Taemin Huh DATA1030 Midterm Project

October 21, 2022

1 DATA1030 Project: Pokemon Battle Analysis

1.1 Exploratory Data Analysis

```
[1]: import pandas as pd
     import numpy as np
     from matplotlib import pylab as plt

     pkmn = pd.read_csv(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh␣
      ↪DATA1030 Project\data\pokemon.csv")
     pkmn = pkmn.rename(index=str, columns={"#": "ID"})
     battle = pd.read_csv(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh␣
      ↪DATA1030 Project\data\combats.csv")
     pkmn.head()
```

```
[1]:    ID           Name Type 1  Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  \
     0   1      Bulbasaur  Grass  Poison  45      49       49       65       65
     1   2        Ivysaur  Grass  Poison  60      62       63       80       80
     2   3       Venusaur  Grass  Poison  80      82       83      100      100
     3   4  Mega Venusaur  Grass  Poison  80     100      123      122      120
     4   5     Charmander   Fire     NaN  39      52       43       60       50

        Speed  Generation  Legendary
     0     45           1      False
     1     60           1      False
     2     80           1      False
     3     80           1      False
     4     65           1      False
```

```
[2]: battle.head()
```

```
[2]:    First_pokemon  Second_pokemon  Winner
     0            266             298     298
     1            702             701     701
     2            191             668     668
     3            237             683     683
     4            151             231     151
```

```
[3]: print("Shape of Pokemon data: ", (pkmn.shape))
     print("Shape of Battle data: ", (battle.shape))
```

```
Shape of Pokemon data:  (800, 12)
Shape of Battle data:  (50000, 3)
```

```
[4]: # Identifying missing values
     pkmn.isnull().sum()
```

```
[4]: ID              0
     Name            1
     Type 1          0
     Type 2        386
     HP              0
     Attack          0
     Defense         0
     Sp. Atk         0
     Sp. Def         0
     Speed           0
     Generation      0
     Legendary       0
     dtype: int64
```

```
[5]: battle.isnull().sum()
```

```
[5]: First_pokemon     0
     Second_pokemon    0
     Winner            0
     dtype: int64
```

```
[6]: # Singling out Pokemone with missing name
     pkmn[pkmn['Name'].isnull()]
```

```
[6]:     ID Name    Type 1 Type 2  HP  Attack  Defense  Sp. Atk  Sp. Def  Speed  \
     62  63  NaN  Fighting    NaN  65     105       60       60       70     95

         Generation  Legendary
     62           1      False
```

```
[7]: # By looking at the online Pokemon database (https://www.serebii.net/pokemon/
      ↪type/fighting/), able to match the missing Pokemon name to "Primeape", which␣
      ↪has exactly the above stat profile among the pool of 7 pure Fighting-type (i.
      ↪e. no Type 2) Pokemons in Generation 1.
     pkmn['Name'][62] = "Primeape"
```

```
C:\Users\User\AppData\Local\Temp\ipykernel_27768\2549842310.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  pkmn['Name'][62] = "Primeape"
```

[8]:
```python
# Feature engineering & merging datasets
winTotal = battle.Winner.value_counts()
winTotal = winTotal.sort_index()
firstAtk_Count = battle.First_pokemon.value_counts()
secondAtk_Count = battle.Second_pokemon.value_counts()
battleTotal = firstAtk_Count + secondAtk_Count
```

[9]:
```python
featEng = pd.DataFrame()
featEng['Total Win Count'] = winTotal
featEng['Total Battle Count'] = battleTotal
# featEng['First Attack Count'] = firstAtk_Count
# featEng['First Attack Rate'] = firstAtk_Count/battleTotal
featEng['Win Rate']= winTotal/battleTotal
pkmnWR = pd.merge(pkmn, featEng, right_index = True, left_on='ID')
pkmnWR
```

[9]:
```
      ID            Name   Type 1  Type 2  HP  Attack  Defense  Sp. Atk  \
0      1       Bulbasaur    Grass  Poison  45      49       49       65
1      2         Ivysaur    Grass  Poison  60      62       63       80
2      3        Venusaur    Grass  Poison  80      82       83      100
3      4   Mega Venusaur    Grass  Poison  80     100      123      122
4      5      Charmander     Fire     NaN  39      52       43       60
..   …               …      …       …  ..    …       …        …
795  796         Diancie     Rock   Fairy  50     100      150      100
796  797    Mega Diancie     Rock   Fairy  50     160      110      160
797  798  Hoopa Confined  Psychic   Ghost  80     110       60      150
798  799   Hoopa Unbound  Psychic    Dark  80     160       60      170
799  800       Volcanion     Fire   Water  80     110      120      130

     Sp. Def  Speed  Generation  Legendary  Total Win Count  \
0         65     45           1      False               37
1         80     60           1      False               46
2        100     80           1      False               89
3        120     80           1      False               70
4         50     65           1      False               55
..       …      …         …        …              …
795      150     50           6       True               39
796      110    110           6       True              116
797      130     70           6       True               60
798      130     80           6       True               89
799       90     70           6       True               75

     Total Battle Count  Win Rate
```

```
0                    133  0.278195
1                    121  0.380165
2                    132  0.674242
3                    125  0.560000
4                    112  0.491071
..                   …         …
795                  105  0.371429
796                  131  0.885496
797                  119  0.504202
798                  144  0.618056
799                  121  0.619835

[783 rows x 15 columns]
```

[10]:
```
pkmnAll = pd.merge(pkmn, featEng, left_on='ID', right_index = True, how='left')
pkmnAll[pkmnAll['Win Rate'].isnull()]
```

