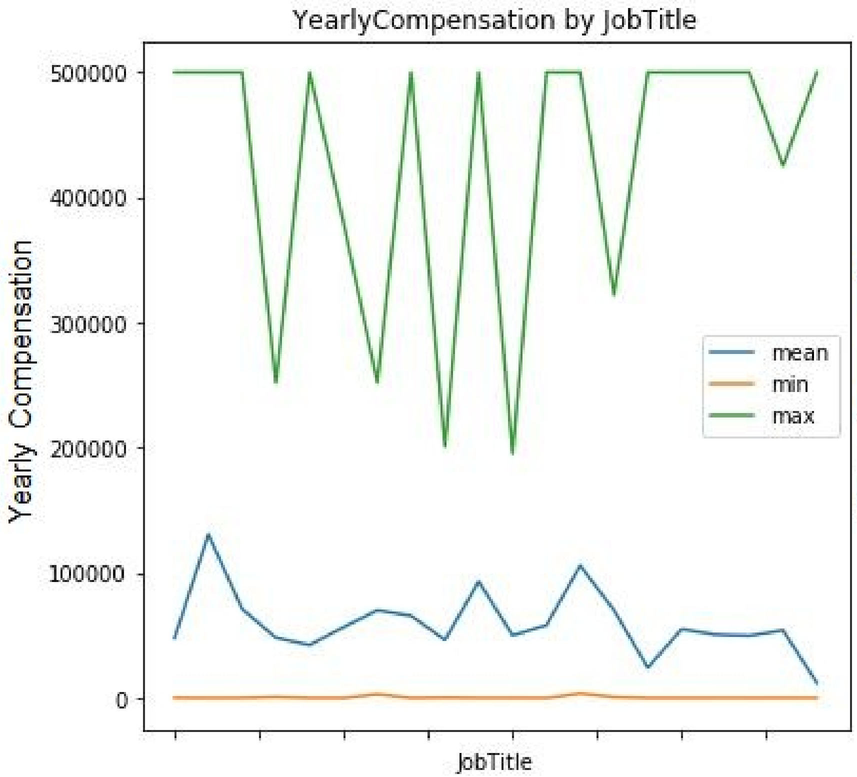
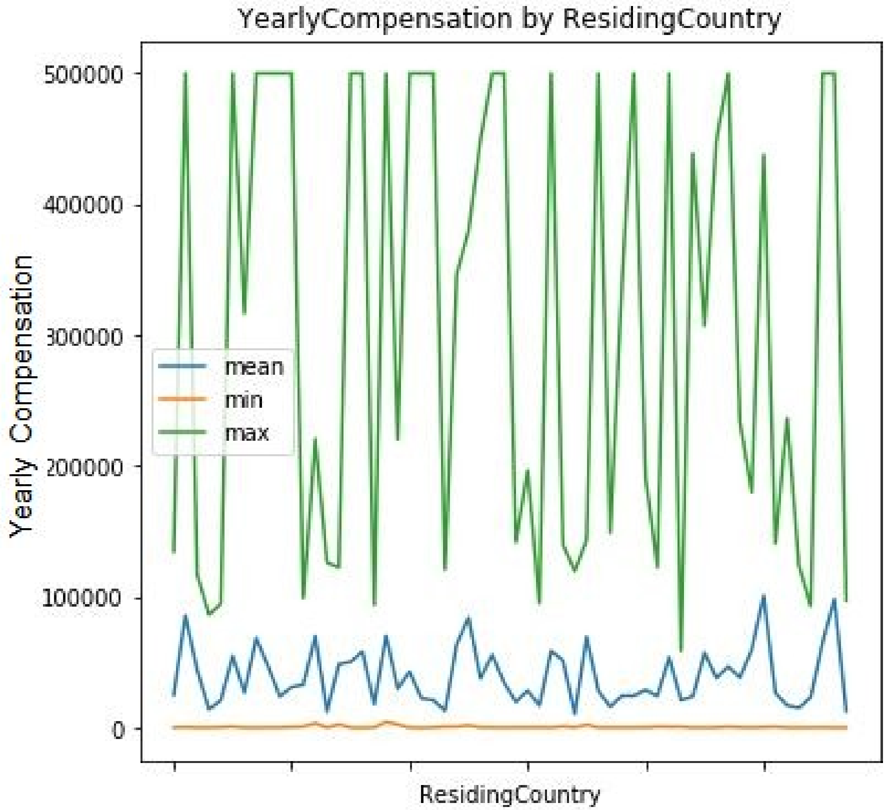
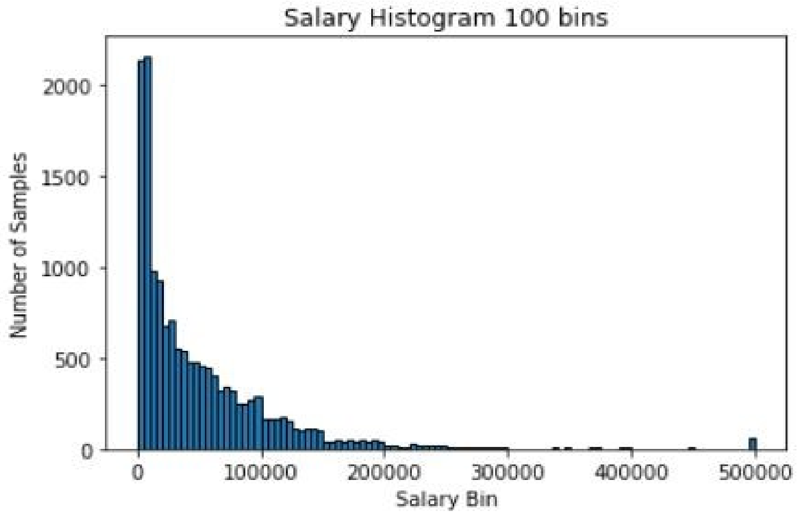
Exploratory Data Analysis

