Research Log

Holden Project: Detecting Phony Data

October 12, 2018

Created a script, random-data-generator.R

This takes the CNA.cct file and creates 50 new phony samples.

If does this by finding the range of each protein in the real data and then generating new samples by randomly generating values from within each protein’s range.

The 100 real and 50 phony samples were then randomly split into train and test sets.

Created an organized file structure for the scripts and data

October 15 2018 & October 16 2018

Added the Classification\_Utils.py module to the analysis scripts folder I and an Ipython note book for test several of the learning models.

Naive Bayes and SVC both got 100% accuracies on the training set. Have not run the test set.

Logistic Regression got 93%

MLP (30,30,30,30): 85%

KNN: 72%

Multinomial Bayes: is being dumb and not working

Random Forest: 94%

Gradient Boosting: 97%

Also created script for doing a PCA plot of the random data

Added code to create bar plot of the classification accuracies and save the plot, at the end of Classification-Random-Data.ipynb

October 18, 2018

Modified the random analysis script so it test on the actual test data set and trains on the whole training set. The test set was run one once and no parameters were changed to adjust to it.

Created a script called distribution-data.R that creates phony samples/individuals based on the set of observed values for each protein and script them samples randomly from a distribution of those. \*\*The made up samples consist entirely of real protein expression values found in the original data.\*\* Similar to the random-data-generator.R script this creates 50 fake samples and makes test and train files.

My hypothesis is that since these are real values being used that it will be harder to classify the fake ones. But just visually I can spot the fakes samples because a real sample’s values often match for several consecutive proteins, but in the fake ones they do not. So I predict the machine learning should still pick up on the pattern and do well.

Ran the Dist data through the basic classifiers. Accuracies were not so good. RF has the highest with 71%, which is barely above base line. MLP took 30 minutes to train… Might be over fitting, there are 4 lays of nodes which may be over kill.

Ideas to increase accuracy.

* Feature selection
* Parameter optimization
* Feature creation
  + range and dist of a sample
  + Benford’s Law compliance

October 19, 2018

Made PCA plots of the Dist dataset

Tried doing a tSNE plot, run time for the first, solving for xx takes FOREVER!

Did feature selection on the data and re ran the models. GB and MLP were not run yet

TODO

Finish making a density plot of a random sampling

October 23, 2018

Started using a package for analyzing Bedford’s law, benford\_py

Created plots of the distribution of each sample individually

TODO: visualize the dist of the fake and the real data all at the same time instead of broken up

TODO: find a way to automate saying if something complies with Benford’s law or not

November 6, 2018

Added the information from benford\_py for first and second digit to the training data. It did not improve accuracy. KNN and SVC both scored only .65.

TODO next: look at distributions of the actual second digit, not just the second digit after the first non zero

Using the first digit after the decimal in combination with the other features it still goes poorly

SVM and KNN both getting .65

If using only the distribution of first digits KNN can do 89% while SMV still only gets .65

KNN

accuracy: 0.89 (+/- 0.12)

Runtime: 0.005766558647155762 minutes

SVC

accuracy: 0.65 (+/- 0.00)

Runtime: 0.006466662883758545 minutes

RF

accuracy: 0.94 (+/- 0.12)

Runtime: 0.02681665023167928 minutes

Gradient Boosting

accuracy: 0.94 (+/- 0.15)

Runtime: 0.0597168763478597 minutes

Niave Bayes

accuracy: 0.96 (+/- 0.08)

Runtime: 0.005083171526590983 minutes

LR

accuracy: 0.65 (+/- 0.01)

Runtime: 0.004949978987375895 minutes

MLP

accuracy: 0.65 (+/- 0.00)

Runtime: 0.172183620929718 minutes

Using the second digit plus the protein data of after the decimal place once again KNN and SMV get .65 percent but using just the decimal place we get the following results:

KNN

accuracy: 0.90 (+/- 0.13)

Runtime: 0.0051833430926005045 minutes

SVC

accuracy: 0.65 (+/- 0.00)

Runtime: 0.007800118128458659 minutes

RF

accuracy: 0.95 (+/- 0.10)

Runtime: 0.026099916299184164 minutes

Gradient Boosting

accuracy: 0.91 (+/- 0.12)

Runtime: 0.05738329887390137 minutes

Niave Bayes

accuracy: 0.98 (+/- 0.06)

Runtime: 0.004850411415100097 minutes

LR

accuracy: 0.65 (+/- 0.00)

Runtime: 0.005133358637491862 minutes

MLP

accuracy: 0.65 (+/- 0.00)

Runtime: 0.1698333740234375 minutes

New Idea, try using both the first and the second digit

November 15th 2018

On Tuesday I made another script that did classification based on the first and second digits distributions. It overall does better than either of the single sets of features.

Also I received imputation software (DreamAI) from Sam which he got from Pei Wang. I put it in the Holden folder for convenience but also git ignored it (listed it in the .gitignore file) so I am not publicly publishing someone else’s work.

January 7th 2019

Ran Script to Generate 100 version of all 3 data sets for the distribution analysis:   
distribution-data-100-all-types.R

Work on getting DreamAI installed on Mary Lou

Started jobs for doing the analysis of CAN and a test of the imputation software on mary lou, ran transcriptomics as not a job on mary lou with 5 different automation script, not scheduling a job was a bad idea, now my computer cannot leave the lab…

TODO:

Graph with error bars for resampling and random

**Do data imputation, create 30 more people**

Create images that explain how data was created

January 8, 2019

Started a test for imputing two sets of 100 NA on Mary Lou to better gauge how long this will take to accomplish creating 30-50 new people.

