MACHINE LEARNING ONDERZOEK

AANTAL DRAAIINGEN

Side swing: accuraat met foutenmarge van 1

**CNN**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 740 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *~~overlap~~*   Validatieset:   * 1 sec *overlap* (50%) |
| **shuffle** | ja |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(64, input\_dim=3,  kernel\_regularizer=regularizers.l2(0.1),  activity\_regularizer=regularizers.l1(0.1)))  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | 0.1667 |

# Experiment 2

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 740 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *~~overlap~~*   Validatieset:   * 1 sec *overlap* (50%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.3551 - accuracy: 0.8988 |
| **Scenario confusion matrix** |  |
| **accuraatheid** |  |

# Experiment 3: beter knippen signalen

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 740 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *~~overlap~~*   Validatieset:   * 1 sec *overlap* (50%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** |  |
| **Scenario confusion matrix** |  |
| **accuraatheid** |  |

# Experiment 4: verhogen window

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 740 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1.5 sec *~~overlap~~*   Validatieset:   * 1.5 sec *overlap* (50%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.3213 - accuracy: 0.8826 |
| **Scenario confusion matrix** |  |
| **accuraatheid** |  |

# Experiment 5: validationset + window

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 740 |
| **Validatie set grootte** | 32 |
| **window** | Trainingsset:   * 2 sec *~~overlap~~*   Validatieset:   * 2 sec *overlap* (50%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 2.1211 - accuracy: 0.5938 |

# Experiment 6: overlap

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 740 |
| **Validatie set grootte** | 32 |
| **window** | Trainingsset:   * 2 sec *overlap* (50%)   Validatieset:   * 2 sec *overlap* (50%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 2.8803 - accuracy: 0.5000 |

# Experiment 7: overlap

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Forward\_180: 19 368 * Backward\_180: 21 648 * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 2818 |
| **Validatie set grootte** | 16 |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 2.5431 - accuracy: 0.5000 |

# Experiment 8: zonder forward en backward 180

|  |  |
| --- | --- |
| **Dataset grootte** | Voor segmentatie: 90 528   * Jump\_slow: 29 360 * Jump\_fast: 20 032 * Side\_swing: 41 128 * Cross\_over: 15 088   Na segmentatie: 1 882 |
| **Validatie set grootte** | 16 |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.8272 - accuracy: 0.7143 |

# Experiment 9: backward 180 (nieuwe metingen)

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 2 354 |
| **Validatie set grootte** | 15 |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.6403 - accuracy: 0.8667 |

# Experiment 10: backward 180 + forward 180 (nieuwe metingen)

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 2 825 |
| **Validatie set grootte** | 16 |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 1.3847 - acc: 0.7500 |

# Experiment 11: backward 180 + forward 180 + validatie data

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 2 825 |
| **Validatie set grootte** | 94 |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 1.3566 - acc: 0.6809 |

# Experiment 12: zonder backward en forward + validatie data

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 1882 |
| **Validatie set grootte** | 88 |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.5001 - acc: 0.8977 |

# Experiment 13: train test split (met springen)

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 2261 |
| **Validatie set grootte** | \ |
| **window** | Trainingsset:   * 2 sec *overlap* (70%)   Validatieset:   * 2 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.0296 - acc: 0.9912 |
| **validatie** |  |

# Experiment 13: window verlagen + predicten op validatiedata

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 2261 |
| **Validatie set grootte** | \ |
| **window** | Trainingsset:   * 1 sec *overlap* (70%)   Validatieset:   * 1 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(Dropout(0.5))  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **labels** | array(['backward\_180', 'cross\_over', 'forward\_180', 'jump\_fast', 'jump\_slow', 'side\_swing'], dtype=object) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** |  |
| **validatie** | Scenario 4 |

# Experiment 14: batchnormalization

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 2261 |
| **Validatie set grootte** | \ |
| **window** | Trainingsset:   * 1 sec *overlap* (70%)   Validatieset:   * 1 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  #model.add(Dense(32, activation='relu', activity\_regularizer=l1(0.001)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(BatchNormalization())  model.add(Dropout(0.3))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(BatchNormalization())  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **labels** | array(['backward\_180', 'cross\_over', 'forward\_180', 'jump\_fast', 'jump\_slow', 'side\_swing'], dtype=object) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** |  |
| **validatie** | Scenario 4: 4, 2, 3, 0, 4, 1, 4, 5 |

# Experiment 15: splitsen verschillende variaties

## Right backwards + right left

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 946 |
| **Validatie set grootte** | \ |
| **window** | Trainingsset:   * 1 sec *overlap* (30%)   Validatieset:   * 1 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  #model.add(Dense(32, activation='relu', activity\_regularizer=l1(0.001)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(BatchNormalization())  model.add(Dropout(0.3))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(BatchNormalization())  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **labels** | array(['backward\_180', 'cross\_over', 'forward\_180', 'jump\_fast', 'jump\_slow', 'side\_swing'], dtype=object) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.1361 - accuracy: 0.9716 |
| **validatie** |  |

## Right forward + right right

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 660 |
| **Validatie set grootte** | 157 |
| **window** | Trainingsset:   * 1 sec *overlap* (30%)   Validatieset:   * 1 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  #model.add(Dense(32, activation='relu', activity\_regularizer=l1(0.001)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(BatchNormalization())  model.add(Dropout(0.3))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(BatchNormalization())  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **labels** | array(['backward\_180', 'cross\_over', 'forward\_180', 'jump\_fast', 'jump\_slow', 'side\_swing'], dtype=object) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 1.0013 - accuracy: 0.8917 |
| **validatie** |  |

## left forward + left right

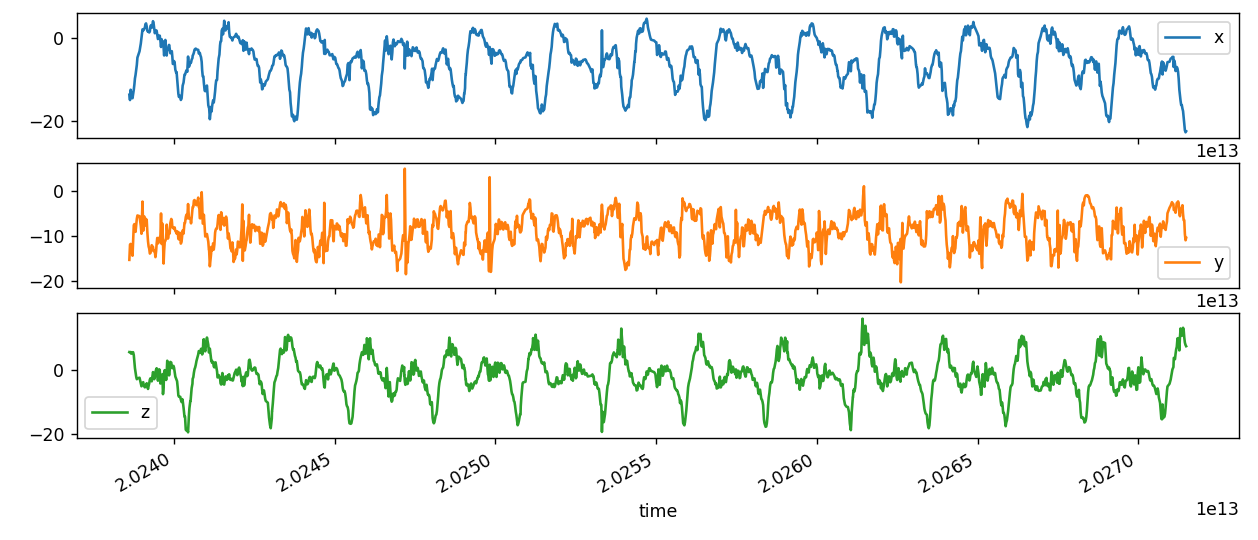
|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 802 |
| **Validatie set grootte** | 155 |
| **window** | Trainingsset:   * 1 sec *overlap* (30%)   Validatieset:   * 1 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  #model.add(Dense(32, activation='relu', activity\_regularizer=l1(0.001)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(BatchNormalization())  model.add(Dropout(0.3))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(BatchNormalization())  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **labels** | array(['backward\_180', 'cross\_over', 'forward\_180', 'jump\_fast', 'jump\_slow', 'side\_swing'], dtype=object) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 0.4904 - accuracy: 0.8387 |
| **validatie** |  |

## left backward + left left

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 546 |
| **Validatie set grootte** | 157 |
| **window** | Trainingsset:   * 1 sec *overlap* (30%)   Validatieset:   * 1 sec *~~overlap~~* (0%) |
| **shuffle** | \ |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **CNN model** | model = Sequential()  #model.add(Dense(32, activation='relu', activity\_regularizer=l1(0.001)))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu', input\_shape=(n\_timesteps,n\_features)))  model.add(BatchNormalization())  model.add(Dropout(0.3))  model.add(Conv1D(filters=64, kernel\_size=3, activation='relu'))  model.add(BatchNormalization())  model.add(MaxPooling1D(pool\_size=2))  model.add(Flatten())  model.add(Dense(100, activation='relu'))  model.add(Dense(n\_outputs, activation='softmax')) |
| **labels** | array(['backward\_180', 'cross\_over', 'forward\_180', 'jump\_fast', 'jump\_slow', 'side\_swing'], dtype=object) |
| **History model fit** |  |
| **Model summary** |  |
| **Learning curve** |  |
| **Confusion matrix** |  |
| **accuraatheid** | loss: 1.1583 - accuracy: 0.8089 |
| **validatie** |  |