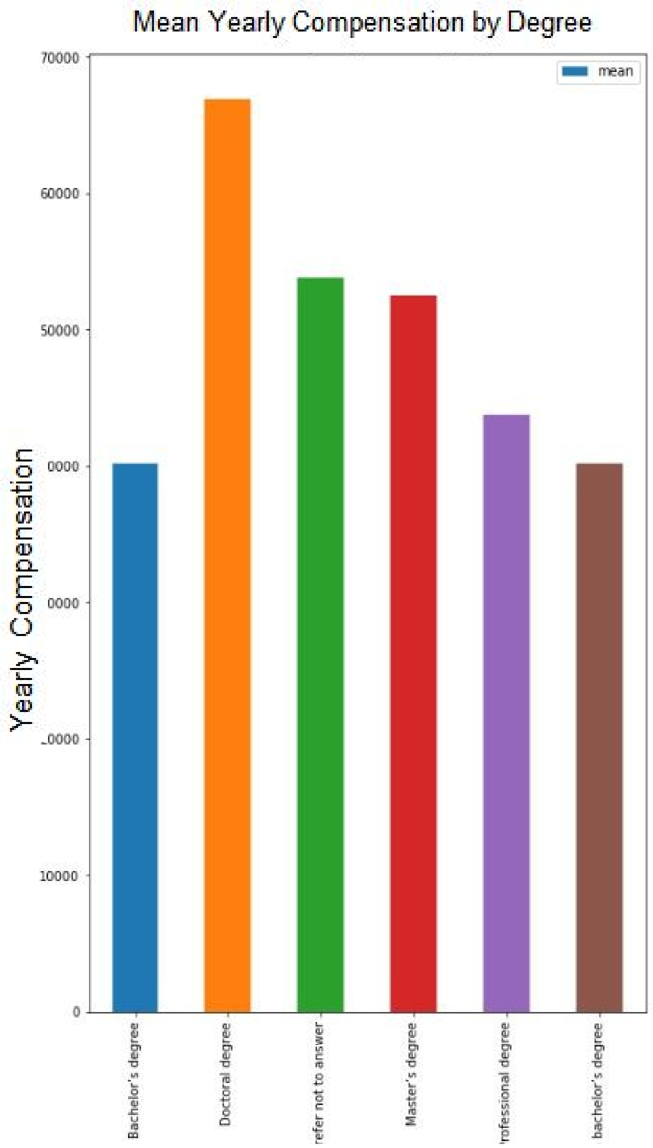


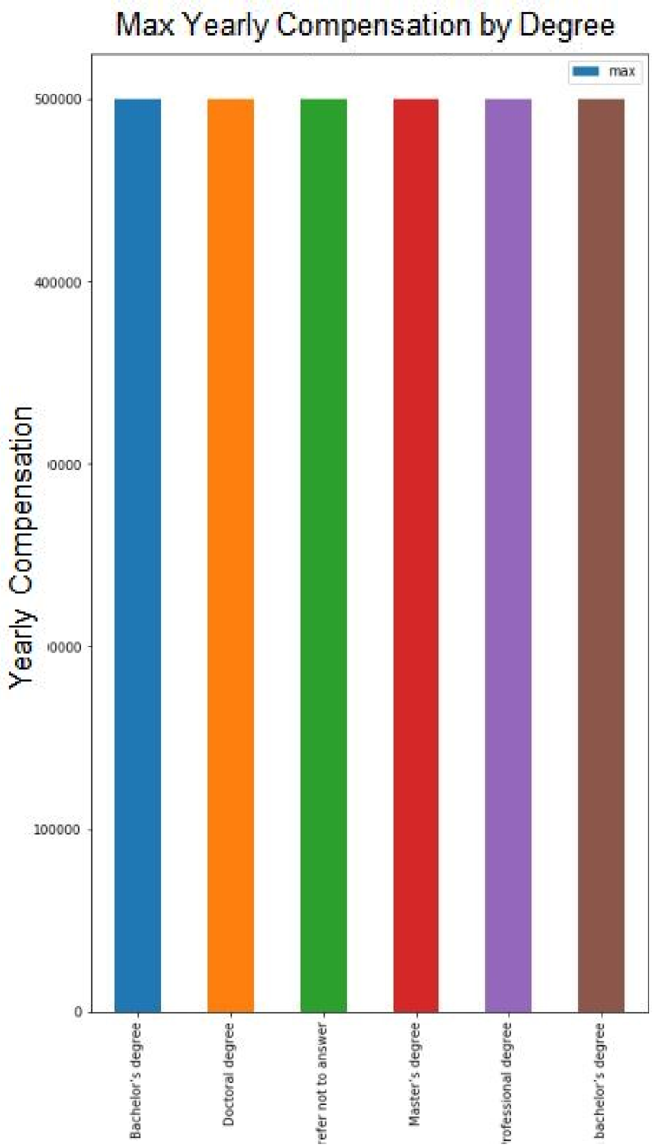


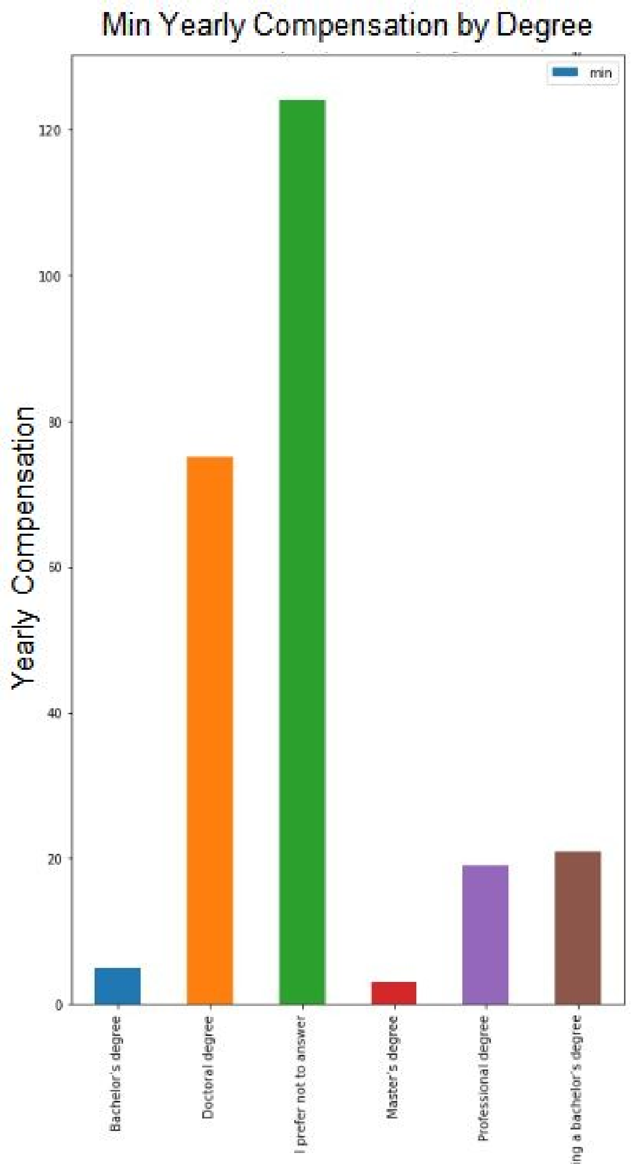
The left two figures respectively show the relationships of JobTitle vs YearlyCompensation, Residing Country vs YearlyCompensation. The mean and maximum YearlyCompensation