Abstract

The goal of this project is to create a weather application that uses the characteristics of today's weather in order to predict if it will rain the following day. This will be useful to city government's and event planners so that they will be able to plan better with the knowledge that either rain is coming or not. In order to do this, I will be connecting to Visual Crossing's API to pull historical weather data and then create a classification model based off of that data. After I have chosen my best model, I will place it into a Streamlit application for end users to interact with.

Design

This project originates from the need to know future weather patterns to accurately prepare for the following day's events. This application will help city government's and event planners to ensure they have enough tents on days that it is raining, or postpone events that would be heavily affected by the rain. This will save the government from wasting money on setting up events that will eventually get canceled because of rain.

Data

The dataset contains 12,000 days of weather information and 29 features for each. Most of the features are numerical, some of continuous, such as the dewpoint and the wind speed, while others are discrete, such as the phases of the moon. There are a few categorical variables that require data cleaning in order to be used in the model, such as conditions and icon.

Algorithms

Feature Engineering

1. Subtracting the feelslike temperature from the actual temperature
2. Converting categorical features to binary dummy variables
3. Create boolean column to represent if it has rained for more than 75% of the previous day
4. Create boolean column to represent if the feelslike is lower than the actual temp

5. Creating new continuous variables that measure the range of the temperature on a particular day
6. Converting sunrise and sunset times to minutes

Models

Logistic regression, k-nearest neighbors, and random forest classifiers, Naive-Bayes, and Decision Trees were used. I eventually settled on creating a VotingClassifier with a random forest, naive bayes, and logistic regression model with a hard voting system.

Model Evaluation and Selection

I split the training set into a training, validation, and test set. Since there was a time component to the data, I broke the data into these groups by date, so the earliest 70% of dates were in the training set, the next 15% were in the validation set, and the remaining 15% were in the test set. Although there was not a large class imbalance between the two target variables, I decided to use F1 as the metric for this model because it seems just as important to correctly identify a rainy day as it is to capture all the rainy days. I tried polynomial features as well, but the increase in F1 score was only .002 for some models and this marginal improvement did not justify the additional computational complexity. I fit the models on the training data and then scored them on the validation set. I then used my best performing model on my test dataset and received an F1 score of .56.

Tools

- Numpy and Pandas for data manipulation
- Scikit-learn for modeling
- Matplotlib and Seaborn for plotting

- Streamlit for web application
- Visual Crossing for Data Acquisition

Presentation

Communication

In addition to the slides and visuals presented, the streamlit app is https://share.streamlit.io/liammoran13/weather-classification/main.