CSE514A Course Project3

Detecting Credit Card Fraud With Imbalanced Data

Team members: Xiaoyao Wang, Lisa Liao, Lijun Yao

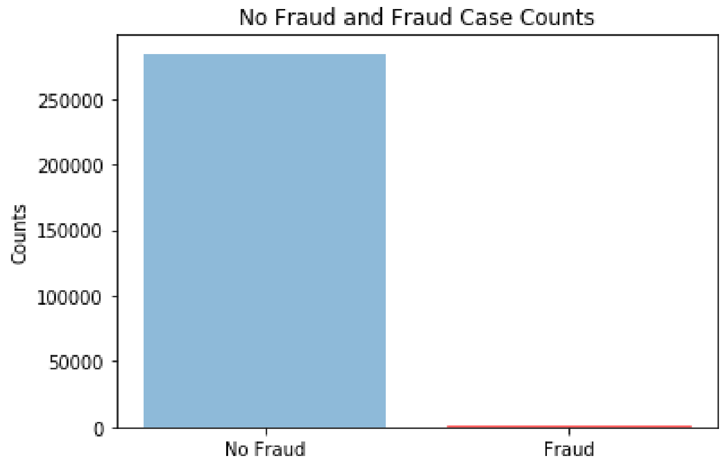
1) The problem(s) you want to solve/analyze and the data you use (1 pt)

Problem:

The problem we would like to solve is to build a predictive model that can detect fraudulent and non-fraudulent transactions among all credit card transactions.

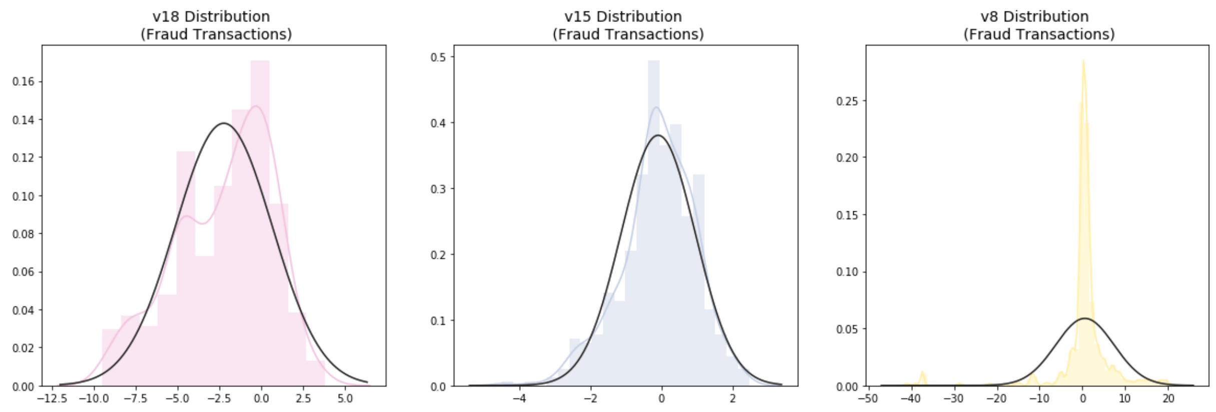
Understanding the data:

The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly imbalanced, the positive class (frauds) account for 0.172% of all transactions.



The dataset contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, original features and background information are not available. Features V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.[1] These 3 columns are all normalized to match the scale of the other feature columns before training.

We then investigated the distribution of some of the features. Here, we plot the distribution of V18, V15 and V8. Compared to V18, V15 and V8 have Gaussian distribution with less extreme values/ outliers.



Due to the highly imbalanced nature of this dataset, a simple random train-test split will likely result in underrepresentation of the fraud cases in training or testing data, and a bias towards the majority. Therefore we apply random undersampling and SMOTE to explore and compare their performance with different classifiers on our dataset.

2) A very clear statement of the objectives you aim to achieve (1 pt)

With different combinations of classifiers and imbalanced data-processing techniques, the objective is to find a model with over 90% true positive and true negative rates without sacrificing false negative and false positive rates. In addition, we wish to find a model with greater than 90% Average Precision Score (AP).

3) The methods you use – you must include at least two methods and the performance metrics you use (which should match the objectives stated above) (1 pt)

Four different supervised learning techniques are used for this binary classification problem: Support Vector Classifier, simple Logistic Regression, Random Forest, and Neural Network (simple Neural Network with only one dense hidden layer).

We employ two different rebalancing techniques for each classifier to deal with imbalanced data: random undersampling and SMOTE, are used to process the training data. Finally, the results of various models are evaluated through 5 metrics to get a better picture of classifier behavior: accuracy, precision, recall, average precision score (AP) and F-1 score. Additionally, we plotted the precision-recall curve and confusion matrix for each classifier.

<u>More on Average Precision Score (AP):</u>

[1] https://www.kaggle.com/mlg-ulb/creditcardfraud

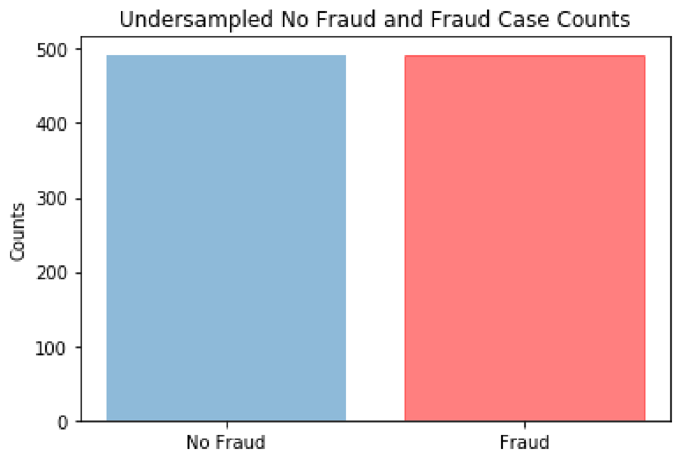
AP summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight:

$$AP = \sum_{n} (R_n - R_{n-1})P_n$$

where P_n and R_n are the precision and recall at the nth threshold. This implementation is not interpolated and is different from computing the area under the precision-recall curve with the trapezoidal rule, which uses linear interpolation and can be too optimistic.[2]

