PREDICTIVE MODELING USING MACHINE LEARNING

DATASET

The dataset used here is well suited for the classification problems and conduct the proposed experiment. It contains information about 10 years of daily weather observations from various weather stations of Australia. The dataset has 142193 rows and 24 columns/attributes which could help in predicting the outcome. The target is to predict if it will Rain tomorrow – yes or no, perfectly suited for Binary classification.
The creator of the dataset has also specified note regarding one of the variable *Risk-MM* which needs to be dropped before developing any model. The information regarding other variables will be discussed in the later sections. Dataset can be downloaded from the below link:

https://www.kaggle.com/jsphyg/weather-dataset-rattle-package

RELATED WORK

The dataset is quite popular and has around 135 kernels which shows that multiple work has already been done on this data. The kernels were sorted and examined based on the hotness of the notebook. Most of the Kaggle users have implemented different machine learning and deep learning approaches to predict the outcome of the target variable.

One of the most famous kernels for the dataset included extensive analysis of all variables and implementation of Logistic Regression model which gave the accuracy of 85.01%. For fine tuning the model, the user has applied Recursive Feature Elimination with Cross Validation(RFECV) to filter out best features and develop a model. However, the accuracy remained around 84.99% lesser than the previous accuracy. Later, Grid Search Cross Validation was used for Hyperparameter optimization which improved the performance and yielded the final accuracy of 85.07%.

The other kernel uses SelectKBest method for selecting three main features based on the Chi2 scores and develops multiple models to predict the target variables. Implemented models - Logistic regression, Random forest, Decision Tree and Support Vector Machines yielded the accuracies of 83.63%, 83.66%, 83.69% and 79.05% respectively. The user has not used any technique to further improve the accuracies of the models.

One kernel implements three machine learning models - Random Forest, Support Vector Classification, and one deep learning model - ANN for the prediction. All three models gave accuracy around 84-85%. The user did not focus on the feature selection and hyperparameter optimization techniques to enhance the accuracy. Another Kernel explores the various feature selection techniques like Univariate Feature Selection based on SelectKBest, RFECV, Principal Component Analysis with Random Forest as the base classifier. Most of the developed models gave the accuracy of 100% raising questions on the way of implementation which is certainly inaccurate given its usage of RiskMM variable which measures the amount of next day rain in mm. The dataset creator has already put out a note saying it should not be used as it will leak the answers to the developed models.

The other highlighting kernel implemented XGBoost along with PCA for developing a model. It yielded the accuracy of 86.69%. Some kernels developed models using K Nearest Neighbour and Naive Bayes also but none of them gave better performance than the previously discussed models. In most cases, they seem to overfit the training data and give less testing accuracy. Furthermore, there were few kernels which focussed only on the visualisation part while some focussed only on the data exploration and pre-processing part. Also, some were related to working on specific cities and drawing inference regarding the probability of rain with the help of visualisation.

The discussion section did not involve much, except for the questions regarding the usability of datasets and some of the variables while making predictions. Though there was one user who pointed out the metrics which should be considered while evaluating the model developed on the basis of imbalance target variable. To conclude, it was observed in reviewing most of the kernels that maximum users who implemented Logistic Regression and Random Forest model with some variations either in feature selection or hyperparameter optimization achieved better accuracy than the other variants. Apart from this, the popular implementation of gradient boosting XGBoost was also among the top performing algorithms in predicting the outcome of the target variable.

RESEARCH GAPS AND PROPOSED WORK

The users have shown multiple implementations of various Machine Learning techniques in predicting the output of the target variable. However, in all the previous work done on this dataset, one thing which all the users have forgot to consider is the class imbalance problem in the target variable. The target variable RainTomorrow is not balanced, the minority class ‘Yes’ is significantly lower than the majority class ‘No’. Also, almost all Kagglers have chosen Accuracy as the metric to evaluate their models which should not be the case with the imbalanced class as the results can be misleading. Instead, the focus should be more on analysing the results through Precision, Recall or F1 score. In this situation, the predictive models developed using conventional machine learning algorithms could be biased and inaccurate. The combination of powerful algorithms with best hyperparameters and resampling of the target class could have helped in achieving a more robust and unbiased model.

