38), and (39) to evaluate the model. For regression models, we will first classify the predicted value. The scores is calculated using `classification_report` from `sklearn.metrics`. We will also provide the confusion matrix heatmap of the model to visualize more which how many data in each class is predicted right or wrong by the models. The result of the evaluation is shown in Table 13 and Figure 20.

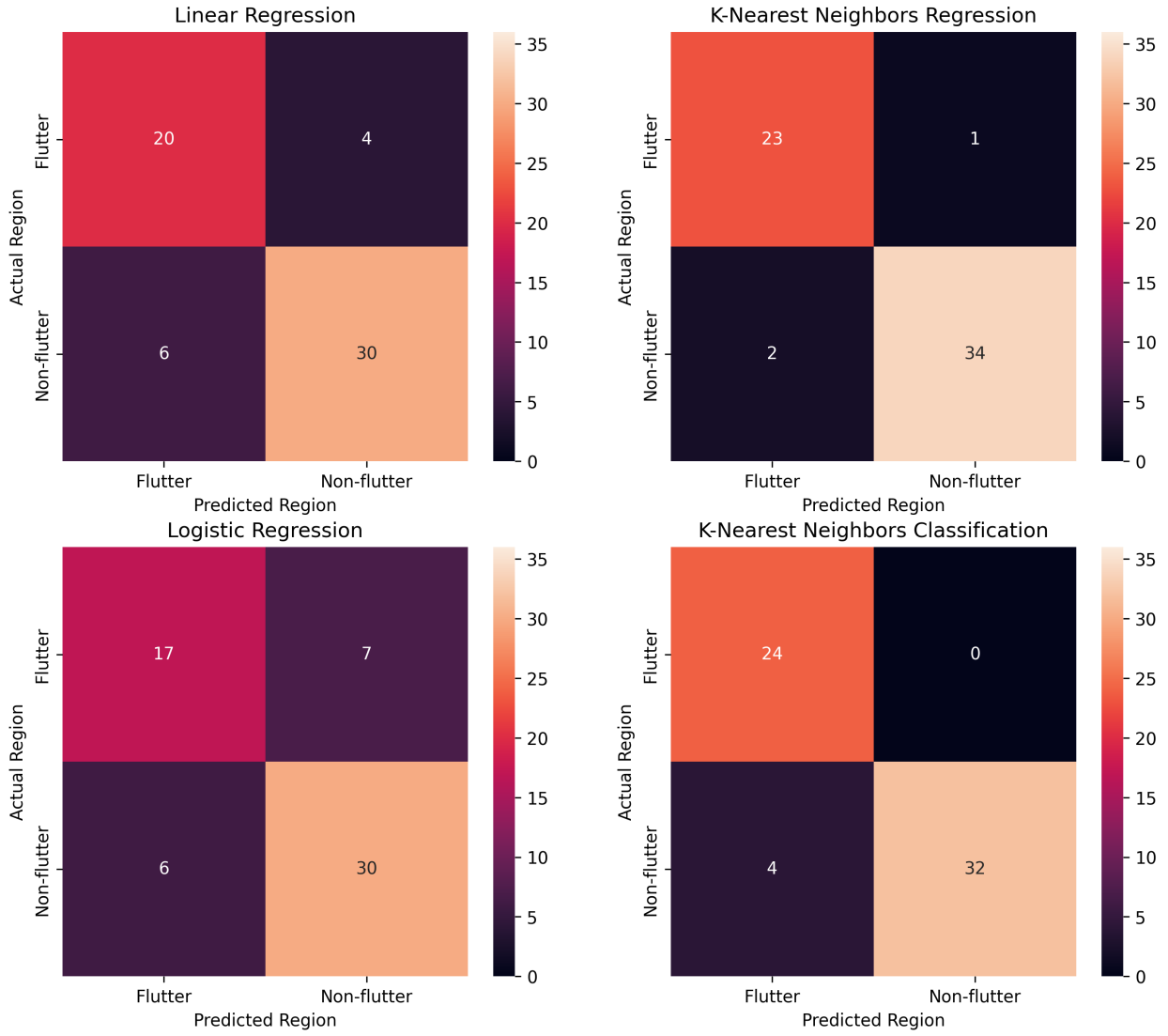


Fig. 20 Confusion matrix heatmap of the regression and classification model in case 3, done on the testing data

Table 13 Classification report of the regression and classification model in case 3, done on the testing data

Model	Class	Precision	Recall	F1-score	Accuracy
Linear Regression	Flutter	0.77	0.83	0.80	0.83
	Non-flutter	0.88	0.83	0.86	
KNN Regression	Flutter	0.92	0.96	0.94	0.95
	Non-flutter	0.97	0.94	0.96	
Logistic Regression	Flutter	0.74	0.71	0.72	0.78
	Non-flutter	0.81	0.83	0.82	
KNN Classification	Flutter	0.86	1.00	0.92	0.93
	Non-flutter	1.00	0.89	0.94	

From the confusion matrix, we can see that both linear and logistic regression model can't handle the false positive case pretty well. Although, linear regression can handle the false negative better than logistic regression. The KNN algorithm models top the parametric models quite well. This is shown from Table 13, where all score metrics of KNN models are larger than those of the parametric models. The KNN regressor has the highest accuracy and F1-score, but the KNN classifier has the highest precision in predicting flutter case and the highest recall in predicting non-flutter case.

Conclusion Our final result is the flutter boundary curve plotted in Figure 21. We can see that the linear regression and logistic regression model has a boundary in form of a line, while the KNN models have boundaries of a complex shape. This is as expected, as the relation between the inputs and the output are shown to be non-linear, KNN as a non-parametric model should "catch" the shape better, than linear regression and logistic regression whcih are parametric models.

