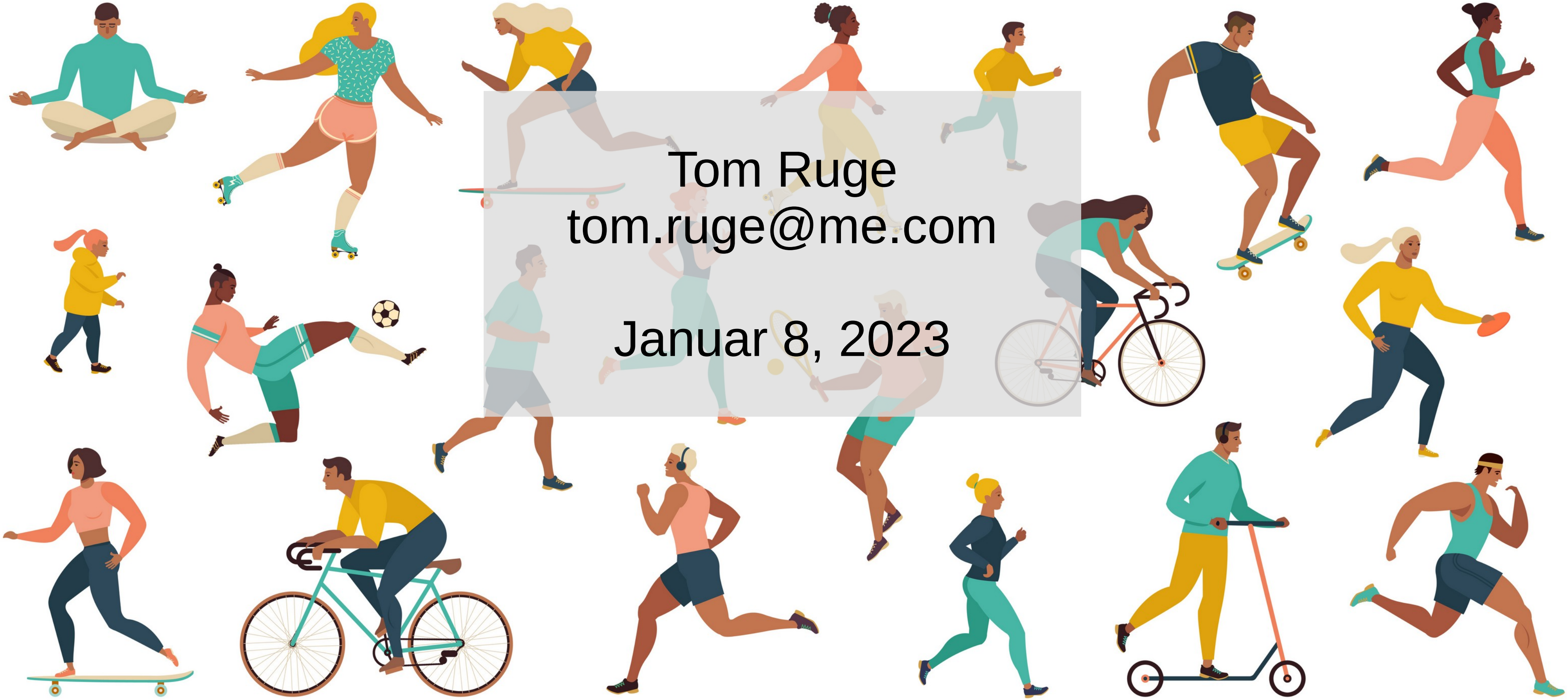
Physical Activity Classification



Tom Ruge
tom.ruge@me.com

Januar 8, 2023

Why and where?

kaggle

Literature

Webpage:

- Dataset with 6 different activities
- Accelerometer and gyroscopic data from smart phone available
- With simple 2 Layer neural network 89% Accuracy could be archieved
- With LSTM a acccuracy of 91 could be reached.

