Satej Soman

CAPP30254: Machine Learning for Public Policy

Spring 2019

HW 3
MACHINE LEARNING PIPELINE IMPROVEMENTS & EVALUATION

Contents

Notes

- Representative code snippets are interspersed with analysis and explanations below; all code is available on GitHub: `https://github.com/satejsoman/capp30254/tree/master/hw3`.
- The pipeline library lives in the `code/pipeline` directory, while the sample application which imports the library is `code/donors_choose.py`.
- The generated table of hyperparameters is uploaded as a separate CSV; it also is available on GitHub as `code/evaluations.csv`.

1 Coding: Pipeline Improvements

1.1 Improvements

The following improvements have been made to the `pipeline` library:

- The library now includes a stage to generate train/test data splits. Due to the object-oriented design, the current implementation can easily be overriden by subclassing or monkey-patching in specific applications.
- Additional metrics, besides accuracy, have been added to the model evaluation. Specifically, precision, recall, and ROC-AUC have been added to the model evaluation stage.

1.2 Additional Models

In the original pipeline design, the model was not hard-coded, so pipeline runs can be parametrized by the model implementation:

```
donors_choose_preprocessors = [
    ...
]
donors_choose_feature_generators = [
    ...
]

models_to_run = {
    ...
}

def model_parametrized_pipeline(description, model):
    return Pipeline(input_path, "funded_in_60_days",
        summarize=False,
        data_preprocessors=donors_choose_preprocessors,
        feature_generators=donors_choose_feature_generators,
        name="donors-choose-" + description,
        model=model
        output_root_dir="output")

for (description, model) in models_to_run.items():
    model_parametrized_pipeline(description, model).run()
```

The above code is purely representative; in implementation, the data generation and feature preprocessing are done in a separate pipeline. The results of this pipeline are serialized and fed in as the inputs to a pipeline that solely trains and tests models. In this way, computation to process data and create feature vectors is not repeated for every trial run.

2 Analysis: Application to DonorsChoose Project Funding Viability

2.1 Background

DonorsChoose.org is a platform for soliciting donations for K-12 schools in need. Teachers post a project proposal highlighting a shortfall of resources at their institution as well as information about what areas of study will benefit from the project meeting its funding requirements. Additional data available on DonorsChoose.org describe the socioeconomic status and geography of the school, as well as characteristics of the teacher.

Responding to shortfalls in educational resources is a time-sensitive matter; donors and teachers should therefore understand the factors that determine whether a project will succeed within a reasonable amount of time (in this analysis, 60 days). Correctly predicting which projects are viable allows for effective use of time and resources for both the donors and the schools.

2.2 Data Exploration

Preliminary inspection of the data identifies columns that should be excluded from further analysis. Project- and school-specific identifiers such as `projectid`, `school_ncesid`, and `schoolid` are not useful for predicting funding success. Additionally, we choose to *exclude* the teacher's preferred prefix (`teacher_prefix`) because it proxies stated gender, a protected class under Title VII.

Table 1 shows summary statistics for other numerical inputs. While latitude and longitude are potentially good predictors, geographic variation can be captured by categorical variables such as `school_city`, `school_state`, and `school_district`.

Further, we can see that two other predictors, the number of students reached and the total price, need to be scaled since many of the other predictors are binary. Without scaling, the use of these predictors will degrade SVM performance, and distort results in k-nearest neighbors analysis.

Finally, we also see that 71% of projects are funded within 60 days. A good baseline classifier would one that randomly predicts success with a probability of 71%.

	MEAN	STD DEV	MIN	50%	MAX
`school_latitude`	36.83	4.96	18.25	36.62	65.67
`school_longitude`	-95.86	18.39	-171.69	-90.10	-66.63
`total_price_including_optional_support`	654.01	1098.02	92.00	510.50	164382.84
`students_reached`	95.45	163.48	1.00	30.00	12143.00
`funded_in_60_days`	0.71	0.45	0.00	1.00	1.00

Table 1: Table of summary statistics for additional numerical vectors.

Additionally, other input vectors have missing values. Depending on the number of missing values and the domain context, we can fill in missing values, or exclude overly sparse vectors from our models. Table 2 shows that variables such as the schools metropolitan area and the projects secondary focus are likely too sparse to support filling with placeholder values, whereas appropriate missing values for resource type, grade level, and students

reached can be found by examining the distributions for those variables. As an example, Figure 1 shows the distribution of `students_reached` - downcoding missing values to `0` is an appropriate approximation for the number of students reached and will not materially change the distribution.

