In [18]:
```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dr
from tensorflow.keras.optimizers import Adam
import pandas as pd
from collections import Counter
import matplotlib.pyplot as plt
```

In [10]:
```
x_train=pd.read_csv("Train_raw.csv")
y_train=x_train['Annot']
col=[str(i) for i in range (452)]
x_train.drop(['Unnamed: 0','Annot'],axis=1,inplace=True)
x_train=x_train[col]
x_train.head()
```

Out[10]:

	0	1	2	3	4	5	6	7	
0	0.314361	0.293304	0.294979	0.291153	0.284532	0.298951	0.287277	0.279160	0.250
1	0.392793	0.353635	0.365580	0.340462	0.351937	0.338264	0.326857	0.359382	0.315
2	0.309705	0.279425	0.282787	0.286763	0.288553	0.289518	0.292288	0.295409	0.298
3	0.980623	1.000000	0.994847	0.972452	0.949903	0.923482	0.887137	0.854327	0.827
4	0.293045	0.318835	0.330447	0.321225	0.291032	0.254603	0.242335	0.243298	0.244

5 rows × 452 columns

In [11]:
```
x_val=pd.read_csv("Test_raw.csv")
y_val=x_val['Annot']
x_val.drop(['Unnamed: 0','Annot'],axis=1,inplace=True)
x_val.head()
```

Out[11]:

	0	1	2	3	4	5	6	7	
0	0.935732	0.910727	0.879865	0.850602	0.811706	0.790849	0.765112	0.751193	0.738
1	0.311535	0.324072	0.299686	0.304333	0.305983	0.307973	0.307386	0.304360	0.298
2	0.348916	0.318369	0.317779	0.316920	0.316518	0.317300	0.319241	0.322825	0.325
3	0.810451	0.804310	0.797393	0.788679	0.779847	0.773938	0.771590	0.774436	0.778
4	0.059137	0.076650	0.112803	0.150386	0.211558	0.217220	0.203146	0.161792	0.157

5 rows × 452 columns

In [12]:
```
Counter(y_train),Counter(y_val)
```

Out[12]:
```
(Counter({1: 36548, 0: 12922}), Counter({1: 36548, 0: 12925}))
```

CNN

In [13]:
```
# Build a more complex model
model = Sequential()
model.add(Conv1D(32, kernel_size=5, activation='relu', input_shape=(x_train.
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(64, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss='binary_crossentropy

# Training loop
epochs = 20
batch_size = 32

history= model.fit(x_train, y_train, epochs=20, batch_size=batch_size,valida
```

```
Epoch 1/20
1546/1546 [==============================] - 48s 30ms/step - loss: 0.3878 -
accuracy: 0.8348 - val_loss: 0.2671 - val_accuracy: 0.9017
Epoch 2/20
1546/1546 [==============================] - 48s 31ms/step - loss: 0.2613 -
accuracy: 0.9049 - val_loss: 0.2129 - val_accuracy: 0.9281
Epoch 3/20
1546/1546 [==============================] - 48s 31ms/step - loss: 0.2175 -
accuracy: 0.9247 - val_loss: 0.1755 - val_accuracy: 0.9389
Epoch 4/20
1546/1546 [==============================] - 45s 29ms/step - loss: 0.1906 -
accuracy: 0.9336 - val_loss: 0.1527 - val_accuracy: 0.9451
Epoch 5/20
1546/1546 [==============================] - 48s 31ms/step - loss: 0.1692 -
accuracy: 0.9405 - val_loss: 0.1433 - val_accuracy: 0.9502
Epoch 6/20
1546/1546 [==============================] - 54s 35ms/step - loss: 0.1548 -
accuracy: 0.9463 - val_loss: 0.1399 - val_accuracy: 0.9513
Epoch 7/20
1546/1546 [==============================] - 48s 31ms/step - loss: 0.1414 -
accuracy: 0.9498 - val_loss: 0.1273 - val_accuracy: 0.9569
Epoch 8/20
1546/1546 [==============================] - 47s 30ms/step - loss: 0.1314 -
accuracy: 0.9554 - val_loss: 0.1188 - val_accuracy: 0.9597
Epoch 9/20
1546/1546 [==============================] - 50s 32ms/step - loss: 0.1224 -
```

```
accuracy: 0.9573 - val_loss: 0.1097 - val_accuracy: 0.9624
Epoch 10/20
1546/1546 [==============================] - 54s 35ms/step - loss: 0.1130 -
accuracy: 0.9599 - val_loss: 0.1157 - val_accuracy: 0.9602
Epoch 11/20
1546/1546 [==============================] - 53s 34ms/step - loss: 0.1058 -
accuracy: 0.9624 - val_loss: 0.1082 - val_accuracy: 0.9643
Epoch 12/20
1546/1546 [==============================] - 52s 33ms/step - loss: 0.1010 -
accuracy: 0.9644 - val_loss: 0.1014 - val_accuracy: 0.9658
Epoch 13/20
1546/1546 [==============================] - 49s 31ms/step - loss: 0.0960 -
accuracy: 0.9658 - val_loss: 0.1047 - val_accuracy: 0.9661
Epoch 14/20
1546/1546 [==============================] - 47s 30ms/step - loss: 0.0903 -
accuracy: 0.9681 - val_loss: 0.0998 - val_accuracy: 0.9675
Epoch 15/20
1546/1546 [==============================] - 52s 34ms/step - loss: 0.0883 -
accuracy: 0.9686 - val_loss: 0.0963 - val_accuracy: 0.9676
Epoch 16/20
1546/1546 [==============================] - 50s 33ms/step - loss: 0.0843 -
accuracy: 0.9700 - val_loss: 0.1024 - val_accuracy: 0.9676
Epoch 17/20
1546/1546 [==============================] - 48s 31ms/step - loss: 0.0807 -
accuracy: 0.9714 - val_loss: 0.1045 - val_accuracy: 0.9686
Epoch 18/20
1546/1546 [==============================] - 47s 30ms/step - loss: 0.0765 -
accuracy: 0.9732 - val_loss: 0.0998 - val_accuracy: 0.9700
Epoch 19/20
1546/1546 [==============================] - 49s 32ms/step - loss: 0.0748 -
accuracy: 0.9738 - val_loss: 0.0924 - val_accuracy: 0.9710
Epoch 20/20
1546/1546 [==============================] - 49s 32ms/step - loss: 0.0686 -
accuracy: 0.9753 - val_loss: 0.1049 - val_accuracy: 0.9703
```