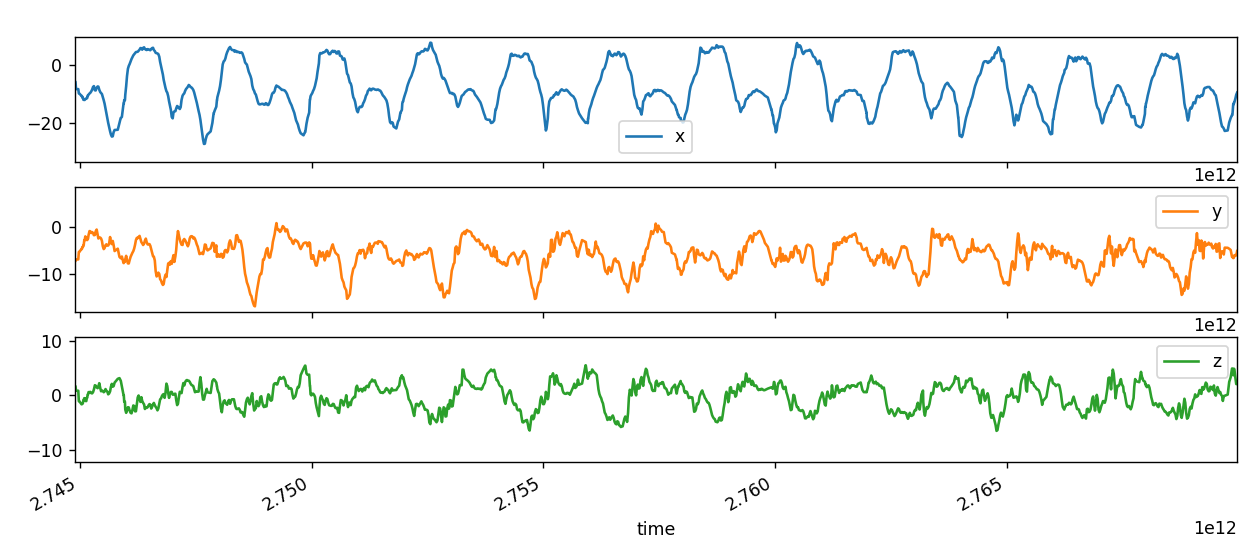
Forward 180 FOUT gepredict als backward 180

