Exercise Report

Ying Guo

1 Data Descriptions

In the training data file, there are 40,000 cases with 100 features and one target variable coded as 0 and 1. In the holdout data file, there are 10,000 cases with 100 features. The goal of the project is to train two classification models with the training set and predict the holdout data with the trained models.

2 Data Preprocessing, Outliers and Missing Values

There are 96 numerical features and four categorical features in the data.

- Outliers. I applied median-absolute-deviation (MAD) based outlier detection for all numerical features. I used a threshold of 3.5. A data point with Z score whose absolute value larger than 3.5 is labeled as an outlier. I found 1927 instances with outliers (about 5% of all cases). Table 1 shows the outliers for Features x0, x1, and x2 as an example. Without knowing what each feature is, it is hard to decide if these outliers are wrong inputs or valid data point from a distribution. I trained the classification models on the training data with and without outliers. The results are very similar.

- Missing values. There are not many missing values in each feature. The number of missing values ranges from 2 to 16. As there is no content information for each feature, I could not decide why there are missing values. So I used the imputation methods for the missing values. For numerical values, I imputed with the median. I choose median over mean because the median is more robust to outliers. For categorical values, I imputed with the most frequent values. In the analysis, this step is implemented within a pipeline.

- Discussion about missing values:

  - If with content information, I can first decide if it is missing completely at random, which rarely occurs in practice. If yes, the case could be safely dropped.

  - If the missing value is continuous and is missing at random, I would impute with the mean if the fitted model is linear regression, as mean is the unbiased estimator. Here I use median because I still have outliers in the data and median is robust to the outlier.

Table 1: Case ID for outliers in Features x0, x1, and x2. No meaningful explanation could be given at this point without knowing feature content.

Case ID	x0	Case ID	x1	Case ID	x2
615	112.38	479	57.81	615	-20.48
5104	-98.57	2934	-65.14	1251	18.93
6933	114.82	4732	59.69	3209	19.14
7626	-106.81	8701	67.69	9202	-18.44
13033	109.10	11285	64.22	11254	21.06
14564	-98.27	12726	-63.60	19342	-19.36
15335	-99.19	15377	-71.56	19719	-18.52
17172	110.46	15621	57.66	21232	-20.44
19369	108.98	16170	68.17	22393	-18.23
20090	134.59	17900	56.37	22805	19.84
22861	-96.88	18266	-63.36	23208	-21.51
22936	-97.40	19313	-72.86	24798	18.78
23235	-98.23	20338	71.07	29665	20.29
25176	-99.36	20616	58.22	33948	-18.33
27499	-101.46	26278	-67.14	34309	19.54
29091	110.48	27928	59.92	36516	-19.30
29377	-95.88	28095	-64.05	37744	19.13
31163	-101.90	29246	60.22		
31938	108.92	30463	-67.98		
34065	-99.46	33661	63.88		
35440	121.44	36568	-65.51		
35983	-98.86	37590	57.48		
		38443	-70.95		
		38517	56.60		
		39514	-65.19		

– if the missing value is categorical, one could impute value "missing" to make missing values as a category. There are very few cases of missing values (See Table 4 in the Appendix) in the categorical variables. So I impute the missing value with the most frequent class.

– Another good choice for imputation is to use models to predict the missing values. A good example is MICE (Multivariate Imputation via Chained Equations). In Python, FancyImpute has a function IterativeImputer, which is similar to MICE but it returns a single imputation.

3 Exploratory Data Analysis

I plotted histograms for the four categorical features, grouped by the two categories of the target variable y. I also computed the proportion of Class 1 in the target variable for each category for the categorical variables. The proportion varies more for car type, days of the week, and month features, but not for the continent features. I also plotted density graph for the numerical features, grouped by the two categories of the target variable y. See "statefarm-finalcode.pdf" for the plots and tables. It is hard to tell which variables could predict the categories of y. I need to fit models to the training data to get a better picture.

