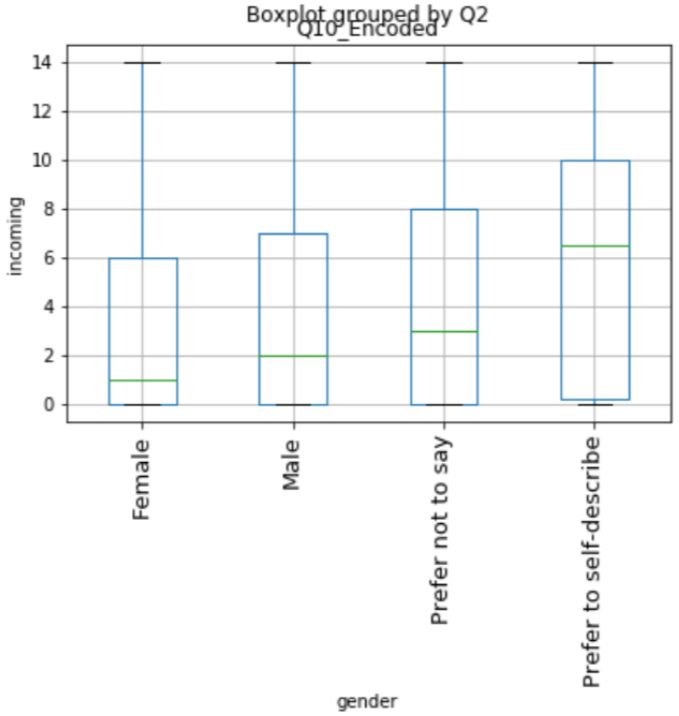
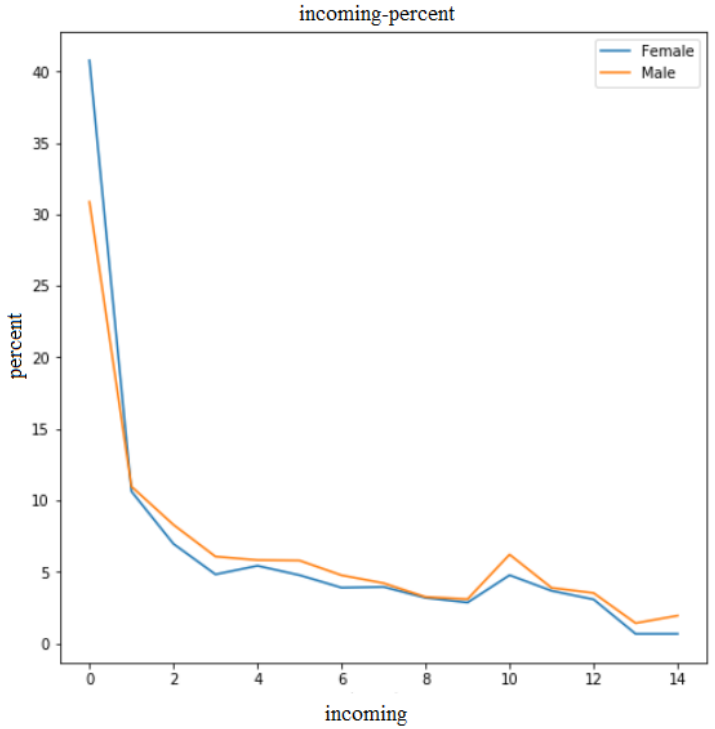
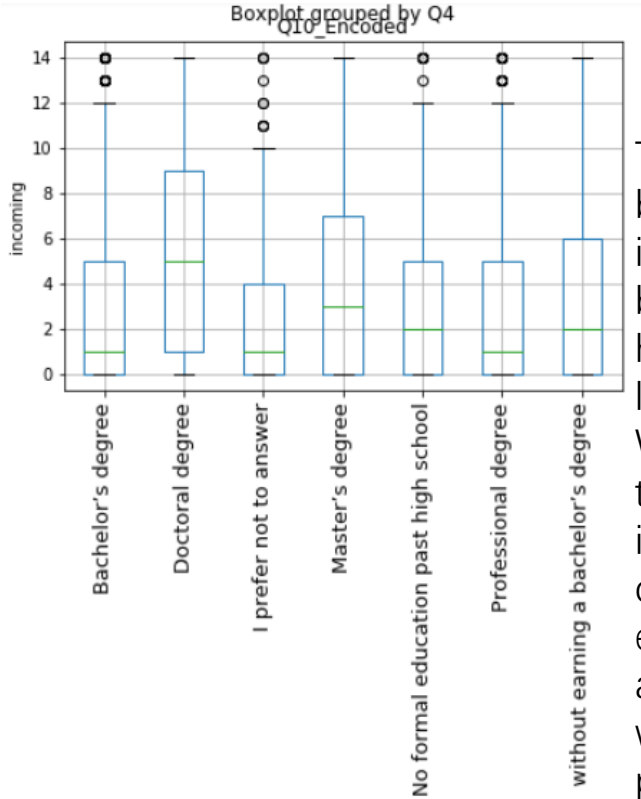
Exploratory Data Analysis

