UNIVERSITY OF ICELAND

A NN approach to predicting gust factors in complex landscape

Brynjar Geir Sigurðsson

June 2024

M.Sc. thesis
in Mechanical Engineering

FACULTY OF INDUSTRIAL ENGINEERING, MECHANICAL ENGINEERING AND COMPUTER SCIENCE

A NN approach to predicting gust factors in complex landscape

Brynjar Geir Sigurðsson

60 ECTS thesis submitted in partial fulfillment of a
***CCMagister Scientiarum* degree in Mechanical Engineering**

Supervisor
Kristján Jónasson

M.Sc. Committee
Kristján Jónasson
Ólafur Pétur Pálsson

Examiner
XXNN3XX

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Reykjavik, June 2024

A NN approach to predicting gust factors in complex landscape

60 ECTS thesis submitted in partial fulfillment of a M.Sc. degree in Mechanical Engineering

Faculty of Industrial Engineering, Mechanical Engineering and Computer Science
School of Engineering and Natural Sciences
University of Iceland
Dunhagi 5
107, Reykjavik Iceland

Telephone: 525 4000

Bibliographic information:
Brynjar Geir Sigurðsson (2024) *A NN approach to predicting gust factors in complex landscape*, M.Sc. thesis, Faculty of Industrial Engineering, Mechanical Engineering and Computer Science, University of Iceland.

Reykjavik, Iceland, June 2024

To all the students who made the wise decision to use LaTeX.

Abstract

English abstract (ca. 250 words).

Útdráttur

Hér kemur útdráttur á íslensku sem er að hámarki 250 orð.

Contents

List of Figures

List of Tables

Listings

Abbreviations

Í þessum kafla mega koma fram listar yfir skammstafanir og/eða breytuheiti. Gefið kaflanum nafn við hæfi, t.d. Skammstafanir eða Breytuheiti. Þessum kafla má sleppa ef hans er ekki þörf.

The section could be titled: Glossary, Variable Names, etc.

If you use acronyms, it is strongly recommended to use a LaTeX acronyms package. For example, to use the package *acronym*, add in the preamble:

```
\usepackage{acronym}
```

and then here in this chapter (to create the list of all acronyms), use:

```
\begin{acronym}[SENS]
  \acro{SENS}{School of Engineering and Natural Sciences}
  \acro{UoI}{University of Iceland}
\end{acronym}
```

In the square bracket above, you need to put the longest acronym, so that the list gets proper indentation based on the longest acronym. Also note that you need to manually sort here that list.

Then, whereever you use the acronym in your text: `\ac{UoI}`: that will automatically expand the acronym at the first use, but use only the short version after the first use. (Use `\acp` if need the plural form. `\acl` if you explicitly want to have the long form only, `\acf` if you explicitly want to have the full long and short form, i.e. like first use of `\ac`.)

Use `\acresetall` to forget about earlier usage (=expansion) of acronyms, e.g. if despite already used in Abstract or Introduction, but you want to expand them later once again (starting from, e.g., in Foundations chapter): add `\acresetall` at start of Introduction chapter (and maybe also again at start of Foundations chapter).

Acknowledgments

Í þessum kafla koma fram þakkir til þeirra sem hafa styrkt rannsóknina með fjárframlögum, aðstöðu eða vinnu. t.d. styrktarsjóðir, fyrirtæki, leiðbeinendur, og aðrir aðilar sem hafa á einhvern hátt aðstoðað við gerð verkefnisins, þ.m.t. vinir og fjölskylda ef við á. Þakkir byrja á oddatölusíðu (hægri síðu).

1 Introduction

Wind gusts are brief increase in wind speed (lasting seconds) as compared to mean wind speed. The gust factor is defined as the peak gust divided by the mean wind speed over some defined time period. The peak wind gust is often defined as the highest 3 second rolling average measured wind speed over a period of 10 minutes, while the mean wind is the average of all measurements in the 10 minute intervalf. This thesis uses this definition. This varies, with the US using a 1 minute interval, leading to 14% higher results [17]. The Navier Stokes Equation (1.1) shows that the change of the wind, in time and space, is dependent upon the pressure gradient, the oscillating force of the earth (the Coriolis force), and frictional force.[7]

$$\frac{\delta \mathbf{V}}{\delta t} + \mathbf{V} \cdot \nabla \mathbf{V} = \underbrace{-\frac{1}{\rho} \nabla P}_{pressure} - \overbrace{f \mathbf{k} x \mathbf{V}}^{oscillation} - g - \underbrace{\frac{\delta(u'\omega')}{\delta z} - \frac{\delta(v'\omega')}{\delta z}}_{resistance} \tag{1.1}$$

Traditionally, numerical weather prediction (NWP) systems are used to forecast and analyze weather patterns[8]. These models describe the transition between discretized packages of atmospheric states using partial differential equations based on physical reality. These results are usually published every hour, or at courser time intervals for climate simulations. With increasing computer power and efficiency the trend is to output data more often[20].They describe the state over the period and so do not necessarily grasp fluctuations well. These fluctuations would include fluctuations in the wind speed, wind gusts[23].

This thesis looks at how best to predict gust factor based on various factors, using several different data sources, including NWP and observations. Being able to accurately predict the wind gust is important as it is often the peak wind gusts that will cause failures in structures. A problem that will become increasingly prevalent in the near future[16].

1.1 Background

The history of numerical weather predictions goes all the way back to the 1920's when Lewis Fry Richardson pioneered the field and tried to produce forecasts. The results were flawed due to noise in the calculations. ENIAC was built in 1945, it was a general purpose computer that was used, among other things, to make predictions. These predictions took 24 hours to make and were predicting 24 hours into the future. It was a proof of concept but not usable[14]. In the 1950's, with the advent of computers the first operational forecasts emerged. In September of 1954, Rossby and his Stockholm based team produced the first real-time barotropic forecasts. The next year the Joint Numerical Weather Prediction Unit (JNWPU), based in Princeton New Jersey, released their first forecasts. These forecasts were for 36 hours at 400, 700 and 900 mb. The results were inferior to subjective human-based forecasts but showed that such forecasts were feasible and promoted further development in the area [11]. The field of NWP has taken great strides since then following the development of computer power an efficiency.

Currently there has been another transformation in the field of weather prediction driven by artificial intelligence. Interest in AI has come in waves. Some progress is made, then interest dwindles. The interest has been increasing steadily since 2010. Notable work that has driven this wave of interest include increase in computational abilities due to parallel processing in graphical processing units (GPU), convolutional neural networks (CNN), which allowed much faster processing of massive (image) datasets and the availability of large datasets online. It is to be noted that images are grid data with some number of channels. Using CNNs could work on any gridded data where there are some spatial features[23]. Since 2018, there has been significant work done in the weather prediction field using AI. In 2018, Dueben and Bauer showed that you can build a NN that can outperform a simple persistence forecast and is competitive with very coarse-resolution atmosphere models of similar complexity for short lead times[9]. Also in 2018, Scher created a deep convolutional neural network (CNN) to emulate a general circulation model (GCM, a numerical model representing the physical processes), training on the GCM which allows it to emulate the dynamics of the model and maintain stability for much longer than Dueben[22]. These two papers were more proof of concept rather than production ready models to replace NWP. They showed that models based on deep learning might, with further development, compete with standard models in the field.

In the last two years there have been even more developments with the emergence of Large AI Weather forecast Models (LWM). In 2024, Ling et al.[13] tried to standardize the definition of LWM in meteorology and came up with 3 rules that need to be met to count as LWM.