FORWARD\_180

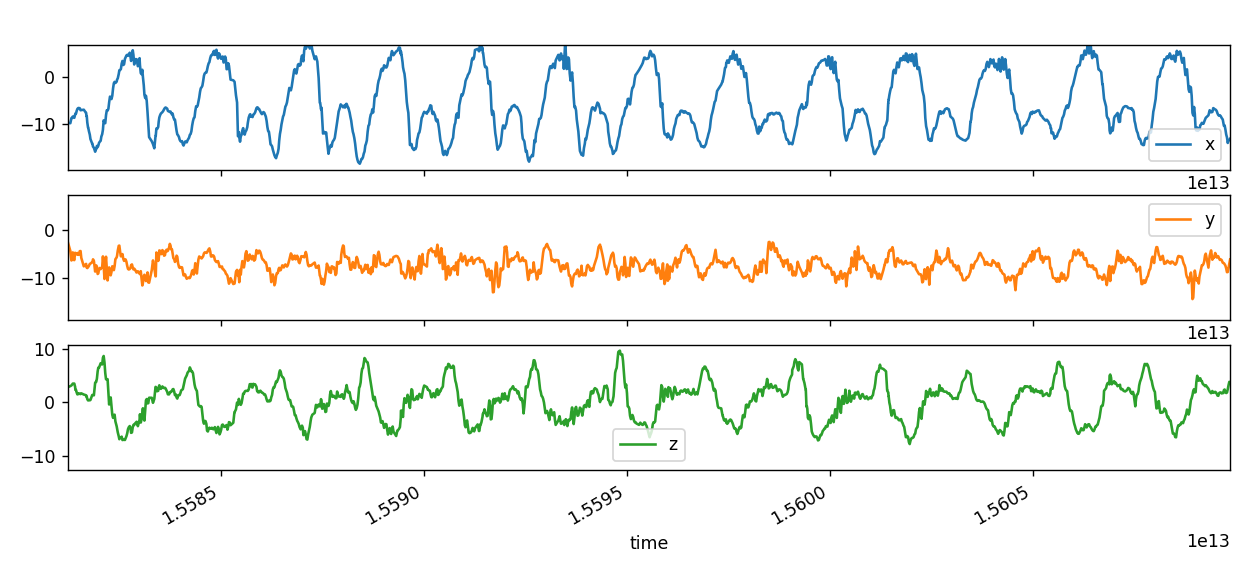


GETRAINDE BACKWARD 180

1

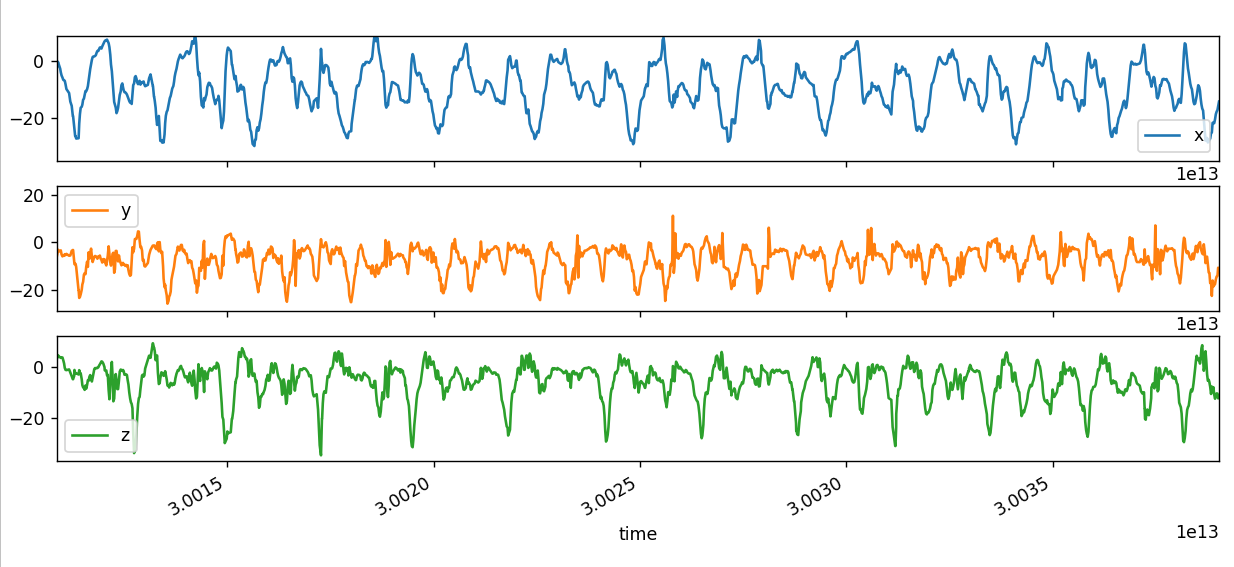


2



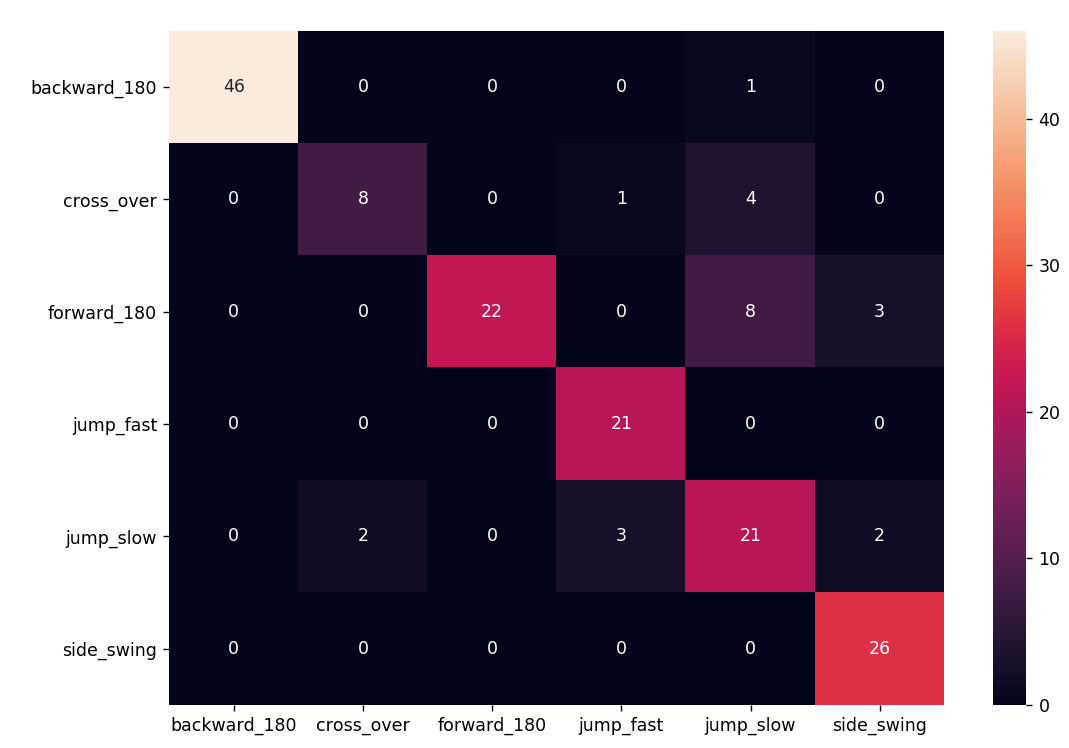
GETRAINDE FORWARD 180

1

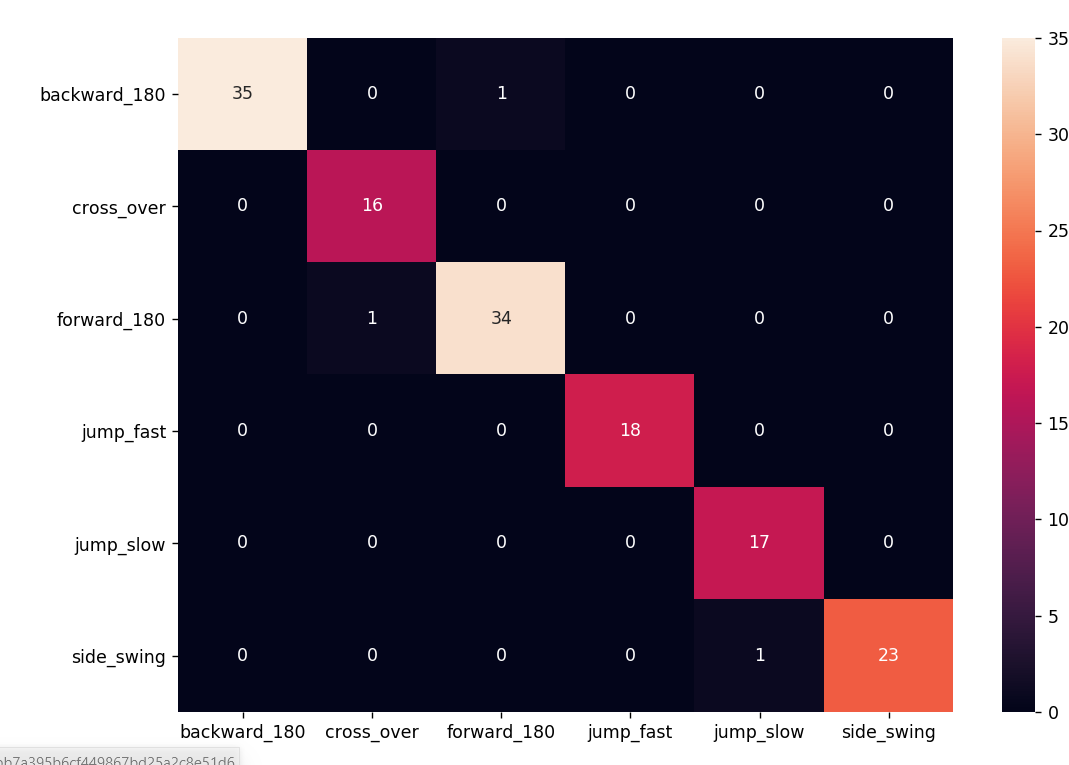


FORWARD BACKWARD

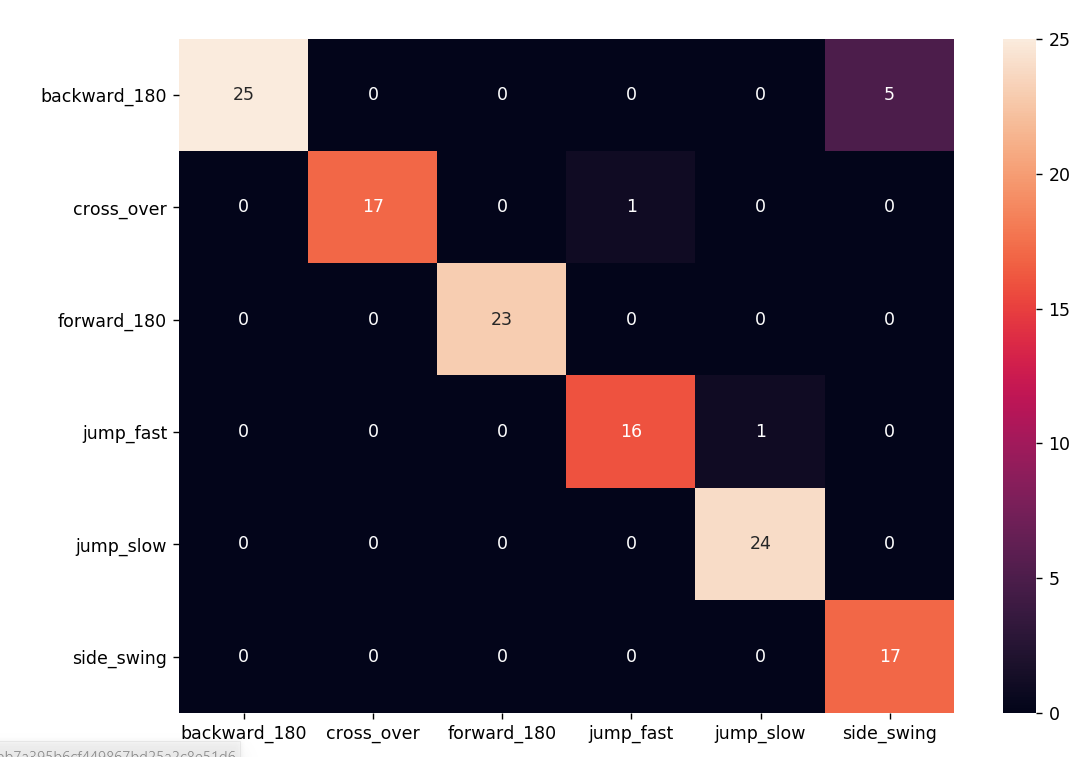
Left left Right right left\_backward\_left\_left



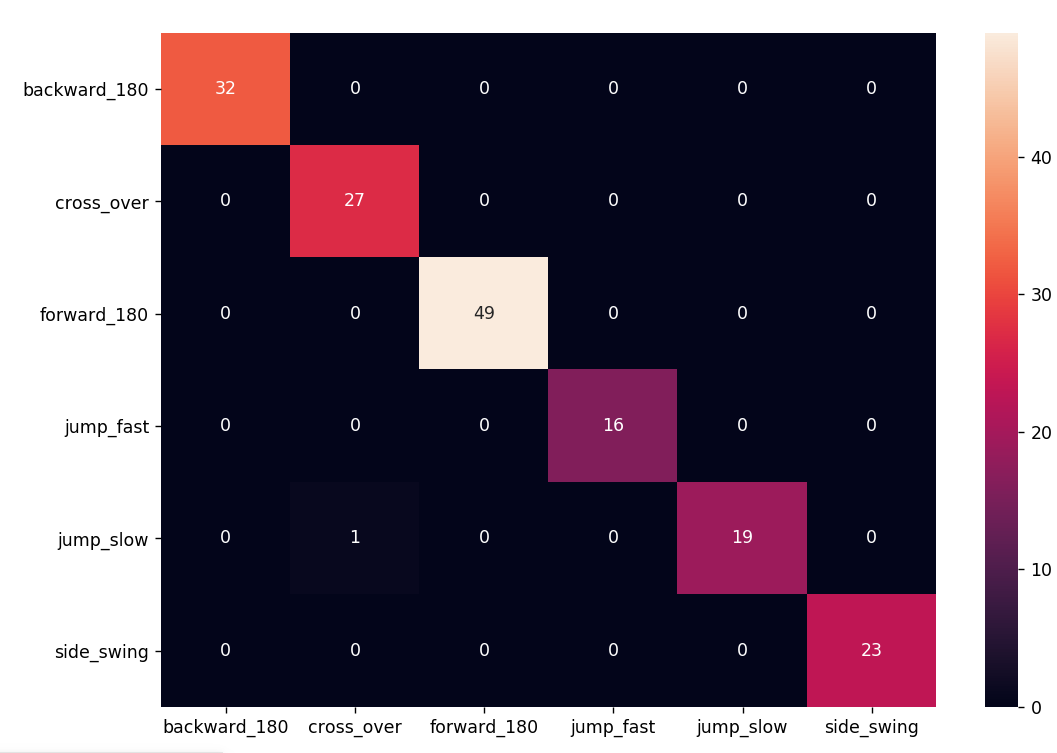
Right right Left left right\_forward\_right\_right



Right left Left rigth left\_forward\_left\_right



Left rigth Right left right\_backward\_right\_left



# Experiment 16: verhogen window -> 1.5

FORWARD BACKWARD

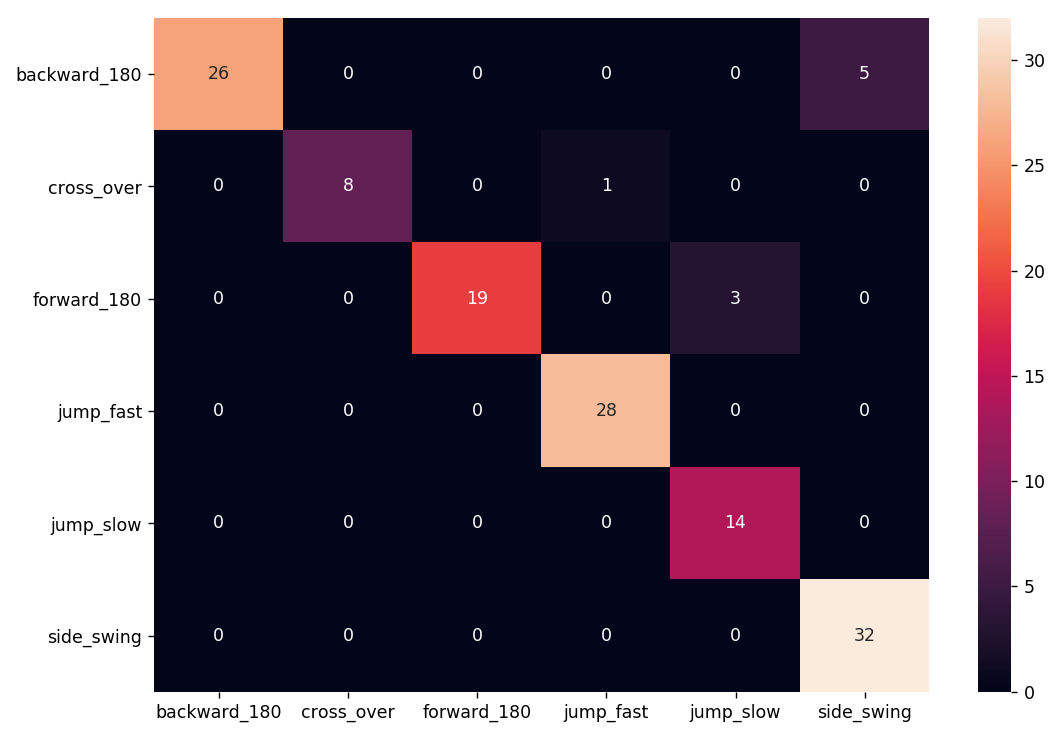
Left left Right right left\_backward\_left\_left



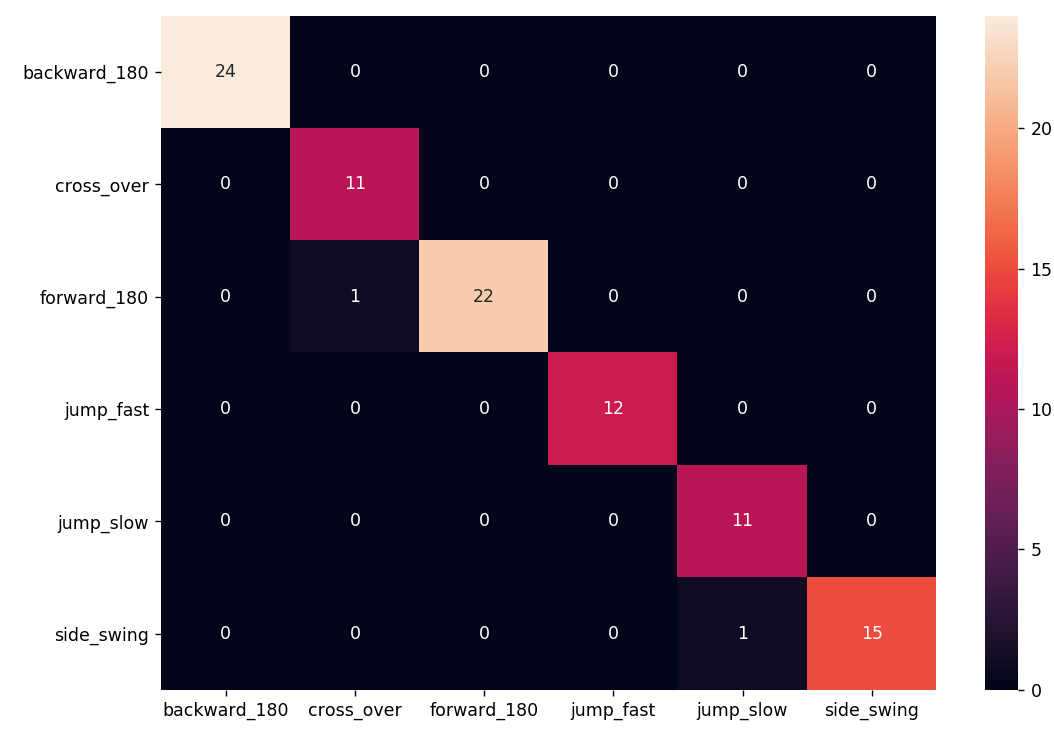
* Meer data nodig?

