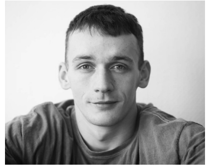
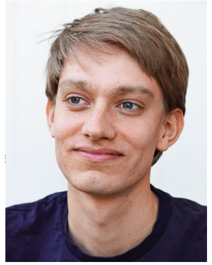
# About us

## Albert Skovgaard Bisgaard



- Education
  - MSc. Mathematical Modelling at DTU, 2021 – 2023
  - MSc. Mathematical Engineering exchange at KU Leuven, 2022 spring
  - BSc. Mathematics and Technology at DTU, 2018 – 2021
- Experience
  - Staff scientist at Department of Neuroscience, University of Copenhagen, current
  - Teaching assistant *Advanced Engineering Mathematics 1* at DTU, current
  - Analyst at Zealth Consultancy, 2019-2021
- Areas of interest
  - Time-varying systems (nonlinear dynamics, time series analysis), signal processing
  - Statistical modelling and multivariate statistics
  - Machine learning (neural networks, support vector machines, clustering techniques)
  - Optimization (dynamic and static), operational research

## Magnus Hamann Poulsen

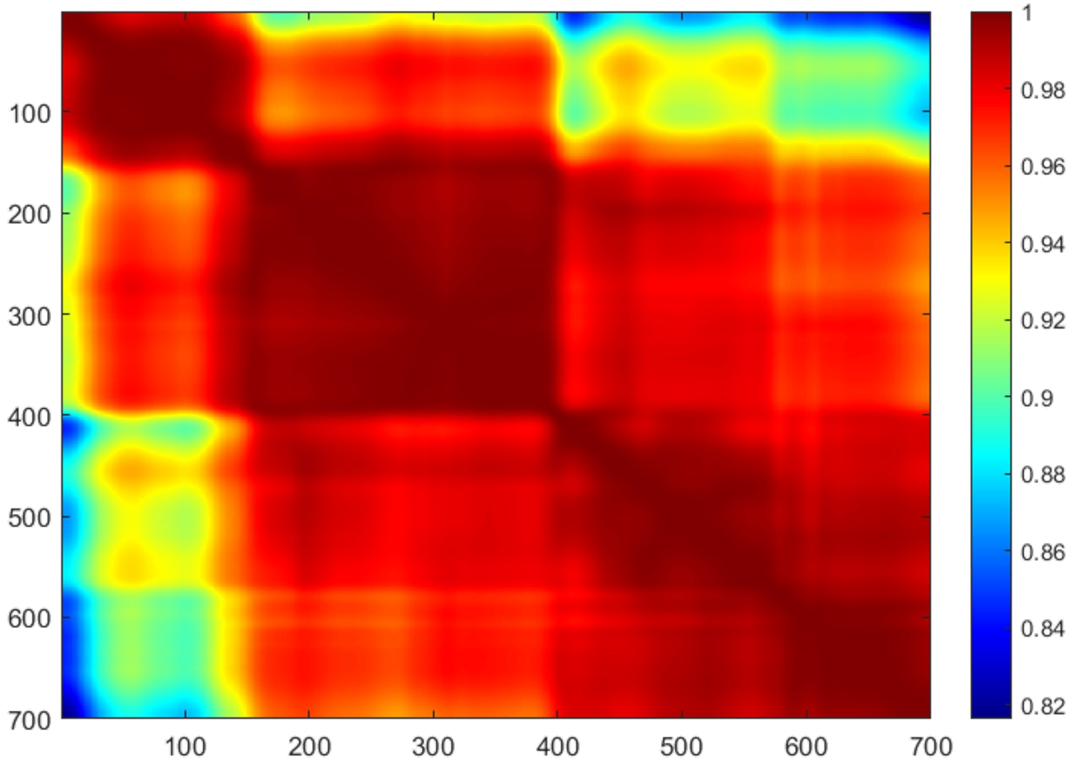


- Education
  - BSc. Mathematics and Technology 2018 – 2021
  - MSc. Mathematical modelling 2021 – 2023
- Experience
  - Student assistant at Global Cost Estimation Topsoe, current
  - Student assistant at Barrowa, 2017-2020
- Areas of interest
  - Digital twins and data engineering
  - Model Predictive control (control of nonlinear dynamics)
  - Time series modelling and prediction
  - Mathematical Optimization
  - Programming of mathematical software
  - Outside study: Athletics and sports team management

