Outline

Introduction & Background
Declaration of Transparency
Technology
Sprint Overview
Results
  ○ Software Engineering
  ○ Pre-Processing / Cleaning
  ○ Model Building & Statistical Evaluation
Key Learnings & Outlook
Appendix: Sprint Details
Introduction & Background
Parkinson's Disease

● Second most common neurodegenerative Disease worldwide

● Typical Symptoms:
  ○ Bradykinesia
  ○ Rigor
  ○ Rest Tremor
  ○ Gait Disorders
  ○ Non-motor symptoms: Depression, urinary dysfunction, autonomous disorders
Purpose & Goal - Specific Use Case

- Application in biomedical research: Parkinson’s Disease (PD)
- (Objective) analysis of PD Symptoms, e.g. rest tremor, is a key success factor in management of the disease
- Many scientific players to solve that problem
- Giant amounts of motion data
- Problem: No / little amount of labels

→ Solution: Fitting various machine learning models enabled by collection of large amounts of expert labels and their meta-information (e.g. confidence)

1 Ching et al 2017, Opportunities and obstacles for deep learning in biology and medicine, published online at biorxiv.org
Workflow

Patient(s) → Video → Segments/ Snippets → Labels by Raters → Aggregated Label → Machine Learning

Motion Data
Declaration of Transparency
HCA LAB

- Frontend
- Human Computation Concept
- Visualization

→ Live Demo

BD LAB

- Software Engineering
- API
- Database
- Preprocessing
- Feature Engineering
- Label Engineering
- Machine Learning
- Statistical Evaluation
Technology
Technology used

LANGUAGES
Python, R

FRAMEWORKS
Django rest framework

LIBRARIES/PACKAGES
SciPy, NumPy, Keras, MLR, XGBoost

DATABASE
mySQL
Sprint Overview

of the Big Data Lab
Sprint Overview

**Sprint 1:** Research and Concept Phase, Data Acquisition

**Sprint 2:** Database Setup / Inner Circle / Model Building 1

**Sprint 3:** Data Cleaning & Model Building 2 (Core)

**Sprint 4:** Model Building 3 / Refine Inner Circle

**Sprint 5:** Statistical Evaluation

CW17  CW20  CW23  CW26  CW29  CW31
Sprint 1: Research and Concept Phase

**Goals:**

- Definition of Core Technologies
- Acquisition & Preparation of Clinical Research Data
- Component Structure of Project
- Assignment of team roles and responsibilities
- Discussion with supervisors and implementation of feedback

**Achievements:**
Sprint 2: Database Setup / Inner Circle / Model Building 1

Goals:

- Prototype Inner Circle of the System
  - Process Input Data
  - Setup prototype database
  - Decision on API
- Research on Machine Learning Technology
- Evaluate Backend-Pipeline
- Implementation of first API version
- Authentication Service
Goals:

- **Connecting the components**
- **Cleaning Service**: Preprocessing of IMU data for Data Science analysis (e.g. read-in, Spectrogram, FFT, Noise reduction, DWT)
- **Machine Learning Component**: Implement Several Machine Learning Models using Test Data
  - Random Forests
  - KNN
  - Trees

Sprint 3: Data Cleaning & Model Building 2 (Core)
Sprint 3: Data Cleaning & Model Building 2 (Core)

Goals:

- Connecting the components
- **Cleaning Service**: Preprocessing of IMU data for Data Science analysis (e.g. read-in, Spectrogram, FFT, Noise reduction, DWT)
- **Machine Learning Component**: Implement Several Machine Learning Models using Test Data
  - Random Forests
  - KNN
  - Trees
Sprint 4: Model Building 3 / Refine Inner Circle

Goals:

- Fine-tune current machine learning models
- Add machine learning models, e.g.
  - Logistic / Linear / Spline Regression
  - Boosting (AdaBoost, XGBoost)
  - Support Vector Machine (SVM)
  - Multi Layer Perceptron (MLP)
  - Convolutional Neural Networks (CNN)
  - Long Short Term Memory (LSTM)
- Gathering more labels from more experts
- Refine Inner Circle & Iterations
Sprint 5: Wrap-Up / Statistical Evaluation

