Rain Prediction using Bayesian Networks

Paul Clotan
Master's Degree in Artificial Intelligence, University of Bologna
paulioan.clotan@studio.unibo.it

May 2, 2024

Abstract

Predicting the weather is crucial, affecting everything from daily decisions like whether to carry an umbrella to preparing for severe events like hurricanes. As a result, having robust models that predict the weather is important for both our safety and comfort.
However, to achieve great performance, AI Forecasting models need vast amounts of data. It would be great if the data needed to train these models could be reduced.
This project explores this idea by using Bayesian Networks to predict the weather when the training data available is reduced.
The results indicate that incorporating domain knowledge can yield performances comparable to those of models trained with significantly more data.

Introduction

Domain

This project models one of the first challenges that arise in weather forecasting, predicting if it will rain tomorrow.
The choice to model the probability of rain in the following day is inspired by the need to have a complex task that needs to be modeled, but at the same time, the domain knowledge required to perform the experiments should be well documented, or, even better, intuitive.
The concepts used to design the Bayesian Networks presented in this work are mostly from the sphere of common knowledge and intuition. For instance, The maximum temperature is affected if there was a rainfall in that day or not.

Aim

The aim of this project is to explore the impact of incorporating domain knowledge during the design phase of the model on reducing the amount of data required to train the model. This is meant to simulate scenarios where there is not enough training data, but domain knowledge is available.
Specifically, we are interested in investigating Bayesian networks, as they offer a straightforward methodology for embedding domain knowledge directly into the model. This advantage can potentially offer a robust modeling environment where the experimentation phase of this project will be faster and more straightforward compared to using other techniques or models.

Method

The following steps were taken when elaborating the project:

- We use the Pandas library as used to assess the situation of the data. The dataset was checked for null values and preprocessed accordingly.
- We leveraged the matplotlib.pyplot library to check the distribution of the data. A great imbalance between the predicting classes was observed(1:4). To address this issue, the sklearn library was used to undersample the majority class.
- We used the sklearn and GridSearchCV create and tune a Decision Tree classifier on the data in order to establish a baseline for our measurements.
- We employed the pandas and matplotlib libraries to turn the continuous features of the dataset into discrete categorical features.
- We utilied the pandas and matplotlib libraries to turn the continuous features of the dataset into discrete categorical features.
- We created and analyzed three different Bayesian Network architectures using the pgmpy library. The performance of the models was measured using different performance metrics while varying the quantity of domain knowledge and training data used in developing and training the a certain model.
- We draw conclusions about the experiments, and used the pgmpy library to analyze the third Bayesian Network.

Results

After the completion of the experiments, the results are optimistic. It was observed that that enhancing the domain knowledge while reducing the training data by over thirty-five percent results in only a minor performance decline of 2-3 percent. Additionally, an increase in domain knowledge coupled with a reduction in training data leads to a noticeable decrease in training time.

Model

In this section, we will discuss both the considerations that were used when creating the Bayesian Network architectures.

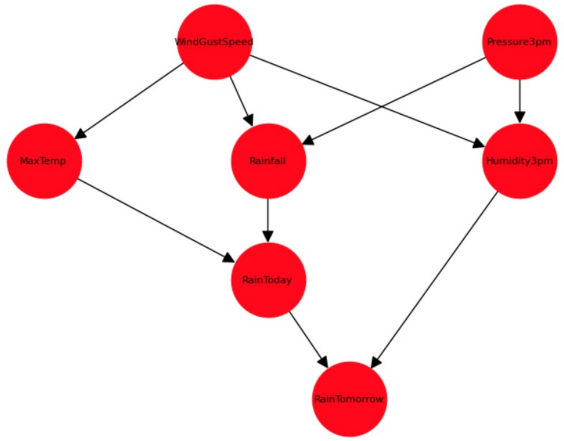

Figure 1: Final Bayesian Network

Three architectures were created using distinct initial assumptions. The first model treated all features as independent. The second and third models were crafted for ease of human understanding. Specifically, the second model, like the first, utilized the same data quantity but structured the system more intuitively. For the third model(Figure 1), training data was decreased and the domain knowledge used to design the network was increased. To reduce the data used, the number of features available to the model was minimized. A cutoff criterion was established to further reduce the features during training, based on each feature's correlation with the predictive label. Furthermore, we discarded features that would repeat the insights of other features present in the dataset.

We decided that the domain experience would be inserted into the model only when determining how to connect the features to one another. As a result, the CPT tables have been set to avoid specific beliefs about the distributions of the variables, favoring an initial state where all the features are considered to have equal importance. This was achieved using the Bayesian Estimator with a Bdeu prior.

It is also important to mention that the continuous features from the dataset were discretized by using either the distribution of the data into the dataset or thresholds inspired by literature. For the variables where inspiration was taken from the literature, a source was provided in the notebook associated with this report.

Analysis

Experimental setup

To assess the results, the need to define a baseline was identified. To obtain a baseline, a Decision Tree classifier was trained and tuned on the training data. After training the Decision Tree, the data was discretized and the three architectures described in the Model category were trained and tested.

The expectation was that the third Bayesian Network, trained with less data but designed with more domain experience would perform similarly with the second model.

Results

The results obtained after our experiments are showcased in the table below.

Table 1: Model Comparison

Model	Accuracy (%)	Training Time
Decision Tree	76.5	Not computed
First Network	72.5	1 minute
Second Network	75.5	0.16 seconds
Third Network	74	0.08 seconds

Our findings highlight the effectiveness of Bayesian networks in classification tasks. By evaluating the results of the last two networks (75.5 and 74 percent accuracy) against the fine-tunned Decision Tree Classifier (76.5 percent accuracy), we find the performance to be comparably effective across models. The results are even more impressing as the Decision Tree Classifier utilized 16 features, while the Second Network used 11 features, and the Third Network used only 7 features(a 35 percent drop in training data used to train the Third Network, compared to the Second Network).

Conclusion

To conclude, we wanted to test the following theory: The data required(nr of features) to have good results in Bayesian Networks drastically decreases as we increase the domain knowledge that we use when we design our network.

The results support our hypothesis. Comparing the performance of the Second Network with the Third Network shows that reducing the number of features from 11 to 7 (a 36 percent decrease in data) only resulted in a minimal performance drop of 1.5 percent. Additionally, the reduction in training time to half is significant, suggesting that this trend could be beneficial in larger datasets.

Links to external resources

Link to the dataset that was used is available here.

References

Source of inspiration for wind discretization categories.
Inspiration source for the relationship between windspeed and humidity.
Inspiration source for the relationship between humidity, temperature and pressure.
Notebooks of inspiration for use of the pgmpy library to create and analyze Bayesian Networks.