Predicting Rain

CS 484 Data Mining

Keeshawn Sun & Atif Hassan

May 13, 2019

# Abstract

Rain has the ability to affect people, businesses, and even governments. For this reason, the purpose of this project is to create a classification model that can accurately predict whether or not it would rain the next day. This was done by implementing a combination of Decision Tree and Clustering based models on various versions of a dataset provided to our team by the Australian government’s Bureau of Meteorology’s online database for climate data. This dataset stored information on the atmospheric conditions surrounding 49 different Australian weather stations over various extended periods of time. Given a specified day and its atmospheric conditions, our team was able to create a model that accurately predicted the occurrence of rain the following day 80% of the time.

# Introduction

Weather patterns can have great effect on economies and society. If it was possible to accurately predict rain forecasts, people, businesses, and governments would be able to plan and make arrangements accordingly . If events have to be moved or flood preparations have to made, they could be done in advanced and without wasting resources. Given data related to a specified day’s atmospheric conditions, our goal is to use the knowledge we obtained throughout the semester to predict if it will rain the next day. We created multiple models using different preprocessing and modelling techniques to determine which one would be the most accurate predictor of rain.

# Solution

The modelling method that we decided to use was a decision tree. We experimented with different preprocessing and feature selection procedures and compared the accuracies and runtime to determine the the most valid model. We used DecisionTreeClassisfier from the scikit-learn machine learning package to run the preprocessed data and determine accuracy.

We encountered a number issues issues with that the dataset that we had to address before we could train the model. Firstly there were many records that had null features and we used different methods to deal with that and determine the best course of action. We ran tests to determine if removing these records would increase accuracy and of clustering could be used to predict the missing values. Next the data set roughly has a 80/20 split between the negative and positive records. We determined that we would test the Synthetic Minority Over-sampling Technique to see if could improve results. We also tested to if whether removing unneeded and promoting important features could increase accuracy.

# Experiments

## Data

The dataset was originated from the Australian government’s Bureau of Meteorology’s online database for climate data and was discorced through a listing on kaggle.com. The data is formatted in a 13MB comma-separated values file with over 142 thousand records and 24 features. The features include weather measurements and observations having to temperature, wind, humidity, cloud cover, and various others.

## Experimental Setup

The experiment used the decision tree method with 4 preprocessing procedures: minimal preprocessing, removing features with many unknown values, with only data entries that have known values for each feature, estimating unknown values with the help of clustering. All four models were also trained and tested using both scaled and unscaled data. We also ran tested without using the SMOTE to see what if we could account for the unbalanced dataset. We used accuracy and runtime to compare the models.

### Base

“base.py” is the file that tests the accuracy and runtime of a decision Tree with minimal preprocessing. The data was prepared by first removing the ‘Date’ feature. Then the features that with values of ‘Yes’ and ‘No’ were replaced with Booleans, with ‘Yes’ replaced with 1 and ‘No’ replaced with 0. Next the values that was described as ‘NA’ were replaced with -1. Lastly the categorical data was transformed into numerical data. This procedure is present is all models and serves as the starting point for their more rigorous procedures.

### Remove

“remove.py” is the file that tests the accuracy and runtime of a decision tree after removing features with many unknown values. In addition the the base procedures, the Evaporation’, ‘Sunshine’, ‘Cloud9am’, and ‘Cloud3pm’ features were removed.

### Known

“known.py” is the file that tests the accuracy and runtime of a decision tree with only data entries that have known values for each feature. In addition the the base procedures, all records with null values were removed. Then SelectKBest was used to determine the importance of each feature and experiments were done to determine the best number of features to include.

### Estimate

“estimate.py” is the file that tests the accuracy and runtime of a decision tree after estimating unknown values with the help of clustering. In addition the the base procedures, k-means clustering was used the group data entries. We created multiple models using different number of clusters and compared their accuracies and runtimes.

