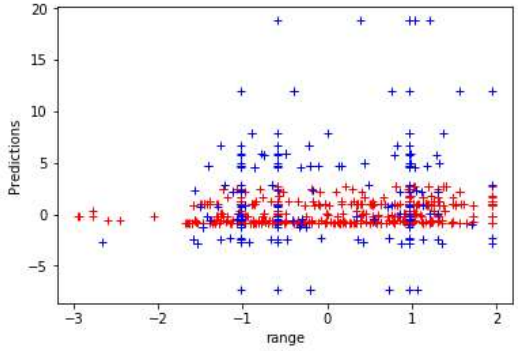
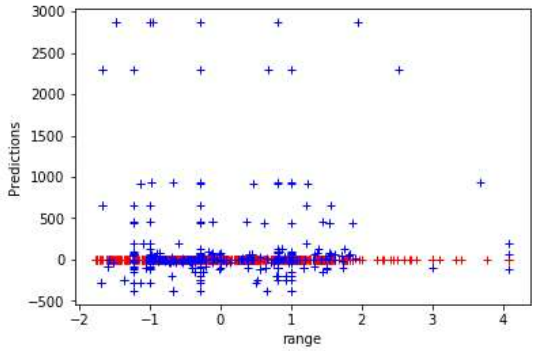
```
Python 3.6.1 |Anaconda 4.4.0 (64-bit)| (default, May 11 2017, 13:25:24) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]:

In [1]:

In [1]:

In [1]:

In [1]:

In [1]: import numpy as np

In [2]: import matplotlib.pyplot as plt

In [3]: import pandas as pd

In [4]: from math import exp

In [5]: from sklearn import gaussian_process


In [6]: from scipy import linalg as l

In [7]: from sklearn.metrics import accuracy_score

In [8]: import warnings

In [9]: warnings.filterwarnings("ignore", category=DeprecationWarning)

In [10]: ActP1 = pd.read_table("Squash1PlayerActivity.txt")

In [11]: ActP2 = pd.read_table("Squash2PlayerActivity.txt")

In [12]: PosP1 = pd.read_table("squash1Position.txt")

In [13]: PosP2 = pd.read_table("squash2Position.txt")

In [14]: act1 = ActP1

In [15]: x_tr1 = act1.iloc[:,:].values

In [16]: x_tr1 = pd.DataFrame(x_tr1)

In [17]: x_tr1 = x_tr1[[0,1,2,4,5,6,7,3]]

In [18]: X1 = pd.DataFrame(x_tr1.iloc[:, :-1].values)

In [19]: y1 = pd.DataFrame(x_tr1.iloc[:, 7].values)

In [20]: y1 =
y1.replace(["S","LL","C","CS","DL","DC","LOL","LOC","KL","KC","VLL","VC","VCS","VDL","VDC","VBN","VBO","VBR","VKL","VKC","BN","BO","BR","BS","COS"],
["1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25"])

In [21]: X1[3] = X1[3].replace(["i","l","s","t","n"],[1,2,3,4,5])

In [22]: X1[4] = X1[4].replace(["f","b"],[1,2])

In [23]: from sklearn.preprocessing import StandardScaler

In [24]: sc_X = StandardScaler()

In [25]: X1 = sc_X.fit_transform(X1)

In [26]: y1 = sc_X.fit_transform(y1)

In [27]: from sklearn.model_selection import train_test_split

In [28]: X_train1, X_test1, y_train1, y_test1 = train_test_split(X1, y1, test_size= 0.2, random_state=0)

In [29]: gp = gaussian_process.GaussianProcess(theta0=1e-2, thetaL=1e-4, thetaU=1e-1)

In [30]: gp.fit(X_train1, y_train1)
Out[30]:
GaussianProcess(beta0=None,
        corr=<function squared_exponential at 0x0000021CFF48F0D0>,
        normalize=True, nugget=array(2.220446049250313e-15),
        optimizer='fmin_cobyla', random_start=1,
        random_state=<mtrand.RandomState object at 0x0000021CFA193D80>,
        regr=<function constant at 0x0000021CFF488E18>,
        storage_mode='full', theta0=array([[ 0.01]]),
        thetaL=array([[ 0.0001]]), thetaU=array([[ 0.1]]), verbose=False)

In [31]: y_pred1, sigma2_pred1 = gp.predict(X_test1, eval_MSE=True)

In [32]: plt.plot(X_train1, y_train1, 'r+')
    ...: plt.plot(X_test1, y_pred1, 'b+')
```

```
    ...: plt.xlabel("range")
    ...: plt.ylabel("Predictions")
Out[32]: <matplotlib.text.Text at 0x21cff54b9e8>
```

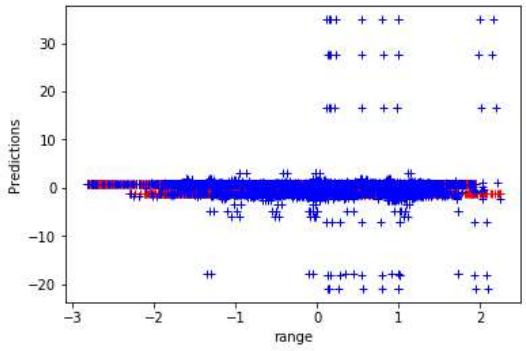