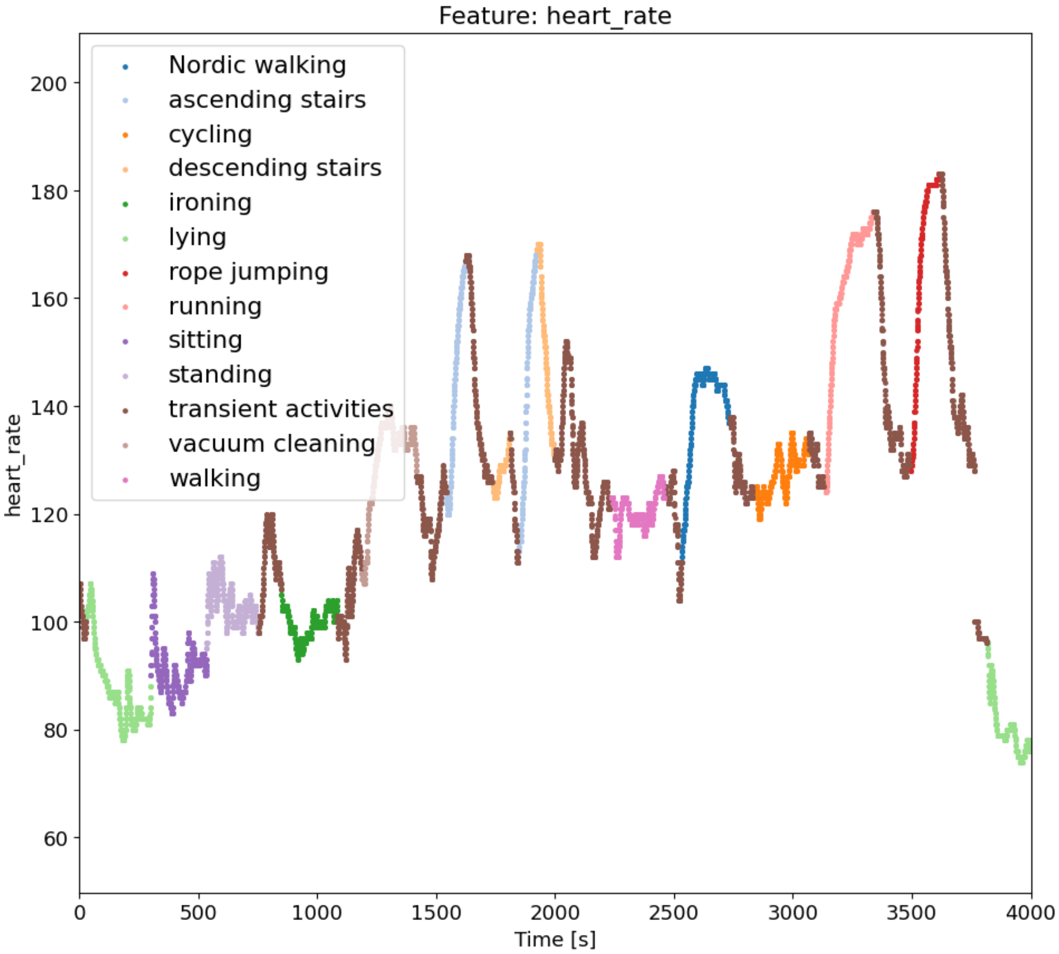
Kaggle Notebook:

- Same Dataset
- Achieved a 99% Accuracy but testing is not realistic
- All other Kaggle notebooks had the same flaw.

