Machine Learning Models Selection on Mushroom Classification

Shuang Song， Xinran Liu, Shaoyang Sun

Abstract

Categorical problem is one of the essential tasks in machine learning, especially for multivariate characteristic data set. In this project, we try to classify the gilled mushrooms into edible and poisonous groups with 22 discrete features by different machine learning models and select the optimal one.

The inspiration is to find the most appropriate model for further prediction, based on the methods of K-Nearest Neighbors, logistic regression and support vector machines. According to the 22 characteristics, we use the method of principal components analysis to reduce the data structure into two-dimension. Then we split data into training sample and testing sample in different ratios and analyze three supervised processes respectively with changing hyper-parameters. In the end, we try to find the most suitable hyper-parameter and the model with best performance by calculating the accuracy scores, summarizing confusion matrices and the results visualization. It turns out that KNN fits this case most with 93.9% accuracy.

Keywords: Classification; K-Nearest Neighbors (K-NN); Logistic Regression; Support Vector machines (SVM)

1. Introduction

1.1 Task Description

Mushroom is an important fungus which contains a good source of vitamin B and a large amount of protein. It helps to prevent cancer and increases immunity power of human. While, some are toxic and can prove dangerous if we eat them. This dataset includes 23 species of gilled mushrooms, identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one.

Therefore, the task is to classify the gilled mushrooms into edible or poisonous group based on its comprehensive characteristics.

1.2 Major challenges and solutions

In this project, the mushroom has 22 features, like color, odor, shape and surface etc. In this case, the major challenge is that we need to build a multidimensional model because there are too many features. To solve this problem, the solution we found is to use the principal components analysis method (PCA) to process the data before building the model. And use the idea of dimensionality reduction to convert multiple indicators into two comprehensive indicators. In this way, we can not only make the data set easier to use, but also reduce the computational overhead of the algorithm.

2. Dataset Description

In the mushroom classification project, we train the models with 22 features which are introduced in Table 1.

Table 1 Variables description

Variable	Definition
Class(y)	poisonous, edible
cap-shape	bell,conical,convex,flat,knobbed,sunken
cap-surface	fibrous,grooves,scaly,smooth
cap-color	brown,buff,cinnamon,gray,green,pink,purple,red,white,yellow
bruises or not	with bruises,without bruises
odor	almond,anise,creosote,fishy,foul, musty,none,pungent,spicy
gill-attachment	attached,descending,free,notched
gill-spacing	close,crowded,distant
gill-size	broad,narrow
gill-color	black,brown,buff,chocolate,gray, green,orange,pink,purple,red, white,yellow
stalk-shape	enlarging,tapering
stalk-root	bulbous,club,cup,equal,rhizomorphs,rooted
stalk-surface-above-ring	fibrous,scaly,silky,smooth
stalk-surface-below-ring	fibrous,scaly,silky,smooth
stalk-color-above-ring	brown,buff,cinnamon,gray,orange, pink,red,white,yellow
stalk-color-below-ring	brown,buff,cinnamon,gray,orange, pink,red,white,yellow
veil-type	partial,universal
veil-color	brown,orange,white,yellow
ring-number	none,one,two
ring-type	cobwebby,evanescent,flaring,large, none,pendant,sheathing,zone
spore-print-color	black,brown,buff,chocolate,green, orange,purple,white,yellow
population	abundant,clustered,numerous, scattered,several,solitary
habitat	grasses,leaves,meadows,paths, urban,waste,woods

In Figure 1 and Figure 2, we visualize 22 features and thus it is obvious that gill color, spore print color and population are the variables that could take priority to consider when classification. On the contrary, the distributions of some features are very close which means it is very difficult to identify whether mushrooms are edible or not based on these characteristics, such as cap shape, cap surface and so on.

