Weather Prediction System
Inf2B Coursework

Matriculation Number: s0943941

Name: Scott Hofman

Weather prediction is an unsolved problem in both statistical analysis and meteorology. Despite this, approximations exist for effectively determining the forecast of the weather. One such method is the use of Bayesian statistics. More specifically, Bayes Theorem (fig. 1)

Bayes Theorem: $$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}.$$ (fig. 1)

can predict the general forecast of future days based on probability. This approach is known as the Naïve Bayes model; "naïve" for the base assumption of independence it makes. To use Naïve Bayes, we assume each datum independent of the other data. By employing Bayes Theorem and assuming weather to be independent, we get the following result. P(A) is the probability of a given weather event, P(B) is the probability of each weather condition, (A|B) is the probability we want to find, and P(B|A) is what we must calculate. For the rest of the paper, weather event/event will refer to fog, snow, rain or none and weather condition /condition will refer to mean temperature, maximum temperature, minimum temperature, dew point, average humidity, maximum humidity, minimum humidity, sea level pressure, wind speed, maximum wind speed, maximum gust speed, and visibility (twelve in total).

To begin, first train the data using a large enough sample space to draw conjectures. This numerical data must be classified as probability data points (P(B|A)) before it can be used. There are two basic approaches to classifying data. I considered discretizing the variables based on how their values related to the mean (for example, mean temperature would be high, medium or low) and then calculating the probabilities based on this. However, this approach takes multiple iterations to learn the fit of the data curve and is therefore unsuitable for our small training data.

Thus, I decided to employ the secondary method of Gaussian distribution. This method requires the assumption that the condition data sets (Mean Temp, or Sea Level Pressure for example) are normally distributed. Despite some data points lacking a normal distribution, the small sample space allows this conjecture to hold. If the training data size increased, it would make sense to switch to a discretization of the data for classification.

Since the data size is small and continuous, we can model the probabilities on a Gaussian distribution. (fig. 2)

Probabilities of Gaussian Distribution: $$P(x=v|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}}\, e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$ (fig. 2)

This formula requires the mean (μ) and variance (σ^2) from each condition of the training data to calculate P(B|A) from Bayes Theorem (fig. 1) . To compute these values in respect to the individual events, we first split the training data into four lists – one for each snow, fog, rain and none. These lists, containing a list of the entire day's numerical weather conditions, pass through another loop that sums both the first element of each list and the square of said element, followed by the second element and so on. This continues, calculating the variance with (fig. 3)

$$\textit{Variance: } s^2 = \frac{\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2/n}{n-1}.$$ (fig. 3)

and the mean by dividing the sum by the length of the list (i.e. number of snows) appending each mean into a list of means and each variance into a list of variances. The n-1 ensures an unbiased estimate of the variance.

After computing a list of means, variances and the prior probabilities (fig. 1 P(A)) (occurrences of single weather event divided by the total events) from the training data, we no longer need said data. We move onto the testing data and apply the Gaussian distribution to get each probability (fig. 1 P(B|A)). Another loop over the testing data computes the probability of P(B|A). For a single day and a single event, it iterates over the twelve data points (v in fig. 2), the mean for the event (μ), and the variance (σ^2) and calculates the probability using the second formula. This iterates through each condition data set for a single day given an event, until all probabilities have been determined.

For example, we will calculate sea pressure given fog. The mean of Sea Pressure Level given fog as calculated from the training data is 1011.549. The variance of Sea Pressure Level given fog is 67.125. For the first day of the testing data, the value of Sea Pressure Level is 1020.48. Plugging these values into the Gaussian distribution, we receive the P(X=1020.48| fog) =0.0268. We compute this twelve times, once per condition.

Because we have assumed independence as the Naïve Bayes Assumption, the laws of probability allow us to multiply the values to attain a total probability. In Bayes Theory (fig. 1), P(B) is a set of conditions related to each other as $P(X_1, X_2, X_3, X_4 \ldots X_{12})$. Assuming independence implies $P(X_1, X_2, X_3, X_4 \ldots X_{12}) = P(X_1)*P(X_2)*\ldots*P(X_{12})$. Each data set probability given a weather event is multiplied together as $P(X_1 \mid A)*P(X_2| A)*\ldots*P(X_{12}| A)$. This result is then multiplied by P(A) (prior probability) to give the top half of Bayes Equation. The bottom half, P(B) can be ignored since its ratio scaling (a positive constant resulting from normal distribution) is applied to each calculated probability and thus is negligible under comparison.

Once we multiply the twelve values calculated via the Gaussian distribution function with the prior probability, we have the probability of a single weather event. We then calculate this probability for the other three events. The result is 4 different probabilities for a given day, one for each event. From this, we can easily calculate our prediction – the weather event with the greatest probability.