vary with these two variables. Comparatively, the minimum YearlyCompensation are almost the same.



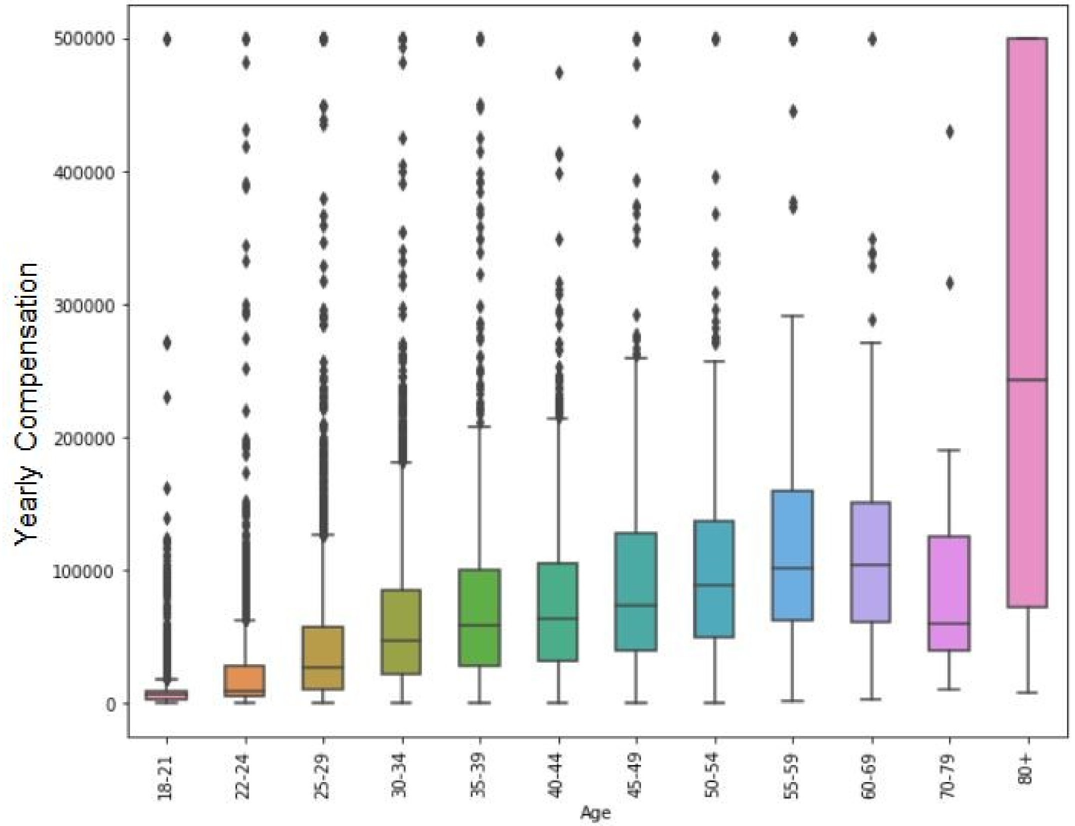
The primary concern for further regression task is existence of outliers in our dataset. The histogram on the left shows the distribution of yearly compensation. From 0 to 300,000 the number of persons keeps dropping. But it's weird that there is a minor peak at 500,000. We have reason to doubt the authentication of these data records. Therefore, for future regression task, we will set a threshold to eliminate the outliers.

