```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6  using DAL;
 7
 8  namespace UpdateDB
 9  {
10      class Program
11      {
12          static void Main(string[] args)
13          {
14              FPLFunctions.numofplayers();
15              //predictions();
16          }
17          public static void predictions()
18          {
19              Dictionary<int, string> dic = FPLFunctions.getdicOfClubs();
20              for (int i = APICall.getCurrentGameweek(0); i < 39; i++)
21              {
22                  foreach (prediction pred in GameFunctions.byGameweek(i))
23                  {
24                      Console.WriteLine("home team: " + dic[pred.hteam]);
25                      Console.WriteLine("away team: " + dic[pred.ateam]);
26                      predict(pred.hteam, pred.ateam, pred.gameID);
27                      Console.WriteLine("\n");
28                  }
29                  Console.WriteLine("done with gw" + i);
30              }
31              Console.WriteLine("done with all");
32              Console.Read();
33          }
34          public static void predict(int htea, int ateam, int matchID)
35          {
36              float avggoalsscoredH = FPLFunctions.getallH() / (float)
                  FPLFunctions.getNumHPlayByTeamId(1) / 20;
37              //Console.WriteLine("avrage num of goals scored at home " +
                  avggoalsscoredH);
38              float avggoalconceededH = FPLFunctions.getallA() / (float)
                  FPLFunctions.getNumHPlayByTeamId(1) / 20;
39              //Console.WriteLine("avrage num of goals conceded at home "+
                  avggoalconceededH);
40
41
42              float hattstrength = attackstrength(htea, avggoalsscoredH, true);
43              float aattstrength = attackstrength(ateam, avggoalconceededH, false);
44              //Console.WriteLine("attacking strength for home team: " +
                  hattstrength);
45              //Console.WriteLine("attacking strength for away team: " +
                  aattstrength);
46
```

```
47
48
49              float hdefstrength = defencetrength(htea, avggoalconceededH ,true);
50              float adefstrength = defencetrength(ateam, avggoalsscoredH ,false);
51              //Console.WriteLine("defence strength for home team: " +
                  hdefstrength);
52              //Console.WriteLine("defenve strength for away team: " +
                  adefstrength);
53
54              float exgH = avggoalconceededH * adefstrength * hattstrength;
55              //Console.WriteLine("xg for home team: "+ exgH);
56              float exgA = avggoalconceededH * hdefstrength * aattstrength;
57              //Console.WriteLine("xg for away team: " + exgA);
58
59
60              //Console.WriteLine("home prob");
61              double[] a = posDeb(exgH, 5);
62              foreach (double d in posDeb(exgH,5))
63              {
64                  //Console.WriteLine(d);
65              }
66              //Console.WriteLine("away prob");
67              double[] b = posDeb(exgA, 5);
68              foreach (double d in b)
69              {
70                  //Console.WriteLine(d);
71              }
72
73
74              double[,] outcomes = multipleOutcomes(a, b);
75              //Console.WriteLine(outcomes.Length);
76              for (int i = 0; i < outcomes.GetLength(0); i++)
77              {
78                  for (int j = 0; j < outcomes.GetLength(1); j++)
79                  {
80                      //Console.Write(outcomes[i,j]+", ");
81                  }
82                  //Console.WriteLine("\n");
83              }
84              double draw = Math.Round(addupoutcomes(outcomes, "d") * 100);
85              double home = Math.Round(addupoutcomes(outcomes, "H") * 100);
86              double away = Math.Round(addupoutcomes(outcomes, "a") * 100);
87              Console.WriteLine("draw: "+draw+"%");
88              Console.WriteLine("Home Win: "+home + "%");
89              Console.WriteLine("Away Win: " + away + "%");
90              GameFunctions.addPrecent(home, draw, away, matchID);
91          }
92          public static float addupoutcomes(double[,] outcomes, string who)
93          {
94              if (who.ToUpper() == "D")
95              {
96                  float total = 0;
```

```
97                  for (int i = 0; i < outcomes.GetLength(1); i++)
98                  {
99                      total += (float)outcomes[i, i];
100                 }
101                 return total;
102             }
103             else if (who.ToUpper() == "H")
104             {
105                 float total = 0;
106                 int not = 1;
107                 for (int i = 1; i < outcomes.GetLength(1); i++)
108                 {
109                     for (int j = 0; j < not; j++)
110                     {
111                         total += (float)outcomes[i, j];
112                     }
113                     not++;
114                 }
115                 return total;
116             }
117             else
118             {
119                 float total = 0;
120                 int not = 1;
121                 for (int i = 0; i < outcomes.GetLength(1); i++)
122                 {
123                     for (int j = not; j < outcomes.GetLength(1); j++)
124                     {
125                         total += (float)outcomes[i, j];
126                     }
127                     not++;
128                 }
129                 return total;
130             }
131         }
132         public static double[,] multipleOutcomes(double[] a, double[] b)
133         {
134             double[,] ret = new double[a.Length, a.Length];
135             for (int i = 0; i < a.Length; i++)
136             {
137                 for (int j = 0; j < a.Length; j++)
138                 {
139                     ret[i, j] = a[i] * b[j];
140                 }
141             }
142             return ret;
143         }
144         public static double[] posDeb(float L, int n)
145         {
146             double[] final = new double[n + 1];
147 
148             for (int i = 0; i < n + 1; i++)
```

```
149             {
150                 final[i] = (Math.Pow(L, i) * Math.Pow(Math.E, -L)) / factorial
                      (i);
151             }
152             return final;
153         }
154         public static double factorial(int n)
155         {
156             if (n == 0)
157                 return 1;
158             int res = 1;
159             while (n != 1)
160             {
161                 res = res * n;
162                 n = n - 1;
163             }
164             return res;
165         }
166         public static float attackstrength(int clubId, float avg, bool home)
167         {
168             int GF = 1;
169             int played = 1;
170             if (home)
171             {
172                 played = FPLFunctions.getNumHPlayByTeamId(clubId);
173                 GF = FPLFunctions.getGFHByTeamId(clubId);
174             }
175             else
176             {
177                 played = FPLFunctions.getNumAPlayByTeamId(clubId);
178                 GF = FPLFunctions.getGFAByTeamId(clubId);
179             }
180             return GF / (float)played / avg;
181         }
182         public static float defencetrength(int clubId, float avg, bool home)
183         {
184             int GA = 1;
185             int played = 1;
186             if (home)
187             {
188                 played = FPLFunctions.getNumHPlayByTeamId(clubId);
189                 GA = FPLFunctions.getGAHByTeamId(clubId);
190             }
191             else
192             {
193                 played = FPLFunctions.getNumAPlayByTeamId(clubId);
194                 GA = FPLFunctions.getGAAAByTeamId(clubId);
195             }
196             return GA / (float)played / avg;
197         }
198     }
199 }
```