Data Mining Assignment 1

Group 19: Xiaoyu Yang [2640948, xyg230], Jiamian Liu[2632301, jlu510], Song Yang[2638728, syg340]

1 Introduction

This report is about data mining assignment 1 with three tasks, including exploring a small dataset, predicting Titanic survival and theory. Each task follows by some basic data ming steps, such as preprocessing, feature engineering, training and evaluating the model.

2 Task 1

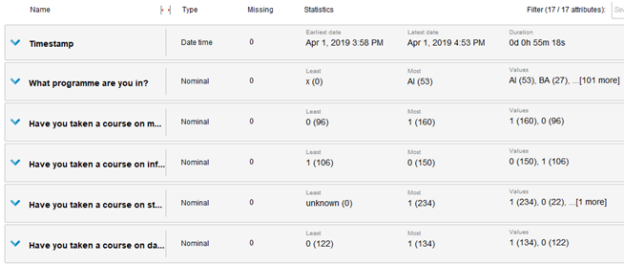
2.1 Task 1.A Exploration

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 includes many different kinds of information of students.

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 the number.

Thirdly, we clean the data, because there is a lot of non-meaningful value for the data. For example, we replace the missing stress level to the most frequent value. Also, we use a unified symbol to represent 'yes' and 'no' which can help us built to 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.

Statistics and Correlations. Statistics: Firstly, we can get some basic statistical information about the data. For example, as figure 1 shows, we can find 'AI student' account the most percentage of students.



Name	Type	Missing	Statistics		Filter (17 / 17 attributes):
Timestamp	Date time	0	Earliest date: Apr 1, 2019 3:58 PM	Latest date: Apr 1, 2019 4:53 PM	Duration: 0d 0h 55m 18s
What programme are you in?	Nominal	0	Least: x (0)	Most: AI (53)	Values: AI (53), BA (27), ... [101 more]
Have you taken a course on m...	Nominal	0	Least: 0 (96)	Most: 1 (160)	Values: 1 (160), 0 (96)
Have you taken a course on inf...	Nominal	0	Least: 1 (106)	Most: 0 (150)	Values: 0 (150), 1 (106)
Have you taken a course on st...	Nominal	0	Least: unknown (0)	Most: 1 (234)	Values: 1 (234), 0 (22), ... [1 more]
Have you taken a course on da...	Nominal	0	Least: 0 (122)	Most: 1 (134)	Values: 1 (134), 0 (122)

Figure 1. part of statistics information of data

After pre-processing the data, we can get an overview of the distribution of each column. 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 those columns.

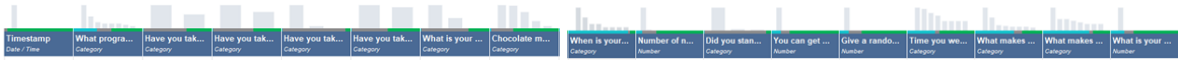


Figure 2. Distribution of all columns

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.

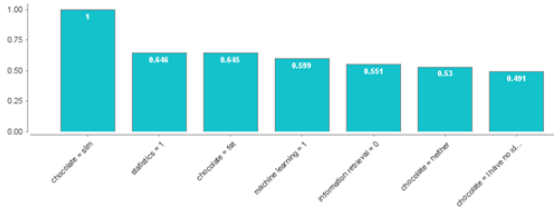


Figure 3. Weight of attributes

Attribut...	chocola...	chocola...	chocola...	chocola...	gender ...	informa...	machin...	statistic...	stress
chocolat...	1	-0.414	-0.458	-0.243	0.106	-0.035	0.007	0.038	-0.040
chocolat...	-0.414	1	-0.366	-0.194	0.081	-0.008	-0.002	0.008	0.068
chocolat...	-0.458	-0.366	1	-0.215	-0.088	-0.024	0.067	0.029	-0.090
chocolat...	-0.243	-0.194	-0.215	1	-0.164	0.082	-0.093	-0.114	-0.022
gender =...	0.106	0.081	-0.088	-0.164	1	-0.091	0.099	0.106	-0.002
informati...	-0.035	-0.008	-0.024	0.082	-0.091	1	-0.293	-0.344	0.148
machine...	0.007	-0.002	0.067	-0.093	0.099	-0.293	1	0.304	-0.172
statistics...	0.038	0.008	0.029	-0.114	0.106	-0.344	0.304	1	-0.026
stress	-0.040	0.068	-0.090	-0.022	-0.002	0.148	-0.172	-0.026	1

Figure 4. Correlation Matrix

Then we try to use some algorithms to predict the gender. It is a classification problem because gender only has three categories. As figure 5 shows, naïve Bayes has the highest accuracy (70%) and the 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.