[10]:
```
      ID                   Name    Type 1   Type 2   HP  Attack  Defense  \
11    12              Blastoise     Water      NaN   79      83      100
32    33              Sandshrew    Ground      NaN   50      75       85
45    46             Wigglytuff    Normal    Fairy  140      70       45
65    66                Poliwag     Water      NaN   40      50       40
77    78             Victreebel     Grass   Poison   80     105       65
89    90               Magneton  Electric    Steel   50      60       95
143  144                  Ditto    Normal      NaN   48      48       48
182  183                Ariados       Bug   Poison   70      90       70
230  231                Shuckle       Bug     Rock   20      10      230
235  236               Ursaring    Normal      NaN   90     130       75
321  322               Hariyama  Fighting      NaN  144     120       60
418  419            Mega Latias    Dragon  Psychic   80     100      120
478  479              Honchkrow      Dark   Flying  100     125       52
555  556                Servine     Grass      NaN   60      60       75
617  618               Maractus     Grass      NaN   75      86       67
654  655              Jellicent     Water    Ghost  100      60       70
781  782  Pumpkaboo Small Size     Ghost    Grass   44      66       70

     Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total Win Count  \
11        85      105     78           1      False              NaN
32        20       30     40           1      False              NaN
45        85       50     45           1      False              NaN
65        40       40     90           1      False              NaN
77       100       70     70           1      False              NaN
89       120       70     70           1      False              NaN
143       48       48     48           1      False              NaN
182       60       60     40           2      False              NaN
230       10      230      5           2      False              NaN
235       75       75     55           2      False              NaN
```

```
321       40       60     50           3     False              NaN
418      140      150    110           3      True              NaN
478      105       52     71           4     False              NaN
555       60       75     83           5     False              NaN
617      106       67     60           5     False              NaN
654       85      105     60           5     False              NaN
781       44       55     56           6     False              NaN

     Total Battle Count  Win Rate
11                  NaN       NaN
32                  NaN       NaN
45                  NaN       NaN
65                  NaN       NaN
77                  NaN       NaN
89                  NaN       NaN
143                 NaN       NaN
182                 NaN       NaN
230                 NaN       NaN
235                 NaN       NaN
321                 NaN       NaN
418                 NaN       NaN
478                 NaN       NaN
555                 NaN       NaN
617                 NaN       NaN
654                 NaN       NaN
781                 NaN       NaN
```

[11]:

```
pkmnWR.describe()
```

[11]:

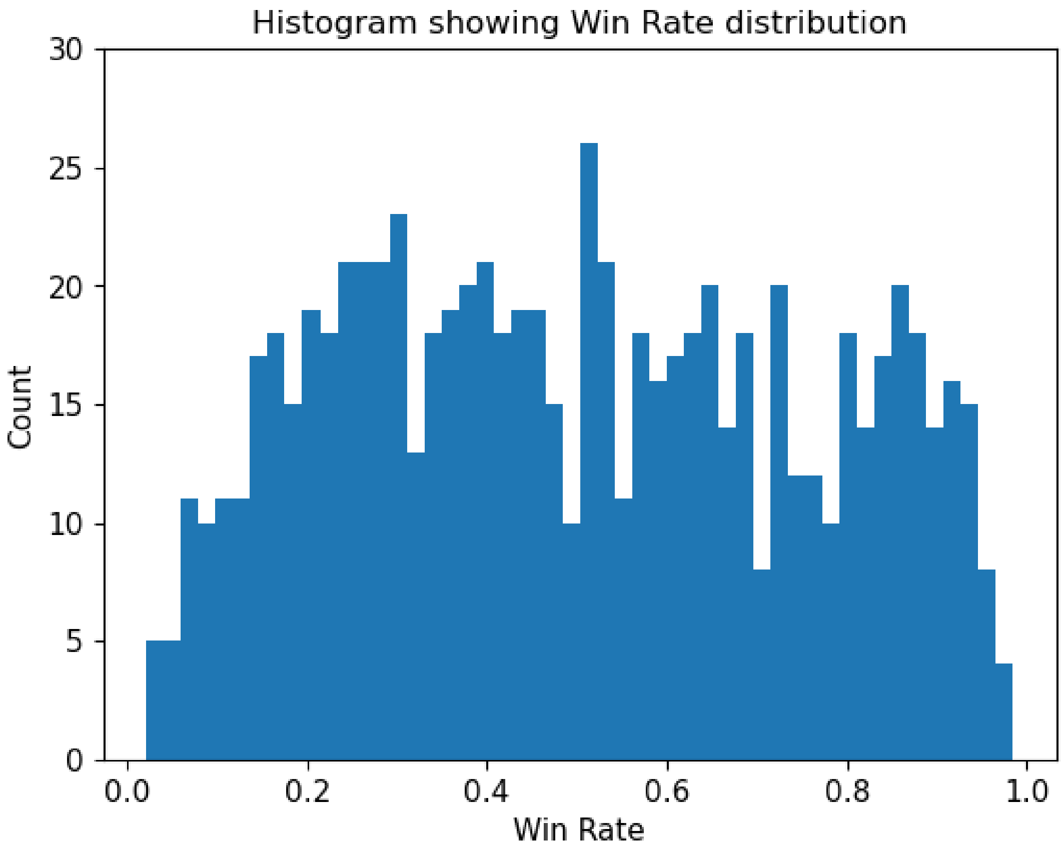
```
                 ID          HP      Attack     Defense     Sp. Atk     Sp. Def  \
count  783.000000  783.000000  783.000000  783.000000  783.000000  783.000000
mean   402.873563   69.140485   79.007663   73.699872   72.839080   71.749681
std    230.297452   25.348783   32.502566   30.879737   32.672868   27.248993
min      1.000000    1.000000    5.000000    5.000000   10.000000   20.000000
25%    204.500000   50.000000   55.000000   50.000000   50.000000   50.000000
50%    403.000000   65.000000   75.000000   70.000000   65.000000   70.000000
75%    601.500000   80.000000  100.000000   90.000000   95.000000   90.000000
max    800.000000  255.000000  190.000000  230.000000  194.000000  200.000000

            Speed  Generation  Total Win Count  Total Battle Count    Win Rate
count  783.000000  783.000000       783.000000          783.000000  783.000000
mean    68.443167    3.339719        63.856960          127.541507    0.501538
std     29.158076    1.656435        32.925941           11.397402    0.254993
min      5.000000    1.000000         3.000000           92.000000    0.021739
25%     45.000000    2.000000        36.000000          120.000000    0.284228
50%     65.000000    3.000000        62.000000          128.000000    0.491071
75%     90.000000    5.000000        91.000000          135.000000    0.717644
```

```
max     180.000000     6.000000         152.000000              164.000000     0.984496
```

