Project for NBA Hackathon Application

-- OBJECTIVE --

Given a training set of data relating to Instagram posts by @nba, including its engagements, predict the engagements for a holdout set.

-- PREPROCESS --

Preprocessing was the bulk of the project, and this involved a lot of trial and error. Many csv files were created to try and produce the best-engineered dataset possible for a model to train on. The following descriptions outline the chronological order of this trial/error process building up to the last csv file. The final predictions made by the model only use the last of these csv files. Feel free to skip to the last attempt, which discusses the last csv file.

1st Attempt:

code- preprocess.py step 1

output- training\_set\_datetimes.csv , holdout\_set\_datetimes.csv

1. split the “Created” column into several time-related columns using the public fastai library (available on Github)
2. ex. Created Year, Created Day, Created Month, etc.
3. problems: unnormalized, many unnecessary columns (fastai’s add\_datepart library caters mostly to financial data analysis), no usage of the description problem

2nd attempt:

code- preprocess.py step 2

output- training\_set\_final\_0.csv , holdout\_set\_final\_0.csv

1. dropped irrelevant problems, such as quarter-of-year, is\_month\_start/is\_month\_end, etc.
2. added a CreatedTotalMinutes column, which counts the number of minutes since midnight that the post was made
3. normalized Followers At Posting, CreatedDayofyear, CreatedTotalMinutes
4. problems: still no text inputs

3rd attempt:

code- preprocess.py step 3, all of preprocess\_text.py

output- training\_set\_final\_1.csv, holdout\_set\_final\_1.csv

1. parsed Descriptions column for Instagram usernames, and for each username:
   1. place it into one of four categories- NBA Player, NBA Team, Celebrity, Organization/Company (done manually)
   2. sometimes the username was misspelled, out of date, or referred to a player’s fan account rather than the player himself, so those errors were fixed as well (done manually)
   3. use GET requests to query Instagram for the number of followers of that account
   4. implemented in preprocess\_text.py steps 1,2,3
2. for each row in dataset:
   1. add 8 columns representing the number of accounts mentioned and sum of account followers, for each of the four categories
   2. ex. caption = “@kylelowry7 of @raptors hits a three against the @warriors!!”
      1. column player = 1
      2. column player\_followers = 3,000,000 (number of Kyle Lowry’s followers)
      3. column team = 2
      4. column team\_followers = 67,000,000 (sum or raptors’, warriors’ followers)
      5. repeat for celebrity, organization/company…
   3. implemented in preprocess\_text.py steps 3,4
3. text-related stats stored in text\_processing/training\_set\_stats\_final.csv and text\_processing/holdout\_set\_stats\_final.csv
4. combined training\_set\_final\_0.csv and holdout\_set\_final\_0.csv with above text data, normalized followers columns
   1. implemented in preprocess.py step 3
5. problems- followers columns are either 0 (no account mentioned) or in the millions (account mentioned) … should I still normalize?

4th attempt:

code- preprocess.py step 4

output- training\_set\_final\_2.csv, holdout\_set\_final\_2.csv

1. repeat above, but don’t normalize the follower columns

-- PROCESSING --

In the end, we used training\_set\_final\_2.csv for training, and holdout\_set\_final\_2 for predicting. Implementation is in process.py

* The predictions were made by a dense, feed-forward neural network
* The method dora\_the\_naive\_hyperparameter\_explorer takes a list of starting node sizes and a list of optimizers, and figures out the best node-size/optimizer combination
* The “best” model was determined using 5-fold cross-validation
* The final model had node structure 64 -> 32 -> 16 -> 8 -> 1 and used the Adadelta optimizer, with a validation MAPE of ~7.5%