//

7 Machine Games in a Box

maXbox Sixpack – **g**ame **a**gainst **m**achine **e**volution (game)

"In the face of ambiguity, refuse the temptation to guess."

– The Zen of Python

This paper from Max Kleiner introduces the basic idea of machine games you play against it. All games are available on board in maXbox as Add Ons or Plug ins, so you don't need to start a script, just download maXbox and start the executable. It **is** that simple, so here **is** an overview:

http://www.softwareschule.ch/examples/machinelearning.jpg

Of course, machine learning (often also referred to **as** Artificial Intelligence, Artificial Neural Network, Big Data, Data Mining **or** Predictive Analysis) **is not** that new field **in** itself **as** they want to believe us. A machine game is a game you play against a machine (in that sense only first four board games):

- Chess (The Board Game.)
- Checkers (incredibly old and has a fascinating history.)
- 4Gewinnt (Connect Four is a two-player game with "perfect information".)
- Reversi (Othello as a two-player game created in the 19th century.)
- Tetris (Creating a horizontal line of ten units without gaps.)
- Asteroids (Good old days of arcade games.)
- Minesweeper (Clear a rectangular board containing mines.)

At its core, most algorithms should have a proof of classification **and** this **is** nothing more than keeping track of which feature gives evidence to which **class**. The way the features are designed determines the model that **is** used to learn. This can be a confusion matrix, a certain confidence interval, a T-Test statistic, p-value **or** something **else** used **in** hypothesis[1] testing. In play-checkers-using-standard-rules or other games you can most decide in four cases:

positive - negative - false positive(false alarm) - false negative(missed alarm)

http://www.softwareschule.ch/examples/decision.jpg

Lets start with some code games precompiled and built in, assuming that you have Python or maXbox already installed (everything at least **as** recent **as** 2.7 should be fine **or** better 3.6 **as** we do and maXbox from 4.2 to 4.6).

"Chess"

You find the chess board in maXbox at

Menu/Options/Chess V4

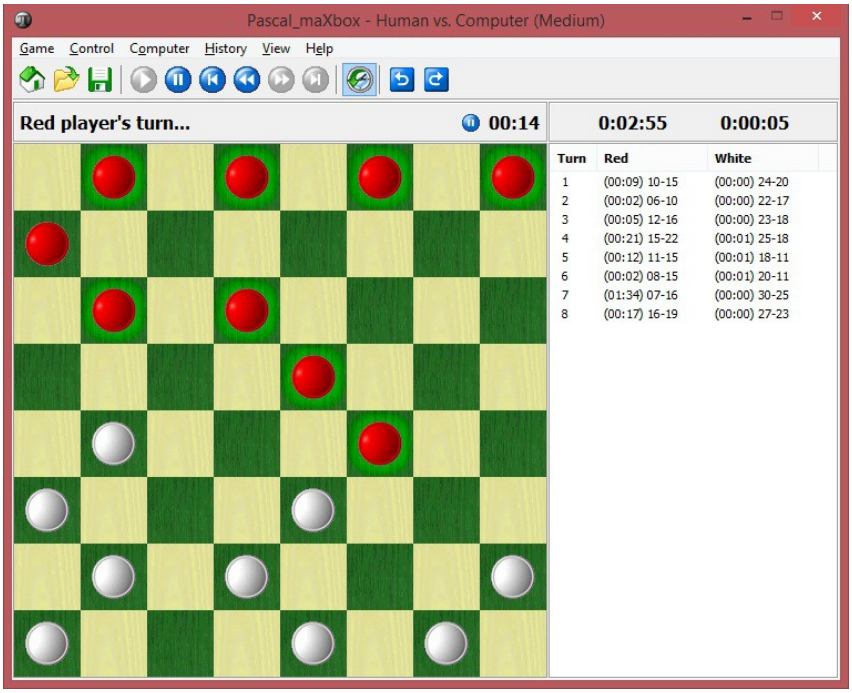
To play against the machine switch on the right side to ComputerMax.
For many cpus the processor utilization will too high. You can also set machine against machine ComputerMax versus Computer then alter the search depth to 5 or 6 to slow down the game steps (otherwise you cant follow it).

1 A thesis with evidence

Easy mode is a little bit hard (search depth 3), and hard is very hard indeed.