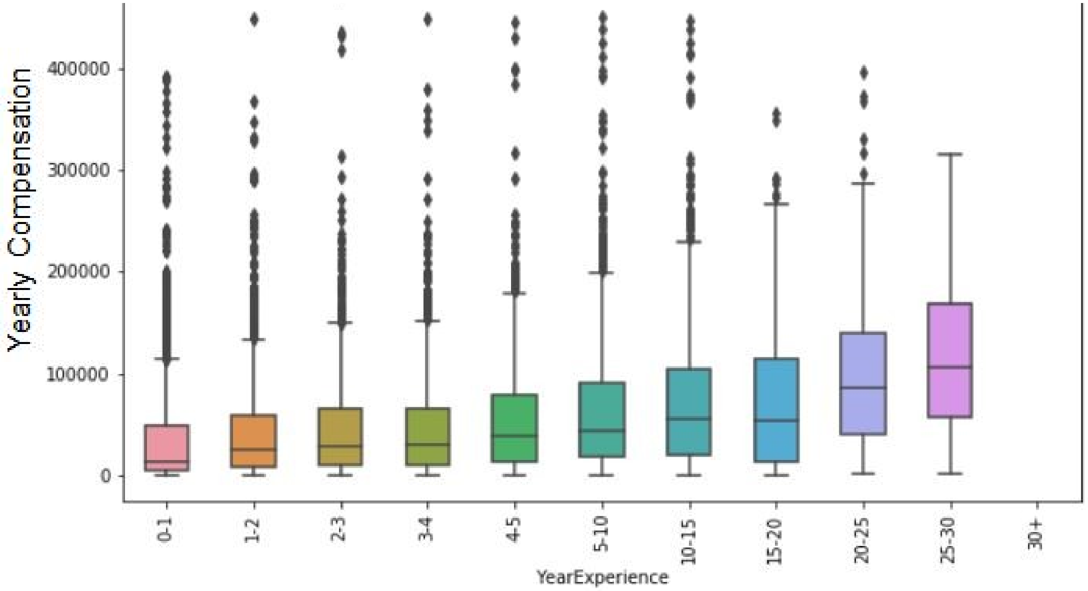






Above figures show mean, max and min Yearly Compensation of people with different degrees. According to mean value, people with higher degree are more likely to earn more salary. People with different degrees all have chance to earn salary as high as 500,000. From the min Yearly Compensation, we notice a weird phenomenon that some people earn as less as 100. These values are likely to be fake and unreliable since respondents may take this survey slightly and give false values. We will set a threshold to delete records less than the threshold in the further deployment.





In the left two figures, we dig into the relationship between Yearly Compensation and length of work, so we respectively show the relationships of Age vs Yearly Compensation, Year Experience vs Yearly Compensation. We can see that older people or those who have longer work experience are more likely to have a higher yearly compensation. Therefore we can normalize these features with with mean value to keep numerical order.

