COMP30027_Assignment3_May_2021

Multiclass classification to predict the cooking time for recipes

1. Introduction

Food.com is an open web platform where users can share recipes. This report aims to develop supervised multiclass classification models to predict the cooking duration from recipe features obtained from Food.com. Further, the performance of models on different feature selection and text encoding methods are evaluated with error analysis.

2. Related work

SVM approaches were designed for binary classification which can be extended for multiclass classification with a one vs one approach. This means one classifier is constructed for each pair of classes which gives O(n^2) time complexity. Thus, selecting feature will be important to reduce computational time and improve the performance of the classification substantially. Although previous work suggests the best performance of a wrapper method (Kohavi & John 1996), this is often time-consuming on the large-dimensional datasets. This project will focus on various feature engineering method for the multiclass classification problem.

3. Classification Method

For this project, a random holdout was utilised to partition the dataset. The classification models were trained on the training set generated by different feature selection methods and the one that gave the highest accuracy score on the validation set was chosen as the best model. After conducting parameter tuning on the best performing model, its performance was evaluated with a test set.

3.1 Datasets

The dataset contains 40000 instances with recipe features including name, ingredients, steps, number of steps, number of ingredients and class labels.

3.1.1 Class Distribution

The original dataset (train.csv) and the training set (including validations) selected by 8:2 random holdout was found to have a similar class distribution which can be considered as a stratified sample (Table 3.1.1.1).

Class label	1.0	2.0	3.0
Original	0.443	0.506	0.0512
Train	0.443	0.505	0.0516

Table 3.1.1.1 Class distribution for the original and training datasets.

Accuracy measures the percentage of the correct prediction made by a classifier. Since the dataset is imbalanced, and class 3.0 occupies a small proportion, using accuracy only is not sufficient for the best measurement. In addition, the macro average will be insensitive to class imbalance as it treats each class equally, and the weighted average will favour the majority class which is not preferable for imbalanced datasets. Since we focus on maximising the correct prediction for this project, we use micro-averaged F1-score, precision, recall and accuracy to evaluate models.

3.1.2 Text Features

Various text conversion methods including CountVectorizer, doc2vec and term-frequency times inverse document-frequency method (tf-idf) were utilised for text features.
The CountVectorizer converts a collection of text documents to a matrix of token counts.
The doc2vec method creates a numeric representation of a document.
The tf-idf conversion calculates
tf-idf(t,d) = tf(t,d)*idf(t) for a term t of a document d in a document set
and idf(t) = log[(1+n)/(1+df(t))]+1, where n is the total number of documents in the document set and df(t) is the document frequency of t.

3.2 Feature Engineering

All the features were combined, and feature

engineering methods were performed such as mutual information and chi-square feature filtering methods, principal component analysis (PCA) and random feature selection (first k features). First, each set of text features are reduced to 50. Then, they are combined and be reduced again to a desirable number of feature columns.

3.3 Classification Models

The model performances on the selected features were evaluated on various machine learning techniques including 0R, Gaussian Naïve Bayes, Decision Tree and SVM.

3.4 Parameter tuning

Best performing parameters were chosen for the best performing model using a grid-search method. The small, stratified subset of a training set was used for faster computation. The resulting parameters were utilised for the final prediction.

4. Results and Evaluation

4.1 Feature Engineering

Table 4.1.1 shows the F1-score of each model on validation sets with different feature engineering methods. Text features converted by the CountVectorizer gave the best F1-score for all the models. For this text feature, feature filtering with mutual information and chi-square gave a high F1-score. In addition, SVM performed best among other models in feature sets.

CountVectorizer	0R	GNB	DT	SVM
MI 50	0.498	0.716	0.707	0.780
Chi-square 50	0.498	0.711	0.707	0.782
PCA 50	0.498	0.640	0.624	0.772
First 50	0.498	0.275	0.672	0.702
tf-idf	0R	GNB	DT	SVM
Chi-square 50	0.498	0.646	0.682	0.690
PCA 50	0.498	0.709	0.623	0.738
First 50	0.498	0.579	0.628	0.661
Doc2vec50	0R	GNB	DT	SVM
MI 50	0.498	0.645	0.569	0.715
PCA 50	0.498	0.616	0.589	0.730
First 50	0.498	0.652	0.544	0.666
All 150 + 2	0.498	0.635	0.577	0.722
Doc2vec100	0R	GNB	DT	SVM
All 300 + 2	0.498	0.614	0.557	0.728

Table 4.1.1 F1-score of classification models with various feature selection and text conversion methods.

