Maskininlärning och tärningsspelet *Etthundra*

Undersökning om en maskinintelligens kan bli konsekvent bättre än människor på tärninggspelet *Etthundra*

Elever: REDACTED
Program: REDACTED
Klass: REDACTED
Handledare: REDACTED
Datum: 2020-03-08

Sammanfattning

Syftet med denna undersökning är att undersöka om en maskinintelligens som tränats med Q-learning metoden kan bli konsekvent bättre än människor på tärningsspelet *Etthundra*. Maskinintelligensen, som fick träna *20 miljoner* spel, testades sedan både mot människor och mot andra, mindre tränade maskinintelligenser. Resultaten visar att maskinintelligensen inte blivit bättre än människor, dock tyder resultatet på att maskinintelligensen presterar bättre efter träning.

Abstract

The purpose of this study is to investigate whether or not a machine intelligence that has been trained according to the Q-learning method consistently can defeat humans in the dice game *Etthundra*. The machine intelligence, which got to train *20 million* games, was then tested against both humans and other, less-trained machine intelligences. The results show that the machine intelligence can not consistently beat humans. However, the results indicate that the machine intelligence performs better after training.

Innehållsförteckning

Sammanfattning **i**

Abstract **i**

1 Inledning **1**
1.1 Bakgrund . . . 1
1.2 Syfte . . . 1
1.3 Frågeställning . . . 1
1.4 Avgränsningar . . . 1

2 Teori **2**
2.1 Etthundra . . . 2
2.2 Maskininlärning . . . 2
2.2.1 Q-learning . . . 3
2.2.2 Maskininlärning och tärningsspelet Etthundra . . . 3
2.2.3 SARSA . . . 4
2.3 Python . . . 4
2.4 Materiel . . . 4
2.5 Metod . . . 4
2.5.1 Maskinintelligens mot människor . . . 4
2.5.2 Maskinintelligens mot maskinintelligens . . . 4
2.6 Källkritik . . . 5

3 Utförande **6**

4 Resultat **7**

5 Diskussion och slutsats **9**

Källhänvisning **11**

A Q-table - exempel **11**

B Data och källkod **11**

1 Inledning

1.1 Bakgrund

Överallt i samhället kan man hitta maskinintelligenser. Man kan bland annat hitta dem i form av smarta röstassistenter, riktad reklam på diverse hemsidor och som självkörande bilar. Maskinintelligens har många användningsområden, men eftersom området är så nytt och relativt outforskat kan maskinintelligens fortfarande inte appliceras på alla tänkbara användningsområden än. Däremot är kunskap inom maskinintelligens mycket attraktivt på arbetsmarknaden och inom en snar framtid kan maskinintelligens komma att förbättra vår vardag ännu mer.[?]

Ett sådant tidigare nämnt användningsområde är självkörande bilar, som skulle kunna hjälpa Sverige att uppnå nollvisionen. Ett annat användningsområde är inom medicin. Maskinintelligenser används redan idag för att assistera doktorer vid diagnostisering av diverse sjukdomar och tillstånd, bland annat cancer.[?] Dessutom skulle en maskinintelligens kunna finna nya kemiska föreningar som potentiellt skulle kunna användas för att bota tidigare obotliga sjukdomar.[?] Ännu ett område tillämpningsområde är miljön. En maskinintelligens skulle kunna uppfinna nya, effektivare motorer som belastar miljön mindre, eller optimera redan existerande motorer och på så sätt hjälpa mänskligheten nå klimatmålen.[?]

1.2 Syfte

Syftet med detta gymnasiearbete är att träna maskinintelligenser med hjälp av Q-learning och sedan undersöka om dessa konsekvent kan besegra människor i tärningsspelet *Etthundra*.

1.3 Frågeställning

Går det att med hjälp av maskininlärning träna maskinintelligenser till att bli bättre än människor på tärningsspelet *Etthundra*?

1.4 Avgränsningar

Denna rapport kommer endast att avhandla maskininlärningsmetoden Q-learning, även om det finns andra mer effektiva modeller.