	# MISSING
school_metro	15224
school_district	172
primary_focus_subject	15
primary_focus_area	15
secondary_focus_subject	40556
secondary_focus_area	40556
resource_type	17
grade_level	3
students_reached	59

Table 2: Vectors with missing values.

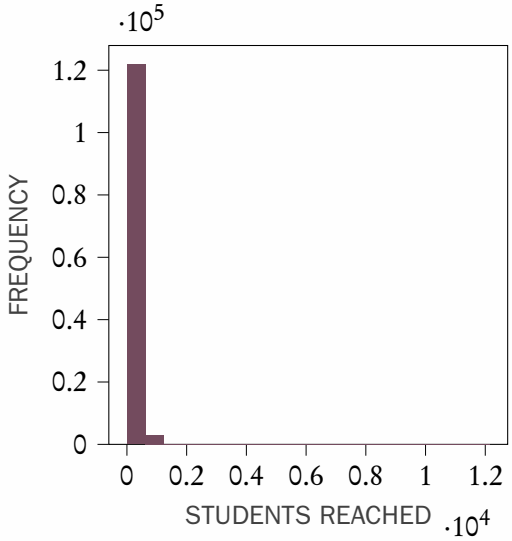

Figure 1: Distribution of number of students reached.

2.3 Feature Selection

As discussed above, we exclude teacher characteristics but focus on project-level, school-specific, and temporal variables. Table 3 lists the chosen features and provides a summary of each.

FEATURE	TYPE	DESCRIPTION
school_city_categorical	categorical	city in which school is located
school_state_categorical	categorical	state in which school is located
primary_focus_subject_categorical	categorical	broad subject on which the project focuses
primary_focus_area_categorical	categorical	specific area on which the project focuses
resource_type_categorical	categorical	type of resources asked for in project
poverty_level_categorical	categorical	socioeconomic indicator of school
grade_level_categorical	categorical	grade level of students affected by project
school_charter_binary	binary	whether the school is a charter school
school_magnet_binary	binary	whether the school is a magnet school
students_reached_clean_scaled	continuous	the number of students reached, scaled to $[0, 1]$
total_price_including_optional_support_scaled	continuous	the total price of the project, scaled to $[0, 1]$
eligible_double_your_impact_match_binary	binary	eligible for donation matching
month_posted	categorical	the month in which the project was posted
funded_in_60_days	binary	**column to predict**

Table 3: Features passed to machine learning models.

2.4 Classifier Models

A number of classifiers are used to try and predict which projects on DonorsChoose will be funded. The implementations available through scikit-learn offer a number of parameters influencing the performance of the classifier.

MODEL	PARAMETER	DESCRIPTION	VALUES
logistic regression	penalty	dimension of regularization function	L^1, L^2
	cost (C)	model penalization for misclassification	10^{-2}, 10^{-1}, 1, 10, 10^2
k-nearest neighbors	neighbors (k)	number of points to use for classification	10, 50, 100
decision tree	split criterion	information content measure for tree splits	Gini, entropy
random forest	-	-	-
gradient boosted classifier	learning rate (α)	gradient vector multiplier	0.1, 0.5, 2.0
bagging	sample fraction	fraction of sample used for bootstrap aggregation	0.1, 0.5, 1.0

Table 4: Classifiers and parameters tested.

2.5 Methodology

All available data were split into three training/validation data sets. The three validation sets were the three periods of six months each preceding 31 December, 2013. Each training set was comprised of all data up until the beginning of each validation set.

The values of the parameters in Table 4 were varied and evaluation metrics were calculated for each training set. The metrics used in this analysis were: precision, recall, F1, area under receiver-operator characteristic curve, and mean accuracy.

3 Report: Classifier Performance

3.1 Metric Choice

As stated in the initial analysis, we should focus on projects likely to succeed - this indicates **recall** is the metric that is most important in choosing a classifier.

3.2 Analysis of Hyperparameter Tuning

Sorting the results of hyperparameter tuning of all models indicates that a simple decision tree is the best classifier. However, the decision trees, regardless of criterion, only perform well on the first two training/test periods.

The

3.3 Temporal Trends

Interestingly, many classifiers suffer from the problem of doing well on the first two training/testing sets but suffering in performance when evaluated on the third set. Further analysis is required to determine if general

equilibrium effects warrant handling projects posted in late 2013 differently. For example, an increase in popularity of the DonorsChoose platform due to initial funding success may cause a flood of less-viable projects to be posted in the second half of 2013.