Decision Boundary Homework

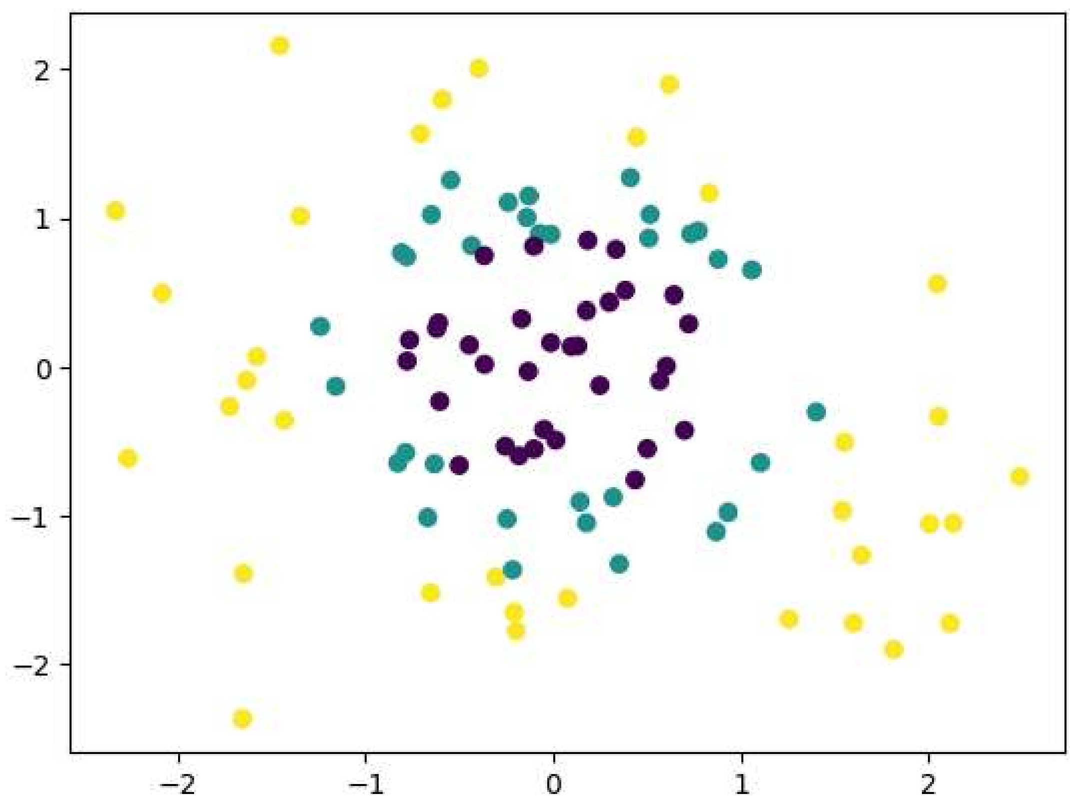
In [24]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
```

Sklearn has a couple of "standard" example data generators for use in working with classification models, these are some models used to synthesize data for practice purposes. Below are a couple of examples.

These are both pretty simple, they are using just two variables

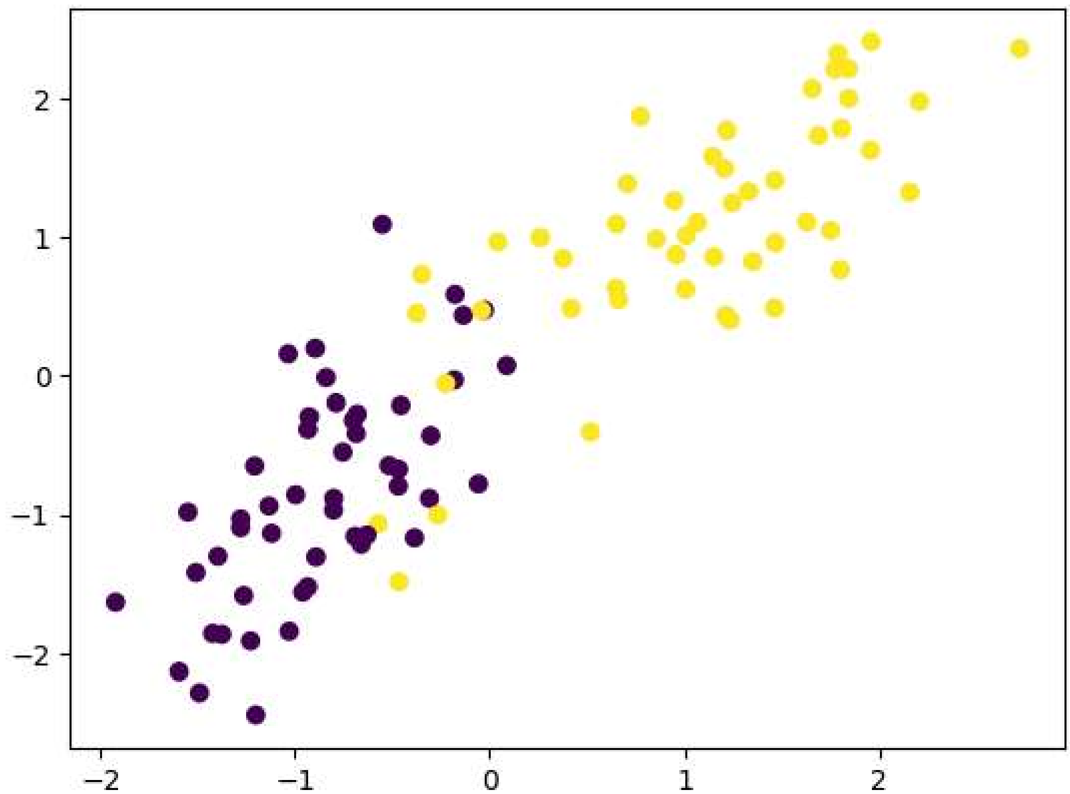
In [ ]:
```
from sklearn.datasets import make_gaussian_quantiles, make_classification

X,y=make_gaussian_quantiles(n_features=2, n_classes=3)
plt.scatter(X[:,0],X[:,1], c=y)
```

Out[ ]:
```
<matplotlib.collections.PathCollection at 0x7b47c3fd4940>
```



In [ ]:
```
X1, y1 = make_classification(
    n_features=2, n_redundant=0, n_informative=2, n_clusters_per_class=1
)
plt.scatter(X1[:,0],X1[:,1], c=y1)
```

Out[ ]: <matplotlib.collections.PathCollection at 0x7b47c1eccd60>



Question/Action

- For each of these data sets, (X,y) and (X1,y1)

a.) Generate two test sets (X_test, y_test),(X1_test,y1_test), by running the generators again, with the same parameters. This will generate test sets with the same distributions but different values.

b.) Build some a number of classifiers for each set -logistic -the logistic again, but add columns for X1^2 and X2^2 as predictors -neural net -random forest -two different boosted methods

```
Do some basic hyperparameter searches on each model (except the
logistic)
```

c.) Write a function that will allow you to pass in the model and the X, y set and then generate from that a plot of the decision boundary for the example

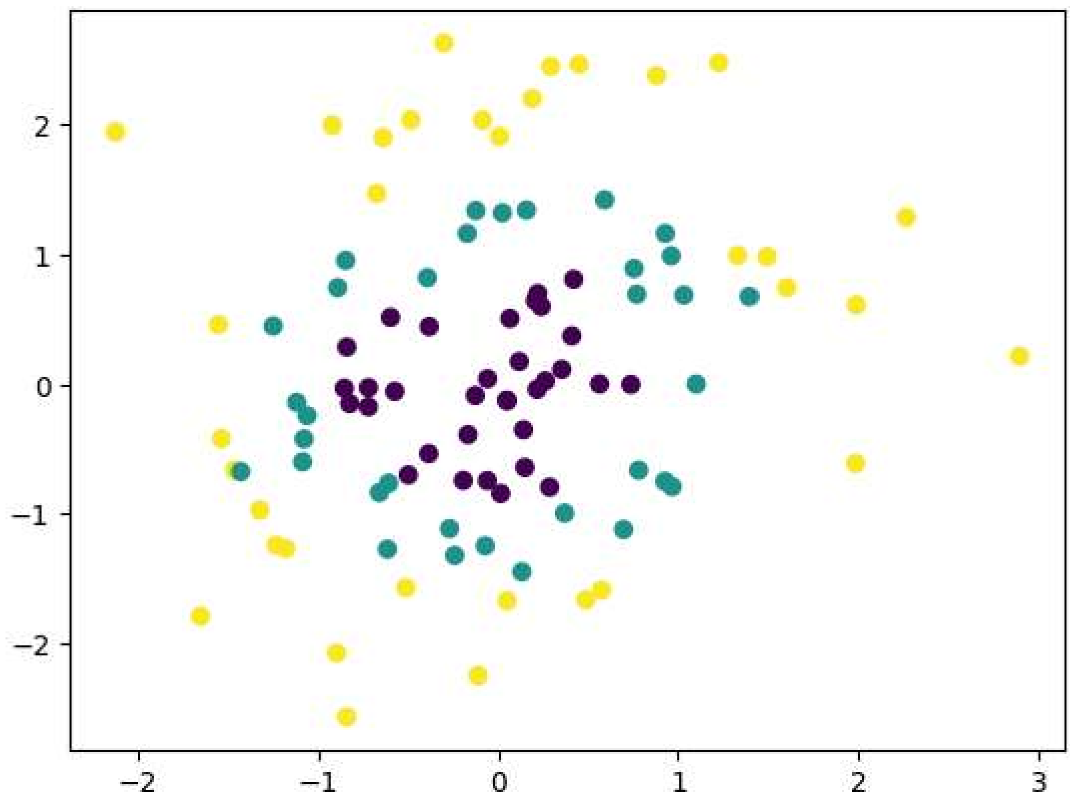
d.) Use your function to plot out the decision boundaries for each of your models and for the two data sets

e.) Based on your examination of the two data sets and the resulting plots, what can you say about the nature of the models? -which of them can generate curved boundaries? Which are all straight line boundaries? -which looks most severely overfit?

a.) Generate two test sets (X_test, y_test),(X1_test,y1_test), by running the generators again, with the same parameters. This will generate test sets with the same distributions but different values.

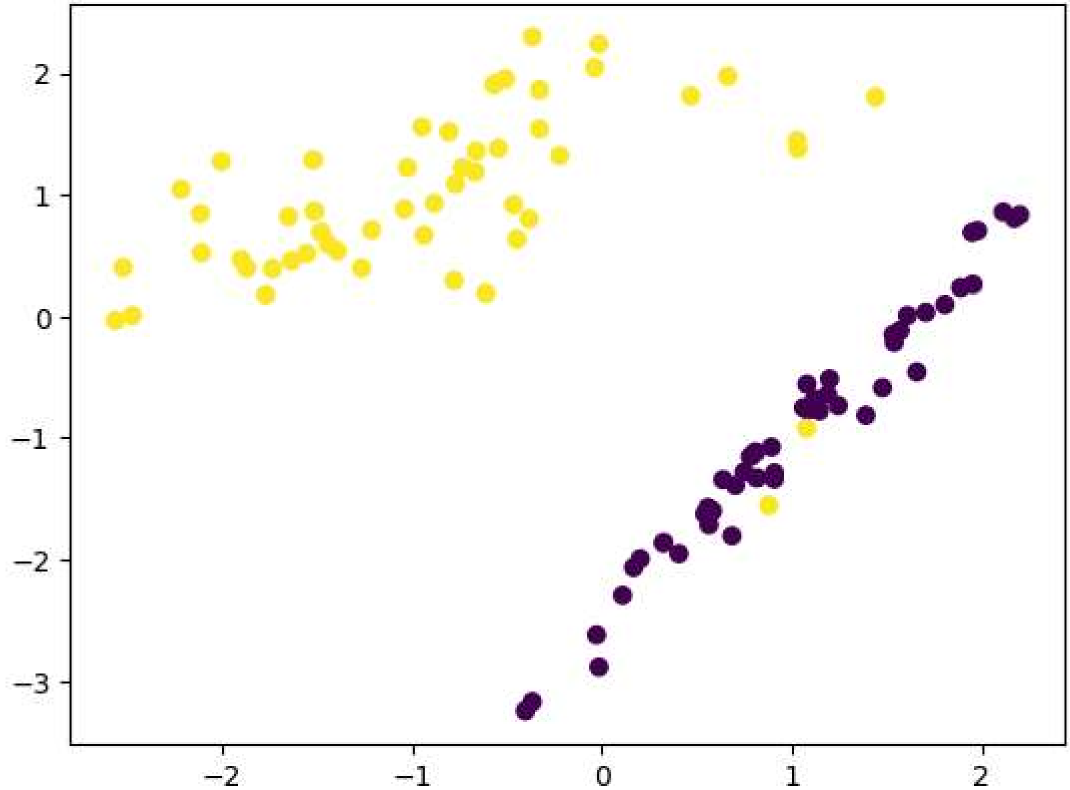
In [ ]:
```
X_test,y_test=make_gaussian_quantiles(n_features=2, n_classes=3)
plt.scatter(X_test[:,0],X_test[:,1], c=y_test)
```

Out[ ]:
```
<matplotlib.collections.PathCollection at 0x7b47c1d70e50>
```



In [ ]:
```
X1_test, y1_test = make_classification(
    n_features=2, n_redundant=0, n_informative=2, n_clusters_per_class=1
)
plt.scatter(X1_test[:,0],X1_test[:,1], c=y1_test)
```

Out[ ]:
```
<matplotlib.collections.PathCollection at 0x7b47c1de5630>
```



b.) Build some a number of classifiers for each set -logistic -the logistic again, but add columns for X1^2 and X2^2 as predictors -neural net -random forest -two different boosted methods

In [31]:
```
from sklearn.linear_model import LogisticRegression