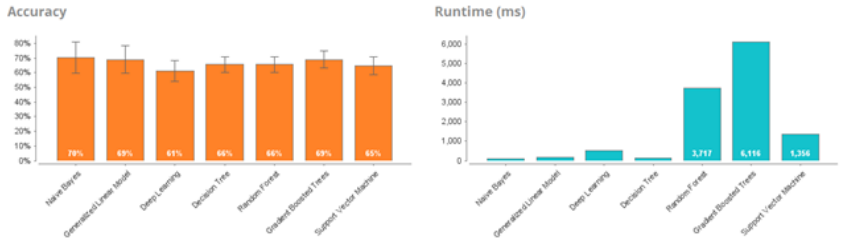


Figure 5. Different Algorithms

Plots about the variables and resulting models. As figure 6 shows, we can see much more male students have attended the machine learning. Both most of male and female student has not attended information retrieval.

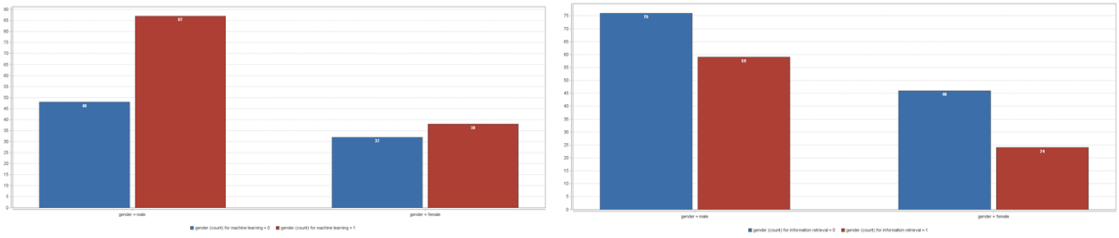


Figure.6 (Left) Machine learning of different genders (left-male, right-female, blue-not attend, red-attend) and (Right) Information retrieval of different genders (left-male, right-female, blue-not attend, red-attend)

As figure 7 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 a machine learning course than male students.

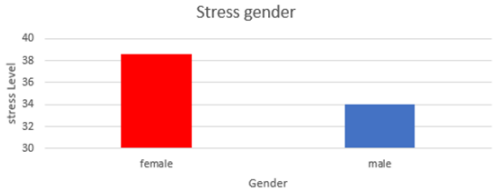


Figure 7. Stress level of different genders

2.2 Task 1.B Basic classification/regression

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 of 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 many feature 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 8 shows, we find the distribution of "SalePrice" is not normal, so we use the log transformation to get normal numeric features.

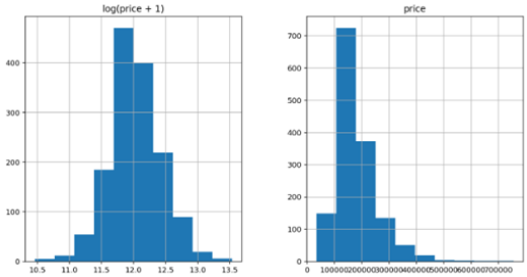


Figure 8. Distribution of Sale Price

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

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 9 shows, the results of training and test are similar which means we eliminated most of the overfitting.

