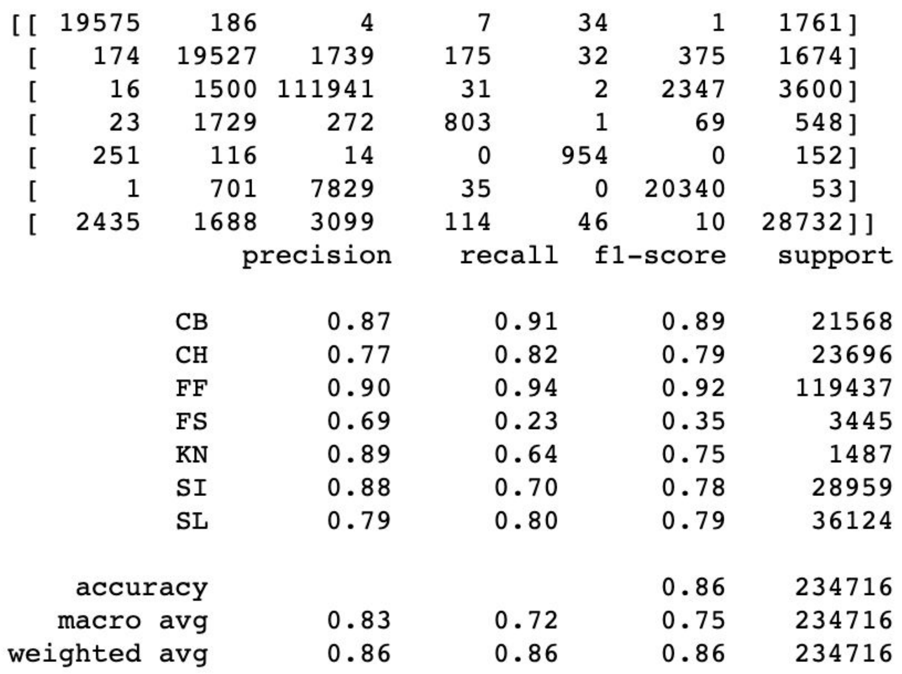
DS Challenge
Attached here is a link to a data file, containing all pitches from the 2011 season and the associated metadata describing the data (attached csv below).

1. Please review the dataset and start outlining the way you would go about the model-building process with the eventual goal of predicting the probability of a fastball, slider, etc., in a real-time environment.

- Preprocess
    - Firstly, I omitted columns with no values/information whatsoever
        - all columns from runner1_id to modified by
    - I dropped all rows with NA values in the label column (pitch_type) since those are of no use while training the model
        - In hindsight, might have created a mini test set out of these rows to use for evaluation
    - I checked which other columns have NA values
        - With my prior baseball knowledge and viewing all the columns that had NA values, I decided to drop all columns with NA values in them because the columns with NA values are essentially irrelevant when it comes to predicting pitch type
    - Detected which columns are non-numeric since those would not be appropriate for future steps
        - Got rid of a few non-numeric columns since they have no effect on pitch type probability
        - Adjusted batter_stand and pitcher_stand to binary value denoting 1 for righties and 0 for lefties
    - Converted Pitch Type using One Hot Encoding to create a separate column for each pitch type
        - Useful for finding correlations specifically with each pitch type and the features
    - Ran a correlation matrix and decided that the important features are all the features with an r^2 value of at least 0.1 in either direction because that value signifies that there is a somewhat significant correlation
    - Took a peek at the range of each feature
        - Realized that features need to scaled in the future to so that each feature contributes approximately proportionately to the final distance and so that gradient descent converges faster as it will be used in the future

- Feature Engineering
    - Insights from initial correlation matrix and viewing the pitch_type breakdown of dataset
        - Variations of Fastball (four-seam, two-seam, cut) can be generalized as simply fastball to increase accuracy
        - Variations of Curveball (curve, knuckle curve) can be generalized as simply curveball to increase accuracy
        - Remove Unidentified since they have no pitch type that we can train on (can be used as test)
        - pitch_out isn't necessarily a pitch type used in game often, no need to train on those pitches
        - Pitches with no correlation:
            - Fastballs (FA): only 204 such rows (<0.05% of data), remove to improve accuracy
            - Forkballs (FO): only 329 such rows (<0.05% of data), remove to improve accuracy
            - Euphus (EP): only 134 such rows (<0.05% of data), remove to improve accuracy
            - SC: Unknown pitch type to me and only 120 such rows (<0.05% of data), remove to improve accuracy
- Standardize
    - Applied a MinMaxScaler to adjust each features range to between 0 and 1 inclusive
    - Re-ran correlation matrix
        - Quick Insights
            - Break Length & pfx_z is a good indicator to classify Fastballs vs Curveballs (makes sense since curves break down and fastballs "break upwards")
            - Start and End speed are good indicators to classify Fastballs, Sinkers, Sliders, Curveballs, Knuckleballs, Changeups

- Modeling
    - Adjusted pitch types to generalizations made in feature engineering
    - Dropped “IN” pitch_type just before modeling because I do know know what pitch type that is and didn’t want uncertainty
        - Asked many people and browsed the web, no luck :/
        - Very curious to know what pitch type that is
    - Baseline Accuracy using majority classifier
        - Majority of dataset is fastballs
        - Majority Classifier would classify all pitches as FF (361299 / 711258) and achieve an accuracy of 50.7%
    - Split dataset into 66.66% train and 33.33% test
    - KNN



