Board Features:
X_1 = Number of 2 X's and an open space in a row/column/diagonal
X_2 = Number of 2 O's and an open space in a row/column/diagonal
X_3 = Player who has taken the center space

We choose these board features because they will help a certain side win. If our program is playing as X, when it is our turn to go, if we see two X's in a row then we want to play the third X and win. The opposite is true for two O's in a row, we want to try and block the opponent from winning. Two X's in a row is a more favorable board state while two O's in a row is a less favorable board state.

For X_3 we choose these because controlling the middle is an advantage in tic tac toe. So our program would want to take the middle if it is still available in order to create a better board state.

At the start of our program we give both AI the same weights for these board features. As the program progresses, AI1 learns and changes these weights to get better at the game. We have our opposing program keep the same default weights to produce the fixed evaluation function. By the end of our program's execution, AI1 would have learned and changed its weights to win or draw more often, while our AI2 will still have the same weights for the board features.

The Writeup For our Tic-Tac-Toe Game

1. anything positive you enjoyed or learned from this assignment, (2) anything negative you didn't like about this assignment, (3) any parts of this assignment you found easy, (4) any parts of this assignment you found challenging or couldn't get working correctly, (5) how your team functioned, including details such as what each team member contributed, how the team communicated with each other, and how team software development & design was accomplished, and(
2. any other remarks you want to make

During this project, our group enjoyed finally putting in what we have been learning in class and finally being able to see how everything comes together. It's one thing working on a subject in class but when you finally apply it in real-world applications it really helps shade a different perspective on what you are learning. Being able to see how this can become a real-life application and even we see it in the everyday games we love to play. One of the things we didn't enjoy about the project was putting together the code and getting it to work. This was our first project and getting everything to work together smoothly wasn't as easy as it seems. Especially getting the machine learning part was the most difficult for our first time in implementing it all together to work. On the other side, the easier side was creating the basic code like getting the board to work and just the user interaction was more on the easier side. The hardest part was just getting the algorithm part working together. In the end, we were able to get the program to work all together and be able to finish the project on time. Our team was able to work together by meeting together on multiple times during the week and we each contributed to different parts of the code. Each member being able to work on the code on their own time and each member contributing enough from each peer.

TicTacToe
TicTacToe learning program for MachineLearning

When you open our file you can see a couple different documents.

In our board features game you can see our initial conditions set for the tic tac toe program and how we went about in creating it.

In our Graph pdf you can see all of our graph outcomes for the different games played by our program and see how the outcome was overall. Seeing a line graph of seeing the win vs lose ratio plays out over time.

In our Main python file "main.py" this is our main code for the tic tac toe game.

To run this file you need to do "cd.main.py"

When looking at our main code you can see that we start off by setting up our global variables and initialize the lists to store our wins, loses, and tie rate. Then we started creating our features for V train and determining the value of the board features for the program to run efficiently. Being able to determine the taken and empty positions on the board and from there letting the program do its thing. Then writing the code for showing how the game will be played out and how the match will occur. We created a class AI for the computer to self learn and being able to create a new value for each weight. With it then playing out and over time training to become efficient