CMPT459 Project

Report

1. Problem statement

In this report, we build several machine learning algorithms to predict the outcome groups based on features in the dataset, e.g. age, gender, country, location, daily cases, etc. The outcome groups are categorized into three different classes, namely "decreased", "hospitalized", "nonhospitalized". Since it is a classification problem, we attempt many classic classification algorithms, including decision tree, KNN, and random forest. For each model, we tune hyperparameters by cross validation method to choose the best hyperparameter settings, and the evaluation metric we choose is F1-score. For data preprocessing, we map all categorical variables into numerical representation, and scale all features so that it has standard normal distribution. To handle data imbalance of labels, we resample the minority of the label, so that each category has the same quantity.

2. Data preparation

Task 1.1 Feature selection

In total, there are 14 features, and 1 label variable, but only some of features are effective for models. Thus, we exclude some of features, which either contains NaN values or the frequency of one value is extremely large. In our case, we eliminate two features: "province", which has NaN values, and "chronic_disease_binary", which has 17131 False values and 81 True values, which is not an effective feature. The remaining features are "age", "gender", "country", "latitude", "longitude", "data_confirmation", "Comfirmed", "Deaths", "Recovered", "Active", "Incident_Rate", "Case_Fatality_Ratio".

Task 1.2 Mapping the features

Then, we should convert all categorical features to numerical ones. The categorical features contain "sex", "country", "date_confirmation", and "outcome_group". For "sex", "country" and "outcome_group", we convert it into integer, while for "date_confirmation", we split the date into year, month and day, and we only keep the month and day, since all dataset share the same "year" value.

Task 1.3 Balancing the classes in the training set

We plot the distribution of class label in Figure 1. It shows the most of instances in "outcome_group" is "hospitalized", which accounts for 76.9%, while the other two categories: "nonhospitalized" and "decreased" consists of only 17.3% and 5.8%, respectively. The following shows the frequency for each class:

```
hospitalized       13241
nonhospitalized     2974
deceased             997
```

To handle that, we resample the dataset so that each category has the same number of

instances, i.e., all categories have exactly 13241 instances, so that the dataset has been balanced.

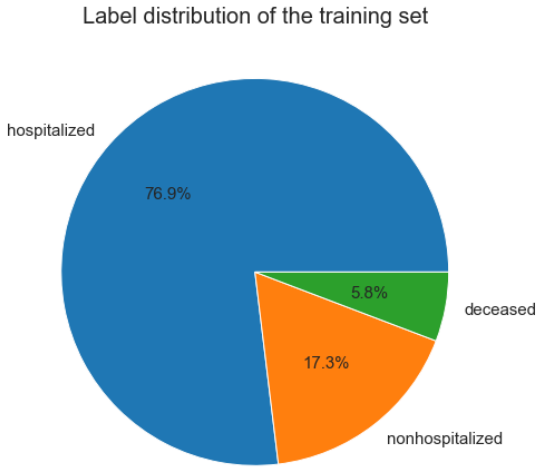

Figure 1. Label distribution of the training set

3. Classification Models

Task 1.4 Building models and hyperparameter tuning

We have built three models: decision tree, KNN and random forest. For each model, we apply 5-fold cross validation to tune hyperparameters, and the evaluation metric for model comparison is F1-score.

For decision tree classifier, we tune two hyperparameters: maximum depth and minimum samples for the leaf node. The max_depth ranges from 1, 2, 5, 10, to 13, and the candidate values for min_samples_leaf are 1, 5, 10, and 20. We use grid search so that every combination of two hyperparameters is considered. Decision tree is fast to train and can handle outliers compared with other models.

For KNN classifier, we tune only one hyperparameter: the number of neighbors. The possible values for n_neighbors are 1, 3, 5, 9, 11, 15, 25, 49, and 99. We use grid search so that every combination of hyperparameter is considered. KNN is better compared with other models because we do not need to train the model, and the model is easy to understand and interpret.

