References

[1] A. Kaluarachchi and S. V. Aparna, “CricAI: A classification based tool to predict the outcome in ODI cricket,” in Information and Automation for Sustainability (ICIAFs), 2010 5th International Conference on, 2010, pp. 250–255.

[2] T. Singh, V. Singla, and P. Bhatia, “Score and winning prediction in cricket through data mining,” in Soft Computing Techniques and Implementations (ICSCTI), 2015 International Conference on, 2015, pp. 60–66.

[3] Y. Saito, M. Kimura, and S. Ishizaki, “Real-time prediction to support decision-making in soccer,” in Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K), 2015 7th International Joint Conference on, 2015, vol. 1, pp. 218–225.

[4] V. V. Sankaranarayanan, J. Sattar, and L. V. Lakshmanan, “Auto-play: A data mining approach to ODI cricket simulation and prediction,” in Proceedings of the 2014 SIAM International Conference on Data Mining, 2014, pp. 1064–1072.

[5] A. Shah, D. Jha, and J. Vyas, “WINNING AND SCORE PREDICTOR (WASP) TOOL.”

[6] R. K. Khan, I. Manarvi, and others, “Evaluating performance of Blackcaps of New Zealand vs. global cricket teams,” in Computers & Industrial Engineering, 2009. CIE 2009. International Conference on, 2009, pp. 1500–1504.

[7] I. Bhandari, E. Colet, and J. Parker. Advanced Scout:Data mining and knowledge discovery in NBA data.Data Mining and Knowledge Discovery, 1(1):121{125,1997.

[8] D. Lutz. A cluster analysis of NBA players. In MITSloan Sports Analytics Conference, 2012.

[9] G. Gartheeban and J. Guttag. A data-driven method for in-game decision making in mlb: when to pull a starting pitcher. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '13, pages 973{979, New York, NY, USA, 2013. ACM.

[10] S. Luckner, J. Schroder, and C. Slamka. On the forecast accuracy of sports prediction markets. In Negotiation, Auctions, and Market Engineering, International Seminar, Dagstuhl Castle, volume 2, pages 227{234,2008.

[11] Khabir Uddin Mughal. Top 10 Most Popular Sports In The World. http://sporteology.com/top-10-popular-sports-world/Accessed 2 February 2015.

[12] Allsopp, P., & Clarke, S. R., Rating teams and analyzing outcomes in one-day and test cricket. Journal ofthe Royal Statistical Society A, (2004).

[13] S.R. Clarke, “Dynamic programming in one-day cricket— optimal scoring rates”, Journal of the Operational Research Society, 1988, Vol. 39, No. pp. 331–337.

[14] B.M De Silva, and T.B. Swartz, Estimation of the magnitude of the victory in one-day cricket. Australia and New Zealand Journal of Statistics, 2001, Vol. 43, pp. 1369-1373.

[15] V. Veppur Sankaranarayanan, "Towards a time-lapse prediction system for cricket matches," PhD Thesis, University of British Columbia, 2014.

[16] D. Prasetio and others, "Predicting football match results with logistic regression," in *Advanced Informatics: Concepts, Theory And Application (ICAICTA), 2016 International Conference On*, 2016, pp. 1–5.

[17] "Weka 3 - Data Mining with Open Source Machine Learning Software in Java." [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/. [Accessed: 07-May-2018].

[18] rpetrusha, "Language Independence and Language-Independent Components." [Online]. Available: https://docs.microsoft.com/en-us/dotnet/standard/language-independence-and-language-independent-components. [Accessed: 07-May-2018].

[19] "Microsoft Launches Its .NET Distribution For Linux And Mac," *TechCrunch*, 29-Apr-2015. .

[20] R. P. Schumaker, O. K. Solieman, and H. Chen, "Open Source Data Mining Tools for Sports," in *Sports Data Mining*, vol. 26, Boston, MA: Springer US, 2010, pp. 89–92.

[21] "Digital Scout." [Online]. Available: https://www.digitalscout.com/. [Accessed: 18-May-2018].

[22] F. Schoonjans, "Logistic regression," *MedCalc*. [Online]. Available: https://www.medcalc.org/manual/logistic_regression.php. [Accessed: 18-May-2018].

[23] "Lightning Fast Data Science Platform | RapidMiner." [Online]. Available: https://rapidminer.com/. [Accessed: 18-May-2018].

[24] heatherbshapiro, "What is Azure Machine Learning Studio?" [Online]. Available: https://docs.microsoft.com/en-us/azure/machine-learning/studio/what-is-ml-studio. [Accessed: 18-May-2018].