[12]:

```
pkmnWR['Win Rate'].plot.hist(bins = 50,fontsize=11)
ax = plt.gca()
ax.set_ylim([0, 30])
plt.xlabel('Win Rate',fontsize=11)
plt.ylabel('Count',fontsize=11)
# arial = {'fontname':'Arial'}
# txt="Figure 1: Histogram showing Win Rate distribution."
# plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.title('Histogram showing Win Rate distribution',fontsize=12)
plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
 ↪Project\figures\WR Histogram.png")
plt.show()
```



[13]:

```
# Pearson correlation table & heatmap
col = ['HP', 'Attack', 'Defense', 'Sp. Atk', 'Sp. Def', 'Speed', 'Legendary',␣
 ↪'Win Rate']
```

```
pkmnWR.loc[:,col].corr()
```

[13]:

```
                 HP    Attack   Defense   Sp. Atk   Sp. Def     Speed  \
HP         1.000000  0.417427  0.265230  0.363244  0.409110  0.179423
Attack     0.417427  1.000000  0.464539  0.395211  0.288078  0.382310
Defense    0.265230  0.464539  1.000000  0.237592  0.490118  0.025762
Sp. Atk    0.363244  0.395211  0.237592  1.000000  0.529276  0.470548
Sp. Def    0.409110  0.288078  0.490118  0.529276  1.000000  0.276715
Speed      0.179423  0.382310  0.025762  0.470548  0.276715  1.000000
Legendary  0.280265  0.348391  0.247921  0.448339  0.367030  0.323420
Win Rate   0.258006  0.500181  0.129426  0.478940  0.324218  0.937742

           Legendary  Win Rate
HP          0.280265  0.258006
Attack      0.348391  0.500181
Defense     0.247921  0.129426
Sp. Atk     0.448339  0.478940
Sp. Def     0.367030  0.324218
Speed       0.323420  0.937742
Legendary   1.000000  0.325007
Win Rate    0.325007  1.000000
```

[14]:

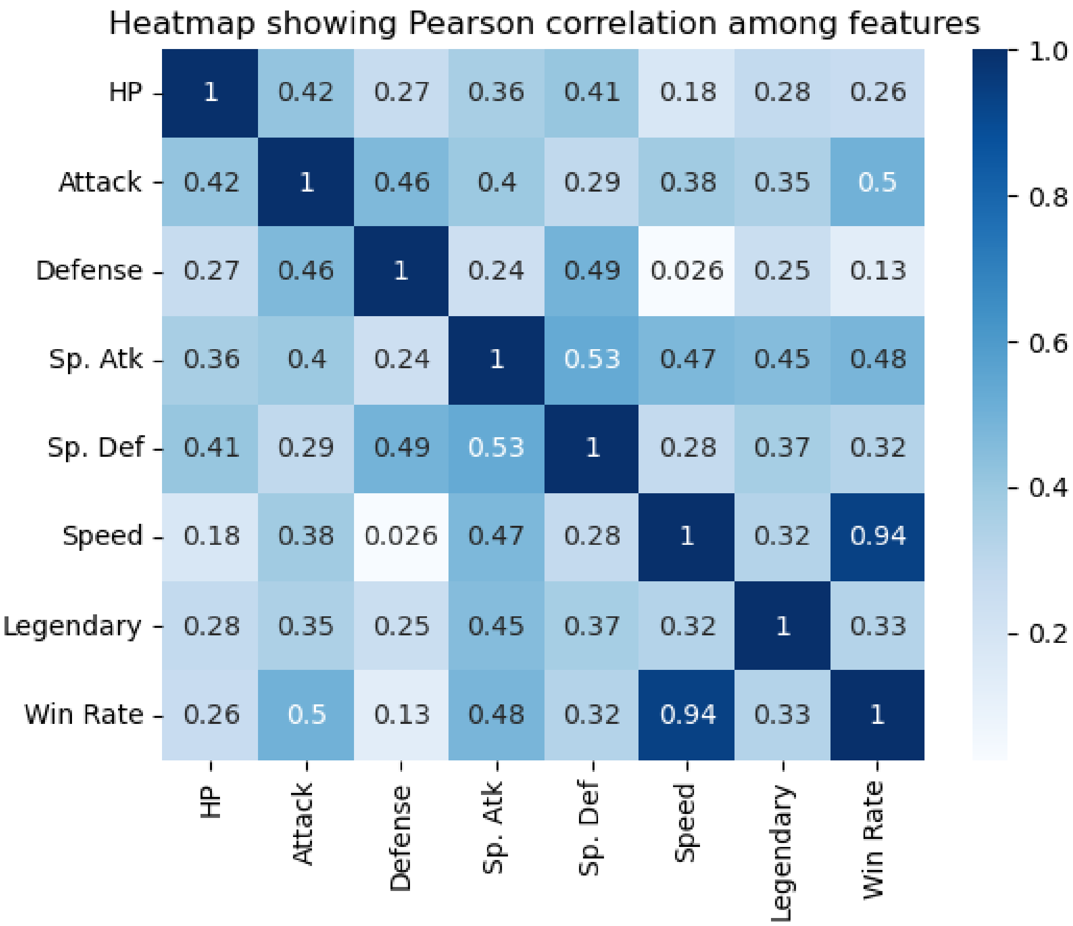
```
# Pearson correlation table & heatmap
import seaborn as sns

corr = pkmnWR.loc[:,col].corr(method='pearson', min_periods=1)
print(pkmnWR.loc[:,col].corr(method='pearson', min_periods=1))
sns.heatmap(corr, cmap="Blues", annot=True)
plt.title('Heatmap showing Pearson correlation among features',fontsize=12)
plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
  ↪Project\figures\Pearson Correlation Heatmap.png")
```

