pulsar6

October 8, 2022

1 Pulsar Emission Data Analysis

2 All Imports that may or may not be needed and used for the notebook

[ ]:
```
#currently including any and all Imports that maybe needed for the project.
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.feature_selection import RFE
import datetime as dt
from sklearn.cluster import KMeans
from sklearn.metrics import pairwise_distances
from scipy.cluster.hierarchy import linkage, dendrogram, cut_tree
from scipy.spatial.distance import pdist
from sklearn.feature_extraction.text import TfidfVectorizer
import matplotlib.dates as mdates
from scipy.stats import pearsonr
from scipy import stats
import statistics
import math
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import acf, pacf
from statsmodels.tsa.tsatools import lagmat
from numpy import array
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dense
from keras.layers import Bidirectional
```

3 Section for extracting from a tar file.

Currently implemented for original TAR File structure.

[ ]:
```
#This is also found in the main file under tarunzip.py
import tarfile
import os
import sys

#tar = tarfile.open("pulseTarFile.tar")
#tar.extractall('./Data')
#tar.close()
```

[ ]:
```
#How to remove outlier data for these datasets.
#pulsar6npcleaned = pulsar6[(np.abs(stats.zscore(pulsar6["Brightness"])) <3)]
#pulsar6npcleaned
```

3.1 Beginning of Exploration

3.1.1 Examining the data

In this section we are determining the total integrity of the data to determine if further comprehensive data cleaning and uniforming processes are needed.

[ ]:
```
colnames = ['Pulse Number', 'Brightness', 'Uncertainty']
pulsar = pd.read_csv("Data/J1644-4559.pulses", sep = ' ', header = None, names␣
↪= colnames)
```

[ ]:
```
pulsar.shape
```

[ ]:
```
(698, 3)
```

[ ]:
```
pulsar.head(25)
```

[ ]:

	Pulse Number	Brightness	Uncertainty
0	1	0.634671	0.002761
1	2	0.736945	0.005207
2	3	0.693834	0.002706
3	4	1.021866	0.010184
4	5	0.673845	0.006236
5	6	0.676883	0.004763
6	7	0.527039	0.002422
7	8	0.673417	0.003174
8	9	0.357076	0.002848
9	10	0.661704	0.005588
10	11	0.545564	0.003835
11	12	0.494655	0.003145
12	13	0.804260	0.005258
13	14	0.513362	0.005700
14	15	0.477025	0.002945

```
15            16    0.399571     0.004712
16            17    0.188069     0.002452
17            18    0.748592     0.005468
18            19    0.723437     0.004548
19            20    0.960154     0.006765
20            21    0.707715     0.006011
21            22    1.074550     0.006831
22            23    0.961340     0.006617
23            24    0.754457     0.004117
24            25    0.773151     0.004920
```

[ ]:
```
pulsar.describe()
```

[ ]:
```
       Pulse Number  Brightness  Uncertainty
count     698.00000  698.000000   698.000000
mean      349.50000    0.654319     0.004445
std       201.63953    0.163945     0.001855
min         1.00000    0.007642     0.002129
25%       175.25000    0.555267     0.003086
50%       349.50000    0.658295     0.003951
75%       523.75000    0.753396     0.005349
max       698.00000    1.159334     0.016097
```

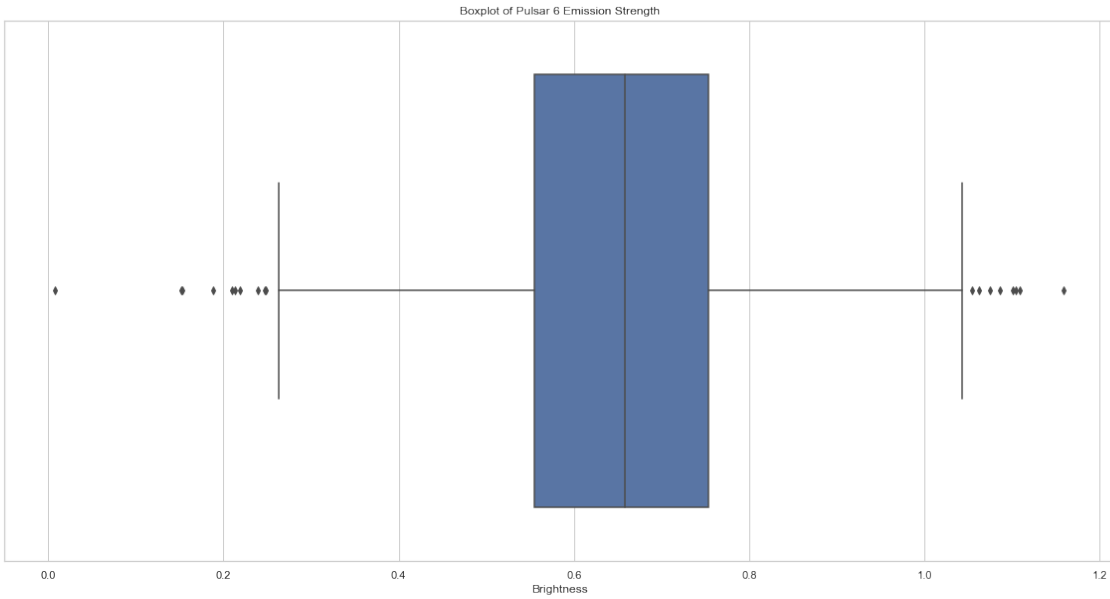
[ ]:
```
nullBoolBrightness = pd.isnull(pulsar["Brightness"])

pulsar[nullBoolBrightness]
```

[ ]:
```
Empty DataFrame
Columns: [Pulse Number, Brightness, Uncertainty]
Index: []
```

[ ]:
```
pulsar["Brightness"].describe()
```

[ ]:
```
count    698.000000
mean       0.654319
std        0.163945
min        0.007642
25%        0.555267
50%        0.658295
75%        0.753396
max        1.159334
Name: Brightness, dtype: float64
```

[ ]:
```
plt.figure(figsize=(20,10))
sns.set_theme(style="whitegrid")
ax = sns.boxplot(x=pulsar["Brightness"]).set_title("Boxplot of Pulsar 6␣
 ↪Emission Strength")
```



[ ]:
```
medianpulse6 = pulsar["Brightness"].median()
print("Median of Pulsar6: ", medianpulse6)
pulsar['Binary'] = np.where(pulsar['Brightness'] > medianpulse6, 1, 0)
```

