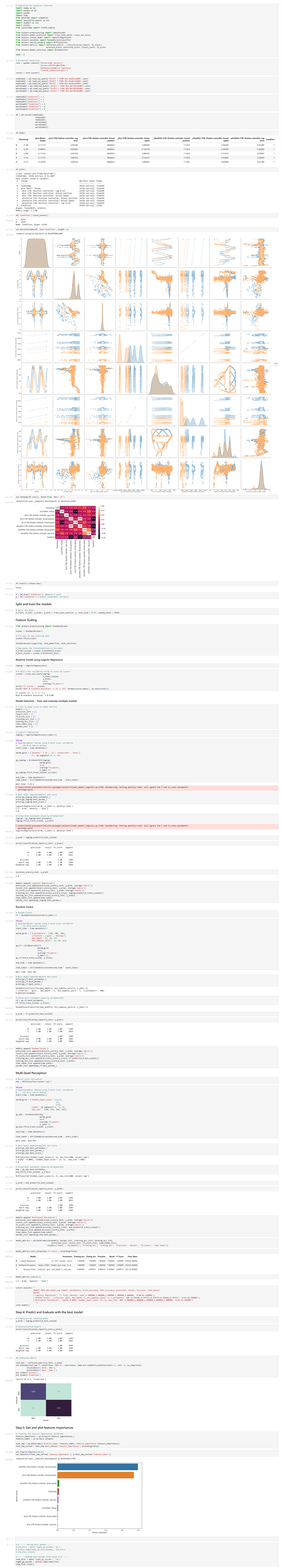
```
In [1]:
# Importing the required libraries
import numpy as np
import pandas as pd
import os
import time
from datetime import timedelta
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import confusion_matrix , classification_report, f1_score,\
from sklearn.model_selection import RandomizedSearchCV, train_test_split, f1_score

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import confusion_matrix , classification_report, f1_score,\
from sklearn.model_selection import RandomizedSearchCV, train_test_split, f1_score

%%%% #

In [2]:
# connection connection
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=LAPTOP-SBNFTQ6S;'
                      'Database=DB_CTRL_TUNNEL;'
                      'Trusted_Connection=yes;')

cursor = conn.cursor()

In [3]:
read1=pd.read_sql_query('SELECT * FROM dbo.ReadCase001', conn)
read2=pd.read_sql_query('SELECT * FROM dbo.ReadCase002', conn)
read3=pd.read_sql_query('SELECT * FROM dbo.ReadCase003', conn)
read4=pd.read_sql_query('SELECT * FROM dbo.ReadCase004', conn)
read5=pd.read_sql_query('SELECT * FROM dbo.ReadCase005', conn)
read6=pd.read_sql_query('SELECT * FROM dbo.ReadCase006', conn)

In [4]:
read1['Condition'] = 1
read2['Condition'] = 2
read3['Condition'] = 3
read4['Condition'] = 4
read5['Condition'] = 5
read6['Condition'] = 6

In [5]:
df = pd.concat([read1,read2,
                read3,read4,
                read5,read6,
                ], ignore_index=True)

In [6]:
df.head()

Out[6]:
```