Jump slow vs cross over

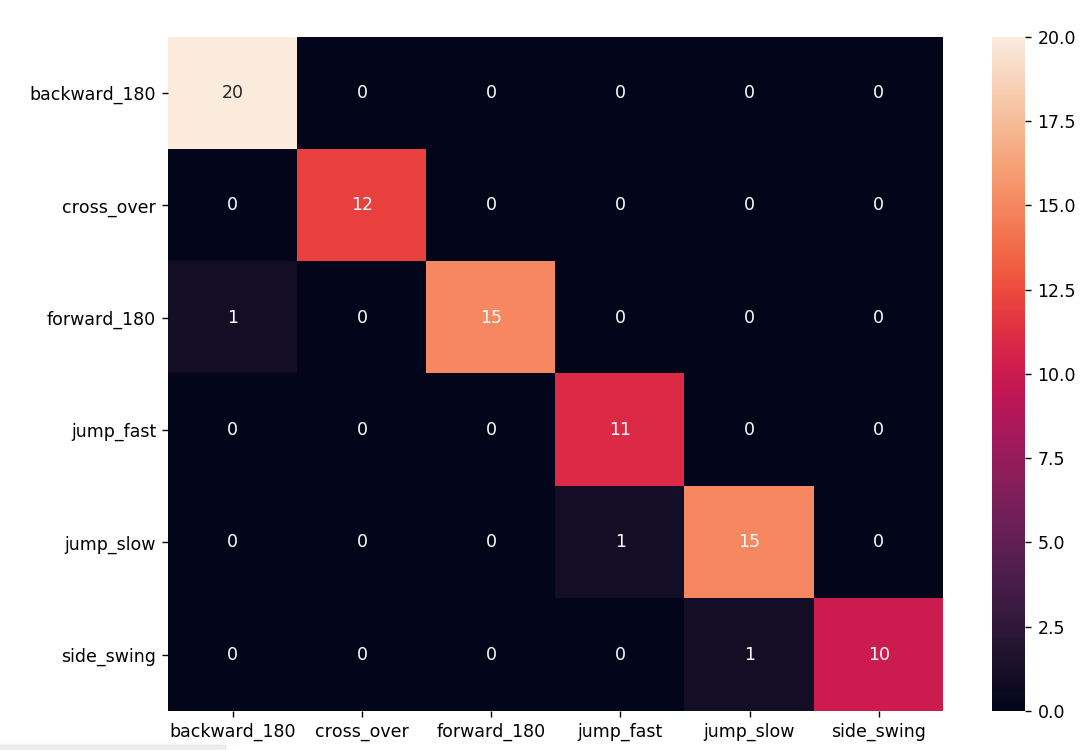
Verkeerd knippen data



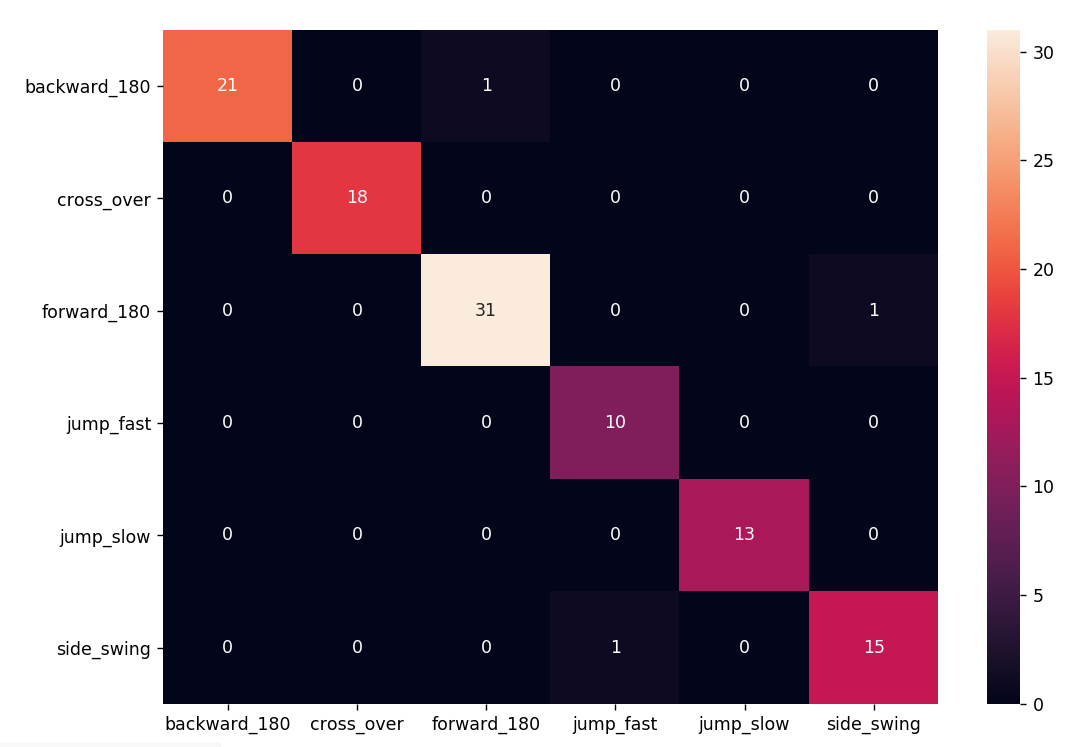
Right right Left left right\_forward\_right\_right