About Random Undersampling:

When class imbalance is present in a dataset, the trained classifiers tend to favor classifying instances into the majority class, which leads to a decrease in the recall of the minority class classification. In addition, noise may be present in features that further degrades classification accuracy[3]. The sheer amount of data in the majority class could indicate the presence of repeating information. Therefore, undersampling tries to leave out some of these "redundant" information in order to rebalance the class ratio. (See challenges section below for drawbacks). In other words, the undersampled dataset is composed of all of the minority class and a portion of the majority class. In order to have a balanced dataset for training, the number of the majority class in the undersampled data is equal to the number of minority class as shown in the figure below (492 fraud cases and 492 non-fraud cases in the undersampled dataset).



Process:

Training and cross validation data: Undersampled dataset composed of 984 datapoints goes went through GridSearchCV to find the optimal hyperparameters to build classifier models. Then the 3 classifiers are trained and validated with a 5-fold cross validation strategy. The undersampled dataset is split into 5 portions, where 4 portions are used as training and the remaining 1 portion is used as validation(test) set.

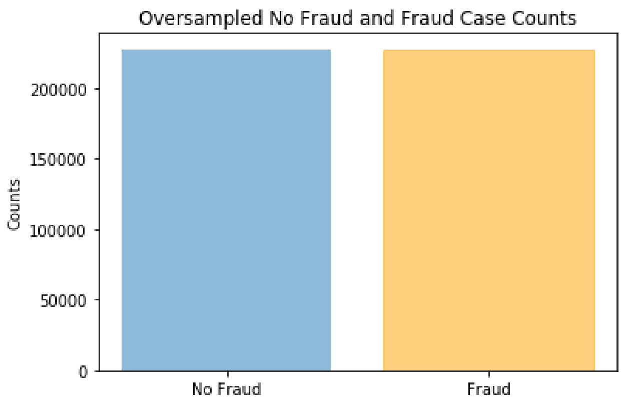
[2] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.
[3] Pozzolo, A.D., & Bontempi, G. (2015). Adaptive Machine Learning for Credit Card Fraud Detection.

Evaluation: The average accuracy, precision score, recall score, F-1 score and average precision score across folds are reported in the results section below. In addition, precision-recall curves and confusion matrices are plotted as well.

About oversampling through SMOTE:

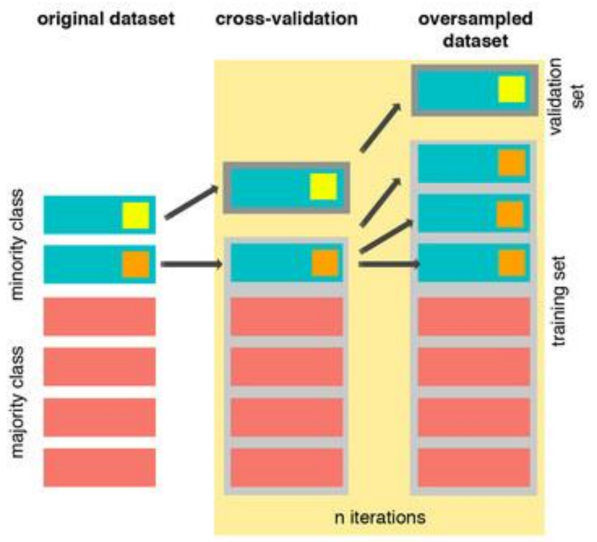
SMOTE, one method of oversampling, stands for Synthetic Minority Over-sampling Technique. It:

- **Solving the Class Imbalance:** SMOTE creates synthetic points from the minority class in order to reach an equal balance between the minority and majority class. (Rikunert, 2017)
- **Location of the synthetic points:** SMOTE picks the distance between the closest neighbors of the minority class, in between these distances it creates synthetic points.
- **Final Effect:** More information is retained since we didn't have to delete any rows unlike in random undersampling.
- **Accuracy verses. Time Tradeoff:** Although it is likely that SMOTE will be more accurate than random under-sampling, it will take more time to train since no rows are eliminated as previously stated.



Process:

Training and cross validation data: We used 227,846 data points (80%) out of 284,807 as training set and the rest 56,961 points (20%) as testing set. By randomly shuffling the data, we put both fraud and non-fraud points into both sets. Then we used SMOTE on the training set to generate synthetic data for fraud case to train the model. The number of oversampled cases can be seen in the figure above. Instead of GridSearchCV, RandomizedSearchCV was used with SMOTE because oversampling required a much longer time for running and RandomizedSearchCV could save some time. One notable thing is that SMOTE occurs "**during**" cross validation and not "prior" to the cross−validation process. Synthetic data are created only for the training set without affecting the validation set as shown by the figure below (Altini, 2015).