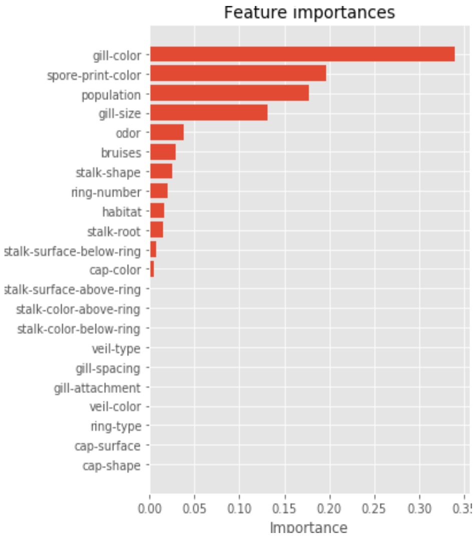


Figure 1 Feature importance

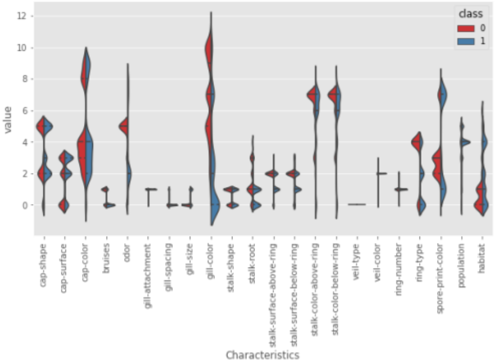


Figure 2 Characteristics

In addition, by drawing histograms of individual features, we can explain how to classify with one characteristic. Take odor as an example. Figure 3 shows that most of mushrooms odorless or foul. And Figure 4 illustrates that most of odorless mushrooms are edible, so do all of almond and anise odor mushrooms. Ones with other kinds of odor are poisonous.

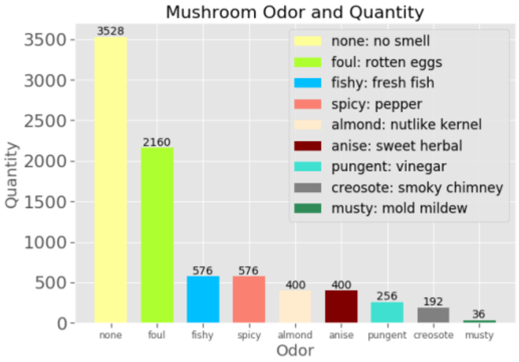


Figure 3 Histogram of odor

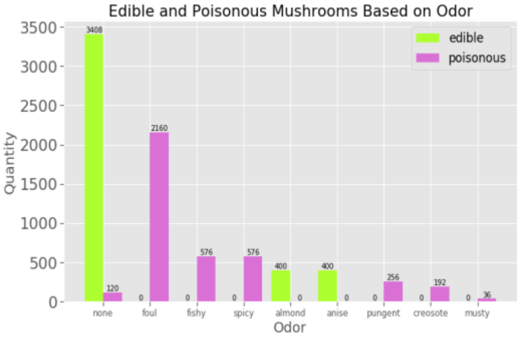


Figure 4 Classification based on odor

Another example is habit. With pie chart Figure 5, Figure 6, the proportions are visualized and most of mushrooms grow in woods and grasses are edible.

