HW#5

April 21, 2020

[1]:
```
import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
from plotnine import *

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import AgglomerativeClustering


from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture

from sklearn.metrics import silhouette_score

import scipy.cluster.hierarchy as sch
from matplotlib import pyplot as plt

%matplotlib inline
```

[2]:
```
# Name: Name of each pokemon

# Type 1: Each pokemon has a type, this determines weakness/resistance to␣
↪attacks

# Type 2: Some pokemon are dual type and have 2

# Total: sum of all stats that come after this, a general guide to how strong a␣
↪pokemon is

# HP: hit points, or health, defines how much damage a pokemon can withstand␣
↪before fainting

# Attack: the base modifier for normal attacks (eg. Scratch, Punch)

```

```
# Defense: the base damage resistance against normal attacks

# SP Atk: special attack, the base modifier for special attacks (e.g. fire␣
↪blast, bubble beam)

# SP Def: the base damage resistance against special attacks

# Speed: determines which pokemon attacks first each round
```

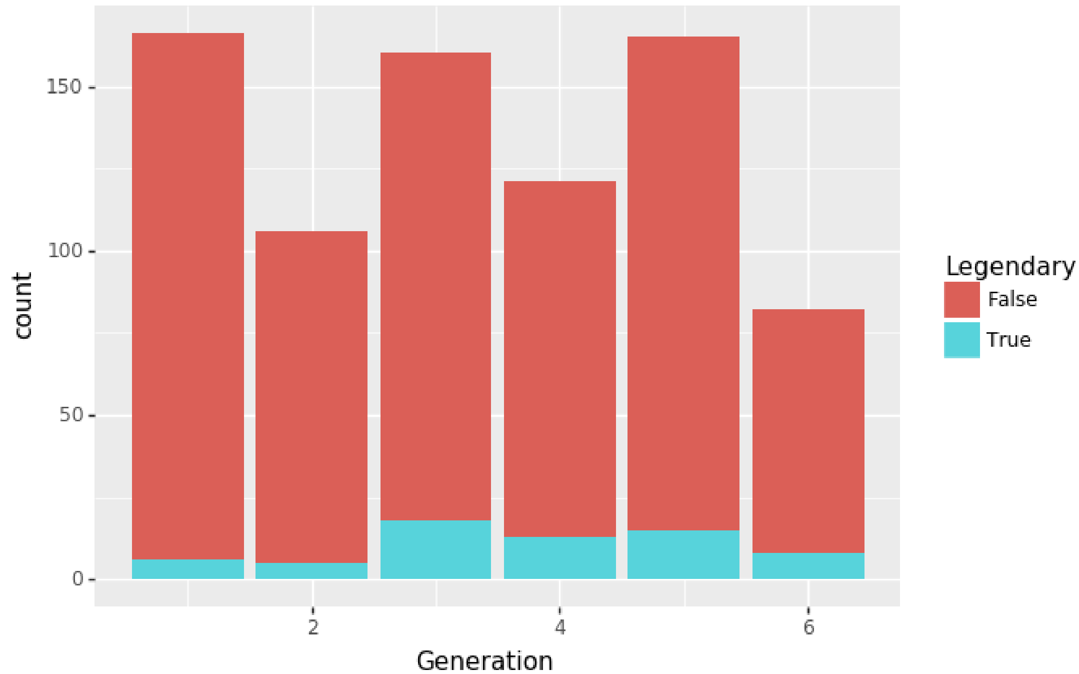
[3]:
```
poke = pd.read_csv("https://raw.githubusercontent.com/cmparlettpelleriti/
↪CPSC392ParlettPelleriti/master/Data/Pokemon.csv")
poke.head()
```

[3]:
```
                    Name Type 1  Type 2  Total  HP  Attack  Defense  Sp. Atk  \
0              Bulbasaur  Grass  Poison    318  45      49       49       65
1                Ivysaur  Grass  Poison    405  60      62       63       80
2               Venusaur  Grass  Poison    525  80      82       83      100
3  VenusaurMega Venusaur  Grass  Poison    625  80     100      123      122
4             Charmander   Fire     NaN    309  39      52       43       60

   Sp. Def  Speed  Generation  Legendary
0       65     45           1      False
1       80     60           1      False
2      100     80           1      False
3      120     80           1      False
4       50     65           1      False
```