## Experimental Results

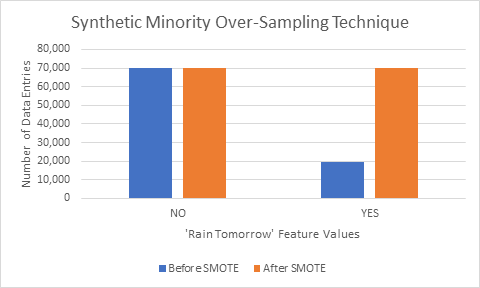


Figure 1. Number of ‘YES’ and ‘NO’ values under the ‘Rain Tomorrow’ feature before and after applying SMOTE. (base.py)

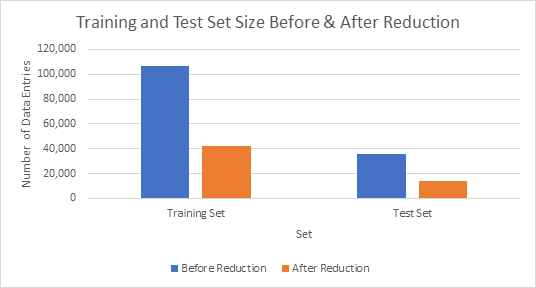


Figure 2. Training and Test set size reduction after removing data entries with unknown features (known.py)

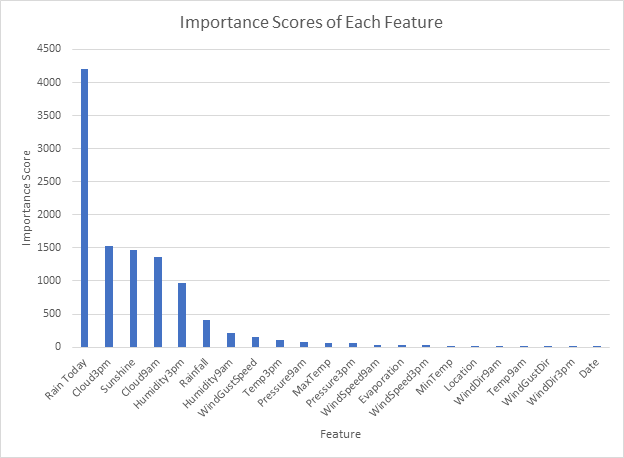


Figure 3. Importance Score of each feature obtained from SelectKBest algorithm (known.py)

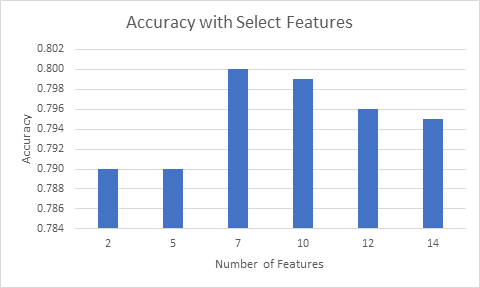


Figure 4. Accuracy values obtained after feature selection (known.py, we use it here because it will provide us with best feature selection since all features are known for all entries.)

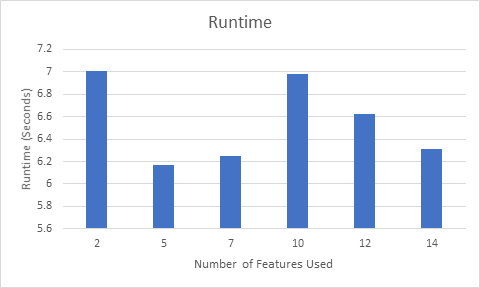


Figure 5. Runtime values when using different numbers of features (known.py)

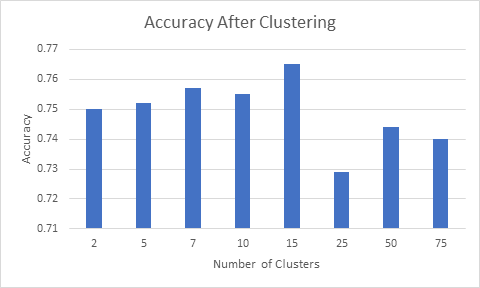


Figure 6. Accuracy after estimating unknown values by clustering (estimate.py)

