In [3]:
```
### Written by vapicuno, May 8, 2023
### Viability ranking analysis of 2023 ADV OU (Spring) metagame
### VR post https://www.smogon.com/forums/threads/adv-ou-viability-ranking.3503019/page-30

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.transforms as transforms
import matplotlib.patches as patches
import matplotlib.cm as cm
import matplotlib.colors as colors
%matplotlib inline
import pandas as pd
import itertools
from timeit import default_timer as timer
from scipy import stats
from scipy import cluster
import pandas as pd
import copy
```

In [45]:
```
### Initialization

# csv input with header=voters and row labels=Pokemon
fnIn = '2023_ADV_rankings_outliers_in.csv'
fnOut = '2023_ADV_rankings_outliers_out.csv'
fnPrevIn = '2022_R2_ADV_rankings_outliers_in.csv'
fnPrevOut = '2022_R2_ADV_rankings_outliers_out.csv'
cutoff = 37 # cut off analysis at this number of Pokemon
rankFracThreshold = 1-6.5/24 # throw out Pokemon that are not ranked by over this fraction of voters
gen = 'ADV OU'
year = '2023'

rank = pd.read_csv(fnIn,header=0,index_col=0)
rankval = rank.values
N = np.shape(rankval)
```

In [46]:
```
### Outlier compensation
rankvalNoOutliers = copy.deepcopy(rankval)
Q1 = np.quantile(rankval,0.16,axis=1) # set reference range
Q3 = np.quantile(rankval,0.84,axis=1) # set reference range
IQR = Q3 - Q1
limit = 1 # set outlier boundary
for ii in range(0,rankval.shape[0]):
    lb = Q1[ii] - limit*IQR[ii]
    ub = Q3[ii] + limit*IQR[ii]
    for jj in range(0,rankval.shape[1]):
        if rankval[ii,jj] == np.nan:
            rankvalNoOutliers[ii,jj] = np.nan
        elif rankval[ii,jj] < lb:
            rankvalNoOutliers[ii,jj] = lb
        elif rankval[ii,jj] > ub:
            rankvalNoOutliers[ii,jj] = ub
```

In [47]:
```
rankNoOutliers = pd.DataFrame(rankvalNoOutliers)
rankNoOutliers.to_csv(fnOut, header=False, index=False)
```

In [48]:
```
### Sort data by outlier-removed ranks
rankval = rank.values
rankavg = np.nanmean(rankvalNoOutliers,axis=1)
rankstd = np.nanstd(rankvalNoOutliers,axis=1)
orderNoOutliers = np.argsort(rankavg)
rankavg = rankavg[orderNoOutliers]
rankstd = rankstd[orderNoOutliers]
monList = rank.index.values
monList = monList[orderNoOutliers]
rankval = rankval[orderNoOutliers,:]
# Ranking Difference
rankMeanSubtracted = rankval - np.repeat(np.array([rankavg]).transpose(),rankv
al.shape[1],axis=1)
```

In [49]:
```
### Final VR order
for n in range(0,N[0]):
    print(str(n+1).zfill(2) + ' ' + monList[n])
```

```
01 Tyranitar
02 Skarmory
03 Metagross
04 Swampert
05 Blissey
06 Zapdos
07 Gengar
08 Salamence
09 Jirachi
10 Celebi
11 Aerodactyl
12 Suicune
13 Claydol
14 Dugtrio
15 Starmie
16 Moltres
17 Charizard
18 Snorlax
19 Magneton
20 Flygon
21 Milotic
22 Breloom
23 Forretress
24 Heracross
25 Hariyama
26 Gyarados
27 Jolteon
28 Cloyster
29 Regice
30 Smeargle
31 Venusaur
32 Kingdra
33 Jynx
34 Medicham
35 Porygon2
36 Raikou
37 Vaporeon
38 Machamp
39 Registeel
40 Ludicolo
41 Weezing
42 Misdreavus
43 Blaziken
44 Steelix
45 Regirock
46 Camerupt
47 Armaldo
48 Glalie
49 Marowak
50 Articuno
51 Houndoom
52 Dusclops
53 Haunter
54 Donphan
55 Alakazam
56 Quagsire
57 Banette
```

```
58 Poliwrath
59 Lapras
60 Roselia
61 Hypno
62 Gardevoir
63 Umbreon
64 Dragonite
65 Golduck
66 Slaking
67 Sceptile
68 Exeggutor
69 Ninjask
70 Pinsir
71 Rhydon
72 Politoed
73 Rapidash
74 Scizor
75 Arcanine
76 Lanturn
77 Slowbro
78 Blastoise
79 Dewgong
80 Ursaring
81 Walrein
82 Lunatone
83 Mantine
84 Solrock
85 Jumpluff
86 Miltank
87 Sableye
88 Entei
89 Nidoqueen
90 Omastar
91 Aggron
92 Swellow
93 Nidoking
```

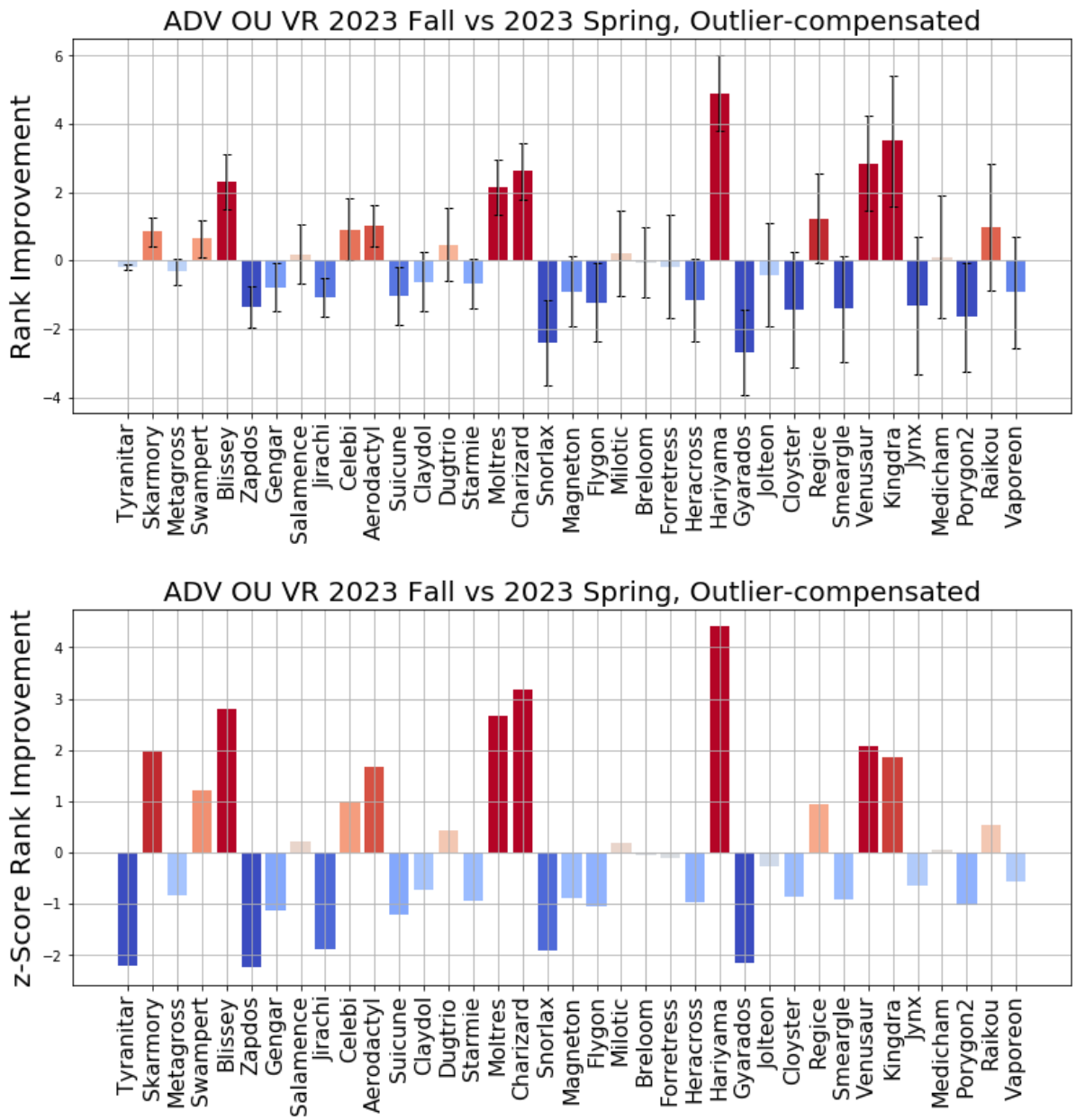
In [50]:
```
### Final VR order after removing frequently unranked Pokemon
orderCount = 0
monCount = 1
orderReduced = []
for n in range(0,N[0]):
    orderCount += 1
    if sum(np.isnan(rankval[n,:])) > rankFracThreshold*N[1]:
        continue
    print(str(monCount).zfill(2) + ' ' + monList[n])
    orderReduced += [orderCount-1]
    monCount += 1
```

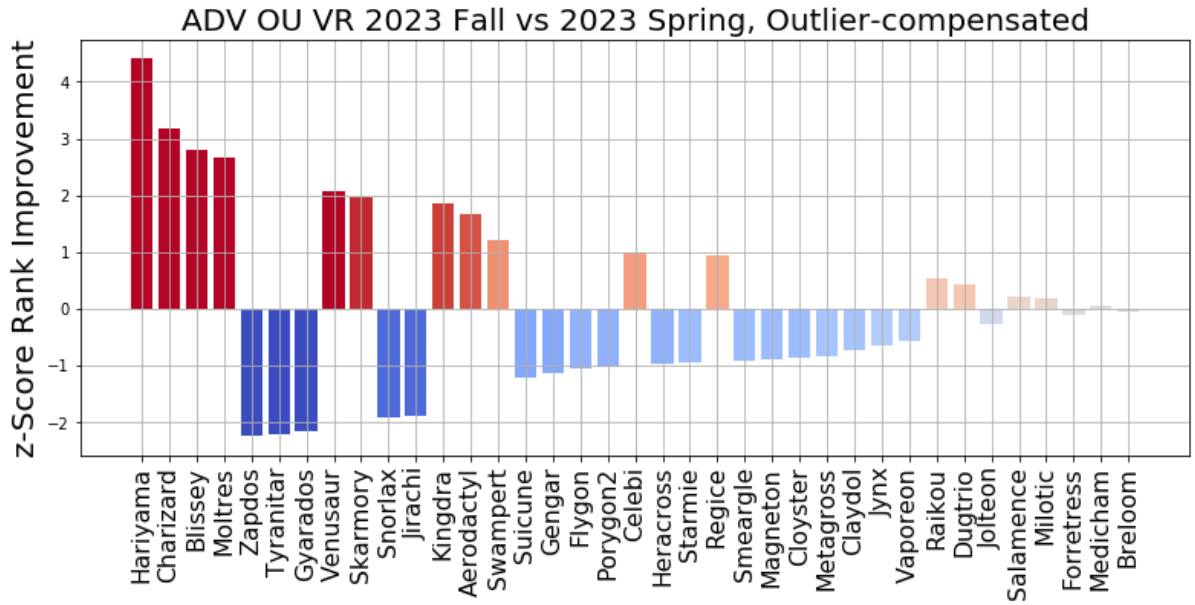
```
01 Tyranitar
02 Skarmory
03 Metagross
04 Swampert
05 Blissey
06 Zapdos
07 Gengar
08 Salamence
09 Jirachi
10 Celebi
11 Aerodactyl
12 Suicune
13 Claydol
14 Dugtrio
15 Starmie
16 Moltres
17 Charizard
18 Snorlax
19 Magneton
20 Flygon
21 Milotic
22 Breloom
23 Forretress
24 Heracross
25 Hariyama
26 Gyarados
27 Jolteon
28 Cloyster
29 Regice
30 Smeargle
31 Venusaur
32 Kingdra
33 Jynx
34 Medicham
35 Porygon2
36 Raikou
37 Vaporeon
38 Machamp
39 Registeel
40 Ludicolo
41 Weezing
42 Misdreavus
43 Blaziken
44 Steelix
45 Regirock
46 Camerupt
47 Armaldo
48 Glalie
49 Marowak
50 Houndoom
51 Roselia
52 Sceptile
```

In [51]:
```
### Analyze previous rankings
rankOld = pd.read_csv(fnPrevIn,header=0,index_col=0)
rankvalOld = rankOld.values
Nold = np.shape(rankvalOld)

rankNoOutliersOld = pd.DataFrame(rankvalNoOutliers)
rankNoOutliersOld.to_csv(fnPrevOut, header=False, index=False)

rankvalNoOutliersOld = copy.deepcopy(rankvalOld)
Q1old = np.quantile(rankvalOld,0.16,axis=1)
Q3old = np.quantile(rankvalOld,0.84,axis=1)
IQRold = Q3old - Q1old
limitOld = 1
for ii in range(0,rankvalOld.shape[0]):
    lb = Q1old[ii] - limitOld*IQRold[ii]
    ub = Q3old[ii] + limitOld*IQRold[ii]
    for jj in range(0,rankvalOld.shape[1]):
        if rankvalOld[ii,jj] == np.nan:
            rankvalNoOutliersOld[ii,jj] = np.nan
        elif rankvalOld[ii,jj] < lb:
            rankvalNoOutliersOld[ii,jj] = lb
        elif rankvalOld[ii,jj] > ub:
            rankvalNoOutliersOld[ii,jj] = ub

rankavgOld = np.nanmean(rankvalNoOutliersOld,axis=1)
rankstdOld = np.nanstd(rankvalNoOutliersOld,axis=1)
orderNoOutliersOld = np.argsort(rankavgOld)
rankavgOld = rankavgOld[orderNoOutliersOld]
rankstdOld = rankstdOld[orderNoOutliersOld]
monListOld = rankOld.index.values
monListOld = monListOld[orderNoOutliersOld]

indexOldIntoNew = [list(monListOld).index(mon) for mon in monList[:cutoff]]
rankOldIntoNew = rankavgOld[indexOldIntoNew]
rankDiff = rankavg[:cutoff] - rankOldIntoNew
```

In [52]:
```
### Compare current and previous rankings
seismic = cm.get_cmap('coolwarm', 32)
rankDiffStd = np.sqrt(rankstdOld[:cutoff]**2/Nold[1]+rankstd[:cutoff]**2/N[1])
zChange = -rankDiff / (rankDiffStd+1e-7)

fig = plt.figure(figsize=(13,5))
ax = plt.axes()
ax.grid(zorder=0)
ax.bar(list(monList[:cutoff]),list(-rankDiff),zorder=1,color=seismic(0.5-rankD
iff/max(rankDiff)),yerr=rankDiffStd,ecolor='k',capsize=3)
plt.xticks(rotation=90,fontsize=580/cutoff)
plt.ylabel('Rank Improvement',fontsize=20)
#plt.title(gen + ' VR ' + year + ' vs ' + str(int(year)-1) + ', Outlier-compen
sated',fontsize=20)
plt.title(gen + ' VR ' + year + ' Fall vs ' + year + ' Spring, Outlier-compens
ated',fontsize=20)

fig = plt.figure(figsize=(13,5))
ax = plt.axes()
ax.grid(zorder=0)
ax.bar(list(monList[:cutoff]),list(zChange),zorder=1,color=seismic(0.5+zChang
e/max(np.abs(zChange))))
plt.xticks(rotation=90,fontsize=580/cutoff)
plt.ylabel('z-Score Rank Improvement',fontsize=20)
#plt.title(gen + ' VR ' + year + ' vs ' + str(int(year)-1) + ', Outlier-compen
sated',fontsize=20)
plt.title(gen + ' VR ' + year + ' Fall vs ' + year + ' Spring, Outlier-compens
ated',fontsize=20)

orderChange = sorted(range(len(monList[:cutoff])),reverse=True,key=lambda x:n
p.abs(zChange[x]))
fig = plt.figure(figsize=(13,5))
ax = plt.axes()
ax.grid(zorder=0)
ax.bar(list(monList[:cutoff][orderChange]),list(zChange[orderChange]),zorder=
1,color=seismic(0.5+zChange[orderChange]/max(np.abs(zChange))))
plt.xticks(rotation=90,fontsize=580/cutoff)
plt.ylabel('z-Score Rank Improvement',fontsize=20)
#plt.title(gen + ' VR ' + year + ' vs ' + str(int(year)-1) + ', Outlier-compen
sated',fontsize=20)
plt.title(gen + ' VR ' + year + ' Fall vs ' + year + ' Spring, Outlier-compens
ated',fontsize=20)
```

Out[52]: Text(0.5, 1.0, 'ADV OU VR 2023 Fall vs 2023 Spring, Outlier-compensated')





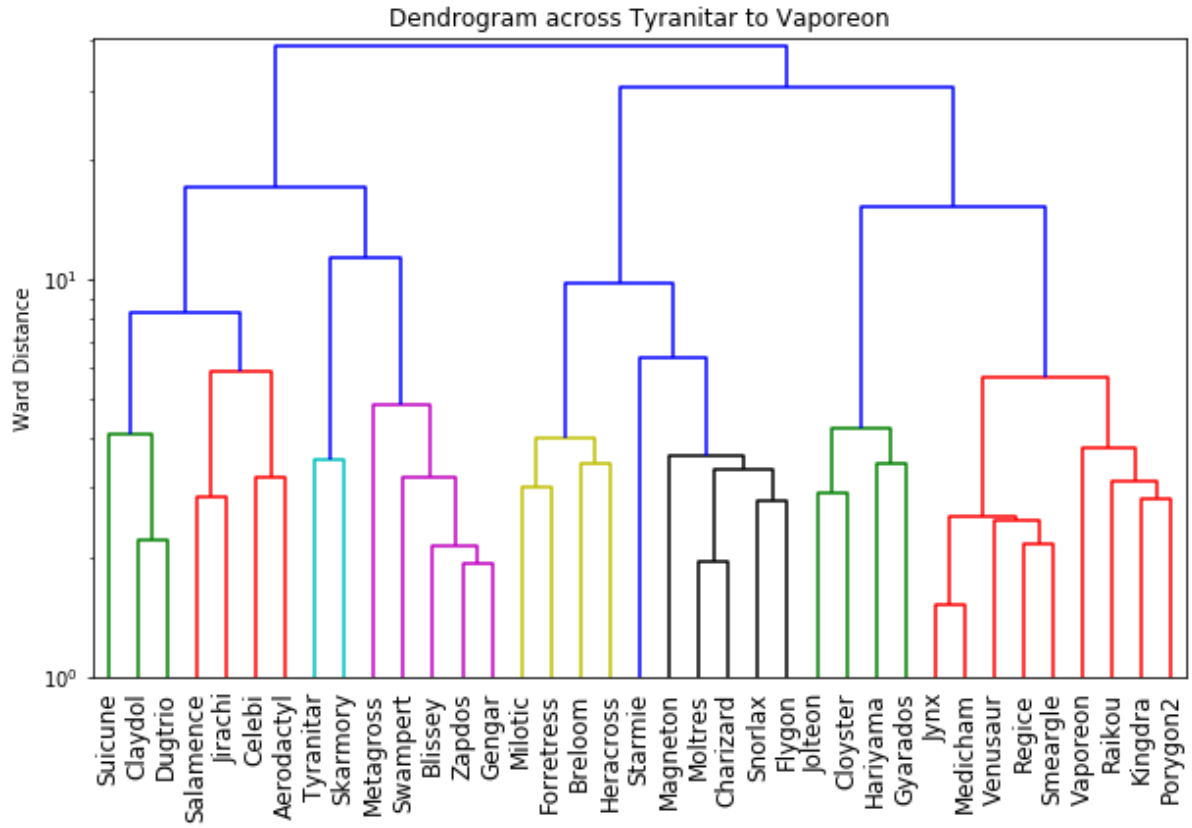
In [53]:
```
### Substitute non-ranked Pokemon according to missingRankMethod
# If using mean rank, missingRankMethod = 0; if substituting by rank 999, miss
ingRankMethod = 1
# Used for dendrogram, dissimilarity matrix and camps analysis
missingRankMethod = 0

rankvalSub = copy.deepcopy(rankval)
for m in np.arange(0,cutoff):
    for n in np.arange(0,N[1]):
        if np.isnan(rankval[m,n]):
            if missingRankMethod == 0:
                rankvalSub[m,n] = rankavg[m]
            elif missingRankMethod == 1:
                rankvalSub[m,n] = 999
```

In [54]:
```
### Handle non-ranked Pokemon and eliminating correlation in case of equal ranks

maxDist = 3.5
dissimMons = np.zeros((cutoff,cutoff))
def compareArr(arr1,arr2):
    assert len(arr1) == len(arr2)
    N = len(arr1)
    arr1cp = copy.deepcopy(arr1)
    arr2cp = copy.deepcopy(arr2)
    logicalArr = np.zeros(N)
    for n in range(N):
        if arr1cp[n] == arr2cp[n]:
            logicalArr[n] = 0.5
        else:
            logicalArr[n] = arr1cp[n] > arr2cp[n]
    return logicalArr

### Perform rank rate comparison across Pokemon pairs and perform logit transform
for m1 in range(cutoff):
    for m2 in range(cutoff):
        if m1 == m2:
            continue
        rate = sum(compareArr(rankvalSub[m1,:],rankvalSub[m2,:])) / N[1]
        dissimMons[m1,m2] = min(abs(np.log(rate/(1-rate))),maxDist)
```

```
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:24: RuntimeWarning: divide by zero encountered in log
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:24: RuntimeWarning: divide by zero encountered in double_scalars
```

In [55]:
```
### Perform hierarchical clustering with Ward linkage
# Adjust the threshold to get the appropriate tiers
tierThreshold = 6 # threshold
Hmons = cluster.hierarchy.ward(dissimMons)
fig = plt.figure(figsize=(10,6))
ax = plt.gca()
ax.set_yscale('log')
ax.set_ylim([1,50])
ax.set_ylabel('Ward Distance')
Dmons = cluster.hierarchy.dendrogram(Hmons,labels=[monList[i] for i in range(c
utoff)],leaf_rotation=90,color_threshold=tierThreshold,leaf_font_size=12)
clustersMons = cluster.hierarchy.fcluster(Hmons, tierThreshold, criterion='dis
tance')
plt.title('Dendrogram across ' + monList[0] + ' to ' + monList[cutoff-1]);
```

```
C:\Users\lambj\anaconda3\lib\site-packages\scipy\cluster\hierarchy.py:830: Cl
usterWarning: scipy.cluster: The symmetric non-negative hollow observation ma
trix looks suspiciously like an uncondensed distance matrix
  return linkage(y, method='ward', metric='euclidean')
C:\Users\lambj\anaconda3\lib\site-packages\scipy\cluster\hierarchy.py:2834: U
serWarning: Attempted to set non-positive bottom ylim on a log-scaled axis.
Invalid limit will be ignored.
  ax.set_ylim([0, dvw])
```



In [56]:
```
# Check consistency of clustering with rankings
# If inconsistent, Pokemon of the same cluster number will be separated.
print('Cluster Labels: ' + ",".join([str(m) for m in clustersMons]))
uniqueLabels = [clustersMons[0]]
for x in clustersMons:
    if x != uniqueLabels[-1]:
        uniqueLabels.append(x)
crossovers = []
for ii in range(len(uniqueLabels)-3):
    if uniqueLabels[ii] == uniqueLabels[ii+2] and uniqueLabels[ii+1] == unique
Labels[ii+3]:
        crossovers.append([uniqueLabels[ii],uniqueLabels[ii+1]])
for p in crossovers:
    idx10 = list(clustersMons).index(p[1])
    idx00 = list(clustersMons).index(p[0],idx10)
    idx11 = list(clustersMons).index(p[1],idx00)
    if idx00-idx10 == 1 and idx11-idx00 > 1:
        print(monList[idx10] + ' is miscategorized / misordered')
    elif idx00-idx10 > 1 and idx11-idx00 == 1:
        print(monList[idx00] + ' is miscategorized / misordered')
    else:
        print(", ".join(monList[idx10:idx11]) + ' are miscategorized / misorde
red')
```