Thus, SVM was selected as the final classification model and trained with features including text features transformed by CountVectorizer and chi-square feature filtering method.

4.2 Parameter tuning

The SVM model was tested on different regularization parameters, kernels, degrees and gamma values. It was found that SVM with regularization parameter 500, gamma 0.0001 and radial basis function (RBF) kernel gave the best performance with 0.77 accuracy and 0.77, 0.78, 0.64 F1-score for class 1.0, 2.0, and 3.0. In comparison, the model without parameter tuning gave an accuracy of 0.73 and F1-score of 0.72, 0.76, 0.05 for 1.0, 2.0, 3.0, which indicates that the parameter tuning effectively improved the performance of the model.

4.3 Feature size

For each model, F1-score was calculated with various feature dimension ranging from 10-150 as illustrated in figure 4.3.1.

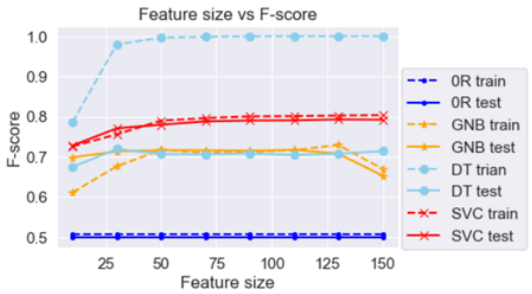

Figure 4.3.1 F1-score for classification models evaluated on the test and training sets with differing feature sizes.

For 0R, the F1-score was constant for both train and validation set regardless of the feature size as 0R always predicts the most common label in the training set. Both Gaussian Naïve Bayes and Decision Tree classifiers stay around 70% on the validation set. However, for the Decision Tree classifier, the F1-score for the training set is significantly high which is close to 100% compared to the validation set. This indicates that the Decision Tree model can be easily overfitted without careful consideration of parameter values and data processing. The F1-score for SVM increased as the feature dimension increases and gave the best F1-score with 0.792 at 140 features. This shows that SVM works well

with a high dimensional feature space.

4.4 Final Performance

For the final model, the SVM classification model performed the best precision, recall, F1-score and accuracy among the 4 models. This is followed by Gaussian Naïve Bayes, Decision Tree and 0R (Table 4.4.1).

	Precision	Recall	F1-score	Accuracy
0R	0.51	0.51	0.51	0.51
GNB	0.73	0.73	0.73	0.73
DT	0.72	0.72	0.72	0.72
SVM	0.80	0.80	0.80	0.80

Table 4.4.1 Precision, recall, F1-score and accuracy for the final models.

The 0R baseline model gave an F1-score of 0.51 by simply predicting the majority class label.

The SVM model uses the RBF kernel function to operate on high dimensional feature space and finds hyperplanes that maximise the margin between two classes with one against one approach. Since this model works for the high dimensional dataset and can be applied to data even if it is not linearly separable, it performed the best for the dataset with high dimensional feature space.

The Gaussian Naïve Bayes model gave around 0.73 F1-score. This model assumes the Gaussian assumption for the attributes for each class. The numerical attributes in the dataset are normally distributed as shown in Figure 4.4.2.

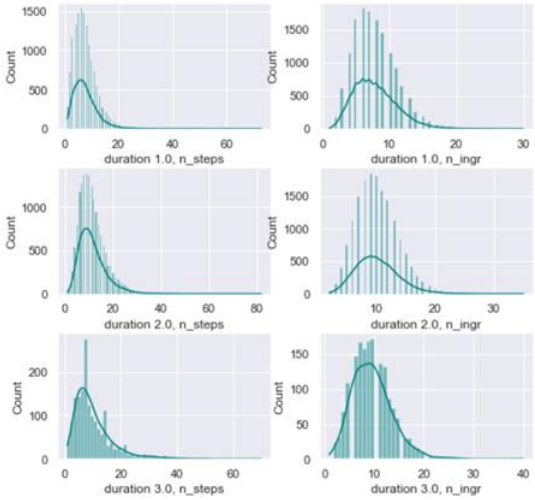

Figure 4.4.2 The distribution of numerical attributes, number of steps and ingredients for each class label.