```
In [33]: act2 = ActP2

In [34]: x_tr2 = act2.iloc[:,:].values

In [35]: x_tr2 = pd.DataFrame(x_tr2)

In [36]: x_tr2 = x_tr2[[0,1,2,4,5,6,7,3]]

In [37]: X2 = pd.DataFrame(x_tr2.iloc[:, :-1].values)

In [38]: y2 = pd.DataFrame(x_tr2.iloc[:, 7].values)

In [39]: y2 =
y2.replace(["S","LL","C","CS","DL","DC","LOL","LOC","KL","KC","VLL","VC","VCS","VDL","VDC","VBN","VBO","VBR","VKL","VKC","BN","BO","BR","BS","COS"],
["1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","23","24","25"])

In [40]: X2[3] = X2[3].replace(["i","l","s","t","n"],[1,2,3,4,5])

In [41]: X2[4] = X2[4].replace(["f","b"],[1,2])

In [42]: from sklearn.preprocessing import StandardScaler

In [43]: sc_X = StandardScaler()

In [44]: X2 = sc_X.fit_transform(X2)

In [45]: y2 = sc_X.fit_transform(y2)

In [46]: from sklearn.model_selection import train_test_split

In [47]: X_train2, X_test2, y_train2, y_test2 = train_test_split(X2, y2, test_size= 0.2, random_state=0)

In [48]: gp = gaussian_process.GaussianProcess(theta0=1e-2, thetaL=1e-4, thetaU=1e-1)

In [49]: gp.fit(X_train2, y_train2)
Out[49]:
GaussianProcess(beta0=None,
        corr=<function squared_exponential at 0x0000021CFF48F0D0>,
        normalize=True, nugget=array(2.220446049250313e-15),
        optimizer='fmin_cobyla', random_start=1,
        random_state=<mtrand.RandomState object at 0x0000021CFA193D80>,
        regr=<function constant at 0x0000021CFF488E18>,
        storage_mode='full', theta0=array([[ 0.01]]),
        thetaL=array([[ 0.0001]]), thetaU=array([[ 0.1]]), verbose=False)

In [50]: y_pred2, sigma2_pred2 = gp.predict(X_test2, eval_MSE=True)

In [51]: plt.plot(X_train2, y_train2, 'r+')
    ...: plt.plot(X_test2, y_pred2, 'b+')
    ...: plt.xlabel("range")
    ...: plt.ylabel("Predictions")
Out[51]: <matplotlib.text.Text at 0x21cffe09c88>
```



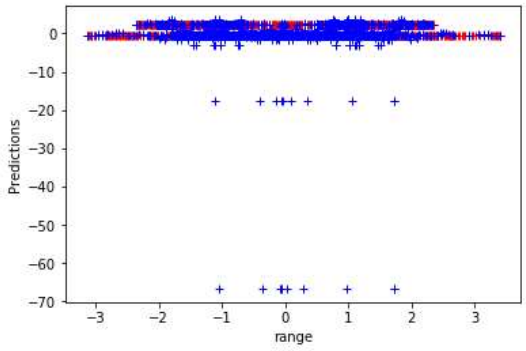

```
In [52]: x_tr3 = PosP1

In [53]: X3 = pd.DataFrame(x_tr3.iloc[:, :-1])
```

```
In [54]: y3 = pd.DataFrame(x_tr3.iloc[:, 10])

In [55]: from sklearn.preprocessing import StandardScaler

In [56]: sc_X = StandardScaler()

In [57]: X3 = sc_X.fit_transform(X3)

In [58]: y3 = sc_X.fit_transform(y3)

In [59]: from sklearn.preprocessing import Imputer

In [60]: imputer = Imputer(missing_values = 'NaN', strategy = 'mean', axis = 0)

In [61]: imputer = imputer.fit(X3[:, 0:9])

In [62]: X3[:, 0:9] = imputer.transform(X3[:, 0:9])

In [63]: imputer1 = Imputer(missing_values = 'NaN', strategy = 'mean', axis = 0)

In [64]: imputer1 = imputer1.fit(y3[:, :])

In [65]: y3[:,:] = imputer1.transform(y3[:,:])

In [66]: from sklearn.model_selection import train_test_split