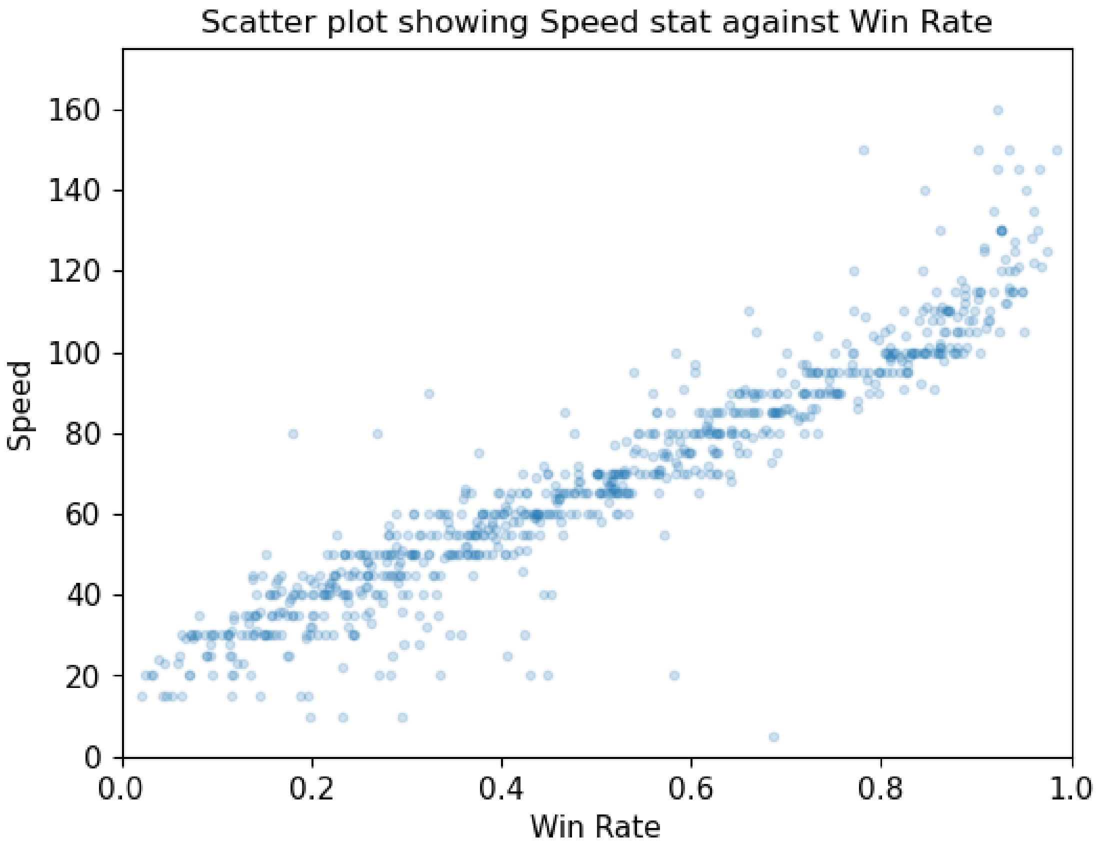
```
                 HP    Attack   Defense   Sp. Atk   Sp. Def     Speed  \
HP         1.000000  0.417427  0.265230  0.363244  0.409110  0.179423
Attack     0.417427  1.000000  0.464539  0.395211  0.288078  0.382310
Defense    0.265230  0.464539  1.000000  0.237592  0.490118  0.025762
Sp. Atk    0.363244  0.395211  0.237592  1.000000  0.529276  0.470548
Sp. Def    0.409110  0.288078  0.490118  0.529276  1.000000  0.276715
Speed      0.179423  0.382310  0.025762  0.470548  0.276715  1.000000
Legendary  0.280265  0.348391  0.247921  0.448339  0.367030  0.323420
Win Rate   0.258006  0.500181  0.129426  0.478940  0.324218  0.937742

           Legendary  Win Rate
HP          0.280265  0.258006
Attack      0.348391  0.500181
Defense     0.247921  0.129426
Sp. Atk     0.448339  0.478940
Sp. Def     0.367030  0.324218
```