modelLogistic1 = LogisticRegression()
# fit the model
modelLogistic1.fit(X, y)

modelLogistic1_pred = modelLogistic1.predict(X_test)
print (accuracy_score(y_test,modelLogistic1_pred))


modelLogistic2 = LogisticRegression()
# fit the model
modelLogistic2.fit(X1, y1)

modelLogistic2_pred = modelLogistic2.predict(X1_test)
print (accuracy_score(y1_test,modelLogistic2_pred))


#modelLogistic1 = LogisticRegression()
# fit the model
#modelLogistic1.fit(X_test, y_test)

#modelLogistic1 = LogisticRegression()
# fit the model
#modelLogistic1.fit(X1_test, y1_test)
```

```
0.33
0.25
```

In [78]:
```
modelLogisticV1 = LogisticRegression()
# fit the model
#modelLogisticV1.fit(np.concatenate((X,X**2),axis=1), y)
modelLogisticV1.fit(X**2, y)
modelLogisticV1_pred = modelLogisticV1.predict(X_test**2)






modelLogisticV2 = LogisticRegression()
# fit the model
#modelLogisticV2.fit(np.concatenate((X1,X1**2),axis=1), y1)
modelLogisticV2.fit(X1**2, y1)

modelLogisticV2_pred = modelLogisticV2.predict(X1_test**2)



print (accuracy_score(y_test,modelLogisticV1_pred))
print (accuracy_score(y1_test,modelLogisticV2_pred))

#modelLogisticV3 = LogisticRegression()
# fit the model
#modelLogisticV3.fit(np.concatenate((X_test,X_test**2),axis=1), y_test)

#modelLogisticV4 = LogisticRegression()
# fit the model
#modelLogisticV4.fit(np.concatenate((X1_test,X1_test**2),axis=1), y1_test)
```

```
0.95
0.47
```

In [26]:
```
# Nural Net

from sklearn.neural_network import MLPClassifier
clf = MLPClassifier(solver='adam', alpha=1e-5, random_state=1, batch_size=int(min(200/
clf.fit(X, y)


clf_pred = clf.predict(X_test)





clf1 = MLPClassifier(solver='adam', alpha=1e-5, random_state=1, batch_size=int(min(200
clf1.fit(X1, y1)

#clf2 = MLPClassifier(solver='adam', alpha=1e-5, random_state=1, batch_size=int(min(20
#clf2.fit(X1_test, y1_test)

#clf3 = MLPClassifier(solver='adam', alpha=1e-5, random_state=1, batch_size=int(min(20
#clf3.fit(X_test, y_test)
clf1_pred = clf1.predict(X1_test)


print (accuracy_score(y_test,clf_pred))
print (accuracy_score(y1_test,clf1_pred))
```

```
Iteration 1, loss = 1.25246050
Iteration 2, loss = 1.24335680
Iteration 3, loss = 1.23481095
Iteration 4, loss = 1.22736959
Iteration 5, loss = 1.21855166
Iteration 6, loss = 1.21241563
Iteration 7, loss = 1.20435838
Iteration 8, loss = 1.19792055
Iteration 9, loss = 1.19154283
Iteration 10, loss = 1.18510584
Iteration 11, loss = 1.17988901
Iteration 12, loss = 1.17455877
Iteration 13, loss = 1.16931349
Iteration 14, loss = 1.16389207
Iteration 15, loss = 1.15928525
Iteration 16, loss = 1.15448941
Iteration 17, loss = 1.15027063
Iteration 18, loss = 1.14645438
Iteration 19, loss = 1.14242135
Iteration 20, loss = 1.13865470
Iteration 21, loss = 1.13492207
Iteration 22, loss = 1.13115301
Iteration 23, loss = 1.12851923
Iteration 24, loss = 1.12577570
Iteration 25, loss = 1.12283303
Iteration 26, loss = 1.12006514
Iteration 27, loss = 1.11770219
Iteration 28, loss = 1.11504854
Iteration 29, loss = 1.11321852
Iteration 30, loss = 1.11116315
Iteration 31, loss = 1.10886888
Iteration 32, loss = 1.10662199
Iteration 33, loss = 1.10461667
Iteration 34, loss = 1.10267922
Iteration 35, loss = 1.10051415
Iteration 36, loss = 1.09915126
Iteration 37, loss = 1.09692967
Iteration 38, loss = 1.09515320
Iteration 39, loss = 1.09367074
Iteration 40, loss = 1.09208283
Iteration 41, loss = 1.09018441
Iteration 42, loss = 1.08870233
Iteration 43, loss = 1.08710978
Iteration 44, loss = 1.08564877
Iteration 45, loss = 1.08403840
Iteration 46, loss = 1.08242289
Iteration 47, loss = 1.08125746
Iteration 48, loss = 1.07965247
Iteration 49, loss = 1.07836416
Iteration 50, loss = 1.07683799
Iteration 51, loss = 1.07558825
Iteration 52, loss = 1.07425664
Iteration 53, loss = 1.07296738
Iteration 54, loss = 1.07180178
Iteration 55, loss = 1.07057006
Iteration 56, loss = 1.06926535
Iteration 57, loss = 1.06804552
Iteration 58, loss = 1.06703266
Iteration 59, loss = 1.06574486
Iteration 60, loss = 1.06464938
```

```
Iteration 61, loss = 1.06376947
Iteration 62, loss = 1.06257155
Iteration 63, loss = 1.06150681
Iteration 64, loss = 1.06042712
Iteration 65, loss = 1.05951102
Iteration 66, loss = 1.05869288
Iteration 67, loss = 1.05767156
Iteration 68, loss = 1.05675483
Iteration 69, loss = 1.05601314
Iteration 70, loss = 1.05500674
Iteration 71, loss = 1.05418258
Iteration 72, loss = 1.05339048
Iteration 73, loss = 1.05257425
Iteration 74, loss = 1.05161894
Iteration 75, loss = 1.05078457
Iteration 76, loss = 1.04998513
Iteration 77, loss = 1.04906744
Iteration 78, loss = 1.04831507
Iteration 79, loss = 1.04757111
Iteration 80, loss = 1.04681259
Iteration 81, loss = 1.04602185
Iteration 82, loss = 1.04536258
Iteration 83, loss = 1.04456407
Iteration 84, loss = 1.04386307
Iteration 85, loss = 1.04309754
Iteration 86, loss = 1.04231076
Iteration 87, loss = 1.04147983
Iteration 88, loss = 1.04072762
Iteration 89, loss = 1.03990747
Iteration 90, loss = 1.03918973
Iteration 91, loss = 1.03829139
Iteration 92, loss = 1.03743520
Iteration 93, loss = 1.03661249
Iteration 94, loss = 1.03571958
Iteration 95, loss = 1.03480495
Iteration 96, loss = 1.03351158
Iteration 97, loss = 1.03256882
Iteration 98, loss = 1.03162066
Iteration 99, loss = 1.03059322
Iteration 100, loss = 1.02969212
Iteration 101, loss = 1.02880152
Iteration 102, loss = 1.02793867
Iteration 103, loss = 1.02682509
Iteration 104, loss = 1.02585497
Iteration 105, loss = 1.02498605
Iteration 106, loss = 1.02416248
Iteration 107, loss = 1.02318925
Iteration 108, loss = 1.02239779
Iteration 109, loss = 1.02146280
Iteration 110, loss = 1.02059435
Iteration 111, loss = 1.01970278
Iteration 112, loss = 1.01885905
Iteration 113, loss = 1.01783398
Iteration 114, loss = 1.01684218
Iteration 115, loss = 1.01601598
Iteration 116, loss = 1.01533525
Iteration 117, loss = 1.01429876
Iteration 118, loss = 1.01352410
Iteration 119, loss = 1.01257397
Iteration 120, loss = 1.01176006
```

```
Iteration 121, loss = 1.01088186
Iteration 122, loss = 1.01007506
Iteration 123, loss = 1.00923703
Iteration 124, loss = 1.00848318
Iteration 125, loss = 1.00756067
Iteration 126, loss = 1.00667551
Iteration 127, loss = 1.00578605
Iteration 128, loss = 1.00480749
Iteration 129, loss = 1.00397244
Iteration 130, loss = 1.00296579
Iteration 131, loss = 1.00196110
Iteration 132, loss = 1.00100565
Iteration 133, loss = 1.00005463
Iteration 134, loss = 0.99921592
Iteration 135, loss = 0.99828906
Iteration 136, loss = 0.99729729
Iteration 137, loss = 0.99677067
Iteration 138, loss = 0.99566829
Iteration 139, loss = 0.99491031
Iteration 140, loss = 0.99402097
Iteration 141, loss = 0.99317278
Iteration 142, loss = 0.99237789
Iteration 143, loss = 0.99151866
Iteration 144, loss = 0.99067202
Iteration 145, loss = 0.98991002
Iteration 146, loss = 0.98911860
Iteration 147, loss = 0.98830747
Iteration 148, loss = 0.98745816
Iteration 149, loss = 0.98670773
Iteration 150, loss = 0.98579522
Iteration 151, loss = 0.98498269
Iteration 152, loss = 0.98418566
Iteration 153, loss = 0.98327374
Iteration 154, loss = 0.98251496
Iteration 155, loss = 0.98169703
Iteration 156, loss = 0.98093177
Iteration 157, loss = 0.98007327
Iteration 158, loss = 0.97921338
Iteration 159, loss = 0.97841163
Iteration 160, loss = 0.97758993
Iteration 161, loss = 0.97680118
Iteration 162, loss = 0.97594386
Iteration 163, loss = 0.97513875
Iteration 164, loss = 0.97429237
Iteration 165, loss = 0.97347340
Iteration 166, loss = 0.97265610
Iteration 167, loss = 0.97180321
Iteration 168, loss = 0.97095423
Iteration 169, loss = 0.97017728
Iteration 170, loss = 0.96929144
Iteration 171, loss = 0.96845797
Iteration 172, loss = 0.96753693
Iteration 173, loss = 0.96666025
Iteration 174, loss = 0.96562113
Iteration 175, loss = 0.96444573
Iteration 176, loss = 0.96348866
Iteration 177, loss = 0.96236400
Iteration 178, loss = 0.96133665
Iteration 179, loss = 0.96023444
Iteration 180, loss = 0.95915140
```