In [ ]:

```
```

In [14]:

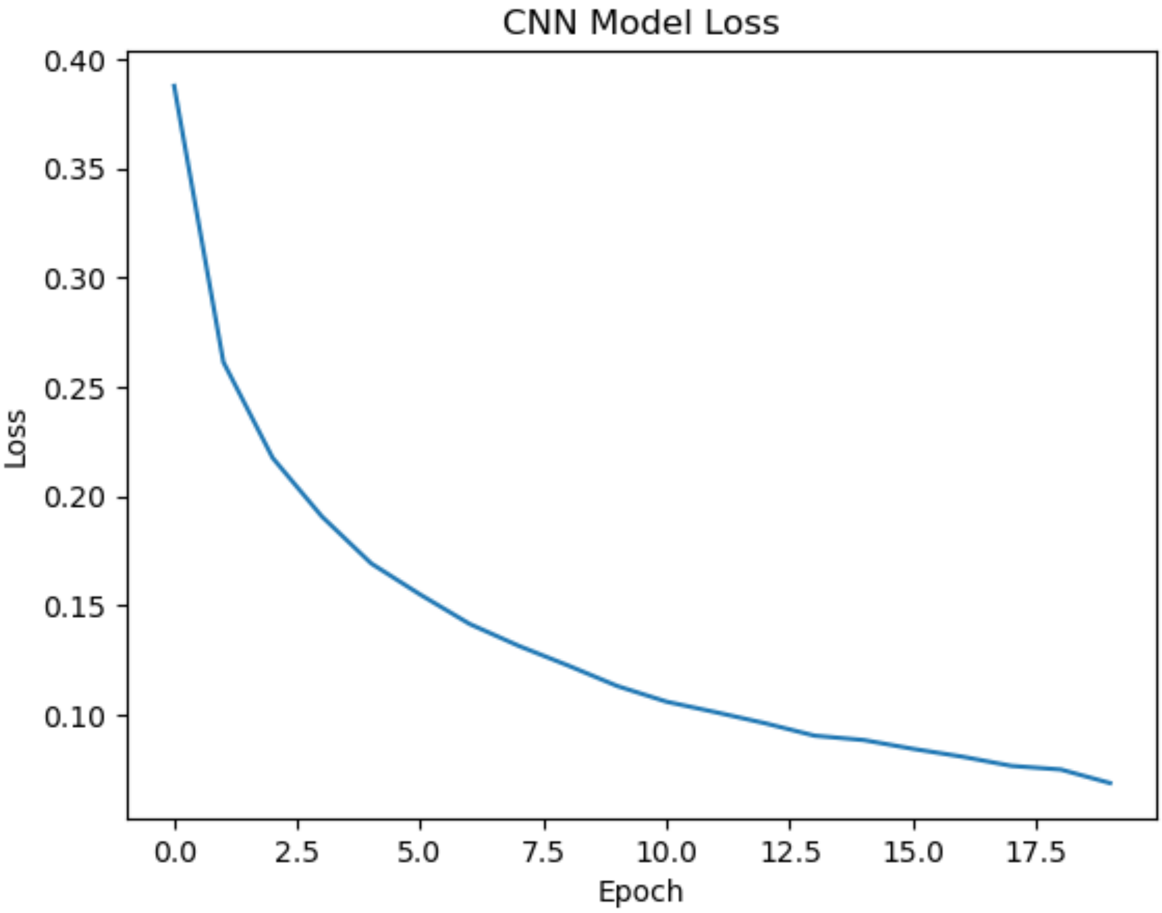
```
# After training, evaluate on the test set
predictions = model.predict(x_val)
predicted_labels = (predictions > 0.5).astype(int)

# Generate the classification report
report = classification_report(y_val, predicted_labels)

# Print the classification report
print(report)
```

```
1547/1547 [==============================] - 16s 10ms/step
              precision    recall  f1-score   support

           0       0.97      0.91      0.94     12925
           1       0.97      0.99      0.98     36548

    accuracy                           0.97     49473
   macro avg       0.97      0.95      0.96     49473
weighted avg       0.97      0.97      0.97     49473
```

