Timothy Lee
Machine Learning Engineer Nanodegree
April 11, 2017

Capstone Report:
Predicting Red Hat Business Value

Define the Problem

Project Overview

Per the Harvard Business Review, "deep insights from customer big data should enable highly skilled employees to be more creative and free up time to connect with customers in new ways that add value." Thus, there's a large demand for data science in identifying likely customers for a company based on their internet activity. This is already happening all around us, with the clearest example being recommendations for certain products based on what we click in Google search. The side panels are filled with advertisements that coincide with links we've clicked in the past; these are called targeted advertisements. This is an example of using one's past activity/experiences to drive future purchases. Prior to this new field of data mining, companies had to broadcast advertisements to an impractically large audience and hope that their message reached the likely buyers for their products. This is expensive and there's no guarantee that it would work. Instead, companies can now hire data scientists and engineers to efficiently focus their ad campaigns and intentionally target their products towards the best possible customers.

Article: https://hbr.org/2016/08/using-data-to-strengthen-your-connections-to-customers

Another example of using data mining on a person's activities to identify their likely outcome is with Goldman-Sachs's handling of TransUnion, a credit-reporting firm they purchased in 2012. They used this company to sell seemingly insignificant characteristics of people to lenders, insurers, etc. to profit nearly $600 million so far. This shows the powerful predictive power of machine learning and big data in the real world.

Article: https://www.wsj.com/articles/how-goldman-sachs-made-more-than-1-billion-with-your-credit-score-1491742835

Problem Statement

The problem being investigated is how to be more efficient in targeting ads towards likely customers; more specifically, how to use a user's internet activities to predict whether they would buy a product or service. Most companies have data regarding the people who browse their products online, but do not

yet have the technology to properly analyze and derive meaning from it. This is a problem because there is a plethora of insight about marketing and customers to be gained from that data.
In the words of the actual Kaggle competition that will be used for this Capstone, participants are tasked with the problem of creating "a classification algorithm that accurately identifies which customers have the most potential business value for Red Hat based on their characteristics and activities. With an improved prediction model in place, Red Hat will be able to more efficiently prioritize resources to generate more business and better serve their customers."

Metrics

Because our problem involves binary classification and the dataset's outcomes are a balanced mix of positive and negative results, the Area Under the ROC Curve (sklearn.roc_auc_score, or AUC score) will be used as the evaluation metric to compare to the benchmark Random Forest model.
ROC curves are great for binary classification visualizations. They plot the False positive rate against the true positive rate for every possible classification threshold, which is the value at which you separate a positive and negative classification. A classifier that separates classes well will have a larger area under the ROC curve.

Analyze the Problem

Data Exploration

Link to dataset: https://www.kaggle.com/c/predicting-red-hat-business-value
This data is from a past Kaggle competition sponsored by a company called Red Hat, a software and services company that uses a subscription model.

The following descriptions are information from the Kaggle competition page.
They provided two files: a people file and an activity file. The people file contains each unique person and their characteristics. The activity file has all the unique activities that a person has performed over time. There is a Boolean business value outcome column in the activity file that indicates whether that specific activity by the user resulted in the user fulfilling Red Hat's desired outcome.

The activities are labelled as a certain 'activity_id' but the meaning of the IDs are hidden to us. However, Red Hat does indicate activities as Type 1 through Type 7. Type 1 activities have more information (i.e. 9 characteristics) whereas Type 2-7 activities only have 1 characteristic.

There are 498,687 activities in the test activity file (act_test.csv) and 2,197,291 activities in train activity file (act_train.csv) for 189,118 people in the people file (people.csv).