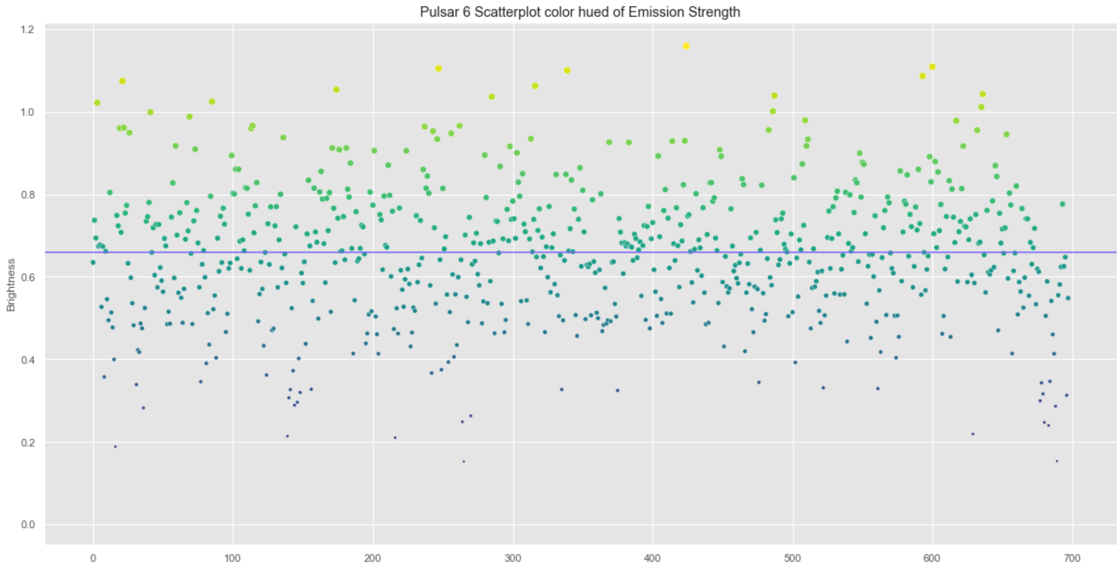
```
Median of Pulsar6:  0.65829515
```

[ ]:
```
pulsar
```

[ ]:
```
     Pulse Number  Brightness  Uncertainty  Binary
0               1    0.634671     0.002761       0
1               2    0.736945     0.005207       1
2               3    0.693834     0.002706       1
3               4    1.021866     0.010184       1
4               5    0.673845     0.006236       1
..            …           …            …       …
693           694    0.776083     0.008928       1
694           695    0.625382     0.006018       0
695           696    0.647559     0.003765       0
696           697    0.312449     0.002901       0
697           698    0.548353     0.009056       0

[698 rows x 4 columns]
```

[ ]:
```
plt.figure(figsize=(20,10))
sns.set_style("darkgrid", {"axes.facecolor": ".75"})
strength = pulsar.Brightness.values
plt.style.use('ggplot')
```

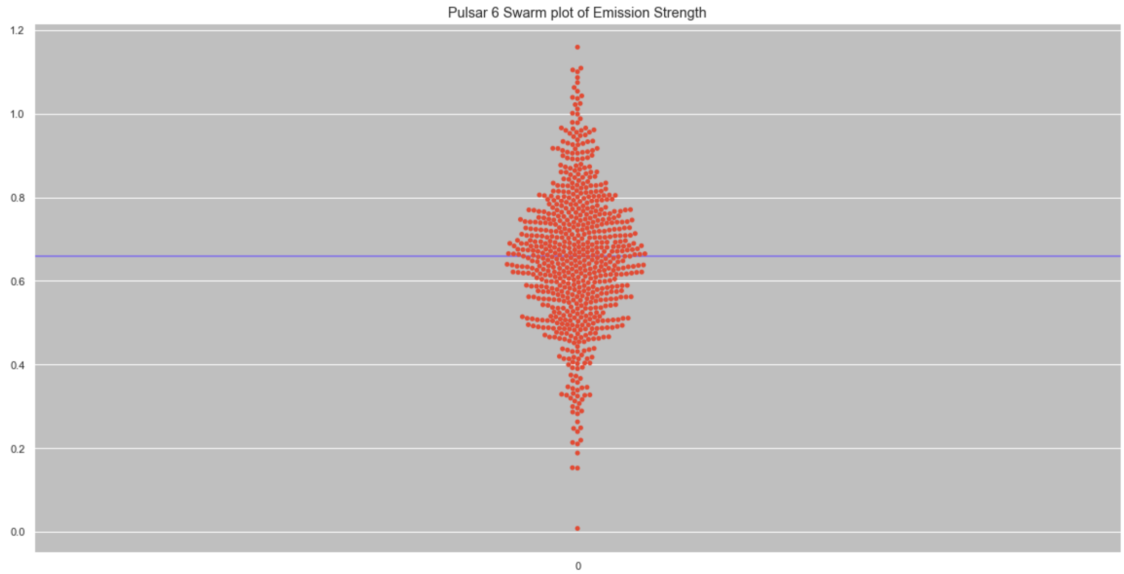
```
ax = sns.scatterplot(data=pulsar["Brightness"], s= strength*50, c=strength,␣
↪cmap="viridis", marker="o").set_title('Pulsar 6 Scatterplot color hued of␣
↪Emission Strength')
ax= plt.axhline( y=0.65829515, ls='-',c='mediumslateblue')
```



[ ]:
```
print(len(pulsar[(pulsar.Brightness > 0.6589028)]))
print(len(pulsar[(pulsar.Brightness < 0.6589028)]))
```