In [15]:
```
print("Confusion matrix:",confusion_matrix(y_val, predicted_labels))
```

```
Confusion matrix: [[11806  1119]
 [  352 36196]]
```

In [19]:
```
#history = model.fit(train_dataset, epochs=10, verbose=1)

# Plot the loss history
plt.plot(history.history['loss'])
plt.title('CNN Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.show()
```



DNN

In [20]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, BatchNormalization, Activation, D
from tensorflow.keras.optimizers import Adam
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import make_classification
```

In [21]:
```
# Build the DNN model
model = Sequential()

# Input layer
model.add(Dense(128, input_dim=len(x_train.columns)))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))  # Dropout for regularization

# Hidden layers
model.add(Dense(64))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))

model.add(Dense(32))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))
```

In [22]:
```
# Output layer
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer=Adam(), loss='binary_crossentropy', metrics=['accura

# Train the model
history=  model.fit(x_train, y_train, epochs=20, batch_size=32, validation_d
```

```
Epoch 1/20
1546/1546 [==============================] - 12s 6ms/step - loss: 0.4864 - a
ccuracy: 0.7724 - val_loss: 0.3869 - val_accuracy: 0.8322
Epoch 2/20
1546/1546 [==============================] - 9s 6ms/step - loss: 0.4147 - ac
curacy: 0.8206 - val_loss: 0.3460 - val_accuracy: 0.8577
Epoch 3/20
1546/1546 [==============================] - 9s 6ms/step - loss: 0.3898 - ac
curacy: 0.8349 - val_loss: 0.3239 - val_accuracy: 0.8632
Epoch 4/20
1546/1546 [==============================] - 10s 6ms/step - loss: 0.3733 - a
ccuracy: 0.8472 - val_loss: 0.3011 - val_accuracy: 0.8835
```

```
Epoch 5/20
1546/1546 [==============================] - 7766s 5s/step - loss: 0.3597 -
accuracy: 0.8548 - val_loss: 0.2933 - val_accuracy: 0.8851
Epoch 6/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3519 - a
ccuracy: 0.8586 - val_loss: 0.2797 - val_accuracy: 0.8902
Epoch 7/20
1546/1546 [==============================] - 14s 9ms/step - loss: 0.3438 - a
ccuracy: 0.8638 - val_loss: 0.2758 - val_accuracy: 0.8943
Epoch 8/20
1546/1546 [==============================] - 16s 10ms/step - loss: 0.3381 -
accuracy: 0.8666 - val_loss: 0.2735 - val_accuracy: 0.8973
Epoch 9/20
1546/1546 [==============================] - 13s 9ms/step - loss: 0.3335 - a
ccuracy: 0.8680 - val_loss: 0.2553 - val_accuracy: 0.9063
Epoch 10/20
1546/1546 [==============================] - 14s 9ms/step - loss: 0.3276 - a
ccuracy: 0.8716 - val_loss: 0.2534 - val_accuracy: 0.9047
Epoch 11/20
1546/1546 [==============================] - 15s 9ms/step - loss: 0.3247 - a
ccuracy: 0.8735 - val_loss: 0.2451 - val_accuracy: 0.9090
Epoch 12/20
1546/1546 [==============================] - 15s 9ms/step - loss: 0.3238 - a
ccuracy: 0.8738 - val_loss: 0.2436 - val_accuracy: 0.9107
Epoch 13/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3191 - a
ccuracy: 0.8771 - val_loss: 0.2409 - val_accuracy: 0.9115
Epoch 14/20
1546/1546 [==============================] - 14s 9ms/step - loss: 0.3141 - a
ccuracy: 0.8782 - val_loss: 0.2361 - val_accuracy: 0.9121
Epoch 15/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3112 - a
ccuracy: 0.8797 - val_loss: 0.2416 - val_accuracy: 0.9113
Epoch 16/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3105 - a
ccuracy: 0.8806 - val_loss: 0.2359 - val_accuracy: 0.9114
Epoch 17/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3087 - a
ccuracy: 0.8812 - val_loss: 0.2305 - val_accuracy: 0.9161
Epoch 18/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3073 - a
ccuracy: 0.8803 - val_loss: 0.2373 - val_accuracy: 0.9117
Epoch 19/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3039 - a
ccuracy: 0.8850 - val_loss: 0.2322 - val_accuracy: 0.9136
Epoch 20/20
1546/1546 [==============================] - 13s 8ms/step - loss: 0.3047 - a
ccuracy: 0.8819 - val_loss: 0.2231 - val_accuracy: 0.9179
```

In [23]:

```
# Evaluate the model on the test set
loss, accuracy = model.evaluate(x_val, y_val)
print(f'Test Loss: {loss}, Test Accuracy: {accuracy}')
```

```
1547/1547 [==============================] - 4s 3ms/step - loss: 0.2231 - ac
curacy: 0.9179
Test Loss: 0.22311055660247803, Test Accuracy: 0.9178541898727417
```

In [24]:
```
pred=model.predict(x_val)
```

```
1547/1547 [==============================] - 4s 3ms/step
```

In [25]:
```
pred=[i[0] for i in pred]
pred
```

Out[25]:
```
[0.18419294,
 0.63367534,
 0.6376862,
 0.8710997,
 0.9443279,
 0.5186675,
 0.4370052,
 0.20339075,
 0.2333843,
 0.29097262,
 0.6383033,
 0.18760417,
 0.80887693,
 0.35308132,
 0.9579945,
 0.29100126,
 0.7805227,
 0.2778485,
 0.9250675,
 0.94407016,
 0.18906356,
 0.45342273,
 0.7107567,
 0.20286085,
 0.94365656,
 0.42687917,
 0.88157165,
 0.07497487,
 0.9376824,
 0.49885288,
 0.35429376,
 0.9559258,
 0.59833777,
 0.33398226,
 0.8862671,
 0.89003086,
 0.93246496,
 0.16317007,
 0.92434704,
 0.6569379,
 0.13350096,
 0.88932896,
 0.8286073,
```

```
0.6259335,
0.95389706,
0.70134705,
0.67413044,
0.8112933,
0.2476563,
0.25129896,
0.42510715,
0.23813501,
0.8703906,
0.8434897,
0.30533248,
0.27060834,
0.73803794,
0.12394169,
0.39604345,
0.41094187,
0.9444314,
0.60189265,
0.15935038,
0.9036491,
0.8491526,
0.6699317,
0.8863214,
0.7002567,
0.88570446,
0.34484723,
0.2016094,
0.90443784,
0.09876853,
0.63045555,
0.28145492,
0.58561546,
0.31467494,
0.9348146,
0.97400075,
0.85901433,
0.9891578,
0.19852564,
0.82281905,
0.29392612,
0.6513334,
0.31110966,
0.9241999,
0.79890376,
0.5274657,
0.72274923,
0.3817028,
0.07121723,
0.95068014,
0.94987553,
0.2733199,
0.73057276,
```

```
0.050888143,
0.78482586,
0.8865531,
0.16226253,
0.67093974,
0.6445756,
0.23602977,
0.45524737,
0.93113106,
0.32971302,
0.5061186,
0.7682842,
0.2774845,
0.9569105,
0.108563535,
0.5865989,
0.29643717,
0.62993014,
0.44354752,
0.36359566,
0.1715993,
0.1943442,
0.13133141,
0.22678438,
0.8868302,
0.34378958,
0.25136465,
0.18175164,
0.038401403,
0.6698308,
0.18252689,
0.5334974,
0.6063142,
0.2011903,
0.23579119,
0.93281174,
0.27706736,
0.4228419,
0.74229544,
0.3735876,
0.9237439,
0.9121762,
0.9458447,
0.91632235,
0.89125365,
0.31724718,
0.85638285,
0.014561476,
0.51826674,
0.41756266,
0.8645759,
0.6832843,
0.80797035,
```

```
0.21457992,
0.4742359,
0.9123622,
0.4156323,
0.92813295,
0.037912,
0.86572886,
0.8229165,
0.18906023,
0.9330188,
0.42795533,
0.15631895,
0.5354202,
0.34745598,
0.928108,
0.9658063,
0.40722588,
0.8713349,
0.94339156,
0.41570386,
0.92239463,
0.2339891,
0.21981715,
0.19221863,
0.8428868,
0.16581592,
0.043049756,
0.100862704,
0.47146794,
0.20393953,
0.92078614,
0.8598217,
0.35312104,
0.58672166,
0.23249997,
0.19952884,
0.6903219,
0.3413039,
0.8939199,
0.979276,
0.28165326,
0.84536594,
0.81307346,
0.7144936,
0.24901499,
0.73974586,
0.31752488,
0.70687795,
0.5095171,
0.9308458,
0.8895228,
0.42296448,
0.68528634,
```

