Hallmark -Data Analytics-exercise

Mandeep Narang

1. Dataset Collection

I have used Python a scripting language for this exercise and all of my work has been conducted and performed using Jupyter notebook. Downloaded the Zip file consisting 100 .csv files in my computer from the provided link. Imported them one by one with a loop and generated one big file by consolidating all the 100 files in one file. Found 200,000 rows and 464 columns in the final dataset.

Data Description:

Columns in the dataset do not have any meaningful name except one column “target”. We have column names from “v2: v462”, “X”, “target”, “text”. “text” column seems to be having some description of transaction. “target” is the variable we need to classify, has two type of values (0 /1). Both numerical(continues) and categorical type of data is present in the columns

Data Exploration:

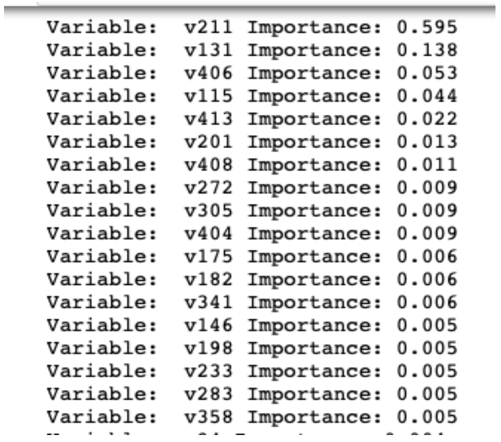
This is a huge dataset for my system, so I decided to find some important features first and then explore them. Though I performed some little check on the data. Missing value check: did not find any missing value, Outliers check: when I tried to remove the outliers from all the columns there was no rows left behind. So, I decided to move forward with this data.

2. Data Preparation

2.1. Feature Selection using Feature Importance

Categorical variables are always thought to shortlist. In this case I do not have any idea of domain of the data, so I cannot remove any categorical variable by its name/significance. Other way is to convert the categorical variables to Dummy variables and then test their significance and importance on "target" statistically

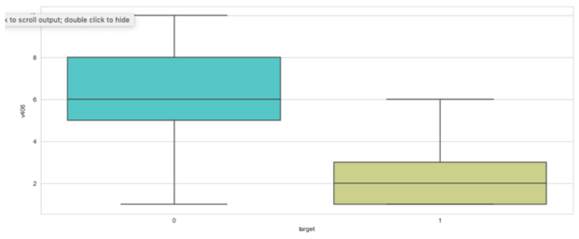
Since there are 462 predictors, I attempted to select the features before any correlation analysis and made a pandas table to view the top features with their importance in classification of label using Random Forest Classifier.



```
Variable:  v211 Importance: 0.595
Variable:  v131 Importance: 0.138
Variable:  v406 Importance: 0.053
Variable:  v115 Importance: 0.044
Variable:  v413 Importance: 0.022
Variable:  v201 Importance: 0.013
Variable:  v408 Importance: 0.011
Variable:  v272 Importance: 0.009
Variable:  v305 Importance: 0.009
Variable:  v404 Importance: 0.009
Variable:  v175 Importance: 0.006
Variable:  v182 Importance: 0.006
Variable:  v341 Importance: 0.006
Variable:  v146 Importance: 0.005
Variable:  v198 Importance: 0.005
Variable:  v233 Importance: 0.005
Variable:  v283 Importance: 0.005
Variable:  v358 Importance: 0.005
```

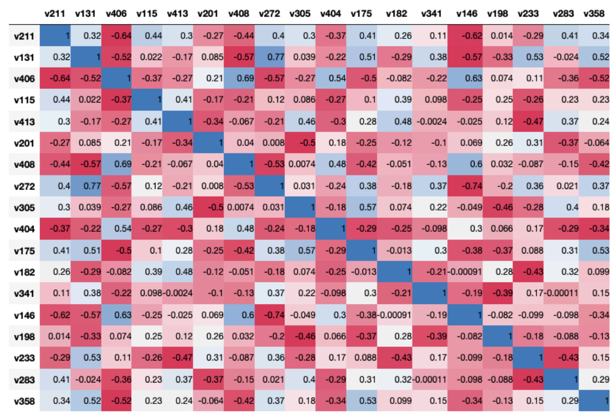
With these above top features, a new data frame is created which has been explored to find distribution of the data.

