Predicting Pokémon Types with Clustering and Classification

Justin Zhang, Isaac Baguisa, Alex Faassen

2025-04-09

Introduction

Motivation: Nostalgia, popularity, and an interesting, numerically-structured game design.

Pokémon Typing: Game mechanic (i.e. pairwise interactions), but also a conceptual grouping based on traits like colour, strength, and theme.

Can clustering and classification methods uncover or predict a Pokémon's type based on its image and statistical features?

Example: Pikachu and Charizard

Pikachu (Primary Type: Electric)

Charizard (Primary Type: Fire)

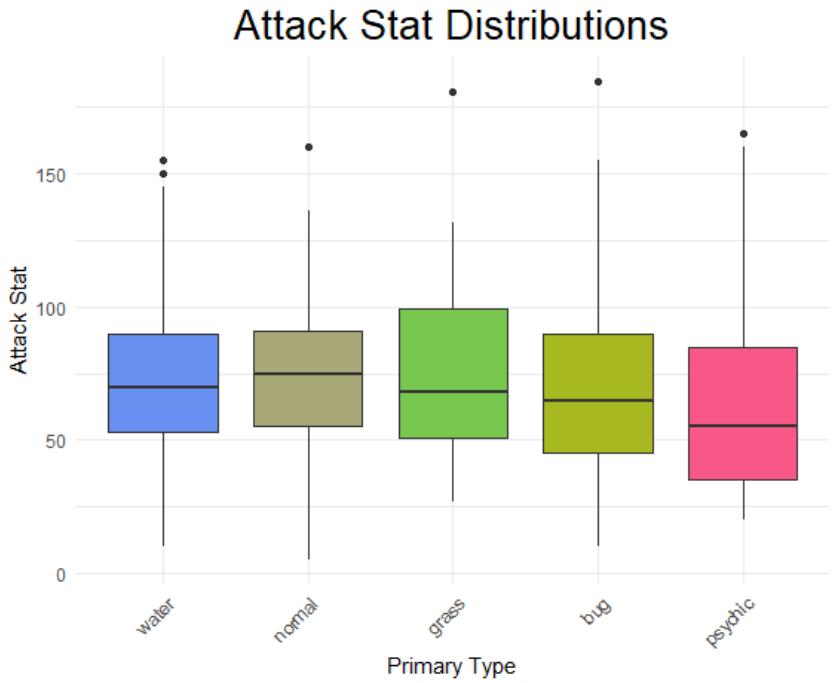
Data Description

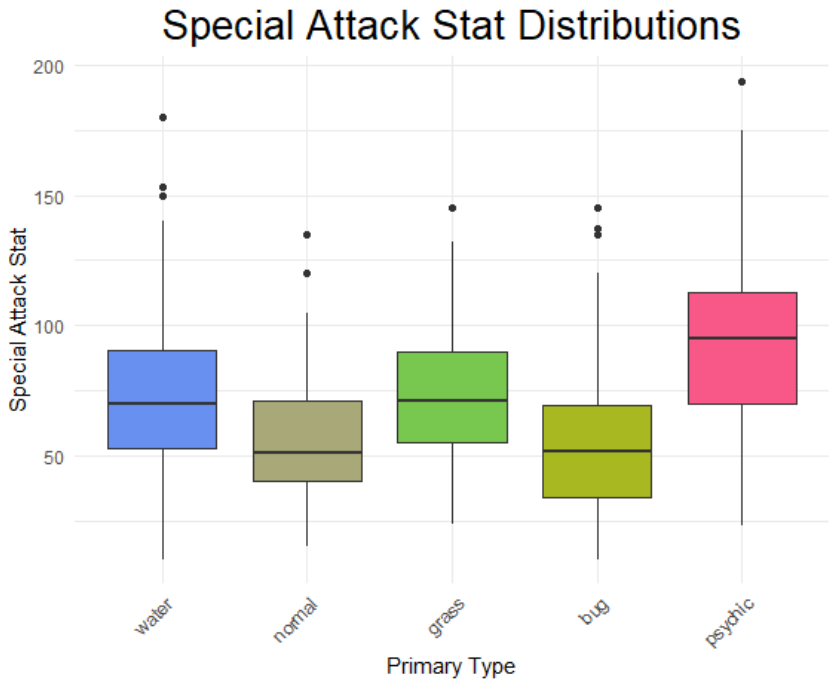
From generations 1-7:

1. **Image Dataset:** 809 Pokémon, 120 x 120 PNGs with RGBA
2. **Stats Dataset:** 801 Pokémon, 41 numerical features (e.g. `hp`, `attack`, `sp_defense`, etc.)

