In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
train = pd.read_csv("../input/scrabble-player-rating/train.csv")
test = pd.read_csv("../input/scrabble-player-rating/test.csv")
games = pd.read_csv("../input/scrabble-player-rating/games.csv")
turns = pd.read_csv("../input/scrabble-player-rating/turns.csv")
```

In [3]:
```
train.head()
```

Out[3]:

| | game_id | nickname | score | rating |
|---|---|---|---|---|
| **0** | 1 | BetterBot | 335 | 1637 |
| **1** | 1 | stevy | 429 | 1500 |
| **2** | 3 | davidavid | 440 | 1811 |
| **3** | 3 | BetterBot | 318 | 2071 |
| **4** | 4 | Inandoutworker | 119 | 1473 |

In [4]:
```
games.head()
```

Out[4]:

| | game_id | first | time_control_name | game_end_reason | winner | created_at | lexicon | initial_time_seconds | increment_seconds | rating_mo |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | BetterBot | regular | STANDARD | 1 | 2022-08-26 03:38:49 | NWL20 | 1200 | 0 | CASU |
| **1** | 2 | Super | regular | STANDARD | 1 | 2022-08-10 19:19:59 | CSW21 | 3600 | 0 | RATI |
| **2** | 3 | BetterBot | regular | STANDARD | 1 | 2022-09-04 08:04:27 | CSW21 | 900 | 0 | RATI |
| **3** | 4 | BetterBot | regular | RESIGNED | 0 | 2022-09-12 02:36:19 | CSW21 | 3600 | 0 | CASU |
| **4** | 5 | STEEBot | regular | STANDARD | 0 | 2022-09-06 04:31:36 | NWL20 | 1200 | 0 | CASU |

In [5]:
```
turns.head()
```

Out[5]:

| | game_id | turn_number | nickname | rack | location | move | points | score | turn_type |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1 | BetterBot | DDEGITT | 8G | DIG | 10 | 10 | Play |
| **1** | 1 | 2 | stevy | AEHOPUX | 7H | HAP | 18 | 18 | Play |
| **2** | 1 | 3 | BetterBot | DEELTTU | 6I | LUTE | 16 | 26 | Play |
| **3** | 1 | 4 | stevy | EMORSUX | 5K | UM | 16 | 34 | Play |
| **4** | 1 | 5 | BetterBot | ACDEITU | L5 | ..DICATE | 28 | 54 | Play |

In [6]:
```
train.shape, games.shape
```

Out[6]:
```
((100820, 4), (72773, 12))
```

In [7]:
```
levels = ['BetterBot','STEEBot','HastyBot']

train_players = train.loc[~train['nickname'].isin(levels)]
test_players = test.loc[~test['nickname'].isin(levels)]
```

In [8]:
```
turn_players = turns.loc[~turns['nickname'].isin(levels)]
train_players["num_moves"] = train_players['game_id'].map(turn_players['game_id'].value_counts())
test_players["num_moves"] = test_players['game_id'].map(turn_players['game_id'].value_counts())
```

```
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a
-view-versus-a-copy

/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a
-view-versus-a-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [9]:
```
train_players
```

Out[9]:

| | game_id | nickname | score | rating | num_moves |
|---|---|---|---|---|---|
| **1** | 1 | stevy | 429 | 1500 | 14 |
| **2** | 3 | davidavid | 440 | 1811 | 14 |
| **4** | 4 | Inandoutworker | 119 | 1473 | 14 |
| **6** | 5 | stevy | 325 | 1500 | 16 |
| **8** | 6 | HivinD | 378 | 2029 | 12 |
| **...** | ... | ... | ... | ... | ... |
| **100810** | 72767 | HAYDEN | 340 | 1395 | 16 |
| **100813** | 72770 | samsiah06 | 97 | 1332 | 16 |
| **100814** | 72771 | BB-8 | 390 | 1500 | 16 |
| **100817** | 72772 | Gtowngrad | 388 | 1364 | 16 |
| **100818** | 72773 | adola | 383 | 2075 | 12 |