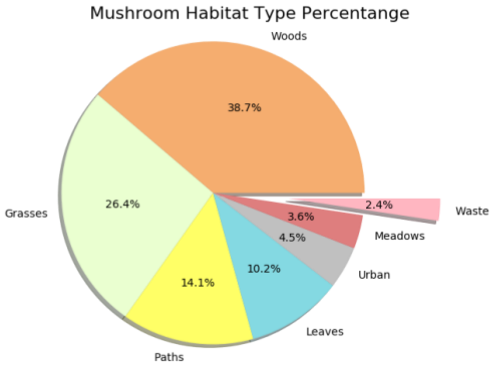


Figure 5 Pie chart of habit

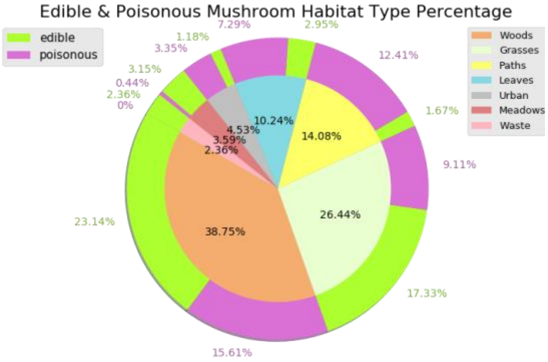


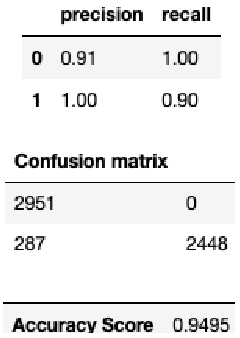
Figure 6 Classification based on habit

3. Experiments

3.1 K-Nearest Neighbor Method

K-Nearest Neighbor Method is one of the simplest methods in data mining classification technology. The core idea of the KNN algorithm is that if most of the k nearest neighbors of a sample in the feature space belong to a certain category, the sample also belongs to this category and has the characteristics of the samples in this category.

We use the K-Nearest Neighbor Method (K=2) to classification mushroom. The results of training set are shown in Figure 7, including precision and recall of each classification, confusion matrix and accuracy score, and the Figure 8 shows the result of test set.

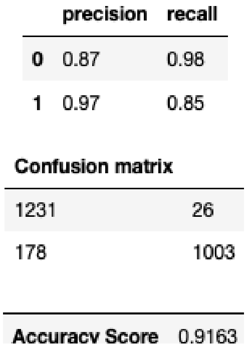


	precision	recall
0	0.91	1.00
1	1.00	0.90

Confusion matrix	
2951	0
287	2448

Accuracy Score	0.9495

Figure 7 KNN train evaluation (when K=2)



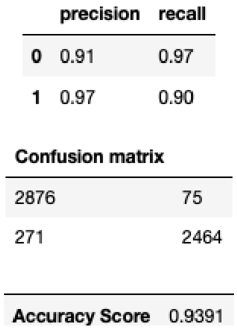
	precision	recall
0	0.87	0.98
1	0.97	0.85

Confusion matrix	
1231	26
178	1003

Accuracy Score	0.9163

Figure 8 KNN test evaluation (when K=2)

In order to increase the robust, we increase the K to 10 and 20. The Figure 9, Figure 10 show the precision and recall of each classification, confusion matrix and accuracy score of training result and test result when K is equal to 10.

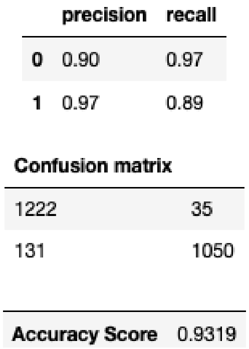


	precision	recall
0	0.91	0.97
1	0.97	0.90

Confusion matrix	
2876	75
271	2464

Accuracy Score	0.9391

Figure 9 KNN train evaluation (when K=10)



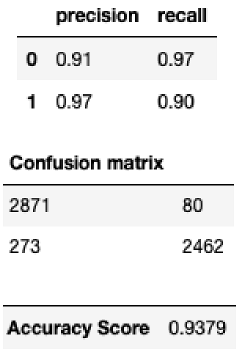
	precision	recall
0	0.90	0.97
1	0.97	0.89

Confusion matrix	
1222	35
131	1050

Accuracy Score	0.9319

Figure 10 KNN test evaluation (when K=10)

The Figure 11 and Figure 12 show the precision and recall of each classification, confusion matrix and accuracy score of training result and test result when K is equal to 20.