Appendix – A

Attribute Selection

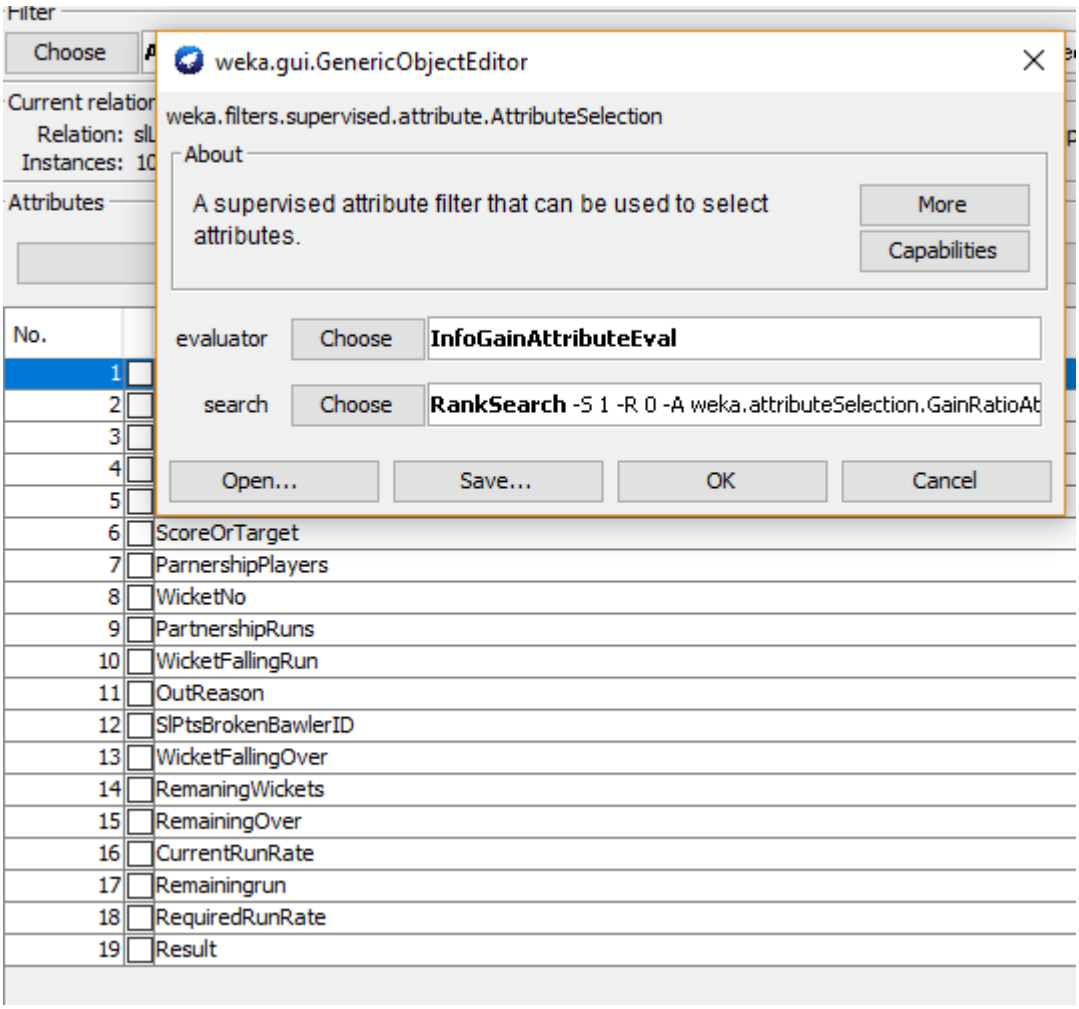


Figure A: 1 Information Gain