ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #35140

# UAV PARAMETER ESTIMATION THROUGH MACHINE LEARNING

**Andres Enriquez Fernandez**

Andres Enriquez Fernandez was born in Ciudad Juarez, Chihuahua, Mexico. He earned his bachelor's degree in Mechanical Engineering at the University of Texas at El Paso. After graduating in Spring of 2006, he started working full-time at an automotive company's technical center as a product development engineer in Ciudad Juarez. While working full-time in Fall of 2017, Andres returned to The University of Texas at El Paso to start the master's graduate school program in Mechanical Engineering. While obtaining his master's degree, he moved to Milwaukee, Wisconsin in 2020 to work full-time at a motorcycle's company development center as a CAE Engineer.

**Dr. Louis J Everett P.E., University of Texas at El Paso**

Dr. Everett is the MacGuire Distinguished Professor of Mechanical Engineering at the University of Texas El Paso. Dr. Everett's current research is in the areas of Mechatronics, Freshman Programs and Student Engagement. Having multiple years of experience in several National Laboratories and Industries large and small, his teaching brings real world experiences to students. As a former NSF Program Director he works regularly helping faculty develop strong education proposals.

**Dr. Miguel Cedeno, The University of Texas at El Paso**

Miguel Cedeno is Adjunct Professor at The University of Texas at El Paso. He received his B.S. in Mechanical Engineering from ESPOCH, his M.S. and Ph.D. in Petroleum Engineering at Missouri University of Science and Technology. His research areas include artificial intelligence, machine learning applied to aerospace and mechanical engineering. He works with CFD applied to refinery equipment design for oil and gas industry. He lectures Thermodynamics, Heat Transfer, Fluid Mechanics, Thermal System Design (Heat Exchanger Design) and VBA Applications for Mechanical Engineers. He is a member of the American Society of Mechanical Engineers (ASME) and the Society of Petroleum Engineers (SPE).

# UAV Parameter Estimation Through Machine Learning

**Andres Enriquez Fernandez, MS**
Master of Science in Mechanical Engineering Candidate
University of Texas at El Paso

**Miguel Cedeno, PhD**
Graduate Advisor, Department of Mechanical Engineering
University of Texas at El Paso

**Louis J Everett, PhD**
Graduate Advisor, Department of Mechanical Engineering
University of Texas at El Paso

## Abstract

Parameter identification of Unmanned Aerial Vehicles (UAV) is very helpful for understanding cause-effect relationships of physical phenomenon, investigating system performance and characteristics, fault diagnostics, control development/tuning, and more. Traditional ways of performing parameter identification involve establishing a mathematical model that describes the system's behavior. The equations in the model contain parameters that are estimated indirectly from measured flight data. This parameter identification process requires knowledge of the physics involved. Also, it necessitates a careful consideration of the aircraft instrumentation for accurate measurements. It also requires careful design of the flight maneuvers to ensure thorough excitation of the flight dynamics involved. Finally, one must select a suitable identification method. The purpose of this paper is to show the application of machine learning for parameter identification of a UAV model. The machine learning algorithm does not require developing parameterized models; hence it is an equation-less identification method. To provide input to the system, a simulation model of the aircraft is generated. The parameters of the model can be modified in the simulation. The aircraft flight measurement data is obtained directly from the model as simulation outputs from a predetermined flight path. The data is submitted to a machine learning algorithm that can read and recognize the data. The machine learning algorithm is trained with a set of flight data that incorporates variations in the parameters to be identified. Finally, the algorithm is tested by feeding unknown flight data to predict the output.
To achieve autonomous and consistent flights, a Software-In-the-Loop (SIL) simulation is constructed between X-Plane and Mission Planner. X-Plane is a realistic flight simulator where the UAV model is created, and flight physics are modeled. Mission Planner is the Ground Control Station (GCS) that generates and sends the flight commands to be executed in X-Plane. Several machine learning regression models are explored including linear regression, regression trees, gaussian process regression, support vector machine and ensembles of regression trees.

*Proceedings of the 2009 ASEE Gulf-Southwest Annual Conference*
*Baylor University*

## Introduction

Some of the biggest challenges in current applications of Unmanned Aerial Vehicles (UAVs) include safe operation, communication between manned and unmanned aircrafts, and robust control systems. Control system design typically needs an accurate model of the aircraft. To increase the model accuracy, model parameters need to be estimated. Dynamic aircraft systems have traditionally been estimated analytically from Newton's second law for rigid-body dynamics (Hoffer, 2014). These dynamic system models have been usually obtained through wind tunnel testing. Some important limitations of wind tunnel tests include the high costs involved, test equipment interface interactions (Bhavithavya, 2013), and limitation on flight regimes. System identification or system ID is an alternative to wind tunnel type model estimation (Bhavithavya, 2013). System identification involves parameter estimation to determine a mathematical model. These parameters are estimated indirectly from measured flight data. This parameter estimation process requires a careful consideration of the aircraft instrumentation for accurate measurements. It also requires careful design of the flight maneuver to ensure thorough excitation of the dynamics. Finally, one must select a suitable identification method.

The purpose of this paper is to show the application of machine learning for parameter identification of a UAV model. The machine learning algorithm does not require developing parameterized models; hence it is an equation-less identification method of an aircraft. To avoid the expense of crashing a real UAV, a simulation model of the aircraft is generated. The parameters of the model can be modified in the simulation. The aircraft flight measurement data is obtained directly from the model as simulation outputs from a predetermined flight path. The data is submitted to a machine learning algorithm that can read and recognize the data. The machine learning algorithm is trained with a set of flight data that incorporates variations in the parameters to be identified. Finally, the algorithm is tested by feeding unknown aircraft data and comparing the prediction of the machine learning algorithm to the known answers.

To obtain simulation data through autonomous UAV operation, a Software-In-the-Loop (SIL) simulation is constructed. In this case, the SIL is created by interconnecting the simulation software known as X-Plane, and the Ground Control Station (GCS) known as Mission Planner. X-Plane is a realistic flight simulator where the UAV model is generated and flown. Mission Planner is a GCS, which for this case, encompasses a software that allows the operator to have control of the UAV autonomously.

## Software In-The-Loop

To control the UAV autonomously and consistently though a predefined path, a Software-In-the-Loop simulation is generated. A SIL allows the evaluation of flight controls and guidance algorithms (Bittar, Figuereido, Guimaraes, & Mendez, 2014) without any hardware. Before having access to simulations, engineers and researchers had to completely rely on actual prototypes. Making a change to any of the aircraft's flight parameters involved replacing actual prototype components. Performing a SIL simulation offers advantages in the time required to perform design

improvements and safety by reducing experimental flight tests (Bittar, Figuereido, Guimaraes, & Mendez, 2014). SIL simulations also reduce project's overall cost by avoiding UAV's potential destructive events like crashes. To generate the SIL simulation for this work, the main two components involved are X-Plane and Mission Planner.

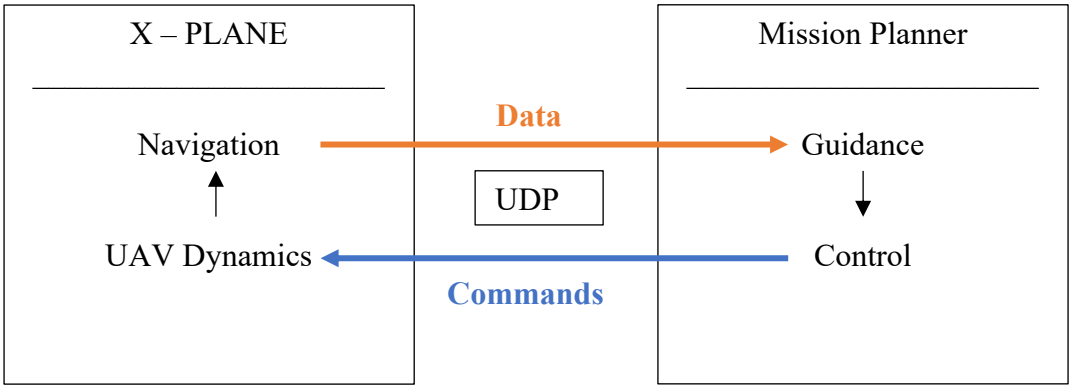


Figure 1. X-Plane – Mission Planner SIL Architecture (Bittar, Figuereido, Guimaraes, & Mendez, 2014)

Figure 1 illustrates the architecture of the SIL generated between X-Plane and Mission Planner. Data packages between X-Plane and Mission Planner are sent via User Datagram Protocol (UDP). Mission Planner evaluates and assigns control commands for the guidance of the aircraft. X-Plane receives commands from Mission Planner and performs the requested control surfaces deflections. Then, X-Plane runs the model physics and sends the generated navigation data to Mission Planner. The data is then exported from Mission Planner to MATLAB. MATLAB is employed to perform all the preprocessing of the data that will be feed into the machine learning algorithms. Data preprocessing includes raw data cleaning that include removing outliers, redundant, and filling missing data. Data preprocessing also includes structuring data appropriately for algorithm readability such as input features and labeled output formats. MATLAB is also used to create and evaluate the performance of different machine learning models utilizing built-in machine learning regression tools such as the Regression Learner App. MATLAB's Regression Learner App is used to predict data by training regression models. Available regression models include linear regression, regression trees, gaussian process regression, support vector machines and ensembles of regression trees ("Train Regression Models," n.d.)