Rule 1: Large Parameter Count. The number of parameters can vary wildly but a

general range might be from tens of millions to billions of parameters

Rule 2: Large Number of Predictands: predicting on different levels (such as pressure levels or height levels) and offering detailed information on the atmospheric vertical structure and surface conditions

Rule 3: Scalability and downstream applicability. This might crystallize in predicting cyclones. Often, the teams responsible for creating these models try to show their applicability to predict cyclones when not trained specifically on cyclone data (e.g. GraphCast)[13]. This is done to show the versability of the models.

Before 2022, LWM had been shown to be able to compete with traditional NWP for some specifc cases as well as making predictions quicker, after training. No model had shown that it could in any way completely replace the traditional systems. In early 2022, Pathak et al.[19] presented FourCastNet. FourCastNet uses an Adaptive Fourier Neural Operator model that leverages transfomer architecture rather than the popular convolutional model architechture. FourCastNet matches the performance of standard forecasting techniques at short lead times for large-scale variables and outperforms for smaller variables. It generates a week-long forecast in less than 2 seconds, orders of magnitude faster than standard physical methods[19]. In 2022, machine learning methods were presented that made predictions much faster than traditional NWP, after a one time training (or at least training that wouldn't have to be redone often). These were in some cases performing better than NWP. In 2023, Remi Lam and the GraphCast team at Google introduced GraphCast. This model was able to outperform the industry standard High Resolution Forecast (HRES) produced by the European Centre for Medium-Range Weather Forecasts (ECMWF). This model as the name suggests leverages graphing connections rather than traditional grid like data structure. The base data is given in latitude and longitude degrees at a resolution of 0.25 degrees. This means points are closer to each other at the poles. Using the graphing structure is supposed to help with bias incurred as a result of this[12].

There has been a lot of progress made over the last 6 years (since 2018) and especially in the last 2 years (since 2022)[13]. The progression from machine learning methods being an interesting idea in the field of numerical weather predictions, to outperforming the standard NWP has been remarkably quick. Two years ago, machine learning methods were able to predict quickly and in some niche cases outperform traditional models. They were not generally competitive with standard weather models. Now they are competitive. It is worth noting that the training of these large models is based on data from traditional large weather models. It will be very interesting to watch what the next few years will have in store for the development of machine learning in weather predictions.

1.2 Methodology and related work

This study looks at data from three sources and an attempt is made to predict the gust factor in a given place in Iceland. It uses reanalysis data, along with elevation data to predict the gust factor. It looks at the data at the point of interest and does not look at the data as a time series. This thesis aims to improve on the baseline model of always predicting the mean gust factor and show that some structure can be learned from reanalysis data about gusts. To do this a neural network was created. Any significant improvement on a base model, that always guesses the mean gust factor, would indicate that the final model has something to contribute.

In 2004, H. Ágústsson and H. Ólafsson[6] looked at the variability of gust factor in complex landscapes. They looked at data from automatic weather stations that measure wind at 10 meters above ground. The data that was studied in 2004 comes from the same source as used in this thesis, but limits itself to a smaller section. They only looked at the years 1999-2001. They looked at three factors and how these three parameters effected the gust factor. These were d_m, D, H, that is direction of wind blowing off a mountain, distance to the mountain and the height of the mountain above the weather station. Their main results were that the gust factor is inversely correlated to the distance from a mountain and correlated to the height of the mountain. The study in 2004 looked at the effect of a dominant point upwind. It did not look at the effects of the landscape more broadly. In this study, landscape upwind is looked at along with landscape downwind. Both the upwind and downwind landscape can influence the gustiness at a certain point [20].

1.2.1 Model architechture

To be able to capture the patterns in the data a neural network was constructed. A NN architechture was chosen as they are known to be able to capture patterns well in complex data and handle high parameter counts. This comes in handy when training on different types of data. It is also easy to construct different types of neural networks and see how they fit well with parts of the dataset. To measure the performance of these models, both to train and test, mean absolute percentage error (MAPE) as defined in Equation (1.2) was used.

$$MAPE = \frac{1}{n}\Sigma_{i=1}^{n}\frac{|y_{predict} - y_{true}|}{y_{predict}} \tag{1.2}$$

This was chosen because the target is the gust factor (the wind gust over the average wind). If the target would have been the wind gust rather than the gust factor then something like mean absolute error might be more appropriate.

1.2.2 Model explainability

Neural networks are often considered as mysterious black boxes[1]. In an attempt to understand the model predictions, methods designed for explainability are used. One such method is Shapley values[15]. Shapley values are calculated as the average marginal contribution of a feature value across all possible coalitions. For any combination of parameters what is the contribution of a given parameter. This means that Shapley values can explain individual predictions. Other machine learning tools, like ELI5 (Explain like I am 5), randomly shuffle a feature and look at the effect on model performance[2].

2 Data gathering and preprocessing

Data was sourced from several streams. The Icelandic Meterological Office (IMO) provided measurements from weather stations all around the Iceland. NWP data was downloaded from Copernicus Arctic Regional Reanalysis dataset (CARRA). A land elevation model was also provided by IMO.

2.1 Automatic Weather Station Data

IMO provided 10 minute measurements from 327 weather stations all around Iceland. The measurements that met the filtering criteria, started in 2004 and ended in 2023 (the year measurements were provided). Of these 327 stations, 212 were from IMO Íslands and placed at 10 meters above ground, while the rest (115) were from the Icelandic Road and Coastal Administration (IRCA) and placed at 6-7 meters above ground[18]. The location of these weather stations can be seen in Figure 2.1. The information that is provided by these Automatic Weather Stations (AWS) is presented in two different type of data files, hourly and 10 minute files. The hourly files are summations of the 10 minute files, with the exception that errors, such as nails, still in the 10 minute files should have been removed from the hourly documents. Nails, are sharp increases from the rest of the data and are unrealistic outliers that are considered measurement errors and are discarded. Each type of document contain the following information: the date and time, the station number (that can be converted to the coordinates using another data set), the average wind speed (f), the wind gust (f_g), the standard deviation of the wind gust, the direction of the wind (d) and the standard deviation for the wind direction. These measurement started at the end of the 20th century, when the first AWS stations was installed. More have been added in the following decades. This thesis does not look at the data as a time series, it tries to make predictions using only the information at a given point in time.

These 10 minute measurements were filtered to only include measurements where the average wind speed was at or above 20 m/s. They were also filtered in such a way that only the highest value in any given 48 hour span is considered. This can be seen in Code Listing 2.1. Any measurement that is within a 48 hour interval

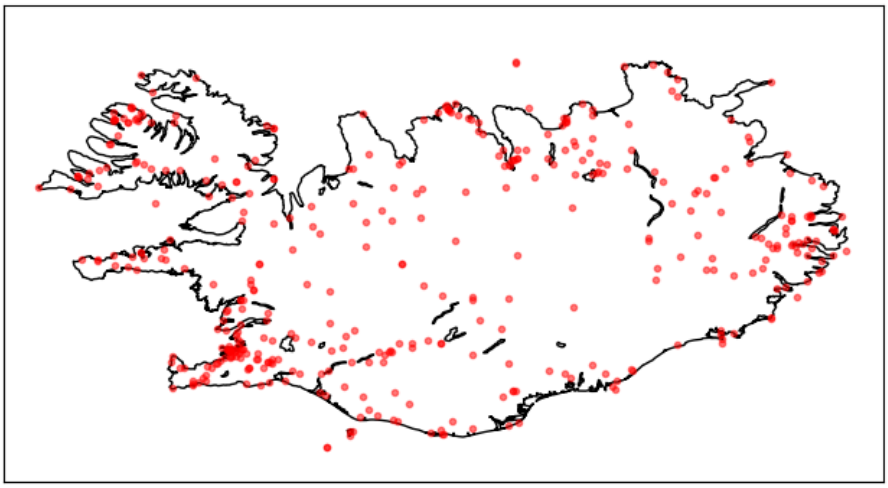


Figure 2.1: Locations of all 412 stations that were looked at in this study. Most of these were from IMO but over a hundred were from IRCA. IMO AWS are placed at 10 meters above ground, while IRCA stations are placed at around 6-7 meters above ground.

centered on of the current highest is removed. This is done for each station and iteratively until all measurements have been removed.