Goals:

- Refinement of ML techniques
  - parameter tuning
  - development of cost-sensitive approaches
- Statistical Evaluation of
  - Machine Learning Models
  - Label Statistics
  - Rater Statistics
- Write final report and presentation

Confusion Matrix and Statistics

<table>
<thead>
<tr>
<th>Reference</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 1 2 3 4</td>
</tr>
<tr>
<td>0</td>
<td>14 27 2 1 0</td>
</tr>
<tr>
<td>1</td>
<td>2 4 1 10 2 0</td>
</tr>
<tr>
<td>2</td>
<td>0 2 0 4 0 4</td>
</tr>
<tr>
<td>3</td>
<td>0 1 0 0 0 2</td>
</tr>
</tbody>
</table>

Overall Statistics

- Accuracy: 0.7422
- 95% CI: (0.6574, 0.8154)
- No Information Rate: 0.5547
- P-Value [Acc > NIR]: 9.076e-06
- Kappa: 0.595

McNemar's Test P-Value: NA

Statistics by Class:

<table>
<thead>
<tr>
<th></th>
<th>Class: 0 Class: 1 Class: 2 Class: 3 Class: 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>0.7324 0.7500 0.8333 0.5743 1.0000</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.8947 0.8152 0.9333 0.9819 0.9920</td>
</tr>
<tr>
<td>Pos Pred Value</td>
<td>0.9566 0.9366 0.9882 0.6667 0.6667</td>
</tr>
<tr>
<td>Neg Pred Value</td>
<td>0.7286 0.8929 0.9819 0.9754 1.0000</td>
</tr>
<tr>
<td>Precision</td>
<td>0.9566 1.0000 0.9882 0.5743 1.0000</td>
</tr>
<tr>
<td>Recall</td>
<td>0.7324 0.7500 0.8333 0.5743 1.0000</td>
</tr>
<tr>
<td>F1</td>
<td>0.8602 0.8706 0.6966 0.6138 0.6667</td>
</tr>
<tr>
<td>Prevalence</td>
<td>0.5547 0.2812 0.8937 0.0546 0.0156</td>
</tr>
<tr>
<td>Detection Rate</td>
<td>0.4862 0.2319 0.8712 0.4912 0.0156</td>
</tr>
<tr>
<td>Detection Prevalence</td>
<td>0.4862 0.2319 0.8712 0.4912 0.0156</td>
</tr>
<tr>
<td>Balanced Accuracy</td>
<td>0.8136 0.7826 0.8869 0.7774 0.9960</td>
</tr>
</tbody>
</table>
Results of the Big Data Lab
Software Engineering

- Spark/Flink vs Pandas
- SQL vs NoSQL
- Django vs Flask
Which Framework for Data Pipeline

- Initially: Apache Flink
  - Problem can be seen as Streaming task?

**Input**

- video files
- metadata
- IMU

**Framework Components**

- segment database updater
- sampler
- cleaning
- feature extraction
- label aggregator
- online learner
- current model updates
- evaluation
Which Framework for Data Pipeline

- Initially: Apache Flink
  - Problem can be seen as Streaming task?
  - lots of issues
    - not really designed to provide a REST API, more computing
    - more for simple operations (filtering, grouping, ..), integration with functions implemented for numpy arrays not native (periodigrams etc)
    - task is not really a streaming task (data comes in as one big batch)
    - ACID is important

Input
- video files
- metadata
- IMU

Processing stages:
- segment database updater
- sampler
- cleaning
- feature extraction
- label aggregator
- online learner
- current model updates
- evaluation
Which Framework for Data Pipeline

- Then: Apache Spark
  - Problem can be seen as batch processing task?
  - lots of issues
    - better, as it is more for batch processing
    - still, different emphasis
    - does not natively integrate with e.g. scipy, no strong ML model building algorithms
Which Framework for Data Pipeline

- Why one technology for everything?
  - API with SQL database (small amount of data, consistency is important)
  - Processing of IMU (larger amount of data, complex computation)
    - pandas/scipy
    - same technologies for model building
    - in case cluster is needed: data parallelism easy, using pyspark for a computation also possible
Django REST vs Flask