348
350

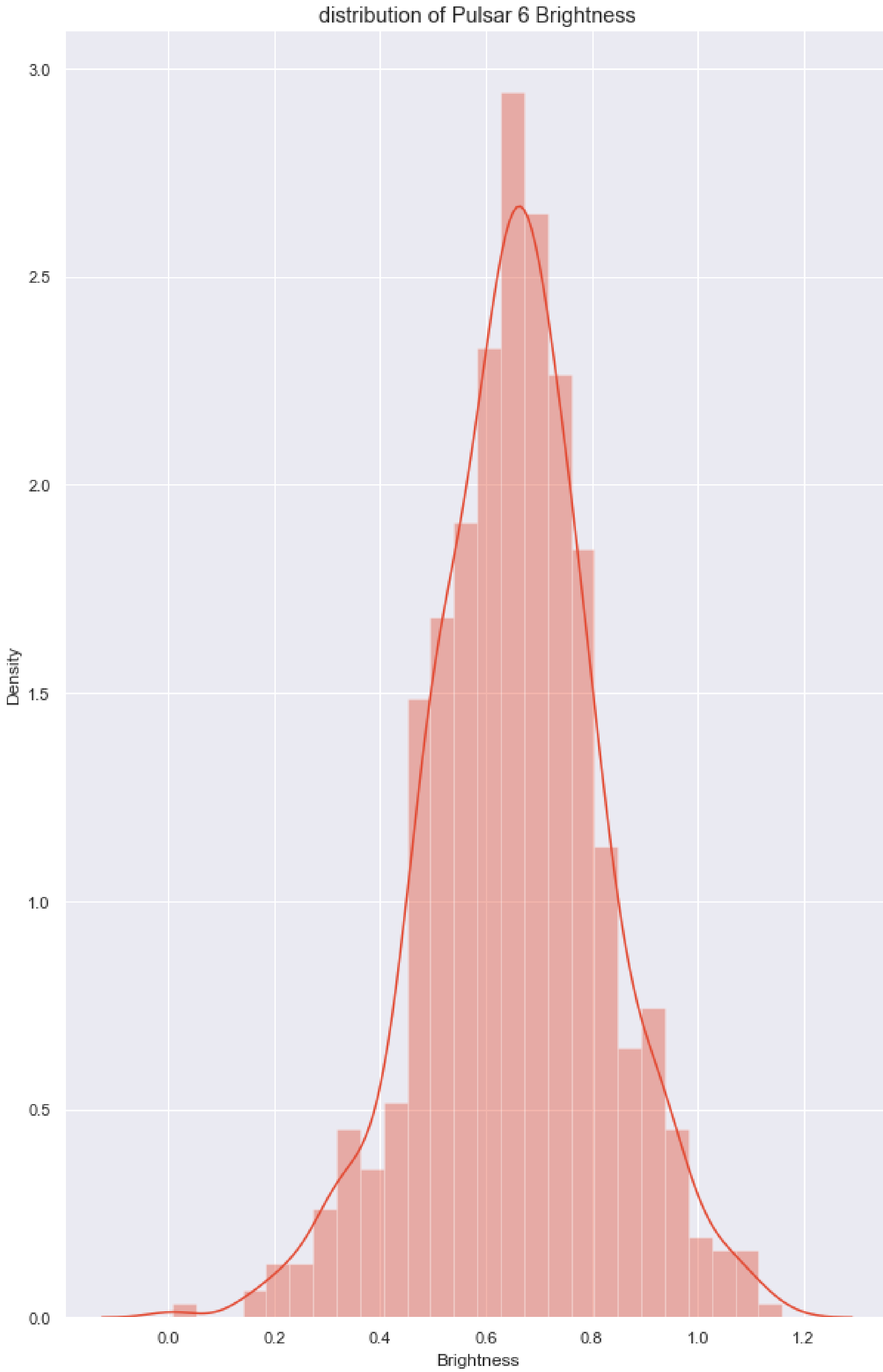
[ ]:
```
plt.figure(figsize=(20,10))
sns.set_style("darkgrid", {"axes.facecolor": ".75"})
strength = pulsar.Brightness.values
ax = plt.axhline( y=0.65829515, ls='-',c='mediumslateblue')
ax = sns.swarmplot(data=pulsar["Brightness"], c="blue").set_title('Pulsar 6␣
↪Swarm plot of Emission Strength')
```




```
[ ]: plt.figure(figsize=(10, 16))
     with sns.axes_style('darkgrid'):
         sns.distplot(pulsar.Brightness)
     plt.title("distribution of Pulsar 6 Brightness")
```

```
c:\Users\oxlay\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

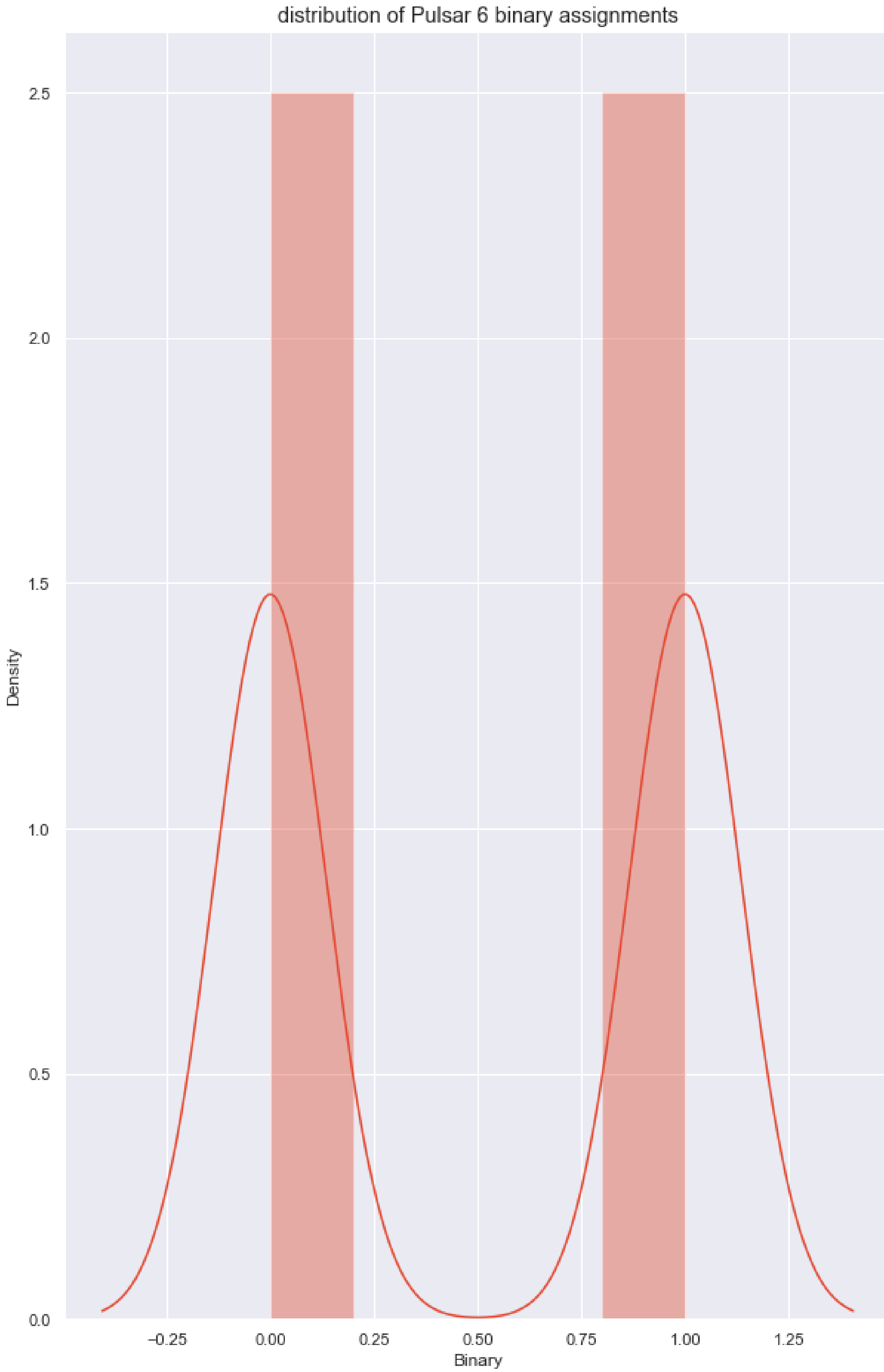
```
[ ]: Text(0.5, 1.0, 'distribution of Pulsar 6 Brightness')
```


distribution of Pulsar 6 Brightness
Density
Brightness
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2

[ ]:
```
plt.figure(figsize=(10, 16))
with sns.axes_style('darkgrid'):
    sns.distplot(pulsar.Binary)
plt.title("distribution of Pulsar 6 binary assignments")
```