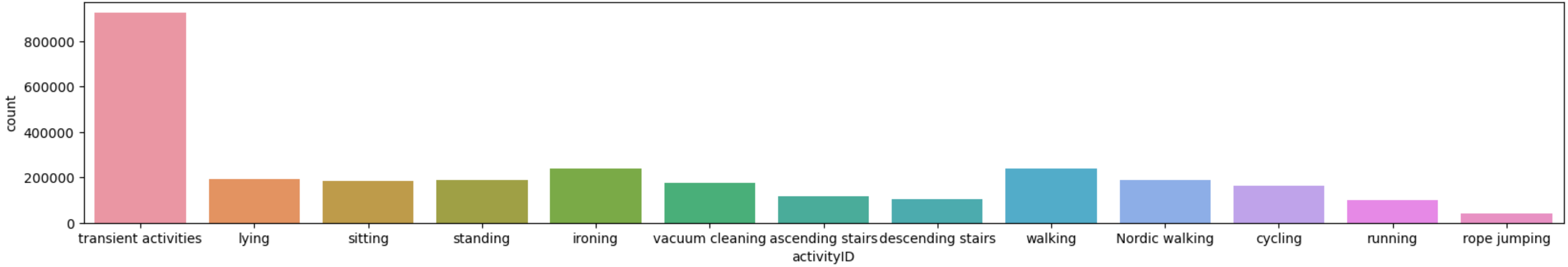
Dataset Characteristics

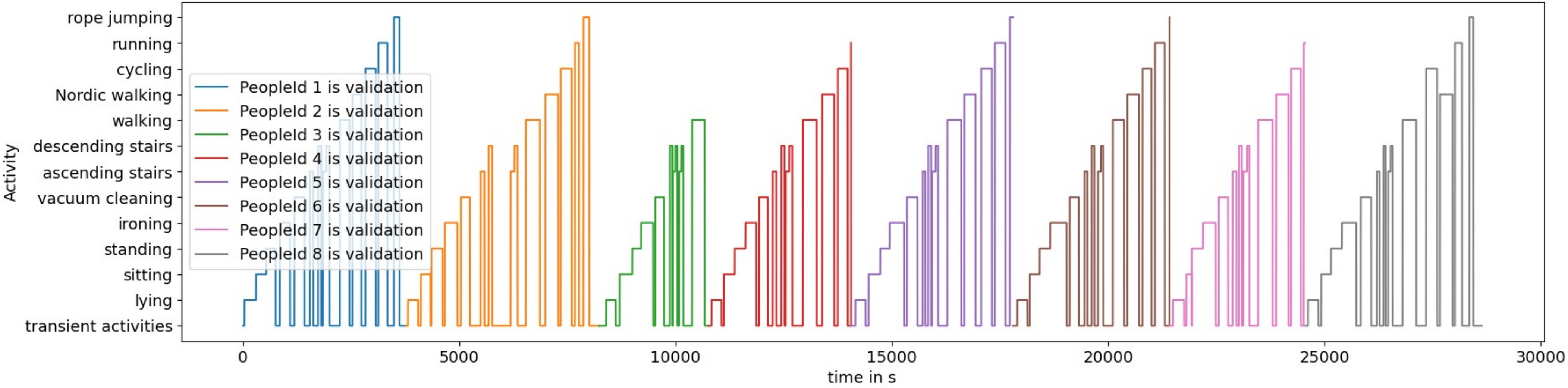
Dataset:

- From Kaggle
- 31 Features(heartbeat, accelerometer, gyroscope, …)
- 13 labels(walking, lying, transient activities, etc …)
- Data measured by 8 different people
- Around 2 million rows



Biases:


count
800000
600000
400000
200000
0
transient activities
lying
sitting
standing
ironing
vacuum cleaning
ascending stairs
descending stairs
walking
Nordic walking
cycling
running
rope jumping
activityID


Activity
rope jumping
running
cycling
Nordic walking
walking
descending stairs
ascending stairs
vacuum cleaning
ironing
standing
sitting
lying
transient activities
PeopleId 1 is validation
PeopleId 2 is validation
PeopleId 3 is validation
PeopleId 4 is validation
PeopleId 5 is validation
PeopleId 6 is validation
PeopleId 7 is validation
PeopleId 8 is validation
0
5000
10000
15000
20000
25000
30000
time in s

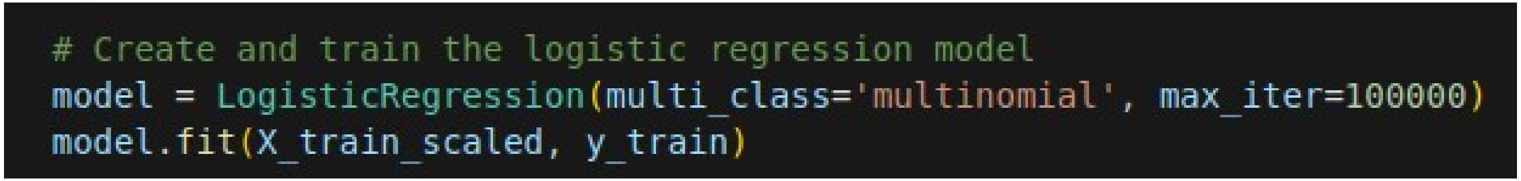
Baseline Model: Data Prep. and Evaluation

- Data undersampled, so that the activities are balanced.