	Timestamp	pCut Motor Torque	pCut CTRL Position controller Lag error	pCut CTRL Position controller Actual position	pCut CTRL Position controller Actual speed	pCut CTRL Position controller Actual torque	pSvolFilm CTRL Position controller Actual position	pSvolFilm CTRL Position controller Actual speed	pSvolFilm CTRL Position controller Lag error	Condition
0	0.48	0.0	0.001562	49449.0	0.000000	0.000000	111263	0.000000	0.000000	1
1	0.58	-0.008057	0.001562	49449.0	-0.146728	-0.732910	111263	-2.036249	0.026001	1
2	0.68	0.115161	0.001562	49449.0	0.146455	0.732910	111263	2.036249	0.026001	1
3	0.78	-0.173071	0.001562	49449.0	-0.732910	-0.732910	111263	-2.036249	0.026001	1
4	0.88	0.230713	0.001562	49449.0	0.732910	0.732910	111263	2.036249	0.026001	1

```
In [7]:
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18720 entries, 0 to 18719
Data columns (total 10 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Timestamp                               18720 non-null  float64
 1   pCut Motor Torque                       18720 non-null  float64
 2   pCut CTRL Position controller Lag error 18720 non-null  float64
 3   pCut CTRL Position controller Actual position 18720 non-null  float64
 4   pCut CTRL Position controller Actual speed 18720 non-null  float64
 5   pCut CTRL Position controller Actual torque 18720 non-null  float64
 6   pSvolFilm CTRL Position controller Actual position 18720 non-null  float64
 7   pSvolFilm CTRL Position controller Actual speed 18720 non-null  float64
 8   pSvolFilm CTRL Position controller Lag error 18720 non-null  float64
 9   Condition                               18720 non-null  int64
dtypes: float64(9), int64(1)
memory usage: 1.4 MB

In [8]:
df['Condition'].value_counts()

Out[8]:
1    3120
2    3120
3    3120
4    3120
5    3120
6    3120
Name: Condition, dtype: int64

In [9]:
!jupyter nbextension enable --py --sys-prefix ipyscales

Enabling notebook extension ipyscales/extension...
      - Validating: OK

In [10]:
import ipyvolume.pylab as p3
import ipyvolume as ipv
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(color_codes=True)
%matplotlib inline
```



```
In [11]:
sns.heatmap(df.corr(), annot=True, fmt='.2')

Out[11]:
<matplotlib.axes._subplots.AxesSubplot at 0x1f4c2c3c508>
```



```
In [12]:
df.shape

Out[12]:
(18720, 10)

In [13]:
df.dtypes

Out[13]:
Timestamp                                 float64
pCut Motor Torque                         float64
pCut CTRL Position controller Lag error   float64
pCut CTRL Position controller Actual position  float64
pCut CTRL Position controller Actual speed  float64
pCut CTRL Position controller Actual torque  float64
pSvolFilm CTRL Position controller Actual position  float64
pSvolFilm CTRL Position controller Actual speed  float64
pSvolFilm CTRL Position controller Lag error  float64
Condition                                   int64
dtype: object

In [14]:
df.isnull().sum()

Out[14]:
Timestamp                                 0
pCut Motor Torque                         0
pCut CTRL Position controller Lag error   0
pCut CTRL Position controller Actual position  0
pCut CTRL Position controller Actual speed  0
pCut CTRL Position controller Actual torque  0
pSvolFilm CTRL Position controller Actual position  0
pSvolFilm CTRL Position controller Actual speed  0
pSvolFilm CTRL Position controller Lag error  0
Condition                                   0
dtype: int64

In [15]:
df.isnull().values.any()

Out[15]:
False

In [16]:
X = df.drop('Condition', axis=1) # input
y = df['Condition'] # output (Dependent Variable)
```

Split and train the models

```
In [17]:
# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 1000)
```

Feature Scaling

```
In [18]:
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

# Fit only to the training data
scaler.fit(X_train)

StandardScaler(copy=True, with_mean=True, with_std=True)

# Now apply the transformations to the data:
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Baseline model using Logistic Regression

```
In [19]:
logreg = LogisticRegression()

# A final cross validation using 15 score as score'
scores = cross_val_score(logreg,
                         X_train_scaled,
                         y_train,
                         cv=5,
                         scoring='f1_macro')

print("F1 scores: ", scores)
print("Mean & standard deviation: {:.2} (+/- {:.2f})".format(scores.mean(), sp.std(scores))

F1 scores: [1. 1. 1. 1. 1.]
Mean & standard deviation: 1.0 (+/- 0.00
```

Model Selection - Train and evaluate multiple models

```
In [20]:
# Since my base level of model metrics
models = []
accuracy_list = []
recall_list = []
f1_score_list = []
training_acc_list = []
testing_acc_list = []
time_taken_list = []
params_list = []

# Logistic Regression
logreg = LogisticRegression(n_jobs=-1)

%%time
# Hyperparameter tuning using 4 fold cross validation
start_time = time.monotonic()
start_time = time.monotonic()

param_grid = {'penalty' : ['l2', 'l1', 'elasticnet', 'none'],
              'C' : np.logspace(-3,3,7),
              }

gs_logreg = GridSearchCV(logreg,
                         param_grid,
                         cv=4,
                         n_jobs=-1,
                         scoring='f1_macro',
                         verbose=1)

gs_logreg.fit(X_train_scaled, y_train)

end_time = time.monotonic()
print(timedelta(seconds=end_time - start_time))

Wall time: 3.19 s
C:\Users\Sathy\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:1330: UserWarning: Setting penalty='none' will ignore the C and l1_ratio parameters
  warnings.warn(
C:\Users\Sathy\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:1330: UserWarning: Setting penalty='none' will ignore the C and l1_ratio parameters
  warnings.warn(

In [21]:
# Best model hyperparameters and score
print(gs_logreg.best_estimator_)
print(gs_logreg.best_params_)
print(gs_logreg.best_score_)

LogisticRegression(C=0.001, n_jobs=-1, penalty='none')
{'C': 0.001, 'penalty': 'none'}
1.0

In [22]:
# Using best estimator found by GridSearchCV
logreg = gs_logreg.best_estimator_
logreg.fit(X_train_scaled, y_train)

LogisticRegression(C=0.001, n_jobs=-1, penalty='none')

In [23]:
y_pred = logreg.predict(X_test_scaled)

In [24]:
print(classification_report(y_test, y_pred))

              precision    recall  f1-score   support

           1       1.00      1.00      1.00       780
           2       1.00      1.00      1.00       780
           3       1.00      1.00      1.00       780
           4       1.00      1.00      1.00       780
           5       1.00      1.00      1.00       780
           6       1.00      1.00      1.00       780

    accuracy                           1.00      4680
   macro avg       1.00      1.00      1.00      4680
weighted avg       1.00      1.00      1.00      4680