Listing 2.1: Filter all measured points so to only keep the highest value in any 48 hour interval for a given station. For a given station, the highest wind speed recorded is selected. Then any measurement in a 48 hour interval, centered on the highest wind speed, for that given station, is removed. This is then done iteratively until we are left with an empty dataframe.

```
for station in tqdm(stations, total = len(stations)):
    subset_df = vedur_df[station == vedur_df.stod]
    subset_df = subset_df.reset_index(drop = True)

    while not subset_df.empty:
        idx = subset_df.f.idxmax()
        time_of_max = subset_df.iloc[idx].timi

        filtered_data.append(subset_df.iloc[idx])

        subset_df = subset_df[ abs(subset_df.timi
            - time_of_max) >= pd.Timedelta(threshold)]

        subset_df = subset_df.reset_index(drop = True)
```

When these two filterings have been done, the end result is 327 stations that meet these criteria. Of these, 212 belong to IMO and 115 belong to IRCA. The number of observations by year can be seen in Figure 2.2. The first observations occurring in 2004 and the latest in 2023.

Looking at the number of measurements that meet these criteria for a given station, Figure 2.3 shows only one station has over 1000 measurements that fit the criterias.

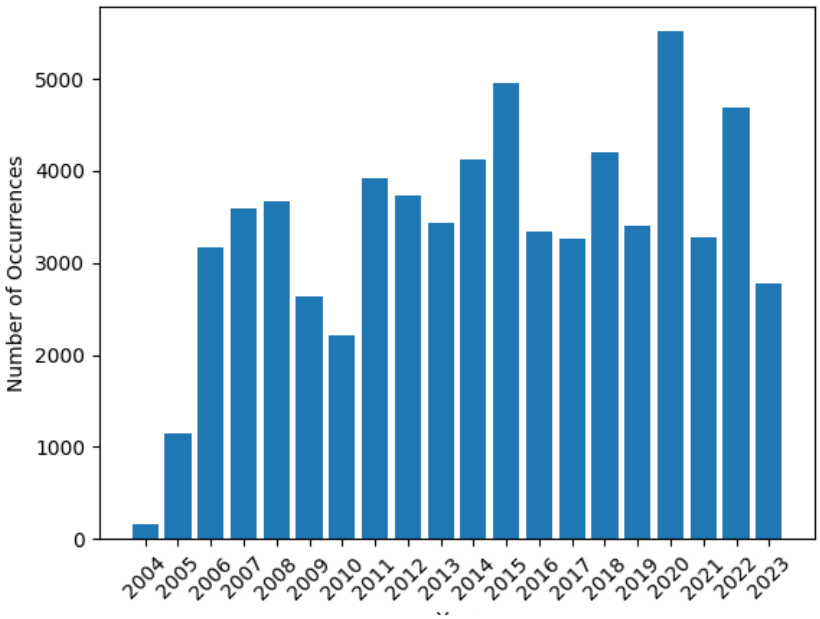


Figure 2.2: The total number of measurements by year after filtering by average wind speed limit of 20 m/s and clearing the neighborhood using filtering method described in code listing. The measurements provided were sparse for years before 2005, as can be seen for 2004. Years before 2005 will thus be ignored.

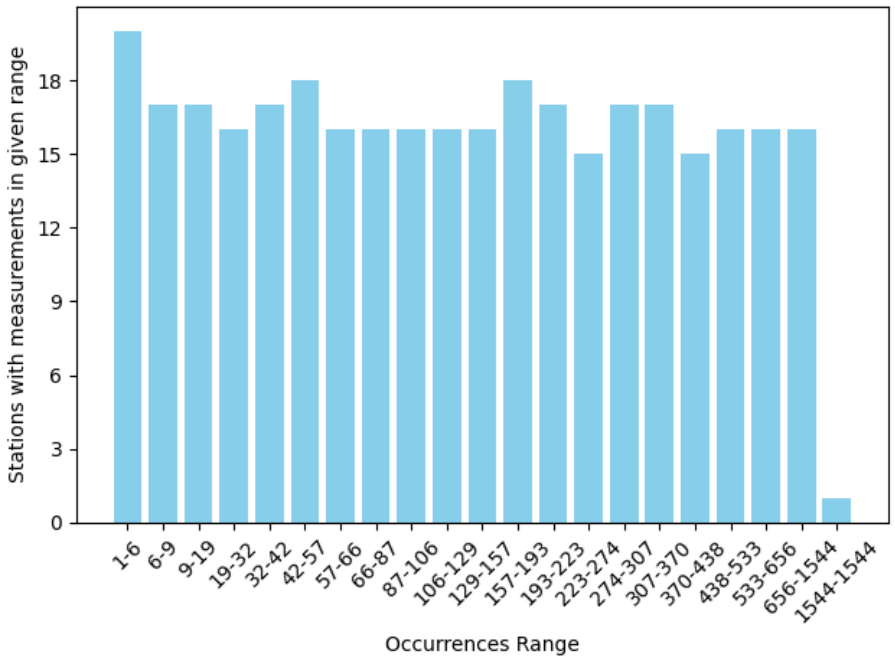


Figure 2.3: The number of stations that have a-b measurements represented in filtered data. That is station x may have c measurements represented and $a <= c < b$. Station x will then add to the count of column labeled $[a, b[$.

2.2 CARRA Data

The CARRA dataset goes back to September 1991 and is currently updated monthly, with a latency of 2-3 months[3]. The oldest IMO data point that fulfills given criteria is from 2004. This is covered by CARRA. A few of the newer points from AWS were not available from CARRA when data was gathered. The CARRA dataset is available for two regions, west and east. Each of these covers a vastly larger area than the area of interest. This leads to having to store a large amount of data, because it is not possible to ask for specific points of interest. To get the data one has two options. Their web interface or using their API client. Using the API client is the only realistic option here, as there were thousands of requests made for different times.

Each request is made by going through all instances of the IMO data after filtering (by mean wind speed reaching 20 m/s over a 10 minute interval). For each such observation in the IMO data, two API calls excluding redundancy are needed. One for the 3 hour interval before and one for the 3 hour interval after, if the data point doesn't fall exactly on one of the three hour interval times. That is if the observation was at 13:00, CARRA data for noon and 15:00 would have to be queried. Date and times were generated automatically from the AWS and requested by calling the

client. Using these datapoints, interpolation was used to get an estimation for the point of the given weather station. The CARRA data contains several types of layers. These are single levels, model levels, height levels, pressure levels. The data for this observation was downloaded from height levels. That is, data was requested at heights of 15, 150, 250 and 500 meters above ground. For each point 4 parameters were requested, wind speed, wind direction, pressure and temperature. Each of these features needed to be interpolated to create data for model to be trained on.

After using this method to request terabytes of data, it was discovered that it was possible to query a specific area. This decreased the size of each file from around 50 MB to around 2 MB. The bottleneck for retrieving the data is the request time. The time the request is queued and running (server preparing the data) before it began downloading. This could range from almost immediate to 30 minutes. This problem was exacerbated by the fact that the climate data store (CDS) was undergoing updates during the winter and spring of 2023/2024, which increased the wait time and sometimes resulted in queries not being responded to. This meant that the time it would take to retrieve the remaining information went from around a day, when the requests were at their quickest, to many months, something that would not be possible given the time frame of the project. Fortunately these problems were only particularly time consuming during mid winter.

