# Track patient recovery in real-time by processing streaming data

**BIOMEDICAL DATA DESIGN**

TA: Haoyin Xu

Group:Zhenyu Xiao
Haobin Zhou
Yimeng Xu
Emma Cardenas

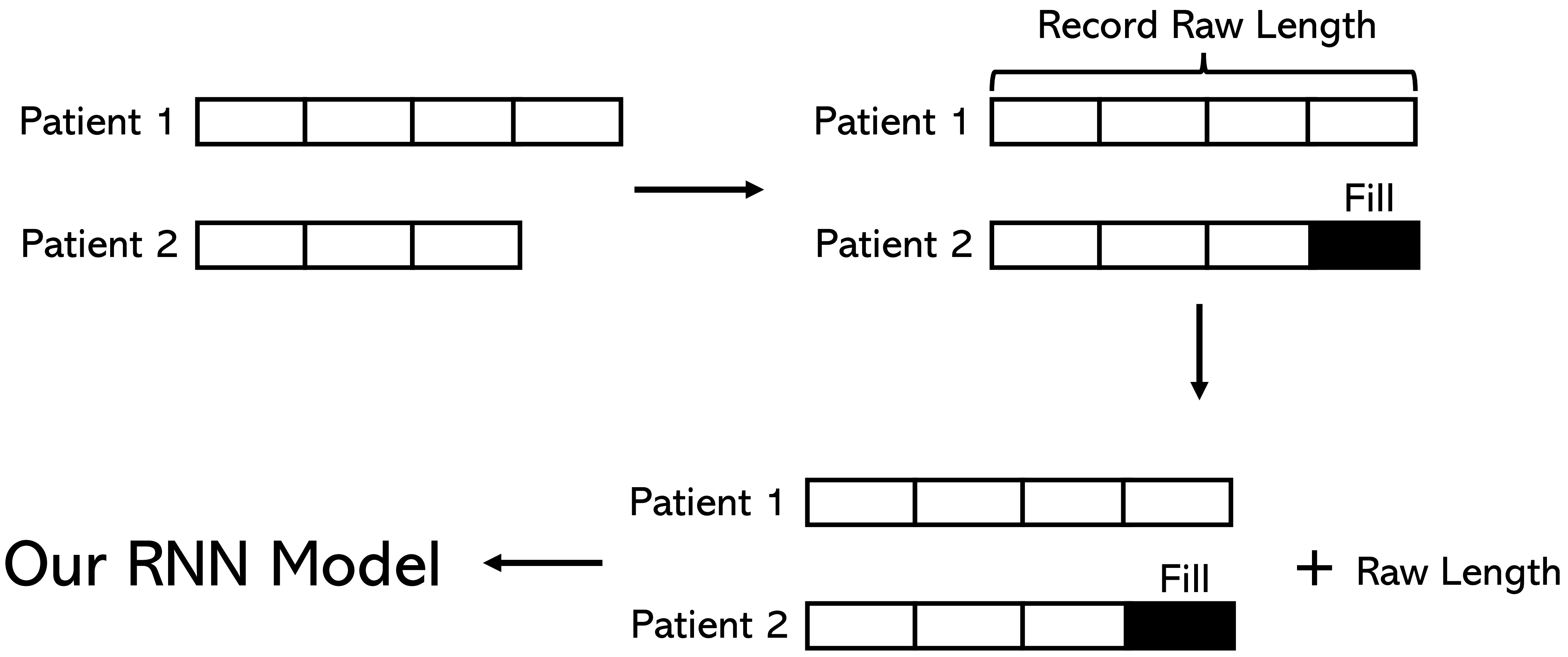
# 01

# ‘Real-time’ Prediction

# 'Real-time' Prediction

## Last Week:

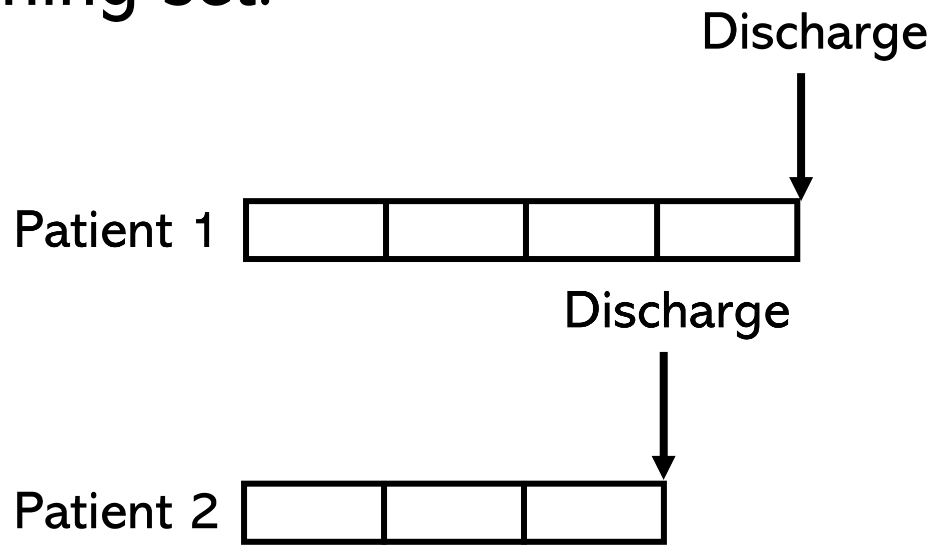
RNN model deal with time series of different lengths