In [25]:
accuracy_score(y_test, y_pred)

Out[25]:
1.0

In [26]:
models.append('Logistic Regression')
precision_list.append(precision_score(y_test, y_pred, average='macro'))
recall_list.append(recall_score(y_test, y_pred, average='macro'))
f1_score_list.append(f1_score(y_test, y_pred, average='macro'))
training_acc_list.append(accuracy_score(y_train, logreg.predict(X_train_scaled)))
testing_acc_list.append(accuracy_score(y_test, y_pred))
time_taken_list.append(time_taken)
params_list.append(gs_logreg.best_params_)
```

Random Forest

```
In [27]:
# Random Forest
rf = RandomForestClassifier(n_jobs=-1)

%%time
# Hyperparameter tuning using 4 fold cross validation
start_time = time.monotonic()
start_time = time.monotonic()

param_grid = {'n_estimators': [100, 200, 300],
              'criterion' : ['gini', 'entropy'],
              'max_depth' : [4, 5, 6, 7, 8],
              'min_samples_split': [5, 10, 15]
              }

gs_rf = GridSearchCV(rf,
                     param_grid,
                     cv=4,
                     n_jobs=-1,
                     scoring='f1_macro',
                     verbose=1)

gs_rf.fit(X_train_scaled, y_train)

end_time = time.monotonic()

time_taken = str(timedelta(seconds=end_time - start_time))

Wall time: 6min 49s

In [28]:
# Best model hyperparameters and score
print(gs_rf.best_estimator_)
print(gs_rf.best_params_)
print(gs_rf.best_score_)

RandomForestClassifier(max_depth=8, min_samples_split=5, n_jobs=-1)
{'criterion': 'gini', 'max_depth': 8, 'min_samples_split': 5, 'n_estimators': 100}
0.9999999999999999

In [29]:
# Using best estimator found by GridSearchCV
rf = gs_rf.best_estimator_
rf.fit(X_train_scaled, y_train)

RandomForestClassifier(max_depth=8, min_samples_split=5, n_jobs=-1)

In [30]:
y_pred = rf.predict(X_test_scaled)

In [31]:
print(classification_report(y_test, y_pred))

              precision    recall  f1-score   support

           1       1.00      1.00      1.00       780
           2       1.00      1.00      1.00       780
           3       1.00      1.00      1.00       780
           4       1.00      1.00      1.00       780
           5       1.00      1.00      1.00       780
           6       1.00      1.00      1.00       780

    accuracy                           1.00      4680
   macro avg       1.00      1.00      1.00      4680
weighted avg       1.00      1.00      1.00      4680

In [32]:
models.append('Random Forest')
precision_list.append(precision_score(y_test, y_pred, average='macro'))
recall_list.append(recall_score(y_test, y_pred, average='macro'))
f1_score_list.append(f1_score(y_test, y_pred, average='macro'))
training_acc_list.append(accuracy_score(y_train, rf.predict(X_train_scaled)))
testing_acc_list.append(accuracy_score(y_test, y_pred))
time_taken_list.append(time_taken)
params_list.append(gs_rf.best_params_)
```

Multi-level Perceptron

```
In [33]:
# Multi-level Perceptron
mlp = MLPClassifier(verbose='mlp')

%%time
# Hyperparameter tuning using 4 fold cross validation
start_time = time.monotonic()
start_time = time.monotonic()

param_grid = {'hidden_layer_sizes': [(3,3),
                                     (4),
                                     (5,5),
                                     ],
              'alpha' : np.logspace(-4, 2, 6),
              'max_iter': [500, 750, 1000, 1500],
              }

gs_mlp = GridSearchCV(mlp,
                      param_grid,
                      cv=4,
                      n_jobs=-1,
                      scoring='f1_macro',
                      verbose=1)

gs_mlp.fit(X_train_scaled, y_train)

end_time = time.monotonic()

time_taken = str(timedelta(seconds=end_time - start_time))

Wall time: 6min 49s

In [34]:
# Best model hyperparameters and score
print(gs_mlp.best_estimator_)
print(gs_mlp.best_params_)
print(gs_mlp.best_score_)

MLPClassifier(alpha=0.0001, hidden_layer_sizes=(5, 5), max_iter=1000, verbose='mlp')
{'alpha': 0.0001, 'hidden_layer_sizes': (5, 5), 'max_iter': 1000}
1.0

In [35]:
# Using best estimator found by GridSearchCV
mlp = gs_mlp.best_estimator_
mlp.fit(X_train_scaled, y_train)

MLPClassifier(hidden_layer_sizes=(5, 5), max_iter=1000, verbose='mlp')

