**Assignment 1 – Group 19**

**Song Yang [12245771]; Jiamian Liu [12270784]; Xiaoyu Yang[12477052]**

**Task1**

**1.A.1 Pre-processing**

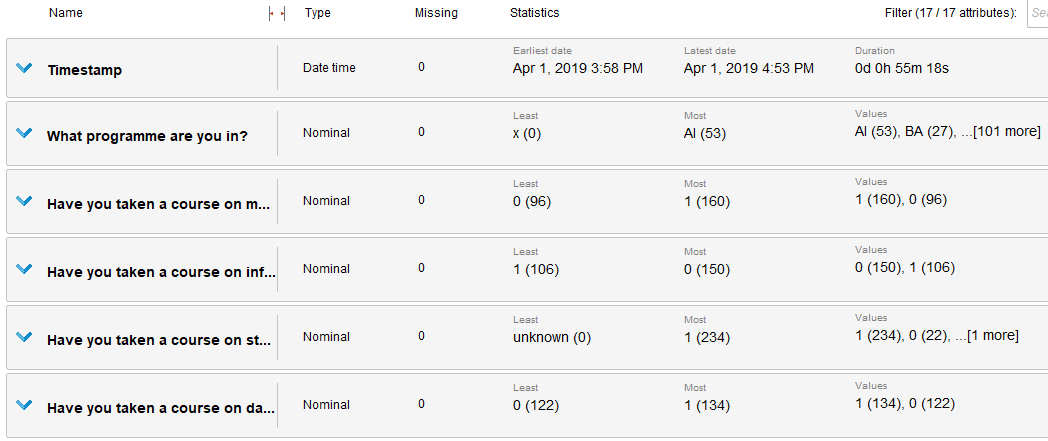
Firstly, we should understand the task question and data. The task is that we need to explore the dataset we got from the course, which include many different kind of information of student.

Secondly, we need to understand the data. The original data have 276 rows and 17 columns which means there are 17 features and 276 samples. Moreover, there are 12 columns are belonged to category, 1 for data/time, 4 for number.

Thirdly, we clean the data, because there are a lot of non-meaningful value for the data. For example, we the replace the missing stress level to the most frequent value. Also, we use unified symbol to represent ‘yes’ and ‘no’ which can help us built find the correlation between different variables. Therefore, we replace ‘ja’ and ‘yes’ to 1, ‘nee’ and ‘no’ to 0. And, we remove the rows of ‘unknown’ in ‘gender’. etc.

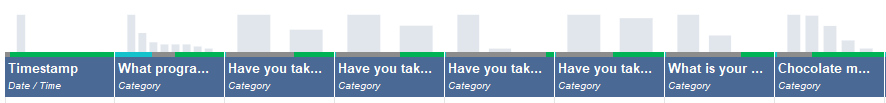
**1.A.2 Statistics and Correlations**

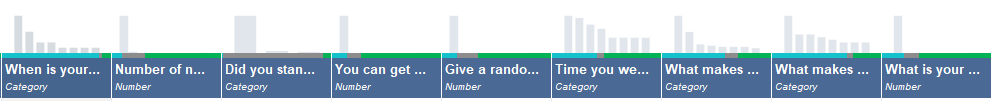
Statistics: Firstly, we can get some baisc statistics information of the data. For example, as the figure 1 shows, we can find ‘AI student’ account the most percentage of student.



**Figure 1. part of stattistics information of datat**

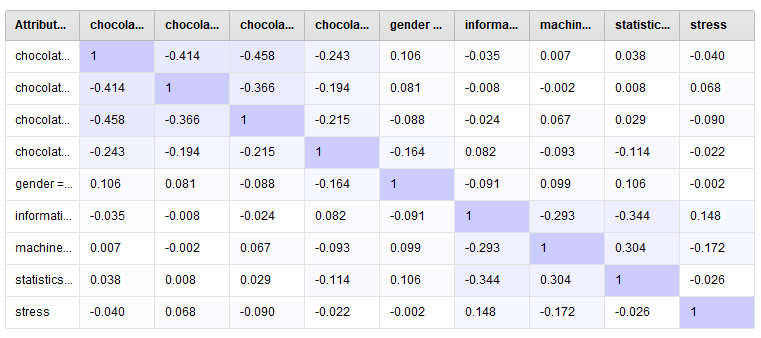
After pre-processing the data, we can get a overview of the distribution of each columns. As figure 2 shows, we can see “What makes a good day for you” has too many categories, which is not useful for our model, so we can remove that columns.



