DISCLAIMER: Running preprocessing scripts first might not give the exact database csv files, which in turn could affect the model. This is because those scripts were changed a bit on the go, along with some manual changes in the csv files.

So, it might be a good idea to run the models first, and then the preprocessing scripts.

We've given which file is required for each model as input for the worst case, so those can always be replaced in case something gets changed. Sorry for the trouble.

Now, for running the models, you can simply run:

For Logistic regression and KNN classifier:

python KNN&Logistic.py

It uses the DB_2.csv file to import data

For Random Forest Classifier:

python RFC_model.py

It uses DB_2 and then does some further changes as mentioned in report

For Neural Nets:

python Neural_Net.py (This will take a lot of time due to the Grid Search)

It uses db2.csv and db2_test.csv (2019 data) and then some further changes are made specifically for implementing neural nets.

For SVM and Naive Bayes Classifier :

python GaussianNB_and_SVM.py

It uses final.csv as the dataset and removes the unnecessary columns from it. It shifts the mean of the player stats by 3. It then divides the dataset into train and test. The 2019 year is chosen as the test set. Then, it uses GaussianNB and (SVM with GridSearchCV) as the 2 models to predict the output. It uses f1-score as the metric. Further, it also calculates the total wins for each team in the year 2019.

For web scraping match lists:

python WebScraper.py

This will generate the match list for a particular year given the link for the cricbuzz website. Files for 2008-2019 are provided.

These were then manually checked for all years(2008-2019), cleaned and combined into a single file.

In the Scraper Code folder,

scrapy crawl matchdetails_scraper -o 2009_matchdetails.csv

The above code should be run from outside the first matchdetails_scraper folder in a virtual environment.It will generate the match details of all the matches played in 2009 and their URLs in the 2009_matchdetails.csv file.

python scorecard_scraper.py

The above command should be run to get the scorecard of all the matches played in 2009 in the 2009_player.csv file

Sample file for 2018 is provided.

Now for feature construction, functions are given and can be tried from the given scripts. It takes the file generated by the web scraper and a separate excel containing pre-constructed player features. There's also a separate script to convert the winner column to 1/0 and to put player features into an excel file. Note that no single final database is included(ignore the names). Different models have used different sets of features to improve performance.

This can be tried by:

python Player_feature_extraction.py
python Preprocessing.py
python classes_to_binary.py

P.S: Exact output might not be reproduced, but these were the functions that were used with some manual changes to get the required databases. Some intermediate files have been given in the data folder to give an idea of how we proceeded in steps.

For generating team_stats :

python team_stats_yearwise.py
Uses 2010_matches.csv upto 2018_matches.csv to generate the team level features.
On running the code, you will get the team level stats for only 1 year. The same code was modified for each year and run again.

For assigning more relevance to recent data :
python relevance.py
Uses the 2010_player_stats.csv to 2018_player_stats.csv to generate player_stats_final.csv by assigning relevance to each year.