My team name on kaggle is “raymond”.

For play-prediction, I tried a few different approaches. First I used a user-based Jaccard similarity. I found all the games played by the user. For each of the games, I found the Jaccard similarity between the users who have played the game and the users who have played the target game. I recorded the highest similarity, set the threshold and filter based on the threshold. This approach did not yield very good results (~68%). Next I tried using cosine similarity. First I constructed a cosine similarity matrix between user and user. For each target game, I found all users who have played that game. I then found the cosine similarity between the user and the target user and kept the highest score. If the highest score is above a certain threshold, which means players who played the game are similar to our target user, I predict true. This yielded a slightly better result of around 70% on the validation score. Lastly, I incorporated the baseline model where we predict true if the game is popular. I counted how many players have played each game, and then divided by a parameter. I first used 1.4 as the parameter, but then got many false positives. I therefore used 1.8. Lastly, I combined this with the cosine similarity, only predicting true if the user-item satisfies both the similarity threshold and the popularity threshold. Using 0.12 as the threshold for similarity and 1.8 for popularity parameter, I was eventually able to achieve an accuracy score of 70.1%.

For the category-prediction, I trained a couple models on the training data and compared each one’s accuracy on the validation set. First, I built the bag of words model. I put all text in lowercase and took out all the punctuations. For the number of most common words, I chose 1000, 2000, 2500, and 3000. 2500 ended up providing the best prediction accuracy so I chose that. Then I fitted the vectors to a logistic regression model with a couple different values of the inverse regularization constant C. I ended up using C=2 as that provides the best accuracy. With this bag-of-words model I was able to achieve an accuracy of 70%. Then I tried to improve the model by adding stemmer to avoid repetition of similar words and improved the accuracy to 72%. Lastly I tried to implement a multinomial naive Bayes model. I used the CountVectorizer library function provided by sklearn and choose min_df equals 5 (minimum

number of text a word needs to be in). I fitted it to a Sklearn Multinomial Naive Bayes and it resulted in an accuracy of 74% with the validation data. With that, I ended up choosing the Bayes classifier.