50410 rows × 5 columns

In [10]:
```
train_merged = pd.merge(train_players, games, on="game_id")
test_merged = pd.merge(test_players, games, on="game_id")
```

In [11]:
```
speed = {'regular':0, 'rapid':1,'blitz':2}

train_merged['time_control_name'] = train_merged['time_control_name'].map(speed)
test_merged['time_control_name'] = test_merged['time_control_name'].map(speed)
```

In [12]:
```
train_robots = train.loc[train['nickname'].isin(levels)]
test_robots = test.loc[test['nickname'].isin(levels)]
```

In [13]:
```
train_Rmerged = pd.merge(train_robots, games, on="game_id")
test_Rmerged = pd.merge(test_robots, games, on="game_id")
```

In [14]:
```
train_merged["score_difference"] = abs(train_merged['score'] - train_Rmerged['score'])
train_merged["level"] = train_Rmerged['nickname']
levels = {'BetterBot':0, 'STEEBot':1, 'Hastybot':2}
train_merged['level'] = train_merged['level'].map(levels)
```

```
test_merged["score_difference"] = abs(test_merged['score'] - test_Rmerged['score'])
test_merged["level"] = test_Rmerged['nickname']
test_merged['level'] = test_merged['level'].map(levels)
```

In [15]:

```
from sklearn import preprocessing

label_encoder = preprocessing.LabelEncoder()

train_merged['level'] = label_encoder.fit_transform(train_merged['level'].values)
test_merged['level'] = label_encoder.fit_transform(test_merged['level'].values)
```

In [16]:

```
train_merged.isnull().sum()
```

Out[16]:

```
game_id                    0
nickname                   0
score                      0
rating                     0
num_moves                  0
first                      0
time_control_name        162
game_end_reason            0
winner                     0
created_at                 0
lexicon                    0
initial_time_seconds       0
increment_seconds          0
rating_mode                0
max_overtime_minutes       0
game_duration_seconds      0
score_difference           0
level                      0
dtype: int64
```

In [17]:

```
test_game_id = test_merged['game_id']
```

In [18]:

```
train_merged.drop(['first','nickname','created_at','game_id'],axis=1,inplace=True)
test_merged.drop(['first','nickname','created_at','rating','game_id'],axis=1,inplace=True)
```

In [19]:

```
train_dummies = pd.get_dummies(train_merged,drop_first=True)
test_dummies = pd.get_dummies(test_merged,drop_first=True)
```

In [20]:

```
train_dummies.drop('rating',axis=1,inplace=True)
train_dummies.head()
```

Out[20]:

| | score | num_moves | time_control_name | winner | initial_time_seconds | increment_seconds | max_overtime_minutes | game_duration_seconds | s |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 429 | 14 | 0.0 | 1 | 1200 | 0 | 1 | 674.844274 | |
| 1 | 440 | 14 | 0.0 | 1 | 900 | 0 | 5 | 492.268262 | |
| 2 | 119 | 14 | 0.0 | 0 | 3600 | 0 | 1 | 350.861141 | |
| 3 | 325 | 16 | 0.0 | 0 | 1200 | 0 | 1 | 642.688722 | |
| 4 | 378 | 12 | 0.0 | 0 | 900 | 0 | 1 | 426.950541 | |

In [21]:

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values=np.nan , strategy='mean')
test_dummies[['time_control_name']] = imputer.fit_transform(test_dummies[['time_control_name']])
train_dummies[['time_control_name']] = imputer.fit_transform(train_dummies[['time_control_name']])
```

In [22]:

```
train_dummies.drop('lexicon_NSWL20',axis=1,inplace=True)
```

In [23]:

```
from sklearn.preprocessing import StandardScaler

std = StandardScaler()
train_dummies_std = std.fit_transform(train_dummies)
test_dummies_std = std.fit_transform(test_dummies)
```

In [24]:

```
from sklearn.model_selection import train_test_split

X,y = train_dummies_std, train_merged.iloc[:,1].values
X_train,X_test,y_train,y_test = train_test_split(X,y,
                                                 test_size = 0.3,
                                                 random_state=0)
X_sub = test_dummies_std
```

In [25]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error


reg = LinearRegression().fit(X_train, y_train)
reg.score(X_test, y_test)

mean_squared_error(y_test, reg.predict(X_test), squared=False)
```