2 Teori

2.1 Etthundra

Tärningsspelet *Etthundra* spelas av två spelare. Målet i spelet är att först vara den som samlar ihop 100 poäng. För att samla ihop poäng turas spelarna om att slå tärningen. När det blir en spelares tur får den slå med tärningen. När spelaren slår med tärningen läggs poängen som tärningen visar till spelarens temporära poäng. Spelaren får sedan fortsätta att slå med tärningen så länge den behagar, eller tills spelaren slår en etta. I fallet då spelaren väljer att avsluta sin tur överförs spelarens temporära poäng till dess permanenta poäng. Därefter nollställs de temporära poängen och turen går över till den andra spelaren. I det andra fallet, då spelaren slår en etta, läggs inte spelarens temporära poäng till dess permanenta poäng utan de temporära poängen nollställs och turen går sedan direkt över till den andra spelaren. Detta fortsätter tills någon av spelarnas permanenta poäng tangerar eller överskrider 100 poäng.

2.2 Maskininlärning

Maskininlärning är ett samlingsord för olika metoder att lära en dator att lösa en viss uppgift utan att den som programmerar metoden behöver kunna lösa uppgiften. Detta i skillnad till konventionell artificiell intelligens där datorn löser uppgifter baserad på logik som en människa har programmerat med hjälp av olika if-satser. När uppgiften som datorn ska lösa är simpel och/eller enkel nog är det oftast lättare och/eller snabbare att låta datorn lösa uppgiften med hjälp av logik programmerad av en människa. När uppgiften som ska lösas är komplicerad och/eller stor, kan det vara fördelaktigt att använda sig av maskininlärning. Ett exempel är självkörande bilar. Att förklara för en dator hur man kör en bil med hjälp av programmerad logik är i princip omöjligt, bilkörning är för nyanserat. För att se varför kan du tänka dig att du står vid en ljussignal i en korsning med en bil framför dig, du ska svänga vänster och väntar på grön signal. Ljussignalen blir grön men bilen framför dig kör inte direkt och du måste vänta. När bilen framför väl kör, kan du inte svänga till vänster direkt eftersom bilarna som kör mot dig från andra sidan även har grönt och du måste vänta på att alla bilarna passerar innan du kan köra vidare. När bilarna har passerat kan du fortsätta din vänstersväng men måste ännu en gång stanna för att det finns fotgängare på övergångsstället på vägen du precis svängde in på. Bara vid en vänstersväng i en korsning behöver man hålla koll på väldigt många variabler och att programmera logik som kör en bil hade därför varit omöjligt. Istället använder man sig av maskininlärning. Genom att samla in data om hur man kör bil med hjälp av bland annat kameror, kan en dator sedan få analysera denna data och försöka att själv dra slutsatser kring hur man kör bil. Datorns slutsatser kring detta kan sedan testas i simulationer och utvärderas för att göra datorn bättre på att köra bil.

Även om maskininlärning brukas användas till stora och komplicerade problem går det även att applicera på relativt lätta och små problem, däribland tärningsspelet *Etthundra*.

2.2.1 Q-learning

Q-learning är en maskininlärningsmetod som baseras på positiv och negativ förstärkning. Datorn låts försöka lösa en uppgift och blir efter försöket belönad eller bestraffad. Om datorn löste uppgiften, eller ett delmål till uppgiften, blir den belönad. Om datorn inte löste uppgiften, eller om den gjorde någonting som anses vara dåligt, blir den bestraffad. Datorn kommer sedan ihåg belöningen eller bestraffningen och associerar denna med det sätt den försökte lösa uppgiften på just denna gång. Denna process av försök till att lösa uppgift, belöning eller bestraffning och sedan associering kallas att datorn tränar och fortsätter ända tills datorn anses lösa uppgiften på ett tillfredsställande sätt. I början av träningen slumpar datorn i princip alla sina drag, men gradvis under träningens gång baseras fler och fler av datorns val på den kunskap den samlat. Hur snabbt denna stegringen sker bestäms av ett ε-värde. Högt ε leder till en långsammare stegring, lågt ε leder till en snabb stegring. Denna stegringen finns för att datorn ska utforska många olika kombinationer av drag och inte anta att det bästa sättet att utföra uppgiften är på det sättet den slutförde uppgiften för första gången.