Utilizing the code, the test data received a percentage of 41%. The prediction I received was:

[‘Snow’, ‘Snow’, ‘Fog’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Fog’, ‘Fog’, ‘Snow’, ‘None’, ‘None’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Snow’, ‘Fog’, ‘Fog’, ‘Fog’, ‘Fog’, ‘Fog’, ‘Snow’, ‘Snow’, ‘Fog’, ‘Snow’, ‘Snow’, ‘Fog’, ‘Fog’]

Compared to the actual result of

[‘Snow’, ‘Snow’, ‘Snow’, ‘Fog’, ‘Snow’, ‘Snow’, ‘Fog’, ‘None’, ‘None’, ‘None’, ‘None’, ‘None’, ‘None’, ‘Rain’, ‘Fog’, ‘Snow’, ‘None’, ‘None’, ‘Snow’, ‘Snow’, ‘Fog’, ‘Fog’, ‘None’, ‘None’, ‘Fog’, ‘None’, ‘Rain’, ‘Rain’, ‘Fog’, ‘Fog‘, ‘Fog’]

To ensure the percentage was not a fluke, I ran the code over the training data and received an accuracy of 70%. These percentages (41 and 70) are acceptable, but there is a way to improve the prediction accuracy of the testing data. By removing the visibility (because fog, rain, none and snow all have visibility 10), the mean temperature (because the information should be correlated to the max and min temperatures), and the minimum humidity (another inconsistent condition set) from the calculations, the testing data rose to 51% while the training data dropped to 61%. If the goal were to attain a higher accuracy for the testing data, I would remove these. However, given that the goal is to model a predictive weather system, an effective long-term program would employ all the data. I have left every attribute active within the code for this reason.

A few minor subtleties are involved in the calculation of these probabilities. First, the numbers we received from the training data are valued either as integers or as floating point variables – this inconsistency could lead to minor inaccuracies due to a loss of precision (a problem in every language). A solution casts every value that can possibly lose accuracy as a float. This limits the loss to several decimal places, where, when dealing with multiplication, should not be greatly affected. Also, since any inconsistencies will be made for each weather event, it will not greatly affect our results upon a final comparison.

A second consideration comes from careful analysis of the variance. If the data points contain no variance, then the value returned by Gaussian data curve will be undefined. To correct this, we add a stipulation that, should the variance equal zero, we return 1 as the probability. Therefore, when we multiply, the value does not change and the overall probability is not affected. A second issue could appear if, for a single day, the variances all equated zero. This would result in a probability of one. However, given the volatility of weather (i.e. what makes this task difficult to predict), we can assume variance will not equal zero in at least one weather condition.

Another problem develops from the selection of the months. If we were to try predicting a month where the weather conditions varied from the test data, it would have a low success rate. We are limited to predicting weather events around the months given as training data. To fix this, we would need a larger sample size to train the data efficiently, which would mean changing the training to discretization.

A major issue with the use of Naïve Bayes stems from its suitability. To use Naïve Bayes, one must assume each variable is independent. The data proves this is not the case. Meantemp will be casually related to Mintemp and Maxtemp, and humidMean will likewise have relation to humidMax, humidMin and potentially dewPoint. Weather is defined as the final result between the interaction of intrinsically related data. To assume these values are independent fails to represent weather. A reasonably successful prediction from our testing data will never match the prediction accuracy of a weather forecasting system, simply because a weather forecasting system will not make assumptions about the weather's independence. Naïve Bayes cannot be a suitable choice for determining weather. Thus, the 41% accuracy achieved seems an appropriate answer, given the large assumption required to use Naïve Bayes.

As an alternative training solution, I would suggest something along the lines of random forests. http://statistics.wharton.upenn.edu/documents/research/Deloncleetal.pdf is a University of Pennsylvania paper that suggests random forests are capable of predicting random surges of unexpected weather patterns, and can handle large amounts of data with relative ease – perfect for predicting the weather. To quote "…there are no classifiers to date that will consistently classify and forecast more accurately." (Deloncle, 1633) Random forests can supplement the Bayesian learning without cost to accuracy.

The inclusion of data from previous days presents the minor issue of Markov chaining. We would have to ensure our model did not make the prediction based on the previous day's weather condition. This would erroneously assume a day of rain would be likely to be followed by another day of rain. Any resident of Scotland would know that weather is fickle and rarely follows a pattern of this type. To prevent this, one would have to take the previous few days conditions into storage for computation later, potentially wasting both time and resources. The computation would need to be compared to the graph of a single data point, presenting the issue of its calculation. I know not how to solve this issue, thus it remains as extra credit in the code.

In conclusion, the weather prediction software designed with a Naïve Bayes model had a success rate of 41% when run over the test data given a Gaussian distribution, and 70% when run over the training data. This improved to a rate of 51% for the testing data after the removal of a couple inputs, but at a loss of overall accuracy – therefore, the prediction software remains at 41%. However, Naïve Bayes model is an unsuitable method for weather prediction, because of the necessary assumption of independence when weather is dependent. A better method would use random forests for learning. Floating point variables are dealt with by assuming the floats, and assuming the loss of a few decimal places to be negligible, given its universality throughout the code. Finally, previous days present a challenge in their storage.

Bibliography

Deloncle, A., R. Berk, F. D'Andrea, and M. Ghil. "Weather Regime Prediction Using Statistical Learning." *Journal of the Atmospheric Sciences* 64.5 (2007): 1619.

Equation diagrams taken from Wikipedia:

http://upload.wikimedia.org/math/6/c/8/6c84cac9b183bb78e05fb51205f7f7a0.png

http://upload.wikimedia.org/math/e/f/6/ef65a6d71f02b376dd42b54ce46abf6e.png

http://upload.wikimedia.org/math/1/8/8/188019d193258f9ba310da979906d24f.png