Data Mining Take Home Final Exam Report

Jingsong Yuan 161006003

1. Algorithms Description

Here are the algorithms I used in classification and regression:

Classification	Regression
SVM Linear	Linear
SVM RBF Kernel	Neural Networks
Neural Networks	GBM
GBM	Random Forest
Random Forest	
Logistic	
SVM Polynomial Kernel	

2. Data Set Description

Here are the data I used in classification and regression:

1) Regression: Boston housing
2) Binary-classification: wdbc, hypothyroid, ionosphere

For cross validation:

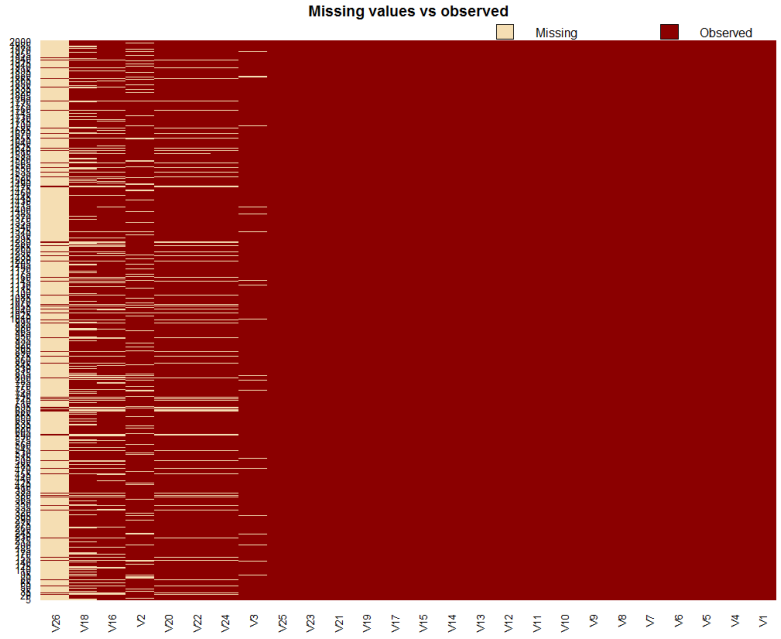
I used 70/30 splits for my training data for parameter tuning. I set the seed to 470 in the code in this cross validation data partition.