Attribute Evaluation

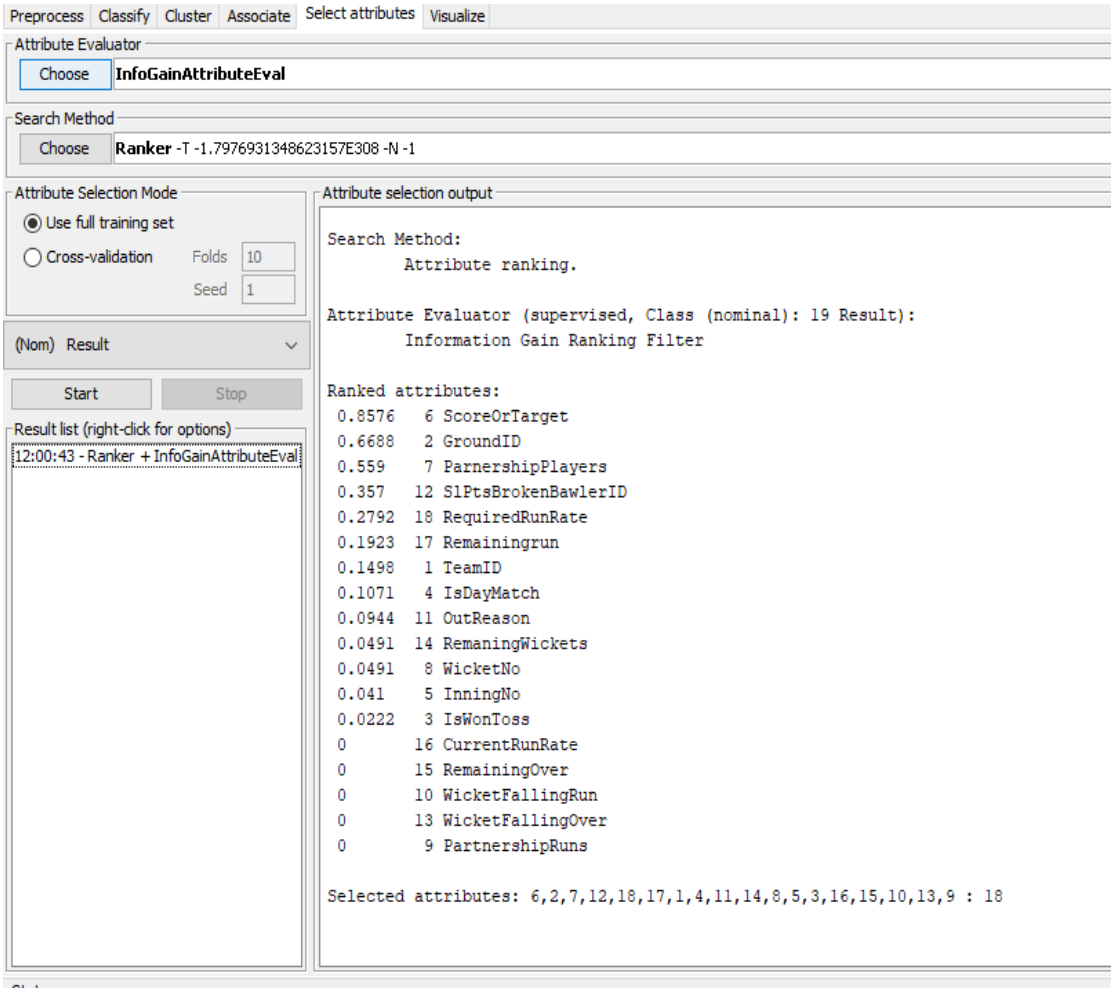
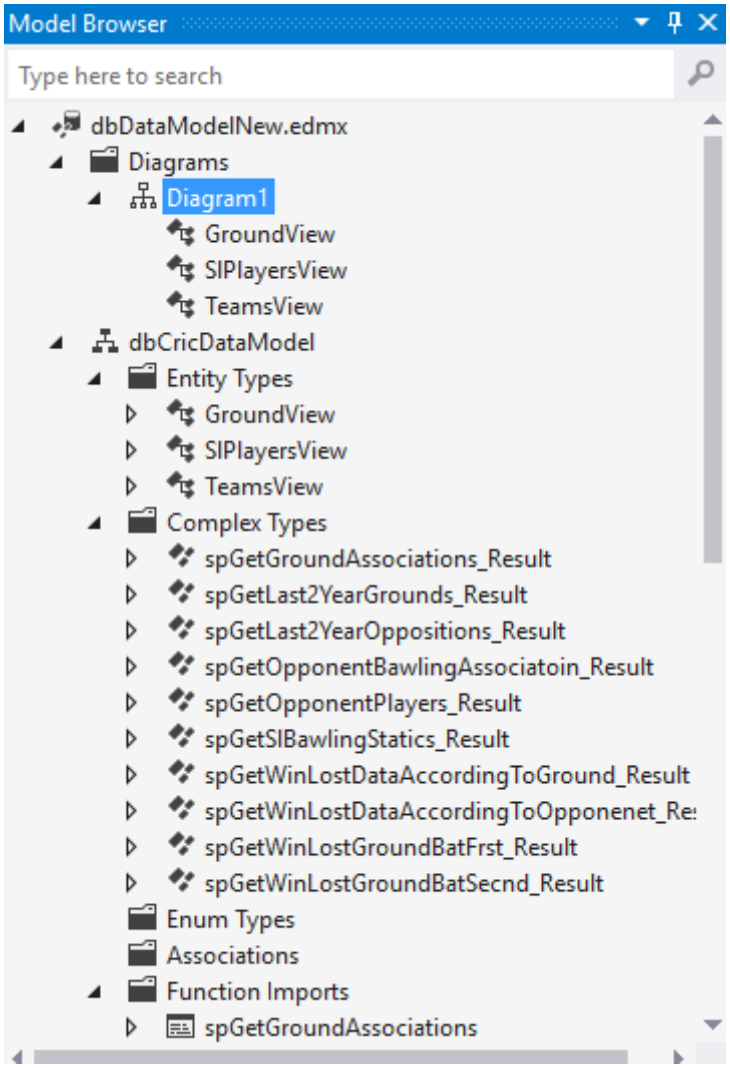