- Flask is simple to learn
  - great for simple tasks
  - great for complex tasks for advanced experts

- Learning Django takes time
  - scalable
  - clear styleguide
  - supports authentication, database Integration, authorization, serialization
## Django REST API

<table>
<thead>
<tr>
<th>Django REST API</th>
<th>url specs</th>
<th>Django REST framework</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampling service</td>
<td>label input service</td>
<td>data model</td>
</tr>
<tr>
<td>rater registration</td>
<td>login/ logout</td>
<td>url specs</td>
</tr>
<tr>
<td>data input services</td>
<td>label access service</td>
<td>authorization specs</td>
</tr>
</tbody>
</table>

```python
urlpatterns = [
    url(r'^subjects/$', views.subject_list),
    url(r'^videos/$', views.video_list),
    url(r'^videos/info/(?P<id>\d+)/$', views.video_info),
    url(r'^raters/$', views.rater_list),
    url(r'^rater/info/$', views.rater_info),
    url(r'^snippets/$', views.snippet_list),
    url(r'^labels/$', views.label_list),
    url(r'^random_snippets/?P<n>[\d]+/$', views.random_snippets),
    url(r'^register/rater/$', views.register_rater),
    url(r'^snippets/wr/?P<n>[\d]+/$', views.wo replacement_snippets),
]
```

```python
class Video(models.Model):
    video_id = models.AutoField(primary_key=True, unique=True)
    subject = models.ForeignKey(Subject, on_delete=models.CASCADE)
    start_unix = models.BigIntegerField(default=0)
    duration = models.PositiveIntegerField(default=0)
    path = models.CharField(max_length=100)
    name = models.CharField(max_length=25)

class Meta:
    ordering = ('video_id',)

class Snippet(models.Model):
    snippet_id = models.AutoField(primary_key=True, unique=True)
    video = models.ForeignKey(Video, on_delete=models.CASCADE)
    offset = models.PositiveIntegerField(default=0)
    duration = models.PositiveIntegerField(default=0)

class Meta:
    ordering = ('snippet_id',)
```
Overview of Decisions made

- Backend of Collex System:
  - Flask vs Django
- Framework for Data Preprocessing
  - Spark/Flink vs numpy/pandas
- Data Model
- API specs
- Storage of IMU data
- Storage of Video data
Preprocessing / Cleaning

Challenges:
- Missing Data
- Timestamp correction
- IMU data from different watches
- Matching IMU data with Video segment
Label Collection

- Raw Data from 19 Patients with Parkinson’s Disease (about 10min each), in total \( n \) segments of motion data, equals to about 11,400s of raw motion data
- accounts for 1,275 windows
- Labels from 4 Expert Raters for 254 snippets each, accounts to 1,016 collected labels
Raw IMU Data - Issues
Cleaning & Preprocessing

- Cleaning of Raw Data: Duplicate Timestamps, Transmission Errors, Matching
- Preprocessing of Raw Data (Discrete Wavelet Transform, PSD, Periodogram, Welch Method, Kalman Filtering)
Feature Engineering

- Energy of each segment
- Energy between 3-7 Hz (characteristic frequency band of rest tremor)
- Maximum energy
- Maximum energy between 3-7 Hz
- Power Spectral Density in the frequency bands between 0 and 31 Hz at a 0.5Hz step size
Label Engineering

- 1 Random Rater
- Mean
- Mode
- Aggregation of Agreement
- Cost-Sensitivity
Cost-sensitive learning

- Different ways to do cost-sensitive learning
  - Thresholding
  - Direct weighting
  - Rebalancing the data
- Rebalancing is done by rejection resampling
  - with a probability of \( 1 - \text{weight} \) reject the observation
  - otherwise add observation to data set
- How to choose weights?
Weighting strategies

\[ \text{weight} = \exp(-\text{std}^4) \]

\[ \text{weight} = \exp(-\text{std}^4) \]

\[ \text{weight} = (1 - \text{std}/2)_+ \]

\[ \text{weight} = 1_{\text{std}=0} \]
Model Building & Statistical Evaluation

Challenges:
- class imbalance
- label noise
- few data
Interrater Agreement
Comparison of different machine learning approaches