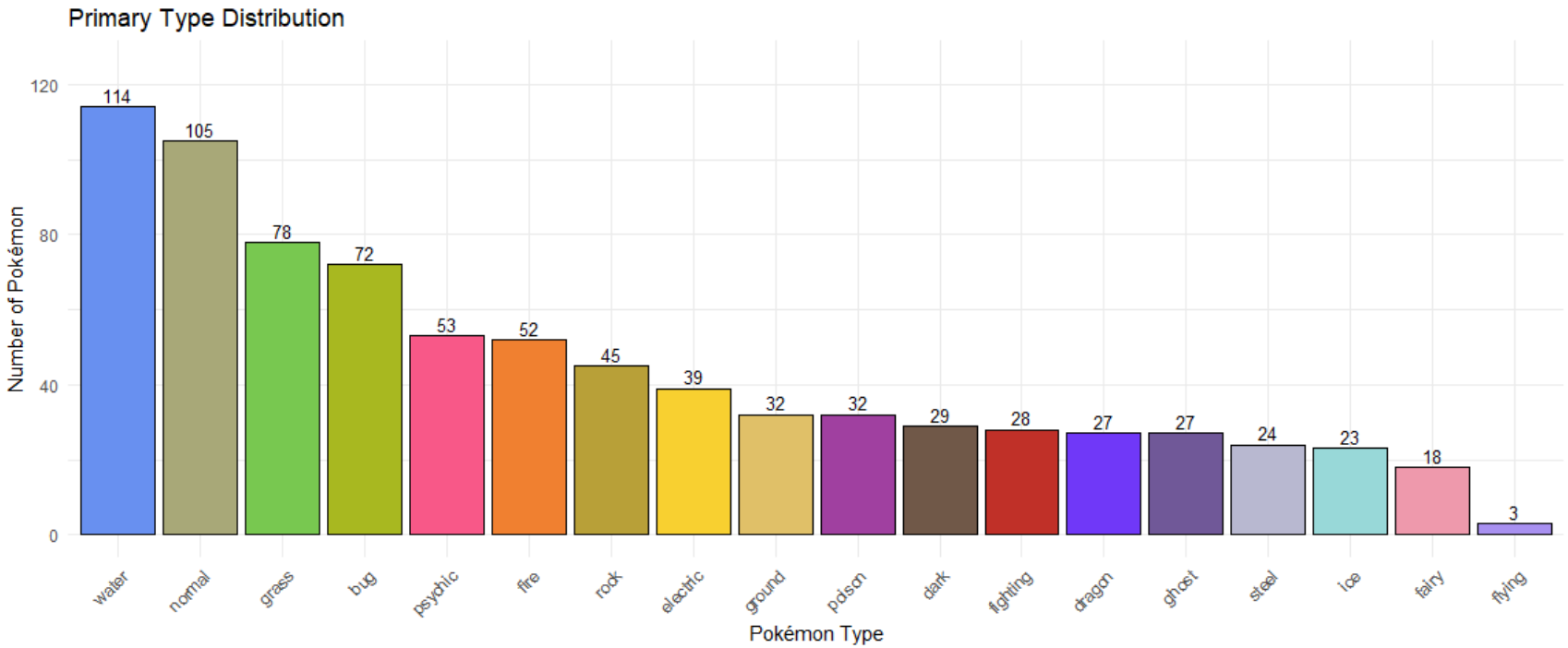
Pre-processing: Flattened RGB image vectors (43,200 features); matched and cleaned datasets (801 shared Pokémon).

