Summer 8-7-2020

Flight Data of Airplane for Wind Forecasting

Astha Sharma
University of New Orleans, asharma6@uno.edu

Flight Data of Airplane for Wind Forecasting

A Thesis

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfillment of the
requirements for the degree of

Master of Science
in
Computer Science

by

Astha Sharma

B.E. Tribhuvan University, 2015

August, 2020

Acknowledgments

I would like to thank my master's thesis advisors, Professor Dr. Md. Tamjidul Hoque and Professor Dr. Mahdi Abdelguerfi of the Department of Computer Science for constantly guiding me and encouraging me to work hard for the past two years. I am very grateful to Dr. Mahdi for the support and motivation he has given me through every stage of my research. I am thankful to Dr. Hoque for providing me an opportunity to learn and grow my knowledge in different areas of Machine Learning and helping me find the answers to my research questions.

I would like to thank Dr. Elias Ioup from the Center for Geospatial Sciences, Naval Research Laboratory, Stennis Space Centre, Mississippi, USA, and Professor Dr. Shaikh Arifuzzman of Department of Computer Science, for serving as my committee members at hardship.

I would also like to thank Pujan Pokhrel for helping me understand different ML approaches, which were beneficial as I worked through my research work.

A very special thanks to my family. It would not have been possible without their support. I would like to thank my mom, dad, sister Dikchya, and brother Vinayak for having faith in me.

Last but not least, I am thankful to my friends Manisha Panta, Reecha Khanal, and Rishav Rajendra for all the fun time and support. Thanks to everyone who was, directly and indirectly, involved in helping me get through my thesis research.

Table of Contents

List of Tables

List of Figures

Abstract

This research solely focuses on understanding and predicting weather behavior, which is one of the important factors that affect airplanes in flight. The future weather information is used for informing pilots about changing flight conditions. In this paper, we present a new approach towards forecasting one component of weather information, wind speed, from data captured by airplanes in flight. We compare NASA's ACT-America project against NOAA's Wind Aloft program for prediction suitability. A collinearity analysis between these datasets reveals better model performance and smaller test error with NASA's dataset. We then apply machine learning and a genetic algorithm to process the data further and arrive at a competitive error rate. The sliding window approach is used to find the best window size, and then we create a forecasting model that predicts wind speed at high altitudes 10 mins ahead of time. Finally, a stacking-based framework was used for better performance than individual learning algorithms to get root means square error (RMSE) of the best combination as 0.674, which is 98.4% better than the state-of-the-art approach.

Keywords: Machine Learning, Weather Forecasting, Genetic Algorithm, kNN imputation, Linear Regression, Extreme Gradient Boosting, Time Series Forecasting, Sliding Window, Stacking-based Framework.

Chapter 1 – Introduction

Weather forecasting is a scientific approach to predict the atmospheric conditions for a given time and location. The informal prediction of weather started millennia ago and was formally established in the 19th century. People used to rely on hand-based calculations depending on the changes observed in barometer pressure, current weather or sky conditions, and position of clouds. However, with the advancement of technology, weather forecasting has now relied upon computer-based models, which take many atmospheric variables into account. But human skills are still required to choose the best forecast model based on accuracy. The errors in forecasting models can be a result of the chaotic nature of the atmosphere, the computational power required to solve spatial equations, the initial miscalculations, and the incomplete understanding of atmospheric processes.

Correct forecasting is essential because it helps protect life and property from potential loss. There are a lot of parameters that are included in weather forecasting like temperature, air pressure, wind speed, wind directions, and rainfall. Each has its own scope of importance. In our paper, we are mostly focused on predicting the wind speed at higher altitudes.

The aviation industry is a risky business, and small changes in the atmospheric conditions can highly affect perfectly fine airplanes flying at greater altitudes from the earth's surface. But like everything else, airplanes are also coupled with the risk factors. Thankfully, science and technology can be used to come up with efficient approaches to reduce danger. Airplanes in flight are highly affected by and susceptible to high altitude winds. These types of wind can be the cause of flight turbulence and potential threats associated with it. A heads-up on upcoming powerful storms can help change the direction or even re-schedule flights to avoid possible hazards.

The most popular form of transportation for long-distance travel in the US [1], airplane, is also a great resource for weather data. The data collected from an airplane during flight includes information about the aircraft's position as well as meteorological and environmental measurements. These data are of great value for analyzing and predicting various natural conditions, like turbulence, which can assist the pilot and flight crew in making decisions about the future and avoid any possible mishap. In addition, these data can be used to monitor the flight progress and provide improved arrival and departure estimates to passengers.

Most of the available fight scheduling applications in the US are based on the information provided by the Wind Aloft Program from the US National Oceanic and Atmospheric Administration (NOAA)[2]. This program collects data via the recurring release of weather balloons and radar. The forecasts models then use linear interpolation to combine information from the available measurements[3]. However, there is evidence that this NOAA data may not be sufficient for making accurate predictions[4]. In our research, we try to put some light on the wind information from NOAA data at different heights above the ground.

The NOAA data come from weather models that are fed with measurements from ground stations along with data from weather balloons, satellites, and other instruments. The wind data are available at different altitudes ranging from 3,000 to 53,000 feet and include information from 9 different regions of America: Northeast, Southeast, Northcentral, South Central, Rocky Mountain, Pacific Coast, Alaska, Hawaii, and West Pacific[5].

Alternatively, the Atmospheric Carbon and Transport-America (ACT-America) campaign from NASA covers 4 seasons and 3 regions of the central and eastern United States and is based heavily on direct in-flight measurements. Using a variety of instruments, airplanes record their positional data as well as meteorological and environmental readings across a variety of surface

and atmospheric conditions. The dataset includes 118 days of data with a temporal resolution of 1 second. There is a total of 34 different features, including latitude, longitude, altitude, ground speed, air temperature, wind speed, and direction [6].

Our objective is to choose a quality dataset from the set of above explained two groups, as a step forward in the direction of accurately forecasting/ nowcasting the wind speed.

In this research, we first describe the model performances for each dataset based on linear regression and chose one over the other for the remaining part of the research. Then we further cleanse the dataset and apply machine learning algorithms to derive useful information about the wind speed. We then evaluate the performance based on the use of different algorithms and group together a bunch of outperforming classifiers to give a result better than the result obtained from each classifier. We also create offline software that predicts the wind speed in the next ten minutes. Finally, we compare the results with some related works.

The remainder part of this thesis is organized as follows. In Chapter 2, we review some research work related to weather forecasting. Chapter 3 is all about classical machine learning techniques (ML) as well as the relevant tools and technologies. In Chapter 4, we describe the setup of our study. Here, we introduce the datasets, feature extraction procedure, the forecasting models, and performance evaluation metrics used in our work. It also includes elaboration on the parameter selection and optimization of several state-of-the-art machine learning (ML) techniques implemented. Chapter 5 presents the Stacking-based framework and the performance comparison to relevant state-of-the-art techniques. In Chapter 6, we define the result and compare it with that from the Literature Review. Finally, Chapter 7 concludes the thesis work with the selection of the best performing predictor framework and future directions.

Chapter 2 – Literature Review

In the literature review, we discuss similarities and differences in the work done in a similar context in the past, as in comparison to what we have done. We have put high efforts in understanding the weather and atmospheric components to implement their correlation in our approached wind model. Efforts have also been made to develop machine learning models for the predictive analysis of airplane data to improve upon the existing NOAA forecasts.