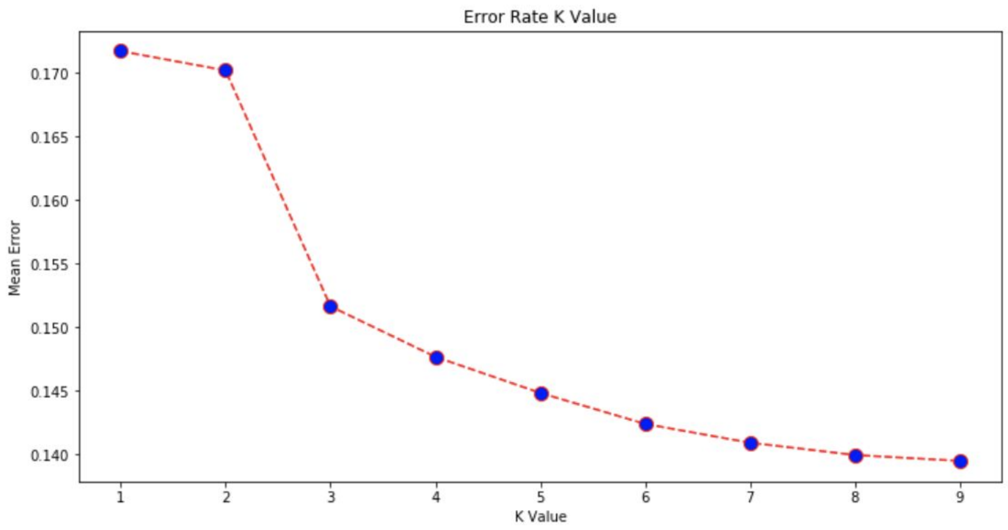
```
[[ 19575    186      4      7     34      1   1761]
 [   174  19527   1739    175     32    375   1674]
 [    16   1500 111941     31      2   2347   3600]
 [    23   1729    272    803      1     69    548]
 [   251    116     14      0    954      0    152]
 [     1    701   7829     35      0  20340     53]
 [  2435   1688   3099    114     46     10  28732]]
              precision    recall  f1-score   support

          CB       0.87      0.91      0.89     21568
          CH       0.77      0.82      0.79     23696
          FF       0.90      0.94      0.92    119437
          FS       0.69      0.23      0.35      3445
          KN       0.89      0.64      0.75      1487
          SI       0.88      0.70      0.78     28959
          SL       0.79      0.80      0.79     36124

    accuracy                           0.86    234716
   macro avg       0.83      0.72      0.75    234716
weighted avg       0.86      0.86      0.86    234716
```

        -
        - Overall Accuracy: 86%
        - Satisfactory accuracy for all pitch types except FS (splitters), 35% accuracy, which didn’t have any strong correlation between features given in dataset so it’s reasonable

- KNN with error rates at each k increment to determine the optimal number of k for this dataset



    -

- Logistic Regression
    - Failed: Failed to Converge

- GCD with Logistic Regression

```
[[ 17715    540      5      0      0     17   3291]
 [   306  13684   4766      0      0   1635   3305]
 [    15    639 111176      0      0   2787   4820]
 [    21   1411    846      0      0    245    922]
 [   653    426      2      0      0      0    406]
 [    12    387  17696      0      0  10561    303]
 [  2313   1306   3852      0      0    123  28530]]
```

```
/Users/ayusharora/opt/anaconda3/lib/python3.7/site-packages/
sklearn/metrics/classification.py:1437: UndefinedMetricWarni
ng: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
```

```
              precision    recall  f1-score   support

          CB       0.84      0.82      0.83     21568
          CH       0.74      0.58      0.65     23696
          FF       0.80      0.93      0.86    119437
          FS       0.00      0.00      0.00      3445
          KN       0.00      0.00      0.00      1487
          SI       0.69      0.36      0.48     28959
          SL       0.69      0.79      0.73     36124

    accuracy                           0.77    234716
   macro avg       0.54      0.50      0.51    234716
weighted avg       0.75      0.77      0.75    234716
```

-
- Accuracy: 77%
- Fails to classify splitters yet again as well as knuckleballs here, causing its overall accuracy to decrease
- Interesting how GCD classifier with logistic regression doesn't fail to converge
    - Scaling the data might be to thank for this model succeeding to process

2. We would like to see you build and evaluate a model that would be acceptable in a production environment after improvement. I understand actually delivering predictions with any degree of accuracy is unlikely in this short time span. Additionally, please provide any associated data analysis (plots, graphs etc..), feature engineering and code assembled during the 3-5 hours in the form of a .pdf or .html file.

- See Attachment

3. Lastly, please provide the future steps you would take from a data/technology perspective to finalize this project and the ways you would measure success.

- Standardizing
    - Future Work
        - Could blend break_angle,pfx_x, and ax since they are very similarly correlated
        - Could blend break_length, pfx_z, and az since they are very similarly correlated
        - Can eliminate one of vy0 and start_speed since they are EXACTLY the same
        - start and end speed are similarly correlated with vy0
        - More feature engineering can be done by looking at features individually and seeing how they relate to each individual pitch_type
- Modeling
    - Logistic Regression
        - Future Work
            - Determine what caused it to fail to converge
    - KNN
        - Would run multiple times with different k until accuracy is maximized
    - K-Means Clustering?
        - Might delve into unsupervised learning to see if there's any insight or increase in accuracy
    - Future Work
        - Rather than outputting a single pitch type as its prediction, output a % prediction probability of each pitch type