Stats


Attack Stat Distributions
Attack Stat
150
100
50
0
water
normal
grass
bug
psychic
Primary Type


Special Attack Stat Distributions
Special Attack Stat
200
150
100
50
water
normal
grass
bug
psychic
Primary Type

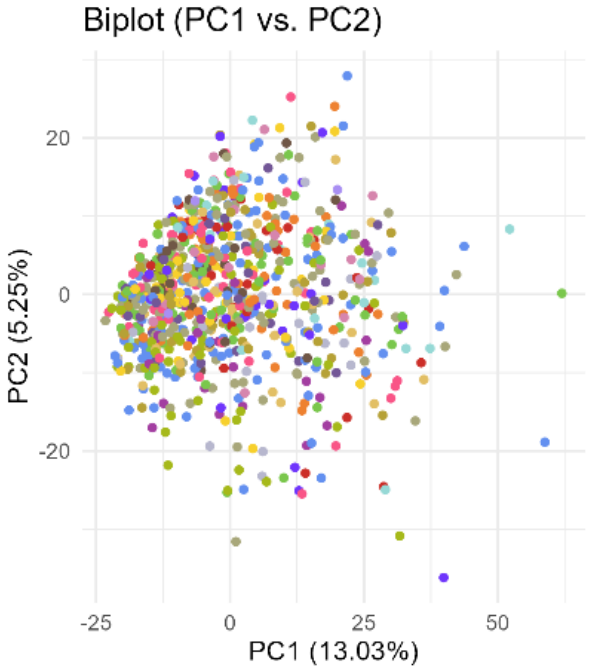
Primary Type Distribution

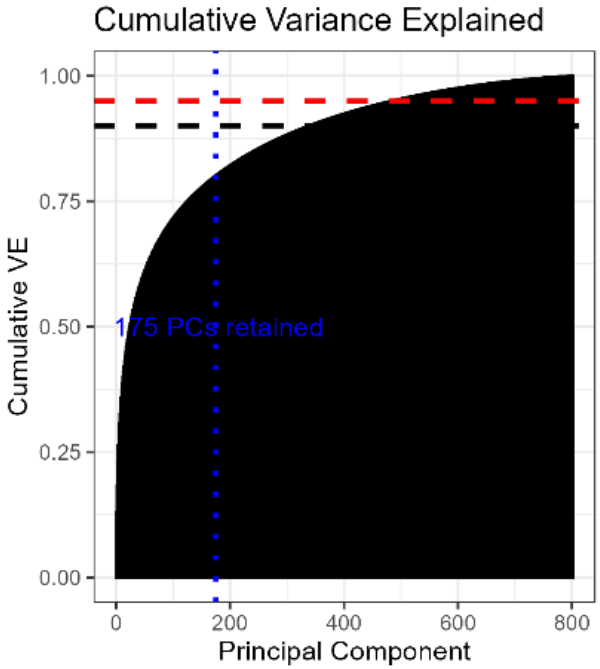


Methods

- **Dimension reduction:** PCA
  - Images and Stats
- **Clustering:** k-means
- **Classification:** LDA, Gradient boosting
  - For comparison