Exploratory Visualization

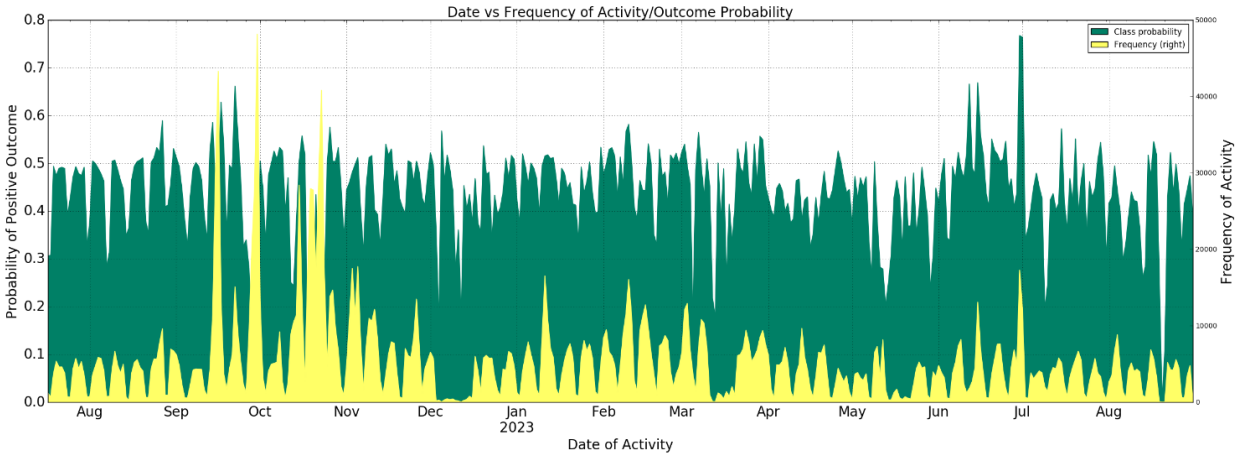

Figure 1. An area plot of Activity Date vs. Activity Frequency/Outcome Probability for the Red Hat training data.

According to Figure 1, you can see that there is a consistent pattern week to week. On the weekends, there is an obvious drop off in both activity frequency and the probability for a positive outcome. Additionally, on the weekdays there is more activity and an increased chance of positive outcome. This is very understandable as businesses aren't likely to make decisions on the weekends when nobody is working. Also, because Red Hat largely caters to business-to-business services, the activity is consistent with a Monday to Friday schedule. This observation also led to adding a new feature called 'isweekend', which will be mightily effective in helping to predict the outcome of activity.

Additionally, you can see a large increase in activity frequency from mid-September to late-November. This is probably because most businesses begin their fiscal years around that time, which is an obvious time to begin researching and subscribing to new services like Red Hat.

Relevant code can be found in visualization.ipynb.

Algorithms and Techniques

The proposed solution is to use supervised learning on the given Red Hat data to develop an algorithm that can accurately predict whether a given person and her activities will have high potential business value. Supervised learning uses training data to predict a target variable. The training data for our context is the data provided by Red Hat and the target variable in the data is 'outcome'. With this properly trained algorithm, Red Hat should be able to accurately identify whether a customer has potential business value.

Because there is a large amount of data that is well structured but diverse, I will be using XGBoost gradient boosted trees as the supervised learning classification algorithm to develop a solution for this problem. This is because there are up to 9 characteristics for each activity which indicates that a tree classifier is ideal for classification. In addition, the activities are very diverse (i.e. Type 1 vs. Type 2) so boosting is necessary to reduce error.

XGBoost is a software library for gradient boosted decision trees that focuses on speed and performance, and appropriately stands for Extreme Gradient Boosting. Its algorithm has three key performance features:

1. Sparse aware: automatically handle missing data values
2. Block structure: parallelization of tree construction
3. Continued training: ability to further boost an already fitted model with new data

Essentially, the algorithm implements gradient boosting decision trees. Boosting is an ensemble technique where an existing model is made more robust by sequentially adding new models to it as additive improvements. The newer models focus on the more troublesome data points that are misclassified on earlier models. Gradient boosting is similar in that it is additive to existing models but it also considers the error of prior models and optimizes their loss with newer models. XGBoost is a great candidate for our problem because of the number of features and the size of the dataset.

Benchmark

To objectively validate my results, I will be using a Random Forest model as a benchmark model against my XGBoost model. This is a less robust and simple algorithm that will show that the gradient boosted trees are either more or less effective for this problem. This classifier works by taking subsets of the dataset and features, building decision trees on each one, and averaging the results to improve accuracy and reduce overfitting.

This benchmark model was performed on the Red Hat data and the area under the curve score was .967. This result was achieved using 100 as the n_estimators parameter for the Random Forest Classifier. This parameter indicates the number of trees to use in each of the Decision Trees performed on the subsets. The value of the parameter was found using GridSearchCV. This classifier uses a parameter grid of multiple values and the model in question (i.e. Random Forest) to find the most effective parameter values. I passed 10, 100, and 1000 into the parameter grid and found 100 to be the most effective value for n_estimators.

Although this benchmark score was decent, I believe the XGBoost model will perform significantly better.

