Predict Suicides to save lives

Capstone Project Report

Ish Gupta 4/19/21 ML Engineer Nanodegree

Definition

Project Overview

National Institutes of Health, and Center for Disease Control and Prevention and the U.S. Census Bureau conducted a National Longitudinal Mortality Study (NLMS) for the purpose of studying effects of differentials in demographic and socio-economic characteristics on mortality.

Mortality information was obtained from death certificates available for deceased persons through the National Center for Health Statistics. Standard demographic and socio-economic variables such as education, income, and employment, as well as information collected from death certificates including cause of death are available for analyses.

Problem Statement

The survey consists of the reasons of death for the people who lost their life, and were included in the Survey. The reasons are related to Health, accidents, and intended loss of life, i.e., Suicide.

The data can be analysed and studied to see if there can be any measures that the government can take to reduce Suicidal attempts, by classifying the deaths caused by intended action, vs accidents, or health reasons.

Giving up on Life, is the most brutal act we can do to ourselves and our loved ones. Let's try, if we can employ data technologies to save one from messing up with not just this Life, but the afterlives as well.

How can we utilize the data?

- Segregate natural and suicidal deaths using Classfication algorithms, starting with xgboost
- analyse them to look for any possible reasons which could potentially be the drivers for suicide
- expand the solution using neural networks and compare the performance with xgboost

Metrics

The metrics I have used for evaluation are:

- Accuracy – performance of a model against the test data set
    - Accuracy will be helpful in confirming the performance on validation dataset, if the model was able to understand the pattern and underlying behaviour of the attributes which possibly drive an individual to commit suicide
    - Usually, with machine learning algorithms- the performance on training dataset is close to 100%, but the performance can be validated when tested against the data (which was not used for training the model)
- Confusion matrix- to see how well did the model do in identifying actual cases of Suicide, and false negatives
    - Confusion matrix can be an easier way to understand on how well did the model do against both of the outcome events, i.e., if the death was a suicide or not.
    - We'll be able to see:
        - How many cases were correctly identified as *Suicide*, i.e., *True Positives*
        - How many cases were incorrectly identified as *Suicide*, i.e., *False Positives*
        - How many cases were correctly identified as *Not a Suicide*, i.e., *True Negatives*

- How many cases were incorrectly identified as *Not a Suicide*, i.e., *False Negatives*
  It will be helpful to build a model which can have least number of False Negatives, and most True Positives, for it can be harmful, when a case is possibly a Suicide, but it gets classified otherwise.

Analysis

Data Exploration

Data overview

The dataset is 10,48,576 rows x 43 columns, with attributes in respect to citizenship, demographics, family, education, profession, income, health, habits, mortality.

Out of the major contributing attributes, the numeric attributes are:

1. Age - age of the person at the time of the survey
2. Follow- The length of follow-up period in days

The categorical features include:

3. Sex- Gender
4. POB - Region of Birth
5. Race - An expanded version of race which separates the American Indian, etc. and Asian, etc. out of the Other category in early CPS files
6. Hisp - Hispanic origin classifies all persons by Mexican, Hispanic (not Mexican) or Non-Hispanic (Not Mexican or Hispanic) origin
7. Ms- marital status
8. RELTRF - Relationship to reference person within the household
9. HHNUM - number of persons residing in the household at the time of the interview
10. SSNYN - Indicator of the presence or absence of Social Security Number on the CPS record
11. STATER - state of residence at the date of interview
12. URBAN - Urban or rural status
13. SMSAST - Is a household located in an SMSA or not? (SMSA - standard metropolitan statistical area)
14. TENURE - the type of ownership of the residence
15. EDUC- highest grade completed
16. VT - Indicates whether person was U.S. veteran
17. CITIZEN - person’s citizenship at the time of survey
18. health - the status of person health at the time of survey
19. Working - if the person was employed
20. ESR - Recoded employment status
21. IND - Job specific industrial classification codes
22. MAJIND - Major industry classification recode of 2007 specific industry
23. RCOW - Recoded Class of Worker
24. ADJINC - Inflation Adjusted Income
25. POVPCT - Income as Percent of Poverty Level
26. HISTATUS - Indicates whether person had health insurance coverage any time in the calendar year prior to the interview
27. HITYPE - the type of health insurance

output variable:

28. Cause113- cause of death – Suicide (1), or accident (2), or other Health related issues (0)

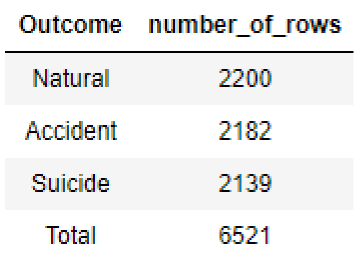
Observations

1. Data attributes

a. The data contains variable which describe an individual's:
   i. Demographics
   ii. Family
   iii. Education
   iv. Occupation
   v. Health
b. The data values are all categorical except Age and the number of follow up days.

2. Outcome class balance*:*

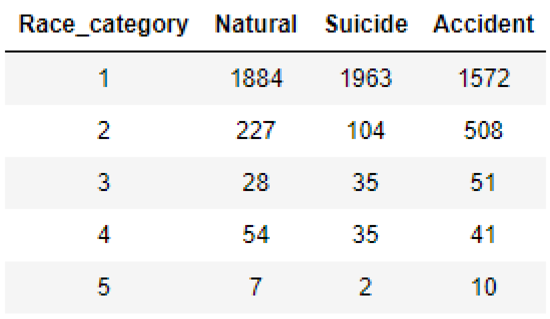
a. The data reports 106 causes of deaths, but we classify them into:
   i. 1- Suicide
   ii. 2 – Accident
   iii. 0 – Natural/health issues
b. The class imbalance was such that Natural deaths comprised more than 99% of the data, hence the data was trimmed for natural deaths using random sample of similar number of records as other two classes. So, the data set was reduced from 10,48,576 x 43 to 6521 x 24 (after dropping unnecessary columns, and treating the class imbalance)
c. Finally, we had balanced proportion of data in respect to the outcome classes.



Outcome	number_of_rows
Natural	2200
Accident	2182
Suicide	2139
Total	6521

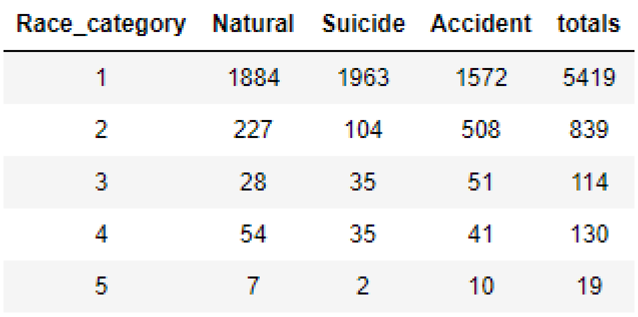
3. Findings w.r.t data attributes*:*

To study the average effect of any attribute towards the outcome, we could be misled by looking at just the numbers. For example, when we look the numbers below, it can seem that most loss of life had happened to Whites (category 1):



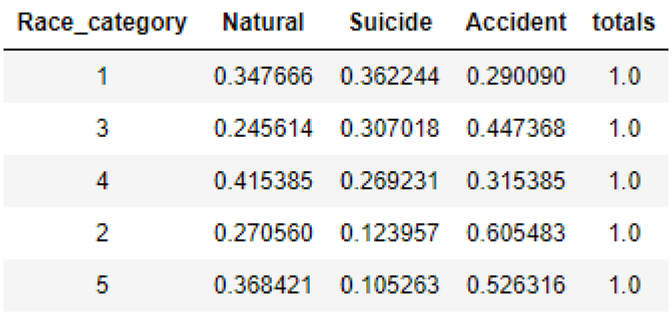
Race_category	Natural	Suicide	Accident
1	1884	1963	1572
2	227	104	508
3	28	35	51
4	54	35	41
5	7	2	10

But it is when we look at the *totals*- it is *Blacks* (category 2), who had most accidents, in **proportion**.