```
            0.5965315,
            0.18871939,
            0.13432562,
            0.36506933,
            0.23108475,
            0.3072423,
            0.6770255,
            0.7672455,
            0.10789282,
            0.038867343,
            0.7092386,
            0.03698682,
            0.9073733,
            0.5133525,
            0.07728422,
            0.4454375,
            0.37516057,
            0.86173266,
            0.20494623,
            0.35301143,
            0.8098329,
            0.908552,
            0.9031597,
            0.89161766,
            0.8789176,
            0.8903514,
            0.20449813,
            0.0927119,
            0.7152648,
            0.24243943,
            0.12393505,
            0.82864034,
            0.5042024,
            0.12401299,
            0.016999215,
            0.70138913,
            0.053974047,
            0.8693963,
            0.74573034,
            0.73179936,
            0.89112073,
            0.9274257,
            0.9208572,
            0.6355239,
            0.9612547,
            0.15862466,
            0.89599943,
            0.47486925,
            0.6787574,
            0.41993693,
            0.74541247,
            0.41194224,
            0.41723183,
```

```
0.6225416,
0.39574587,
0.47429538,
0.6720782,
0.16860321,
0.55859435,
0.18314624,
0.96335006,
0.23623048,
0.4026494,
0.46784857,
0.18096945,
0.51010776,
0.7973787,
0.70179313,
0.1998611,
0.033942778,
0.86006004,
0.7177898,
0.8754223,
0.8956438,
0.9219211,
0.68753505,
0.8868101,
0.2096684,
0.94326204,
0.8740444,
0.23499021,
0.35148022,
0.16116144,
0.113040164,
0.90818554,
0.7087575,
0.35445386,
0.2879164,
0.7395761,
0.1276561,
0.9320105,
0.95342875,
0.16201662,
0.6082063,
0.93724495,
0.3122123,
0.60907626,
0.02677365,
0.84782475,
0.9616719,
0.9854899,
0.22463593,
0.7602758,
0.9600597,
0.9553769,
0.28857875,
```

```
0.74040085,
0.22822729,
0.84547466,
0.70777416,
0.42561316,
0.024051998,
0.55414486,
0.6738975,
0.33133015,
0.9408959,
0.4384365,
0.24255018,
0.16981742,
0.31762555,
0.91911066,
0.9173226,
0.87405527,
0.5562393,
0.5335782,
0.5024226,
0.8772542,
0.9252908,
0.10889319,
0.4269773,
0.21402253,
0.7019941,
0.54457515,
0.68295246,
0.9018261,
0.8985227,
0.3532138,
0.81262785,
0.9368702,
0.2500687,
0.8995721,
0.32372662,
0.86407155,
0.8215981,
0.9021505,
0.27903464,
0.89003795,
0.19108956,
0.61078036,
0.87221617,
0.9375698,
0.23074618,
0.44801828,
0.21199341,
0.18872654,
0.505618,
0.32613614,
0.31712615,
0.6469603,
```

```
0.4233555,
0.2679255,
0.9850394,
0.8378536,
0.9014365,
0.0075694853,
0.06615484,
0.2109477,
0.25270438,
0.81182736,
0.32505155,
0.3118746,
0.020959705,
0.06608531,
0.043991297,
0.61199427,
0.25947577,
0.84273034,
0.8562208,
0.15286236,
0.23948745,
0.7840636,
0.43718427,
0.38432926,
0.6761616,
0.16387172,
0.13932903,
0.44742873,
0.72605604,
0.21410756,
0.4961795,
0.2170792,
0.30648118,
0.9295411,
0.6183671,
0.9290957,
0.16860586,
0.7224432,
0.8828057,
0.89986837,
0.79111165,
0.14295447,
0.9165964,
0.8222693,
0.8728764,
0.18034002,
0.34291273,
0.35257107,
0.08104077,
0.811089,
0.8880052,
0.40482327,
0.07315669,
```

```
0.95502025,
0.764807,
0.22981083,
0.6773001,
0.19256374,
0.6117431,
0.7025206,
0.7745952,
0.61756456,
0.92427623,
0.6739614,
0.9340723,
0.49353248,
0.8543007,
0.5660546,
0.899662,
0.6533815,
0.91564393,
0.8704635,
0.8197821,
0.5882469,
0.51902044,
0.10289539,
0.9181969,
0.915226,
0.33029136,
0.98865676,
0.7580314,
0.5677536,
0.4078434,
0.85706586,
0.028001957,
0.22759983,
0.54872495,
0.13778931,
0.6302744,
0.23010421,
0.16069682,
0.59951687,
0.8427563,
0.951277,
0.9507094,
0.20846975,
0.27505484,
0.64494264,
0.20207638,
0.75353616,
0.19646382,
0.23873027,
0.8954309,
0.5383171,
0.27240458,
0.361563,
```

