Predicting Melbourne Weather

Ayda Arya, Maxwell Bennet and Minh Ngo

Credit: GordonBellPhotography

Executive Summary

Our aim for this project was to predict the next day's rain based on the past few days' (or longer) climate data. This data includes well-known weather measurements such as the temperature, rainfall, and wind. It also contains other metrics such as the mean sea level pressure, which is often used on television weather segments, as well as the dew point temperature and apparent temperature which are derived from the other measurements. The data provided to us had measurements taken every ten minutes, which we used to transform the data into daily aggregated measurements.

The measurements were taken from Melbourne's Olympic Park weather station, which were then logged by Simon Clarke. This data was provided to us as a text file, from which our code extracted data which we could work with for the rest of the project.

In the early data exploration phase, we found that none of the variables had a strong linear correlation, with the exception of variables that are derived from others. As such, we expect poor performance from the linear regression model.

Introduction

Project Outline

For this project, we worked to predict the rainfall of the next day based on historical data. This problem may be interpreted as either a binary classification problem or a continuous regression problem. Our group explored both.

Data Summary

Our data was provided to us as a space separated text file. The data was collected by the Bureau of Meteorology's Olympic Park weather station in Melbourne, then logged and provided to us by Simon Clarke.

The following are the variables in the dataset we were given, in the same order as the file:

- Time variables: `Year, month, date, hour` and `minute.`
- `Dry bulb temperature`: the standard temperature measurement in degrees Celsius.
- `Dew point temperature`: the measurement of the moisture content of the air in degrees Celsius.
- `Apparent temperature`: the measure of how hot it feels in degrees Celsius based on various other variables.
- `Relative humidity`: the percentage of moisture in the air in percentage, relative to the saturation level.
- `Wind direction`: in cardinal directions.
- `Average wind-speed`: in kilometres per hour.
- `Wind-gust`: maximum wind-speed in kilometres per hour.
- `Mean sea level pressure (MSLP)`: the pressure interpolated to mean sea level in Hectopascal Pressure Unit.
- `Rainfall`: cumulative measurement of rainfall since 9 am in millimetres.

We also computed the cartesian wind speeds (`x_speed`, `y_speed`) from the wind direction and wind speed variables. These are much more useful for modelling. Additionally, we added a rainfall difference value (`rainfall_diff`) which is calculated as the change in rainfall over a period of time in millimetres.

The data provided was mostly of a uniform file format. In some places, there were variances which impeded our importing.

For some parts of the data, the file became tab-delimited instead of variable space delimited. In this case, we replaced each tab with a single space.

Since the data was delimited by potentially multiple spaces and tabs, pandas import would not work properly on the raw file. We worked around this by importing the file as raw text first, and then processing it in text form to a point where pandas would recognise it in the way we expect.

The data contained many duplicate rows. These were removed once in DataFrame format.

There are some cases where the data contained zeroes in the `wind direction`, `average wind speed`, `wind-gust`, `MSLP`, and `rainfall` columns. These rows also contained the `MSLP` in the `wind direction` column. We set the columns with no data to numpy NaN values, while moving the `MSLP` to its correct position.

Some rows contained values of "-9999", "-9999.0", and "-". We interpreted these values as NaN values in all instances, and replaced them as such.

Expected Relationships

We expect that previous rain is a good predictor of future rain. If it rained yesterday, then it is more likely that it will rain today. We also expect the converse to be true.

We expect that the MSLP will be a useful predictor of future rain. MSLP is used generally as a climate predictor, so that should continue here.

High humidity should predict some rain. Likewise, if the humidity is low we do not expect it to rain.

Modelling Challenges

We needed to provide our models with the three days prior as input data, and the current day's rainfall as the target value. The target value will be a boolean value, or a vector of outputs corresponding to bins.

Our data contained far more days with no rainfall compared to days with non-zero rainfall. This caused our models to favour predicting no rain, just because there were more samples of that category.

We considered whether or not we should include the rainfall from the previous day as input for our models, as this could be avoiding the true purpose of the project of finding relationships between climate predictors and rainfall. We decided to include rainfall, as this information would be available on a given day if we are trying to do a prediction.