Considering all the identified research gaps, the objective of this research is to implement a robust model which could be generalised well even on the unseen data. In the proposed work, class imbalance is handled using SMOTE technique which chelps in generating new and synthetic data to upsample the minority class which is probability of rain being 'Yes' in our case. The important features were selected based on Chi2 scores using SelectKBest(). Later, multiple models with the algorithms like Logistic Regression, Decision Tree, Bagging and Random Forest were implemented using the Randomised Search Cross Validation hyperparameter optimization technique. The accuracies of all these models were later tested on test data and compared with each other. Out of all the developed models, the best model based on the accuracy was then evaluated using Confusion Matrix on other metrics like Recall, Precision, F1-Score and ROC curve.

EXPLORATORY DATA ANALYSIS

DATASET DESCRIPTION

The dataset is huge and consist of 142193 instances and 24 attributes. From the summary of the dataset, it was observed that it is a mixture of Categorical and Numerical variables. Categorical variables have datatype object while Numerical variables have datatype float64.

There are total 7 categorical variables out of which one is a date variable and is denoted by 'Date'. The other 6 categorical variables are 'Location', 'WindGustDir', 'WindDir9am', 'WindDir3pm', 'RainToday' and 'RainTomorrow' in which RainToday and RainTomorrow are binary in nature while others are multi-class in nature. All the values will be converted in the numeric format which is discussed later in the data preparation section.

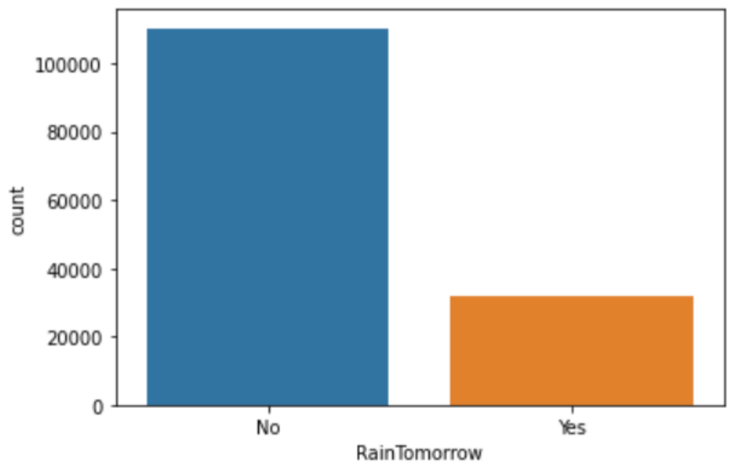
Total 17 Numerical variables are present in our dataset and all of them are continuous in nature. These are given by 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustSpeed', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm' and 'RISK_MM'.

The dataset creator has notified regarding the variable 'RISK_MM' and needs to be dropped before developing any model. Also, there are plenty of null values in the dataset which were noticed after sorting the counts of each variable. The columns like Evaporation, Sunshine, Cloud9am, Cloud3pm have very less data when compared to the other columns. We will deal with all these attributes in the data preparation stage.

The box plot drawn for some of the numeric variables showed the presence of outliers in the variables. These outliers need to be removed and standardized data must be used for our model building which we will do while preparing our data. The multivariate analysis was also done to find the patterns and relationships between the different attributes available in the dataset. The correlation heatmap revealed that the variables 'MinTemp', 'MaxTemp', 'Temp9am', 'Temp3pm', 'WindGustSpeed', 'WindSpeed3pm', ‘Pressure9am’, ‘Pressure3pm’ are highly correlated with each other. Most of the discussed plots/diagrams in this section were taken from the linked notebook [1].

TARGET VARIABLE INVESTIGATION

RainTomorrow is the target variable which is Binary in nature. It contains the outcome regarding the prediction of rain in the form of ‘No’ and ‘Yes’. The frequency distribution shows that there are total 110316 values for ‘No’ category and 31877 values for ‘Yes’ category. This shows that the classes in the target variable are highly imbalanced. When converted in percentage it shows that out of the total values, ‘No’ appears 77.58% times and ‘Yes’ appears only 22.42% times which is significantly low and the model developed without considering it, will be biased and not robust in nature. The plot for the same can be observed below:



There are variety of techniques of oversampling and undersampling which could be considered before creating a model to avoid the biasness which is discussed in the later sections.