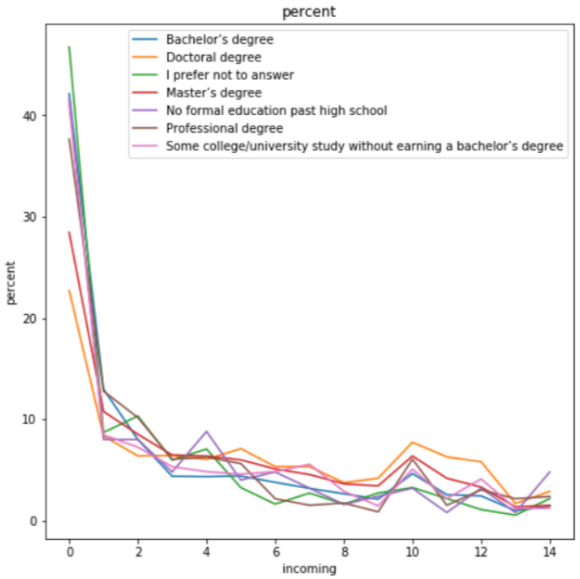
In this part, we plot the distribution of the gender in different salary buckets by the form of percentage and we also plot a boxplot to show the median of different gender.





As we can see in the left fig, all then gender has samples in rank 14, but it is shown that the quantile of females' salary is lower than males'. In the percent fig, there are 40% women's salary in rank0 while only 30% men's salary in rank0, in other rank woman's percent is 1-2% lower than man which causes the mean salary difference between different gender.(The other two gender do not shown in pic because of few sample)