```
Iteration 181, loss = 0.95798452
Iteration 182, loss = 0.95681134
Iteration 183, loss = 0.95564288
Iteration 184, loss = 0.95440442
Iteration 185, loss = 0.95311635
Iteration 186, loss = 0.95199793
Iteration 187, loss = 0.95053759
Iteration 188, loss = 0.94895601
Iteration 189, loss = 0.94753428
Iteration 190, loss = 0.94583320
Iteration 191, loss = 0.94411383
Iteration 192, loss = 0.94234441
Iteration 193, loss = 0.94052141
Iteration 194, loss = 0.93879862
Iteration 195, loss = 0.93706875
Iteration 196, loss = 0.93524029
Iteration 197, loss = 0.93353201
Iteration 198, loss = 0.93166349
Iteration 199, loss = 0.93009383
Iteration 200, loss = 0.92829636
Iteration 201, loss = 0.92665490
Iteration 202, loss = 0.92472779
Iteration 203, loss = 0.92270620
Iteration 204, loss = 0.92075839
Iteration 205, loss = 0.91866656
Iteration 206, loss = 0.91669029
Iteration 207, loss = 0.91445150
Iteration 208, loss = 0.91242131
Iteration 209, loss = 0.91031958
Iteration 210, loss = 0.90801145
Iteration 211, loss = 0.90578181
Iteration 212, loss = 0.90352713
Iteration 213, loss = 0.90113874
Iteration 214, loss = 0.89856030
Iteration 215, loss = 0.89621610
Iteration 216, loss = 0.89387471
Iteration 217, loss = 0.89131965
Iteration 218, loss = 0.88836256
Iteration 219, loss = 0.88567837
Iteration 220, loss = 0.88299085
Iteration 221, loss = 0.87978871
Iteration 222, loss = 0.87681879
Iteration 223, loss = 0.87353602
Iteration 224, loss = 0.87033993
Iteration 225, loss = 0.86720398
Iteration 226, loss = 0.86401343
Iteration 227, loss = 0.86076694
Iteration 228, loss = 0.85766401
Iteration 229, loss = 0.85417162
Iteration 230, loss = 0.85090657
Iteration 231, loss = 0.84759383
Iteration 232, loss = 0.84409597
Iteration 233, loss = 0.84081841
Iteration 234, loss = 0.83745643
Iteration 235, loss = 0.83403210
Iteration 236, loss = 0.83051337
Iteration 237, loss = 0.82714128
Iteration 238, loss = 0.82363183
Iteration 239, loss = 0.81985192
Iteration 240, loss = 0.81632429
```

```
Iteration 241, loss = 0.81269212
Iteration 242, loss = 0.80940063
Iteration 243, loss = 0.80602842
Iteration 244, loss = 0.80256420
Iteration 245, loss = 0.79871070
Iteration 246, loss = 0.79470746
Iteration 247, loss = 0.79100777
Iteration 248, loss = 0.78732578
Iteration 249, loss = 0.78371320
Iteration 250, loss = 0.78010234
Iteration 251, loss = 0.77650641
Iteration 252, loss = 0.77266750
Iteration 253, loss = 0.76909454
Iteration 254, loss = 0.76510827
Iteration 255, loss = 0.76135255
Iteration 256, loss = 0.75744585
Iteration 257, loss = 0.75345202
Iteration 258, loss = 0.74925371
Iteration 259, loss = 0.74526405
Iteration 260, loss = 0.74130410
Iteration 261, loss = 0.73780142
Iteration 262, loss = 0.73381490
Iteration 263, loss = 0.73021481
Iteration 264, loss = 0.72640027
Iteration 265, loss = 0.72313227
Iteration 266, loss = 0.71961550
Iteration 267, loss = 0.71618800
Iteration 268, loss = 0.71262931
Iteration 269, loss = 0.70935219
Iteration 270, loss = 0.70592696
Iteration 271, loss = 0.70261301
Iteration 272, loss = 0.69936915
Iteration 273, loss = 0.69633008
Iteration 274, loss = 0.69313327
Iteration 275, loss = 0.69006159
Iteration 276, loss = 0.68720482
Iteration 277, loss = 0.68399689
Iteration 278, loss = 0.68107464
Iteration 279, loss = 0.67812959
Iteration 280, loss = 0.67536888
Iteration 281, loss = 0.67243898
Iteration 282, loss = 0.66955460
Iteration 283, loss = 0.66676219
Iteration 284, loss = 0.66404036
Iteration 285, loss = 0.66125898
Iteration 286, loss = 0.65853502
Iteration 287, loss = 0.65581331
Iteration 288, loss = 0.65318538
Iteration 289, loss = 0.65044249
Iteration 290, loss = 0.64788670
Iteration 291, loss = 0.64507930
Iteration 292, loss = 0.64237889
Iteration 293, loss = 0.63946372
Iteration 294, loss = 0.63675170
Iteration 295, loss = 0.63396915
Iteration 296, loss = 0.63121466
Iteration 297, loss = 0.62857609
Iteration 298, loss = 0.62598885
Iteration 299, loss = 0.62327866
Iteration 300, loss = 0.62063084
```

```
Iteration 301, loss = 0.61795710
Iteration 302, loss = 0.61524251
Iteration 303, loss = 0.61225498
Iteration 304, loss = 0.60948855
Iteration 305, loss = 0.60670045
Iteration 306, loss = 0.60334705
Iteration 307, loss = 0.60022078
Iteration 308, loss = 0.59732327
Iteration 309, loss = 0.59420109
Iteration 310, loss = 0.59094983
Iteration 311, loss = 0.58736637
Iteration 312, loss = 0.58393715
Iteration 313, loss = 0.58030949
Iteration 314, loss = 0.57654245
Iteration 315, loss = 0.57303644
Iteration 316, loss = 0.56968333
Iteration 317, loss = 0.56562160
Iteration 318, loss = 0.56211716
Iteration 319, loss = 0.55835544
Iteration 320, loss = 0.55473341
Iteration 321, loss = 0.55121182
Iteration 322, loss = 0.54749239
Iteration 323, loss = 0.54374510
Iteration 324, loss = 0.54040433
Iteration 325, loss = 0.53716188
Iteration 326, loss = 0.53413763
Iteration 327, loss = 0.53088312
Iteration 328, loss = 0.52798599
Iteration 329, loss = 0.52499483
Iteration 330, loss = 0.52180490
Iteration 331, loss = 0.51883649
Iteration 332, loss = 0.51605874
Iteration 333, loss = 0.51337016
Iteration 334, loss = 0.51042294
Iteration 335, loss = 0.50783346
Iteration 336, loss = 0.50515479
Iteration 337, loss = 0.50240644
Iteration 338, loss = 0.50003404
Iteration 339, loss = 0.49760135
Iteration 340, loss = 0.49533649
Iteration 341, loss = 0.49292186
Iteration 342, loss = 0.49064372
Iteration 343, loss = 0.48842480
Iteration 344, loss = 0.48610183
Iteration 345, loss = 0.48388228
Iteration 346, loss = 0.48163669
Iteration 347, loss = 0.47954123
Iteration 348, loss = 0.47731664
Iteration 349, loss = 0.47519864
Iteration 350, loss = 0.47306697
Iteration 351, loss = 0.47109854
Iteration 352, loss = 0.46900936
Iteration 353, loss = 0.46698574
Iteration 354, loss = 0.46487589
Iteration 355, loss = 0.46289387
Iteration 356, loss = 0.46088857
Iteration 357, loss = 0.45893318
Iteration 358, loss = 0.45705027
Iteration 359, loss = 0.45503446
Iteration 360, loss = 0.45311274
```

```
Iteration 361, loss = 0.45134791
Iteration 362, loss = 0.44933122
Iteration 363, loss = 0.44755648
Iteration 364, loss = 0.44568349
Iteration 365, loss = 0.44390129
Iteration 366, loss = 0.44211344
Iteration 367, loss = 0.44029503
Iteration 368, loss = 0.43857693
Iteration 369, loss = 0.43687823
Iteration 370, loss = 0.43511531
Iteration 371, loss = 0.43344582
Iteration 372, loss = 0.43174686
Iteration 373, loss = 0.42999202
Iteration 374, loss = 0.42836459
Iteration 375, loss = 0.42677147
Iteration 376, loss = 0.42505300
Iteration 377, loss = 0.42341419
Iteration 378, loss = 0.42175519
Iteration 379, loss = 0.42024941
Iteration 380, loss = 0.41850656
Iteration 381, loss = 0.41703030
Iteration 382, loss = 0.41553837
Iteration 383, loss = 0.41377140
Iteration 384, loss = 0.41232192
Iteration 385, loss = 0.41073255
Iteration 386, loss = 0.40920350
Iteration 387, loss = 0.40765060
Iteration 388, loss = 0.40621153
Iteration 389, loss = 0.40469259
Iteration 390, loss = 0.40319351
Iteration 391, loss = 0.40171260
Iteration 392, loss = 0.40023442
Iteration 393, loss = 0.39883728
Iteration 394, loss = 0.39743231
Iteration 395, loss = 0.39589152
Iteration 396, loss = 0.39457322
Iteration 397, loss = 0.39294224
Iteration 398, loss = 0.39176063
Iteration 399, loss = 0.39007823
Iteration 400, loss = 0.38872427
Iteration 401, loss = 0.38728616
Iteration 402, loss = 0.38589506
Iteration 403, loss = 0.38450966
Iteration 404, loss = 0.38320726
Iteration 405, loss = 0.38166324
Iteration 406, loss = 0.38042261
Iteration 407, loss = 0.37893247
Iteration 408, loss = 0.37756568
Iteration 409, loss = 0.37641409
Iteration 410, loss = 0.37478365
Iteration 411, loss = 0.37347335
Iteration 412, loss = 0.37214663
Iteration 413, loss = 0.37086523
Iteration 414, loss = 0.36959203
Iteration 415, loss = 0.36815889
Iteration 416, loss = 0.36676315
Iteration 417, loss = 0.36562464
Iteration 418, loss = 0.36423473
Iteration 419, loss = 0.36296037
Iteration 420, loss = 0.36158967
```

```
Iteration 421, loss = 0.36043251
Iteration 422, loss = 0.35911027
Iteration 423, loss = 0.35780567
Iteration 424, loss = 0.35650193
Iteration 425, loss = 0.35529227
Iteration 426, loss = 0.35406156
Iteration 427, loss = 0.35274048
Iteration 428, loss = 0.35170572
Iteration 429, loss = 0.35040617
Iteration 430, loss = 0.34935085
Iteration 431, loss = 0.34793200
Iteration 432, loss = 0.34673417
Iteration 433, loss = 0.34545587
Iteration 434, loss = 0.34434271
Iteration 435, loss = 0.34307177
Iteration 436, loss = 0.34184552
Iteration 437, loss = 0.34057656
Iteration 438, loss = 0.33946354
Iteration 439, loss = 0.33817137
Iteration 440, loss = 0.33698193
Iteration 441, loss = 0.33588223
Iteration 442, loss = 0.33456061
Iteration 443, loss = 0.33347324
Iteration 444, loss = 0.33224789
Iteration 445, loss = 0.33104476
Iteration 446, loss = 0.32999480
Iteration 447, loss = 0.32870302
Iteration 448, loss = 0.32763010
Iteration 449, loss = 0.32644933
Iteration 450, loss = 0.32529945
Iteration 451, loss = 0.32415104
Iteration 452, loss = 0.32304815
Iteration 453, loss = 0.32186662
Iteration 454, loss = 0.32083992
Iteration 455, loss = 0.31991740
Iteration 456, loss = 0.31866522
Iteration 457, loss = 0.31765736
Iteration 458, loss = 0.31642076
Iteration 459, loss = 0.31537140
Iteration 460, loss = 0.31434536
Iteration 461, loss = 0.31325710
Iteration 462, loss = 0.31221688
Iteration 463, loss = 0.31131310
Iteration 464, loss = 0.30995750
Iteration 465, loss = 0.30893565
Iteration 466, loss = 0.30805711
Iteration 467, loss = 0.30687594
Iteration 468, loss = 0.30589578
Iteration 469, loss = 0.30488244
Iteration 470, loss = 0.30377849
Iteration 471, loss = 0.30280530
Iteration 472, loss = 0.30182144
Iteration 473, loss = 0.30089741
Iteration 474, loss = 0.29978604
Iteration 475, loss = 0.29885540
Iteration 476, loss = 0.29770771
Iteration 477, loss = 0.29688882
Iteration 478, loss = 0.29578367
Iteration 479, loss = 0.29492760
Iteration 480, loss = 0.29380054
```

