Kai Mcconnell

Final Project Pokemon Battle Neural Network Predictor

Introduction:

For this project my goal was to work with an interesting dataset with real world applications and create a neural network to predict a given value. In this case the dataset I chose was the weedle cave Pokemon dataset. The reason I chose this dataset was because I thought it had interesting data which would work well in creating my neural network model, and because I love Pokemon. My love of Pokemon makes my neural network valuable to the real world application of predicting the result of Pokemon battles between different Pokemon. The goal of using this dataset will be to create a model which can predict the winning pokemon of a battle given the statistics of the 2 pokemon involved.

Experiment:

The dataset contains 3 different csv files. The first csv file contains all 800 Pokemon and their stats, the second contains a set of Pokemon battles that occur between 2 Pokemon and the result of the battle, and a the third contains a set of future battles to be predicted.

Pokemon Dataset first 10 values:

	#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
0	1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	FALSE
1	2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	FALSE
2	3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	FALSE
3	4	Mega Venusaur	Grass	Poison	80	100	123	122	120	80	1	FALSE
4	5	Charmander	Fire	NaN	39	52	43	60	50	65	1	FALSE

5	6	Charmeleon	Fire	NaN	58	64	58	80	65	80	1	FALSE
6	7	Charizard	Fire	Flying	78	84	78	109	85	100	1	FALSE
7	8	Mega CharizardX	Fire	Dragon	78	130	111	130	85	100	1	FALSE
8	9	Mega CharizardY	Fire	Flying	78	104	78	159	115	100	1	FALSE
9	1(	Squirtle	Water	NaN	44	48	65	50	64	43	1	FALSE

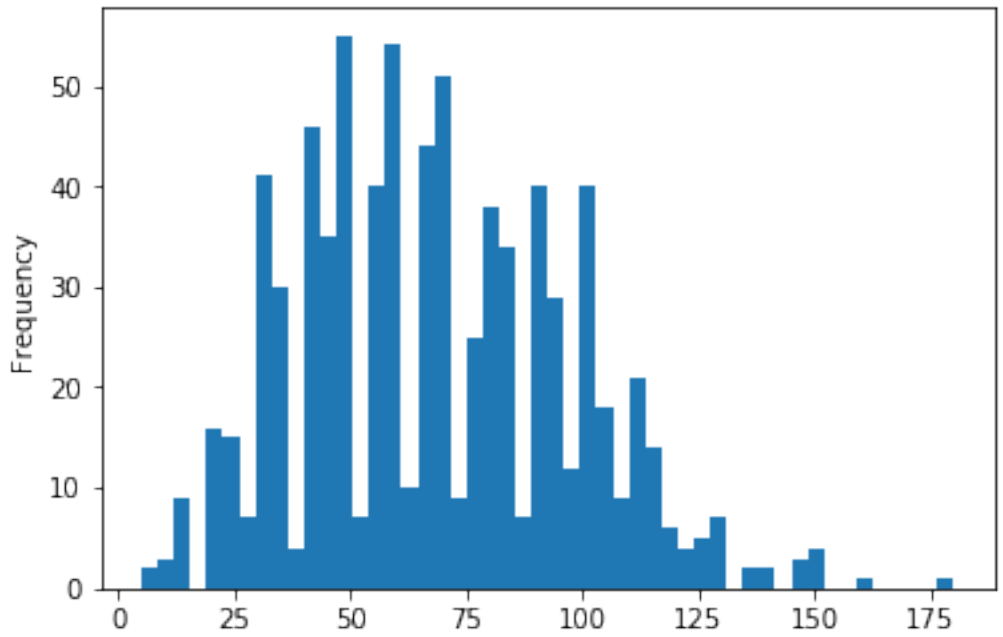
Battle Dataset first 10 values:

	First_pokemon	Second_pokemon	Winner
0	266	298	298
1	702	701	701
2	191	668	668
3	237	683	683
4	151	231	151
5	657	752	657
6	192	134	134
7	73	545	545
8	220	763	763
9	302	31	31