"Checkers"

Checkers is a classic board game that comes to life many years ago.
Menu/Options/Checkers

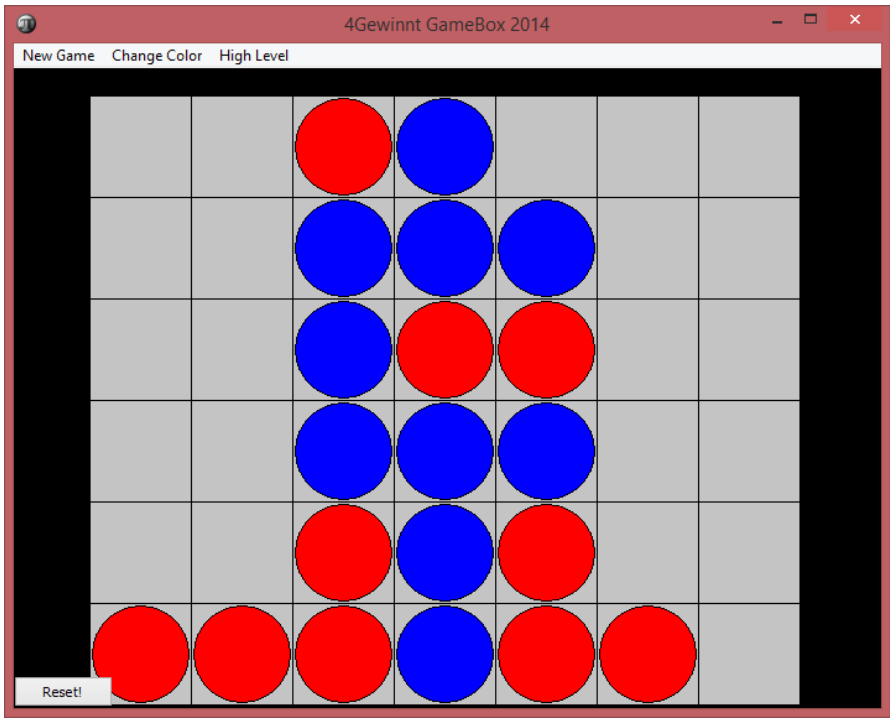


The board game Checkers is played on an 8x8 board and each player has 12 checker pieces. This is a head to head game, so play against the computer's artificial intelligence or a friend's intelligence! Red checkers will always go first. Checkers may only move diagonally one space, with the exception of jumping. A checker may move more than one space if they can jump one of the opponent's

checker pieces which is located immediately in their diagonal vicinity and onto a free space. Multiple jumps can occur. To move a checker, click the checker you wish to move and then click the spot you'd like to move onto.
When you jump a checker of the opposite color, the checker is removed from the board. Move your checkers into your opponent's row to get kinged. Kinged checkers can move forward and backward so these are very valuable. Whichever player jumps and removes all their player's checkers wins.
The first Checkers-like playing board was discovered by archaeologists in a dig in Mesopotamia (now Iraq) in the city of Ur. Carbon dating showed the board to be from about 3000 BCE, or more than 5000 years old. It has also been the focus of several inventive computer programmers interested in AI.
https://www.thesprucecrafts.com/play-checkers-using-standard-rules-409287

"4Gewinnt"

Connect Four (also known as 4Gewinnt, Captain's Mistress, Four Up, Plot Four, Find Four, Four in a Row, Four in a Line and Gravitrips (in Soviet Union)) is a two-player connection game in which the players first choose a color and then take turns dropping one colored disc from the top into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the next available space within the column.



The objective of the game is to be the first to form a horizontal, vertical, or diagonal line of four of one's own discs. Connect Four is a solved game. The first player can always win by playing the right moves.
Connect Four also belongs to the classification of an adversarial, zero-sum game, since a player's advantage is an opponent's disadvantage.
One measure of complexity of the Connect Four game is the number of possible games board positions. For classic Connect Four played on 6 high, 7 wide grid, there are 4,531,985,219,092 positions for all game boards populated with 0 to 42 pieces.