However, the rest of the attributes made by CountVectorizer are the frequency of each word in the text features which violates the gaussian assumption. Therefore, the performance of the model was lower compared to SVM.

Decision Tree gave a 0.72 F1-score. The depth of a decision tree grows linearly as the dimension of feature space increases and gives a long hypothesis. This indicates that the model is less generalisable and susceptible to the effect of irrelevant features. Thus, the score performed lower than SVM but could be improved with proper parameter tuning.

4.5 Learning Curves

Figure 4.5.1 shows the data trade-off of the model. More training instance results in constructing a better model with lower mean squared error (MSE). More evaluation instance results in more reliable evaluation which means more model variance and less evaluation variance.

As shown in Figure 4.5.1, the MSE of training for SVM decreases and the MSE of evaluation increases as the training size increases. This indicates that a large proportion of the training results in less model variance and more evaluation variance. The optimal training size seems to be around 0.7-0.8. The high model variance is also evident for the Decision Tree classifier to a greater extent with an MSE of around 0.35 on the test and 0 on the training. On the contrary, 0R has constantly high training error which indicates the high model bias and zero variance.

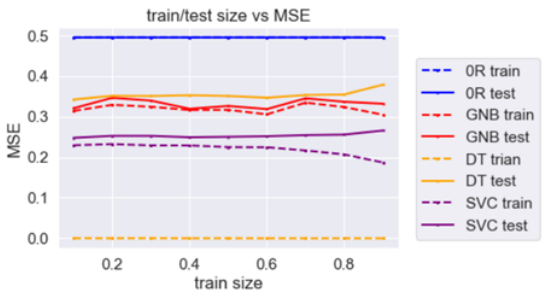

Figure 4.5.1 Percentage of training instances against F1-score on train and test sets.

4.6 Error analysis of the best performing model

The SVM performed well on the dataset as

you can see from the high number of diagonal elements in the confusion matrix (Figure 4.6.1). For the soft-margin SVM model, the regularisation parameter is used to allow some data points to violate the decision boundary. This will prevent overfitting the model especially when data contains some outliers, sampling bias and small training instances. Even though the number of correct predictions for the class 3.0 is small, the F1-score for 3.0 was found to be 0.72 which is decent considering the small sample size of the 3.0 class in the overall population.

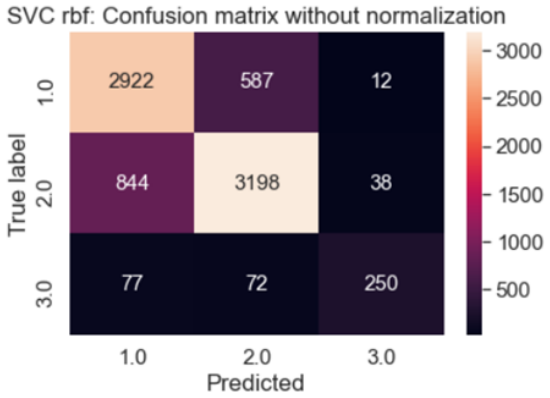

Figure 4.6.1 SVM confusion matrix

Random holdout and stratification strategy were used to control the bias and variance of the model. However, the text transformation methods including CountVectorizer and doc2vec were applied before the train/validation split which may have leaked the information of the validations during training.

5. Conclusion

It was found that SVM with regularization parameter 500 and gamma 0.0001 performed 0.80 F1-score after pre-processing, feature engineering and parameter tuning. This was 29 points higher than the benchmark 0R classifier.

Since the dataset was imbalanced with only 5% of class 3.0 instances, it turns out to have lower performance for the minority class. For future consideration, the Synthetic Minority Oversampling Technique could be incorporated to improve the performance of prediction for minority class by synthesising new examples.

6. References

Majumder, BP, Li, S, Ni, J & McAuley, J 2019, 'Generating personalized recipes from historical user preferences', Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing(EMNLP-IJCNLP), pp.5976–5982.

Kohavi, R & John, GH 1996, 'Wrappers for feature subset selection', *Artificial Intelligence*, vol.97, no.1-2, pp.273-324.

Forman, G 2003, 'An Extensive Empirical Study of Feature Selection Metrics for Text Classification', *Journal of Machine Learning Research*, vol.3, no.7-8, pp.1289-1305.