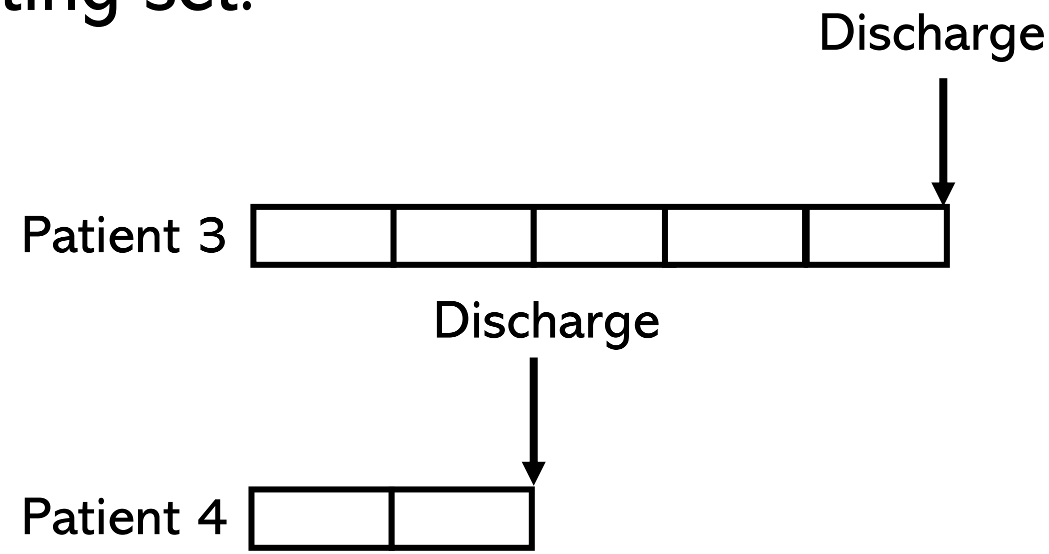
01 ‘Real-time’ Prediction

Last Week:

Training set:




Testing set:



01

# ‘Real-time’ Prediction

Repeat for five times:

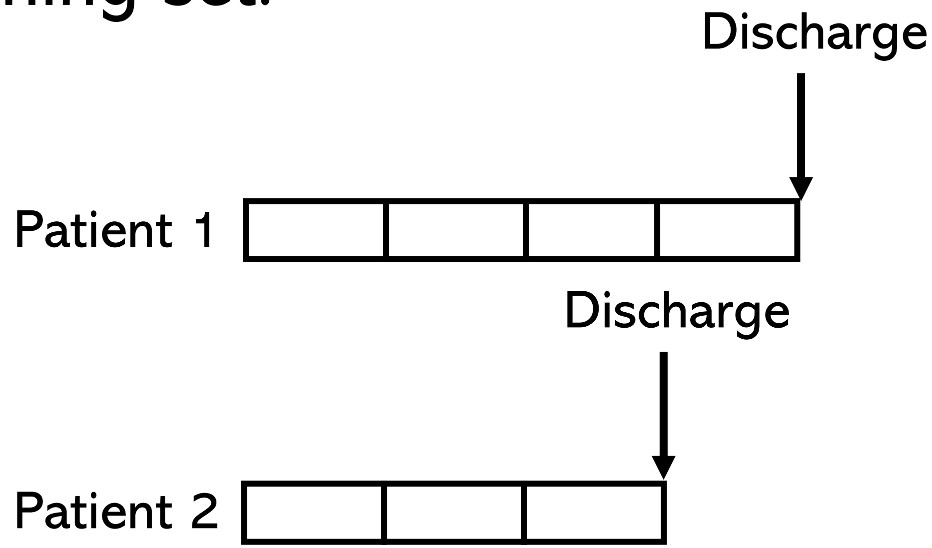
```
Epoch 8/20, Loss: 0.10321918874979019
Epoch 9/20, Loss: 0.009393540211021 9
Epoch 10/20, Loss: 0.07064346969127655
Epoch 11/20, Loss: 0.09824777394533157
Epoch 12/20, Loss: 0.09720230847597122
Epoch 13/20, Loss: 0.023129863664507866
Epoch 14/20, Loss: 0.07097922265529633
Epoch 15/20, Loss: 0.5772714018821716
Epoch 16/20, Loss: 1.3697293996810913
Epoch 17/20, Loss: 0.0018030975479632616
Epoch 18/20, Loss: 0.28388893604278564
Epoch 19/20, Loss: 0.38625386357307434
Epoch 20/20, Loss: 0.011136839166283607
Accuracy: 0.9139784946236559
Precision: 0.8926553672316384
Recall: 0.9239766081871345
...
 [ 13 169]]
Accuracy: [0.9139784946236559, 0.8924731182795699, 0.9274193548387096, 0.8978494623655914, 0.9327956989247311]
avg_Acc:0.9129032258064516
Time taken: 31197.8478 seconds
```