```
Iteration 481, loss = 0.29303221
Iteration 482, loss = 0.29184185
Iteration 483, loss = 0.29094937
Iteration 484, loss = 0.29001010
Iteration 485, loss = 0.28897376
Iteration 486, loss = 0.28806235
Iteration 487, loss = 0.28752441
Iteration 488, loss = 0.28622409
Iteration 489, loss = 0.28543858
Iteration 490, loss = 0.28440109
Iteration 491, loss = 0.28348468
Iteration 492, loss = 0.28254008
Iteration 493, loss = 0.28166008
Iteration 494, loss = 0.28072589
Iteration 495, loss = 0.27988676
Iteration 496, loss = 0.27895817
Iteration 497, loss = 0.27803368
Iteration 498, loss = 0.27718542
Iteration 499, loss = 0.27639075
Iteration 500, loss = 0.27541074
Iteration 1, loss = 0.90172150
Iteration 2, loss = 0.89022423
Iteration 3, loss = 0.87837307
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/neural_network/_multilayer_perceptro
n.py:686: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (500) reached
and the optimization hasn't converged yet.
  warnings.warn(
```

```
Iteration 4, loss = 0.86787407
Iteration 5, loss = 0.85597044
Iteration 6, loss = 0.84550253
Iteration 7, loss = 0.83392755
Iteration 8, loss = 0.82307967
Iteration 9, loss = 0.81202447
Iteration 10, loss = 0.80132376
Iteration 11, loss = 0.79016590
Iteration 12, loss = 0.77962600
Iteration 13, loss = 0.76890119
Iteration 14, loss = 0.75822536
Iteration 15, loss = 0.74738611
Iteration 16, loss = 0.73733969
Iteration 17, loss = 0.72669745
Iteration 18, loss = 0.71637174
Iteration 19, loss = 0.70604750
Iteration 20, loss = 0.69583577
Iteration 21, loss = 0.68574226
Iteration 22, loss = 0.67631065
Iteration 23, loss = 0.66651566
Iteration 24, loss = 0.65715326
Iteration 25, loss = 0.64756990
Iteration 26, loss = 0.63782130
Iteration 27, loss = 0.62857612
Iteration 28, loss = 0.61919359
Iteration 29, loss = 0.61004063
Iteration 30, loss = 0.60090847
Iteration 31, loss = 0.59212838
Iteration 32, loss = 0.58299690
Iteration 33, loss = 0.57400075
Iteration 34, loss = 0.56544869
Iteration 35, loss = 0.55699335
Iteration 36, loss = 0.54837480
Iteration 37, loss = 0.54044959
Iteration 38, loss = 0.53188619
Iteration 39, loss = 0.52398105
Iteration 40, loss = 0.51598926
Iteration 41, loss = 0.50809546
Iteration 42, loss = 0.50047571
Iteration 43, loss = 0.49310287
Iteration 44, loss = 0.48583752
Iteration 45, loss = 0.47849239
Iteration 46, loss = 0.47098148
Iteration 47, loss = 0.46388068
Iteration 48, loss = 0.45709826
Iteration 49, loss = 0.45042680
Iteration 50, loss = 0.44345459
Iteration 51, loss = 0.43704702
Iteration 52, loss = 0.43063059
Iteration 53, loss = 0.42448115
Iteration 54, loss = 0.41794371
Iteration 55, loss = 0.41196418
Iteration 56, loss = 0.40587786
Iteration 57, loss = 0.40006769
Iteration 58, loss = 0.39453266
Iteration 59, loss = 0.38894264
Iteration 60, loss = 0.38331233
Iteration 61, loss = 0.37806819
Iteration 62, loss = 0.37309259
Iteration 63, loss = 0.36819002
```

```
Iteration 64, loss = 0.36308601
Iteration 65, loss = 0.35843562
Iteration 66, loss = 0.35391359
Iteration 67, loss = 0.34925153
Iteration 68, loss = 0.34512466
Iteration 69, loss = 0.34099783
Iteration 70, loss = 0.33682958
Iteration 71, loss = 0.33296512
Iteration 72, loss = 0.32902943
Iteration 73, loss = 0.32538156
Iteration 74, loss = 0.32167931
Iteration 75, loss = 0.31818290
Iteration 76, loss = 0.31465003
Iteration 77, loss = 0.31143623
Iteration 78, loss = 0.30814236
Iteration 79, loss = 0.30483761
Iteration 80, loss = 0.30173857
Iteration 81, loss = 0.29863597
Iteration 82, loss = 0.29570611
Iteration 83, loss = 0.29310765
Iteration 84, loss = 0.29025009
Iteration 85, loss = 0.28769216
Iteration 86, loss = 0.28513593
Iteration 87, loss = 0.28252883
Iteration 88, loss = 0.28017429
Iteration 89, loss = 0.27770780
Iteration 90, loss = 0.27552167
Iteration 91, loss = 0.27343011
Iteration 92, loss = 0.27122989
Iteration 93, loss = 0.26927613
Iteration 94, loss = 0.26721234
Iteration 95, loss = 0.26537001
Iteration 96, loss = 0.26342485
Iteration 97, loss = 0.26171434
Iteration 98, loss = 0.25988487
Iteration 99, loss = 0.25815412
Iteration 100, loss = 0.25657959
Iteration 101, loss = 0.25497198
Iteration 102, loss = 0.25345878
Iteration 103, loss = 0.25198095
Iteration 104, loss = 0.25048446
Iteration 105, loss = 0.24907055
Iteration 106, loss = 0.24765870
Iteration 107, loss = 0.24638124
Iteration 108, loss = 0.24502921
Iteration 109, loss = 0.24381655
Iteration 110, loss = 0.24270990
Iteration 111, loss = 0.24147300
Iteration 112, loss = 0.24045012
Iteration 113, loss = 0.23922968
Iteration 114, loss = 0.23821360
Iteration 115, loss = 0.23712818
Iteration 116, loss = 0.23612694
Iteration 117, loss = 0.23515279
Iteration 118, loss = 0.23420417
Iteration 119, loss = 0.23314215
Iteration 120, loss = 0.23235412
Iteration 121, loss = 0.23145059
Iteration 122, loss = 0.23055873
Iteration 123, loss = 0.22971299
```

```
Iteration 124, loss = 0.22899793
Iteration 125, loss = 0.22805565
Iteration 126, loss = 0.22734196
Iteration 127, loss = 0.22658892
Iteration 128, loss = 0.22583340
Iteration 129, loss = 0.22510805
Iteration 130, loss = 0.22440910
Iteration 131, loss = 0.22372110
Iteration 132, loss = 0.22301739
Iteration 133, loss = 0.22235932
Iteration 134, loss = 0.22167840
Iteration 135, loss = 0.22104988
Iteration 136, loss = 0.22043098
Iteration 137, loss = 0.21984548
Iteration 138, loss = 0.21919507
Iteration 139, loss = 0.21860652
Iteration 140, loss = 0.21813690
Iteration 141, loss = 0.21754047
Iteration 142, loss = 0.21698099
Iteration 143, loss = 0.21644561
Iteration 144, loss = 0.21590977
Iteration 145, loss = 0.21538919
Iteration 146, loss = 0.21486248
Iteration 147, loss = 0.21443452
Iteration 148, loss = 0.21391915
Iteration 149, loss = 0.21341183
Iteration 150, loss = 0.21293316
Iteration 151, loss = 0.21249172
Iteration 152, loss = 0.21208916
Iteration 153, loss = 0.21157849
Iteration 154, loss = 0.21115764
Iteration 155, loss = 0.21072836
Iteration 156, loss = 0.21032432
Iteration 157, loss = 0.20986477
Iteration 158, loss = 0.20942901
Iteration 159, loss = 0.20904668
Iteration 160, loss = 0.20863650
Iteration 161, loss = 0.20820638
Iteration 162, loss = 0.20778327
Iteration 163, loss = 0.20743898
Iteration 164, loss = 0.20699071
Iteration 165, loss = 0.20660722
Iteration 166, loss = 0.20622015
Iteration 167, loss = 0.20584862
Iteration 168, loss = 0.20545800
Iteration 169, loss = 0.20505275
Iteration 170, loss = 0.20465400
Iteration 171, loss = 0.20426338
Iteration 172, loss = 0.20388146
Iteration 173, loss = 0.20346449
Iteration 174, loss = 0.20310845
Iteration 175, loss = 0.20271947
Iteration 176, loss = 0.20233295
Iteration 177, loss = 0.20197111
Iteration 178, loss = 0.20160998
Iteration 179, loss = 0.20124394
Iteration 180, loss = 0.20096854
Iteration 181, loss = 0.20059708
Iteration 182, loss = 0.20029779
Iteration 183, loss = 0.19995601
```

```
Iteration 184, loss = 0.19966196
Iteration 185, loss = 0.19936112
Iteration 186, loss = 0.19907626
Iteration 187, loss = 0.19875019
Iteration 188, loss = 0.19848010
Iteration 189, loss = 0.19822437
Iteration 190, loss = 0.19798778
Iteration 191, loss = 0.19764600
Iteration 192, loss = 0.19738560
Iteration 193, loss = 0.19715135
Iteration 194, loss = 0.19685127
Iteration 195, loss = 0.19662033
Iteration 196, loss = 0.19637319
Iteration 197, loss = 0.19608474
Iteration 198, loss = 0.19585817
Iteration 199, loss = 0.19560036
Iteration 200, loss = 0.19536424
Iteration 201, loss = 0.19506658
Iteration 202, loss = 0.19483513
Iteration 203, loss = 0.19465730
Iteration 204, loss = 0.19442696
Iteration 205, loss = 0.19414657
Iteration 206, loss = 0.19394085
Iteration 207, loss = 0.19369135
Iteration 208, loss = 0.19347540
Iteration 209, loss = 0.19324261
Iteration 210, loss = 0.19302435
Iteration 211, loss = 0.19279108
Iteration 212, loss = 0.19262603
Iteration 213, loss = 0.19236214
Iteration 214, loss = 0.19218930
Iteration 215, loss = 0.19193753
Iteration 216, loss = 0.19171067
Iteration 217, loss = 0.19149766
Iteration 218, loss = 0.19130936
Iteration 219, loss = 0.19115226
Iteration 220, loss = 0.19089827
Iteration 221, loss = 0.19073447
Iteration 222, loss = 0.19048503
Iteration 223, loss = 0.19030497
Iteration 224, loss = 0.19011103
Iteration 225, loss = 0.18989381
Iteration 226, loss = 0.18969044
Iteration 227, loss = 0.18952567
Iteration 228, loss = 0.18931603
Iteration 229, loss = 0.18912446
Iteration 230, loss = 0.18895619
Iteration 231, loss = 0.18876078
Iteration 232, loss = 0.18858598
Iteration 233, loss = 0.18841593
Iteration 234, loss = 0.18819411
Iteration 235, loss = 0.18803231
Iteration 236, loss = 0.18790028
Iteration 237, loss = 0.18765407
Iteration 238, loss = 0.18749095
Iteration 239, loss = 0.18730659
Iteration 240, loss = 0.18714834
Iteration 241, loss = 0.18696131
Iteration 242, loss = 0.18676275
Iteration 243, loss = 0.18659962
```