Right left Left rigth left\_forward\_left\_right

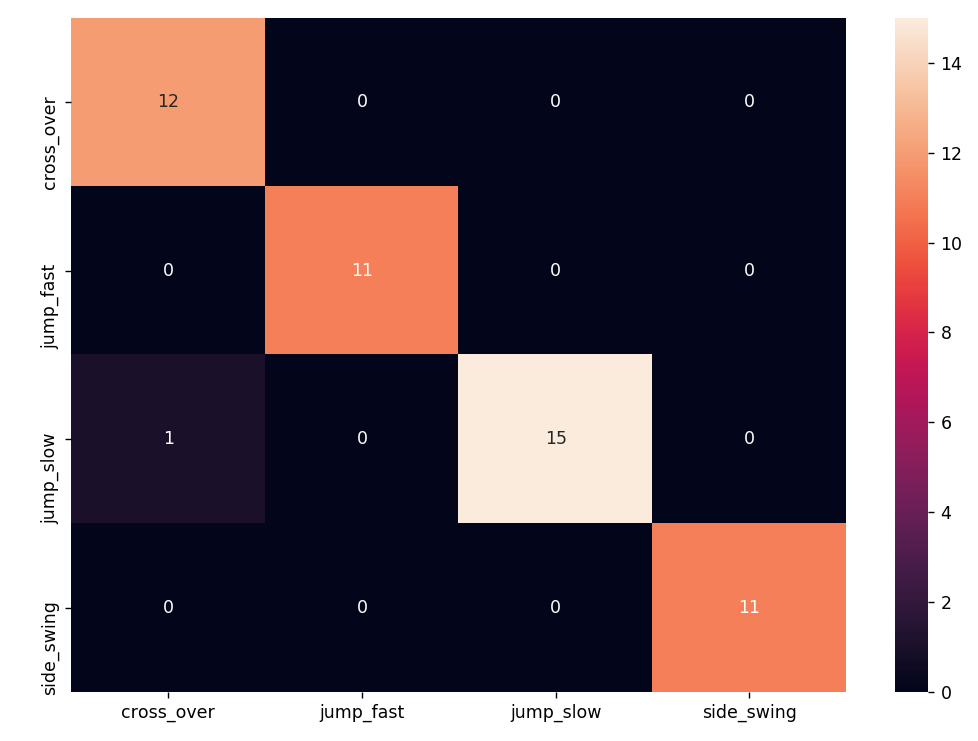


Left rigth Right left right\_backward\_right\_left

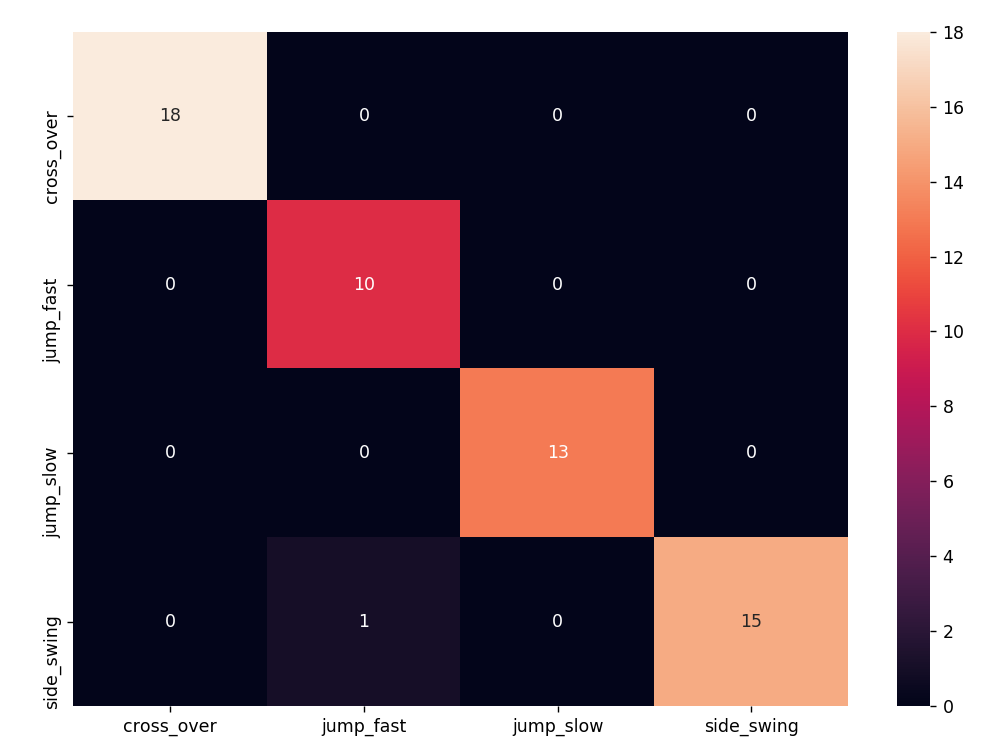


ZONDER 180s

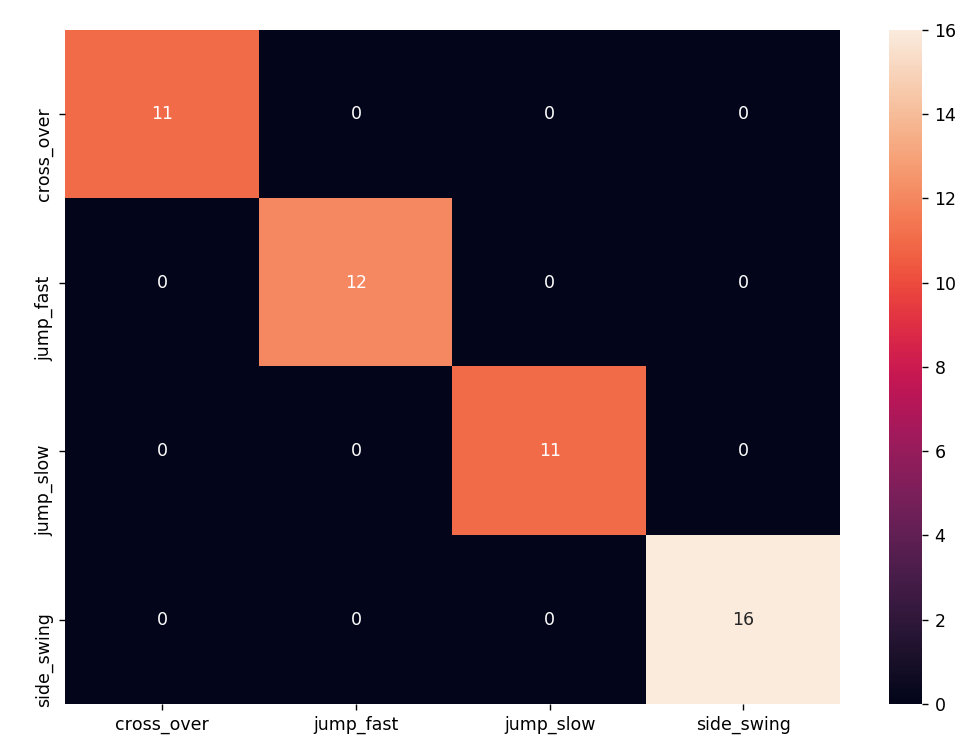
Left forward – left right



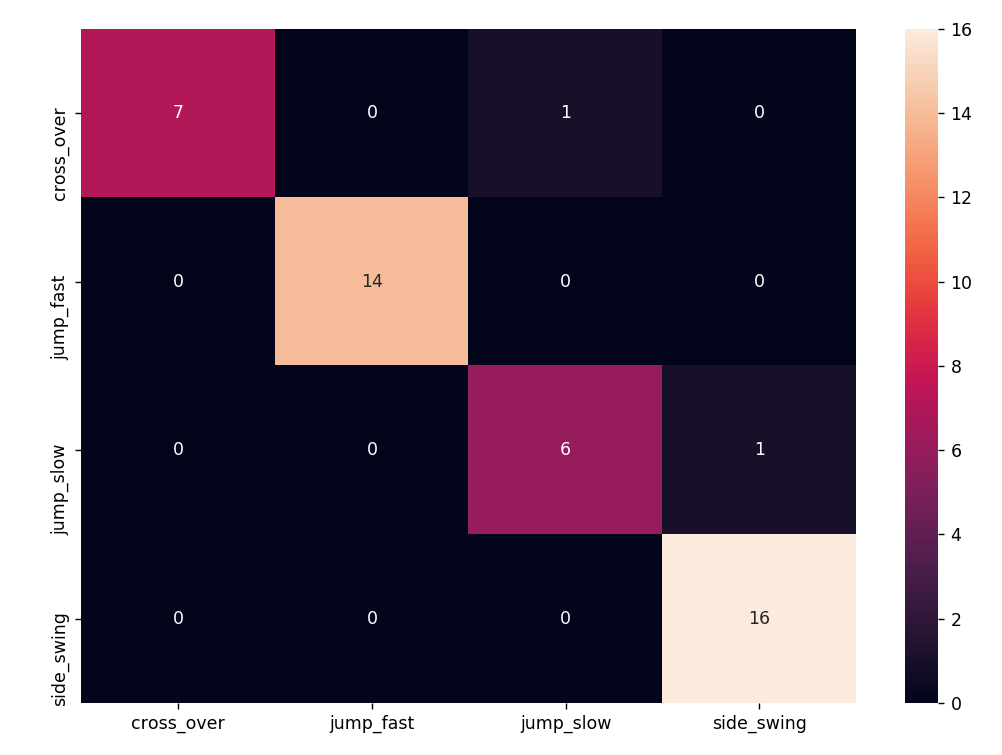
right\_backward\_right\_left



right\_forward\_right\_right



left\_backward\_left\_left



**EVALUATIE (niet geknipt signaal)**

**ZONDER 180s**

**Left forward – left right**

SIDE SWING

array([0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0], dtype=int64)

array([1, 0, 5, 5, 5, 5, 5, 5, 5, 5, 5, 1], dtype=int64)

JUMP SLOW

array([0, 3, 3, 2, 2, 2, 2, 2, 2, 2, 3, 0], dtype=int64)

array([1, 0, 5, 4, 4, 4, 4, 4, 4, 4, 0, 1], dtype=int64)

JUMP FAST

array([0, 2, 1, 1, 1, 1, 1, 3, 0], dtype=int64)

array([1, 4, 3, 3, 3, 3, 3, 0, 1], dtype=int64)

CROSS OVER

array([0, 2, 0, 0, 0, 0, 0, 0, 0, 2, 0], dtype=int64)

array([1, 4, 1, 1, 1, 1, 1, 1, 1, 0, 1], dtype=int64)

left\_backward\_left\_left

SIDE SWING

array([3, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3], dtype=int64)

JUMP SLOW

array([3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3], dtype=int64)

JUMP FAST

array([3, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3], dtype=int64)

CROSS OVER

array([3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 3], dtype=int64)

right\_forward\_right\_right

SIDE SWING

array([0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 0], dtype=int64)

JUMP SLOW

array([0, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 0], dtype=int64)

JUMP FAST

array([0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0], dtype=int64)

CROSS OVER

array([1, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)

right\_backward\_right\_left

SIDE SWING

array([3, 3, 3, 3, 3, 3, 3, 3], dtype=int64)

JUMP SLOW

array([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 0], dtype=int64)

JUMP FAST

array([1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 2, 0], dtype=int64)

CROSS OVER

array([3, 0, 0, 0, 0, 0], dtype=int64)

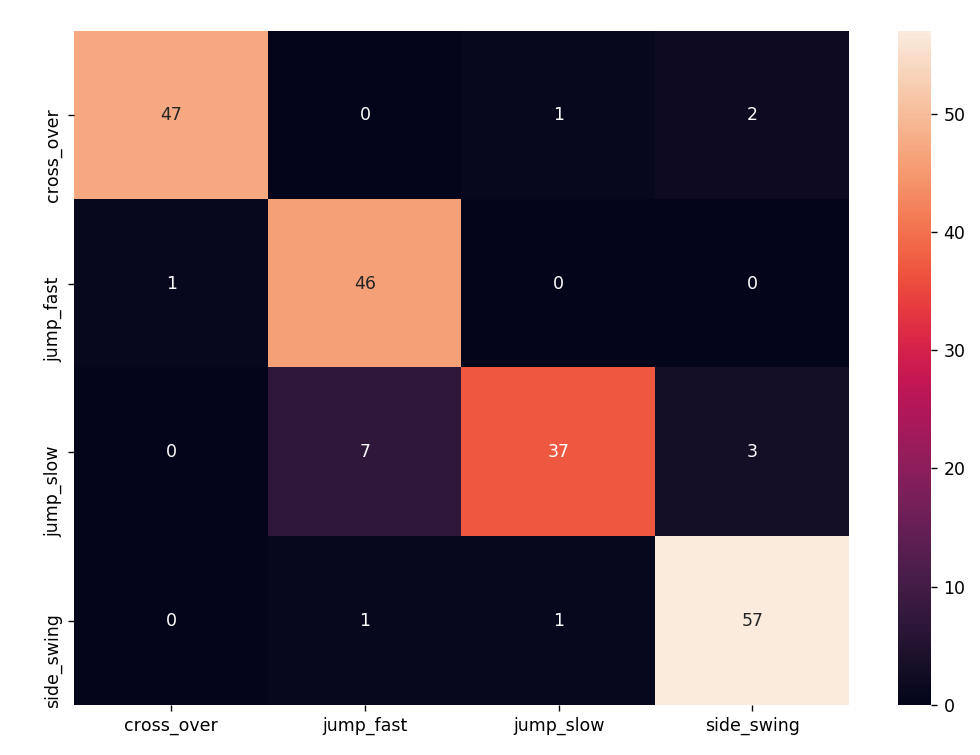
* Zekerheid meestal lager bij verkeerd geclassificeerde
* Bij starten sessie enkel effectief springen behouden want eerste en laatste segmenten is zekerheid vr andere sprong soms heel hoog

# Experiment 18: enkel proefpersoon2

Niet veel verschil (ietsje beter)