Race_category	Natural	Suicide	Accident	totals
1	1884	1963	1572	5419
2	227	104	508	839
3	28	35	51	114
4	54	35	41	130
5	7	2	10	19

So, dividing the values by their total- we get below which shows *category 4*(*Asian or Pacific Islander*) had most natural deaths, *Whites* (*category 1*) most suicides, and *Blacks* (*category 2*) most accidents.



Race_category	Natural	Suicide	Accident	totals
1	0.347666	0.362244	0.290090	1.0
3	0.245614	0.307018	0.447368	1.0
4	0.415385	0.269231	0.315385	1.0
2	0.270560	0.123957	0.605483	1.0
5	0.368421	0.105263	0.526316	1.0

Using this approach, we can see that with:

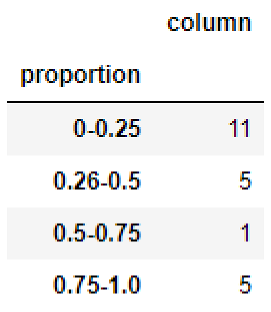
a. Gender variable (*SEX*), it can be noted that it is Females committing most suicides
b. Age group attribute (*AGE*) shows, it is mostly the youth committing Suicides
c. With Race, we could see that comparatively- White people have committed more Suicides, and Black people who had lost more lives in accidents
d. Marital Status variable shows, married people had more numbers in Accidents, and Separated in suicides

4. *Unimportant features*

a. Some unimportant attributes which could not have any effect on the outcome variable but describe the outcome event only, were removed. The list can be referred, here.

5. *Missing Values*

a. Occupation (*MAJOCC*), industry (*MAJIND*):
The industry which an individual works for, and the kind of job that he/she does in that industry, can be a useful attribute but it had missing values in more than 40% of the data.
The reason survey tells for this to be blank is, the individual could be without a job, or not working. So, I had decided to not drop the variable but, impute the missing values with '-1'.
b. The data had missing values as below. 11 columns had less than 25% of data missing, 5 had upto 50% of missing values, 1 upto 75%, and 5 with even more than that. The attributes which were missing more than 40% of the data were dropped, and rest were imputed using MissForest.



	column
proportion	
0-0.25	11
0.26-0.5	5
0.5-0.75	1
0.75-1.0	5

Data cleaning

Removing Less significant features:

I started off looking at the data variables which would not make a signification contribution towards the outcome, which can be:

1. Variables which are just for record keeping and have distinct value per row
    a. *RECORD* - sequential number for each record on the file
2. Variables with a single unique value throughout the data
    a. Variables related to smoking and tobacco usage, which were mostly blanks in the data we had
        i. found: *SMOK100, AGESMK, SMOKSTAT, SMOKHOME, CURRUSE, EVERUSE*
3. attributes with inclination towards an outcome event
    a. check for any variables inclined towards any particular outcome event, i.e., If there are any attributes which had a valid value for one outcome event, and not the other.
        i. found: none
4. Variables missing more than 40% of the values
    a. *OCC:* 4 Digit Occupation Code
    b. *IND*: 270 unique Job specific industrial classification codes
    c. *RCOW*: 5 class of worker codes to explain if the individual works/worked in private, government, self-employed, work-without-pay, or has never worked
    d. *CITIZEN*: person's citizenship at the time of survey
    e. *HEALTH*: status of person's health
    f. *INDDEA*: indicator, if the person is alive or not
    g. *INDALG*: mortality indicator, basis the algorithmic calculation
5. Variables describing the outcome (when it happened, or after it has happened)
    a. *DAYOD:* day of the week on which the decedent died
    b. *HOSP*: place of death
    c. *HOSPD*: location of death relative to a hospital
    d. *CAUSE113:* alternated with 'Suicide' – for 3 distinct categories:
        i. *0* – Natural death
        ii. *1* – Suicide
        iii. *2* - Accident
6. variables completely missing in one or the other case of outcome event
    a. found: none