2.1 Traditional Approaches for Wind Forecasting

The very first weather report was published in "The Times" on August 1, 1861 [7][8]. The captain of "The Beagle," Robert FitzRoy, was concerned about the weather and the threats associated with the ship in a sail. So using some instruments like a barometer and a couple of thermometers at the ports of Britain's coast, he collected weather data which he used along with his instincts ruled by observation of the sky and the atmosphere to forecast the weather [7]. In 1911, the first marine weather forecast was issued via radio transmission, which included gale and storm warnings around Great Britain [8]. The first public radio forecast in the United States was made in 1925 by Edward B. Rideout, on the Edison Electric Illuminating station in Boston [8]. Today, there are a number of ways in which weather can be forecasted. A consensus of forecast models based on various parameters like model biases and performance can help reduce forecast errors [9]. There are a number of fields that need a special type of weather forecasting, and one of them is aviation.

The aviation industry is very sensitive to changing weather. Fog, turbulence, icing are forms of weather hazard that can cause trouble in landing and takeoffs as well as airplanes in-flight [10]. Thunderstorms are another significant problem for most of the aircraft, which results in-flight turbulence because of their updrafts and outflow boundaries [10]. So with this information, it is viable to say that it is very important to accurately predict the weather, especially when it comes

to the aviation industry. However, most of the flight scheduling applications available in the United States today depend on the data provided by the Nation Oceanic and Atmospheric Administration (NOAA). For our research purpose, we will put some light on the significance of this particular set of data. The information that NOAA *Winds Aloft* service [5] provides on winds, we will refer to it as the NOAA data for simplicity.

In general, the commercial aircraft fly at heights between 23000 and 41000 feet. It is found that winds at these altitudes can vary from 30 to 120 knots [12]. The NOAA data includes wind measures from 176 different weather stations throughout the US by lofting high-altitude weather balloons in every 6 hours. The wind speed and magnitude are measured at a set of altitudes starting from the 3000 feet to 53000 feet above the sea level. In addition to that, a set of measurements from corresponding wind stations and wind data from radar observations are combined using ad hoc rules, like weighted average, to provide estimates of the winds over a wider range [2][13].

Reports have shown that over the past 50 years, pilot voice is used in weather models. And for more than the last 20 years, efforts have been made to employ data from commercial aircraft for better weather modeling [14]. In these past 20 years, the privately-owned commercial aircraft data was made accessible to the government organizations for predicting weather phenomena. There is another program called Aircraft Meteorological Data Relay (AMDAR) [14], which focuses on providing meteorological data from aircraft. To date, AMDAR has centered efforts around constructing infrastructure to collect and disseminate the data. Efforts have also made qualitative analyses and in simple linear models using AMDAR data [15].

Our work is differentiated from prior work as we focus on the comparative analysis of two publicly available datasets, one from NOAA and other from the featureful and rich NASA data. We also dig deep into the filtering dataset and use it to create a predictive model for forecasting

winds at high altitudes. One of the reasons for using a rich dataset is to understand the environmental features that actually boost the possible wind speed. Although the environment variables cannot be controlled, the results from the wind predicting model can be used by pilots to make wise decisions about the schedules as well as the direction and position of the aircraft to avoid any disasters. One of the key benefits of the proposed method is that it allows for better predictions by mining available data.

2.2 The Microsoft Project

One of the major inspirations for this research is a similar project from Microsoft [3], in which they conduct a comparative analysis of possible approaches for wind prediction at a continental scale. The tried to predict the wind at different heights for given locations. There were 1653 observations from 496 aircraft. However, they had removed all the observations with 0 and over 100 knots of speed. They used Gaussian processes (GP) [16] for their predictive analysis. GP has been used to model natural phenomena, including spatial interdependencies [17], [18]. As an example, they have been applied in modeling wind energy and power forecast [19]. In the case of Microsoft, they directly observed the aircraft ground speed and extended the GP-based techniques to incorporate data that is auxiliary to the phenomenon being modeled. The results, as mentioned in Microsoft's research paper using different approaches, are described in Table 1.

Table 1. Microsoft Research Results

Approach	**RMS Error**
NOAA data	51.53
Gaussian Process Estimate	50.93
Gaussian Process + Airplane Data	43.66

One of the problems with Microsoft's project is that its data is not publicly available. However, in our research, we have removed the barrier by using open-source datasets that are easily available. Therefore, this research can be used as a benchmark for comparison.

Chapter 3 – Tools and Techniques

In this chapter, we discuss all the different technical approaches we had applied to efficiently solve the problem definition associated with our dataset.

3.1 Machine Learning Techniques

This section comprises details about all the state-of-the-art machine learning methods, their working principles, strengths, and weaknesses.

3.1.1 k-Nearest Neighbors (kNN)

The k-Nearest Neighbors [20] algorithm is one of the simplest, non-parametric methods in machine learning which can be used for classification and regression problems. It is also known as instance-based or lazy learning because the algorithm makes local approximation and does not undergo explicit training before classification. The training data is stored in memory and is used during the testing phase. It is a non-parametric technique for estimating a decision boundary or a regressive curve without making a strong assumption [20]

The algorithm is based on feature similarity. It works by storing the entire training dataset and then finding the k most-similar training patterns as the prediction is made. The similarity can be measured in terms of distance (typically, the Euclidean distance) between the data instances. Then an instance is classified to a particular class based on the majority of votes among its identified k neighbors. We generally take k as an odd value. Since the algorithm stores all the training data, this algorithm is computationally expensive. However, it can be highly accurate in the case of nonlinear data. Overall, it is a very simple algorithm and often achieves very good performance.

3.1.2 Linear Regression

Linear regression is a simple and attractive regression algorithm which is derived from statistics and is studied as a model for interpreting the relationship between input and output variables.

It is a **linear model**, a model that assumes a linear relationship between the input variables (x) against the single output variable (y). Or we can also say that y can be calculated from a linear combination of the input variables (x). The unknown model parameters are estimated from the data. Like in all other regression analyses, linear regression also focuses on the conditional probability distribution of the responses obtained from the predictors.

Linear regression has been studied for a long time (more than 200 years old) and is the first type of regression analysis to be analyzed rigorously and to be used extensively in practical applications. This is because of the fact that models that depend linearly on their unknown parameters fit easily than models that are non-linearly related to their parameters. Additionally, it is also because the statistical properties of the resulting estimators are easier to determine.

3.1.3 Random Forest (RF)

Random forest [21] is an ensemble learning algorithm for both classification and regression problems that utilizes the predictive ability of multiple trained learners to create a single but better performance model. The algorithm creates a lot of individual decision trees based on randomly selected feature subspace from the training dataset. Using Bagging, each decision tree is trained on the randomly selected subset of the training data, in order to properly classify an unlabeled data instance, each decision tree votes for a class label meaning. Each decision tree-labels the instance as one of the output classes. The decision on the class of the instance is made on the highest votes obtained from the individual tree results in the forest. Despite being a complex form of algorithm

and requiring more computational resources, Random Forest is extremely flexible, has high accuracy, and can handle different feature types, including binary, categorical, and numerical.

Randomization is applied in RF when selecting the best node to split. While constructing multiple decision trees in RF, randomization (for selecting the best node to do a split) can be achieved using various algorithms like Gini index heuristics, Chi-Square, information gain between the features, or using splitting value classically equal to $\sqrt{M}$, where M is the number of features in the dataset.

3.1.4 Bagging Regressor

Bagging, also known as Bootstrap Aggregation, is an ensemble method that creates different samples of the training dataset and creates a unique classifier for each of those samples. The result from these multiple classifiers are then combined based on say, average value, or voting. The thing to remember here is that each sample of the training dataset is different, which in-turn gives different (trained) classifiers and definitely a different focus and perspective on the problem.

