ML project

May 15, 2017

1 Identifying fraud from email a mchine learning project

1. Project Overview

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, a significant amount of typically confidential information entered into the public record, including tens of thousands of emails and detailed financial data for top executives. The data set comntains a total of 146 data points (individual names) and 21 features. From the 146 individuals 18 of them are labeled person of interest (poi), and 127 are labeled non porson of interest (non_poi). From the 21 features available some features have many missing values for example the feature "loan_advances" has the most missing values. In this project, I played detective, and built a person of interest identifier based on financial and email data made public as a result of the Enron scandal. Before, I started feature selection (engineering) and algorithm selection, I did a quick EDA to see if any outliers exist in the data. I found an outlier called "TOTAL" and removed it.

2. Feature selection:

In order to get a better precision and recall, I selected the list of features that I believe are intuitively relevant followed by automatic feature selection techniques such as PCA and univariat feature selections SelectKbest and SelectPercentile. The automatic feature selections have two fold functions, to reduce the number of features to avoid over fitting since my dataset is small and to extract the best features or combinations. I also tried to engineer a new single financial feature from the financial features. The single fiancial feature was obtained by adding the financial features ('salary','total_stock_value','exercised_stock_options','bonus']) for each individual. Hence my new feature was features_list1 = ['financial_feature','deferral_payments', 'total_payments', 'restricted_stock_deferred', 'expenses','deferred_income', 'other', 'long_term_incentive', 'restricted_stock', 'director_fees', 'to_messages', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi']. By combining SelectKBest(f_classif) feature selector and GaussianNB classifier, precsion of 0.40769, recall of 0.32350 and F1 score of ˜ 0.36 was obtained for the features with the new feature added.

Even though, the precision and recall obtained from the previous features are good, I used the following features as my final features ['salary', 'deferral_payments', 'bonus', 'restricted_stock_deferred', 'deferred_income', 'total_stock_value','expenses', 'exercised_stock_options', 'other', 'long_term_incentive', 'restricted_stock','director_fees', 'to_messages', 'from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi'], then scaled them using the scikit-learn MinMaxScaler to avoid problems caused by different units in the dataset and finally the final features that yield the best result were selected by SelectKBest(f_classif) with f_classif scoring function. The implementation of SelectKBest(f_classif) with GridSearchCV() for parameter tuning choosen 8 features with their corrosponding scores and p values.

[('exercised_stock_options', '25.10', '0.000'), ('total_stock_value', '24.47', '0.000'), ('bonus', '21.06', '0.000'), ('salary', '18.58', '0.000'), ('deferred_income', '11.60', '0.001'), ('long_term_incentive', '10.07', '0.002'), ('restricted_stock', '9.35', '0.003'), ('shared_receipt_with_poi', '8.75', '0.004')]

3. Algorithm Selection:

I selected and tested the following algorithms.

1. Logistic regression : Precision: 0.43473 Recall: 0.17650 F1: 0.25107. This algorithm takes considerably longer time
2. SVM (with linear kernel): Yielded the worst result for the same feature. Can't predict true positives at all.
3. Gaussian naive Bayes(with PCA): recision: 0.52788 Recall: 0.27450 F1: 0.36118 F2: 0.30365.
4. Gaussian naive Bayes with the new fiancial feature: Precsion 0.40769, Recall of 0.32350 and F1 score 0.36.
5. Gaussian naive Bayes: Yielded the best result Precision: 0.42486 Recall: 0.34350 F1: 0.37987 F2: 0.35718.

The best algorithm selected is Gaussian naive Bayes.