```
0.8510124,
0.8526662,
0.5047241,
0.89328486,
0.6388107,
0.2069389,
0.92721707,
0.83791155,
0.5376575,
0.032163095,
0.111334085,
0.8005527,
0.24577035,
0.8369072,
0.35148513,
0.19845913,
0.35328123,
0.6987232,
0.23887493,
0.17138182,
0.6567702,
0.8477984,
0.5207947,
0.5485604,
0.38043162,
0.74429387,
0.18790963,
0.89345765,
0.52544945,
0.3171719,
0.025208687,
0.8071378,
0.89630777,
0.21206,
0.9309223,
0.39293805,
0.40380475,
0.3995134,
0.3428519,
0.59673136,
0.9077165,
0.5432631,
0.86710465,
0.9439879,
0.41794455,
0.8331584,
0.24965028,
0.67655,
0.85216767,
0.5823092,
0.7220168,
0.74847454,
0.49138027,
```

```
0.47493592,
0.89879644,
0.68247575,
0.78407633,
0.5164542,
0.513588,
0.58972347,
0.9348177,
0.9144131,
0.44920403,
0.7193228,
0.44981894,
0.98477626,
0.2923789,
0.6696693,
0.19522499,
0.7188965,
0.17339535,
0.22986606,
0.01724276,
0.92513984,
0.8934821,
0.4991613,
0.49052155,
0.39859095,
0.5920055,
0.29262903,
0.89934057,
0.87101674,
0.4965965,
0.30097505,
0.93703514,
0.96336263,
0.011108674,
0.8452599,
0.15989196,
0.5285561,
0.098709315,
0.93438727,
0.20734367,
0.8970762,
0.9429,
0.79082066,
0.8597049,
0.94666857,
0.5569404,
0.83397,
0.568875,
0.25857377,
0.8756004,
0.43386844,
0.9481339,
0.6896956,
```

```
0.23278388,
0.68222594,
0.46873885,
0.37814924,
0.37253612,
0.11232947,
0.72935694,
0.8982686,
0.9210848,
0.24910726,
0.73585355,
0.86868024,
0.7059185,
0.5692367,
0.28575152,
0.9255026,
0.448912,
0.4702511,
0.98720133,
0.97437084,
0.81556845,
0.8593001,
0.5517212,
0.27950695,
0.2975709,
0.1377862,
0.7246163,
0.8727577,
0.047007546,
0.11272893,
0.92724293,
0.17192474,
0.14629208,
0.7495191,
0.53437686,
0.16065823,
0.033919577,
0.8133708,
0.027153388,
0.80974144,
0.78510773,
0.22696713,
0.46331537,
0.04771744,
0.8195394,
0.925182,
0.060723517,
0.7986343,
0.042328775,
0.7378543,
0.6900731,
0.9143704,
0.9345981,
```

```
0.9228371,
0.19556649,
0.9739136,
0.044151284,
0.6307028,
0.17239413,
0.81055343,
0.5169437,
0.025911324,
0.18054245,
0.164718,
0.9205804,
0.31015173,
0.91602653,
0.01134838,
0.8447435,
0.562983,
0.91795474,
0.511537,
0.19780889,
0.6763593,
0.6945847,
0.37078467,
0.32535395,
0.8623347,
0.8064387,
0.28249818,
0.92400986,
0.92277485,
0.7506895,
0.9083525,
0.9012972,
0.40586746,
0.20413391,
0.80263895,
0.3174946,
0.25710478,
0.90886825,
0.93949425,
0.87791145,
0.01930515,
0.83248883,
0.1973087,
0.3022428,
0.20060259,
0.29530504,
0.1646627,
0.2941904,
0.029859664,
0.15451041,
0.76301986,
0.8536704,
0.37772226,
```

```
0.07831995,
0.1571301,
0.8414655,
0.12713501,
0.3692018,
0.74806553,
0.6058888,
0.77312475,
0.88071,
0.6038197,
0.46773475,
0.9193907,
0.9371002,
0.30739316,
0.91190547,
0.2904308,
0.19558899,
0.0282529,
0.7190334,
0.8045084,
0.9099645,
0.8731137,
0.651478,
0.9314663,
0.7127678,
0.33715132,
0.9157908,
0.9449282,
0.9693788,
0.8258523,
0.90036535,
0.1937194,
0.44107488,
0.9127428,
0.2736368,
0.88918394,
0.19300678,
0.40328088,
0.82990116,
0.25526407,
0.64343226,
0.32414842,
0.4666978,
0.5571046,
0.79130316,
0.918161,
0.71182156,
0.11539808,
0.1295058,
0.66840714,
0.13099957,
0.70366806,
0.41223684,
```