Relevant code can be found in analysis_nb.ipynb.

Implement Algorithms

Relevant code can be found in redhat_nb.ipynb.

Data Preprocessing

1. The first necessary preprocessing step was to import the data from the multiple files and merge them using the people_id key. This allows us to work with a single dataset with characteristics from both the person and activity in our feature set.

2. Next, the characteristics were either filled with a value of format 'Type X' (where X is an integer), a date of format YYYY-MM-DD, or as a Boolean variable (i.e. true or false). To make this data easier to work with, the following was performed:
    a. The 'Type X' characteristics with null values were filled with 'Type 0'
    b. The value of all 'Type X' columns were replaced with their extracted digit
        i. E.g. Type 1 would be replaced with 1
    c. The Boolean columns were replaced with 0s and 1s (i.e. 0 for False and 1 for True)
    d. The date columns were split into separate year, month, and day columns. Additionally, an 'isweekend' column was added, as the visualization earlier indicated a strong correlation between weekend activity and negative outcomes.
3. Other empty values in columns outside of characteristics were filled with 'NA'
4. To reduce dimensionality of outliers (i.e. unique values in categorical columns), those unique values in each categorical column were replaced with 9999999. This is to simplify the size of the feature set when one hot encoding.
5. The non-feature columns, people_id and activity_id, were dropped from the dataset, as they were irrelevant for creating a predictive model
6. The dataset was split into training and validation sets using sklearn.cross_validation.train_test_split()
7. The categorical columns were transformed into numerical 0/1 columns using sklearn.preprocessing.OneHotEncoder
8. Use scipy.sparse.hstack to horizontally stack the categorical (one hot encoded) and non-categorical columns for the training and validation sets
9. Create XGBoost DMatrices for the stacked training and validation sets and their outcome labels

Implementation

The XGBoost model was trained using the optimized parameters and the DMatrix with training data. The training was performed using 300 rounds and with an early stopping parameter of 5.

The model was first run without a specified booster, so it defaulted to the 'gbtree' booster. This resulted in a AUC score of .954 which wasn't better than the Random Forest benchmark score of .967.

Relevant code can be found in gbtree.ipynb.

Refinement

Firstly, GridSearchCV was performed on the XGBoost model to find the best parameter values. Because the sheer length of time involved in running this technique, only max_depth was optimized to 11. This parameter was chosen because it controls overfitting by limiting the maximum depth of each tree.

Relevant code can be found in gridsearch.ipynb.

To improve upon the poor score, the booster parameter value was changed to 'gblinear'. This resulted in an AUC score of .997, which is well above the benchmark model and highly accurate. This new booster is a regularized linear model that uses linear regression techniques to solve categorical problems instead

of continuous ones. This is still a great a candidate for our problem because our outcome response variable has two categories: yes and no.

Collect Results

about the performance of the models used, visualize significant quantities, and validate/justify these values

Model Evaluation and Validation

- eta – .02, learning rate
- silent – 1, no running messages printed
- objective – 'binary:logistic', the loss function to be minimized
    - binary classification with logistic regression
    - This measures the performance of the model given a certain set of parameters. This is the function that is being optimized.
- nthread – 4, number of cores to run script on
- eval_metric – 'auc', metric used for validation
    - Area Under the ROC Curve Score
- subsample – .8, the ratio of the data to use for each tree
- colsample_bytree – .8, subsample ratio of columns to construct tree
- booster – 'gblinear', the function to use for boosting
    - This function is a generalized linear model that uses multiple iterations of training to build a strong classifier. Latter iterations focus more on misclassified training data to additively build a robust solution model.
- max_depth – 11, maximum depth of a tree
    - a larger value creates more complex trees that are more prone to over fitting

Justification

Percent Improvement:

(0.996772831138 - 0.966989848292) * 100 / 0.996772831138

= 2.987940874351204

= 3% increase in AUC score

The XGBoost model had a 3% better AUC score than the Random Forest model.