3.1 Multicollinearity

I also want to find out if there is multicollinearity in the feature space, in other words, if there are some independent variables that are highly correlated. With multicollinearity, the coefficient estimates are not robust and become very sensitive to small changes in the model. For all pairs of features in the feature space, the highest absolute correlation is 0.412, suggesting the feature space does not contain highly correlated features.

4 Model Fitting

Now the data is clean. I performed several classification algorithms on the data: Random Forest (RF), Gradient Boosting Machine (GBM), k-Nearest Neighbors (KNN), Logistic Regression using L1 regularization, Logistic Regression with Recursive Feature Elimination, Principal Component Analysis (PCA) with RF, PCA with KNN, PCA with Logistic Regression with L2 regularization. The requirement of the exercise is to report two methods with the best AUC score for the holdout test. Therefore, I report the two methods with the best accuracy scores and AUC scores in detail: Random Forest and Gradient Boosting Algorithm (see Table 2 for details).

4.1 SMOTE Resampling

Before I talked about the model fitting, I want to mention that the training data is an imbalanced dataset. There are 8,120 cases of Class 1 and 31,880 cases of Class 0 in the target variable. This is

Table 2: Accuracy scores and AUC scores for each algorithm. "SMOTE" means whether SMOTE is used, "Outlier" means if the outlier is removed, "Dim. Reduction" means if the algorithm is able to select feature and reduce dimensionality, "Accuracy" means accuracy score, and "AUC "means AUC score. The two algorithms in boldface are reported with predictions for holdout data.

Model	SMOTE	OUTLIER	Dim. Reduction	Accuracy	AUC
Random Forest	Yes	No	Yes	0.925	0.975
Random Forest	Yes	Yes	Yes	0.922	0.978
Random Forest	No	No	Yes	0.873	0.984
Gradient Boosting Algorithm	Yes	N0	Yes	0.955	0.981
Gradient Boosting Algorithm	No	No	Yes	0.953	0.983
Knearest Neighbor	Yes	No	No	0.468	0.866
Knearest Neighbor	No	No	No	0.926	0.960
Logistic Regression (L1 regularization)	No	No	Yes	0.891	0.909
Logistic Regression(RFE)	No	No	Yes	0.891	0.907
PCA + RF	No	No	Yes	0.965	0.982
PCA + RF	Yes	No	Yes	0.966	0.982
PCA + Logistic Regression (L2)	Yes	No	Yes	0.832	0.905
PCA + Logistic Regression (L2)	No	No	Yes	0.885	0.904
PCA + KNN	No	No	Yes	0.915	0.961

an imbalanced dataset, and the ratio of Class 1 to Class 0 instances is about 20:80 or more concisely 1:4. We have accuracy paradox with an imbalanced dataset. It is the case where the accuracy measures show that one has excellent accuracy (such as 90%), but the accuracy is only reflecting the underlying class distribution. So I used an oversampling method, Synthetic Minority Over-sampling Technique (SMOTE). In this analysis, for each classification methods, I fitted the model with and without SMOTE. In the end, I report the two models that have the best accuracy scores and AUC scores.

