Method

The two questions we aim to investigate are: 1. How do average base stats and type distributions vary across generations? 2. Do CAP Pokemon exhibit distinctive stat distributions compared to Pokemon from Generations 1 to 9? The column in the obtained data frame provides us with the name of the Pokemon, the Pokemon's generation an indication of CAP, and the Pokemon's attributes.

Model 1 Linear Regression: We utilized linear regression to examine varying average base stats and types of Pokemon across generations. Our initial approach was to identify the independent and dependent variables. Since the average base stat and type of Pokemon varies, while generation stays constant within the range of one to nine, we decided to utilize the linear regression model to establish a relationship between the stat and type (dependent variable Y) and the generations (independent variable X). Hence the equation we used is:

$$Y = b0 + b1x1 + b2x2 + \ldots + Bnxn$$

where

y is the predicted value of the dependent variable, which is the stat and type of Pokemon

b0 to bn are the coefficients representing the influence of each predictor on y,

x0 to xn are the independent variables, which is the generation of Pokemon.

In matrix form, this becomes:

$$y = Xb$$

where

X is the matrix containing rows for each Pokemon and columns for the predictors

b is the coefficient vector that represents the weights for each predictor

y is the predicted outcome vector, which is the predicted base stat total for each Pokemon

We determined that the linear regression suits the question we aim to address because it quantifies the impact of predictor generation on the stats and type of Pokemon. This makes the relationship between the two variables interpretable for further analysis. However, the assumption of linearity between the stat of Pokemon and generation is a potential pitfall. There

may exist the possibility of not capturing the entirety of the data when non-linear relationships between the variables are introduced.

Model 2 Perceptron: To examine whether CAP Pokemon exhibit distinctive stat distributions compared to Pokemon from Generations 1 to 9, we utilized Perceptron as our second model. Even during the initial data collection phase, we utilized boolean flags to distinguish between CAP Pokemons with others. This inspired us to implement the perceptron, which is a linear classifier designed to separate the two classes CAP Pokemon and non-CAP Pokemon. It learns the decision boundary to classify each Pokemon based on its stats, polynomial expansions, and type indicators. For a given Pokemon, the decision function is:

$$f(x) = wx + b$$

where

w is the weight vector

x is the base stat, type indicators, and polynomial expansions

b is the bias term

The perceptron also predicts the Pokemons class by a piecewise function:

$$y = 1 \ if \ f(x) > 0 \ (CAP \ Pokemon)$$

$$y = -1 \ if \ f(x) \leq 0 \ (non - CAP \ Pokemon)$$

During training, the model updates its weights using gradient descent to minimize classification errors, also known as hinge loss. This process allows us to find the percent boundary between the two classes, and by examining features like the base stat and type of Pokemon, the perception allows us to identify patterns that distinguish CAP Pokemon from generational Pokemon. The assumption behind the implementation of the perceptron is the assumption of linear separation, that a weight vector exists that can separate CAP Pokemon from regular Pokemon. However, if the stat layouts of CAP Pokemon are too similar to regular Pokemon, any weight vector created from this model will likely also misclassify many regular Pokemon when trying to predict for CAP Pokemon or vice versa. Thus, in that situation, this assumption will be violated and act as a pitfall of this model.

Results

Model 1: Linear Regression

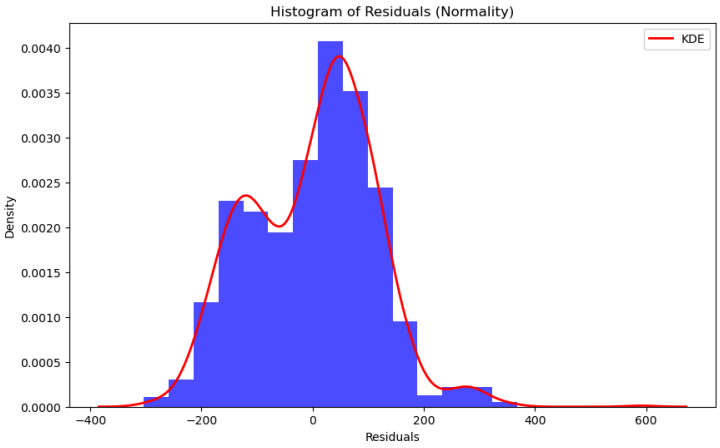

Figure 3. Histogram of Residuals

Figure 3 is the histogram of residuals from the linear regression. The x-axis represents the residuals, a representation of predicted and actual base stat values of generational Pokemon. The y-axis represents the density of residual occurrences and the frequency of occurrence of residuals of a particular magnitude. The model shows that the residuals are approximately normally distributed, as indicated by the bell-shaped curve, with a peak near 0. This suggests that the model often predicts Pokemon's base stats close to the true values. However, a slight right skew and the presence of outliers on both extremes indicate areas where the model overestimates or underestimates the stats, particularly for Pokémon with unusual characteristics. Again, we assume that this occurred due to the presence of a unique Pokemon. The red KDE curve aligns closely with the histogram bars in the center but diverges at the tails, confirming minor deviations from normality. These results suggest that the model reasonably satisfies the normality assumption but could benefit from handling the outliers and addressing skewness to improve accuracy.