Solution Methods

We approached this problem using three different modelling techniques. Those were linear regression, logistic regression, and decision trees. We included linear regression as a starting point, despite expecting underwhelming results. We investigated both the binary classification problem and the multiclass classification problem, thresholding and binning the data as appropriate. We used SMOTE to balance the number of samples of each classification so that our models had a reduced bias.

Data Quality

Provide basic descriptive statistics or figures to point out any potential quality issues with the data sets you have used.

	air_temp	apparent_temp	dew_pt_temp	humidity	wind_speed	mslp	X_speed	Y_speed	rainfall_diff	rainfall_9am	wind_gust
count	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000	4085.000000
mean	15.913948	12.013184	9.026013	66.491910	19.428895	1017.159052	-2.627048	4.210774	0.014197	1.669400	42.959119
std	4.662940	6.634680	3.684530	11.479122	7.388564	7.249135	5.397078	15.563333	0.044553	4.664734	14.764393
min	5.796639	0.000000	-1.689231	24.300000	0.000000	990.162774	-26.804715	-35.770216	0.000000	0.000000	0.000000
25%	12.218321	7.996923	6.281818	59.290323	13.766129	1012.192810	-5.340590	-8.218327	0.000000	0.000000	32.000000
50%	15.355385	11.712030	8.552672	66.600000	18.425373	1017.165000	-1.620839	2.352477	0.000000	0.000000	39.000000
75%	19.018321	16.491406	11.429365	74.223022	24.074074	1022.225954	0.602757	16.902931	0.008264	1.000000	52.000000
max	34.187097	32.654032	22.671698	99.978261	53.807143	1037.080000	19.509774	53.353347	0.904839	54.600000	109.000000

Figure 1. Descriptive statistics of daily resampled data.

Exactly three rows contained missing values in the rainfall column. This is because there were jumps in the data such that the entire rainfall day was skipped, but not the full 24-hour day. These rows were interpolated so as not to create disruptions in future modelling. The impact of adjusting three rows in a set of 4085 rows is minimal on model performance.

To evaluate the quality of the data, we analysed their correlations to see how interdependent the variables are. If the value of correlation is close to 1 and -1, it means that the two quantities are closely related to each other and have a big influence on Melbourne weather. Below is a heatmap of the correlations between variables.

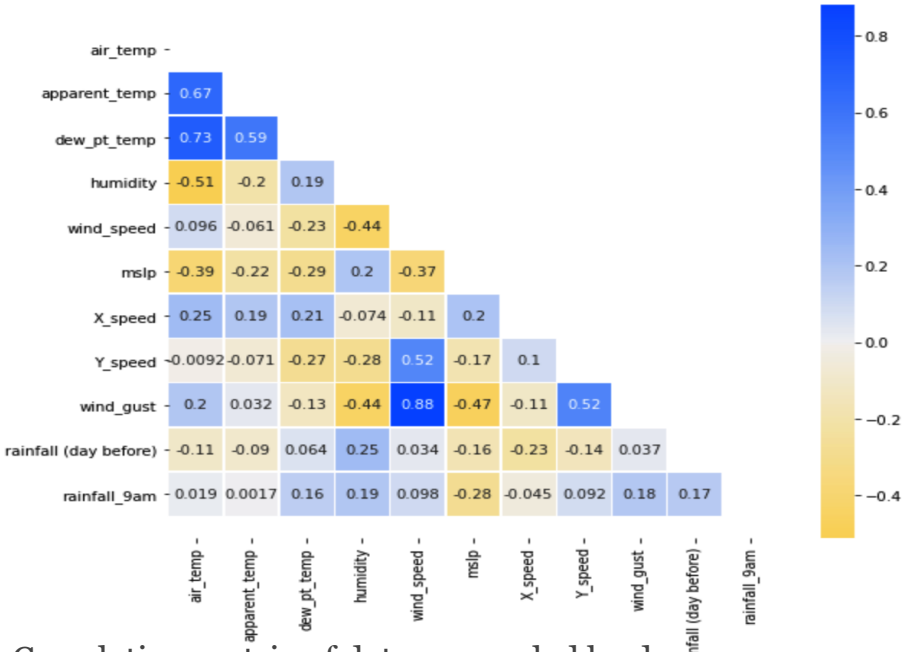

Figure 2. Correlation matrix of data resampled by day

As we can see from the heatmap, the highest correlation is between wind_gust and wind_speed (0.88), so it shows a strong, positive, linear relationship between these two variables. Therefore, the two quantities are strongly associative and have a great influence on the Melbourne weather. In contrast, the lowest correlation is between apparent_temp and rainfall_9am (0.0017 close to 0), thus, almost these two variables are not related to each other and do not have much influence on Melbourne weather.