## Machine Learning

Machine learning deals with self-identification of patterns using statistical theory embedded in learning algorithms. In the context of machine learning, learning is the process of finding patterns in data (Nasteski, 2017). Depending on the desired algorithm outcome, machine learning algorithms are organized in two main groups, supervised and unsupervised learning. Supervised learning is applied when there is known or labeled output data that the algorithm is trained to predict. The objective of supervised learning is to generate an artificial system that is trained to learn the mapping between the system's input and outputs, and to predict the outputs from a different set of input data that is unknown to the algorithm (Lui & Wu, Y, 2012). For this

application is focused on supervised learning since the desired output known and it is part of the training data. Linear regression through Machine Learning provides an alternative to model building through mathematical equations. Machine Learning techniques can be used to estimate a set of model parameters at once without specific physical knowledge of the system to be estimated (Gabrielli, 2017). In the next section, the most broadly used supervised machine learning models in this work, namely, Support Vector Machine (SMV) and Ensembles of Regression Trees, are introduced.

## Linear Regression

Linear regression in the context of Machine Learning has the primary objective of modeling relationships between dependent and independent variables (Nasteski, 2017). Consider a set of independent variables $x$ as a subset of $\mathbb{R}^d$($d$-dimensional vector), and a set of known or labeled data $y$ as a set of real numbers. Linear regression can be represented as follows.

$$L_d = \{\boldsymbol{x} \mapsto \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b : \boldsymbol{w} \in \mathbb{R}^d, b \in \mathbb{R}\} \quad \text{Eq. 1}$$

Where $L_d$ are functions parametrized by $\boldsymbol{w} \in \mathbb{R}^d$and $b \in \mathbb{R}$, $\langle \boldsymbol{w}, \boldsymbol{x} \rangle$ is the inner or dot product of $\boldsymbol{w}$ and $\boldsymbol{x}$ (Shalev-Shwartz & Ben-David , 2014). The loss function for *h(x)* predictions is defined as:

$$l(h) = \frac{1}{m}\sum_{i}^{m}(h(x_i) - y_i)^2 \quad \text{Eq. 2}$$

## Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning model for data classification and regression. SVM models a linear prediction over mapped samples to a higher dimensional space (Verrelst, et al., 2012). The objective in SVM models is to generate a high-dimensional surface or hyper-plane that can separate data categories (Rodriguez-Galiano, Sanchez-Castillo, Chica-Olmo, & Chica-Rivas, 2015). Given a decision tolerance $\epsilon$, the main objective of Support Vector Regression (SVR) is to find a hyperplane where all data points lie inside the decision surfaces at a distance $\epsilon$ from the hyperplane (Smola & Vishwanathan, 2008).

$$|y_i - (\langle \boldsymbol{w}, x_i \rangle + b)| \leq \epsilon \quad \text{Eq. 3}$$

The Loss Function of Support Vector Regression is given by (Smola & Vishwanathan, 2008):

$$l(w, x, y) = max(0, |\boldsymbol{y} - \langle \boldsymbol{w}, \boldsymbol{x} \rangle| - \epsilon) \quad \text{Eq. 4}$$

## Regression Trees

On a Regression Tree, regression models are fitted to the terminal nodes or leaves. The goal is to arrive to the appropriate leaf to make the prediction (Loh, 2011). The process starts by splitting the root node in such a way that the resulting child nodes are 'less impure' or have less variation in the observations than the parent node. The objective is to minimize the 'impurity' of the tree by performing the optimal node splits (Rodriguez-Galiano, Sanchez-Castillo, Chica-Olmo, & Chica-Rivas, 2015). There are several impurity measures, and the discussion is beyond the scope of this work (Rodriguez-Galiano, Sanchez-Castillo, Chica-Olmo, & Chica-Rivas, 2015) and (Loh, 2011).

## Ensembles of Regression Trees

Support Vector Machine (SVM) is a supervised machine learning model for data classification and regression. SVM models a linear prediction over mapped samples to a higher dimensional space (Verrelst, et al., 2012). The objective in SVM models is to generate a high-dimensional surface or hyper-plane that can separate data categories (Rodriguez-Galiano, Sanchez-Castillo, Chica-Olmo, & Chica-Rivas, 2015). Given a decision tolerance $\epsilon$, the main objective of Support Vector Regression (SVR) is to find a hyperplane where all data points lie inside the decision surfaces at a distance $\epsilon$ from the hyperplane (Smola & Vishwanathan, 2008).

## Flight Simulation Data Repeatability

To assess the repeatability of the flight simulation data, three consecutive flights were performed with the same UAV model. The test flights were consecutive and used the same flight path from Mission Planner. The flights were conducted at the same location under the same environmental conditions. Once plotted together, the flight curves should overlap since the simulated flight conditions are identical between flights and no variation due to these simulated conditions is expected. Once the model is found to be able to provide repeatable results, flight conditions can be altered with more confidence to incorporate real life condition variations in future studies. Incorporating flight condition variation is outside the scope of this work. An initial inspection of the data revealed an undesired source of variation. It was observed that the start of data recording before takeoff varies from flight to flight. This caused varying amounts of unnecessary "dead time" data having values of zero across all the flights features. Figure 2 shows three sequential runs of the same UAV.

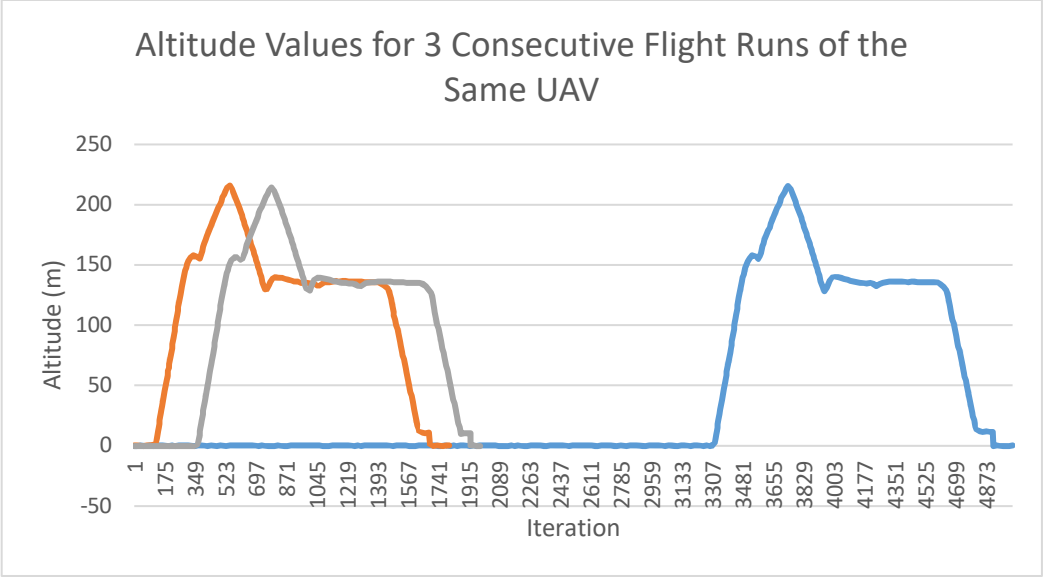


Figure 2: Altitude Values for 3 Consecutive Flight Runs of the Same UAV.

