# Introduction:

The aim of this project is to use machine learning classification to determine weather types from ground based observations. Machine learning classification is a way of teaching a computer to differentiate between certain given classes, in this case weather types, given certain features, in this case ground observations. For this to work there needs to be a correlation between the features and the classes. This report will explore whether ground observations can be used to predict the weather type and how precise it can do it.

# Data:

For this project, a set of data was provided. The data consist of ground based weather observations made from 126 weather observation stations. The observations are made hourly during the period from 24th of October to 7th of November. This gives 45178 data points. Each data point consists of nine observational features, three locational features, one time feature and a weather type

Some of the data provided had missing values. These values were all given the same value to keep the dimensions of the data points equal, and so they could be further modified if necessary.

The data was given in two files, one basic and one advanced. The basic file only had three categories for weather types whereas the advanced had eleven types.

# Methods:

One class and one script was provided for this project. The class, Weather, holds all the data from the ground based weather stations and enables extraction and manipulation of that data. The script, FeatureExtract, enabled the creation of the weather object, manipulation of it and the possibility to choose which features were kept for further processing. Two methods were added to the Weather class to be able to extract the features selected in FeatureExtract from the weather object.

Four classifiers were used in this project; decision tree, neural network, nearest neighbours and random forest. All the classifiers are from the sklearn python package.

A decision tree classifier is a classifier that attempts to sort the data by learning simple true or false decision rules from the features. This is a so called white box model because it enables a user to look at the model and understand the decisions made. As can be seen in the figure, a decision tree has many boxes, or leaves. When the decision tree tries to predict a class from a set of features it propagates down the tree, from leaf to leaf, until it reaches a final leaf which defines it’s predicted class.

An artificial neural network classifier consists of an input layer, a set of hidden layers, each containing neurons and an output layer. The neural network learns to predict a class from a set of features by applying different weights to the features at each neuron. The input layer represents the features, where each neuron in the input layer is a feature. Each feature is then sent to all the neurons in the first hidden layer where they are applied a weight and summed up. Each hidden neuron has an activation function, meaning that the sum of the weighted inputs need to be a certain value for the neuron to activate and send a signal to the next layer. This propagates through the network untill the output layer receives an input which decides the output class. This is a black box model, meaning that it is hard to understand the decisions made by each neuron.

A nearest neighbour classifier tries to predict the class by looking at the training data and picking out a predefined number samples that have features that closely resembles that of the input. In a way, it is grouping the training data in n (features) space with boundaries that defines the class. A nearest neighbour classifier is therefore different from many other classifiers, it doesn’t try to make rules that decides what the output will be but instead looks at the training data and compares it to the input features.

A random forest classifier creates a set of decision trees, each with a different subset of features and weights. The splits are also decided by choosing the random best split instead of the best. This creates a forest of decision trees that will each produce an output class. The class that is finally returned is the one that the majority of the trees produce.

A set of optimising techniques were used to try and increase precision and reduce overfitting.

Firstly, the data was standardised with the sklearn standardScaler. This means that the range of the data was changed so it’s mean became zero and its variance became one.

A grid-search was done on all the classifiers, meaning their initializing parameters were changed to find the ones that gave the best results for the given dataset.

A method for over sampling the data was implemented. This method looked at the class imbalance in the training data and made copies of the features and class of the underrepresented classes until all the classes were equally represented.

A method for comparing the results from two or more classifiers trained and tested on the same data was implemented. This method would keep the testing results that were the same for all the classifiers. If it was not equal in all the classifiers it would keep the results that most of the classifiers agreed on, and if none of the classifiers agreed it would randomly pick the result from one of them, with all classifiers weighted equally.

# Results:

When comparing the classifiers there are two things that were looked at; precision and recall. The precision of a classifier is the ratio of true positives to false positives. I.e. what fraction of the predicted classes are actually that class. The recall of a classifier is the ratio of true positives to false negatives. I.e. what fraction of that class did the classifier predict correctly.

From the classification report of the decision tree, using the features; Temperature, Visibility, Pressure, Pressure Trend, Humidity and Dew Point, and no optimising methods we got an average precision and recall of 0.69. There is also a big difference in the precision and recall between clear, cloudy and precipitation. This means that the classifier is overfitted to cloudy.

By standardising the data, performing a grid-search to find the best initializing parameters for the classifier, over-sampling the data, and using the features that, through trial and error, proved to give the best results (Temperature, Visibility, Pressure, Pressure Trend, Humidity, Dew Point, Wind direction, Latitude, Longitude) we see big improvements.

Both precision and recall improved for all classes, and the over fitting decreased some but is still there. This can be seen easier by comparing the confusion matrices.

To get better results it seems other classifiers need to be tested. Below is a graph of the results from them.

From these results, the most interesting ones are the results from the random forest classifier and the results from the result comparison method applied to the three classifiers when the data is over sampled.

The random forest classifier produces the results with the highest avg/total precision and recall, whereas the result comparison method produces the least overfitted results