Exploratory Visualization

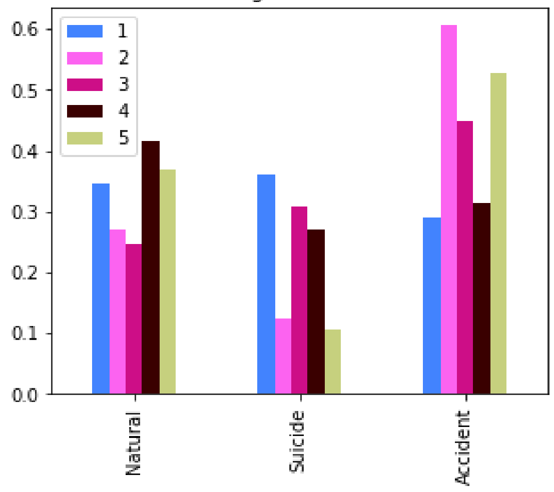
The variables which have <= 6 distinct categories in the data were studied for their effect on deaths.

Race

While checking Race, we could see that:

1. Race = 2 (Black) has had most accidents in comparison to other 4 categories of Race
2. Race = 1 (White) has had most suicides, and
3. Race = 4 (Asian or Pacific Islander) have had most deaths due to health issues

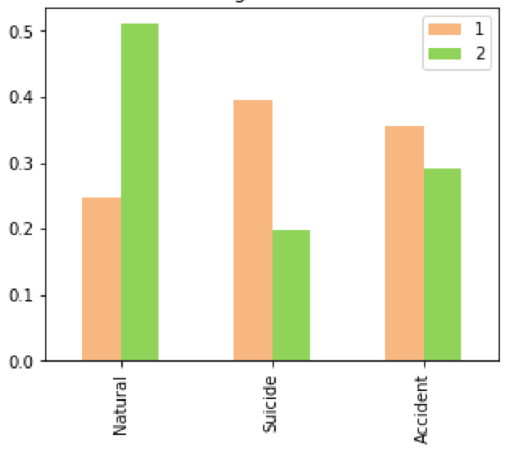
Fig. 1 - Race



Gender:

Females (category 2) have had the greatest number of natural deaths, but Males exceed in Suicides and Accidents. This could mean, that males have better health than women, but are more prone towards succumbing to emotions, and accidents.

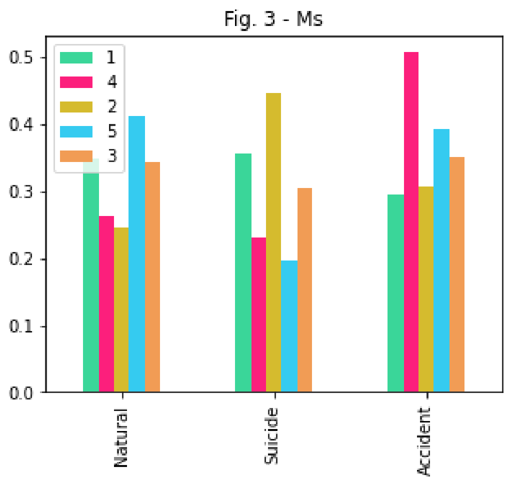
Fig. 2 - Sex



Marital Status

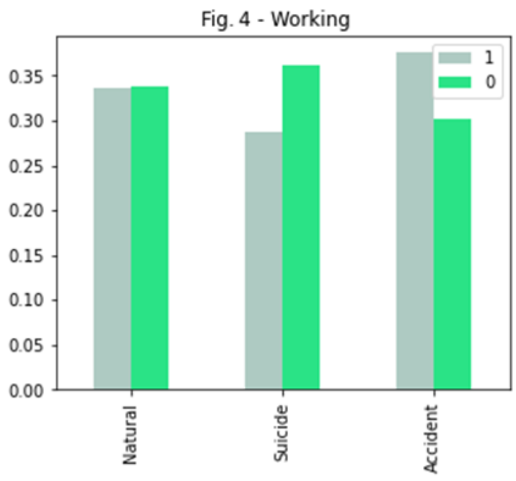
1. Never Married (category 5) people top the accidents
2. Widowed (category 2) top the Suicides
3. Separated (category 4) top natural deaths

This could mean, unmarried people are comparatively careless on roads, widowed accumulate emotional stress overtime, and end up taking their lives- which can be taken as an attribute to counsel the individuals (by the Administration) in order to help them out.