Out[25]: 139.7434783445873

In [26]:
```
from sklearn.ensemble import GradientBoostingRegressor

gbr = GradientBoostingRegressor(random_state=0)
gbr.fit(X_train, y_train)

mean_squared_error(y_test, gbr.predict(X_test), squared=False)
```

Out[26]: 122.87455537050629

In [27]:
```
from sklearn import svm
svm = svm.SVR()
svm.fit(X_train, y_train)

mean_squared_error(y_test, svm.predict(X_test), squared=False)
```

Out[27]: 137.35485427888196

In [28]:
```
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import tensorflow as tf
```

In [29]:
```
from tensorflow.keras import layers
from tensorflow import keras

def build_model():
    model = keras.Sequential([
        layers.Dense(500, activation="relu"),
        layers.Dense(250, activation="relu"),
        layers.Dense(250, activation="relu"),
        layers.Dense(50, activation="relu"),
        layers.Dense(1)
    ])
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
                  loss='mean_absolute_error')
    return model
```

In [30]:
```
model = build_model()
history = model.fit(
    X_train,
    y_train,
    epochs=100,
```

```
    verbose=2,
    # Calculate validation results on 20% of the training data.
    validation_data = (X_test,y_test))
```

```
Epoch 1/100
1103/1103 - 4s - loss: 176.9052 - val_loss: 99.1616
Epoch 2/100
1103/1103 - 2s - loss: 100.7436 - val_loss: 100.9564
Epoch 3/100
1103/1103 - 3s - loss: 97.7667 - val_loss: 97.2308
Epoch 4/100
1103/1103 - 3s - loss: 96.8294 - val_loss: 91.6271
Epoch 5/100
1103/1103 - 2s - loss: 94.4746 - val_loss: 92.0466
Epoch 6/100
1103/1103 - 2s - loss: 92.7907 - val_loss: 88.3915
Epoch 7/100
1103/1103 - 2s - loss: 92.7113 - val_loss: 96.9077
Epoch 8/100
1103/1103 - 3s - loss: 90.6830 - val_loss: 86.7921
Epoch 9/100
1103/1103 - 2s - loss: 90.4512 - val_loss: 85.6791
Epoch 10/100
1103/1103 - 2s - loss: 89.5835 - val_loss: 85.6063
Epoch 11/100
1103/1103 - 2s - loss: 88.9124 - val_loss: 88.0993
Epoch 12/100
1103/1103 - 3s - loss: 88.2120 - val_loss: 86.9371
Epoch 13/100
1103/1103 - 2s - loss: 87.3493 - val_loss: 87.7547
Epoch 14/100
1103/1103 - 2s - loss: 88.0782 - val_loss: 83.2880
Epoch 15/100
1103/1103 - 2s - loss: 85.9196 - val_loss: 97.6502
Epoch 16/100
1103/1103 - 3s - loss: 85.5704 - val_loss: 80.5785
Epoch 17/100
1103/1103 - 3s - loss: 84.5858 - val_loss: 89.0124
Epoch 18/100
1103/1103 - 2s - loss: 85.1149 - val_loss: 80.9868
Epoch 19/100
1103/1103 - 2s - loss: 83.6014 - val_loss: 84.7586
Epoch 20/100
1103/1103 - 2s - loss: 83.5298 - val_loss: 86.7967
Epoch 21/100
1103/1103 - 3s - loss: 83.6922 - val_loss: 83.6873
Epoch 22/100
1103/1103 - 3s - loss: 82.9357 - val_loss: 79.0191
Epoch 23/100
```

```
1103/1103 - 3s - loss: 82.5063 - val_loss: 82.7828