- 1 person test and 7 person train. Iterate over all people

- Because of undersample, accuracys may vary → do fitting 5 times and take average accuracy

Baseline Model: Results



```
# Create and train the logistic regression model
model = LogisticRegression(multi_class='multinomial', max_iter=100000)
model.fit(X_train_scaled, y_train)
```

Average accuracy of 47%

standart deviation basically 0

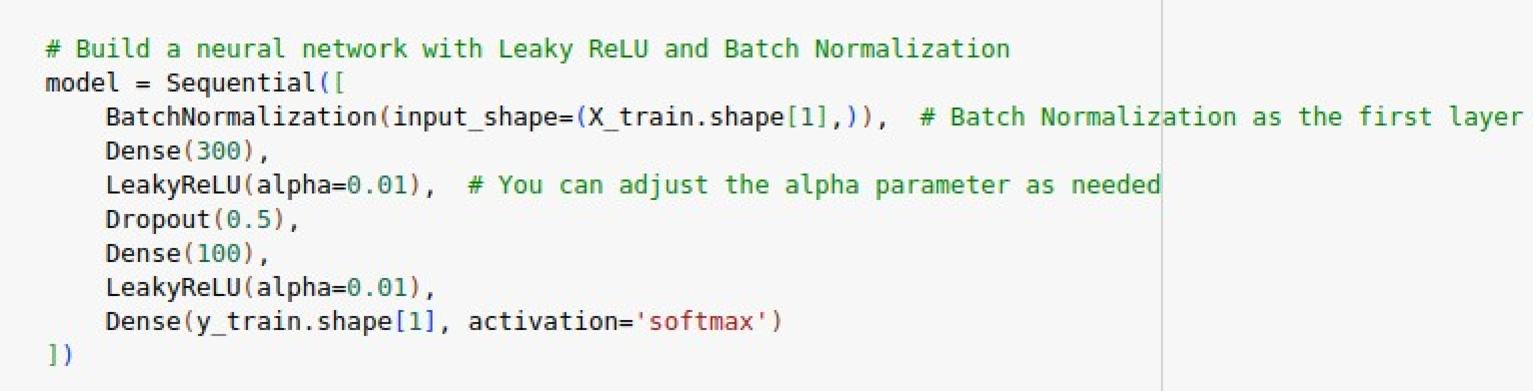
Test People ID: 6
Accuracy around 63%

Test People ID: 5
Accuracy around 33%

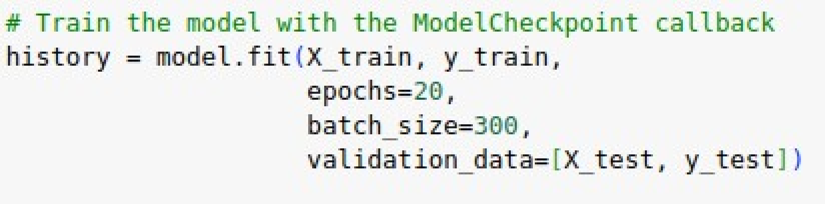
Model 1 with Simple NN: Data Prep. and Evaluation

- Data undersampled, so activitys are balanced
- random shuffle to prevent sampling bias
- 1 person test and 7 person train and iterate over all people
- take the best validation accuracy from all epochs
- do fitting 5 times and take average fitting accuracy for evaluation