IMPLEMENTATION

In order to implement the proposed work, the experiment was conducted in different stages to get the best out of all the models. The below sub-sections describe the process of implementing all the models in details.

Data Preparation

After the process of data exploration, the various things observed in the raw data were fixed which could have potentially affected the model implementation. The data pre-processing steps taken before feeding to all the models have been described below.

[1] https://www.kaggle.com/prashant111/extensive-analysis-eda-fe-modelling/notebook

Removal of unwanted columns

In the first stage of data pre-processing, some of the columns were removed which were not essential for our data modelling. As observed in the data exploration process, the columns like Evaporation, Sunshine, Cloud9am, Cloud3pm had less than 60% of the records when compared to the actual data. Also, the location column was removed as the research is not location specific. Date column would have not contributed in predicting the output, thus that column was also discarded. Another column RISK_MM was removed as per the guideline provided by the dataset creator. Since this variable specifies amount of rainfall for the next day, it can leak future information to our model and influence the predictions heavily made by the implemented model. Thus, all these columns were not considered for training our model.

Missing Values

It was observed during data exploration that there were lot of missing values. These missing values were dropped and not used during the training of our model. Earlier, the count of samples was 142193, after removal of null values there were 112925 samples left for our model building.

Removal of Outliers

Detecting outliers is an essential part of data pre-processing as non-removal of them can impact the model performance. It was done using Z-score normalisation technique and after removal of all the outliers there were 107868 samples left to develop our model.

Encoding of Categorical Variables

The next step involved one hot encoding of categorical variables as the model needs to be trained using the numerical data. Firstly, the target variable ‘RainTomorrow’ which predicts if it will rain tomorrow, was encoded to 0 and 1 where 0 corresponds to ‘No’ and 1 means ‘Yes’. Later, the other variables ‘WindGustDir’, ‘WindDir3pm’, ‘WindDir9am’ were converted into two-dimensional array as they had more than two categories.

Data Standardization

Our main objective is to classify Binary target variable thus it a good approach to standardize all the data in between 0 and 1. This was done using the MinMaxScaler function and whole data was normalized between 0 and 1.

SMOTE for Resampling

As discussed already, the data in the target variable is imbalanced and needs to be resampled. The new synthetic data could give more data to our classifier to get trained on and predict output without any biasness. For this purpose, the imblearn’s SMOTE or Synthetic Minority Oversampling Technique was used which uses a nearest neighbours algorithm to generate new and synthetic data in order to increase the count of the minority class. After SMOTE, both the classes in the target variable were balanced with each category giving the count of 85475.

Feature Selection

The feature selection technique helps in selecting the important features while training the data in order to increase the performance of the model. By selecting only, the relevant features, the redundant and misleading data will be removed from the dataset which will eventually help in reducing overfitting and improving accuracy. It will also reduce the algorithm complexity and algorithms will train faster. Here, for simplification reason Chi-squared(Chi2) statistical test was conducted using SelectKBest function to find the top five features which have the strongest relationship with the output variable. The variables RainToday,Rainfall,Humidity3pm,Humidity9am and WindDir9am_N were identified as the top five features which greatly influence the target variable.

After feature selection, the dataset was divided into training and testing set in the ratio of 70% and 30% respectively and various models were implemented.

Model Training

Logistic Regression

The first model which was trained using the final prepared data was Logistic Regression. The reason for choosing this algorithm was mainly to test how the binary classifiers will perform in case of increasing the sample size using the synthetic data. Also, it was found that there were multiple kernels which got good accuracy using this algorithm. In order to find the best performing Logistic Regression model, a grid for hyperparameters was declared. It consisted of below parameters :

```
Solvers    = ['newton-cg', 'lbfgs', 'liblinear']
penalty    = ['l2']
c_values = [100, 10, 1.0, 0.1, 0.01]
```

The solvers used in this case support mostly L2 regularisation and often helps to converge faster for high dimensional data. C-values specify the strength of regularisation, usually smaller values specify stronger regularisation and reduces overfitting. For this purpose, the bigger pool of c_values were used here to conduct the experiment. A five-fold cross validation with ten iterations was then used to perform the randomised search to find the best set of attributes and the model developed using those attributes was saved as the best model. The best set of parameters estimated by the randomised search were - 'solver': 'newton-cg', 'penalty': 'l2' and 'C': 0.1 . The implemented model was tested on the test data which gave the final accuracy of **74.62%** which is not good when compared to the kernels who carried out their experiment without taking class imbalance into consideration.