In Data Exploration we found some interesting insights from the visualizations. For example:



we can see from the above image that how 1 is influenced by lower values of the variable.

2.2. Correlation Analysis



	v211	v131	v406	v115	v413	v201	v408	v272	v305	v404	v175	v182	v341	v146	v198	v233	v283	v358
v211	1	0.32	-0.64	0.44	0.3	-0.27	-0.44	0.4	0.3	-0.37	0.41	0.26	0.11	-0.62	0.014	-0.29	0.41	0.34
v131	0.32	1	-0.52	0.022	-0.17	0.085	-0.57	0.77	0.039	-0.22	0.51	-0.29	0.38	-0.57	-0.33	0.53	-0.024	0.52
v406	-0.64	-0.52	1	-0.37	-0.27	0.21	0.69	-0.57	-0.27	0.54	-0.5	-0.082	-0.22	0.63	0.074	0.11	-0.36	-0.52
v115	0.44	0.022	-0.37	1	0.41	-0.17	-0.21	0.12	0.086	-0.27	0.1	0.39	0.098	-0.25	0.25	-0.26	0.23	0.23
v413	0.3	-0.17	-0.27	0.41	1	-0.34	-0.067	-0.21	0.46	-0.3	0.28	0.48	-0.0024	-0.025	0.12	-0.47	0.37	0.24
v201	-0.27	0.085	0.21	-0.17	-0.34	1	0.04	0.008	-0.5	0.18	-0.25	-0.12	-0.1	0.069	0.26	0.31	-0.37	-0.064
v408	-0.44	-0.57	0.69	-0.21	-0.067	0.04	1	-0.53	0.0074	0.48	-0.42	-0.051	-0.13	0.6	0.032	-0.087	-0.15	-0.42
v272	0.4	0.77	-0.57	0.12	-0.21	0.008	-0.53	1	0.031	-0.24	0.38	-0.18	0.37	-0.74	-0.2	0.36	0.021	0.37
v305	0.3	0.039	-0.27	0.086	0.46	-0.5	0.0074	0.031	1	-0.18	0.57	0.074	0.22	-0.049	-0.46	-0.28	0.4	0.18
v404	-0.37	-0.22	0.54	-0.27	-0.3	0.18	0.48	-0.24	-0.18	1	-0.29	-0.25	-0.098	0.3	0.066	0.17	-0.29	-0.34
v175	0.41	0.51	-0.5	0.1	0.28	-0.25	-0.42	0.38	0.57	-0.29	1	-0.013	0.3	-0.38	-0.37	0.088	0.31	0.53
v182	0.26	-0.29	-0.082	0.39	0.48	-0.12	-0.051	-0.18	0.074	-0.25	-0.013	1	-0.21	-0.00091	0.28	-0.43	0.32	0.099
v341	0.11	0.38	-0.22	0.098	-0.0024	-0.1	-0.13	0.37	0.22	-0.098	0.3	-0.21	1	-0.19	-0.39	0.17	-0.00011	0.15
v146	-0.62	-0.57	0.63	-0.25	-0.025	0.069	0.6	-0.74	-0.049	0.3	-0.38	-0.00091	-0.19	1	-0.082	-0.099	-0.098	-0.34
v198	0.014	-0.33	0.074	0.25	0.12	0.26	0.032	-0.2	-0.46	0.066	-0.37	0.28	-0.39	-0.082	1	-0.18	-0.088	-0.13
v233	-0.29	0.53	0.11	-0.26	-0.47	0.31	-0.087	0.36	-0.28	0.17	0.088	-0.43	0.17	-0.099	-0.18	1	-0.43	0.15
v283	0.41	-0.024	-0.36	0.23	0.37	-0.37	-0.15	0.021	0.4	-0.29	0.31	0.32	-0.00011	-0.098	-0.088	-0.43	1	0.29
v358	0.34	0.52	-0.52	0.23	0.24	-0.064	-0.42	0.37	0.18	-0.34	0.53	0.099	0.15	-0.34	-0.13	0.15	0.29	1

These selected features do not seem to be correlated enough to cause collinearity issues. Hence, I decide to go forward without eliminating or treating the columns.