Epoch 24/100
1103/1103 - 2s - loss: 81.4425 - val_loss: 80.9511
Epoch 25/100
1103/1103 - 3s - loss: 81.7161 - val_loss: 104.7168
Epoch 26/100
1103/1103 - 2s - loss: 82.4007 - val_loss: 84.0168
Epoch 27/100
1103/1103 - 2s - loss: 81.2734 - val_loss: 80.9904
Epoch 28/100
1103/1103 - 3s - loss: 81.0794 - val_loss: 90.5179
Epoch 29/100
1103/1103 - 2s - loss: 80.8105 - val_loss: 78.3139
Epoch 30/100
1103/1103 - 3s - loss: 79.5972 - val_loss: 82.0417
Epoch 31/100
1103/1103 - 2s - loss: 80.4638 - val_loss: 78.5543
Epoch 32/100
1103/1103 - 2s - loss: 79.6034 - val_loss: 81.4076
Epoch 33/100
1103/1103 - 2s - loss: 79.5804 - val_loss: 78.0300
Epoch 34/100
1103/1103 - 3s - loss: 79.2522 - val_loss: 77.5533
Epoch 35/100
1103/1103 - 2s - loss: 79.5034 - val_loss: 78.9911
Epoch 36/100
1103/1103 - 3s - loss: 78.3107 - val_loss: 78.9318
Epoch 37/100
1103/1103 - 2s - loss: 78.6509 - val_loss: 78.4894
Epoch 38/100
1103/1103 - 3s - loss: 79.2648 - val_loss: 77.1254
Epoch 39/100
1103/1103 - 2s - loss: 78.1063 - val_loss: 85.2974
Epoch 40/100
1103/1103 - 3s - loss: 78.2698 - val_loss: 79.6996
Epoch 41/100
1103/1103 - 2s - loss: 78.3141 - val_loss: 77.0302
Epoch 42/100
1103/1103 - 3s - loss: 77.9557 - val_loss: 78.7211
Epoch 43/100
1103/1103 - 2s - loss: 78.3152 - val_loss: 86.9173
Epoch 44/100
1103/1103 - 2s - loss: 77.9701 - val_loss: 90.8606
Epoch 45/100
1103/1103 - 2s - loss: 77.2282 - val_loss: 76.8731
```

```
Epoch 46/100
1103/1103 - 2s - loss: 77.4195 - val_loss: 79.3340
Epoch 47/100
1103/1103 - 3s - loss: 76.8128 - val_loss: 78.6413
Epoch 48/100
1103/1103 - 2s - loss: 77.1689 - val_loss: 82.1263
Epoch 49/100
1103/1103 - 2s - loss: 76.4724 - val_loss: 77.4503
Epoch 50/100
1103/1103 - 2s - loss: 76.3742 - val_loss: 77.6938
Epoch 51/100
1103/1103 - 3s - loss: 76.4305 - val_loss: 78.8828
Epoch 52/100
1103/1103 - 3s - loss: 76.4106 - val_loss: 76.3813
Epoch 53/100
1103/1103 - 3s - loss: 76.4994 - val_loss: 82.3976
Epoch 54/100
1103/1103 - 2s - loss: 76.3338 - val_loss: 78.9307
Epoch 55/100
1103/1103 - 3s - loss: 76.1759 - val_loss: 77.6145
Epoch 56/100
1103/1103 - 3s - loss: 75.9738 - val_loss: 77.9865
Epoch 57/100
1103/1103 - 2s - loss: 75.8739 - val_loss: 78.0223
Epoch 58/100
1103/1103 - 2s - loss: 75.4275 - val_loss: 83.8802
Epoch 59/100
1103/1103 - 2s - loss: 75.1444 - val_loss: 79.5284
Epoch 60/100
1103/1103 - 3s - loss: 74.8298 - val_loss: 77.6694
Epoch 61/100
1103/1103 - 2s - loss: 74.9866 - val_loss: 86.3382
Epoch 62/100
1103/1103 - 2s - loss: 75.3421 - val_loss: 76.4321
Epoch 63/100
1103/1103 - 2s - loss: 74.9949 - val_loss: 74.4768
Epoch 64/100
1103/1103 - 3s - loss: 74.5877 - val_loss: 75.9845
Epoch 65/100
1103/1103 - 3s - loss: 74.3773 - val_loss: 78.5946
Epoch 66/100