```
Iteration 244, loss = 0.18644957
Iteration 245, loss = 0.18628634
Iteration 246, loss = 0.18613208
Iteration 247, loss = 0.18596496
Iteration 248, loss = 0.18579790
Iteration 249, loss = 0.18565722
Iteration 250, loss = 0.18550551
Iteration 251, loss = 0.18536298
Iteration 252, loss = 0.18521539
Iteration 253, loss = 0.18506336
Iteration 254, loss = 0.18495400
Iteration 255, loss = 0.18484377
Iteration 256, loss = 0.18462232
Iteration 257, loss = 0.18448791
Iteration 258, loss = 0.18436258
Iteration 259, loss = 0.18421035
Iteration 260, loss = 0.18410745
Iteration 261, loss = 0.18392722
Iteration 262, loss = 0.18381806
Iteration 263, loss = 0.18367992
Iteration 264, loss = 0.18355847
Iteration 265, loss = 0.18339955
Iteration 266, loss = 0.18332897
Iteration 267, loss = 0.18311966
Iteration 268, loss = 0.18300150
Iteration 269, loss = 0.18290972
Iteration 270, loss = 0.18278862
Iteration 271, loss = 0.18261040
Iteration 272, loss = 0.18247634
Iteration 273, loss = 0.18239460
Iteration 274, loss = 0.18227639
Iteration 275, loss = 0.18211863
Iteration 276, loss = 0.18199094
Iteration 277, loss = 0.18184804
Iteration 278, loss = 0.18173616
Iteration 279, loss = 0.18159880
Iteration 280, loss = 0.18144893
Iteration 281, loss = 0.18136770
Iteration 282, loss = 0.18118460
Iteration 283, loss = 0.18108293
Iteration 284, loss = 0.18098576
Iteration 285, loss = 0.18079904
Iteration 286, loss = 0.18068134
Iteration 287, loss = 0.18061105
Iteration 288, loss = 0.18045736
Iteration 289, loss = 0.18031751
Iteration 290, loss = 0.18021483
Iteration 291, loss = 0.18015679
Iteration 292, loss = 0.18000878
Iteration 293, loss = 0.17986524
Iteration 294, loss = 0.17978824
Iteration 295, loss = 0.17965517
Iteration 296, loss = 0.17951351
Iteration 297, loss = 0.17943511
Iteration 298, loss = 0.17928565
Iteration 299, loss = 0.17918786
Iteration 300, loss = 0.17906745
Iteration 301, loss = 0.17896505
Iteration 302, loss = 0.17891453
Iteration 303, loss = 0.17872975
```

```
Iteration 304, loss = 0.17862732
Iteration 305, loss = 0.17851900
Iteration 306, loss = 0.17842198
Iteration 307, loss = 0.17832292
Iteration 308, loss = 0.17824067
Iteration 309, loss = 0.17815589
Iteration 310, loss = 0.17811244
Iteration 311, loss = 0.17792543
Iteration 312, loss = 0.17781699
Iteration 313, loss = 0.17770112
Iteration 314, loss = 0.17760548
Iteration 315, loss = 0.17751990
Iteration 316, loss = 0.17741562
Iteration 317, loss = 0.17733624
Iteration 318, loss = 0.17723507
Iteration 319, loss = 0.17715045
Iteration 320, loss = 0.17703931
Iteration 321, loss = 0.17693450
Iteration 322, loss = 0.17683024
Iteration 323, loss = 0.17673687
Iteration 324, loss = 0.17660708
Iteration 325, loss = 0.17651185
Iteration 326, loss = 0.17642083
Iteration 327, loss = 0.17634481
Iteration 328, loss = 0.17623692
Iteration 329, loss = 0.17614825
Iteration 330, loss = 0.17605916
Iteration 331, loss = 0.17593283
Iteration 332, loss = 0.17586546
Iteration 333, loss = 0.17575322
Iteration 334, loss = 0.17564652
Iteration 335, loss = 0.17557936
Iteration 336, loss = 0.17546367
Iteration 337, loss = 0.17537302
Iteration 338, loss = 0.17528205
Iteration 339, loss = 0.17520446
Iteration 340, loss = 0.17509713
Iteration 341, loss = 0.17501915
Iteration 342, loss = 0.17491669
Iteration 343, loss = 0.17483368
Iteration 344, loss = 0.17471180
Iteration 345, loss = 0.17464744
Iteration 346, loss = 0.17453564
Iteration 347, loss = 0.17443074
Iteration 348, loss = 0.17435032
Iteration 349, loss = 0.17428403
Iteration 350, loss = 0.17417594
Iteration 351, loss = 0.17408033
Iteration 352, loss = 0.17414560
Iteration 353, loss = 0.17396530
Iteration 354, loss = 0.17383976
Iteration 355, loss = 0.17370746
Iteration 356, loss = 0.17368203
Iteration 357, loss = 0.17353041
Iteration 358, loss = 0.17344188
Iteration 359, loss = 0.17336586
Iteration 360, loss = 0.17326491
Iteration 361, loss = 0.17316592
Iteration 362, loss = 0.17307031
Iteration 363, loss = 0.17299617
```

```
Iteration 364, loss = 0.17293813
Iteration 365, loss = 0.17281136
Iteration 366, loss = 0.17272162
Iteration 367, loss = 0.17263215
Iteration 368, loss = 0.17253899
Iteration 369, loss = 0.17245925
Iteration 370, loss = 0.17235941
Iteration 371, loss = 0.17231334
Iteration 372, loss = 0.17220007
Iteration 373, loss = 0.17208679
Iteration 374, loss = 0.17201301
Iteration 375, loss = 0.17191421
Iteration 376, loss = 0.17187719
Iteration 377, loss = 0.17174427
Iteration 378, loss = 0.17168585
Iteration 379, loss = 0.17155223
Iteration 380, loss = 0.17154276
Iteration 381, loss = 0.17138146
Iteration 382, loss = 0.17129246
Iteration 383, loss = 0.17120438
Iteration 384, loss = 0.17114636
Iteration 385, loss = 0.17103029
Iteration 386, loss = 0.17094099
Iteration 387, loss = 0.17089329
Iteration 388, loss = 0.17076650
Iteration 389, loss = 0.17066130
Iteration 390, loss = 0.17060802
Iteration 391, loss = 0.17049104
Iteration 392, loss = 0.17042682
Iteration 393, loss = 0.17031588
Iteration 394, loss = 0.17022187
Iteration 395, loss = 0.17013178
Iteration 396, loss = 0.17004526
Iteration 397, loss = 0.16996494
Iteration 398, loss = 0.16990118
Iteration 399, loss = 0.16974774
Iteration 400, loss = 0.16968204
Iteration 401, loss = 0.16956225
Iteration 402, loss = 0.16950047
Iteration 403, loss = 0.16939219
Iteration 404, loss = 0.16929554
Iteration 405, loss = 0.16920859
Iteration 406, loss = 0.16910722
Iteration 407, loss = 0.16903283
Iteration 408, loss = 0.16893831
Iteration 409, loss = 0.16885312
Iteration 410, loss = 0.16876734
Iteration 411, loss = 0.16864815
Iteration 412, loss = 0.16855915
Iteration 413, loss = 0.16845375
Iteration 414, loss = 0.16838974
Iteration 415, loss = 0.16828123
Iteration 416, loss = 0.16821643
Iteration 417, loss = 0.16809555
Iteration 418, loss = 0.16800059
Iteration 419, loss = 0.16791657
Iteration 420, loss = 0.16782404
Iteration 421, loss = 0.16781243
Iteration 422, loss = 0.16771251
Iteration 423, loss = 0.16755036
```

```
Iteration 424, loss = 0.16746489
Iteration 425, loss = 0.16736350
Iteration 426, loss = 0.16730376
Iteration 427, loss = 0.16717460
Iteration 428, loss = 0.16709183
Iteration 429, loss = 0.16700363
Iteration 430, loss = 0.16695806
Iteration 431, loss = 0.16682427
Iteration 432, loss = 0.16673714
Iteration 433, loss = 0.16663189
Iteration 434, loss = 0.16655727
Iteration 435, loss = 0.16647746
Iteration 436, loss = 0.16638905
Iteration 437, loss = 0.16632380
Iteration 438, loss = 0.16618744
Iteration 439, loss = 0.16614146
Iteration 440, loss = 0.16602025
Iteration 441, loss = 0.16596043
Iteration 442, loss = 0.16582270
Iteration 443, loss = 0.16577367
Iteration 444, loss = 0.16566223
Iteration 445, loss = 0.16556865
Iteration 446, loss = 0.16548176
Iteration 447, loss = 0.16542248
Iteration 448, loss = 0.16531945
Iteration 449, loss = 0.16520770
Iteration 450, loss = 0.16512702
Iteration 451, loss = 0.16508345
Iteration 452, loss = 0.16497028
Iteration 453, loss = 0.16494078
Iteration 454, loss = 0.16478455
Iteration 455, loss = 0.16468803
Iteration 456, loss = 0.16462460
Iteration 457, loss = 0.16452188
Iteration 458, loss = 0.16443972
Iteration 459, loss = 0.16437994
Iteration 460, loss = 0.16424791
Iteration 461, loss = 0.16416216
Iteration 462, loss = 0.16404108
Iteration 463, loss = 0.16394059
Iteration 464, loss = 0.16385305
Iteration 465, loss = 0.16378331
Iteration 466, loss = 0.16366400
Iteration 467, loss = 0.16356027
Iteration 468, loss = 0.16347727
Iteration 469, loss = 0.16340180
Iteration 470, loss = 0.16327755
Iteration 471, loss = 0.16321367
Iteration 472, loss = 0.16308082
Iteration 473, loss = 0.16297716
Iteration 474, loss = 0.16285192
Iteration 475, loss = 0.16276242
Iteration 476, loss = 0.16268870
Iteration 477, loss = 0.16255953
Iteration 478, loss = 0.16244625
Iteration 479, loss = 0.16238222
Iteration 480, loss = 0.16225985
Iteration 481, loss = 0.16219688
Iteration 482, loss = 0.16207948
Iteration 483, loss = 0.16204515
```

```
Iteration 484, loss = 0.16194338
Iteration 485, loss = 0.16180692
Iteration 486, loss = 0.16173883
Iteration 487, loss = 0.16163632
Iteration 488, loss = 0.16160413
Iteration 489, loss = 0.16151269
Iteration 490, loss = 0.16138770
Iteration 491, loss = 0.16134984
Iteration 492, loss = 0.16122830
Iteration 493, loss = 0.16124558
Iteration 494, loss = 0.16104789
Iteration 495, loss = 0.16094218
Iteration 496, loss = 0.16092323
Iteration 497, loss = 0.16076925
Iteration 498, loss = 0.16067836
Iteration 499, loss = 0.16060737
Iteration 500, loss = 0.16054915
0.93
0.34
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/neural_network/_multilayer_perceptro
n.py:686: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (500) reached
and the optimization hasn't converged yet.
  warnings.warn(
```

In [60]:

```python
#-random forest

from sklearn.ensemble import RandomForestClassifier

rnd_clf = RandomForestClassifier(n_estimators=500, max_leaf_nodes=50, n_jobs=-1)
rnd_clf.fit(X, y)



y_pred_rf = rnd_clf.predict(X_test)

print (sum(y_pred_rf==y_test)/y_test.size)
```

```
0.88
```

In [61]:

```python
rnd_clf2 = RandomForestClassifier(n_estimators=500, max_leaf_nodes=8, n_jobs=-1)
rnd_clf2.fit(X1, y1)


y_pred_rf = rnd_clf.predict(X1_test)

print (sum(y_pred_rf==y1_test)/y1_test.size)
```

```
0.14
```

In [52]:

```python
# Gradient Boosting


from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
gb_clf=GradientBoostingClassifier(learning_rate=0.1, n_estimators=100, subsample=1, ma

gb_clf.fit(X, y)

y_pred_gb = rnd_clf.predict(X_test)
print (accuracy_score(y_test,y_pred_gb))
```

```
0.42
```

In [53]:
```python
# Gradient Boosting2


from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
gb_clf2=GradientBoostingClassifier(learning_rate=0.1, n_estimators=100, subsample=1, m

gb_clf2.fit(X1, y1)

y2_pred_gb = rnd_clf.predict(X1_test)
print (accuracy_score(y1_test,y2_pred_gb))
```