As it can be seen in the figure above, the three flights show very similar patterns. However, it is unclear to see if the flights show any variation between them. This is because it is difficult to compare curves when they do not overlap. If not addressed, besides making it difficult to compare graphs for repeatability, this will also have at least two undesired effects in the learning process. Firstly, it will affect the performance of the learning algorithm since it will be trying to find relations in sections that do not have any effect on flight performance. Secondly, it increases the size of the data files. As a result, the algorithm will take longer to process the data delaying the learning process. To solve this issue, the data leading to the initiation of the take-off maneuver was eliminated. Figure 3 shows altitude values for three consecutive flight runs of the same UAV after removing the data before take-off.

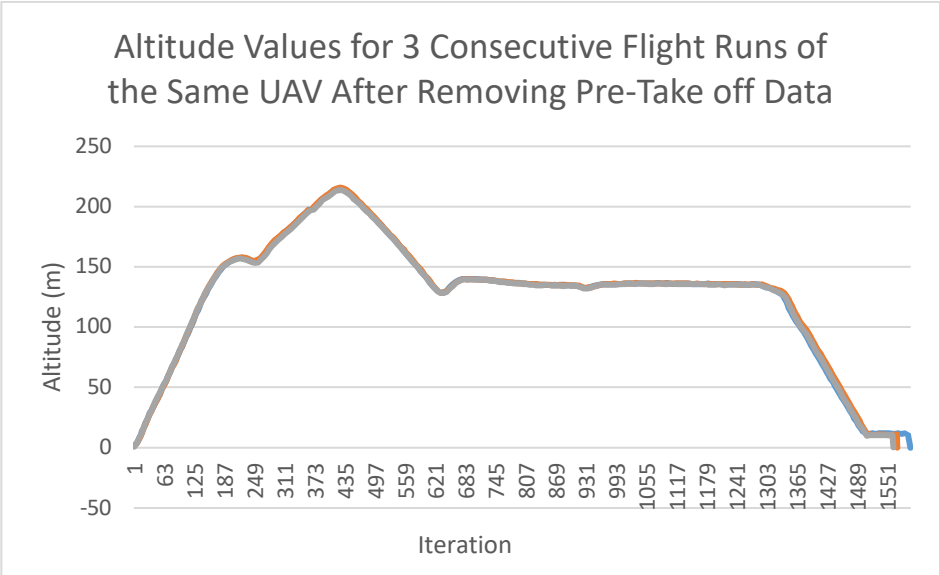


Figure 3: Altitude Values for 3 Consecutive Flight Runs of the Same UAV After Removing Pre-Take off Data.

Once the pre-take-off data is removed, the beginning point of all flight curves is set at the start of the take-off maneuver. The result is plotter in Figure 3 where it becomes clear that the curves virtually overlap as expected. There is variation at the last curve section, and it could be fixed by eliminating datapoints after the UAV has already landed. There is a sudden drop that occurs when the drone is disconnected at the end of the simulation. This data can be disregarded since it has no effect on the flight performance. For further pre-take off data removing, the air-speed measurement output is employed. The reason is that once the UAV starts gaining speed, the air-speed measurements will start increasing from a constant near zero value, making it easy to identify for preprocessing.

## Feature Reduction

The objective in feature reduction is to find relevant features that are found to have the greatest impact in the algorithm learning process. There are two main approaches in feature reduction according to (Kotsiantis, Kanellopoulos, & Pintelas, 2006), namely, filter and wrapper. Filter performs feature reduction based on discriminating criteria independent of the Machine Learning algorithm (Corrales, Corrales, & Ledezma, 2018). On the other hand, wrapper methods use the Machine Learning model to evaluate and select the appropriate features (Kotsiantis, Kanellopoulos, & Pintelas, 2006). For this work, the wrapper approach is adopted where several features are used to train the regression model and prediction tests are performed to assess the model. To apply the wrapper reduction approach; barometric altitude, roll, pitch, yaw and airspeed data is fed individually to train an SVM regression model. Eighteen flight simulations were performed where the only difference between flights, is the aircraft's weight. The flights are conducted in the same location under the same wind and temperature conditions. The goal of the algorithm is to predict the weight of the UAV. Among the different parameters that can be predicted, weight is selected as the best candidate due to several reasons. The main reason being that weight is independent of other parameters, meaning that the aircraft model weight can be modified without affecting any other aircraft component or flight parameter. That is not the case with some of the parameters, for example, changing the airfoil shape also affects the lift, drag and moment coefficients. Having these interactions can be distracting and add unnecessary sources of variation that will increase the complexity of the model. Another important reason for selecting the weight is that it can be easily and directly modified in the parameters file of the model. Unlike other parameters, the weight is modified simply by changing the density value directly. This is very convenient and efficient when creating model variations. Other parameters, when modified in the model, create changes in many variables within the parameter file. Finding the relation between these variables adds complexity to the model. Also, model preparation time can increase significantly as well due to the unknown relations between these model parameter variables. For this work, the main objective is proving if the proposed use of machine learning for parameter estimation is feasible. Modifying the weight alone is sufficient to find initial challenges and limitations of the method. Furthermore, it can provide information on which models are more appropriate to test more complex parameters in the future. First, the feature data is fed to the regression model for training. Next, the model's prediction is tested with a different set of input data. To assess the quality of the predictions, the Mean Squared Error (MSE) is obtained for each model. The MSE is defined as the sum of the difference between the known *y's* and the model predictions squared divided by the number of instances. In this case, the know *y's* are the known

UAV weights, and the model prediction is the weight output from the trained algorithm. Table 1 shows the weights used for the test.

Table 1: UAV Weight Variations.

| Weight (lb.) | Weight (lb.) | Weight (lb.) |
|---|---|---|
| 0.95 | 12.35 | 23.75 |
| 2.85 | 14.25 | 25.65 |
| 4.75 | 16.15 | 27.55 |
| 6.65 | 18.05 | 29.45 |
| 8.55 | 19.95 | 31.35 |
| 10.45 | 21.85 | 33.25 |

Once the flight data is available from the simulation runs, the SVM model is trained in MATLAB. The data is split between training and testing randomly in a ratio of 89% training and 11% testing. The model's MSE is then computed for each feature and compared against the other models. Due to the randomization of the training, six runs will be performed for each feature. Figure 4 below displays the results.

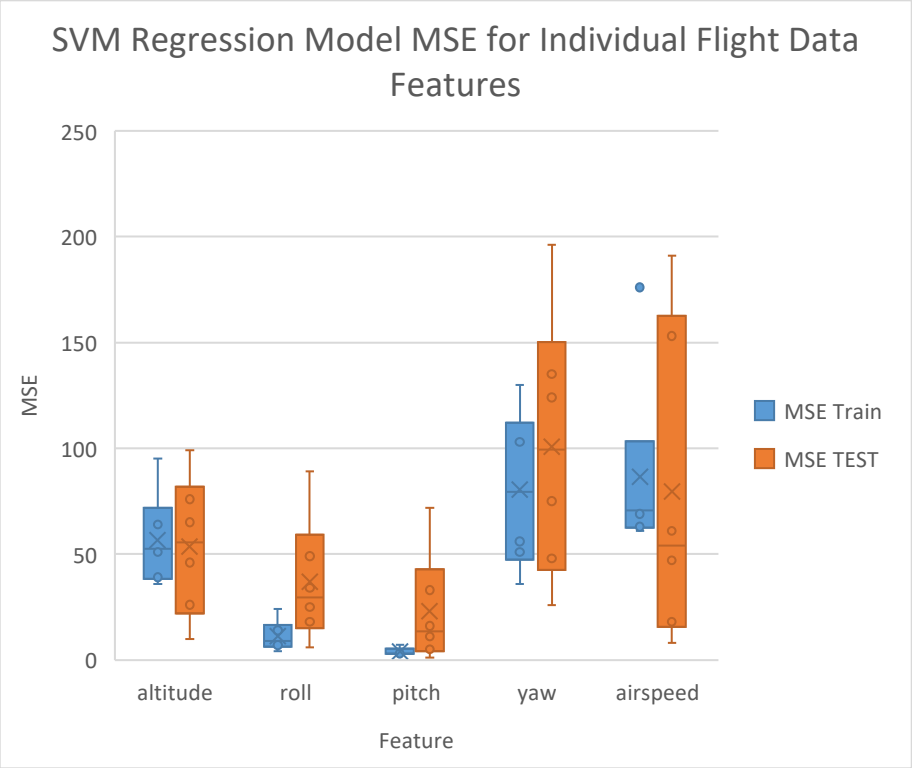

Figure 4: SVM Regression Algorithm MSE for Individual Flight Data Features.