It helps reduce variance to escape from the overfitting problem. Generally applied to decision tree methods, it can be used with different other methods. Bagging is a special case of a model averaging approach.

3.1.5 Stochastic Gradient Descent

Stochastic gradient descent is one of the iterative methods for improving an objective function with proper smoothness properties. Since it replaces the actual gradient (obtained from the dataset) by an estimate (calculated from random subsets of data), it can be considered as a stochastic approximation of the gradient descent optimization. It reduces the computational burden and achieves faster iterations instead of a slightly lower convergence rate when it comes to computation of big data.

It is one of the important optimization techniques used in machine learning and is a popular algorithm for training a wide range of models, including support vector machine, linear regression, logistic regression, graphical models, etc.

3.1.6 Gradient Boosting

Gradient Boosting [22] is a machine learning technique for both classification and regression problems. It helps produce a predictive model which is in the form of an ensemble of weak predictive models (typically, the decision trees). It implements the gradient descent algorithm to optimize an arbitrary differentiable loss function as it builds the model using the concept of boosting. It is also known as GBRT (Gradient Boosted Regression Trees) and MART (Multiple Additive Regression Trees). Using gradient boosting, one can generate a set of weak classifiers or the decision trees and train them based on random subsets of data in a gradual, additive, and sequential manner. This algorithm is used to find the shortcomings of models using gradients on the loss function. The loss function generally depends on the problem space. It is one of the potential techniques for constructing predictive models as it allows us to optimize user-specified cost function and often works great with categorical and numerical values. However, it can be computationally expensive and memory exhaustive as it requires a large number of trees and a large grid search for parameter tuning.

3.1.7 eXtreme Gradient Boosting (XGB)

The XGBC [23] algorithm is an efficient application of a gradient boosting framework. The gradient boosting approach facilitates the creation of new models that predict the residuals or errors of prior models and then add them together to generate a final prediction. It is called gradient boosting because it utilizes a gradient descent algorithm to optimize the loss function or user-specified cost function. It aims to provide a scalable, portable, distributed, and parallel gradient

boosting and is specially designed for greater speed and better performance. It is an open-source software library that supports gradient boosting algorithm, stochastic gradient boosting with sub-sampling at the row, column, and column per split levels and regularized gradient boosting with both L1 and L2 regularization. It has gained much popularity and attention lately because of its salient features, which make it different from other gradient boosting algorithms.

3.2 Technologies Used

In addition to the machine learning approach, we have also adapted to some technology to support the processing of big data in our project.

3.2.1 Spark

Spark is a cluster computing framework that uses a collection of objects called Resilient Distributed Datasets (RDDs) that allow users to perform the in-memory computation on large clusters [24]. RDDs are fault-tolerant, parallel data structures which make it possible to hold intermediate results in memory, control their partitioning in order to optimize placement of data and manipulate them using a rich set of operators [24]. As an outcome of the intermediate results being stored in memory, Spark is proven to be much efficient as compared to Hadoop or any similar technology for iterative analytics like PageRank calculation, k-means clustering, and linear regression [25].

3.2.2 Apache Spark Streaming

Time-sensitive data needs real-time processing. Traditional MapReduce may not be a viable solution for such a case as it is suitable for offline batch processing where latency is not a big of a deal [26]. However, if the input data is being fed repetitively in discrete sets, multiple passes of the map and reduce tasks would create a computational overhead. This problem can be eliminated by using Spark. Using Apache Spark Streaming, we can enable the program to store the

intermediate results within memory and when new data arrives, it is batched to perform quick and efficient transformations on them [26].

Chapter 4 – Experimental Setup

In this section, we put light on the experimental composition of our research, including initial selection, training, and validation of the dataset, feature extraction, and engineering of the predictive model.

4.1 Dataset Collection

The first step to our research starts with the data collection. Machine Learning approach works better when there is enough information to train the model. So, data was one of our major constraints. We could only work on our ideas if we had enough data on the problem. Therefore, we intensively searched for the relevant dataset on the internet and landed with two of them that closely met our requirements. One of them was from the Wind Aloft program by National Oceanic and Atmospheric Administration (NOAA), and another was from The Atmospheric Carbon and Transport (ACT) America by National Aeronautics and Space Administration (NASA). The basic statistics on both the dataset is given in Table 2.

Table 2. Dataset Statistics

Datasets	Number of Instances	Number of Features
WIND ALOFT	170	5
ACT AMERICA	1.5M	33

4.1.1 NOAA based Dataset

The data obtained from the Wind Aloft program by NOAA is being referred to as the NOAA data. We collected the NOAA data from their website itself. There was a total of 170 different locations for which the wind information was recorded at 3K – 39K feet above the sea level. It also included wind information from nine different reasons for the American continent, including Northeast,

Southeast, Northcentral, South Central, Rocky Mountain, Pacific Coast, Alaska, Hawaii, and West Pacific. The wind information provided by them included information about the direction, speed, and temperature in a single block. It was then broken down into individual information based on the decoding technique defined on their website. The latitude and longitude were calculated based on the individual station location.

4.1.2 Airplane Dataset

Another set of data from NASA's ACT America project is referred to as the Airplane data. This dataset includes information from the two well-equipped airplanes. The sensors installed in the aircraft were used to collect atmospheric information like latitude, longitude, wind speed, wind direction, and air pressure. There is a total of 34 different variable information included in the dataset. This campaign covered four seasons and three regions of central and eastern United States. It contains a total of 118 days of data where individual file reflects flight information from each day. The spatial coverage for data is given as N: 49.11 S: 27.23 E: -71.91 W: -106.49, whereas, the temporal coverage is from July 11, 2016, to March 10, 2017. The temporal resolution of data was of 1 second.

This is a rare type of dataset with so many different environmental variables that can possibly be affecting the wind speed at higher altitudes. Therefore, we came to the conclusion of utilizing its huge coverage to derive useful information for our future predictive model.

4.2 Dataset Selection

The availability of two different datasets gave us an opportunity to dive in through the performance of each on correctly predicting wind speed based on the relevant features and then make a comparative analysis among the two. The decision to use one over the other or even going for a stacking-based approach was dependent on the results obtained from the evaluation of the existing

datasets. In case of the airplane data, there were 33 different features, so we wanted to understand how these features are correlated to the wind speed. Our goal was to find the most relevant set of data that would give us the more accurate wind information. Therefore, we turned into finding the Pearson's Correlation Coefficient for each feature against the wind speed for sample population and listed out the top 6 correlated features to wind speed. We then used the WEKA tool to perform a comparative analysis on the sample dataset. Linear regression was applied to the data and the Root Mean Squared Error was calculated.

In the case of the NASA dataset, the original set of 34 features was reduced down to 6 by selecting only the features most strongly correlated to wind speed, wiz., Mach Number, Ground Speed, Track Angle, Drift Angle, Static Air Temperature and Wind Direction. For the NOAA data, since there are only a few features provided (direction, temperature, latitude, longitude, and altitude), all were included. The NOAA data was trained using 10-fold cross-validation at all 170 different site locations at 30,000 feet height. A sample data of the first 3 days from the airplane data was also trained using 10 FCV. The RMSE, using Linear Regression, for both sources are given in Table 3.