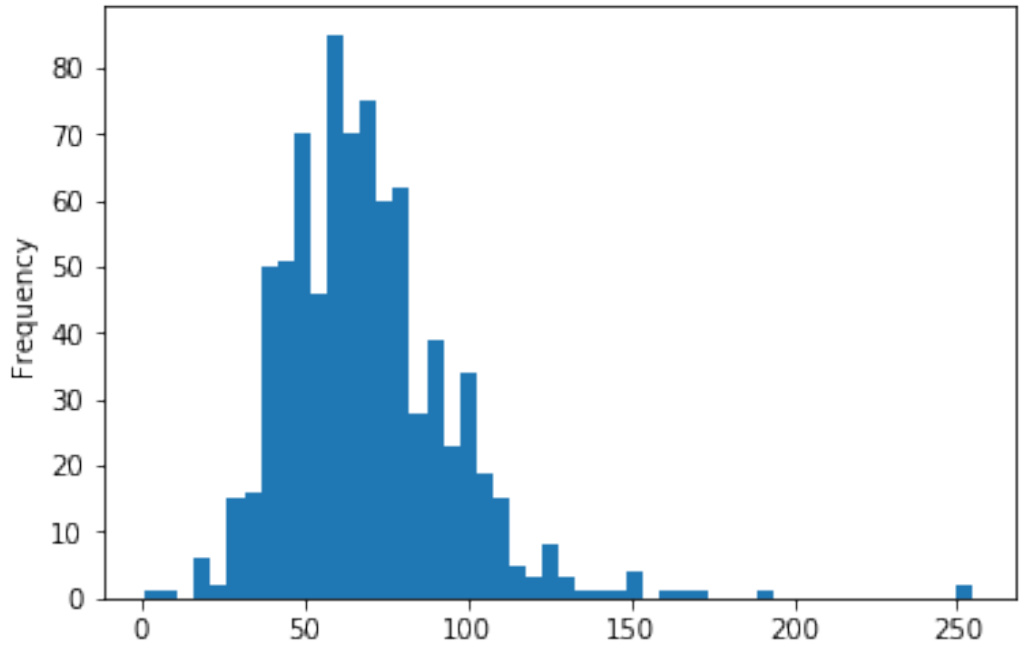
In my case I only needed two of the files the csv file containing the stats of all 800 pokemon and the csv file containing the set of pokemon battles and their winners. The first thing I needed to do

is explore the data contained in these two files and see how their values correlated to each other. The first step I took was in looking at the distribution of pokemons base stats. The following figures show what the distribution for each stat looks like.

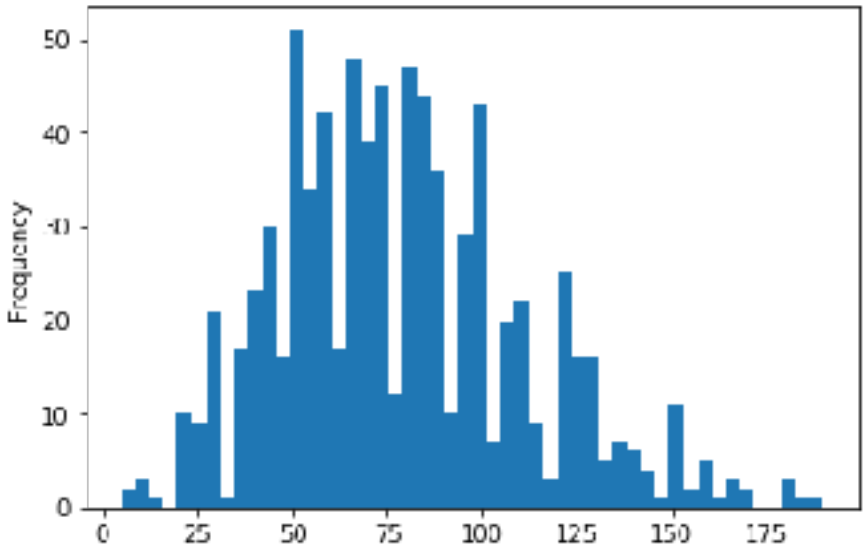
Speed Distribution:



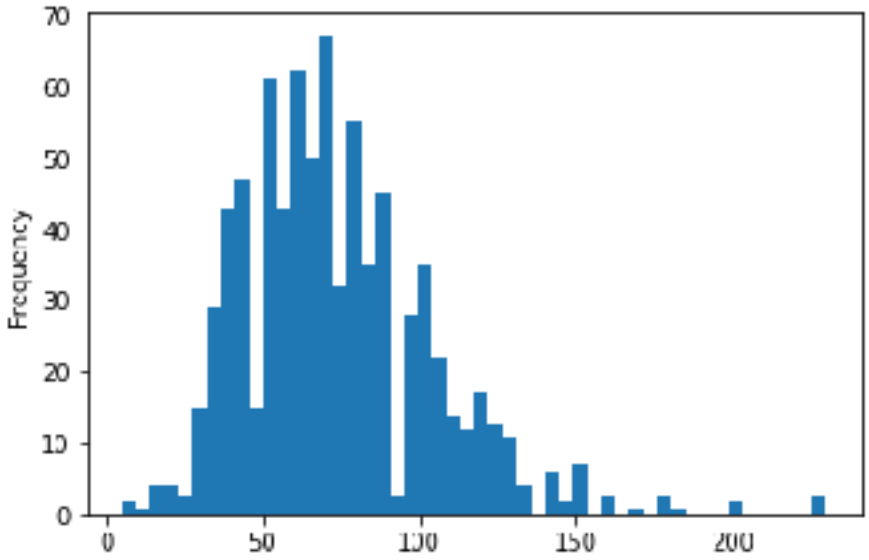
HP distribution:



Attack Distribution:



Defense Distribution:



Following the histograms I looked at the number of each pokemon type1.

# of each type of pokemon:

Water 112

Normal 98

Grass 70

Bug 69

Psychic 57

Fire 52

Electric 44

Rock 44

Dragon 32

Ghost 32

Ground 32

Dark 31

Poison 28

Fighting 27

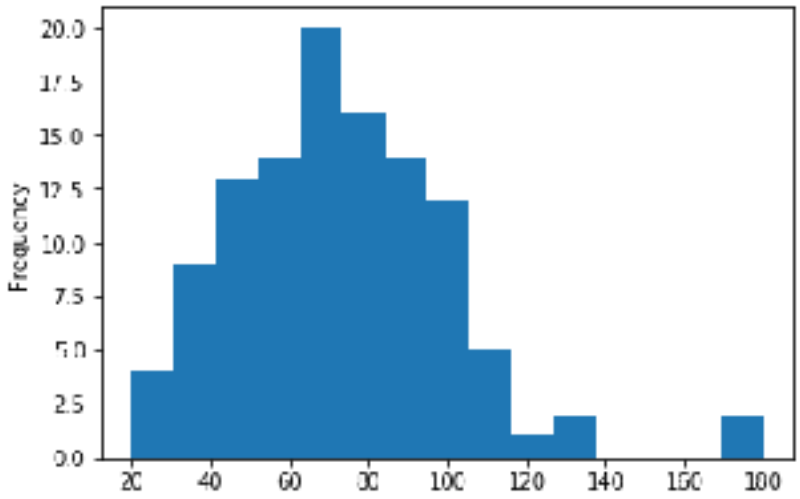
Steel 27

Ice 24

Fairy 17

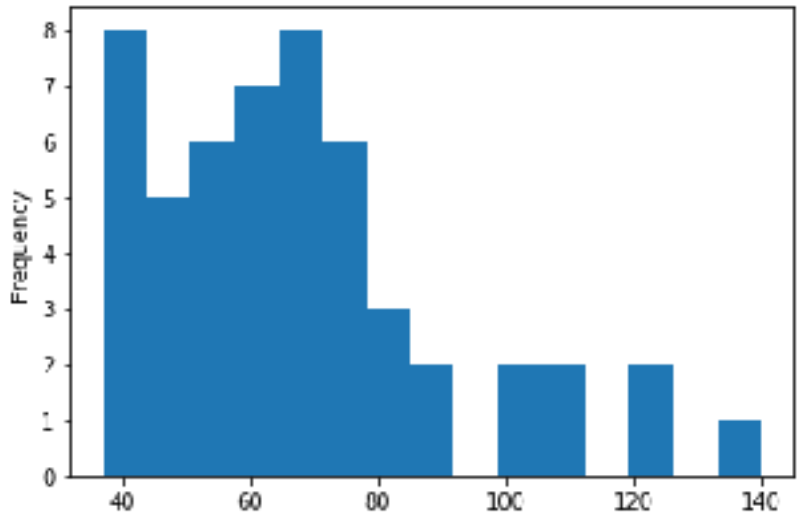
Flying 4

This told me the distribution of types I had to work with and lead to me adding to the dataset to explore these values. I worked to add two new columns to the 800 pokemon dataset to help with data exploration. These two new columns were the number of wins a pokemon had in the battle dataset, and the win rate of that pokemon. This involved taking data from the battle data set and using it to calculate the values for the 2 new columns in the pokemon dataset. What these new columns allowed me to do was analyze the data more thoroughly. This new route of analysis meant I could look at the stat distributions as I had before but for specific pokemon types. For example here is the distribution of the defense stat for water pokemon:



Which if you compare this figure to the defense distribution of the fire type you can see that water pokemon tend to have higher defense than fire pokemon.

Defense Distribution of Fire Type Pokemon



Moving forward I could finally look at some visualizations which helped me guess what my model would use to predict the winner of a pokemon battle.