Model 1 with Simple NN: Results



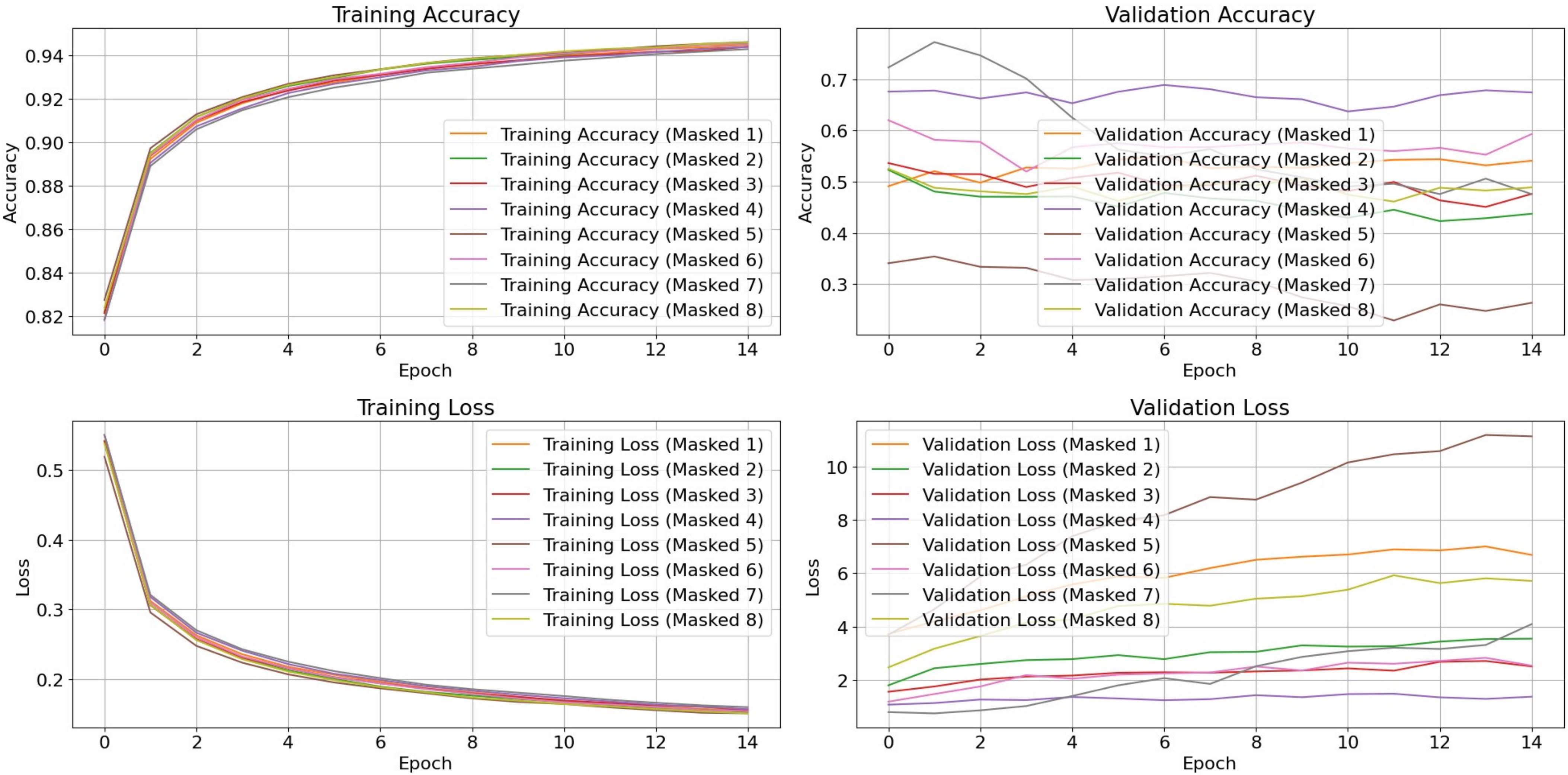
```python
# Build a neural network with Leaky ReLU and Batch Normalization
model = Sequential([
    BatchNormalization(input_shape=(X_train.shape[1],)),  # Batch Normalization as the first layer
    Dense(300),
    LeakyReLU(alpha=0.01),  # You can adjust the alpha parameter as needed
    Dropout(0.5),
    Dense(100),
    LeakyReLU(alpha=0.01),
    Dense(y_train.shape[1], activation='softmax')
])
```



```python
# Train the model with the ModelCheckpoint callback
history = model.fit(X_train, y_train,
                    epochs=20,
                    batch_size=300,
                    validation_data=[X_test, y_test])
```