```
c:\Users\oxlay\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```

[ ]:
```
Text(0.5, 1.0, 'distribution of Pulsar 6 binary assignments')
```


distribution of Pulsar 6 binary assignments
Density
Binary
0.0
0.5
1.0
1.5
2.0
2.5
−0.25
0.00
0.25
0.50
0.75
1.00
1.25

3.2 Binary Classification

[ ]:
```
X = pulsar[['Brightness', 'Uncertainty']]
y = pulsar['Binary']
```

[ ]:
```
X.head()
```

[ ]:

	Brightness	Uncertainty
0	0.634671	0.002761
1	0.736945	0.005207
2	0.693834	0.002706
3	1.021866	0.010184
4	0.673845	0.006236

[ ]:
```
y.head()
```

[ ]:
```
0    0
1    1
2    1
3    1
4    1
Name: Binary, dtype: int32
```

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

X_train, X_test, y_train, y_test = train_test_split(X, y , test_size=0.20)
```

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

train_scaler = StandardScaler()
X_train = train_scaler.fit_transform(X_train)

test_scaler = StandardScaler()
X_test = test_scaler.fit_transform(X_test)
```

[ ]:
```
model = LogisticRegression()

model.fit(X_train, y_train)
```

[ ]:
```
LogisticRegression()
```

[ ]:
```
predictions = model.predict(X_test)
```

[ ]:
```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(y_test, predictions)

TN, FP, FN, TP = confusion_matrix(y_test, predictions).ravel()

print('True Positive(TP)  = ', TP)
print('False Positive(FP) = ', FP)
print('True Negative(TN)  = ', TN)
print('False Negative(FN) = ', FN)
```

```
True Positive(TP)  =  68
False Positive(FP) =  3
True Negative(TN)  =  69
False Negative(FN) =  0
```

[ ]:
```
accuracy =  (TP + TN) / (TP + FP + TN + FN)

print("Accuracy of the model is ", accuracy)
```

```
Accuracy of the model is  0.9785714285714285
```

3.3 Bidirectional LSTM Model

[ ]:
```
brightness_list = list(pulsar['Brightness'])
brightness_list[:10]
```

[ ]:
```
[0.6346714,
 0.7369454,
 0.6938341,
 1.021866,
 0.6738453,
 0.6768825,
 0.5270392,
 0.6734173,
 0.3570756,
 0.6617037]
```

[ ]:
```
def split_list(blist, steps):
    X, y = list(), list()
    for i in range(len(blist)):
        # find the end of this pattern
        end_ix = i + steps
        # check if we are beyond the sequence
        if end_ix > len(blist)-1:
            break
        # gather input and output parts of the pattern
        list_x, list_y = blist[i:end_ix], blist[end_ix]
        X.append(list_x)
        y.append(list_y)
```

```python
    return array(X), array(y)
```

[ ]:
```python
X, y = split_list(brightness_list, 100)
X = X.reshape((X.shape[0], X.shape[1], 1))
X[:1]
```

[ ]:
```
array([[[0.6346714 ],
        [0.7369454 ],
        [0.6938341 ],
        [1.021866  ],
        [0.6738453 ],
        [0.6768825 ],
        [0.5270392 ],
        [0.6734173 ],
        [0.3570756 ],
        [0.6617037 ],
        [0.5455637 ],
        [0.4946546 ],
        [0.8042599 ],
        [0.5133624 ],
        [0.4770252 ],
        [0.3995709 ],
        [0.1880686 ],
        [0.7485923 ],
        [0.723437  ],
        [0.960154  ],
        [0.7077145 ],
        [1.07455   ],
        [0.9613396 ],
        [0.7544566 ],
        [0.7731512 ],
        [0.6326247 ],
        [0.9491536 ],
        [0.5976236 ],
        [0.5358455 ],
        [0.4823253 ],
        [0.00764172],
        [0.3384609 ],
        [0.4226295 ],
        [0.4172967 ],
        [0.4866613 ],
        [0.474994  ],
        [0.2820107 ],
        [0.5241109 ],
        [0.7348035 ],
        [0.7454862 ],
        [0.7803042 ],
```

```
[0.9990716 ],
[0.6589876 ],
[0.7188401 ],
[0.6039976 ],
[0.7266428 ],
[0.5741913 ],
[0.7266481 ],
[0.6223379 ],
[0.5905332 ],
[0.5636434 ],
[0.6927038 ],
[0.6748244 ],
[0.4850771 ],
[0.5154094 ],
[0.4861331 ],
[0.7445553 ],
[0.8274334 ],
[0.5971083 ],
[0.9172228 ],
[0.7009897 ],
[0.5611873 ],
[0.755055  ],
[0.548961  ],
[0.4883782 ],
[0.5705138 ],
[0.6909211 ],
[0.7797759 ],
[0.7110825 ],
[0.9880999 ],
[0.6565107 ],
[0.4854965 ],
[0.7354586 ],
[0.9091333 ],
[0.7606739 ],
[0.6818882 ],
[0.5743544 ],
[0.3454394 ],
[0.6301639 ],
[0.6635159 ],
[0.5985485 ],
[0.3900236 ],
[0.5116962 ],
[0.4355926 ],
[0.7949124 ],
[1.0245    ],
[0.5212468 ],
[0.5545024 ],
```

```
 [0.4035    ],
 [0.6934347 ],
 [0.6132591 ],
 [0.7468123 ],
 [0.6344795 ],
 [0.7662029 ],
 [0.7272725 ],
 [0.4662485 ],
 [0.5103499 ],
 [0.6203424 ],
 [0.6333662 ],
 [0.8940519 ]]])
```

[ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y , test_size=0.20)
```

[ ]:
```
model = Sequential()
model.add(Bidirectional(LSTM(50, activation='relu'), input_shape=(100, 1)))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam',␣
 ↪metrics=['accuracy'])
```

[ ]:
```
history = model.fit(X_train, y_train, epochs=200, verbose=1, batch_size=10)
```

```
Epoch 1/200
48/48 [==============================] - 2s 20ms/step - loss: 0.6519 - accuracy:
0.0000e+00
Epoch 2/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6433 - accuracy:
0.0000e+00
Epoch 3/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6435 - accuracy:
0.0000e+00
Epoch 4/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6434 - accuracy:
0.0000e+00
Epoch 5/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6426 - accuracy:
0.0000e+00
Epoch 6/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6430 - accuracy:
0.0000e+00
Epoch 7/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6425 - accuracy:
0.0000e+00
Epoch 8/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6425 - accuracy:
0.0000e+00
```