4. Parameter tuning:

Parameter tuning is the technique of obtaining the best parameters of the model (regressor or classifier) that improve its performance on an independent data set. If the paramaeter of the model are not optimally tuned to the best configuration, the model's performance would be lower than it could do. In this project parameter tuning was done using combination of automatic feature selections scikit-learn GridSearchCV. For Gaussian Naive Bayes, Logistic regression, and SVM, SelectKBest and PCA was tuned to determine the optimal features. The parameters tuned for each algorithm or pipeline operator are shown below:
Select K Best (SelectKBest): scoring function (chi2, f_classif), K (5,8,10,12,15,16).
Principal Component Analysis (PCA): n_components, (2,4,5,6,7,8,9,10,11,12, 13, 14, 15).
Gaussian Naive Bayes (GaussianNB): None, no input parameters to tune
logistic regression (LogisticRegression): penality= ('l1', 'l2'),C=(0.01, 0.1, 1, 10)
Support Vector Machine (SVC): {'kernel': ['rbf'], 'gamma': [1e-3, 1e-4],'C': [1, 10, 100, 1000]}

5. Validation:

If we train and test a model with the same data set, the model would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called overfitting and is a classic mistake in machine learning. To avoid it, we usually split the data into training and testing sets for cross validation. This can be done, by methods such as K-Fold,where we split our dataset into K units and the algorithm is trained using all but one of the partitions, and tested on the remaining partition. The partitions are then rotated several times so that the algorithm is trained and evaluated on all of the data. However, using this method wouldn't shuffle our data, and is not ideal choice for a data like the one we are dealing now since it is unbalnced (More NON POIS than POIS). Hence, I used StratifiedShuffleSplit method to get a more accurate result.

6. Evaluation metric:

Many metrics can be used to measure whether or not an algorithm is learning to perform its task more effectively. For supervised learning problems, many performance metrics measure the number of prediction errors. In our classification task in which our machine learning system observes persons and must predict whether these persons are person of interest(POI) or not person of interest(NON POI) in enron corporate fraud. Accuracy, or the fraction of instances that were classified correctly, is an intuitive measure of ourclassifier algorithm's performance. While accuracy does measure the athe classifying algorithms's performance, it does not differentiate between POIS that were classified as NON POIs, and NON POIS that were classified as being POI. In some applications, the costs associated with all types of errors may be the same. In this problem, however, failing to identify POI is likely to be a more severe error than mistakenly classifying NON POIs as being POIs. That's why evaluation metrics, precision-recall-f1 score is needed. In our case, when our classifier correctly classifies a person as being POI, the prediction is called a true positive.

When the system incorrectly classifies a NON POI as being POI, the prediction is a false positive. Similarly, a false negative is an incorrect prediction that the person is NON POI, and a true negative is a correct prediction that a person is NON POI.

True positives (TP) is when the model thinks is positive, and it's actually true.
False positives (FP) is when the model thinks is positive, but it isn't (false).
False negatives (FN) is when the model thinks is negatives, but it isn't (false).
True negatives (TN) is when the model thinks is negatives, and it is actually true.

Precision : The fraction of persons classified as POI that are actually POI
P=$\frac{TP}{TP+FP}$

Recall (Sometimes also called sensitivity in medical domains): is the fraction of the truly positive instances that our classifier recognizes. For our POI identifier, recall is the fraction of POIS that were truly classified as POI.
$R = \frac{TP}{TP+FN}$

Since our model is designed to identify POIs, recall is a more important metric than accuracy and even precision. Therefore, our model is inclined more to improve the recall than the precsion. Hence, even though we obtained a higher precision (˜ 0.5) with Gaussian naive Bayes classifier (linear kernel) and principal compnent analysis (PCA) feature selection combinations, we choose the same algorithm but with univariat feature selection (SelectKbest) since it gave a maximu Recall of 0.34350 with reasonable Precision of: 0.42486 and averge performance metric F1 score of : 0.37987.

References

1. http://scikit-learn.org/stable/
2. Mastering Machine Learning with scikit-learn, Gavin Hackeling 2014, Packt Publishing.
3. http://broadwater.io/identifying-fraud-from-enron-email/
4. https://github.com/lyvinhhung/Udacity-Data-Analyst-Nanodegree/tree/master/p5%20-%20Identify%20Fraud%20from%20Enron%20Email