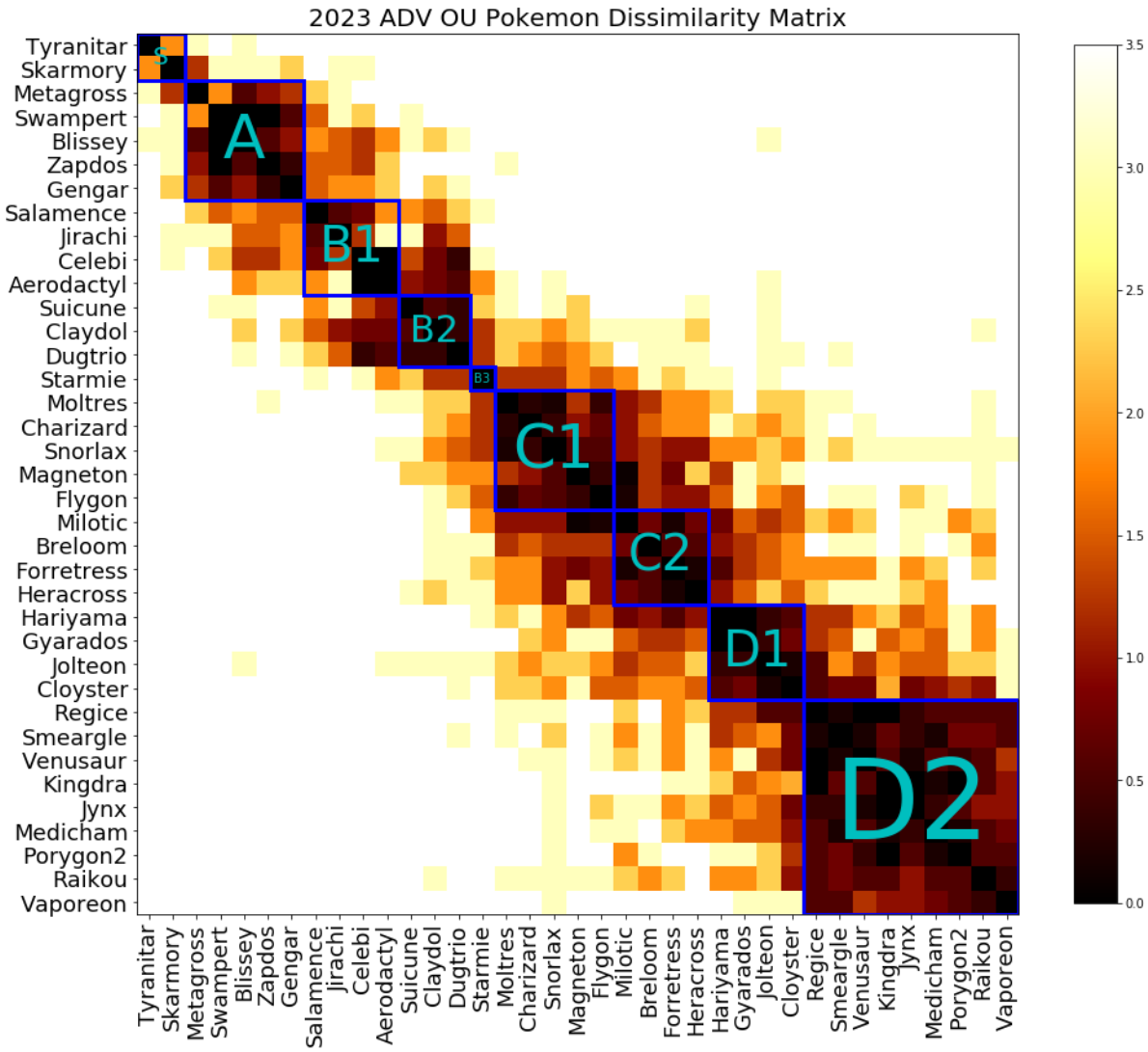
```
Cluster Labels: 3,3,4,4,4,4,4,2,2,2,2,1,1,1,7,6,6,6,6,6,5,5,5,5,8,8,8,8,9,9,
9,9,9,9,9,9,9
```

In [57]:
```
# Tier Modification if inconsistent above, as well as any visual changes from
dendrogram
#clustersMons[list(monList).index('Porygon2')] = clustersMons[list(monList).in
dex('Regice')]
#clustersMons[list(monList).index('Dugtrio')] = clustersMons[list(monList).ind
ex('Aerodactyl')]

### Obtain clusters
clusterDividers = [0] + [clustersMons[i+1]-clustersMons[i] != 0 for i in range
(len(clustersMons)-1)] + [1]
clusterIndices = np.insert(np.nonzero(clusterDividers)[0],0,0)

### Modify cluster indices (insert lower tier cutoffs before len(monList), che
ck graph below)
clusterIndicesMod = np.append(clusterIndices,[49,len(monList)])
clusterIndicesMod.sort()

### Tier names (see dendrogram to determine numerical subtiers)
tierNames = ['S','A','B1','B2','B3','C1','C2','D1','D2','E1','E2']
```

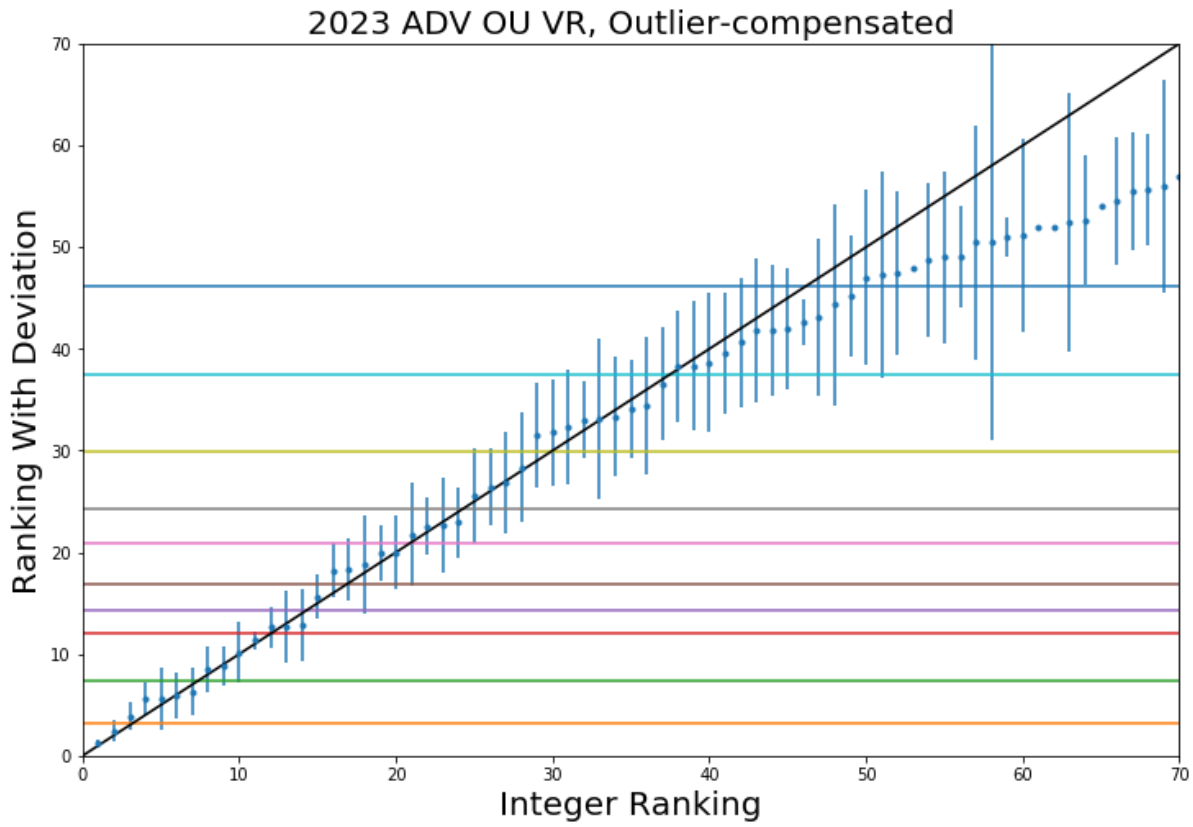
In [58]:
```
### Print VR tier writeup to be used in Smogon forums
textVR = ''
for n in range(len(clusterIndicesMod)-1):
    textVR += tierNames[n] + ': '
    for m in range(clusterIndicesMod[n],clusterIndicesMod[n+1]):
        if sum(np.isnan(rankval[m,:])) > rankFracThreshold*N[1]:
            continue
        textVR += ':' + monList[m] + ':'
    textVR += '\n'
print(textVR)
```

```
S: :Tyranitar::Skarmory:
A: :Metagross::Swampert::Blissey::Zapdos::Gengar:
B1: :Salamence::Jirachi::Celebi::Aerodactyl:
B2: :Suicune::Claydol::Dugtrio:
B3: :Starmie:
C1: :Moltres::Charizard::Snorlax::Magneton::Flygon:
C2: :Milotic::Breloom::Forretress::Heracross:
D1: :Hariyama::Gyarados::Jolteon::Cloyster:
D2: :Regice::Smeargle::Venusaur::Kingdra::Jynx::Medicham::Porygon2::Raikou::V
aporeon:
E1: :Machamp::Registeel::Ludicolo::Weezing::Misdreavus::Blaziken::Steelix::Re
girock::Camerupt::Armaldo::Glalie::Marowak:
E2: :Houndoom::Roselia::Sceptile:
```

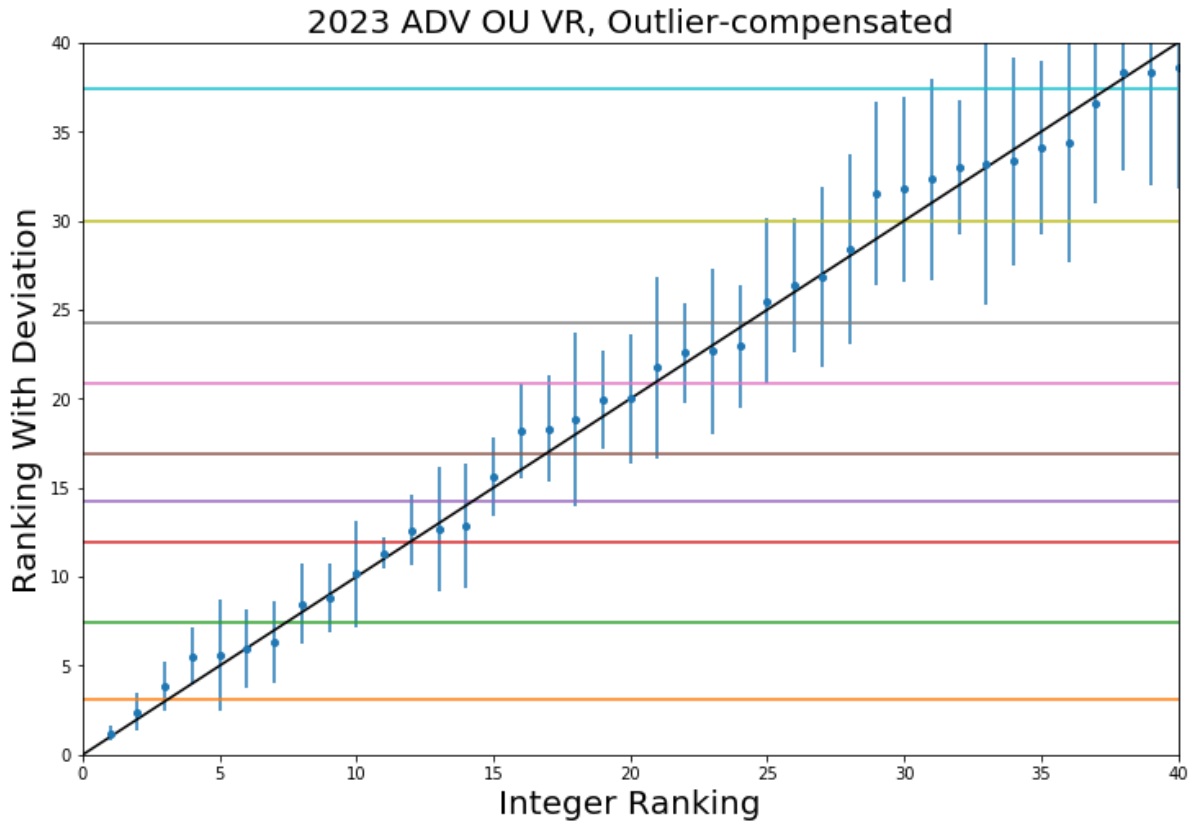
In [59]:
```
### Draw dissimilarity matrix from rank rates across pairs of Pokemon
fig = plt.figure(figsize=(16,16))
ax = plt.gca()
h = ax.imshow(dissimMons,interpolation='none',cmap='afmhot',vmin=0,vmax=maxDis
t)
for c in range(len(clusterIndicesMod)-1):
    ax.plot([clusterIndicesMod[c]-0.5,clusterIndicesMod[c+1]-0.5],[clusterIndi
cesMod[c]-0.5,clusterIndicesMod[c]-0.5],color='b',linewidth=3)
    ax.plot([clusterIndicesMod[c]-0.5,clusterIndicesMod[c+1]-0.5],[clusterIndi
cesMod[c+1]-0.5,clusterIndicesMod[c+1]-0.5],color='b',linewidth=3)
    ax.plot([clusterIndicesMod[c]-0.5,clusterIndicesMod[c]-0.5],[clusterIndice
sMod[c]-0.5,clusterIndicesMod[c+1]-0.5],color='b',linewidth=3)
    ax.plot([clusterIndicesMod[c+1]-0.5,clusterIndicesMod[c+1]-0.5],[clusterIn
dicesMod[c]-0.5,clusterIndicesMod[c+1]-0.5],color='b',linewidth=3)
fig.colorbar(h,ax=ax,shrink=0.8)
# ax.set_title('Dissimilarity Across S/A/B' + ' Camps\n Red = ' + namesA[camps
[0][0]] + ' Camp Favors Y; ' + namesA[camps[1][0]] + ' Camp Favors X\n Blue =
' + namesA[camps[1][0]] + ' Camp Favors Y; ' + namesA[camps[0][0]] + ' Camp Fa
vors X',fontsize=18)
ax.set_xticks(range(0,dissimMons.shape[1]))
ax.set_yticks(range(0,dissimMons.shape[0]))
ax.set_xticklabels([monList[i] for i in range(cutoff)],fontsize=18)
ax.set_yticklabels([monList[i] for i in range(cutoff)],fontsize=18)
ax.set_xlim([-1/2,cutoff-1/2])
ax.set_ylim([cutoff-1/2,-1/2])
ax.set_title(year + ' ' + gen + ' Pokemon Dissimilarity Matrix',fontsize=20)
plt.setp(ax.get_xticklabels(), rotation=90, ha="right",va="center",rotation_mo
de="anchor");
for ii in range(len(clusterIndicesMod)-1):
    pos = (clusterIndicesMod[ii]+clusterIndicesMod[ii+1]-1)/2
    if pos > cutoff:
        continue
    tierFontSize = (clusterIndicesMod[ii+1]-clusterIndicesMod[ii])*10
    text = ax.text(pos, pos, tierNames[ii],ha="center",va="center",color="c",f
ontsize=tierFontSize)
```


2023 ADV OU Pokemon Dissimilarity Matrix
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon
S
A
B1
B2
B3
C1
C2
D1
D2
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0

In [60]:
```
### Draw VR plot
monsLim = 70 # decide on a reasonable limit
# monsLim = len(monList)
fig = plt.figure(figsize=(12,8))
ax = plt.axes()
plt.errorbar(np.arange(1,N[0]+1),rankavg,yerr=rankstd,marker='.',linestyle='no
ne')
for c in clusterIndicesMod[1:-1]:
    plt.plot([0,monsLim],[(rankavg[c-1]+rankavg[c])/2]*2)
plt.plot([0,monsLim],[0,monsLim],color='k')
plt.xlim([0,monsLim])
plt.ylim([0,monsLim])
plt.xlabel('Integer Ranking',fontsize=20)
plt.ylabel('Ranking With Deviation',fontsize=20)
plt.title(year + ' ' + gen + ' VR, Outlier-compensated',fontsize=20);
```



In [61]:
```
### Draw zoomed in VR plot
monsLimZoom = 40 # zoom to this number
fig = plt.figure(figsize=(12,8))
ax = plt.axes()
plt.errorbar(np.arange(1,N[0]+1),rankavg,yerr=rankstd,marker='.',linestyle='no
ne',markersize=8)
for c in clusterIndicesMod[1:-1]:
    plt.plot([0,monsLim],[(rankavg[c-1]+rankavg[c])/2]*2)
plt.plot([0,monsLim],[0,monsLim],color='k')
plt.xlim([0,monsLimZoom])
plt.ylim([0,monsLimZoom])
plt.xlabel('Integer Ranking',fontsize=20)
plt.ylabel('Ranking With Deviation',fontsize=20)
plt.title(year + ' ' + gen + ' VR, Outlier-compensated',fontsize=20);
```

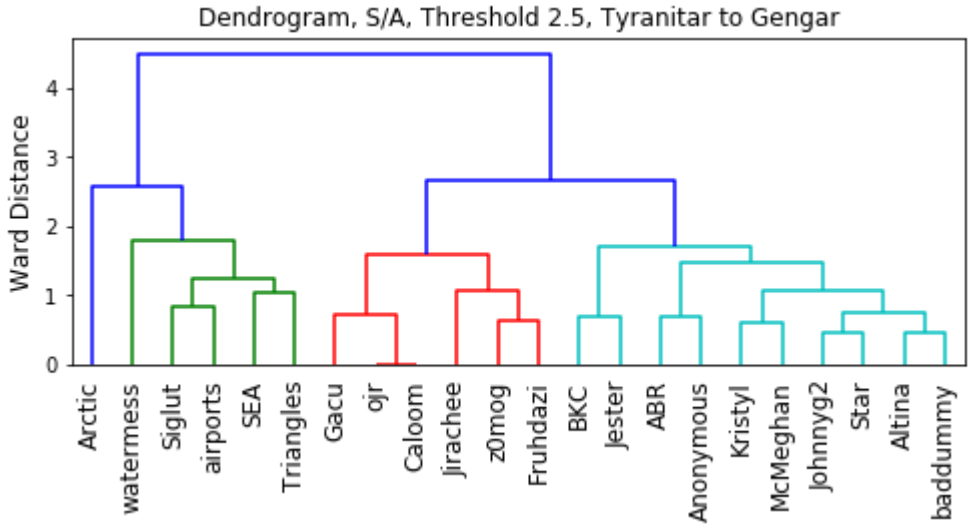


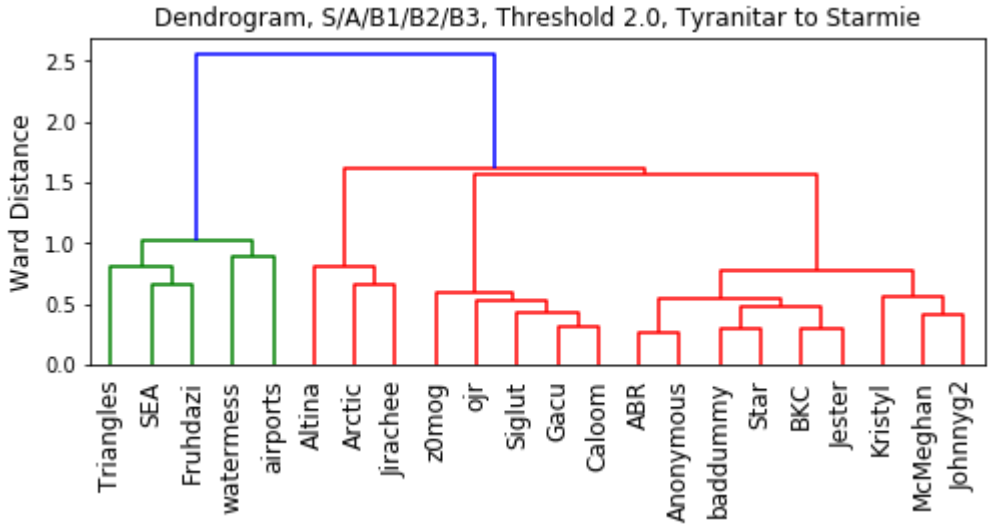
In [62]:
```
### Calculate ranking correlations for dissimilarity matrix across voters
C = len(clusterIndicesMod) - 1
totalCountArr = np.zeros((N[1],N[1],C,C))
cumulativeCountArr = np.zeros((N[1],N[1],C,C))
corrArr = np.zeros((N[1],N[1],C,C))
tArr = np.zeros((N[1],N[1],C,C))
for n1 in np.arange(0,N[1]):
    start = timer();
    for n2 in np.arange(0,N[1]):
        if n1 > n2:
            corrArr[n1,n2,:,:] = corrArr[n2,n1,:,:]
            totalCountArr[n1,n2,:,:] = totalCountArr[n2,n1,:,:]
        elif n1 == n2:
            continue
        else:
            for c1 in range(C):
                for c2 in range(c1+1):
                    for m1 in range(clusterIndicesMod[c1],clusterIndicesMod[c1
+1]):
                        for m2 in range(clusterIndicesMod[c2],clusterIndicesMo
d[c2+1]):
                            if m2 < m1:
                                corr = np.sign(rankvalSub[m1,n1]-rankvalSub[m
2,n1])*np.sign(rankvalSub[m1,n2]-rankvalSub[m2,n2])
                                corrArr[n1,n2,c1,c2] += corr
                                if corr != 0:
                                    totalCountArr[n1,n2,c1,c2] += 1

for c1 in range(C):
    for c2 in range(c1+1):
        tArr[:,:,c1,c2] = np.sum(corrArr[:,:,c2:c1,c2:c1],axis=(2,3))
        cumulativeCountArr[:,:,c1,c2] = np.sum(totalCountArr[:,:,c2:c1,c2:c1],
axis=(2,3))

tArr = tArr / cumulativeCountArr
for n1 in range(N[1]):
    tArr[n1,n1,:,:] = 1
```

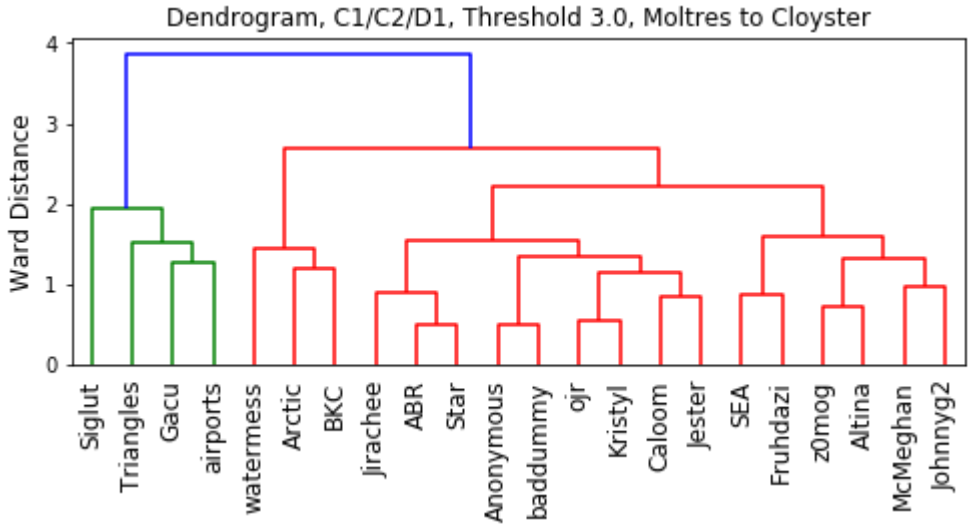
```
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:21: RuntimeW
arning: invalid value encountered in sign
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:31: RuntimeW
arning: invalid value encountered in true_divide
```

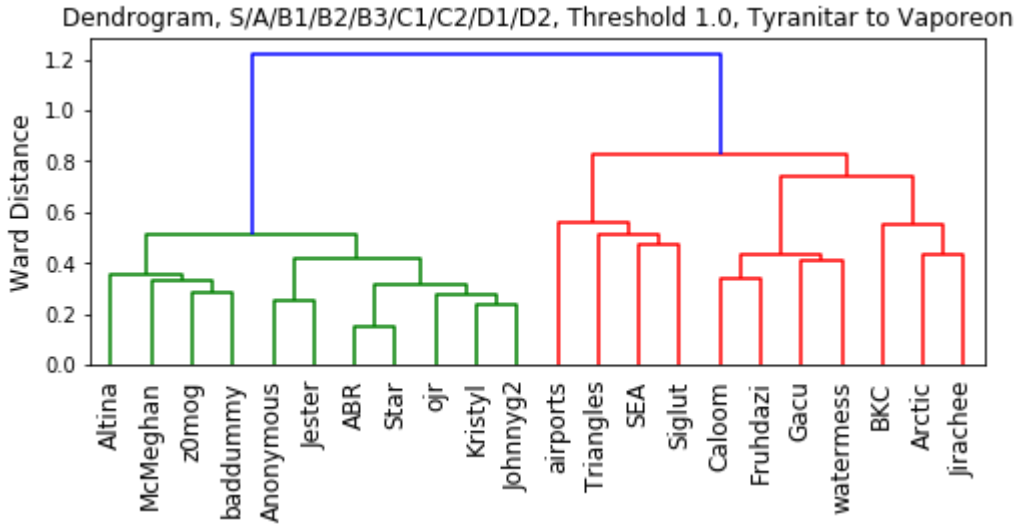
In [63]:
```
### Find dendrogram for interested tiers
# Interested tier spans are written in a list of lists called segments
# Each entry of segments captures the span from the tier represented by the fi
rst to second index
# The indices correspond to tier divisions defined by the clusterIndicesMod
segments = [[0,2],[0,5],[5,8],[0,9]]
names = list(rank.columns)
Hlist = []
Dlist = []
thresholdSegments = [2.55,2,3,1]

for i in range(len(segments)):
    pair = segments[i]
    fig = plt.figure(figsize=(8,3))
    ax = plt.axes()
    ax.set_ylabel('Ward Distance',fontsize=12)
    title = 'Dendrogram, ' + tierNames[pair[0]]
    for n in range(pair[0]+1,pair[1]):
        title += '/' + tierNames[n]
    title += ', Threshold ' + '{:.1f}'.format(thresholdSegments[i])
    title += ', ' + monList[clusterIndicesMod[pair[0]]] + ' to ' + monList[clu
sterIndicesMod[pair[1]]-1]

    ax.set_title(title,fontsize=12)
    Hlist += [cluster.hierarchy.ward(1-tArr[:,:,pair[1],pair[0]])]
    Dlist += [cluster.hierarchy.dendrogram(Hlist[-1],labels=names,color_thresh
old=thresholdSegments[i],leaf_rotation=90,leaf_font_size=12)]
```

```
C:\Users\lambj\anaconda3\lib\site-packages\scipy\cluster\hierarchy.py:830: Cl
usterWarning: scipy.cluster: The symmetric non-negative hollow observation ma
trix looks suspiciously like an uncondensed distance matrix
  return linkage(y, method='ward', metric='euclidean')
```