2.2.1 Statistical Analysis of Measurements

The data gathered from AWS showed inconsistencies due to significant gaps in data coverage in the early 2000s and 90s. This is shown in Figure 2.4. It is important to note that Figure 2.4 and subsequent figures show number of measurements in logarithmic scale. Late in the process of making this thesis, it was discovered that data was missing. Most data points before 2005 had not been provided. Eventually this missing data was provided but the decision was made to exclude that data due to time constraints. Using only data from 2005 to 2023, means that all years cover the entire year. Thus the data was filtered to only include these years. For some reason there were a hundred or so measurements from 1990.

It is a good sanity check to look at figures like Figure 2.4. If there were massive discrepancies by year, then some data is missing. Another way to look at the distribution of measurement over time is to look at it by month, as in Figure 2.5. The total measurements is evenly distributed over the months. As expected higher winds occur during the winter months, with only a few hundred wind measurements reaching 20.

Another thing to look at is the number of average wind speed measurements by interval wind speed interval. This is shown in Figure 2.6. This figure shows the

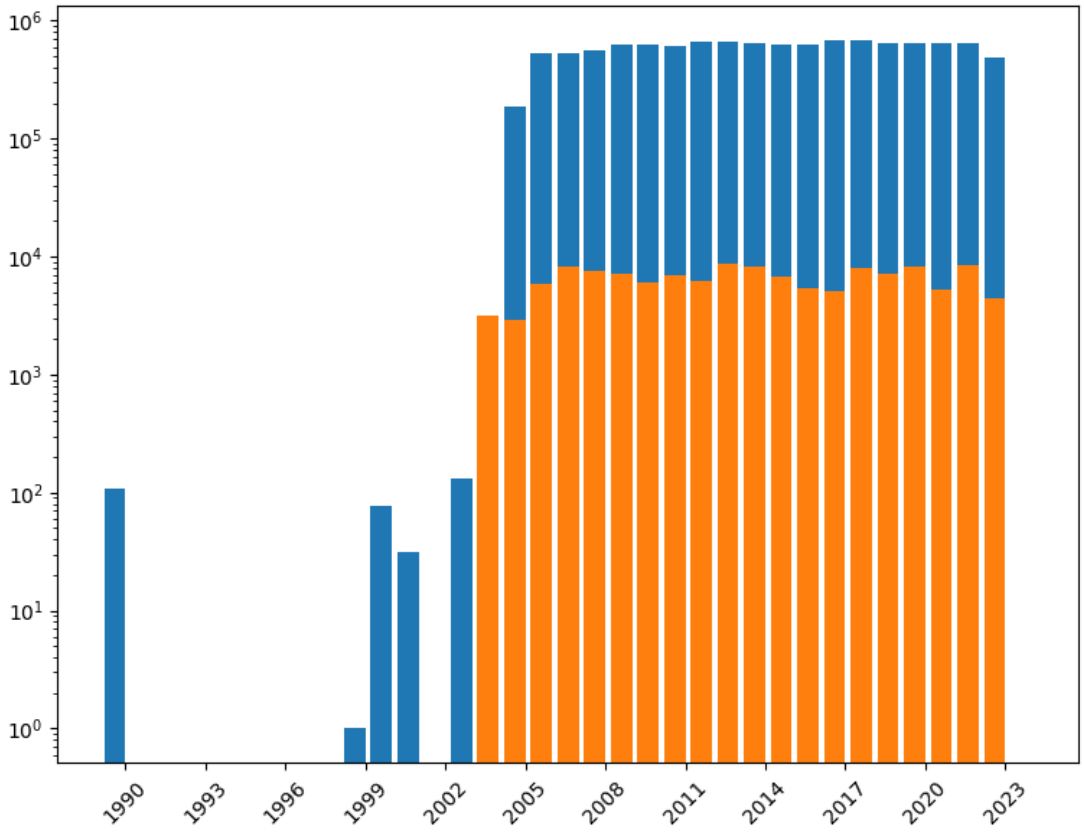


Figure 2.4: Number of measurement data points by year. The effect of filtering by average windspeed limit shown in the orange overlay. It is important to note that the y-axis scale is logarithmic. If this was not the case, then the filtered values would not be clearly displayed.

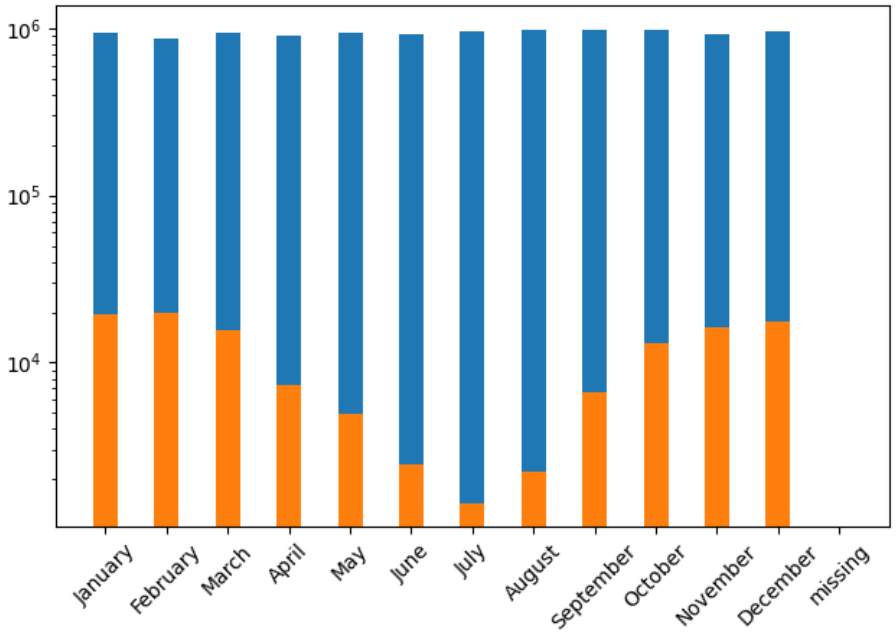


Figure 2.5: Number of measurement data points by month. The effect of filtering by average windspeed limit shown in the orange overlay. This filtering shows us what we would expect to see. We can see that we have fewer extreme wind speeds during summer as compared to winter.

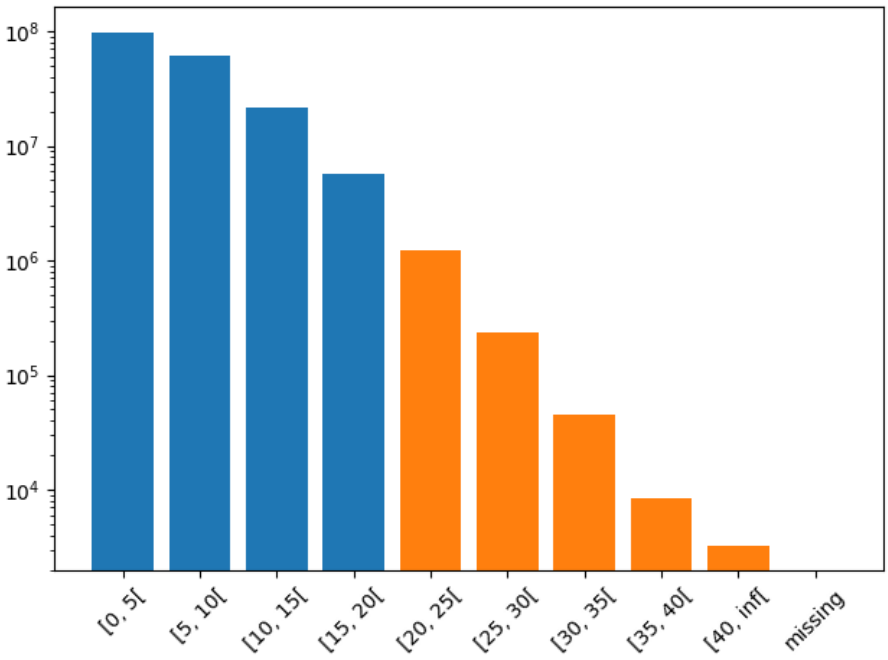


Figure 2.6: Number of measurement data points by wind speed. The effect of filtering by average windspeed limit shown in the orange overlay. The filtering obviously just selects the columns to the right and including column $[20, 25[$.

expected, the number of measurements at higher wind speeds decrease. The orange overlay shows the section that is considered in this thesis.