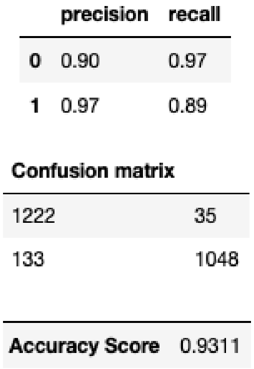


	precision	recall
0	0.91	0.97
1	0.97	0.90

Confusion matrix	
2871	80
273	2462

Accuracy Score	0.9379

Figure 11 KNN train evaluation (when K=20)



	precision	recall
0	0.90	0.97
1	0.97	0.89

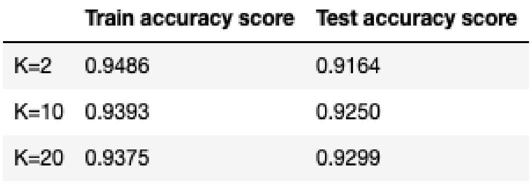
Confusion matrix	
1222	35
133	1048

Accuracy Score	0.9311

Figure 12 KNN test evaluation (when K=20)

In order to select the best model, we change the ratio of the number of training set to test set from7:3 to 9:1 and the result can be seen as Table 2.

Table 2 Accuracy score



	Train accuracy score	Test accuracy score
K=2	0.9486	0.9164
K=10	0.9393	0.9250
K=20	0.9375	0.9299

From the results, we can draw a conclusion that when the training set accounts for 70% of the total number, accuracy score and test accuracy score are closer comparing with the training set accounts for 90% of the total number. So, when the ratio of the number of training set to test set is 9:1, the model may have over fitting issues, and the model is better when the training set accounts for 70%. About the K, the accuracy score in figures show that as K increases, the robustness of the model continues to increase, but when K is too large, the test accuracy score decreases.

In conclusion, when training set accounts for 70% of the total number and K is equal to 10, the K-Nearest Neighbor Method model is best.

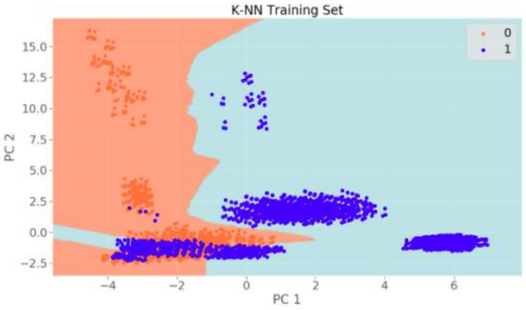


Figure 13 Training sample visualization

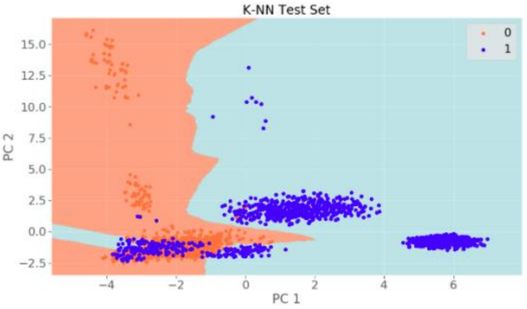


Figure 14 Test sample visualization

3.2 Logistic Regression

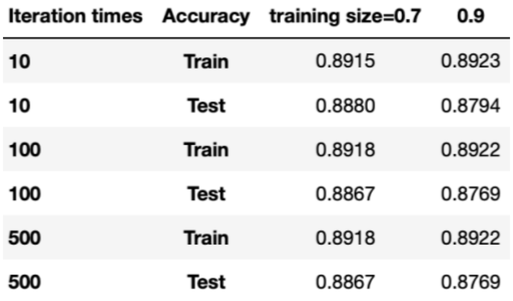
Logistic regression transforms its output using the logistic sigmoid function to return a probability value, which is a machine learning algorithm used to assign observations to a discrete set of classes. In this case, the classification is the binary type.

There are 15 kinds of hyper-parameter in the model, like the learning rate, number of iteration time, the type and of penalty, the weight of classes and so on. I would discuss the first three types for optimal hyper-parameter choice.