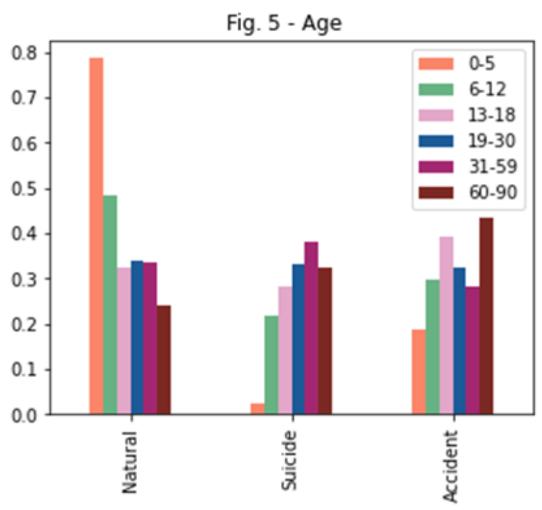
Working

People who are working happen to commit more suicides in comparison, with the people not involved in any employment. This could be related to not having good Work-Life balance, and the resulting stress.



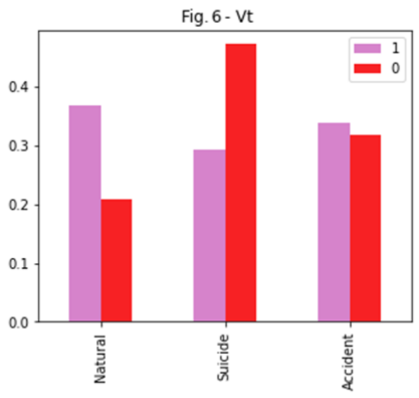
Age

1. Infants have died most natural deaths
2. Senior citizens have had most accidents
3. People within the range of 31-59 are the ones with the greatest number of Suicides, followed by youngsters within age 19-30. This is known that this is the age of passion, when all of the people are busy in making their career, family. When failures come, they may lead to frustration, depression like issues.



Veteran

This shows that non-Veterans (category 2) top in committing Suicides, while Veterans face most natural deaths, and accidents.



Algorithms and Techniques

Benchmark

- Random Forest was utilized to more of exploring the data in terms of looking at the variables of importance
- XGBoost, could reach 68.3% of accuracy in 3 variants, where one was with the basic default settings, the second and third with finding optimal parameters using cross validation.
- Pytorch was then explored to observe the performance, and its comparison to the XGBoost performance.

Methodology

Data Pre-processing

In the final version of EDA, I could finalize below steps to clean the data, and bring it into a shape to be ready for building Classifiers:

1. Categorize eligible variables into classes:

   a. distribute age into classes: ["0-5", "6-12", "13-18", "19-30", "31-59", "60-90”]
   b. categorize POB (place of birth) field into categorized regions
   c. categorize STATER (state of residence) codes into political divisions
2. convert follow up days into years
3. create new variable:
   a. “Suicide” (using *CAUSE113*) to denote the reason of death was natural, accidental, or intentional:
      i. 1- Suicide
      ii. 2 – Accident
      iii. 0 - Natural
   b. “Working” (using *OCC*) to know if an individual is/was working? (using Occ)
4. Impute *MAJIND, MAJOCC* with ‘-1’ for missing values, rather than removing them to study mortality rate is affected by occupation, industry an individual works
5. type caste all variables into category type
6. The data was widely imbalanced for deaths due to health issues, for the proportion was more than 95%, so the outcome event has to be randomly trimmed in proportion with the Suicides, and Accidents so as to balance the data, which gave ~ 6300 rows vs 1048576.
7. impute missing values using MissForest
8. save the dataset for use by other scripts which will build a Classifier using XGBoost, and PyTorch
9. create an initial Random Forest Classifier to see first level observations.
10. Drop *FOLLOW,* which denoted the days an individual was followed up for survey. Since, it cannot be an effective contributor towards a person harming his/her life.
11. Re- process a RandomForestClassifier, and study if there are any features which can be considered to drop out of the data.
12. Study permutation importance of variables to check feature importance, use RFE (sklearn) to validate the observations.
13. Filter the unimportant variables out, and re-process a classifier to see improvements.
14. Now, use find most optimal parameters for a Classifier using cross validation (GridSearchCV), and check if it helps improving the accuracy of observations, and if it reports any changes in the feature importance.
15. Use the data persisted in step 7 for further scripts