```
0.9013214,
0.80620956,
0.7371623,
0.96179783,
0.0073906695,
0.02495693,
0.94850945,
0.5654813,
0.03966768,
0.52776635,
0.93364465,
0.9570618,
0.9007238,
0.90782255,
0.8094947,
0.6750968,
0.8827372,
0.57038105,
0.93449706,
0.3930513,
0.9075495,
0.9157232,
0.82633543,
0.38059765,
0.46075237,
0.4194437,
0.8600811,
0.14344233,
0.15320578,
0.36481032,
0.2261427,
0.98105216,
0.75009483,
0.54939044,
0.011440972,
0.294748,
0.95330083,
0.9221047,
0.7571019,
0.45748913,
0.8238881,
0.74626476,
0.27263978,
0.47391045,
0.8989991,
0.23872076,
0.8537687,
0.4411986,
0.35098374,
0.13219024,
0.97027373,
0.7910736,
0.055697802,
```

```
0.91839033,
0.4554372,
0.21508387,
0.7454623,
0.8731902,
0.6485298,
0.98917514,
0.45306075,
0.14129496,
0.90924656,
0.3722217,
0.90688235,
0.6371941,
0.16477056,
0.29882762,
0.3077086,
0.49403873,
0.3576224,
0.9200327,
0.1947541,
0.21821654,
0.85482675,
0.33194068,
0.83787334,
0.5903706,
0.7235985,
0.18713267,
0.82441884,
0.58623165,
0.7091832,
0.029492337,
0.42534965,
0.9710059,
0.40449753,
0.80024797,
0.34390974,
0.94979924,
0.7435434,
0.47038507,
0.87551826,
0.5210577,
0.39099973,
0.19411084,
0.11700783,
0.46178317,
0.87525755,
0.5278252,
0.5777411,
0.2194617,
0.94065934,
0.79472303,
0.29245153,
0.45513654,
```

```
0.9440804,
0.93494445,
0.8797442,
0.90150964,
0.34753263,
0.5888554,
0.29538733,
0.22320792,
0.9456428,
0.068719104,
0.8929813,
0.80932105,
0.3150151,
0.04134795,
0.22267081,
0.88127995,
0.52181345,
0.47399276,
0.17978868,
0.53456455,
0.7426306,
0.19555824,
0.2582394,
0.3376389,
0.8796281,
0.4854141,
0.95589477,
0.51577985,
0.80735815,
0.62284726,
0.60194176,
0.3580538,
0.24009661,
0.6195014,
0.88832706,
0.07876007,
0.81882745,
0.57861656,
0.79774433,
0.9586063,
0.21105857,
0.16060846,
0.7852418,
0.7304039,
0.80652434,
0.26272082,
0.88839227,
0.2321175,
0.7999449,
0.35038015,
0.22480252,
0.21095711,
0.15344797,
```

```
0.9122226,
0.29393324,
0.31086946,
0.19980761,
0.87168807,
0.8124139,
0.93154335,
0.85062814,
0.4471148,
0.17543523,
0.82400143,
0.48554888,
0.62080455,
0.112126596,
0.12516041,
0.40452114,
0.89579266,
0.95799893,
0.119912006,
0.984425,
0.6022482,
0.9509952,
0.5016754,
0.22048527,
0.19508252,
0.086429544,
0.5443711,
0.49400753,
0.5681968,
0.6207125,
0.9504013,
0.27586496,
0.40194255,
0.9712813,
0.6959516,
0.13353138,
0.8401921,
0.6071392,
0.17386921,
0.9063933,
0.15905057,
0.92879254,
0.86033547,
0.95351815,
0.91513866,
0.74167234,
0.931529,
0.82497656,
0.75811124,
0.9123425,
0.87112826,
0.2812994,
0.933306,
```

```
0.28083652,
0.13455217,
0.23458584,
0.9491965,
0.8776758,
0.47357816,
0.9078905,
0.74711436,
0.2436693,
0.42267993,
0.9069367,
0.9796173,
0.81845134,
0.13771324,
0.30353844,
0.7845466,
0.5700864,
0.6924208,
0.91299593,
0.25130266,
0.31475922,
0.6319702,
0.48968068,
0.24337867,
0.4387546,
0.76217616,
0.5462165,
0.23719262,
0.15433247,
0.30299076,
0.9268261,
0.5827366,
0.8413565,
0.8605111,
0.6875819,
0.6281687,
0.10585568,
0.5516188,
0.41442907,
0.78519225,
0.18502577,
0.97329605,
0.41816407,
0.53323895,
0.93463117,
0.021028906,
0.77076155,
0.6694411,
0.9058134,
0.49766922,
0.6253161,
0.39432675,
0.9265711,
```