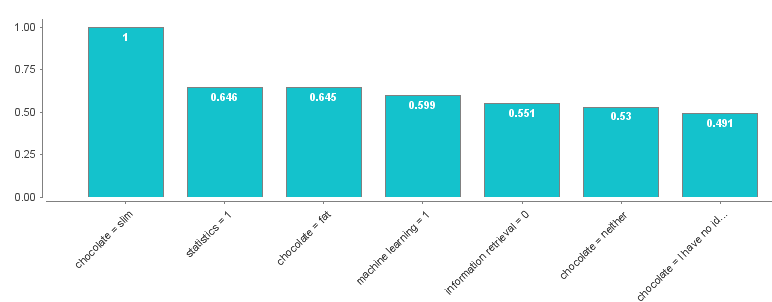


**Figure 2. Distribution of all columns**

Correlations: We find some correlations between “machine learning”, “information retrieval”, “statistics”, “chocolate” and “stress” to predict the “gender”. We removed some independent variables, such as “timestamp” and “random number”. As figure 3 and figure 4 show, “chocolate=slim” is the most negative relevant attribute to the gender.

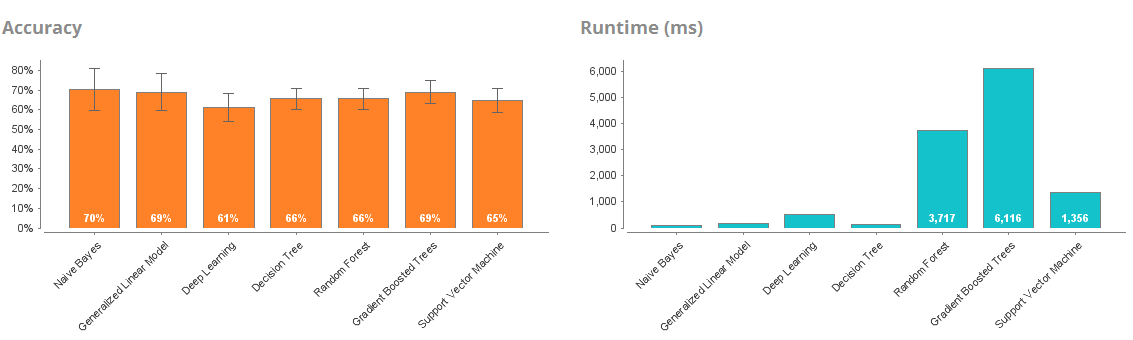


**Figure 3. Correlation Matrix**



**Figure 4. weight of attribute**

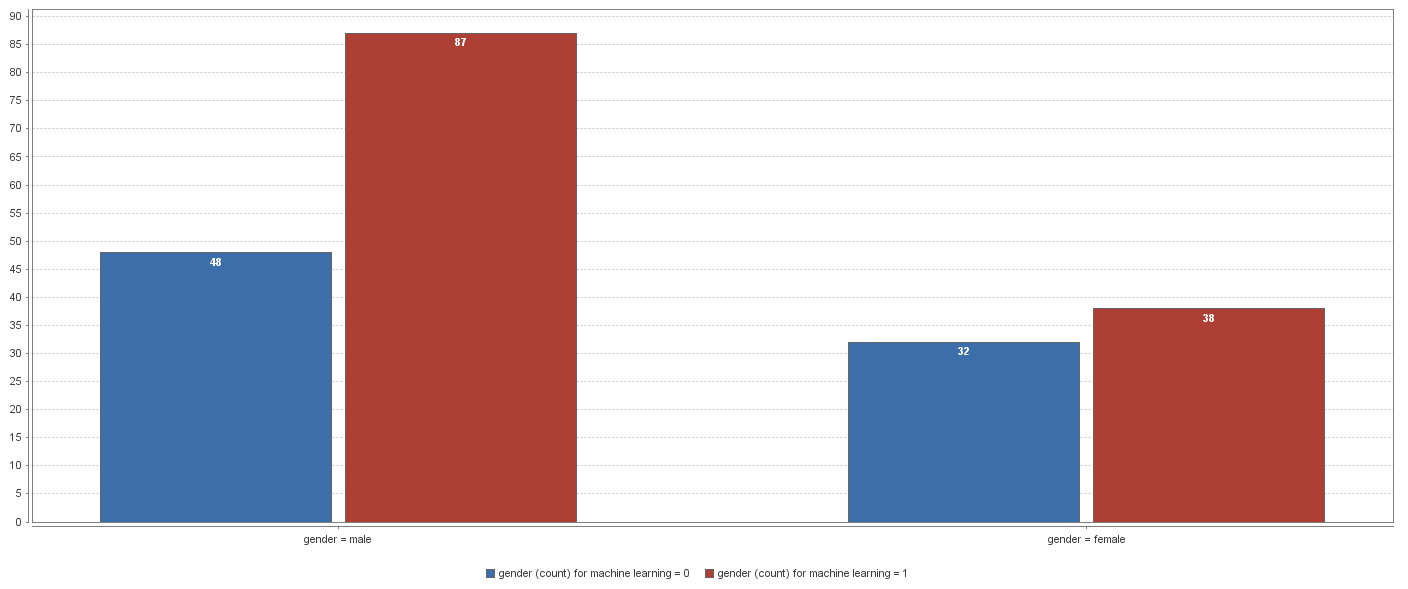
Then we try to use some algorithms to predict the gender. It is a classification problem, because gender only has tree categories. As figure 5 shows, naïve Bayes has the highest accuracy (70%) and shortest time. Gradient boosted trees also get good accuracy, but it takes much more time. The reason may is the dataset is simple and using naïve Bayes is enough.