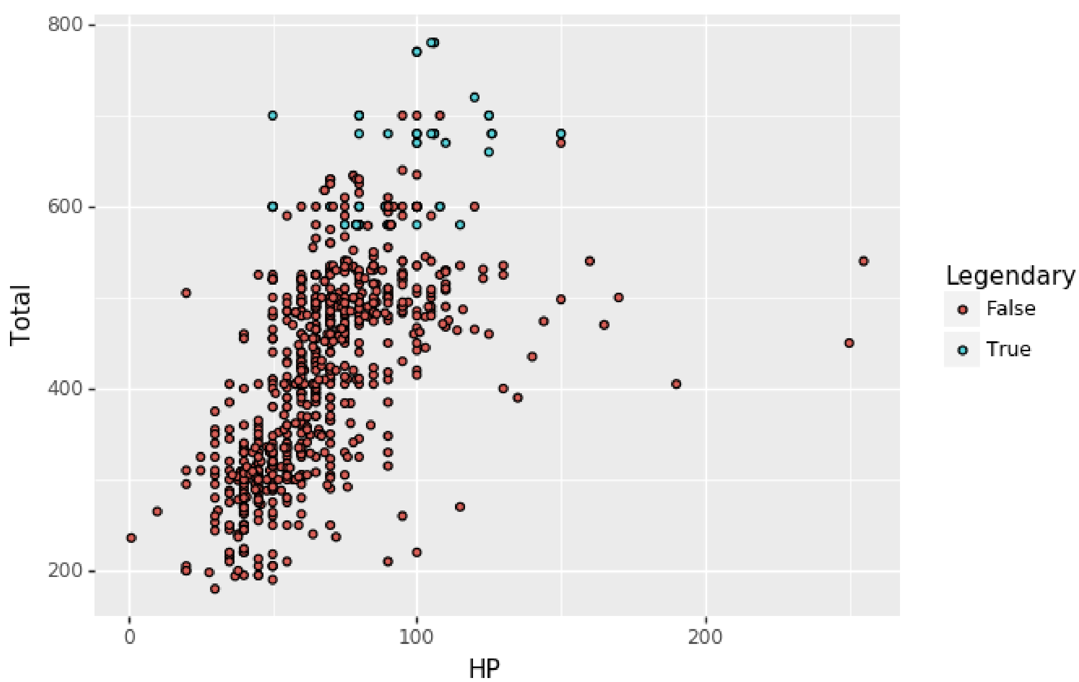
1 1. Explore Data

[4]:
```
ggplot(poke, aes("Generation", fill = "Legendary")) + geom_bar()
```



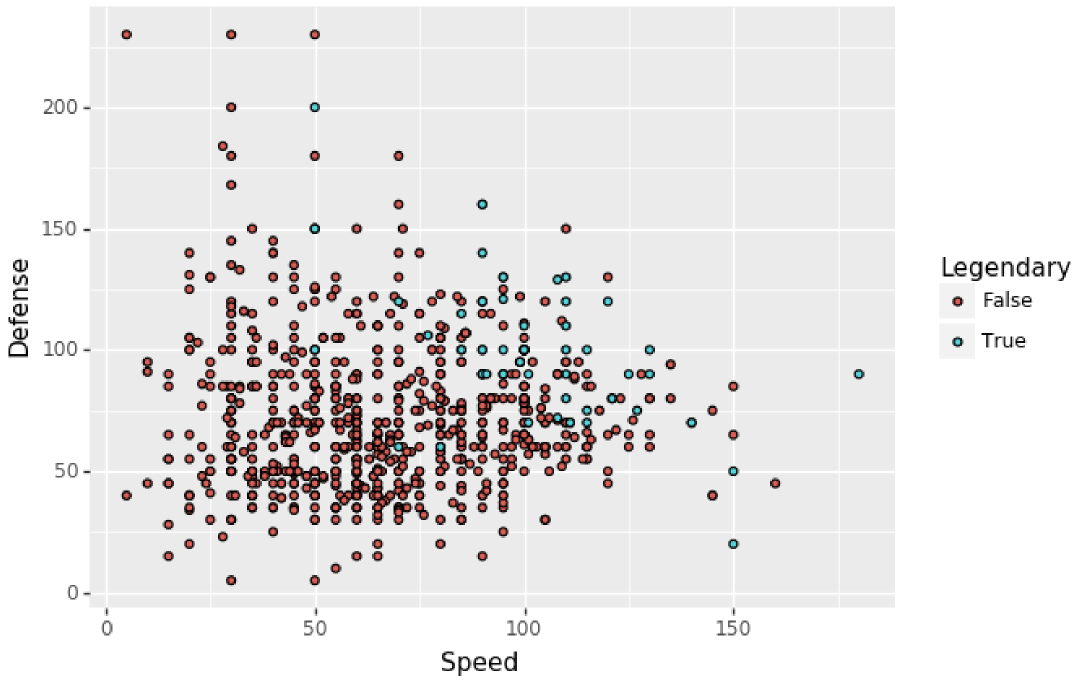
[4]: <ggplot: (317225441)>

[5]:
```
ggplot(poke, aes("HP", "Total")) + geom_point(aes(fill = "Legendary"))
```



[5]: <ggplot: (316648725)>

[6]:
```
ggplot(poke, aes("Speed", "Defense")) + geom_point(aes(fill = "Legendary"))
```



[6]: <ggplot: (317712945)>

I see that most of the legendary pokemons are equiped with the best attacks and defenses which makes sense.

2 2. K-means model

[7]:
```
features = ["Legendary","HP", "Total", "Speed","Attack", "Defense"]
X = poke[features]

z = StandardScaler()
X[features] = z.fit_transform(X)

km = KMeans(n_clusters = 5)

km.fit(X)

membership = km.predict(X)
```

```
X["cluster"] = membership

silhouette_score(X, membership)
```

[7]: 0.45044341226229606

which features did you use and why? - I used “Legendary”,“HP”, “Total”, “Speed”,“Attack”, and “Defense” to member the pokemon because I felt theses were important variables to identify pokemon

did you standardize your variables? why or why not? - yes I did because not all variables were not measured on the same scale.

which k works best? what metrics did you use to determine this? - I ran the model several times with different k values and found that between 4 and 5 n_clusters was the best amount for maximizing our silhouette score

3 3. Gaussian Mixture model