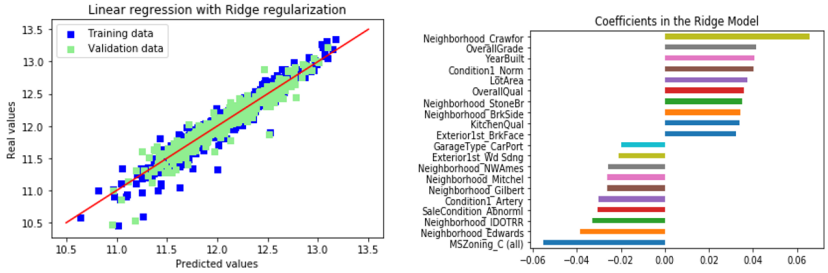


Figure 9. Predictions and important coefficients of Ridge regularization

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)))

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 10 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.

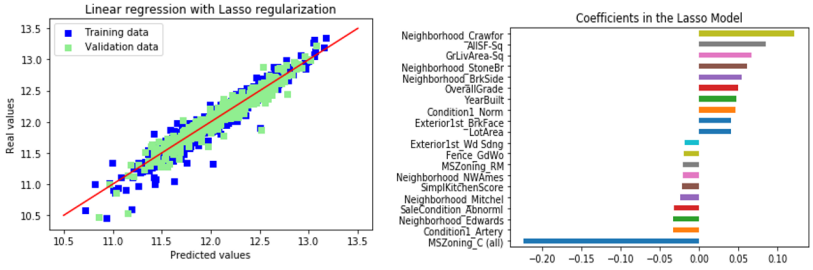


Figure 10. Predictions and important coefficients of Lasso regularization

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)

3 Task 2

In this task we came up with a model to predict whether someone survived the Titanic disaster or not based on a training set of already known people. We are mainly offered 2 datasets: train.csv as the training set for establishing survive model, test.csv as the testing set. The whole task is coded in python. Please refer to attachment task2_draw.py and task2_analyze.py for detailed information. N.B. The team name on Kaggle is VU-DM-group19 (because VU-DM-19 already exists).

3.1 Task 2.A Preparation

Explore data. After reading dataset, it is important for us to get the basic information. The basic information of training set is shown in left of Fig.11, and the basic information of testing set is in the right.

```
df_train.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

```
df_test.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
PassengerId    418 non-null int64
Pclass         418 non-null int64
Name           418 non-null object
Sex            418 non-null object
Age            332 non-null float64
SibSp          418 non-null int64
Parch          418 non-null int64
Ticket         418 non-null object
Fare           417 non-null float64
Cabin          91 non-null object
Embarked       418 non-null object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

Figure 11. Basic information of two dataset

After looking into the detailed information, we could know that there are 12 columns and 891 rows data in the train dataset. “Name”, “Sex”, “Ticket”, “Cabin” and “Embarked” are the type of object, and the other are integer or float which means may be directly calculated. Besides, there are missing values in “Age”, “Cabin”, “Fare” and “Embarked”. Especially, “Cabin” misses a lot. The descriptive statistical results are clearly shown in Figure 12.

```
df_train.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
df_train[['Name','Sex','Ticket','Cabin','Embarked']].describe()
```

	Name	Sex	Ticket	Cabin	Embarked
count	891	891	891	204	889
unique	891	2	681	147	3
top	Lovell, Mr. John Hall ("Henry")	male	CA. 2343	G6	S
freq	1	577	7	4	644

Figure 12. Descriptive Statistical Results

We could know the survive rate is only around 38.4%, which means a high death rate. And now we need to find out if the other 11 features for survive are significant or not. We could know that “PassengerId” is only to identify the identity, so it is an irrelevant feature. We first draw the correlation heat map to check if there are correlation > 0.5 as shown in Fig.13 left. We could know that there is obvious correlation between P class and survived. Then we draw the detailed information as shown in Fig.13 middle. We could know the survival opportunity of 1 class is larger than the others. So the final model should include “Pclass”. For “Name”, we should find out the important part – title. We should first classify the similar title into one group, such as “Mlle”, “Ms” represented by “Miss”. The result is shown in Fig.13 right. We could know that “Master”, “Miss”, “Mrs” are larger than “Mr” and “Rare”. For “Sex”, “Ticket” and “Fare” as shown in Fig.14, we also divide them into different group and find the inner correlation. N.B. “Ticket” may shows people knows each other share one ticket. According to figure, the 3 features should be included in the final model.