Iteration 1, loss = 1.79178825
Iteration 2, loss = 1.78911284
Iteration 3, loss = 1.78650139
Iteration 4, loss = 1.78394966
Iteration 5, loss = 1.78145350
Iteration 6, loss = 1.77900911
Iteration 7, loss = 1.77661290
Iteration 8, loss = 1.77426151
Iteration 9, loss = 1.77195176
Iteration 10, loss = 1.76968060
Iteration 11, loss = 1.76744514
Iteration 12, loss = 1.76524262
Iteration 13, loss = 1.76307040
Iteration 14, loss = 1.76092600
Iteration 15, loss = 1.75880705
Iteration 16, loss = 1.75671130
Iteration 17, loss = 1.75463659
Iteration 18, loss = 1.75258086
Iteration 19, loss = 1.75054211
Iteration 20, loss = 1.74851844
Iteration 21, loss = 1.74650802
Iteration 22, loss = 1.74450911
Iteration 23, loss = 1.74252004
Iteration 24, loss = 1.74053918
Iteration 25, loss = 1.73856496
Iteration 26, loss = 1.73659587
Iteration 27, loss = 1.73463044
Iteration 28, loss = 1.73266726
Iteration 29, loss = 1.73070497
Iteration 30, loss = 1.72874224
Iteration 31, loss = 1.72677779
Iteration 32, loss = 1.72481037
Iteration 33, loss = 1.72283880
Iteration 34, loss = 1.72086194
Iteration 35, loss = 1.71887870
Iteration 36, loss = 1.71688806
Iteration 37, loss = 1.71488904
Iteration 38, loss = 1.71288073
Iteration 39, loss = 1.71086227
Iteration 40, loss = 1.70883286
Iteration 41, loss = 1.70679176
Iteration 42, loss = 1.70473829
Iteration 43, loss = 1.70267183
Iteration 44, loss = 1.70059183
Iteration 45, loss = 1.69849781
Iteration 46, loss = 1.69638935
Iteration 47, loss = 1.69426611
Iteration 48, loss = 1.69212781
Iteration 49, loss = 1.68997424
Iteration 50, loss = 1.68780525
Iteration 51, loss = 1.68562073
Iteration 52, loss = 1.68342064
Iteration 53, loss = 1.68120500
Iteration 54, loss = 1.67897389
Iteration 55, loss = 1.67672746
Iteration 56, loss = 1.67446592
Iteration 57, loss = 1.67218955
Iteration 58, loss = 1.66989870
Iteration 59, loss = 1.66759378
Iteration 60, loss = 1.66527525
Iteration 61, loss = 1.66294363
Iteration 62, loss = 1.66059950
Iteration 63, loss = 1.65824350
Iteration 64, loss = 1.65587630
Iteration 65, loss = 1.65349859
Iteration 66, loss = 1.65111112
Iteration 67, loss = 1.64871468
Iteration 68, loss = 1.64631008
Iteration 69, loss = 1.64389819
Iteration 70, loss = 1.64147990
Iteration 71, loss = 1.63905611
Iteration 72, loss = 1.63662776
Iteration 73, loss = 1.63419581
Iteration 74, loss = 1.63176125
Iteration 75, loss = 1.62932509
Iteration 76, loss = 1.62688835
Iteration 77, loss = 1.62445206
Iteration 78, loss = 1.62201726
Iteration 79, loss = 1.61958499
Iteration 80, loss = 1.61715630
Iteration 81, loss = 1.61473224
Iteration 82, loss = 1.61231385
Iteration 83, loss = 1.60990216
Iteration 84, loss = 1.60749818
Iteration 85, loss = 1.60510289
Iteration 86, loss = 1.60271724
Iteration 87, loss = 1.60034212
Iteration 88, loss = 1.59797837
Iteration 89, loss = 1.59562677
Iteration 90, loss = 1.59328802
Iteration 91, loss = 1.59096273
Iteration 92, loss = 1.58865141
Iteration 93, loss = 1.58635446
Iteration 94, loss = 1.58407216
Iteration 95, loss = 1.58180464
Iteration 96, loss = 1.57955187
Iteration 97, loss = 1.57731369
Iteration 98, loss = 1.57508977
Iteration 99, loss = 1.57287966
Iteration 100, loss = 1.57068276
Iteration 101, loss = 1.56849835
Iteration 102, loss = 1.56632559
Iteration 103, loss = 1.56416353
Iteration 104, loss = 1.56201111
Iteration 105, loss = 1.55986715
Iteration 106, loss = 1.55773040
Iteration 107, loss = 1.55559955
Iteration 108, loss = 1.55347322
Iteration 109, loss = 1.55134999
Iteration 110, loss = 1.54922844
Iteration 111, loss = 1.54710713
Iteration 112, loss = 1.54498464
Iteration 113, loss = 1.54285957
Iteration 114, loss = 1.54073057
Iteration 115, loss = 1.53859636
Iteration 116, loss = 1.53645574
Iteration 117, loss = 1.53430763
Iteration 118, loss = 1.53215109
Iteration 119, loss = 1.52998532
Iteration 120, loss = 1.52780971
Iteration 121, loss = 1.52562387