[8]:
```
features = ["Legendary","HP", "Total", "Speed","Attack", "Defense"]

X = poke[features]
z = StandardScaler()
X[features] = z.fit_transform(X)
Xdf = X


n_components = [2,3,4,5,6]

sils = []
for n in n_components:
    gmm = GaussianMixture(n_components = n)
    gmm.fit(X)
    colName = str(n) + "assign"
    clusters = gmm.predict(X)

    Xdf[colName] = clusters

    sils.append(silhouette_score(X, clusters))

print(sils)
```

```
[0.5029335973278094, 0.5061383802739656, 0.4911189789750171, 0.5033830610048823,
0.5392913667951059]
```

which features did you use and why? - I used “Legendary”,“HP”, “Total”, “Speed”,“Attack”, and “Defense” to member the pokemon because I felt theses were important variables to identify pokemon

did you standardize your variables? why or why not? - yes I did because not all variables were not measured on the same scale.

which number of components works best? what metrics did you use to determine this? - After running the model for 2,3,4,5,and 6 num pf components, the silhouette score showed that 5 components was the ideal amount of components.

4 4. Hierarchical model

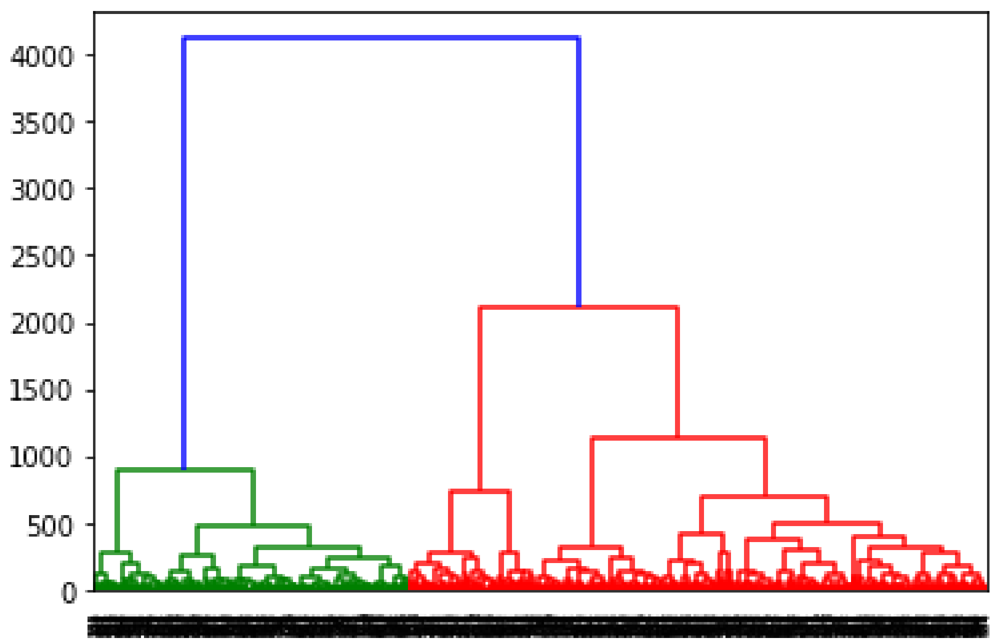
[9]:
```
features = ["Legendary","HP", "Total", "Speed","Attack", "Defense"]

X = poke[features]

hac = AgglomerativeClustering(affinity = "euclidean",
                              linkage = "ward")

hac.fit(X)

dendro = sch.dendrogram(sch.linkage(X, method='ward'))
```