3.2.1 Iteration Times

In Table 3, it shows how the iteration times would affect the outcomes of the model. As we increase the iteration times, the accuracy of both training and testing scores would increase. But if we iterate the gradient descent process for too much times, there won't be more benefit and on the other hand, it takes longer time for calculation. Therefore, 100 times of iteration is suitable for the case.

Table 3

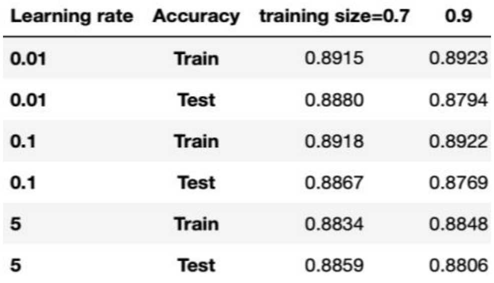


Iteration times	Accuracy	training size=0.7	0.9
10	Train	0.8915	0.8923
10	Test	0.8880	0.8794
100	Train	0.8918	0.8922
100	Test	0.8867	0.8769
500	Train	0.8918	0.8922
500	Test	0.8867	0.8769

3.2.2 Learning Rate

According to Table 4, the smaller the learning rate, the better the model would perform for most of training and testing samples. Theoretically, small learning rate does better in finding the global minimum cost, and it should match larger iteration times. In this experiment, the combination of 100 iteration times and 0.01 learning rate is optimal.

Table 4

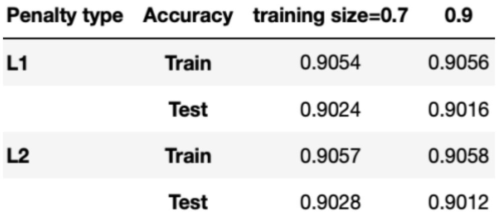


Learning rate	Accuracy	training size=0.7	0.9
0.01	Train	0.8915	0.8923
0.01	Test	0.8880	0.8794
0.1	Train	0.8918	0.8922
0.1	Test	0.8867	0.8769
5	Train	0.8834	0.8848
5	Test	0.8859	0.8806

3.2.3 Penalty

To deal with overfitting, we can add L1 or L2 to regularize the parameters. L1 is in absolute form and L2 is the quadratic form of regularization. According to Table 5, L2 fits the case better.

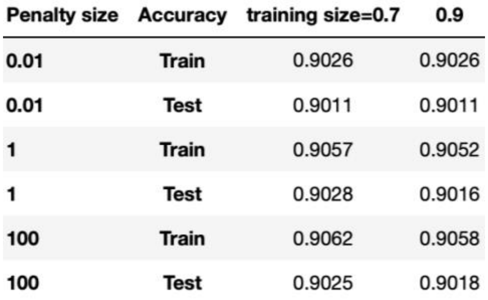
Table 5



Penalty type	Accuracy	training size=0.7	0.9
L1	Train	0.9054	0.9056
	Test	0.9024	0.9016
L2	Train	0.9057	0.9058
	Test	0.9028	0.9012

The penalty size, C, reflects model's tolerance over the noises. If C is close to 0, it means the model is very strict over noises and the bias would increase accordingly. But if C is too large, the gap between training and testing scores, or the variance, would increase and cause overfitting problem. Therefore, C equals 1, which is proper here.

Table 6



Penalty size	Accuracy	training size=0.7	0.9
0.01	Train	0.9026	0.9026
0.01	Test	0.9011	0.9011
1	Train	0.9057	0.9052
1	Test	0.9028	0.9016
100	Train	0.9062	0.9058
100	Test	0.9025	0.9018

3.2.4 Splitting Ratio

We find in every case that large training sample and limited testing sample would induce overfitting. If testing score is much worse than training one, it won't be an good model for prediction. Therefore, we choose to train 70% of the data.