Table 3. RMSE from applying 10FCV on Datasets

Source	#Observations	RMSE
Wind Aloft	170	20.0503
ACT-America	45126	31.9042

We also performed two simple analysis on this dataset at different heights in order to see if there is any height-wise data dependency:

i. Use training data from one height and analyze the prediction on all remaining heights

ii. Used training data from the lowest and highest heights and analyzed the consistency of the prediction of immediate layers and the layers thereafter.

The RMSE and MAE result from (i) at 30000 ft height is obtained as 23.4266 and 17.7324, respectively. The test results are given in Table 4.

Table 4. Model Prediction at different test heights

Test heights	RMSE	MAE
3000	42.3677	34.62
6000	38.3919	30.1
9000	34.4851	25.92
12000	30.8608	22.75
16000	25.1507	17.52
24000	21.334	12.36
34000	10.4867	7.09
39000	16.5777	11.68
45000	34.1917	24.81
53000	38.2205	29.25

The RMSE and MAE result from (ii) at minimum height 3000 ft was is obtained as 7.4168 and 6.3287 respectively, and at maximum height, 53000 ft was obtained as 13.7568 and 10.7422 respectively. The test results are given in Table 5.

Table 5. Model Prediction at different test heights

Training heights	Test heights	RMSE	MAE
3000ft	6000ft	8.3558	5.76
	9000ft	13.0813	10.18

	12000ft	17.9335	14.55
	18000ft	24.5426	19.76
	24000ft	29.3036	23.92
	30000ft	42.3676	34.62
	34000ft	46.8712	39.99
	39000ft	52.5115	45.14
	45000ft	31.6993	26.05
	53000ft	18.4978	15.03
53000ft	3000ft	18.497	15.0294
	6000ft	19.4907	16.9295
	9000ft	20.2745	17.3897
	12000ft	22.5671	18.7397
	18000ft	24.8981	20.29
	24000ft	27.8131	21.4903
	30000ft	38.2202	29.2497
	34000ft	41.9856	33.7597
	39000ft	47.0656	38.7496
	45000ft	17.9293	12.7

From Tables 3 and 4, we can conclude that we need to build separate models for different layers if we chose NOAA dataset over NASA data. On the other hand, the airplane data look much promising and have more features and observations than the NOAA data. Our research after that focused only on airplane data.

4.3 Data Analysis

The airplane data chosen for the rest of the research needed a bit of analysis and cleansing before we start off using it for our predictive model. Therefore, we began preprocessing our airplane dataset by sampling data from 5 days (selected randomly) having a reasonable number of input rows. The random sampling approach was employed because it gives an equal probability of selection for each element in the full dataset, thereby reducing the probability of biased results.

A deeper analysis of the available dataset revealed that more than 72% of the total rows had one or more missing fields. Almost 23–41% of the columns had missing values. This suggested that the available dataset is noisy. Simply dropping the rows with missing values would have been undesirable since it would mean losing a sizeable fraction of the data and potentially decreasing overall accuracy. We thus required a technique that could address gaps in data without losing samples.

We adopted the very popular technique of replacing missing values called kNN imputation. Based on the kNN algorithm, kNN imputation is widely known because of its great performance in machine learning applications. Here, the average of the *k* nearest neighbors at a fixed distance is used as the imputation estimate. We used Euclidean distance as the fixed distance parameter. The value for *k* was decided after computing the root mean squared error (RMSE) for a range of different values, from $k = 10$ to 1,500. Our result showed the minimum RMSE of 6.177 at $k = 500$. Therefore, k = 500 was used for imputing the missing values in the dataset.

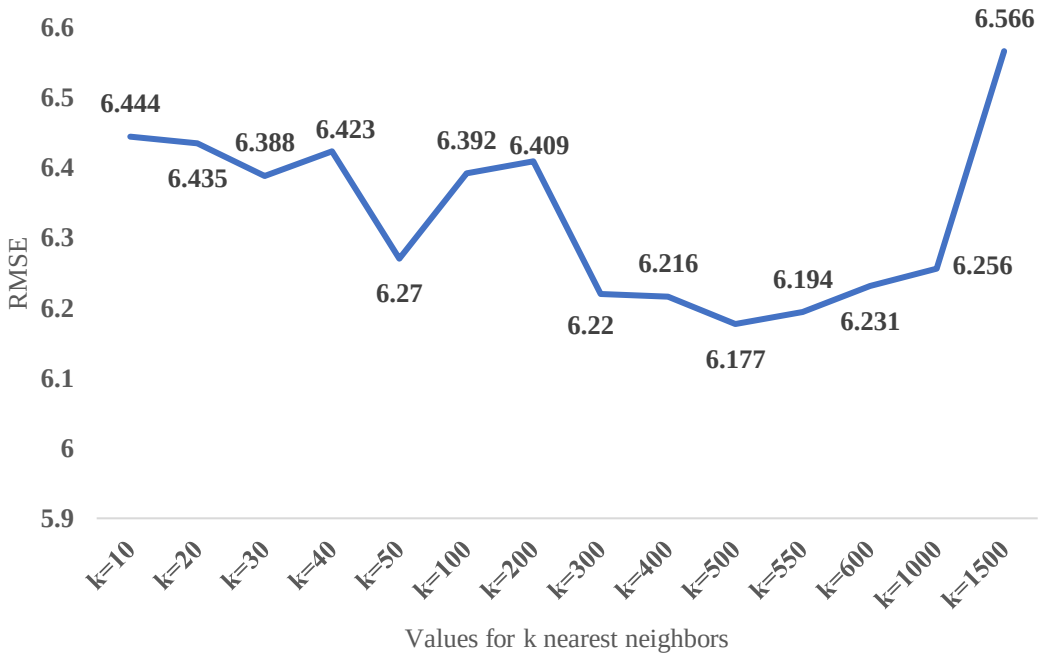

Figure 1. Corresponding RMSE for different k values

4.4 Feature Selection

Careful feature selection and filtering was a critical step of this research as we wanted to retain only the useful variables that are most related to the wind speed feature. For the feature selection process, we used the powerful genetic algorithm (GA) approach. GA gives a clear idea of feature selection without requiring expertise about the project's domain and inclination. For instance, we can determine whether the Mach number of an airplane is highly correlated to wind speed without necessarily understanding the principles behind that variable.

Two algorithms — Extreme Gradient Boosting (XGBoost) and Linear Regression — were used to analyze the fitness function, and the better algorithm was selected based on the output RMSE. Our GA ran for 300 generations for both fitness function algorithms. The following standard parameters were set for our GA: *Population Size* of 20, *Crossover Rate* of 80%, *Mutation Rate* of 5%, and *Elite Rate* of 10%.

At the end of 300 generations, XGBoost gave a total of 6 fittest chromosomes: indicated airspeed, Mach number, track angle, roll angle, potential temperature, and wind direction. Linear regression gave a total of 10: latitude, GPS altitude, ground speed, vertical speed, true heading, pitch angle, static pressure, sun azimuth, partial pressure water vapor, and saturated vapor pressure H_2O.

Since XGBoost reduced the number of chromosomes to 6 and obtained a fitness score (28.91) far better than linear regression (42.24), it was the better performer. Therefore, for prediction, we examined only these 6 features plus the wind speed.

4.5 Engineering the predictive model

After carefully working on the dataset for some time, we finally had it filtered with the selected feature set and reduced the noise as a result of the missing values. We then advanced to creating the predictive wind model.

4.5.1 Sliding Window Technique

We approached the time series forecasting with the sliding window technique. This approach takes a set of observations sequential in time and creates a model to fit in historical data. The model then predicts future outputs based on historical evidence.