Iteration 122, loss = 1.52342766
Iteration 123, loss = 1.52122120
Iteration 124, loss = 1.51900492
Iteration 125, loss = 1.51677960
Iteration 126, loss = 1.51454636
Iteration 127, loss = 1.51230665
Iteration 128, loss = 1.51006222
Iteration 129, loss = 1.50781503
Iteration 130, loss = 1.50556722
Iteration 131, loss = 1.50332102
Iteration 132, loss = 1.50107866
Iteration 133, loss = 1.49884233
Iteration 134, loss = 1.49661410
Iteration 135, loss = 1.49439588
Iteration 136, loss = 1.49218940
Iteration 137, loss = 1.48999618
Iteration 138, loss = 1.48781747
Iteration 139, loss = 1.48565424
Iteration 140, loss = 1.48350719
Iteration 141, loss = 1.48137669
Iteration 142, loss = 1.47926276
Iteration 143, loss = 1.47716503
Iteration 144, loss = 1.47508277
Iteration 145, loss = 1.47301488
Iteration 146, loss = 1.47095997
Iteration 147, loss = 1.46891643
Iteration 148, loss = 1.46688253
Iteration 149, loss = 1.46485650
Iteration 150, loss = 1.46283660
Iteration 151, loss = 1.46082115
Iteration 152, loss = 1.45880859
Iteration 153, loss = 1.45679750
Iteration 154, loss = 1.45478664
Iteration 155, loss = 1.45277501
Iteration 156, loss = 1.45076181
Iteration 157, loss = 1.44874650
Iteration 158, loss = 1.44672875
Iteration 159, loss = 1.44470844
Iteration 160, loss = 1.44268567
Iteration 161, loss = 1.44066071
Iteration 162, loss = 1.43863405
Iteration 163, loss = 1.43660630
Iteration 164, loss = 1.43457814
Iteration 165, loss = 1.43255031
Iteration 166, loss = 1.43052358
Iteration 167, loss = 1.42849870
Iteration 168, loss = 1.42647641
Iteration 169, loss = 1.42445741
Iteration 170, loss = 1.42244233
Iteration 171, loss = 1.42043172
Iteration 172, loss = 1.41842604
Iteration 173, loss = 1.41642567
Iteration 174, loss = 1.41443090
Iteration 175, loss = 1.41244193
Iteration 176, loss = 1.41045886
Iteration 177, loss = 1.40848172
Iteration 178, loss = 1.40651045
Iteration 179, loss = 1.40454490
Iteration 180, loss = 1.40258484
Iteration 181, loss = 1.40062997
Iteration 182, loss = 1.39867993
Iteration 183, loss = 1.39673432
Iteration 184, loss = 1.39479273
Iteration 185, loss = 1.39285475
Iteration 186, loss = 1.39091999
Iteration 187, loss = 1.38898809
Iteration 188, loss = 1.38705875
Iteration 189, loss = 1.38513172
Iteration 190, loss = 1.38320681
Iteration 191, loss = 1.38128387
Iteration 192, loss = 1.37936284
Iteration 193, loss = 1.37744374
Iteration 194, loss = 1.37552665
Iteration 195, loss = 1.37361170
Iteration 196, loss = 1.37169904
Iteration 197, loss = 1.36978885
Iteration 198, loss = 1.36788130
Iteration 199, loss = 1.36597654
Iteration 200, loss = 1.36407468
Iteration 201, loss = 1.36217577
Iteration 202, loss = 1.36027980
Iteration 203, loss = 1.35838670
Iteration 204, loss = 1.35649636
Iteration 205, loss = 1.35460864
Iteration 206, loss = 1.35272337
Iteration 207, loss = 1.35084045
Iteration 208, loss = 1.34895980
Iteration 209, loss = 1.34708139
Iteration 210, loss = 1.34520525
Iteration 211, loss = 1.34333141
Iteration 212, loss = 1.34145990
Iteration 213, loss = 1.33959073
Iteration 214, loss = 1.33772389
Iteration 215, loss = 1.33585936
Iteration 216, loss = 1.33399710
Iteration 217, loss = 1.33213707
Iteration 218, loss = 1.33027923
Iteration 219, loss = 1.32842354
Iteration 220, loss = 1.32656997
Iteration 221, loss = 1.32471850
Iteration 222, loss = 1.32286913
Iteration 223, loss = 1.32102187
Iteration 224, loss = 1.31917674
Iteration 225, loss = 1.31733376
Iteration 226, loss = 1.31549295
Iteration 227, loss = 1.31365433
Iteration 228, loss = 1.31181791
Iteration 229, loss = 1.30998370
Iteration 230, loss = 1.30815170
Iteration 231, loss = 1.30632191
Iteration 232, loss = 1.30449432
Iteration 233, loss = 1.30266892
Iteration 234, loss = 1.30084570
Iteration 235, loss = 1.29902464
Iteration 236, loss = 1.29720572
Iteration 237, loss = 1.29538892
Iteration 238, loss = 1.29357421
Iteration 239, loss = 1.29176156