Sättet på vilket datorn kommer ihåg om ett drag är bra eller dåligt är genom en tabell, ett Q-table. På vänster sida av tabellen finns varje giltig position i spelet *Etthundra* och överst finns de möjliga dragen, att slå eller att stanna. Varje kombination av position och drag får en siffra associerad med sig baserad på hur bra det är att utföra detta draget från just denna positionen. För exempel, se bilaga A. Att beräkna värdet av siffran för en kombination av position och drag är enkelt om draget leder till en belöning eller en bestraffning. Värdet av siffran blir då bara det värde som associeras med en belöning respektive bestraffning, exempelvis 100 respektive -100. För att beräkna värdet av siffran för en kombination av position och drag som inte direkt leder till en belöning eller bestraffning används en SARSA-algoritm. Med hjälp av denna kan värdet för siffran beräknas då algoritmen tar hänsyn till om ett drag i framtiden leder till en belöning eller bestraffning. Detta eftersom ett drag som leder till ett drag som ger belöning är bättre än ett drag som leder till ett drag som inte ger belöning, eller värre, en bestraffning. När datorn har tränat färdigt och den bes spela *Etthundra*, är allt den behöver göra att observera vilken position som spelet är i inför varje slag och sedan välja det draget med det högsta associerade värdet. Se bilaga A för exempel.

2.2.2 Maskininlärning och tärningsspelet Etthundra

Antalet giltiga positioner i *Etthundra* är ungefär 500 000. Varje spelare kan ha en poäng mellan 0 och 100 och en temporär poäng mellan 0 till 100. Detta leder till att antalet giltiga positioner är $100^3 = 1000000$, men eftersom att det inte är nödvändigt för en spelare som har 60 poäng att slå mer än 40 poäng på en runda leder detta till att antalet giltiga positioner ungefär halveras. Därav finns det ungefär 500 000 giltiga positioner och dimensionen av Q-tablet är därför ungefär: 500 000 positioner x 2 drag.

2.2.3 SARSA

Värdet för en kombination av position och drag Q, beräknas med hjälp av SARSA-algoritmen:

$$Q(S, A) \leftarrow Q(S, A) + \alpha(R + \gamma \cdot Q(S', A') - Q(S, A) \tag{1}$$

Där S står för position (engelska state), A står för drag (engelska action), R för värdet av belöning/bestraffning (engelska reward), S' för nästa position och A' för nästa drag. Variabeln α bestämmer hur mycket värdet Q förändras av ny information. Variabeln γ bestämmer hur mycket värdet Q ska baseras på framtida belöningar/bestraffningar.

2.3 Python

Programmeringsspråket Python är mycket lämpat för maskininlärning. Detta då Python är mycket modulärt, d.v.s det finns många färdigskrivna bibliotek med färdig kod som man lätt kan implementera i sina egna program. Detta gör det mycket lätt att implementera svåra och avancerade koncept kopplade till maskininlärning med få rader kod.

Python är också väldigt användarvänligt då språket är väldigt likt engelskan, vilket gör att många lär sig Python som sitt första programmeringsspråk. Detta har gjort Python till ett av de mest använda programmeringsspråken vilket underlättar om man skulle få problem med sin kod, då det antagligen finns resurser på Internet som behandlar just det problem du upplever.[?]

Dessa anledningar gör Python till det självklara valet när man ska programmera och träna maskinintelligenser.

2.4 Materiel

En dator med Python installerat.

2.5 Metod

2.5.1 Maskinintelligens mot människor

För att undersöka hur bra maskinintelligensen presterade gentemot människor fick ett antal personer möta maskinintelligenserna. Varje person fick spelet *Etthundra* förklarat för sig, därefter fick de börja spela mot en förutbestämd nivå av maskinintelligensen. Totalt testades fyra nivåer av maskinintelligensen: 100 (*nivå 1*), *5 miljoner* (*nivå 2*), *10 miljoner* (*nivå 3*) och slutligen *20 miljoner* (*nivå 4*) tränade spel. Alla mänskliga spelare spelade vartannat spel som spelare ett och vartannat spel som spelare två. Totalt spelade varje spelare fyra spel. Efter varje färdigspelat spel antecknades resultatet.

2.5.2 Maskinintelligens mot maskinintelligens

För att se om maskinintelligensen blivit bättre, fick den mest tränade maskinintelligensen, *nivå 4*, möta: *nivå 1*, *nivå 2* och *nivå 3*. Under en match spelades 3000 spel. Totalt spelades alltså 9000 spel mellan den mest tränade maskinintelligensen, *nivå 4*, och de olika träningsnivåerna. Resultaten antecknades efter varje färdigspelad match.

2.6 Källkritik

Alla källor som har använts i denna rapport har valts ut med stor omsorg. Endast förstahandskällor har använts för att öka källornas trovärdighet.