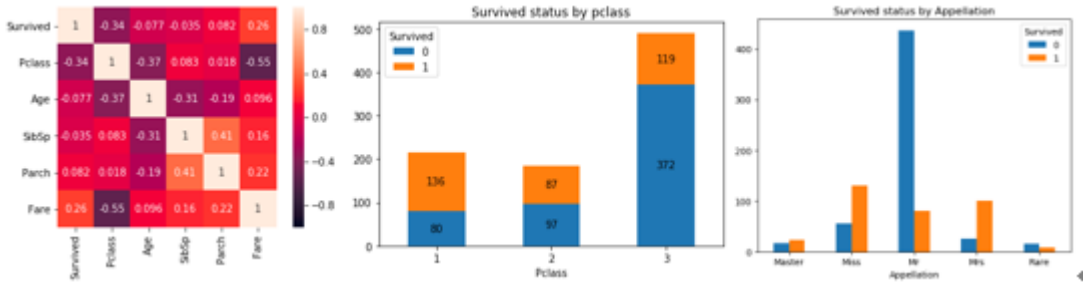


Figure 13. Correlation map, Pclass and Name

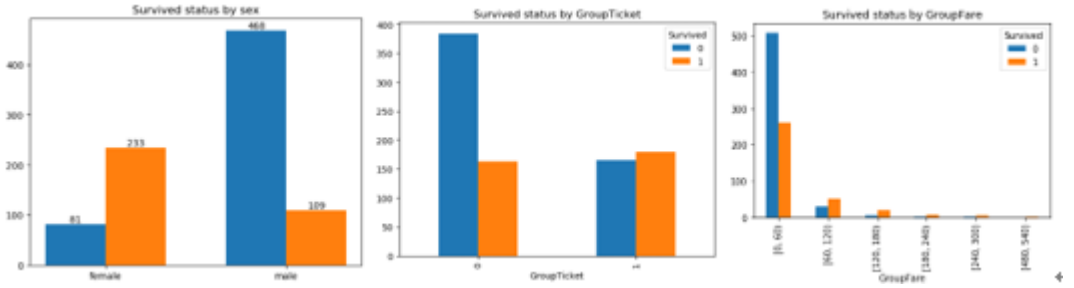


Figure 14. Sex, Ticket and Fare

We could know that main SibSp is 0, when the value is 1,2, the survive rate increases, when larger, rate decrease. Parch is similar, when the value is 1,2,3, the survive rate increases, when larger, rate decrease. Because the limitation of page, we will not show plot here. For 'Age", "Cabin" and "Embarked", which exists missing values, we will do the exploration work in section "Prepare the data" at the same time.

Prepare the data. Except for the data we selected, we now process the missing values. For "Embarked", "S" is the most value, so we use the mode number for this value. For "Cabin", which miss a lot, we choose use "NO" for the missing values. For "Age", we split it into 10 groups based on max and min values. We choose the median value relevant to the title, which means different title represents age to some extends. The 3 features are grouped and as shown in Fig.16. As expected, the three features are significant. For Age, the survival rates for kids are larger.