Looking at Figure 2.7 something strange seems to be happening. The orange overlay shows the number of gust values where the wind speed is over 20 m/s and some of the gust values is lower than that. This shows the relevancy of looking at such figures. These data points, where the average wind speed is higher than the gust speed, are filtered out later in the preprocessing pipeline. Note that there are seemingly no missing values for wind speed, wind gust and wind direction, but there is missing standard deviations for these parameters.

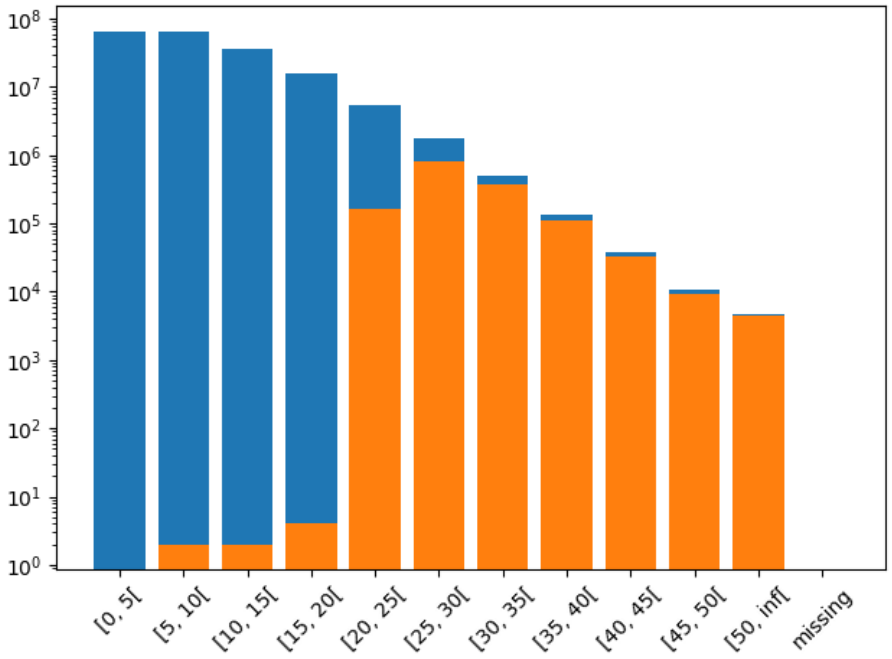


Figure 2.7: Number of measurement data points by wind gust. The effect of filtering by average wind speed limit shown in the orange overlay. Here we might expect to see some discrepancies in the columns. That is, the orange overlays should not completely cover all columns. This is because we select by windspeed and not gust. There should be no coverage below the average wind speed limit of 20 m/s.

2.3 Elevation data

IMO provided a TIFF file containing the elevation of Iceland on a 20 meter by 20 meter grid. This file encompasses Iceland and is around 685 MB. A good amount of time was spent finding a data structure that was best for lookup when trying to find points within a certain area. The country was divided into 10 parts (as only around 13% of the file was able to be read into memory at each time as a part of dictionary object) with boundary boxes. The Python package Rasterio allowed for quick lookup with it's index and the affine transform. Using this package it is possible to quickly look up elevation given coordinates using matrix calculations.

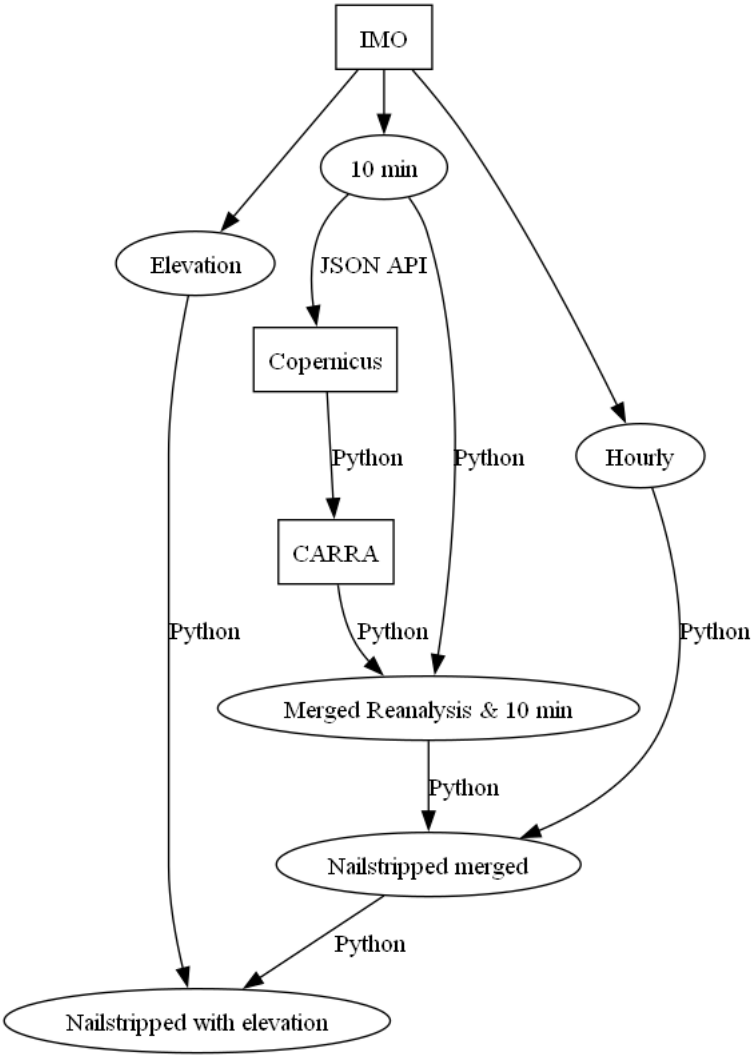
3 Data processing and structure

3.1 Combining data sources

This project used three main data sources, which need to be queried, filtered and combined to prepare the data for use in the models. When working with hundred of thousands of rows, the efficiency of the code is very important. Iterating through those rows might be necessary at times but will increase the time exponentially as compared to using vectorizing methods were possible. The three data sources were all in a different format. Measurement data from IMO was in text files, elevation data was in GeoTiff and reanalysis data from CARRA was in a GRIB format. To use the data to train, these three data sources needed to be combined into one file. This was done based on the measurement data from the IMO. A limit was set on the average wind speed and it was used to select measurement points. Along with the average wind speed having to be above a certain limit, to ensure that the same weather for the same location is not duplicated. The data from IMO was supplied for 10 minute increments, while CARRA data is in 3 hour intervals. This means that to use the CARRA data to predict the measured values from IMO, temporal interpolation would need to be done. Along with the temporal interpolation, note that the CARRA data is given in a rectangular grid where the distance between each point is around 2.5 km while the the information from the IMO is given at specific locations. The elevation information was given by a 20 by 20 rectangular grid that covers Iceland. When combining these data sources an interpolation method needs to be decided upon. Here linear interpolation was applied, both temporally and spatially. The choice of interpolation method, although potentially impactful on the results, was not specifically addressed in this study.