Accuracy: 91.3%

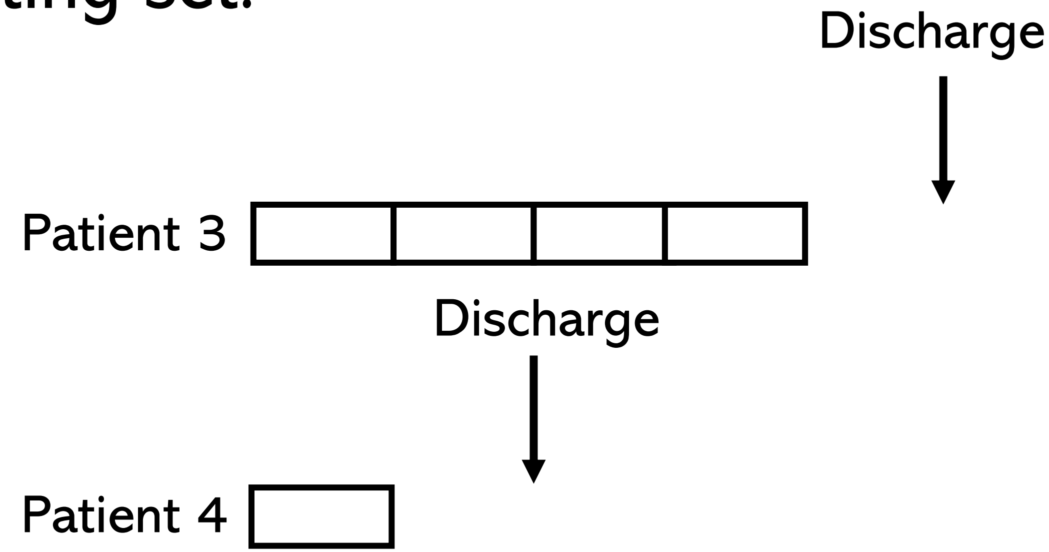
# 01 ‘Real-time’ Prediction

This Week:

Training set:




Testing set:




For the testing part, we don’t give the last hour to predict the final mortality

01

# ‘Real-time’ Prediction

Repeat for four times:

```
Epoch 10/20, Loss: 0.6931630969047546
Epoch 11/20, Loss: 0.3337962329387665
Epoch 12/20, Loss: 0.6931714415550232
Epoch 13/20, Loss: 0.6932192444801331
Epoch 14/20, Loss: 0.3200587034225464
Epoch 15/20, Loss: 0.3239499628543854
Epoch 16/20, Loss: 0.5084646940231323
Epoch 17/20, Loss: 0.6931554675102234
Epoch 18/20, Loss: 0.6961817741394043
Epoch 19/20, Loss: 0.693185031414032
Epoch 20/20, Loss: 0.693149983882904
Accuracy: 0.8579088471849866
Precision: 0.8520710059171598
Recall: 0.8372093023255814
F1 Score: 0.844574780058651
Confusion Matrix:
[[176  25]
 [ 28 144]]
```

```
Epoch 10/20, Loss: 0.3345186114311218
Epoch 11/20, Loss: 0.6931914687156677
Epoch 12/20, Loss: 0.31854328513145447
Epoch 13/20, Loss: 1.0125386714935303
Epoch 14/20, Loss: 0.4884851574897766
Epoch 15/20, Loss: 0.3698061406612396
Epoch 16/20, Loss: 0.8132229447364807
Epoch 17/20, Loss: 0.31506800651550293
Epoch 18/20, Loss: 0.3157444894313812
Epoch 19/20, Loss: 0.695896565914154
Epoch 20/20, Loss: 0.3164122700691223
Accuracy: 0.8679245283018868
Precision: 0.8315217391304348
Recall: 0.8947368421052632
F1 Score: 0.8619718309859156
Confusion Matrix:
[[169  31]
 [ 18 153]]
```

```
Epoch 10/20, Loss: 0.31515413522720337
Epoch 11/20, Loss: 1.2940282821655273
Epoch 12/20, Loss: 0.6931993365287781
Epoch 13/20, Loss: 0.31462329626083374
Epoch 14/20, Loss: 0.3369888663291931
Epoch 15/20, Loss: 0.31822213530540466
Epoch 16/20, Loss: 0.3157409131526947
Epoch 17/20, Loss: 0.6965146064758301
Epoch 18/20, Loss: 0.6935627460479736
Epoch 19/20, Loss: 0.3151668310165405
Epoch 20/20, Loss: 0.6933030486106873
Accuracy: 0.806970509383378
Precision: 0.9243697478991597
Recall: 0.6358381502890174
F1 Score: 0.7534246575342467
Confusion Matrix:
[[191   9]
 [ 63 110]]
```

```
Epoch 10/20, Loss: 0.6934880018234253
Epoch 11/20, Loss: 0.6932013630867004
Epoch 12/20, Loss: 0.6931668519973755
Epoch 13/20, Loss: 0.6941850781440735
Epoch 14/20, Loss: 0.31728488206863403
Epoch 15/20, Loss: 0.6931740045547485
Epoch 16/20, Loss: 0.693154513835907
Epoch 17/20, Loss: 0.6931524872779846
Epoch 18/20, Loss: 1.3090211153030396
Epoch 19/20, Loss: 0.6931576132774353
Epoch 20/20, Loss: 0.3151927590370178
Accuracy: 0.848404255319149
Precision: 0.8155339805825242
Recall: 0.8983957219251337
F1 Score: 0.8549618320610687
Confusion Matrix:
[[151  38]
 [ 19 168]]
```