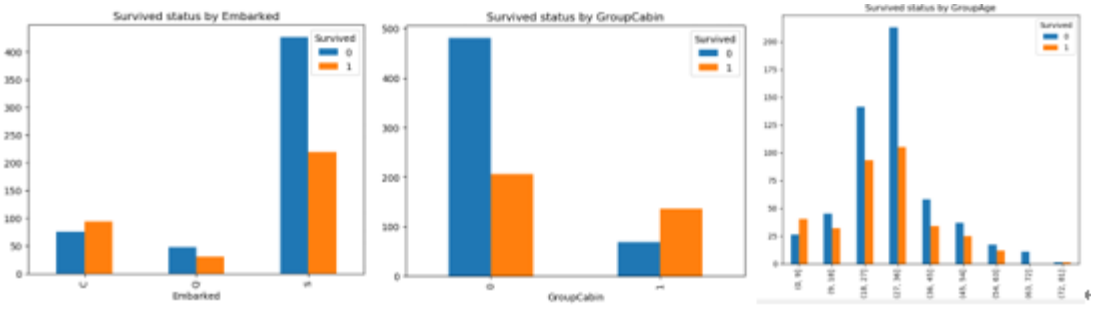


Figure 15. Embarked, Cabin and Age

We now have the following 10 needed features: Pclass, Appellation (from Name), Sex, GroupAge (from Age), SibSp, Parch, GroupTicket (from Ticket), GroupFare (from Fare), GroupCabin (from Cabin) and Embarked.
After drawing the graphs, we combine the training set and test set with the name of combined_train_test. We define the feature "Survived" of test set is 0. And then we re-generate the 10 needed feature above, the generated dataset is what we use in the model. We now notice there is NaN in Fare, so we fill the NaN with the mean price based on Pclass.

3.2 Task 2.B Classification and evaluation

Setup. And now we need to extract new features from the old ones via the following process: Transfer qualitative variable into quantitative variable: Appellation: such as "Mr" to "0"; Sex: such as "female" to 0; Embarked: such as "S" to 0. Group as range: GroupAge: such as "Age<9" as 0; GroupFare: such as "Fare<60" as 0. Combine similar features: SibSp and Parch as FamilySize.

After deleting irrelevant features, we now have following features for model: Appellation, Survived, Pclass, Sex, Age, Fare, Embarked, GroupTicket, GroupCabin, FamilySize. We then split the processed data into train dataset (for X, y value) and test dataset to apply them into model.

Evaluate and Results on Kaggle. N.B. The team name on Kaggle is VU-DM-group19 (because VU-DM-19 already exists, maybe other groups used the wrong group number).

In this step, we first applied random forest classifier in our model to train the dataset. Because random forest classifier always performs better in the dataset with not too much features. Based on reference and experiences, after several testing and cross validation, we set our parameters as: n_estimators=150, min_samples_leaf=2, max_depth=6
The result on Kaggle is 0.79904, which performs good and meet the estimation.

Then we try to improve our model to get a better score. We then used logistic regression, multilayer perceptron classifier, and then combined the 3 classifier into voting classifier.

The result of logistic regression is: 0.77033, MLP classifier: 0.78468, Voting classifier: 0.78468. All the new results do not show improvement. The reason is that parameters need to be further adjusted to get a higher score. And moreover, the data process procedure need to consider other factors, such as more precise group for each feature, if the seats position effects the survival ratio. We will improve our work in future.

4 Task 3

4.1 Task 3.A Research: State of the art solutions (10 points)

Description of the competition. Kaggle DM Competition - Humpback Whale Identification. The competition ended about a month ago. It was a quite popular competition with 2131 teams in total. To help whale protection, scientists use Photo Surveillance Systems to monitor marine activities. They use the unique mark of the whale tail to identify whales in successive images and to analyze their activities in detail. For the past four decades, most of the work has been based on the manual work of scientists, which has also resulted in a large amount of data being underutilized.

In this competition, it is required to build algorithms to identify whale individuals in the image. It is a problem with few-shot learning and fine-grained classification. But the difficulty lies in the severe imbalance of the training samples and the existence of nearly one-third of the 'new whale' data. The commonly used classification method can't handle this large amount of unlabeled data, only the innovation of traditional methods can get high scores. The solution was evaluated according to the Mean Average Precision @ 5 (MAP@5) [1]:

$$MAP@5 = \frac{1}{U}\sum_{u=1}^{U}\sum_{k=1}^{min(n,5)} P(k) \times rel(k)$$

The winner - Earhian|Venn|Tom|A.L.@KAIL [2]. In order to add 'new whale' to the network for training, they classify each type of whale and use triplet loss to make feature metrics. Through a large number of experiments, SE-resneXt154 was used as the optimal backbone. After adding a series of tricks (to be discussed in the following section of *Analysis and Comparison*), and finally, with 4 fold cross-validation, and class balance post-processing, they got the first place in public & private leaderboard of 0.973.