The first step consisted of selecting the sliding window size. We considered a set of window sizes, ranging from 2 to 14. Again, RMSE was the deciding factor. After calculating the RMSE using four regression algorithms, XGBoost, KNN, Random Forest, and linear regression, we obtained the least RMSE at window size 9. The least RMSE using linear regression was obtained at window size 10. However, the difference in RMSE at windows 9 and 10 is nominal. For consistency, however, and considering the 1-second resolution of the data, we settled on a window

size of 9 for all the algorithms. Figure 2 shows the RMSE values obtained at different window sizes for all the algorithms used at this step.

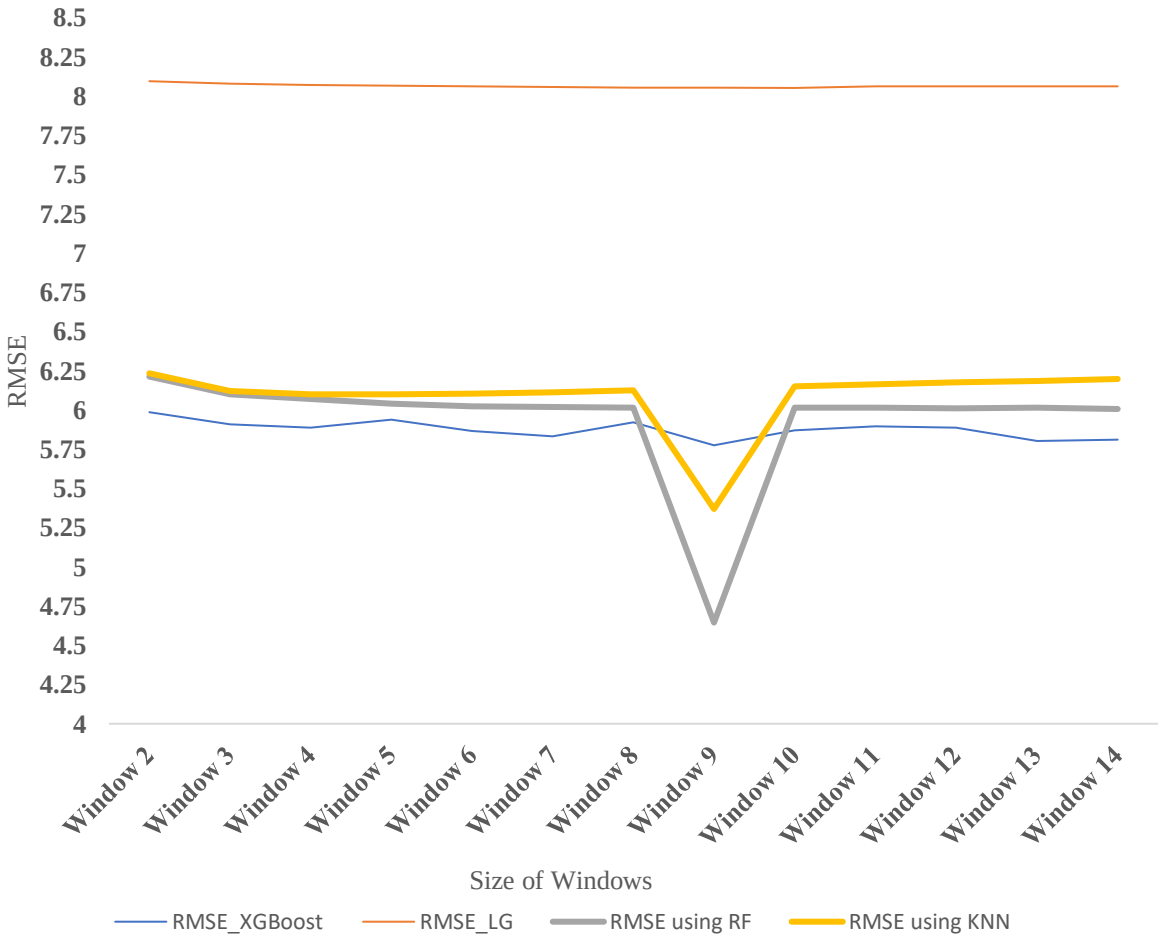

Figure 2. RMSE values obtained from using XGBoost, KNN, RF and Linear Regression

The better understanding of error values obtained at each window size using different algorithms can be visualized from the data in Table 6.

Table 6. RMSE at different window size using XGBC, LR, RF, and KNN

Different Window Size	RMSE_XGBoost	RMSE_LR	RMSE_RF	RMSE_KNN
Window 2	5.986	8.096	6.213	6.234
Window 3	5.909	8.082	6.1	6.123

Different Window Size	RMSE_XGBoost	RMSE_LR	RMSE_RF	RMSE_KNN
Window 4	5.887	8.072	6.071	6.1
Window 5	5.94	8.066	6.042	6.098
Window 6	5.866	8.063	6.023	6.103
Window 7	5.834	8.06	6.018	6.113
Window 8	5.922	8.057	6.014	6.124
Window 9	5.775	8.055	4.645	5.369
Window 10	5.869	8.053	6.013	6.152
Window 11	5.896	8.062	6.014	6.164
Window 12	5.889	8.062	6.011	6.175
Window 13	5.801	8.062	6.013	6.186
Window 14	5.809	8.062	6.008	6.196

4.5.2 Averaging

After deciding to move forward with the window size of 9, the next step was to take an average of all the variables in the dataset for a finite interval such that our data size reduces and gives us information about the wind speed and the correlated variables for this duration. After carefully understanding the total data points and the need for enough data for training and testing purposes for our future predictive model, we decided to take an average of 10 minutes. Since our data had a 1 second resolution so we took an average of every 600 (**60** – *for each second* ***10** – *for each minute*) points in each file. We ignored the remainder of every file. After averaging, we merged them into a single file with a total of 2127 data points. After this point, the file contained sorted variables and their averaged values for the best performing window size.

These averaged files were also checked for errors using different predefined models in order to verify if the quality of data is retained. Table 7 discloses the error rates of different models on the given dataset.

Table 7. Performance evaluation

Models	VAR	RMSE	R2
Linear Regression	0.215	7.703	0.215
KNN	0.913	2.644	0.913
Random Forest	0.993	0.772	0.993
Bagging	0.978	1.327	0.978

4.5.3 Training Model for Forecast

At this step, we focused on feeding data to the ML algorithms in a slightly different way. The features (X) from the first data point were linked to the goal (y) of the second data point, the features from the second data point were linked to the y of the third data point, and so on. We did this in order to help the machine understand the nature of the upcoming (10 minutes ahead) wind speed based on the current environmental features. We used different algorithms like Linear Regression, kNN, Random Forrest, Bagging, etc., to understand the performance of each on the given dataset. Table 6 shows the model performance evaluation under parameters specified under *Models and Parameters* in Chapter 4.

Table 8. Model Performance Evaluation

Models	VAR	RMSE	R2
Linear Regression	0.997	6.538	0.997
KNN	0.993	10.066	0.993

Models	VAR	RMSE	R2
Random Forest	0.998	5.540	0.998
Bagging	0.997	6.236	0.997

The same table can be better visualized in the column chart, figure 3.