1103/1103 - 2s - loss: 74.4991 - val_loss: 76.2350
Epoch 67/100
1103/1103 - 2s - loss: 74.5602 - val_loss: 79.2707
Epoch 68/100
```

```
1103/1103 - 3s - loss: 74.8520 - val_loss: 77.2568
Epoch 69/100
1103/1103 - 2s - loss: 74.0970 - val_loss: 75.0852
Epoch 70/100
1103/1103 - 2s - loss: 73.8947 - val_loss: 74.9514
Epoch 71/100
1103/1103 - 2s - loss: 74.0189 - val_loss: 77.2876
Epoch 72/100
1103/1103 - 2s - loss: 73.9837 - val_loss: 80.9216
Epoch 73/100
1103/1103 - 3s - loss: 74.3308 - val_loss: 75.1939
Epoch 74/100
1103/1103 - 2s - loss: 74.0263 - val_loss: 76.5290
Epoch 75/100
1103/1103 - 2s - loss: 73.4171 - val_loss: 76.5148
Epoch 76/100
1103/1103 - 2s - loss: 73.5938 - val_loss: 76.8915
Epoch 77/100
1103/1103 - 3s - loss: 73.5207 - val_loss: 74.6526
Epoch 78/100
1103/1103 - 2s - loss: 73.7710 - val_loss: 74.9933
Epoch 79/100
1103/1103 - 2s - loss: 73.2820 - val_loss: 78.9608
Epoch 80/100
1103/1103 - 3s - loss: 73.0307 - val_loss: 76.2580
Epoch 81/100
1103/1103 - 3s - loss: 73.0324 - val_loss: 74.2666
Epoch 82/100
1103/1103 - 3s - loss: 72.8053 - val_loss: 75.2877
Epoch 83/100
1103/1103 - 3s - loss: 72.8953 - val_loss: 75.2880
Epoch 84/100
1103/1103 - 3s - loss: 73.0056 - val_loss: 74.7380
Epoch 85/100
1103/1103 - 3s - loss: 72.9746 - val_loss: 75.2932
Epoch 86/100
1103/1103 - 3s - loss: 72.9020 - val_loss: 76.4409
Epoch 87/100
1103/1103 - 2s - loss: 72.2952 - val_loss: 75.6871
Epoch 88/100
1103/1103 - 2s - loss: 72.7774 - val_loss: 75.4422
Epoch 89/100
1103/1103 - 3s - loss: 72.8038 - val_loss: 75.8537
Epoch 90/100
1103/1103 - 3s - loss: 72.3951 - val_loss: 73.4247
```

```
Epoch 91/100
1103/1103 - 2s - loss: 71.9555 - val_loss: 77.8701
Epoch 92/100
1103/1103 - 2s - loss: 72.7075 - val_loss: 74.0696
Epoch 93/100
1103/1103 - 2s - loss: 72.6290 - val_loss: 74.9616
Epoch 94/100
1103/1103 - 3s - loss: 72.2311 - val_loss: 76.5037
Epoch 95/100
1103/1103 - 2s - loss: 72.1545 - val_loss: 75.4523
Epoch 96/100
1103/1103 - 2s - loss: 71.6935 - val_loss: 73.9845
Epoch 97/100
1103/1103 - 2s - loss: 72.0296 - val_loss: 76.3727
Epoch 98/100
1103/1103 - 2s - loss: 72.2567 - val_loss: 76.3834
Epoch 99/100
1103/1103 - 3s - loss: 71.9420 - val_loss: 75.4885
Epoch 100/100
1103/1103 - 3s - loss: 71.9109 - val_loss: 76.4392
```

In [31]:

```
model.evaluate(X_test,
               y_test, verbose=0)
```

Out[31]:

```
76.43922424316406
```

In [32]:

```
mean_squared_error(y_test, model.predict(X_test), squared=False)
```

Out[32]:

```
115.27066801759108
```

In [33]:

```
sub_pred = model.predict(X_sub)

sub= pd.DataFrame(test_game_id.values)
sub["rating"] = pd.DataFrame(sub_pred)
sub.rename(columns={0:'game_id'},inplace=True)
submission = sub
submission.to_csv('submission.csv', index=False)
```