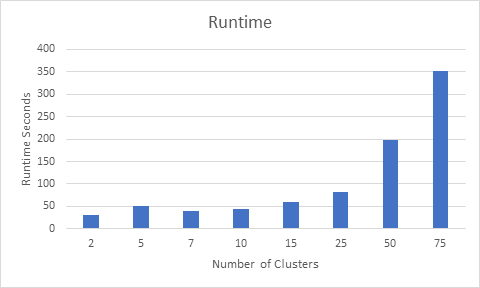


Figure 7. Runtime after estimating unknown values by clustering (estimate.py)

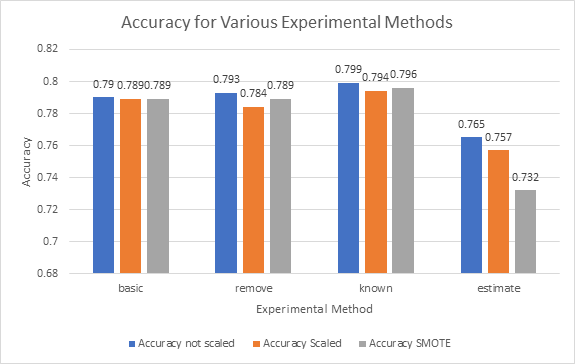


Figure 8. Accuracy for experiments

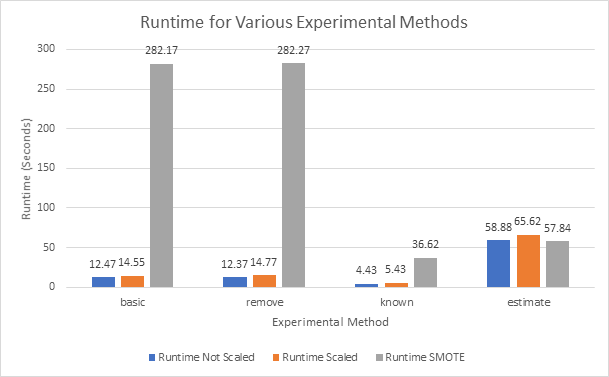


Figure 9. Runtime for experiments

# Analysis & Conclusion

Using SMOTE allowed us to compensate for our unbalanced dataset by over-sampling the the positive records (Fig. 1). However we determined that it did not yield significantly higher accuracies (Fig. 8) and increased the runtime drastically (Fig 9).

In the case of the known models, the elimination of the records with null values sharply reduced the number of records that were used for training and testing (Fig. 2) which yield better runtimes (Fig. 9). After using SelectKBest, we are able to determine the importance of each feature (Fig. 3). We compared the accuracies (Fig. 4) and the runtimes (Fig. 5) with the different numbers of features and determined that by using 7 features we would achieve the best accuracy without and unreasonable runtime.

For the estimate model, we results shows us that 15 clusters would achieve the best accuracy (Fig 6) without sacrificing speed (Fig. 7). However, in comparison to the other models, the estimate models fall short. In all three variations the accuracies are much lower (Fig 8). The difference between the accuracies of the best estimate model, estimate - not scaled, and the worst of the other three, remove - scaled model, is 2.45%. When it comes to runtime, despite being better the two worst runtimes of the other 9 models, they consistently clock in at about 1 minute.

When evaluating the accuracies of the base, remove, known models, only a small difference is observed (Fig 8). They all fall between .78 and .80 and the largest difference in accuracy is between the remove - scaled model (.784) and the known - not scaled model (.799) is only 1.90%. From our experiments, we can determine that the best model for predicting rainfall is the known model with unscaled data. It provided the best accuracy, .799, and at the same time the lowest runtime, 4.43 seconds.

# Contributions

Keeshawn Sun: Determined data preprocessing procedures. Built and tested models. Extracted valuable accuracies and run times. Created graphs.

Atif Hassan: Wrote the report