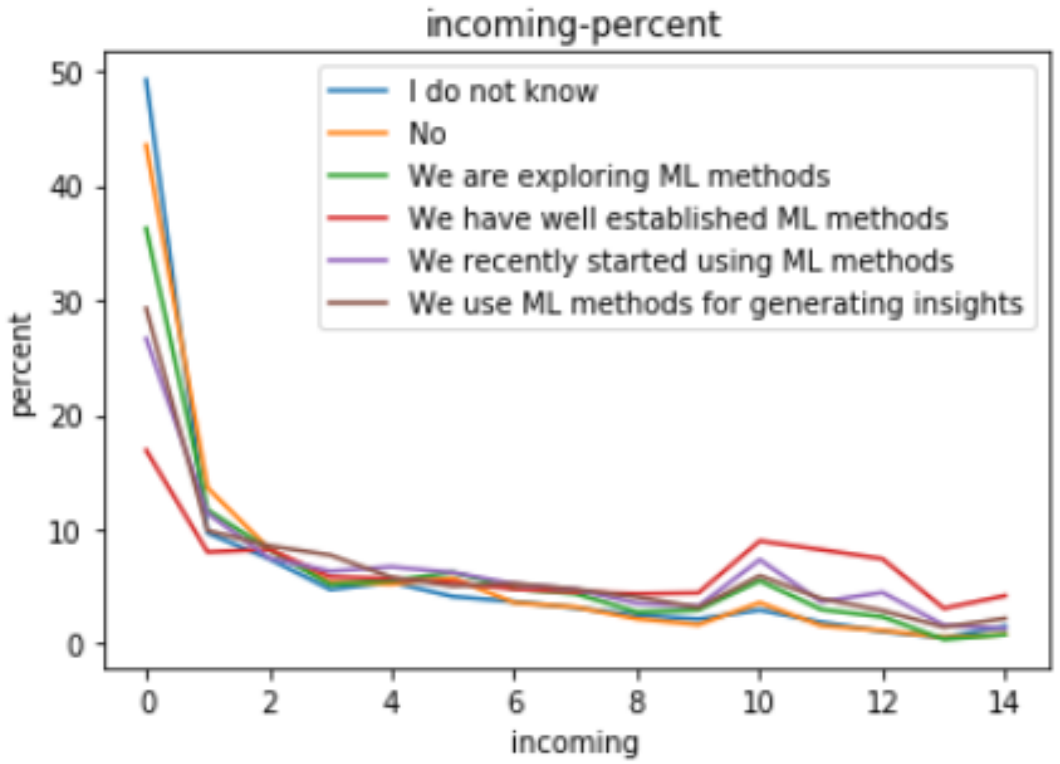


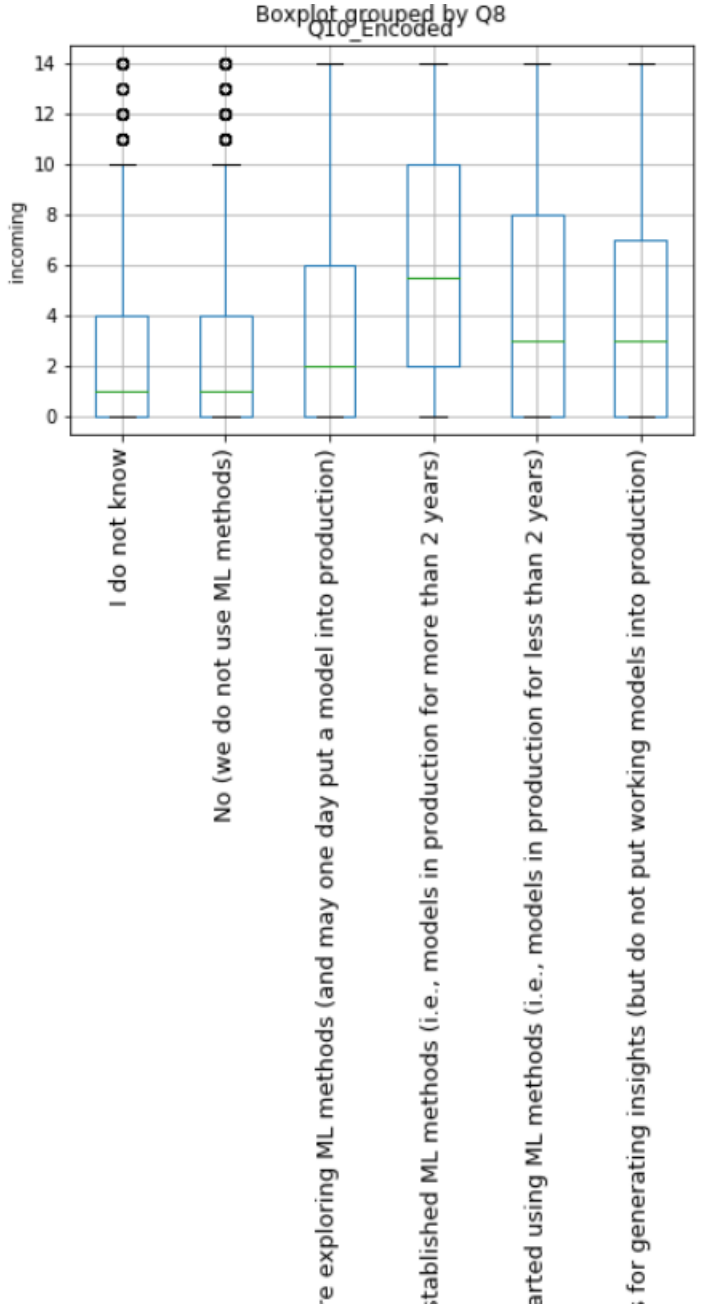


The the relationship between education and incoming. According to box plot, people with higher degree are more likely to earn more salary. What should point out is that the highest incoming of Bachelors is only 12 same as no education. the rank0 is always the main part while every line have two peaks at 4 and 10. what should point out is that the distribution does not change to much but it still matters due to the bachelor has more than 40% in rank0 while phd only has 20% in rank0