<table>
<thead>
<tr>
<th>Model</th>
<th>1RR</th>
<th>1ER</th>
<th>SSD</th>
<th>Mod</th>
<th>Avg</th>
<th>Agr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Regression*</td>
<td>-</td>
<td>-</td>
<td>.61</td>
<td>-</td>
<td>.58</td>
<td>1.06</td>
</tr>
<tr>
<td>Log. Regr. L1/L2**</td>
<td>.31</td>
<td>.31</td>
<td>.48</td>
<td>.31</td>
<td>-</td>
<td>.71</td>
</tr>
<tr>
<td>Random Forest**</td>
<td>.68</td>
<td>.64</td>
<td>.61</td>
<td>.69</td>
<td>.41*</td>
<td>.84</td>
</tr>
<tr>
<td>AdaBoost**</td>
<td>.64</td>
<td>.72</td>
<td>.73</td>
<td>.66</td>
<td>-</td>
<td>.81</td>
</tr>
<tr>
<td>XGBoost**</td>
<td>.63</td>
<td>.69</td>
<td>-</td>
<td>.67</td>
<td>-</td>
<td>.83</td>
</tr>
<tr>
<td>SVM**</td>
<td>.51</td>
<td>.61</td>
<td>.58</td>
<td>.51</td>
<td>-</td>
<td>.71</td>
</tr>
<tr>
<td>FF Neural Network**</td>
<td>.42</td>
<td>.51</td>
<td>-</td>
<td>.40</td>
<td>.42</td>
<td>.53</td>
</tr>
<tr>
<td>CNN**</td>
<td>.52</td>
<td>.52</td>
<td>-</td>
<td>.40</td>
<td>.41</td>
<td>.53</td>
</tr>
<tr>
<td>LSTM (raw)**</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>.34</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3: Overview of different Model Performances. Abbreviations: SLA = Best Performance Small Label Amount, LLA = Best Performance Large Label Amount, 1RR = 1 Random Rater Baseline, 1ER = 1 Expert Rater, SSD = Small Standard Deviation, Mod = Mode, Avg = Average / Mean, Agr = Agreement, WCS = Weighted / Cost-Sensitive Approach, *MSE, **Accuracy, in **bold**: best performance
How do raters influence results of the ML model?

- in reality: train and evaluate on labels of **one** rater
- → high variance in rater’s results
- results of one model (same hyperparameters, same evaluation method)

<table>
<thead>
<tr>
<th></th>
<th>Student</th>
<th>Doctor 1</th>
<th>Expert</th>
<th>Doctor 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF classification*</td>
<td>0.688</td>
<td>0.742</td>
<td>0.652</td>
<td>0.640</td>
</tr>
</tbody>
</table>

- Which setting to choose?
- How to find something near to the “ground truth”?* evaluated by 10-fold CV
Linear / Logistic / Spline Regression

Linear Regression (MSE 0.558)

Spline Regression (MSE 0.468)
SVM - without Tuning

- Kernel SVMs: Gaussian Kernel
- Without Tuning: ACC = 0.477 (Mode)
SVM - Tuning

- With Tuning: ACC = 0.529
  (Cost-Parameter C and Kernel Width by Grid Search)
SVM - Challenges for Dataset

- **Number of Features**
  - no accuracy gain
- **Class Imbalance**
  - SMOTE: ACC = 0.577
AdaBoost

Resample Result
Task: mode
Learner: classif.ada.multiclass
Aggr perf: acc.test.mean=0.654, mmce.test.mean=0.346
Runtime: 79.4886
XGBoost

- Classification with tree boosters (acc 0.673)
- Regression with linear boosters (mse 0.39)
- Tuning by grid search
Random Forest for mode

Confusion Matrix and Statistics

<table>
<thead>
<tr>
<th>Prediction</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>232</td>
<td>85</td>
<td>27</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>182</td>
<td>300</td>
<td>31</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>37</td>
<td>42</td>
<td>100</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>7</td>
<td>27</td>
<td>67</td>
</tr>
</tbody>
</table>

Overall Statistics

- Accuracy: 0.6941
- 95% CI: (0.668, 0.7198)
- No Information Rate: 0.3749
- P-Value [Acc > NIR]: < 2e-16
- Kappa: 0.5612
- McNemar's Test P-Value: 0.05879