Accuracy: 84.5%

Recall: 84.3%

A significant drop!!

# 02

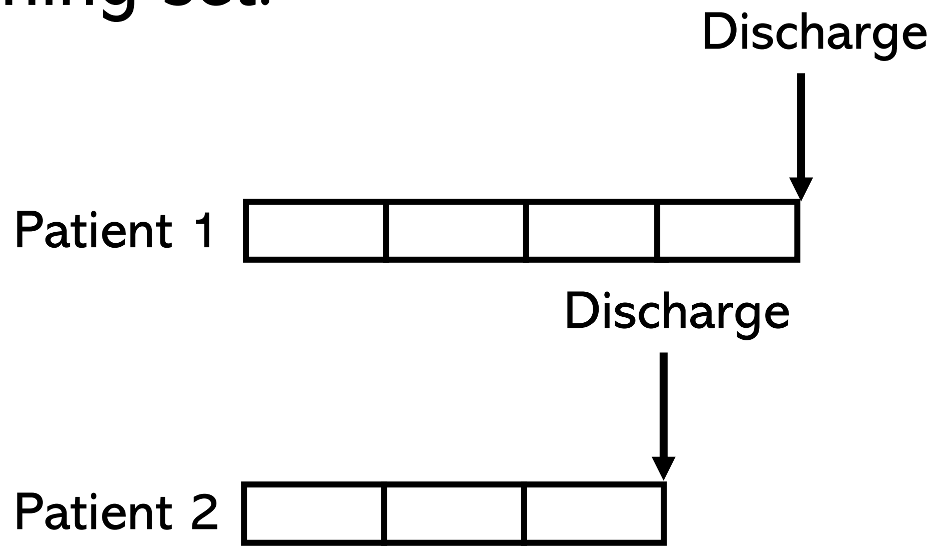
# Next Step

# 02 Change the training set

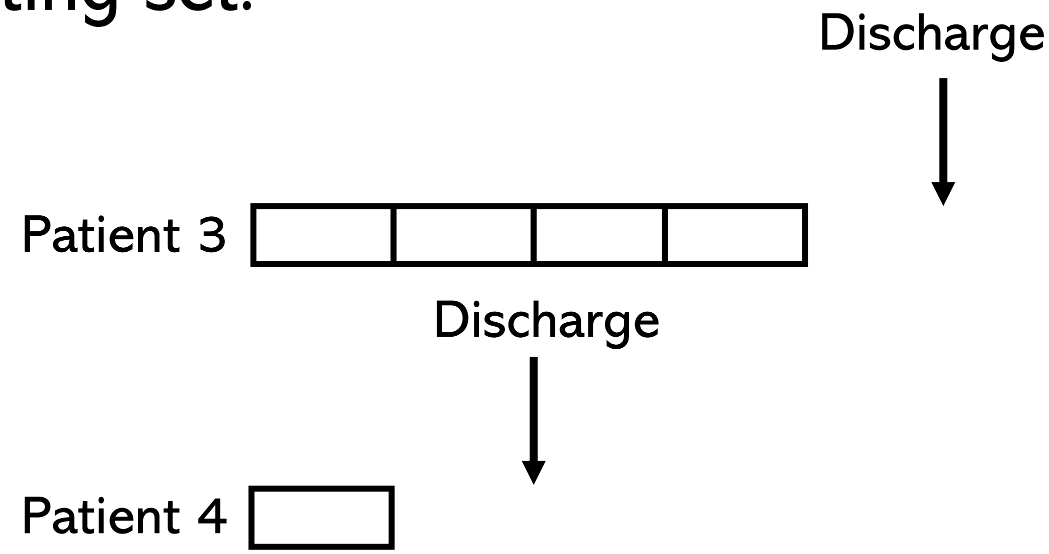
Now:

Changes:

Training set:




Testing set:




For the testing part, we don’t give the last hour to predict the final mortality

02

## Change the parameters

Learning rate

Loss

Drop out

…

# Thank you

# LSTM

Last week: Machine learning

↓

First day’s flattened data

| |
|---|
| Patient1 |
| Patient2 |
| Patient3 |

↓

This week: LSTM

↓

First two day’s flattened data as two time stamps

| Time stamp1 | Time stamp2 |
|---|---|
| Patient1 | Patient1 |
| Patient2 | Patient2 |
| Patient3 | Patient3 |