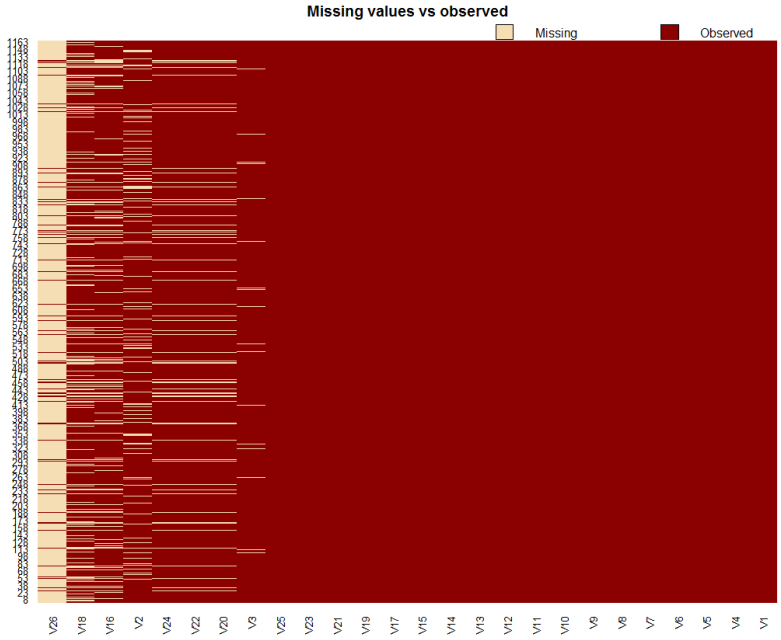
3. Data Preprocessing

3.1. Missing Data

The last variable named “TBG” of the hypothyroid data is missing too many values. And some other variables miss some data randomly. The usual method is to delete the “TBG” feature and impute the other missing values with mean of the nonmissing values for that features. However, I found that the missing data is relevant to the other features. So, deleting them or imputing mean values to them may cause some errors. As a result, I keep the feature “TBG” and change the missing label to “-1” for algorithm modeling. Besides, in order to test the performance between deleting the feature “TBG” and keeping it, I run the experiment using both preprocessing data. And I found deleting the feature “TBG” indeed decrease the performance. That’s why and how I handle the missing data.

Missing data visualization for the hypothyroid data set:





3.2. Irrelevant Data

In the wdbc data, the first feature “ID number” is meaningless to the prediction methods, so I would delete this feature before training.

In the ionosphere data, the second feature is all “0” without variation, so this feature could be deleted before training without influence the performance.

3.3. Data representation

Some variables contain binary labels represented by letters. For algorithm modeling, I replaced the binary labels to “1” and “0”.

4. Classification Problems

4.1. Parameters Tuning Results

I use the classification error to evaluate the performance of different parameters setting. And I implemented a caret-like function to evaluate it. At first, I tried the "caret" function to evaluate different parameters' performance, however, it seems not very time efficient. As a result, I think the classification error can fully evaluate the performance.

In this part, I present my program output as the table.

4.1.1. Random Forest

The following table shows the experimental result of validation error of Random Forest using 3 different data sets with different parameters.

RandomForest-Hypothyroid(ntree=10000)					
m-try	5	2	7	15	25
validation error	0.008333333	0.011666667	0.00833333	0.005	0.005
test error				0.0172	
RandomForest-ionosphere(ntree=10000)					
m-try	5.8309519	2	7	15	25
validation error	0.04545455	0.045454545	0.06060606	0.090909	0.10606
test error	0.0610687				
RandomForest-wdbc(ntree=10000)					
m-try	5.47722558	2	7	15	25
validation error	0.05555556	0.055555556	0.05555556	0.055556	0.05556
test error	0.02230483				

The parameters in the Random Forest algorithm contain ntree and m-try.

And from my research and experiment test, I found the ntree would not influence the performance of the method if the value is big enough.

From this table, we can see:

1. The best parameter setting for Hypothyroid is ntree=10000 and m-try=15, the validation error has the min value as 0.005. So I chose these parameter setting to test the data and obtain the test error as 0.0172.

2. The best parameter setting for Ionosphere is ntree=10000 and m-try=5.8309519, which is the default value as sqrt(p). The p is the numbers of the features in X. And the validation error has the min value as 0.045. So I chose these parameter setting to test the data and obtain the test error as 0.061.

3. The best parameter setting for Ionosphere is ntree=10000 and m-try=5.47722558, which is the default value as sqrt(p). The p is the numbers of the features in X. And the validation error has the min value as 0.056. So I chose these parameter setting to test the data and obtain the test error as 0.0223.

4.1.2. Linear SVM

The following table shows the experimental result of validation error of linear SVM using 3

different data sets with different parameters.

SVM_Linear-Hypothyroid													
lambda	0.0001	0.00032	0.001	0.0032	0.01	0.032	0.1	0.32	**1**	3.2	10	32	100
validation error	0.046666667	0.04666667	0.045	0.03333	0.02667	0.0166667	0.013333333	0.013333333	**0.0117**	0.01333	0.01333	0.01333	0.01333
test error									**0.0292**				
SVM_Linear-Ionosphere													
lambda	0.0001	0.00032	0.001	0.0032	0.01	0.032	**0.1**	0.32	1	3.2	10	32	100
validation error	0.318181818	0.31818182	0.31818182	0.31818	0.18182	0.1060606	**0.106060606**	0.106060606	0.13636	0.15152	0.15152	0.15152	0.13636
test error							**0.106870229**						
SVM_Linear-WDBC													
lambda	0.0001	0.00032	0.001	0.0032	0.01	0.032	**0.1**	0.32	1	3.2	10	32	100
validation error	0.444444444	0.277777778	0.11111111	0.077778	0.04444	0.04444444	**0.011111111**	0.011111111	0.01111	0.022222	0.04444	0.044444	0.04444
test error							**0.029739777**						

The parameter in this algorithm contain lambda.