[10]:
```
hac = AgglomerativeClustering(n_clusters =2,
                               affinity = "euclidean",
                              linkage = "ward")
hac.fit(X)
```

[10]:

```
AgglomerativeClustering(affinity='euclidean', compute_full_tree='auto',
                        connectivity=None, distance_threshold=None,
                        linkage='ward', memory=None, n_clusters=2)
```

[11]:

```
membershiph = hac.labels_
membershiph
```

[11]:

```
array([1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1,
       0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,
       0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0,
       1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1,
       0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1,
       0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,
       0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0,
       1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0,
       1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,
       0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0,
       0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0])
```

[12]:

```
silhouette_score(X,membershiph)
```

[12]: 0.5002392714696835

which features did you use and why? - I used “Legendary”,“HP”, “Total”, “Speed”,“Attack”, and “Defense” to member the pokemon because I felt theses were important variables to identify pokemon

did you standardize your variables? why or why not? - yes I did because not all variables were not measured on the same scale.

which number of clusters works best? what metrics did you use to determine this? - I used 2 clusters because it produced the best silhouette score, also this amount of clusters is the least computationally expensive.

5 5. Compare the results from each model

Were the clusters created by each method similar? - Yes I would say the models were very similar.

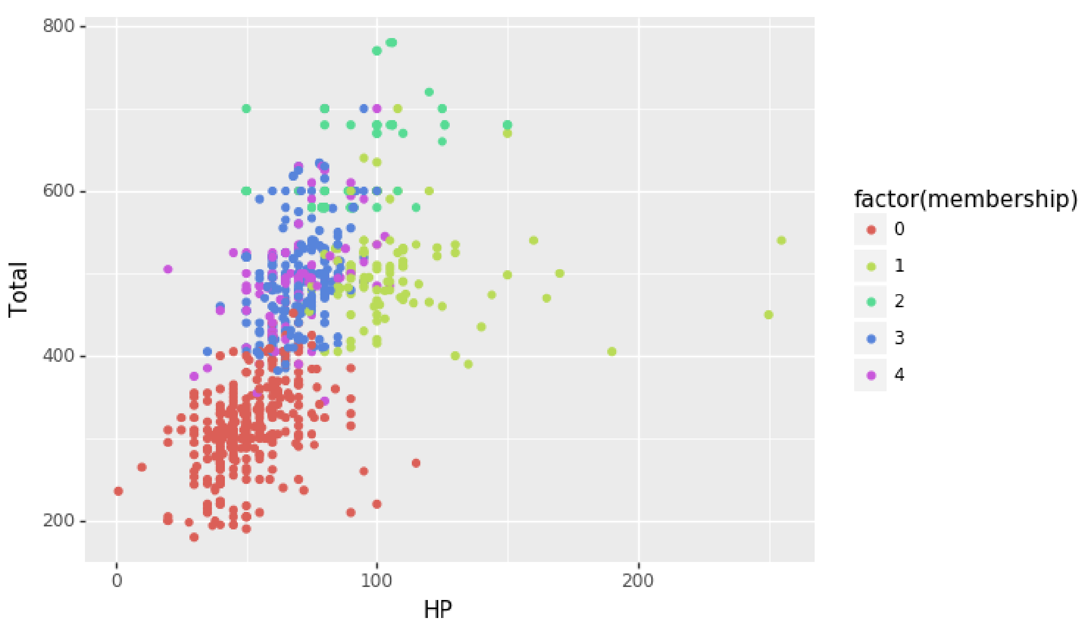
Describe the clusters from each method (in other words, are the mean values for features different between clusters? - I did not see a huge differenct in the mean values between clusters

How would you describe each cluster to someone who hasn’t seen the data before? e.g. “This cluster is very fast, and has a low attack….etc”) - I would describe these clusters by saying which were closer to legendary or not.