Winrates by Type1

Type1: Grass 43.994778067885115 %

Type1: Fire 58.02808302808303 %

Type1: Water 46.784417064311654 %

Type1: Bug 43.05936073059361 %

Type1: Normal 53.876673830385194 %

Type1: Poison 42.993979200875756 %

Type1: Electric 63.03778526000748 %

Type1: Ground 53.689894574440736 %

Type1: Fairy 32.86843328684333 %

Type1: Fighting 46.67271627344223 %

Type1: Psychic 54.61748633879782 %

Type1: Rock 40.55388957488093 %

Type1: Ghost 48.02968270214944 %

Type1: Ice 44.07275089314713 %

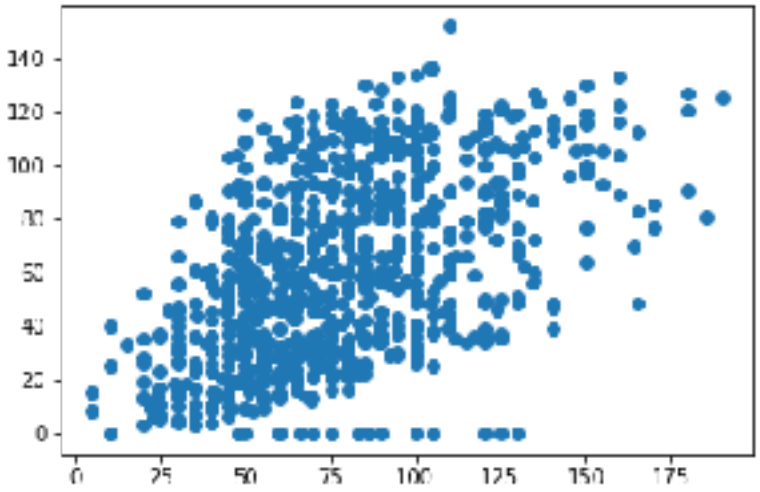
Type1: Dragon 63.326551373346895 %

Type1: Dark 63.64109232769831 %

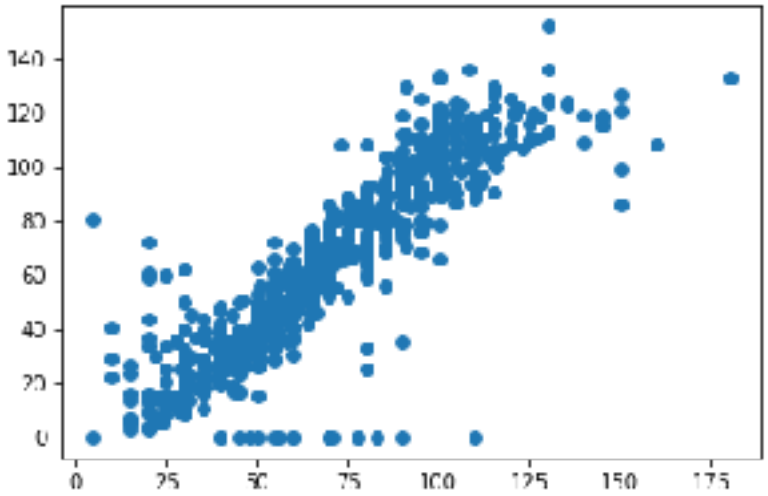
Type1: Steel 42.949802594472644 %

Type1: Flying 75.73221757322176 %

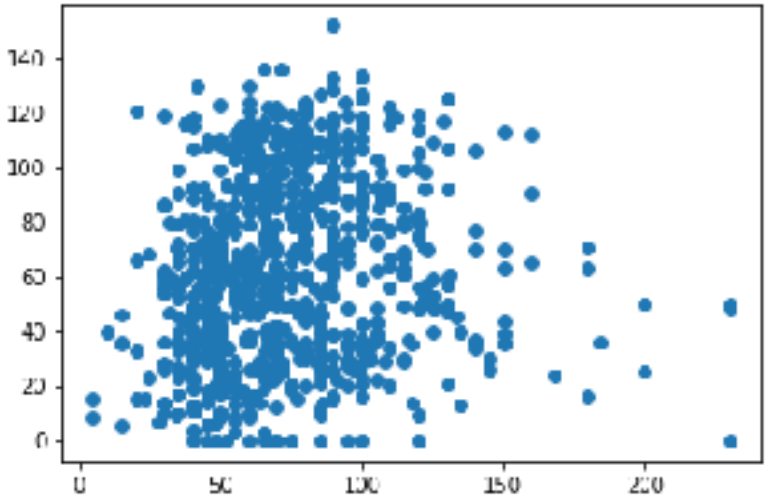
Correlation of the Attack stat to number of wins:



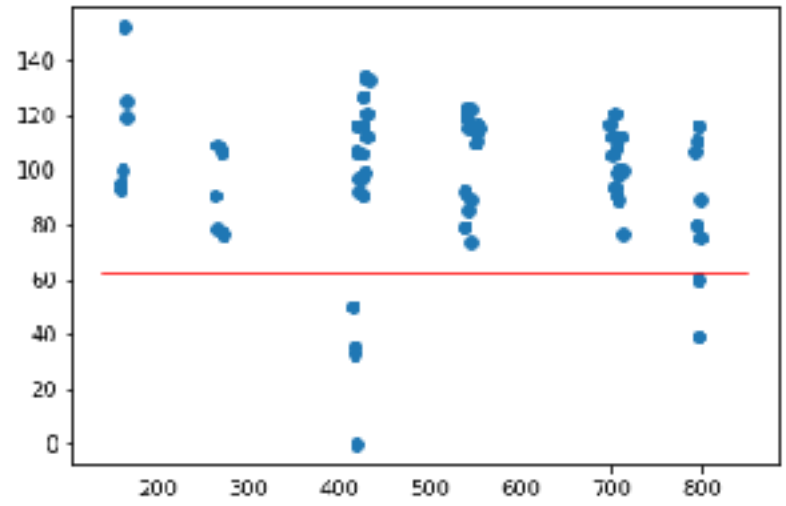
Correlation of the Speed stat to number of wins:



Correlation of the defense stat to number of wins:



Number of legendary pokemon with greater than average wins:



Winrates by Generation

gen1 21.026 %

gen2 11.932 %

gen3 19.416 %

gen4 16.32 %

gen5 21.324 %

gen6 9.982000000000001 %

After all this data analysis I felt prepared to work my dataset into a usable form for my neural network. In order to succeed in creating a neural network to predict the result of a pokemon battle I needed to put these two datasets together to create a dataset which contains the stats of two Pokemon and the result of their battle. In order to achieve this in the Jupiter notebook I took all the features of the pokemon dataset and put them in replacing the pokemon number which existed in the battle dataset. This meant that my final dataset looked like this:

	Winner	1#	2#	1Type1	2Type1	1Type2	2Type2	1HP	2HP	\
0	298.0	266.0	298.0	11.0	0.0	4.0	12.0	0.196078	70.0	

	1Attack	...	1SpAtk	2SpAtk	1SpDef	2SpDef	1Speed	2Speed	\
0	64.0	...	45.0	60.0	50.0	40.0	41.0	60.0	

	1Generation	2Generation	1Legendary	2Legendary
0	2.0	3.0	0.0	0.0

Taking this dataset I could then create a neural network to build a model that could predict the winning pokemon of a pokemon battle. The neural network I used had an input layer, 6 hidden layers of a width double that of the number of inputs, and an output layer which output one of the 800 pokemon. I found that the more epochs I had the higher my accuracy but the longer my model creation took. I also split the data randomly into training and testing. For this neural network I used 40% of my data to train and 60% to test. The most epochs I used was 20 and they gave me a 69% prediction rate. However I found an even better model when I normalized the combined dataset to only contain values between 0 and 1. This normalization caused the data to change to look like this:

	Winner	1#	2#	1Type1	2Type1	1Type2	2Type2	1HP	\
0	0.3725	0.3325	0.3725	0.647059	0.0	0.222222	0.666667	0.196078	

	2HP	1Attack	...	1SpAtk	2SpAtk	1SpDef	2SpDef	\
0	0.27451	0.336842	...	0.231959	0.309278	0.217391	0.173913	

	1Speed	2Speed	1Generation	2Generation	1Legendary	2Legendary
0	0.227778	0.333333	0.333333	0.5	0.0	0.0

Using this normalized dataset I found my neural network much more successful. In the end I just needed an input layer which had a node for each 22 features, a hidden layer of the same width as the input layer and an output layer which had an output for each pokemon. I also split the data into training and testing however this neural network was so powerful I was able to train the model on 10% of the data and achieve impressive prediction results for the other 90% of the data. The results of this simpler neural network built on the normalized data were extraordinary. I was able to achieve a successful prediction rate of 99.8%. This means that out of 1000 pokemon battles there would be 2 wrong results which is amazing.

https://www.kaggle.com/terminus7/pokemon-challenge#pokemon.csv