```
 0.3700513,
 0.005803528,
 0.28626478,
 ...]
```

In [26]:
```
predicted=[]
for i in range (len(pred)):
    if pred[i]>0.5:
        predicted.append(1)
    else:
        predicted.append(0)
```

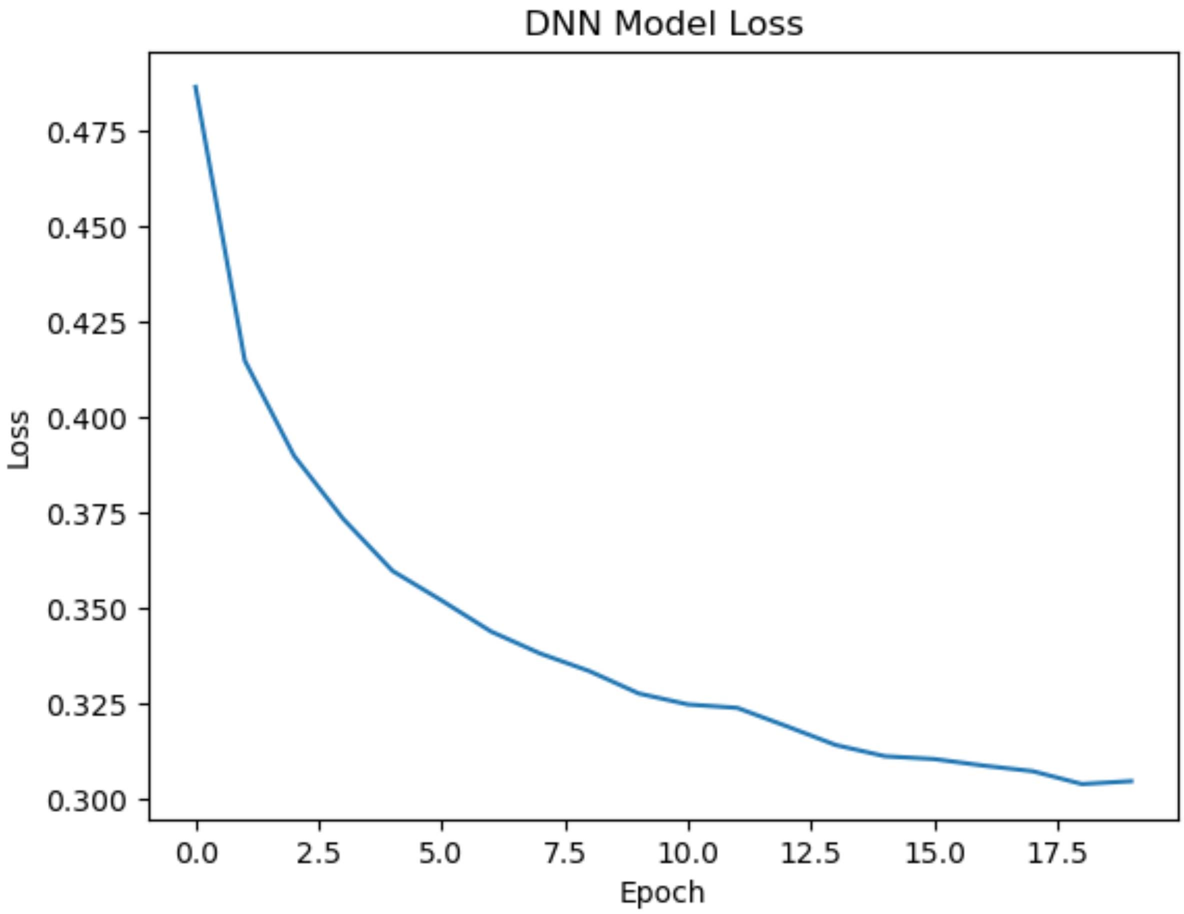
In [27]:
```
print(classification_report(predicted,y_val))
```

```
              precision    recall  f1-score   support

           0       0.77      0.90      0.83     11167
           1       0.97      0.92      0.95     38306

    accuracy                           0.92     49473
   macro avg       0.87      0.91      0.89     49473
weighted avg       0.92      0.92      0.92     49473
```

In [28]:
```
print(confusion_matrix(predicted,y_val))
```

```
[[10014  1153]
 [ 2911 35395]]
```

In [30]:
```
plt.plot(history.history['loss'])
plt.title('DNN Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.show()
```


DNN Model Loss
Loss
0.475
0.450
0.425
0.400
0.375
0.350
0.325
0.300
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch

```
In [ ]:
```