Dendrogram, S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, Tyranitar to Vaporeon
Ward Distance
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Altina
McMeghan
z0mog
baddummy
Anonymous
Jester
ABR
Star
ojr
Kristyl
Johnnyg2
airports
Triangles
SEA
Siglut
Caloom
Fruhdazi
Gacu
watermess
BKC
Arctic
jirachee

In [64]:
```
### Generates all statistical plots based on tier spans defined above in list
segments

for n in range(len(segments)):
    pair = segments[n]
    order = Dlist[n]['leaves']
    rankMeanSubtractedCutoff = rankMeanSubtracted[:cutoff,order]
    namesSegment = [names[i] for i in order]
    # marker = itertools.cycle
(('v','^','<','>','*','d','o','p','P','h','H','X','D'))
    marker = itertools.cycle(('v','^','<','>','*','d','X'))
    clustersSegmentUnordered = cluster.hierarchy.fcluster(Hlist[n], thresholdS
egments[n], criterion='distance')
    clustersSegment = clustersSegmentUnordered[order]
    clustersSegmentDividers = [0] + [clustersSegment[i+1]-clustersSegment[i] !
= 0 for i in range(len(clustersSegment)-1)] + [1]
    clustersSegmentIndices = np.insert(np.nonzero(clustersSegmentDividers)[0],
0,0)
    segmentLen = np.diff(clustersSegmentIndices)

    ### Plot Relative Ranks
    cArr = ['r'] + ['g'] + ['b'] + ['m'] + ['c']
    cArr.insert(np.argmax(segmentLen),'y')
    color = itertools.cycle(tuple(cArr))
    delta = 0.2
    fig = plt.figure(figsize=(12,5))
    ax = plt.axes()
    ax.grid(zorder=0)
    offset = lambda p: transforms.ScaledTranslation(p/72.,0, plt.gcf().dpi_sca
le_trans)
    rankMeanSubtractedSegmentCutoff = []
    rankStdSubtractedSegmentCutoff = []
    monListCutoff = list(monList[:cutoff])
    for p in range(0,len(segmentLen)):
        offsetNum = 5*((p-len(clustersSegmentIndices)+2)/(len(clustersSegmentI
ndices)-1)+0.5)
        rankMeanSubtractedSegment = rankMeanSubtractedCutoff[:,clustersSegment
Indices[p]:clustersSegmentIndices[p+1]]
        trans = plt.gca().transData
        # ax.scatter(list(monList[:Dcut]),list(-np.mean(rankMeanSubtractedADca
mp,axis=1)),err=list(np.std(rankMeanSubtractedADcamp,axis=1)),zorder=1,label=n
amesAD[p],marker = next(marker),s=200,alpha=0.5,color=cArrAD[p])
        rankMeanSubtractedSegmentCutoff += [-np.nanmean(rankMeanSubtractedSegm
ent,axis=1)]
        rankStdSubtractedSegmentCutoff += [np.nanstd(rankMeanSubtractedSegmen
t,axis=1)/np.sqrt(segmentLen[p]-1)]
        if segmentLen[p] < 4:
            continue
        ax.errorbar(monListCutoff,list(rankMeanSubtractedSegmentCutoff[p]),lis
t(rankStdSubtractedSegmentCutoff[p]),zorder=1,label=namesSegment[clustersSegme
ntIndices[p]],
                    linestyle='None',marker=next(marker),markersize=10,alpha=
0.5,color=next(color),transform=trans+offset(offsetNum))

    ax.set_ylim(ax.get_ylim())
    ax.set_xlim(ax.get_xlim())
```

```
    ax.plot([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.5],[-
99,99],color='k')
    ax.fill([-99,-99,clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]
-0.5],[-99,99,99,-99],'k',alpha=0.1)
    ax.plot([clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.5],[-
99,99],color='k')
    ax.fill([clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.5,cut
off*2,cutoff*2],[-99,99,99,-99],'k',alpha=0.1)
    plt.xticks(rotation=90,size=18)
    ax.legend(loc='upper center',bbox_to_anchor=(1.1,1),ncol=1)
    plt.ylabel('Relative Rank In Favor',fontsize=20)
    title = year + ' ' + gen + ' VR Relative Statistics: ' + tierNames[pair
[0]]
    for m in range(pair[0]+1,pair[1]):
        title += '/' + tierNames[m]
    plt.title(title,fontsize=20)

    ### Plot Ranking Correlations
    tClustered = np.array([[tArr[i,j,pair[1],pair[0]] for j in order] for i in
order])
    for ii in range(0,tClustered.shape[0]):
        tClustered[ii,ii] = np.nan
    fig = plt.figure(figsize=(12,12))
    ax = plt.gca()
    h = ax.imshow(tClustered,interpolation='none',cmap='viridis')
    fig.colorbar(h,ax=ax,shrink=0.8)
    title = 'Ranking Correlation'
    title = tierNames[pair[0]]
    for m in range(pair[0]+1,pair[1]):
        title += '/' + tierNames[m]
    title += ', ' + monList[clusterIndicesMod[pair[0]]] + ' to ' + monList[clu
sterIndicesMod[pair[1]]-1]

    ax.set_title(title,fontsize=18)
    ax.set_xticks(range(0,tClustered.shape[1]))
    ax.set_yticks(range(0,tClustered.shape[0]))
    ax.set_xticklabels([names[i] for i in order],fontsize=18)
    ax.set_yticklabels([names[i] for i in order],fontsize=18)
    plt.setp(ax.get_xticklabels(), rotation=90, ha="right",va="center",rotatio
n_mode="anchor");
    for ii in range(0,tClustered.shape[0]):
        for jj in range(0,tClustered.shape[1]):
            if ii != jj:
                text = ax.text(ii, jj, int(100*tClustered[ii, jj]),
                               ha="center", va="center", color="w",fontsize=1
2)

    ### Plot Sorted Differences by Z Score
    rankMeanSubtractedCutoffOtherSegments = []
    rankStdSubtractedCutoffOtherSegments = []
    rankStdDifference = []
    orderSegment = []
    seismic = cm.get_cmap('coolwarm', 32)
    dataNormalizer = colors.Normalize()
    for p in range(len(segmentLen)):
        rankSubtractedCutoffOtherSegments = rankMeanSubtractedCutoff[:,np.arra
y(np.concatenate((range(clustersSegmentIndices[p]),range(clustersSegmentIndice
```

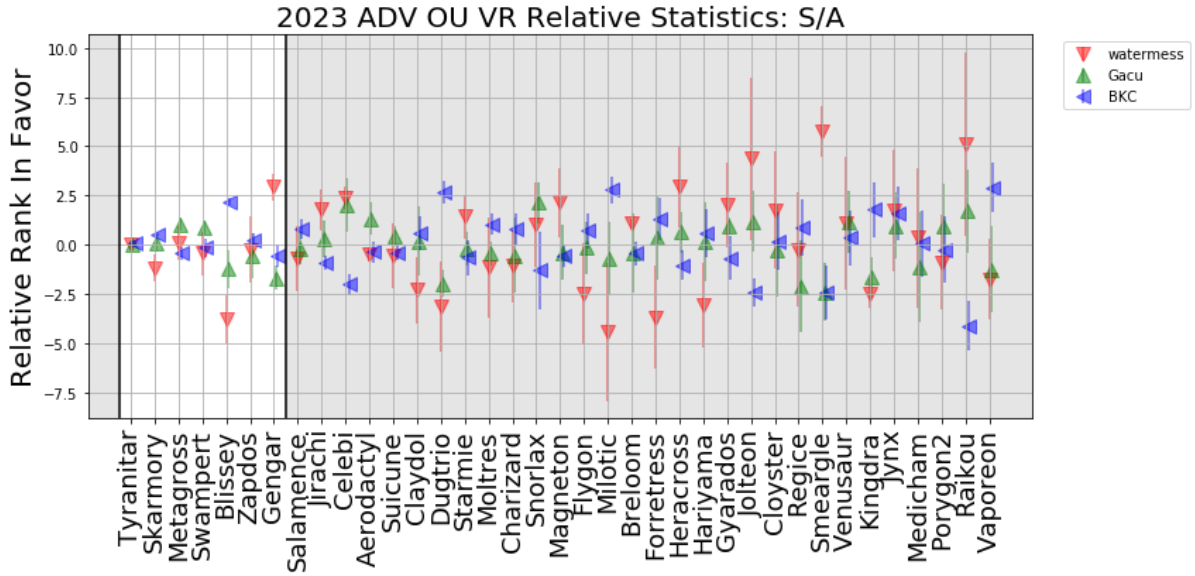
```
s[p+1],len(order))))],dtype=int)]
        rankMeanSubtractedCutoffOtherSegments += [-np.nanmean(rankSubtractedCu
toffOtherSegments,axis=1)]
        rankStdSubtractedCutoffOtherSegments += [np.nanstd(rankSubtractedCutof
fOtherSegments,axis=1)/np.sqrt(len(order)-segmentLen[p]-1)]
        rankStdDifference += [np.sqrt(rankStdSubtractedSegmentCutoff[p]**2 + r
ankStdSubtractedCutoffOtherSegments[p]**2)]
        zSegment = (rankMeanSubtractedSegmentCutoff[p] - rankMeanSubtractedCut
offOtherSegments[p]) / (rankStdDifference[p]+1e-7)

        orderSegment += [sorted(range(len(monListCutoff)),reverse=True,key=lam
bda x:np.abs(zSegment[x]))]
        if segmentLen[p] < 4:
            continue
        # order by z score
        fig = plt.figure(figsize=(12,5))
        ax = plt.axes()
        ax.grid(zorder=0)
        offset = lambda p: transforms.ScaledTranslation(p/72.,0, plt.gcf().dpi
_scale_trans)
        limit = min(max(abs(zSegment[abs(zSegment)<20]))*1.05,5)
        ax.bar(range(len(monListCutoff)),zSegment[orderSegment[p]],zorder=1,co
lor=seismic(zSegment[orderSegment[p]]/(2)+0.5))
        ax.set_xticks(range(len(monListCutoff)))
        ax.set_xticklabels([monListCutoff[i] for i in orderSegment[p]],fontsiz
e=360/cutoff)
        plt.xticks(rotation=90,size=16)
        ax.set_ylabel('Z Score of Relative Rank in Favor',fontsize=15)
        title = tierNames[pair[0]]
        for m in range(pair[0]+1,pair[1]):
            title += '/' + tierNames[m]
        title += ', Threshold ' + '{:.1f}'.format(thresholdSegments[i])
        title += ', ' + namesSegment[clustersSegmentIndices[p]] + ' camp'
        title += ', ' + str(segmentLen[p]) + '/' + str(len(order)) + ' voters'
        ax.set_title(title,fontsize=18)
        ax.set_ylim([-limit,limit])

        # order by rank
        fig = plt.figure(figsize=(12,5))
        ax = plt.axes()
        ax.grid(zorder=0)
        offset = lambda p: transforms.ScaledTranslation(p/72.,0, plt.gcf().dpi
_scale_trans)
        limit = min(max(abs(zSegment[abs(zSegment)<20]))*1.05,5)
        ax.bar(range(len(monListCutoff)),zSegment,zorder=1,color=seismic(zSegm
ent/(2)+0.5))
        ax.set_xticks(range(len(monListCutoff)))
        ax.set_xticklabels(monListCutoff,fontsize=360/cutoff)
        plt.xticks(rotation=90,size=16)
        ax.set_ylabel('Z Score of Relative Rank in Favor',fontsize=15)
        title = tierNames[pair[0]]
        for m in range(pair[0]+1,pair[1]):
            title += '/' + tierNames[m]
        title += ', Threshold ' + '{:.1f}'.format(thresholdSegments[i])
        title += ', ' + namesSegment[clustersSegmentIndices[p]] + ' camp'
        title += ', ' + str(segmentLen[p]) + '/' + str(len(order)) + ' voters'
        ax.set_title(title,fontsize=18)
```

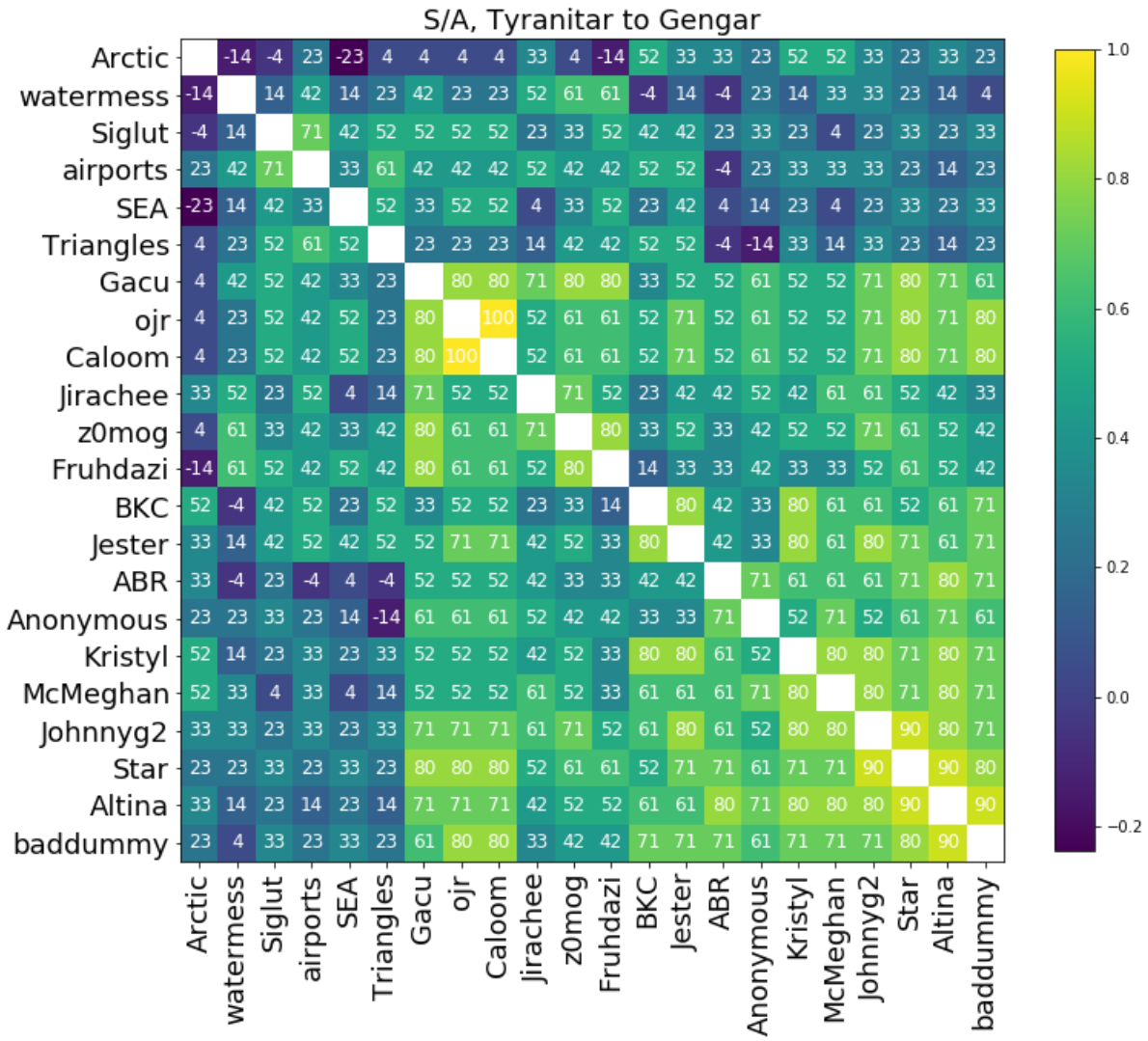
```
        ax.set_ylim([-limit,limit])
        ax.set_xlim(ax.get_xlim())
        ax.plot([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.
5],[-99,99],color='k')
        ax.fill([-99,-99,clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair
[0]]-0.5],[-99,99,99,-99],'k',alpha=0.1)
        ax.plot([clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.
5],[-99,99],color='k')
        ax.fill([clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.
5,cutoff*2,cutoff*2],[-99,99,99,-99],'k',alpha=0.1)

        ### Plot Dissimilarity Matrix
        dissimSegment = np.zeros((cutoff,cutoff))
        totalCountSegment = np.zeros((cutoff,cutoff))
        for m1 in range(cutoff):
            totalCountSegment[m1,m1] = 1
        for m1 in range(cutoff):
            for m2 in range(cutoff):
                if m1 == m2:
                    continue

                for n1 in range(clustersSegmentIndices[p],clustersSegmentIndic
es[p+1]):
                    for n2 in np.array(np.concatenate((range(clustersSegmentIn
dices[p]),range(clustersSegmentIndices[p+1],len(order)))),dtype=int):
                        sign1 = np.sign(rankvalSub[m1,order[n1]]-rankvalSub[m
2,order[n1]])
                        sign2 = np.sign(rankvalSub[m1,order[n2]]-rankvalSub[m
2,order[n2]])
                        corr = sign1 * sign2
                        if corr != 0:
                            dissimSegment[m1,m2] += - (1 - corr)/2 * sign1
                            totalCountSegment[m1,m2] += 1
        dissimSegment = dissimSegment / totalCountSegment
        for m1 in range(cutoff):
            dissimSegment[m1,m1] = 0
        fig = plt.figure(figsize=(12,12))
        ax = plt.gca()
        h = ax.imshow(dissimSegment,interpolation='none',cmap='seismic',vmin=-
1,vmax=1)
        ax.plot([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[1]]-0.
5],[clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.5],color='k')
        ax.plot([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[1]]-0.
5],[clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.5],color='k')
        ax.plot([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.
5],[clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[1]]-0.5],color='k')
        ax.plot([clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.
5],[clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[1]]-0.5],color='k')
        ax.fill([-0.5,-0.5,clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pa
ir[0]]-0.5],[-0.5,cutoff-0.5,cutoff-0.5,-0.5],'k',alpha=0.1)
        ax.fill([clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.
5,cutoff-0.5,cutoff-0.5],[-0.5,cutoff-0.5,cutoff-0.5,-0.5],'k',alpha=0.1)
        ax.fill([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.
5,clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.5],[clusterIndic
esMod[pair[1]]-0.5,cutoff-0.5,cutoff-0.5,clusterIndicesMod[pair[1]]-0.5,],'k',
alpha=0.1)
        ax.fill([clusterIndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.
```

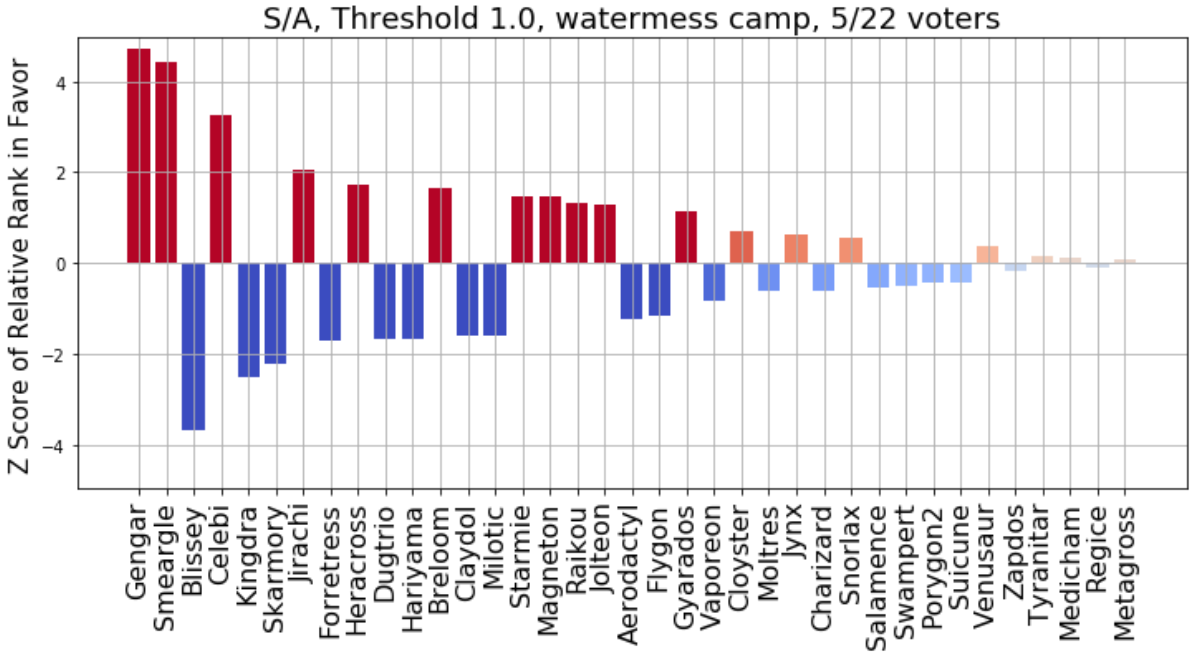
```
5,clusterIndicesMod[pair[1]]-0.5,clusterIndicesMod[pair[1]]-0.5],[-0.5,cluster
IndicesMod[pair[0]]-0.5,clusterIndicesMod[pair[0]]-0.5,-0.5],'k',alpha=0.1)

        fig.colorbar(h,ax=ax,shrink=0.8)
        title = 'Dissimilarity '
        title += tierNames[pair[0]]
        for m in range(pair[0]+1,pair[1]):
            title += '/' + tierNames[m]
        title += ', Threshold ' + '{:.1f}'.format(thresholdSegments[i])
        title += ', ' + str(segmentLen[p]) + '/' + str(len(order)) + ' voters
in '
        title += namesSegment[clustersSegmentIndices[p]] + ' Camp'
        title += ' \n Red = ' + namesSegment[clustersSegmentIndices[p]] + ' Ca
mp Favors Y; ' + ' Remainder Favors X\n Blue = ' + ' Remainder Favors Y; ' + n
amesSegment[clustersSegmentIndices[p]] + ' Camp Favors X'
        ax.set_title(title,fontsize=18)
        ax.set_xticks(range(0,dissimSegment.shape[1]))
        ax.set_yticks(range(0,dissimSegment.shape[0]))
        ax.set_xticklabels([monList[i] for i in range(cutoff)],fontsize=520/cu
toff)
        ax.set_yticklabels([monList[i] for i in range(cutoff)],fontsize=520/cu
toff)
        plt.setp(ax.get_xticklabels(), rotation=90, ha="right",va="center",rot
ation_mode="anchor");
        for ii in range(0,dissimSegment.shape[0]):
            for jj in range(0,dissimSegment.shape[1]):
                if dissimSegment[ii,jj] != 0:
                    text = ax.text(jj, ii, int(100*dissimSegment[ii, jj]),
                                   ha="center", va="center", color="w",fontsiz
e=360/cutoff)
```