In addition, some useful plots can also help us to draw conclusions about the weather of Melbourne.

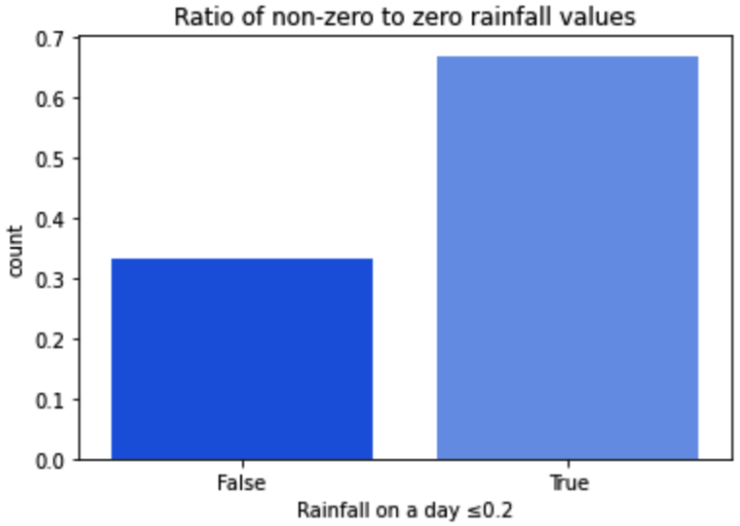

Figure 3. Proportion of days that have rainfall below the threshold given by the BOM of 0.2mm.

From the graph above, we can see that most of the days about 67% are not rainy, we only have 33% of the days when it rains.

Model Development

Setup and Approach

For the setup stage of the modelling, we wanted to have a Pandas DataFrame with values from three days prior. In order to achieve this, we first had to get resampled daily values. Pandas has a `.resample()` feature that automatically resamples between the 12am to 12am window. There were some steps we had to take before using this however.

As previously mentioned in the introduction, `rainfall` updates at 9am and is cumulative, meaning we have to be careful about how we resample this variable. Consider the diagram below:

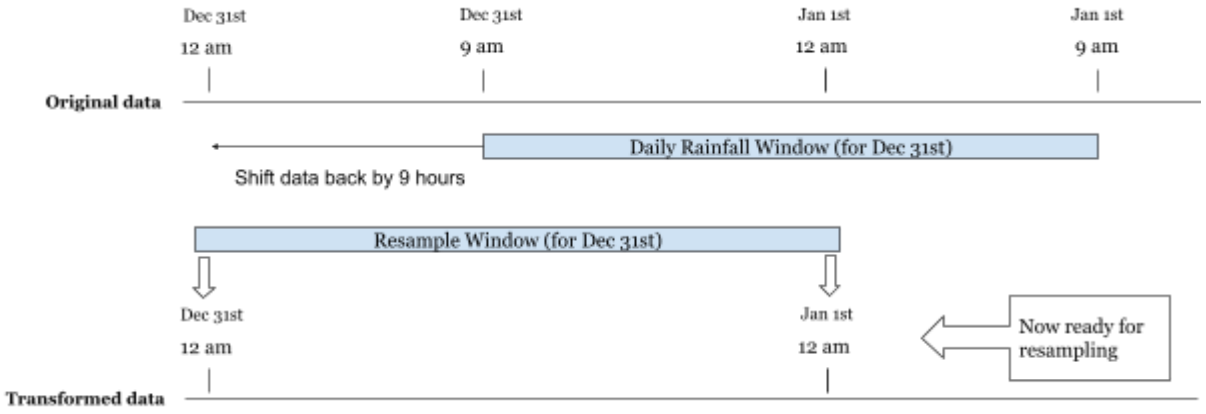

Figure 4. Diagram illustrating the transformation required to appropriately compute the rainfall since 9am for a given day.