From this table, we can see:

1. The best parameter setting for Hypothyroid is lambda=1, the validation error has the min value as 0.0117. So I chose this parameter setting to test the data and obtain the test error as 0.0292.
2. The best parameter setting for Ionosphere is lambda =0.1.And the validation error has the min value as 0.106. So I chose this parameter setting to test the data and obtain the test error as 0.1068.
3. The best parameter setting for Ionosphere is lambda=0.1. And the validation error has the min value as 0.0111. So I chose these parameter setting to test the data and obtain the test error as 0.0297.

4.1.3. RBF-SVM

The following table shows the experimental result of validation error of RBF SVM using 3 different data sets with different parameters.

SVM_RBF-Hypothyroid(lambda=3.2)										
sigma	0.05	0.1	0.2	0.3	0.4	0.5	0.7	1	2	**5**
validation error	0.043333333	0.04333333	0.04333333	0.04167	0.03667	0.0316667	0.021666667	0.021666667	0.02167	**0.0117**
test error										**0.03181**
SVM_RBF-Ionosphere(lambda=3.2)										
sigma	0.05	0.1	0.2	0.3	0.4	0.5	0.7	**1**	2	5
validation error	0.303030303	0.3030303	0.3030303	0.287879	0.15152	0.1515152	0.075757576	**0.03030303**	0.04545	0.060606
test error								**0.04580153**		
SVM_RBF-WDBC(lambda=3.2)										
sigma	0.05	0.1	0.2	0.3	0.4	0.5	0.7	1	2	**5**
validation error	0.444444444	0.44444444	0.44444444	0.44444	0.44444	0.4444444	0.444444444	0.277777778	0.04444	**0.01111**
test error										**0.0297**

The parameters in this algorithm contain lambda and sigma.

And from my experiment test, I found the lambda=3.2, this method has the best performance. So I set the lambda as 3.2 and then tune the sigma.

From this table, we can see:

1. The best parameter setting for Hypothyroid is sigma=5, the validation error has the min value as 0.0117. So I chose this parameter setting to test the data and obtain the test error as 0.0318.
2. The best parameter setting for Ionosphere is sigma = 1.And the validation error has the min value as 0.0303. So I chose this parameter setting to test the data and obtain the test error as 0.0458.
3. The best parameter setting for Ionosphere is sigma = 5. And the validation error has the min value as 0.0111. So I chose these parameter setting to test the data and obtain the test error as 0.0297.

4.1.4. Polynomial-SVM

The following table shows the experimental result of validation error of Polynomial SVM using 3 different data sets with different parameters.

SVM_Poly-Hypothyroid(lambda=3.2)							
degree	1	2	3	**4**	5	7	10
validation error	0.013333333	0.015	0.01833333	**0.0133**	0.02	0.035	0.021666667
test error				0.02322			
SVM_Poly-Ionosphere(lambda=0.01)							
degree	1	**2**	3	4	5	7	10
validation error	0.181818182	**0.0606061**	0.07575758	0.106061	0.09091	0.1060606	0.090909091
test error		0.10687023					
SVM_Poly-WDBC(lambda=3.2)							
degree	1	**2**	3	4	5	7	10
validation error	0.022222222	**0.0222222**	0.06666667	0.1	0.06667	0.0555556	0.111111111
test error		0.21189591					

The parameters in the algorithm contain lambda and degree.

And from my experiment test, I found the lambda=3.2, this method has the best performance. So I set the lambda as 3.2 and then tune the degree.

From this table, we can see:

1. The best parameter setting for Hypothyroid is degree =4, the validation error has the min value as 0.0133. So I chose this parameter setting to test the data and obtain the test error as 0.0232.