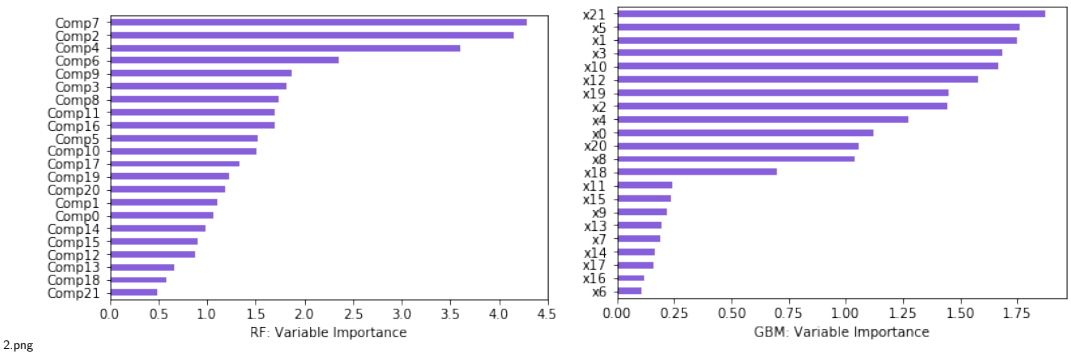
4.2 Pipeline

Before fitting the model, I split up the data to 70% *training data* and 30% test data, randomly without replacement. Then I created a pipeline. First, it preprocesses the data. For the numerical feature, it imputes the missing value with the median for that feature and then standardizes it; for the categorical feature, it imputes the missing value with the most frequent class and gets dummy variables. Secondly, it creates a balanced sample with SMOTE. If using PCA, the pipeline uses SMOTE resampling right after the running of PCA. Thirdly, it fits the classification model on the *training data*, while tuning the hyperparameter with five-fold cross-validation. For the result, I report the classification accuracy score (the number of correct predictions made by the model over all kinds of predictions made), and AUC score. For RF and GBM, I also plotted the variable importance plots.

The two boldfaced algorithms in Table 2 are used to predict the holdout data, because they have the best accuracy scores and AUC scores. The following sections report details for each of those two best algorithms.

4.3 Random Forest with PCA

I built the RF classifier on the reduced dimensions provided by PCA, tuning hyper-parameters with grid search on the number of components for PCA, the number of trees, quality of the split criteria, number of features to consider when looking for the best split for RF, with 5-fold cross-validation. SMOTE was performed right after PCA. The best performing model selects 110 components from PCA, and uses 500 trees, entropy as split criteria, and the square root of the total number of features as the number of features to consider when looking for the best split for RF. The accuracy score is 0.965, and the ROC AUC score is 0.9824. The good thing about RF is that it provides the variable importance. The variable importance plot shows that the top 3 most important components are Comp7, Comp2, and Comp4 shown by the left figure of Figure 1. But it is tricky to explain what each component represents, because each component resulting from PCA is a combination of the original features. The coefficients for each feature in a given component could be outputted. Table 3 shows ten features with the highest absolute coefficients in Comp7 and Comp2, which are the two most important components shown by the variable importance plot. Further understanding of the components depends on knowing feature contents.



2.png

Figure 1. Left: The Variable Importance Plot from Random Forest with PCA; Right: The Variable Importance Plot from GBM.

Table 3: Ten features with the highest absolute coefficients for Comp7 and Comp2

Comp7		Comp2	
Feature	Coeff	Feature	Coeff
x8	-0.3074	x5	0.3234
x22	0.2805	x90	-0.2730
x97	-0.2612	x53	-0.2701
x61	0.2572	x50	0.2544
x51	-0.2565	x43	-0.2436
x40	-0.2563	x66	-0.2157
x4	-0.2170	x75	0.2119
x95	-0.2168	x77	-0.2108
x63	0.2074	x47	-0.2041
x77	-0.2019	x71	0.2003

The total number of features is 126 after getting dummy variables for categorical features. One thing to note is that PCA selects 110 components out of 126 features, suggesting it does not combine many features and does not reduce the dimensionality a lot. This result makes sense since we have a very low correlation among features, which means independence between features.

4.4 Gradient Boosting Machine (GBM)

I built Gradient Boosting Machine and used grid search on hyperparameter learning rate, maximum depth of the individual estimators, number of features to consider when looking for the best split, with 5-fold cross-validation. No SMOTE is performed. The best performing GBM uses 0.2 learning rate, maximum depth of the individual estimators as 8, and the square root of the total number of features as the number of features to consider when looking for the best split. The accuracy score is 0.953, and the ROC AUC score is 0.9834. The variable importance plot shows that the top 5 most important features are x21, x5, x1, x2 and x10, shown by the right figure of Figure 1.

