Title: Machine Learning Engineer Nanodegree

Speciality

Project: Capstone Proposal - Predicting US High School Graduation Success

> Author: jtmoogle @github.com Date: Feb 8, 2018

Proposal

The purpose is to predict the success if US High School Graduates can meet 90% graduation rate in 2020[1]. In this proposal, we will leverage the publicly available data sources for AT&T 2015 'Data for Diplomas' Hakathon[2], and use machine learning/ML techniques such as classification and regression algorithms to evaluate problems as follows:

Problem	Pattern ML	Predictive Model
1. **Prediction if achieving 90% graduation goal**	Supervised Learning	Classification model
2. **Prediction of graduation rate**	Supervised Learning	Regression model

Tasks will be included to

- Model and predict individual performance of school district
- Compare predictive accuracy of different algorithms
- (Stretch target) If the feature is suitable for classification, a further comparative analysis will be done that will be useful for regression.

Goal is to explore insights, and understand the factors related to success and failure, so we can propose actionable plans to increase the U.S. high school graduation rate reaching 90% by 2020[1]

Domain Background

Children are our future, higher education bring children better fortune. Our kids will benefit from the goal of "Increase high school graduation rate in 2020 to 90% with a "better future."

For decades, the high school graduation rate increased from 71.7% in 2001 to 81.4% in 2013 nationally [2]. People used a variety of machine learning techniques and predictive analytics, so they gain actionable insights and improve educational opportunities early. An example from Tacoma public schools in Washington State, which uses the ML models to predict the risk of dropping out from schools, so teachers could intervene early to help students at risks. In doing so, graduation rates increased from 55% to 78% by 2014[4].

By 2020, we need more 9% (about 310,000 more) graduates to meet our goal of 90% on-time graduation rate by 2020. Students in low-income, minority, and special education students, big cities/big districts, and big states seems struggle and challenge to reach graduation rate 90% on-time [2].

Problem Statement

This proposal will use the supervised ML techniques of classification and regression models to evaluate problems below:

1. Prediction if high school will have a success or failure of 90% or more graduates on-time This will be in supervised *classification* process.
2. Prediction of high school graduation rate This will be in supervised *regression* process. Goals
3. Explore potential factors or insights: significant impact of features on the graduation rates, and seek potential linkages
4. Understand factors related to success and failure: evaluate the performance and predictive power of fitting models in the school district level

Experimental Tasks

- Feature selection: identify which predictor (feature/variables) really matter to establish a model that accurately predicts at or exceed 90% graduation rate

- Model and predict individual performance of school district
- Compare prediction accuracy of different algorithms
- (Stretch target) conduct a further comparative analysis if the feature is suitable for classification, that will be useful for regression.

Analyzing the results may lead to the ideas of increasing graduation rates. In doing so, we can act on before problem becomes serious, and can propose workable solutions feasible to bring our goal closer to 90%

Datasets

The dataset is from Data for Diplomas_Merged Data.zip which contain 2011-2012 High School Graduation & 2012 Census data.

- GRADUATION_WITH_CENSUS.csv is data file for 2012 Four-year regulatory Adjusted Cohort Graduation Rates (ACGR) joined with the maximum overlapping 2010 Census data
- The analysis will be focus on the year of 2011/2012

Inputs

A total of 9907 records were observed for 580 variables, each record represents each school district. The variables used are population, ethnicity, gender, age, geography, and family educational background, household. The ALL_DATA_SCHEMA_M.pdf contain definition for all fields

We performed the following to reduce 580 variables to less 35 variables.