For both KNN models, we can see that the region with large Mach is predicted poorly. This is due to the lack of data points provided by the dataset. To improve the result even more, we may find or add more data points to the dataset to increase its accuracy, tune the hyperparameter of KNN models, or even use more models, such as tree-based models and boosting algorithm models.

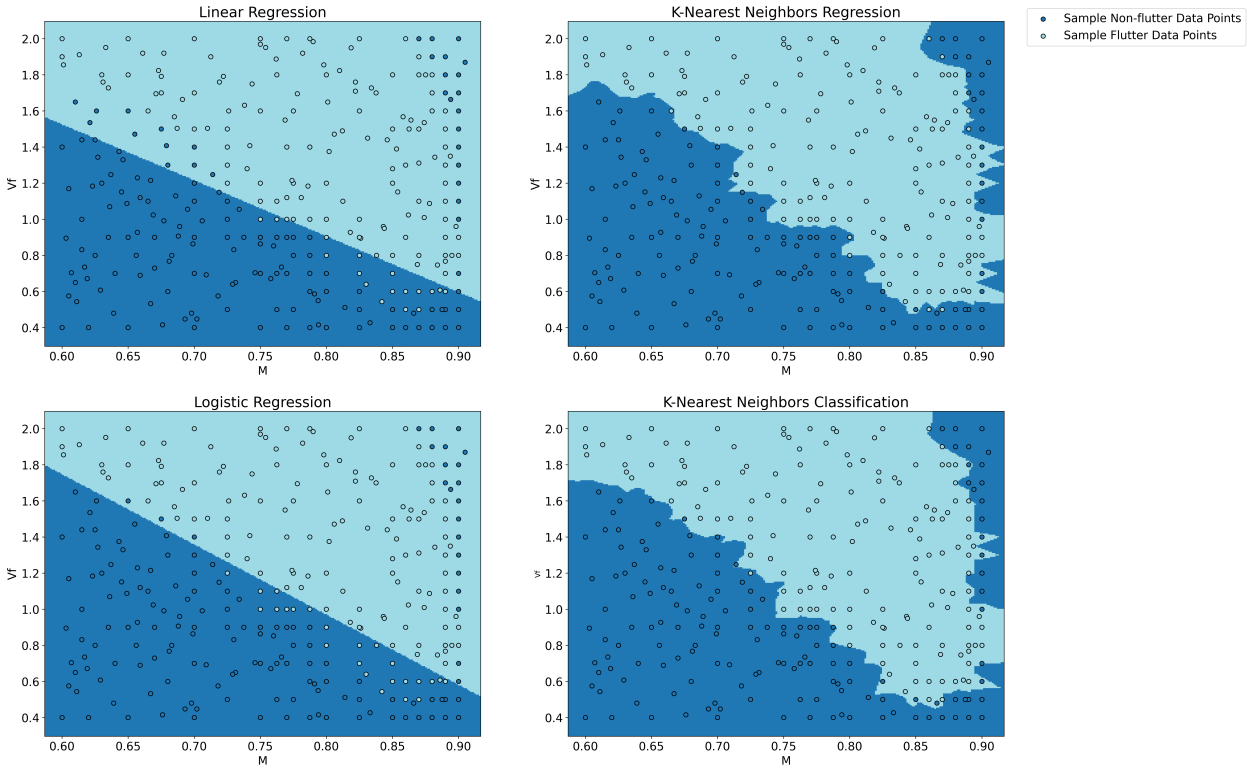


Fig. 21 Flutter boundary of the regression and classification model in case 3

Appendix

Source Code

The source code of this project is available on `https://github.com/hafizh-ender/TugasBesarSDS_Kelompok5`.

Presentation Video

The presentation can be accessed on `https://youtu.be/W9TJMT3xBdI`

Task Division

Table 14 Task Division of the Project

Name	NIM	Task
Rizky Amalis Zhuraida	13620071	Analysis of problem 2, KNN regression part and final conclusion (coding, PPT, and video)
Abisatya Hadyan Dhananjaya	13621046	Analysis of problem 2, linear regression part (coding, PPT, and video) Main video editor
Hafizh Renanto Akhmad	13621060	Analysis of problem 3 (coding, PPT, and video) Write the basic theory of the report Put all answers together in the report
Isna Nur Firdausi	13621078	Analysis of problem 1 (coding, PPT, and video) Initiate PPT slide design Write the introduction in the report

References

[1] James, G., *An Introduction to Statistical Learning with Applications in R*, Springer, 2017.

[2] Fathurrohim, L., Zuhal, L. R., Palar, P. S., and Dwianto, Y. B., "Maximizing the thrust performance of flexible caudal fin panels via experimental optimization," *Ocean Engineering*, Vol. 266, 2022, p. 112969. https://doi.org/https://doi.org/10.1016/j.oceaneng.2022.112969, URL https://www.sciencedirect.com/science/article/pii/S0029801822022521.

[3] Palar, P., Parussini, L., Bregant, L., Shimoyama, K., and Zuhal, L., "On kernel functions for bi-fidelity Gaussian process regressions," *Structural Optimization*, Vol. 66, No. 2, 2023. https://doi.org/10.1007/s00158-023-03487-y, funding Information: The authors acknowledge financial support from Penelitian Dasar Unggulan Perguruan Tinggi research scheme administered by Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, Republic of Indonesia. Publisher Copyright: © 2023, The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature.