Machine Learning Project of Dataset #3

Shiyao Wang, Jia Luo, Yiwen Yang, Nuo Wang

COMP 6915, Machine Learning

Professor Pena-Castillo, Lourdes and Professor Hu, Ting

April 11, 2016,

Contents

1 Introduction **3**

2 Preprocessing **4**

2.1 Missing Data 4

2.2 Dummy Variable 4

2.3 Feature Selection 4

2.4 Preprocessing Reuslts 5

3 Failed Attempts **6**

3.1 Multilayer perceptron 6

3.2 Restricted Boltzmann Machine 6

3.3 Logistic Regression 7

4 Good Approach **8**

4.1 K-Nearest Neighbors (KNN) 8

4.2 Support vector machine (SVM) 8

4.3 Naive Bayes 9

4.4 Decision Tree 10

5 Final Results **11**

5.1 Final results of KNN method 11

5.2 Final results of SVM method 14

5.3 Final Results-Naive Bayes and Decision Tree 16

5.4 Comparison between four algorithms 17

6 Conclusion **18**

7 Future Work **18**

Machine Learning Project of Dataset #3

Shiyao Wang, Jia Luo, Yiwen Yang, Nuo Wang

Abstract

This study investigates the effectiveness of various machine learning methods for dataset #3 related to genetic association. We introduce our preprocessing methods, as well as some failed attempts, including Multilayer Perceptron, Restricted Boltzmann Machine and Logistic Regression. These algorithms are considered not suitable for our dataset. Then some other approaches have been tried, such as K-Nearest Neighbors, Naive Bayes, Decision Trees and Kernel based SVM, and reasonable results are obtained. Finally, some conclusions are discussed based on the results.

1. Introduction

Genetic studies associated with a certain disease are pressing and far-reaching in human health and they have been a popular topic over the past decades. Genetic association studies look for genetic attributes, i.e. single nucleotide polymorphisms (SNPs) that have high susceptibility risks associated with a certain disease. Usually DNA sequences are collected from both diseased patients and healthy individuals. SNPs that are most relevant to this disease should be found by comparing these two populations [1, 2].

In machine learning, this is a classification and feature selection problem. In order to obtain reasonable results, various machine learning methods have been explored in the project. Due to their own advantages and limitations, some attempts cannot outcome good results. However, several approaches provide decent outputs.

The rest of the report is organized as follows: Section 2 introduces preprocessing techniques among original data. Section 3 describes the failed attempts. Section 4 proposes some good approaches. The empirical results are then illustrated in Section 5 and summarized in Section 6.

2. Preprocessing

2.1. Missing Data

The raw dataset has 1694 features, including 1692 SNPs along with gender and smoking status. The first column "cancer" is the class label with 0 indicates healthy and 1 for disease. The dataset has 2048 samples, and part of features are missing in several samples. In order to cope with missing features in the dataset, three basic methods are applied [3, 4, 5, 6, 7, 8, 9, 10, 11]:

- **Sample Deletion:** If one sample contains a certain percent of missing features, this sample would be ignored.
- **Feature Deletion:** If one feature is missing in a certain proportion of samples, this feature would be ignored.
- **Imputation:** In general, a missing value would be replaced by the mean value for numerical feature and the most likely value for discrete one. In our case, most likely value would be used.

For Sample and Feature Deletion, two different thresholds (0.05, 0.95) are investigated.

2.2. Dummy Variable

All the SNP features are qualitative value with different labels (0,1,2) indicating different status, as well as smoking status (1,2,3). To ensure that the distance between each two states in the same feature is equal, dummy variables or the hamming distance should be considered. In our project, dummy variables are introduced. For a given feature with three possible status, three binary values are used accordingly to represent each status [14].

2.3. Feature Selection

For feature selection, three selection criteria are developed:

- **Gin Index and Cross Entropy:** Two measures that can indicate the impurity of a certain feature are proposed. For any feature that has a value of Gini Index or Cross Entropy lower than a certain threshold would be ignored. The feature is decided by measuring a feature with a distribution of (0.8, 0.1, 0.1). The threshold is 0.1 for Gini Index, and 0.2 for Cross Entropy.

- **Complex:** This criterion use Gini Index to measure the impurity of the final result, i.e., cancer status. The value of one feature is measured by the minimum Gini Index of final output among every state of this feature. This value can indicate how one feature can separate the cancer status. For instance, if all individuals with smoking state 1 have cancer status 1, it means that for state 1, the Gini Index equals to 0, then the value of smoking state would be 0. Therefore, we can use this value to select features. Threshold of this value (0.2) is decided based on smoking status and gender status (all around 0.18).

2.4. Preprocessing Reuslts

		Samples	Features
Original Data		2048	1694
Sample Deletion	$\theta = 0.05$	1123	1694
	$\theta = 0.95$	1644	1694
Feature Deletion	$\theta = 0.05$	1123	1586
	$\theta = 0.95$	1644	1690

Table 1. Listwise Deletion Result

		Features	Dummy Variables
Data After Deletion	$\theta = 0.05$	1586	4715
	$\theta = 0.95$	1690	5020
Gini Index	$\theta = 0.05$	1264	3788
	$\theta = 0.95$	1500	4483
Cross Entropy	$\theta = 0.05$	1098	3292
	$\theta = 0.95$	1439	4304
Complex	$\theta = 0.05$	698	2065
	$\theta = 0.95$	144	415

Table 2. Feature Selection Result

3. Failed Attempts

3.1. Multilayer perceptron

The first approach we try is the multilayer perceptron (MLP)[15, 16, 17, 18]. It is powerful and it can distinguish data easily. Therefore, we choose it as the first test algorithm. This algorithm has a simple conception and perform well in classification tasks. It is a feed forward artificial neural network, which maps sets of input data onto a set of appropriate outputs. Its structure is displayed in Figure 1.

However, when we tried to train this network, the input size is quite large. Millions of floating numbers would be included in the connection weights. Even after preprocessing, the final input length is still not small enough. In MLP, conventionally, the number of units in hidden layer should be at least equal to the length of the input. Even if we set a smaller hidden layer size, for instance, a net with 200 hidden layers, still requires at least 20 seconds to update once. Without implementing on GPU processor, the convergence of MLP is actually time-consuming..

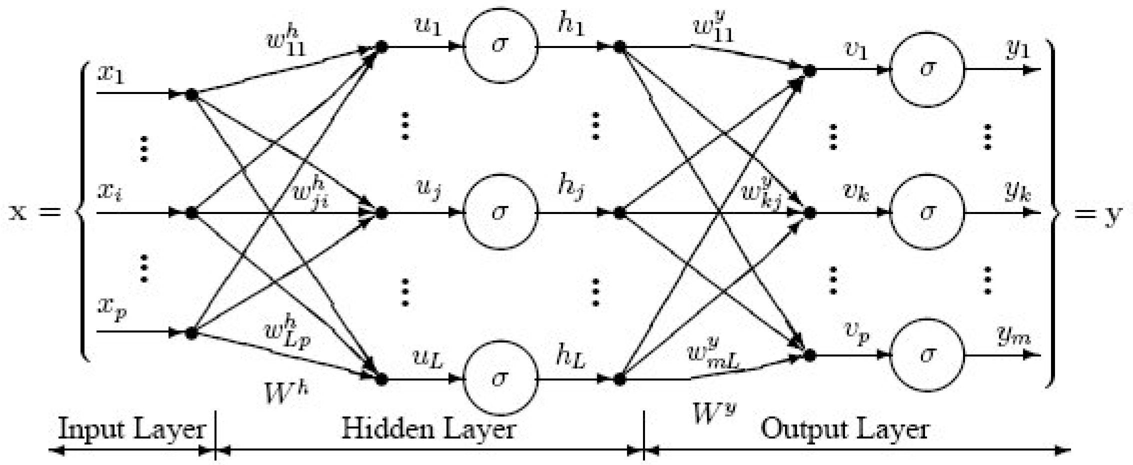


Figure 1. Multilayer Perceptron

3.2. Restricted Boltzmann Machine

We failed with MLP due to its impracticable huge weight matrix, so we tend to try a neural net with smaller weights. We thus decided to test Restricted Boltzmann Machine (RBM)[19]. RBM is a generative stochastic artificial neural network that can learn a probability distribution over the set of input. RBM only contains two layers: visible layer and hidden layer. The structure of RBM is shown in Figure 2. This structure greatly reduced the complexity of weight matrix, which would be more convenient for initialization and update of the weight matrix. The processing time would decrease.

However, RBM is normally adopted in unsupervised learning. It is not appropriate for our task. The energy function of RBM reduces slowly in our experiment while the program can hardly get converge in reasonable time.

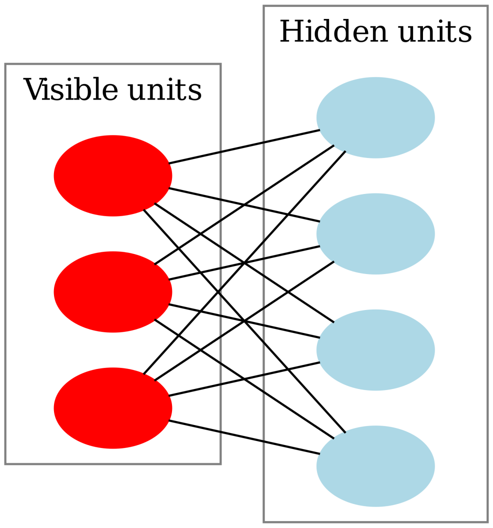


Figure 2. Restricted Boltzmann Machine

3.3. Logistic Regression

After we failed with two neural networks, we found neural networks may be not suitable for this dataset with a large input size. We have two ways left. The method should either perform feature selection itself or be abled to handle large input size. So we tried Logistic Regression. The reason we choose Logistic Regression is that its P-values can be used to perform a backward feature selection [12, 20].

If we use gradient descent in Logistic Regression, it still requires too much time to converge. Instead, we try to use Newton's gradient descent, in which we need to construct a Hessian matrix, and compute the reverse of it. However, the Hessian matrix in this case is similar to the weight matrix in neural nets, which contains millions of floating numbers. It is numerically unstable and time-consuming to calculate its reverse.

4. Good Approach

After we failed with Logistic Regression, finding algorithms that capable for high dimensional inputs is the only way left. Then we tried four algorithms satisfy this demand.

4.1. K-Nearest Neighbors (KNN)

KNN[21] is one of the typical algorithms in machine learning, which represents a simple analyze procedure in human brains. It evaluates the distance of each training input and captures the nearest k neighbors to compute the desire output according to these neighbors (Figure 3). It is easy to implement, and simple to understand. The calculation process is also simple. The algorithm only computes the distance and then sorts to find nearest neighbors to decide the class. Therefore, KNN are capable for any input size.

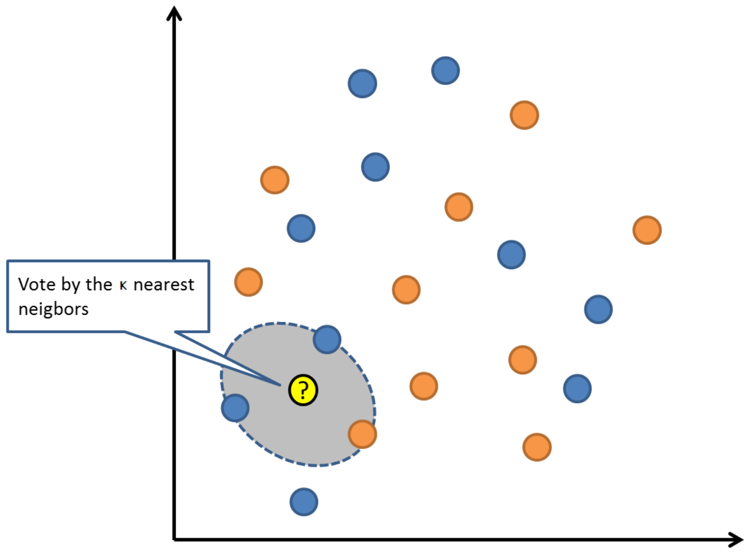


Figure 3. K-Nearest Neighbors

4.2. Support vector machine (SVM)

Support Vector Network[22, 23], also refers to Support Vector Machine (SVM) is widely used in pattern recognition, such as handwriting recognition, speaker identification, charmed quark detection, and face detection. It performs well in binary classification task, so we decide to test it with our dataset.

For a binary classification problem $y[-1,1]$, the basic idea of SVM is to find a max-margin separating hyperplane (Figure 4 and Figure 5).

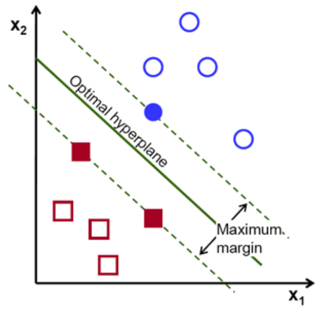


Figure 4. Support Vector Machine

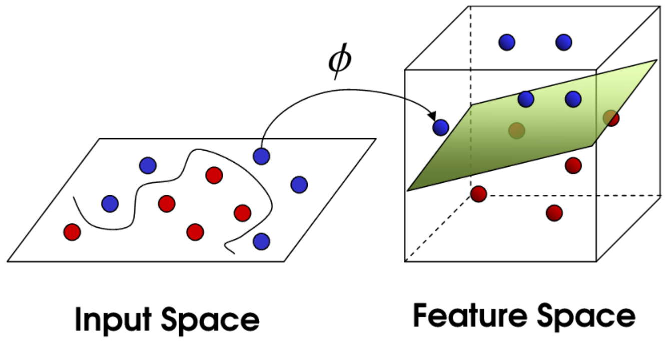


Figure 5. Hyperplane in SVM

A kernel is a similarity function, i.e., distance measurement which estimates the distance between two inputs. Kernel methods enable SVM to operate in higher-dimensional, implicit feature space without ever computing the coordinates of the data in that space[24].

The reason to use kernel is that in many cases, computing the kernel itself is easy, but computing the feature vector corresponding to the kernel is very hard. For example, the corresponding feature vector for the Gaussian RBF kernel[25] is infinite dimensional. Yet, computing the kernel is almost trivial.

$$Gaussian\ RBF\ Kernel : K(x,x') = exp(\gamma||x-x'||_2^2)$$

4.3. Naive Bayes

In machine learning, Bayes classifiers[26, 27] are a family of simple probabilistic classifiers based on Bayes' theory (Figure 6) with strong (naive) independence assumptions between features.

Bayes' theory calculates the probability for each hypothesis, and then gives the most possible prediction. It gives the prediction based on statistical data, which acts as a weak classifier, but computes efficiently. Also, since the final outputs are based on statistic analyzing, missing values would not affect them. These features make a Bayes classifier quite suitable for our task. We choose a simple representation of Bayes classifiers, Naive Bayes[28].

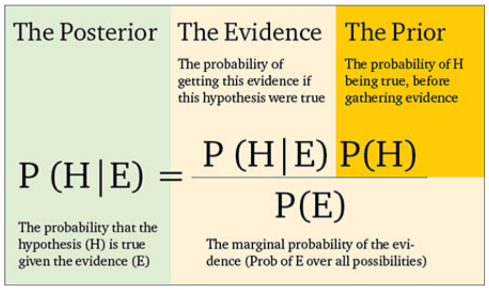

The Posterior
The Evidence
The probability of getting this evidence if this hypothesis were true
The Prior
The probability of H being true, before gathering evidence
P (H|E) = P (H|E) P(H) / P(E)
The probability that the hypothesis (H) is true given the evidence (E)
The marginal probability of the evidence (Prob of E over all possibilities)

Figure 6. Bayes' Theory

4.4. Decision Tree

Since we have so many qualitative values, tree-based method (Figure 7) is also considered [13, 29, 30], because qualitative values are easily handled in decision trees (no need of dummy variables).

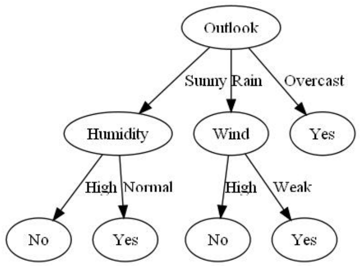


Figure 7. Simple Decision Tree

C4.5 algorithm[31] has been performed to generate the decision tree. C4.5 is an extension of ID3 algorithm. It uses a concept of information gain instead of information entropy in ID3, which can be used to decide a preferred value of the features most rapidly narrow down the state of this feature.

Pseudocode for C4.5 algorithm: C4.5_Build_Tree(T)

```
Step 1   Evaluate Information Gain over all attributes $A_1, A_2, A_3, ..., A_n$, and select best spliting attribute $B$.
Step 2   If $B$ is categorical with $b$ values, partition $T$ into $b$ nodes.
           Else compute the exact threshold value and partition $T$ into $T_1, T_2$.
Step 3   For all nodes $T_i$,
           If $T_i$ satisfies the homogeneity criterion, $T_i$ is a leaf node; assign it a class label.
             Else C4.5_Build_Tree($T_i$)
```

5. Final Results

We analyze previous four algorithms which are KNN, SVM, Bayes and Decision Tree with different evaluation scheme. For each of the algorithms, three kinds of data are being adopted. First one is the raw data without preprocessing. Data after imputation, but without feature selection come next. Data gone through all preprocessing technique, including feature selection have been tried at the end. For feature selection, three criteria discussed before which are Gini Index, Cross Entropy and Complex, are used. During data imputation process, two threshold values (0.05 and 0.95) are tested in listwise deletion (sample deletion and feature deletion). Detail results would be present and discussed below.

5.1. Final results of KNN method

Among different groups of the input data we get after the preprocessing process, we do the training and testing using 4-fold cross validation. We use the ROC curves to illustrate the trend. For each of the k values (12 values including 5, 7, 9, 11, 13, 17, 21, 25, 31, 45, 51, 61), we plot one curve. The threshold is ranged from 0 to 1 with a step of 0.05, so we have 21 points in one curve.

The Figure 8 is the ROC curve of the data set which is not applied to any preprocessing. As we can see, , the algorithm performs well and the curve is near the topleft corner of the coordinate axis.

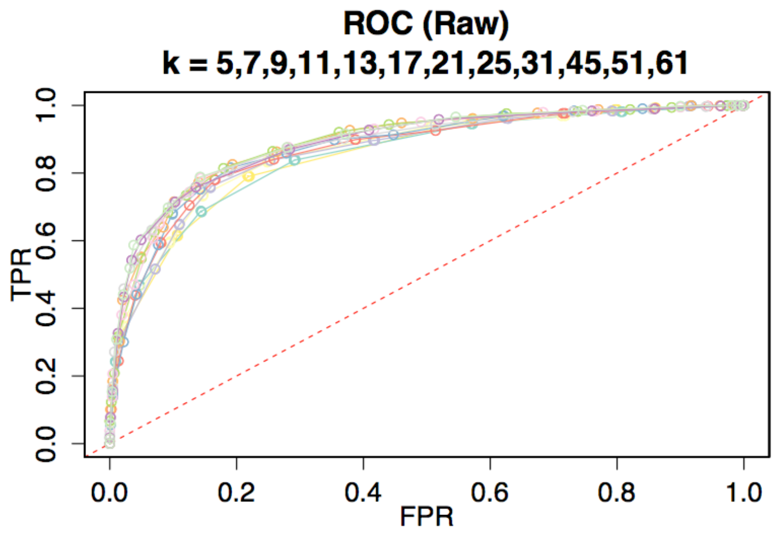


Figure 8. Result for Raw Data with different K

The results of the data without feature selection with two different threshold (0.05, 0.95) can be seen in Figure 9 and Figure 10. The result is as good as the previous one. Meanwhile, with threshold value 0.95 performs better than threshold value 0.05.

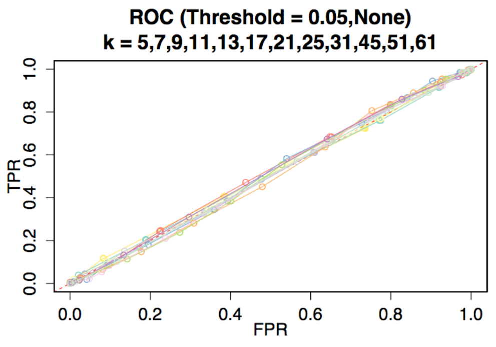


Figure 9. Result after Data imputation

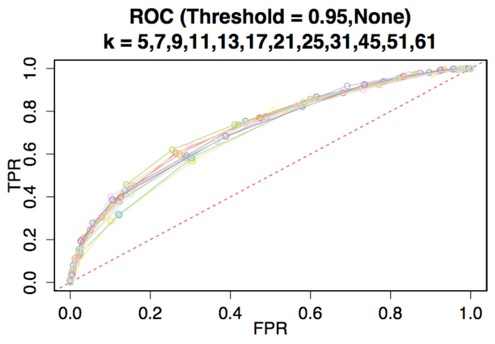


Figure 10. Result after Data imputation

Result of the Gini Index and Cross Entropy criteria are measured in Figure 11, 12 and Figure 13, 14.

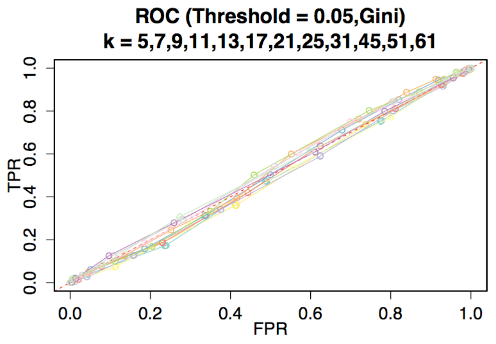


Figure 11. Result after Gini Index Selection

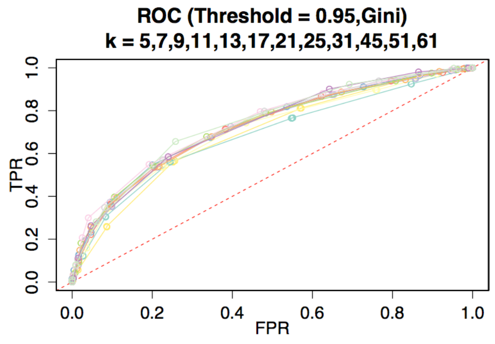


Figure 12. Result after Gini Index Selection

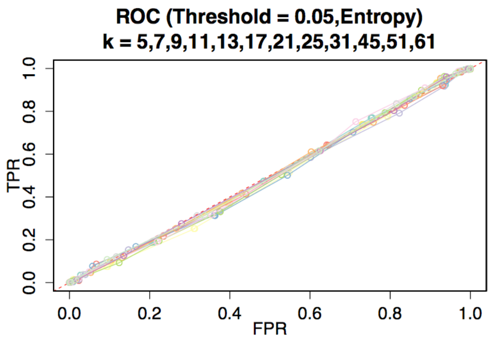


Figure 13. Result after Cross Entropy Selection

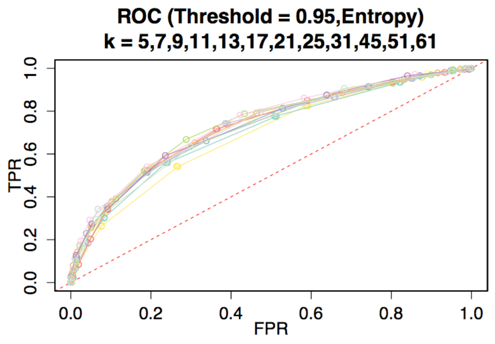


Figure 14. Result after Cross Entropy Selection

At last, ROC curves of the Complex criterion are illustrated in Figure 15 and 16. Although the performance under the threshold value 0.05 is still as discouraging as previous circumstances, the result of threshold value 0.95 is good and comparable to the performance under the raw data.

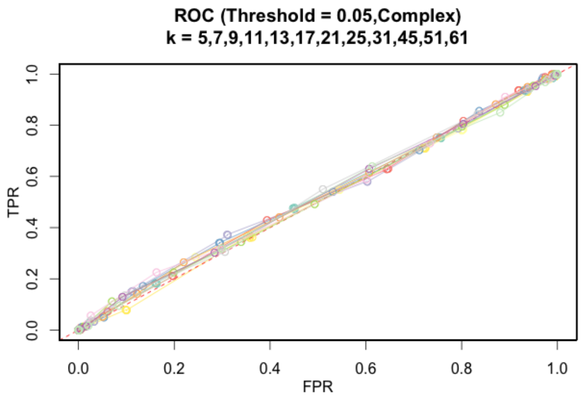


Figure 15. Result after Complex Selection

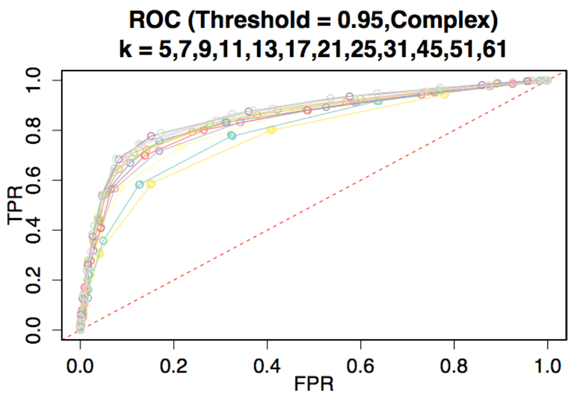


Figure 16. Result after Complex Selection

We also compare the results of different data with the same K value. We apply this comparison with one sample K value with high accuracy. The K value here is 31. As threshold value 0.05 just create almost random classifier. We compare those results under threshold value 0.95. In Figure 17, the algorithm perform better with raw data (NP, stands for No Preprocessing) and Complex selection method (Com). The Gini Index selection (G) and Cross Entropy Selection (E) are not helpful for the data after data imputation (NF, stands for No Feature Selection).

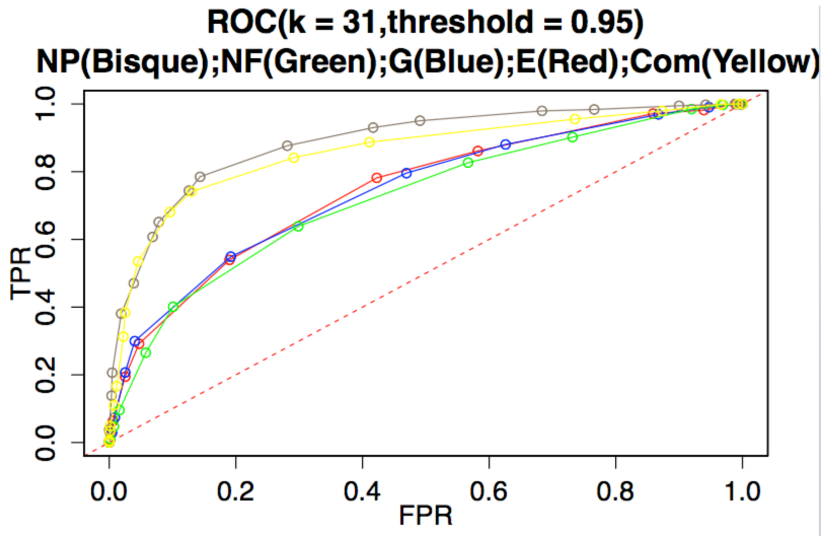
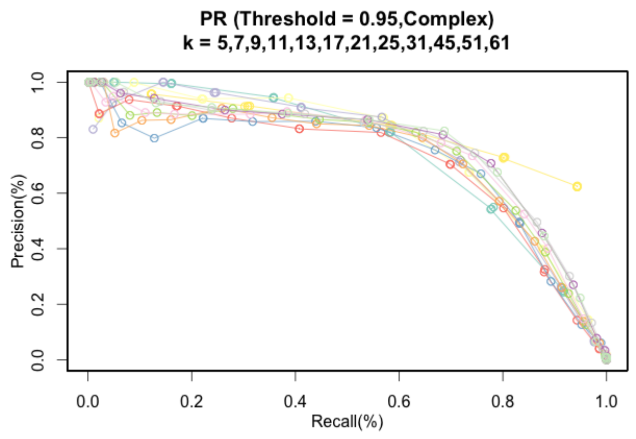


Figure 17. Comparison with same K (K=31)

Also, since the dataset is imbalance between two classes, the ROC may give an optimistic view of performance. Therefore, Precision-Recall(PR) Curve is plotted for two results in Figure 18 and 19.



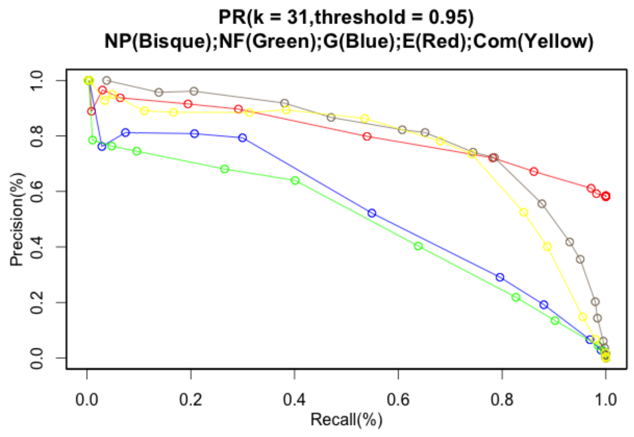
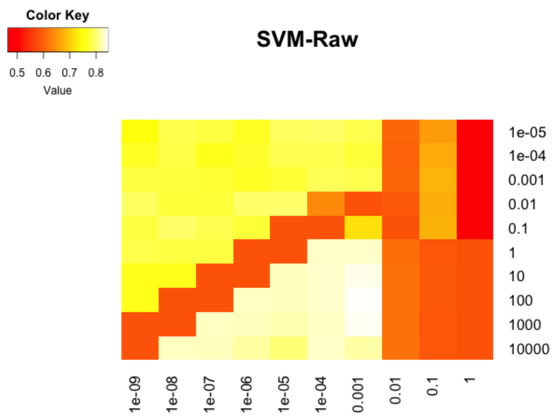


Figure 18. Result after Complex Selection **Figure 19. Comparison with same K (K=31)**

5.2. Final results of SVM method

We illustrates the result of SVM method using heat maps where the horizontal value represents parameter Gamma and the vertical one represents parameter Complexity. When the raw data is used directly, the result is better than others with threshold value 0.05. As can be seen, good results lie in the area where parameter c and gamma are both in the "middle" region of their scale. When the threshold is 0.95, the performance becomes quite good compare to previous cases, especially for the Complex method.



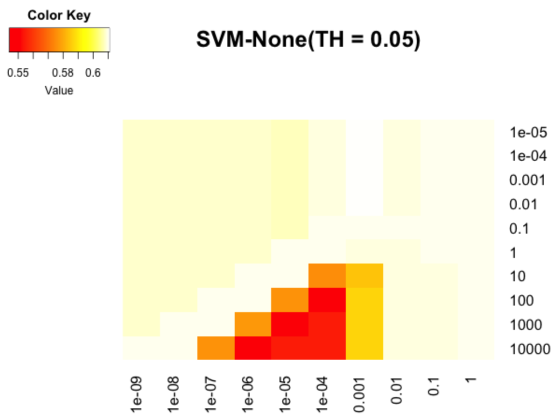


Figure 20. Result for Raw Data **Figure 21. No Selection with Threshold=0.05**

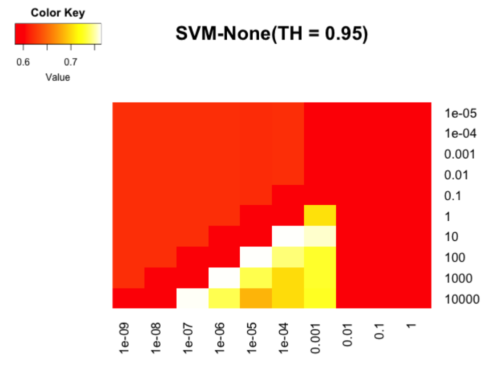


Figure 22. No Selection with Threshold=0.95

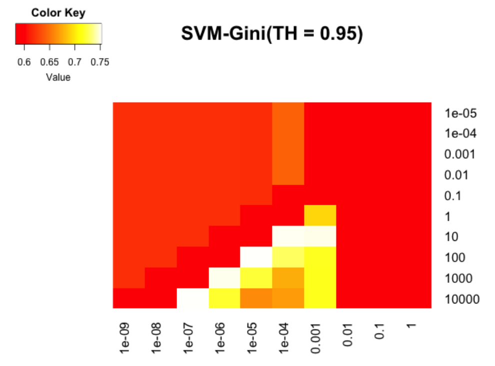


Figure 23. Gini with Threshold=0.95

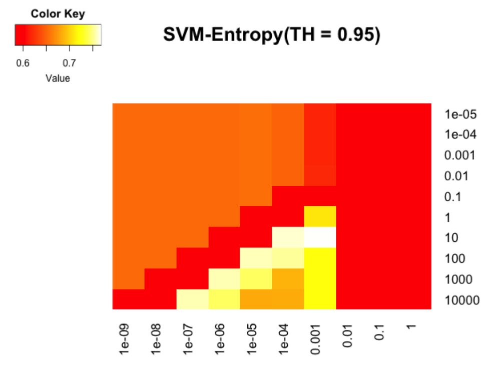


Figure 24. Entropy with Threshold=0.95

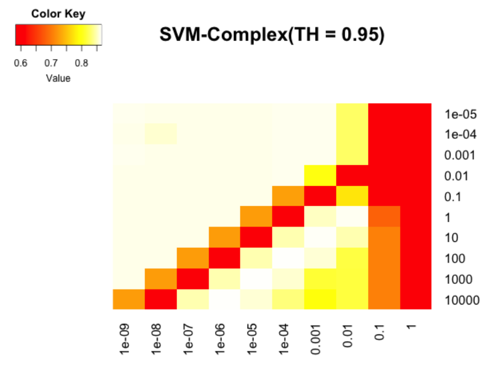


Figure 25. Complex with Threshold=0.95

In order to compare the results for different data processing technique, we also compare the best result for each one (Figure 26). We can see from the diagram below that the Complex selection technique contributes the best result.

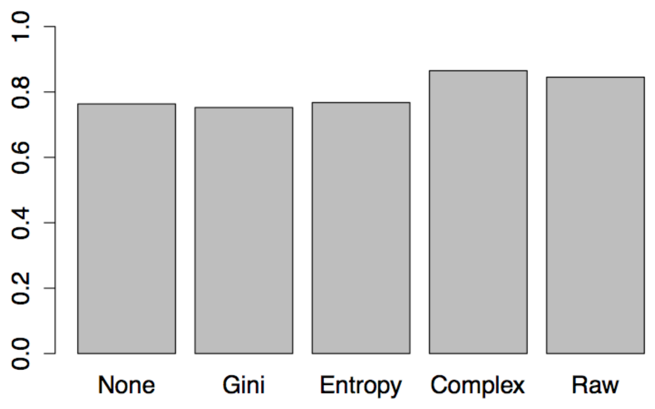


Figure 26. Comparison between different data processing technique

In order to compare with KNN, ROC and PRC of particular Gamma and C(omplexity) are plotted in Figure 27 and 28.

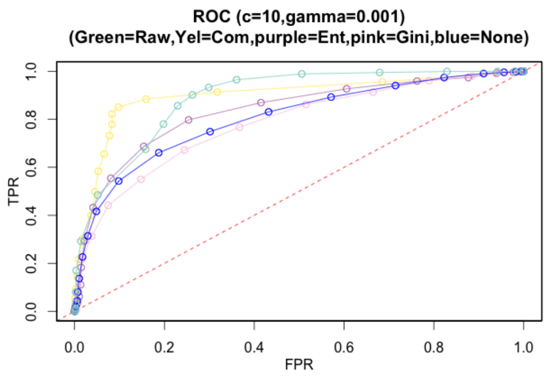


Figure 27. ROC for Gamma=0.001, C=10

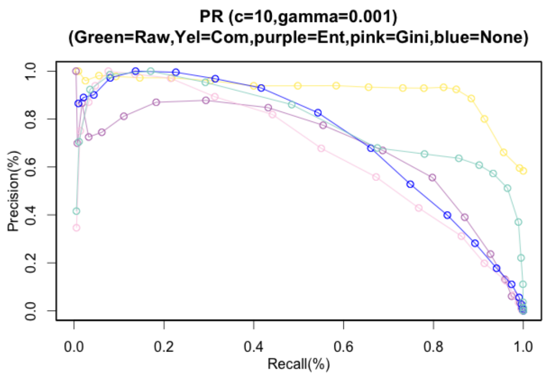
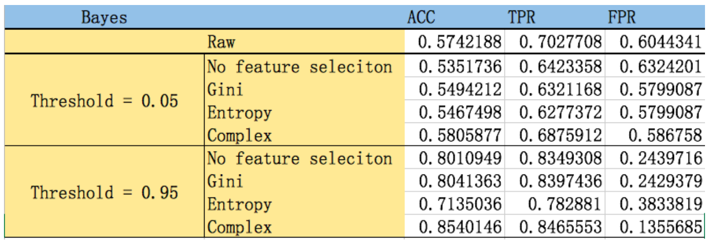


Figure 28. PRC for Gamma=0.001, C=10

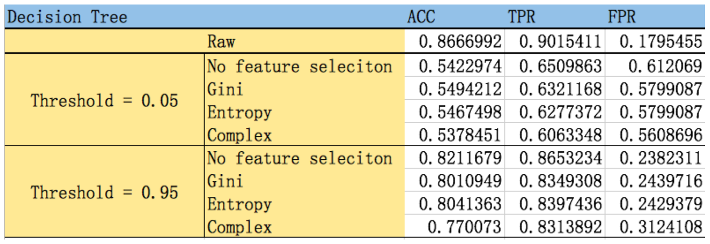
5.3. Final Results-Naive Bayes and Decision Tree

For these two methods, there is no threshold or parameter settings for us to control the algorithm. We can only conclude the Accuracy, TPR and FPR under the different data processing techniques for the results. Naive Bayes results is illustrated in Figure 29. Figure 30 shows the result of Decision Tree.