# Experiment 19: alles terug samen (zonder 180s)

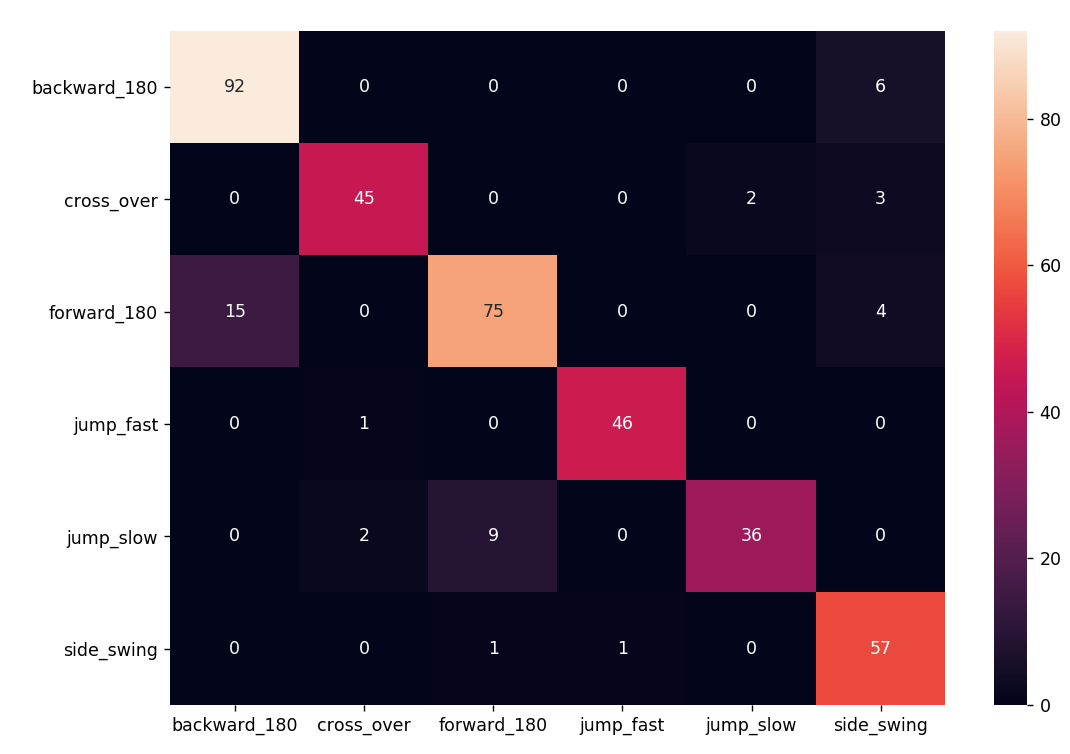
loss: 0.2376 - accuracy: 0.9212



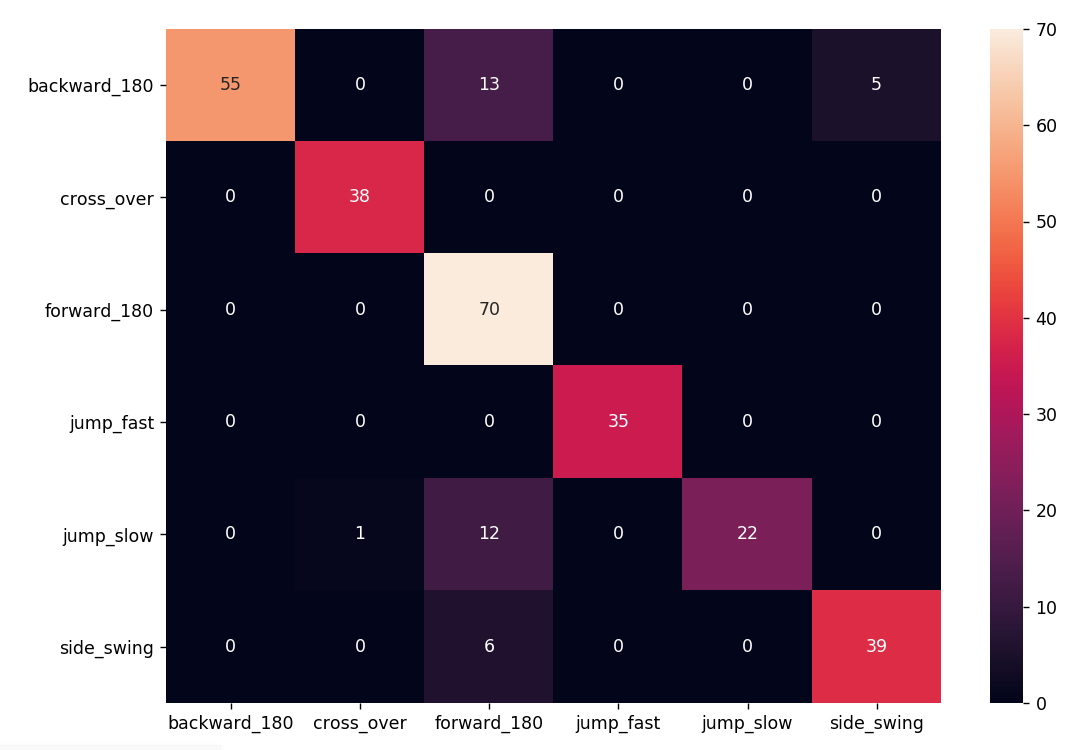
# Experiment 20: alles terug samen (met 180s)