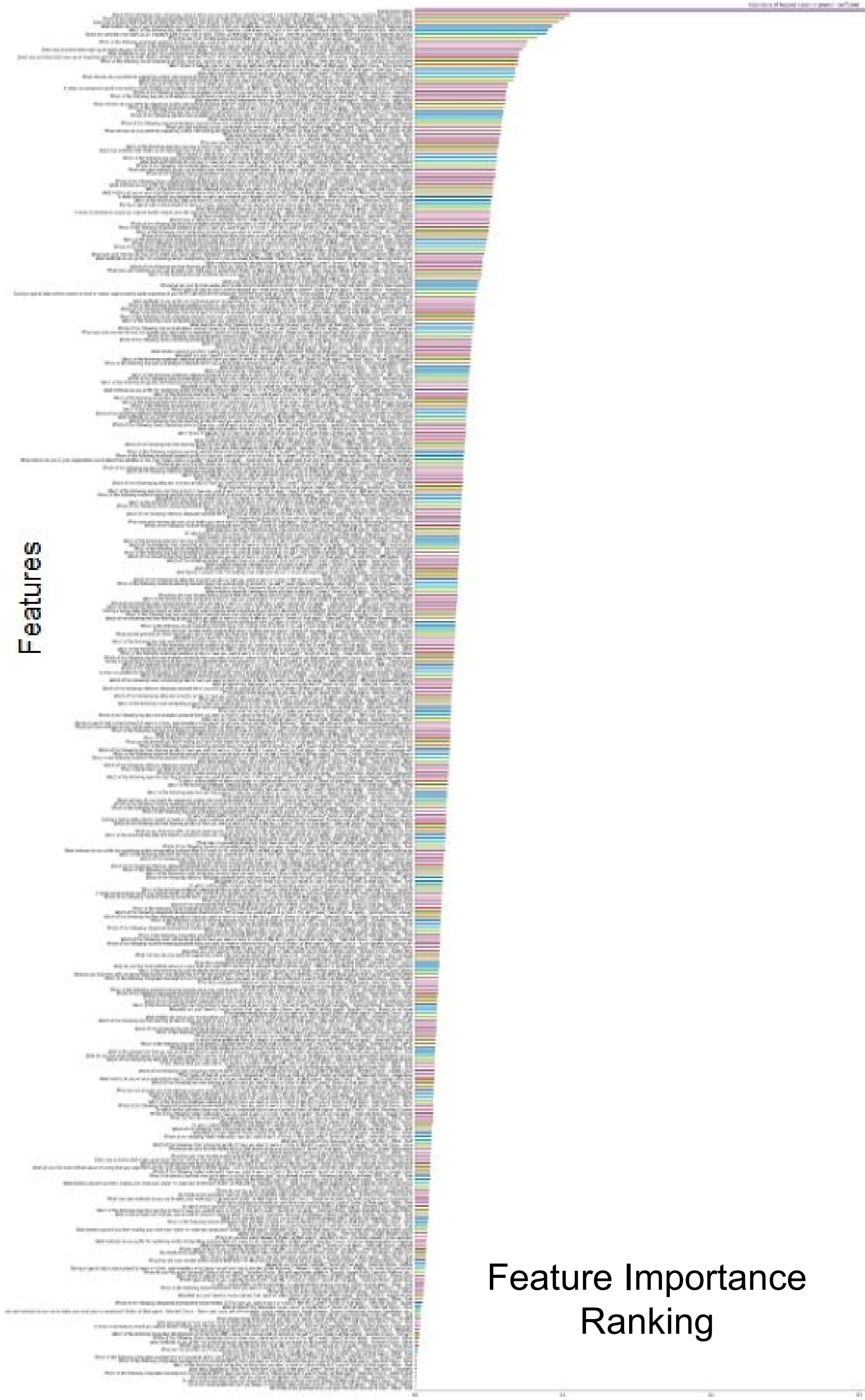
In the right figure, we rank the features according to their pearson correlation with yearly compensation.