```
0.21
```

In [46]:
```python
import xgboost

xgb_reg = xgboost.XGBClassifier()
xgb_reg.fit(X,y)
y_pred_XG = xgb_reg.predict(X_test)

print (accuracy_score(y_test,y_pred))
```

```
0.85
```

In [47]:
```python
import xgboost

xgb_reg2 = xgboost.XGBClassifier()
xgb_reg2.fit(X1,y1)
y1_pred = xgb_reg.predict(X1_test)

print (accuracy_score(y1_test,y1_pred))
```

```
0.15
```

c.) Write a function that will allow you to pass in the model and the X, y set and then generate from that a plot of the decision boundary for the example

In [33]:
```python

```

Out[33]:
```
array([1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1,
       0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1,
       1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1])
```

In [81]:
```python
def decisionBoundary(model, X,y,special):
  # define bounds of the domain
  min1, max1 = X[:, 0].min()-1, X[:, 0].max()+1
  min2, max2 = X[:, 1].min()-1, X[:, 1].max()+1
  # define the x and y scale
  x1grid = np.arange(min1, max1, 0.1)
  x2grid = np.arange(min2, max2, 0.1)
  # create all of the lines and rows of the grid
  xx, yy = np.meshgrid(x1grid, x2grid)
  # flatten each grid to a vector
  r1, r2 = xx.flatten(), yy.flatten()
  r1, r2 = r1.reshape((len(r1), 1)), r2.reshape((len(r2), 1))
```

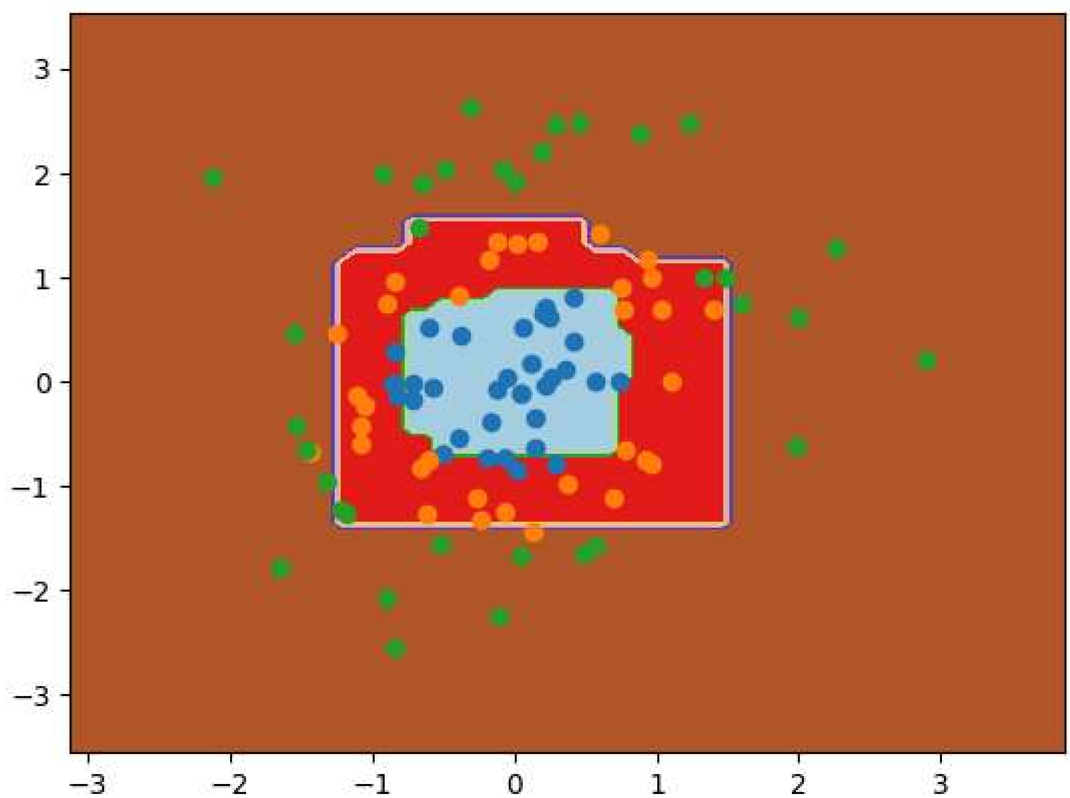
```
# horizontal stack vectors to create x1,x2 input for the model
grid = np.hstack((r1,r2))

if special == False:
  y_pred_grid = model.predict(grid)
else:
  y_pred_grid = model.predict(grid**2)
zz = y_pred_grid.reshape(xx.shape)
plt.contourf(xx, yy, zz, cmap='Paired')

for class_value in range(len(np.unique(y))):
  # get row indexes for samples with this class
  row_ix = np.where(y == class_value)
  # create scatter of these samples
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```

d.) Use your function to plot out the decision boundaries for each of your models and for the two data sets

XGboost

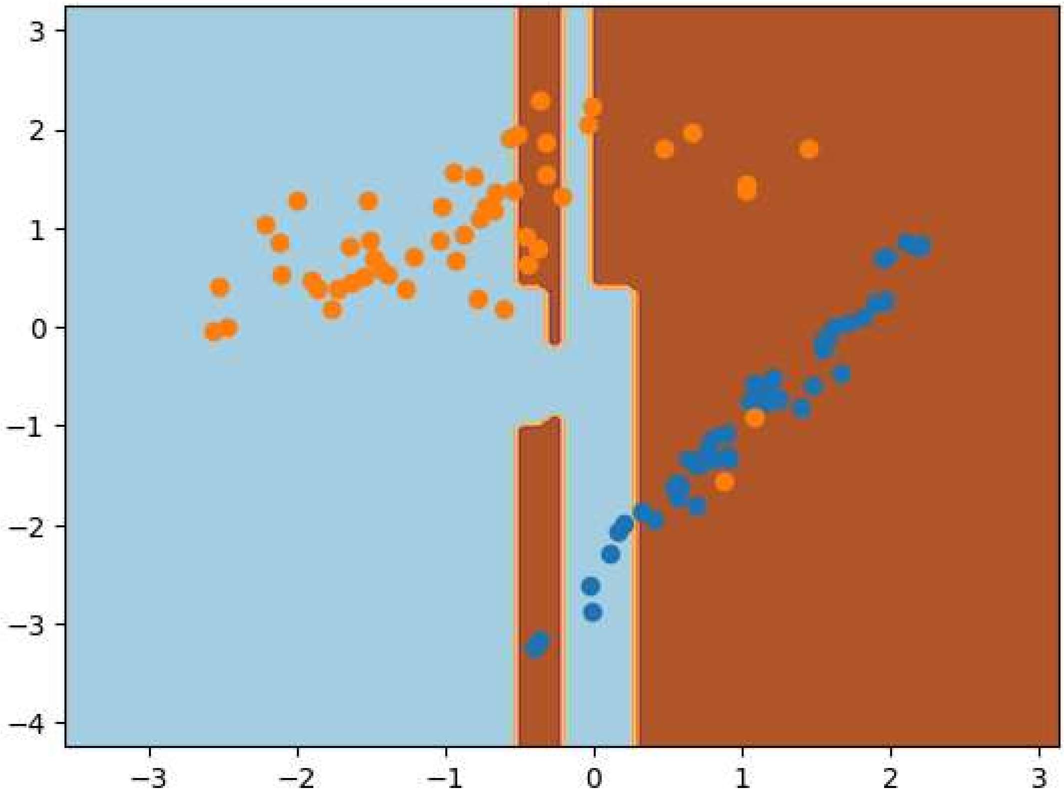
In [96]:
```
#XHboost
decisionBoundary(xgb_reg,X_test,y_test,False)
```

```
&lt;ipython-input-81-125e0d01568d&gt;:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [95]:
```
decisionBoundary(xgb_reg2,X1_test,y1_test,False)
```

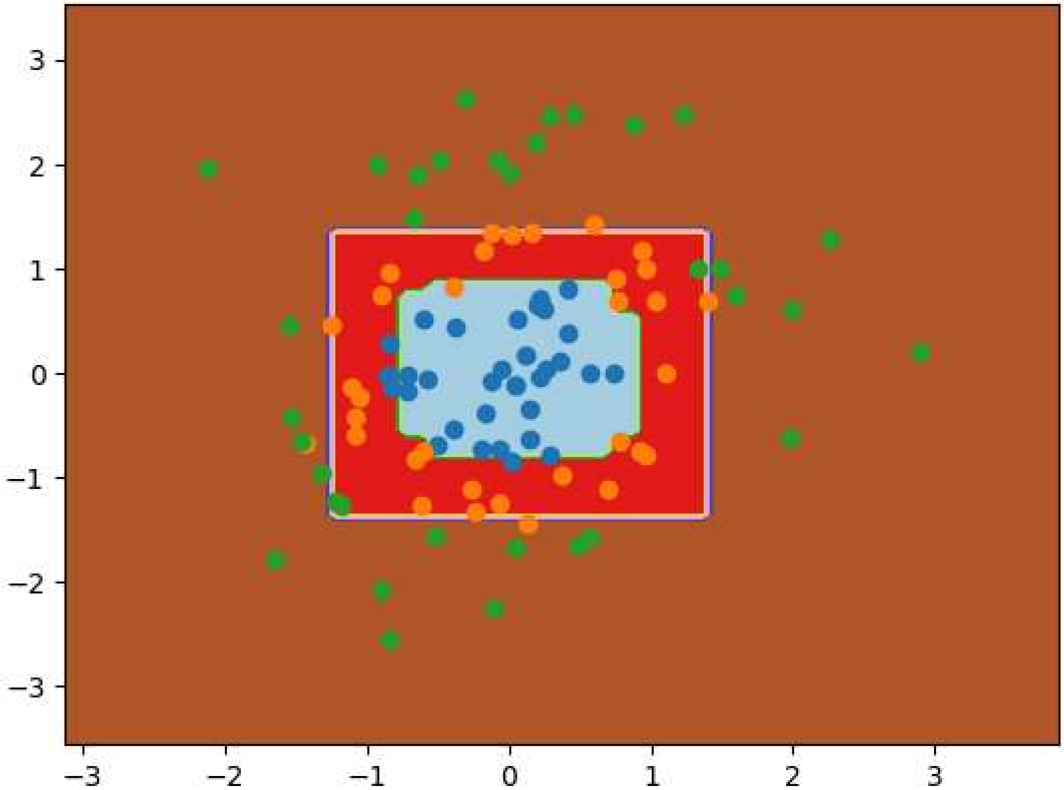
```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [94]:

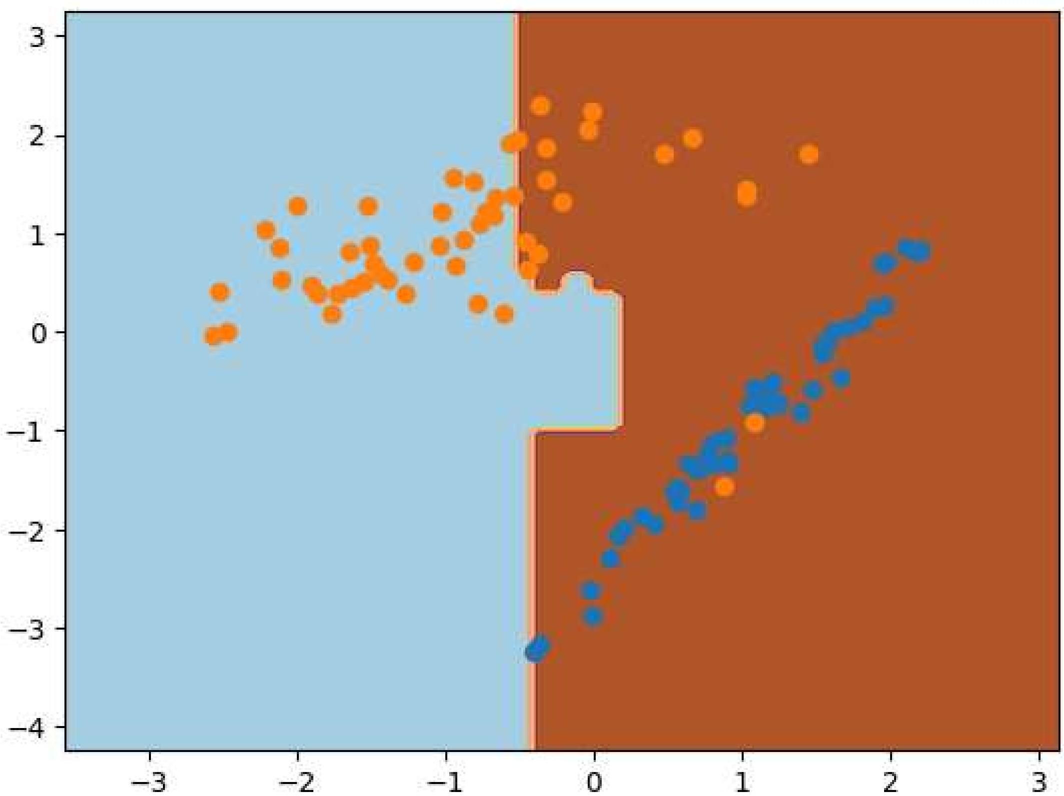
```
#Gradient Boost

decisionBoundary(gb_clf,X_test,y_test,False)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



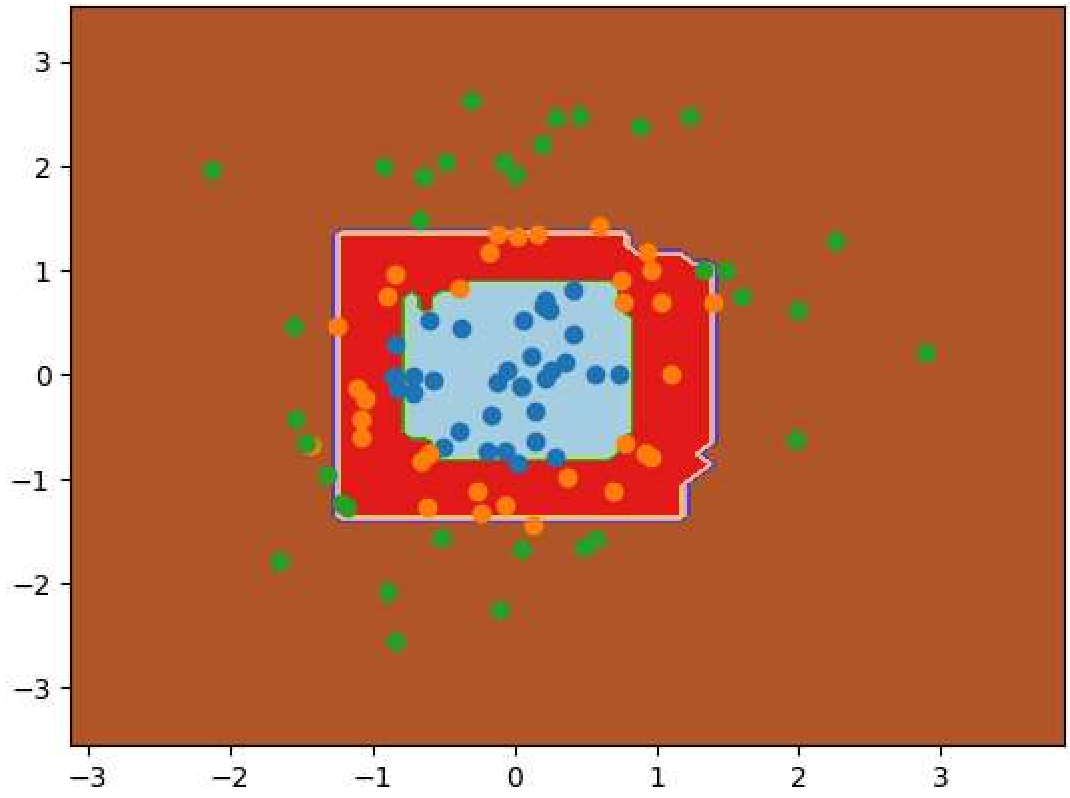
In [93]:
```
decisionBoundary(gb_clf2,X1_test,y1_test,False)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



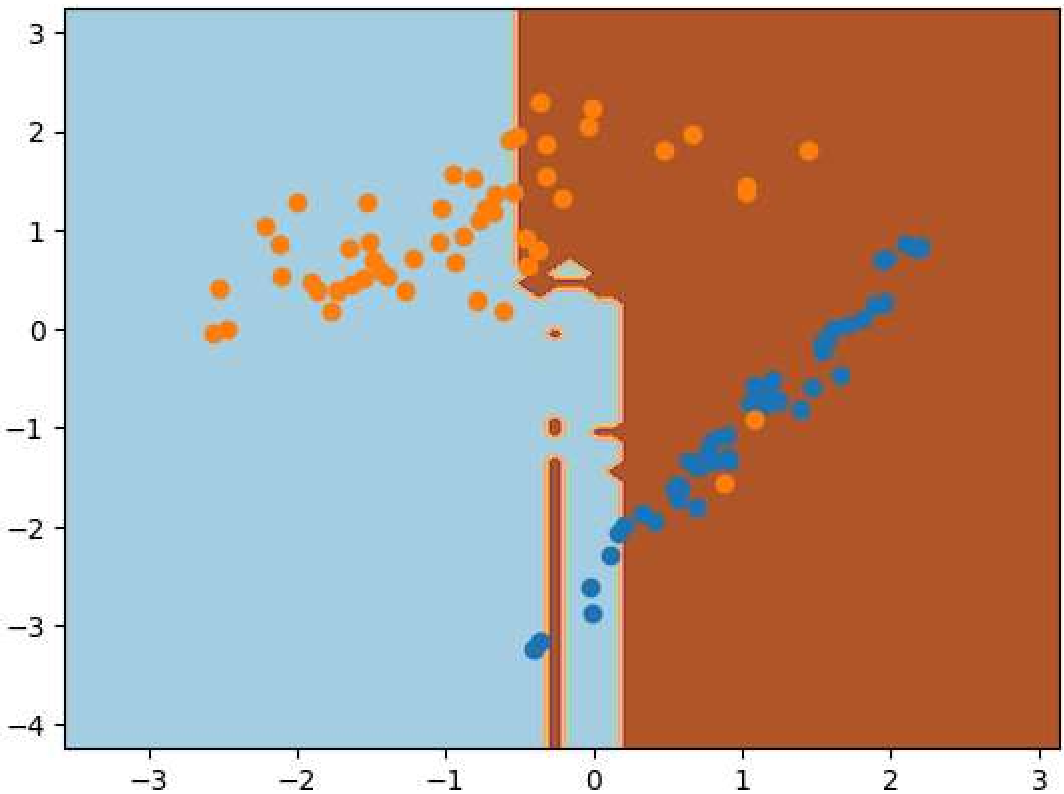
In [92]:
```
# Random Forest
decisionBoundary(rnd_clf,X_test,y_test,False)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



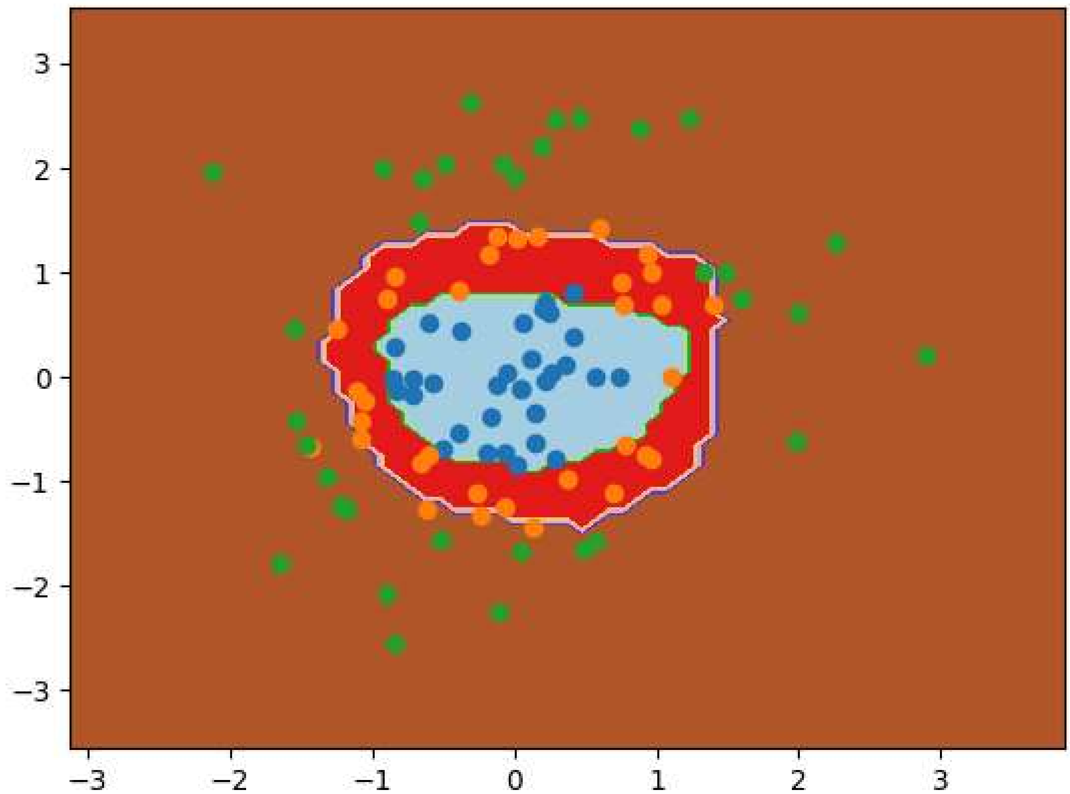
In [91]:
```
decisionBoundary(rnd_clf2,X1_test,y1_test,False)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



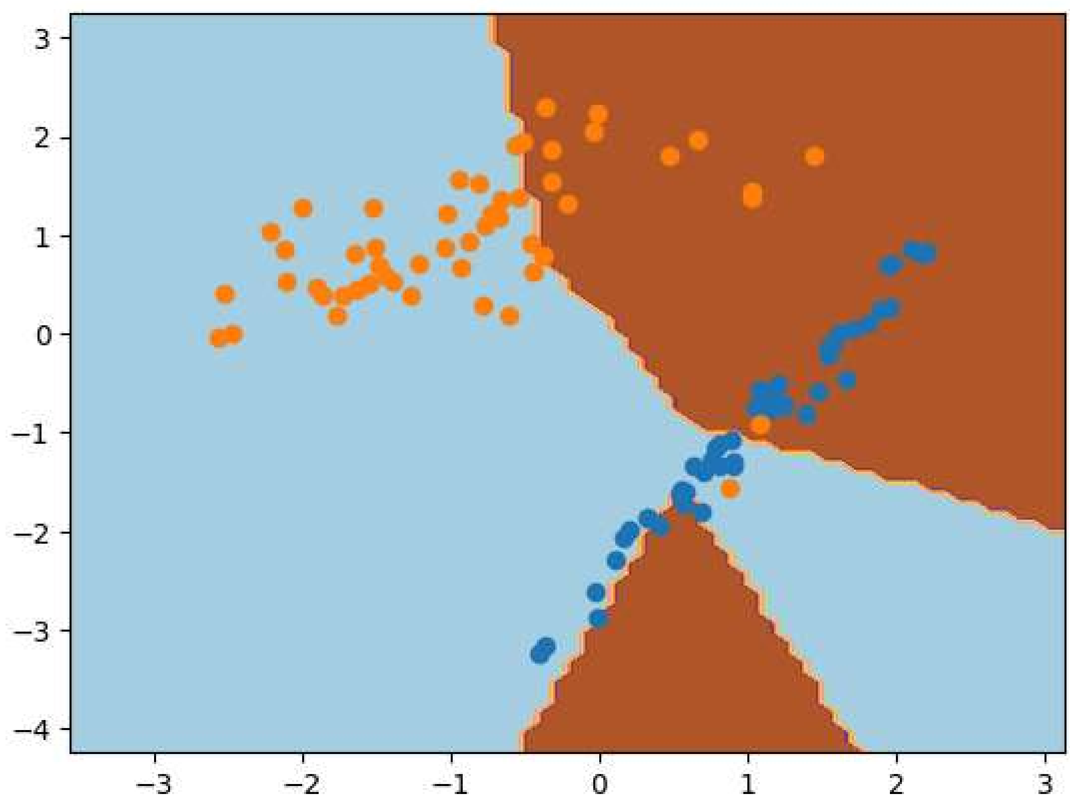
In [90]:
```
# Nural net


decisionBoundary(clf,X_test,y_test,False)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [89]:
```
decisionBoundary(clf1,X1_test,y1_test,False)
```

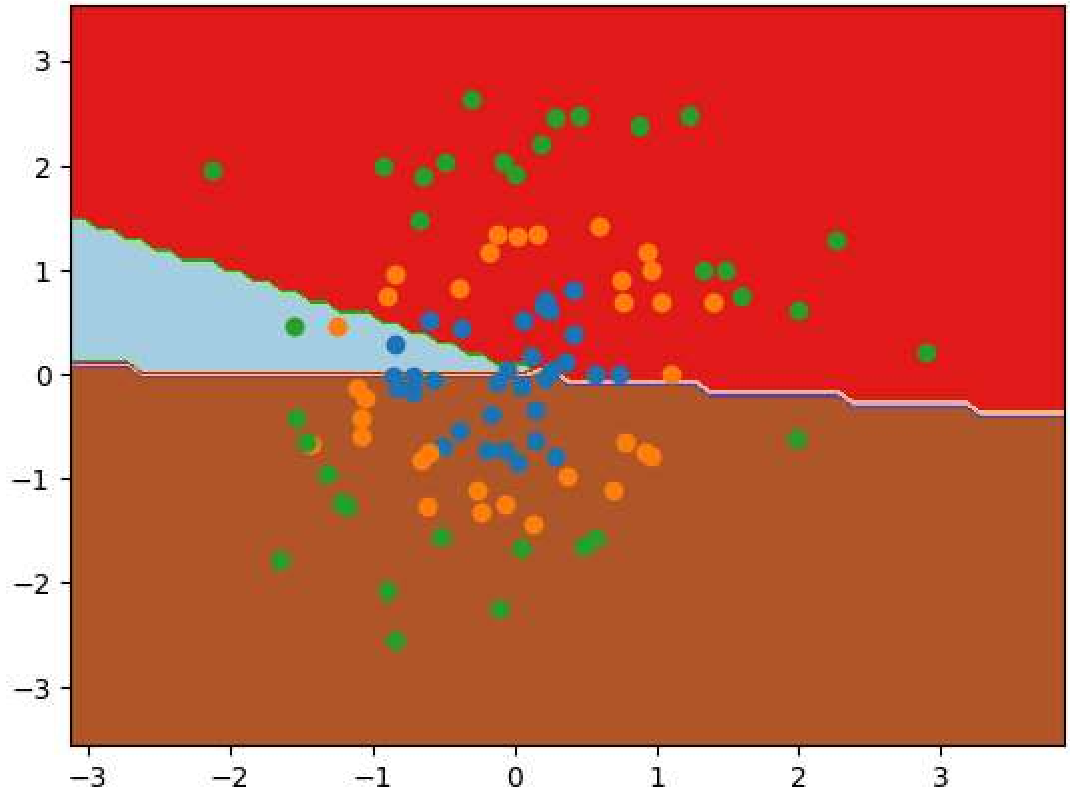
```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [88]:
```
#Logistic Regular
```