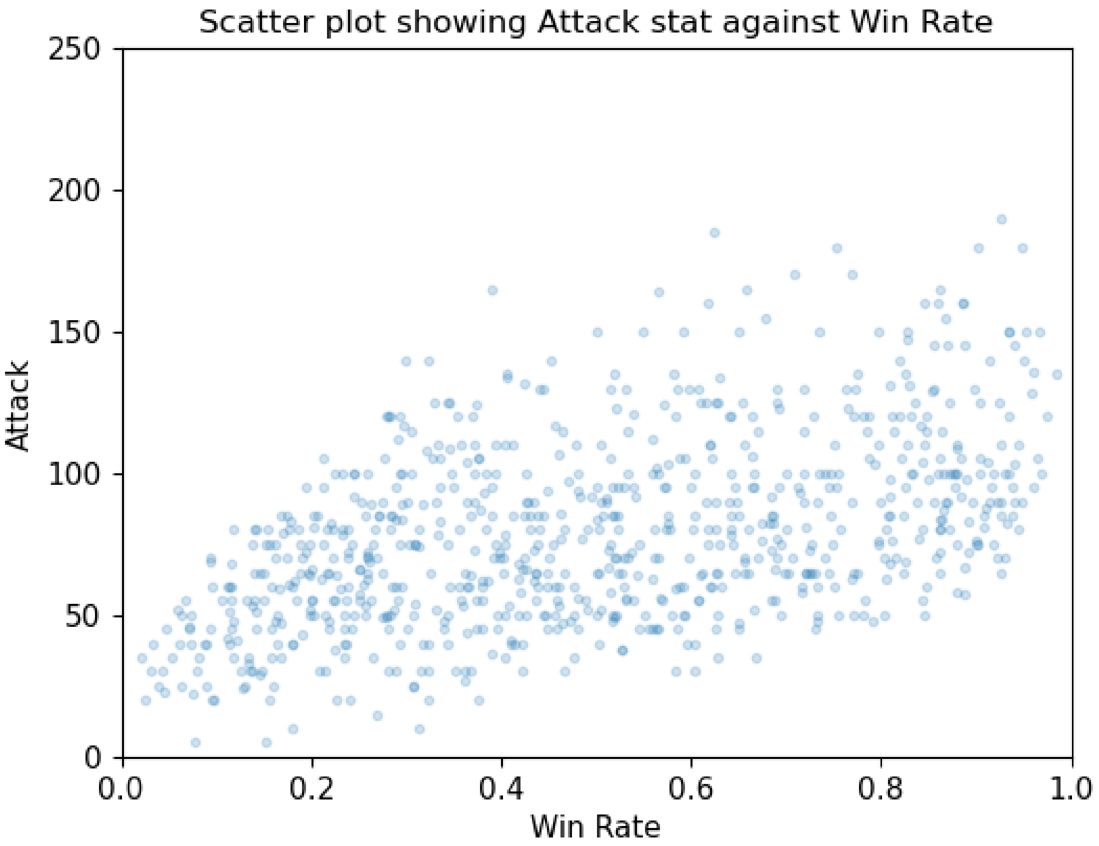
```
Speed        0.323420  0.937742
Legendary    1.000000  0.325007
Win Rate     0.325007  1.000000
```



```
[15]: pkmnWR.plot.scatter('Win Rate', 'Speed',s=10,alpha=0.2,fontsize=11)
      ax = plt.gca()
      ax.set_xlim([0, 1])
      ax.set_ylim([0, 175])
      plt.xlabel('Win Rate',fontsize=11)
      plt.ylabel('Speed',fontsize=11)
      # txt="Scatter plot showing Speed stat against Win Rate."
      # plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center',␣
        ↪fontsize=10)
      plt.title('Scatter plot showing Speed stat against Win Rate',fontsize=12)
      plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
        ↪Project\figures\Speed-WR Scatter Plot.png")
      plt.show()
```

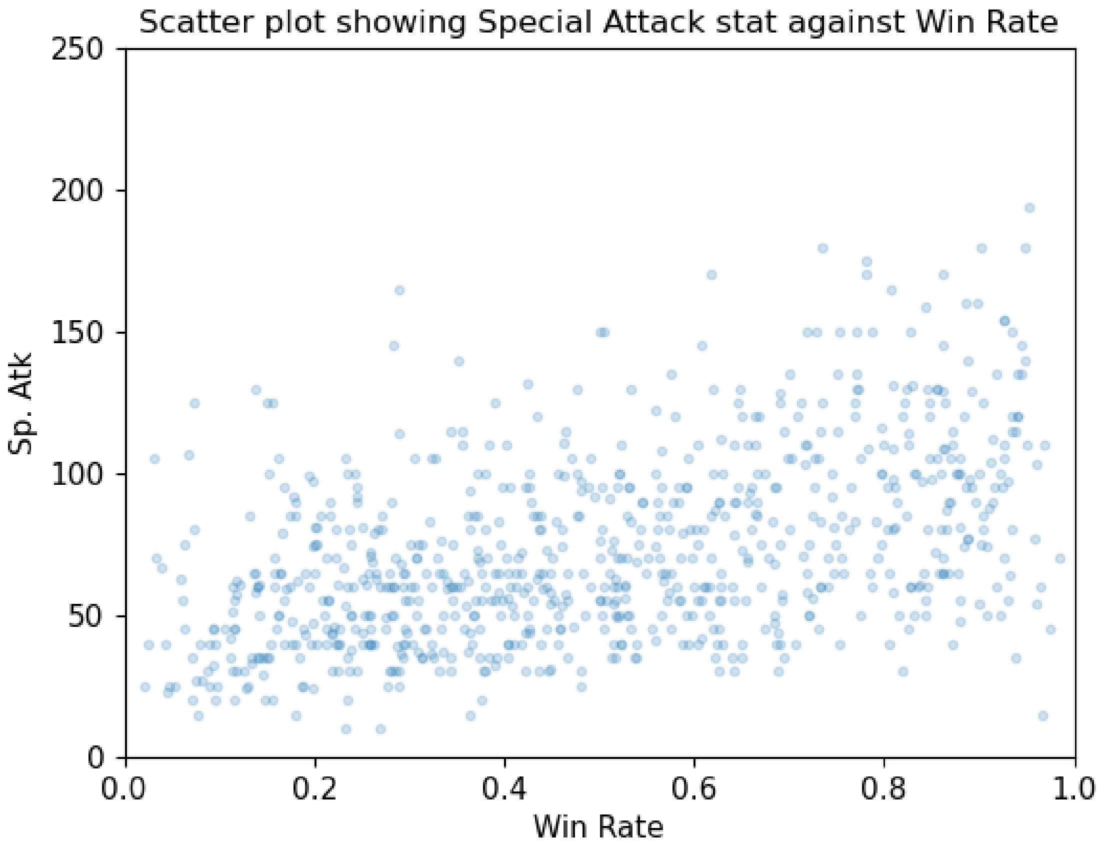


[16]:
```
pkmnWR.plot.scatter('Win Rate', 'Attack',s=10,alpha=0.2,fontsize=11)
ax = plt.gca()
ax.set_xlim([0, 1])
ax.set_ylim([0, 250])
plt.xlabel('Win Rate',fontsize=11)
plt.ylabel('Attack',fontsize=11)
# txt="Scatter plot showing Attack stat against Win Rate."
# plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.title('Scatter plot showing Attack stat against Win Rate', fontsize=12)
plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
 ↪Project\figures\Attack-WR Scatter Plot.png")
plt.show()
```