3.2.5 Outcomes of Optimal Choice

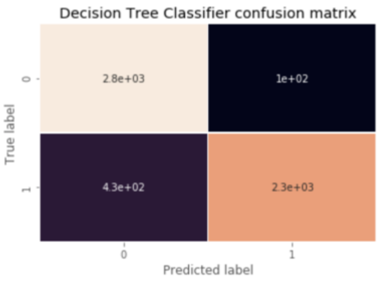
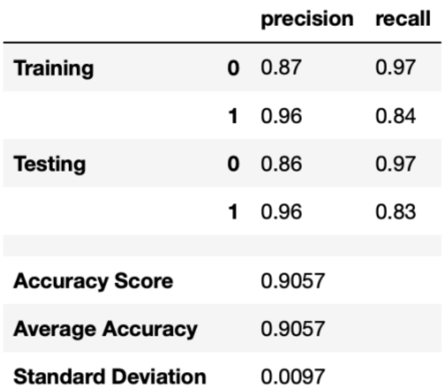


Figure 15 Confusion Matrix

Table 7



		precision	recall
Training	**0**	0.87	0.97
	1	0.96	0.84
Testing	**0**	0.86	0.97
	1	0.96	0.83
Accuracy Score		0.9057	
Average Accuracy		0.9057	
Standard Deviation		0.0097	

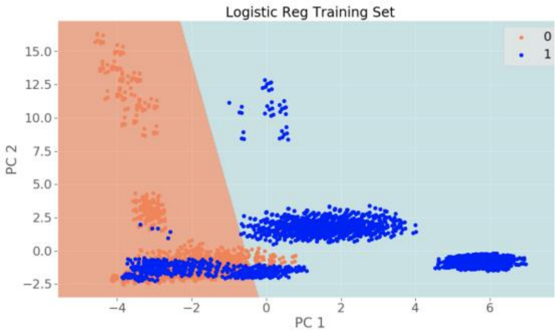


Figure 16 Training sample visualization

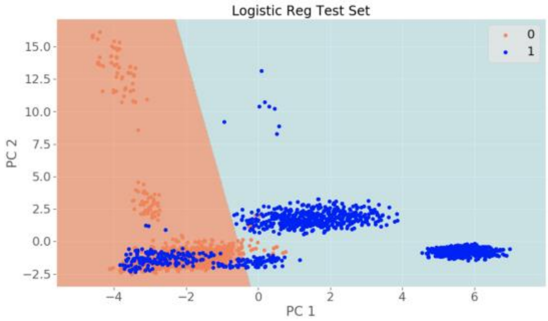


Figure 17 Testing sample visualization

3.3 Support Vector Machine

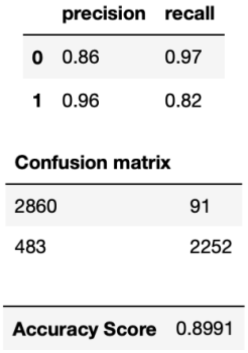
Support vector machines (SVM) are supervised learning models with associated learning algorithms that analyze data used for classification or regression. In this part, we will apply the SVM model to classify mushrooms into edible and poisonous ones.

In SVM model, the kernel type used in the algorithm has to be specified. Specifically, we will use linear, polynomial, and Radial basis function kernel respectively.

3.3.1 Linear Kernel SVM

Firstly, with the linear kernel SVM, we can calculate the Confusion Matrix (including accuracy, precision and recall rates) and accuracy scores, shown below.

Table 8 Linear Kernel Training evaluation

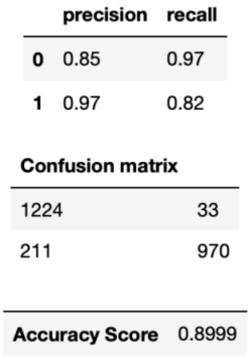


	precision	recall
0	0.86	0.97
1	0.96	0.82

Confusion matrix	
2860	91
483	2252

Accuracy Score	0.8991

Table 9 Linear Kernel Testing evaluation



	precision	recall
0	0.85	0.97
1	0.97	0.82

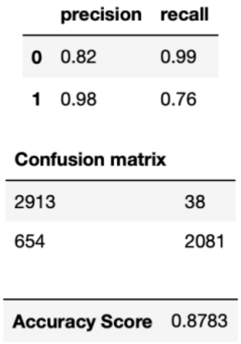
Confusion matrix	
1224	33
211	970

Accuracy Score	0.8999

According to training and testing results, we can see that the training score and testing score are 0.8991 and 0.8999 respectively, which means both models's bias and variance are not high. Therefore, the linear kernel SVM model is good to deal with the classification problem.