# 01 LSTM

LSTM: 2 days

```
              precision    recall  f1-score   support

         0.0       0.48      1.00      0.65       154
         1.0       0.00      0.00      0.00       165

    accuracy                           0.48       319
   macro avg       0.24      0.50      0.33       319
weighted avg       0.23      0.48      0.31       319

[[154   0]
 [165   0]]
```

All Zero

Too few time stamps

LSTM:48 hours
(with interpolation)

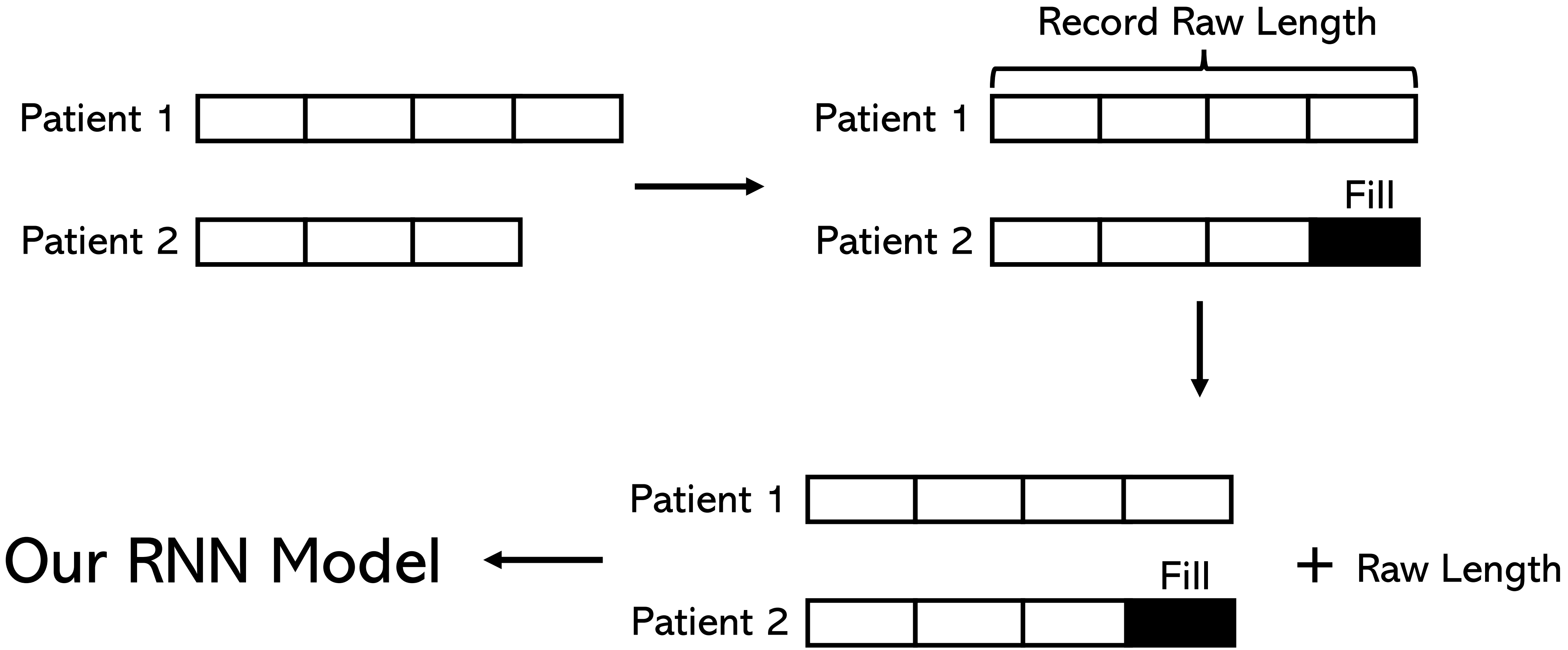
```
...
weighted avg       0.73      0.73      0.73       342

[[119  54]
 [ 39 130]]
```

Add more time stamps

02 **'Real-time' Prediction**

RNN model deal with time series of different lengths



02

# 'Real-time' Prediction

## Fill the series and record the raw length

```python
# load the data
data = pd.read_csv('balanced_LSTM_long.csv')

# Tagged feature columns and label columns
feature_columns = [col for col in data.columns if col not in ('patientunitstayid', 'observationoffset', 'actualicumortality')]
label_column = 'actualicumortality'

# Initialize the list of features and labels
sequences = []
labels = []

# Group and create sequences for each patient
for _, group in data.groupby('patientunitstayid'):
    # For each patient, the features are converted to a tensor
    sequence = torch.tensor(group[feature_columns].values, dtype=torch.float)
    sequences.append(sequence)

    # Assuming that each patient has the same label, so we only take the label of the first record
    labels.append(group[label_column].iloc[0])

lengths = torch.tensor([len(seq) for seq in sequences])
print(lengths.numpy())

# Fill the sequence
padded_sequences = pad_sequence(sequences, batch_first=True)
```