Figure 4 shows a box and whisker plot of the results obtained. The box and whisker plot shows the median of the data in the center of the box. The box limits represent the lower and upper quartiles of the data. Finally, the whiskers represent upper and lower extreme data values. The blue boxes correspond to the model's MSE for the training and the orange boxes correspond to the test MSE. Yaw and airspeed show a high mean MSE values for test. This is referred to as the models having high variance. This means the model's predictions to new sets of data is not accurate compared to the other models with lower variance. Models for roll and pitch show small MSE values for training. This is defined as low bias. Then, we can say that the models for roll and pitch show low bias. However, it can also be observed that both roll and pitch models have larger variance than bias. Having high variance and low bias is a sign of an overfit model. Having a good model requires to have a balance of a low bias with a low level of variance. All models show high levels of variance. Altitude shows good balance between bias and variance. However, it is desirable to decrease both to get a model with good predicting accuracy. Roll and pitch models are overfit with the employed SVM algorithms. Results of this study show that the altitude, roll and pitch show smaller level of variance compared to yaw and airspeed. For purposes of showing whether the ML tools can be employed for parameter estimation of flight systems, the altitude, roll and pitch will be prioritized over yaw and airspeed.

## Results and Discussion

Feature reduction yielded that the most significant features affecting the regression model are altitude, roll and pitch. In this section, MATLAB's regression learner tool results are presented for altitude. Data partitioning is performed to the altitude input data for each flight. Also, curves for

altitude's running average for the past 20 timesteps are included to incorporate the time behavior aspect of the data in the learning algorithm. The data is divided into four, two thousand datapoint sections. The input data for training comes from eighteen flights corresponding to eighteen weights, see Table 1.The input data contains one hundred and forty-four curves in total.

## Altitude – SVM Optimizable Regression

The first regression model utilized for altitude data is an SVM regression optimizable model from MATLAB's regression learner. Prior to training the model, the input data is split randomly between testing and validation or testing. For this model, twenty-five percent of the data was saved for model validation. The prediction results are shown in Figure 5 below.

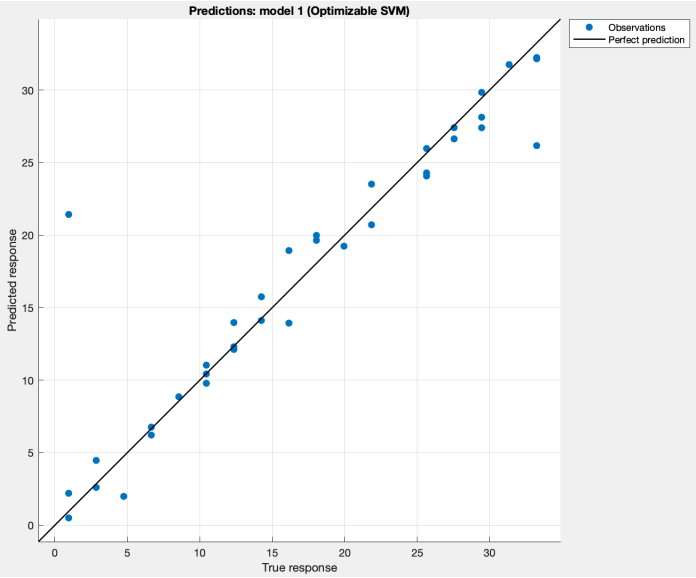

Figure 5: Altitude SVM Optimizable Prediction Plot.

The model can fit the test data with a MSE of 14.6. Some outlier points are spotted in the data at low weight values. To test the model, two additional flights that are not part of the model's input are generated separately. In other words, the data from these two flights is unknown to the regression model. To perform the testing of the model, a section of the flight is fed to the algorithm and the weight prediction is obtained. Figure 6 shows the prediction results for each flight section.

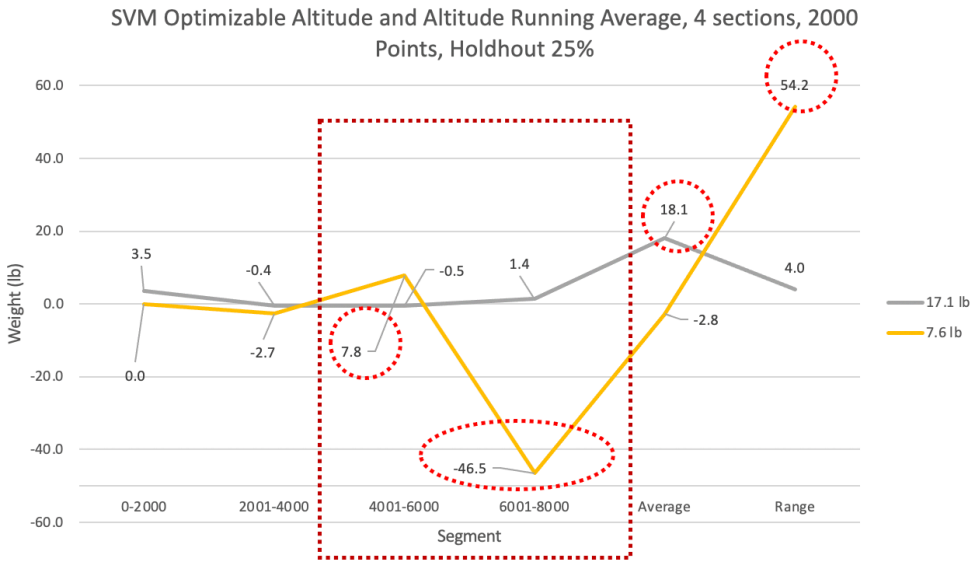


Figure 6: Altitude SVM Optimizable Model Test Results Plot.

The plot shows curves of delta values between the known weight and the model prediction. The first four sections show the deltas for the four, two-thousand data sections of each flight. After the four sections, the average of the weight predictions for the four previous sections is plotted. Finally, the plot shows the range of the prediction computed by subtracting the minimum delta value from the maximum delta value from the four flight sections. As it can be seen in the plot in Figure 6, the results show that the model is able to predict the new UAV weight within 3.5 lb. of the actual value for the first two sections. However, the model fails to predict the weight accurately for the last two sections of the flight. This affect's the overall predictability of the model when the average of the predictions is calculated. SVM regression employs hyperplanes to make predictions. This could explain why the model is uncapable of making accurate prediction across all data as the data on these flight sections might lie outside the model's decision hyperplanes. To improve the regression model prediction across all sections, ensembles of regression trees are explored in the next section.

## Altitude – Ensembles of Regression Trees

The SVM model employed in the last section failed to accurately predict the UAV's weight for all flight data partitions. To improve the prediction on all sections, ensembles of regression trees are now employed. As in the previous SVM model, the data is split between training and testing/validation. The same twenty-five percent of the data is held for validation of the model. Figure 7 contains the result of the model.

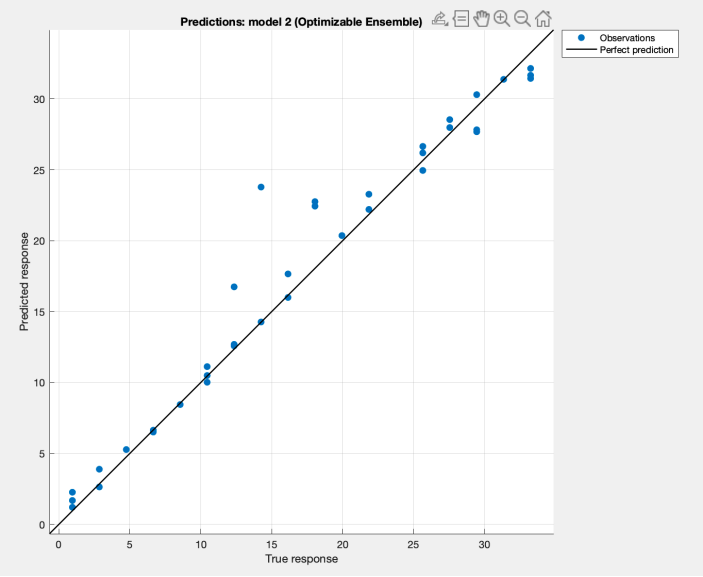

Figure 7: Altitude Ensemble of Trees Optimizable Prediction Plot.