Bayes		ACC	TPR	FPR
	Raw	0.5742188	0.7027708	0.6044341
Threshold = 0.05	No feature seleciton	0.5351736	0.6423358	0.6324201
	Gini	0.5494212	0.6321168	0.5799087
	Entropy	0.5467498	0.6277372	0.5799087
	Complex	0.5805877	0.6875912	0.586758
Threshold = 0.95	No feature seleciton	0.8010949	0.8349308	0.2439716
	Gini	0.8041363	0.8397436	0.2429379
	Entropy	0.7135036	0.782881	0.3833819
	Complex	0.8540146	0.8465553	0.1355685

Figure 29. Result for Naive Bayes



Decision Tree		ACC	TPR	FPR
	Raw	0.8666992	0.9015411	0.1795455
Threshold = 0.05	No feature seleciton	0.5422974	0.6509863	0.612069
	Gini	0.5494212	0.6321168	0.5799087
	Entropy	0.5467498	0.6277372	0.5799087
	Complex	0.5378451	0.6063348	0.5608696
Threshold = 0.95	No feature seleciton	0.8211679	0.8653234	0.2382311
	Gini	0.8010949	0.8349308	0.2439716
	Entropy	0.8041363	0.8397436	0.2429379
	Complex	0.770073	0.8313892	0.3124108

Figure 30. Result for Decision Tree

5.4. Comparison between four algorithms

The four algorithms are successfully implemented in our project. In order to compare the results of these four methods, we should choose a representative of each algorithm to perform the comparative analysis. As can be seen from the previous results, the performance of threshold value 0.95 is way better than the result of threshold value 0.05. Complex selection is considered to be most powerful feature selection criteria among three. Also, all these algorithms can handle raw data well, we should also compare the raw result among them.

Accuracy and computation time are compared in Figure 31 and 32.

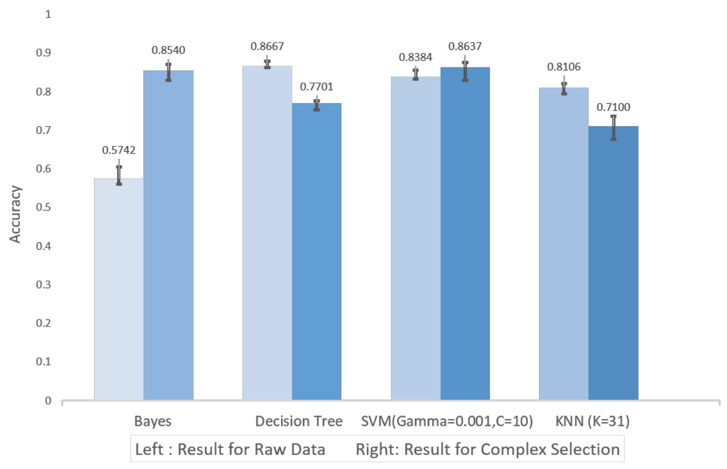


Figure 31. Accuracy Comparison between 4 algorithms

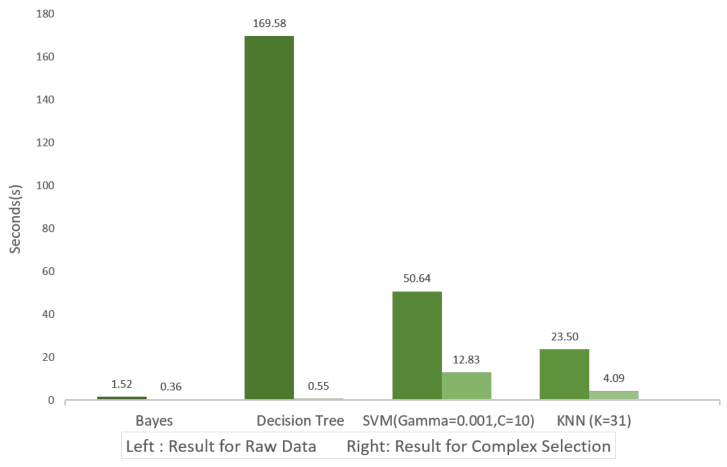


Figure 32. Computation Time Comparison between 4 algorithms

6. Conclusion

We can observe from previous tables and figures that for most algorithms, except Naive Bayes, the result become worse after preprocessing, which leads to a puzzle that it seems that we should leave the raw data without any optimization techniques.

The possible explanation for this result may lie in the dataset and the method we use.

- The dataset has some intrinsic meaning which may diminish or warp after some preprocessing. The aim of preprocessing is to better explain and represent the element of the data to a lower dimensional input. It suggests us that SNP data might actually have numerical meaning instead of categorical aspect.
- SVM, Bayes and Decision Tree are algorithms that capable of handling missing components. Therefore preprocessing may have little effect.
- During preprocessing progress, we abandon parts of the data because of the missing feature exceeds a certain threshold. However, those abandoned data may have important feature information. The facts that higher threshold (0.95) contributes to a better performance actually suggests that we are ignoring too much data.
- Complex selection criteria performs well, and increase the performance after preprocessing, it suggests that the problem is the choice of feature selection technique. Gini Index and Cross Entropy might not be a suitable solution for our task.

7. Future Work

There are still works left in our project, we could try these directions in further research.

- Treat SNP as numerical feature and observe the performance of the algorithms under this assumption.
- Try different imputation strategies, and investigate the effect.
- Use different deletion threshold to experiment.
- Adopt Complex selection criteria for different data set to test its performance.