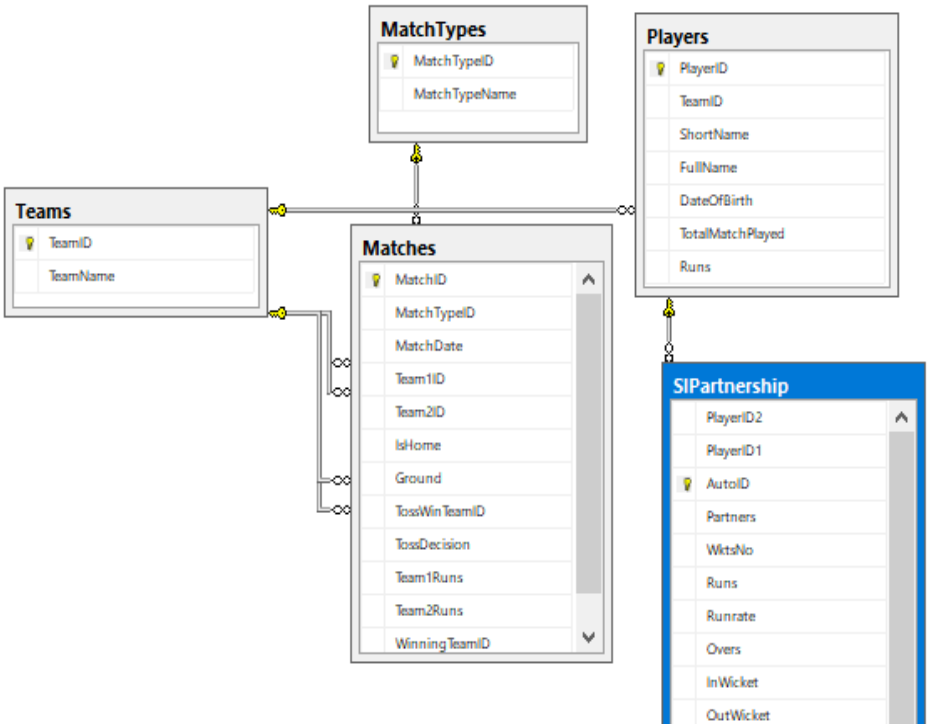
Figure A: 2 Attribute Selection

Appendix - B

Backend Data Module Implementation



Appendix B: 1 Entity Framework Model Browser



Appendix B: 2 Back-End Database Diagram

Appendix - C

Model Evaluation Summary

```
Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          54               98.1818 %
Incorrectly Classified Instances         1                1.8182 %
Kappa statistic                          0.9364
Mean absolute error                      0.0216
Root mean squared error                  0.1371
Relative absolute error                  7.0812 %
Root relative squared error             35.5068 %
Total Number of Instances               55

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.9       0          1         0.9       0.947      0.978    Won
                 1         0.1        0.978     1         0.989      0.978    Lost
Weighted Avg.    0.982     0.082      0.982     0.982     0.981      0.978
```

Appendix C: 1 Naive Bayes Model Creation Summary

Classifier output

```
Size of the tree :         83


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          77               97.4684 %
Incorrectly Classified Instances         2                2.5316 %
Kappa statistic                          0.9217
Mean absolute error                      0.0253
Root mean squared error                  0.143
Relative absolute error                  7.3896 %
Root relative squared error             34.7324 %
Total Number of Instances               79

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.882     0          1         0.882     0.938      0.998    Won
                 1         0.118      0.969     1         0.984      0.998    Lost
Weighted Avg.    0.975     0.092      0.975     0.975     0.974      0.998
```

Appendix C: 2 Decision Tree Model Creation Summary

```
Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          61               77.2152 %
Incorrectly Classified Instances        18               22.7848 %
Kappa statistic                          0.0379
Mean absolute error                      0.225
Root mean squared error                  0.3489
Relative absolute error                 65.666  %
Root relative squared error             84.7424 %
Total Number of Instances               79

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.059     0.032      0.333     0.059      0.1        0.867    Won
                 0.968     0.941      0.789     0.968      0.87       0.867    Lost
Weighted Avg.    0.772     0.746      0.691     0.772      0.704      0.867
```