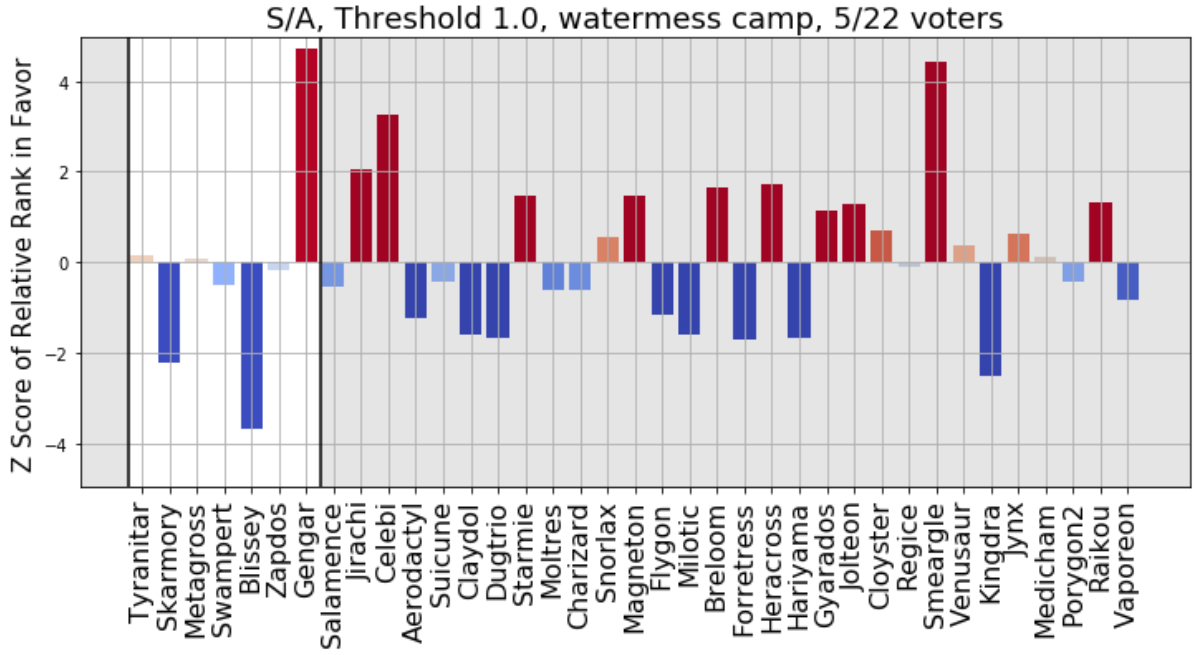
```
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:34: RuntimeW
arning: invalid value encountered in true_divide
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:58: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:98: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:118: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:163: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:98: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:118: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:163: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:21: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:58: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:98: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:118: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
```

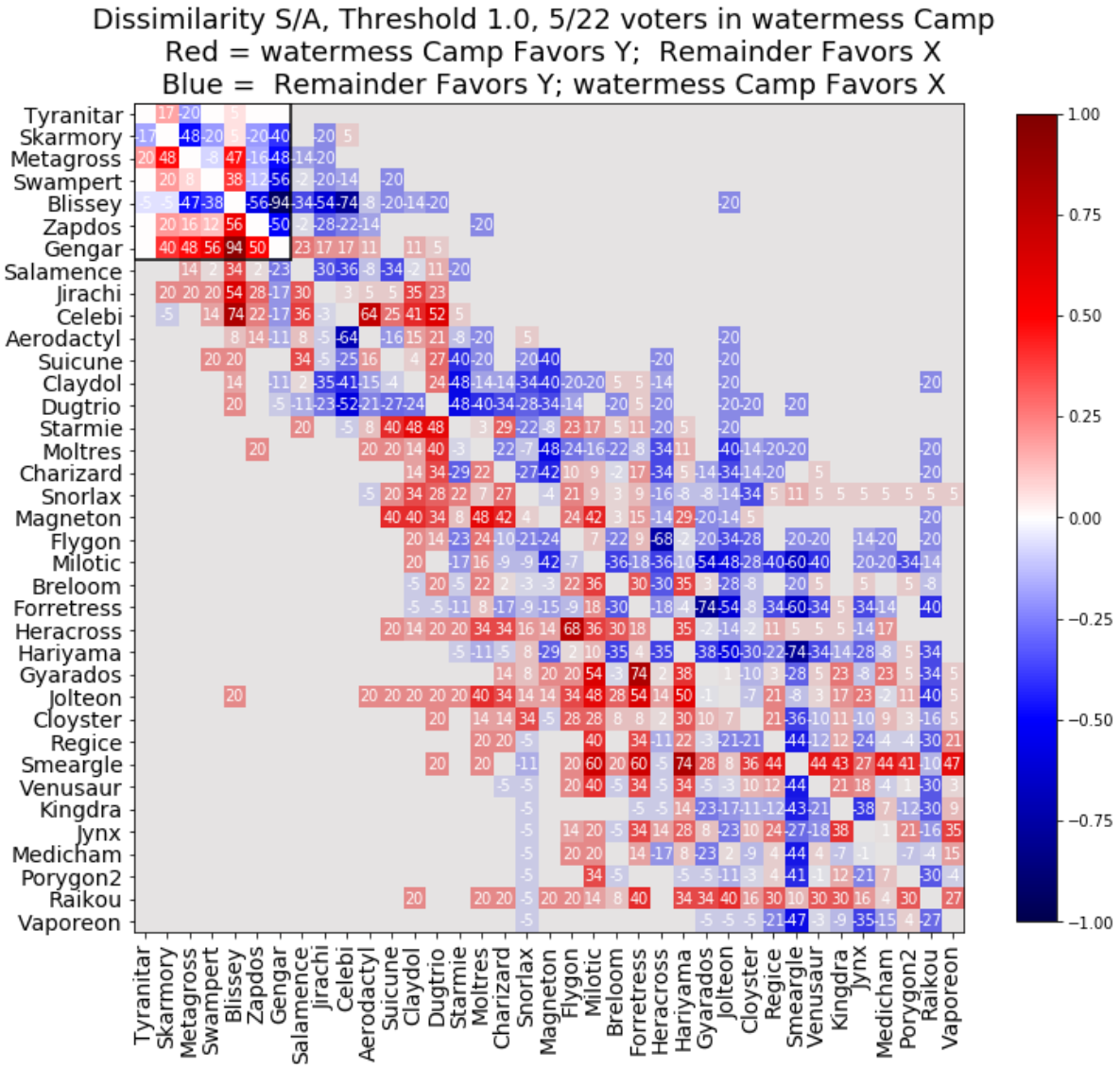
```
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:163: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:98: RuntimeW
arning: More than 20 figures have been opened. Figures created through the py
plot interface (`matplotlib.pyplot.figure`) are retained until explicitly clo
sed and may consume too much memory. (To control this warning, see the rcPara
m `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:118: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
C:\Users\lambj\anaconda3\lib\site-packages\ipykernel_launcher.py:163: Runtime
Warning: More than 20 figures have been opened. Figures created through the p
yplot interface (`matplotlib.pyplot.figure`) are retained until explicitly cl
osed and may consume too much memory. (To control this warning, see the rcPar
am `figure.max_open_warning`).
```

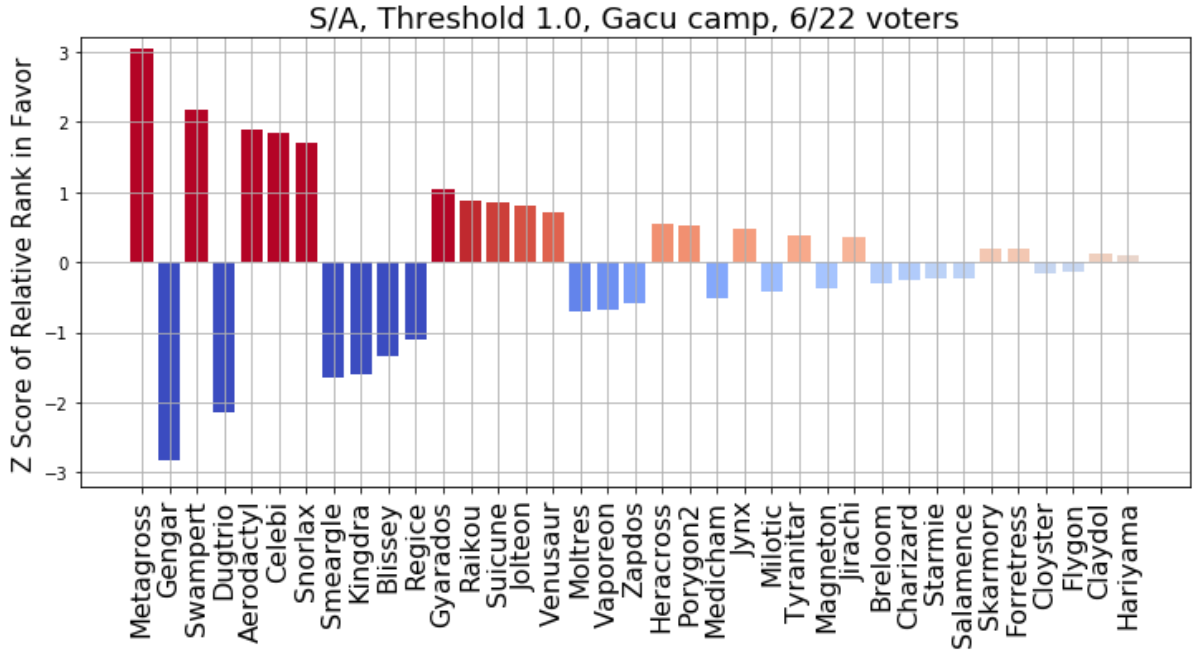


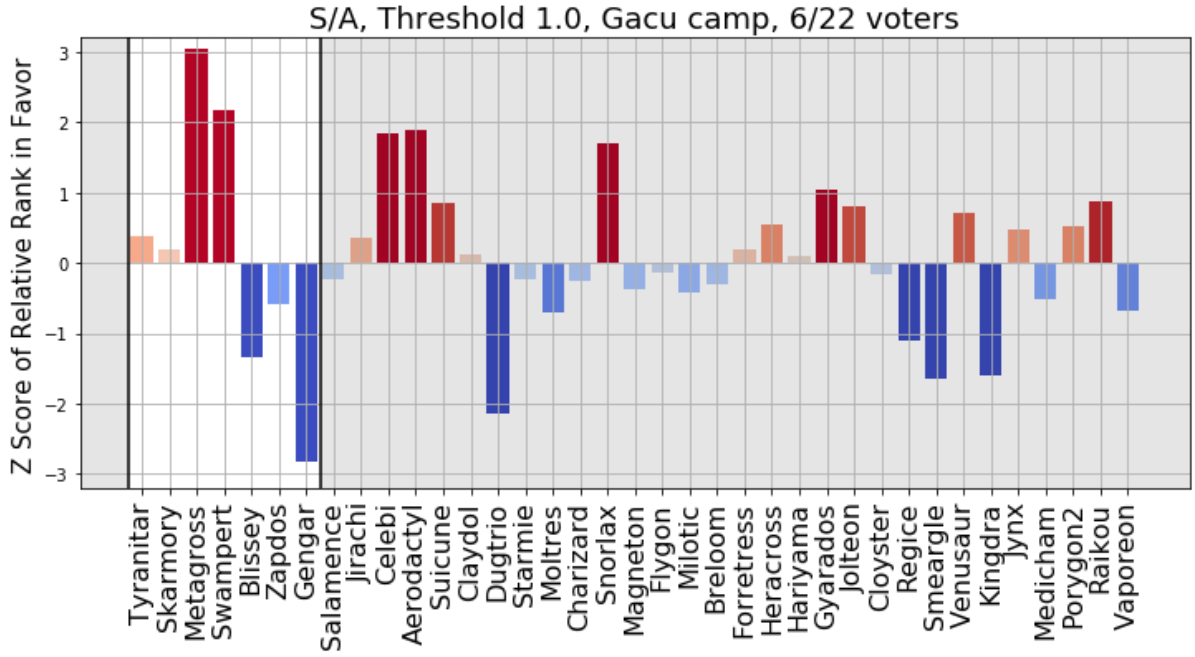

S/A, Tyranitar to Gengar
Arctic -14 -4 23 -23 4 4 4 4 33 4 -14 52 33 33 23 52 52 33 23 33 23
watermess -14 14 42 14 23 42 23 23 52 61 61 -4 14 -4 23 14 33 33 23 14 4
Siglut -4 14 71 42 52 52 52 52 23 33 52 42 42 23 33 23 4 23 33 23 33
airports 23 42 71 33 61 42 42 42 52 42 42 52 52 -4 23 33 33 33 23 14 23
SEA -23 14 42 33 52 33 52 52 4 33 52 23 42 4 14 23 4 23 33 23 33
Triangles 4 23 52 61 52 23 23 23 14 42 42 52 52 -4 -14 33 14 33 23 14 23
Gacu 4 42 52 42 33 23 80 80 71 80 80 33 52 52 61 52 52 71 80 71 61
ojr 4 23 52 42 52 23 80 100 52 61 61 52 71 52 61 52 52 71 80 71 80
Caloom 4 23 52 42 52 23 80 100 52 61 61 52 71 52 61 52 52 71 80 71 80
Jirachee 33 52 23 52 4 14 71 52 52 71 52 23 42 42 52 42 61 61 52 42 33
z0mog 4 61 33 42 33 42 80 61 61 71 80 33 52 33 42 52 52 71 61 52 42
Fruhdazi -14 61 52 42 52 42 80 61 61 52 80 14 33 33 42 33 33 52 61 52 42
BKC 52 -4 42 52 23 52 33 52 52 23 33 14 80 42 33 80 61 61 52 61 71
Jester 33 14 42 52 42 52 52 71 71 42 52 33 80 42 33 80 61 80 71 61 71
ABR 33 -4 23 -4 4 -4 52 52 52 42 33 33 42 42 71 61 61 61 71 80 71
Anonymous 23 23 33 23 14 -14 61 61 61 52 42 42 33 33 71 52 71 52 61 71 61
Kristyl 52 14 23 33 23 33 52 52 52 42 52 33 80 80 61 52 80 80 71 80 71
McMeghan 52 33 4 33 4 14 52 52 52 61 52 33 61 61 61 71 80 80 71 80 71
Johnnyg2 33 33 23 33 23 33 71 71 71 61 71 52 61 80 61 52 80 80 90 80 71
Star 23 23 33 23 33 23 80 80 80 52 61 61 52 71 71 61 71 71 90 90 80
Altina 33 14 23 14 23 14 71 71 71 42 52 52 61 61 80 71 80 80 80 90 90
baddummy 23 4 33 23 33 23 61 80 80 33 42 42 71 71 71 61 71 71 71 80 90
Arctic watermess Siglut airports SEA Triangles Gacu ojr Caloom Jirachee z0mog Fruhdazi BKC Jester ABR Anonymous Kristyl McMeghan Johnnyg2 Star Altina baddummy
1.0 0.8 0.6 0.4 0.2 0.0 −0.2

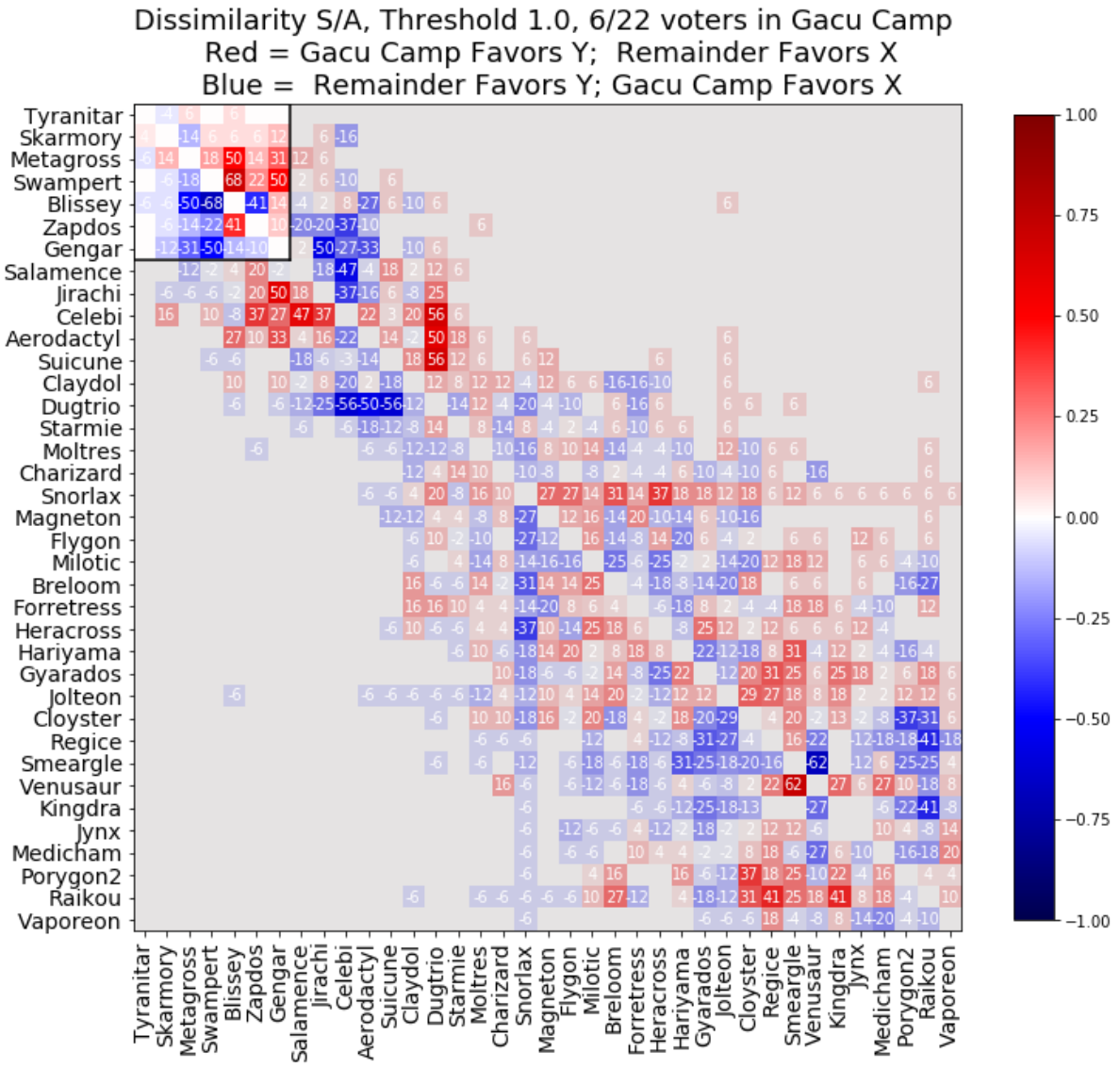

S/A, Threshold 1.0, watermess camp, 5/22 voters
Z Score of Relative Rank in Favor
4 2 0 −2 −4
Gengar Smeargle Blissey Celebi Kingdra Skarmory Jirachi Forretress Heracross Dugtrio Hariyama Breloom Claydol Milotic Starmie Magneton Raikou Jolteon Aerodactyl Flygon Gyarados Vaporeon Cloyster Moltres Jynx Charizard Snorlax Salamence Swampert Porygon2 Suicune Venusaur Zapdos Tyranitar Medicham Regice Metagross

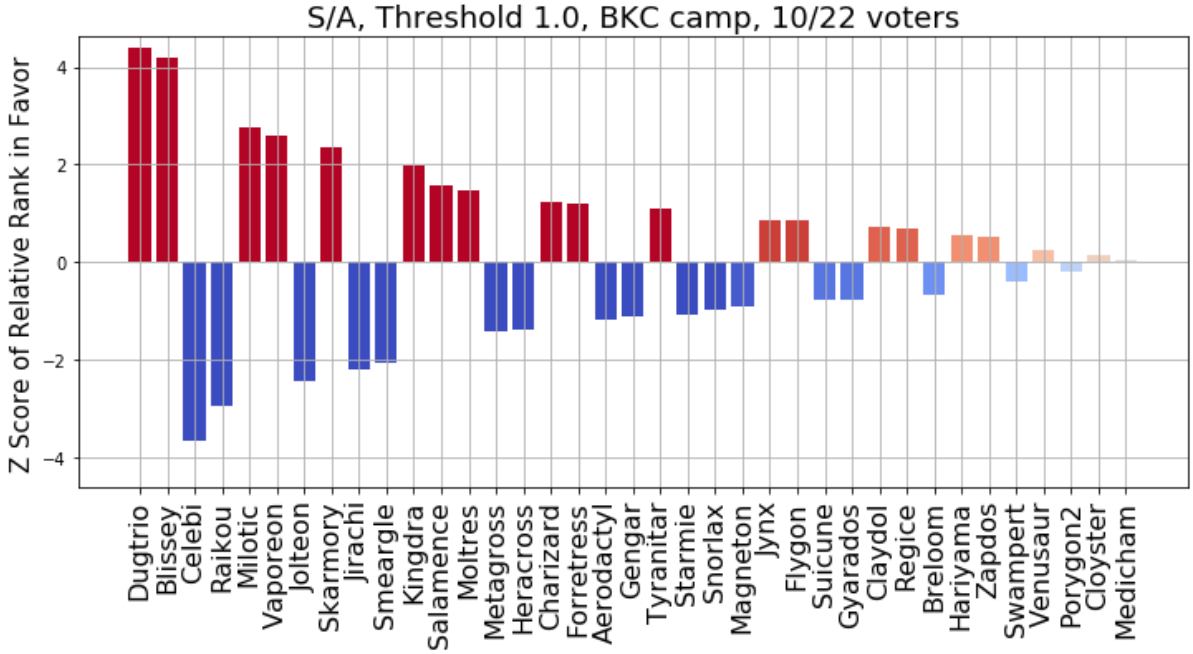

S/A, Threshold 1.0, watermess camp, 5/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

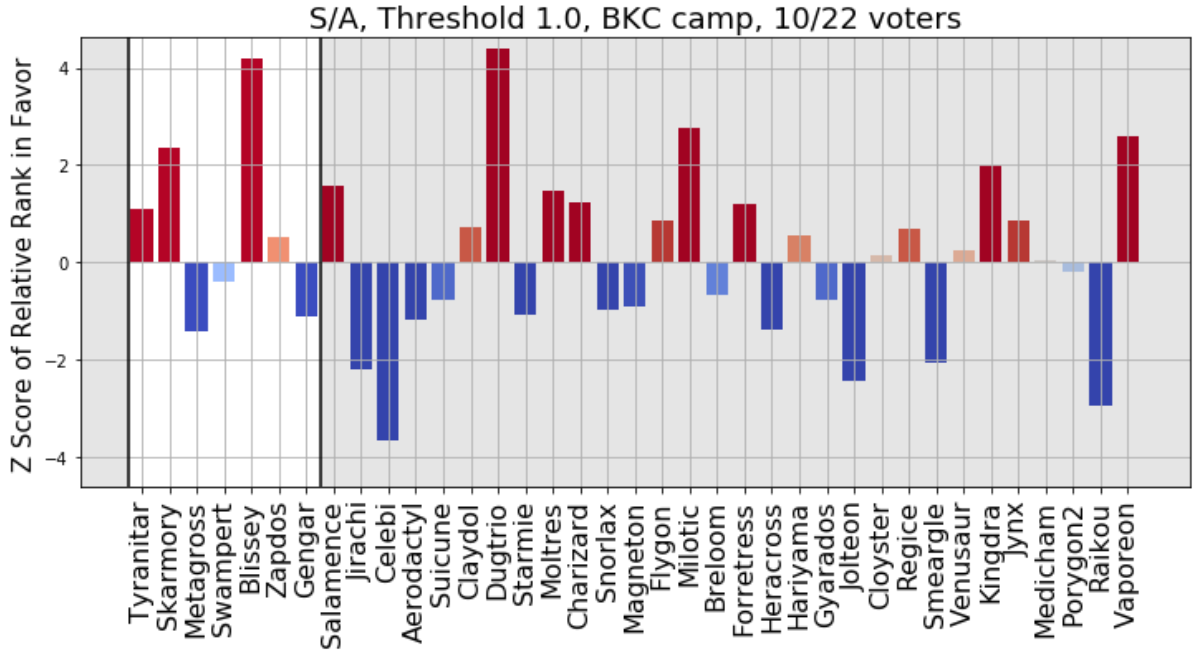

Dissimilarity S/A, Threshold 1.0, 5/22 voters in watermess Camp
Red = watermess Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; watermess Camp Favors X
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