The proposal data analysis PDF list the python codes which read raw data file, and pre-processed the following logics

- Feature columns
  - Drop columns name embedding 'MOE_', which mean margin error column
  - Select columns name embedding partial text: pct=percentage, Inc/INC=income, avg=average, House, Area, ALL_, COHORT
  - Filter only columns whose datatype is integer or float
  - Drop rows if column has 10 or more NaN values found
  - Impute NaN with zero by filling in missing data with zero
- Target columns: Get target data for target column
- Perform Stepwise selection, the forward-backward feature selection based on p-value from statsmodels.api.OLS

Features	Classifier process	Regression process	Description
County.1	include	include	County ID Number
CWD_COHORT_1112	include		Number of children with disabilities in the graduation cohort
leaid11	include		Local Education Agency (district) NCES ID
MAS_COHORT_1112	include		Number of Asian/Pacific Islander students in the graduation cohort
MBL_COHORT_1112	include	include	Number of Black students in the graduation cohort
MHI_COHORT_1112	include		Number of Hispanic students in the graduation cohort
pct_Age5p_German_ACS_08_12		include	percentage of population ages 5 years and over who speak English less than "very well" and speak German at home.
pct_Age5p_Navajo_ACS_08_12	include	include	percentage of population ages 5 years and over who speak English less than "very well" and speak Navajo at home
pct_Age5p_OthPacIsl_ACS_08_12	include		percentage of population ages 5 years and over who speak English less than "very well" and speak some other Pacific Island language at home
pct_Age5p_Scandinav_ACS_08_12		include	percentage of population ages 5 years and over who speak English less than "very well" and speak a Scandinavian language at home
pct_Age5p_WGerman_ACS_08_12		include	percentage of population ages 5 years and over who speak English less than "very well" and speak another West Germanic language at home
pct_Census_UAA_CEN_2010	include	include	percentage of addresses was returned to the Census with the postal code "Undeliverable as Addressed"

Features	Classifier process	Regression process	Description
pct_Civ_unemp_16p_ACS_08_12	include	include	percentage of civilians ages 16 years and over in the labor force that are unemployed Persons
pct_College_ACS_08_12	include	include	percentage of population aged 25 years and over that have a college degree or higher
pct_Female_No_HB_ACS_08_12	include	include	The percentage of all ACS occupied housing units with a female householder and no husband of householder present
pct_Females_CEN_2010		include	percentage of population that is female
pct_HHD_PPL_Und_18_CEN_2010	include	include	percentage of all census occupied housing units where one or more people are ages 18 years or under
pct_Hispanic_CEN_2010	include	include	percentage of total population that identify as "Mexican", "Puerto Rican", "Cuban", or "another Hispanic, Latino, or Spanish origin"
pct_Inst_GQ_CEN_2010	include	include	percentage of Census population who live in group quarters and are primarily ineligible, unable, or unlikely to participate in labor force while residents
pct_Males_CEN_2010	include	include	percentage of Census total population that is male Persons 2010
pct_MLT_U10p_ACS_08_12	include	include	percentage of all ACS housing units that are in a structure that contains 10 or more housing units
pct_Mobile_Homes_ACS_08_12	include		percentage of all ACS housing units that are considered mobile homes
pct_MrdCple_HHD_CEN_2010	include		percentage of all Census occupied housing units where the householder and his or her spouse are listed as members of the same household;
pct_NH_AIAN_alone_ACS_08_12	include	include	percentage of total population indicate no Hispanic origin and their only race as "Asian Indian", "Chinese", "Filipino", "Korean", "Japanese", "Vietnamese", or "Other Asian"
pct_NH_AIAN_alone_CEN_2010		include	percentage of population indicate no Hispanic origin and their only race as "American Indian or Alaska Native" or report entries such as Navajo, Blackfeet, Inupiat, Yup'ik, or Central/South American Indian
pct_NH_Blk_alone_CEN_2010	include	include	percentage of total population indicate no Hispanic origin and their only race as "Black, African Am., or Negro" or report entries such as African American, Kenyan, Nigerian, or Haitian
pct_NH_NHOPI_alone_ACS_08_12	include		percentage of the population that indicate no Hispanic origin and their only race as "Native Hawaiian", "Guamanian or Chamorro", "Samoan", or "Other Pacific Islander"
pct_NH_NHOPI_alone_CEN_2010		include	percentage of the 2010 Census total population that indicate no Hispanic origin and their only race as "Native Hawaiian", "Guamanian or Chamorro", "Samoan", or "Other Pacific Islander"
pct_NO_PH_SRVC_ACS_08_12		include	ercentage of ACS occupied housing units that do not have a working telephone and available service
pct_No_Plumb_ACS_08_12		include	percentage of all ACS housing units that do not have complete plumbing facilities
pct_Pop_25_44_CEN_2010		include	percentage of the 2010 Census total population that is between 25 and 44 years old
pct_Pop_45_64_ACS_08_12		include	percentage of the ACS population that is between 45 and 64 years old
pct_Pop_45_64_CEN_2010	include		percentage of the Census population that is between 45 and 64 years old
pct_Pop_5_17_CEN_2010	include		percentage of the Census population that is between 5 and 17 years old
[illegible]	include	include	percentage of Census population that is under under 5 years