The procedure of combining these sources was as follows and can be seen in Figure 3.1. The measured data from the AWS is filtered by using a limit on the average wind speed. The gust factor generally drops with increased wind speed (although not always dependent on factors such as the landscape [20]). Even so being able to predict the gust factor is more important for higher average wind speed due to then higher wind gusts. After this stripped dataset over every AWS has been created it is used to query the CARRA data by using their API. The CARRA API needs to be queried for given hours, days, months, years and a given area. That

Figure 3.1: A flow chart showing how data sources were combined



is, if queried for a given hour, it returns that hour for every day that is queried. Similarly if queried for a given day, it returns that day for every month. In light of these restraints, it was decided to query month by month. Querying only the days needed (both the days included in the 10 minute measurements and the days needed for interpolation if close to midnight) but every hour of the day (UTC 00, 03, 06, 09, 12, 15, 18 and 21) and then take these values and interpolate for the 10 minute values. After querying and downloading the data for the height levels and variables requested, points of interest are interpolated and values stored in a pandas dataframe. After this the downloaded data is discarded and the next month is queried. This drastically decreases the amount of data that needs to be stored as compared to downloading the entire area and keeping all the data points (a reduction from several terabytes to less than a gigabyte).

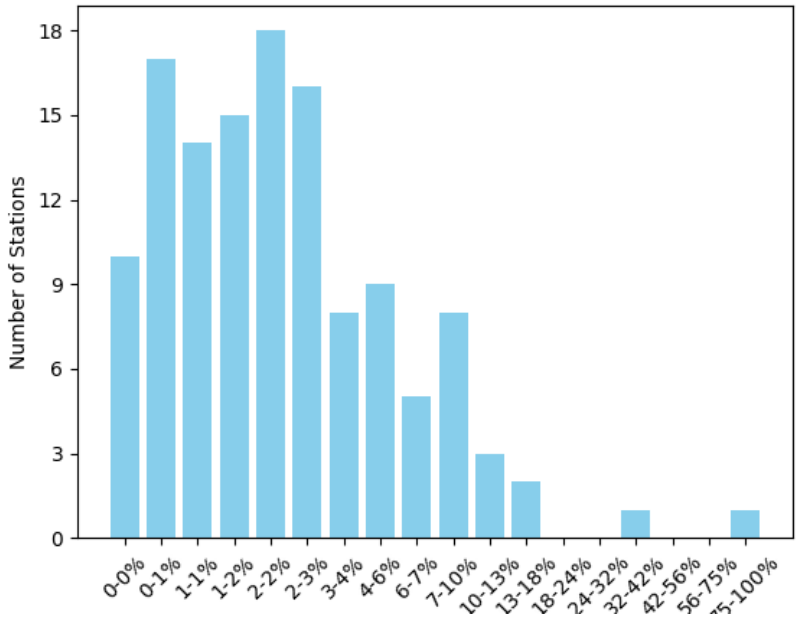


Figure 3.2: The number of stations whose number of nails are in a certain percentage range of the overall

Once CARRA data has been merged with AWS data, using station and time columns, then this combined file needs to checked for nails. This is done by using the hourly data (which is supposedly error free). As the data has been filtered in for the highest average wind speed in a 48 hour interval, the hourly data can be used to find nails. The hourly data is combined with the merged AWS and 10 min data. Then filtering is applied on the average wind. If the average wind speed differs the rows are dropped. This can be used to look at which stations have high percentage error. This can be seen in Figure 3.2.

Figure 3.2 shows that most stations have nails in less that 10% measurements. The stations that have higher than 10% error, have only few observations that meet the criteria ($<$ 10 measurements) and are ignored.

The elevation data comes in a GeoTIFF file that covers Iceland. It is a rectangular grid of resolution 20 meters. For every point of interest (every weather station), the elevation of that given point along with other points surrounding the weather station is retrieved. For each point retrieved interpolation needs to be done. This is done in a similar manner to the interpolation of the CARRA data. The four points bounding the point of interest were used to linearly interpolate the value of the point of interest. This information is included in the training data as the landscape is known to influence both the average wind and the gustiness [20].

3.2 Data Structure

Once data has been retrieved for all three sources and processed, including interpolating values, it needs to be made ready to use by the model, for both training, validation and test. The starting point is a dataframe that contains measured information from AWS. This includes the average wind, the wind gust, wind direction along with the station number and coordinates. When selecting the CARRA data certain height levels are chosen, which heights above ground the reanalysis parameters are requested. These present as separate lines in the CARRA dataframe. Information for one observation is represented in as many lines as height levels requested in the reanalysis data. These rows need to be combined on the position (the weather station) and time so that each row contains all reanalysis information for a given measurement of a weather station. When this is done it is possible to combine the AWS IMO data and CARRA reanalysis data on the location and time columns. The last data source is the elevation. A couple of different sections of land around the weather stations have been looked at. A sector of a circle looking upwind, two sectors looking upwind and downwind and a circle around the point. In any case the points, that represent these sections, were selected as shown in Code Listing 3.1. A range of angles are defined based on the wind direction d at some distance from the given point. This means that the resultant points (equal in number to the length of angleRange by k) form sectors at several distances from the given weather station.

Listing 3.1: Sector elevation points generated

```
angles = [(angle + (90 - d)) * pi/180 for angle in angleRange]
length_rng = [(exp(i * log(n + 1)/ k) - 1) * 1000
                  for i in  range(1, k + 1)]
points = np.array([[(X + l * cos(angle), Y + l * sin(angle))
                    for angle in angles] for l in length_rng])
```

The result is a dataframe that has measured data from AWS, which gives us our target, reanalysis data from CARRA, which gives us weather variables to train on, and finally elevation points in the landscape to include in our training data. An example of what the data looks like can be seen in Table 3.1.

Looking at Table 3.1 note that the last column shown before ellipsis contains the first principle component in principles component analysis. In total there are 10 columns representing the principle component analysis of the elevation data. Also note that the first four columns represent two variables that describe the stability of the air. These are the Richardson number (Ri)[24] and Brunt–Väisälä frequency[10] (N), and are calculated using Equations (3.1) and (3.2)[6]. These values are calculated using reanalysis data at three different height levels. Thus Ri_{01} refers to the Richardson number calculated between height levels 0 and 1. Exactly the same notation is used with the Brunt–Väisälä frequency. The height levels used were 15,

Table 3.1: An example of structure of data used to train model. Data points include the derived variables Ri and N, the elevation of the station, direction of wind and relative direction of the wind (that is the direction of the wind relative to center of Iceland), along with some combination of wind speed, pressure and temperature at the different height levels. Finally there are the elevation points around a given station. The example below only shows data derived variables elevation and relative corner.

Ri_01	Ri_12	N_01	N_12	station_elevation	relative_corner	elevation_point_0	...
-1.18e+00	2.67e+04	-8.57e-06	6.78e-05	3.34e+01	2.73e+00	1.36e+01	...