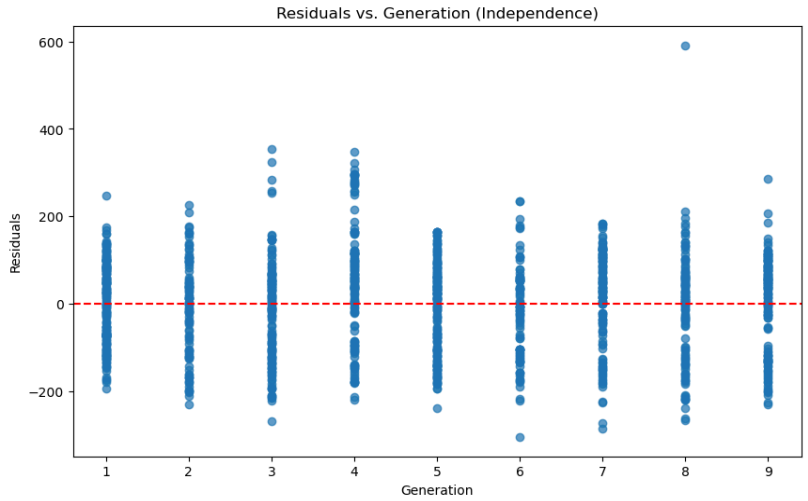

Figure 4. Residual plot

Figure 4 is the residual plot derived from the linear regression model that shows the relationship between residuals and generation, and in turn, provides information regarding the independence of our derived model. We saw that the residuals are fairly spread across all generations, with no clear trend or systematic structure. This suggests that the linear regression model reasonably meets the assumption of independence and homoscedasticity, which also means that Pokemon stats across generations are correctly reflected. However, some notable outliers do exist where some Pokémon have stats that deviate significantly from what the model predicts. We concluded that this result may be caused due to the presence of Pokemon that show exceptionally high stat distributions such as Legendary Pokemon, which are unique in the realm of Pokemon Showdown.

Model 2: Perceptron

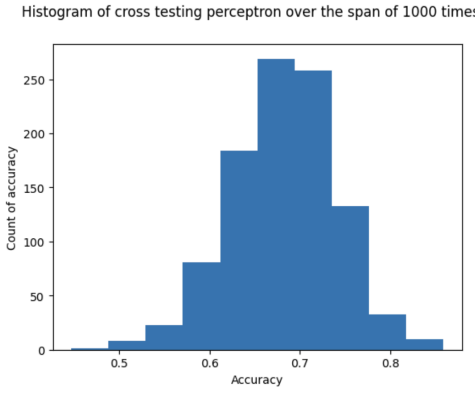

Figure 5. Histogram of cross testing perceptron

The histogram in Figure 5 shows the accuracy distribution of the perceptron model during cross-validation testing, with most accuracy values clustering between 50%-80%. This shows that by using the perceptron, we were loosely able to distinguish between CAP Pokémon and non-CAP Pokémon using features like base stats, polynomial expansions, and type indicators. However, since there are 1245 regular Pokemon and 75 CAP Pokemon, a naive algorithm that only predicts regular Pokemon would have a higher 94% accuracy rate.

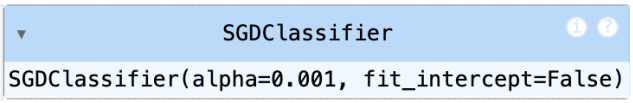

Figure 7. SDGClassifier

Figure 5 is the SGDClassifier configuration used in this project to apply a Perceptron model for Pokémon classification. The model's hyperparameters include fit_intercept=False, which assumes the dataset already added an intercept column and prevents an extra one from being added, and alpha=0.01, which regulates regularization to avoid overfitting. Through iterative weight updates using stochastic gradient descent, the SGDClassifier creates a weight vector that allows our Perceptron model to predict whether certain Pokemon are CAP.