Implementation

Algorithms

Random Forest

Overview

Random forest algorithm being an ensemble of Decision Trees, i.e., combines a number of weak learners into one strong learner as it builds multiple classification trees for prediction, where each prediction is voted by the trees built while training. For each tree built in process- a random sample of rows from the training data is used, with a subset of features to split each tree. Each tree grows to the extent specified in the parameters.

The main advantage of Random Forest over Decision trees, is to combine the predictions of many decision trees into a single ensemble model(**bagging**), which in turn reduces the variance from a flexible technique like decision trees.

Strength

- runtimes are quite fast
- able to deal with unbalanced and missing data
- ensemble of multiple decision trees
- avoids overfitting

Weakness

- does a good job at classification but not for regression problem as it does not give precise continuous predictions beyond a range in training data, which can result in overfitting.
- can feel like a black box approach for we have very little control on what the model does. We can at best try different parameters and random seeds

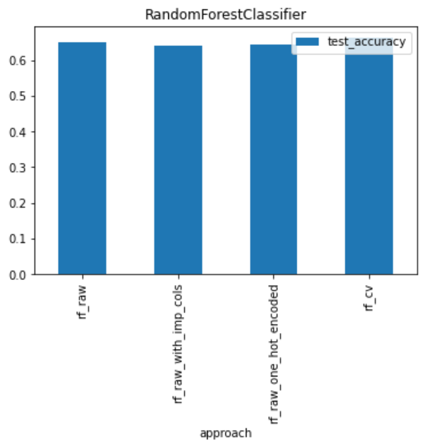
Reason for choice:

- explore the data at the surface for important variables
- to prepare the data in turn, for modelling with XGBoost, and pytorch

Architecture used

- Default *RandomForestClassifier* with *n_estimators = 100*
- Default *RandomForestClassifier* with *n_estimators = 100* only on the important variables discovered in the first step
- Default *RandomForestClassifier* with *n_estimators = 100* after the class variables are encoded using OneHotEncoding
- Optional parameters discovered using CV (round I):
  - bootstrap = True, max_depth = 10, max_features = auto, min_samples_leaf = 4, min_samples_split = 2, n_estimators = 800

Performance



XGBoost

Overview

Boosting algorithm helps in reducing variance and bias in a machine learning ensemble. The algorithm helps in the conversion of weak learners into strong learners by combining N number of learners.

Strength

- In contrast to bagging techniques like Random Forest, in which trees are grown to their maximum extent, boosting makes use of trees with fewer splits
- The small trees being not too deep, are comparatively more interpretable
- It prunes the tree with the *Similarity score* before entering into the actual data modelling
- good option for unbalanced datasets but we cannot trust random forest in these types of cases
- Boosting is a resilient method that curbs over-fitting easily
- XGBoost incorporates a sparsity-aware split finding algorithm to handle different types of sparsity patterns in the data

Weakness

- implementation of boosted models is relatively slow
- boosting is sensitive to outliers since every classifier is obliged to fix the errors in the predecessors

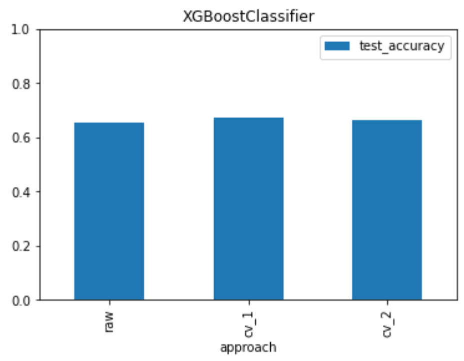
Reason for choice

- A go-to algorithm for classification problems
- To see how well it can perform with the use case, and in comparison, with the pytorch classifier