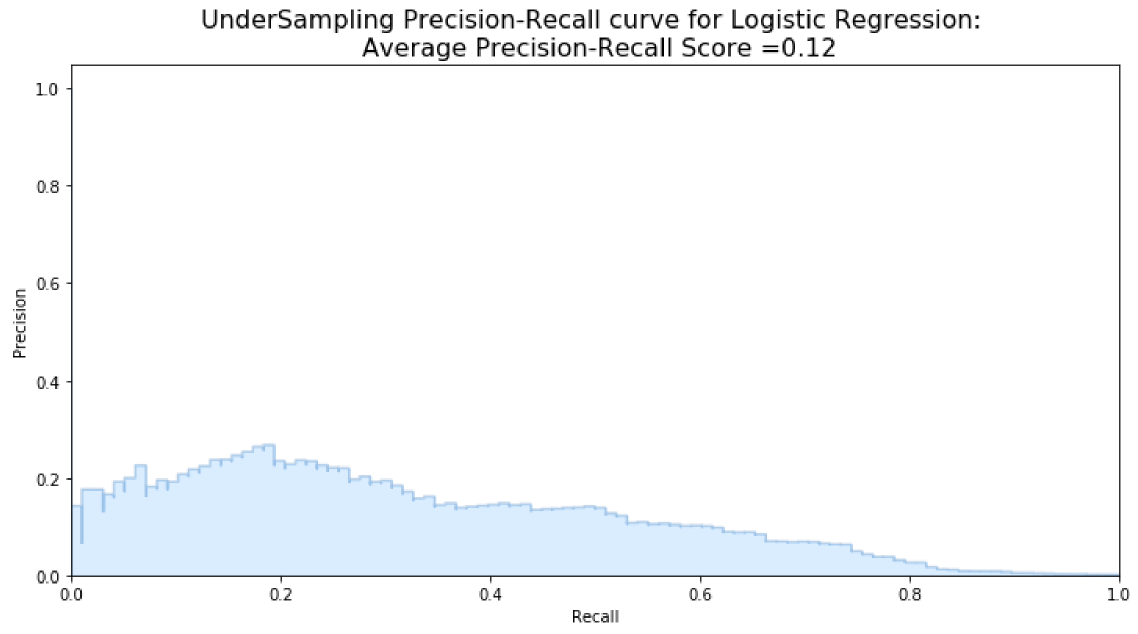


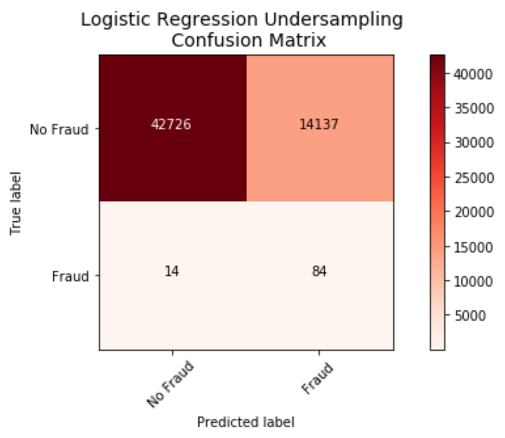
Evaluation: Same as undersampling.

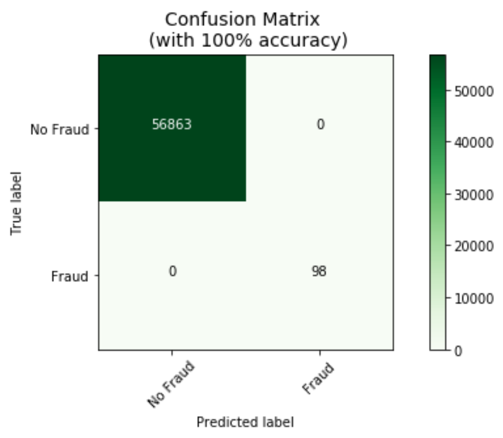
4) The results you obtained – you must have at least two methods and their results (3 pts)

Undersampling:

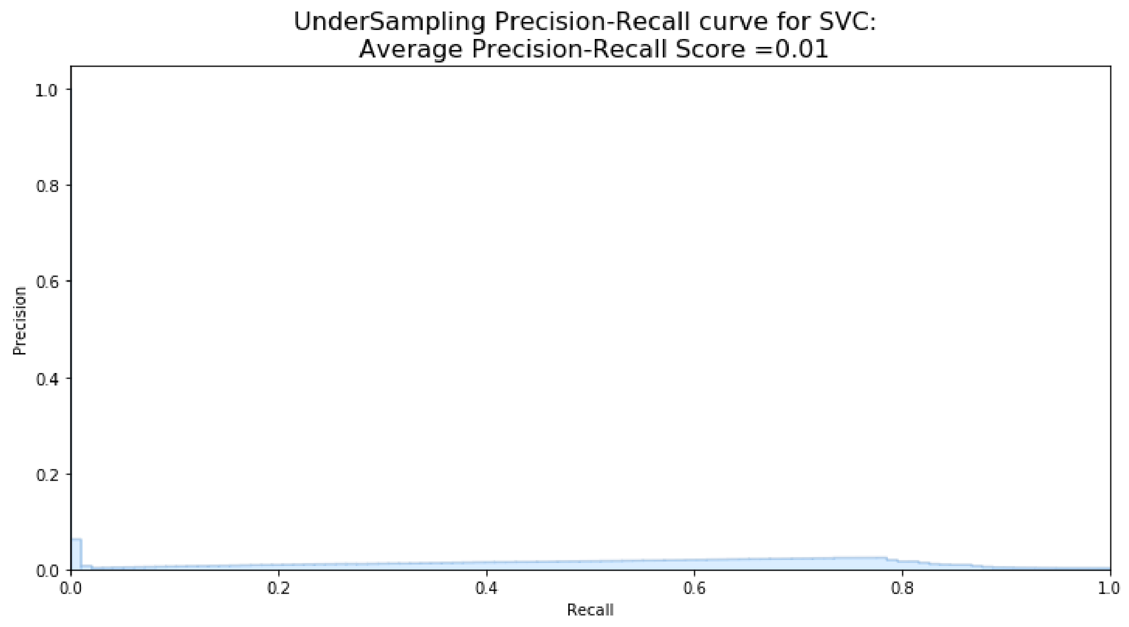
- Logistic Regression**:**

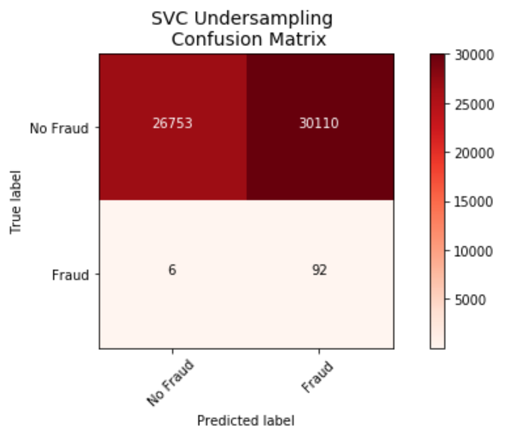


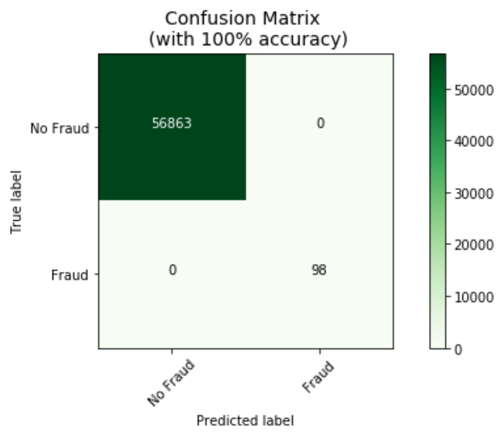

Logistic Regression Undersampling
Confusion Matrix
No Fraud
Fraud
True label
Predicted label
42726
14137
14
84
40000
35000
30000
25000
20000
15000
10000
5000


Confusion Matrix
(with 100% accuracy)
No Fraud
Fraud
True label
Predicted label
56863
0
0
98
50000
40000
30000
20000
10000
0

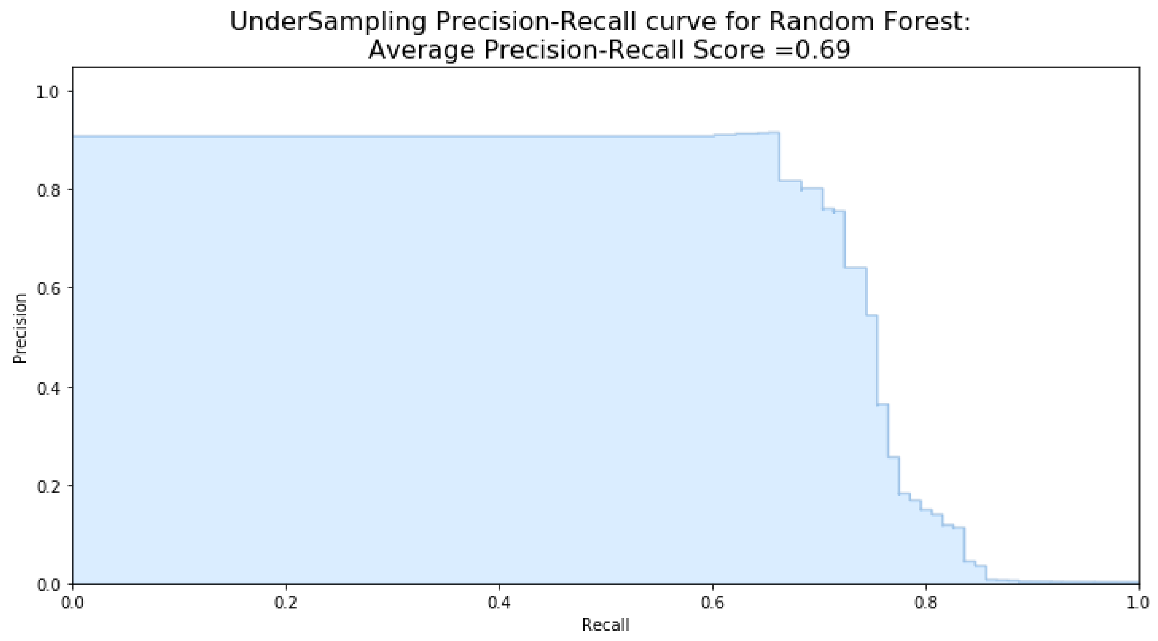
- SVC:

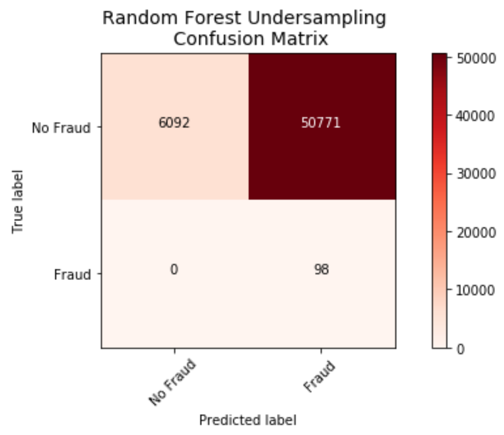

UnderSampling Precision-Recall curve for SVC:
Average Precision-Recall Score =0.01
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

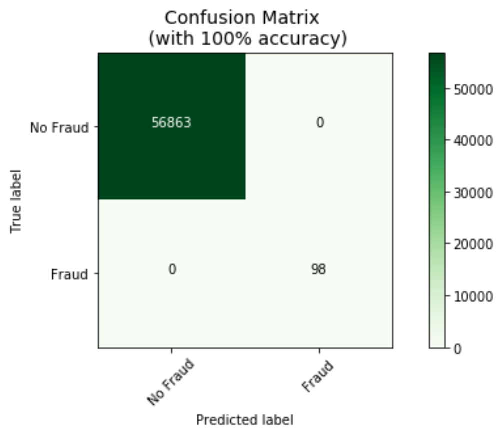

SVC Undersampling
Confusion Matrix
No Fraud
Fraud
True label
Predicted label
26753
30110
6
92
30000
25000
20000
15000
10000
5000