3.3.2 Polynomial Kernel SVM

Table 10 Polynomial Kernel Training evaluation

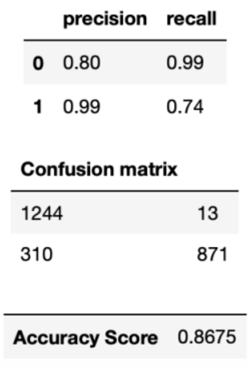


	precision	recall
0	0.82	0.99
1	0.98	0.76

Confusion matrix	
2913	38
654	2081

Accuracy Score	0.8783

Table 11 Polynomial Kernel Testing evaluation



	precision	recall
0	0.80	0.99
1	0.99	0.74

Confusion matrix	
1244	13
310	871

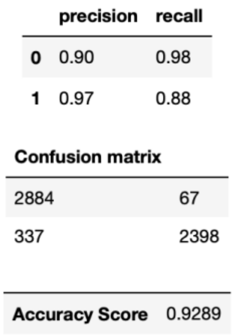
Accuracy Score	0.8675

Based on the training and testing evaluation tables, the accuracy scores illustrate that the polynomial model is not bad. Comparatively, linear performs better than polynomial with mushroom classification.

3.3.3 RBF Kernel SVM

Thirdly, a radial basis function kernel SVM is used, and the performance is explained by the Table 12, Table 13.

Table 12 RBF Kernel Training evaluation

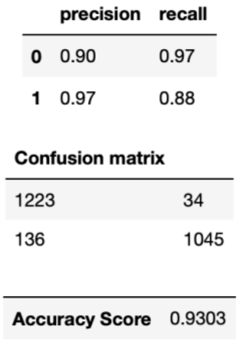


	precision	recall
0	0.90	0.98
1	0.97	0.88

Confusion matrix	
2884	67
337	2398

Accuracy Score	0.9289

Table 13 RBF Kernel Testing evaluation



	precision	recall
0	0.90	0.97
1	0.97	0.88

Confusion matrix	
1223	34
136	1045

Accuracy Score	0.9303

The training and testing accuracy scores are 0.9289 and 0.9303 respectively, i.e. the RBF kernel SVM model has extremely low bias and variance which means the model not only fits the training sample excellent but only could be used to predict precisely.

3.3.4 Best performance model

With the comparison of the three SVM models above, it is obvious that RBF Kernel model performs best.

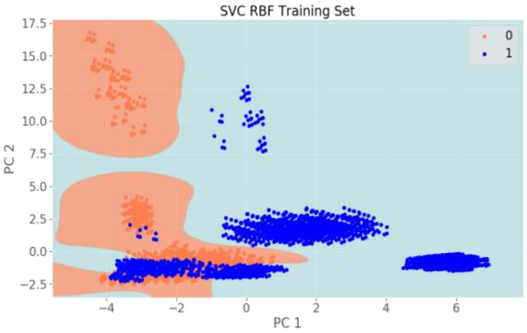


Figure 18 RBF Kernel SVM Training Model

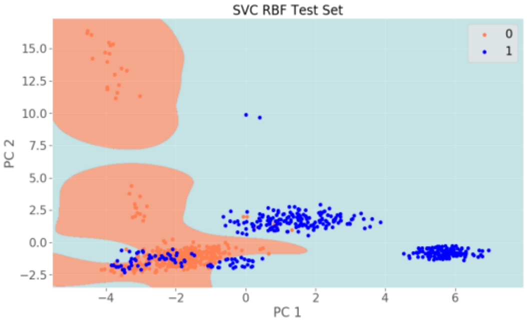


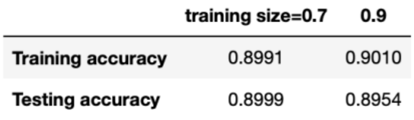
Figure 19 RBF Kernel SVM Testing Model

3.3.5 Different Sample Size

In this part, we compare the effect of changing training sample size on the prediction model performance. To be specific, we will re-train three-type models with 90% training sample size and 10% testing size.