Average accuracy of 57.1%
Standart deviation of 0.2%

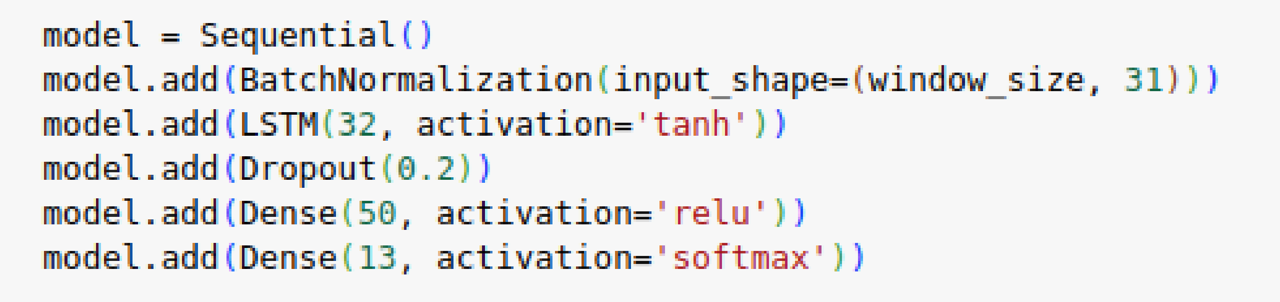
→ Already better!


Training Accuracy
Accuracy
0.82
0.84
0.86
0.88
0.90
0.92
0.94
Epoch
0
2
4
6
8
10
12
14
Training Accuracy (Masked 1)
Training Accuracy (Masked 2)
Training Accuracy (Masked 3)
Training Accuracy (Masked 4)
Training Accuracy (Masked 5)
Training Accuracy (Masked 6)
Training Accuracy (Masked 7)
Training Accuracy (Masked 8)
Validation Accuracy
Accuracy
0.3
0.4
0.5
0.6
0.7
Epoch
Validation Accuracy (Masked 1)
Validation Accuracy (Masked 2)
Validation Accuracy (Masked 3)
Validation Accuracy (Masked 4)
Validation Accuracy (Masked 5)
Validation Accuracy (Masked 6)
Validation Accuracy (Masked 7)
Validation Accuracy (Masked 8)
Training Loss
Loss
0.2
0.3
0.4
0.5
Epoch
Training Loss (Masked 1)
Training Loss (Masked 2)
Training Loss (Masked 3)
Training Loss (Masked 4)
Training Loss (Masked 5)
Training Loss (Masked 6)
Training Loss (Masked 7)
Training Loss (Masked 8)
Validation Loss
Loss
2
4
6
8
10
Epoch
Validation Loss (Masked 1)
Validation Loss (Masked 2)
Validation Loss (Masked 3)
Validation Loss (Masked 4)
Validation Loss (Masked 5)
Validation Loss (Masked 6)
Validation Loss (Masked 7)
Validation Loss (Masked 8)

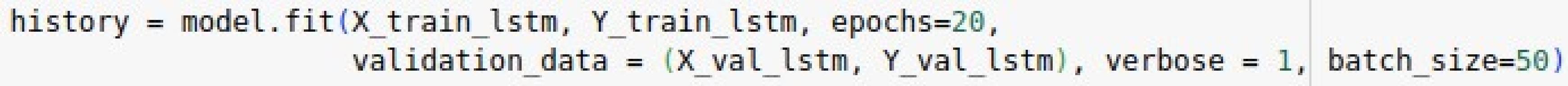
Model 2 with LSTM: Data Prep. and Evaluation

- Data undersampled, so activitys are balanced
- random shuffle of windows, to prevent sampling bias
- 1 person test and 7 person train. Iterate over all people
- take the best validation accuracy from all epochs
- do fitting 5 times and take average fitting accuracy for evaluation