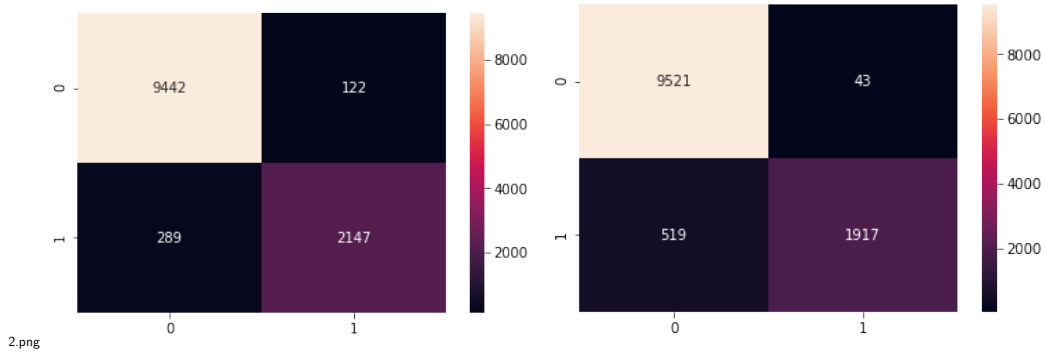


Figure 2. Left: The Confusion Matrix from Random Forest with PCA;
Right: The Confusion Matrix from GBM.

5 Random Forest vs. Gradient Boosting Machine

Confusion matrix comparison: as shown in Figure 2, the true positive predicted by RF with PCA is higher than GBM (here I define classifying a case to Class 1 when it is actually Class 1 as positive). GBM has higher false negatives and lower false positives, compared to RF with PCA. Depending on the goal of the classification task, if false negatives are far less adverse than false positives, we want to use GBM who has lower false positive. After knowing the nature of the problem, we can decide which algorithm work better.

Similarity: both Random Forest and Gradient Boosting Machine are non-parametric methods. This means that they have no assumptions about the space distribution and the classifier structure. They are both ensemble learning methods and make predictions by combining the outputs from individual trees. Both Random Forest and GBM require less data cleaning and is not influenced by outliers and missing

values to a fair degree. Both of them can handle a large data set with higher dimensionality. They can handle many input variables and identify the most significant variables. They are considered as methods for dimensionality reduction. Moreover, both methods output variable importance. However, both methods are very easy to overfit the data. Both Random Forest and GBM can suffer from skewed class distributions. I used SMOTE with Random Forest and GBM for the training dataset. However, SMOTE does not improve the accuracy score or AUC score for GBM. So I did not predict the holdout data using SMOTE for GBM.

Difference: the biggest difference is that GBM trains trees sequential and RF trains independently. GBMs are more sensitive to overfitting if the data is noisy. Training for GBM generally takes longer. In general, GBM is harder to tune than RF.

6 Prediction

I use the fitted model to predict the holdout data. I performed the Kolmogorov-Smirnov Test on each numerical feature from the training and holdout data sets. It turns out that p values for x15, x27, x51, x60, x63, x71, 78 are smaller than 0.05. However, since the sample size here is very large (40,000 and 10,000), even a very small difference in the two samples might result in a significant p-value. By looking at the data summary statistics of those features (see Table 5 for some examples), I cannot find clear evidence of those data are from a different population. Thus, it is safe to use the fitted model to predict the holdout data. The predicted probability for each class is reported in "result1.csv" (PCA with RF) and result2.csv" (GBM).

