International Interaction Game

Signorino’s Backward Induction Model

Explain here the Signorino tree based model

Backward Induction Functions

```
In[ ]:= q10classical[U2War1_, U2Cap2_, l1_, l2_] :=
         Exp[l2 * U2War1] / (Exp[l2 * U2War1] + Exp[l2 * U2Cap2])

        notq10classical[U2War1_, U2Cap2_, l1_, l2_] :=
         1 - q10classical[U2War1, U2Cap2, l1, l2]

        q11classical[U2War1_, U2Cap2_, l1_, l2_] :=
         Exp[l2 * U2War1] / (Exp[l2 * U2War1] + Exp[l2 * U2Cap2])

        notq11classical[U2War1_, U2Cap2_, l1_, l2_] :=
         1 - q11classical[U2War1, U2Cap2, l1, l2]

        p8classical[U1War2_, U1Cap1_, l1_ : 1, l2_ : 1] :=
         Exp[l1 * U1War2] / (Exp[l1 * U1War2] + Exp[l1 * U1Cap1])

        notp8classical[U1War2_, U1Cap1_, l1_ : 1, l2_ : 1] :=
         1 - p8classical[U1War2, U1Cap1, l1, l2]

        p12classical[U1War2_, U1Cap1_, l1_ : 1, l2_ : 1] :=
         Exp[l1 * U1War2] / (Exp[l1 * U1War2] + Exp[l1 * U1Cap1])

        notp12classical[U1War2_, U1Cap1_, l1_ : 1, l2_ : 1] :=
         1 - p12classical[U1War2, U1Cap1, l1, l2]

        q9classical[U1War2_, U1Cap1_, U2War2_, U2Cap1_, U2Nego_, l1_ : 1, l2_ : 1] :=
         Module[{p12val, notp12val, UP2N12},
           p12val = p12classical[U1War2, U1Cap1, l1, l2];
           notp12val = notp12classical[U1War2, U1Cap1, l1, l2];
           UP2N12 = p12val * U2War2 + notp12val * U2Cap1;
           Exp[l2 * UP2N12] / (Exp[l2 * UP2N12] + Exp[l2 * U2Nego])]

        notq9classical[U1War2_, U1Cap1_, U2War2_, U2Cap1_, U2Nego_, l1_ : 1, l2_ : 1] :=
         1 - q9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2]

        p7classical[U1War1_, U1Cap2_, U2War1_, U2Cap2_, U1Nego_, l1_ : 1, l2_ : 1] :=
 Module[{q11val, notq11val, UP1N11},
  q11val = q11classical[U2War1, U2Cap2, l1, l2];
  notq11val = notq11classical[U2War1, U2Cap2, l1, l2];
  UP1N11 = q11val * U1War1 + notq11val * U1Cap2;
  Exp[l1 * UP1N11] / (Exp[l1 * UP1N11] + Exp[l1 * U1Nego])]

notp7classical[U1War1_, U1Cap2_, U2War1_, U2Cap2_, U1Nego_, l1_ : 1, l2_ : 1] :=
 1 - p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2]

q6classical[U1War1_, U1War2_, U1Cap1_, U1Cap2_, U2War2_,
  U2Cap1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, l1_ : 1, l2_ : 1] :=
 Module[{p8val, notp8val, UP2N8, q11val, notq11val, U2N11, UP2N7},
  p8val = p8classical[U1War2, U1Cap1, l1, l2];
  notp8val = notp8classical[U1War2, U1Cap1, l1, l2];
  UP2N8 = p8val * U2War2 + notp8val * U2Cap1;
  q11val = q11classical[U2War1, U2Cap2, l1, l2];
  notq11val = notq11classical[U2War1, U2Cap2, l1, l2];
  U2N11 = q11val * U2War1 + notq11val * U2Cap2;
  UP2N7 = p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2] * U2N11 +
    notp7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2] * U2Nego;
  Exp[l2 * UP2N8] / (Exp[l2 * UP2N8] + Exp[l2 * UP2N7])]

notq6classical[U1War1_, U1War2_, U1Cap1_, U1Cap2_, U2War2_,
  U2Cap1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, l1_ : 1, l2_ : 1] :=
 1 - q6classical[U1War1, U1War2, U1Cap1, U1Cap2, U2War2,
   U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2]

p5classical[U1Cap1_, U2Cap1_, U1Cap2_, U2Cap2_, U1War1_,
  U2War1_, U1War2_, U2War2_, U1Nego_, U2Nego_, l1_ : 1, l2_ : 1] :=
 Module[{p12val, notp12val, UP1N12, q10val, notq10val, UP1N10, q9val,
   notq9val, UP1N9}, p12val = p12classical[U1War2, U1Cap1, l1, l2];
  notp12val = notp12classical[U1War2, U1Cap1, l1, l2];
  UP1N12 = p12val * U1War2 + notp12val * U1Cap1;
  q10val = q10classical[U2War1, U2Cap2, l1, l2];
  notq10val = notq10classical[U2War1, U2Cap2, l1, l2];
  UP1N10 = q10val * U1War1 + notq10val * U1Cap2;
  q9val = q9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
  notq9val = notq9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
  UP1N9 = q9val * UP1N12 + notq9val * U1Nego;
  Exp[l1 * UP1N10] / (Exp[l1 * UP1N10] + Exp[l1 * UP1N9])]

notp5classical[U1Cap1_, U2Cap1_, U1Cap2_, U2Cap2_, U1War1_,
  U2War1_, U1War2_, U2War2_, U1Nego_, U2Nego_, l1_ : 1, l2_ : 1] :=
 1 - p5classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2, U1War1,
   U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2]

p4classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_, U2Cap1_, U1War1_,
   U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_, l1_ : 1, l2_ : 1] :=
  Module[{q11val, notq11val, UP1N11, p8val, notp8val, UP1N8,
    p7val, notp7val, UP1N7, q6val, notq6val, UP1N6},
   q11val = q11classical[U2War1, U2Cap2, l1, l2];
   notq11val = notq11classical[U2War1, U2Cap2, l1, l2];
   UP1N11 = q11val * U1War1 + notq11val * U1Cap2;
   p8val = p8classical[U1War2, U1Cap1, l1, l2];
   notp8val = notp8classical[U1War2, U1Cap1, l1, l2];
   UP1N8 = p8val * U1War2 + notp8val * U1Cap1;
   p7val = p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
   notp7val = notp7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
   UP1N7 = p7val * UP1N11 + notp7val * U1Nego;
   q6val = q6classical[U1War1, U1War2, U1Cap1, U1Cap2,
     U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
   notq6val = notq6classical[U1War1, U1War2, U1Cap1,
     U1Cap2, U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
   UP1N6 = q6val * UP1N8 + notq6val * UP1N7;
   Exp[l1 * UP1N6] / (Exp[l1 * UP1N6] + Exp[l1 * U1Acq1])]

notp4classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_, U2Cap1_, U1War1_,
   U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_, l1_ : 1, l2_ : 1] :=
  1 - p4classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1,
    U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2]

q3classical[U1Cap1_, U2Cap1_, U1Cap2_, U2Cap2_, U1War1_, U2War1_,
   U1War2_, U2War2_, U1Nego_, U2Nego_, U2Acq2_, l1_ : 1, l2_ : 1] :=
  Module[{p12val, notp12val, UP2N12, q10val, notq10val,
    UP2N10, q9val, notq9val, UP2N9, p5val, notp5val, UP2N5},
   p12val = p12classical[U1War2, U1Cap1, l1, l2];
   notp12val = notp12classical[U1War2, U1Cap1, l1, l2];
   UP2N12 = p12val * U2War2 + notp12val * U2Cap1;
   q10val = q10classical[U2War1, U2Cap2, l1, l2];
   notq10val = notq10classical[U2War1, U2Cap2, l1, l2];
   UP2N10 = q10val * U2War1 + notq10val * U2Cap2;
   q9val = q9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
   notq9val = notq9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
   UP2N9 = q9val * UP2N12 + notq9val * U2Nego;
   p5val = p5classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
     U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
   notp5val = notp5classical[U1Cap1, U2Cap1, U1Cap2,
     U2Cap2, U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
   UP2N5 = p5val * UP2N10 + notp5val * UP2N9;
   Exp[l2 * UP2N5] / (Exp[l2 * UP2N5] + Exp[l2 * U2Acq2])]

notq3classical[U1Cap1_, U2Cap1_, U1Cap2_, U2Cap2_, U1War1_, U2War1_,
   U1War2_, U2War2_, U1Nego_, U2Nego_, U2Acq2_, l1_ : 1, l2_ : 1] :=
  1 - q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2, U1War1,
   U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2]

q2classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_, U2Cap1_, U1War1_, U2War1_,
  U2Cap2_, U1Nego_, U2Nego_, U1Acq1_, U2Acq1_, U2SQ_, l1_ : 1, l2_ : 1] :=
 Module[{q11val, notq11val, UP2N11, p7val, notp7val, UP2N7, p8val,
   notp8val, UP2N8, q6val, notq6val, UP2N6, p4val, notp4val, UP2N4},
  q11val = q11classical[U2War1, U2Cap2, l1, l2];
  notq11val = notq11classical[U2War1, U2Cap2, l1, l2];
  UP2N11 = q11val * U2War1 + notq11val * U2Cap2;
  p7val = p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
  notp7val = notp7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
  UP2N7 = p7val * UP2N11 + notp7val * U2Nego;
  p8val = p8classical[U1War2, U1Cap1, l1, l2];
  notp8val = notp8classical[U1War2, U1Cap1, l1, l2];
  UP2N8 = p8val * U2War2 + notp8val * U2Cap1;
  q6val = q6classical[U1War1, U1War2, U1Cap1, U1Cap2,
    U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
  notq6val = notq6classical[U1War1, U1War2, U1Cap1,
    U1Cap2, U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
  UP2N6 = q6val * UP2N8 + notq6val * UP2N7;
  p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
    U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
  notp4val = notp4classical[U1War2, U1Cap1, U1Cap2, U2War2,
    U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
  UP2N4 = p4val * UP2N6 + notp4val * U2Acq1;
  Exp[l2 * UP2N4] / (Exp[l2 * UP2N4] + Exp[l2 * U2SQ])]

notq2classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_, U2Cap1_, U1War1_, U2War1_,
  U2Cap2_, U1Nego_, U2Nego_, U1Acq1_, U2Acq1_, U2SQ_, l1_ : 1, l2_ : 1] :=
 1 - q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
   U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2]

p1classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
  U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
  U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
 Module[{p12val, notp12val, UP1N12, q11val, notq11val, UP1N11, q9val,
   notq9val, UP1N9, p7val, notp7val, UP1N7, p8val, notp8val, UP1N8, q10val,
   notq10val, UP1N10, q6val, notq6val, UP1N6, p5val, notp5val, UP1N5,
   p4val, notp4val, UP1N4, q3val, notq3val, UP1N3, q2val, notq2val, UP1N2},
  p12val = p12classical[U1War2, U1Cap1, l1, l2];
  notp12val = notp12classical[U1War2, U1Cap1, l1, l2];
  UP1N12 = p12val * U1War1 + notp12val * U1Cap1;
  q11val = q11classical[U2War1, U2Cap2, l1, l2];
  notq11val = notq11classical[U2War1, U2Cap2, l1, l2];
  UP1N11 = q11val * U1War1 + notq11val * U1Cap2;
  q9val = q9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
  notq9val = notq9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
```

```
    UP1N9 = q9val * UP1N12 + notq9val * U1Nego;
    p7val = p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
    notp7val = notp7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
    UP1N7 = p7val * UP1N11 + notp7val * U1Nego;
    p8val = p8classical[U1War2, U1Cap1, l1, l2];
    notp8val = notp8classical[U1War2, U1Cap1, l1, l2];
    UP1N8 = p8val * U1War2 + notp8val * U1Cap1;
    q10val = q10classical[U2War1, U2Cap2, l1, l2];
    notq10val = notq10classical[U2War1, U2Cap2, l1, l2];
    UP1N10 = q10val * U1War1 + notq10val * U1Cap2;
    q6val = q6classical[U1War1, U1War2, U1Cap1, U1Cap2,
      U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
    notq6val = notq6classical[U1War1, U1War2, U1Cap1,
      U1Cap2, U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
    UP1N6 = q6val * UP1N8 + notq6val * UP1N7;
    p5val = p5classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
    notp5val = notp5classical[U1Cap1, U2Cap1, U1Cap2,
      U2Cap2, U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
    UP1N5 = p5val * UP1N10 + notp5val * UP1N9;
    p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
      U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
    notp4val = notp4classical[U1War2, U1Cap1, U1Cap2, U2War2,
      U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
    UP1N4 = p4val * UP1N6 + notp4val * U1Acq1;
    q3val = q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
    notq3val = notq3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
    UP1N3 = q3val * UP1N5 + notq3val * U1Acq2;
    q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
      U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
    notq2val = notq2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
      U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
    UP1N2 = q2val * UP1N4 + notq2val * U1SQ;
    Exp[l1 * UP1N3] / (Exp[l1 * UP1N3] + Exp[l1 * UP1N2])]

  notp1classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
    U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
    U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
   1 - p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
     U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2]
```