Confusion Matrix
(with 100% accuracy)
No Fraud
Fraud
True label
Predicted label
56863
0
0
98
50000
40000
30000
20000
10000
0

- Random Forest:

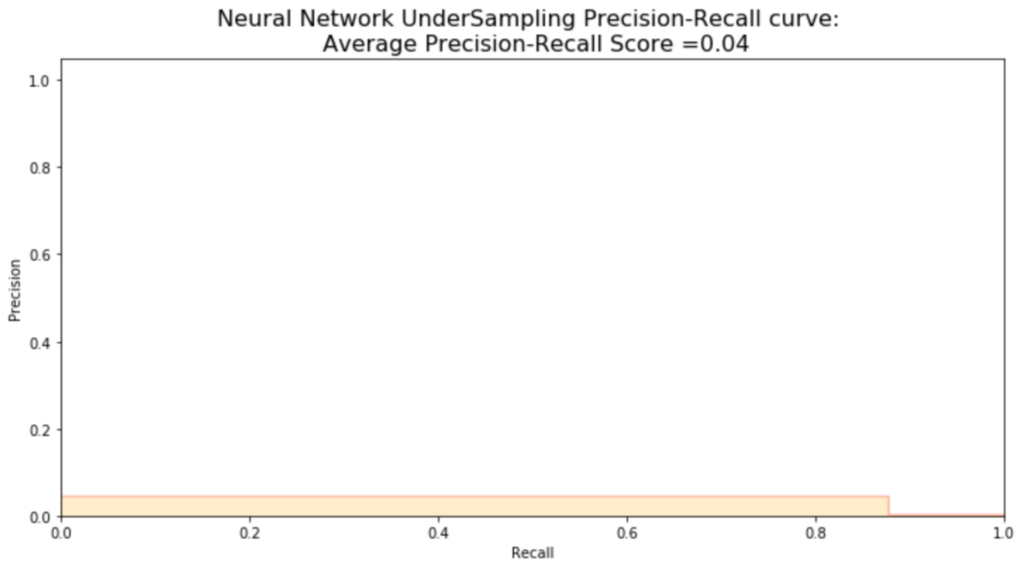


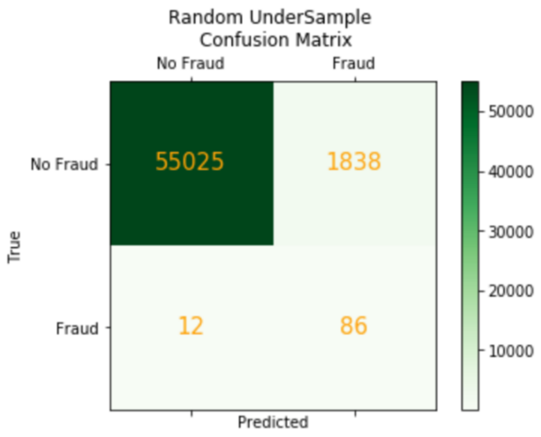


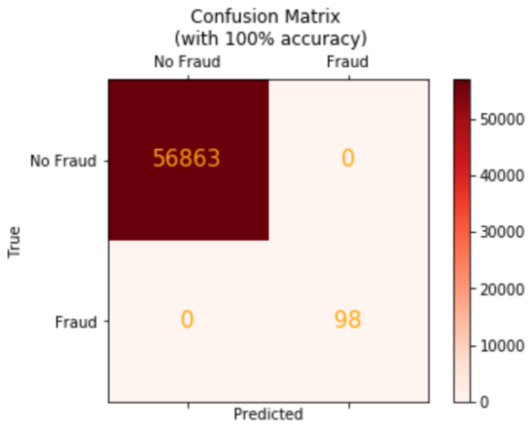


- Neural Network:

```
Average precision-recall score: 0.04
```





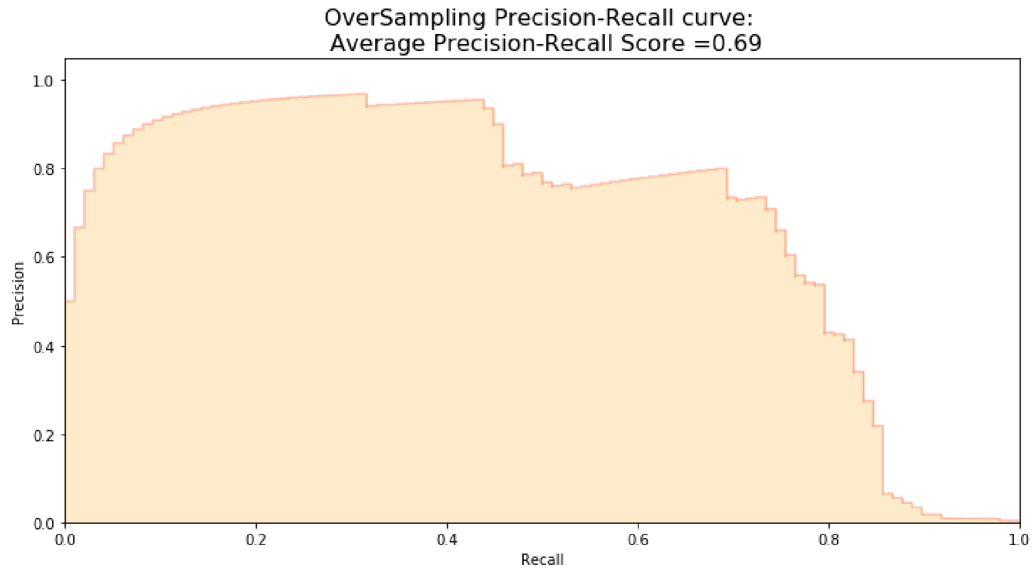


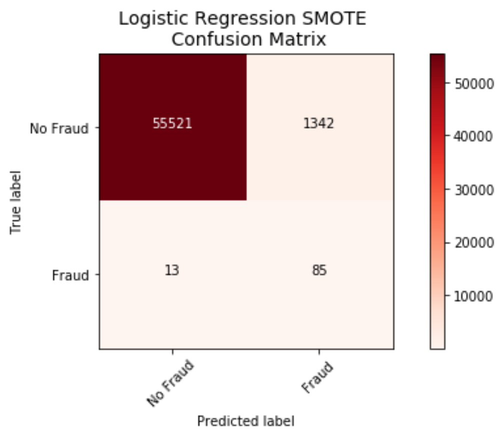
Based on the precision-recall curve, both Logistic Regression and SVC have very low scores compared to Random Forest. Due to the imbalanced nature of the data, many observations could be predicted as false negatives, which means that a non-fraud transaction is predicted, but it is in fact a fraudulent one. Recall score captures the behavior of the classifiers related to these false negatives. Therefore in effort of increasing recall when training(i.e. Increasing proportion of fraud with undersampling so that the classifiers don't overfit to the majority non-fraud class), we sacrificed precision and in turn have a very low average precision score when we moved back to our original larger dataset.