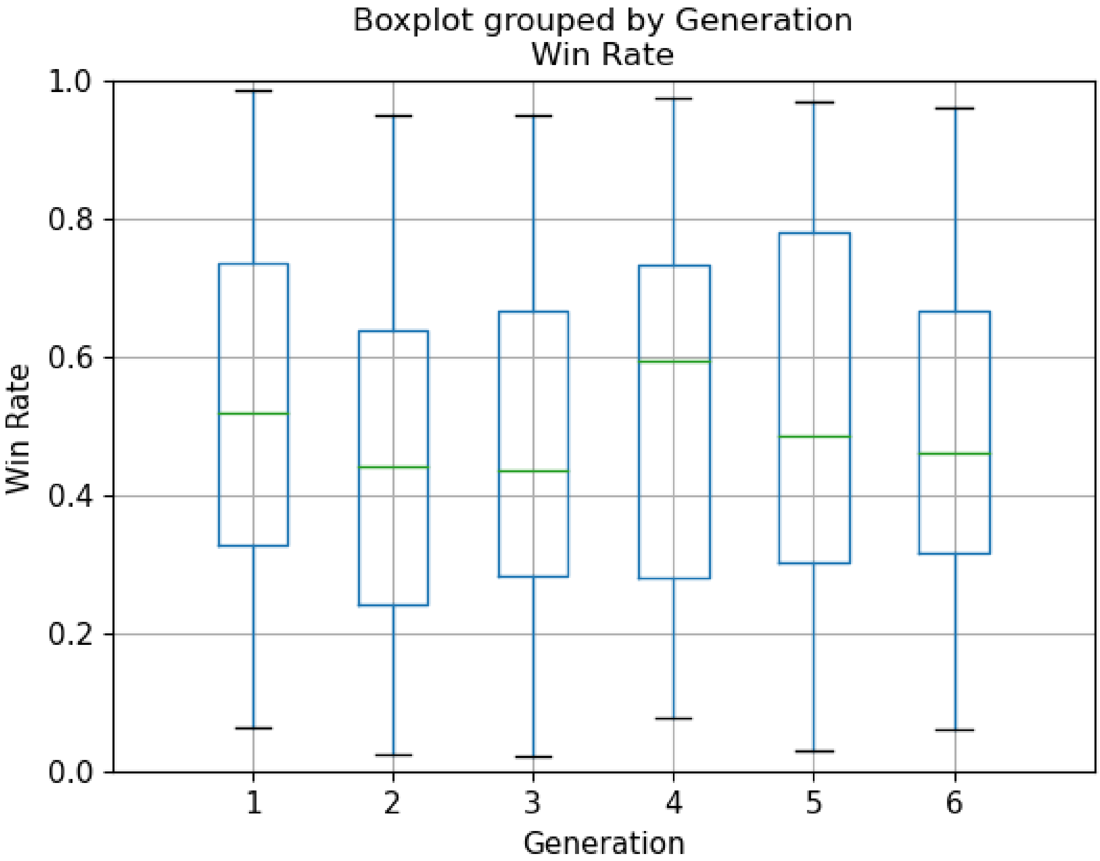
[17]:

```
pkmnWR.plot.scatter('Win Rate', 'Sp. Atk',s=10,alpha=0.2,fontsize=11)
ax = plt.gca()
ax.set_xlim([0, 1])
ax.set_ylim([0, 250])
plt.xlabel('Win Rate',fontsize=11)
plt.ylabel('Sp. Atk',fontsize=11)
# txt="Scatter plot showing Special Attack stat against Win Rate."
# plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.title('Scatter plot showing Special Attack stat against Win␣
 ↪Rate',fontsize=12)
plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
 ↪Project\figures\SpAtk-WR Scatter Plot.png")
plt.show()
```



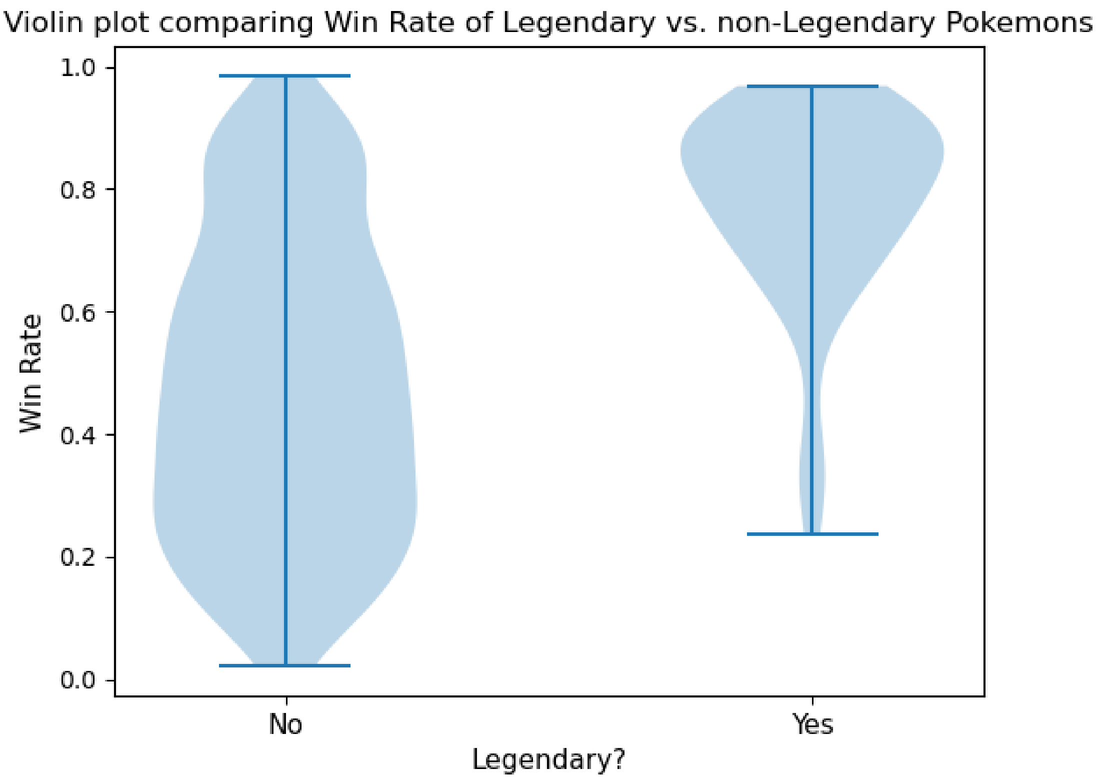
[18]:

```
pkmnWR[['Win Rate','Generation']].boxplot(by='Generation',fontsize=11)
ax = plt.gca()
ax.set_xlim([0, 7])
ax.set_ylim([0, 1])
plt.ylabel('Win Rate',fontsize=11)
plt.xlabel('Generation',fontsize=11)
# txt="Box plot comparing Win Rate of Pokemons across 6 generations."
# plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
 ↪Project\figures\Generation-WR Box Plot.png")
plt.show()
```