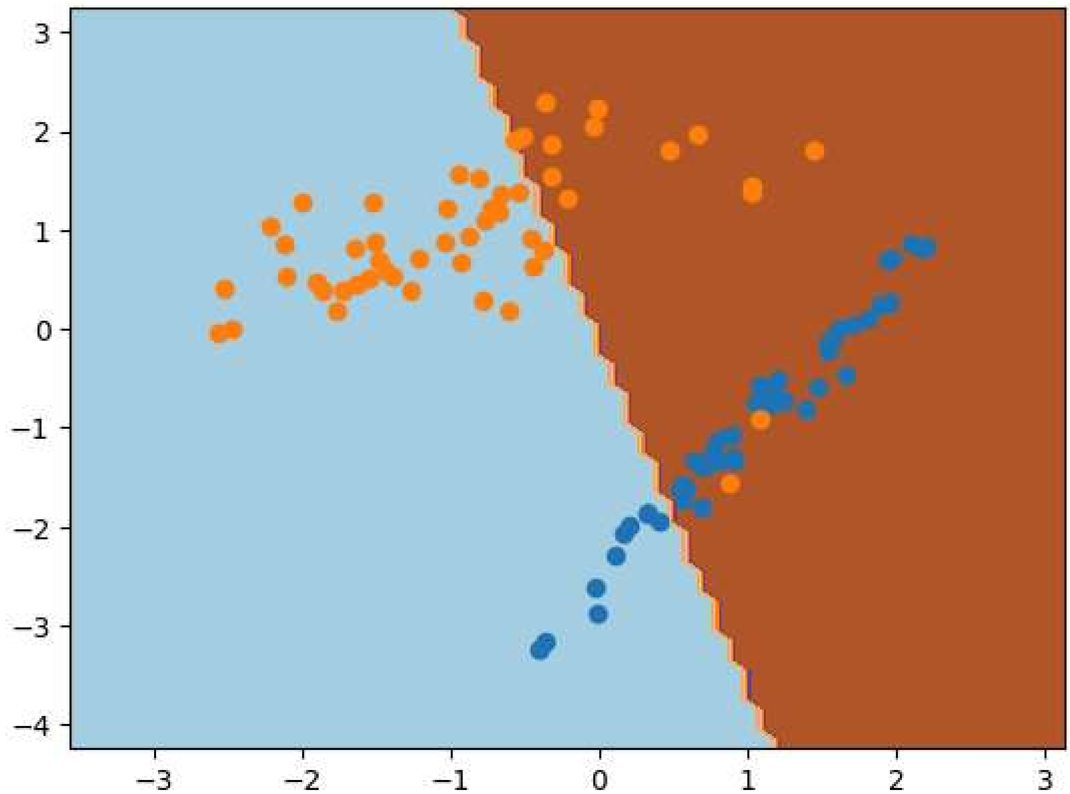
```
decisionBoundary(modelLogistic1,X_test,y_test,False)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [87]:
```
decisionBoundary(modelLogistic2,X1_test,y1_test,False)
```

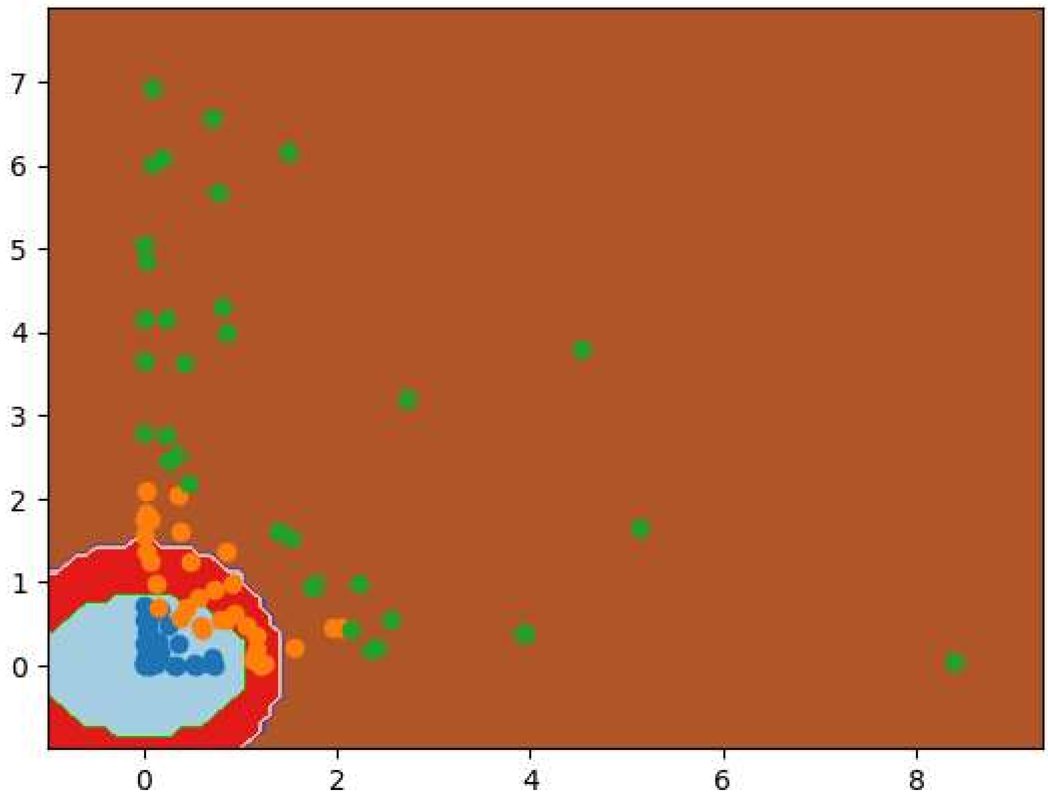
```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [97]:

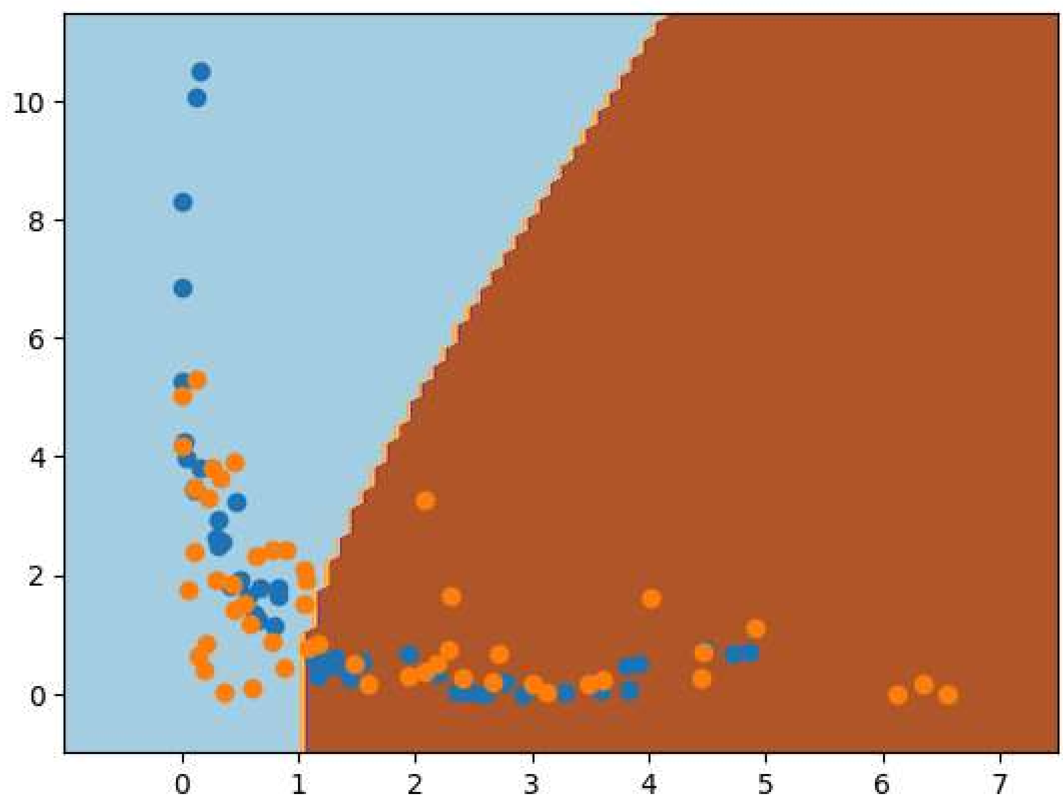
```
#LogisticX^2 and X1^2



decisionBoundary(modelLogisticV1,X_test**2,y_test,True)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



In [86]:
```
decisionBoundary(modelLogisticV2,X1_test**2,y1_test,True)
```

```
<ipython-input-81-125e0d01568d>:27: UserWarning: No data for colormapping provided vi
a 'c'. Parameters 'cmap' will be ignored
  plt.scatter(X[row_ix, 0], X[row_ix, 1], cmap='Paired')
```



e.) Based on your examination of the two data sets and the resulting plots, what can you say about the nature of the models? -which of them can generate curved boundaries? Which are all straight line boundaries?

Curved : LogisticX^2 and X1^2,Nural net Stright: XGboost, Gradient boost, random forest , regualr logistic -which looks most severely overfit? modelLogisticV1 lossts most severly overfit.

```
In [ ]:
```