Outcome Functions

```
In[•]:= SQclassical[U1War2_, U1Cap1_, U1Cap2_, U2War2_, U2Cap1_, U1War1_, U2War1_,
          U2Cap2_, U1Nego_, U2Nego_, U1Acq1_, U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_,
  U2SQ_, l1_ : 1, l2_ : 1] := Module[{notp1val, notq2val}, notp1val =
   notp1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
    U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
  notq2val = notq2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
    U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
  notp1val * notq2val]

ACQ1classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
  U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
  U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
 Module[{notp1val, q2val, notp4val}, notp1val = notp1classical[U1War2,
    U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1, U2Cap2, U1Nego,
    U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
  q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
    U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
  notp4val = notp4classical[U1War2, U1Cap1, U1Cap2, U2War2,
    U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
  notp1val * q2val * notp4val]

ACQ2classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
  U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
  U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
 Module[{p1val, notq3val},
  p1val = p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
    U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
  notq3val = notq3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
    U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
  p1val * notq3val]

NEGOclassical[U1War2_, U1Cap1_, U1Cap2_, U2War2_, U2Cap1_,
  U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_, U2Acq1_,
  U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] := Module[
  {p1val, notp1val, q2val, q3val, p4val, notp5val, notq6val, notp7val, notq9val},
  p1val = p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
    U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
  notp1val =
   notp1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
    U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
  q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
    U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
  q3val = q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
    U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
  p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
    U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
  notp5val = notp5classical[U1Cap1, U2Cap1, U1Cap2,
    U2Cap2, U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
  notq6val = notq6classical[U1War1, U1War2, U1Cap1,
      U1Cap2, U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
    notp7val = notp7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
    notq9val = notq9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
    notp1val * q2val * p4val * notq6val * notp7val + p1val * q3val * notp5val * notq9val]

CAP1classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
    U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
    U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
  Module[{notp1val, q2val, p4val, q6val, notp8val, p1val,
     q3val, notp5val, q9val, notp12val}, notp1val =
     notp1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
      U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
    p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
      U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
    q6val = q6classical[U1War1, U1War2, U1Cap1, U1Cap2,
      U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
    notp8val = notp8classical[U1War2, U1Cap1, l1, l2];
    p1val = p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    q3val = q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
    notp5val = notp5classical[U1Cap1, U2Cap1, U1Cap2,
      U2Cap2, U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
    q9val = q9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
    notp12val = notp12classical[U1War2, U1Cap1, l1, l2];
    notp1val * q2val * p4val * q6val * notp8val +
     p1val * q3val * notp5val * q9val * notp12val]

CAP2classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
    U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
    U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
  Module[{notp1val, q2val, p4val, notq6val, p7val,
     notq11val, p1val, q3val, p5val, notq10val}, notp1val =
     notp1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
      U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
    p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
      U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
    notq6val = notq6classical[U1War1, U1War2, U1Cap1,
      U1Cap2, U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
    p7val = p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
    notq11val = notq11classical[U2War1, U2Cap2, l1, l2];
    p1val = p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    q3val = q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
    p5val = p5classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
    notq10val = notq10classical[U2War1, U2Cap2, l1, l2];
    notp1val * q2val * p4val * notq6val * p7val * notq11val +
     p1val * q3val * p5val * notq10val]

WAR1classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
    U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
    U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
  Module[{notp1val, p1val, q2val, p4val, notq6val,
     p7val, q11val, q3val, q10val, p5val}, notp1val =
     notp1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    p1val = p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
      U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
    p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
      U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
    notq6val = notq6classical[U1War1, U1War2, U1Cap1,
      U1Cap2, U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
    p7val = p7classical[U1War1, U1Cap2, U2War1, U2Cap2, U1Nego, l1, l2];
    q11val = q11classical[U2War1, U2Cap2, l1, l2];
    q3val = q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
    q10val = q10classical[U2War1, U2Cap2, l1, l2];
    p5val = p5classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
      U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
    notp1val * q2val * p4val * notq6val * p7val * q11val + p1val * q3val * p5val * q10val]

WAR2classical[U1War2_, U1Cap1_, U1Cap2_, U2War2_,
    U2Cap1_, U1War1_, U2War1_, U2Cap2_, U1Nego_, U2Nego_, U1Acq1_,
    U2Acq1_, U1Acq2_, U2Acq2_, U1SQ_, U2SQ_, l1_ : 1, l2_ : 1] :=
  Module[{notp1val, p1val, q2val, p4val, q6val, p8val,
     q3val, notp5val, q9val, p12val}, notp1val =
     notp1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    p1val = p1classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1, U2War1,
      U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U1Acq2, U2Acq2, U1SQ, U2SQ, l1, l2];
    q2val = q2classical[U1War2, U1Cap1, U1Cap2, U2War2, U2Cap1, U1War1,
      U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, U2Acq1, U2SQ, l1, l2];
    p4val = p4classical[U1War2, U1Cap1, U1Cap2, U2War2,
      U2Cap1, U1War1, U2War1, U2Cap2, U1Nego, U2Nego, U1Acq1, l1, l2];
    q6val = q6classical[U1War1, U1War2, U1Cap1, U1Cap2,
      U2War2, U2Cap1, U2War1, U2Cap2, U1Nego, U2Nego, l1, l2];
```

```
p8val = p8classical[U1War2, U1Cap1, l1, l2];
q3val = q3classical[U1Cap1, U2Cap1, U1Cap2, U2Cap2,
  U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, U2Acq2, l1, l2];
notp5val = notp5classical[U1Cap1, U2Cap1, U1Cap2,
  U2Cap2, U1War1, U2War1, U1War2, U2War2, U1Nego, U2Nego, l1, l2];
q9val = q9classical[U1War2, U1Cap1, U2War2, U2Cap1, U2Nego, l1, l2];
p12val = p12classical[U1War2, U1Cap1, l1, l2];
notp1val * q2val * p4val * q6val * p8val + p1val * q3val * notp5val * q9val * p12val]
```