Alla artiklar som används är från högt rankade universitet, alla med god tidigare historia gällande tidigare publicerade artiklar. Således kan de artiklar som använts anses vara mycket trovärdiga.

Den undersökning som rapporten har refererat till två gånger är från Stackoverflow, en fråga-svar hemsida för utvecklare. Den är flitigt använd av industrin och är en mycket tillförlitlig källa. Varje år genomför de en undersökning för att ta reda på en rad parametrar och deras undersökningar har alltid många deltagare. Detta gör att en stor mängd data ackumuleras samt att deltagarna är från hela världen, vilket ger undersökningen mycket och tillförlitlig data. Således kan undersökningen ses som en mycket trovärdig källa som representerar hela utvecklar-industrin.

3 Utförande

Undersökningen startade med att en maskinintelligens programmerades utifrån Q-learning modellen, se 2.2.1. Q-learning modellen baserades på SARSA ekvationen, se ekvation 1.

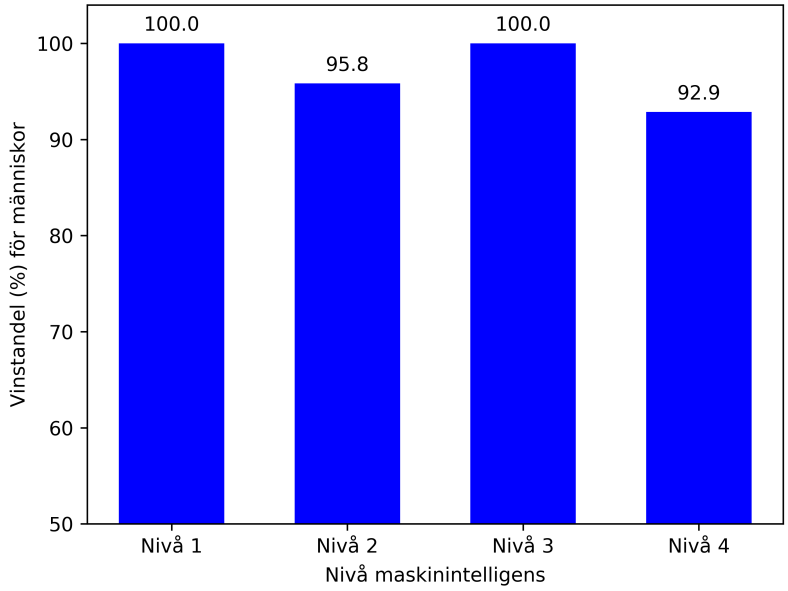
Därefter fick två instanser av maskinintelligensen träna mot varandra tills de spelat totalt *20 miljoner* spel. Under tiden de tränade sparades maskinintelligensens utveckling med jämna intervaller, för att möjliggöra undersökning av dess utveckling.

Sedan testades instanserna mot både människor och mot varandra, se 2.5. Resultaten registrerades i CSV-format, se bilaga B.

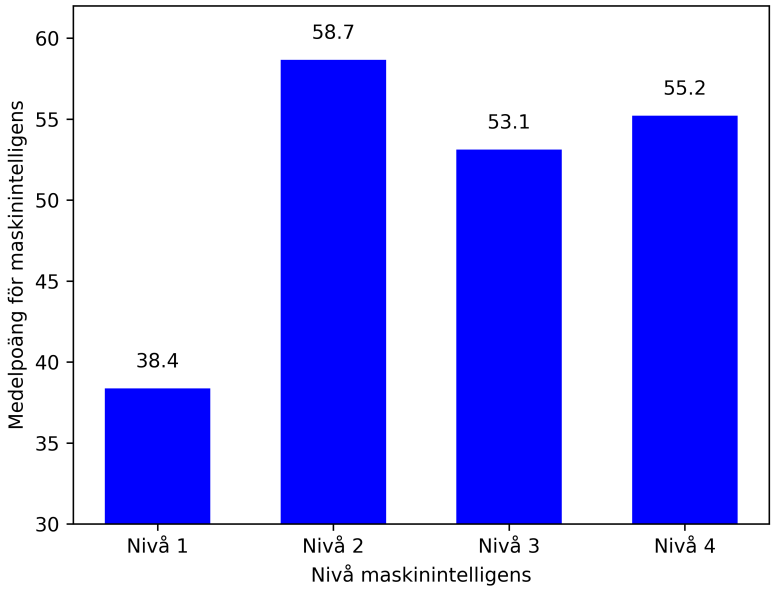
Därefter sammanställdes resultaten i illustrerande grafer för att göra resultatet nedbrytbart.