Decision Tree

The decision tree classifiers usually perform better when there is the case of Class imbalance and balancing the class using synthetic data points in the target variable. I order to test this, a grid of max_depth of the tree ranging from 3 to 15 was defined.

The other effective parameters like criterion, min_samples_split etc. were not specified and kept as default for the simplification and avoid complexity in training the algorithm.

After performing randomised search with 5-fold cross validation for ten iterations, the best Decision Tree model was developed using the best-found parameter 'max_depth' of 14 . The model gave the accuracy of **82.34%** on test set of the data, a lot better than the Logistic Regression. With the accuracy, it was clear that decision trees and its variants could be the better choice for such data.

Bagging Classifier

The accuracy obtained with the simple decision tree was the motivation behind implementing the next model using bagging classifier for the prediction of rain. Bagging is a combination of Bootstrapping and Aggregating to form one ensemble model. A default decision tree was formed using each of the bootstrapped subsamples pulled from the data. It is a powerful technique which helps in reducing the variance observed in the decision trees by using randomization in its construction procedure and make a final predictor based on the aggregated results of the sampled Decision Trees. Most of the parameters were kept as default by not specifying in the grid except for the n_estimators which represents the number of trees.

The number of trees were chosen from the grid of [10,100,1000] with five-fold cross validation to perform the randomised search for ten iterations. The number of trees with the count of 1000 were found the best fit for our data. As expected, the accuracy improved drastically from the other classifiers. It gave accuracy of **87.41%** on the test data and was the top performing model.

Random Forest

The other implementation of Ensemble is Random Forest which can be considered as the Bagging classifier with slight tweak in its way of implementation. Similar to Bagging, bootstrapped subsamples are pulled from the larger dataset. A decision tree is formed on each subsample, but each decision tree splits on different features which is not the case with Bagging where splitting occurs at similar features at each node throughout the process. This level of differentiation in Random Forest usually helps in producing a better ensemble model. Taking this into consideration, Random Forest was used to implement the final model in order to enhance accuracy which was achieved with the Bagging classifier.

The grid for the hyperparameter optimization was defined as below :

n_estimators = [10, 100, 1000]
max_features = ['sqrt', 'log2']

where n_estimators represent the range of number of trees to be considered in the forest and max_features represent the number of features to consider when looking for the best split. The number of trees were chosen as much as in the Bagging classifier to observe if there is any change in the accuracy. The best parameters were in our case and the model based on those parameters was saved as the best model. The final accuracy observed on the test data was **87.47%** which is not that much different than what was achieved using Bagging, yet with the difference of 0.01% the Random Forest comes out as the winner among all the tried models.

One thing to notice here is – randomised search cross validation technique was chosen to find the best hyperparameters, grid search with cross validation could have further enhance the accuracies of all the models. But grid search often gets complex if the larger grid is specified and comes at the cost of higher computational power and more training time. Randomised search on other hand helps in tuning models by choosing randomly and thus saves lot of training time and computational issues.

Considering all the limitations, still our model developed using Random Forest gave the highest accuracy and outperformed all the models in the kernels of the dataset. This model was further evaluated on different metrics which is discussed in the below section.

EVALUATION

Confusion Matrix

Confusion matrix compares the predicted target values generated by our model with the target values of our test data. Thus, the terms like True Positives, True Negatives, False Positives and False Negatives help in finding the difference between the actual and predicted values. True positives (TP) are the number of instances which were classified as positive by the classifier that are actually positive while True Negatives (TN) are the number of instances classified as negative by the classifier that are actually negative. On the other hand, False positives (FP) are the number of instances classified as positive by the classifier that are actually negative and False negatives (FN) are the number of instances classified as negative by the classifier that are actually positive. The model was evaluated using the confusion matrix and the results can be understood via the below table. *(The generated confusion matrix plot can be observed in the code)*

Actual Target Value	***Predicted Target Value***	***Value***	
Yes	Yes	21324	TP
Yes	No	4245	FN
No	No	23519	TN
No	Yes	2197	FP

Based on the values obtained using confusion matrix, the following metrics were evaluated:

Classification Accuracy

Classification accuracy is defined as the number of instances in the dataset which were classified correctly from the overall predictions made by the classifier. The calculation for classification accuracy is done using formula : (TP+TN)/(TP+TN+FP+FN). The random forest model gave the classification accuracy of 87.47% which signifies that our model was able to correctly classify 87.47% of the inputs.