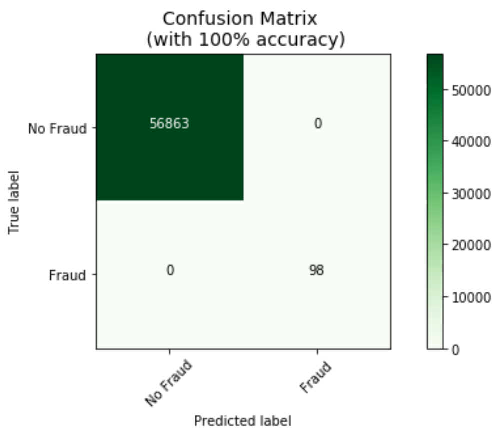
The confusion matrix of the 4 classifiers tells a different story. SVC severely overfitted to fraud cases where it predicted more non-fraudulent cases as fraudulent and have a high false positive rate. Despite having the highest average precision score, Random Forest overfitted to fraudulent cases even worse than SVC. It is able to correctly classify all 98 fraudulent cases correctly but it produced a lot more 50771 cases as false positives (classifying non-fraudulent as fraudulent) as well. On the other hand, Logistic Regression did a decent job at predicting true fraudulent cases (84 out of 98) but has a higher false positive rate (14137 out of 56863) than neural network (1838 out of 56863). Therefore overall, Neural Network performed the best across all metrics.

Oversampling (SMOTE):

- Logistic Regression:

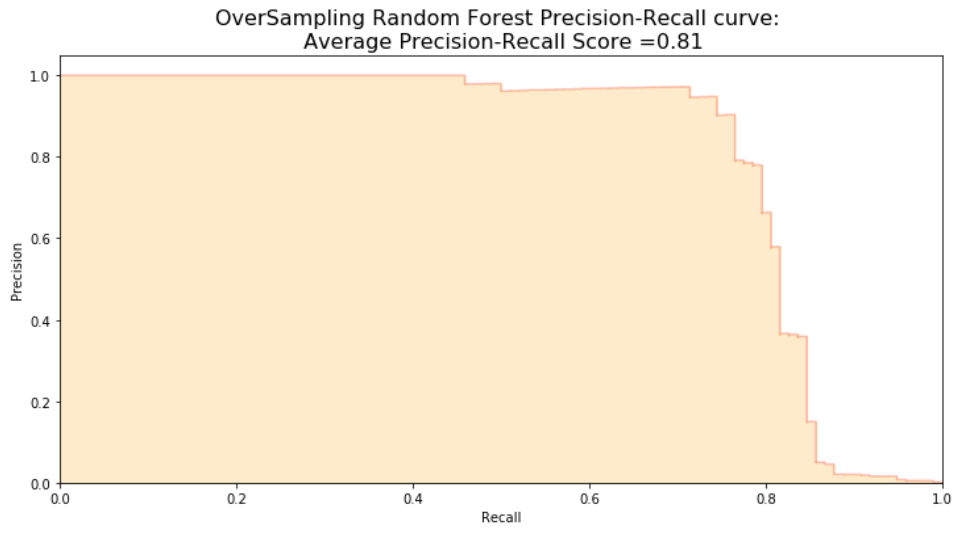


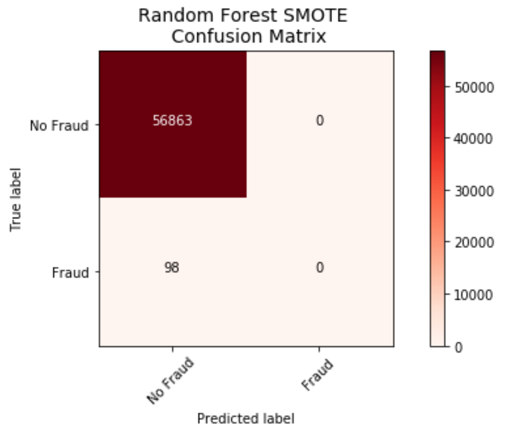


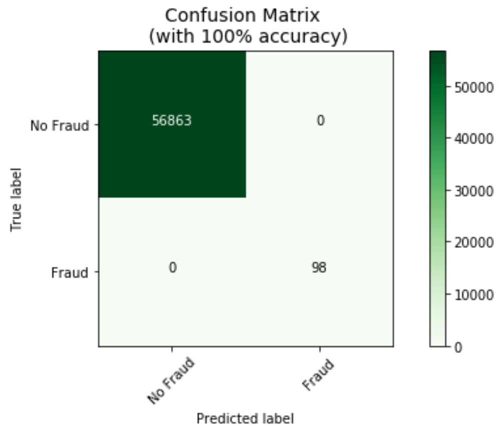


- SVC: Unfortunately the SVC took over 24 hours to run and was an extremely ineffective method in this case, so we gave it up.

