Predict Rain in Australia.

by Y. Kostrov

Contents

Overview

The purpose of this project is to use weather data set from Kaggle to predict rainfall for the next day, based on the data about today's weather.

Business Problem

- Predicting rainy weather for the next day is a very important task. It plays a role in farming and other kinds of business, including restaurants, museums, etc.. Good weather forecast plays important role for tourist too.
- Usually weather is predicted by using complicated deterministic models involving partial differential equations.
- I would like to see how well the rain can be predicted by using Machine Learning.

Data Used in the Project

This data set contains about 10 years of daily weather observations from many locations across Australia.

RainTomorrow is the target variable to predict. It means – did it rain the next day, Yes or No? This column is Yes if the rain for that day was 1mm or more.

Cleaning/Modifying Data

- I extracted the month out of the Date column and saved it into the Month column.
- There were quite a lot of missing data in the numerical columns.
- I have filled the missing data with average values for the same region and the same month.
- I scaled the data.

Modeling: Creating Baseline Models

I have built the following classifiers to compare the results based on "recall" score as a primary metric and "precision" score as a secondary metric:

- Logistic Regression Classifier
- Random Forest Classifier
- KNeighbors Classifier
- Support Vector Machines Classifier
- XG Boost Classifier
- Naive Bayes Classifier

Explanation of Recall

- I used **recall** as a metric to select the base model and optimize by later on.
- Recall is defined as:

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} = \frac{\text{True Positive}}{\text{Total Actual Positive}}$$

Explanation of Recall

- Recall calculates how many of the Actual Positives our model captures by marking it as Positive (True Positive).
- Thus Recall is a better model metric when there is a high cost associated with False Negative.
- In our case False Negative is predicting "No Rain" when there is a "Rain Tomorrow".

Explanation of Recall

- For instance, in rain prediction.
- If it rains tomorrow (Actual Positive) is predicted as no rain tomorrow (Predicted Negative), then the person who relies on the prediction will be really upset since being unprepared for bad weather.

Explanation of Precision

- There is a secondary metric I will be watching, called "precision".
- Precision is defined as:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} = \frac{\text{True Positive}}{\text{Total Predicted Positive}}$$

Explanation of Precision

- Precision describes how precise/accurate your model is out of those predicted positive, how many of them are actual positive.
- Precision is a good measure when we worry about the costs of False Positive.

Explanation of Precision

- In our rain prediction, a false positive means "No Rain" tomorrow (actual negative) has been identified as "Rain" tomorrow.
- It is not that bad, since a person will carry an umbrella or rain coat for nothing.

How Well Baseline Models Performed

- Out of the box four out of six "vanilla" classifiers "Logistic Regression", "Random Forest", "Support Vector Machines", and "XG Boost" performed well.
- They achieve from 93% to 95.8% scores on "recall" metric.

I tuned up the four before mentioned classifiers trying to improve their recall score.

- Tuning up of the following four models
  1. Logistic Regression
  2. Random Forest
  3. Support Vector Machines
  4. XG Boost

  gives us the following results:

Logistic Regression Classifier

After tuning hyper parameters for Logistic Regression classifier we have the following results:

- Logistic Regression classifier went up from 93.6% to 100% on validation data.
- Logistic Regression classifier achieves 100% on test data that I have not used for training.
- The precision score went down from 85.5% to 75.7% on validation data.
- The precision score on test data is 75.9%.

Random Forest Classifier

After tuning hyper parameters for Random Forest classifier we have the following results:

- Random Forest classifier went up from 94.87% to 94.97% on validation data.
- Random Forest classifier achieves 95% on test data that I have not used for training.
- The precision score stays at 86% on validation data.
- The precision score on test data is 86%.

HGBoost Classifier

After tuning hyper parameters for HGBoost classifier we have the following results:

- HGBoost classifier went up from 94% to 97.7% on validation data.
- HGBoost classifier achieves 97.7% on test data that I have not used for training.
- The precision score went from 87% to 80.6% on validation data.
- The precision score on test data is 80.6%.

Support Vector Machines Classifier

After tuning hyper parameters for Support Vector Machines classifier we have the following results:

- Support Vector Machines classifier went up from 95.9% to 100% on validation data.
- Support Vector Machines classifier achieves 100% on test data that I have not used for training.
- The precision score went from 85.1% to 75.7% on validation data.
- The precision score on test data is 75.9%.

Conclusions

- It seems that the best choice for the model is HGBoost since it has the best balance between recall score at 97.7% on the test data and precision score at 80.6% on the test data.
- If one wants to neglect the precision score (labeling a lot of non-rainy days as rainy), then the best choice is Logistic Regression. Even though it is close in performance to Support Vector Machines, it is lighter and easier to retrain.

Ways to Improve the Project

- I would like to optimize the code.
- Learn more about weather and related data.
- Use more powerful computer for Support Vector Machines training.

THE END
THANK YOU!