6 GGPLOT Per Method

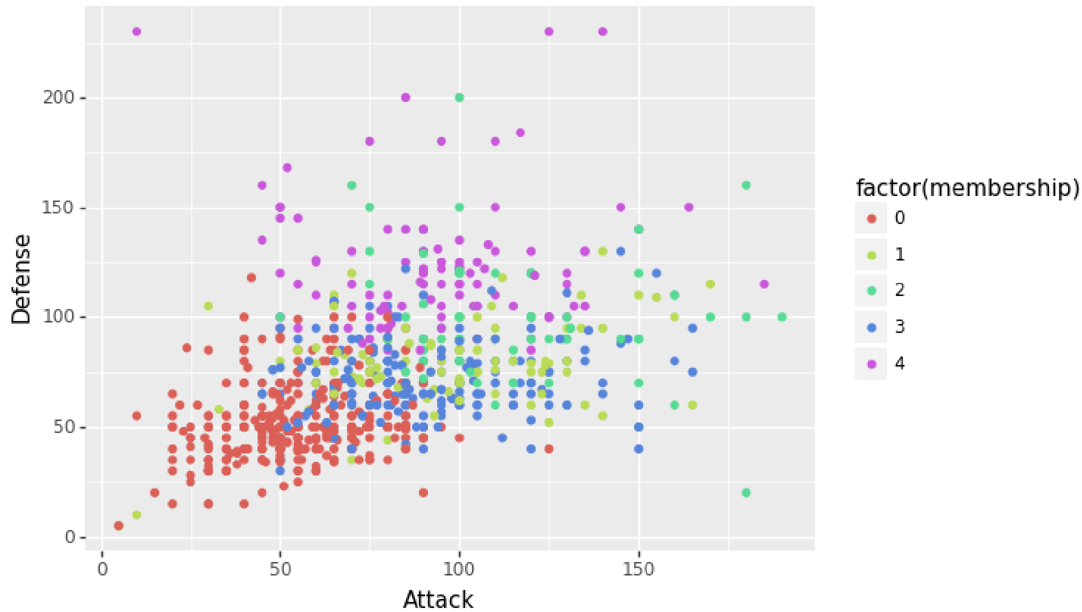
7 K-means model

[13]:
```
(ggplot(X, aes("HP", "Total", color = "factor(membership)")) + geom_point())
```



[13]: <ggplot: (317687565)>

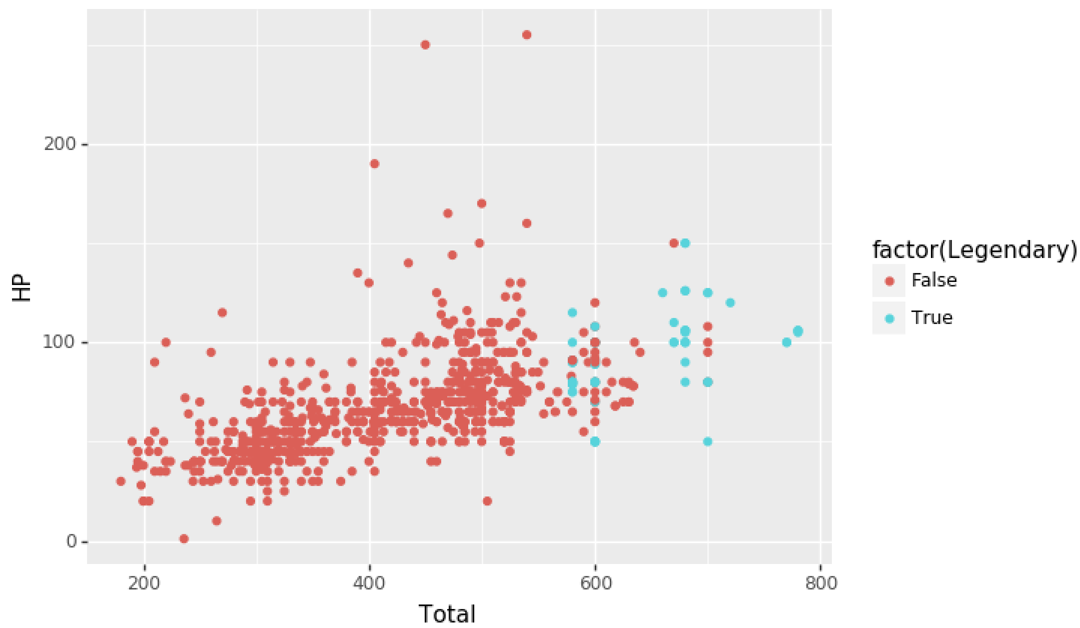
[14]:
```
(ggplot(X, aes("Attack", "Defense", color = "factor(membership)")) +␣
↪geom_point())
```