- Random Forest:

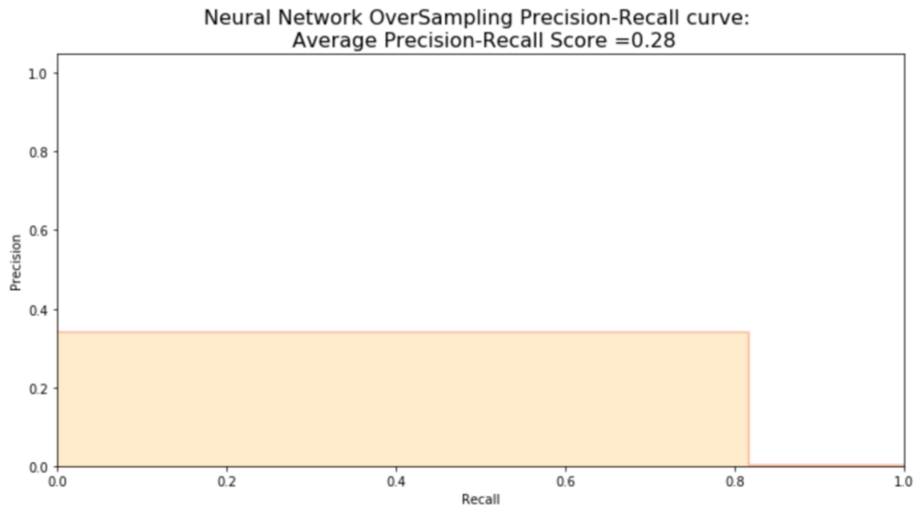


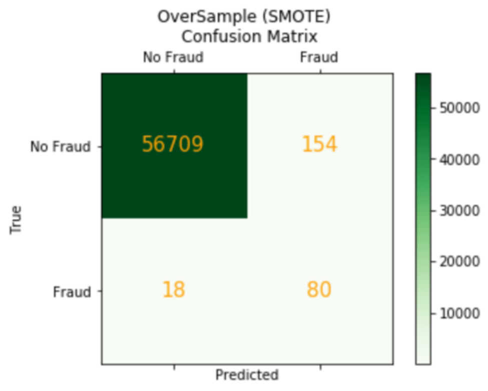


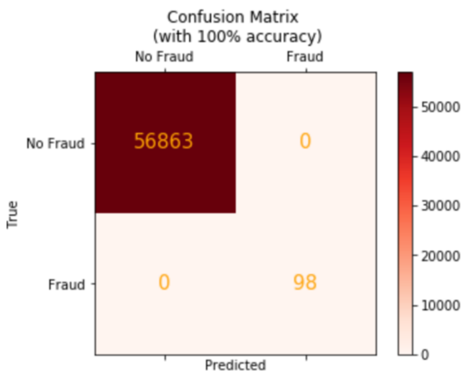


- Neural Network:

```
Average precision-recall score: 0.28
```







The phenomenon of Precision/Recall Tradeoff is also present in the case of oversampled **training** dataset just as in the case of undersampling. The more precise (selective) our model is, the less cases it will detect. The Logistic Regression gained a very low Precision Score (0.055) but a high Recall Score (0.911). In contrast, Random Forest had a balanced Precision Score (0.618) and Recall Score (0.673), and it had a higher accuracy score and a higher average precision-recall score than Logistic Regression. However, if we look into the **testing** dataset's confusion matrix, Random Forest completely failed to detect any fraud case while Logistic Regression successfully detected 85 out of 98. Random Forest obviously had an overfitting problem in the case of SMOTE. Neural Network performed worse than Logistic Regression in terms of identifying fraud transactions (80/98), but better in identifying non-fraudulent cases.

5) A thorough comparison of the results from the 2 methods, using figures and tables if necessary (2 pt)

Our main goal is to explore how our models behave in both the random undersample and oversample data frames and see whether they can predict accurately both non-fraud and fraud cases. We shouldn't emphasize only in detecting fraud cases but also emphasize correctly categorizing non-fraud transactions, because we don't want customers to lose money due to fraud transactions and we also don't want their normal transactions be regarded as fraud ones thus rejected. One thing notable is that predictions and accuracies may be subjected to change since we implemented data shuffling on both types of dataframes.

Precision - **training dataset**	Logistic Regression	SVC	Random Forest
Undersampling	0.000	0.000	0.000
Oversampling	0.055	NA	0.618

Recall - training dataset	Logistic Regression	SVC	Random Forest
Undersampling	0.47	0.60	0.95
Oversampling	0.911	NA	0.673

In terms of the training dataset, precision or recall alone is not a good measurement for the imbalanced data as it does not consider the proportion of the fraud relative to the non-fraud. In the case of undersampling, the three methods listed in the table above all got high recall but low precision, so did logistic regression in oversampling. Such phenomenon is normal in the case of imbalanced data. In contrast, the balanced precision and recall scores of random forest with SMOTE indicate the possible overfitting problem. Therefore we looked directly into the confusion matrix of testing data combined with the average precision score to compare different methods. Below is a summary of testing results.