The model fits the test data with a Mean Square Error MSE of 4.9. The plot shows an improvement compared to the SVM model with some points lying far from the perfect prediction line. To test the model, predictions are made to flight data that is unknown to the algorithm. The results are plotted in Figure 8 below.

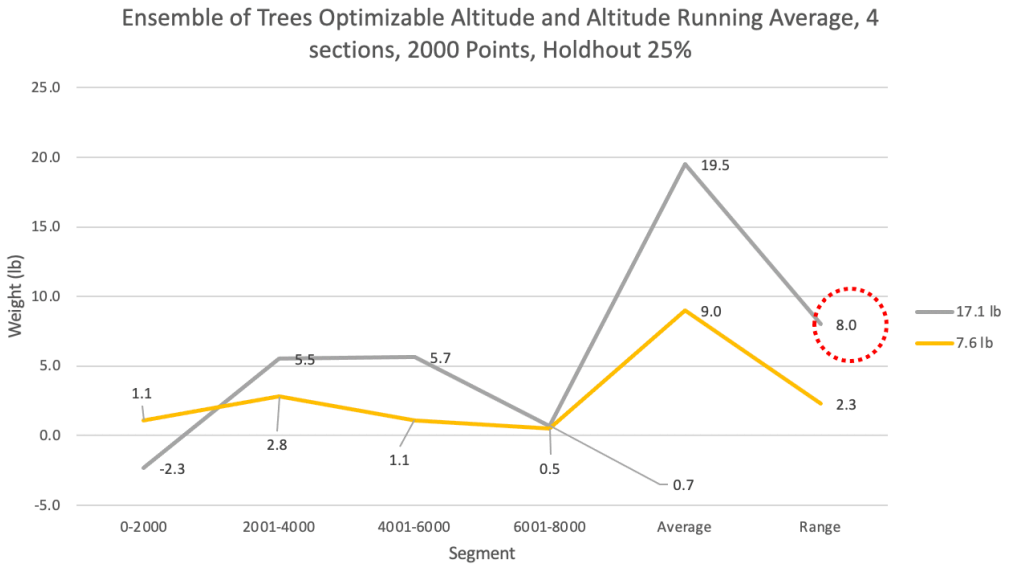

Figure 8: Altitude Ensemble of Trees Optimizable Model Test Results Plot.

As it can be observed in the plot above, the prediction results are more consistent across all flight data sections. The maximum delta is 5.7lb between predicted and actual values. This improves the overall average of the results. The predicted weight for the 17.1 lb. UAV is at 19.5 lb. and at 8.0 lb. for the 7.6 lb. UAV. Although the overall averages are close for both flights, the variation of the predictions for the different sections is not ideal. The validation of the training model has been performed with a twenty-five percent holdout for the past two models. An

alternative to holdout validation is cross validation. The main objective of cross validation is to optimize the training of the algorithm by selecting the best train/validation split as part of the model training process. In the next section, the same ensemble of trees used is trained with cross validation to evaluate the effect in the variation of the predictions.

## Altitude – Ensembles of Regression Trees with Cross Validation

In this section the results of ensembles of regression trees with cross validation is presented. An ensemble of regression trees model is selected since it yields consistent results across all flight data sections. Three different models are trained and tested for altitude data. Each of the three models is validated with a different amount of cross validation folds, five, ten and fifteen. Figure 9 summarizes the results.

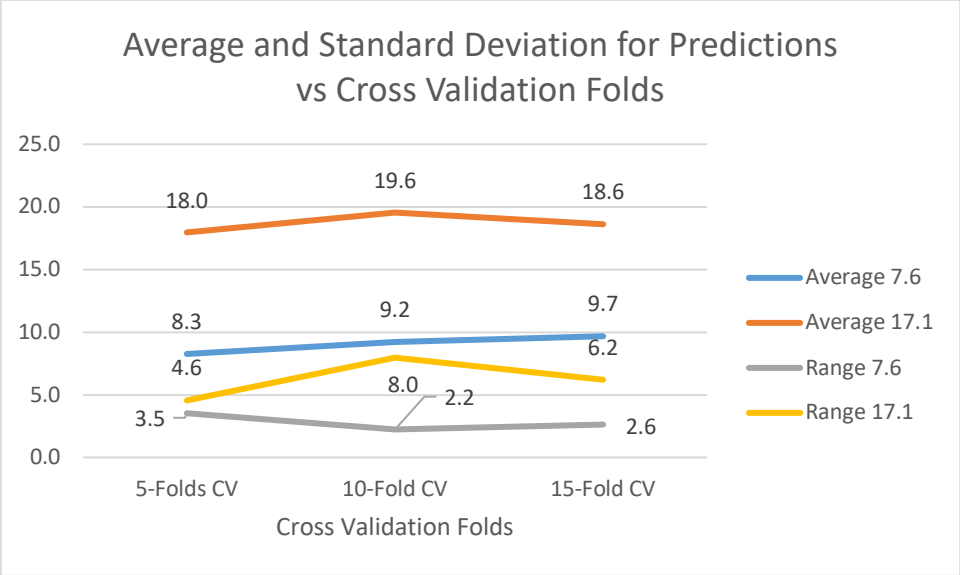


Figure 9: Average and Range for Prediction vs Cross Validation Folds.

The plot contains the average and range values of three repetitions for each fold-number value tested. This is done for both 7.6 lb. and 17.1 lb. UAV weights. The results indicate that after five folds, the predictions do not show a significant improvement. It is noted that for increased folds, the models tend to overpredict without much improvement in the range of the predictions. To save training run time a value of five folds is adapted for the remainder of this work.

## Altitude – Ensembles of Regression Trees with 5 Folds Cross Validation

Results for altitude are presented in this section. The results are obtained from an ensemble of regression trees optimizable model with a cross validation with 5 folds. The predicted versus true response plot is shown in Figure 10.

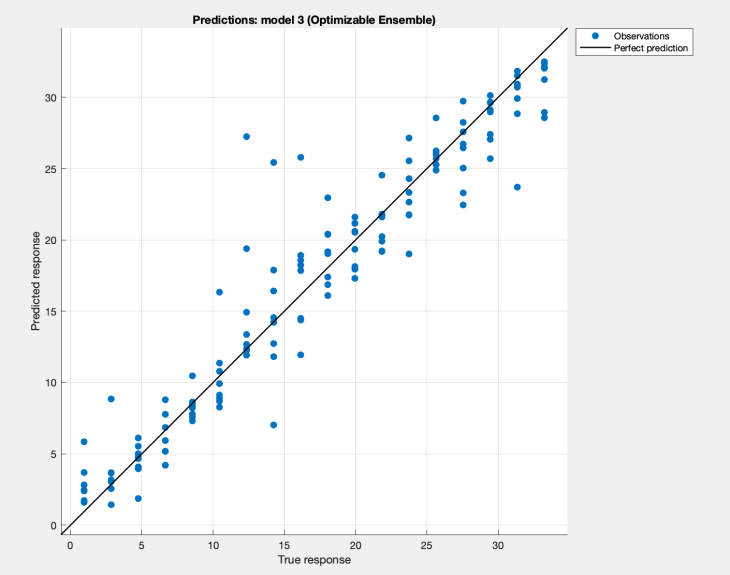


Figure 10: Altitude Ensemble of Trees Optimizable Cross Validation 5 Folds Prediction Plot.