Aux Functions

loadData

```
In[•]:= loadData[filename_String] :=
         Module[{rawData, headers, dataRows, groundtruth, utilityData,
           cleanedData, requiredColumns, missingColumns},

          Print["Loading CSV file: ", filename];
          rawData = Import[filename, "CSV"];

          If[Head[rawData] =!= List || Length[rawData] < 2,
           Print["Error: Could not load CSV file or file is empty"];
           Return[$Failed]
          ];

          headers = First[rawData];
          dataRows = Rest[rawData];

          Print["Loaded ", Length[dataRows],
           " rows with ", Length[headers], " columns"];
          cleanedData =
           Map[Function[row, Map[Function[cell, If[NumericQ[cell], cell, If[
                 StringQ[cell] && StringMatchQ[cell, NumberString], ToExpression[cell],
                 cell]]], row]], dataRows];
          groundtruth = cleanedData⟦All, -1⟧;

          utilityData =
           Association[Table[headers⟦i⟧ → cleanedData⟦All, i⟧, {i, Length[headers]}]];

          requiredColumns = {"wrTu1wr2", "wrTu1cp1", "wrTu1cp2", "wrTu1wr1", "wrTu1neg",
            "wrTu1ac1", "wrTu1ac2", "wrTu1sq", "wrTu2wr2", "wrTu2cp1", "wrTu2wr1",
            "wrTu2cp2", "wrTu2neg", "wrTu2ac1", "wrTu2ac2", "wrTu2sq"};
          missingColumns = Select[requiredColumns, ! KeyExistsQ[utilityData, #] &];

          If[Length[missingColumns] > 0,
           Print["Warning: Missing required columns: ", missingColumns];
          ];

          Association[
           "groundtruth" → groundtruth,
           "data" → utilityData,
           "nrows" → Length[cleanedData],
           "headers" → headers,
           "filename" → filename]
         ]
```