```
Epoch 9/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6423 - accuracy:
0.0000e+00
Epoch 10/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6418 - accuracy:
0.0000e+00
Epoch 11/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6416 - accuracy:
0.0000e+00
Epoch 12/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6417 - accuracy:
0.0000e+00
Epoch 13/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6413 - accuracy:
0.0000e+00
Epoch 14/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6412 - accuracy:
0.0000e+00
Epoch 15/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6413 - accuracy:
0.0000e+00
Epoch 16/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6409 - accuracy:
0.0000e+00
Epoch 17/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6407 - accuracy:
0.0000e+00
Epoch 18/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6405 - accuracy:
0.0000e+00
Epoch 19/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6407 - accuracy:
0.0000e+00
Epoch 20/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6401 - accuracy:
0.0000e+00
Epoch 21/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6399 - accuracy:
0.0000e+00
Epoch 22/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6395 - accuracy:
0.0000e+00
Epoch 23/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6402 - accuracy:
0.0000e+00
Epoch 24/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6400 - accuracy:
0.0000e+00
```

```
Epoch 25/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6392 - accuracy:
0.0000e+00
Epoch 26/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6405 - accuracy:
0.0000e+00
Epoch 27/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6406 - accuracy:
0.0000e+00
Epoch 28/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6401 - accuracy:
0.0000e+00
Epoch 29/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6399 - accuracy:
0.0000e+00
Epoch 30/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6393 - accuracy:
0.0000e+00
Epoch 31/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6398 - accuracy:
0.0000e+00
Epoch 32/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6397 - accuracy:
0.0000e+00
Epoch 33/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6395 - accuracy:
0.0000e+00
Epoch 34/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6391 - accuracy:
0.0000e+00
Epoch 35/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6386 - accuracy:
0.0000e+00
Epoch 36/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6388 - accuracy:
0.0000e+00
Epoch 37/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6390 - accuracy:
0.0000e+00
Epoch 38/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6391 - accuracy:
0.0000e+00
Epoch 39/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6390 - accuracy:
0.0000e+00
Epoch 40/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6387 - accuracy:
0.0000e+00
```

```
Epoch 41/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6390 - accuracy:
0.0000e+00
Epoch 42/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6387 - accuracy:
0.0000e+00
Epoch 43/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 44/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6388 - accuracy:
0.0000e+00
Epoch 45/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 46/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6389 - accuracy:
0.0000e+00
Epoch 47/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 48/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 49/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6395 - accuracy:
0.0000e+00
Epoch 50/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 51/200
48/48 [==============================] - 1s 25ms/step - loss: 0.6389 - accuracy:
0.0000e+00
Epoch 52/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6387 - accuracy:
0.0000e+00
Epoch 53/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6394 - accuracy:
0.0000e+00
Epoch 54/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6387 - accuracy:
0.0000e+00
Epoch 55/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6390 - accuracy:
0.0000e+00
Epoch 56/200
48/48 [==============================] - 1s 21ms/step - loss: 1.0027 - accuracy:
0.0000e+00
```

```
Epoch 57/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6400 - accuracy:
0.0000e+00
Epoch 58/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6394 - accuracy:
0.0000e+00
Epoch 59/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6399 - accuracy:
0.0000e+00
Epoch 60/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6394 - accuracy:
0.0000e+00
Epoch 61/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6393 - accuracy:
0.0000e+00
Epoch 62/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6393 - accuracy:
0.0000e+00
Epoch 63/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6392 - accuracy:
0.0000e+00
Epoch 64/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6393 - accuracy:
0.0000e+00
Epoch 65/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6390 - accuracy:
0.0000e+00
Epoch 66/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6391 - accuracy:
0.0000e+00
Epoch 67/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6390 - accuracy:
0.0000e+00
Epoch 68/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6388 - accuracy:
0.0000e+00
Epoch 69/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6387 - accuracy:
0.0000e+00
Epoch 70/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6388 - accuracy:
0.0000e+00
Epoch 71/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6386 - accuracy:
0.0000e+00
Epoch 72/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6386 - accuracy:
0.0000e+00
```

```
Epoch 73/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 74/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6386 - accuracy:
0.0000e+00
Epoch 75/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 76/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 77/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6387 - accuracy:
0.0000e+00
Epoch 78/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 79/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 80/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 81/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 82/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 83/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 84/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 85/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 86/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 87/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 88/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6381 - accuracy:
0.0000e+00
```

```
Epoch 89/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 90/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 91/200
48/48 [==============================] - 1s 23ms/step - loss: 0.6386 - accuracy:
0.0000e+00
Epoch 92/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 93/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6381 - accuracy:
0.0000e+00
Epoch 94/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 95/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 96/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 97/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 98/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 99/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 100/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6376 - accuracy:
0.0000e+00
Epoch 101/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 102/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6378 - accuracy:
0.0000e+00
Epoch 103/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 104/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6376 - accuracy:
0.0000e+00
```

```
Epoch 105/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 106/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 107/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6376 - accuracy:
0.0000e+00
Epoch 108/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 109/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 110/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 111/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 112/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 113/200
48/48 [==============================] - 1s 21ms/step - loss: 10.3085 -
accuracy: 0.0000e+00
Epoch 114/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6385 - accuracy:
0.0000e+00
Epoch 115/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 116/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6382 - accuracy:
0.0000e+00
Epoch 117/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6384 - accuracy:
0.0000e+00
Epoch 118/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 119/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 120/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6381 - accuracy:
0.0000e+00
```

```
Epoch 121/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6383 - accuracy:
0.0000e+00
Epoch 122/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 123/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6381 - accuracy:
0.0000e+00
Epoch 124/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 125/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 126/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6378 - accuracy:
0.0000e+00
Epoch 127/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 128/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 129/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 130/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6378 - accuracy:
0.0000e+00
Epoch 131/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 132/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 133/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 134/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6380 - accuracy:
0.0000e+00
Epoch 135/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 136/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6376 - accuracy:
0.0000e+00
```

```
Epoch 137/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6376 - accuracy:
0.0000e+00
Epoch 138/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 139/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 140/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 141/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 142/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 143/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6378 - accuracy:
0.0000e+00
Epoch 144/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6376 - accuracy:
0.0000e+00
Epoch 145/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 146/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 147/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 148/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 149/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 150/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 151/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 152/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
```

```
Epoch 153/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 154/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6377 - accuracy:
0.0000e+00
Epoch 155/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6373 - accuracy:
0.0000e+00
Epoch 156/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6373 - accuracy:
0.0000e+00
Epoch 157/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6373 - accuracy:
0.0000e+00
Epoch 158/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 159/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6370 - accuracy:
0.0000e+00
Epoch 160/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 161/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 162/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 163/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 164/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 165/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6370 - accuracy:
0.0000e+00
Epoch 166/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 167/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 168/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6374 - accuracy:
0.0000e+00
```

```
Epoch 169/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 170/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 171/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6369 - accuracy:
0.0000e+00
Epoch 172/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6374 - accuracy:
0.0000e+00
Epoch 173/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6375 - accuracy:
0.0000e+00
Epoch 174/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 175/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 176/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6378 - accuracy:
0.0000e+00
Epoch 177/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 178/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6379 - accuracy:
0.0000e+00
Epoch 179/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 180/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 181/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 182/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6373 - accuracy:
0.0000e+00
Epoch 183/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6370 - accuracy:
0.0000e+00
Epoch 184/200
48/48 [==============================] - 1s 22ms/step - loss: 0.6368 - accuracy:
0.0000e+00
```

