School Budget Line Item Classification

A Multi-class, Multi-label Classification Problem

Milestone Report for Capstone Project

Springboard, DSCT

01Jun2018

Mark Sausville
saus@ieee.org

School Budgets: Multi-class, multi-label classification, Capstone Milestone Report

Overview

Budgets for schools and school districts are huge, complex, and unwieldy. Education Resource Strategies (ERS) is a non-profit that analyzes school budgets using human expert analysts with the goal of letting districts be more effective in their spending.

The task is to correctly label every budget line item in nine different classifications with appropriate labels for each class. These labels allow ERS to understand how schools are spending money and tailor their strategy recommendations to improve outcomes for students, teachers, and administrators.

An effective machine learning model would let human analysts complete work much faster by eliminating a substantial amount of time-consuming hand analysis and allow ERS to serve its core client base (public schools and school districts) more effectively.

The data consists of 14 columns of descriptive free-form text data and 2 numerical columns related to the nature of the expenditure.

The project is a multi-class, multi-label classification problem with challenges in text processing, feature design and feature engineering, model selection and model tuning. It also has the advantage of participation in an active competition, so models can be objectively judged for quality.

The data is hosted as part of a competition at Drivendata.org. Along with the competition documentation, DrivenData has created a tutorial demonstrating several relevant modeling techniques.

The goal of this project is to analyze the performance of the presented models, and independently produce a competitive and high-quality model.

This project breaks down into the following steps:

- Acquire the data.
- Explore and describe the characteristics of the data.
- Analyze the structure and performance of the models that DrivenData has presented.
- Independently produce a competitive model.

Data Acquisition

The data are available at this link. DrivenData also provides documentation on the dataset and required submission format.

The files are:

- TrainingData.csv – 400k rows, 9 targets, 16 features
- TestData.csv – 50 rows, 16 features

The labels of the test data are withheld and used by the hosting organization to judge the competition.

Problem Description

The goal is to fit a model that predicts for each row the most likely label for each of the 9 target columns, minimizing the log loss. Log loss (cross-entropy) is a convenient metric for judging the competition as it provides a single number summarizing prediction quality.

In this document and the associated notebooks, I use the term ‘multi-target, multi-label’ in preference to the more usual ‘multi-class, multi-label’ because the terms ‘class’ and ‘label’ are often used interchangeably. Multi-target-multi-label unambiguously distinguishes this kind of problem as producing multiple independent predictions, each with its own set of labels.

Data Description

All rows have 9 target columns, each containing 1 label. Each row in the budget has 14 free-form text features and two floating point features. Any feature may be null.

The features in the dataset

The dataset contains the following 16 feature columns:

- FTE (float) - If an employee, the percentage of full-time that the employee works.
- Facility_or_Department - If expenditure is tied to a department/facility, that department/facility.
- Function_Description - A description of the function the expenditure was serving.
- Fund_Description - A description of the source of the funds.
- Job_Title_Description - If this is an employee, a description of that employee's job title.
- Location_Description - A description of where the funds were spent.
- Object_Description - A description of what the funds were used for.
- Position_Extra - Any extra information about the position that we have.
- Program_Description - A description of the program that the funds were used for.
- SubFund_Description - More detail on Fund_Description
- Sub_Object_Description - More detail on Object_Description
- Text_1 - Any additional text supplied by the district.
- Text_2 - Any additional text supplied by the district.
- Text_3 - Any additional text supplied by the district.
- Text_4 - Any additional text supplied by the district.
- Total (float) - The total cost of the expenditure.

The targets in the dataset

The dataset contains the following 9 target columns.

Target Column	Number of labels	Sample of labels
Function	37	'Teacher Compensation', 'NO_LABEL', 'Substitute Compensation'…
Object_Type	11	'NO_LABEL', 'Base Salary/Compensation', 'Benefits'…
Operating_Status	3	'PreK-12 Operating', 'Non-Operating', 'Operating, Not PreK-12'
Position_Type	25	'Teacher', 'NO_LABEL', 'Substitute'…
Pre_K	3	('NO_LABEL', 'Non PreK', 'PreK')
Reporting	3	'School', 'NO_LABEL', 'Non-School'
Sharing	5	'School Reported', 'NO_LABEL', 'School on Central Budgets'…
Student_Type	9	'NO_LABEL', 'Unspecified', 'Special Education'…
Use	8	'Instruction', 'NO_LABEL', 'O&M'…

Feature Example

The table below shows all 16 feature names and sample values for one row.

Feature Name	Feature Value
Object_Description'	NaN
'Text_2'	SPECIAL EDUCATION INSTRUCTION
'SubFund_Description'	LOCAL
'Job_Title_Description'	Teacher, Special Education
'Text_3'	NaN
'Text_4'	NaN
'Sub_Object_Description'	NaN
'Location_Description'	NaN
'FTE'	1.0
'Function_Description'	NaN
'Facility_or_Department'	NaN
'Position_Extra'	NaN
'Total'	67397.91883
'Program_Description'	NaN
'Fund_Description'	NaN
'Text_1'	NaN

Label Example

The table below shows the 9 target names and label value for the features above.

Target	Label
Function	Teacher Compensation
Use	Instruction
Sharing	School Reported
Reporting	School
Student_Type	Special Education
Position_Type	Teacher
Object_Type	Base Salary/Compensation
Pre_K	NO_LABEL
Operating_Status	PreK-12 Operating

Submission Format

DrivenData requires that submission be in the following format:

- 50064 rows, 104 columns of floating point probabilities
- Label columns with “TargetName__PossibleLabel”, e.g. Function__Teacher_Compensation, replacing spaces in labels with underscores.