Statistics by Class:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Class: 0</th>
<th>Class: 1</th>
<th>Class: 2</th>
<th>Class: 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity</td>
<td>0.646</td>
<td>0.6955</td>
<td>0.7792</td>
<td>0.7126</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.8429</td>
<td>0.8395</td>
<td>0.8539</td>
<td>0.96534</td>
</tr>
<tr>
<td>Pos Pred Value</td>
<td>0.7345</td>
<td>0.6955</td>
<td>0.7545</td>
<td>0.62037</td>
</tr>
<tr>
<td>Neg Pred Value</td>
<td>0.9226</td>
<td>0.8865</td>
<td>0.9159</td>
<td>0.9785</td>
</tr>
<tr>
<td>Precision</td>
<td>0.7345</td>
<td>0.6955</td>
<td>0.7545</td>
<td>0.62037</td>
</tr>
<tr>
<td>Recall</td>
<td>0.6048</td>
<td>0.6955</td>
<td>0.7392</td>
<td>0.7126</td>
</tr>
<tr>
<td>F1</td>
<td>0.7140</td>
<td>0.6955</td>
<td>0.6667</td>
<td>0.6780</td>
</tr>
<tr>
<td>Prevalence</td>
<td>0.3758</td>
<td>0.3451</td>
<td>0.2078</td>
<td>0.07216</td>
</tr>
<tr>
<td>Detection Rate</td>
<td>0.2684</td>
<td>0.2400</td>
<td>0.1412</td>
<td>0.0525</td>
</tr>
<tr>
<td>Detection Prevalence</td>
<td>0.3545</td>
<td>0.3451</td>
<td>0.2157</td>
<td>0.08471</td>
</tr>
<tr>
<td>Balanced Accuracy</td>
<td>0.7720</td>
<td>0.7675</td>
<td>0.7926</td>
<td>0.8480</td>
</tr>
</tbody>
</table>

RF Feature Importance

mode (19 features), filter = information.gain
Random Forest cost-sensitive
Cost-Sensitive Approach & Agreement Aggregation

- tested with Random Forest
- Full agreement portion: 19.4%

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Small Dataset</td>
<td>.84</td>
</tr>
<tr>
<td>(b) Large Dataset</td>
<td>.62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>RF 1 randomly picked rater</td>
<td>.65</td>
</tr>
<tr>
<td>RF mode of all raters</td>
<td>.69</td>
</tr>
<tr>
<td>RF /w cost-sensitivity</td>
<td>.80</td>
</tr>
</tbody>
</table>
Random Forest regression

- MSE of 0.417
- Even there weighting according to raters confidence achieved good results
Feed Forward NN

- Lower performance than classical machine learning methods
- Aggregation approach slightly improved performance
Deep Learning: CNN & LSTM

- too little data
- difficult to choose hyperparameters
- method not as robust, sensitive to variance in labels
Class Imbalance

→ SMOTE 5nn
→ class weights
Conceptual Considerations (1)
Conceptual Considerations (2)

Feature Engineering
- Manual/Automatic feature engineering
- dimensionality reduction
- filtering
- data augmentation
- outlier removal
- balancing of classes
- ...

Label Engineering
Conceptual Considerations (3)
Summary & Key Learnings

- Aggregation of multiple labels is a very helpful method to infer ground truth
- Aggressive Sample Weighting is gaining accuracy
- Cost-Sensitive Approach, which takes the costs for each label within one segment into account is gaining accuracy
- Large Inter-Rater Disagreement / Variability
- Inter-Rater Variability is very valuable in inferring ground truth
- Quality of Labels over Quantity of labels for medical data?
Outlook

- Add Active Learning Component
- Refining the current use case
  - Collecting more labels from more experts
  - Collecting more data (more severe cases)
- Other use cases
is a formalized framework system to collect
large-scale, high-quality, correctly labeled training
data from experts for machine learning applications in
the biomedical domain
Team Members & Roles

Ahmet Gündüz: Software Development, Preprocessing, Deep Learning


Julia M Moosbauer: Software Development, Statistics, Model Building

Franz MJ Pfister: Software Development, Model Building, Domain Knowledge
Thank you for your attention.