[14]: <ggplot: (316658233)>

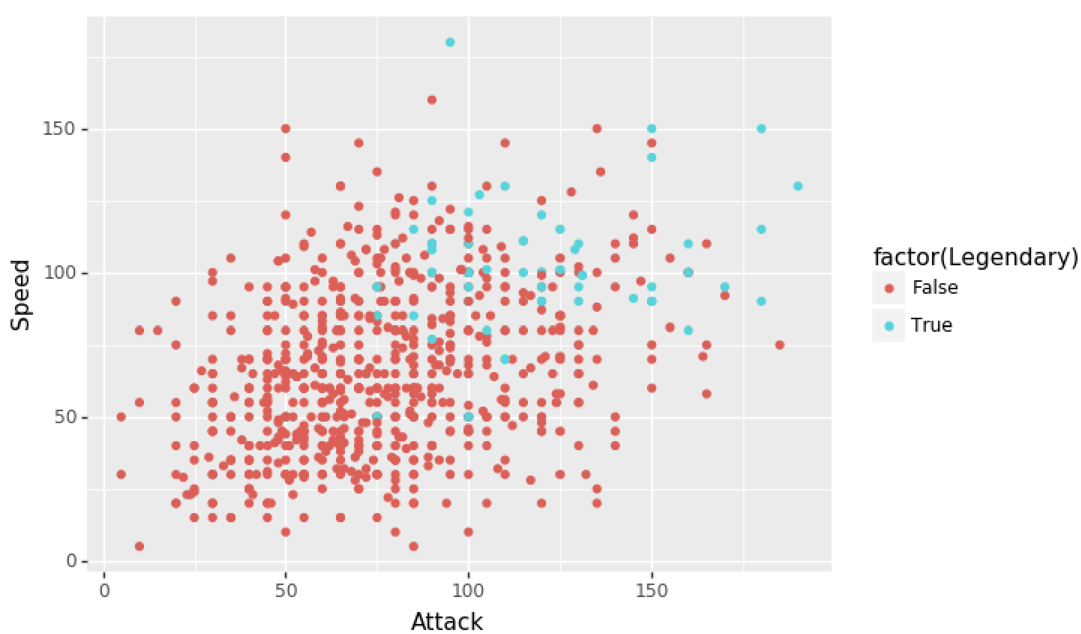
8 Gaussian Mixture model

[18]:
```
(ggplot(X, aes("Total","HP", color = "factor(Legendary)")) + geom_point())
```