The game was first sold under the Connect Four trademark by Milton Bradley in February 1974.
https://en.wikipedia.org/wiki/Connect_Four

"Reversi"

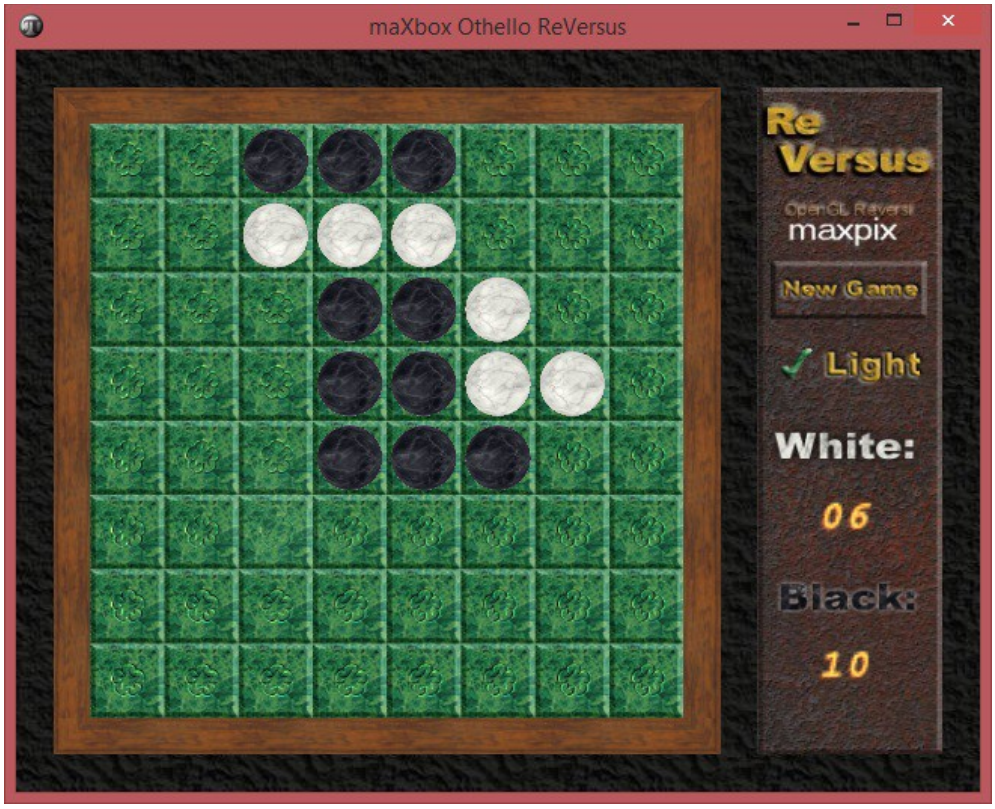
The Game is also called Othello. Also called "Othello", this game has millions of people addicted. Place your piece on an empty square so that one (or more) of the opponent's pieces are **between** yours. You win by having **more of your pieces** than the opponents at the end of the game.
Othello is quite simple to learn, but can take a lifetime to master.

To play the game with *maXbox4.exe* you need to download some OpenGL resources: opengldata.zip
https://sourceforge.net/projects/maxbox/files/Examples/13_General/opengldata.zip/download
Unpack it to ../source/opengldata: After starting at

Menu/Options/Add ons/Reversi

it says "Reversi start in single mode ../source/OpenGLData" so this game does have a modal window to check memory leaks as a singleton for the OpenGL library: C:\WINDOWS\SYSTEM32\opengl32.dll and C:\WINDOWS\SYSTEM32\GLU32.dll

For lack of some shader functions you can switch off Light on the right side of the form.

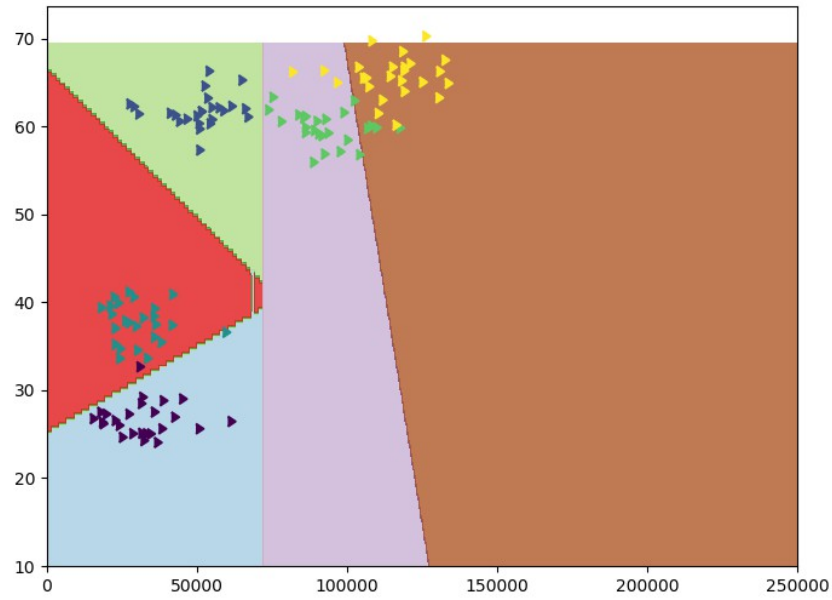



```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

Note: While it might initially seem like flipping as many discs as possible is the key to victory, this actually makes you more vulnerable. Most positions on the board can be outflanked; the edges of the board and the corners are the most stable positions.

Or another strategy could be to cluster your stones to gain more points like a cluster algorithm. It returns coordinate matrices **from** coordinate vectors. Make

N-D coordinate arrays for vectorized evaluations of N-D scalar/vector fields over N-D grids, given one-dimensional coordinate arrays x1, x2,..., xn.



Or just use predict for a given point:

```
print(svc.predict([[100000, 60]])) -  print(svc.predict([[50000, 30]]))
```

You should choose to ask a more meaningful question. Without some context, you might as well flip a coin like reversi.

"Tetris"

Tetris is a tile-matching puzzle video game, originally designed and programmed by Russian game designer Alexey Pajitnov. It was released on June 6, 1984, while he was working for the Dorodnitsyn Computing Centre of the Academy of Science of the Soviet Union in Moscow. He derived its name from the Greek numerical prefix tetra- and tennis, Pajitnov's favorite sport.

When such a line is created, it gets destroyed, and any block above the deleted line will fall. When a certain number of lines are cleared, the game enters a new level. As the game app progresses, each level causes the Tetriminos to fall faster, and the game ends when the stack of Tetriminos reaches the top of the playing field and no new Tetriminos are able to enter. Some games also end after a finite number of levels or lines. Play at:

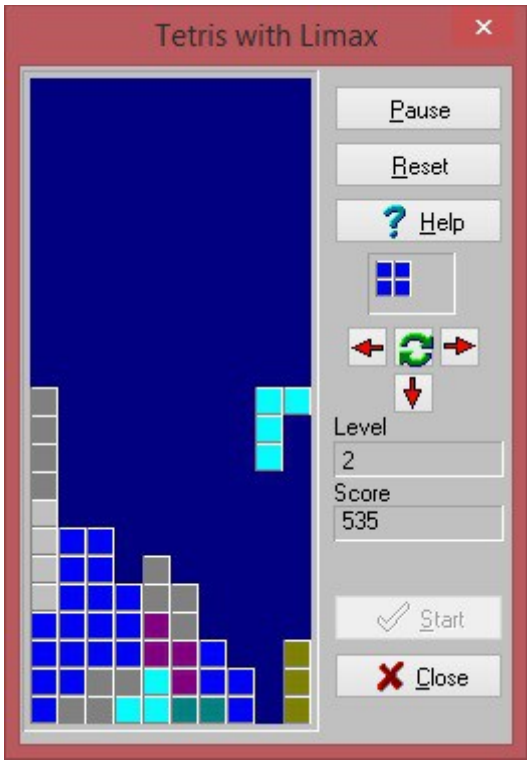
Menu/Options/Add ons/Tetris

Tetris can be like a confusion matrix in machine learning with four cases:

- True positive : you find the pattern to place it
- True negative : you avoid a false pattern position
- False positive : you missed (reject) a right piece/place (false alarm)
- False negative : you put (accept) the wrong piece (like screenshot below)

Of course it depends on the system hypothesis and reasoning, could be an alarm or a signal from your brain to made the right move.

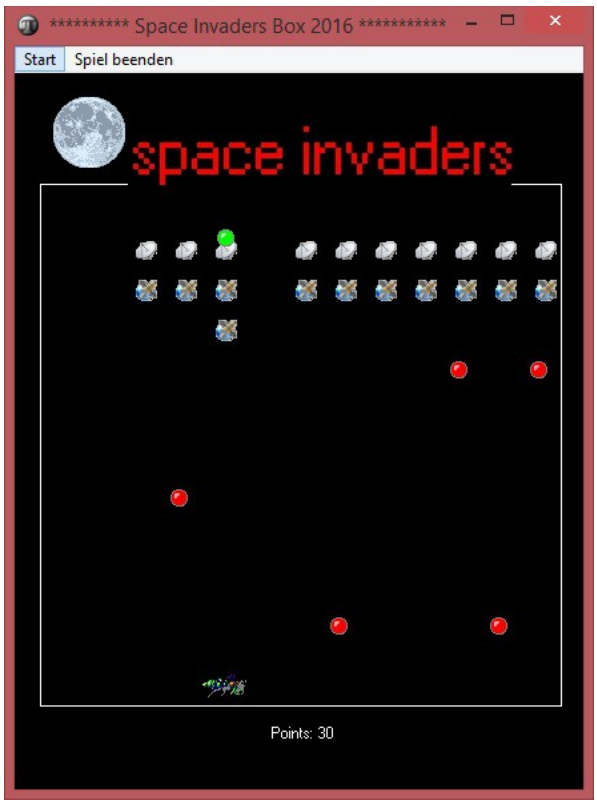
```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)
```



Write some code, run some tests, fiddle with features, run a test, fiddle with features, realize everything is slow, and decide to use more layers or factors. The scoring formula for the majority of Tetris products is built on the idea that more difficult line clears should be awarded more points.

"Asteroids"

Here at last we talk about Space Invaders and Asteroids – the race against machines after skynet.
Remember this old classic? Space Invaders is one of the most addicting games that was ever made. The idea is very simplistic. You are a space ship who must destroy the invading enemy space ships as they descend upon your little 8-bit world. Use the [SPACE] bar and cursors to fire your gun at them and blow each ship to pieces – I'll be back in four pieces.



https://sourceforge.net/projects/maxbox/files/Examples/13_General/714_spaceinvadersUnit2.PAS/download

Asteroids is a great game and an old classic and it's been reworked in flash to take you back to the 1980s and the pinnacle of arcade gaming. Try to achieve the highest score possible by destroying asteroids and enemy flying saucers.