Appendix C: 3 AdaBoost Model Creation Summary

```
Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          57               72.1519 %
Incorrectly Classified Instances        22               27.8481 %
Kappa statistic                         -0.0483
Mean absolute error                      0.2969
Root mean squared error                  0.3797
Relative absolute error                 86.6652 %
Root relative squared error             92.221  %
Total Number of Instances               79

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.059     0.097      0.143     0.059      0.083      0.829    Won
                 0.903     0.941      0.778     0.903      0.836      0.829    Lost
Weighted Avg.    0.722     0.759      0.641     0.722      0.674      0.829
```

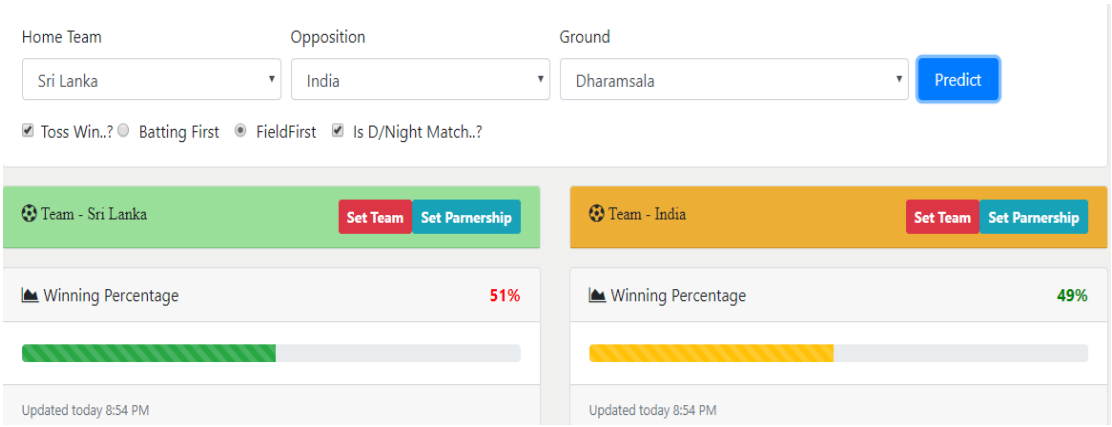
Appendix C: 4 RandomForest Model Creation Summary

```
Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          55               69.6203 %
Incorrectly Classified Instances        24               30.3797 %
Kappa statistic                         -0.1435
Mean absolute error                      0.3692
Root mean squared error                  0.4611
Relative absolute error                107.7608 %
Root relative squared error            111.9925 %
Total Number of Instances               79

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0         0.113      0          0         0          0.4      Won
                 0.887     1          0.764      0.887     0.821      0.402    Lost
Weighted Avg.    0.696     0.809      0.6        0.696     0.644      0.402
```

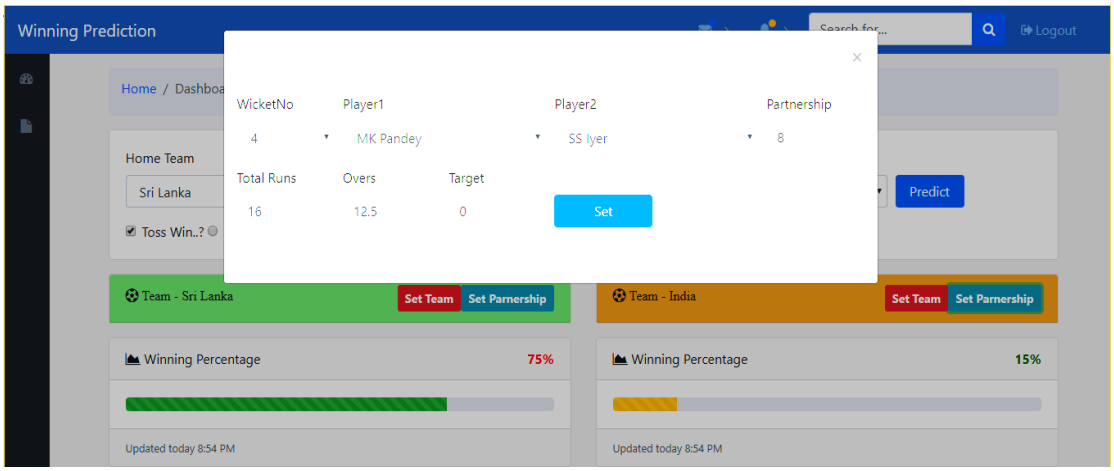
Appendix C: 5 Bagging Model Creation Summary