7 Appendix

Table 4: The frequency for each category of all four categorical features

x34		x35		x68		x93	
volkswagon	12557	wednesday	20756	Jul	11146	asia	35434
Toyota	10922	thursday	17726	Jun	9289	america	3136
bmw	7288	tuesday	898	Aug	8115	euorpe	1423
Honda	5195	friday	550	May	4833	NaN	7
tesla	2286	monday	59	Sep	3441		
chrystler	1209	NaN	11	Apr	1638		
nissan	339			Oct	886		
ford	160			Mar	397		
mercades	26			Nov	160		
chevrolet	10			Feb	52		
NaN	8			Dec	20		
				Jan	12		
				NaN	11		

Table 5: Sample summary statistics for Features x0 - x19 in both training and holdout data

Training	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9
count	39989	39989	39993	39991	39992	39989	39993	39988	39996	39992
mean	6.16	-3.57	0.22	-1.74	0.08	-0.54	0.02	-0.01	-3.06	-0.02
std	29.10	17.19	5.24	36.60	21.18	13.60	4.11	2.42	13.45	2.47
min	-106.81	-72.86	-21.51	-157.57	-79.90	-55.05	-15.96	-9.30	-54.42	-9.67
25%	-13.62	-15.15	-3.30	-26.47	-14.22	-9.77	-2.77	-1.64	-12.06	-1.68
50%	6.25	-3.66	0.26	-1.64	0.11	-0.53	0.02	0.00	-3.07	-0.04
75%	25.57	7.81	3.76	23.04	14.37	8.67	2.77	1.62	5.91	1.64
max	134.59	71.07	21.06	145.57	89.86	52.63	18.55	11.92	54.26	9.49
Holdout	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9
count	9997	9999	9998	9996	10000	10000	9996	9999	9997	9999
mean	6.24	-3.91	0.22	-1.97	0.49	-0.64	-0.04	0.00	-2.72	-0.02
std	29.01	17.04	5.25	36.39	20.95	13.86	4.11	2.42	13.46	2.47
min	-111.49	-72.42	-22.33	-139.60	-75.91	-52.45	-13.75	-8.71	-55.03	-9.68
25%	-13.49	-15.25	-3.33	-26.95	-13.65	-10.04	-2.81	-1.63	-11.73	-1.70
50%	6.06	-4.16	0.27	-1.81	0.55	-0.57	-0.02	0.01	-2.83	-0.04
75%	25.36	7.63	3.75	22.71	14.50	8.77	2.69	1.61	6.29	1.63
max	130.30	56.09	18.33	135.28	79.67	55.59	20.50	9.20	68.55	9.34
Training	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19
count	39990	39994	39988	39989	39997	39994	39993	39990	39987	39993
mean	5.23	0.01	-0.43	-0.02	-0.03	-0.03	-0.04	-0.04	6.64	0.22
std	28.34	1.53	1.71	8.55	8.49	9.95	8.80	7.39	26.95	31.85
min	-112.35	-6.04	-7.42	-36.42	-34.50	-40.66	-38.32	-29.01	-100.19	-129.18
25%	-13.77	-1.02	-1.57	-5.80	-5.75	-6.76	-5.98	-5.06	-11.42	-21.37
50%	5.31	0.00	-0.43	-0.06	-0.03	-0.02	-0.03	-0.03	6.78	-0.06
75%	24.32	1.03	0.72	5.80	5.69	6.66	5.89	4.99	24.78	21.58
max	121.68	6.40	6.66	35.21	35.07	48.85	30.96	28.61	105.80	126.51
Holdout	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19
count	9999	9997	10000	9994	9998	9997	9998	9997	9998	9998
mean	5.32	0.00	-0.42	0.07	0.08	0.23	-0.05	0.07	6.88	-0.23
std	27.99	1.53	1.71	8.53	8.48	9.83	8.75	7.44	26.95	31.80
min	-118.81	-6.12	-7.33	-29.79	-35.00	-35.13	-33.35	-26.29	-101.90	-110.74
25%	-13.49	-1.01	-1.57	-5.70	-5.53	-6.39	-5.93	-4.87	-10.99	-21.94
50%	5.57	0.01	-0.42	0.07	0.10	0.36	-0.14	0.11	6.78	-0.54
75%	24.27	1.03	0.73	5.83	5.76	6.84	5.88	5.05	24.80	20.68
max	107.37	5.72	6.77	33.42	34.62	37.32	32.04	28.50	103.33	131.14