[19]:

```
dataset = [pkmnWR[pkmnWR['Legendary']==False]['Win Rate'].values,
           pkmnWR[pkmnWR['Legendary']==True]['Win Rate'].values]
plt.violinplot(dataset = dataset)
plt.xticks([1,2],['No','Yes'],fontsize=11)
plt.ylabel('Win Rate',fontsize=11)
plt.xlabel('Legendary?',fontsize=11)
# txt="Violin plot comparing Win Rate of Legendary vs. non-Legendary Pokemons."
# plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center',␣
 ↪fontsize=10)
plt.title('Violin plot comparing Win Rate of Legendary vs. non-Legendary␣
 ↪Pokemons',fontsize=12)
plt.savefig(r"C:\Users\User\Desktop\DSI\DATA1030-Fall2022\Taemin Huh DATA1030␣
 ↪Project\figures\Legendary Violin Plot.png")
plt.show()
```



1.2 Preprocessing

[20]:
```
from sklearn.model_selection import train_test_split
y = pkmnWR['Win Rate']
X = pkmnWR.loc[:, pkmnWR.columns != 'Win Rate']

print(X.head())
print(X.shape)
print(y)
```

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

   Speed  Generation  Legendary  Total Win Count  Total Battle Count
0     45           1      False               37                 133
1     60           1      False               46                 121
2     80           1      False               89                 132
3     80           1      False               70                 125
```

```
4     65           1      False               55                 112
(783, 14)
0      0.278195
1      0.380165
2      0.674242
3      0.560000
4      0.491071
         …
795    0.371429
796    0.885496
797    0.504202
798    0.618056
799    0.619835
Name: Win Rate, Length: 783, dtype: float64
```

[21]:

```
def basic_split(X,y,train_size,val_size,test_size,random_state):

    X_train, X_other, y_train, y_other = train_test_split(X,y,train_size =␣
 ↪train_size,random_state=random_state)
    print('training set:',X_train.shape, y_train.shape)
    print(X_other.shape, y_other.shape)

    X_val, X_test, y_val, y_test = train_test_split(X_other,y_other,train_size␣
 ↪= val_size/(val_size+test_size),random_state=random_state)
    print('validation set:',X_val.shape, y_val.shape)
    print('test set:',X_test.shape, y_test.shape)

    print(X_train.head())
    return X_train, y_train, X_val, y_val, X_test, y_test

print(basic_split(X,y,0.6,0.2,0.2,7))
```

```
training set: (469, 14) (469,)
(314, 14) (314,)
validation set: (157, 14) (157,)
test set: (157, 14) (157,)
      ID                 Name  Type 1  Type 2   HP  Attack  Defense  Sp. Atk  \
95    96               Grimer  Poison     NaN   80      80       50       40
221  222            Dunsparce  Normal     NaN  100      70       70       65
139  140             Magikarp   Water     NaN   20      10       55       15
541  542               Palkia   Water  Dragon   90     120      100      150
7      8     Mega Charizard X    Fire  Dragon   78     130      111      130

     Sp. Def  Speed  Generation  Legendary  Total Win Count  \
95        50     25           1      False               25
221       65     45           2      False               26
139       20     80           1      False               25
541      120    100           4       True              115
```

```
7          85    100            1      False               119

     Total Battle Count
95                  142
221                 137
139                 139
541                 149
7                   139
(      ID                Name    Type 1  Type 2   HP  Attack  Defense  Sp. Atk  
\
95    96              Grimer    Poison     NaN   80      80       50       40   
221  222           Dunsparce    Normal     NaN  100      70       70       65   
139  140            Magikarp     Water     NaN   20      10       55       15   
541  542              Palkia     Water  Dragon   90     120      100      150   
7      8    Mega Charizard X      Fire  Dragon   78     130      111      130   
..   …                   …       …     …   …     …       …       …   
593  594             Gurdurr  Fighting     NaN   85     105       85       40   
515  516           Rhyperior    Ground    Rock  115     140      130       55   
550  551  Shaymin Land Forme     Grass     NaN  100     100      100      100   
204  205            Jumpluff     Grass  Flying   75      55       70       55   
183  184              Crobat    Poison  Flying   85      90       80       70   

     Sp. Def  Speed  Generation  Legendary  Total Win Count  \
95        50     25           1      False               25   
221       65     45           2      False               26   
139       20     80           1      False               25   
541      120    100           4       True              115   
7         85    100           1      False              119   
..       …    …         …        …              …   
593       50     40           5      False               25   
515       55     40           4      False               47   
550      100    100           4       True              117   
204       95    110           2      False              114   
183       80    130           2      False              113   

     Total Battle Count  
95                  142  
221                 137  
139                 139  
541                 149  
7                   139  
..                  …  
593                 118  
515                 104  
550                 136  
204                 135  
183                 122  
```