```
Epoch 185/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 186/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6370 - accuracy:
0.0000e+00
Epoch 187/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 188/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6369 - accuracy:
0.0000e+00
Epoch 189/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6369 - accuracy:
0.0000e+00
Epoch 190/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6366 - accuracy:
0.0000e+00
Epoch 191/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6372 - accuracy:
0.0000e+00
Epoch 192/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6365 - accuracy:
0.0000e+00
Epoch 193/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6367 - accuracy:
0.0000e+00
Epoch 194/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6369 - accuracy:
0.0000e+00
Epoch 195/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6369 - accuracy:
0.0000e+00
Epoch 196/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6367 - accuracy:
0.0000e+00
Epoch 197/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6366 - accuracy:
0.0000e+00
Epoch 198/200
48/48 [==============================] - 1s 21ms/step - loss: 0.6371 - accuracy:
0.0000e+00
Epoch 199/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6366 - accuracy:
0.0000e+00
Epoch 200/200
48/48 [==============================] - 1s 20ms/step - loss: 0.6366 - accuracy:
0.0000e+00
```

[ ]:
```
y_pred = model.predict(X_test, verbose=0)
y_pred[:10]
```

[ ]:
```
array([[0.8022064 ],
       [0.70461446],
       [0.66105044],
       [0.6896631 ],
       [0.68478507],
       [0.7015957 ],
       [0.6640747 ],
       [0.50144947],
       [0.63427603],
       [0.72587335]], dtype=float32)
```

[ ]:
```
model.evaluate(X_test, y_test)
```

```
4/4 [==============================] - 0s 7ms/step - loss: 0.6307 - accuracy:
0.0000e+00
```

[ ]:
```
[0.6307459473609924, 0.0]
```

3.4 ML Evaluation.

3.4.1 Logistic Regression

This model appears to have gained some insight in the data and accurately defined a majority of the data. The accuracy of the model is >95% which indicates that it was able to determine a trend and apply it in a useful manner in the predictions during evaluation. Further, the confusion matrix further supports the high accuracy and likely usefulness of the model with only 3 false assignments. However, in analysis this is only to determine if there is a correlation between binary assignment and the emission strength x error in measurement. This doesn't aid us in our overall randomness determination, but it does determine that uncertainty has a role in the binary assignment and the overall trust of emission strength.

3.4.2 Bidirectional LSTM

This model is very error prone as the loss value is consistently at 60& or higher at every epoch during training and at exactly 63.07% in evaluation with a 0% accuracy this indicates that there is either a great error in the formation of the model, data used or trend being obtained. Alternatively it could indicate that there is no trend there to predict. Likely this indicates that the model is not valuable for any meaningful analysis.

4 Preliminary runs test

4.0.1 Math Logic

$$Z = \frac{R - \tilde{R}}{s_R}$$

$$\tilde{R} = \frac{2_{n1n2}}{n1 + n2} + 1$$

$$s_R^2 = \frac{2_{n1n2}(2n1n2 - n1 - n2)}{(n1 + n2)^2(n1 + n2 - 1)}$$

link to resource: https://www.geeksforgeeks.org/runs-test-of-randomness-in-python/

$ Z_{critical} = 1.96 $ as the confidence interval level of 95% thus this is a 2 tailed test. If the probability as corrosponding to this confidence interval $ H_{null} $ will be rejected as it is not statistically significant as denoted by $|Z| > Z_{critical} $

There is also code attempting to change it from a z-score probability to a P-score for ease of understanding and clarity.

5 FUNCTION CODE FOR RUNS TEST

[ ]:
```
binaryData1 = pulsar['Binary'].tolist()
print("pulsar6 original: ",binaryData1)
```

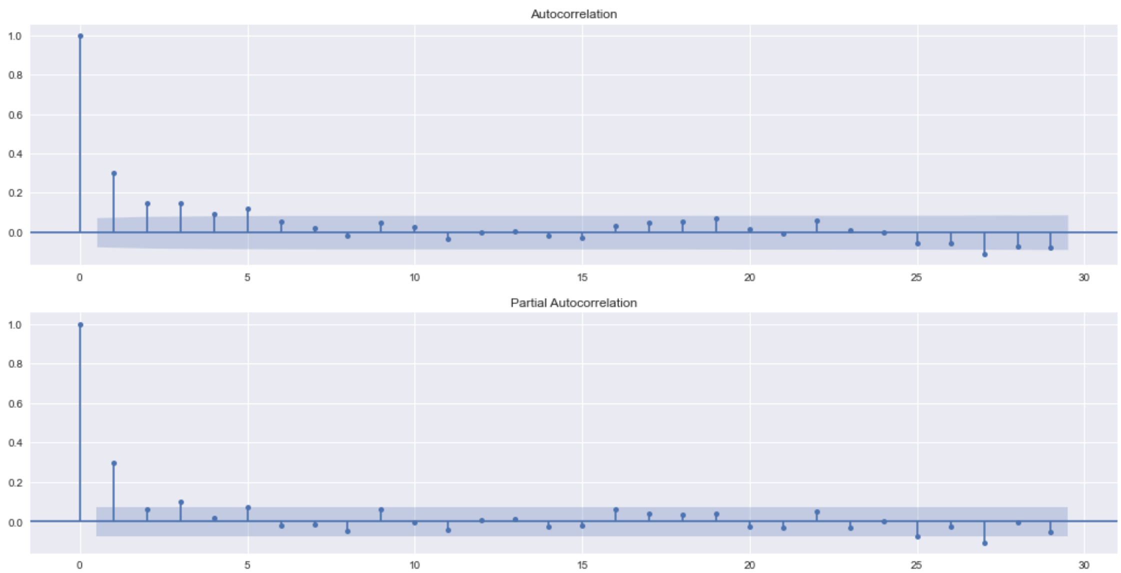
```
pulsar6 original:  [0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1,
1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0,
1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1,
1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1,
0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1,
1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1,
1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1,
1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1,
0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1,
0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1,
0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1,
1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0,
0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1,
0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1,
1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1,
1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1,
0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0,
0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0,
1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0,
0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0]
```

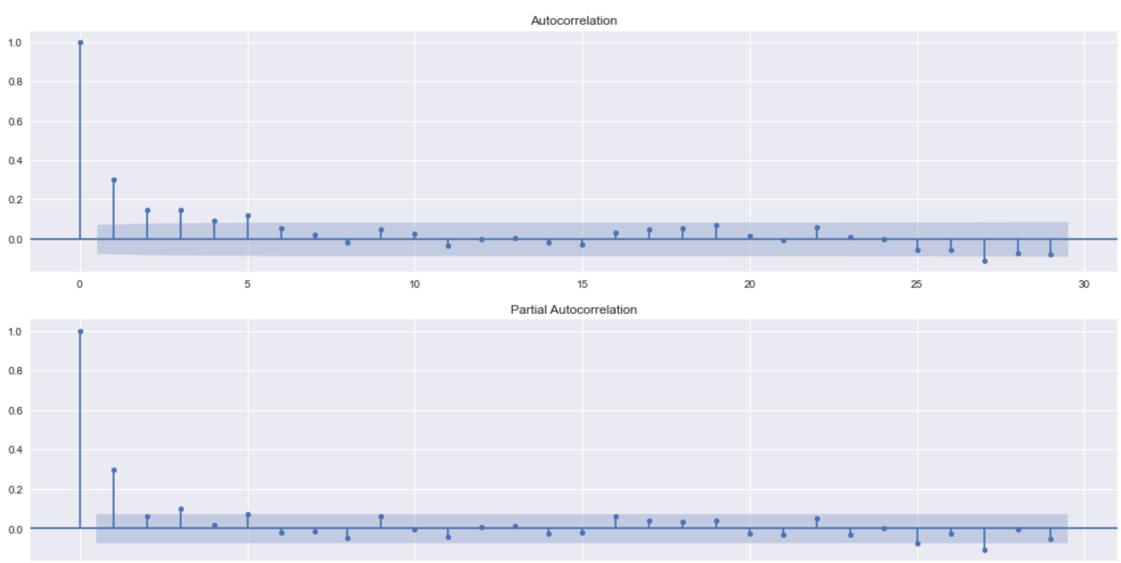
6 Below we begin autocorrelation and autocovariance analysis

To get started with this I am playing around with guide from: https://towardsdatascience.com/a-step-by-step-guide-to-calculating-autocorrelation-and-partial-autocorrelation-8c4342b784e8

[ ]:
```
plt.style.use("seaborn")
plt.rcParams["figure.figsize"] = (18, 9)