Appendix - D

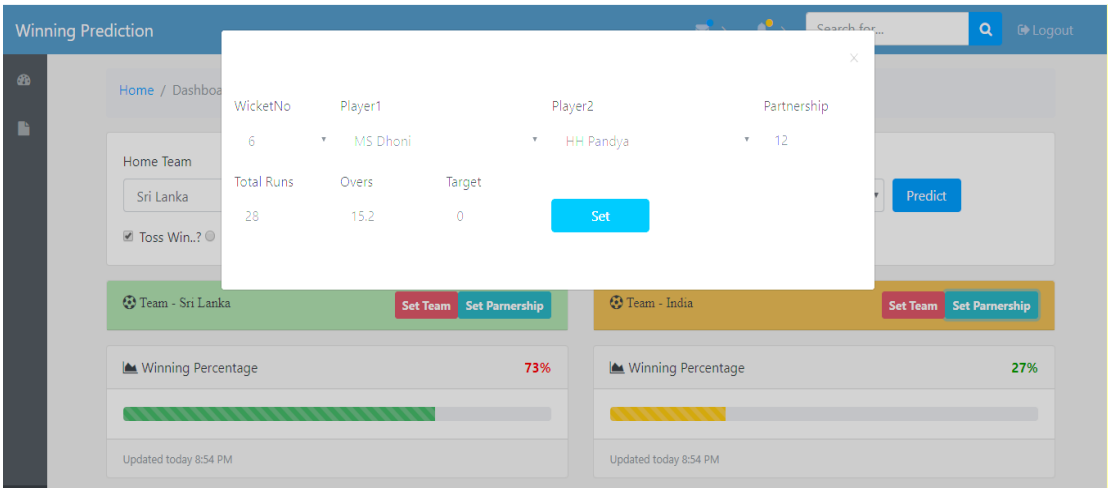
CRIC-Win Predictor Tool Test Result



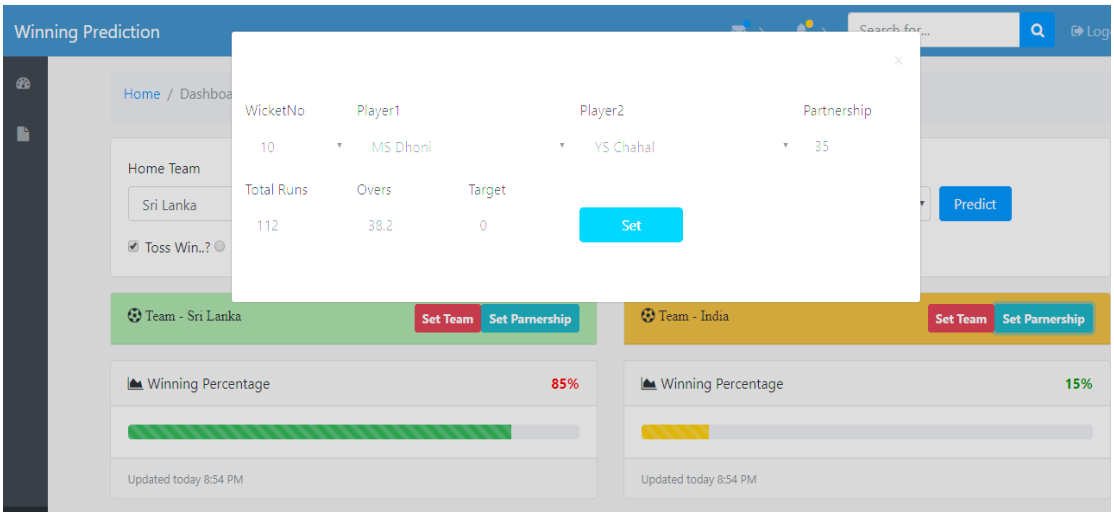
Appendix D: 1 SL vs India Match Outcome



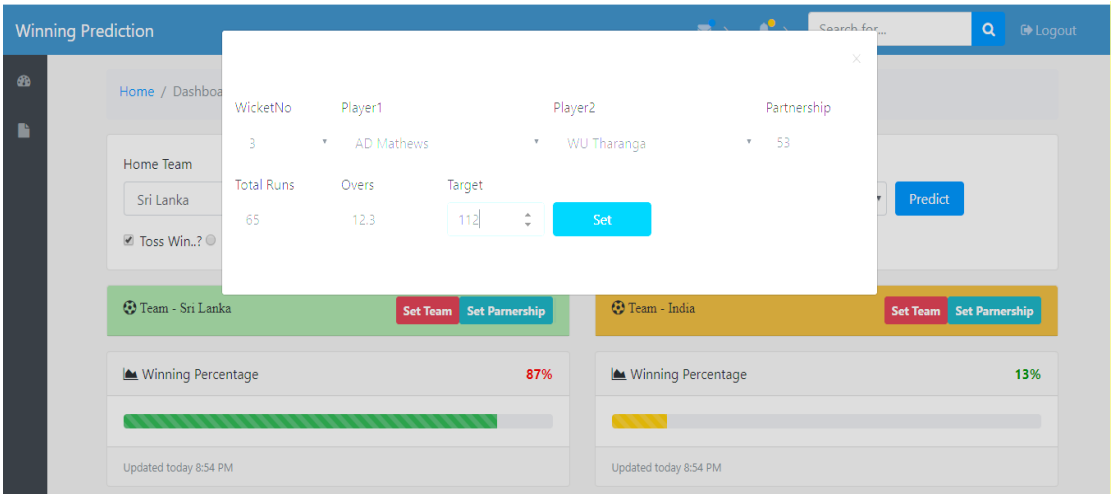
Appendix D: 2 1st Inning 4th Wicket Partnership



Appendix D: 3 1st Inning 6th Wicket Partnership



Appendix D: 4 1st Inning Last Wicket Partnership



Appendix D: 5 2nd Inning 3rd Wicket Partnership

Appendix - E

Code Snippet

```
static void Main(string[] args)
      {
         Entities db = new Entities();
         List<SlPartnership> lstSlMatch2017 = new List<SlPartnership>();
         HtmlWeb hw = new HtmlWeb();
         HtmlDocument hDoc =
hw.Load(@"http://stats.espncricinfo.com/ci/engine/stats/index.html?class=2;home_or_away=
1;host=8;result=1;result=2;spanmin1=14+Jan+2008;spanval1=span;team=8;template=results;
type=allround;view=results");
         HtmlNodeCollection rows =
hDoc.DocumentNode.SelectNodes("//*[@id=\"ciHomeContentlhs\"]/div[3]/table[3]/tbody/tr"
);
         for (int i = 1; i <= rows.Count; i++)
         {
            HtmlNodeCollection columns =
hDoc.DocumentNode.SelectNodes(string.Format("//*[@id=\"ciHomeContentlhs\"]/div[3]/tab
le[3]/tbody/tr" + "[" + "{0}" + "]" + " /td", i));

            SlPartnership obj = new SlPartnership
            {
               Partners = columns[0].InnerText,
               WktsNo = columns[1].InnerText,
               Runs = columns[2].InnerText,
               Overs = columns[3].InnerText,
               Runrate = columns[4].InnerText,
               InWicket = columns[3].InnerText,
               OutWicket = columns[4].InnerText,
               FirstOrSecndInning = columns[5].InnerText,
               Opposition = columns[7].InnerText,
               Ground = columns[8].InnerText,
               MatchStart = columns[9].InnerText,
               IsDayMatch = false,
               IsWinMatch = false,
               IsWonToss = false

            };
            //lstSlMatch2017.Add(obj);


         }