Image Dimension Reduction: PCA



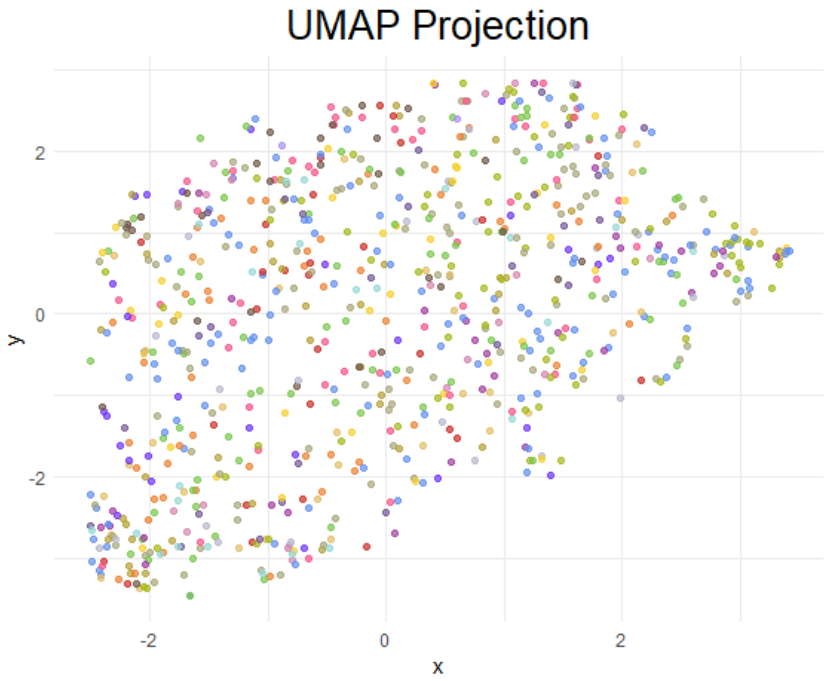


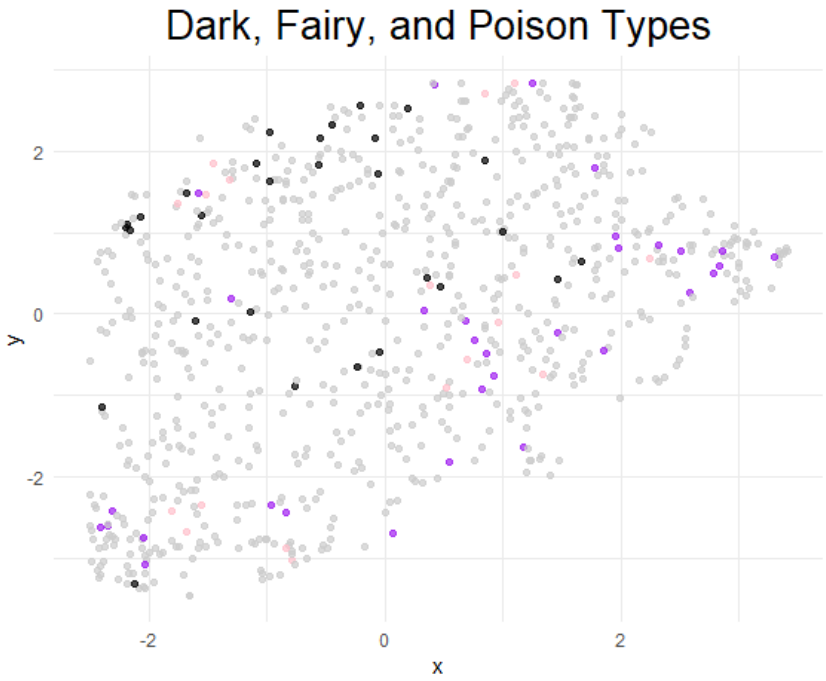
Example: Image Compression

Original: Abomasnow

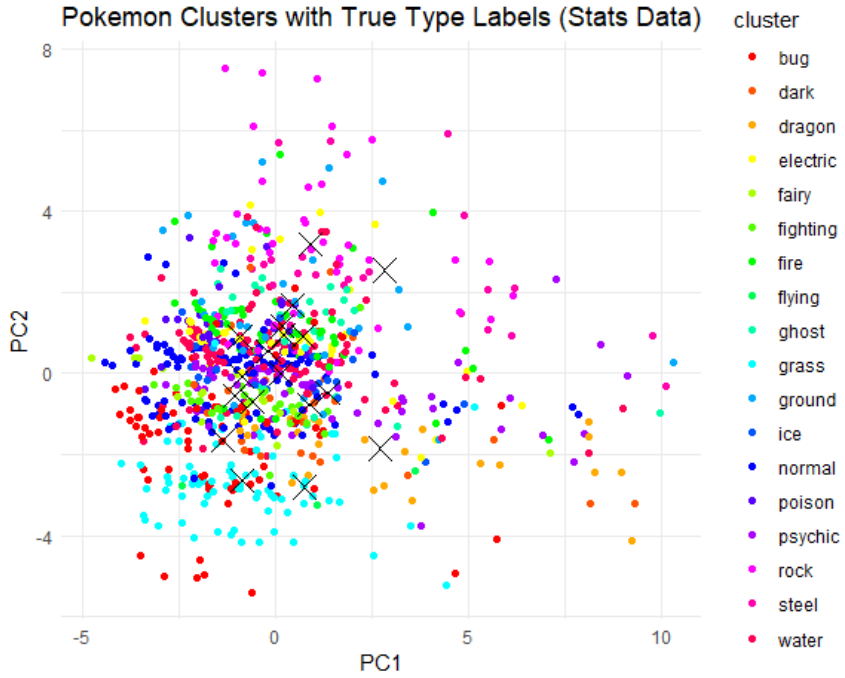
PCA Reconstructed (175 PCs)