```
[469 rows x 14 columns], 95     0.176056
221    0.189781
139    0.179856
541    0.771812
7      0.856115
         …
593    0.211864
515    0.451923
550    0.860294
204    0.844444
183    0.926230
Name: Win Rate, Length: 469, dtype: float64,       ID           Name    Type 1
Type 2   HP  Attack  Defense  Sp. Atk  \
767  768      Tyrantrum      Rock  Dragon   82     121      119       69
728  729      Diggersby    Normal  Ground   85      56       77       50
753  754     Aromatisse     Fairy     NaN  101      72       72       99
196  197  Mega Ampharos  Electric  Dragon   90      95      105      165
266  267        Pupitar      Rock  Ground   70      84       70       65
..   …            …       …     …  …     …      …      …
199  200      Azumarill     Water   Fairy  100      50       80       60
414  415       Regirock      Rock     NaN   80     100      200       50
261  262        Blissey    Normal     NaN  255      10       10       75
749  750       Doublade     Steel   Ghost   59     110      150       45
423  424        Groudon    Ground     NaN  100     150      140      100

     Sp. Def  Speed  Generation  Legendary  Total Win Count  \
767       59     71           6      False               82
728       77     78           6      False               68
753       89     29           6      False               25
196      110     45           2      False               37
266       70     51           2      False               41
..       …    …         …        …              …
199       80     50           2      False               44
414      100     50           3       True               50
261      135     55           2      False               40
749       49     35           6      False               44
423       90     90           3       True              106

     Total Battle Count
767                 152
728                 128
753                 129
196                 128
266                 139
..                  …
199                 128
414                 144
261                 128
```

```
749                 132
423                 133

[157 rows x 14 columns], 767    0.539474
728    0.531250
753    0.193798
196    0.289062
266    0.294964
         …
199    0.343750
414    0.347222
261    0.312500
749    0.333333
423    0.796992
Name: Win Rate, Length: 157, dtype: float64,       ID                      Name
Type 1   Type 2   HP  Attack  Defense  \
31    32                   Raichu  Electric      NaN   60      90       55
669  670                  Lampent     Ghost     Fire   60      40       60
345  346                   Gulpin    Poison      NaN   70      43       53
459  460      Wormadam Sandy Cloak       Bug   Ground   60      79      105
201  202                 Politoed     Water      NaN   90      75       75
..   …                         …        …       …   …      …       …
421  422                   Kyogre     Water      NaN  100     100       90
194  195                  Flaaffy  Electric      NaN   70      55       55
485  486                 Bronzong     Steel  Psychic   67      89      116
708  709  Landorus Incarnate Forme    Ground   Flying   89     125       90
676  677                Cryogonal       Ice      NaN   70      50       30

     Sp. Atk  Sp. Def  Speed  Generation  Legendary  Total Win Count  \
31        90       80    110           1      False              105
669       95       60     55           5      False               57
345       43       53     40           3      False               21
459       59       85     36           4      False               22
201       90      100     70           2      False               65
..       …       …    …          …        …               …
421      150      140     90           3       True               92
194       80       60     45           2      False               24
485       79      116     33           4      False               37
708      115       80    101           5       True               89
676       95      135    105           5      False              119

     Total Battle Count
31                  121
669                 135
345                 110
459                 129
201                 119
..                  …
```

```
421                  128
194                  108
485                  141
708                  102
676                  148

[157 rows x 14 columns], 31     0.867769
669    0.422222
345    0.190909
459    0.170543
201    0.546218
         …
421    0.718750
194    0.222222
485    0.262411
708    0.872549
676    0.804054
Name: Win Rate, Length: 157, dtype: float64)
```

[22]:

```
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder, MinMaxScaler

X_train, y_train, X_val, y_val, X_test, y_test = basic_split(X,y,0.6,0.2,0.2,7)

# Pre-processing with OneHotEncoder and MinMaxScaler

onehot_ftrs = ['Type 1', 'Type 2', 'Generation', 'Legendary']
minmax_ftrs = ['HP','Attack', 'Defense', 'Sp. Atk', 'Sp. Def', 'Speed']

preprocessor = ColumnTransformer(
    transformers=[
        ('onehot', OneHotEncoder(sparse=False,handle_unknown='ignore'),␣
 ↪onehot_ftrs),
        ('minmax', MinMaxScaler(), minmax_ftrs)])

clf = Pipeline(steps=[('preprocessor', preprocessor)])

X_train_prep = clf.fit_transform(X_train)
X_val_prep = clf.transform(X_val)
X_test_prep = clf.transform(X_test)

print('X_train shape:',X_train.shape)
print('X_train_prep shape:',X_train_prep.shape)
print(X_train_prep)
```

```
training set: (469, 14) (469,)
(314, 14) (314,)
```

```
validation set: (157, 14) (157,)
test set: (157, 14) (157,)
      ID              Name  Type 1  Type 2   HP  Attack  Defense  Sp. Atk  \
95    96            Grimer  Poison     NaN   80      80       50       40   
221  222         Dunsparce  Normal     NaN  100      70       70       65   
139  140          Magikarp   Water     NaN   20      10       55       15   
541  542            Palkia   Water  Dragon   90     120      100      150   
7      8  Mega Charizard X    Fire  Dragon   78     130      111      130   

     Sp. Def  Speed  Generation  Legendary  Total Win Count  \
95        50     25           1      False               25   
221       65     45           2      False               26   
139       20     80           1      False               25   
541      120    100           4       True              115   
7         85    100           1      False              119   

     Total Battle Count  
95                  142  
221                 137  
139                 139  
541                 149  
7                   139  
X_train shape: (469, 14)
X_train_prep shape: (469, 51)
[[0.         0.         0.         … 0.16304348 0.16666667 0.13793103]
 [0.         0.         0.         … 0.29891304 0.25       0.27586207]
 [0.         0.         0.         … 0.02717391 0.         0.51724138]
 …
 [0.         0.         0.         … 0.48913043 0.44444444 0.65517241]
 [0.         0.         0.         … 0.24456522 0.41666667 0.72413793]
 [0.         0.         0.         … 0.32608696 0.33333333 0.86206897]]
```