[18]: <ggplot: (320273381)>

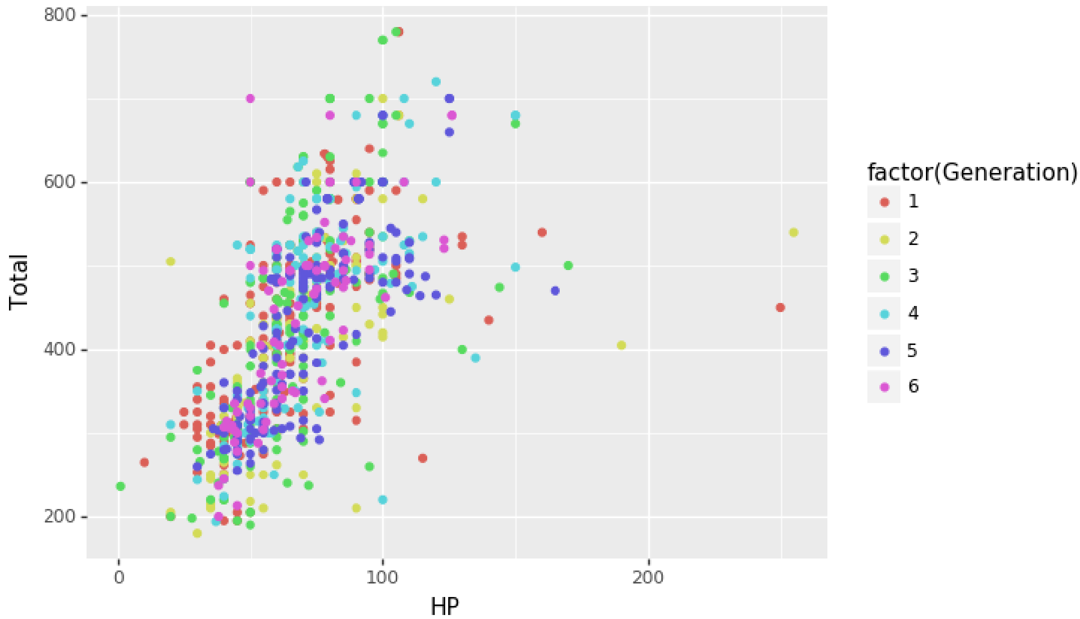
[19]:
```
(ggplot(X, aes("Attack","Speed", color = "factor(Legendary)")) + geom_point())
```



[19]: <ggplot: (320297417)>

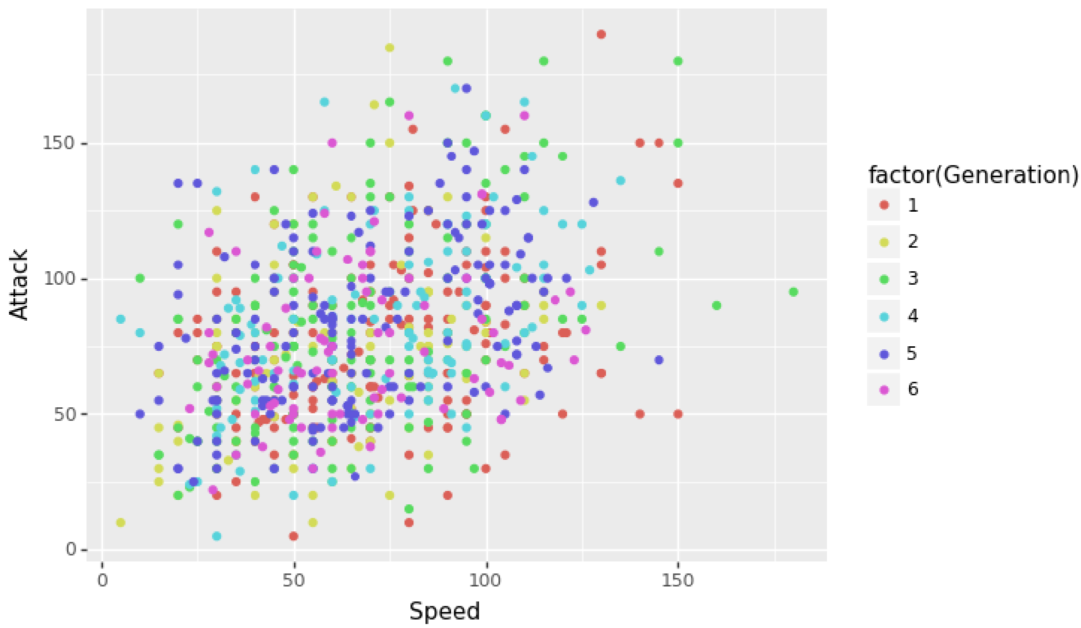
9 Hierarchical model

[20]:
```
(ggplot(poke, aes("HP", "Total")) + geom_point(aes(color =␣
↪"factor(Generation)")))
```



[20]: <ggplot: (320173553)>

[21]:
```
(ggplot(poke, aes("Speed", "Attack")) + geom_point(aes(color =␣
↪"factor(Generation)")))
```




```
[21]: <ggplot: (320297577)>
```

```
[ ]:
```

```
[ ]:
```