Avg Precision - testing dataset	Logistic Regression	SVC	Random Forest	Neural Network
Undersampling	0.12	0.01	0.69	0.04
Oversampling	0.69	NA	0.81	0.28

		True Positive	**False Negative**	**True Negative**	**False Positive**	**Accuracy**	**Precision**	**Recall/Sensitivity**	**F-1**	**Specificity**
Benchmark with 100% accuracy		**56863**		**98**						
Under-sampling	Logistic Regression	42726	14137	84	14	0.7516	0.9997	0.7514	0.8579	0.0059
	SVC	26753	30110	92	6	0.4713	0.9998	0.4705	0.6399	0.0030
	Random Forest	6092	50771	98	0	0.1087	1.0000	0.1071	0.1935	0.0019
	Neural Network	55025	1838	86	12	0.9675	0.9998	0.9677	0.9835	0.0447
Over-sampling	Logistic Regression	55521	1342	85	13	0.9762	0.9998	0.9764	0.9879	0.0596
	SVC	NA	NA	NA	NA	NA	NA	NA	NA	NA
	Random Forest	56863	0	0	98	0.9983	0.9983	1.0000	0.9991	0.0000
	Neural Network	56709	154	80	18	0.9970	0.9997	0.9973	0.9985	0.3419

We could see that the first three methods in undersampling were bad at identifying the non-fraud cases, and thus got low accuracy, recall, and F-1 scores. The Random Forest in oversampling completely failed to detect the fraud transactions, but this result was not reflected in the accuracy, precision, recall, or F-1 scores due to the extreme imbalance in the dataset. However, the specificity score shows this problem to some extent. The average precision score should be a good measure for imbalanced dataset but it also failed to show the problem of detecting the fraud with oversampled Random Forest.

Generally, we could conclude from the confusion matrix that the oversampling method outperformed the undersampling method in the credit card dataset, especially in identifying the non-fraud cases (shown both in Logistic Regression and Neural Network), but in exchange it may take a higher risk of failing to detect the fraud cases (shown in Random Forest). Among all four classifiers, Neural Network performed the best compared to others in both undersampling and oversampling data frames and gave stable results in both cases.

6) Your interpretation of the results and insights into the problem(s) you studied (1 pts)

The severe overfitting in random undersampling to one class can be attributed to the drastic downsampling of non-fraudulent cases when we performed undersampling. In an effort to rebalance the dataset through emphasis on the fraudulent cases, the classifiers placed too much "emphasis" on trying to correctly predict fraudulent cases, which leads to a high false positive rate across the board (three out of four classifiers) with the exception of Neural Network, which have a better mechanism (e.g. back propagation) in adjusting hyperparameters and finding a pattern in the data to avoid overfitting. Similar behavior can be observed in application of SMOTE in a lesser degree, where one (Random Forest) out of four classifiers failed to classify any true fraudulent cases. Theses overfitting means that the classifiers applied with undersampling is more likely to report a fraud when there isn't one, which can cause great inconvenience to the customers when their cards get frozen when they are making normal purchases. On the other hand, Random Forest Classifier applied with SMOTE will not report a fraud case when there is one, which will also cause losses to credit card users as well. However, a simple Neural Network seem to work well with either rebalancing technique and is able to correctly classify true frauds and true non-frauds without sacrificing its false positive rate.

The ability to correctly detect and classify fraudulent credit card transactions can be a great tool in the finance industry. Not only can it reduce dispute cases, but also increase card holders' satisfaction. There are many other techniques besides sampling methods that can be employed to deal with binary classifications with an imbalanced dataset such as (Pozzolo & Bontempi, 2015):

- Cost-based methods - consist into sampling training instances from the majority and minority classes by taking into consideration the ratio of the misclassification costs.
- Distance-based methods - make use of distance measures between input points either to undersample or to remove noisy and borderline examples of each class.
- Imbalance learning - for example, a SVM optimized in terms of F-measure is presented by Callut and Dupont (2005).
- Cost-sensitive learning - consider misclassification costs in the learning phase without the need of sampling the two classes.

7) Lessons you learned and what you may do differently if you have to restart (1 pts)

There are several drawbacks to undersampling. First, there is potential risk of removing useful information and thus leading to suboptimal performance; Second, the training sample is only composed of 984 samples, which is rather small for training a robust classifier.

For oversampling/SMOTE, there is a huge risk of overfitting since we are repeatedly adding data/patterns that are specific to this dataset. The overfitting may have a more significant effect on classifier performance on unseen data, especially when the class ratio is big.

Another challenge is finding an appropriate metric in order to quantify performances of our classifiers on such an imbalanced dataset. If we have to restart, we would better structure the start of our project in a way that facilitates research of the most fitting metric in this situation. One well-accepted measures for unbalanced classification in credit card fraud detection could be the rank-based formulation of AUC. We may also use cost-matrix that associates a cost to each entry of the confusion matrix to consider the cost of a missed fraud (FN) compared to the transaction amount.

References:

Altini, M. (2015). Dealing with Imbalanced Data: Undersampling, Oversampling and Proper Cross-validation. Available at: https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation [Access at Dec 01, 2019]

Davis, Jesse & Goadrich, Mark. (2006). The Relationship Between Precision-Recall and ROC Curves. Proceedings of the 23rd International Conference on Machine Learning, ACM. 06. 10.1145/1143844.1143874.

Jérôme Callut and Pierre Dupont. F β support vector machines. *In Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 3, pages 1443–1448. IEEE, 2005.

Pozzolo, A.D., & Bontempi, G. (2015). Adaptive Machine Learning for Credit Card Fraud Detection.

Rikunert. (2017). SMOTE explained for noobs - Synthetic Minority Over-sampling TEchnique line by line. Available at: http://rikunert.com/SMOTE_explained [Access at Dec 01, 2019]

Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.