2. The best parameter setting for Ionosphere is degree = 2.And the validation error has the min value as 0.0606. So I chose this parameter setting to test the data and obtain the test error as 0.1069.

3. The best parameter setting for Ionosphere is degree = 2. And the validation error has the min value as 0.0222. So I chose these parameter setting to test the data and obtain the test error as 0.2119.

The error of the SVM using the WDBC achieved the quite large value of 0.2119. And it is because the iteration number reaching the max value in this algorithm. So this caused the big error on the test data.

4.1.5. Logistic Regression for Classification

The following table shows the experimental result of validation error of Logistic Regression using 3 different data sets.

Logistic Regression for Classification			
	Hypothyroid	Ionosphere	WDBC
validation error	0.0115	0.05454545	0
test error	0.022355976	0.15267176	0.063197026

There is no parameter to be tuned in this method.

1. The test error of the method using Hypothyroid data is 0.0224.

2. The test error of the method using Ionosphere data is 0.05454.

1. The test error of the method using WDBC data is 0.0632.

4.1.6. Neutral Network for Classification

The following table shows the experimental result of validation error of Neutral Network using 3 different data sets with different parameters.

The parameters in the algorithm contain hidden value and threshold value. I test the hidden in [1,

2, 3] and threshold in [1, 0.1, 0.01, 0.001].

Neural Network for Classification(hidden=1)					
threshold		1	0.1	0.01	0.001
validation error	Hypothyroid	0.04666667	0.01333333	0.015	0.01333
	Ionosphere	0.27272727	0.16666667	0.15152	0.18182
	WDBC	0.14444444	0.11111111	0.15556	0.08889

Neural Network for Classification(hidden=8)					
threshold		1	0.1	0.01	0.001
validation error	Hypothyroid	0.01667	0.011667	0.016667	0.01833
	Ionosphere	0.07576	0.106061	0.15152	0.121212
	WDBC	0.05556	0.05556	0.07778	0.06667

Neural Network for Classification(hidden=15)					
threshold		1	0.1	0.01	0.001
validation error	Hypothyroid	0.01833	0.01	0.015	0.01167
	Ionosphere	0.13636	0.07576	0.07576	0.10606
	WDBC	0.05556	0.05556	0.03333	0.03333
test error	Hypothyroid		0.024076		
	Ionosphere		0.0687		
	WDBC			0.07063	

From this table, we can see:

1. The best parameter setting for Hypothyroid is hidden =15 and the threshold = 0.1, the validation error has the min value as 0.01. So I chose this parameter setting to test the data and obtain the test error as 0.024076.

2. The best parameter setting for Ionosphere is hidden =15 and the threshold = 0.1. And the validation error has the min value as 0.0758. So I chose this parameter setting to test the data and obtain the test error as 0.0687.

3. The best parameter setting for Ionosphere is hidden =15 and the threshold = 0.01. And the validation error has the min value as 0.0333. So I chose these parameter setting to test the data and obtain the test error as 0.07063.

From the experiment, I found that the neural network is quite unstable, the prediction is quite different when the hidden is small, for example, when hidden is 1. So, for better performance the hidden should be large enough but balancing the running time. And the threshold acts as the partial derivatives of the error function as stopping criteria. The smaller the value, the more running time would be used. And I think when the value of threshold equals 1, it can achieve a good performance with a fast running time.

4.1.7. GBM for Classification

The following table shows the experimental result of validation error of GBM using 3 different data sets with different parameters.

The parameters in the algorithm contain shrinkage value, iteration depth and iteration value.

The shrinkage value is learning rate. Basically, when the shrinkage is higher, the accuracy is higher. But it would increase the iteration number which taking much running time. And I found when the shrinkage is 0.01, the performance is quite good. And the running time of the method is acceptable.

The iteration number can be obtained in the function automatically using error evaluation and cross-validation.