References

[1] J. Marchini and B. Howie, "Genotype imputation for genome-wide association studies," *Nature Reviews Genetics*, vol. 11, no. 7, pp. 499–511, 2010.

[2] B. Howie, J. Marchini, and M. Stephens, "Genotype imputation with thousands of genomes," *G3: Genes, Genomes, Genetics*, vol. 1, no. 6, pp. 457–470, 2011.

[3] F. E. Grubbs, "Procedures for detecting outlying observations in samples," *Technometrics*, vol. 11, no. 1, pp. 1–21, 1969.

[4] S. H. Walker and D. B. Duncan, "Estimation of the probability of an event as a function of several independent variables," *Biometrika*, vol. 54, no. 1-2, pp. 167–179, 1967.

[5] L. Györfi, L. Devroye, and G. Lugosi, "A probabilistic theory of pattern recognition," 1996.

[6] D. B. Rubin, *Multiple imputation for nonresponse in surveys*. John Wiley & Sons, 2004, vol. 81.

[7] ——, "Inference and missing data," *Biometrika*, vol. 63, no. 3, pp. 581–592, 1976.

[8] W. Young, G. Weckman, and W. Holland, "A survey of methodologies for the treatment of missing values within datasets: limitations and benefits," *Theoretical Issues in Ergonomics Science*, vol. 12, no. 1, pp. 15–43, 2011.

[9] O. Dekel, O. Shamir, and L. Xiao, "Learning to classify with missing and corrupted features," *Machine learning*, vol. 81, no. 2, pp. 149–178, 2010.

[10] M. Saar-Tsechansky and F. Provost, "Handling missing values when applying classification models," 2007.

[11] B. M. Marlin, "Missing data problems in machine learning," Ph.D. dissertation, University of Toronto, 2008.

[12] D. Williams, X. Liao, Y. Xue, and L. Carin, "Incomplete-data classification using logistic regression," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 972–979.

[13] Y. Ding and J. S. Simonoff, "An investigation of missing data methods for classification trees applied to binary response data," *The Journal of Machine Learning Research*, vol. 11, pp. 131–170, 2010.

[14] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[15] F. Rosenblatt, *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[16] M. Minsky and S. Papert, "Perceptron: an introduction to computational geometry," *The MIT Press, Cambridge, expanded edition*, vol. 19, no. 88, p. 2, 1969.

[17] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural Networks, 1989. IJCNN., International Joint Conference on*. IEEE, 1989, pp. 593–605.

[18] M. W. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)a review of applications in the atmo-

spheric sciences," *Atmospheric environment*, vol. 32, no. 14, pp. 2627–2636, 1998.

[19] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 791–798.

[20] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 215–242, 1958.

[21] K. Fukunaga and P. M. Narendra, "A branch and bound algorithm for computing k-nearest neighbors," *Computers, IEEE Transactions on*, vol. 100, no. 7, pp. 750–753, 1975.

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[23] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.

[24] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[25] B. Schölkopf, K.-K. Sung, C. J. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik, "Comparing support vector machines with gaussian kernels to radial basis function classifiers," *Signal Processing, IEEE Transactions on*, vol. 45, no. 11, pp. 2758–2765, 1997.

[26] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[27] N. Friedman and R. Kohavi, "Data mining tasks and methods: classification: Bayesian classification," in *Handbook of data mining and knowledge discovery*. Oxford University Press, Inc., 2002, pp. 282–288.

[28] I. Rish, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22. IBM New York, 2001, pp. 41–46.

[29] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[30] ——, "Simplifying decision trees," *International journal of man-machine studies*, vol. 27, no. 3, pp. 221–234, 1987.

[31] ——, *C4. 5: programs for machine learning*. Elsevier, 2014.

Appendix

A. Feature Ranking Result

In Complex selection method with deletion threshold 0.95, we obtain a list of 144 features. These features are considered to be important for our task. Four algorithms perform relatively well with these features. Here is a list of them:

sex, smkstat, LPL_09, RERG_24, GSK3B_23, MGC29463_01, APOE_03, OGG1_12, IRS1_03, LDLR_08, NFKB1_33, SLC6A3_03, MASP1_54, GDF15_01, DRD4_15, PARP1_12, ERCC2_09, SCARB1_08, STK11_03, MBL2_65, CYP2C19_03, CYP19A1_15, SAT2_03, GHR_25, CAV1_19, MYBL2_06, ALOX12_02, AKR1C3_28, TNFRSF6_04, asa_rs2273535, asa_rs5601842, rs11191454, rs1800067, rs1801280, rs2020897, rs2020914, rs2228332, rs33923703, rs34625968, rs3740393, rs3862792, rs45455492, rs7386783, asa_rs2230743, xrcc1_632, xrcc1_26602, MC1R_142, MC1R_294, MC1R_4161, rs12801239, rs4593053, PTCH_1315, xrcc1_1678, ERCC4_2, gstp1_114, rs7178, xrcc1_194, ERCC4_1, XPA_6, xpc_pat, KRAS_13, CARD15_19, rs3740391, BCL2L1_03, COMT_03, CTNNB1_02, ALOX5_06, VDR_12, xpd_312, tp53_01, PLA2G6_10, CRP_02, XRCC5_19, TP73L_46, RAD51_17, BIRC3_02, rs1052571, BLM_05, IL15RA_04, rs938886, INSR_61, NEDD5_01, BRCA1_05, TLR2_05, CYP19A1_27, MTRR_22, CASP9_27, ARHGDIB_03, AMT_06b, rs34987347, LIPC_17, CYP7B1_03, IGF1_27, INSR_59, CASP8_06, rs1799929, MASP1_46, TNKS_22, GSK3B_38, rs45483697, TP73L_15, IGF1R_27b, rs1867380, rs35717727, ERCC5_02b, IGF1_22, IGFBP6_11, ALOX5_10, MSH2_03, SRA1_03, ENPP1_04, CBS_01, FTHFD_06, TNFRSF5_03, FZD7_06, IGFBP3_04, TYMS_01, SLAMF1_04, rs11191439, rs12507582, rs156697, rs2070676, rs2234631, rs2234636, rs2572023, rs3218076, rs640573, rs7085854, rs45520831, SLC23A2_05, SLC19A1_01, KRT23_04, MASP1_43, rs2020955, ERCC3_02, RB1CC1_10, IL15_01, TEP1_11, rs11509439, rs1695, rs2289965, FANCA_16, CFH_05, APC_19