fig, ax = plt.subplots(2,1)

plot_acf(pulsar['Brightness'], ax=ax[0])
plot_pacf(pulsar['Brightness'], ax=ax[1], method="ols")
```

[ ]:





[ ]:
```
acf(pulsar['Brightness'], nlags=10)
```

```
c:\Users\oxlay\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:667:
FutureWarning: fft=True will become the default after the release of the 0.12
release of statsmodels. To suppress this warning, explicitly set fft=False.
  warnings.warn(
```

[ ]:
```
array([ 1.        ,  0.29929122,  0.14656878,  0.14948301,  0.09384681,
        0.11707783,  0.05493324,  0.02160374, -0.01711482,  0.04777   ,
        0.02563995])
```

[ ]:
```
acfpulsar = pd.DataFrame()
for lag in range(0,11):
    acfpulsar[f"B_lag_{lag}"] = pulsar['Brightness'].shift(lag)


acfpulsar
```

[ ]:
```
       B_lag_0   B_lag_1   B_lag_2   B_lag_3   B_lag_4   B_lag_5   B_lag_6  \
0     0.634671       NaN       NaN       NaN       NaN       NaN       NaN
1     0.736945  0.634671       NaN       NaN       NaN       NaN       NaN
2     0.693834  0.736945  0.634671       NaN       NaN       NaN       NaN
3     1.021866  0.693834  0.736945  0.634671       NaN       NaN       NaN
4     0.673845  1.021866  0.693834  0.736945  0.634671       NaN       NaN
..         …         …         …         …         …         …         …
693   0.776083  0.623757  0.581248  0.555266  0.152886  0.286132  0.413354
694   0.625382  0.776083  0.623757  0.581248  0.555266  0.152886  0.286132
695   0.647559  0.625382  0.776083  0.623757  0.581248  0.555266  0.152886
696   0.312449  0.647559  0.625382  0.776083  0.623757  0.581248  0.555266
697   0.548353  0.312449  0.647559  0.625382  0.776083  0.623757  0.581248

       B_lag_7   B_lag_8   B_lag_9  B_lag_10
0          NaN       NaN       NaN       NaN
1          NaN       NaN       NaN       NaN
2          NaN       NaN       NaN       NaN
3          NaN       NaN       NaN       NaN
4          NaN       NaN       NaN       NaN
..         …         …         …         …
693   0.460095  0.541486  0.346502  0.239302
694   0.413354  0.460095  0.541486  0.346502
695   0.286132  0.413354  0.460095  0.541486
696   0.152886  0.286132  0.413354  0.460095
697   0.555266  0.152886  0.286132  0.413354

[698 rows x 11 columns]
```

[ ]:
```
acfpulsar.corr()["B_lag_0"].values
```

[ ]:
```
array([ 1.        ,  0.29938402,  0.14710414,  0.15003691,  0.09455452,
        0.11800036,  0.05537751,  0.02179885, -0.01724535,  0.04863954,
        0.02621294])
```

6.0.1 Getting every 5th as per the auto correlation

6.0.2 Creating a new set of discrete 100 sets and examining them specifically

6.0.3 Further Random testing to move into extensive testing

Getting every 5th as per the auto correlation

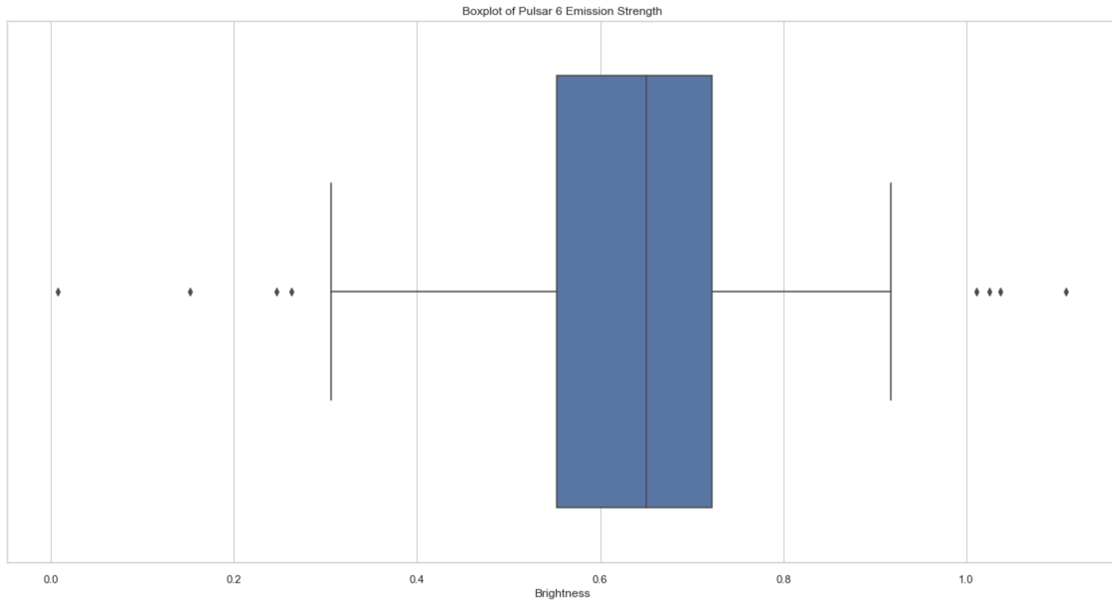
[ ]:
```
held5ths = pulsar[pulsar.index % 5 == 0]
held5ths
```

[ ]:
```
     Pulse Number  Brightness  Uncertainty  Binary
0               1    0.634671     0.002761       0
5               6    0.676883     0.004763       1
10             11    0.545564     0.003835       0
15             16    0.399571     0.004712       0
20             21    0.707715     0.006011       1
..            …          …            …       …
675           676    0.618826     0.002507       0
680           681    0.246916     0.004276       0
685           686    0.541486     0.003149       0
690           691    0.555266     0.003657       0
695           696    0.647559     0.003765       0