Iteration 240, loss = 1.28995094
Iteration 241, loss = 1.28814232
Iteration 242, loss = 1.28633566
Iteration 243, loss = 1.28453093
Iteration 244, loss = 1.28272809
Iteration 245, loss = 1.28092710
Iteration 246, loss = 1.27912792
Iteration 247, loss = 1.27733052
Iteration 248, loss = 1.27553485
Iteration 249, loss = 1.27374088
Iteration 250, loss = 1.27194857
Iteration 251, loss = 1.27015788
Iteration 252, loss = 1.26836878
Iteration 253, loss = 1.26658123
Iteration 254, loss = 1.26479520
Iteration 255, loss = 1.26301066
Iteration 256, loss = 1.26122757
Iteration 257, loss = 1.25944590
Iteration 258, loss = 1.25766562
Iteration 259, loss = 1.25588670
Iteration 260, loss = 1.25410911
Iteration 261, loss = 1.25233283
Iteration 262, loss = 1.25055783
Iteration 263, loss = 1.24878409
Iteration 264, loss = 1.24701158
Iteration 265, loss = 1.24524029
Iteration 266, loss = 1.24347020
Iteration 267, loss = 1.24170130
Iteration 268, loss = 1.23993357
Iteration 269, loss = 1.23816700
Iteration 270, loss = 1.23640158
Iteration 271, loss = 1.23463731
Iteration 272, loss = 1.23287418
Iteration 273, loss = 1.23111220
Iteration 274, loss = 1.22935137
Iteration 275, loss = 1.22759170
Iteration 276, loss = 1.22583321
Iteration 277, loss = 1.22407592
Iteration 278, loss = 1.22231985
Iteration 279, loss = 1.22056504
Iteration 280, loss = 1.21881153
Iteration 281, loss = 1.21705936
Iteration 282, loss = 1.21530858
Iteration 283, loss = 1.21355923
Iteration 284, loss = 1.21181137
Iteration 285, loss = 1.21006505
Iteration 286, loss = 1.20832032
Iteration 287, loss = 1.20657723
Iteration 288, loss = 1.20483583
Iteration 289, loss = 1.20309617
Iteration 290, loss = 1.20135830
Iteration 291, loss = 1.19962227
Iteration 292, loss = 1.19788813
Iteration 293, loss = 1.19615592
Iteration 294, loss = 1.19442569
Iteration 295, loss = 1.19269748
Iteration 296, loss = 1.19097134
Iteration 297, loss = 1.18924731
Iteration 298, loss = 1.18752543
Iteration 299, loss = 1.18580574
Iteration 300, loss = 1.18408827
Iteration 301, loss = 1.18237305
Iteration 302, loss = 1.18066011
Iteration 303, loss = 1.17894947
Iteration 304, loss = 1.17724114
Iteration 305, loss = 1.17553514
Iteration 306, loss = 1.17383147
Iteration 307, loss = 1.17213013
Iteration 308, loss = 1.17043111
Iteration 309, loss = 1.16873439
Iteration 310, loss = 1.16703994
Iteration 311, loss = 1.16534772
Iteration 312, loss = 1.16365768
Iteration 313, loss = 1.16196977
Iteration 314, loss = 1.16028392
Iteration 315, loss = 1.15860006
Iteration 316, loss = 1.15691811
Iteration 317, loss = 1.15523798
Iteration 318, loss = 1.15355957
Iteration 319, loss = 1.15188278
Iteration 320, loss = 1.15020749
Iteration 321, loss = 1.14853358
Iteration 322, loss = 1.14686092
Iteration 323, loss = 1.14518937
Iteration 324, loss = 1.14351879
Iteration 325, loss = 1.14184902
Iteration 326, loss = 1.14017990
Iteration 327, loss = 1.13851125
Iteration 328, loss = 1.13684289
Iteration 329, loss = 1.13517462
Iteration 330, loss = 1.13350624
Iteration 331, loss = 1.13183753
Iteration 332, loss = 1.13016827
Iteration 333, loss = 1.12849822
Iteration 334, loss = 1.12682713
Iteration 335, loss = 1.12515473
Iteration 336, loss = 1.12348074
Iteration 337, loss = 1.12180484
Iteration 338, loss = 1.12012673
Iteration 339, loss = 1.11844607
Iteration 340, loss = 1.11676251
Iteration 341, loss = 1.11507567
Iteration 342, loss = 1.11338516
Iteration 343, loss = 1.11169057
Iteration 344, loss = 1.10999147
Iteration 345, loss = 1.10828741
Iteration 346, loss = 1.10657791
Iteration 347, loss = 1.10486249
Iteration 348, loss = 1.10314064
Iteration 349, loss = 1.10141186
Iteration 350, loss = 1.09967564
Iteration 351, loss = 1.09793145
Iteration 352, loss = 1.09617877
Iteration 353, loss = 1.09441709
Iteration 354, loss = 1.09264592
Iteration 355, loss = 1.09086479
Iteration 356, loss = 1.08907326
Iteration 357, loss = 1.08727093