Note: In the above correlation matrix “v272” is positively related with “v131” and “v146” is negatively corelated with “v211”. There could be a possibility of endogeneity or multicollinearity but I do not have any domain knowledge so cannot remove any one of them.

2.3. Data Sampling

Before using the data in the learning algorithms, the data was divided into two parts named Train and Test. Train data was 75% of the total data and rest was Test. This data was selected randomly

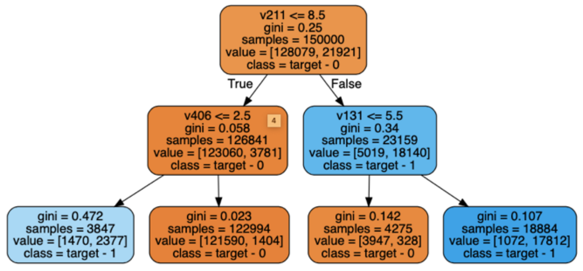
2.4. Classification model selection

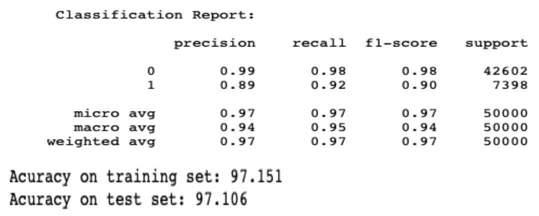
Since our goal is to classify the target variable and interpret the directionality of the individual variables on ‘target’ we can use any of the classification algorithm. I wanted to cross validate my results, so I decided to perform two powerful algorithms Decision Tree and Logistic Regression on the data

3. Model & Results:

3.1. Decision Tree:

I wanted to make decision tree flexible and automatically changeable with changing the criteria, so I defined one function for tree and used that function in interactive package function to plot tree. By running the model, I found the following results:

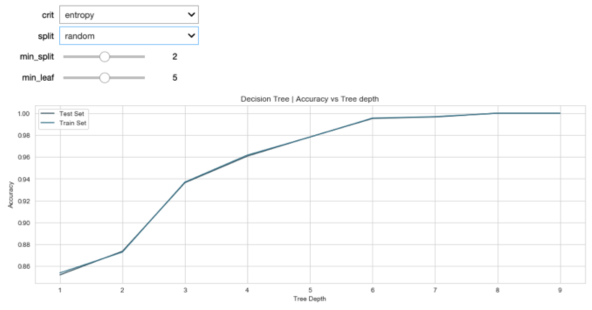




```
     Classification Report:

                  precision    recall  f1-score   support

               0       0.99      0.98      0.98     42602
               1       0.89      0.92      0.90      7398

       micro avg       0.97      0.97      0.97     50000
       macro avg       0.94      0.95      0.94     50000
    weighted avg       0.97      0.97      0.97     50000

Acuracy on training set: 97.151
Acuracy on test set: 97.106
```

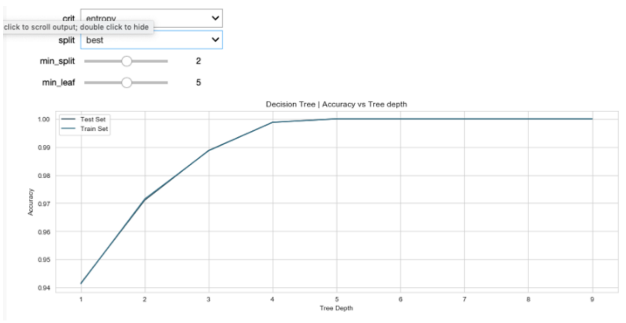
Interpretation of Tree:
We can see that our model is performing very finely it is showing 97% of accuracy for both training and test data.
I used classification report to check the precision of the model. From the above tree we can say that for any new entry if “v211” >8.5 and “v131”>5.5 then the target will be in class 1.
If for any new entry “v211” <=8.5 and “v406” >2.5 then it will be in class 0.