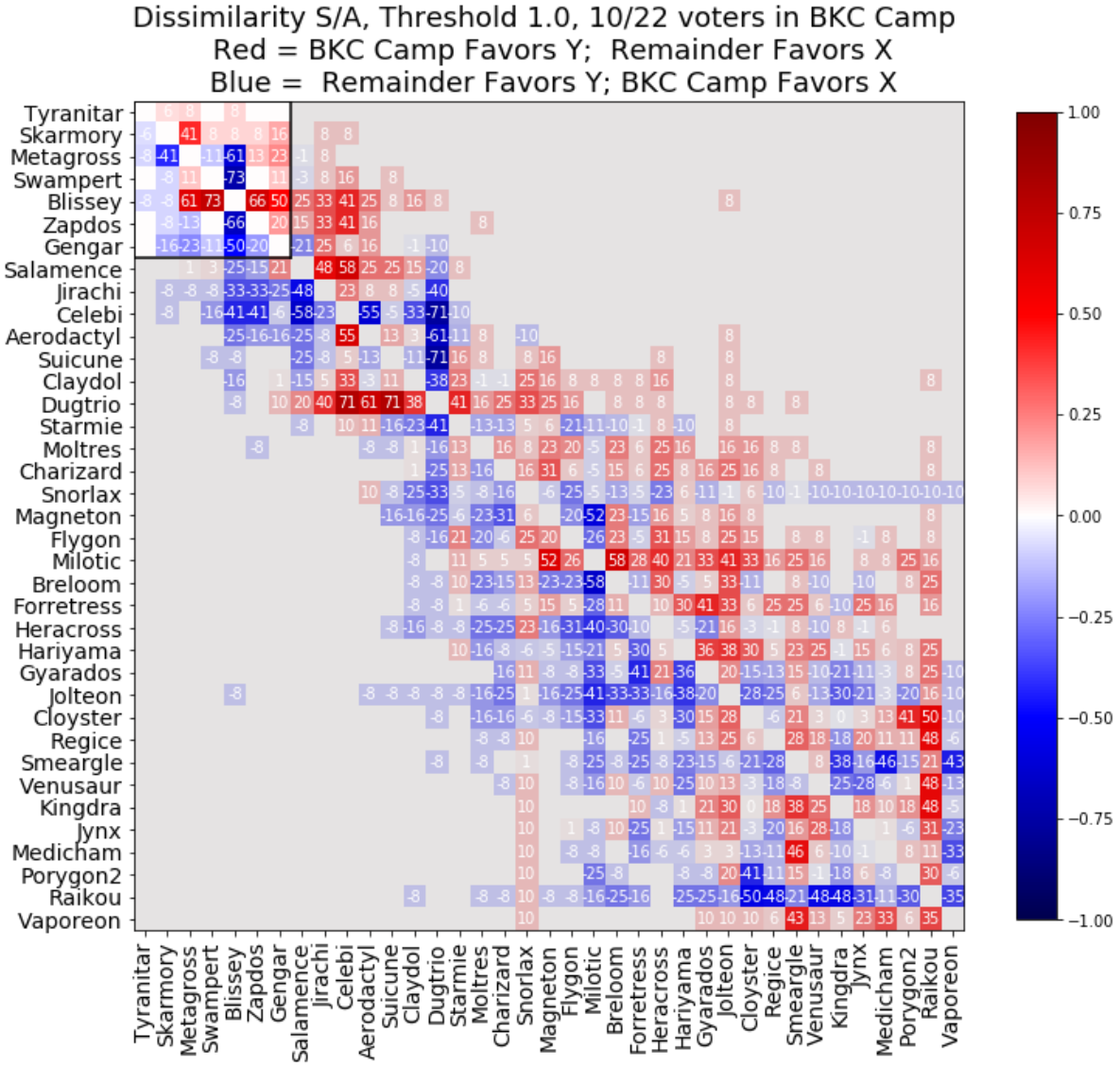

S/A, Threshold 1.0, Gacu camp, 6/22 voters
Z Score of Relative Rank in Favor
3
2
1
0
−1
−2
−3
Metagross
Gengar
Swampert
Dugtrio
Aerodactyl
Celebi
Snorlax
Smeargle
Kingdra
Blissey
Regice
Gyarados
Raikou
Suicune
Jolteon
Venusaur
Moltres
Vaporeon
Zapdos
Heracross
Porygon2
Medicham
Jynx
Milotic
Tyranitar
Magneton
Jirachi
Breloom
Charizard
Starmie
Salamence
Skarmory
Forretress
Cloyster
Flygon
Claydol
Hariyama

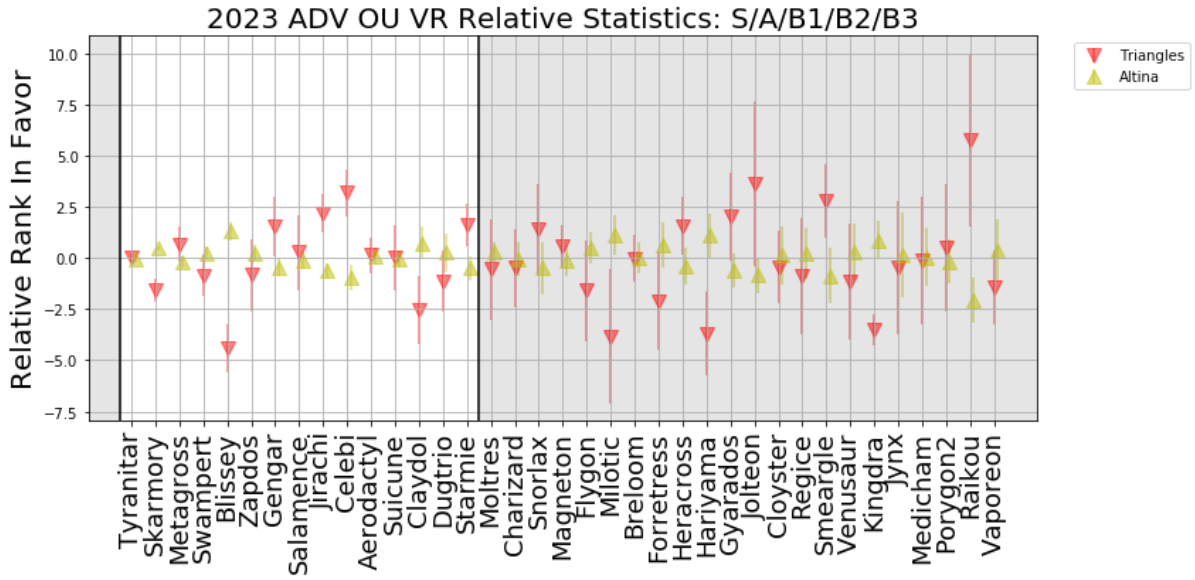

S/A, Threshold 1.0, Gacu camp, 6/22 voters
Z Score of Relative Rank in Favor
3
2
1
0
−1
−2
−3
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

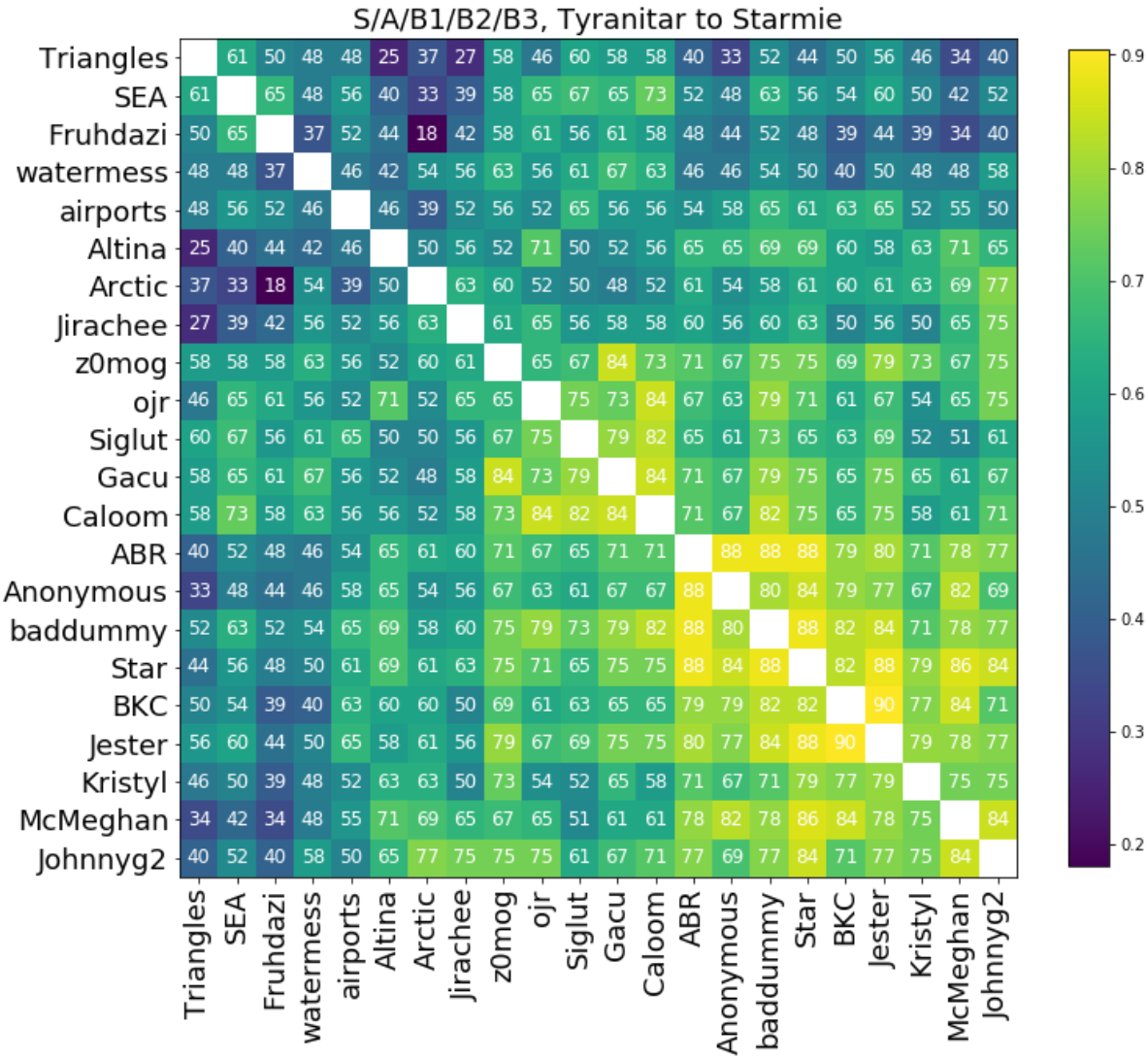

Dissimilarity S/A, Threshold 1.0, 6/22 voters in Gacu Camp
Red = Gacu Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; Gacu Camp Favors X
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

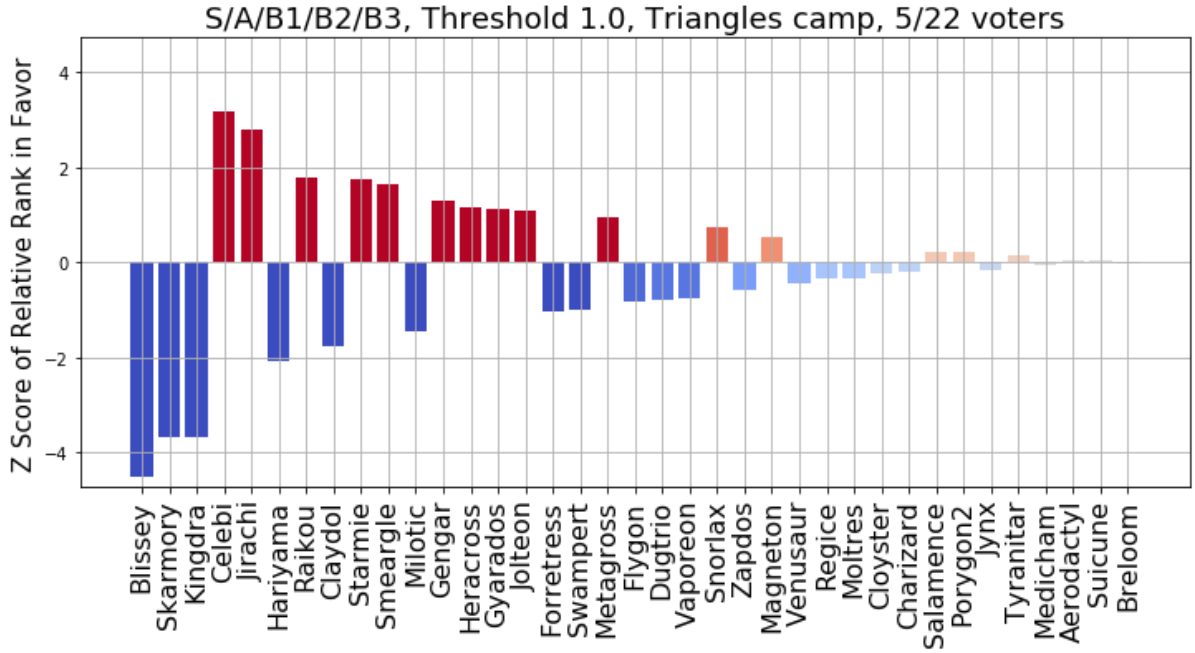

S/A, Threshold 1.0, BKC camp, 10/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Dugtrio
Blissey
Celebi
Raikou
Milotic
Vaporeon
Jolteon
Skarmory
Jirachi
Smeargle
Kingdra
Salamence
Moltres
Metagross
Heracross
Charizard
Forretress
Aerodactyl
Gengar
Tyranitar
Starmie
Snorlax
Magneton
Jynx
Flygon
Suicune
Gyarados
Claydol
Regice
Breloom
Hariyama
Zapdos
Swampert
Venusaur
Porygon2
Cloyster
Medicham

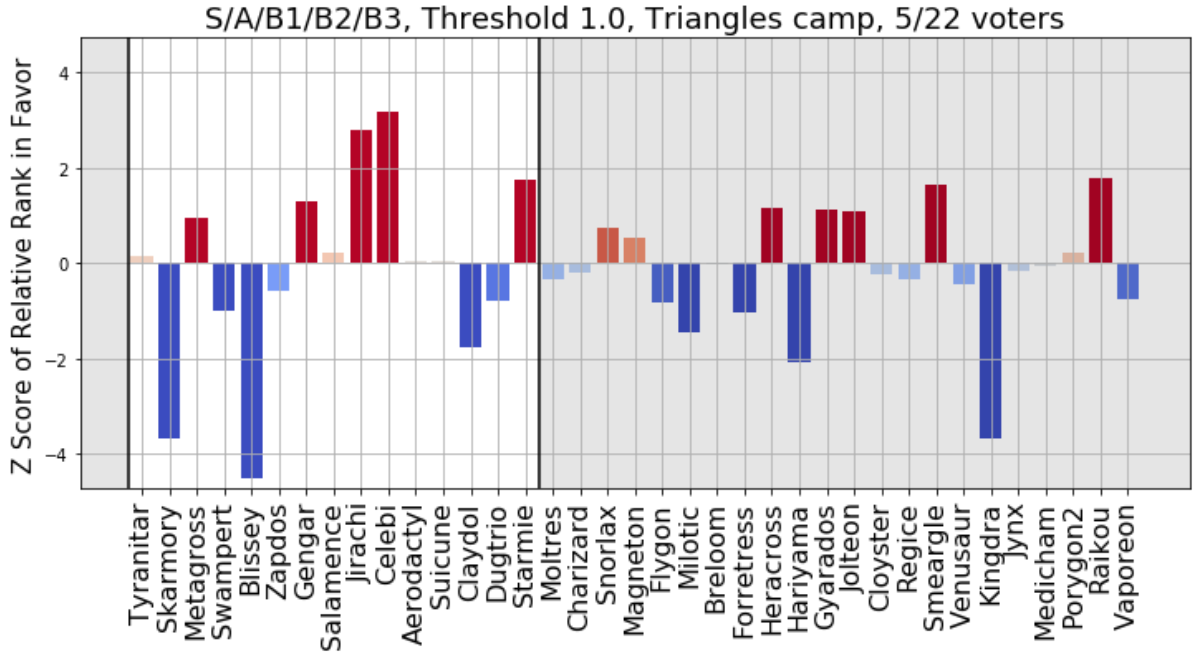

S/A, Threshold 1.0, BKC camp, 10/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

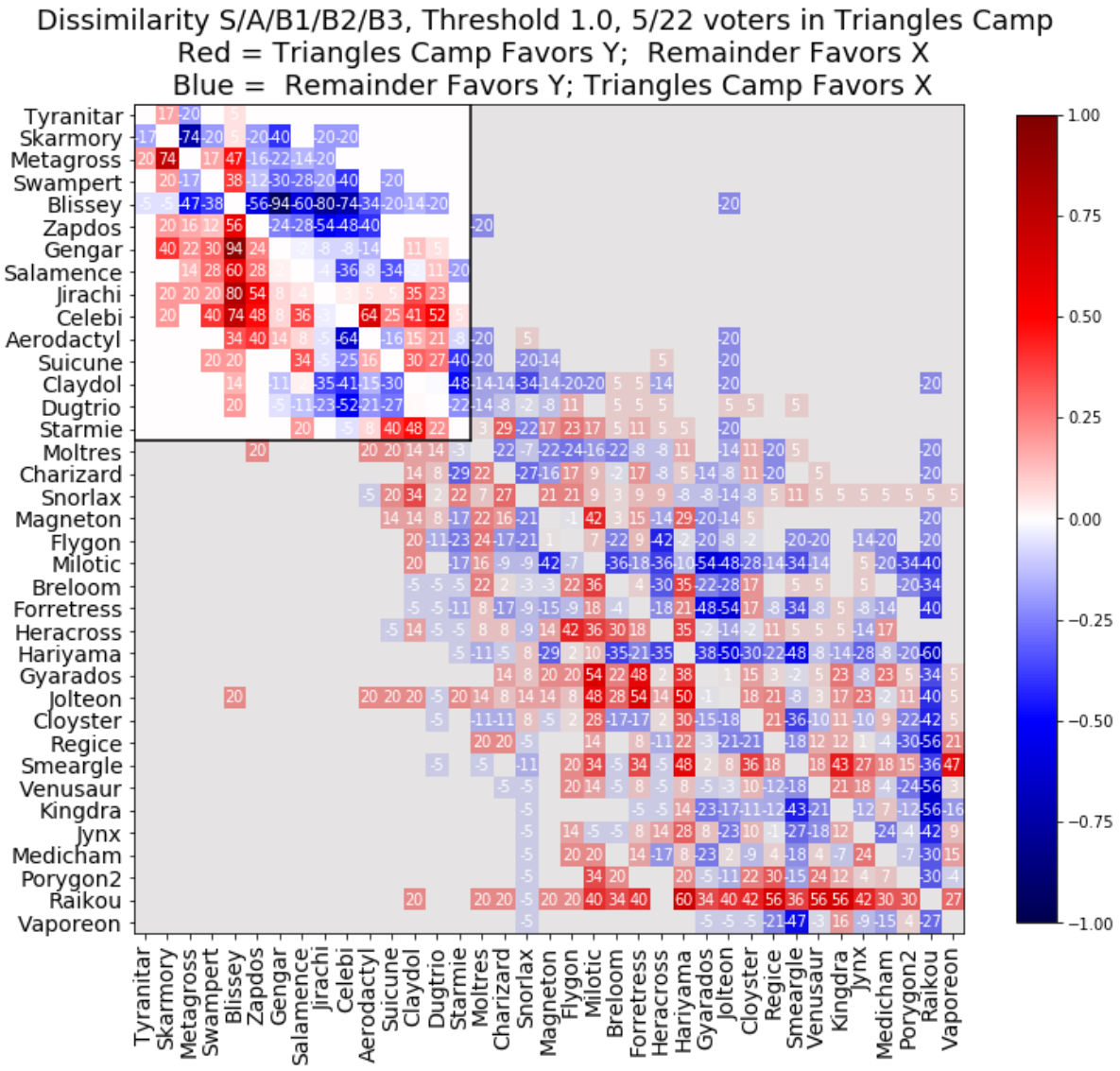

Dissimilarity S/A, Threshold 1.0, 10/22 voters in BKC Camp
Red = BKC Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; BKC Camp Favors X
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

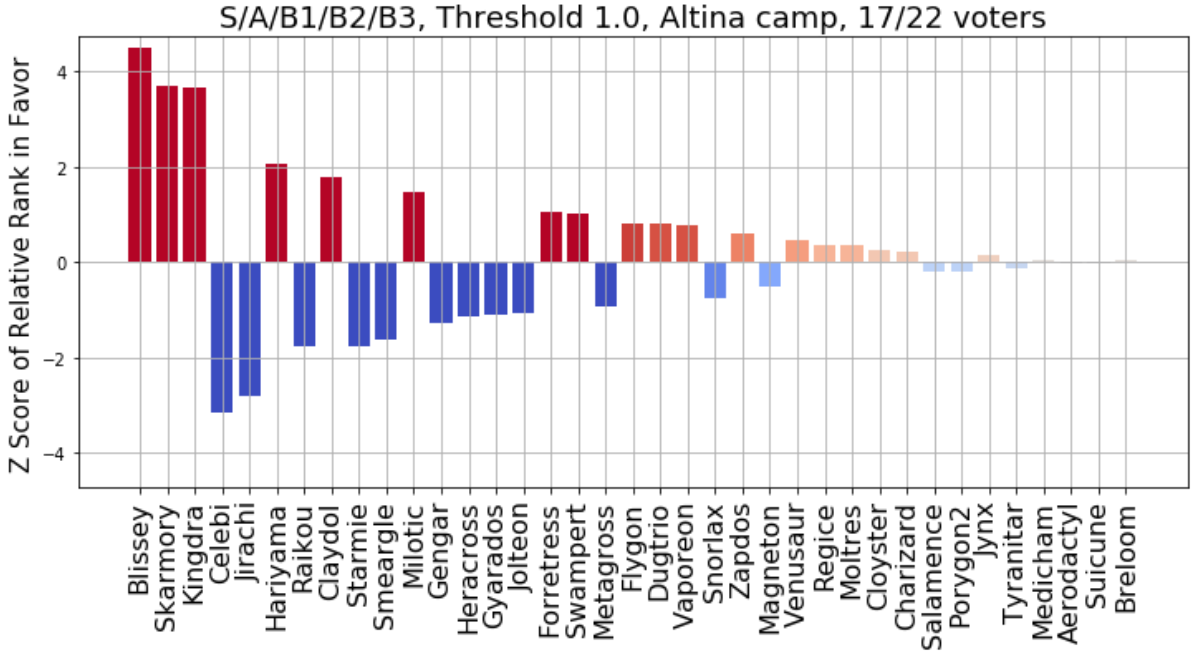

2023 ADV OU VR Relative Statistics: S/A/B1/B2/B3
Relative Rank In Favor
Triangles
Altina
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

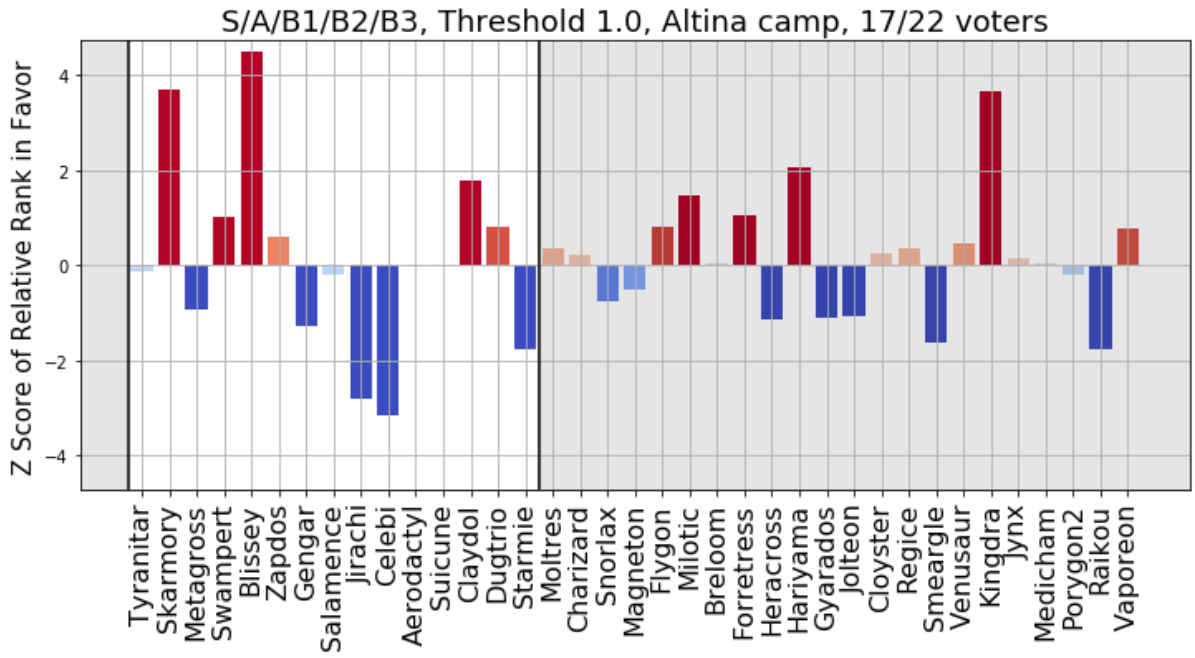

S/A/B1/B2/B3, Tyranitar to Starmie
Triangles 61 50 48 48 25 37 27 58 46 60 58 58 40 33 52 44 50 56 46 34 40
SEA 61 65 48 56 40 33 39 58 65 67 65 73 52 48 63 56 54 60 50 42 52
Fruhdazi 50 65 37 52 44 18 42 58 61 56 61 58 48 44 52 48 39 44 39 34 40
watermess 48 48 37 46 42 54 56 63 56 61 67 63 46 46 54 50 40 50 48 48 58
airports 48 56 52 46 46 39 52 56 52 65 56 56 54 58 65 61 63 65 52 55 50
Altina 25 40 44 42 46 50 56 52 71 50 52 56 65 65 69 69 60 58 63 71 65
Arctic 37 33 18 54 39 50 63 60 52 50 48 52 61 54 58 61 60 61 63 69 77
Jirachee 27 39 42 56 52 56 63 61 65 56 58 58 60 56 60 63 50 56 50 65 75
z0mog 58 58 58 63 56 52 60 61 65 67 84 73 71 67 75 75 69 79 73 67 75
ojr 46 65 61 56 52 71 52 65 65 75 73 84 67 63 79 71 61 67 54 65 75
Siglut 60 67 56 61 65 50 50 56 67 75 79 82 65 61 73 65 63 69 52 51 61
Gacu 58 65 61 67 56 52 48 58 84 73 79 84 71 67 79 75 65 75 65 61 67
Caloom 58 73 58 63 56 56 52 58 73 84 82 84 71 67 82 75 65 75 58 61 71
ABR 40 52 48 46 54 65 61 60 71 67 65 71 71 88 88 88 79 80 71 78 77
Anonymous 33 48 44 46 58 65 54 56 67 63 61 67 67 88 80 84 79 77 67 82 69
baddummy 52 63 52 54 65 69 58 60 75 79 73 79 82 88 80 88 82 84 71 78 77
Star 44 56 48 50 61 69 61 63 75 71 65 75 75 88 84 88 82 88 79 86 84
BKC 50 54 39 40 63 60 60 50 69 61 63 65 65 79 79 82 82 90 77 84 71
Jester 56 60 44 50 65 58 61 56 79 67 69 75 75 80 77 84 88 90 79 78 77
Kristyl 46 50 39 48 52 63 63 50 73 54 52 65 58 71 67 71 79 77 79 75 75
McMeghan 34 42 34 48 55 71 69 65 67 65 51 61 61 78 82 78 86 84 78 75 84
Johnnyg2 40 52 40 58 50 65 77 75 75 75 61 67 71 77 69 77 84 71 77 75 84
Triangles SEA Fruhdazi watermess airports Altina Arctic Jirachee z0mog ojr Siglut Gacu Caloom ABR Anonymous baddummy Star BKC Jester Kristyl McMeghan Johnnyg2
0.9 0.8 0.7 0.6 0.5 0.4 0.3 0.2