extractUtilities

```
In[•]:= extractUtilities[data_, rowIndex_Integer] := Module[{row, utils},
          If[rowIndex < 1 || rowIndex > data["nrows"],
 Print["Error: Row index ",
  rowIndex, " out of range [1, ", data["nrows"], "]"];
 Return[$Failed]
];

row = data["data"];
utils = Association[];

(*Player 1 utilities*)
utils["U1War2"] = If[KeyExistsQ[row, "wrTu1wr2"] &&
   NumericQ[row["wrTu1wr2"]⟦rowIndex⟧], row["wrTu1wr2"]⟦rowIndex⟧, 0.0];
utils["U1Cap1"] = If[KeyExistsQ[row, "wrTu1cp1"] &&
   NumericQ[row["wrTu1cp1"]⟦rowIndex⟧], row["wrTu1cp1"]⟦rowIndex⟧, 0.0];
utils["U1Cap2"] = If[KeyExistsQ[row, "wrTu1cp2"] &&
   NumericQ[row["wrTu1cp2"]⟦rowIndex⟧], row["wrTu1cp2"]⟦rowIndex⟧, 0.0];
utils["U1War1"] = If[KeyExistsQ[row, "wrTu1wr1"] &&
   NumericQ[row["wrTu1wr1"]⟦rowIndex⟧], row["wrTu1wr1"]⟦rowIndex⟧, 0.0];
utils["U1Nego"] = If[KeyExistsQ[row, "wrTu1neg"] &&
   NumericQ[row["wrTu1neg"]⟦rowIndex⟧], row["wrTu1neg"]⟦rowIndex⟧, 0.0];
utils["U1Acq1"] = If[KeyExistsQ[row, "wrTu1ac1"] &&
   NumericQ[row["wrTu1ac1"]⟦rowIndex⟧], row["wrTu1ac1"]⟦rowIndex⟧, 0.0];
utils["U1Acq2"] = If[KeyExistsQ[row, "wrTu1ac2"] &&
   NumericQ[row["wrTu1ac2"]⟦rowIndex⟧], row["wrTu1ac2"]⟦rowIndex⟧, 0.0];
utils["U1SQ"] = If[KeyExistsQ[row, "wrTu1sq"] &&
   NumericQ[row["wrTu1sq"]⟦rowIndex⟧], row["wrTu1sq"]⟦rowIndex⟧, 0.0];

(*Player 2 utilities*)
utils["U2War2"] = If[KeyExistsQ[row, "wrTu2wr2"] &&
   NumericQ[row["wrTu2wr2"]⟦rowIndex⟧], row["wrTu2wr2"]⟦rowIndex⟧, 0.0];
utils["U2Cap1"] = If[KeyExistsQ[row, "wrTu2cp1"] &&
   NumericQ[row["wrTu2cp1"]⟦rowIndex⟧], row["wrTu2cp1"]⟦rowIndex⟧, 0.0];
utils["U2War1"] = If[KeyExistsQ[row, "wrTu2wr1"] &&
   NumericQ[row["wrTu2wr1"]⟦rowIndex⟧], row["wrTu2wr1"]⟦rowIndex⟧, 0.0];
utils["U2Cap2"] = If[KeyExistsQ[row, "wrTu2cp2"] &&
   NumericQ[row["wrTu2cp2"]⟦rowIndex⟧], row["wrTu2cp2"]⟦rowIndex⟧, 0.0];
utils["U2Nego"] = If[KeyExistsQ[row, "wrTu2neg"] &&
   NumericQ[row["wrTu2neg"]⟦rowIndex⟧], row["wrTu2neg"]⟦rowIndex⟧, 0.0];
utils["U2Acq1"] = If[KeyExistsQ[row, "wrTu2ac1"] &&
   NumericQ[row["wrTu2ac1"]⟦rowIndex⟧], row["wrTu2ac1"]⟦rowIndex⟧, 0.0];
utils["U2Acq2"] = If[KeyExistsQ[row, "wrTu2ac2"] &&
   NumericQ[row["wrTu2ac2"]⟦rowIndex⟧], row["wrTu2ac2"]⟦rowIndex⟧, 0.0];
utils["U2SQ"] = If[KeyExistsQ[row, "wrTu2sq"] &&
   NumericQ[row["wrTu2sq"]⟦rowIndex⟧], row["wrTu2sq"]⟦rowIndex⟧, 0.0];
utils["Agent1"] =
 If[KeyExistsQ[row, "ISOShNm1"], row["ISOShNm1"]⟦rowIndex⟧, "Unknown"];
utils["Agent2"] =
 If[KeyExistsQ[row, "ISOShNm2"], row["ISOShNm2"]⟦rowIndex⟧, "Unknown"];
 (*Additional information*)
 utils["groundtruth"] = data["groundtruth"]⟦rowIndex⟧;
 utils["ccode1"] = If[KeyExistsQ[row, "ccode1"] &&
    NumericQ[row["ccode1"]⟦rowIndex⟧], row["ccode1"]⟦rowIndex⟧, 0];
 utils["ccode2"] = If[KeyExistsQ[row, "ccode2"] &&
    NumericQ[row["ccode2"]⟦rowIndex⟧], row["ccode2"]⟦rowIndex⟧, 0];
 utils["year"] = If[KeyExistsQ[row, "year"] && NumericQ[row["year"]⟦rowIndex⟧],
   row["year"]⟦rowIndex⟧, 0];
 utils]
```

calculateOutcomes

```
In[∘]:= calculateOutcome[data_, rowIndex_Integer, l1_ : 1, l2_ : 1] :=
         Module[{utils, outcomes, sqProb, acq1Prob,
           acq2Prob, negoProb, cap1Prob, cap2Prob, war1Prob, war2Prob,
           prediction, roundedProbs, outcomeProbs, maxVal, maxOutcomes},

          utils = extractUtilities[data, rowIndex];
          If[utils === $Failed,
           Return[$Failed]
          ];

          sqProb =
           Quiet[
            SQclassical[utils["U1War2"],
             utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
             utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
             utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
             utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];

          acq1Prob =
           Quiet[
            ACQ1classical[utils["U1War2"],
             utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
             utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
             utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
             utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];

          acq2Prob =
           Quiet[
            ACQ2classical[utils["U1War2"],
             utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
             utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
             utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
             utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];
```

```
negoProb =
 Quiet[
  NEGOclassical[utils["U1War2"],
   utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
   utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
   utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
   utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];

cap1Prob =
 Quiet[
  CAP1classical[utils["U1War2"],
   utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
   utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
   utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
   utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];

cap2Prob =
 Quiet[
  CAP2classical[utils["U1War2"],
   utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
   utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
   utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
   utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];

war1Prob =
 Quiet[
  WAR1classical[utils["U1War2"],
   utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
   utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
   utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
   utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];

war2Prob =
 Quiet[
  WAR2classical[utils["U1War2"],
   utils["U1Cap1"], utils["U1Cap2"], utils["U2War2"],
   utils["U2Cap1"], utils["U1War1"], utils["U2War1"], utils["U2Cap2"],
   utils["U1Nego"], utils["U2Nego"], utils["U1Acq1"], utils["U2Acq1"],
   utils["U1Acq2"], utils["U2Acq2"], utils["U1SQ"], utils["U2SQ"], l1, l2]];


(*Create outcomes association*)
outcomes =
 Association[
  "SQ" → sqProb,
  "ACQ1" → acq1Prob,
    "ACQ2" → acq2Prob,
    "NEGO" → negoProb,
    "CAP1" → cap1Prob,
    "CAP2" → cap2Prob,
    "WAR1" → war1Prob,
    "WAR2" → war2Prob];

  (*Calculate prediction *)
  outcomeProbs =
   Association[
    "SQ" → sqProb,
    "ACQ1" → acq1Prob,
    "ACQ2" → acq2Prob,
    "NEGO" → negoProb,
    "CAP1" → cap1Prob,
    "CAP2" → cap2Prob,
    "WAR1" → war1Prob,
    "WAR2" → war2Prob];

  (*Round probabilities to 4 decimal places for comparison*)
  roundedProbs =
   Association[# → Round[outcomeProbs[#], 0.0001] & /@ Keys[outcomeProbs]];

  maxVal = Max[Values[roundedProbs]];
  (*Get all outcomes that have the maximum probability (after rounding)*)
  maxOutcomes = Keys[Select[roundedProbs, # == maxVal &]];
  (*Randomly select one if there are ties*)
  prediction = RandomChoice[maxOutcomes];

  (*Add all components to outcomes*)
  outcomes["groundtruth"] = utils["groundtruth"];
  outcomes["prediction"] = prediction;
  outcomes["total"] = Total[{sqProb, acq1Prob,
     acq2Prob, negoProb, cap1Prob, cap2Prob, war1Prob, war2Prob}];
  outcomes["utilities"] = utils;
  outcomes]
```