Multi-target, multi-label classification and the tutorial's approach

The DrivenData tutorial's approach to classification is to one-hot encode all targets and use sklearn.multiclass's OneVsRestClassifier to drive a logistic regression classifier across all 104 resulting binary columns. This classifier can then produce probability predictions in the correct format, treating the problem as 104 separate binary classification problems.

Sklearn classifiers, in general cannot deal with multi-target, multi-label (with some exceptions). While logistic regression will properly classify in single-target, multi-label problems, it will not work in the more general setting.

It should be noted that some Sklearn classifiers can natively deal with multi-target, multi-label tasks (notably RandomForestClassifier). These classifiers, however produce prediction outputs in a different format (see first_models.ipynb, mod2 for details).

The choice of OneVsRestClassifier has several ramifications. First, Sklearn does not provide generally applicable metrics for multi-target, multi-label problems. For the competition, DrivenData uses (and provides) a specific metric that calculates the average log loss across all targets.

Second, standard classification metrics (such as accuracy, precision, recall, F1, etc.) require predictions in order to work correctly. The predictions from the OneVsRestClassifier() are the wrong format for these metrics, as the classifier produces 104 different and independent binary predictions for each possible label.

In this project, we built a specific tool (flat-to-labels) that restructures the classifier output (104 columns of probabilities) to 9 columns of label outputs (matching the input targets) in order to be able to apply the standard Sklearn classification quality metrics to each model.

The models in the tutorial

The DrivenData tutorial presents five different models.

Mod0

A rudimentary model that uses only the numerical data, ignoring all the text data. This has the advantage of producing a working model that produces output in the correct syntax, with correct semantics. This model is then used to predict probabilities on the holdout data that can be used for submission to the competition. The framework illustrated in this model is carried forward throughout the tutorial.

Mod1

This model adds text processing to the workflow, illustrating several important features:

- ***Pipeline*** and related tools that allow different feature types to be processed separately and recombined before classification.
- ***CountVectorizer*** is used to create sparse word-count vectors for input to the classifier with default setting.

The pipeline structure in this model is retained throughout the rest of the tutorial, facilitating model changes and enhancements.

Mod2

RandomForestClassifier replaces the OneVsRestClassifier demonstrating the flexibility provided by the pipeline.

Mod3

This model improves the configuration of the CountVectorizer by adding bigrams and modifying the tokenization pattern.

Mod4

The final model introduces the HashingVectorizer for reduction of memory usage and a custom tool (provided by DrivenData) that adds interaction features. Essentially, the cross-product of all tokens is produced providing an indicator matrix that shows when tokens appear together in the data. This is an implementation of a technique used in a previous winning entry.

Initial Findings

In the first phase of this project we did the following:

- Acquired the data.
- Performed an exploratory data analysis (see EDA_ddbpfe.ipynb).
- Split the data into train and test sets
- Coded all 5 models. (see first_models.ipynb)
- Fitted all models and made predictions for holdout set (see first_models.ipynb)
- Submitted predictions to the competition.
- Coded tools to correctly normalize and restructure model output so that the detailed performance of each model can be observed.
- Measured performance of all models with standard metrics on all 9 targets in the dataset.

Example Result

The table below shows competition result and aggregate classification metrics for the first model. The final report will present and discuss all results in detail.

metric scores on test set Model0		competition score	log loss	accuracy	F1	ROC AUC
All targets	**aggregate**	**1.3557**	**0.5599**	**0.1532**	**0.5645**	**1.3557**
Scores on individual targets	Function	2.5994	0.2760	0.0312	0.5665	2.5994
	Use	1.7791	0.4253	0.1183	0.6974	1.7791
	Sharing	0.4372	0.8588	0.3083	0.5666	0.4372
	Reporting	2.0767	0.3729	0.0512	0.6146	2.0767
	Student_Type	0.6171	0.7663	0.2892	0.5726	0.6171
	Position_Type	0.8652	0.6410	0.2611	0.5592	0.8652
	Object_Type	1.1127	0.6337	0.1554	0.5064	1.1127
	Pre_K	1.2525	0.5570	0.0798	0.5218	1.2525
	Operating_Status	1.4616	0.5082	0.0845	0.4752	1.4616

Issues raised by first phase

There are two main questions that need to be answered in the next phase.

1. While the metrics scores from train/test on the dataset generally improve from the simpler to more complex models, the scores from competition decline. For example, the RandomForestClassifier has aggregate log loss of 0.288 on the test set as measured by the supplied metric and my independent calculation of aggregate log loss, but scores the worst of all models submitted to the competition.
2. The models from the tutorial make multiple changes at once, so it is difficult to attribute an improvement to one feature if more than one element of feature processing or modeling is changed between models.

Next steps

We will rebuild the models and add one change at a time and score the models (with several metrics). In this way we will discover what changes in feature processing and classification are responsible for enhancing or degrading model performance and their relative strength or weakness.

We will investigate why models that are better with train/test split do not score better in the competition, but the main thrust will be to understand proper use of the modeling techniques.

Using the best tools (as discovered by understanding the contribution of each model change), we will construct and submit a competitive model.

We will produce final report and presentation.