Construct Conclusions

Free-Form Visualization

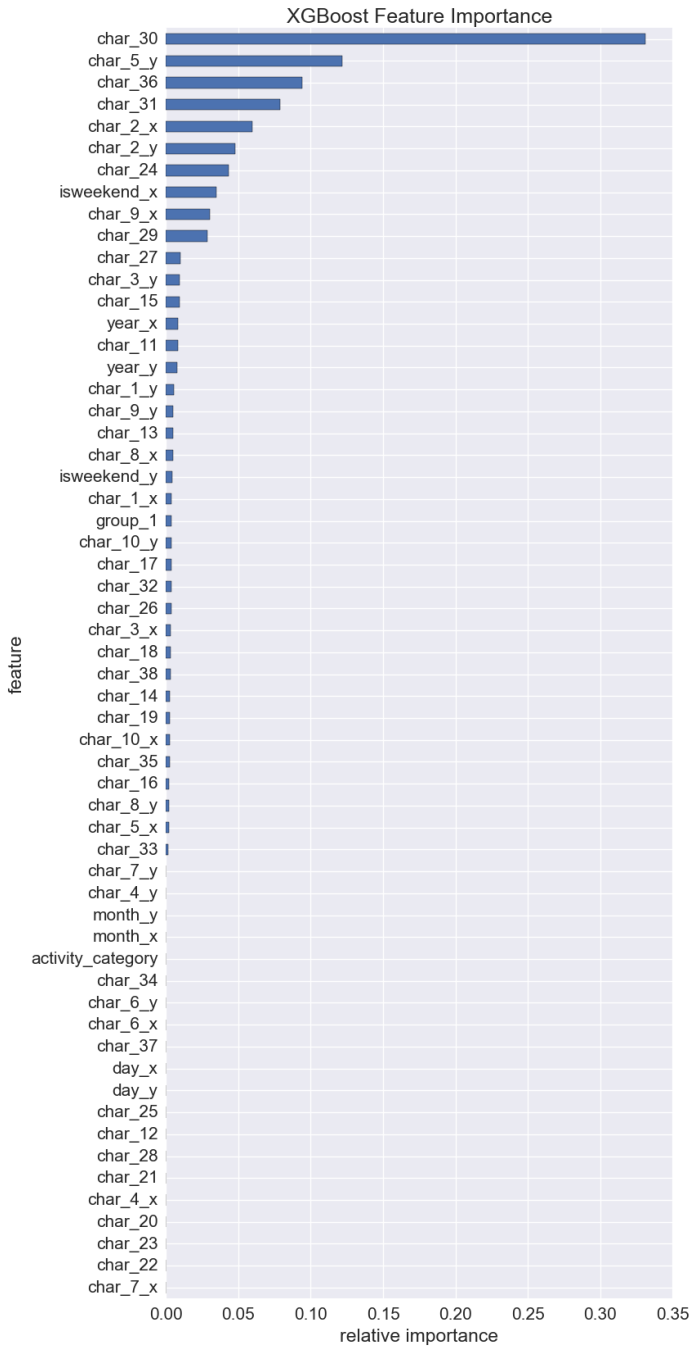

Figure 2. A horizontal bar plot of feature importance.

The XGBoost model using the 'gblinear' Booster parameter was difficult to extract feature importance from. For the visualization portion of the results, the 'gbtree' Booster was used instead. This is a more traditional tree based model that allows individual tree importance to be analyzed, albeit with a lower AUC ROC score. The score for this model was .954.

Figure 2 shows the relative importance of the features. The top four features were from the activity characteristics: char_30, char_5, char_36, and char_31. This shows that a person's actions are more indicative of their choices than their personal characteristics.

Of the top ten features, only 3 features were from the person characteristics: char_2, is_weekend, and char_9. This confirms that our initial discussion of adding the is_weekend feature is indeed a significant factor in predicting Red Hat outcomes.

Relevant code can be found in gbtree.ipynb.

Reflection

This problem started with deidentified people and activity data. The purpose was to use the given characteristics of an activity and its associated person to predict the Red Hat outcome.

Before fitting any classifiers to the data, the data was transformed using various techniques and wrapped with a DMatrix to pass to the XGBoost model.

GridSearchCV was used to find the best parameter values to pass to the model.

The training data was used to train the XGB model. The validation data was used to calculate the AUC score from the trained model.

An aspect I found interesting is that there is a single characteristic (char_30) that has exponentially more predictive power than the rest of the features. Red Hat did not mention what this characteristic is, but the existence of a powerful one activity that can significantly predict a successful outcome is unexpected.

Improvement

One interesting aspect of the project I found difficult was to use GridSearchCV on multiple parameters at once. The computing power and time required to run this model on a multi combination parameter grid was overwhelming and couldn't be completed. Therefore, the optimal parameter values are still unknown although the AUC score is still very nearly 1.0.