processDataset

```
In[•]:= processDataset[data_, l1_ : 1, l2_ : 1] := Module[{results, i},
          Print["Processing ", data["nrows"], " rows..."];
          results = {};
          Do[Module[{result},
            If[Mod[i, 50] == 0,
             Print["Processing row ", i, "/", data["nrows"]]
            ];
            result = calculateOutcome[data, i, l1, l2];
            If[result =!= $Failed,
             AppendTo[results, result],
             Print["Warning: Failed to process row ", i]
            ]
           ], {i, 1, data["nrows"]}
          ];
          Print["Successfully processed ",
           Length[results], " out of ", data["nrows"], " rows"];
          results
         ]
```

extractPredictionsAndGroundtruth

```
In[∘]:= extractPredictionsAndGroundtruth[results_] :=
         Module[{predictions, groundTruth, outcomes},
          outcomes = {"ACQ1", "ACQ2", "CAP1", "CAP2", "NEGO", "SQ", "WAR1"};
          predictions =
           Table[Module[{outcomeProbs, maxVal, maxOutcomes, maxOutcome, roundedProbs},
             outcomeProbs = Association["ACQ1" → If[KeyExistsQ[results⟦i⟧, "ACQ1"] &&
                  NumericQ[results⟦i⟧["ACQ1"]], results⟦i⟧["ACQ1"], 0], "ACQ2" →
                If[KeyExistsQ[results⟦i⟧, "ACQ2"] && NumericQ[results⟦i⟧["ACQ2"]],
                 results⟦i⟧["ACQ2"], 0], "CAP1" → If[KeyExistsQ[results⟦i⟧, "CAP1"] &&
                  NumericQ[results⟦i⟧["CAP1"]], results⟦i⟧["CAP1"], 0], "CAP2" →
                If[KeyExistsQ[results⟦i⟧, "CAP2"] && NumericQ[results⟦i⟧["CAP2"]],
                 results⟦i⟧["CAP2"], 0], "NEGO" → If[KeyExistsQ[results⟦i⟧, "NEGO"] &&
                  NumericQ[results⟦i⟧["NEGO"]], results⟦i⟧["NEGO"], 0],
               "SQ" → If[KeyExistsQ[results⟦i⟧, "SQ"] && NumericQ[results⟦i⟧["SQ"]],
                 results⟦i⟧["SQ"], 0], "WAR1" → If[KeyExistsQ[results⟦i⟧, "WAR1"] &&
                  NumericQ[results⟦i⟧["WAR1"]], results⟦i⟧["WAR1"], 0]];
             (*Find outcome with maximum probability*)
             (*Round probabilities to 4 decimal places for comparison*)roundedProbs =
              Association[# → Round[outcomeProbs[#], 0.0001] & /@ Keys[outcomeProbs]];
             maxVal = Max[Values[roundedProbs]];
             (*Get all outcomes that have the maximum probability (after rounding)*)
             maxOutcomes = Keys[Select[roundedProbs, # == maxVal &]];
             (*Randomly select one if there are ties*)
             maxOutcome = RandomChoice[maxOutcomes];
             maxOutcome], {i, Length[results]}];
          groundTruth = Table[If[KeyExistsQ[results⟦i⟧, "groundtruth"],
             results⟦i⟧["groundtruth"], "UNKNOWN"], {i, Length[results]}];
          Association["predictions" → predictions, "groundtruth" → groundTruth]]
```

calculateAccuracy

```
In[∘]:= calculateAccuracy[results_] := Module[{predTruth, correct},
          predTruth = extractPredictionsAndGroundtruth[results];
          correct =
           MapThread[Equal, {predTruth["predictions"], predTruth["groundtruth"]}];
          N[Count[correct, True] / Length[correct]]]
```

plotConfusionMatrix

```
In[∘]:= plotConfusionMatrix[results_, l1_ : 1, l2_ : 1, title_ : "Confusion Matrix"] :=
         Module[{predictions, groundTruths, outcomes, confusionData, accuracy},

          Module[{predTruth}, predTruth = extractPredictionsAndGroundtruth[results];
           predictions = predTruth["predictions"];
           groundTruths = predTruth["groundtruth"];];

          accuracy = N[Count[MapThread[Equal, {predictions, groundTruths}], True] /
              Length[results]];

          outcomes = {"ACQ1", "ACQ2", "CAP1", "CAP2", "NEGO", "SQ", "WAR1"};
          confusionData = Table[Count[MapThread[List, {groundTruths, predictions}],
              {actualOutcome, predictedOutcome}],
            {actualOutcome, outcomes}, {predictedOutcome, outcomes}];

          Print[Style[title, 16, Bold]];
          Print[Style["λ1 = " <> ToString[l1] <> " λ2 = " <>
              ToString[l2] <> " | Accuracy = " <> ToString[N[accuracy]], 14]];
          Print[""];
          Grid[
           Prepend[MapThread[Prepend, {confusionData, outcomes}],
                  Prepend[outcomes, Style["Actual \\ Predicted", Bold]]],
           Frame → All,
           Alignment → Center,
           Background → {None, {LightBlue, None}},
           ItemStyle → {Automatic, {Bold, Automatic}},
           Spacings → {2, 1},
           FrameStyle → Thick,
           Dividers → {{2 → Thick}, {2 → Thick}}]]
```