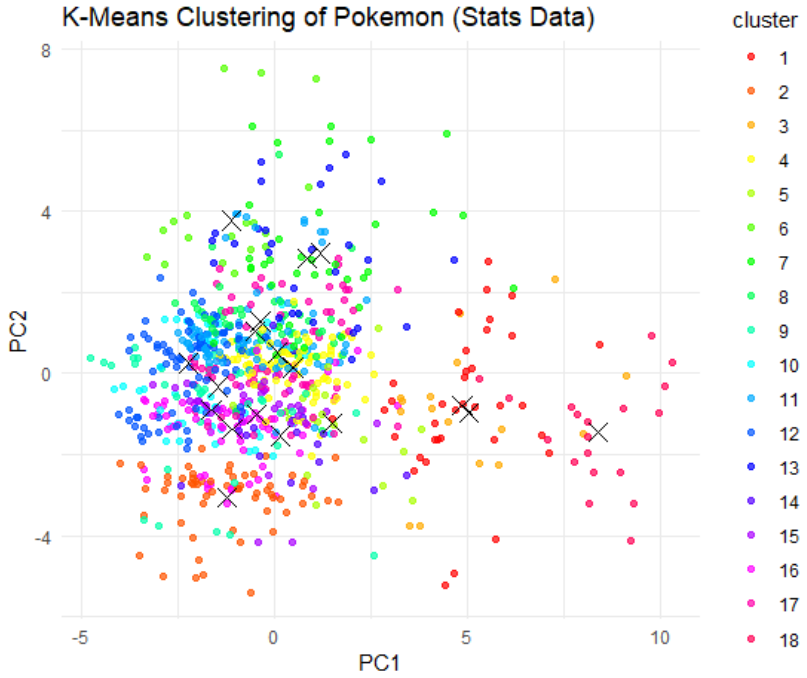
Image Dimension Reduction: UMAP





K-Means Clustering





Clustering Accuracies

Method	Accuracy_Stats	Accuracy_Image
K-means	0.5243446	0.1810237
K-means++	0.5293383	0.1747815
Weighted K-means	0.2034956	0.1423221
Optimal K-means K = 3	0.1722846	0.1473159

Clustering Accuracy Comparison: Stats vs Image Data

Cluster Assignments for Stats Data

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
bug	3	6	0	3	0	0	4	2	2	12	1	19	2	0	1	16	1	0
dark	1	0	0	10	0	0	2	1	0	0	0	4	0	5	0	3	1	2
dragon	0	0	2	0	9	0	0	0	0	0	0	0	0	11	0	0	0	5
electric	5	0	0	9	0	1	4	2	1	0	0	15	0	0	0	1	1	0
fairy	1	0	0	0	0	0	0	0	17	0	0	0	0	0	0	0	0	0
fighting	0	0	0	0	0	0	1	0	0	0	0	1	0	1	25	0	0	0
fire	3	0	0	2	1	2	1	38	0	0	0	0	0	0	0	4	0	1
flying	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
ghost	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26	1
grass	2	50	1	0	0	1	2	0	5	14	0	0	0	0	2	0	1	0
ground	1	0	0	2	4	5	1	0	0	0	0	8	6	0	0	0	4	1
ice	2	0	0	7	0	0	0	0	0	0	1	6	3	0	0	1	3	0
normal	2	2	1	38	1	4	0	0	4	0	1	22	0	0	2	26	0	2
poison	0	0	0	2	1	2	0	3	0	17	3	1	0	0	1	2	0	0
psychic	4	0	11	0	0	0	0	0	5	0	0	0	0	0	1	0	30	2
rock	4	0	0	0	1	4	3	0	0	0	6	2	18	1	0	4	2	0
steel	3	0	1	0	0	0	17	0	0	0	0	0	0	0	0	0	1	2
water	4	2	0	4	1	0	1	0	3	2	76	7	0	0	1	7	3	3