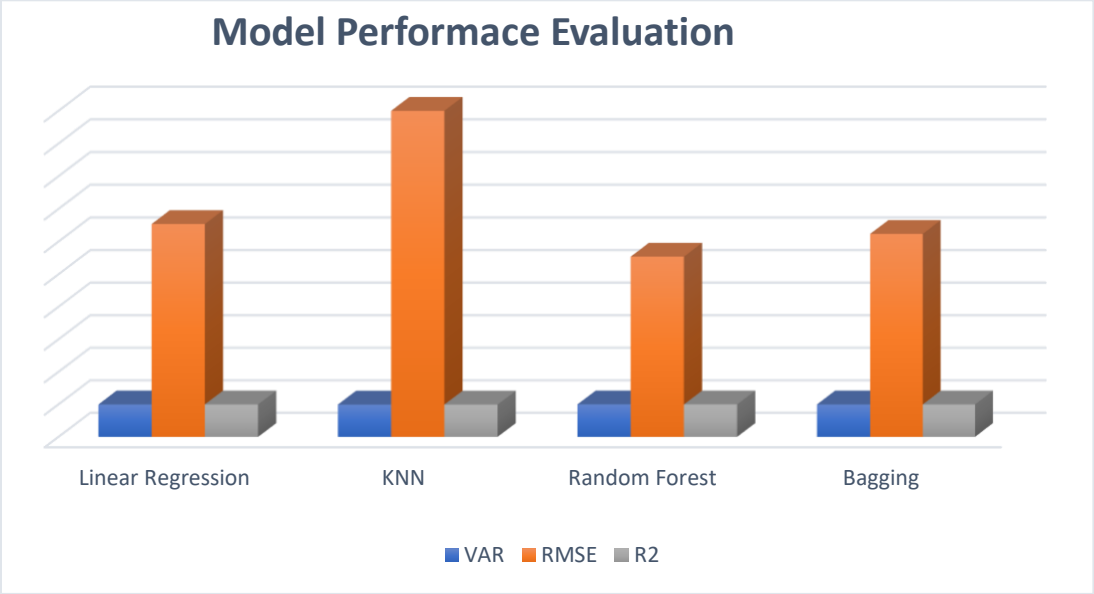

Figure 3. Var, MSE, and R2 of different ML models

4.5.4 Offline Predictive Model

Depending on the best performing model, i.e., Random Forest, we created an offline wind predictive model. A python script was generated to run pass the arguments (6 features obtained from the feature selection process in Chapter 3) to the model to give the wind value in knots. The result obtained was fairly accurate, with an error rate of 0.59.

3.5.5 ML Models and Parameters

We tried different algorithms for our problem domain and then evaluated the performance of each model using different evaluation metrics like RMSE, Variance, and R2. Based on the results

obtained from each model, we derived a conclusion on using the best performer model for predicting wind speed at higher altitudes.

We investigated six different supervised machine learning algorithms under different parameter settings. We used simple yet but effective and widely used algorithms such as k-Nearest Neighbors (kNN) [20], Linear Regression, Random Forest (RF) [21], Bagging, Gradient Boosting Classifier (GBC) [22], and Stochastic Gradient Descent.

We used the novel Scikit-learn library in order to build the model and fine-tune the parameters of different learning algorithms, as mentioned above. The parameters were defined as $k = 100$ for k – Nearest Neighbors, 3000 estimators for Random Forest, 10 estimators for Bagging, 0.02 learning rate, and 500 estimators for Stochastic Gradient Descent. The Linear Regressor model was set up with default parameters.

Chapter 5 – Machine Learning Approach

5.1 No Free Lunch Theorem

In Machine Learning, there is a famous *No Free Lunch* theorem developed by Wolpert *et al.* [33]. According to this theorem, every problem is unique, and there is no specific algorithm that is defined to work best for every problem. Hence the name No Free Lunch. Therefore, if we are working with a machine learning approach for our problem domain, we must at least try a couple of algorithms to see which one is performing better and only use the best or a combination of best performers for deriving the solution model.

The choice of a learning algorithm is dependent on various factors like the nature and size of data under consideration, roughness of the decision boundary, the problem definition, and the computational time.

5.2 Framework for Stacking-Based Models

In this section, we describe the novel Stacking-based machine learning framework. Stacking multiple best performing classifiers give a result that is better than all the classifiers considered individually. We applied the Stacking technique to generate better wind predictor.

Stacking, an ensemble technique, combines several machine learning algorithms to create one predictive model. The prediction probabilities from selected base learners are augmented to the original feature set to build a new feature-set. Then the meta-classifier is trained on this new feature-set, reinforcing the final predictions [27][28][29].

Stacking implementation has at least two levels of learning stages. In our study, we prepared one layer of base learners and one layer of meta-learner. In the first stage of learning, we generated ML models for based layers using the Scikit-learn library. All the first level prediction probabilities from these base models were used as features and augmented with the original feature vector.

Then, the augmented feature vector was used for training the final level of the learner or the meta-classifier [30][31][32].

Considering machine learning algorithms are based on different working principles, we explored several state-of-the-art ML algorithms – KNN, RF, Linear Regression, Bagging, and XGBC. The selection of base and meta-learners was influenced by the underlying principle of the selected algorithm. We created different combinations of base learners with KNN, RF, Linear Regression, Bagging, and XGBC. Likewise, linear regression was chosen as the meta-classifier. As shown in Table 9, we generated different combinations of base classifiers, including the meta-classifier, leading to different Stacking-based models.

Table 9. Executed combinations of Stacked Models.

Models	Combination of Base Classifiers	Meta classifier	RMSE
1	RF+KNN	LR	0.674
2	RF+KNN+XGB	LR	0.737
3	RF+KNN+LR	XGBC	0.681
4			
5			
6			

5.2.1 Training Procedure

In the training phase, we used a state-of-the-art framework to generate the subsets of the dataset for the parent nodes (note that each subset of a dataset contains a feature-set with all its child nodes) and invoked our proposed stacked generalization-based framework to train different tiers of learners. We used the 10-fold cross-validation technique on our dataset while we ran it through different base classifiers to generate a new subset with predicted probabilities for each feature of

the dataset. All prediction probabilities were concatenated with the original dataset to generate a new subset of the training dataset, which was finally used as a training set to train meta-classifier. This way, a stacked model was developed by training different combinations of base and meta classifiers. We also calculated the model performances using RMSE to get the actual error rate due to stacking. The cross-validation is done in order to ensure that we are not overfitting out model.

Figure 4 represents the training phase for our proposed Stacking-based framework, where part (a) illustrates training base classifiers with the instances of child nodes ($X_{INITIAL}$), which gives a set of prediction probabilities (X_{PROB}) and part (b) illustrates a new feature set (X_{FINAL}) resulted by augmentation of the set of prediction probabilities (X_{PROB}) with the feature vector ($X_{INITIAL}$).