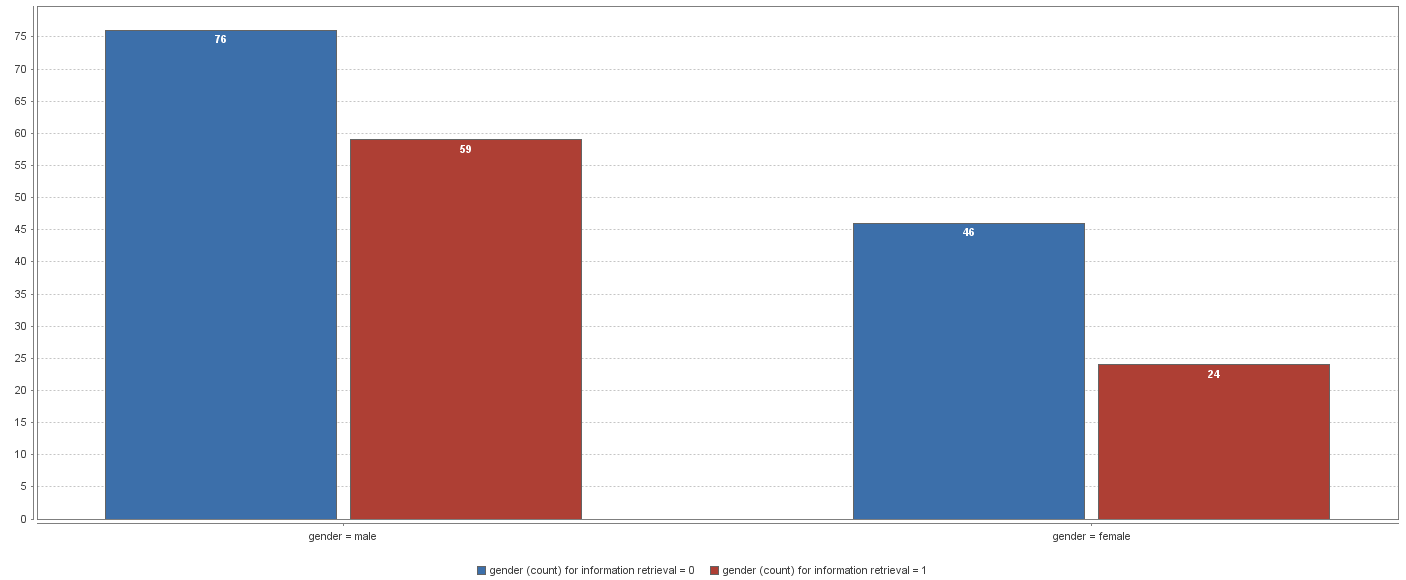
**Figure 5. Different Algorithms**

**1.A.3 Plots about the variables and resulting models**

As figure 6 and figure 7 show, we can see much more male students have attended the machine learning. Both most of male and female student have not attended information retrieval.



**Figure 6 machine learning of different genders (left-male, right-female, blue-not attend, red-attend)**



**Figure 7 information retrieval of different genders (left-male, right-female, blue-not attend, red-attend)**

As figure 8 shows, average female students’ stress level (38) is more than male students (34). Comparing with the above plots, the reason may be is that less female students have attended machine learning course than male students.



**Figure 6. stress level of different genders**

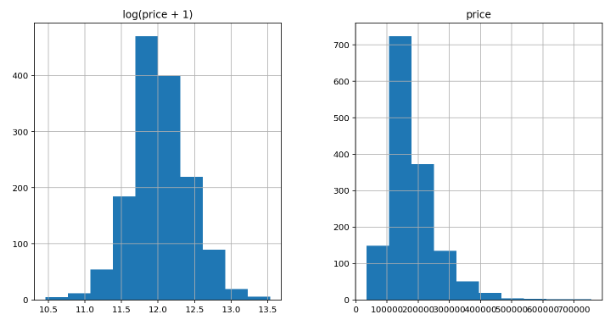
**1.B Basic classification/regression**

**1.B.1 Pre-processing**

I choose the “House Price” dataset from Kaggle, because it contains 79 explanatory variables describing almost every aspect of houses. It is a regression task, because the house price results within a continuous output.

Firstly, we need to understand the dataset. The house price dataset contains 5 rows × 81 columns. Then, we pre-process the dataset. We gave some value to the missing data, including putting ‘typical’ to ‘NA’ of “Heating QC". Also, we can combine some features because there are too may features to build a model. For example, we combine "OverallQual" and "OverallCond" to "OverallGrade".