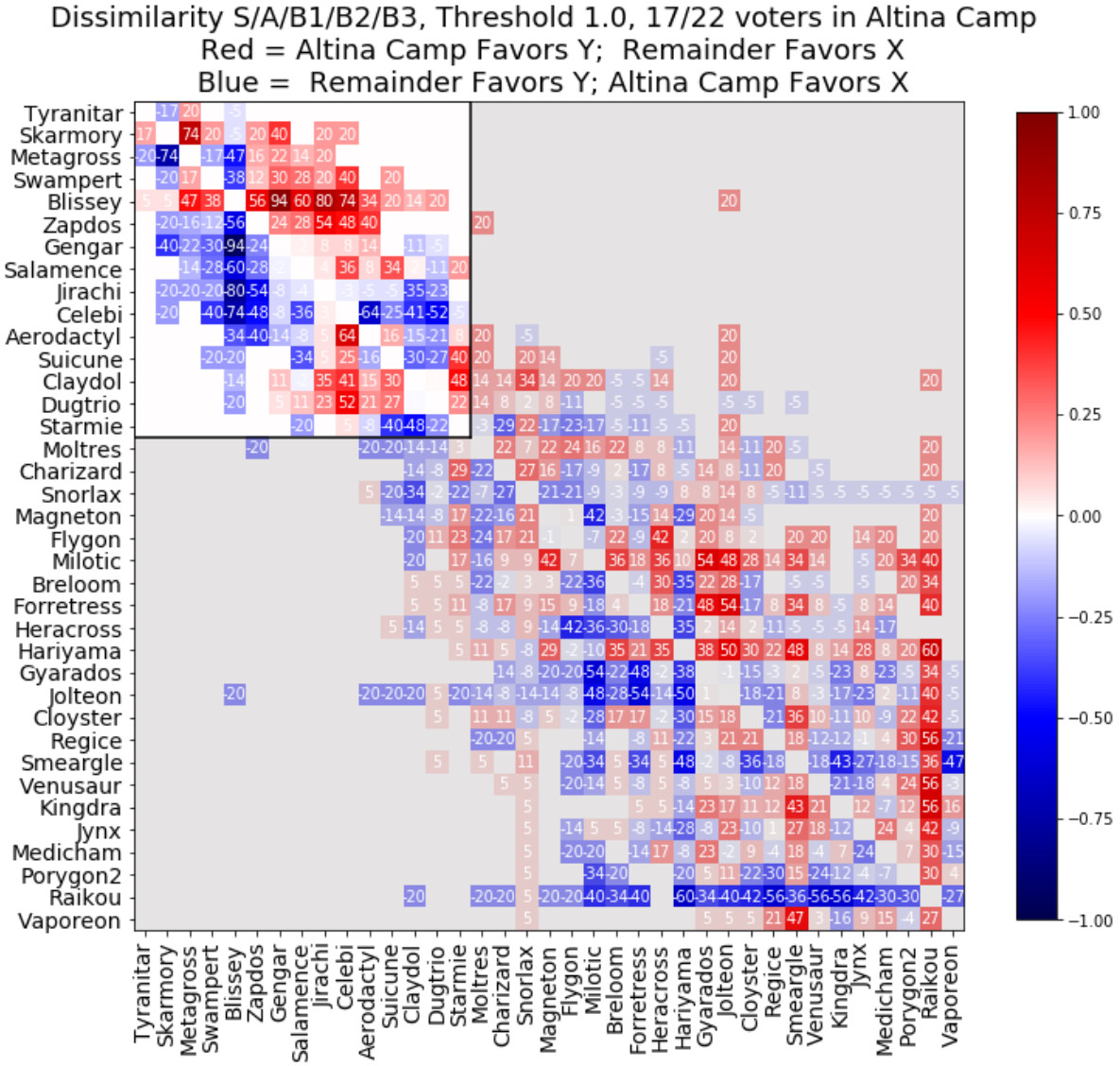

S/A/B1/B2/B3, Threshold 1.0, Triangles camp, 5/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Blissey
Skarmory
Kingdra
Celebi
Jirachi
Hariyama
Raikou
Claydol
Starmie
Smeargle
Milotic
Gengar
Heracross
Gyarados
Jolteon
Forretress
Swampert
Metagross
Flygon
Dugtrio
Vaporeon
Snorlax
Zapdos
Magneton
Venusaur
Regice
Moltres
Cloyster
Charizard
Salamence
Porygon2
Jynx
Tyranitar
Medicham
Aerodactyl
Suicune
Breloom

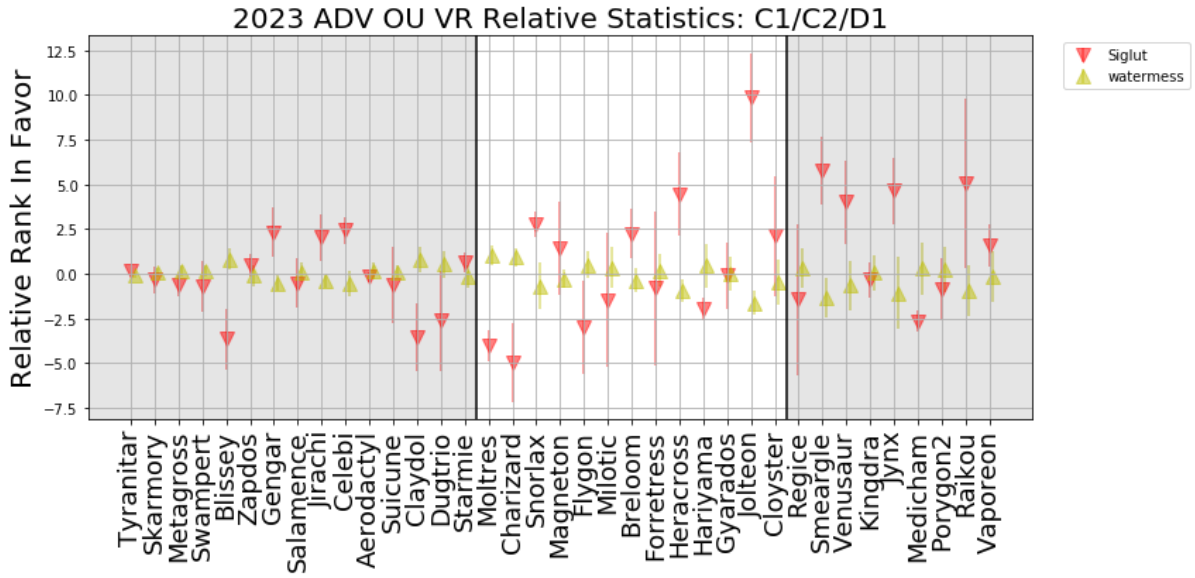

S/A/B1/B2/B3, Threshold 1.0, Triangles camp, 5/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

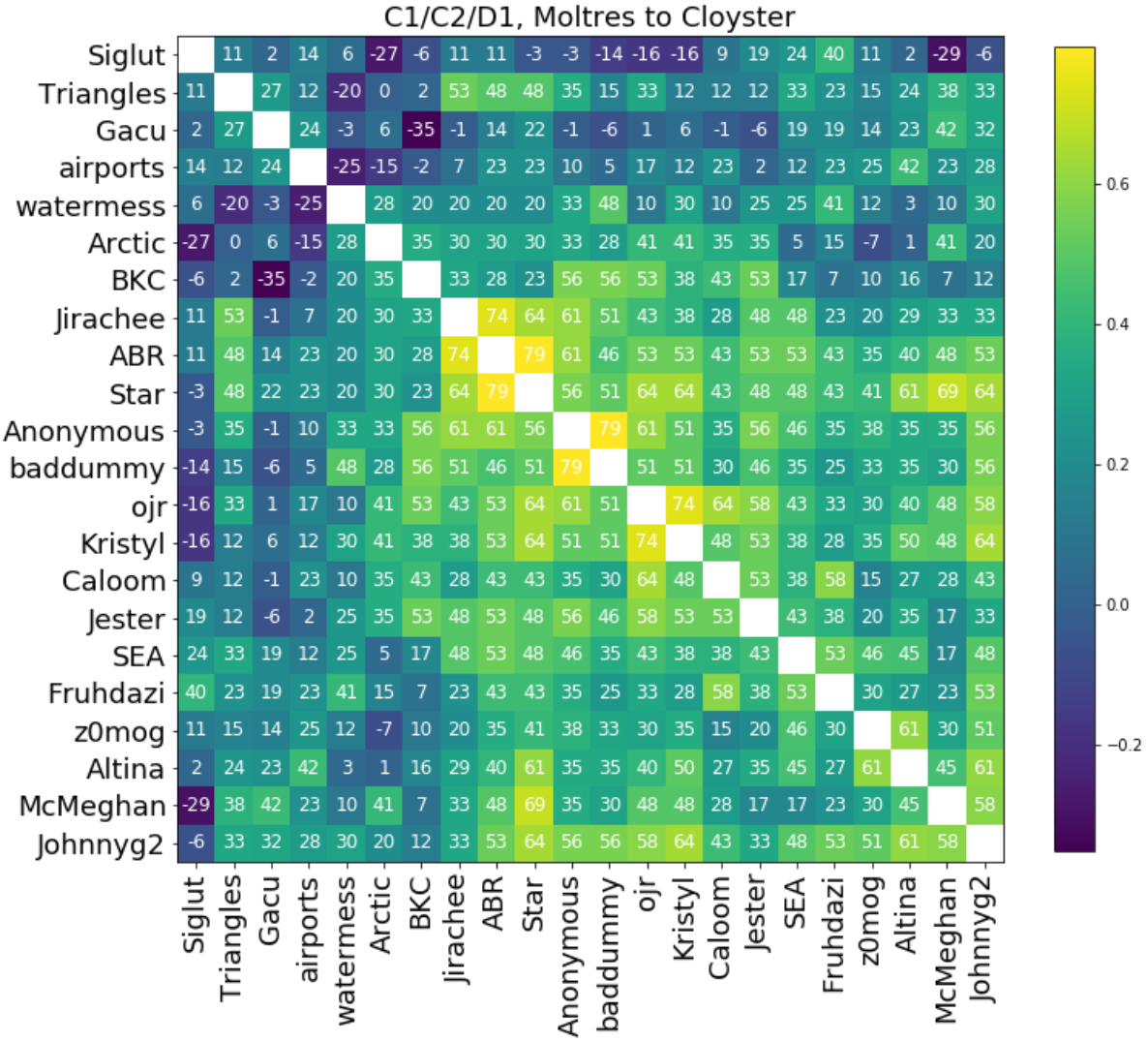

Dissimilarity S/A/B1/B2/B3, Threshold 1.0, 5/22 voters in Triangles Camp
Red = Triangles Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; Triangles Camp Favors X
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

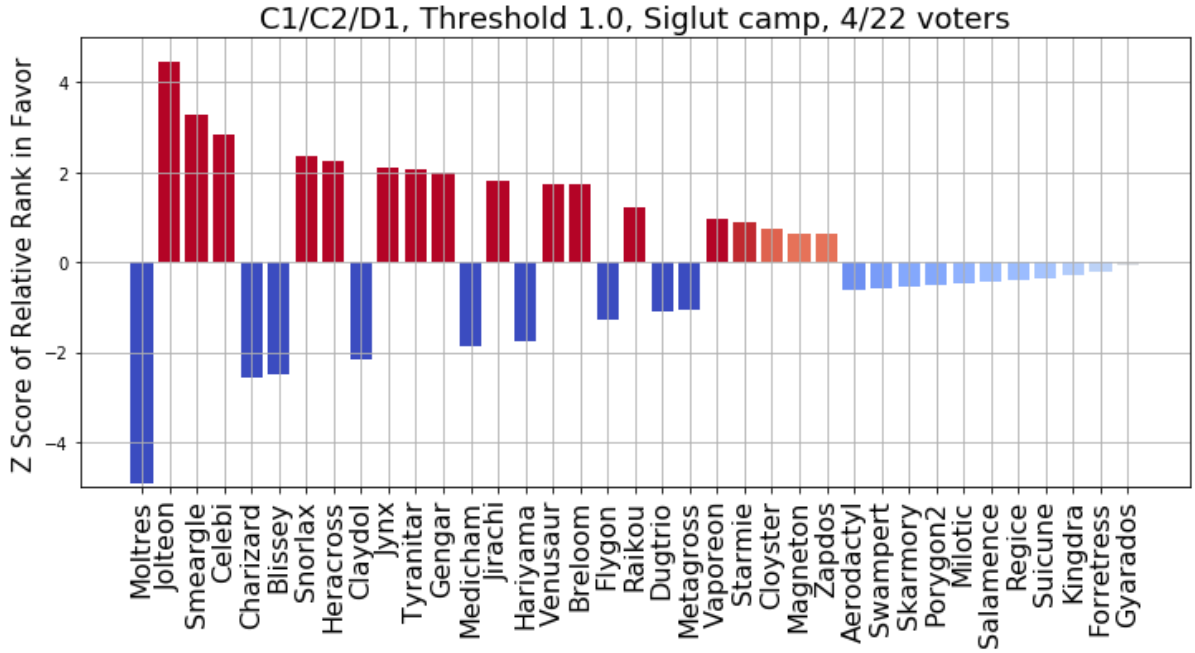

S/A/B1/B2/B3, Threshold 1.0, Altina camp, 17/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Blissey
Skarmory
Kingdra
Celebi
Jirachi
Hariyama
Raikou
Claydol
Starmie
Smeargle
Milotic
Gengar
Heracross
Gyarados
Jolteon
Forretress
Swampert
Metagross
Flygon
Dugtrio
Vaporeon
Snorlax
Zapdos
Magneton
Venusaur
Regice
Moltres
Cloyster
Charizard
Salamence
Porygon2
Jynx
Tyranitar
Medicham
Aerodactyl
Suicune
Breloom

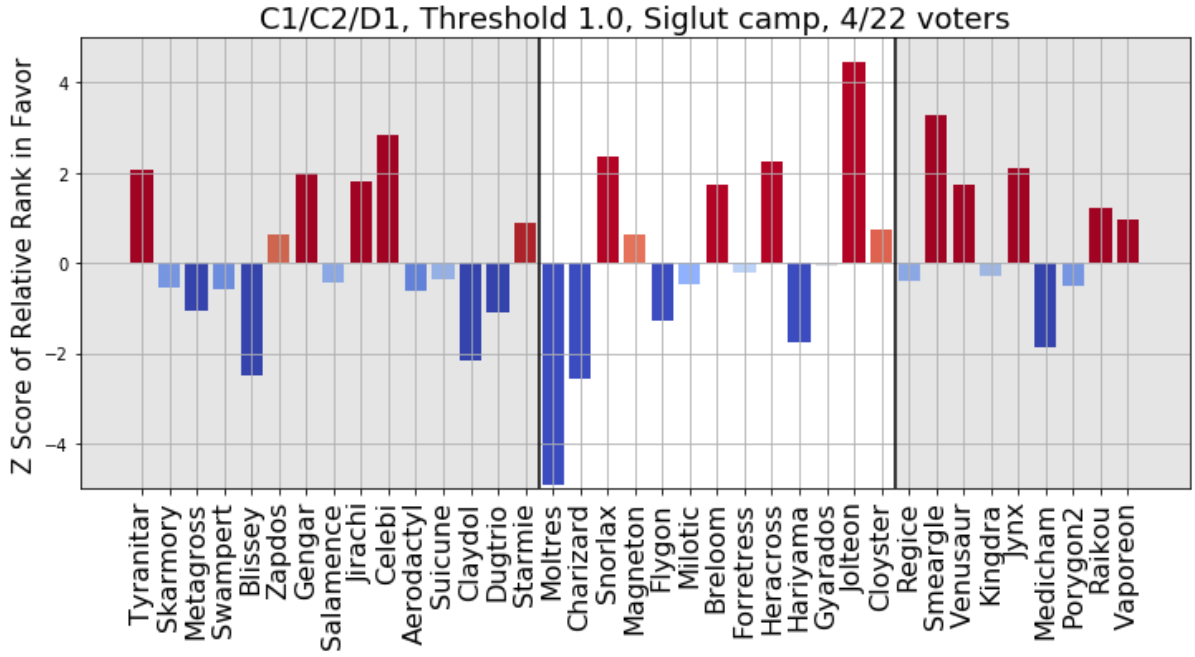

S/A/B1/B2/B3, Threshold 1.0, Altina camp, 17/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

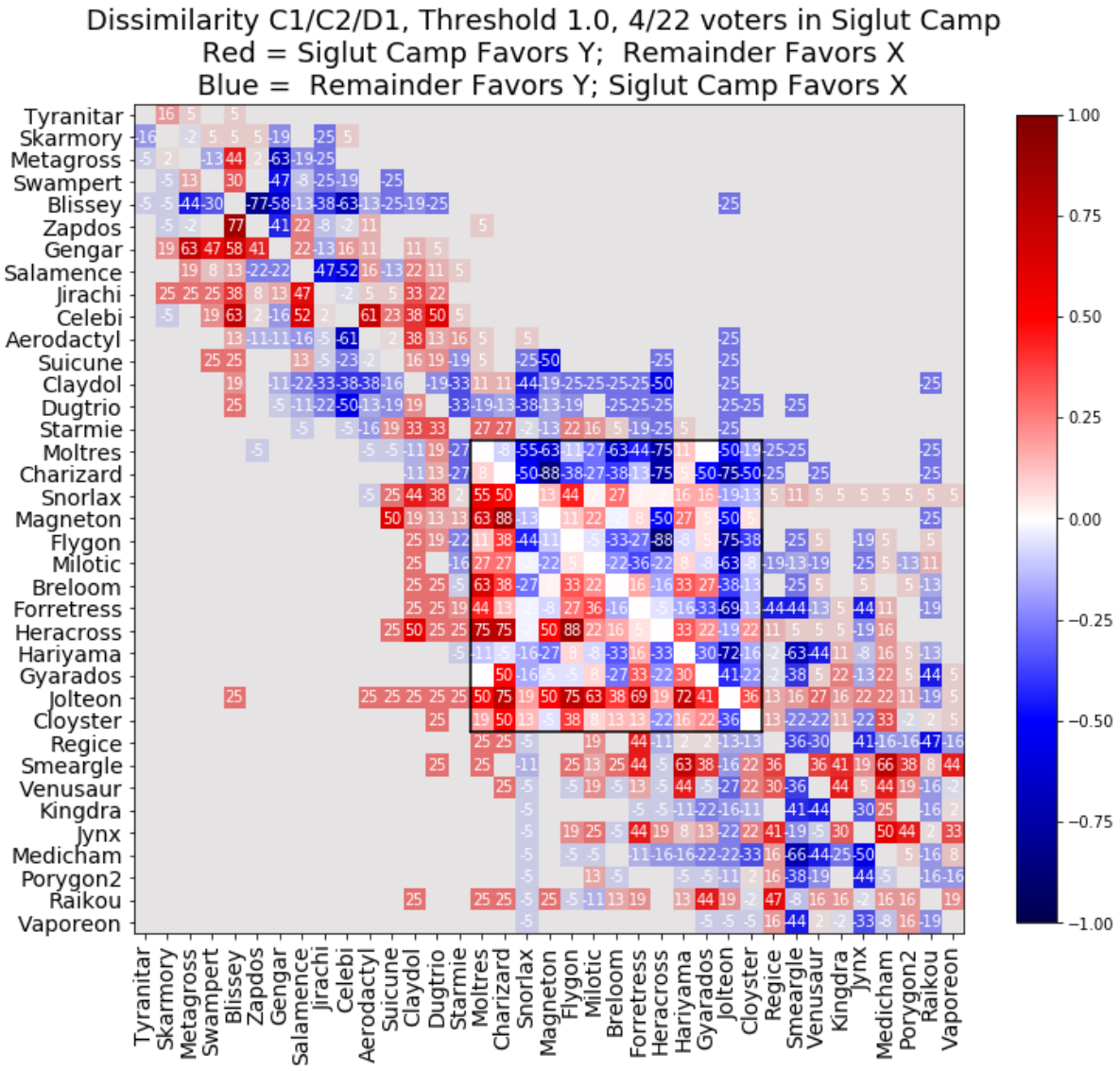

Dissimilarity S/A/B1/B2/B3, Threshold 1.0, 17/22 voters in Altina Camp
Red = Altina Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; Altina Camp Favors X
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

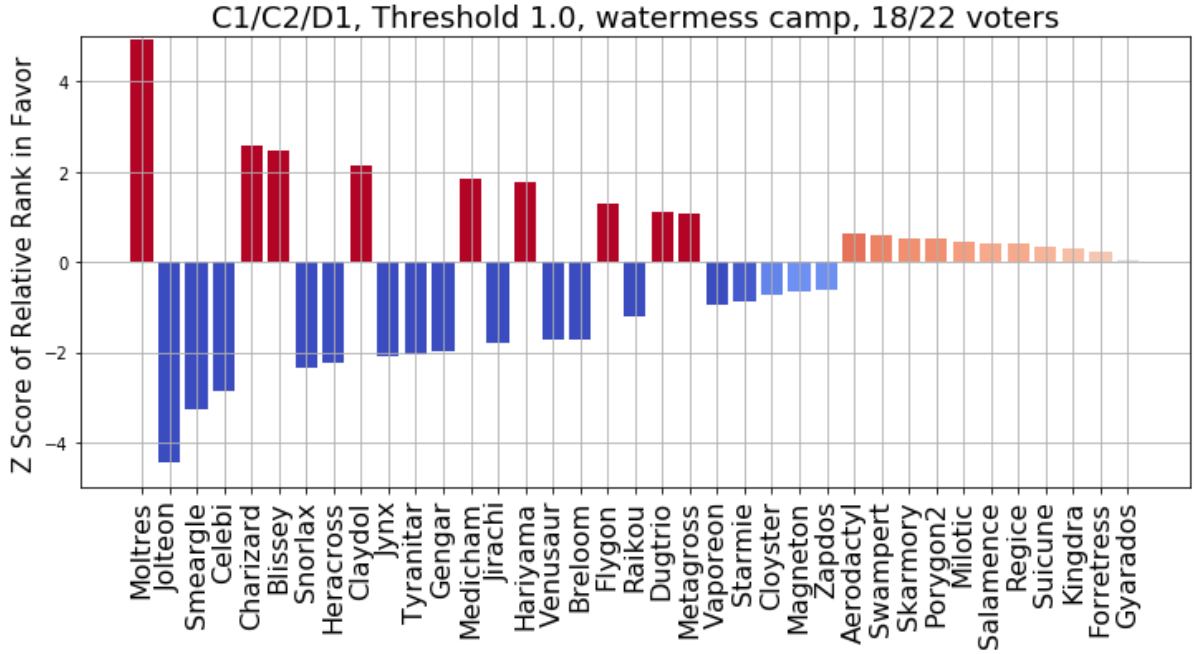

2023 ADV OU VR Relative Statistics: C1/C2/D1
Relative Rank In Favor
Siglut
watermess
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