Cluster Assignments for Image Data

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
bug	0	9	1	4	2	0	5	7	1	0	3	3	2	2	3	24	6	0
dark	0	1	0	9	2	0	2	3	2	0	0	0	0	0	1	6	2	1
dragon	0	1	0	8	4	2	2	1	1	0	1	1	2	0	0	1	1	2
electric	0	2	0	6	1	0	7	0	1	1	2	3	0	1	1	5	6	3
fairy	0	0	0	1	3	0	1	1	0	0	1	5	0	1	0	1	4	0
fighting	0	2	1	1	4	0	6	1	1	0	1	3	2	2	0	1	1	2
fire	0	1	1	1	6	0	6	7	0	3	6	1	1	2	3	8	3	3
flying	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1
ghost	1	1	0	0	1	0	3	3	2	1	1	1	0	1	0	9	2	1
grass	0	6	3	8	7	0	10	3	0	2	6	1	2	3	0	13	10	4
ground	1	6	0	3	1	0	1	2	0	2	3	1	1	2	2	4	2	1
ice	1	1	0	0	3	0	2	0	0	1	1	1	1	5	1	1	5	0
normal	0	10	3	6	9	0	10	2	1	1	4	9	1	5	7	17	16	4
poison	0	3	0	4	1	0	1	0	1	2	1	0	1	4	0	6	8	0
psychic	1	1	1	2	5	1	11	4	1	1	2	2	1	0	1	10	7	2
rock	0	2	1	3	4	0	5	2	2	2	4	3	0	3	1	8	4	1
steel	1	2	1	1	1	0	1	0	2	1	2	1	2	1	1	6	0	1
water	3	14	1	4	14	0	10	2	0	3	8	5	3	5	1	21	15	5

Clustering Takeaways

- Stats cluster primary types better than images
  - at best clustering classified 52% of Pokemon type
- Limitations include imbalanced data among Pokemon types, which can be accounted for in Supervised Models
- Structure of the data may not capture the complexity of the Pokemon types in a way that creates efficient and distinct clusters

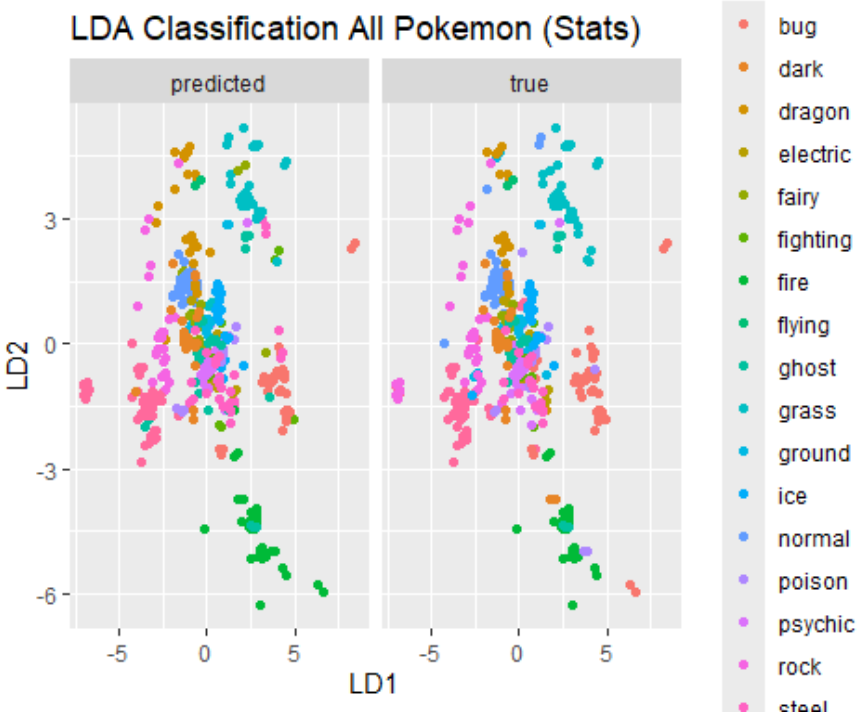
Supervised Model

- Main purposes:
  - Baseline accuracy measurement for clustering analysis
  - Determine whether there are distinguishable features for each type
- 2 models:
  - Linear Discriminant Analysis
  - Gradient Boosting