GBM-Hypothyroid(shrinkage=0.01)			
interactionDepth	**1**	5	10
iteration	**1191**	371	450
validation error	**0.0067**	0.008333	0.00833
test error	0.02064		
GBM-ionosphere((shrinkage=0.01)			
interactionDepth	1	**5**	10
iteration	893	**455**	525
validation error	0.09091	**0.04545**	0.04545
test error		0.08397	
GBM-wdbc((shrinkage=0.01)			
interactionDepth	**1**	5	10
iteration	**1115**	492	736
validation error	**0.0111**	0.03333	0.02222
test error	**0.0372**		

From this table, we can see:

1. The best parameter setting for Hypothyroid is shrinkage =0.01, interactionDepth=1 and the iteration = 1191, the validation error has the min value as 0.0067. So I chose this parameter setting to test the data and obtain the test error as 0.02064.

2. The best parameter setting for ionosphere is shrinkage =0.01, interactionDepth=5 and the iteration = 455, the validation error has the min value as 0.04545. So I chose this parameter setting to test the data and obtain the test error as 0.08397.

3. The best parameter setting for wdbc is shrinkage =0.01, interactionDepth=1 and the iteration = 1115, the validation error has the min value as 0.0111. So I chose this parameter setting to test the data and obtain the test error as 0.0372.

From the experiment, I found that mostly, when the interaction depth is 1, the method can be a quite good performance. It is because 1 implies an additive model. The additive model is good for hypothyroid data and wdbc data but not so good for ionosphere.

4.2. Test Data Performance

Using the tuned parameters above, I compare the performance between different classification methods:

Test Error	Data Set		
Classification	hypothyroid	ionosphere	wdbc
RandomForest	0.017196905	0.0610687	0.0223
SVM_Linear	0.02923474	0.10687023	0.02974
SVM_RBF	0.031814273	0.04580153	0.02974
SVM_Poly	0.02321582	0.10687023	0.2119
Logistic Regression	0.02235598	0.15267176	0.063197
Neural Network	0.024075666	0.06870229	0.07063
GBM	0.02063629	0.08396947	0.03717

As we can see:

1. For the hypothyroid data, the random forest method has the smallest test error, therefore, I would chose random forest to prediction in this data set.
2. For the ionosphere data, the SVM with RBF kernel method has the smallest test error, therefore, I would chose SVM with RBF kernel method to prediction in this data set.
3. For the hypothyroid data, the random forest method has the smallest test error, therefore, I would chose random forest to prediction in this data set.

Explanation:

For the ionosphere data set, all the method performed with the worst accuracy. It is because the ionosphere's all 34 predictor attributes are continuous. And the second feature in the data set is all "0" without the variation. So the prediction accuracy for the ionosphere is much lower.

4.3. Model Averaging

Considering the test error is very similar, therefore I would use voting as my model averaging method.

Through voting method, I improved test error to be 0.015423 on the hypothyroid data set. And improved test error to be 0.0601 on the ionosphere data. But in the wdbc data, the SVM_poly error is much higher than the others. So I drop this model and averaging the others. I obtained the error of 0.02107.

5. Regression Problems

5.1. Parameters Tuning Results

I use the mean square error(MSE) to evaluate the performance of different parameters setting. And I implemented a caret-like function to evaluate it. At first, I tried the "caret" function to evaluate different parameters' performance, however, it seems not very time efficient. As a result, I think the classification MSE can fully evaluate the performance.

In this part, I present my program output as the table.

5.1.1 Ridge Regression

The following table shows the experimental result of validation MSE of ridge regression using 3 different data sets with different parameters.

Ridge Regression													
lambda	1.00E-04	1.00E-03	1.00E-02	0.1	1	10	1.00E+02	1.00E+03	1.00E+04	1.00E+05	1.00E+06	1.00E+07	1.00E+08
validation MSE	12.79093	12.79131	12.79973	13.2209	20.4477	26.8233	38.38943	56.2586	61.85525	62.56192	62.6345	62.6418	62
test MSE	378.3199												

The parameter in this algorithm contain lambda.