The MSE achieved for the data from this model is at 8.3. The prediction shows a balanced distribution around the ideal line with some scatter in the mid-range of the weight. The summary results of the test flight are shown below in Figure 11

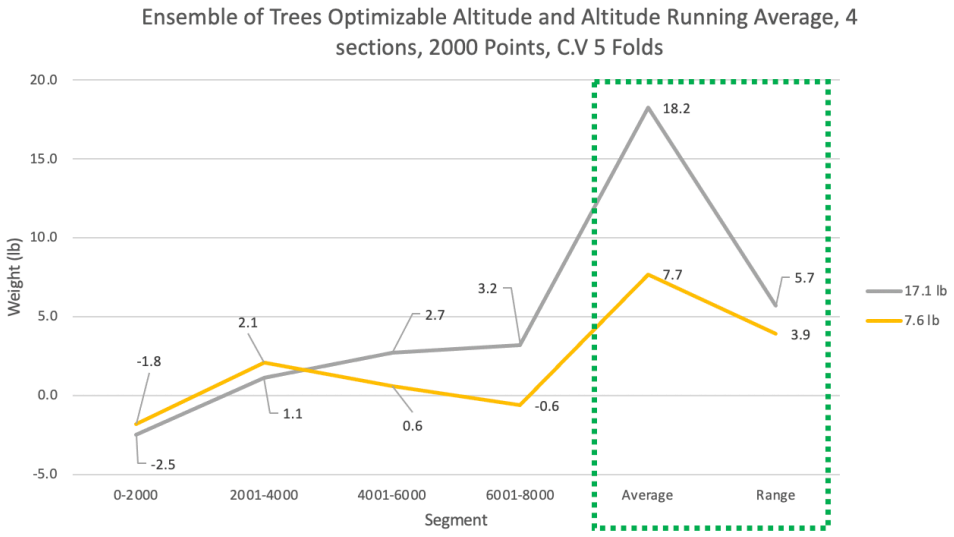


Figure 11: Altitude Ensemble of Trees Optimizable Model Cross Validation 5 Folds Test Results Plot.

The results show an improvement in both the average weight and range of the predictions. The range across the data section lies within 3.2 lb. with an overall range of 5.7 lb. for the 17.1 lb. UAV. For the 7.6 lb. UAV, the range is within 2.5 lb. across the data sections and below 4 lb. in the overall range. The average prediction values are at 7.7 lb. and 18.2 lb. for the 7.6 lb. and 17.6 lb. weights respectively. In the following sections, the same approach is applied to the roll and pitch features.

## Roll Results

In this section, the roll feature results are presented. Similar to altitude, the model used for regression is ensemble of trees optimizable model with cross validation and 5 folds. The same data format used in the altitude predictions is used for roll. The model prediction vs real values plot is shown in Figure 12.

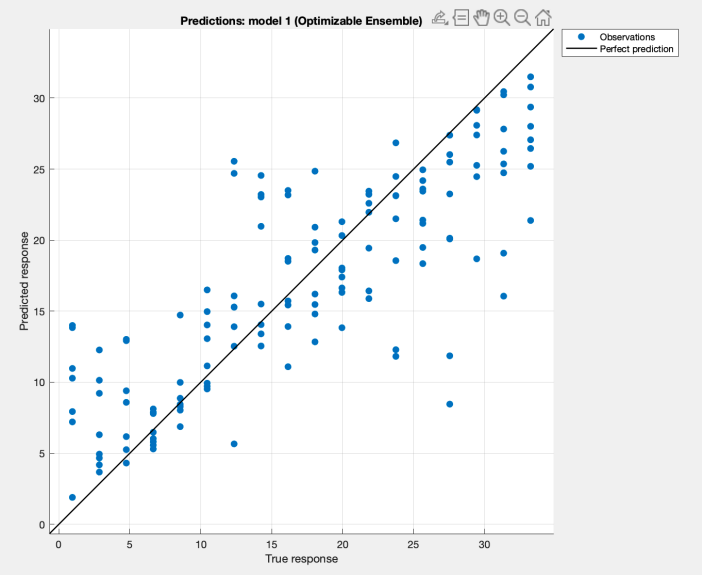


Figure 12 Roll Ensemble of Trees Optimizable Cross Validation 5 Folds Prediction Plot.

The predicted vs true response plot shows large spread of the predictions around the ideal line. This is reflected in a high resulting MSE for the model of 32.4. The prediction for new flight data is summarized in Figure 13.

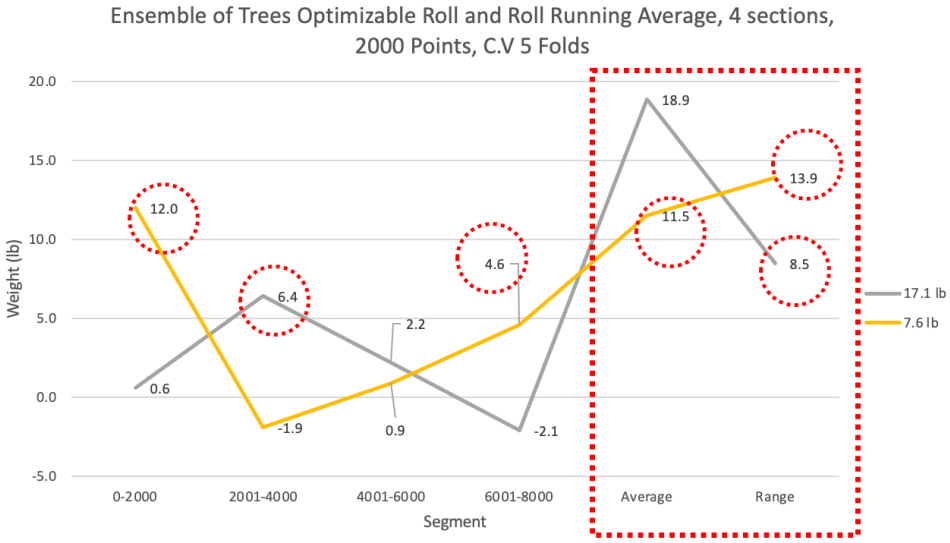


Figure 13: Roll Ensemble of Trees Optimizable Model Cross Validation 5 Folds Test Results Plot.

The model predictions show large delta values across all flight data sections. The results indicate that the roll feature model is not capable of making accurate weight predictions compared to altitude. The average prediction for the 7.6 lb. UAV is at 11.5 lb. This is shown in the high range value of 13.9 lb. across sections. The average prediction for the 17.1 lb. is at 18.9 lb. Although the prediction magnitude is close to the actual value, the range across flight data sections is still high at 8.5 lb. Next, the pitch feature is tested with the same regression model and results are discussed in the next section.

## Pitch Results

Lastly, the pitch feature is evaluated individually with the same model and data input structure used for the altitude and roll features. Again, the regression model used is ensemble of trees optimizable with cross validation and 5 folds. First, the resulting plot of predicted vs actual response is shown in Figure 14.

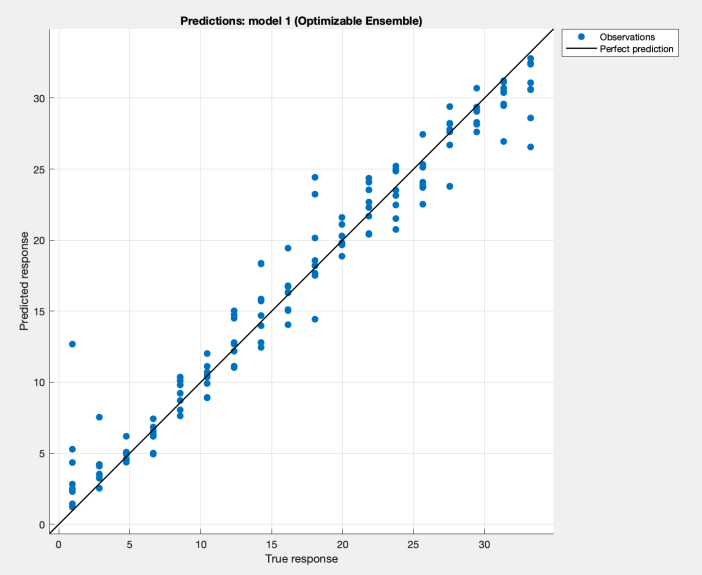


Figure 14: Pitch Ensemble of Trees Optimizable Cross Validation 5 Folds Prediction Plot.