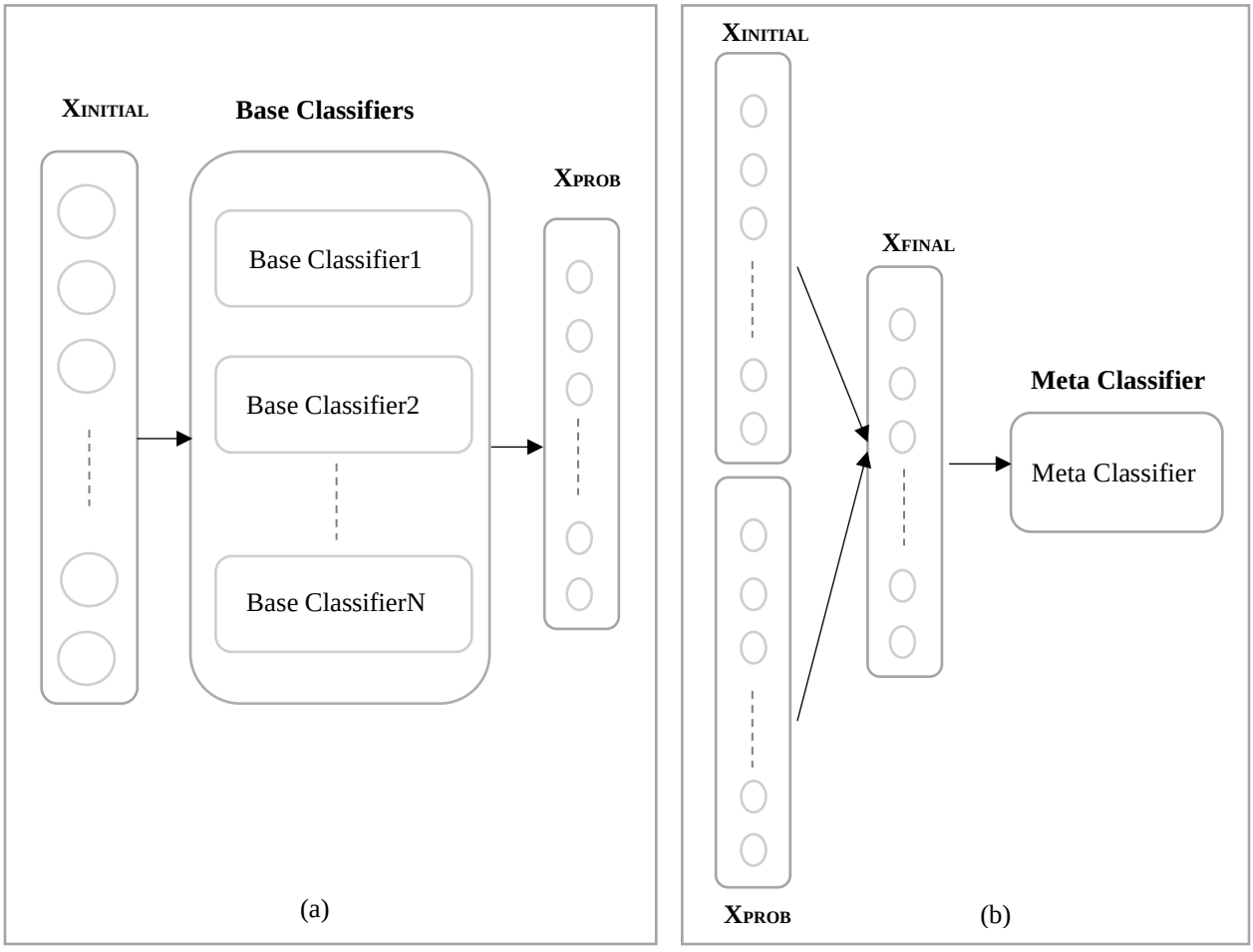

Figure 4. Training the Stack-based framework[33].

5.2.2 Grid Search

The hyperparameters of a model is a characteristic external to the model but affect the performance. The values of hyperparameter are set before the learning process is initiated, such as the value of *k* in k-Nearest Neighbors, the number of hidden layers in Neural Network, etc. Grid search is a technique to find the optimal hyperparameters for a model to give the most accurate predictions [34].

We used *GridSearchCV* from *sklearn* library to find the best hyperparameters for different models, including k-Nearest Neighbors, Random Forest, Bagging, Gradient Boosting, and Extreme Gradient Boosting. The outperforming hyperparameters were k=200 for k-Nearest Neighbors, max_depth = 5, n_estimators = 50 for Random Forest, n_estimators = 50, max_depth = 5, min_child_weight = 2 for extreme Gradient Boosting.

Chapter 6 – Results and Discussions

For this paper, we performed a handful of experiments like comparing the available datasets to deciding on choosing the best, analyzing and improving the quality of the selected dataset, and finally, presented a method of predicting the speed of one of the critical atmospheric phenomena, wind.

In the beginning, when we were not very familiar with both the dataset, NASA's dataset looked more promising because of the number of feature sets, the total number of observations and the spatial and seasonal coverage. However, we did some experiments to see if the NOAA's dataset can be used for further analysis. Since the NOAA website had the data spread according to the different heights, we tried to see if there were any height-wise data dependency. Therefore, we tried a couple of approaches to understand data behavior. First, we trained our model (using Linear Regression) at one height and tested the model at all other heights, one at a time. And secondly, we trained the model (Linear Regression based) at the lowest and highest heights and tested them against all other heights, one at a time. In both cases, we did not see any kind of trend that could be used to support data dependability. As a result, we completely switched to using NASA's dataset for the rest of the experiment.

A simple analysis of sample data from the NASA dataset helped us understand the ratio of missing values present in the dataset. Dropping the missing values was not an option because it would cut off most of the observations, leading to lower accuracy. Therefore we used an approach called kNN imputation, where the missing values were replaced with the mean of the k-nearest neighbors. Therefore, we first found out the best value for k neighbors, which was obtained from the root mean square error obtained from using different k values on the given dataset. This

experiment gave us the best value of k as 500, figure 1. Using this value, we performed the kNN imputation to cover the missing values and complete our dataset.

The next step required filtering of only useful features, and this was done by using the Genetic Algorithm for feature selection. We used some standard parameters like Population Size of 20%, Crossover Rate of 80%, Mutation Rate of 5%, and Elite Rate of 10% and ran two algorithms XGBoost and Linear Regression as the fitness function to obtain two different sets of fittest chromosomes. Since XGBoost gave the lowest fitness score (lower the better) and the least number of chromosomes, therefore, we stuck with this output. The entire dataset was then filtered based on only these six features, namely, *Indicated Air Speed, Mach Number, Track Angle, Roll Angle, Potential Temperature,* and *Wind Direction.*

Next, we applied the sliding window technique, where we used different window sizes ranging from 2 to 15 and calculated the RMSE at each window to see which window size is giving us the lowest error. For this experiment, we used four different ML algorithms, XGBoost, Linear Regression, Random Forest, and k-Nearest Neighbor, of which three showed window 9 to be the best performer. Therefore, we used the window size 9 dataset.

At this stage, we had a noise-free filtered dataset. However, it still needed some operation because the dataset we were using until this point was recorded at a time difference of one second. Therefore, we took an average of every ten minutes of data, which led to a drastic reduction in data points from over a million to 2127 observations. Thereafter, we used this dataset on different classifiers to check the model performance (Table 8). The training was done in such a way that the features from one line were mapped to the wind speed at the next line. This was done to help the model understand the nature of the wind speed in every other ten minutes. And among all the

classifiers Random Forest was performing the best, it was used to create an offline wind predictive model.

Last but not least, we tried stacking-based models using the different combinations of base and meta layers. The best performing stacking-based model had Random Forest and KNN at base layer and Linear Regression at the meta layer.

Our approach is valuable and general enough for use in similar cases and publicly available datasets. As a result of our different experiments, we have been able to obtain competitive results at each step of the project, as demonstrated in Table 10.

Table 10. RMSE at different stages of the project.

State	Datapoints	RMSE
Initial State	45126	31.904
After kNN Imputation	78023	6.177
After Sliding Window	1595422	5.775
After Training Model	2127	5.540

Our RMSE obtained at different stages of the project show a significant improvement. Recall that the best RMSE obtained by Microsoft's project discussed earlier was 43.66. This is a good indication that our project is headed in the right direction.