Window: 1.5

loss: 0.5595 - accuracy: 0.8886

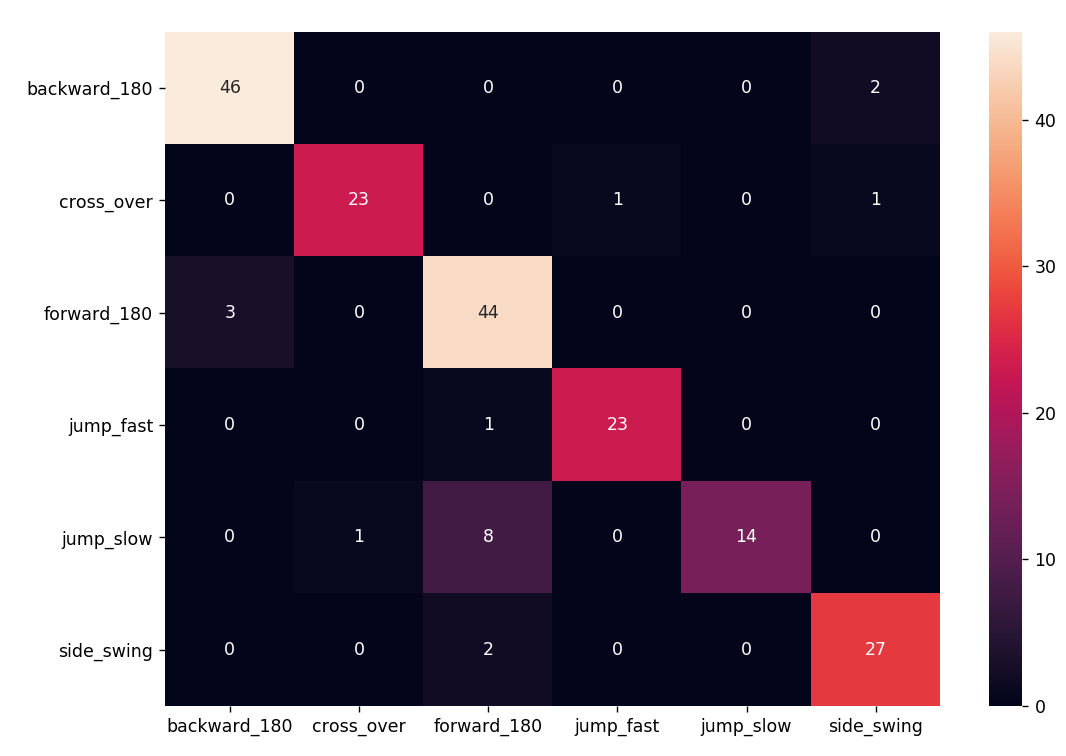


Window: 2



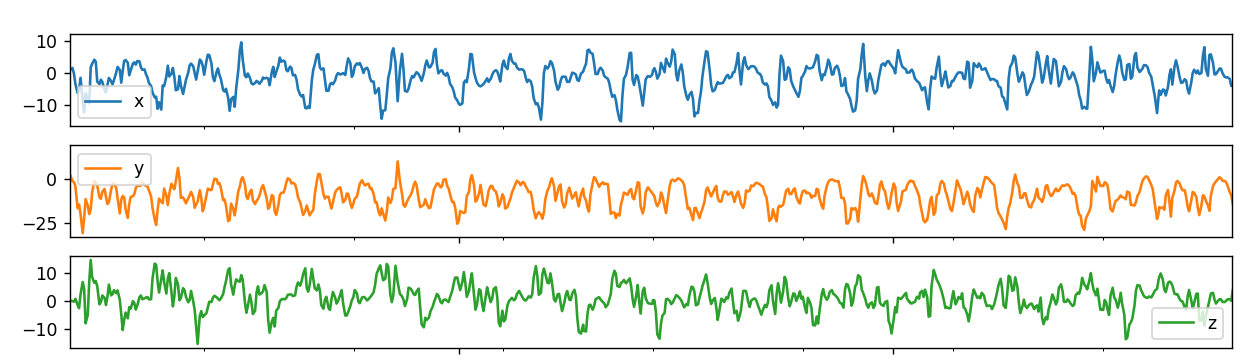
Window: 3

loss: 0.6616 - accuracy: 0.9031

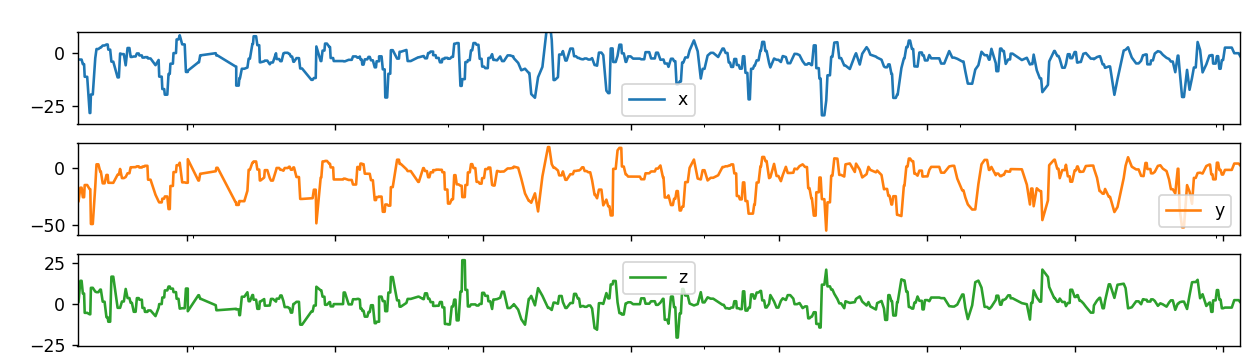


# Experiment 21: cross over fast en slow

SLOW



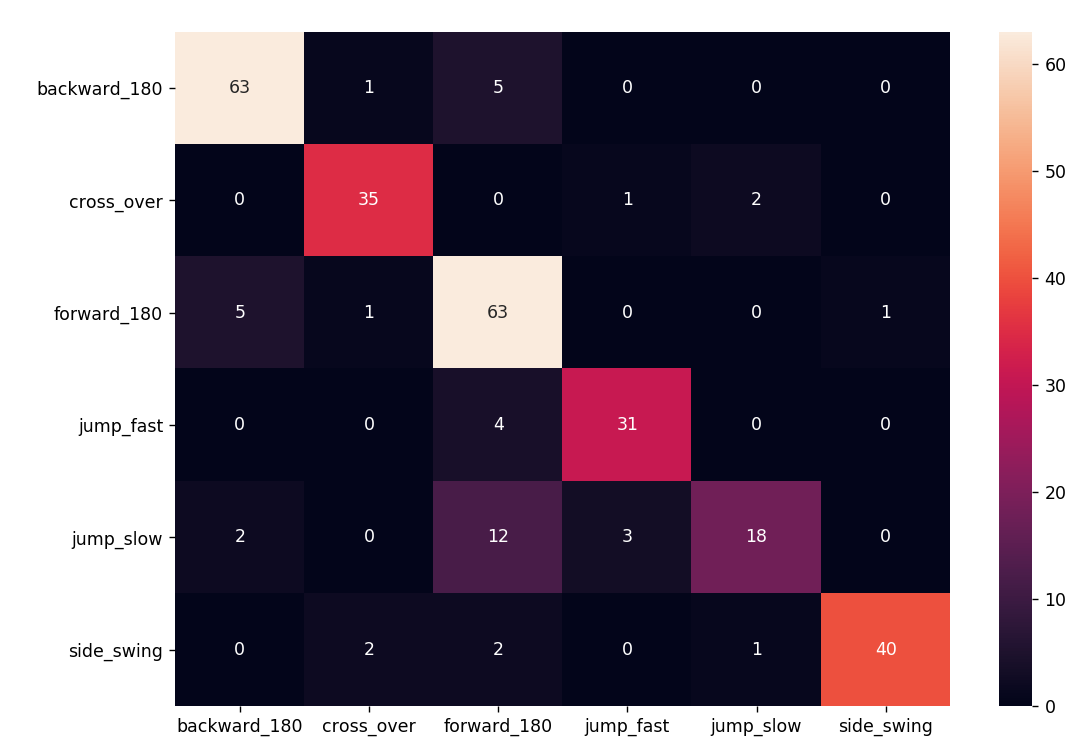
FAST



# Experiment 22: alles in 1 model (180s met springen)

loss: 0.5076 - accuracy: 0.8562

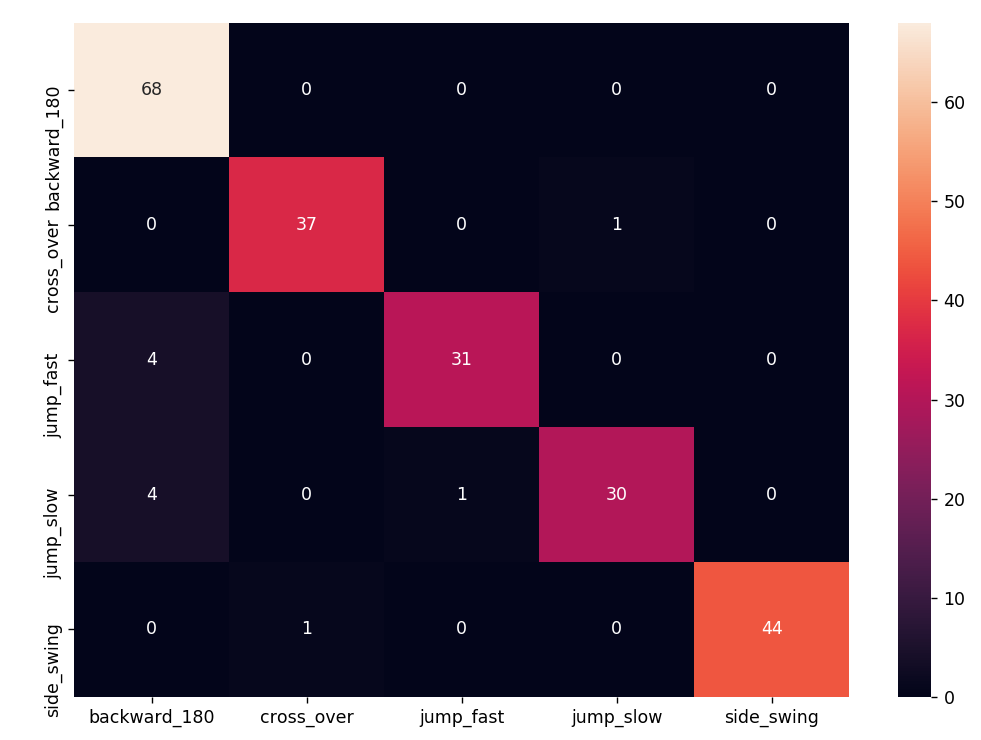
Window = 2



Forward 180: gemeten met sprong ervoor en erachter -> confusion

Zonder sprong, niet conform met werkelijkheid

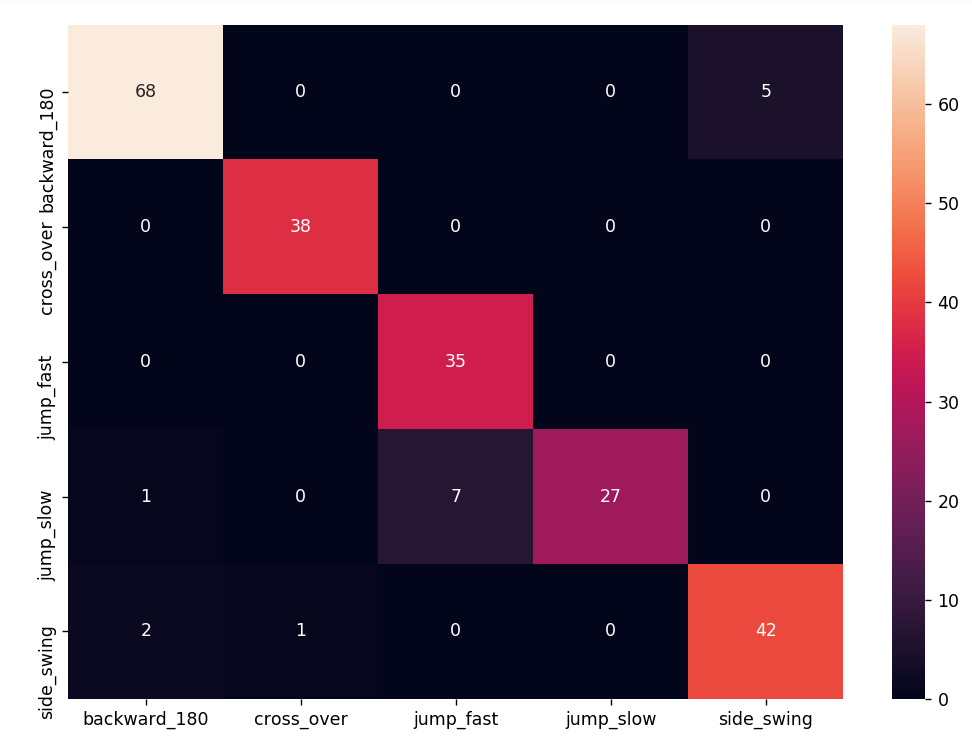
# Experiment 22: alles in 1 model (180 met springen, zonder forward 180)



# Experiment 23: alles in 1 model (180 zonder springen, zonder forward 180)

Window: 2

loss: 0.5625 - accuracy: 0.9292



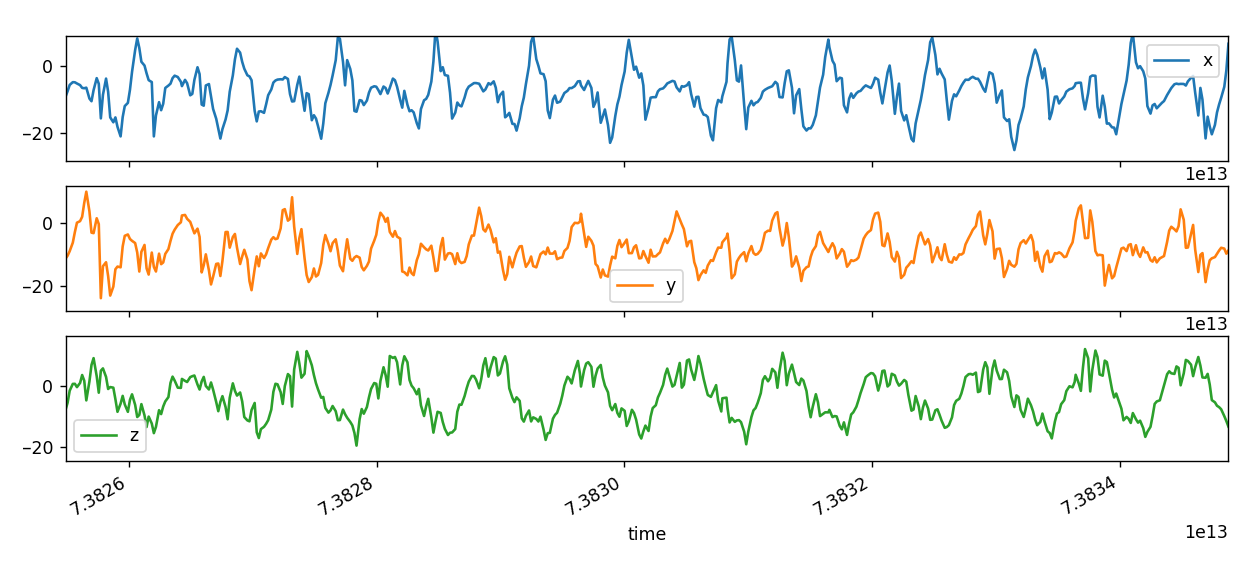
* Iets lagere accuraatheid op nieuwe data