        db.SlPartnership.AddRange(lstSlMatch2017);
        db.SaveChanges();

      }
```

Appendix E: 1 Web Cricket Data Scraping

```
HomeServices objHomeServices;
        public ActionResult Home()
        {
            ViewBag.slPlayerGrid = BGrid.CreateBGrid<SlPlayersView>("slPlayerGrid",
GetFirstApprovalPendingGridModel(), new List<SlPlayersView>(), new GridOptions() {
EditOption = false, EditFunctionName = "edit" });
            ViewBag.OppPlayerGrid = BGrid.CreateBGrid<SlPlayersView>("OppPlayerGrid",
GetFirstApprovalPendingGridModel(), new List<SlPlayersView>(), new GridOptions() {
EditOption = false, EditFunctionName = "edit" });
            return View();
        }

        public JsonResult GetTeams(bool isOverall)
        {
            objHomeServices = new HomeServices();
            var reslt = objHomeServices.GetTeamData(isOverall);
            return Json(reslt, JsonRequestBehavior.AllowGet);
        }

        public JsonResult getOpponentPlayers(bool isOverall,int TeamID)
        {
            objHomeServices = new HomeServices();
            var reslt = objHomeServices.GetOpponentAllPlayers(isOverall,TeamID);
            return Json(reslt, JsonRequestBehavior.AllowGet);
        }


        public JsonResult GetGronds(bool isOverall)
        {
            objHomeServices = new HomeServices();
            var reslt = objHomeServices.GetGrounds(isOverall);
            return Json(reslt, JsonRequestBehavior.AllowGet);
        }

        public JsonResult GetGroundAssociations(string GroundID)
        {
            objHomeServices = new HomeServices();
            var reslt = objHomeServices.GetGroundAssocations(GroundID);
            return Json(reslt, JsonRequestBehavior.AllowGet);
        }

        public JsonResult GetOppositionBawlingAssociations(string TeamID)
        {
            objHomeServices = new HomeServices();
            var reslt = objHomeServices.GetOppositionBawlingAssociations(TeamID);
            return Json(reslt, JsonRequestBehavior.AllowGet);
        }
```