getFirstNEntries

```
In[∘]:= getFirstNEntries[resultAllData_, N_Integer] := Module[{extractedEntries},
          (*Input validation*)
          If[! ListQ[resultAllData],
           Print["Error: resultAllData must be a list"];
           Return[$Failed]
          ];

          If[N ≤ 0,
           Print["Error: N must be a positive integer"];
           Return[$Failed]
          ];

          If[Length[resultAllData] == 0,
 Print["Warning: resultAllData is empty"];
 Return[{}]
];

(*Extract only the specified components from each entry*)
extractedEntries =
 Table[Module[{entry, utils},
   entry = resultAllData〚i〛;
   utils =
    If[KeyExistsQ[entry, "utilities"], entry["utilities"], Association[]];

   Association[
    "SQ" → If[KeyExistsQ[entry, "SQ"], Round[entry["SQ"], 0.0001], 0],
    "ACQ1" → If[KeyExistsQ[entry, "ACQ1"], Round[entry["ACQ1"], 0.0001], 0],
    "ACQ2" → If[KeyExistsQ[entry, "ACQ2"], Round[entry["ACQ2"], 0.0001], 0],
    "NEGO" → If[KeyExistsQ[entry, "NEGO"], Round[entry["NEGO"], 0.0001], 0],
    "CAP1" → If[KeyExistsQ[entry, "CAP1"], Round[entry["CAP1"], 0.0001], 0],
    "CAP2" → If[KeyExistsQ[entry, "CAP2"], Round[entry["CAP2"], 0.0001], 0],
    "WAR1" → If[KeyExistsQ[entry, "WAR1"], Round[entry["WAR1"], 0.0001], 0],
    "WAR2" → If[KeyExistsQ[entry, "WAR2"], Round[entry["WAR2"], 0.0001], 0],
    "prediction" →
     If[KeyExistsQ[entry, "prediction"], entry["prediction"], "UNKNOWN"],
    "groundtruth" →
     If[KeyExistsQ[entry, "groundtruth"], entry["groundtruth"], "UNKNOWN"],
    "Agent1" → If[KeyExistsQ[utils, "Agent1"], utils["Agent1"], "UNKNOWN"],
    "Agent2" → If[KeyExistsQ[utils, "Agent2"], utils["Agent2"], "UNKNOWN"]]
  ], {i, Min[N, Length[resultAllData]]}
 ];

(*Return extracted entries with informative message*)
If[N ≥ Length[resultAllData],
 Print["Note: Requested ", N, " entries but only ", Length[resultAllData],
  " available. Returning all entries with extracted components."];
 Return[extractedEntries],
 Print["Returning first ", N, " entries out of ",
  Length[resultAllData], " total entries with extracted components."];
 Return[extractedEntries]]
```

Experiments

Setting 0: Checking model correctness

```
In[ ]:= rowIndex = 2;
```

```
In[•]:= (* datasetPath =
          "/Users/162191/Documents/Github/quantum_international_interaction_game/BN/
            dataset/balanced_data.csv"; *)

In[•]:= datasetPath =
         "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
           dataset/balanced_data.csv"
        (* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
           dataset\\balanced_data.csv"; *)

Out[•]=
        /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
          balanced_data.csv

In[•]:= data = loadData[datasetPath];

        Loading CSV file:
         /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
           balanced_data.csv

        Loaded 579 rows with 149 columns

In[•]:= result = calculateOutcome[data, rowIndex];

In[•]:= (* checking the outcome for a single dyad *)

In[•]:= result = calculateOutcome[data, rowIndex];
        displayOutcomes[result, rowIndex];
```

Balanced Dataset

Setting 1: Lamda1 = 1 | Lambda2 = 1

```
In[•]:= datasetPath =
         "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
           dataset/balanced_data.csv"
        (* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
           dataset\\balanced_data.csv"; *)
        data = loadData[datasetPath];

Out[•]=
        /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
          balanced_data.csv

        Loading CSV file:
         /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
           balanced_data.csv

        Loaded 579 rows with 149 columns

In[•]:= l1 = 1;
        l2 = 1;

In[•]:= resultAllData = processDataset[data, l1, l2];
```

```
Processing 579 rows...

Processing row 50/579

Processing row 100/579

Processing row 150/579

Processing row 200/579

Processing row 250/579

Processing row 300/579

Processing row 350/579

Processing row 400/579

Processing row 450/579

Processing row 500/579

Processing row 550/579

Successfully processed 579 out of 579 rows
```

```
In[∘]:= samplePredictions = getFirstNEntries[resultAllData, 3]

        Returning first 3 entries out of 579 total entries with extracted components.

Out[∘]=
        {<|SQ → 0.121, ACQ1 → 0.1124, ACQ2 → 0.1849, NEGO → 0.1001, CAP1 → 0.0571,
           CAP2 → 0.0687, WAR1 → 0.2536, WAR2 → 0.1021, prediction → WAR1,
           groundtruth → SQ, Agent1 → ESTONIA, Agent2 → UNITED KINGDOM|>,
          <|SQ → 0.213, ACQ1 → 0.0125, ACQ2 → 0.4264, NEGO → 0.0915, CAP1 → 0.0135,
           CAP2 → 0.0727, WAR1 → 0.0789, WAR2 → 0.0915, prediction → ACQ2,
           groundtruth → SQ, Agent1 → FRANCE, Agent2 → CHILE|>,
          <|SQ → 0.1259, ACQ1 → 0.2326, ACQ2 → 0.0548, NEGO → 0.1131, CAP1 → 0.0866,
           CAP2 → 0.0266, WAR1 → 0.258, WAR2 → 0.1024, prediction → WAR1,
           groundtruth → SQ, Agent1 → ARGENTINA, Agent2 → FRANCE|>}

In[∘]:= accuracy = calculateAccuracy[resultAllData]
Out[∘]=
        0.17962

In[∘]:= plotConfusionMatrix[resultAllData, l1, l2, "Signorino Confusion Matrix"]
```

```
Signorino Confusion Matrix
λ1 = 1 λ2 = 1 | Accuracy = 0.17962
```

Out[∘]=

Actual \ Predicted	ACQ1	ACQ2	CAP1	CAP2	NEGO	SQ	WAR1
ACQ1	0	5	0	0	0	0	1
ACQ2	0	78	0	0	0	0	21
CAP1	0	42	0	0	0	1	13
CAP2	3	70	0	0	0	0	26
NEGO	0	71	0	0	0	2	26
SQ	7	64	0	0	0	1	27
WAR1	0	74	0	0	0	0	25