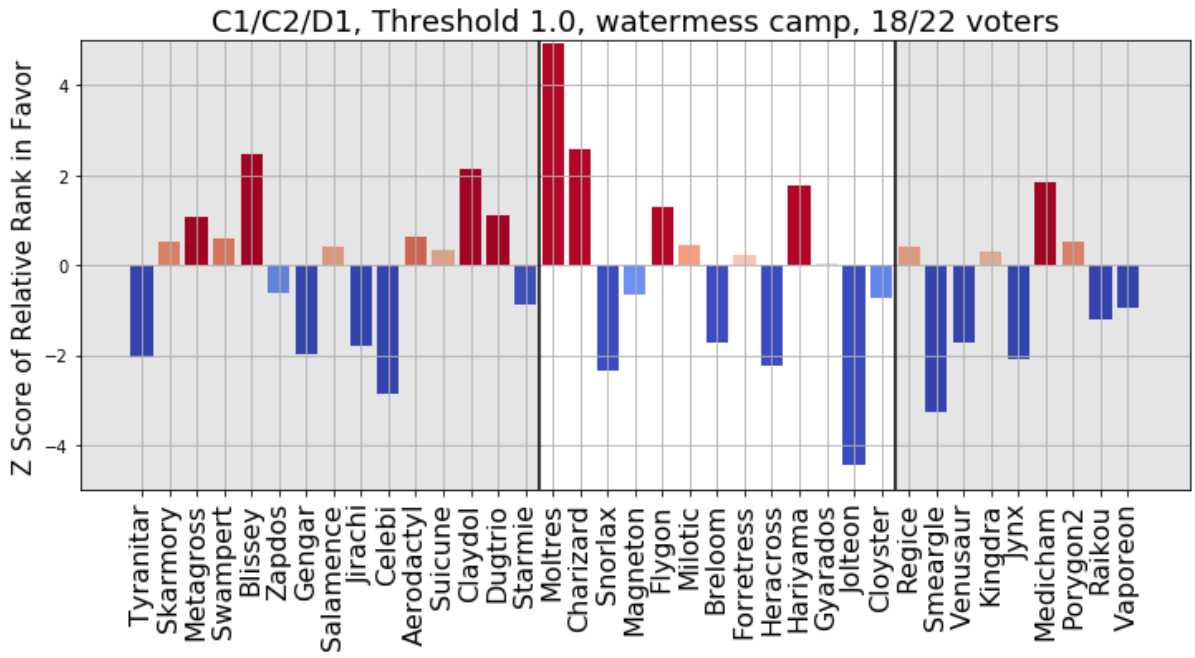

C1/C2/D1, Moltres to Cloyster
Siglut 11 2 14 6 -27 -6 11 11 -3 -3 -14 -16 -16 9 19 24 40 11 2 -29 -6
Triangles 11 27 12 -20 0 2 53 48 48 35 15 33 12 12 12 33 23 15 24 38 33
Gacu 2 27 24 -3 6 -35 -1 14 22 -1 -6 1 6 -1 -6 19 19 14 23 42 32
airports 14 12 24 -25 -15 -2 7 23 23 10 5 17 12 23 2 12 23 25 42 23 28
watermess 6 -20 -3 -25 28 20 20 20 20 33 48 10 30 10 25 25 41 12 3 10 30
Arctic -27 0 6 -15 28 35 30 30 30 33 28 41 41 35 35 5 15 -7 1 41 20
BKC -6 2 -35 -2 20 35 33 28 23 56 56 53 38 43 53 17 7 10 16 7 12
Jirachee 11 53 -1 7 20 30 33 74 64 61 51 43 38 28 48 48 23 20 29 33 33
ABR 11 48 14 23 20 30 28 74 79 61 46 53 53 43 53 53 43 35 40 48 53
Star -3 48 22 23 20 30 23 64 79 56 51 64 64 43 48 48 43 41 61 69 64
Anonymous -3 35 -1 10 33 33 56 61 61 56 79 61 51 35 56 46 35 38 35 35 56
baddummy -14 15 -6 5 48 28 56 51 46 51 79 51 51 30 46 35 25 33 35 30 56
ojr -16 33 1 17 10 41 53 43 53 64 61 51 74 64 58 43 33 30 40 48 58
Kristyl -16 12 6 12 30 41 38 38 53 64 51 51 74 48 53 38 28 35 50 48 64
Caloom 9 12 -1 23 10 35 43 28 43 43 35 30 64 48 53 38 58 15 27 28 43
Jester 19 12 -6 2 25 35 53 48 53 48 56 46 58 53 53 43 38 20 35 17 33
SEA 24 33 19 12 25 5 17 48 53 48 46 35 43 38 38 43 53 46 45 17 48
Fruhdazi 40 23 19 23 41 15 7 23 43 43 35 25 33 28 58 38 53 30 27 23 53
z0mog 11 15 14 25 12 -7 10 20 35 41 38 33 30 35 15 20 46 30 61 30 51
Altina 2 24 23 42 3 1 16 29 40 61 35 35 40 50 27 35 45 27 61 45 61
McMeghan -29 38 42 23 10 41 7 33 48 69 35 30 48 48 28 17 17 23 30 45 58
Johnnyg2 -6 33 32 28 30 20 12 33 53 64 56 56 58 64 43 33 48 53 51 61 58
Siglut Triangles Gacu airports watermess Arctic BKC Jirachee ABR Star Anonymous baddummy ojr Kristyl Caloom Jester SEA Fruhdazi z0mog Altina McMeghan Johnnyg2
0.6
0.4
0.2
0.0
−0.2

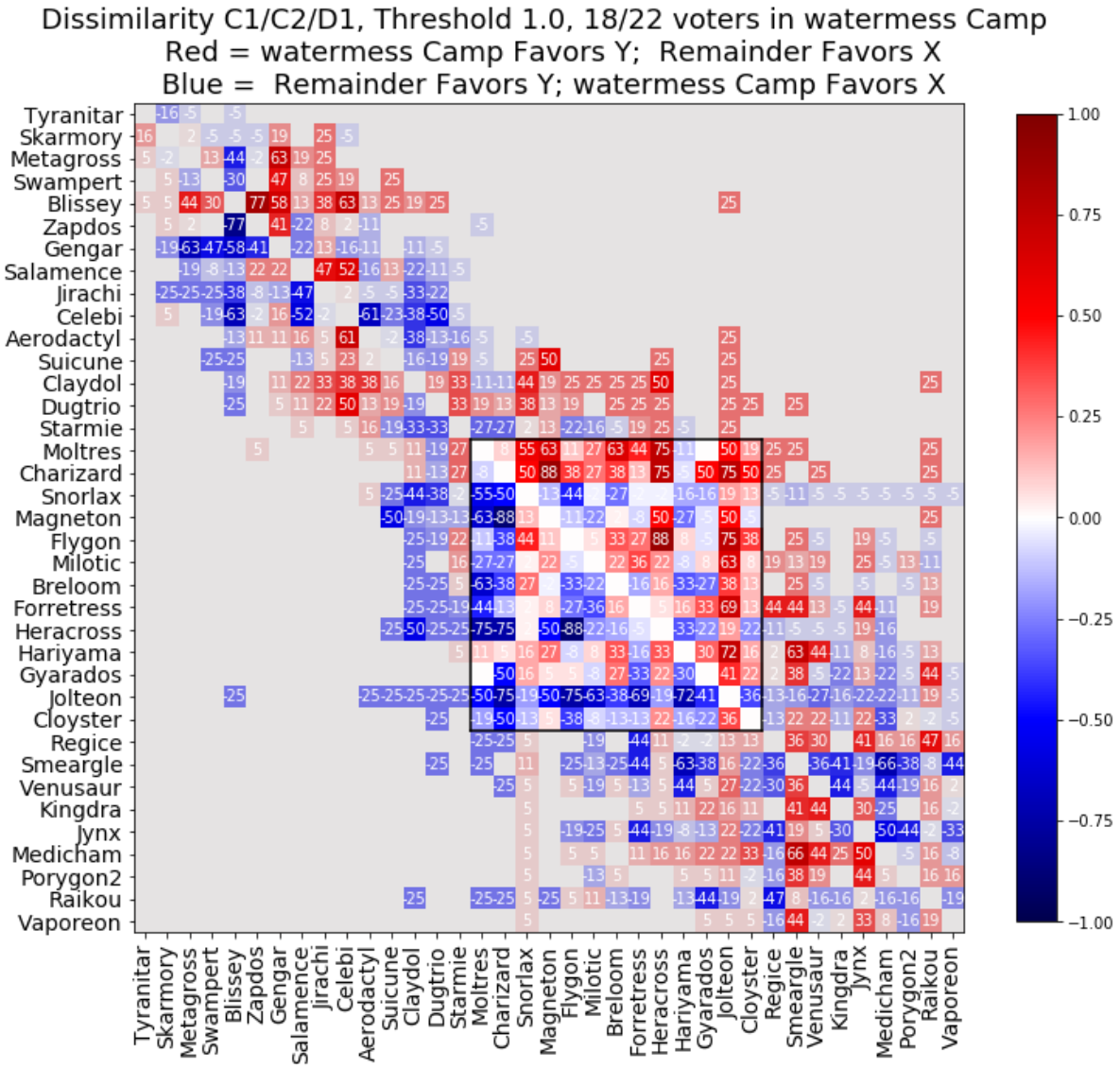

C1/C2/D1, Threshold 1.0, Siglut camp, 4/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Moltres
Jolteon
Smeargle
Celebi
Charizard
Blissey
Snorlax
Heracross
Claydol
Jynx
Tyranitar
Gengar
Medicham
Jirachi
Hariyama
Venusaur
Breloom
Flygon
Raikou
Dugtrio
Metagross
Vaporeon
Starmie
Cloyster
Magneton
Zapdos
Aerodactyl
Swampert
Skarmory
Porygon2
Milotic
Salamence
Regice
Suicune
Kingdra
Forretress
Gyarados

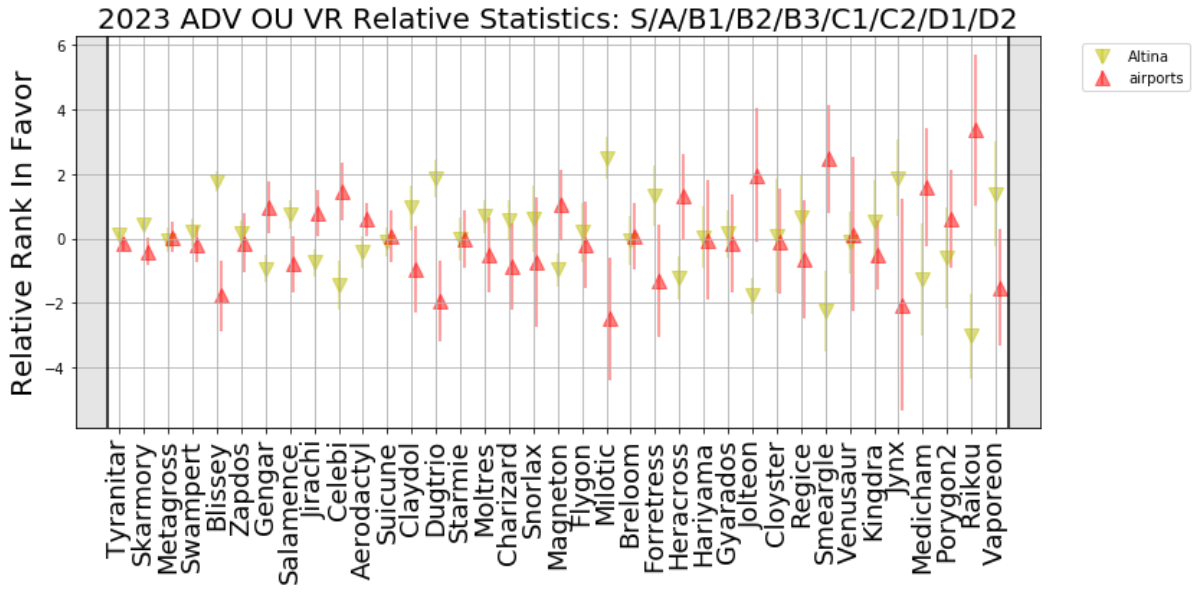

C1/C2/D1, Threshold 1.0, Siglut camp, 4/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

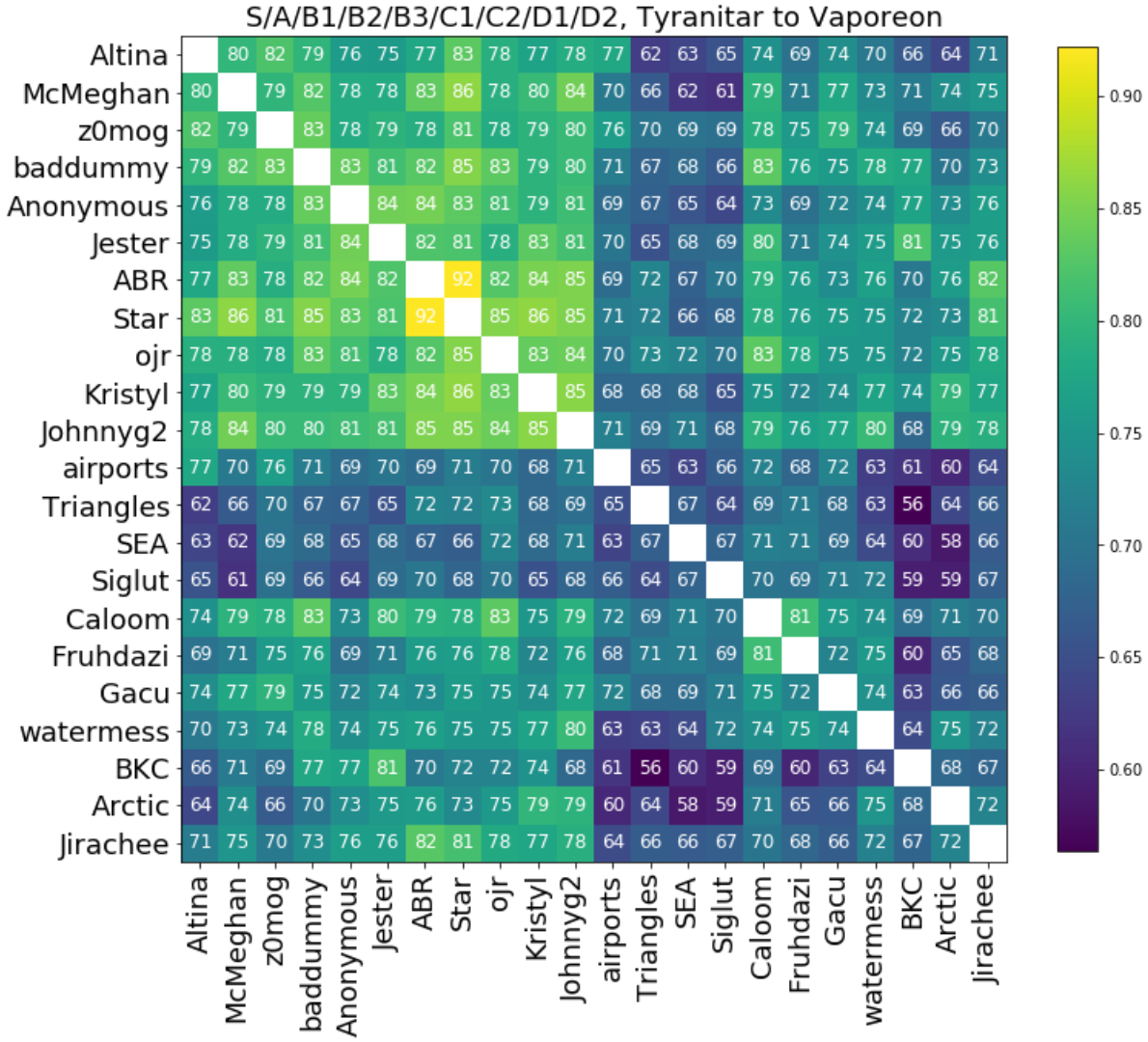

Dissimilarity C1/C2/D1, Threshold 1.0, 4/22 voters in Siglut Camp
Red = Siglut Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; Siglut Camp Favors X
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

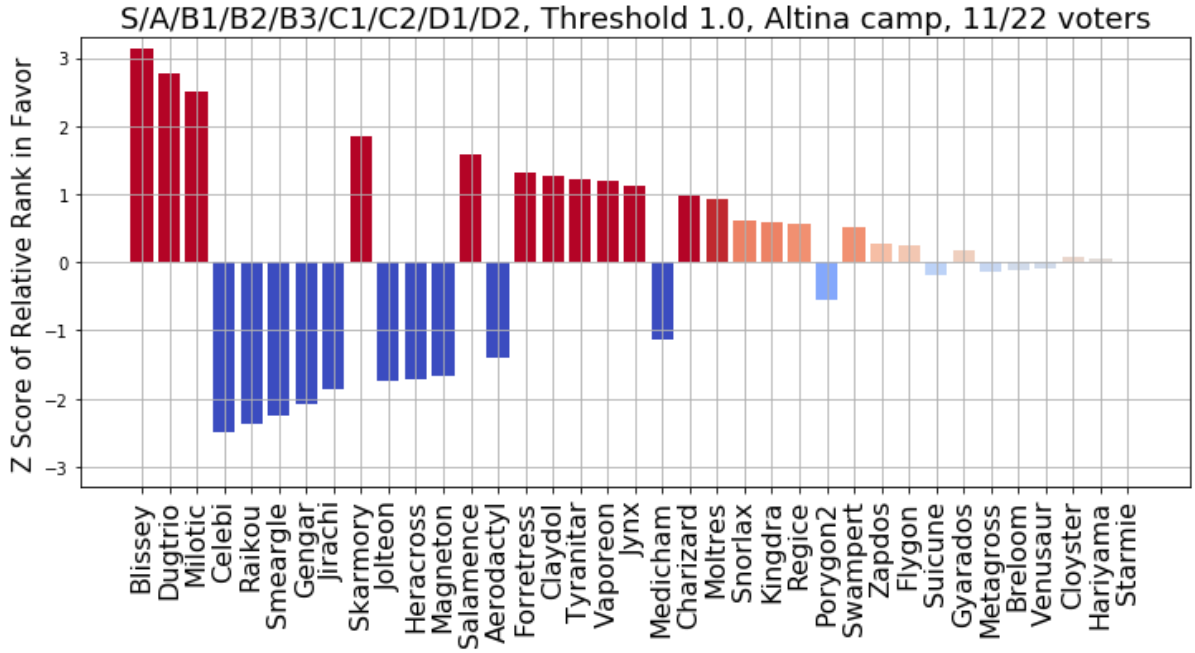

C1/C2/D1, Threshold 1.0, watermess camp, 18/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Moltres
Jolteon
Smeargle
Celebi
Charizard
Blissey
Snorlax
Heracross
Claydol
Jynx
Tyranitar
Gengar
Medicham
Jirachi
Hariyama
Venusaur
Breloom
Flygon
Raikou
Dugtrio
Metagross
Vaporeon
Starmie
Cloyster
Magneton
Zapdos
Aerodactyl
Swampert
Skarmory
Porygon2
Milotic
Salamence
Regice
Suicune
Kingdra
Forretress
Gyarados

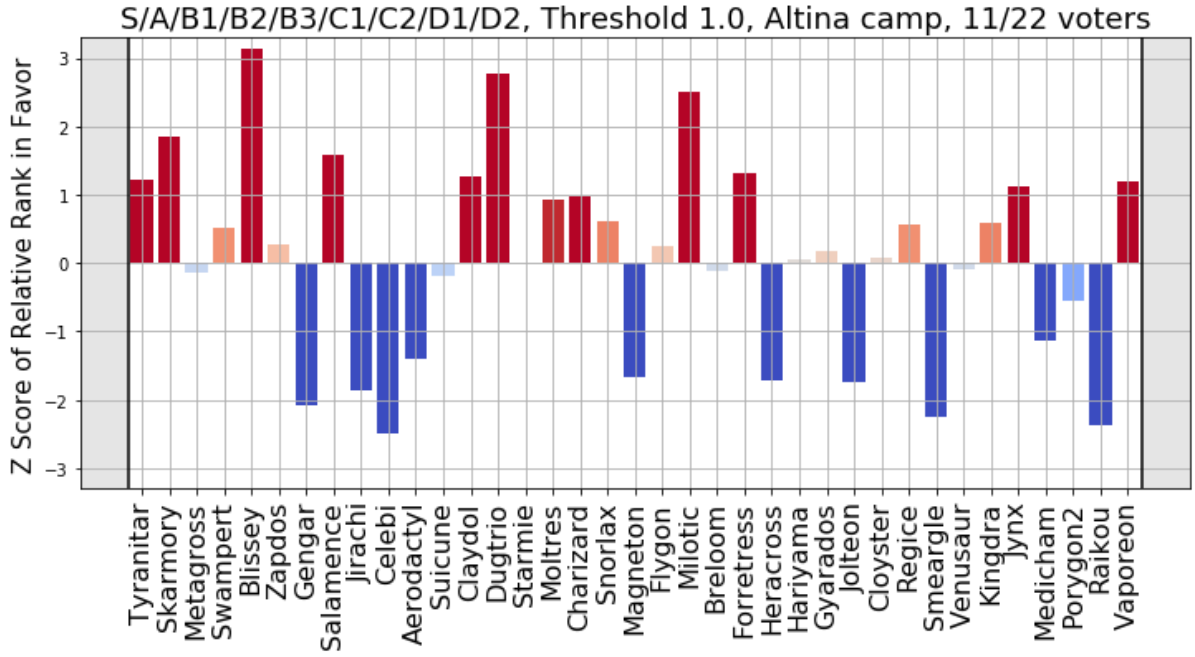

C1/C2/D1, Threshold 1.0, watermess camp, 18/22 voters
Z Score of Relative Rank in Favor
4
2
0
−2
−4
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

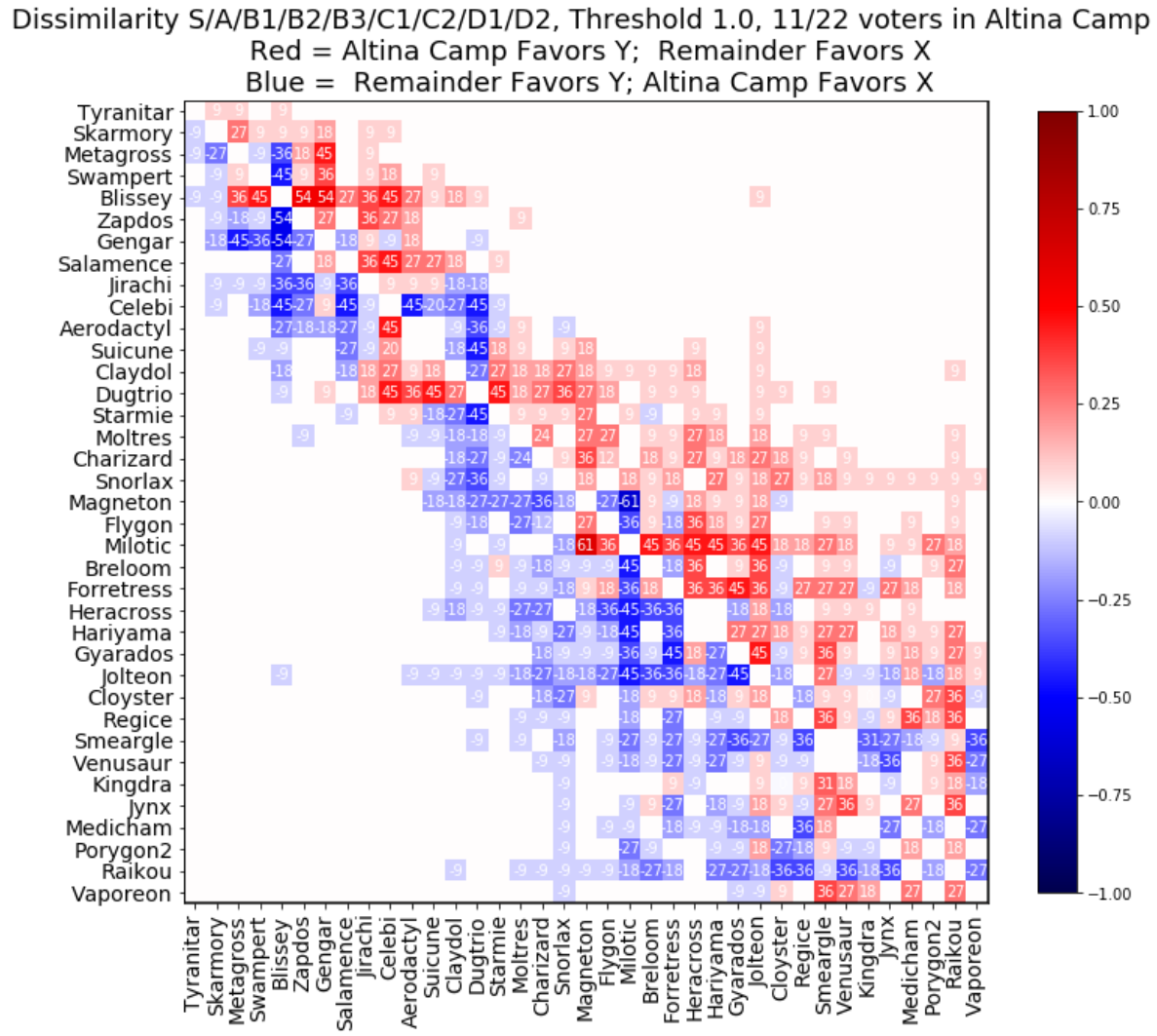

Dissimilarity C1/C2/D1, Threshold 1.0, 18/22 voters in watermess Camp
Red = watermess Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; watermess Camp Favors X
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