From this table, we can see:

The best parameter setting for the data is lambda=0.0001, the validation MSE has the min value as 12.79. So I chose this parameter setting to test the data and obtain the test MSE as 378.32.

5.1.2 Random Forest

The following table shows the experimental result of validation MSE of random forest using 3 different data sets with different parameters.

Random Forest for Regression(Ntree=10000)				
mtry	4.3	10	2	**6**
validation MSE	7.36418	7.4122	9.22176	**7.1535**
test MSE				**32.479**

The parameters in the Random Forest algorithm contain ntree and m-try.

And from my research and experiment test, I found the Ntree would not influence the performance of the method if the value is big enough.

From this table, we can see:

The best parameter setting for the data is ntree=10000 and mtry=6, the validation MSE has the min value as 7.1535. So I chose these parameter setting to test the data and obtain the test error as 34.479.

From the experiment, I found that if the mtry value is too small, the performance of the method would be worse(such as 2). But when the mtry value as large as some value(such as 6), the performance would not improve as the value enlarged. So I think the mtry=6 is suitable to obtain a good performance saving method running time.

5.1.3 Neural Network

The following table shows the experimental result of validation error of random forest using 3 different data sets with different parameters.

Neural Network for Regression(hidden=1)				
threshold	1	**5**	10	20
validation MSE	8.381779	**8.25858**	8.26182	9.88876

Neural Network for Regression(hidden=2)				
threshold	**1**	5	0.1	20
validation MSE	**5.15915**	11.32589	5.97277	11.8427
test MSE	38.78669			

Neural Network for Regression(hidden=3)				
threshold	1	**5**	10	20
validation MSE	10.049	**9.2921**	12.3057	9.59101
test MSE				

The parameters in this algorithm contain hidden and threshold.

From this table, we can see:

The best parameter setting for the data is hidden=2 and threshold=1, the validation MSE has the min value as 5.159. So I chose these parameters setting to test the data and obtain the test error as 38.787.

From the experiment, I found that the hidden value and the threshold should not be tuned to be too large which is different from using this method for classification.

5.1.4 GBM

The following table shows the experimental result of validation MSE of GBM using 3 different data sets with different parameters.

The parameters in the algorithm contain shrinkage value, iteration depth and iteration value.

The shrinkage value is learning rate. Basically, when the shrinkage is higher, the accuracy is higher. But it would increase the iteration number which taking much running time. And I found when

the shrinkage is 0.01, the performance is quite good. And the running time of the method is acceptable.

The iteration number can be obtained in the function automatically using error evaluation and cross-validation.

GBM(shrinkage=0.01)			
interactionDepth	**1**	5	10
iteration	**1833**	9999	9976
validation MSE	**8.914**	7.76165	7.60796
test MSE	9.80516		

From this table, we can see:

1. The best parameter setting for the data is shrinkage =0.01, interactionDepth=1 and the iteration = 1833, the validation MSE has the min value as 8.914. So I chose this parameter setting to test the data and obtain the test MSE as 9.80516.

5.2 Test Data Performance

Using the tuned parameters above, I compare the performance between different regression methods:

Regression	MSE
Ridge Regression	378.31991
Neural Network	38.78669
Random Forest	32.4789
GBM	9.805163

As we can see, the GBM get smallest MSE. Therefore, I would select the GBM model in this data set.

Explanation:

The GBM got a best performance compared to the other regression method here. It is because the GBM is ensembles of weak prediction models especially using decision trees. It used the boosting method to get the suitable cost function. Basically, it trains learners sequentially and then combines these learners to get a more complicated model. Thus, the GBM get a best accuracy in this regression.

5.3. Model Averaging

In this case, the MSE is very different from different regression methods here. So it should not be averaged by using voting here. So I would use the stacking model averaging method with weight optimization.

6. Program Outputs

For better format and understanding, I combine the outputs into the table shown above.