Setting 2: Lamda1 = 0.5 | Lambda2 = 0.5

```
In[∘]:= datasetPath =
         "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
           dataset/balanced_data.csv"
        (* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
           dataset\\balanced_data.csv"; *)
        data = loadData[datasetPath];
```

```
Out[∘]=
/Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
  balanced_data.csv

Loading CSV file:
 /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
   balanced_data.csv
Loaded 579 rows with 149 columns
```

```
In[∘]:= l1 = 0.5;
        l2 = 0.5;
```

```
In[∘]:= resultAllData = processDataset[data, l1, l2];
```

```
Processing 579 rows...

Processing row 50/579

Processing row 100/579

Processing row 150/579

Processing row 200/579

Processing row 250/579

Processing row 300/579

Processing row 350/579

Processing row 400/579

Processing row 450/579

Processing row 500/579

Processing row 550/579

Successfully processed 579 out of 579 rows
```

```
In[ ]:= samplePredictions = getFirstNEntries[resultAllData, 3]

Returning first 3 entries out of 579 total entries with extracted components.

Out[ ]=
{<|SQ → 0.1694, ACQ1 → 0.1119, ACQ2 → 0.2361, NEGO → 0.0895, CAP1 → 0.0615,
   CAP2 → 0.0854, WAR1 → 0.164, WAR2 → 0.0822, prediction → ACQ2,
   groundtruth → SQ, Agent1 → ESTONIA, Agent2 → UNITED KINGDOM|>,
  <|SQ → 0.2107, ACQ1 → 0.0541, ACQ2 → 0.3356, NEGO → 0.0896, CAP1 → 0.039,
   CAP2 → 0.0831, WAR1 → 0.0865, WAR2 → 0.1015, prediction → ACQ2,
   groundtruth → SQ, Agent1 → FRANCE, Agent2 → CHILE|>,
  <|SQ → 0.1681, ACQ1 → 0.157, ACQ2 → 0.153, NEGO → 0.1038, CAP1 → 0.077,
   CAP2 → 0.0625, WAR1 → 0.1949, WAR2 → 0.0838, prediction → WAR1,
   groundtruth → SQ, Agent1 → ARGENTINA, Agent2 → FRANCE|>}
```

```
In[ ]:= accuracy = calculateAccuracy[resultAllData]

Out[ ]=
0.169257
```

```
In[ ]:= plotConfusionMatrix[resultAllData, l1, l2, "Signorino Confusion Matrix"]
```

Signorino Confusion Matrix

```
λ1 = 0.5 λ2 = 0.5 | Accuracy = 0.169257
```

Out[•]=

Actual \ Predicted	ACQ1	ACQ2	CAP1	CAP2	NEGO	SQ	WAR1
ACQ1	0	5	0	0	0	0	1
ACQ2	0	90	0	0	0	0	9
CAP1	0	52	0	0	0	0	4
CAP2	0	88	0	0	0	0	11
NEGO	0	91	0	0	0	3	5
SQ	0	81	0	0	0	1	17
WAR1	0	92	0	0	0	0	7

Setting 3: Lamda1 = 2 | Lambda2 = 2

```
In[•]:= datasetPath =
         "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
           dataset/balanced_data.csv"
        (* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
           dataset\\balanced_data.csv"; *)
        data = loadData[datasetPath];
```

```
Out[•]=
        /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
          balanced_data.csv

        Loading CSV file:
         /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
           balanced_data.csv

        Loaded 579 rows with 149 columns
```

```
In[•]:= l1 = 2;
        l2 = 2;
```

```
In[•]:= resultAllData = processDataset[data, l1, l2];
```

```
Processing 579 rows...

Processing row 50/579

Processing row 100/579

Processing row 150/579

Processing row 200/579

Processing row 250/579

Processing row 300/579

Processing row 350/579

Processing row 400/579

Processing row 450/579

Processing row 500/579

Processing row 550/579

Successfully processed 579 out of 579 rows
```

```
In[ ]:= samplePredictions = getFirstNEntries[resultAllData, 3]

Returning first 3 entries out of 579 total entries with extracted components.

Out[ ]=
{<|SQ → 0.0904, ACQ1 → 0.1386, ACQ2 → 0.066, NEGO → 0.1642, CAP1 → 0.0472,
   CAP2 → 0.0234, WAR1 → 0.3193, WAR2 → 0.1508, prediction → WAR1,
   groundtruth → SQ, Agent1 → ESTONIA, Agent2 → UNITED KINGDOM|>,
  <|SQ → 0.1228, ACQ1 → 0.0002, ACQ2 → 0.6035, NEGO → 0.1087, CAP1 → 0.001,
   CAP2 → 0.0534, WAR1 → 0.0628, WAR2 → 0.0476, prediction → ACQ2,
   groundtruth → SQ, Agent1 → FRANCE, Agent2 → CHILE|>,
  <|SQ → 0.0606, ACQ1 → 0.4044, ACQ2 → 0.0047, NEGO → 0.0949, CAP1 → 0.0786,
   CAP2 → 0.0026, WAR1 → 0.2444, WAR2 → 0.1098, prediction → ACQ1,
   groundtruth → SQ, Agent1 → ARGENTINA, Agent2 → FRANCE|>}
```

```
In[ ]:= accuracy = calculateAccuracy[resultAllData]

Out[ ]=
0.183074
```

```
In[ ]:= plotConfusionMatrix[resultAllData, l1, l2, "Signorino Confusion Matrix"]
```

Signorino Confusion Matrix

```
λ1 = 2 λ2 = 2 | Accuracy = 0.183074
```

Out[ ]=

Actual \ Predicted	ACQ1	ACQ2	CAP1	CAP2	NEGO	SQ	WAR1
ACQ1	0	5	0	0	0	0	1
ACQ2	7	70	0	0	0	1	21
CAP1	1	35	0	0	1	2	17
CAP2	10	59	0	0	0	1	29
NEGO	6	56	0	0	2	2	33
SQ	15	48	0	0	2	1	33
WAR1	3	59	0	0	2	2	33

Setting 4: Lamda1 = 0.1 | Lambda2 = 0.1

In[ ]:=

```
datasetPath =
 "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
   dataset/balanced_data.csv"
(* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
   dataset\\balanced_data.csv"; *)
data = loadData[datasetPath];
```

Out[ ]=

```
/Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
  balanced_data.csv

Loading CSV file:
 /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
   balanced_data.csv

Loaded 579 rows with 149 columns
```

In[ ]:=

```
l1 = 0.1;
l2 = 0.1;
```

In[ ]:=

```
resultAllData = processDataset[data, l1, l2];
```

```
Processing 579 rows...

Processing row 50/579

Processing row 100/579

Processing row 150/579

Processing row 200/579

Processing row 250/579

Processing row 300/579

Processing row 350/579

Processing row 400/579

Processing row 450/579

Processing row 500/579

Processing row 550/579

Successfully processed 579 out of 579 rows
```

```
In[•]:= samplePredictions = getFirstNEntries[resultAllData, 3]

        Returning first 3 entries out of 579 total entries with extracted components.

Out[•]=
        {<|SQ → 0.2316, ACQ1 → 0.1214, ACQ2 → 0.2497, NEGO → 0.0921, CAP1 → 0.0631,
           CAP2 → 0.082, WAR1 → 0.0934, WAR2 → 0.0668, prediction → ACQ2,
           groundtruth → SQ, Agent1 → ESTONIA, Agent2 → UNITED KINGDOM|>,
          <|SQ → 0.2363, ACQ1 → 0.1101, ACQ2 → 0.2662, NEGO → 0.0922,
           CAP1 → 0.0602, CAP2 → 0.0807, WAR1 → 0.0813, WAR2 → 0.073,
           prediction → ACQ2, groundtruth → SQ, Agent1 → FRANCE, Agent2 → CHILE|>,
          <|SQ → 0.2294, ACQ1 → 0.1282, ACQ2 → 0.2371, NEGO → 0.0945, CAP1 → 0.0655,
           CAP2 → 0.0793, WAR1 → 0.0995, WAR2 → 0.0666, prediction → ACQ2,
           groundtruth → SQ, Agent1 → ARGENTINA, Agent2 → FRANCE|>}

In[•]:= accuracy = calculateAccuracy[resultAllData]

Out[•]=
        0.172712

In[•]:= plotConfusionMatrix[resultAllData, l1, l2, "Signorino Confusion Matrix"]
```