Chapter 7 – Conclusions

The number of commercial and military aircraft flying in the sky each day is massive and only expected to increase. Applying the in-flight data these aircraft collect to wind speed prediction can be efficient and cost-effective. Although the ten minutes ahead forecast may not be very significant in making flight schedules, it can be used in multiple other decision-makers about the airplane in flight, like deciding whether or not to move forward in case of possible turbulence, decisions about changing the direction of the aircraft, decisions about landing or takeoff and decisions about changing the trajectory of the aircraft.

This wind model is also closely related to turbulence experienced in flight, which depends on the wind speed at a particular position and altitude. We can, therefore, extend this project to create a predictive model that can be used to optimize flight time based on wind speed. Improving wind speed models also has applications in the creation of more fuel-efficient aircraft designs. Nevertheless, the result of this project is impeccable when compared to that of the NOAA and Microsoft models (Table 11).

Table 11. Output Comparison.

Projects	RMS Error
Wind Aloft from NOAA	51.53
Microsoft Research Project	43.66
Our Project	5.54

Current airplane flight planner applications are using weather information from the NOAA-based Wind Aloft program, which is quite noisy and less accurate. With a better system in place,

keeping track of flights can help manage arrivals and departures more efficiently and assist in making decisions about flight schedules.

References

[1] *Air Traffic by Numbers*. Federal Aviation Administration, June 2019.
[2] Federal Aviation Administration (FAA)/ Aviation Supplies & Academics (ASA). *Aviation Weather Services: FAA Advisory Circular 00-45g, Change 1*. July 2010.
[3] A. Kapoor, et al. "Airplane Aloft as a Sensor Network for Wind Forecasting." *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, 2014, pp. 25–34.
[4] C. Roberts. "America Has Gotten Bad at Predicting Weather - but There's a Plan to Fix It." *Observer Media Group Inc.*, August 26 2019.
[5] National Oceanic and Atmospheric Administration. "Wind/Temp Forecast." *Aviation Weather Center*, 2019, https://www.aviationweather.gov/windtemp/help.
[6] Yang, M., et al. "ACT-America: L1 Meteorological and Aircraft Navigational Data." *ORNL DAAC*, 2018.
[7] Laskow, Sarah. "The Very First Forecast." *Theatlantic.Com*, November 20. 2014, https://www.theatlantic.com/technology/archive/2014/11/the-very-first-forecast/382911/.
[8] "Weather Forecasting." *Wikipedia*, https://en.wikipedia.org/wiki/Weather_forecasting.
[9] Kimberlain, Todd. *TC Genesis, Track, and Forecasting*.
[10] Aircraft Owners and Pilots Association. *Aircraft Icing*. 2/2/2007. Accessed May 26 2008.
[11] National Weather Service Forecast Office Dodge City, Kansas. *Aviation Hazards They Didn't Tell You About*. September 10, 2008.
[12] Boccia, L., et al. "Low Multipath Antennas for GNSS-Based Attitude Determination Systems Applied to High Altitude Platforms." *GPS Solutions*, 2008.
[13] *Jeppsen Private Pilot Manual*. Sanderson, 2001.
[14] National Oceanic and Atmospheric Administration. *Aircraft Data Web*. https://amdar.noaa.gov/.
[15] http://amdar.noaa.gov/docs/mamrosh-ams-98/.
[16] C.E. Rasmusen, and C. Williams. "Gaussian Processes for Machine Learning." *MIT Press*, 2006.
[17] N. A. C. Cressie. "Statistics for Spatial Data (Revised Edition)." *Wiley*, 1993.
[18] Guhaniyogi, R., et al. "Adaptive Gaussian Predictive Process Models for Large Spatial Datasets." *Environmetrics*, vol. 22(8), 2011.
[19] Jiang, X., et al. "Adaptive Gaussian Process for Short-Term Wind Speed Forecasting." *2010 Conference on ECAI*, 2010.
[20] Altman, N. S. "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression." *The American Statistician*, vol. 46, no. 3, 1992, pp. 175–85.
[21] Breiman, L. "Random Forests." *Machine Learning*, vol. 45, no. 1, Oct. 2001, pp. 5–32.
[22] Friedman, J. H. "Stocastic Gradient Boosting." *Comput Stat. Data Anal.*, vol. 38, no. 4, 2002, pp. 367–78.
[23] Chen, T., and C. Guestrin. "XGBoost: A Scalable Tree Boosting System." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (San Francisco, California, USA, 2016)*, 2016.
[24] Zaharia, Matei., et al. "Resilient Distributed Datasets: A Fault-Tolerant Abstraction for in-Memory Cluster Computing." *9th USENIX Conference on Networked System Design and Implementation*, USENIX Association, 2012.
[25] Gopalani, Satish, and Arora Rohan. "Comparing Apache Spark and Map Reduce with Performance Analysis Using K-Means." *International Journal of Computer Applications*, vol. 113, no. 1, 2015.
[26] Matei Zaharia, et al. "Spark: Cluster Computing with Working Sets." *2nd USENIX Conference on Hot Topics in Cloud Computing*, 2010, pp. 10–10.
[27] Sumaiya Iqbal, Md Tamjidul Hoque, "PBRpredict-Suite: A Suite of Models to Predict Peptide Recognition Domain Residues from Protein Sequence," in ***Oxford Bioinformatics Journal***, 2018 [Published].

[28] Avdesh Mishra, Pujan Pokhrel, Md Tamjidul Hoque, "StackDPPred: A Stacking based Prediction of DNA-binding Protein from Sequence," in ***Oxford Bioinformatics Journal***, 2018 [Published].
[29] Michael Flot, Avdesh Mishra, Aditi Sharma Kuchi, Md Tamjidul Hoque, "StackSSSPred: A Stacking-Based Prediction of Supersecondary Structure from Sequence," ***Book Chapter*** (Chapter 5, pp 101-122), in: Kister A. (eds) Protein Supersecondary Structures. Methods in Molecular Biology, vol 1958. Humana Press, New York, NY, 2019 [Published].
[30] Corey Maryan, Md Tamjidul Hoque, Christopher Michael, Elias Ioup, Mahdi Abdelguerfi, "Machine Learning Applications in Detecting Rip Channels from Images," ***Applied Soft Computing, Elsevier Journal***, 2019 [Published].
[31] Aditi Sharma Kuchi, Md Tamjidul Hoque, Mahdi Abdelguerfi, and Maik Flanagin, "Machine Learning Applications in Detecting Sand Boils from Images," ***Array, Elsevier Journal***, 2019 [Published].
[32] Duaa Mohammad Alawad, Avdesh Mishra, and Md Tamjidul Hoque, "AIBH: Accurate Identification of Brain Hemorrhage using Genetic Algorithm based Feature Selection and Stacking," Machine Learning and Knowledge Extraction (MAKE) journal, ***MDPI***, 2020 [Published].
[33] Panta, M., et al. *Traning Procedure for Stacking Based Model*. https://player.slideplayer.com/100/17310465/slides/slide_15.jpg.
[34] Krishni. *An Introduction to Grid Search*. January 5 2019, https://medium.com/datadriveninvestor/an-introduction-to-grid-search-ff57adcc0998.

Vita

The author Astha Sharma was born in Lalitpur, Nepal. She received his bachelor's degree in Computer Engineering from the Tribhuvan University in Kathmandu, Nepal, in 2015. After two and a half years of professional experience, she joined the Graduate program of Computer Science at The University of New Orleans. She worked at Canizaro Livingston Gulf States Center for Environmental Informatics center as a graduate assistant while working on her thesis. This research work was conducted under the supervision of Dr. Md Tamjidul Hoque, Dr. Elias Ioup and Dr. Mahdi Abdelguerfi in 2019/2020