Iteration 358, loss = 1.08545745
Iteration 359, loss = 1.08363253
Iteration 360, loss = 1.08179595
Iteration 361, loss = 1.07994760
Iteration 362, loss = 1.07808750
Iteration 363, loss = 1.07621587
Iteration 364, loss = 1.07433312
Iteration 365, loss = 1.07244000
Iteration 366, loss = 1.07053760
Iteration 367, loss = 1.06862733
Iteration 368, loss = 1.06671102
Iteration 369, loss = 1.06479102
Iteration 370, loss = 1.06287000
Iteration 371, loss = 1.06095102
Iteration 372, loss = 1.05903744
Iteration 373, loss = 1.05713276
Iteration 374, loss = 1.05524060
Iteration 375, loss = 1.05336450
Iteration 376, loss = 1.05150788
Iteration 377, loss = 1.04967395
Iteration 378, loss = 1.04786566
Iteration 379, loss = 1.04608566
Iteration 380, loss = 1.04433624
Iteration 381, loss = 1.04261931
Iteration 382, loss = 1.04093636
Iteration 383, loss = 1.03928844
Iteration 384, loss = 1.03767615
Iteration 385, loss = 1.03609966
Iteration 386, loss = 1.03455866
Iteration 387, loss = 1.03305239
Iteration 388, loss = 1.03157966
Iteration 389, loss = 1.03013888
Iteration 390, loss = 1.02872810
Iteration 391, loss = 1.02734503
Iteration 392, loss = 1.02598713
Iteration 393, loss = 1.02465161
Iteration 394, loss = 1.02333554
Iteration 395, loss = 1.02203591
Iteration 396, loss = 1.02074971
Iteration 397, loss = 1.01947398
Iteration 398, loss = 1.01820586
Iteration 399, loss = 1.01694262
Iteration 400, loss = 1.01568172
Iteration 401, loss = 1.01442083
Iteration 402, loss = 1.01315787
Iteration 403, loss = 1.01189102
Iteration 404, loss = 1.01061877
Iteration 405, loss = 1.00934000
Iteration 406, loss = 1.00805397
Iteration 407, loss = 1.00676035
Iteration 408, loss = 1.00545928
Iteration 409, loss = 1.00415135
Iteration 410, loss = 1.00283762
Iteration 411, loss = 1.00151960
Iteration 412, loss = 1.00019923
Iteration 413, loss = 0.99887880
Iteration 414, loss = 0.99756090
Iteration 415, loss = 0.99624827
Iteration 416, loss = 0.99494384
Iteration 417, loss = 0.99365060
Iteration 418, loss = 0.99237154
Iteration 419, loss = 0.99110952
Iteration 420, loss = 0.98986722
Iteration 421, loss = 0.98864709
Iteration 422, loss = 0.98745126
Iteration 423, loss = 0.98628151
Iteration 424, loss = 0.98513918
Iteration 425, loss = 0.98402511
Iteration 426, loss = 0.98293966
Iteration 427, loss = 0.98188268
Iteration 428, loss = 0.98085350
Iteration 429, loss = 0.97985090
Iteration 430, loss = 0.97887311
Iteration 431, loss = 0.97791783
Iteration 432, loss = 0.97698227
Iteration 433, loss = 0.97606317
Iteration 434, loss = 0.97515685
Iteration 435, loss = 0.97425931
Iteration 436, loss = 0.97336630
Iteration 437, loss = 0.97247346
Iteration 438, loss = 0.97157648
Iteration 439, loss = 0.97067120
Iteration 440, loss = 0.96975367
Iteration 441, loss = 0.96882024
Iteration 442, loss = 0.96786760
Iteration 443, loss = 0.96689284
Iteration 444, loss = 0.96589344
Iteration 445, loss = 0.96486731
Iteration 446, loss = 0.96381276
Iteration 447, loss = 0.96272850
Iteration 448, loss = 0.96161360
Iteration 449, loss = 0.96046750
Iteration 450, loss = 0.95929000
Iteration 451, loss = 0.95808125
Iteration 452, loss = 0.95684172
Iteration 453, loss = 0.95557220
Iteration 454, loss = 0.95427377
Iteration 455, loss = 0.95294777
Iteration 456, loss = 0.95159579
Iteration 457, loss = 0.95021960
Iteration 458, loss = 0.94882112
Iteration 459, loss = 0.94740241
Iteration 460, loss = 0.94596560
Iteration 461, loss = 0.94451287
Iteration 462, loss = 0.94304638
Iteration 463, loss = 0.94156827
Iteration 464, loss = 0.94008060
Iteration 465, loss = 0.93858535
Iteration 466, loss = 0.93708440
Iteration 467, loss = 0.93557950
Iteration 468, loss = 0.93407222
Iteration 469, loss = 0.93256393
Iteration 470, loss = 0.93105579
Iteration 471, loss = 0.92954875
Iteration 472, loss = 0.92804350
Iteration 473, loss = 0.92654048
Iteration 474, loss = 0.92503988
Iteration 475, loss = 0.92354166
