Machine Learning Engineer Nanodegree

Telstra Network Disruptions Capstone Write Up

Srinivasa Amirapu
July 31th, 2018

The task was to predict the severity of service disruptions on Telstra's network. It was a 3-class classification problem evaluated on multiclass logloss. Here I'll share my approach to solving this problem.

1) **Understanding Data and Features:**
   Features are extracted from log files and other sources: event_type.csv, log_feature.csv, resource_type.csv, severity_type.csv. All above features are categorical except for "volume". The data set is in a relational format, split among multiple files. The following provides a description of data in each file.

- Event Type Data
- Log Feature Data
- Resource Type Data
- Severity Type Data
- Training Data
- Testing Data

Training Data

Data Fields	Definition
ID	identifies a unique location-time point
Location	identifier of location
Fault_Severity	categorical. 0: no fault, 1: a few faults, 2: many faults

- Event Type Data

Data Fields	Definition
ID	identifies a unique location-time point
EventType	type of event that occured at that ID (can be multiple events per ID)

- Log Feature Data

Data Fields	Definition
ID	identifies a unique location-time point
Log_Feature	type of feature logged for that ID
Volume	number of times the feature was logged for that ID

- Resource Type Data

Data Fields	Definition
ID	identifies a unique location-time point
Resource_Type	type of resource assocaited with that ID

- Severity Type Data

Data Fields	Definition
ID	identifies a unique location-time point
Severity_Type	type of severity level logged for that ID

1.1) Exploratory Data Analysis

1) Chi-square test is performed to understand correlation between categorical variables Analysis shows location and id are strongly correlated and we can drop one of them.

```
array([[14121,   118],
       [14121,   118],
       [14121,   118],
       ...,
       [17067,   885],
       [17067,   885],
       [17067,   885]], dtype=int64)
```

2) Spearman correlation test is performed to understand correlation between categorical vs numericals.

	id	location	fault_severity	event_type	resource_type	severity_type	log_feature	volume
id	1.000000	-0.027298	-0.035930	0.019441	-0.013694	0.027083	-0.010111	-0.002106
location	-0.027298	1.000000	0.271742	-0.383049	0.429917	-0.424742	-0.304589	0.071210
fault_severity	-0.035930	0.271742	1.000000	-0.261497	0.290473	-0.360594	-0.238632	-0.046634
event_type	0.019441	-0.383049	-0.261497	1.000000	-0.596874	0.414419	0.523528	0.091495
resource_type	-0.013694	0.429917	0.290473	-0.596874	1.000000	-0.392140	-0.491933	-0.043653
severity_type	0.027083	-0.424742	-0.360594	0.414419	-0.392140	1.000000	0.382624	0.126415
log_feature	-0.010111	-0.304589	-0.238632	0.523528	-0.491933	0.382624	1.000000	0.043322
volume	-0.002106	0.071210	-0.046634	0.091495	-0.043653	0.126415	0.043322	1.000000

2) Data Preparation

Step 1: Import Modules

Import all the python,numpy,pandas and scikit-learn modules

Step 2: Import Datasets

Each row in the main dataset (train.csv, test.csv) represents a location and a time point. They are identified by the "id" column, which is the key "id" used in other data files. Fault severity has 3 categories: 0,1,2 (0 meaning no fault, 1 meaning only a few, and 2 meaning many). “fault_severity” is a measurement of actual reported faults from users of the network and is the target variable (in train.csv).

Step 3: Data preprocessing: Data merging to create a single Customer Analytics Record (CAR)

	id	event_type	resource_type	severity_type	log_feature	volume
0	6597	event_type 11	resource_type 8	severity_type 2	feature 68	6
1	8011	event_type 15	resource_type 8	severity_type 2	feature 68	7
2	2597	event_type 15	resource_type 8	severity_type 2	feature 68	1
3	5022	event_type 15	resource_type 8	severity_type 1	feature 172	2
4	5022	event_type 15	resource_type 8	severity_type 1	feature 56	1