Then, we try to find the most relevant features to house price. Therefore, we compute the correlation of features. The top 3 relevant features are SalePrice 1.000, OverallQual: 0.817, GrLivArea: 0.701. As figure 7 shows, we find the distribution of “SalePrice” is not normal, so we use the log transformation to get a normal numeric features.



**Figure 7. Distribution of Sale Price**

**1.B.2 Models**

The independent variables are “Fence: Fence quality” and “MiscFeature: Miscellaneous feature not covered in other categorie”, because they do not have any relevant with other variables. We choose two different linear regression algorithms to solve the regression problem.

1. **Linear Regression with Ridge regularization**

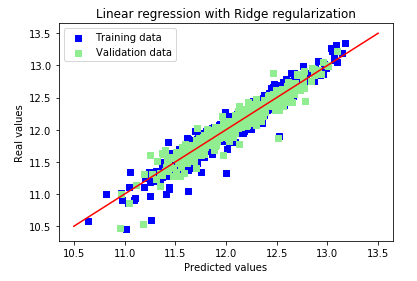
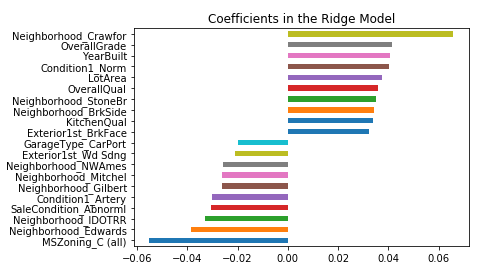
First, we use linear regression with ridge regularization, because regularization is a good way to filter noise and prevent overfitting. Regularization can introduce additional information to penalize extreme parameter weights. We add the squared sum of the weights to our cost function, and the params of ridge regularization shows as below.