Features	Classifier process	Regression process	Description
pct_Pop_Under_5_CEN_2010	include	include	old percentage of the ACS eligible population that are classified
pct_Prs_Blw_Pov_Lev_ACS_08_12	include	include	as below the poverty level given their total family or household income
pct_PUB_ASST_INC_ACS_08_12		include	percentage of all ACS occupied housing units that receive public assistance income
pct_Rel_Under_6_CEN_2010		include	percentage of 2010 Census family-occupied housing units with a related child under 6 years old
pct_Sngl_Prns_HHD_CEN_2010	include		percentage of all 2010 Census occupied housing units where a householder lives alone
pct_TEA_Update_Leave_CEN_2010	include	include	percentage of addresses from which a Census form was expected to be delivered for mail return that were in an Update/Leave type of enumeration area in the 2010 Census
pct_URBAN_CLUSTER_POP_CEN_2010		include	percentage of the 2010 Census total population that lives in a densely settled area containing 2,500 to 49,999 people
pct_Vacant_CEN_2010		include	percentage of addresses in a 2010 Census mailback area that were confirmed as vacant housing units
pct_Vacant_Units_ACS_08_12	include		percentage of all ACS housing units where no one is living regularly at the time of interview; units occupied at the time of interview entirely by persons who are staying two months or less and who have a more permanent residence elsewhere are classified as vacant
PUB_ASST_INC_ACS_08_12		include	Number of ACS households that receive public assistance income
State.1		include	State ID Number

Output Variable

The output variable represent the high school district performance on graduation

1. Classification target variable is "Success_Pass_90" derived from "ALL_RATE_1112"
2. Regression target variable is "ALL_RATE_1112'. Float number, in range of 0 to 100
    - integer, value 1, successful if ALL_RATE_1112 is equal to or greater than 90.0
    - integer, value 0, unsuccessful if ALL_RATE 1112 is less than 90.0

Solution Statement

We will calculate feature importance or coefficient of determination, R square value for feature selection to determine good candidates fitting models. Goal to reduce to 30% or less features if feasible.

The dataset will be divided into 70% training, 30% testing. The two process will train a set of predictive models respectively for classification and regression. The best candidate selection of supervised learning algorithms will experiment at least 3 for each model: K-Nearest Neighbors (KNN), Gaussian Naive Bayes(NB), Random Forest (RF), Support Vector Machines (SVM), Decision Tree, Linear Regression, Logistic Regression, Ridge, and Lasso etc.

These models will be evaluated using different performance metrics for classification and regression. The results will be compared side by side at the final process. F-score for Classification, Root Mean Squared Error (RMSE) value for regression.

Benchmark Model

For classification benchmark, we will use the naive prediction that predict all school district 'successfully' meet 90% graduation goal, none of school district fail. The output value will be 1, success for all school districts. The 'recall' will be 1.0

For regression benchmark, we will use the naive prediction that predict all school district will have '100%' graduation rate. The output value is 100.0 for all school district. The recall will be 1.0

The final model results will be compared with the naive predictor benchmarks.

Evaluation Metrics

For classification metrics, we will calculate average, standard deviation of F-score (Precision, Recall) used for finding the correctness and accuracy of the model. The best value at 1 and worse score at 0

Possible results of prediction

Confusion Matrix	Predicted as True	Predicted As False
Actually True	True Positive (TP)	False Negative (TN)
Actually False	False Positive (FP)	True Negative (TN)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F = 2 \cdot \frac{1}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

For regression metrics, we will calculate Root Mean Squared Error (RMSE) used measure the difference between the predicted by a model and actual value observed, how accurately the model predicts the response.