250 and 500 meters above the ground and the three different sets of height levels were (15 m, 250 m), (250 m, 500 m) and (15 m, 500 m).

$$Ri = \frac{g \cdot dT \cdot dz}{T_{ave} \cdot dU^2} \, [] \tag{3.1}$$

$$N = \sqrt{\frac{g \cdot dT}{T_{ave} \cdot dz}} \, [\text{Hz}] \tag{3.2}$$

Here, g is the acceleration due to gravity, dT is the temperature difference between the two height levels, dz is the elevation difference, T_{ave} is the average temperature (that is the average of the two temperatures in the height levels) and dU is the wind speed difference between the two height levels. Both of these numbers provide some insight about the stability of the air. A lower value for the Richardson number indicates a higher turbulence. A typical range of values could be between 0.1 and 10, with values below 1 indicating significant turbulence[24]. When the square of the Brunt-Väisälä frequency is negative, then the air is unstable (an air parcel will move away from its original position)[10]. These are derived factors from the reanalysis data and as such there shouldn't be a significant information gain using Ri and N as opposed to having the raw data. However, including these factors instead of every reanalysis variable requested might speed up training as well as making the model more easily explainable with the use of Shapley values or other tools for explainability. Using Shapley a feature importance value is attributed to a given feature by creating all possible permutations of any possible length (up to number of features) and seeing how the predictions are skewed when the given parameter is included or excluded. This needs to be done for all parameters. The time complexity of this is very high (2^n coalitions)[15]. Most implementations use some approximations, which still can take a considerable amount of time for models with a high parameter count and many examples.

4 Model architecture

4.1 Model structure

The structure of the neural network is such that it contains some number n of fully connected layers and batch normalization for each layer, along with regularization. All of these layers have the same number of units. The last layer has a dropout of 50%. In addition to these layers there is one more output layer. This is simply a dense layer with 1 unit. A grid search was performed to determine the hyperparameters that minimize the loss. Hyperparameters are parameters that are set before the training begins[4]. These hyperparameters include number of units in each layer, number of epochs to train for, number of layers, batch size, optimizer and penalty to enforce in the regularization. The possible combinations tried can be seen in Table 4.1.

Table 4.1: Hyperparamter search with best performing combination shown. Hyperparameter search was done using hyperband algorithm that initially searches randomly for the best parameters but then hones in on the what is working and as such is neither exhaustive nor completely random. This means that an upper limit will nont be set on the number of combinations to try like with randomsearch.

Parameter	Range of values	Selected
Layers	min_value = 4, max_value = 15, step = 1	**11**
Units	min_value = 32, max_value = 512, step = 32	**64**
Penalties	min_value = 1e-4, max_value = 1, sampling = log	**0.000168**
Epochs	min_value = 50, max_value = 1000, step = 50	**750**
Optimizers	Adam, RMSprop, Adamax	**RMSProp**
Activation	ReLU, ELu, Softmax	**eLU**

4.2 Feature selection and importance

Using the three datasources, a model is trained. The data from IMO only represents the ground truth and is not used as part of the training data. The gust (f_g) and

wind speed (f) are used to calculate the target gust factor. All weather parameters used in predictions come from the CARRA reanalysis data, with variables wind speed, wind direction, temperature, pressure. All of these variables are queried in three height levels (15, 250, 500 meters above ground) at the location of a given weather station. The second data source for the training data, is the elevation data. Using the GeoTIFF file that IMO provided, several different landscape elevation distributions were looked at. A sector upwind, a sector upwind and downwind as well as a circle surrounding a weather station.

To begin with all the available parameters were used to train the model along with the derived variables. This is done to be able to then use tools such as Shapley to see which features are impacting the predictions. Using Shapley values to see which features, will then enable the exclusion of features that don't affect the model prediction and simplify the training data. This means starting with much higher dimensionality of our training data and reduce it as much as possible without impacting our results. This crystallizes in the landscape points. Starting with n points that describe the elevation of the landscape around our weather station. This n might be in the hundreds. Looking at concentric circles outward from a given weather station, there might be 10 circles (largest with radius 20 km), each with 72 points (5° spacing) and end up with 720 points for the landscape. This might slow down training. In addition, certain landscape features might be more important than others. There might be a large mountain up- or downwind of a given weather station, that might be the most important feature. To address this principal component analysis (PCA) is used to try to reduce the dimensionality of our landscape elevation data. PCA is a statistical procedure that allows the summarization of multivariate data to lower dimension and thus can ease visualization or increase speed of training a complex model with only minimal adverse effect to performance[21].

Using all the variables from CARRA, along with the two derived factors (Ri, N) and two sectors (upwind and downwind) downscaled to 10 features using PCA, the feature importance can be seen in Figure 4.1. The waterfall graph in Figure 4.1 shows that the Richardson number for higher levels is most important.

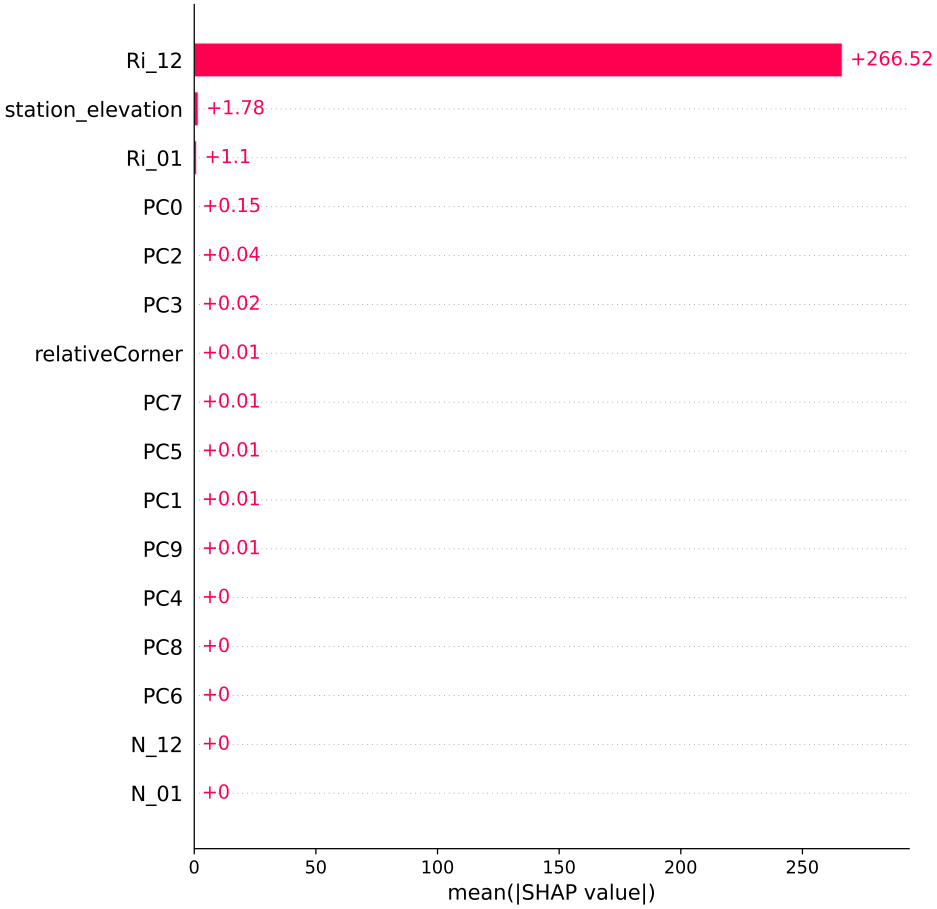


Figure 4.1: Feature importance of a neural network as described with 5 hidden layers and 256 units in each

5 Results

5.1 Results

A baseline model was constructed. This model looked at the gust factor for some training data and took the average of this and predicted this average everytime. The baseline model gave an average error of around 6.54 %. This sets a goal. A model that does not significantly improve on this baseline suggests either failure to capture essential patterns in the data or that the data itself may lack the necessary information for substantial improvements upon the baseline. Using the previously described neural network architecture a mean absolute percentage error of 4.78 % was achived. This is some improvement upon the baseline error, with a decrease in error of around 1.76 %. The power generated by wind mill increases with wind speed cubed [5]. The highest wind gusts in Iceland are around 70 m/s. Knowing the gust factor to 2 percentage points better than before can allow for the anticipation power generated by wind mills in the kilowatts per square meter of swept area.