Model Fitting and optimization:
To optimize the Decision tree we use learning curve with different criteria.



crit: entropy
split: random
min_split: 2
min_leaf: 5

In this curve we are taking split criteria as random we can see that our tree provides approx. 99% accuracy at tree depth 6. But by changing this criterion we can see that if we choose split type best then our algorithm provides 99% accuracy at tree depth 4.

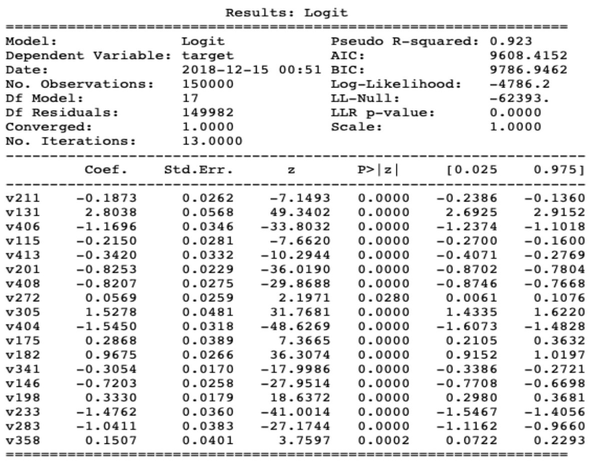


crit: entropy
split: best
min_split: 2
min_leaf: 5

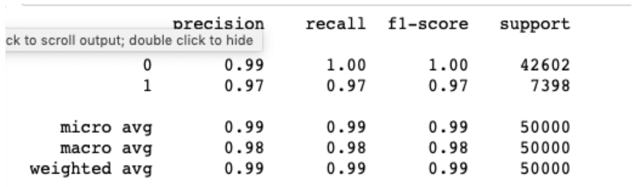
We can see that our model is neither under nor over fit to the test data. This is a generalized model, it is performing very well on both seen and unseen data.

3.2. Logistic Regression

The logistic regression is very powerful algorithm in case of feature direction interpretation. I trained the logistic regression on train data and then predict the results on the test data.



```
                         Results: Logit
==================================================================
Model:              Logit            Pseudo R-squared: 0.923
Dependent Variable: target           AIC:              9608.4152
Date:               2018-12-15 00:51 BIC:              9786.9462
No. Observations:   150000           Log-Likelihood:   -4786.2
Df Model:           17               LL-Null:          -62393.
Df Residuals:       149982           LLR p-value:      0.0000
Converged:          1.0000           Scale:            1.0000
No. Iterations:     13.0000
-------------------------------------------------------------------
         Coef.    Std.Err.      z       P>|z|    [0.025    0.975]
-------------------------------------------------------------------
v211    -0.1873    0.0262    -7.1493   0.0000   -0.2386   -0.1360
v131     2.8038    0.0568    49.3402   0.0000    2.6925    2.9152
v406    -1.1696    0.0346   -33.8032   0.0000   -1.2374   -1.1018
v115    -0.2150    0.0281    -7.6620   0.0000   -0.2700   -0.1600
v413    -0.3420    0.0332   -10.2944   0.0000   -0.4071   -0.2769
v201    -0.8253    0.0229   -36.0190   0.0000   -0.8702   -0.7804
v408    -0.8207    0.0275   -29.8688   0.0000   -0.8746   -0.7668
v272     0.0569    0.0259     2.1971   0.0280    0.0061    0.1076
v305     1.5278    0.0481    31.7681   0.0000    1.4335    1.6220
v404    -1.5450    0.0318   -48.6269   0.0000   -1.6073   -1.4828
v175     0.2868    0.0389     7.3665   0.0000    0.2105    0.3632
v182     0.9675    0.0266    36.3074   0.0000    0.9152    1.0197
v341    -0.3054    0.0170   -17.9986   0.0000   -0.3386   -0.2721
v146    -0.7203    0.0258   -27.9514   0.0000   -0.7708   -0.6698
v198     0.3330    0.0179    18.6372   0.0000    0.2980    0.3681
v233    -1.4762    0.0360   -41.0014   0.0000   -1.5467   -1.4056
v283    -1.0411    0.0383   -27.1744   0.0000   -1.1162   -0.9660
v358     0.1507    0.0401     3.7597   0.0002    0.0722    0.2293
==================================================================
```



```
              precision    recall  f1-score   support
ck to scroll output; double click to hide
           0       0.99      1.00      1.00     42602
           1       0.97      0.97      0.97      7398

   micro avg       0.99      0.99      0.99     50000
   macro avg       0.98      0.98      0.98     50000
weighted avg       0.99      0.99      0.99     50000
```