Square root of the average of squared errors

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - p)^2} \quad where\ p_i\ is\ predicted\ value,\ p\ is\ modeled\ value$$

Lower values of RMSE indicate better fit.

Project Design

The following tasks will be conducted

1. Data Collection:
   - Access data- manually pre-download the raw files listed in dataset section. File format could be excel, dat or text files
2. Exploratory Analysis:
   - Data exploration- exam file content, tables and graphs of the observation data
   - Basic statistics- calculate descriptive statistics and visualization
3. Preprocess data: Verify data quality
   - Clean up data: (1) identify missing values/fields/columns and delete NA variables (2) identify and remove near zero variables, which have no variability, and are not useful in constructing a prediction model (3) remove non-relevant Variables toward target variable: *graduation rate* (4) identify and remove high correlated Variables: show the effect of removing descriptors with high absolute correlations
   - Data Transformation against the raw data, such as log-transform for skewed data, label encoding, one-hot encoding
4. Fitting model: Supervised learning regression
   - create sample data- shuffle and split data into training (70%) and testing (30%) subsets in order to eliminate any bias in the ordering of the dataset.
   - build prediction models on training set with (1) Linear regression (2) Random Forest Regression (3) K-Nearest Neighbors (KNN).
5. Model Evaluation:
   - Use test dataset to evaluate and compare model performance to avoid overfitting
   - Compare cross validation result- expect the best performance model if (1) the highest/best accuracy, the lowest/least/smallest error value
6. Deployment: All analysis will be performed in this study can be reproducible, and files will be available in github capstone repository.
   - Program language: python for data manipulation
   - Tool: jupyter notebook for report
   - Machine learning libraries: sckikt-learn
   - Report: PDF files

References

[1] The GradNation Campaign: Our Goal: Increase the nation's on-time high school graduation rate to 90% for the class of 2020

[2] 2015 Building a Grad Nation Report: Progress and Challenge in Ending the High School Dropout Epidemic
Number of Additional Graduates Needed to Reach a 90 Percent Graduation Rate by State and Subgroup

[3] An Intro to Data for Diploma by devpost.com
Data for Diploma by devpost.com - Resources
Data for Diploma by devpost.com - Submissions

[4] ML Predicts School Dropout Risk & Boosts Graduation Rates
Predicting student dropout risks, increasing graduation rates with cloud analytics.

[5] GeneLesinskiaStevenCornsbCihanDagli, "Application of an Artificial Neural Network to Predict Graduation Success at the United States Military Academy", Oct 2016

[5] Stamos T. Karamouzis and Andreas Vrettos, "n Artificial Neural Network for Predicting Student Graduation Outcomes ", Oct 2008

[3] Suchita Borkar, K.Rajeswari, "Attributes Selection for Predicting Students' Academic Performance using Education Data Mining and Artificial Neural Network"

[4] Making better use of graduation rates to assess school success

- Sebastian Raschka, Aug 2014 Predictive modeling, supervised machine learning, and pattern classification — the big picture
- A Machine Learning Approach to Prioritizing Students at Risk of not Graduating High School on Time
- Predicting STEM attrition using student transcript data
- Application of an Artificial Neural Network to Predict Graduation Success at the United States Military Academy, 2016, by Gene Lesinskia, Steven Cornsb, Cihan Dagli
- 7 tools in every data scientist's toolbox
- Big Data in Education 1.1 Introduction
- Data Science by Dr. Saed Sayad
- Education in the United States
- GradNation leaders sound the alarm as U.S. remains off-track to reaching 90 percent
- Predictors of Postsecondary Success
- Progress and Challenge in Ending the High School Dropout Epidemic in 2015 Building a Grad Nation report
- Public High School Graduation Rates
- Stability Selection by Nicolai Meinshausen and Peter Buhlmann, in 2009
- Supervised and Unsupervised Machine Learning Algorithms
- Supervised learning
- U.S. Department of Education
- Udacity MLND Capstone Project Description - Education
- Udacity Machine Learning Enginner Nanodegree Program
- Unsupervised learning -Prediction of Graduation Delay Based on Student Characteristics and Performance Selecting good features – Part IV: stability selection, RFE and everything side by side, Ando Saabas