Analysis and Comparison. They used RGB + Mask's four-channel input. (Mask comes from their trained segmentation model, and it works well with 450 open-label data training in the forum: MAP for local cross-validation: 0.96+). While in some other teams, for example, the team of Sanakoyeu, Pleskov, Shakhray turned the color of all images to grey.

In the forum, the team of Heng CherKeng put forward the assumption that "the whale tail can be flipped to get a new category". This assumption did not receive a lot of approval votes, but it achieved very good results in their experiments. The third place team, pudae also used the same skills to improve the score.
When the model got a score of 0.96+, they added around 2000 test images (with confidence > 0.96) into their training set.

They found that as the performance of the model increases, the number of labels is correlated with scores. So they used the following strategy: Suppose the five predictions are: class_1, class_2, class_3, class_4, class_5

If (1) class_1's confidence minus class_2's confidence < 0.3, (2) and class_2 did not appear in top1, (3) and class_1 appears multiple times in top2, then class_1 and class_2 are swapped. It is worth mentioning that the 4th place, David He used full resolution images and traditional keypoint matching techniques to take advantage of SIFT and ROOTSIFT. To solve the false positive problem, he trained a U-Net to segment the whale from the background. Interestingly, he used post-processing to give more TOP-1 predictions to a category with only one training sample. The lessons learned from it is that we should never be blinded by the power of deep learning, thus underestimating the ability of traditional methods.

4.2 Task 3.B Theory: MSE vs MAE

Formulae.

$$MSE = \frac{\sum_1^N (y_i - y_i^p)^2}{n}$$

y_i is the vector of observed values of the variable. y_i^p is a vector of n predictions. MSE can evaluate the degree of changes of the data. The smaller the value of MSE, the better the accuracy of the prediction model to describe the experimental data.

$$MAE = \frac{\sum_1^n |y_i - y_i^p|}{n}$$

y_i is the vector of actual values of the variable. y_i^p is a vector of n predictions. It is the average of the absolute errors. The actual situation of the predicted value error can be better reflected.

Discussion of using MSE or MAE[4]. In general, MSE is more easy to calculate. MAE is more robust to anomalies, which means it is less sensitive to outliers.

MSE. If we have unexpected values that we should care about, MSE it the a good choice. However, If we make a single very bad prediction, the squaring will make the error even worse, especially when we have lots of noisy data.

MAE. The MAE is a linear score which means that all the individual differences are weighted equally in the average. MAE penalizes huge errors that not as that badly as MSE does. Therefore, MAE is not that sensitive to outliers as mean square error. When we have outliers in the data, we should choose MAE. However, if they are unexpected value we should still care about, we will use MSE.

Identical results of MSE and MAE. If we use a dataset which is evenly distributed errors, we will get the identical results in both MSE and MAE. The reason is that all errors have the same magnitude. Moreover, when forecasted values and square of the difference between actual values have a positive distance, the MSE and MAE will get a identical results.

Experiment. For exploring the difference between MSE and MAE, we chose the "House Price" dataset from Kaggle**[5]**, because it is a good dataset to help us understand the advanced regression techniques. This dataset includes 79 explanatory attributes describing almost every aspect of residential homes in Ames. The size of the dataset is 1459*80 on testing set, and 1460*81 on training set, including attributes price, neighborhood, house style and so on.

We test MSE and MAE on two models, including Linear Regression without regularization and Linear Regression with Ridge regularization (L2 penalty). MSE decrease after Ridge regularization (0.33 to 0.01), but MAE is not change too much after Ridge regularization (0.12 to 0.08). After Ridge regularization, the linear regression can get a better results both on MSE and MAE, which means Linear Regression with Ridge regularization can predict the house price better than Linear Regression without regularization.

Linear Regression without regularization.

MSE on Training set : 0.3335865483847917 MSE on Test set : 0.20605733818150082
MAE on Training set : 0.1270387881197999 MAE on Test set : 0.17387523444533734

Linear Regression with Ridge regularization (L2 penalty).