Iteration 476, loss = 0.92204557
Iteration 477, loss = 0.92055114
Iteration 478, loss = 0.91905771
Iteration 479, loss = 0.91756446
Iteration 480, loss = 0.91607045
Iteration 481, loss = 0.91457463
Iteration 482, loss = 0.91307588
Iteration 483, loss = 0.91157307
Iteration 484, loss = 0.91006504
Iteration 485, loss = 0.90855064
Iteration 486, loss = 0.90702877
Iteration 487, loss = 0.90549840
Iteration 488, loss = 0.90395864
Iteration 489, loss = 0.90240875
Iteration 490, loss = 0.90084824
Iteration 491, loss = 0.89927688
Iteration 492, loss = 0.89769477
Iteration 493, loss = 0.89610237
Iteration 494, loss = 0.89450048
Iteration 495, loss = 0.89289027
Iteration 496, loss = 0.89127331
Iteration 497, loss = 0.88965158
Iteration 498, loss = 0.88802747
Iteration 499, loss = 0.88640373
Iteration 500, loss = 0.88478349
Iteration 501, loss = 0.88316999
Iteration 502, loss = 0.88156650
Iteration 503, loss = 0.87997615
Iteration 504, loss = 0.87840178
Iteration 505, loss = 0.87684577
Iteration 506, loss = 0.87531000
Iteration 507, loss = 0.87379581
Iteration 508, loss = 0.87230393
Iteration 509, loss = 0.87083450
Iteration 510, loss = 0.86938705
Iteration 511, loss = 0.86796048
Iteration 512, loss = 0.86655313
Iteration 513, loss = 0.86516284
Iteration 514, loss = 0.86378695
Iteration 515, loss = 0.86242241
Iteration 516, loss = 0.86106585
Iteration 517, loss = 0.85971366
Iteration 518, loss = 0.85836202
Iteration 519, loss = 0.85700702
Iteration 520, loss = 0.85564477
Iteration 521, loss = 0.85427150
Iteration 522, loss = 0.85288364
Iteration 523, loss = 0.85147787
Iteration 524, loss = 0.85005122
Iteration 525, loss = 0.84860110
Iteration 526, loss = 0.84712540
Iteration 527, loss = 0.84562252
Iteration 528, loss = 0.84409139
Iteration 529, loss = 0.84253151
Iteration 530, loss = 0.84094296
Iteration 531, loss = 0.83932645
Iteration 532, loss = 0.83768333
Iteration 533, loss = 0.83601562
Iteration 534, loss = 0.83432599
Iteration 535, loss = 0.83261773
Iteration 536, loss = 0.83089470
Iteration 537, loss = 0.82916124
Iteration 538, loss = 0.82742220
Iteration 539, loss = 0.82568284
Iteration 540, loss = 0.82394880
Iteration 541, loss = 0.82222603
Iteration 542, loss = 0.82052071
Iteration 543, loss = 0.81883911
Iteration 544, loss = 0.81718747
Iteration 545, loss = 0.81557188
Iteration 546, loss = 0.81399815
Iteration 547, loss = 0.81247168
Iteration 548, loss = 0.81099741
Iteration 549, loss = 0.80957977
Iteration 550, loss = 0.80822259
Iteration 551, loss = 0.80692904
Iteration 552, loss = 0.80570158
Iteration 553, loss = 0.80454191
Iteration 554, loss = 0.80345090
Iteration 555, loss = 0.80242858
Iteration 556, loss = 0.80147413
Iteration 557, loss = 0.80058586
Iteration 558, loss = 0.79976126
Iteration 559, loss = 0.79899696
Iteration 560, loss = 0.79828880
Iteration 561, loss = 0.79763188
Iteration 562, loss = 0.79702060
Iteration 563, loss = 0.79644870
Iteration 564, loss = 0.79590933
Iteration 565, loss = 0.79539511
Iteration 566, loss = 0.79489824
Iteration 567, loss = 0.79441054
Iteration 568, loss = 0.79392353
Iteration 569, loss = 0.79342852
Iteration 570, loss = 0.79291670
Iteration 571, loss = 0.79237931
Iteration 572, loss = 0.79180778
Iteration 573, loss = 0.79119384
Iteration 574, loss = 0.79052966
Iteration 575, loss = 0.78980800
Iteration 576, loss = 0.78902241
Iteration 577, loss = 0.78816737
Iteration 578, loss = 0.78723844
Iteration 579, loss = 0.78623251
Iteration 580, loss = 0.78514797
Iteration 581, loss = 0.78398485
Iteration 582, loss = 0.78274501
Iteration 583, loss = 0.78143220
Iteration 584, loss = 0.78005206
Iteration 585, loss = 0.77861222
Iteration 586, loss = 0.77712139
Iteration 587, loss = 0.77558944
Iteration 588, loss = 0.77402744
Iteration 589, loss = 0.77244766
Iteration 590, loss = 0.77086350
Iteration 591, loss = 0.76928929
Iteration 592, loss = 0.76774026
Iteration 593, loss = 0.76623232
```