Model 2 with LSTM: Results

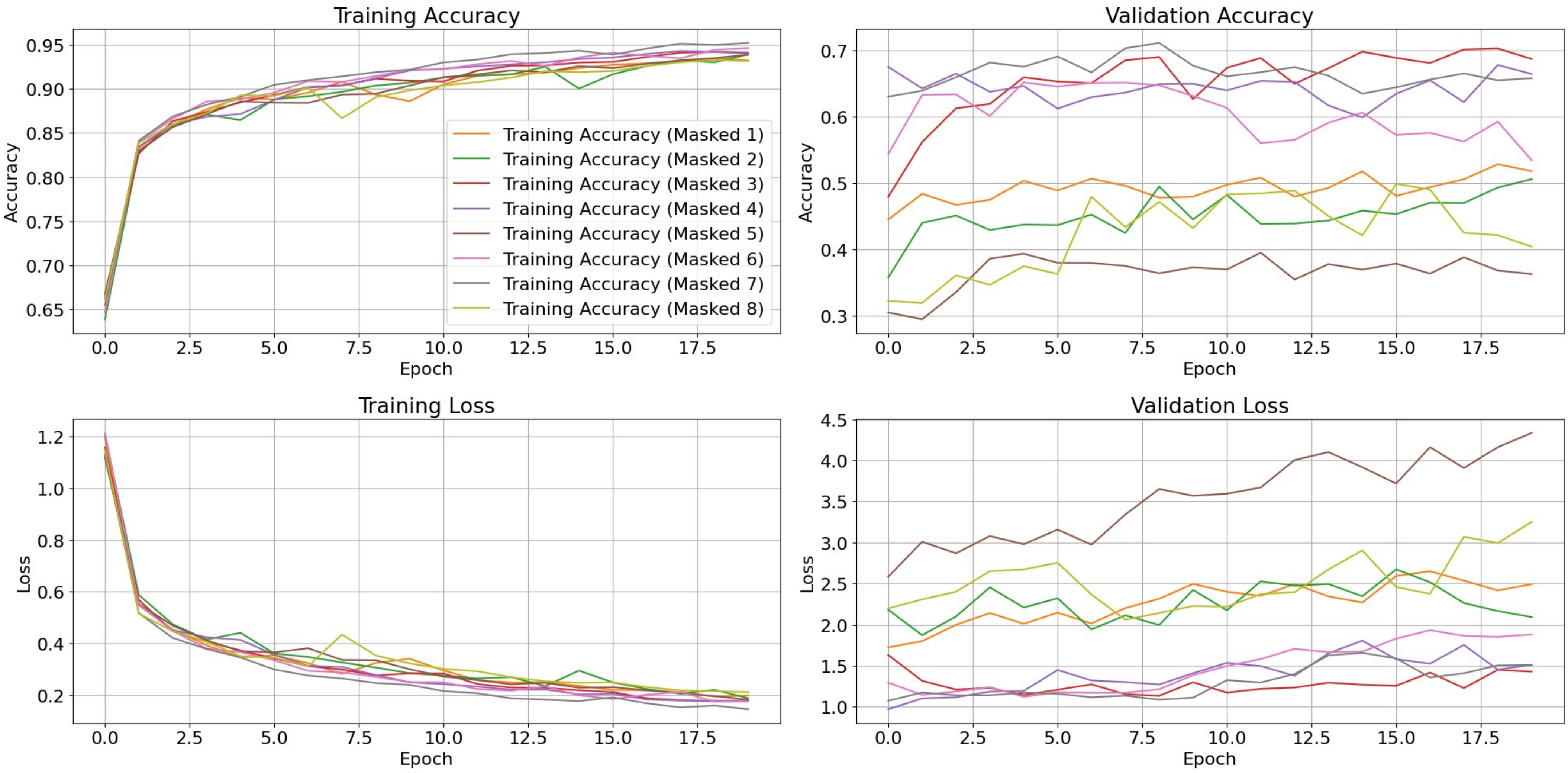


```
model = Sequential()
model.add(BatchNormalization(input_shape=(window_size, 31)))
model.add(LSTM(32, activation='tanh'))
model.add(Dropout(0.2))
model.add(Dense(50, activation='relu'))
model.add(Dense(13, activation='softmax'))
```



```
history = model.fit(X_train_lstm, Y_train_lstm, epochs=20,
                    validation_data = (X_val_lstm, Y_val_lstm), verbose = 1, batch_size=50)
```

Average accuracy of 59%
Standard deviation of 1%


Training Accuracy
Accuracy
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch
Training Accuracy (Masked 1)
Training Accuracy (Masked 2)
Training Accuracy (Masked 3)
Training Accuracy (Masked 4)
Training Accuracy (Masked 5)
Training Accuracy (Masked 6)
Training Accuracy (Masked 7)
Training Accuracy (Masked 8)
Validation Accuracy
Accuracy
0.7
0.6
0.5
0.4
0.3
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch
Training Loss
Loss
1.2
1.0
0.8
0.6
0.4
0.2
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch
Validation Loss
Loss
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Epoch

Results

- Both neural nets performed at least 10% better then the logistical regression.