In [67]: X_train3, X_test3, y_train3, y_test3 = train_test_split(X3, y3, test_size= 0.3, random_state=0)

In [68]: gp = gaussian_process.GaussianProcess(theta0=1e-2, thetaL=1e-4, thetaU=1e-1)

In [69]: gp.fit(X_train3, y_train3)
Out[69]:
GaussianProcess(beta0=None,
        corr=<function squared_exponential at 0x0000021CFF48F0D0>,
        normalize=True, nugget=array(2.220446049250313e-15),
        optimizer='fmin_cobyla', random_start=1,
        random_state=<mtrand.RandomState object at 0x0000021CFA193D80>,
        regr=<function constant at 0x0000021CFF488E18>,
        storage_mode='full', theta0=array([[ 0.01]]),
        thetaL=array([[ 0.0001]]), thetaU=array([[ 0.1]]), verbose=False)

In [70]: y_pred3, sigma2_pred3 = gp.predict(X_test3, eval_MSE=True)

In [71]: plt.plot(X_train3 , y_train3, 'r+')
    ...: plt.plot(X_test3, y_pred3,"b+")
    ...: plt.xlabel("range")
    ...: plt.ylabel("Predictions")
Out[71]: <matplotlib.text.Text at 0x21c81defac8>
```




```
In [72]: x_tr4 = PosP2

In [73]: X4 = pd.DataFrame(x_tr4.iloc[:, :-1])

In [74]: y4 = pd.DataFrame(x_tr4.iloc[:, 10])

In [75]: from sklearn.preprocessing import StandardScaler

In [76]: sc_X = StandardScaler()

In [77]: X4 = sc_X.fit_transform(X4)

In [78]: y4 = sc_X.fit_transform(y4)

In [79]: from sklearn.preprocessing import Imputer

In [80]: imputer = Imputer(missing_values = 'NaN', strategy = 'mean', axis = 0)

In [81]: imputer = imputer.fit(X4[:, 0:9])

In [82]: X4[:, 0:9] = imputer.transform(X4[:, 0:9])

In [83]: imputer1 = Imputer(missing_values = 'NaN', strategy = 'mean', axis = 0)

In [84]: imputer1 = imputer1.fit(y4[:, :])

In [85]: y4[:,:] = imputer1.transform(y4[:,:])
```

```
In [86]: from sklearn.model_selection import train_test_split

In [87]: X_train4, X_test4, y_train4, y_test4 = train_test_split(X4, y4, test_size= 0.3, random_state=0)

In [88]: gp = gaussian_process.GaussianProcess(theta0=1e-2, thetaL=1e-4, thetaU=1e-1)

In [89]: gp.fit(X_train4, y_train4)
Out[89]:
GaussianProcess(beta0=None,
        corr=<function squared_exponential at 0x0000021CFF48F0D0>,
        normalize=True, nugget=array(2.220446049250313e-15),
        optimizer='fmin_cobyla', random_start=1,
        random_state=<mtrand.RandomState object at 0x0000021CFA193D80>,
        regr=<function constant at 0x0000021CFF488E18>,
        storage_mode='full', theta0=array([[ 0.01]]),
        thetaL=array([[ 0.0001]]), thetaU=array([[ 0.1]]), verbose=False)

In [90]: y_pred4, sigma2_pred4 = gp.predict(X_test4, eval_MSE=True)

In [91]: plt.plot(X_train4 , y_train4, 'r+')
    ...: plt.plot(X_test4, y_pred4, 'b+')
    ...: plt.xlabel("range")
    ...: plt.ylabel("Predictions")
Out[91]: <matplotlib.text.Text at 0x21c8338ce10>
```




```
In [92]:
```