The top 3 most important features are:

1. Used AWS in the past or not
2. Used EC2 in the past or not
3. Built prototypes to explore applying machine learning to new areas in work or not

The top 3 less important features are

1. Used Udemy as online platform to take data science courses or not
2. Used DataCamp as online platform to take data science courses or not
3. Use Other online platform to take data science courses or not



Feature Selection and Model Implementation

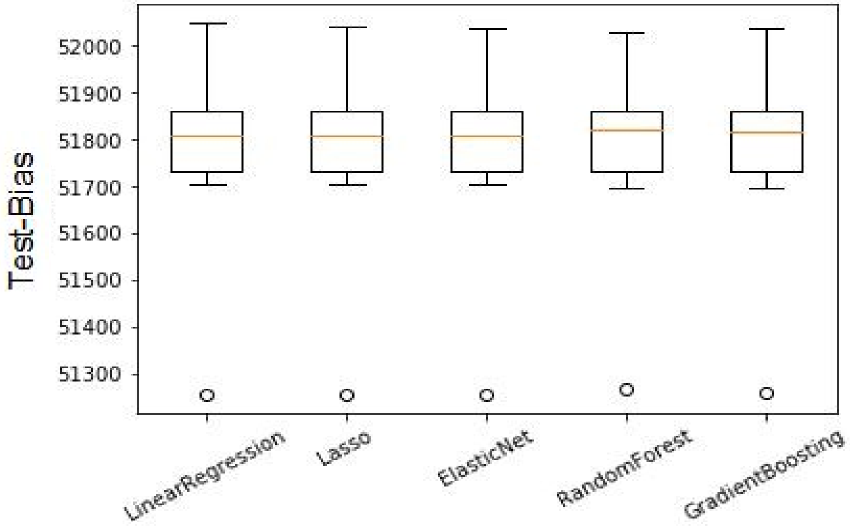
After cleaning data and featurization, we implemented two kinds of feature selection methods "Principal Component Analysis(PCA)" denoted by 'F1' and "SelectFromModel-Lasso" denoted by 'F2'. Then we implemented five machine learning methods, which are "Linear Regression", "Lasso", "ElasticNet", "Random Forest", "Gradient Boosting" to do regression task.

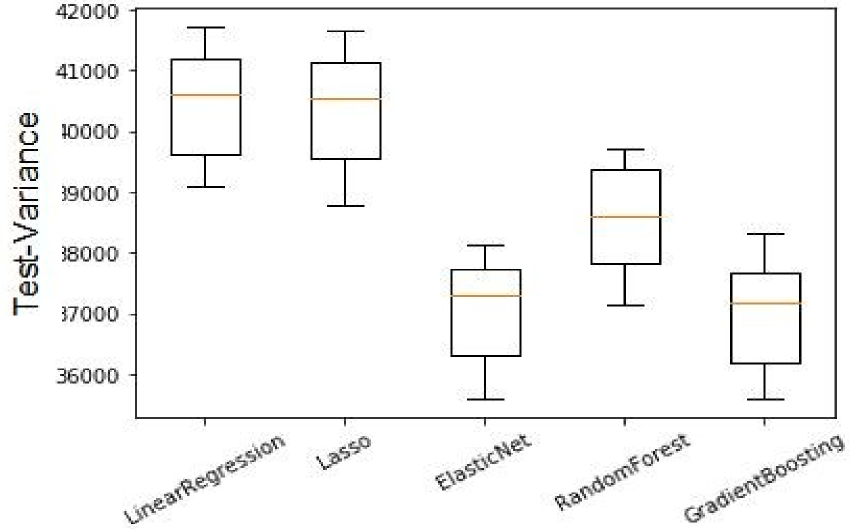
Algorithm	R2-F1	R2-F2	RMSE-F1	RMSE-F2	MAE-F1	MAE-F2
LinearRegression	0.504585	0.584582	36385.268611	33325.829170	24985.537382	22320.800607
Lasso	0.504709	0.585777	36380.798480	33280.844648	24977.825815	22260.589416
ElasticNet	0.505481	0.581069	36360.055688	33471.304581	24830.968799	22256.419912
RandomForest	0.330818	0.583253	42306.977638	33379.665461	29912.679960	21008.869324
GradientBoosting	0.387567	0.582384	40481.930780	33416.478943	27916.286427	21330.622497