Exploratory Data Analysis

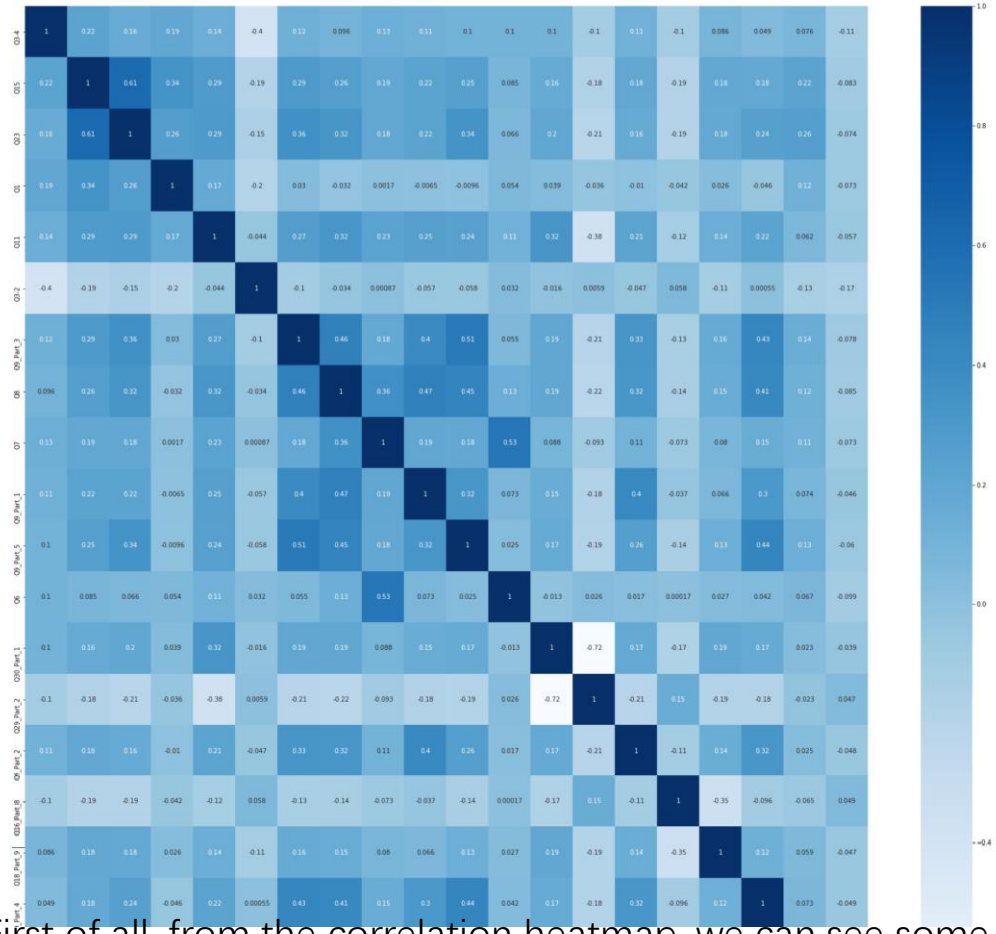
Salary vs Machine learning

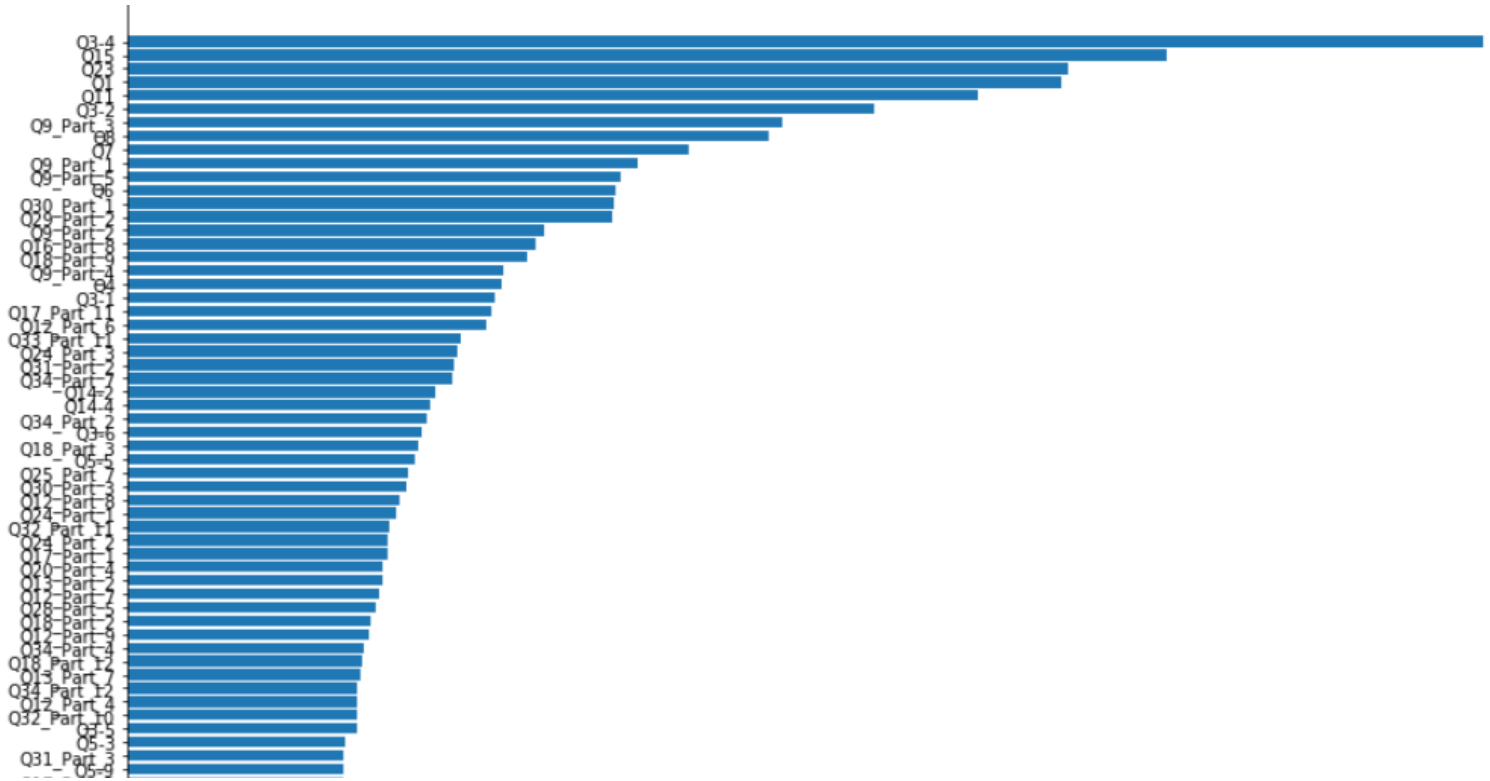




As we can see from the right picture, the quantile (no matter25, 50or 75) is increasing by the experience with machine learing. Especially with the classes ”I do not know” and “No”, the upper limit is only 10 while others’ upper limit is 14. And It also can be seen from the percentage picture. There are more than 40% people whose answer is no and no idea in rank0 which is much more than other classes. While in higher rank like 10-14, the number of people who have established ML method is more than others. This shows that machine learning method has a strong relationship with incomings.

Feature importance , correlation and selection





The left fig shows the correlation between each feature (only top 20), and the right graph shows the Pearson correlation coefficient between features and target.