The plot shows improved fitting compared to the roll feature. The resulting MSE for pitch is at 4.4. This shows in a better and narrower data-points distribution around the ideal values line in the plot. As in previous models, a test is conducted with unknown flight data for the algorithm. The results are shown in Figure 15

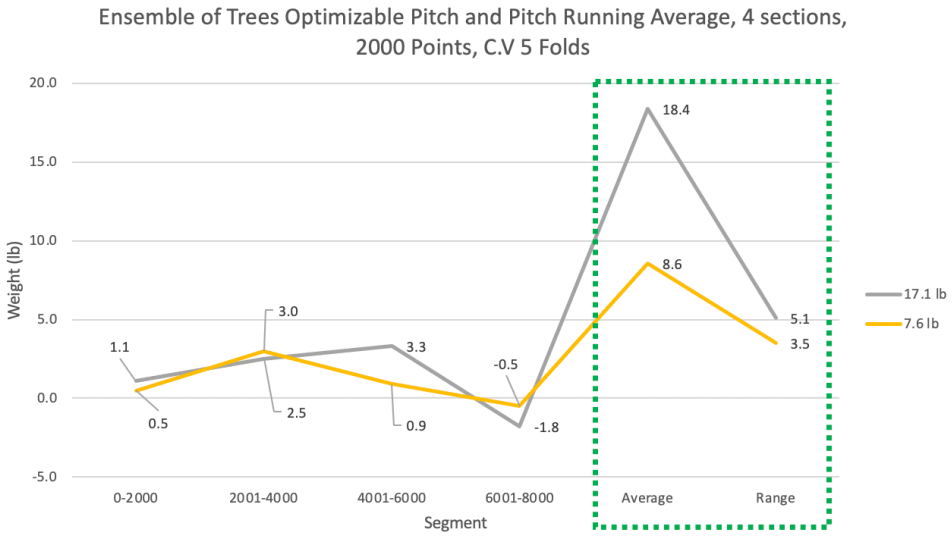


Figure 15: Pitch Ensemble of Trees Optimizable Model Cross Validation 5 Folds Test Results Plot.

The resulting model can predict weight within 3.3 lb. across all flight data sections. The average predicted weights are 18.4 lb. for 17.1 lb. and 8.6 lb. for 7.6 lb. The range is at 5.1 lb. and 3.5 lb. for the 17.1 lb. and 7.6 lb. respectively. The results obtained up to this point show that the best predictors for weight are altitude and pitch. In the final section of this chapter, a model is constructed by integrating altitude and pitch sections in the training model. The model's prediction power is then tested by feeding individual sections of either altitude or pitch.

## Altitude Plus Pitch Results

The final predictions are made by integrating the altitude and pitch features. This means that the amount of input data curves is doubled from 144 to 288 curves. Half the curves correspond to the altitude feature and the other half correspond to the pitch feature. Figure 16 shows the input data format used for this model.

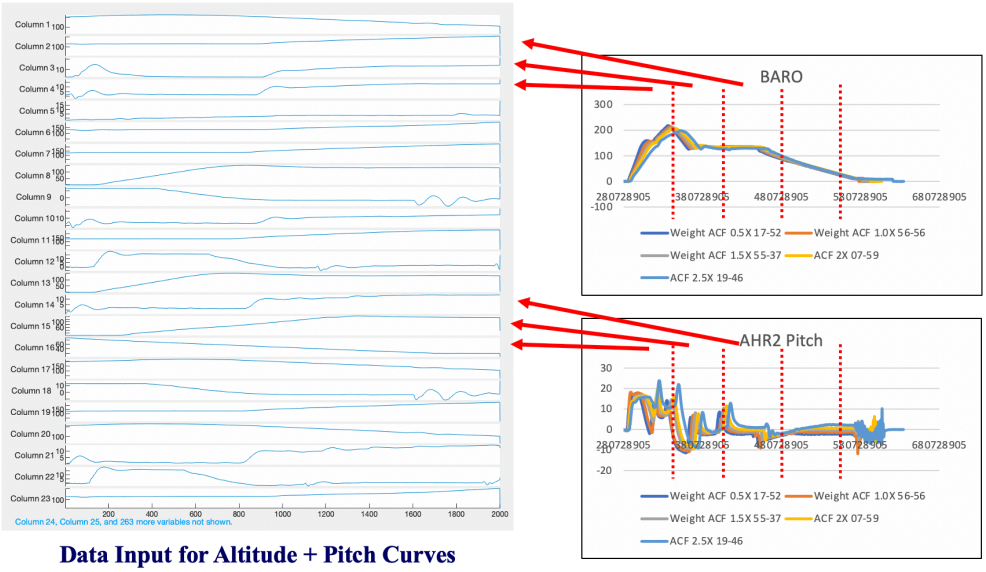


Figure 16: Altitude Plus Pitch Input Data Curves.

The same model is utilized for the integrated data model. An optimizable ensemble of trees model with cross validation with five folds is trained. Once the model is trained, the algorithm is tested with flight data that has not been seen by the training of the model. The first plot reviewed is the predicted versus actual response plot in Figure 17.

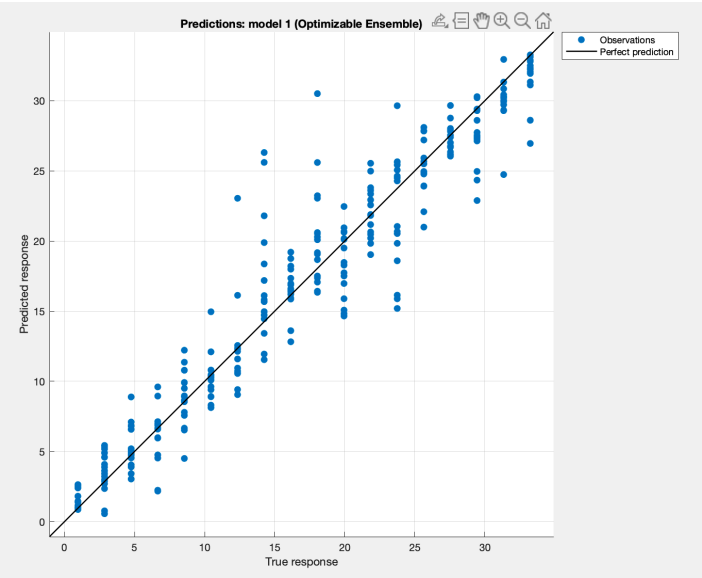


Figure 17: Altitude Plus Pitch Ensemble of Trees Optimizable Cross Validation 5 Folds Prediction Plot.

The resulting MSE for this model is 7.1. The datapoint variation follows a pattern similar to the altitude's predicted response plot in Figure 10 with a slightly improved MSE value from 8.3 to 7.1. The test flight's predictions for this model were performed slightly different than in the previous sections. The reason is that there are two different features being used for testing and not only one. To illustrate how the data curves for testing were inputted to the model, see Figure 18 below.

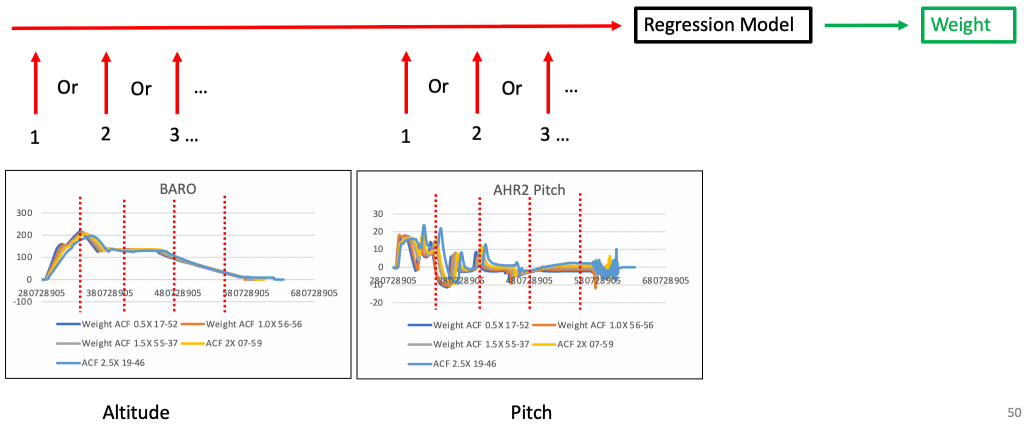


Figure 18: Altitude Plus Pitch Ensemble of Trees Optimizable Cross Validation 5 Folds Prediction Plot.

As shown in Figure 18, individual sections from either altitude or pitch are fed to the algorithm individually. To generate predictions, a section of each flight was input to the algorithm. This means that a total of eight sections were tested, four sections for altitude and four sections for pitch. To summarize the results, the average results for each section was averaged across both features (altitude and pitch). The results of the test flights are presented in Figure 19 below.