In [ ]:

Appendix: Code

```
#4 --- Stepwise selection
# cited: Does scikit-learn have forward selection/stepwise regression algorithm?
# https://datascience.stackexchange.com/questions/937/does-scikit-learn-have-forward-selection-stepwise-regression-algorithm
#
import pandas as pd
import numpy as np
import statsmodels.api as sm
import json, codecs

def stepwise_selection(X, y, initial_list=[], threshold_in=0.01, threshold_out = 0.05):
    """ Perform a forward-backward feature selection based on p-value from statsmodels.api.OLS
    Arguments:
        X - pandas.DataFrame with candidate features
        y - list-like with the target
        initial_list - list of features to start with (column names of X)
        threshold_in - include a feature if its p-value < threshold_in
        threshold_out - exclude a feature if its p-value > threshold_out
    Returns: list of selected features
    Always set threshold_in < threshold_out to avoid infinite looping.
```

```
    See https://en.wikipedia.org/wiki/Stepwise_regression for the details
    """
    included = list(initial_list)
    while True:
        changed=False
        # forward step
        excluded = list(set(X.columns)-set(included))
        new_pval = pd.Series(index=excluded)

        for new_column in excluded:
            model = sm.OLS(y, sm.add_constant(pd.DataFrame(X[included+[new_column]]))).fit()
            #debug(model.summary())
            new_pval[new_column] = model.pvalues[new_column]
        best_pval = new_pval.min()
        if best_pval < threshold_in:
            best_feature = new_pval.argmin()
            included.append(best_feature)
            changed=True
            print( 'Add  {:30} with p-value {:.6}'.format(best_feature, best_pval))

        # backward step
        model = sm.OLS(y, sm.add_constant(pd.DataFrame(X[included]))).fit()
        #debug(model.summary())
        # use all coefs except intercept
        pvalues = model.pvalues.iloc[1:]
        worst_pval = pvalues.max() # null if pvalues is empty
        if worst_pval > threshold_out:
            changed=True
            worst_feature = pvalues.argmax()
            included.remove(worst_feature)
            print('Drop {:30} with p-value {:.6}'.format(worst_feature, worst_pval))
        if not changed:
            break
    return included
```

In [ ]:

```
#4.1 --- regression
savefname='saved/rgs_stepwise_result.txt'
redofit=True

rgs_X = pd.DataFrame( rgs_feature_data, columns= rgs_feature_cols)
rgs_y = rgs_target_data

if (not redofit) and (os.path.exists(savefname)):
    info( '{} exist.  stepwise_selection loaded from a file'.format(savefname))
    with open(savefname) as data_file:
        rgs_selresult = json.load(data_file)
else:
    rgs_selresult = stepwise_selection(rgs_X, rgs_y)
    print('Stepwise selection features:')
    display(rgs_selresult)
    with open(savefname, 'wb') as f:
        info( 'Save stepwise_selection to a file {}'.format(savefname))
        json.dump(rgs_selresult, codecs.getwriter('utf-8')(f), ensure_ascii=False)
```

In [ ]:

```
#4.2 --- classification
savefname='saved/cls_stepwise_result.txt'
redofit=True

cls_X = pd.DataFrame( cls_feature_data, columns= cls_feature_cols)
cls_y = cls_target_data

if (not redofit) and (os.path.exists(savefname)):
    info( '{} exist.  stepwise_selection loaded from a file'.format(savefname))
    with open(savefname) as data_file:
        cls_selresult = json.load(data_file)
else:
    cls_selresult = stepwise_selection(cls_X, cls_y)
    print('Stepwise selection features:')
    display(cls_selresult)
    with open(savefname, 'wb') as f:
        info( 'Save stepwise_selection to a file {}'.format(savefname))
        json.dump(cls_selresult, codecs.getwriter('utf-8')(f), ensure_ascii=False)
```