Interpretation of Logistic regression and feature directionality:
With one unit of any feature increase the odds of obtaining label = 1 increase by exponential of coef of the feature.

- With one unit increase in “v211”, the odds of obtaining label =1 increase by approx. 0.83
- With one unit increase in “v131”, the odds of obtaining label =1 increase by approx. 16.5
- With one unit increase in “v406”, the odds of obtaining label =1 increase by approx. 0.3
- With one unit increase in “v115”, the odds of obtaining label =1 increase by approx. 0.8
- With one unit increase in “v413”, the odds of obtaining label =1 increase by approx. 0.7
- With one unit increase in “v201”, the odds of obtaining label =1 increase by approx. 0.4

3.3. Model Comparison:

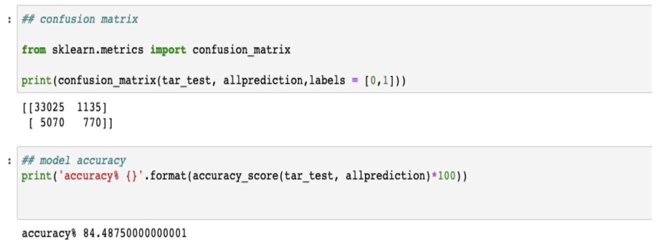
There are several ways to compare models, but we can see the accuracy on both the models are close to 99. In this exercise I used default version of Decision tree and logistic regression and they are showing very satisfying results, so I did not think of boosting the trees or using gradient descends in logistic regression.

In the first step tree was showing 97% accuracy but after optimizing tree we can achieve up to 99% with tree depth 4 and split type best. So that tree will be equally accurate to logistic regression.
 We can also compare their confusion matrix and see how they are similar and how they are different.
More over in this case I can say that we can use any of these models to classify.

4. Textual Data Association and importance:

There is one ‘text’ field associated with each entry, our goal is to check if the 'text' column could be helpful in classifying the 'target' variable. We cannot use this ‘text’ into any model and also cannot convert all of them to dummy variables because the unique number of ‘text’ records are around (31357), creating this much of dummy variable will not be a justice to the model. Otherwise to take this column into account we can convert this column text to numbers using TfidfVectorizer algorithm. I am not going to directly include this in my classification model, but we can see if there is some association between two fields i.e., ‘text’ and ‘target’ then we can say including this would be a better step. So, to see association if we use and MultinomialNB to predict the 'target' using those ‘text’ (after converting them to numbers) only and if we see a significant level of accuracy in the classification model then we can say that it is a useful field and we can improve our previous model by adding this field into account.

Model Fitting and optimization:



```
## confusion matrix

from sklearn.metrics import confusion_matrix

print(confusion_matrix(tar_test, allprediction,labels = [0,1]))
```

```
[[33025  1135]
 [ 5070   770]]
```

```
## model accuracy
print('accuracy% {}'.format(accuracy_score(tar_test, allprediction)*100))
```

```
accuracy% 84.48750000000001
```

We can see Jupyter file for coding part but above results shows that out model is giving 84.5% accuracy, and this is a good number for justification.
So we can say that adding this field will be a better step towards classification of ‘target’