COLLEX // A-Team
Ahmet Gündüz
Gunnar CS Koenig
Julia M Moosbauer
Franz MJ Pfister
Sprint Overview

Detailed Protocol
<table>
<thead>
<tr>
<th>Tasks of Sprint 1</th>
<th>MRP</th>
<th>Description</th>
<th>MRP %</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>all</td>
<td>Conceptual Considerations, Definition of Objectives, Targets, Potential</td>
<td>20%</td>
</tr>
<tr>
<td>Use Case Definition</td>
<td>FP</td>
<td>Definition of different biomedical use cases, concepts of implementation</td>
<td>5%</td>
</tr>
<tr>
<td>Software System Design</td>
<td>all</td>
<td>Define System Design, Evaluate Frameworks (Spark, Flink, python vs. Scala, Pandas)</td>
<td>15%</td>
</tr>
<tr>
<td>Research on Methods</td>
<td>all</td>
<td>Study literature on annotation collection services, sampling, aggregation, system design, etc.</td>
<td>15%</td>
</tr>
<tr>
<td>Database Evaluation &amp; Selection</td>
<td>AG/GK</td>
<td>SQL vs. NoSQL, SQLite vs. MySQL vs. PostgreSQL etc.,</td>
<td>5%</td>
</tr>
<tr>
<td>Specify Components</td>
<td>all</td>
<td>Consider design and functionality of components that define the COLLEX system</td>
<td>5%</td>
</tr>
<tr>
<td>Data Segmentation</td>
<td>JM/FP</td>
<td>Develop strategy to split data into specific segments</td>
<td>7.5%</td>
</tr>
<tr>
<td>Object Oriented Database Model</td>
<td>all</td>
<td>Develop the Database Model</td>
<td>5%</td>
</tr>
<tr>
<td>Segment Decision Borders Flagging</td>
<td>FP</td>
<td>Flag borders for stable segments</td>
<td>5%</td>
</tr>
<tr>
<td>Aggregation Concept</td>
<td>JM/GK</td>
<td>Meta-Information, Computation, Parameters, Concepts</td>
<td>5%</td>
</tr>
<tr>
<td>Data Pipeline Tech Eval. &amp; Selection</td>
<td>AG/GK</td>
<td>Assess &amp; compare different core technologies (e.g. Flink / Spark / Python)</td>
<td>10%</td>
</tr>
<tr>
<td>API Evaluation &amp; Selection</td>
<td>AG/GK</td>
<td>Assess &amp; compare different ways to build an API, different technologies (Flask, Django, Java Play Framework)</td>
<td>5%</td>
</tr>
<tr>
<td>Concept Decisions</td>
<td>all</td>
<td>Discussion and Decision on Concepts</td>
<td>2.5%</td>
</tr>
<tr>
<td>Task</td>
<td>MRP</td>
<td>Description</td>
<td>MRP Percentage</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Inner Loop Concept &amp; Implementation</td>
<td>all</td>
<td>Define Concept of the Inner Loop of the Technology Stack, first version of implementation</td>
<td>15 %</td>
</tr>
<tr>
<td>Django Tutorial</td>
<td>AG/GK</td>
<td>Get to learn Django rest framework, study tutorial material</td>
<td>10 %</td>
</tr>
<tr>
<td>Sampling Service</td>
<td>GK/JM</td>
<td>First version of sampling service, randomly draw from segments</td>
<td>10 %</td>
</tr>
<tr>
<td>Aggregation Service</td>
<td>JM</td>
<td>First version of aggregation service</td>
<td>5 %</td>
</tr>
<tr>
<td>Frontend Connection</td>
<td>JM</td>
<td>Connecting the first versions of the components with the frontend</td>
<td>5 %</td>
</tr>
<tr>
<td>Data model</td>
<td>AG/GK</td>
<td>ERM, refinement of data model, decisions on data types and representation of categorical variables, definition of Django models</td>
<td>5 %</td>
</tr>
<tr>
<td>Video storage</td>
<td>FP</td>
<td>Folder structure, accessing videos</td>
<td>5 %</td>
</tr>
<tr>
<td>Data Segmentation</td>
<td>FP</td>
<td>Cutting data (video / motion data) into pieces according to flagging</td>
<td>5 %</td>
</tr>