As we can see, the solution we used to this problem was to shift all of the data back by nine hours in a new DataFrame, so that the cutoff for a day's rainfall would land on the 9am of the following day. Conveniently, there is a Pandas function that can be used for this specific purpose: `timedelta`. The code below demonstrates the final method used for resampling our `rainfall` and other values:

```
#storing rainfall values
rainfall_df = df['rainfall_9am']
#adding 9 hours to the index
rainfall_df.index = rainfall_df.index - timedelta(hours = 9)
#storing wind gust values
wind_gust_df = df['wind_gust']
#resampling rainfall/ wind gust by max and everything else by mean
```

```
df=df.drop(['rainfall_9am','wind_gust'],axis=1).resample('D').mean
()
df['rainfall_9am'] = rainfall_df.resample('D').max()
df['wind_gust'] = wind_gust_df.resample('D').max()
#remove any NaN values resulting from this method
df = df.interpolate(method="time")
```

Figure 5. Code block demonstrating daily resampling of all columns. Notice how `wind_gust` and `rainfall_9am` use the maximum value, whereas all other values are computed using the mean.

Note that the `wind gust` for a given day is the maximum measurement over that day. For `rainfall` it is the largest value over the shifted day. Since we are resampling to a larger time period, it is important to keep note of this and take the resampled value as the largest value over the sample. For all other values the resampled value is the mean.

As formerly mentioned, we decided to use linear regression, logistic regression and random forest classifier to predict `rainfall` values on a particular day given the data entries on the three days prior. We include linear regression as the most basic model available. We do not expect it to perform nearly as well as the nonlinear models, due to the nonlinearity inherent to the problem. Logistic regression is the basic nonlinear classification model. We include it as a baseline for a potential solution. The random forest classifier is a well-suited complex model. As such, it is expected to perform at least as well as the logistic regression.

As well as this, we split the problem up into binary and multi-categorical problems where the boundaries for the latter are defined as: [0, 0.2], [0.3, 5.0], [5.1, 10.0], [10.1, 25.0], [25.1, 54.6] in millimetres.

We also decided to implement SMOTE from imbalanced-learn in order to balance the rainfall values so the disproportionate distribution of data wouldn't compromise the accuracy of the model.

Limitations and Assumptions

Something worth mentioning in considering our models is that they rely on us knowing the values for all variables from the previous three days. This data will not be available if you're wanting to guess the amount of rainfall for say a week in advance.

Additionally, these models take into account all variables in the final dataset, and this is not always convenient as that information will not always be readily

available. Other models can be made using less variables, however the effect of constructing such models may need investigating.

Finally, the only metric predicted by this model is `rainfall`. If we were to investigate these models on different variables such as `temperature`, they may not be as accurate.

Results

Accuracy of Models

Much to our satisfaction, our best-performing model returned an accuracy score of around 92%, and this was our Random Forest multi-category model with

	problem type	binary	multi-category
names	smote/no smote		
linear regression	SMOTE	0.153753	0.253344
	no SMOTE	0.114708	0.088727
logistic regression	SMOTE	0.663312	0.354449
	no SMOTE	0.694002	0.664627
random forest	SMOTE	0.783166	0.923105
	no SMOTE	0.723378	0.686659

Figure 6. Performance summary.

SMOTE. Unsurprisingly, Linear Regression consistently returned the poorest results out of all of the models. The resulting accuracy scores for all the combinations of modelling strategies can be found in the pivot table below.

Interestingly, the Logistic Regression model performed much worse comparatively when multi-categorisation, although this may have to do with the fact that Logistic Regression models are supposed to execute with probability comparisons and one-hot encoding.