For random forest classifier, we tune two hyperparameters: maximum depth and minimum samples for the leaf node. The max_depth ranges from 1, 2, 5, 10, to 13, and the candidate values for min_samples_leaf are 1, 5, 10, and 20. We use grid search so that every combination of two hyperparameters is considered. Random forest is an ensemble learning method, which can aggregate base models, so that it can avoid overfitting problem.

Table 1 shows mean macro F1-score, mean macro F1-score for class “decreased” and mean overall accuracy across validation sets.

Based on the results, we can conclude that KNN can achieve the highest mean macro F1-score value compared to decision tree and random forest.

Model	Hyperparameters	Mean macro F1-score across the validation sets	Mean F1-score on "decreased" across the validation sets	Mean overall accuracy across the validation sets
Decision tree	max_depth = 1, min_samples_leaf = 1	0.5156	0.0	0.6362
	max_depth = 1, min_samples_leaf = 5	0.5156	0.0	0.6362
	max_depth = 1, min_samples_leaf = 10	0.5156	0.0	0.6362
	max_depth = 1, min_samples_leaf = 20	0.5156	0.0	0.6362
	max_depth = 2, min_samples_leaf = 1	0.7450	0.6508	0.7529
	max_depth = 2, min_samples_leaf = 5	0.7450	0.6508	0.7529
	max_depth = 2, min_samples_leaf = 10	0.7450	0.6508	0.7529
	max_depth = 2, min_samples_leaf = 20	0.7450	0.6508	0.7529
	max_depth = 5, min_samples_leaf = 1	0.7906	0.6830	0.7958
	max_depth = 5, min_samples_leaf = 5	0.7904	0.6829	0.7956
	max_depth = 5, min_samples_leaf = 10	0.7902	0.6827	0.7954
	max_depth = 5, min_samples_leaf = 20	0.7902	0.6826	0.7953
	max_depth = 10, min_samples_leaf = 1	0.8580	0.7954	0.8586
	max_depth = 10, min_samples_leaf = 5	0.8557	0.7943	0.8562
	max_depth = 10, min_samples_leaf = 10	0.8505	0.7874	0.8515
	max_depth = 10, min_samples_leaf = 20	0.8433	0.7762	0.8437
	max_depth = 13, min_samples_leaf = 1	0.9052	0.8628	0.9049
	max_depth = 13, min_samples_leaf = 5	0.8988	0.8562	0.8985
	max_depth = 13, min_samples_leaf = 10	0.8827	0.8349	0.8827
	max_depth = 13, min_samples_leaf = 20	0.8647	0.8074	0.8651
KNN	n_neighbors = 1	0.9797	0.9719	0.9797
	n_neighbors = 3	0.9782	0.9712	0.9780
	n_neighbors = 5	0.9645	0.9534	0.9642
	n_neighbors = 9	0.9344	0.9141	0.9336
	n_neighbors = 11	0.9212	0.8967	0.9201
	n_neighbors = 15	0.8947	0.8614	0.8932
	n_neighbors = 25	0.8558	0.7979	0.8550
	n_neighbors = 49	0.8265	0.7501	0.8270
	n_neighbors = 99	0.8078	0.7192	0.8098
Random forest	max_depth = 1, min_samples_leaf = 1	0.5213	0.0113	0.6419
	max_depth = 1, min_samples_leaf = 5	0.5255	0.0179	0.6418
	max_depth = 1, min_samples_leaf = 10	0.5231	0.0267	0.6407
	max_depth = 1, min_samples_leaf = 20	0.5192	0.0368	0.6413
	max_depth = 2, min_samples_leaf = 1	0.6553	0.3339	0.7102
	max_depth = 2, min_samples_leaf = 5	0.6519	0.3605	0.7147
	max_depth = 2, min_samples_leaf = 10	0.6821	0.3942	0.7181
	max_depth = 2, min_samples_leaf = 20	0.6986	0.3954	0.7146
	max_depth = 5, min_samples_leaf = 1	0.7959	0.7000	0.7999