[140 rows x 4 columns]
```

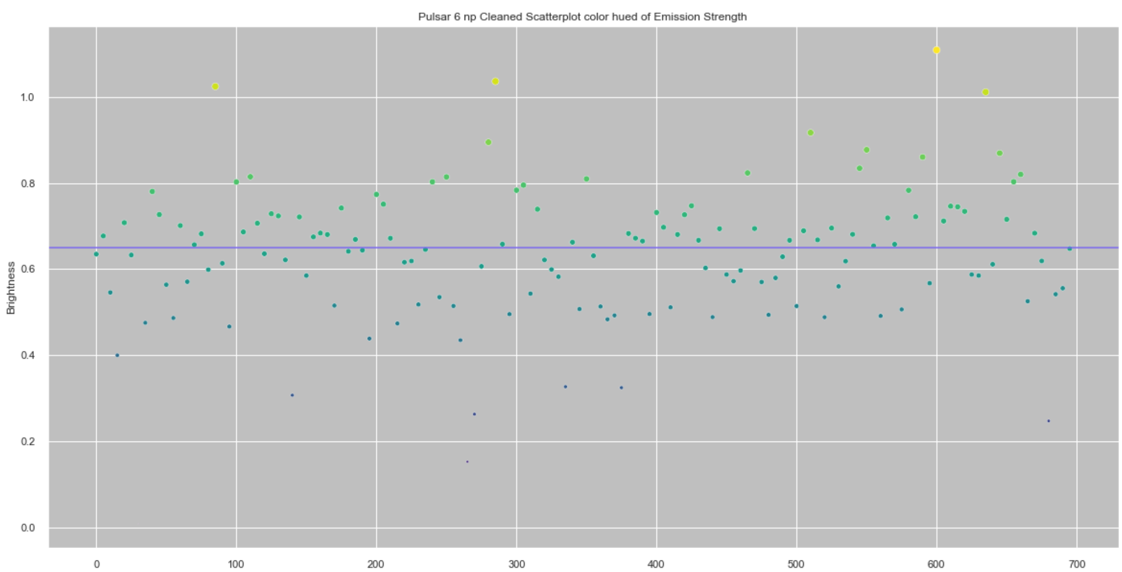
[ ]:
```
medianheld5ths = held5ths["Brightness"].median()
medianheld5ths
```

[ ]:
```
0.6508051
```

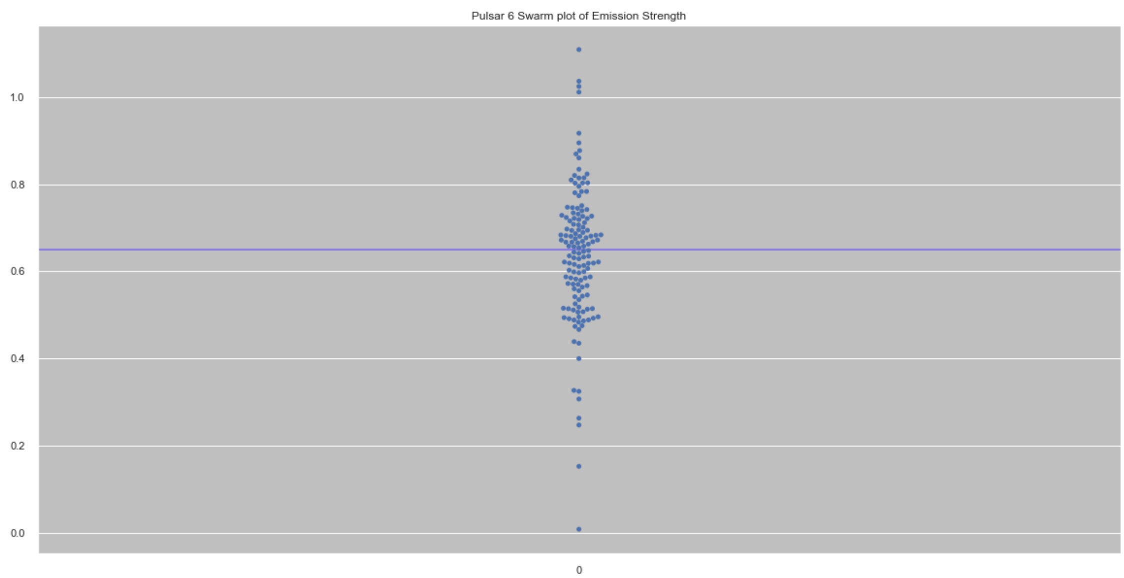
[ ]:
```
plt.figure(figsize=(20,10))
sns.set_theme(style="whitegrid")
ax = sns.boxplot(x=held5ths["Brightness"]).set_title("Boxplot of Pulsar 6␣
↪Emission Strength")
```



[ ]:
```
plt.figure(figsize=(20,10))
sns.set_style("darkgrid", {"axes.facecolor": ".75"})
strength = held5ths.Brightness.values
ax = sns.scatterplot(data=held5ths["Brightness"], s= strength*50, c=strength,␣
 ↪cmap="viridis", marker="o").set_title('Pulsar 6 np Cleaned Scatterplot color␣
 ↪hued of Emission Strength')
ax = plt.axhline( y=0.6508051, ls='-',c='mediumslateblue')
```



[ ]:
```
plt.figure(figsize=(20,10))
sns.set_style("darkgrid", {"axes.facecolor": ".75"})
strength = held5ths.Brightness.values
ax = plt.axhline( y=0.6508051, ls='-',c='mediumslateblue')
ax = sns.swarmplot(data=held5ths["Brightness"], c="blue").set_title('Pulsar 6␣
↪Swarm plot of Emission Strength')
```



[ ]:
```
print(len(held5ths[(held5ths.Brightness > 0.6508051)]))
print(len(held5ths[(held5ths.Brightness < 0.6508051)]))
```

```
70
70
```

Randomness testing

[ ]:
```
np.savetxt(r'every5thbinarypulsar6.txt', held5ths.Binary, fmt='%d',␣
↪delimiter='')
np.savetxt(r'allpulsar6.txt', pulsar.Binary, fmt='%d', delimiter='')
```

[ ]:
```
pulsar.Binary
```

[ ]:
```
0      0
1      1
2      1
3      1
4      1
      ..
693    1
694    0
```

```
695    0
696    0
697    0
Name: Binary, Length: 698, dtype: int32
```