Confusion Matrix / Feature Importance

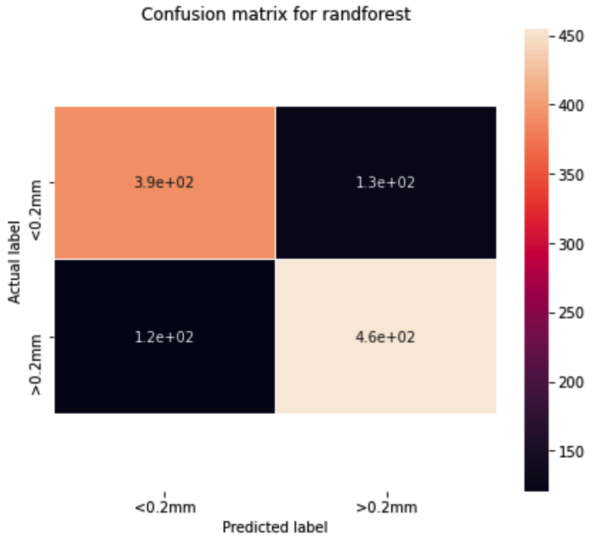

Figure 7. Confusion matrix for SMOTE random forest classifier.

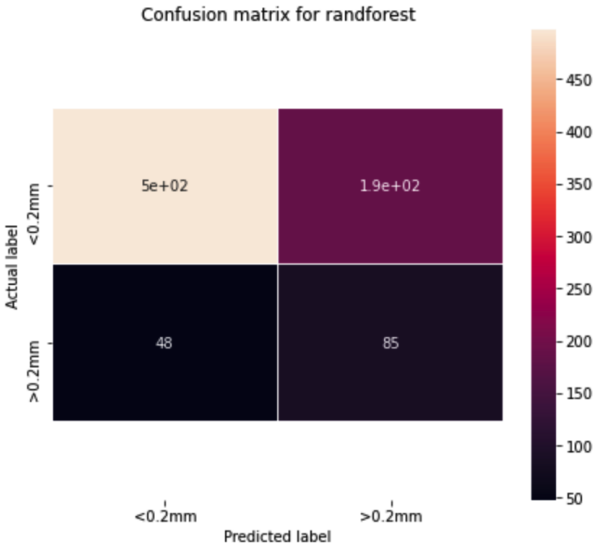

Figure 8. Confusion matrix for non-SMOTE random forest classifier. Note the extreme false negative rate, the model almost always predicts that it will not rain.

You'll notice from Figure 7 that the correctly identified and misrepresented predictions are roughly proportionate, whereas this is not at all the case without oversampling in Figure 8. This is a result of the oversampling with SMOTE, and is desirable for a well-functioning model.

Below are feature importances of the binary and multi-category Random Forest models respectively where the darker colours imply lower feature importance:

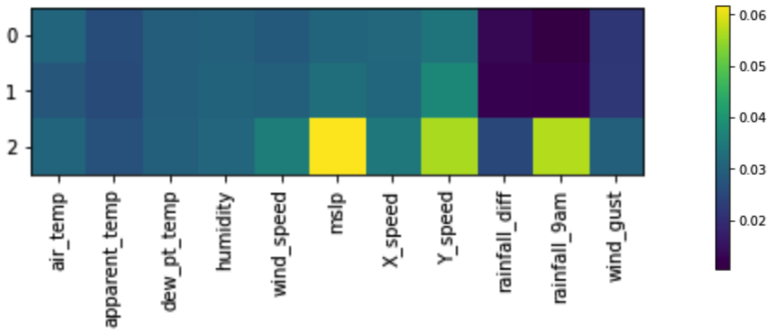

Figure 9. Feature importance table for random forest binary classifier. The bottom row represents data from the day before, and each row above that represents one day further back.

Looking at the feature importance map of our random forest SMOTE binary model, we can note that `MSLP`, `Y-Speed` and `rainfall` from the previous day are the most important features. In general, the data from the previous day is far more important than that of two days or more prior.

Conclusions

The aforementioned Random Forest suffices as a fairly accurate model for predicting daily rainfall. Oversampling with SMOTE was the most effective way to improve model performance for Random Forest in particular, and Random Forest in addition to SMOTE responded well to being split up into further bins.

While we are satisfied with this model's performance, our recommendation for future projects would be further investigation of the exclusion of different variables, as well as attempting to predict other variables. A good extension is also trying to use data from a week prior or incorporating seasonal analysis with time series methods such as ARIMA to improve the accuracy of such models.

Interpreting these results into real-world scenarios, we find that they line up well with what the layman might tell about the weather. If it rained yesterday, it'll probably rain today, but otherwise rain is uncommon. Adding the MSLP then increases the possible complexity to that of an amateur meteorologist, incorporating climate knowledge into predictions.