- The first model performed with an accuracy around 57%

- The LSTM performed with an accuracy around 59% but worse then expected

Challenges and Errors

- The dataset contains many biases.

- Only 8 people

- Running time….

Conclusion and Outlook

Conclusion:

- As expected LSTM performed better then simple neural and the LSTM performed better then the sinple neural net. But the LSTM performed worse then expected.

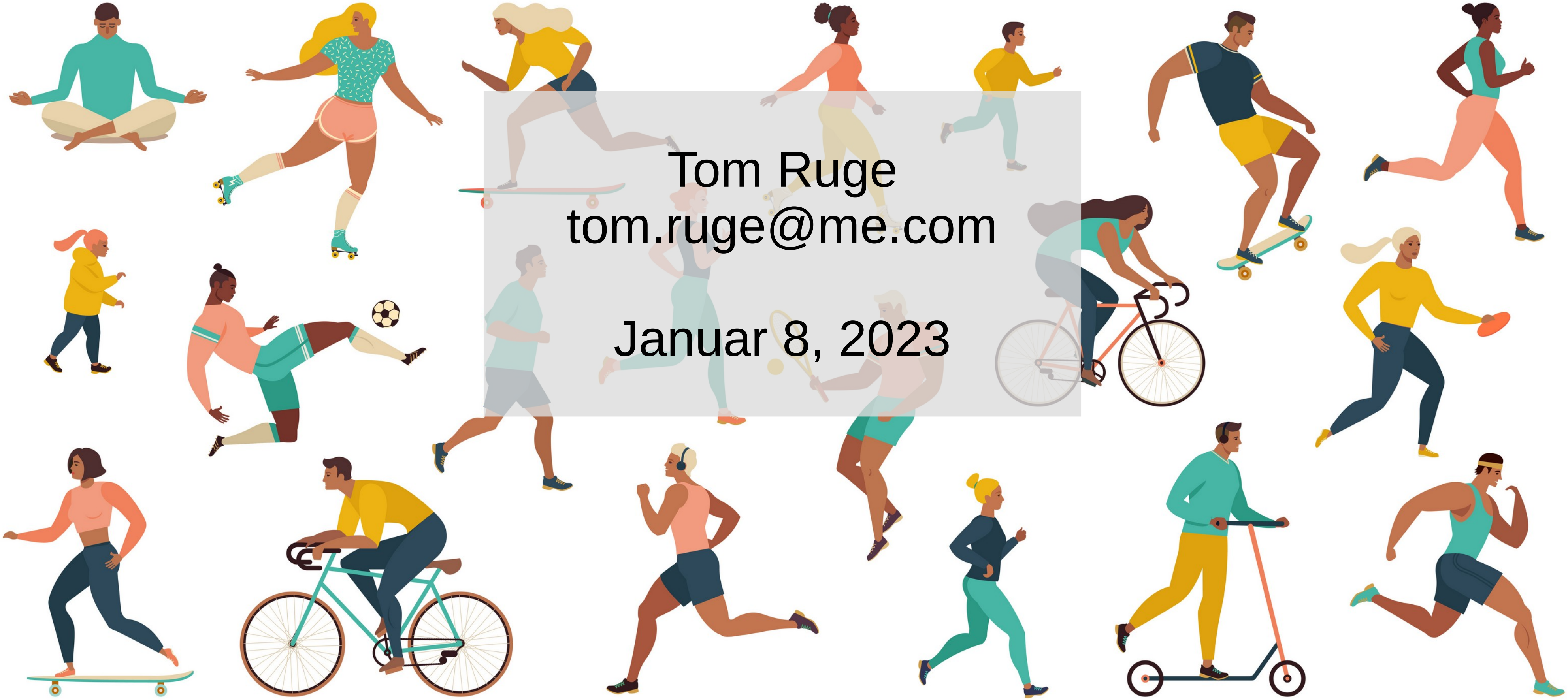
Outlook:

- Better investigation of Hyperparameters

- reduce number of features

- average the classifications to improve accuracy.

Questions, Comments?



Tom Ruge
tom.ruge@me.com

Januar 8, 2023