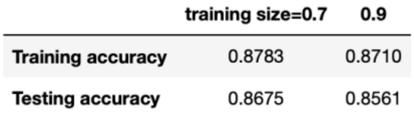
By repeating Kernel SVM models above, similar classification results can be obtained, but with different performance which is demonstrated directly by training and testing accuracy scores shown in Table 14, Table 15, Table 16.

Table 14 Accuracy scores of Linear Kernel



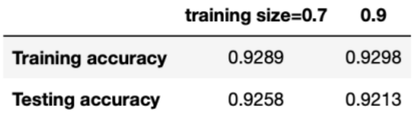
	training size=0.7	0.9
Training accuracy	0.8991	0.9010
Testing accuracy	0.8999	0.8954

Table 15 Accuracy scores of Polynomial Kernel



	training size=0.7	0.9
Training accuracy	0.8783	0.8710
Testing accuracy	0.8675	0.8561

Table 16 Accuracy scores of Polynomial Kernel



	training size=0.7	0.9
Training accuracy	0.9289	0.9298
Testing accuracy	0.9258	0.9213

Based on the results above, it is safe to say that increasing the training sample size will improve the model itself and decrease the model bias. However, due to small proportion of testing data, the models perform not as good as training process when they are used to predict.

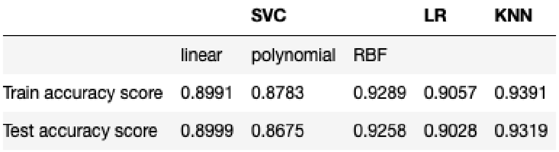
3.3.6 Optimal Choice

Consequently, we will choose Radial Basis Function Kernel SVM with 70% training sample size in SVM method part.

4. Conclusion and Future Work

In conclusion, we use the K-Nearest Neighbor Method (KNN), Support Vector Classification (SVC) and Logistic Regression to classify the mushrooms. In details, in order to find the best model, first we change the ratio of the number of the training set to test set for each model. Second, In KNN model, we change the number of K. In logistic regression, we change the alpha, number of iteration and the penalties C. In SVC, we use the polynomial kernel SVM, linear kernel SVM and RBF kernel SVM methods. Third, we pick an optimal in each model and compare the accuracy score of the three models.

Table 17 Accuracy score of three method



	SVC			LR	KNN
	linear	polynomial	RBF		
Train accuracy score	0.8991	0.8783	0.9289	0.9057	0.9391
Test accuracy score	0.8999	0.8675	0.9258	0.9028	0.9319

From Table 17, we can find that the KNN method has the highest test accuracy. Therefore, in this problem, we use the KNN method to classify the mushroom and the accuracy score is 0.9319.

About this problem, we use the three method, in the future we can use more method to classify the mushroom. For example, we can use naive Bayes classifier because the futures of the mushroom are independent, and we can also use the random forest method to classify the mushroom, because random forest method can calculate the importance of variables and the distance between data points.

5. References

[1] Ringnér, M. (2008). What is principal component analysis. Nature biotechnology, 26(3), 303-304.

[2] Maurya, P., & Singh, N. P. (2020). Mushroom Classification Using Feature-Based Machine Learning Approach. In Proceedings of 3rd International Conference on Computer Vision and Image Processing (pp. 197-206). Springer, Singapore.

[3] Mari, S. I., & Wu, S. (1999). Improving support vector machine classifiers by modifying kernel functions. Neural Networks, 12(6), 783-789.