Architecture used

- Default XGBClassifier with:
  - *min_child_weight = 0, learning_rate = 0.01, n_estimators= 10*
  - *early_stopping_rounds = 10, eval_metric = 'aucpr'*
- optimal parameters discovered using cross validation (run I):
  - colsample_bytree = 1.0, gamma = 0.3, learning_rate = 0.05, max_depth = 3, min_child_weight = 5, scale_pos_weight = 1
  - *early_stopping_rounds = 25, eval_metric = 'aucpr'*
- optimal parameters discovered using cross validation (run II):
  - colsample_bytree = 0.5, gamma = 0.15, learning_rate = 0.09, max_depth = 4, min_child_weight = 2
  - *early_stopping_rounds = 25, eval_metric = 'aucpr'*

Performance



Pytorch

Overview

An open-source python-based scientific computing package which provides Deep Learning research platform that provides maximum flexibility and speed on scientific computing for deep learning

Strength

- ➢ Wraps numpy to utilize the power of GPUs
- ➢ Numpy is integrated into pytorch using tensors and provides easy bridge to interact between the two
- ➢ Tensors can also be used on a GPU that supports CUDA to accelerate computing
- ➢ Easy to use/debug, is developer friendly

Reason for choice

- ➢ Explore the capability for a normal binary classification
- ➢ Compare the performance with xgboost, which is more commonly used in classification/regression problems
- ➢ Well known for text analysis, image classification, audio

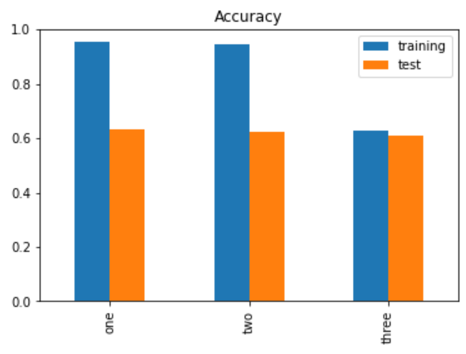
Architecture used

i. a 3-layer classifier using 3 different approaches for comparison

ii. All the approaches use Linear function for hidden and output layers, with 500 epochs, where dimensions look like below:
   - Hidden Layer 1: (144, 64)
   - Hidden Layer 2: (64, 64)
   - Output Layer: (64, 2)

iii. Approaches:
   - Approach 1: Feed-Forward with back propagation (using Adam optimizer):
     a. Hidden Layer 1 checked by *ReLU* regularized by *batch normalization*
     b. Hidden Layer 2 checked by *ReLU* regularized by *batch normalization*, and *dropout* (0.2)
     c. Output layer normalized by *softmax* function
   - Approach 2: Feed-Forward with back propagation (using Adam optimizer):
     a. Hidden Layer 1 regularized by *batch normalization*, which is then regularized by *dropout* (0.35), and checked by ReLU, i.e., ReLU (Dropout (BatchNorm1d(Linear), 0.35))
     b. Hidden Layer 2 regularized by *batch normalization*, which is then regularized by dropout (0.35), and checked by *ReLU*, i.e., *ReLU (Dropout (BatchNorm1d(Linear), 0.35))*
     c. Output layer normalized by *sigmoid* function
   - Approach 3: Feed-Forward with back propagation (using Stochastic Gradient Descent):
     a. Hidden Layer 1 checked by *ReLU*
     b. Hidden Layer 2 checked by *ReLU*
     c. Output layer normalized by *LogSoftmax* function

Performance



Project structure

The project contains 3 scripts/notebooks used in processing the data, and building a classifier.