Appendix E: 2 Fetch Data from Backend Module

```
public WinLostCountView GetGroundWiseBattingSecns(int groundID)
      {
         try
         {
            db = new dbCricDataEntities();
            List<spGetWinLostDataAccordingToGround_Result> lst =
db.spGetWinLostDataAccordingToGround(groundID).ToList();
            WinLostCountView objWinlossView = new WinLostCountView();
            foreach (var item in lst)
            {
               if (item.Result == "Won")
               {
                  if (item.Count == 0)
                  {
                     objWinlossView.WinCount = 0;
                  }
                  else
                  {
                     objWinlossView.WinCount = Convert.ToInt16(item.Count);
                  }

               }
               else if (item.Result == "Lost")
               {
                  if (item.Count == 0)
                  {
                     objWinlossView.LossCount = 0;
                  }
                  else
                  {
                     objWinlossView.LossCount = Convert.ToInt16(item.Count);
                  }

               }

            }
            if (lst.Count == 0)
            {
               objWinlossView.LossCount = 0;
               objWinlossView.LossCount = 0;
            }
            return objWinlossView;
         }
         catch (Exception)
         {

            throw new Exception("Run Time Error Occured");
         }
      }
```

Appendix E: 3 Backend Service Module code Snippet

```
package OdiPredict;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import weka.classifiers.Classifier;
import weka.core.Instances;
import weka.core.converters.ConverterUtils;

/**
 *
 * @author Dinesh
 */
@WebService(serviceName = "last2yearsOverall")
public class last2yearsOverall {

    @WebMethod(operationName = "NaiveBayersLasttwoYearsSlOverallMatchPredict")
    public String NaiveBayersLasttwoYearsSlOverallMatchPredict(String modelFileSerialized,
String testFileARFF)
    throws Exception
    {
    // Deserialize the classifier.
    Classifier classifier =
        (Classifier) weka.core.SerializationHelper.read(
            modelFileSerialized);

    // Load the test instances.
    Instances testInstances = ConverterUtils.DataSource.read(testFileARFF);

    // Mark the last attribute in each instance as the true class.
    testInstances.setClassIndex(testInstances.numAttributes()-1);

    int numTestInstances = testInstances.numInstances();
//    System.out.printf("There are %d test instances\n", numTestInstances);

    //Loop over each test instance.
    String reslt = "";
    for (int i = 0; i < numTestInstances; i++)
    {
        // Get the true class label from the instance's own classIndex.
        String trueClassLabel =
            testInstances.instance(i).toString(testInstances.classIndex());

        // Make the prediction here.
        double predictionIndex =
            classifier.classifyInstance(testInstances.instance(i));

        // Get the predicted class label from the predictionIndex.
        String predictedClassLabel =
            testInstances.classAttribute().value((int) predictionIndex);

        // Get the prediction probability distribution.
        double[] predictionDistribution =
            classifier.distributionForInstance(testInstances.instance(i));
```

```
      reslt +=  "ID:"+Integer.toString(i) +","+"Prediction:"+predictedClassLabel.toString()+",";
//System.out.println(i+ trueClassLabel.toString()+ predictedClassLabel.toString());
       // Print out the true label, predicted label, and the distribution.
//        System.out.printf("%5d: true=%-10s, predicted=%-10s, distribution=",
//                          i, trueClassLabel, predictedClassLabel);


       // Loop over all the prediction labels in the distribution.
       for (int predictionDistributionIndex = 0;
            predictionDistributionIndex < predictionDistribution.length;
            predictionDistributionIndex++)
       {
          // Get this distribution index's class label.
          String predictionDistributionIndexAsClassLabel =
             testInstances.classAttribute().value(
                predictionDistributionIndex);

          // Get the probability.
          double predictionProbability =
             predictionDistribution[predictionDistributionIndex];

          reslt +="Probability
"+predictionDistributionIndexAsClassLabel.toString()+":"+Double.toString(Math.round(pred
ictionProbability * 100D) / 100D)+" ,";
       }
       reslt += "\n";
    }
    return reslt;
}
}
```

Appendix E: 4 Weka API Web Service Code Snippet