https://sourceforge.net/projects/maxbox/files/Examples/13_General/714_astroids_GAME5.PAS/download

Load the script and the missing resources will be downloaded, unpack and started. Or you use from Menu/Help/Get Web Script and type:
http://www.softwareschule.ch/examples/asteroid.txt

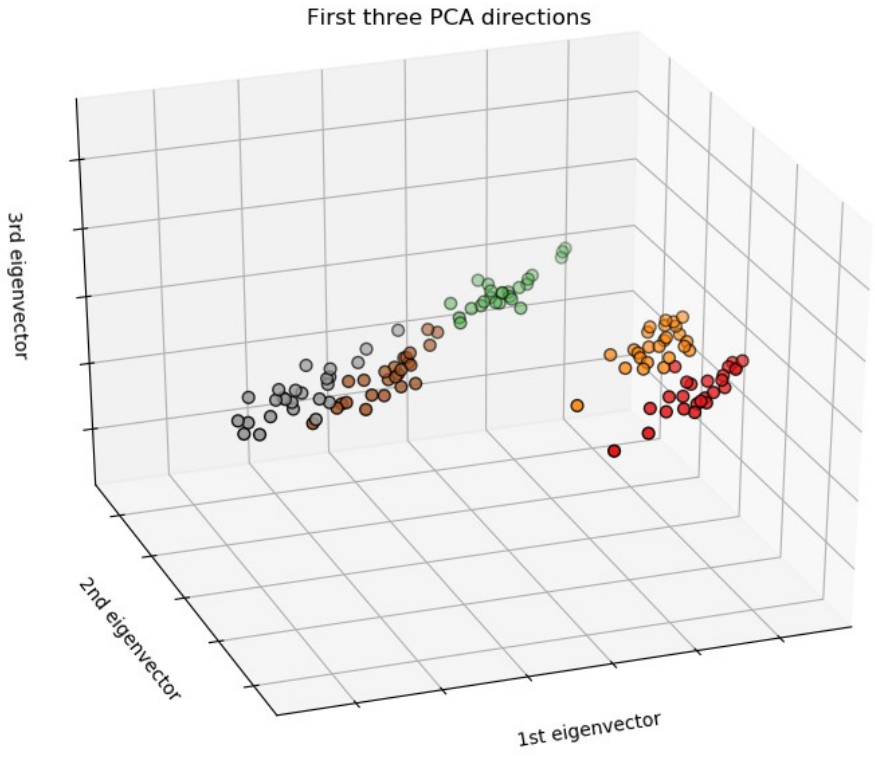
Maybe you get better stability with set ***ProcessMessagesOFF*** in menu or script.
These two games remember me of data reduction (in other words you reduce some figures in game or a plot)

This **is** dimensional reduction **as** the plot on
http://www.softwareschule.ch/examples/machinelearning.jpg
shows, its more a preparation but could also necessary to data reduction **or** to find a thesis.

Principal component analysis (PCA) **is** often the first thing to **try** out **if** you want to cut down number of features **and** do **not** know what feature extraction method to use.

PCA **is** limited **as** its a linear method, but chances are that it already goes far enough **for** your model to learn well enough.
Add to this the strong mathematical properties it offers **and** the speed at which it finds the transformed feature data space **and is** later able to transform between original **and** transformed features; we can almost guarantee that it also will become one of your frequently used machine learning tools.

Your ship has some really cool physics that take into account thrust, direction, and speed. Pressing the forward arrow speeds your ship up and pressing left or right will turn you counter-clockwise or clockwise in a 360 degree rotation. This allows for some quick maneuvering when trying to avoid asteroids or dodge enemy ship fire.



As PCA simply transforms the **input** data, it can be applied both to classification **and** regression problems. In this section, we will use a classification task to discuss the method.

The script can be found at:
http://www.softwareschule.ch/examples/811_mXpcatest_dmath_datascience.pas
..\examples\811_mXpcatest_dmath_datascience.pas

It may be seen that:

- High correlations exist between the original variables, which are therefore **not** independent
- According to the eigenvalues, the last two principal factors may be neglected since they represent less than 11 % of the total variance. So, the original variables depend mainly on the first two factors
- The first principal factor **is** negatively correlated with the second **and** fourth variables, **and** positively correlated with the third variable
- The second principal factor **is** positively correlated with the first variable
- The table of principal factors show that the highest scores are usually associated with the first two principal factors, **in** agreement with the previous results