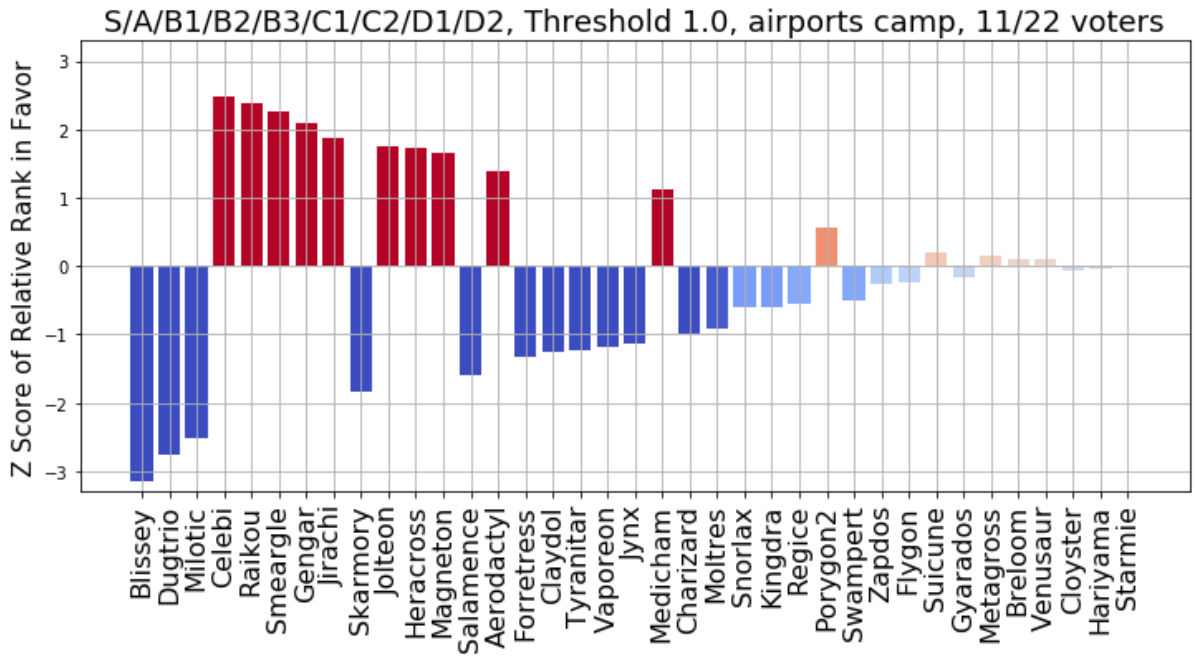

2023 ADV OU VR Relative Statistics: S/A/B1/B2/B3/C1/C2/D1/D2
Relative Rank In Favor
Altina
airports
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

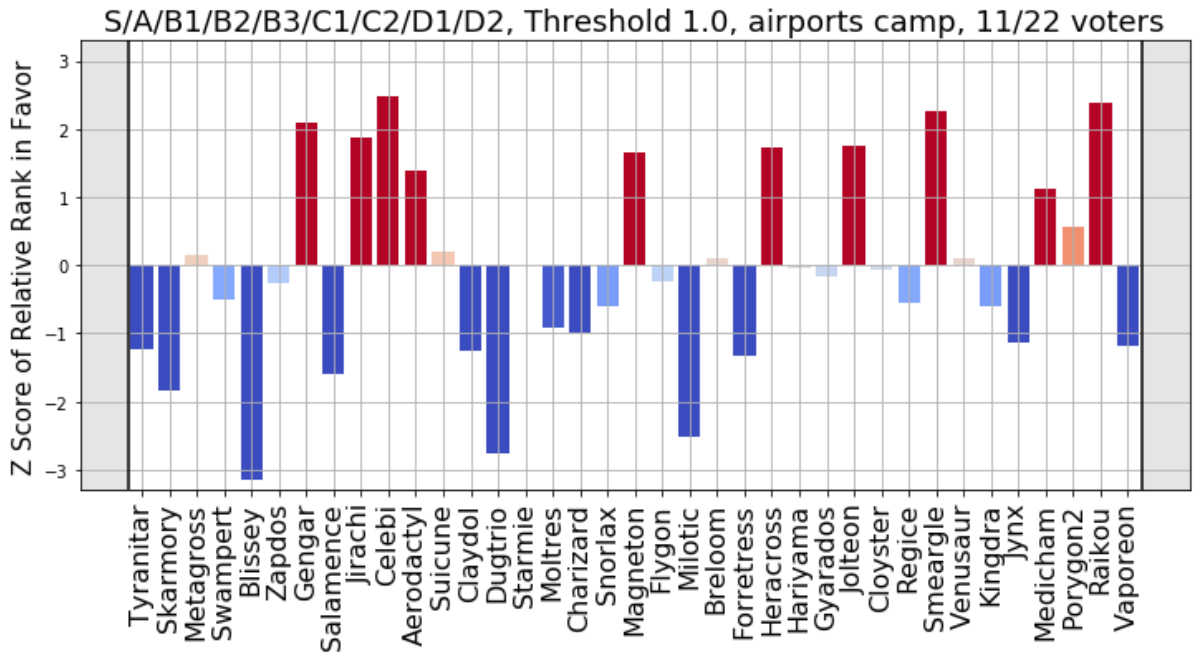

S/A/B1/B2/B3/C1/C2/D1/D2, Tyranitar to Vaporeon
Altina 80 82 79 76 75 77 83 78 77 78 77 62 63 65 74 69 74 70 66 64 71
McMeghan 80 79 82 78 78 83 86 78 80 84 70 66 62 61 79 71 77 73 71 74 75
z0mog 82 79 83 78 79 78 81 78 79 80 76 70 69 69 78 75 79 74 69 66 70
baddummy 79 82 83 83 81 82 85 83 79 80 71 67 68 66 83 76 75 78 77 70 73
Anonymous 76 78 78 83 84 84 83 81 79 81 69 67 65 64 73 69 72 74 77 73 76
Jester 75 78 79 81 84 82 81 78 83 81 70 65 68 69 80 71 74 75 81 75 76
ABR 77 83 78 82 84 82 92 82 84 85 69 72 67 70 79 76 73 76 70 76 82
Star 83 86 81 85 83 81 92 85 86 85 71 72 66 68 78 76 75 75 72 73 81
ojr 78 78 78 83 81 78 82 85 83 84 70 73 72 70 83 78 75 75 72 75 78
Kristyl 77 80 79 79 79 83 84 86 83 85 68 68 68 65 75 72 74 77 74 79 77
Johnnyg2 78 84 80 80 81 81 85 85 84 85 71 69 71 68 79 76 77 80 68 79 78
airports 77 70 76 71 69 70 69 71 70 68 71 65 63 66 72 68 72 63 61 60 64
Triangles 62 66 70 67 67 65 72 72 73 68 69 65 67 64 69 71 68 63 56 64 66
SEA 63 62 69 68 65 68 67 66 72 68 71 63 67 67 71 71 69 64 60 58 66
Siglut 65 61 69 66 64 69 70 68 70 65 68 66 64 67 70 69 71 72 59 59 67
Caloom 74 79 78 83 73 80 79 78 83 75 79 72 69 71 70 81 75 74 69 71 70
Fruhdazi 69 71 75 76 69 71 76 76 78 72 76 68 71 71 69 81 72 75 60 65 68
Gacu 74 77 79 75 72 74 73 75 75 74 77 72 68 69 71 75 72 74 63 66 66
watermess 70 73 74 78 74 75 76 75 75 77 80 63 63 64 72 74 75 74 64 75 72
BKC 66 71 69 77 77 81 70 72 72 74 68 61 56 60 59 69 60 63 64 68 67
Arctic 64 74 66 70 73 75 76 73 75 79 79 60 64 58 59 71 65 66 75 68 72
Jirachee 71 75 70 73 76 76 82 81 78 77 78 64 66 66 67 70 68 66 72 67 72
Altina McMeghan z0mog baddummy Anonymous Jester ABR Star ojr Kristyl Johnnyg2 airports Triangles SEA Siglut Caloom Fruhdazi Gacu watermess BKC Arctic Jirachee
0.90
0.85
0.80
0.75
0.70
0.65
0.60

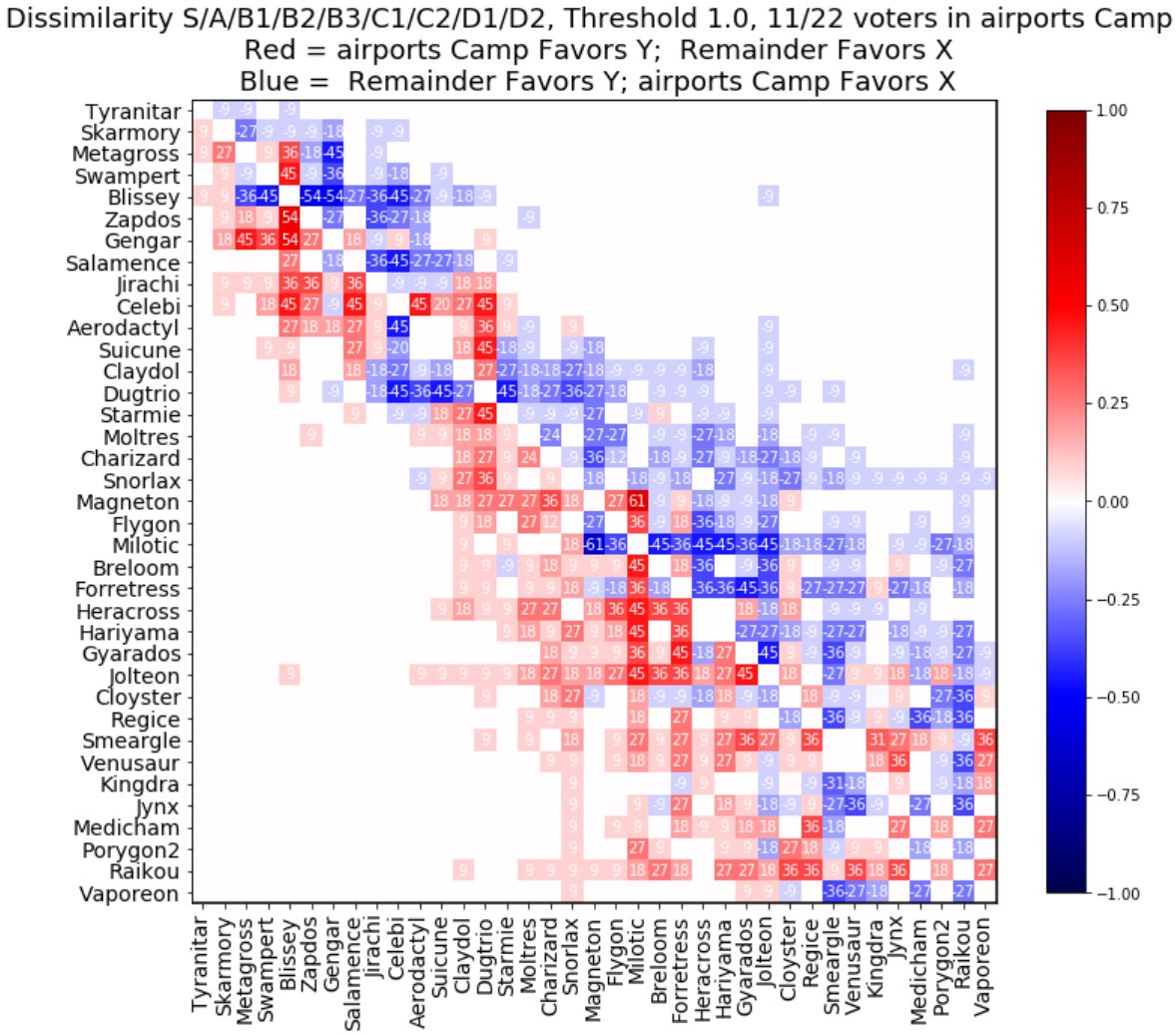

S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, Altina camp, 11/22 voters
Z Score of Relative Rank in Favor
3
2
1
0
−1
−2
−3
Blissey
Dugtrio
Milotic
Celebi
Raikou
Smeargle
Gengar
Jirachi
Skarmory
Jolteon
Heracross
Magneton
Salamence
Aerodactyl
Forretress
Claydol
Tyranitar
Vaporeon
Jynx
Medicham
Charizard
Moltres
Snorlax
Kingdra
Regice
Porygon2
Swampert
Zapdos
Flygon
Suicune
Gyarados
Metagross
Breloom
Venusaur
Cloyster
Hariyama
Starmie


S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, Altina camp, 11/22 voters
Z Score of Relative Rank in Favor
3
2
1
0
−1
−2
−3
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon


Dissimilarity S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, 11/22 voters in Altina Camp
Red = Altina Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; Altina Camp Favors X
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

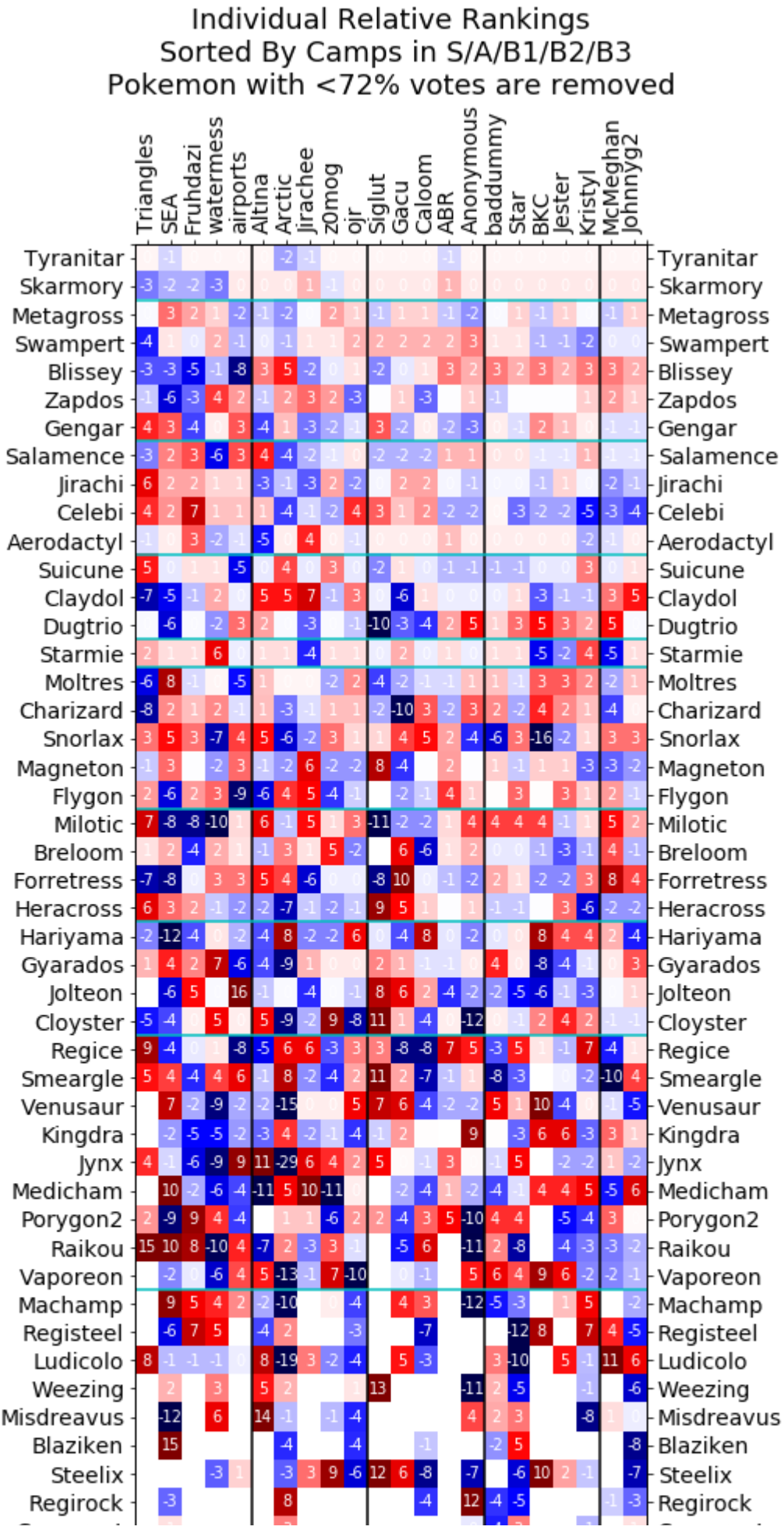

S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, airports camp, 11/22 voters
Z Score of Relative Rank in Favor
3
2
1
0
−1
−2
−3
Blissey
Dugtrio
Milotic
Celebi
Raikou
Smeargle
Gengar
Jirachi
Skarmory
Jolteon
Heracross
Magneton
Salamence
Aerodactyl
Forretress
Claydol
Tyranitar
Vaporeon
Jynx
Medicham
Charizard
Moltres
Snorlax
Kingdra
Regice
Porygon2
Swampert
Zapdos
Flygon
Suicune
Gyarados
Metagross
Breloom
Venusaur
Cloyster
Hariyama
Starmie


S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, airports camp, 11/22 voters
Z Score of Relative Rank in Favor
3
2
1
0
−1
−2
−3
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon

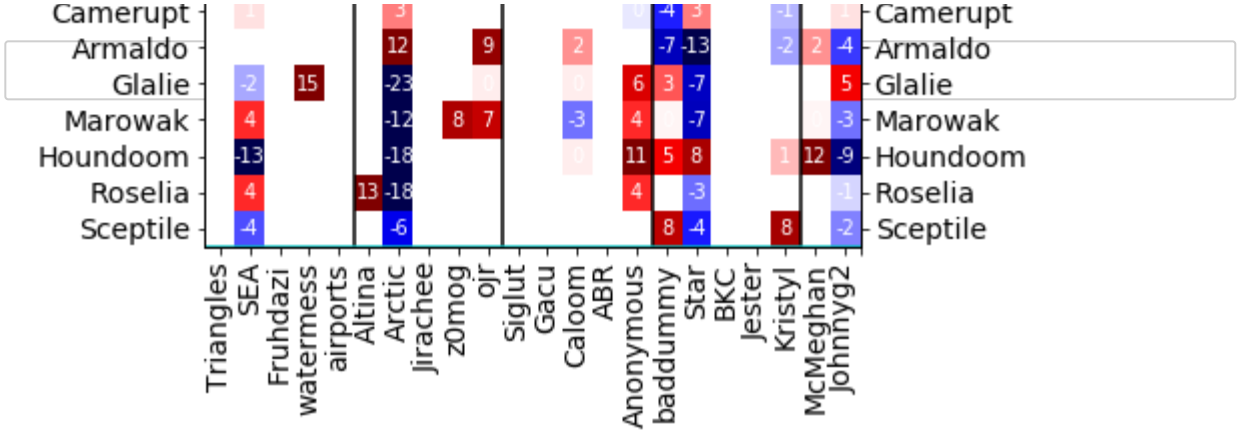

Dissimilarity S/A/B1/B2/B3/C1/C2/D1/D2, Threshold 1.0, 11/22 voters in airports Camp
Red = airports Camp Favors Y; Remainder Favors X
Blue = Remainder Favors Y; airports Camp Favors X
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

In [65]:
```
### Individual Relative Ranks

# Define range for clustering
refRange = 1
# Order voters by camps
pair = segments[refRange]
order = Dlist[refRange]['leaves']
namesInd = [names[i] for i in order]

## Option 1 plots relative rank wrt mean rank
rankFinalSubtractedOrder = rankMeanSubtracted[:,order]

## Option 2 plots relative rank wrt integer rank
# rankFinalSubtracted = rankval - np.repeat(np.array([np.arange(1,N[0]+1)]).tr
anspose(),rankval.shape[1],axis=1)
# rankFinalSubtractedOrder = rankFinalSubtracted[:,order]

# Eliminate underranked Pokemon
rankFinalSubtractedOrder = rankFinalSubtractedOrder[orderReduced,:]
Nr = np.shape(rankFinalSubtractedOrder)
monListReduced = monList[orderReduced]

fig = plt.figure(figsize=(Nr[1]/3,Nr[0]/3))
ax = plt.gca()
h = ax.imshow(-rankFinalSubtractedOrder,interpolation='none',cmap='seismic',vm
in=-10,vmax=10)

fig.colorbar(h,ax=ax,shrink=0.8)
title = 'Individual Relative Rankings\n Sorted By Camps in '
title += tierNames[pair[0]]
for m in range(pair[0]+1,pair[1]):
    title += '/' + tierNames[m]
title += '\nPokemon with <' + str(int(100*rankFracThreshold)) + '% votes are r
emoved'
ax.set_title(title,fontsize=18)
ax.set_xticks(range(0,Nr[1]))
ax.set_yticks(range(0,Nr[0]))

ax.get_xaxis().set_visible(False)
ax1 = ax.twiny()
ax1.xaxis.tick_top()
ax2 = ax.twiny()
ax2.xaxis.tick_bottom()

ax.get_yaxis().set_visible(False)
ax3 = ax.twinx()
ax3.yaxis.tick_left()
ax4 = ax.twinx()
ax4.yaxis.tick_right()

ax1.set_xticks(np.append(np.arange(0.5,Nr[1]),Nr[1]))
ax2.set_xticks(np.append(np.arange(0.5,Nr[1]),Nr[1]))
ax3.set_yticks(np.append(Nr[0],np.arange(Nr[0]-0.5,0,-1)))
ax4.set_yticks(np.append(Nr[0],np.arange(Nr[0]-0.5,0,-1)))
ax1.set_xticklabels([namesInd[i] for i in range(Nr[1])],fontsize=520/cutoff)
ax2.set_xticklabels([namesInd[i] for i in range(Nr[1])],fontsize=520/cutoff)
```

```
ax3.set_yticklabels(['']+[monListReduced[i] for i in range(Nr[0])],fontsize=52
0/cutoff)
ax4.set_yticklabels(['']+[monListReduced[i] for i in range(Nr[0])],fontsize=52
0/cutoff)
plt.setp(ax1.get_xticklabels(), rotation=90, ha="left",va="center",rotation_mo
de="anchor");
plt.setp(ax2.get_xticklabels(), rotation=90, ha="right",va="center",rotation_m
ode="anchor");
for ii in range(0,Nr[0]):
    for jj in range(0,Nr[1]):
        if not np.isnan(-rankFinalSubtractedOrder[ii,jj]):
            text = ax.text(jj, ii, int(np.round(-rankFinalSubtractedOrder[ii,j
j])),
                           ha="center", va="center", color="w",fontsize=360/cu
toff)

ax.set_aspect('auto')
for n in np.arange(1,Nr[1]/5):
    ax.plot([n*5-0.5,n*5-0.5],[-0.5,ax.get_ylim()[0]],color='k')

tempBool = np.zeros(N[0])
tempBool[clusterIndicesMod[clusterIndicesMod!=N[0]]] = 1
tempBool = tempBool[orderReduced]
clusterIndicesModReduced = np.nonzero(tempBool)
clusterIndicesModReduced = np.append(clusterIndicesModReduced,Nr[0])

for n in clusterIndicesModReduced:
    ax.plot([-0.5,ax.get_xlim()[1]],[n-0.5,n-0.5],color='c')
plt.delaxes(fig.axes[1])
```


Individual Relative Rankings
Sorted By Camps in S/A/B1/B2/B3
Pokemon with <72% votes are removed
Triangles
SEA
Fruhdazi
watermess
airports
Altina
Arctic
Jirachee
z0mog
ojr
Siglut
Gacu
Caloom
ABR
Anonymous
baddummy
Star
BKC
Jester
Kristyl
McMeghan
Johnnyg2
Tyranitar
Skarmory
Metagross
Swampert
Blissey
Zapdos
Gengar
Salamence
Jirachi
Celebi
Aerodactyl
Suicune
Claydol
Dugtrio
Starmie
Moltres
Charizard
Snorlax
Magneton
Flygon
Milotic
Breloom
Forretress
Heracross
Hariyama
Gyarados
Jolteon
Cloyster
Regice
Smeargle
Venusaur
Kingdra
Jynx
Medicham
Porygon2
Raikou
Vaporeon
Machamp
Registeel
Ludicolo
Weezing
Misdreavus
Blaziken
Steelix
Regirock

In [ ]:


Camerupt
Armaldo
Glalie
Marowak
Houndoom
Roselia
Sceptile
Triangles
SEA
Fruhdazi
watermess
airports
Altina
Arctic
Jirachee
z0mog
ojr
Siglut
Gacu
Caloom
ABR
Anonymous
baddummy
Star
BKC
Jester
Kristyl
McMeghan
Johnnyg2