1. *ml_toolkit.py* contains utility methods, and is being referenced in all the scripts
2. *eda.ipynb* – This script:
   a. explores the data from the beginning
   b. does necessary data cleaning, as listed in above points
   c. transforms the data by:
      i. removing unwanted variables
      ii. creating and replacing already available attributes with new concise ones
      iii. imputing missing values
      iv. one-hot encoding the data, to have an individual attribute per categorical value of all the variables
      v. build an initial classifier (RandomForestClassifier) to perform Feature selection
   d. builds RandomForestClassifier using:
      i. the data with categorical variables
      ii. with the same data, but after removing some unimportant variables as identified in the above iteration
      iii. the one-hot encoded data
      iv. and finally with the encoded data, but with optimal parameters obtained using cross validation
   e. records accuracy of the RandomForestClassifeir from each step to compare in the final step.
3. *process_xg_classifier.ipynb – This script:*
   a. uses the data prepared in *eda.ipynb*
   b. to use cross validation in finding optimal parameters for an XGBoost Classifier
   c. performs another round of cross validation to confirm the optimal parameters
   d. records accuracy of the above classifiers for comparison in the final step
4. *process_nn_classifier.ipynb* - This script uses the data prepared in *eda.ipynb*, and processes the neural network architecture as described here.

Refinement

- I started with XGBoost classifier to segregate Suicides and Accidents
- Then, I wanted to compare the performance of XGBoost with Pytorch

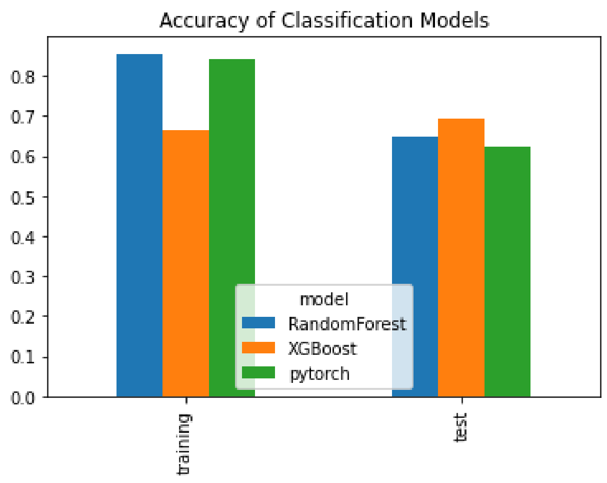
- While looking at the capstone acceptance metrics on Udacity, as I started working on the EDA script, I felt I could also incorporate a Random Forest Classifier, for it will be interesting to compare the performance.
- I had multiple hit and trials with different choices of model parameters in XGBoost, and pytorch, but I started raw in RandomForest, and ended up doing a model using CV in RF too.
- After all this, I realized, maybe the problem statement that I am trying to resolve by comparing Suicides with Accidents might be not that fruitful, but the other way round, where we can see mortality events in cases of Health-related issues, and Suicides, and finally happened to update the design accordingly.
- The code for pytorch was repetitive for the execution, and then I thought of optimizing it by putting into individual methods, and called them in an iteration over 3 model choices.

Results

Model Evaluation and Validation

Performance comparison

Comparison of the performance across 3 algorithms, RandomForest, XGBoost, and pytorch classifier:



Understanding

Justification

- XGBoost performed the best in comparison, but Pytorch can do a much better job if we had a larger dataset for all the outcome events
- False Negatives can be a problem in such a use case, where we wrongly classify a Suicide, to be not a Suicide. But that was improved when we tweaked the XGBoost model
- Out of the three models in pytorch:
  - second had the least False Negatives, but it was still more than XGBoost classifier.
  - But after applying softmax activation function to the first choice, it exceeded the second model in having lowest False negatives
- With the available volume, XGBoost performs fairly well in segregating the Suicides vs deaths resulting from health related issues.

Disclosure

The National Longitudinal Mortality Study is a collaborative effort between the US Census Bureau and the National Heart, Lung, and Blood Institute (NHLBI), National Cancer Institute (NCI), National Institute on Aging (NIA), and the National Center for Health Statistics (NCHS). The views expressed in this paper are those of the authors and do not necessarily reflect the views of the Census Bureau, NHLBI, NCI, NIA or NCHS.