Methodology:

Firstly, a function *getFeature* was defined with the purpose of generating feature scores for each endpoint. The idea is to give this function a group of end points and the function with extract feature score from the end points. The *getFeature* function returns a dictionary of scores for each feature for each node

Which features to use? Looking at the data from a social perspective ideally it was important to select features such as *Preferential Attachment* which basically calculates a score for how popular a node is (how many connections it already has, think of an influencer). This can be a great indicator for our model to suggest if a connection will be made between two end points depending on the popularity of either point. *Common Neighbours* was also selected, just like mutual friends, a connection is much more likely to be created if two nodes (or people) have other nodes in common (mutuals). Finally, the *Jaccard Coefficient* was used as this calculates the shared number of nodes or the ratio of shared nodes (friends) to the total number of unique nodes. This is a measure of similarity between the two nodes.

Once this function was created the machine learning model used to predict links could be built. The first stage, a model had to be chosen and trained from the edge list file. Logistic regression was chosen as the specific model to be used and several functions were created to train the model.

Using a function called *get\_node\_pairs,* this function would take our edge list file and create two lists of positive and made-up negative pairs. The positive pairs were simply a list of the original edges in the network and the negative ones were generated randomly. It is worth mentioning that having a 1:1 ratio of positive to negative pairs in the training set is a good starting point for training a predicition model. However from test different ratios, a 1:3 positive to negative ratio yielded the highest accuracy scores.

Second step, the pairs could be passed through a function called *extract\_features* which would iterate through each node pairing and create a list of feature scores for each. These feature scores will be what we use to train our logistic regression model to determine if a link will be present or not.

Third step involved assigning both positive (1) y labels to the positive pairs and negative (0) y labels to the negative pairs. This was done in the *prepare\_labels* function.

Fourth step, training the logistic model which was done in the *train\_model* function.

Final step was to evaluate our model using the *accuracy\_score* and *roc\_auc\_score* method.

Running the above functions on the original edge list file the model returns an accuracy of 83.7% and an AUC score of 70.7%. This model was then used to predict links within the solution input file and return scores of 84% for accuracy and 72.2% for AUC.

Potential steps to improve this metric to be to potentially and explain-ably use other models like decision tree or using the attribute file, train to model to associate nodes with similar attributes to be more likely to connect.