# 02 'Real-time' Prediction

## RNN

### How to use the inputs?

```python
def forward(self, input_seq, input_length):
    #print(input_seq.size())
    batch_size, _, _ = input_seq.size()
    seq_len = int(input_length.item())  # Get the value of the length from the tensor
    #print(seq_len)
    # 初始化隐藏状态
    h0 = torch.zeros(1, batch_size, self.hidden_size)
    c0 = torch.zeros(1, batch_size, self.hidden_size)
    # Initialize the previous day's output
    prev_output = torch.zeros(batch_size, self.output_size)

    # Process the entire sequence recursively
    for t in range(seq_len):
        # Combining the day's input with the previous day's output
        combined_input = torch.cat((input_seq[:, t, :], prev_output), dim=1).unsqueeze(1)
        # Through LSTM
        if t == 0:
            lstm_out, (h, c) = self.lstm(combined_input, (h0, c0))
        else:
            lstm_out, (h, c) = self.lstm(combined_input, (h, c))
        # Use only the last output of the sequence
        final_output = self.fc(lstm_out[:, -1, :])
        # Updating the previous day's output
        prev_output = self.sigmoid(final_output)

    return prev_output
```

### Results

```
Epoch 1/10, Loss: 0.3596692979335785
Epoch 2/10, Loss: 0.7099694013595581
Epoch 3/10, Loss: 0.3183300793170929
Epoch 4/10, Loss: 0.6963921189308167
Epoch 5/10, Loss: 0.35637760162353516
Epoch 6/10, Loss: 0.6995091438293457
Epoch 7/10, Loss: 0.7040725350379944
Epoch 8/10, Loss: 0.6934269070625305
Epoch 9/10, Loss: 0.5024394392967224
Epoch 10/10, Loss: 0.6532086133956909
```

Accuracy~85%

# 02 ‘Real-time’ Prediction

Time-consuming

```
Epoch 17/100, Loss: 0.31905150413513184
Epoch 18/100, Loss: 0.6933485865592957
Epoch 19/100, Loss: 0.6933465003967285
Epoch 20/100, Loss: 0.3626611530780792
Epoch 21/100, Loss: 0.3548220694065094
Epoch 22/100, Loss: 0.6931563019752502
Epoch 23/100, Loss: 0.3166859745979309
Epoch 24/100, Loss: 0.3259027302265167
...
Epoch 71/100, Loss: 0.6931471824645996
Epoch 72/100, Loss: 0.3139694035053253
Epoch 73/100, Loss: 0.3144867420196533
Epoch 74/100, Loss: 0.31408748030662537
```

Whole night but failed!!

Check if there are mistakes

## 02 'Real-time' Prediction

Another easier method:

Use the last day of patients to train

↓

Use each single day's data as input to predict

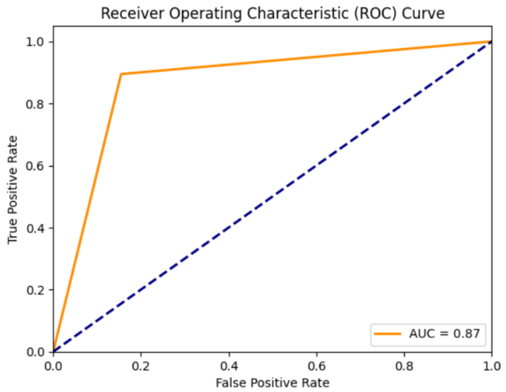
We assume that different days' data are independent

# 01 ‘Real-time’ Prediction

Repeat for four times:

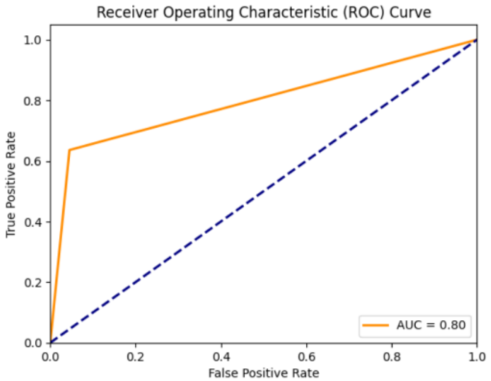
```
Epoch 10/20, Loss: 0.6931630969047546
Epoch 11/20, Loss: 0.3337962329387665
Epoch 12/20, Loss: 0.6931714415550232
Epoch 13/20, Loss: 0.6932192444801331
Epoch 14/20, Loss: 0.3200587034225464
Epoch 15/20, Loss: 0.3239499628543854
Epoch 16/20, Loss: 0.5084646940231323
Epoch 17/20, Loss: 0.6931554675102234
Epoch 18/20, Loss: 0.6961817741394043
Epoch 19/20, Loss: 0.693185031414032
Epoch 20/20, Loss: 0.693149983882904
Accuracy: 0.8579088471849866
Precision: 0.8520710059171598
Recall: 0.8372093023255814
F1 Score: 0.844574780058651
Confusion Matrix:
[[176  25]
 [ 28 144]]
```

```
Epoch 10/20, Loss: 0.3345186114311218
Epoch 11/20, Loss: 0.6931914687156677
Epoch 12/20, Loss: 0.31854328513145447
```