Precision

Precision is often referred as the Positive Predictive Value it gives the fraction of positive instances classified correctly by the classifier. It is calculated using formula : (TP)/(TP+FP). The precision for our model evaluated using test dataset was 90.65% which shows that the model predicted outcome for rain predictability as ‘Yes’ for around 91% of the inputs when the actual outcome was also ‘Yes’.

Recall

Recall is also called the True Positive Rate or Sensitivity as it measures the proportion of actual positives that are correctly identified as such. It can be calculated using formula: (TP)/(TP+FN). In this case, the Recall for our model when evaluated using test dataset was 83.39% meaning the model identified the true values, when it actually did rain the next day around 84% of the time. The developed model performed lot better in terms of Recall when compared to the other users who even after getting a decent accuracy, got very less recall value.

F1 Score

F1 score is also called as F Score or F Measure and uses precision and recall to compute the score. It depicts the balance between precision and recall. In simple terms if precision increases, recall decreases, and recall increases if precision decreases. It can be calculated using the formula: 2*(Precision*Recall)/(Precision+Recall). Our model when evaluated on test data gave the F1 score as 86.87% which is good as it is not even overpredicting and under-predicting the outcome.

AUC-ROC

Receiver Operating Characteristics (ROC) curve is a graphical plot which shows the ability of the classifier by plotting the true positive rate against the false positive rate. The curve was plotted by keeping true positive rate (Sensitivity) on Y-axis and false positive rate (1-Specificity) on X-axis. AUC(Area Under Curve) represents the degree or measure of separability and shows how much our model was capable of distinguishing between classes. Higher the AUC, better the model is at predicting No as No and Yes as Yes. In our case, AUC score was 87.5%. ROC curve for our model can be observed below:

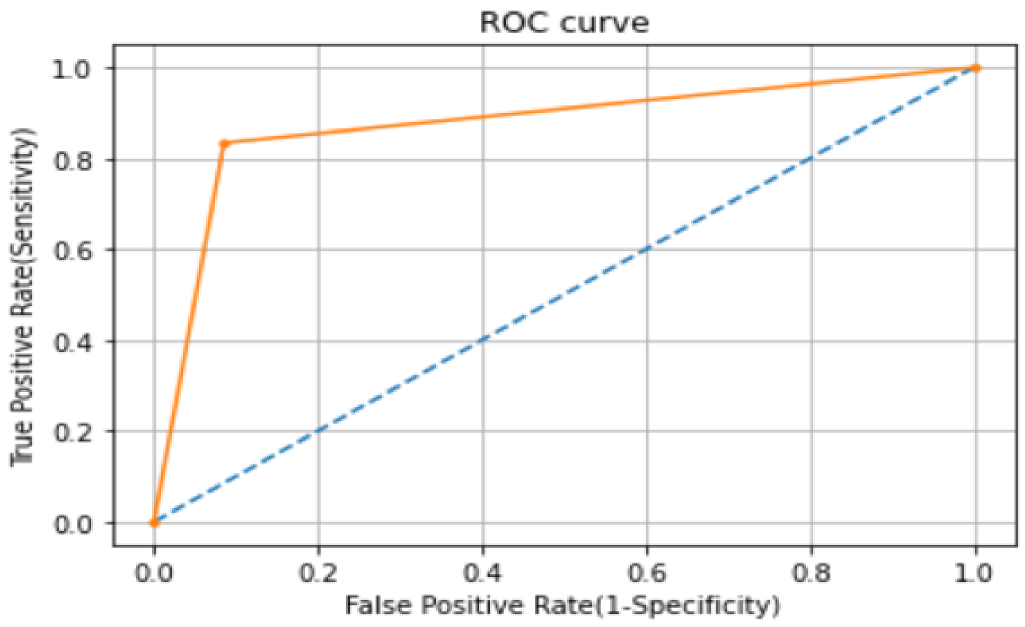


Figure 1:ROC curve for Random Forest