Methods

- Use PCA reduced datasets
- 80-20 train-test split, stratified for Pokemon types
- **Linear Discriminant Analysis:**
  - full analysis and rank-reduced DA for $L = 1, \ldots, 10$
  - Relatively robust to outliers
- **Gradient Boosting:**
  - Tuned for learning rate, max depth = 3, itertions = 100
  - Good with non-linear data

LDA: Full


LDA Classification All Pokemon (Stats)
predicted
true
LD2
LD1
3
0
-3
-6
-5
0
5
bug
dark
dragon
electric
fairy
fighting
fire
flying
ghost
grass
ground
ice
normal
poison
psychic
rock
steel

LDA Accuracy

L	accuracy_train	accuracy_test
1	0.4574132	0.4371257
2	0.6167192	0.5808383
3	0.7176656	0.6886228
4	0.8091483	0.7904192
5	0.8454259	0.8263473
6	0.8517350	0.8383234
7	0.8738170	0.8622754
8	0.8864353	0.8862275

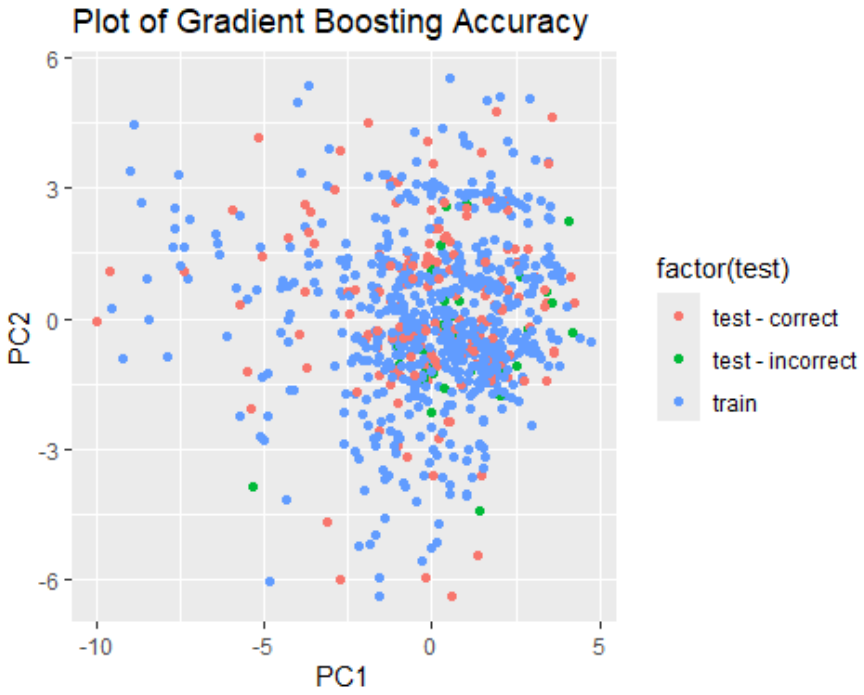
LDA Accuracy for Stats Data

LDA Accuracy: By Generation

	gen1-train	gen2-train	gen3-train	gen4-train	gen1-test	gen2-test	gen3-test	gen4-test
3	0.9824561	0.9459459	0.970297	0.9480519	0.9189189	0.6923077	0.7647059	0.5333333
4	0.9912281	1.0000000	0.960396	0.9870130	0.9189189	0.7692308	0.8235294	0.6000000
5	0.9912281	1.0000000	0.970297	0.9870130	0.9189189	0.8076923	0.8529412	0.6333333
6	0.9912281	1.0000000	0.970297	1.0000000	0.9189189	0.8076923	0.8235294	0.6333333