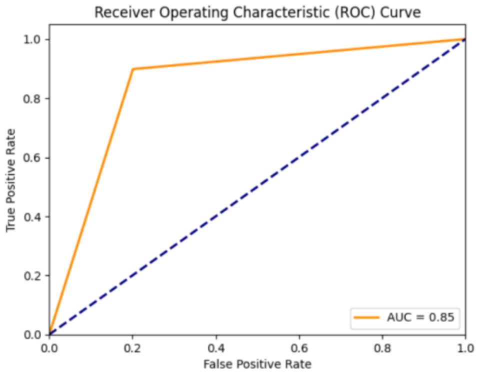

Accuracy: 84.0%

Recall: 85.2%




```
Recall: 0.6358381502890174
F1 Score: 0.7534246575342467
Confusion Matrix:
[[191   9]
 [ 63 110]]
```

```
Epoch 10/20, Loss: 0.6934880018234253
Epoch 11/20, Loss: 0.6932013630867004
Epoch 12/20, Loss: 0.6931668519973755
Epoch 13/20, Loss: 0.6941850781440735
Epoch 14/20, Loss: 0.31728488206863403
Epoch 15/20, Loss: 0.6931740045547485
Epoch 16/20, Loss: 0.69315451
Epoch 17/20, Loss: 0.69315248
Epoch 18/20, Loss: 1.30902111
Epoch 19/20, Loss: 0.69315761
Epoch 20/20, Loss: 0.31519275
Accuracy: 0.848404255319149
Precision: 0.8155339805825242
Recall: 0.8983957219251337
F1 Score: 0.8549618320610687
Confusion Matrix:
[[151  38]
 [ 19 168]]
```




A significant drop!!

03

# Next step

```
Epoch 1/10, Loss: 0.5323466062545776
Epoch 2/10, Loss: 0.7005636096000671
Epoch 3/10, Loss: 0.34396567940711975
Epoch 4/10, Loss: 0.321659862995147
Epoch 5/10, Loss: 0.37011444568634033
Epoch 6/10, Loss: 1.1123342514038086
Epoch 7/10, Loss: 0.6933222413063049
Epoch 8/10, Loss: 0.6972675323486328
Epoch 9/10, Loss: 0.7365317940711975
Epoch 10/10, Loss: 0.6936031579971313
Accuracy: 0.8203753351206434
Precision: 0.7586206896551724
Recall: 0.8953488372093024
F1 Score: 0.8213333333333334
Confusion Matrix:
[[152  49]
 [ 18 154]]
```