<tr>
<td>ML Model Selection</td>
<td>all</td>
<td>Research on Model Selection and how to approach this kind of data</td>
<td>15 %</td>
</tr>
<tr>
<td>Build Dummy REST API</td>
<td>AG/GK</td>
<td>Implement preliminary version of the REST API</td>
<td>5 %</td>
</tr>
<tr>
<td>API authentication</td>
<td>AG</td>
<td>Adding authentication to API</td>
<td>5 %</td>
</tr>
<tr>
<td>Specify Statistics</td>
<td>all</td>
<td>Decide on specific statistics that should be measured / computed</td>
<td>5 %</td>
</tr>
<tr>
<td>Database Connection</td>
<td>AG/GK</td>
<td>Connect the system to the database and test connection; first queries</td>
<td>10 %</td>
</tr>
<tr>
<td>Tasks of Sprint 3</td>
<td>MRP</td>
<td>Description</td>
<td>MRP Percentage</td>
</tr>
<tr>
<td>-----------------------------------------------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Inner Loop Refinement</td>
<td>all</td>
<td>Iterative improvement of the inner loop, adding several views, more efficient sampling</td>
<td>10%</td>
</tr>
<tr>
<td>Preprocessing</td>
<td>AG/FP/GK</td>
<td>Reading in the motion data; addressing the challenges of different data formats</td>
<td>5%</td>
</tr>
<tr>
<td>Motion Data Cleaning pt. 1</td>
<td>AG/FP/GK</td>
<td>Addressing the problems of missing data lines, Matching of Video / IMU data</td>
<td>10%</td>
</tr>
<tr>
<td>Segmentation of Snippets</td>
<td>JM</td>
<td>Addressing the problems of different snippet lengths</td>
<td>5%</td>
</tr>
<tr>
<td>Frontend Connection</td>
<td>JM/FP</td>
<td>Connecting the first versions of the components with the frontend</td>
<td>5%</td>
</tr>
<tr>
<td>Tuning Window Size</td>
<td>AG/GK/FP</td>
<td>Structuring ML problem, extraction of segments, reviewing window size choice (information vs overfitting tradeoff)</td>
<td>10%</td>
</tr>
<tr>
<td>Dimensionality Reduction</td>
<td>AG</td>
<td>Testing of dimensionality red. methods, performance measures, evaluate PCA, etc.</td>
<td>5%</td>
</tr>
<tr>
<td>Transformation of Data</td>
<td>AG/GK</td>
<td>Implementation of FFT, Welch, Kalman Filter and Periodogram, etc.</td>
<td>15%</td>
</tr>
<tr>
<td>Random Forest</td>
<td>JM/FP</td>
<td>First version of Random Forest</td>
<td>10%</td>
</tr>
<tr>
<td>Trees</td>
<td>JM/FP</td>
<td>Implementation of trees</td>
<td>5%</td>
</tr>
<tr>
<td>K Nearest Neighbor</td>
<td>JM/FP</td>
<td>Implementation and first round of tuning of KNN</td>
<td>5%</td>
</tr>
<tr>
<td>Further Considerations</td>
<td>all</td>
<td>Handling of Class Imbalance, weighing approaches, cost-sensitive approaches</td>
<td>5%</td>
</tr>
<tr>
<td>Data insertion scripts and API Tests</td>
<td>GK/AG</td>
<td>functionality to push data from csv into django using API prototype and to make rest api calls using python/requests package</td>
<td>10%</td>
</tr>
<tr>
<td>Task</td>
<td>MRP</td>
<td>Description</td>
<td>Percentage</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>------------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Inner Circle Finish</td>
<td>AG/GK</td>
<td>Iterative improvement of the inner loop, adding several views, more efficient</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sampling, finishing all components of the inner circle</td>
<td></td>
</tr>
<tr>
<td>Label Collection</td>
<td>JM/FP</td>
<td>Collecting labels from experts, real-time test of the system</td>
<td>10%</td>
</tr>
<tr>