	max_depth = 5, min_samples_leaf = 5	0.7943	0.7022	0.8002
	max_depth = 5, min_samples_leaf = 10	0.7937	0.7001	0.7995
	max_depth = 5, min_samples_leaf = 20	0.7940	0.6985	0.7986
	max_depth = 10, min_samples_leaf = 1	0.8648	0.7951	0.8652
	max_depth = 10, min_samples_leaf = 5	0.8570	0.7877	0.8578
	max_depth = 10, min_samples_leaf = 10	0.8504	0.7768	0.8504
	max_depth = 10, min_samples_leaf = 20	0.8476	0.7577	0.8389
	max_depth = 13, min_samples_leaf = 1	0.9234	0.8853	0.9242
	max_depth = 13, min_samples_leaf = 5	0.9060	0.8623	0.9075
	max_depth = 13, min_samples_leaf = 10	0.8876	0.8312	0.8852
	max_depth = 13, min_samples_leaf = 20	0.8604	0.7922	0.8617

Table 2. Mean macro F1-score, macro F1-score on “decreased” and mean overall accuracy across validation sets

Task 1.5 Overfitting

Since the training set is highly imbalanced, we use two methods to alleviate overfitting issues. The first one is the oversampling, which balances the number of instances for each category; The second one is the evaluation metric. Instead of accuracy, which is a biased metric towards the majority of the label, i.e. if the model predicts the majority of the class, the model can achieve a very high accuracy score. However, if we choose F1-score as our metric, we can handle the issue appropriately. In random forest, when max_depth = 1 and min_samples_leaf = 1, the F1-score is 0.5213, but the accuracy score is 0.6419, which means the model is prune to achieve high accuracy, if the F1-score is relatively low. However, when F1-score is over 0.8, both accuracy and F1-score can achieve the same level.

Task 1.6 Comparative study

We will choose the best model among hyperparameter space, which achieves the highest cross validation f1-score. For decision tree, the model achieves the highest F1-score when max_depth = 13 and min_sample_leaf = 1, and mean macro F1-score is 0.9052; For KNN, the model achieves the highest F1-score when n_neighbors = 1 and mean macro F1-score is 0.9719; For random forest, the model achieves the highest F1-score when max_depth = 13 and min_sample_leaf = 1 and mean macro F1-score is 0.9242. The result indicates KNN is more promising model, and random forest is better than decision tree, which means random forest has better generalization capacity compared with decision tree model.

4. Model predictions

Task 1.7

When n_neighbors is set to 1 in KNN, we predict the results based on the features of the test set. After prediction, we show the distribution of each category below:

```
hospitalized       3318
nonhospitalized     699
deceased            287
```

The distribution indicates the label distribution almost fits the distribution of the training set, so our best model is not prune to the majority of the classes.

5. Conclusion

In conclusion, we have investigated three machine learning algorithms to predict the outcome group of the COVID-19 dataset, including how to handle imbalance of the label variable, how to tune hyperparameters, and which evaluation metric we should choose to select the best model. Finally, the best model we choose is KNN, with n_neighbors = 1, and it can achieve 0.9719 mean macro F1-score on cross validation.

6. Lessons learnt and future work

Based on the model performance, we have learnt that some simple model like KNN can achieve higher performance than complex model, such as decision tree and ensemble learning model random forest. In further study, we will attempt other models, namely boosting algorithms (Adaboost, Gradient Boost, XGBoost, etc), neural networks, SVM, etc, and try to understand which features contribute to prediction to better interpret the model.

7. References

- All lectures of CMPT459, https://coursys.sfu.ca/2022fa-cmpt-459-d1/pages/Lecture_notes
- *Classification accuracy is enough: more performance measures you can use*, https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/
- Confusion Matrix, Accuracy, Precision, Recall, F1 Score, https://medium.com/analytics-vidhya/confusion-matrix-accuracy-precision-recall-f1-score-ade299cf63cd

8. Contribution

Zhuo Liu: task 1.1 – task 1.4 and the report (Data preparation, Conclusion, Lessons learnt and future work)

Teliang Yu: task 1.5 – task 1.7 and the report (Classification models, Model predictions)