4 Resultat

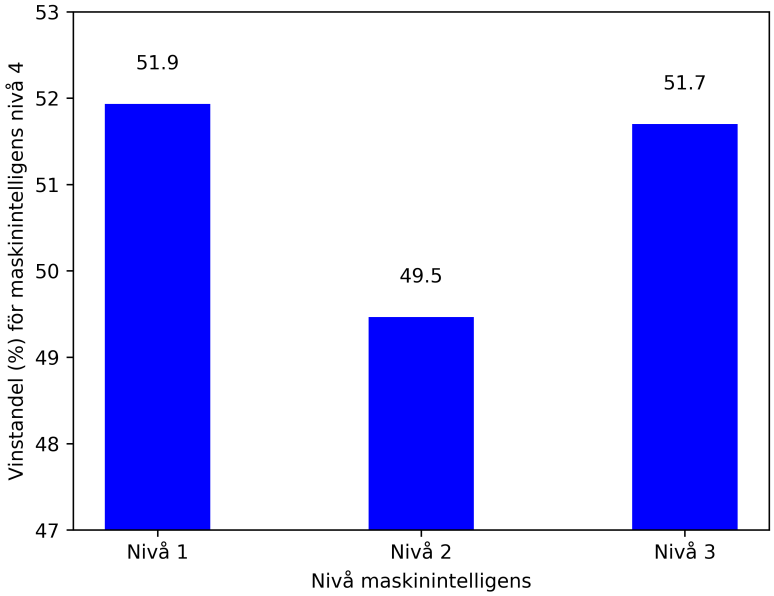
När människor mötte maskinintelligens *nivå 1* vann människorna 100% av spelen. Respektive värde för maskinintelligens av *nivå 2*, *nivå 3* och *nivå 4* är 95,8%, 100% samt 92,9%. Se figur 1. Under dessa spelen uppnådde maskinintelligens *nivå 1* i genomsnitt 38,4 poäng, betydligt lägre än vad de andra nivåerna presterade. Dessa uppnådde istället en medelpoäng mellan 53 och 59 poäng. Se figur 2. När maskinintelligensen *nivå 4* mötte maskinintelligenserna av *nivå 1* och *nivå 3* vann den en majoritet av spelen. Dock så förlorade den majoriteten av spelen när den mötte maskinintelligensen *nivå 2*. Se 3. För all data som tabellerna baseras på, se bilaga B.



Figur 1: Figuren visar på hur stor andel av spelade spel som människa vinner mot maskinintelligens av en given nivå.



Figur 2: Figuren visar medelpoängen för en maskinintelligens av en viss nivå då den mötte människor.



Figur 3: Figuren visar vinstandelen i procent för maskinintelligens nivå 4 då den möter en maskinintelligens av en given nivå.

5 Diskussion och slutsats

Figur 1 visar på att maskinintelligensen ej är bättre än människor på tärningsspelet *Etthundra*. Även om maskinintelligens *nivå 4* vinner flest spel kan ingen slutsats dras från denna figur förutom att människor är bättre än maskinintelligensen. Detta då resultatet baseras på en relativt liten mängd spel samt att skillnaden i vinstandel är liten.

Däremot visar figur 2 att maskinintelligensen blir bättre då den har tränat fler spel. Detta reflekteras i att medelpoängen maskinintelligensen ackumulerar under ett spels gång ökar i takt med att maskinintelligensen tränat fler spel. Överlägset sämst är maskinintelligens *nivå 1* vars medelpoäng är 38,4. Marginellt bäst presterar maskinintelligens *nivå 2*. Denna data är mer tillförlitlig än datan i figur 1 då poängackumulation är mer nyanserat än en vinst eller en förlust. Man kan förlora ett spel med 1 poäng eller med 100 poäng. Bilden av maskinintelligensernas prestation blir således mer detaljerad eftersom man får poäng oftare än man vinner eller förlorar och därför kan figur 2 anses mer tillförlitlig än figur 1.