<td>Cleaning pt. 2 and preprocessing</td>
<td>AG/GK</td>
<td>Addressing Cleaning and preprocessing of data, see above problems of missing</td>
<td>15%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data lines</td>
<td></td>
</tr>
<tr>
<td>Linear / Logistic Regression</td>
<td>FP</td>
<td>Implementation of Linear Regression as well as L1 / L2 Logistic Models</td>
<td>10%</td>
</tr>
<tr>
<td>Spline Regression</td>
<td>FP</td>
<td>Implementation of Spline Regression Models</td>
<td>5%</td>
</tr>
<tr>
<td>Boosting</td>
<td>JM</td>
<td>Implementation of AdaBoost / XGBoost including evaluation methods</td>
<td>10%</td>
</tr>
<tr>
<td>Random Forest</td>
<td>JM/FP</td>
<td>Implementation of Random Forest including evaluation methods</td>
<td>10%</td>
</tr>
<tr>
<td>FNN / Deep Learning</td>
<td>AG/GK</td>
<td>Implementation of Feed-Forward Neural Network and testing of deep learning</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>methods</td>
<td></td>
</tr>
<tr>
<td>Aggregation Methods</td>
<td>all</td>
<td>Evaluation and implementation of different aggregation methods, testing of</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cost-sensitive approaches</td>
<td></td>
</tr>
<tr>
<td>Statistical Evaluation</td>
<td>all</td>
<td>Statistical Evaluation of all Models</td>
<td>5%</td>
</tr>
<tr>
<td>Research on Security</td>
<td>GK/AG</td>
<td>Researching methods to improve security: Is Django runserver safe? Transmission</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>of Video? Local concept short-term, VPN and MFA long-term</td>
<td></td>
</tr>
<tr>
<td>Tasks of Sprint 5</td>
<td>MRP</td>
<td>Description</td>
<td>%</td>
</tr>
<tr>
<td>------------------------------------------------------</td>
<td>------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SVM</td>
<td>FP</td>
<td>Implementation of SVMs and Tuning</td>
<td>10 %</td>
</tr>
<tr>
<td>Tuning</td>
<td>JM</td>
<td>Further tuning of consisting models (Boosting, Random Forests)</td>
<td>10 %</td>
</tr>
<tr>
<td>Implementation of RNN</td>
<td>AG/GK</td>
<td>Preparation of raw data and coding RNN model</td>
<td>10 %</td>
</tr>
<tr>
<td>Tuning and Cross Validation of NN Approaches</td>
<td>AG/GK</td>
<td>Writing code for 10-fold Cross Validation on the data, evaluating different</td>
<td>10 %</td>
</tr>
<tr>
<td></td>
<td></td>
<td>model/parameter/aggregation methods, visualization of results</td>
<td></td>
</tr>
<tr>
<td>Cost-Sensitive Approach</td>
<td>JM</td>
<td>Evaluating for Cost-Sensitive Multiclass Classification methods</td>
<td>10 %</td>
</tr>
<tr>
<td>SMOTE algorithm</td>
<td>JM</td>
<td>Implementation of SMOTE algorithm for imbalanced multiclass problems</td>
<td>10 %</td>
</tr>
<tr>
<td>Evaluation of Rater’s impact</td>
<td>all</td>
<td>Evaluating impact of high-quality labels</td>
<td>5 %</td>
</tr>
<tr>
<td>Write Final Report</td>
<td>FP</td>
<td>Final Report</td>
<td>15 %</td>
</tr>
<tr>
<td>Deploying on AWS server</td>
<td>GK/AG</td>
<td>Deploying the REST API on AWS server</td>
<td>5 %</td>
</tr>
<tr>
<td>Revision of Final Report</td>
<td>all</td>
<td>Revision of final report</td>
<td>10 %</td>
</tr>
<tr>
<td>Final Presentation</td>
<td>all</td>
<td>Preparation of Final Presentation</td>
<td>10 %</td>
</tr>
<tr>
<td>Statistical evaluation of all Models</td>
<td>all</td>
<td>Final statistical comparison and evaluation</td>
<td>10 %</td>
</tr>
</tbody>
</table>