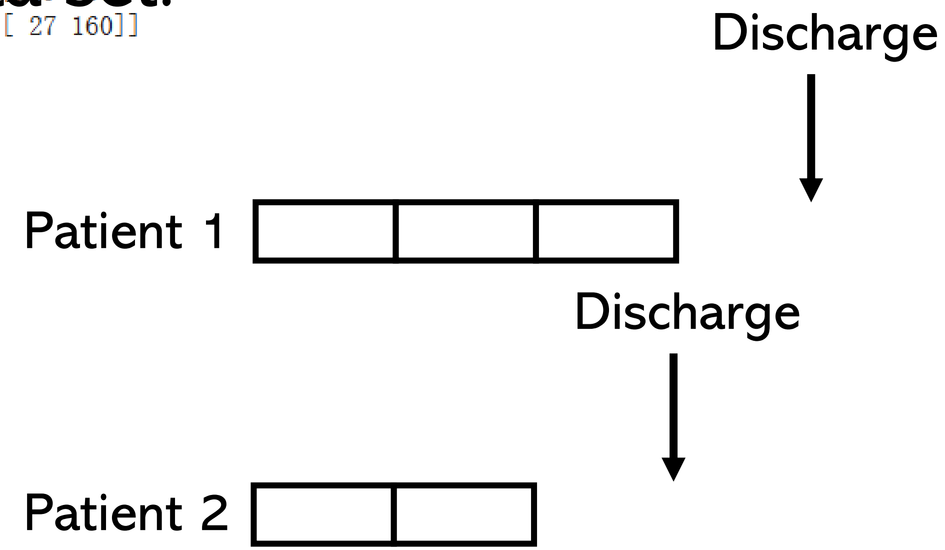
```
Epoch 1/10, Loss: 0.9933973550796509
Epoch 2/10, Loss: 0.717208981513977
Epoch 3/10, Loss: 0.3334490954875946
Epoch 4/10, Loss: 0.5397564768791199
Epoch 5/10, Loss: 0.5919001698493958
Epoch 6/10, Loss: 0.6943417191505432
Epoch 7/10, Loss: 0.6934819221496582
Epoch 8/10, Loss: 0.31822171807289124
Epoch 9/10, Loss: 0.6941342949867249
Epoch 10/10, Loss: 0.31616759300231934
Accuracy: 0.8471849865951743
Precision: 0.8333333333333334
Recall: 0.838150289017341
F1 Score: 0.8357348703170029
Confusion Matrix:
[[171  29]
 [ 28 145]]
```

```
Epoch 1/10, Loss: 0.7011728882789612
Epoch 2/10, Loss: 0.710917592048645
Epoch 3/10, Loss: 0.3280715346336365
Epoch 4/10, Loss: 0.691260457038794
Epoch 5/10, Loss: 0.6948522925376892
Epoch 6/10, Loss: 1.030002236366272
Epoch 7/10, Loss: 0.6941815614700317
Epoch 8/10, Loss: 0.32503750920295715
Epoch 9/10, Loss: 0.693243145942688
Epoch 10/10, Loss: 0.6932666301727295
Accuracy: 0.848404255319149
Precision: 0.8421052631578947
Recall: 0.8556149732620321
F1 Score: 0.8488063660477454
Confusion Matrix:
[[159  30]
 [ 27 160]]
```

```
Epoch 1/10, Loss: 0.6513708829879761
Epoch 2/10, Loss: 0.6125394105911255
Epoch 3/10, Loss: 0.7148727178573608
Epoch 4/10, Loss: 0.7017617225646973
Epoch 5/10, Loss: 0.33441460132598877
Epoch 6/10, Loss: 0.3995670676231384
Epoch 7/10, Loss: 0.6942034959793091
Epoch 8/10, Loss: 0.3185052275657654
Epoch 9/10, Loss: 0.6968369483947754
Epoch 10/10, Loss: 0.321229487657547
Accuracy: 0.7789757412398922
Precision: 0.6943231441048034
Recall: 0.9298245614035088
F1 Score: 0.7949999999999999
Confusion Matrix:
[[130  70]
 [ 12 159]]
```

```
Epoch 1/10, Loss: 0.7399792075157166
Epoch 2/10, Loss: 0.8454279899597168
Epoch 3/10, Loss: 0.33754962682724
Epoch 4/10, Loss: 0.697210967540741
Epoch 5/10, Loss: 0.3270319104194641
Epoch 6/10, Loss: 0.32441461086273193
Epoch 7/10, Loss: 0.5552095174789429
Epoch 8/10, Loss: 0.32459592819213867
Epoch 9/10, Loss: 1.3068009614944458
Epoch 10/10, Loss: 0.32740145921707153
Accuracy: 0.8191489361702128
Precision: 0.7430167597765364
Recall: 0.8580645161290322
F1 Score: 0.7964071856287425
Confusion Matrix:
[[175  46]
 [ 22 133]]
```

## Data set:



# 03 Next step

Data set:

Discharged

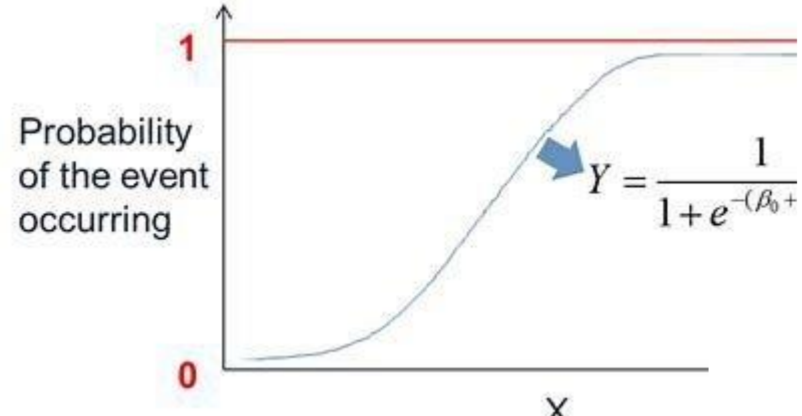
Record Raw Length

Patient 1

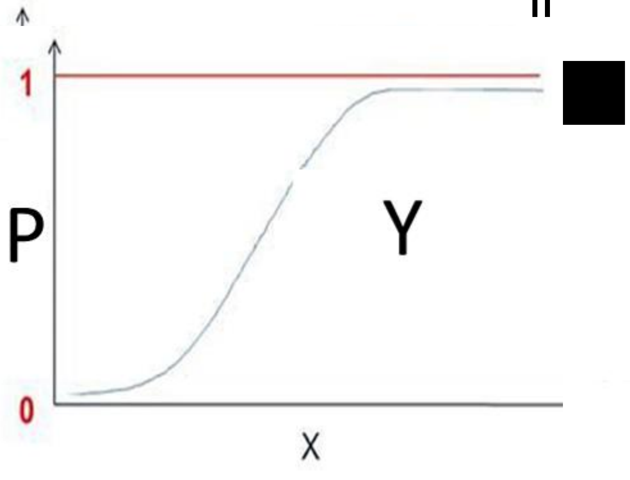
Discharged

Fill

Patient 2




$$Y = \frac{1}{1 + e^{-(\beta_0 + \beta_1 * x)}}$$




Y