First of all, from the correlation heatmap, we can see some features have correlation between them. Take Q23 and Q15 as an example, these features are both the question about writing code. So they definitely have relationship. In order to remove this unnecessary redundant data relationships, We choose to use PCA to deal with the whole data.

Second, as shown in the Pearson correlation coefficient. The Q3-4:"Which country you live in USA" , the Q15:" How long have you been writing code to analyze data (at work or at school)?" and the Q23" For how many years have you used machine learning methods?" are the top 3 important feature.

In the process of selecting feature we choose lasso. Lasso is L1-regularized linear regression and what we do is finding best lambda in the test site and apply it to the whole part. Then remove the feature whose weight is 0.

Model Implementation, Tuning and Testing

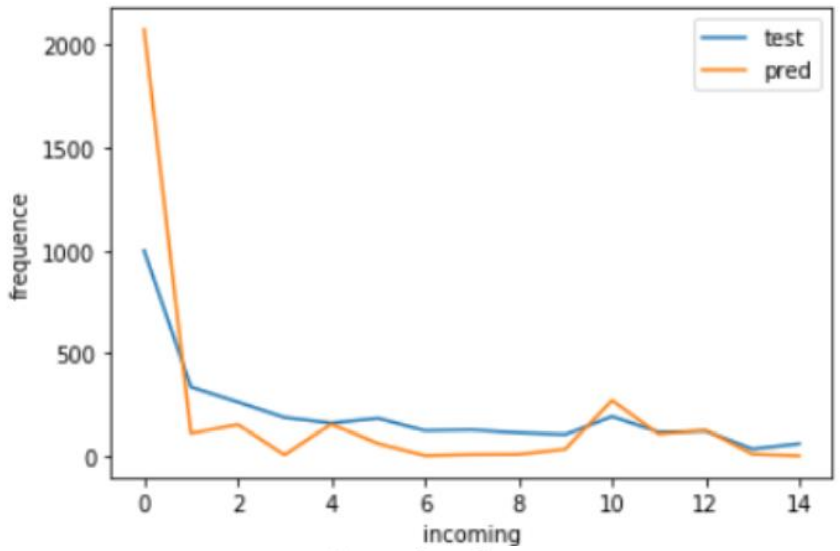
The principle we used in build the model is that build 14 Classifiers and transform y into 14 forms like y0={0:0, 1-14:1} y1={0-1:0, 2-14:1} y2={0-2:0, 3-14:1} y3={0-3:0, 4-14:1}, as the function we define below. Then we use each y and X_train to train the model. After doing that we get 14 different model. Then we use these models to predict the X_train and return probability of samples in 0 or 1. And then use the probability that sample in rank 0-13 minus the probability that the sample I rank 0-12 to get the probability the sample in rank 13. And then rank12... Finally we get the probability of belonging to each rank. Then we choose the largest probability and class the sample into this rank.(what we get is a df in the right)

