# CMPT 318 Project

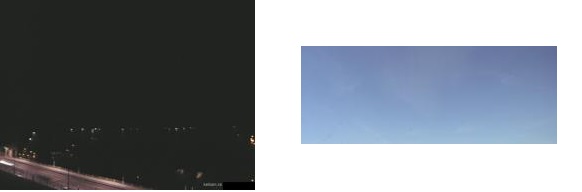
## Can the weather be accurately classified based on simple vison classifiers?

In this project, I attempt to guess the English Bay weather based only on simple vision classifiers of the image. I chose only classifiers that make sense based on simple ways humans would classify images by looking at them, using their colors, brightness, and “busyness”. Using these simple parameters, I experimented with various simple machine learning techniques to find the most successful. Then, I analyzed the problems with this approach and ways this program could be approved.

## Data cleaning and formatting

To extract the relevant data for the project, I wanted to match all the weather descriptions in the weather excel files to their relevant images. To do this, I first read in the columns describing the date, time, and weather of each image from the excel files. Only some of the times given had weather description, so I deleted each row that had no description. In retrospect, I probably could have done some sort of interpolation of the weather conditions in order to get more data, especially when the same weather conditions persisted throughout the day.

Then, for each row of the weather data I searched through the image files to find the corresponding image. Some rows had no available image, so I deleted those. From this set of image files, I decided to limit my analysis to the sky portion of the image, so I got rid of the half that showed the city. I also removed all of the images below a certain brightness, since it didn’t make much sense to analyze images of totally dark sky.



*Example of removed image Section of the image that I worked with*

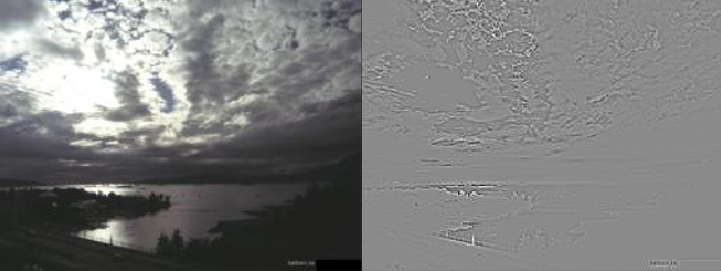
## Classifier definitions and calculations

I used 3 main classifiers to try and tell the different weather patterns apart: Brightness, Colour, and Gradient. I wanted to use classifiers that were easy to relate to human visual systems. Color seemed like a no brainer, so I started with that.

For colour, I wanted to have a measure that was brightness invariant, so that a pixel with colour (50, 50, 100) wasn’t much less blue then a pixel of color (100, 100, 200). The formula I used was dividing the blueness from the total brightness. In this case, a white pixel would have approx. brightness of .33 and a blue pixel would have brightness greater then .33. For each image, I took the average of each pixels blueness aggregated into one value for each image.

Brightness seemed like another core part of our visual system, as well as being even simpler to implement. For brightness, I simply added up the values of each pixel to obtain a single number for each image. This brightness parameter was also used for filtering images below a certain brightness, as nighttime and close to nighttime pictures polluted the image pool with strange data points.

M most complicated classifier was the gradient classifier. I tested some machine learning algorithms, and the images they had the most problems with using my current parameters were telling those that were cloudy or partly cloudy apart. I wanted to use a parameter to represent the “busyness” of the image, how much was going on. Obviously, in an image with all clouds or all sky, the image would be less busy. But in the case of many patchy clouds, the image would be busier. To represent this idea, I used a concept called the gradient. The gradient of an image is 0 at pixels that don’t change on any side, and increases in magnitude for large changes of intensity. An image and its gradient are shown below:



*Gradient of image with highest measured gradient from dataset*

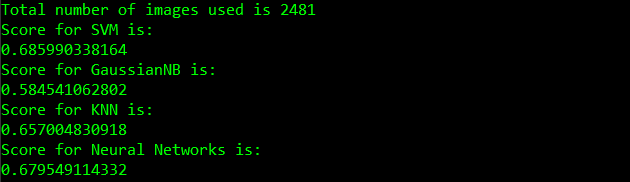
In this image, the lighter and darker pixels represent high positive/negative gradients respectively, and the grey pixels represent no gradient. To get a single number, I took the absolute value of the gradient values, and summed them over the sky pixels. Although this measure is a bit more complicated, I still think it represents a real part of the human visual system.

Lastly, for the sake of ease of use, I included the measures of redness and greenness as well, since it was trivial to include them after implementing the blue pixels, although they only increased accuracy by a few percentage points.

## Machine Learning Models and Results

Although images are complicated, my methods reduced them to a simple 5 number classification, very simple to plug into various machine learning methods. Before doing anything with the data, I used StandardScaler to scale it all to the same size, since the colour values were a different order of magnitude then the others.

The methods I tested were Naïve Bayesian, K Nearest Neighbors, Support Vector Machines, and the neural network classifier used in class, MLPclassifier. The approximate percentages for each classifier are given below, they tended to vary by about +-3% per trial, below is shown one of my higher score trials:



I used SVC for all of my setup and tuning, so I was very surprised when KNN and neural networks obtained an almost identical accuracy score. I believe that the reason these accuracy scores are so close is because they are reaching the limitations of my simple image descriptors to tell the weather apart.

The score for Neural Networks tended to be about 2 percent higher then the next highest score, leading me to think that it could achieve the best results with some tuning. However, I know nothing about neural networks, and just changing the parameters slightly didn’t seem to change much. This would be something to further explore in a larger project.

## Discussion and Improvements

Due to the three best machine learning methods achieving similar results, I believe that using my simple descriptors, only an accuracy of about 70% is possible. There is simply not enough data in those 5 simple numbers to tell more of them apart. However, I think this is a surprisingly high accuracy for such simple descriptors. There are many more complicated parameters that could be added including magnitude of Fourier transform frequencies, numbers obtained from dimensionality reduction, and pixel counting and contrasting methods. However, from just these simple visual ideas, I was able to achieve a decent accuracy.

## Accomplishment Statement

-Created a program to predict the weather based on image taken of sky by training a machine learning model, in order to test the limits of simple visual descriptors in analyzing images

-Imported and cleaned data from historical weather logs with python library Pandas in order to match weather patterns to particular images

-Extracted various visual descriptors such as measures of colour, brightness, and busyness from database of images using python library OpenCV in order to classify the images and input them into my machine learning model