In table above, to choose better feature bunch, we compute the R-Square(R2)/Root Mean Square Error (RMSE)/Mean Absolute Error(MAE) scores on testing set. Note that during the training, we do 10-fold cross-validation and the testing scores are the mean of 10 folds. From the test performance, we can see that F2 is better than F1 from every measurement. Therefore, in the following steps, we discard F1 and only develop on F2 bunch ("SelectFromModel-Lasso").

Algorithm	R2-Mean	R2-Variance	RMSE-Mean	RMSE-Variance	MAE-Mean	MAE-Variance
LinearRegression	0.584582	0.022330	33325.829170	1422.097882	22320.800607	462.086330
Lasso	0.585777	0.021063	33280.844648	1441.114786	22260.589416	471.831081
ElasticNet	0.581069	0.021440	33471.304581	1507.973160	22256.419912	520.076451
RandomForest	0.583253	0.029997	33379.665461	1789.502425	21008.869324	704.257489
GradientBoosting	0.582384	0.025997	33416.478943	1653.933987	21330.622497	632.595015

To compare the models performance on chosen feature bunch, we use R2/RMSE/MAE to measure the accuracy. R2 value and RMSE score are almost the same across different algorithms. The MAE score of Random Forest is the lowest but it has high variance.





Above we plotted the bias and variance of five algorithms cross 10-fold. The bias are almost the same, but the variance are quite different. LinearRegression and Lasso have the larger variance, while ElasticNet and GradientBoosting have lower variance. Considering that GradientBoosting performs better than ElasticNet in accuracy, we choose GradientBoosting as our best model.

Model Tuning, Testing and Conclusion

We use GridSearch Cross-Validation to tune the hyperparameters.
1. For each algorithm, we define the hyperparameter space beforehand.
2. With every candidate hyperparameter, GridSearchCV will first split the data into K folds, train the model on K-1 folds and then score on test data. We choose to use R2 as performance measure score. R2 measures how well the regression line approximates the real data points, it also portrays percent of variance in the data explained by regression model. GridSearchCV will give the best hyperparameters based on the average R2 score cross K times training.

Algorithm	R2	RMSE	MAE
LinearRegression	0.609853	32337.127780	21622.296833
Lasso	0.609822	32338.420808	21605.431551
ElasticNet	0.609853	32337.137689	21621.014060
RandomForest	0.896974	16617.246984	9424.845054
GradientBoosting	0.846006	20316.004314	13305.775134

Based on the hyperparameter tuning, we obtained the best models of each algorithm and we listed their performance on the whole training set in the above table. According to above table, the models that perform better are RandomForest and GradientBoosting. RandomForest seems to be the best. However, from our analysis in last step, GradientBoosting has much lower variance than RandomForest and the bias are almost the same. So we choose GradientBoosting as our best model. The best hyper-parameter are {'max_depth': 7, 'min_samples_leaf': 2, 'n_estimators': 110}.

Measurement	Train	Test
R2	0.846006	0.621837
RMSE	20316.004314	31592.944441
MAE	13305.775134	19926.766717
Bias	51771.136888	51377.338307
Variance	43318.851196	40983.183941
Total	95089.988084	92360.522249

We used our optimal model to make predictions on the training and test set. The results are listed above. From every perspective, the performance on training set is better than that on testing set.
Our model is overfitting. First we can see the R2 score is much higher for training than testing, which means that the model fits training set so well that it has captured the noise of the data. The variance difference between training set and testing set is much higher than bias difference, and the variance in training stage is a little higher. Therefore we can conclude that the model is overfitting.
To improve the testing performance, we can fit multiple models and use validation or cross-validation to compare their predictive accuracies on test data. In the case of GradientBoosting, the maximum tree depth also plays a huge role in determining the fit, so we can decrease the max depth to significantly reduces overfitting.