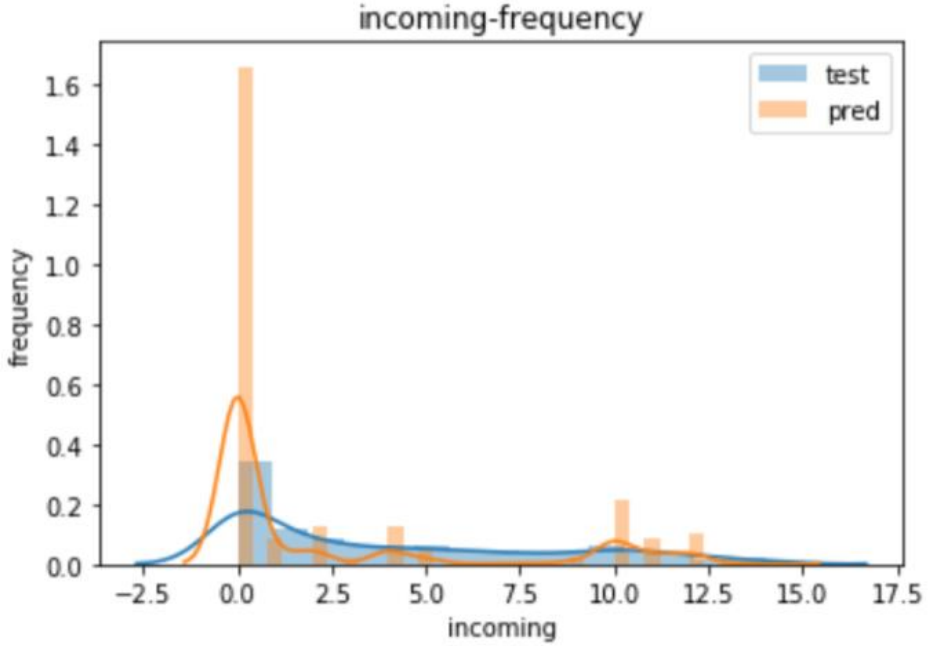
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0.355378	0.112162	0.073611	0.055603	0.051644	0.051899	0.046398	0.038547	0.030094	0.027365	0.055978	0.036064	0.033426	0.014440	0.017392
1	0.305278	0.107412	0.083363	0.056773	0.062976	0.057919	0.054806	0.047162	0.032632	0.030695	0.057883	0.038836	0.032600	0.014290	0.017376
2	0.382027	0.119464	0.078979	0.052338	0.051974	0.049280	0.038770	0.035018	0.024244	0.024708	0.048617	0.033982	0.030021	0.013500	0.017078
3	0.256267	0.094534	0.071286	0.066394	0.080872	0.070077	0.064862	0.048685	0.038552	0.032880	0.067267	0.040143	0.035701	0.015057	0.017423
4	0.313613	0.115488	0.076717	0.069129	0.076276	0.059737	0.048767	0.043531	0.027368	0.029622	0.048145	0.031263	0.030007	0.013455	0.016881

	C value	solver
0	0.01	'liblinear'
1	0.0025	'sag'
2	0.001	'liblinear'
3	0.01	'lbfgs'
4	0.005	'lbfgs'
5	0.001	'liblinear'
6	0.0025	'liblinear'
7	0.005	'liblinear'
8	0.01	'liblinear'
9	0.01	'lbfgs'
10	0.01	'lbfgs'
11	0.01	'lbfgs'
12	0.01	'lbfgs'
13	0.005	'liblinear

In the tuning process, the hyperparameter need to be tuned is C value and solver. Instead using build in model , we write a loop to do the grid search work.
The hyperparameter we used before tuning is C=1 and solver='liblinear' and the accuracy on the training set got by doing cross-validation is 32.43%.
After tuning the hyperparameter for each model and re-run the whole model the accuracy becomes 33.31%. There is 1% higher than before.
The accuracy on the testing set is 34.9%. there is not too much difference between them. It can not be seen as overfitting. It tends to be a little underfitting for the whole 14 model.
Why the accuracy that low is that in the 0,1-14 classification and 0-1,2-14classification rank, the models for these works do not work well. the accuaracy is only 78%, which limit the upper limit of the entire model.

Conclusion and Visualization





This is the histogram of the y_test and the y_pred. In the fig we can see clearly that the peaks of the distributions of them are coincident which appears in rank0 and rank10. However, in the predicting ,there are too much sample be identified as 0 and it is 2 times more than the real situation and that's the reason why the accuracy keeps low . According to this situation we can conclude that the model is underfitting and the reasons are below.

First all of the data cleaning is not perfect, we just encoding them and we gave some feature which is not totally linear ranks(like education degree and working year). So we cause some noise by this and we make some noise when we handle the missing value.
Second, in the modeling processing we just tuning the hyperparameter for each single model instead of doing it to the whole model. And we ignore the effect caused by hyperparameter changing to the whole predicting model. So the improvement is little.
Finally, the model is too simple to handle such a huge mount of data , if we want to get a higher accuracy, a more complex model is required. logistic model works pretty well in 0-1 classification question. But when comes to the multi-class question, especially it have 15 classes, it is not expected to have a good accuracy. perhaps we can combine LR with other model. For example we can use LR model to classify the 0-11,12-14. And in 0-11 classes we do not use LR we use other kinds of model to do the predict work.