Figur 3 förstärker datan i figur 2 då denna också pekar på att maskinintelligens *nivå 2* är den som presterar bäst. Det enda som talar emot detta är figur 1, där *nivå 4* presterar bäst, men som tidigare nämnts baseras den på minst mängd data och är därför minst tillförlitlig när man ska jämföra de olika nivåerna av maskinintelligens mot varandra.

Eftersom maskinintelligensen enbart tränade emot en annan maskinintelligens som hade tränat lika mycket som den själv, lär den sig hela tiden av att spela mot någon som är lika bra, marginellt sämre eller marginellt bättre än sig själv. Detta betyder att den får belöningar för drag och/eller kombinationer av drag som inte skulle lett till en vinst om den mötte en bättre spelare. Maskinintelligensen lär sig därför att spela dåligt, då det räcker med att spela dåligt för att vinna över en sämre motståndare. Det skulle därför vara bättre om maskinintelligensen fick träna mot en väldigt bra motståndare redan från början. Då skulle den lärt sig vad som fungerar och vad som inte fungerar när man möter en bra motståndare, inte vad som fungerar mot en dålig motståndare. Det bästa hade varit om maskinintelligensen fått träna genom att spela mot människor. Maskinintelligensen hade då lärt sig vad som är ett bra sätt att spela mot människor respektive vad som är ett dåligt sätt att spela mot människor. Detta är dock orimligt då maskinintelligensen behöver träna flera miljoner spel för att komma upp i en rimlig nivå. Om man räknar lågt och antar att ett *Etthundra* spel tar en minut att spela och att maskinintelligensen konsekvent kan slå människor efter *1 miljon* tränade spel, skulle det behövas 695 dagars kontinuerligt spelande för att uppnå detta. Det näst bästa alternativet hade varit att spela mot logik baserat på det optimala sättet att spela *Etthundra*. Då hade maskinintelligensen fått möta en perfekt "spelare" och hade således behövt lära sig att spela på en väldigt hög nivå för att ha en chans att vinna.

En annan förbättring hade varit att experimentera med ε-värdet. ε-värdet bestämmer hur snabbt stegringen går från att maskinintelligensen slumpmässigt bestämmer alla sina drag under träningen, till att alla dess drag är beräknade. Om detta värdet är för lågt, leder det till att maskinintelligensen slutar att slumpmässigt välja sina drag tidigt och därför hinner den ej utforska många och utvärdera många olika positioner. Dess beräknade drag baseras då på en liten mängd data. Maskinintelligensen saknar med ett lågt ε-värde erfarenheter och har därför inte tillräckligt med kunskap för att ta bra beslut. Om ε-värdet däremot är för högt kommer maskinintelligensen ta längre tid på sig

innan den slutar att slumpmässigt välja sina drag. Detta är i sig självt inget dåligt, särskilt inte om maskinintelligensen hela tiden tränar mot en stark motståndare, då leder enbart ett högt ε-värde till att träningsprocessen tar längre tid. Om det istället är så att det är två maskinintelligenser som möter varandra, är ett för högt ε-värde dåligt. Detta då maskinintelligensen och dess motståndare båda kommer att välja sina drag slumpmässigt länge och de lär sig då att spela mot en motståndare som spelar slumpmässigt. Experimentation med α och γ-värden i SARSA-algoritmen (se 2.2.3) hade även kunnat ge ett bättre resultat.

A Q-table - exempel

Tabell 1: Exempel på hur ett Q-table kan se ut. Värdena är påhittade.

Poäng	**Slå**	**Stanna**
40 - 23 - 7	30,32	10,2
40 - 23 - 12	20,74	23,12
52 - 46 - 0	40,85	8,39
52 - 46 - 6	32,16	13,43

Kolumnen längst till vänster beskriver hur positionen ser ut. Exempelvis beskriver den andra raden (40-23-7) att spelaren har 40 poäng, att motståndaren har 23 poäng och att spelarens temporära poäng är 7. De två andra kolumnerna visar ett värde baserat på hur bra datorn tycker det är att göra respektive drag, slå eller stanna. När datorn utvärderas för att se hur bra den presterar slumpar den ej sina drag utan baserar enbart sina drag på värdena i kolumn två och tre. Exempelvis skulle datorn i positionen 40-23-12 välja att stanna då 23,12 är större än 20,74.

B Data och källkod

Datan och källkoden för denna undersökning ligger uppe på GitHub:

https://github.com/Zelmyx/Gymnasisearbete-Etthundra