Step 4: Data Cleansing to remove text from variables

Define Cleanse() method to strip off text from variables

Step 6: Drop "fault_severity" from train dataset as it is the target variable

	id	location
0	14121	location 118
1	9320	location 91
2	14394	location 152
3	8218	location 931
4	14804	location 120

	id	location
0	14121	118
1	9320	91
2	14394	152
3	8218	931
4	14804	120

Step 7: Merge the train dataframe without the "fault_severity" column and the combined dataframe of "event_type ... etc"

	id	location	event_type	resource_type	severity_type	log_feature	volume
0	14121	118	34	2	2	312	19
1	14121	118	34	2	2	232	19
2	14121	118	35	2	2	312	19
3	14121	118	35	2	2	232	19
4	9320	91	34	2	2	315	200

Step 8: Convert Categoricals using Get_Dummies

Step 9: groupby "id"

	id	0	1	2	3	4	5	6	7	8	...	1572	1573	1574	1575	1576	1577	1578	1579	1580	vol
0	14121	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	76
1	9320	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	632
2	14394	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	4
3	8218	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	44
4	14804	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	96

3) **Split data** using train_test_split method from sklearn.model_selection
Dataset is slightly imbalanced across class labels 0,1,2. I choose stratify parameter.In this context, stratification means that the train_test_split method returns training and test subsets that have the same proportions of class labels as the input dataset.

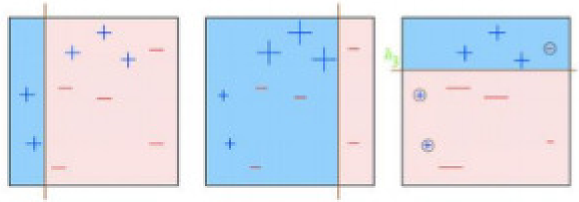
3) **Model Selection**:

A wide class of models have been used for Network Disruption Prediction

- Generalized Linear Models
- Support Vector Machines
- K-Nearest Neighbors
- Naïve Bayes
- Decision Trees and Ensemble Methods like AdaBoost and Gradient Boosting

Among Ensemble Methods, I have chosen Gradient Boosting Classifier as benchmark model because of its better performance and accuracy.

Boosting is a sequential technique which works on the principle of ensemble. It combines a set of weak learners and delivers improved prediction accuracy.



5) Model Training and Tuning: Train the gradient boost classifier model using optimal parameters (best fit model) obtained from GridSearchCV technique.Scores obtained for unoptimized model vs optimized model are below :

Unoptimized model:

Accuracy score on testing data: 0.7133

F-score on testing data: 0.5864

Optimized Model:

Final accuracy score on the testing data: 0.7233

Final F-score on the testing data: 0.6018

We can further tune and improve the F-score by tuning parameters which can be divided into 3 categories:

- ***Tree-Specific Parameters***: These affect each individual tree in the Model

Min_samples_split, min_samples_leaf,

min_weight_fraction_leaf, max_depth, max_leaf_nodes,

max_features

- ***Boosting Parameters***: These affect the boosting operation in the model

Learning_rate, n_estimators, subsample

• ***Miscellaneous Parameters***: Other parameters for overall functioning Loss, init, random_state, verbose, warm_start, presort

6) Measure accuracy:

It would seem that using **accuracy** as a metric for evaluating a particular model's performance would be appropriate. But this is a multi-class classification problem where model's ability to precisely predict fault severity at a location is *more important* than the model's ability to **recall** those locations. We can use **F-score** as a metric that considers both precision and recall:

Below is the classification report generated for the optimized model obtained above :

	precision	recall	f1-score	support
fault_severity_0	0.76	0.95	0.84	4784
fault_severity_1	0.66	0.32	0.43	1871
fault_severity_2	0.67	0.48	0.56	726
avg / total	0.73	0.74	0.71	7381

7) Predicted Probabilities for Classes 1,2,3: Calculate predicted values along with probabilities for test data set.

	id	predict_0	predict_1	predict_2
0	11066.0	0.810560	0.100919	0.088521
1	18000.0	0.533543	0.145241	0.321216
2	16964.0	0.813410	0.097757	0.088832
3	4795.0	0.545782	0.328023	0.126195
4	3392.0	0.378244	0.440102	0.181654

References:

1) https://www.kaggle.com/c/telstra-recruiting-network

2) https://www.telstra.com.au/consumer-advice/customer-service/mass-service-disruption

3) http://gereleth.github.io/Telstra-Network-Disruptions-Writeup/