Figure out what is returned from DreamAI, does it pass by reference or do I need to reassign values?

Started a test doing two sets of 1000 NAs on MaryLou

ToDo Today

1. Start Full Imputation Job for at least 1 dataset
2. Finish analysis of Distribution data

January 9th 2019

Running a full imputation of a single person failed on MaryLou after several hours last night.

I think the problem is that I reassigned data to the out put of DreamAI, maybe it passes by reference?

Went ahead and ran it again under that assumption

Got bar plots with error bars running for Transcriptomics, did not work with ggplot for python, used MatPlotLib

January 10th 2019

Running 3 imputations of 100 took 11 hours, started two jobs

1. Full set of data subbing 100 at a time
2. Full set of data subbing 1000 at a time
3. Test to see how long a single run of 1000 takes
4. Test to how long 5000 takes

I gave both the max time cap of 162 hrs

January 11, 2019

Plotted results of resampling on proteomics, they are lower than expected.

ToDo:

Generate results for random samples

\*\* look at PCA and or numerical frequencies of resampling data, proteomics is goofy compared to the other two

January 16 2019

Finished creating automation scripts to break imputation up into many steps. Each row will be imputed 100 NAs at a time with 24 of those running on each node at the same time on different threads so there are about 8 jobs per person created and 30 people being created. There any a lot of jobs.

Jan 18 2018

Non of the 24 threaded 100 imputaions jobs have finished, I am afraid running many processes will not speed it up as much as I hoped and a wall time will be hit.

I just submitted 540 jobs doing 1000 imputations each. No parallel threads, just one node processor and one job for each. Previous runs of similar stuff took ~44 hours

All those jobs failed almost as soon as I started them so did many of 100 batch. It only sent email confirmation that a few failed (like less than 30, but if you get the report from fsl website you can see they all failed)

There seems to have been an issue with my create-people1000.sh script. When I ran create-people1000.sh manually with correct parameters, it worked fine well at least they did not fail instantly.

Those scripts were all started from Holden2/Phony-Scripts on MaryLou.

Jan 21 2018

On Friday I started jobs doing 1000 at a time but using only the knn method, this is much faster than the full ensemble method.

All the jobs finished but the out put has all NAs, not actual values current started a single job to debug why this is. It will be writing 4 files

test-1-1-pre-knn.csv : this is confirm there are NA introduced in the data to begin with

test-1-1-post-knn.csv : this is to see is data is imputed on a pass by reference basis.

test-1-1-knn-out.csv : this is to test if it the imputed data is return, not passed by reference

1-1-CNAknn.csv : to see if the problem is how I NA out all the non-imputed data prior to writing the results

January 23 2019

The last job I started failed while trying to write to csv

test-1-1-pre-knn.csv : worked

test-1-1-post-knn.csv : looks the same as test-1-1-pre-knn.csv

test-1-1-knn-out.csv : failed

1-1-CNAknn.csv :

January 25, 2019

I believe I figured out put from DreamAI

It returns a list

The list contains a matrix

The matrix is a flattened version (is one column with 1715600 rows) of what you passed in

You can un-flatten the matrix using the follow commands

So I have started trying to use an alternative imputation package in R, MICE.

January 29, 2019

Started a job imputing just 10 NA using DreamAI with the output and my reformatting of it being written to disk as .rds files so I can easily play with the output of the function and double checkout it

ToDo: run a test on MICE on the super computer : running, not done

January 30, 2019

The DreamAI job worked, but I need to transpose the data, I was creating a new protein instead of a new person, just started job to test that. Maybe it will run faster this way?

The MICE job failed for time out issues at 8 hours. This makes me not want to use it because it was only doing 10 values which makes it really really slow.

I started running tests using missForest. It so far works and give intelligible results quicker than any other method so far.

In my emails with Shabranti thye said that missForrest for them can take 2-3 days to run, but mine goes faster than anything else, way faster than the rest of theirs.

I also emailed Shabranti a couple questions I have about their package, input vs output differences and why when I specify one method it does all 7.

ToDo: run full person imputation using missForest

February 4th 2019

The MissForest Job timed out after 24 hours, it accomplished 5000 out of 17,000 NAs.

Started 5 jobs of 400 each using missForest should take just under 24 hours to complete, this will constitute 1 person

ToDo: plots the distributions of the Real and Fake random and digit distribution data and see how that compares to what is described in Hill’s 1995 paper (you should have in this Mendely)

Payne Suggested training on CNA and Transcriptomics and testing on Proteomics, does it do better? Or work at all

Look at how the transcript omics have been normalized\*\*\*

February 5th 2019

The jobs for imputing with missForest all failed while writing the results because subsets did not have the same dimensions as the original when adding column names, DUM!!!

Started a new job with no column name reassignment

Created a new version of the missForrest-4000 script that creates the correct size column names and save the output as RDS files prior to that so that I can work on debugging later on if needed: started 5 new jobs with this

Generated Plots of average digit frequency of phony vs real data

Created a module for getting digit preferences that can be imported into each analysis script as needed. This way I can avoid code duplication and possible cross file changes