Signorino Confusion Matrix

λ1 = 0.1 λ2 = 0.1 | Accuracy = 0.172712

Out[ ]=

Actual \ Predicted	ACQ1	ACQ2	CAP1	CAP2	NEGO	SQ	WAR1
ACQ1	0	6	0	0	0	0	0
ACQ2	0	99	0	0	0	0	0
CAP1	0	56	0	0	0	0	0
CAP2	0	99	0	0	0	0	0
NEGO	0	97	0	0	0	2	0
SQ	0	98	0	0	0	1	0
WAR1	0	99	0	0	0	0	0

Setting 5: Lamda1 = 10 | Lambda2 = 10

In[ ]:=

```
datasetPath =
 "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
   dataset/balanced_data.csv"
(* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
   dataset\\balanced_data.csv"; *)
data = loadData[datasetPath];
```

Out[ ]=

```
/Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
  balanced_data.csv

Loading CSV file:
 /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
   balanced_data.csv
Loaded 579 rows with 149 columns
```

In[ ]:=

```
l1 = 10;
l2 = 10;
```

In[ ]:=

```
resultAllData = processDataset[data, l1, l2];
```

```
Processing 579 rows...

Processing row 50/579

Processing row 100/579

Processing row 150/579

Processing row 200/579

Processing row 250/579

Processing row 300/579

Processing row 350/579

Processing row 400/579

Processing row 450/579

Processing row 500/579

Processing row 550/579

Successfully processed 579 out of 579 rows
```

```
In[•]:= samplePredictions = getFirstNEntries[resultAllData, 3]

        Returning first 3 entries out of 579 total entries with extracted components.

Out[•]=
        {<|SQ → 0.2047, ACQ1 → 0.0012, ACQ2 → 0., NEGO → 0.7666,
           CAP1 → 0.0001, CAP2 → 0., WAR1 → 0.009, WAR2 → 0.0185, prediction → NEGO,
           groundtruth → SQ, Agent1 → ESTONIA, Agent2 → UNITED KINGDOM|>,
         <|SQ → 0., ACQ1 → 0., ACQ2 → 0.4461, NEGO → 0.5114, CAP1 → 0.,
           CAP2 → 0.0119, WAR1 → 0.0268, WAR2 → 0.0038, prediction → NEGO,
           groundtruth → SQ, Agent1 → FRANCE, Agent2 → CHILE|>,
         <|SQ → 0., ACQ1 → 0.9504, ACQ2 → 0., NEGO → 0.0001, CAP1 → 0.0002,
           CAP2 → 0., WAR1 → 0.0481, WAR2 → 0.0011, prediction → ACQ1,
           groundtruth → SQ, Agent1 → ARGENTINA, Agent2 → FRANCE|>}
```

```
In[•]:= accuracy = calculateAccuracy[resultAllData]

Out[•]=
        0.169257
```

```
In[•]:= plotConfusionMatrix[resultAllData, l1, l2, "Signorino Confusion Matrix"]
```

Signorino Confusion Matrix

λ1 = 10 λ2 = 10 | Accuracy = 0.169257

Out[ ]=

Actual \ Predicted	ACQ1	ACQ2	CAP1	CAP2	NEGO	SQ	WAR1
ACQ1	0	4	0	0	2	0	0
ACQ2	6	29	0	0	43	16	5
CAP1	5	15	0	0	23	4	9
CAP2	7	24	0	0	46	12	10
NEGO	9	24	0	0	36	18	12
SQ	15	16	0	0	42	16	10
WAR1	3	24	0	0	43	12	17

Setting 6: Lambda1 = 0.0001 | Lambda2 = 0.0001

In[ ]:=

```
datasetPath =
  "/Users/162191/Documents/GitHub/quantum_international_interaction_game/
    dataset/balanced_data.csv"
(* "D:\\home\\Documents\\Github\\quantum_international_interaction_game\\BN\\
    dataset\\balanced_data.csv"; *)
data = loadData[datasetPath];
```

Out[ ]=

```
/Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
   balanced_data.csv

Loading CSV file:
 /Users/162191/Documents/GitHub/quantum_international_interaction_game/dataset/
   balanced_data.csv
Loaded 579 rows with 149 columns
```

In[ ]:=

```
l1 = 0.0001;
l2 = 0.0001;
```

In[ ]:=

```
resultAllData = processDataset[data, l1, l2];
```

```
Processing 579 rows...

Processing row 50/579

Processing row 100/579

Processing row 150/579

Processing row 200/579

Processing row 250/579

Processing row 300/579

Processing row 350/579

Processing row 400/579

Processing row 450/579

Processing row 500/579

Processing row 550/579

Successfully processed 579 out of 579 rows
```

```
In[•]:= samplePredictions = getFirstNEntries[resultAllData, 3]

        Returning first 3 entries out of 579 total entries with extracted components.

Out[•]=
        {<|SQ → 0.25, ACQ1 → 0.125, ACQ2 → 0.25, NEGO → 0.0937, CAP1 → 0.0625,
           CAP2 → 0.0781, WAR1 → 0.0781, WAR2 → 0.0625, prediction → SQ,
           groundtruth → SQ, Agent1 → ESTONIA, Agent2 → UNITED KINGDOM|>,
          <|SQ → 0.25, ACQ1 → 0.125, ACQ2 → 0.25, NEGO → 0.0937, CAP1 → 0.0625,
           CAP2 → 0.0781, WAR1 → 0.0781, WAR2 → 0.0625, prediction → SQ,
           groundtruth → SQ, Agent1 → FRANCE, Agent2 → CHILE|>,
          <|SQ → 0.25, ACQ1 → 0.125, ACQ2 → 0.25, NEGO → 0.0938, CAP1 → 0.0625,
           CAP2 → 0.0781, WAR1 → 0.0781, WAR2 → 0.0625, prediction → SQ,
           groundtruth → SQ, Agent1 → ARGENTINA, Agent2 → FRANCE|>}
```

```
In[•]:= accuracy = calculateAccuracy[resultAllData]
Out[•]=
        0.169257
```

```
In[•]:= plotConfusionMatrix[resultAllData, l1, l2, "Signorino Confusion Matrix"]
```

Signorino Confusion Matrix

λ1 = 0.0001 λ2 = 0.0001 | Accuracy = 0.188256

Out[•]=

Actual \ Predicted	ACQ1	ACQ2	CAP1	CAP2	NEGO	SQ	WAR1
ACQ1	0	2	0	0	0	4	0
ACQ2	0	52	0	0	0	47	0
CAP1	0	27	0	0	0	29	0
CAP2	0	40	0	0	0	59	0
NEGO	0	49	0	0	0	50	0
SQ	0	42	0	0	0	57	0
WAR1	0	47	0	0	0	52	0