LDA Accuracy for Stats Data - by Generation

Gradient Boosting: Results

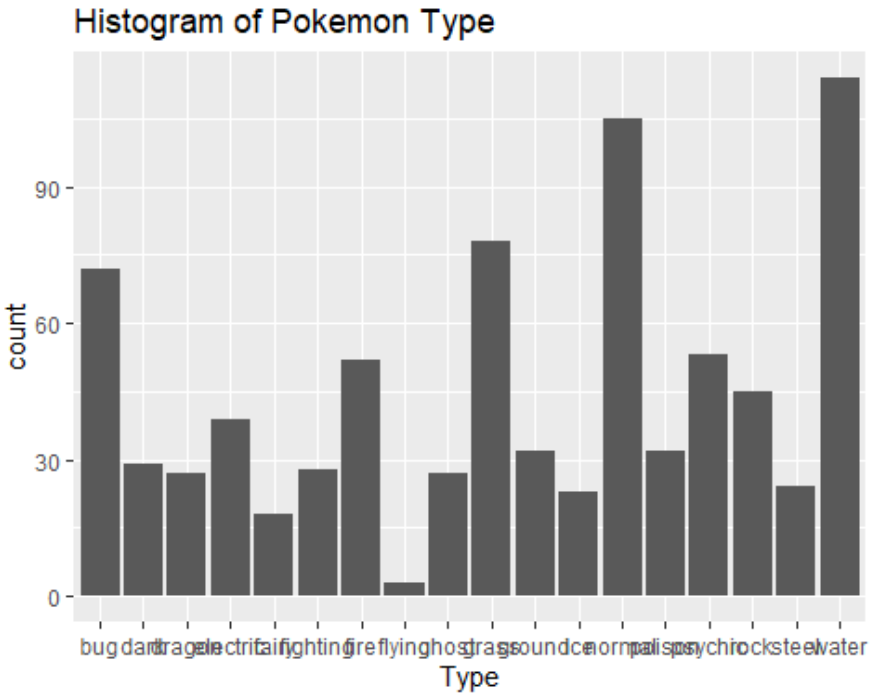


Gradient Boosting Accuracy

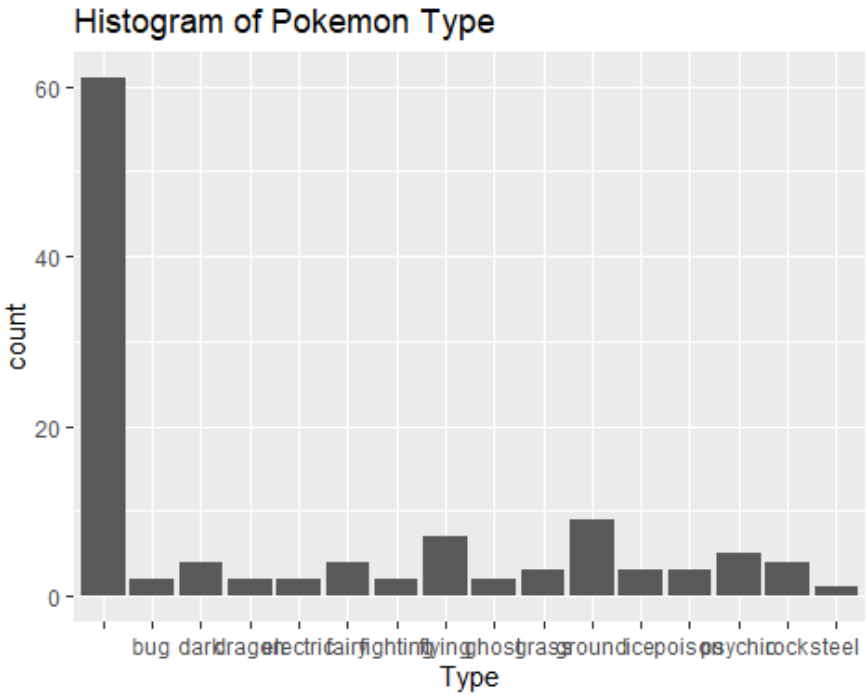
shrinkage	train_accuracy	test_accuracy
0.01	0.7271293	0.6047904
0.10	0.9968454	0.8143713
0.25	1.0000000	0.8383234

Gradient Boosting Accuracy

Limitation: Non-uniformity in Pokemon types



Limitation: Dual Typing



Biases

- Computation of classification accuracy
  - non-uniformity of Pokemon types
- Use of ‘against_(type)’ variables -inflates accuracy by ~20%

Recommendations

- Tackle the limitations and biases we mentioned
- Consider confounding variables with (ex. is_legendary)
- Use descriptional image stats rather then image pixels directly
- Cross validation on Pokemon generation
- Try other models - ex. GMM for clustering