Window: 1

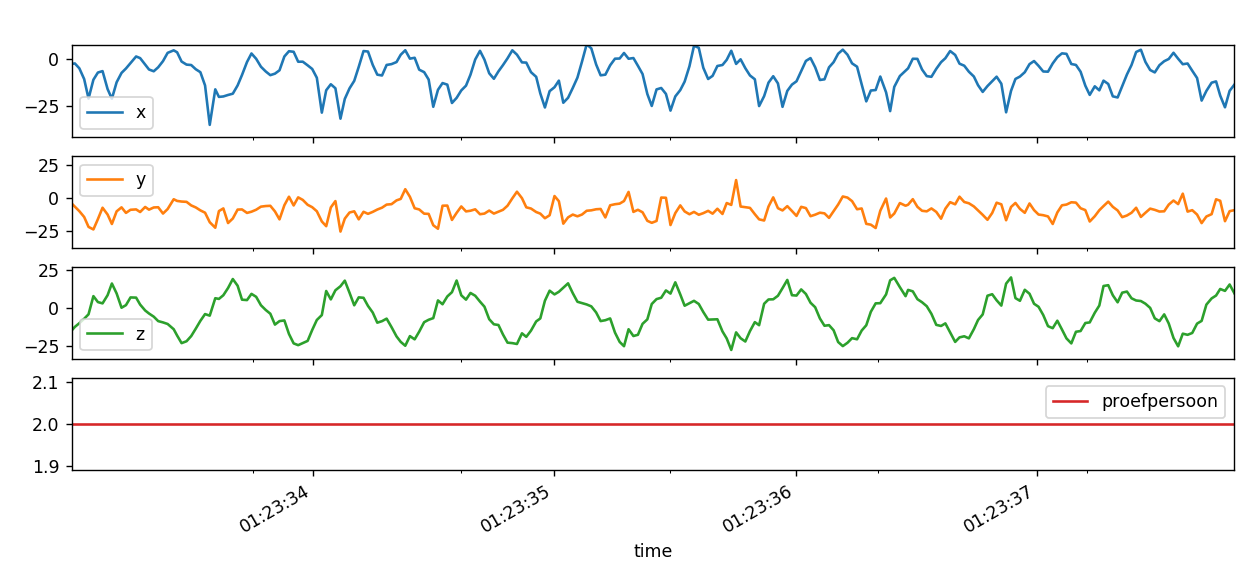
# Experiment 24: jump\_run vs jump fast vs jump slow

Left forward

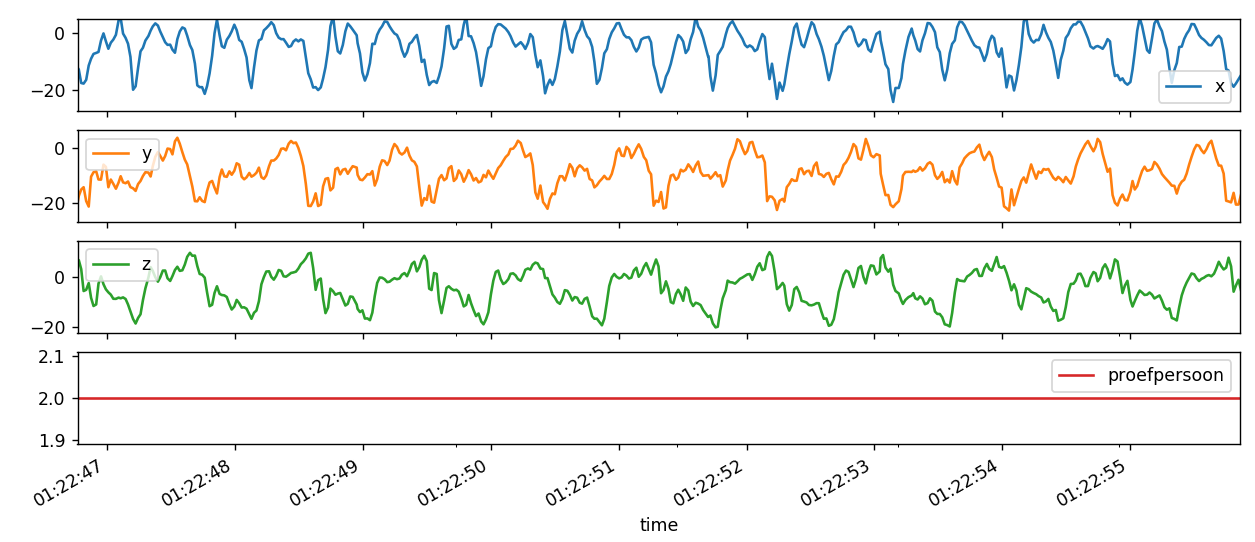
JUMP RUN



JUMP FAST

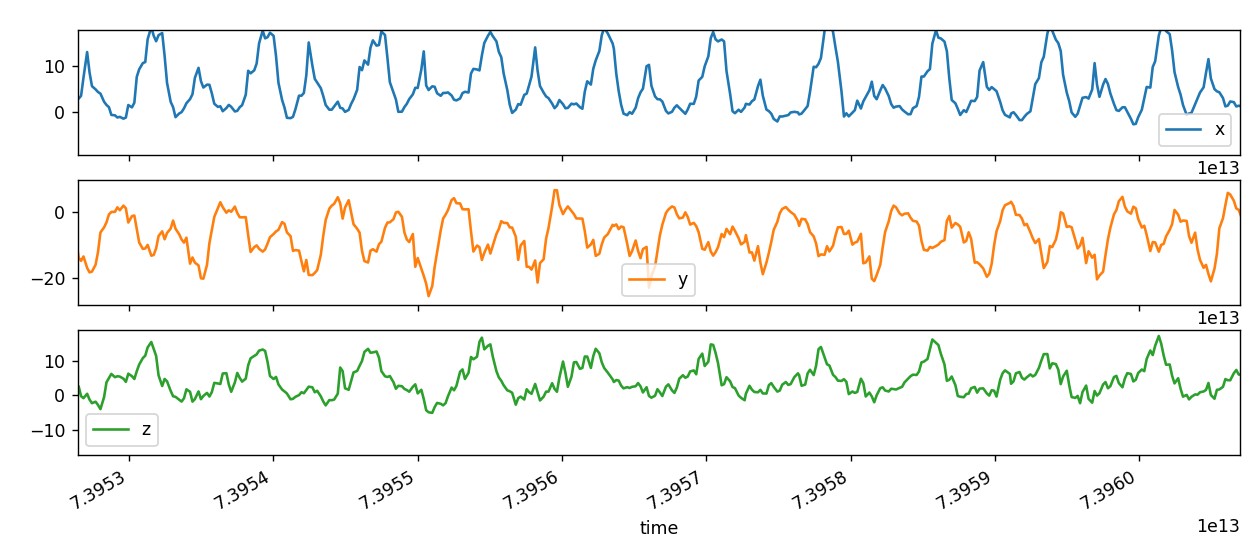


JUMP SLOW

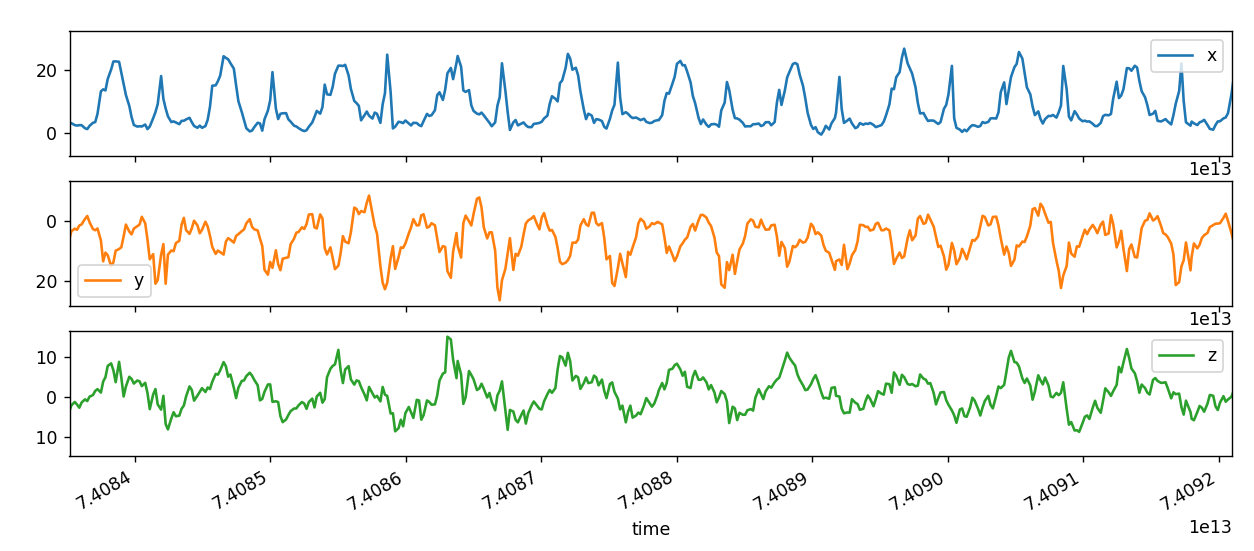


Right forward

JUMP RUN



JUMP RUN TURN RIGHT



* verschil

**SVC**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **Dimension reduction** | 6 |
| **gridsearch** | {'C': [0.1,1, 10, 100],  'gamma': [1,0.1,0.01,0.001],  'kernel': ['linear','rbf', 'poly', 'sigmoid']} |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 0.675  Test set score: 0.662 |
| **Scenario confusion matrix** |  |
| **accuraatheid** | Test set score: 0.175 |

**LinearSVC**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | {'C': [0.1,1, 10, 100]} |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 0.615  Test set score: 0.604 |

**Random Forest**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | {'n\_estimators': [100, 1000]} |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 1.000  Test set score: 0.746 |

**AdaBoost**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | {'learning\_rate': np.arange(0.10, 1, 0.2),  'n\_estimators': [10, 50, 100, 1000],  'base\_estimator': [RandomForestClassifier(), ExtraTreesClassifier()]} |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 1.000  Test set score: 0.750 |

**Naive Bayes**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | / |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 1.000  Test set score: 0.750 |

**Kneighbors**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | {'n\_neighbors': [1,2,4,8,10],  'weights': ['uniform', 'distance'],  'algorithm': ['ball\_tree', 'kd\_tree', 'brute']} |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 0.650  Test set score: 0.592 |

**SGD**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | {'alpha': [1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1],  'learning\_rate': ['constant', 'optimal', 'invscaling', 'adaptive'],  'max\_iter': [10, 100, 1000], #np.ceil(10\*\*6 / n)  'shuffle' : [True],  'eta0' : [1]} |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 0.618  Test set score: 0.592 |

**MLP**

# Experiment 1

|  |  |
| --- | --- |
| **Dataset grootte** | Na segmentatie: 3408   * Forward\_180: 744 * Backward\_180: 832 * Jump\_slow: 1128 * Jump\_fast: 760 * Side\_swing: 1616 * Cross\_over: 568 |
| **Validatie set grootte** | 0.2 train test split |
| **window** | Trainingsset:   * 1 sec *overlap* (50%) |
| **Dimension reduction** | 6 |
| **Klassen** | Jump\_fast  Jump\_slow  Side\_swing  Cross\_over  Forward\_180  Backward\_180 |
| **gridsearch** | / |
| **Confusion matrix** |  |
| **accuraatheid** | Training set score: 0.670  Test set score: 0.643 |