Ridge MSE on Training set : 0.013739965614874986 Ridge MSE on Test set : 0.013836295334691301
Ridge MAE on Training set : 0.08087532745309364 Ridge MAE on Test set : 0.08519738464044861

4.3 Task 3.C Theory: Analyze a less obvious dataset

Inspecting Dataset & Modelling Techniques. We have 5574 labeled text messages, including ham: 4827, spam: 747. Therefore, there are much more training example for ham than spam, we need to be care of the imbalance. Moreover, we should use NLP technique to process the regular texts data.

Data Transformations. Normalization: We can see there are too many different words which should belong to one attribute, so we need to replace them. For example, we need to replace phone number (0097677886) to 'phonenumber' based on regular expressions. Then, we remove punctuation and lowercase the entire corpus. Removing stop words: Then, we need to remove some non-meaningful stop words, such as "had", "do" and "this". Stemming: We combine the same corpus contains words with various suffixes. Feature engineering: Firstly, we tokenize the terms by bag of words model. Also, we used the n-gram model to get all unigrams and bigrams. Then, we computed the term frequency (tf)

and inverse document frequency (idf). Eventually, we transform the text to a matrix of numbers with one row per training example and one column per n -gram. The shape of X-ngrams matrix is (5574, 36457), so it is a sparse matrix.

Modeling. We used SVM (support vector machines) to train the model. Mover, we chose the linear kernel, because we have too many features (36, 457) and it is can help to reduce the computation. We split the dataset to 80/20 training and test set. (X_train, X_test, y_train, y_test = train_test_split(X_ngrams, y_enc, test_size=0.2, random_state=42, stratify=y_enc)) Then, we got 0.951 F1 score. As the figure 16 shows, the confusion matrix have some positive mistake which means it predicts a sms is spam but the sms actually is a ham.

		predicted	
		spam	ham
actual	spam	965	1
	ham	13	136

Figure 16. Confusion Matrix

Analysis. Firstly, we used 10-fold cross-validation (80/20 splitting) without regularization. As the figure 17 shows, the performance of training set is almost not change, because we evaluate SVM model on the same data. However, the model seems has high variance and overfitting on the validation set, because the F1 score of validation set is always below the training set. Finally, we got some top n-grams of spam sms, such as 'phonenumber', 'numbrp', 'service', 'free' and so on.

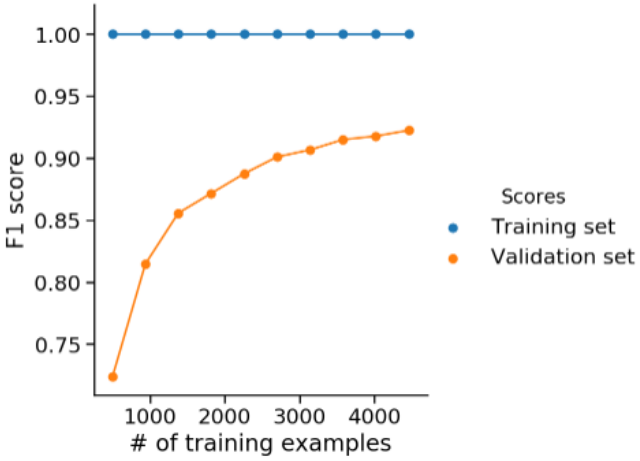


Figure 17. Learning Curves

Improvement. Preprocessing: We can add some inside features on the dataset, such as average word length and message length. Modeling: The model seems has high variance and overfitting on the validation set. so we can use smaller set of features by removing unigrams features and tune the regularization hyperparameter. We can use GridSearchCV to gets better hyperparameters, because it trains a series of candidate models using every combination of hyperparameters.

References

1. https://www.kaggle.com/c/humpback-whale-identification/overview/evaluation
2. https://www.kaggle.com/c/humpback-whale-identification/discussion/82366#latest-507782
3. https://www.kaggle.com/c/humpback-whale-identification/discussion/82356
4. https://towardsdatascience.com/how-to-select-the-right-evaluation-metric-for-machine-learning-models-part-1-regrression-metrics-3606e25beae0
5. https://www.kaggle.com/c/house-prices-advanced-regression-techniques
6. https://www.kaggle.com/c/titanic/overview/tutorials.
7. https://www.quora.com/How-does-one-solve-the-titanic-problem-in-Kaggle