5.2 Discussion

The error improvement is modest in absolute terms but represents 21% over the baseline. This would indicate significant improvement. A point to consider, that might heavily impact the results, is that the reanalysis data, sourced from CARRA, represents some kind of 3 hour average and therefore will be shifted towards lower values for wind speed when compared to the 10 minute measurements provided by IMO. The baseline model uses measured information to calculate the average gust factor, while the model does not have access to any measured data directly (the reanalysis data is trained on some measured data). The expected shape of the measured wind shape distribution, after filtering and nailstripping, is a bell curve with a steep tail to the right and no tail to the left and this proves to be true. If the reanalysis data would closely follow the measured wind speed, the same distribution should be observed. This is not the case. The distributions can be seen in Figure (5.1)

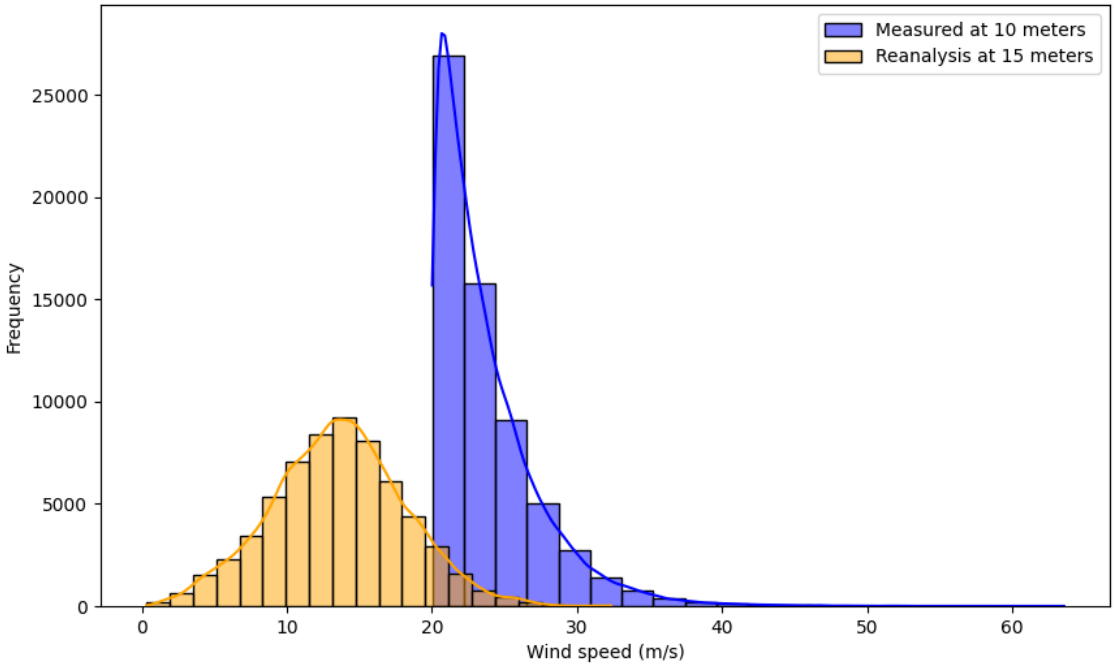


Figure 5.1: Measured wind speed and corresponding points for reanalysis. The two distributions vary greatly. Some variation is expected, as the reanalysis represents some kind of average.

Using any information from the measurement data to improve the data, would induce data leakage. Any simple mathematical transform would not be expected to help the model as the model would simply be able to learn the pattern. A simple check was done to confirm this. The reanalysis wind speed was transformed using code shown code listing 5.1.

Listing 5.1: Custom transform method

```
def custom_distribution_transform(x, threshold=20):
    max_decay, max_growth = 1.2, 6
    decay = lambda z: 1 + (max_decay-1)
        * np.exp(threshold - z)
    growth = lambda z: 1 + (max_growth - 1)
        * np.exp(z - threshold)
    amplification_factors = np.where(x <= threshold,
         growth(x), decay(x))
    y = amplification_factors * x + 8
    return y
```

This shift changes the shape of the reanalysis data distribution and shifts it to the right. The updated distribution can be seen in Figure (5.2).

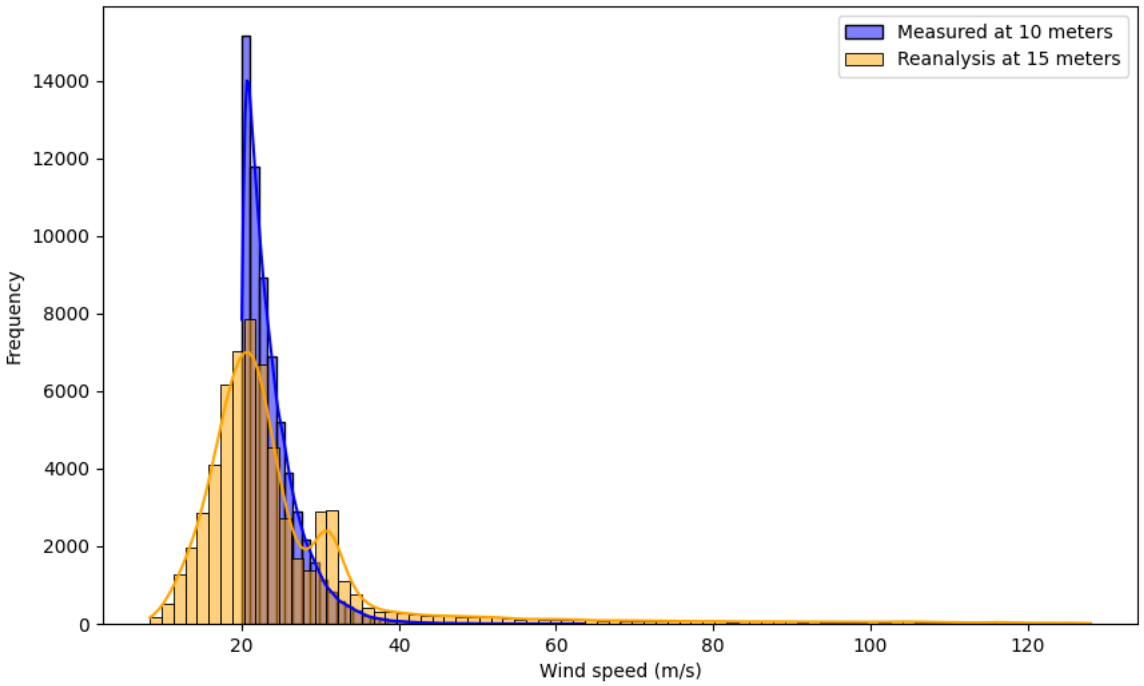


Figure 5.2: The distributions of the transformed 15 meter height wind speed of reanalysis data, along with the 10 meter measured wind speed.

Training two identical models, one with the transformed reanalysis data and the other with original reanalysis data, results in two near identical errors, as expected. As no new information was input, the data was only manipulated, the model evaluates to the same error. The model that used the transformed reanalysis data was slightly quicker to converge.

Looking only at the error, the performance of the model can be gauged. However it does not help us explain why the model made these predictions. Here feature interpretation can be of great help. Using Shapley values and the python SHAP package, the contribution of each attribute can be given a value, as previously mentioned. Figure **??** shows the average contribution of each feature.