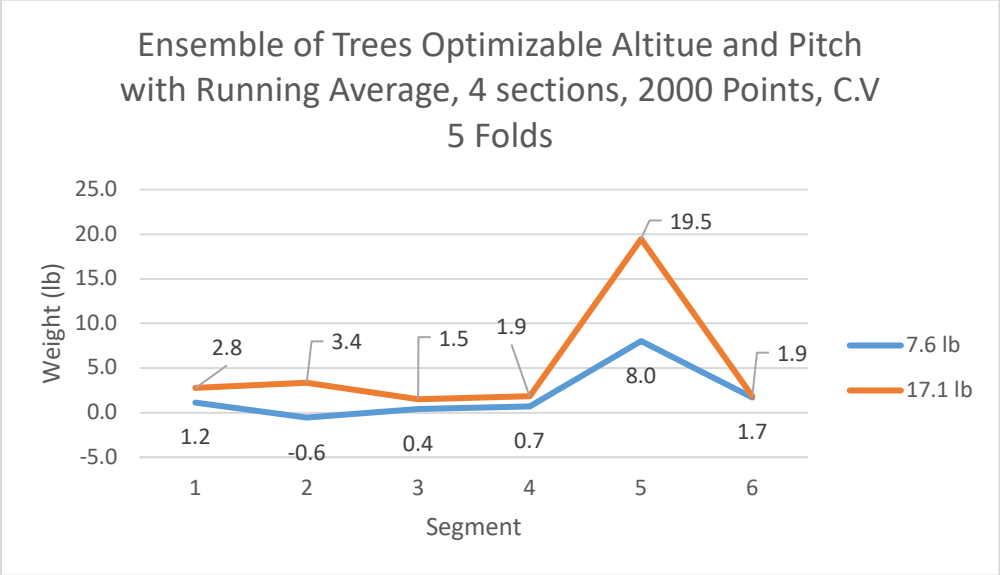


Figure 19: Altitude Plus Pitch Ensemble of Trees Optimizable Model Cross Validation 5 Folds Test Results Plot.

The final results display weight prediction deltas within 3.4 lb. across all sections for both 7.6 lb. and 17.1 lb. weights. The average predictions are at 19.5 lb. and 8.0 lb. for the 17.1 lb. and 7.6 lb. respectively. The overall range lies within 2 lb. for both UAVs across all sections for both features. This means that the model is capable to predict the UAV's weight regardless of the data segment and feature used (between altitude and pitch) for making weight predictions.

## Conclusions

The feasibility of Machine Learning for parameter estimation of UAVs has been demonstrated. Due to the nonlinearity present in the input data, the regression trees models present an advantage and show better prediction power than SVM regression models over the complete range of input data. Cross validation with 5 folds improved the fitness of the model in both altitude and pitch features. The coupling of individual performing features shows that there is no sacrifice in accuracy in predictions. Feature coupling is feasible and allows to make predictions with partial data input from either feature. Overall prediction variation for altitude and pitch model integration is below 2 lb. for the performed prediction flight tests. Expand the application to the estimation of other parameters like flight coefficients (lift, drag, moment) to evaluate the adaptability of the method. Incorporate flight maneuvers that ensure the excitation of all dynamic flight modes and evaluate if model predictions can be improved. Increasing the amount of flight data for training and testing regression models. Expand the application of Machine Learning to Neural Networks to evaluate if they can better predict aircraft parameters. Finally, the final improved predicting model should be validated with real flight data to assess its real prediction power.

## References


1. Jategaonkar, R. V. (2015). Flight vehicle system identification a time-domain methodology (2nd ed., Vol. 245). Reston, VA: AIAA, American Institute of Aeronautics & Ast.
2. Bhavithavya, K. K. (2013). System Identification of Fixed-Wing Unmanned Aerial Vehicle. 5th ed. Bangalore: Council of Scientific and Industrial Research National Aerospace Laboratories.
3. Hoffer, N. V. (2014). System Identification of a Small Low-Cost Unmanned Aerial Vehicle Using Flight Data from Low-Cost Sensors. All Graduate Theses and Dissertations. https://digitalcommons.usu.edu/etd/4274
4. Nasteski, V. (2017). An overview of the supervised machine learning methods. Horizons.b, 4, 51-62. doi:10.20544/horizons.b.04.1.17.p05
5. Morelli, E. (2006). Practical Aspects of the Equation-Error Method for Aircraft Parameter Estimation. AIAA Atmospheric Flight Mechanics Conference and Exhibit. doi:10.2514/6.2006-6144
6. Johnson, M. L., Faunt, L. M. (1992). [1] Parameter estimation by least-squares methods. Methods in Enzymology Numerical Computer Methods, 1-37. doi:10.1016/0076-6879(92)10003-v
7. Gabrielli, L. (2017). Introducing deep machine learning for parameter estimation in physical modelling. Edinburgh, UK: Proceedings of the 20th International Conference on Digital Audio Effects (DAFx-17)
8. Liu, Q., & Wu, Y. (2012). Supervised Learning. In N. M. Seel (Ed.), Encyclopedia of the Sciences of Learning (2012th ed.). https://doi.org/10.1007/978-1-4419-1428-6_451
9. Nadler, D. W. (2019). Decision support: Using machine learning through MATLAB to analyze environmental data. Journal of Environmental Studies and Sciences, 9(4), 419-428. doi:10.1007/s13412-019-00558-9
10. Train Regression Models in Regression Learner App: The MathWorks. (n.d.). Documentation (R2020b). Retrieved October 28, 2020, from https://www.mathworks.com/help/stats/train-regression-models-in-regression-learner-app.html
11. Regression Learner App: The MathWorks (n.d.). Documentation (R2020b). Retrieved October 28, 2020, from https://www.mathworks.com/help/stats/regression-learner-app.html
12. How X-Plane Works: X-Plane (2017, February 10). Retrieved October 31, 2020, from https://www.x-plane.com/zh/desktop/how-x-plane-works
13. Bittar, A., Figuereido, H. V., Guimaraes, P. A., & Mendes, A. C. (2014). Guidance Software-In-the-Loop simulation using X-Plane and Simulink for UAVs. 2014 International Conference on Unmanned Aircraft Systems (ICUAS). doi:10.1109/icuas.2014.6842350
14. X – Plane 11 Desktop Manual: X-Plane. (2020). Retrieved October 31, 2020, from https://www.x-plane.com/manuals/desktop
15. Plane Maker Manual: X-Plane. (2020). Retrieved November 21, 2020, from https://developer.x-plane.com/manuals/planemaker/
16. Airfoil Maker Manual: X-Plane (2020). Retrieved November 21, 2020, from https://developer.x-plane.com/manuals/airfoil_maker/
17. Michailidis, M. G., Agha, M., Rutherford, M. J., & Valavanis, K. P. (2019). A Software in the Loop (SIL) Kalman and Complementary Filter Implementation on X-Plane for UAVs.

2019 International Conference on Unmanned Aircraft Systems (ICUAS). doi:10.1109/icuas.2019.8797942
18. Mission Planner Overview: ArduPilot Dev Team (2020). Retrieved November 01, 2020, from https://ardupilot.org/planner/docs/mission-planner-overview.html
19. Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press. doi:10.1017/cbo9781107298019
20. Verrelst, J., Muñoz, J., Alonso, L., Delegido, J., Rivera, J. P., Camps-Valls, G., & Moreno, J. (2012). Machine learning regression algorithms for biophysical parameter retrieval: Opportunities for Sentinel-2 and -3. Remote Sensing of Environment, 118, 127-139. doi:10.1016/j.rse.2011.11.002
21. Smola, A., Vishwanathan S. V. N., (2008) Introduction to Machine Learning. Cambridge: Cambridge University Press. (Smola & Vishwanathan, 2008)
22. Introducing Machine Learning [PDF]. (2016). The MathWorks.
23. Rodriguez-Galiano, V., Sanchez-Castillo, M., Chica-Olmo, M., & Chica-Rivas, M. (2015). Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines. Ore Geology Reviews, 71, 804-818. doi:10.1016/j.oregeorev.2015.01.001
24. Loh, W.-Y. (2011). Classification and regression trees. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. **1**(1), 14–23. http://doi.wiley.com/10.1002/widm.8
25. Regression Tree Ensembles: The MathWorks (n.d.). Retrieved November 03, 2020, from https://www.mathworks.com/help/stats/regression-tree-ensembles.html
26. Dietterich, T.G. (2000). An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. Machine Learning 40, 139–157. https://doi.org/10.1023/A:1007607513941
27. S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas. (2006). Data preprocessing for supervised learning. Int. J. Comput. Sci., vol. 1, no. 1, pp. 111–117.
28. Corrales, D., Corrales, J., & Ledezma, A. (2018). How to Address the Data Quality Issues in Regression Models: A Guided Process for Data Cleaning. Symmetry, 10(4), 99. doi:10.3390/sym10040099