```
Const
  N    = 11;  { Number of observations }
  Nvar = 4;   { Number of variables }
```

Of course, its **not** always this **and** that simple. Often, we dont know what number of dimensions **is** advisable **in** upfront. In such a case, we leave n_components **or** Nvar parameter unspecified when initializing PCA to let it calculate the full transformation. After fitting the data, explained_variance_ratio_ contains an array of ratios **in** decreasing order: The first value **is** the ratio of the basis vector describing the direction of the highest variance, the second value **is** the ratio of the direction of the second highest variance, **and** so on.

"Minesweeper"

Minesweeper strategy is the art of solving games. Techniques include learning patterns and where to click first, using guessing tactics and developing efficient clicking and mouse movement.

Sometimes in Minesweeper you need to guess. A typical case is a 50/50 situation where one mine is hidden in two squares. Guess quickly and move on. Thinking does not improve your chance of guessing correctly, it only wastes time. Waiting

to see if you guessed right also wastes time, so assume you survived and try to keep playing. Do not delay taking forced guesses - solving the rest of the board first is a waste of time if you end up guessing the wrong square.

Many players are impatient and guess instead of solving. Do not guess unless it is necessary.

Conclusion

That said, the question of whether to have opponents learn and/or adapt to the player at run-time is largely not a technological question - it's a design question. That is, it's not an issue whether it's difficult - rather, it's a question of whether it should be done at all, and how it would impact the player's experience. Would the player enjoy it if the computer quickly learned how to counter the player's particular strategies, and crushed him? Oftentimes, the designer's answer is a resounding "no".

At this point, automatic ML application to games is still pretty rare. What's much more common is automatic data collection and analysis, combined with manual, human interpretation and application to design.

Ref:

Building Machine Learning Systems with Python
Second Edition March 2015

DMath Math library for Delphi, FreePascal and Lazarus May 14, 2011

http://www.softwareschule.ch/box.htm

https://en.wikipedia.org/wiki/Tetris

http://www.minesweeper.info/wiki/Strategy

https://www.quora.com/What-are-some-examples-of-computer-games-which-use-machine-learning

Doc:

https://maxbox4.wordpress.com

https://www.tensorflow.org/

http://satirist.org/learn-game/

http://ai-depot.com/GameAI/Learning.html

ValueError: The truth value of an array with more than one element is ambiguous. Use a.any() or a.all()

ypeError: data type not understood

from mpl_toolkits.mplot3d import Axes3D