ridge = RidgeCV(alphas = [0.01, 0.03, 0.06, 0.1, 0.3, 0.6, 1, 3, 6, 10, 30, 60])

ridge = RidgeCV(alphas = [alpha \* .6, alpha \* .65, alpha \* .7, alpha \* .75, alpha \* .8, alpha \* .85, alpha \* .9, alpha \* .95, alpha, alpha \* 1.05, alpha \* 1.1, alpha \* 1.15, alpha \* 1.25, alpha \* 1.3, alpha \* 1.35, alpha \* 1.4], cv = 10)

ridge.fit(X\_train, y\_train); y\_train\_rdg = ridge.predict(X\_train); y\_test\_rdg = ridge.predict(X\_test)

Ridge picked 316 features and eliminated the other 3 features. As figure 8 shows, the results of training and test are similar which means we eliminated most of the overfitting.

**Figure 8. predictions and important coefficients of Ridge regularization**

Then, we cross validate the model, and we can see the RMSE as below. We can see we get a good RMSE, that means our model can predict the house price.

Ridge RMSE on Training set: 0.11540572328450789

Ridge RMSE on Test set: 0.11642721377799559

**(**rmse= np.sqrt(-cross\_val\_score(model, X\_train, y\_train, scoring = scorer, cv = 10))**)**

1. **Linear Regression with Lasso regularization**

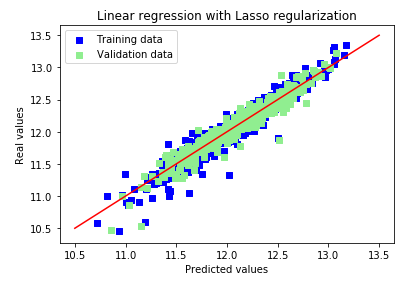
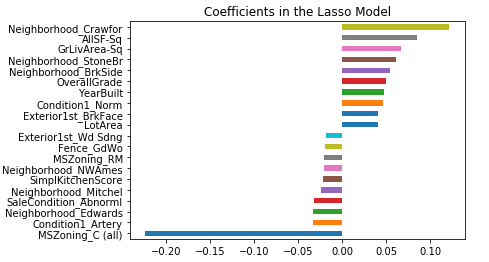
Then we choose linear Regression with Lasso regularization, because it is simply replace the square of the weights by the sum of the absolute value of the weights. We can see the params of Lasso regularization as below.

lasso = LassoCV(alphas = [0.0001, 0.0003, 0.0006, 0.001, 0.003, 0.006, 0.01, 0.03, 0.06, 0.1,

0.3, 0.6, 1], max\_iter = 50000, cv = 10)

lasso = LassoCV(alphas = [alpha \* .6, alpha \* .65, alpha \* .7, alpha \* .75, alpha \* .8, alpha \* .85, alpha \* .9, alpha \* .95, alpha, alpha \* 1.05, alpha \* 1.1, alpha \* 1.15, alpha \* 1.25, alpha \* 1.3, alpha \* 1.35, alpha \* 1.4], max\_iter = 50000, cv = 10)

Lasso picked 111 features and eliminated the other 208 features. As figure 9 shows, the results of training and test are also very similar. It gives big weights to Neighborhood categories. The potential reason is that house prices change a whole lot from one neighborhood to another in the same city.

**Figure 9. predictions and important coefficients of Lasso regularization**

Then, we cross validate the model, and we can see the RMSE as below. We can see we get a better RMSE, that means Lasso regularization model can predict the house price better than Ridge regularization.

Lasso RMSE on Training set: 0.11411150837458059

Lasso RMSE on Test set: 0.11583213221750707