# Challenges

High dimensionality and many observations require much computational power

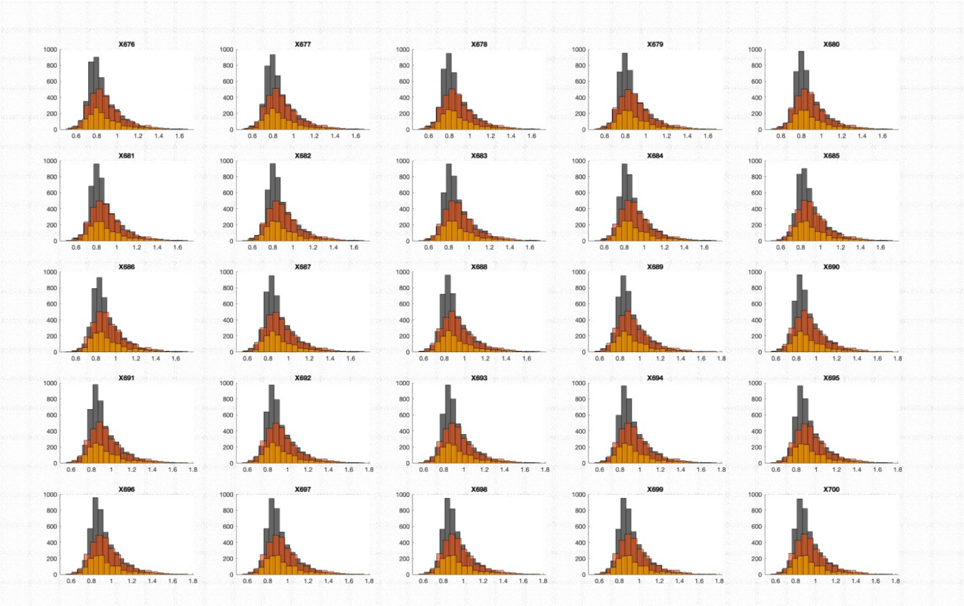
Multidimensionality poses visualization issues




Correlation matrix

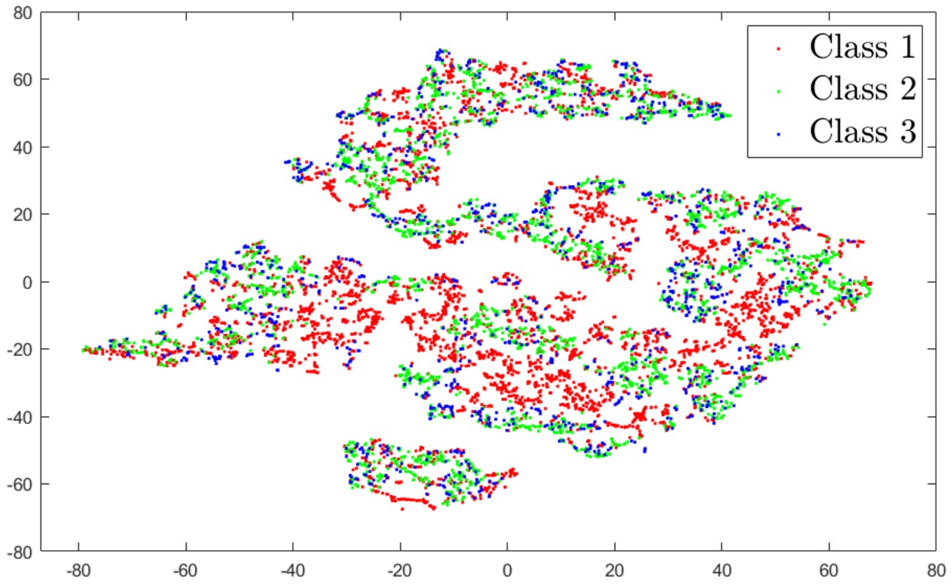
# Challenges

- Distributions of wavelengths are not easily separable (variable 691-700 seen in plot).



Histograms

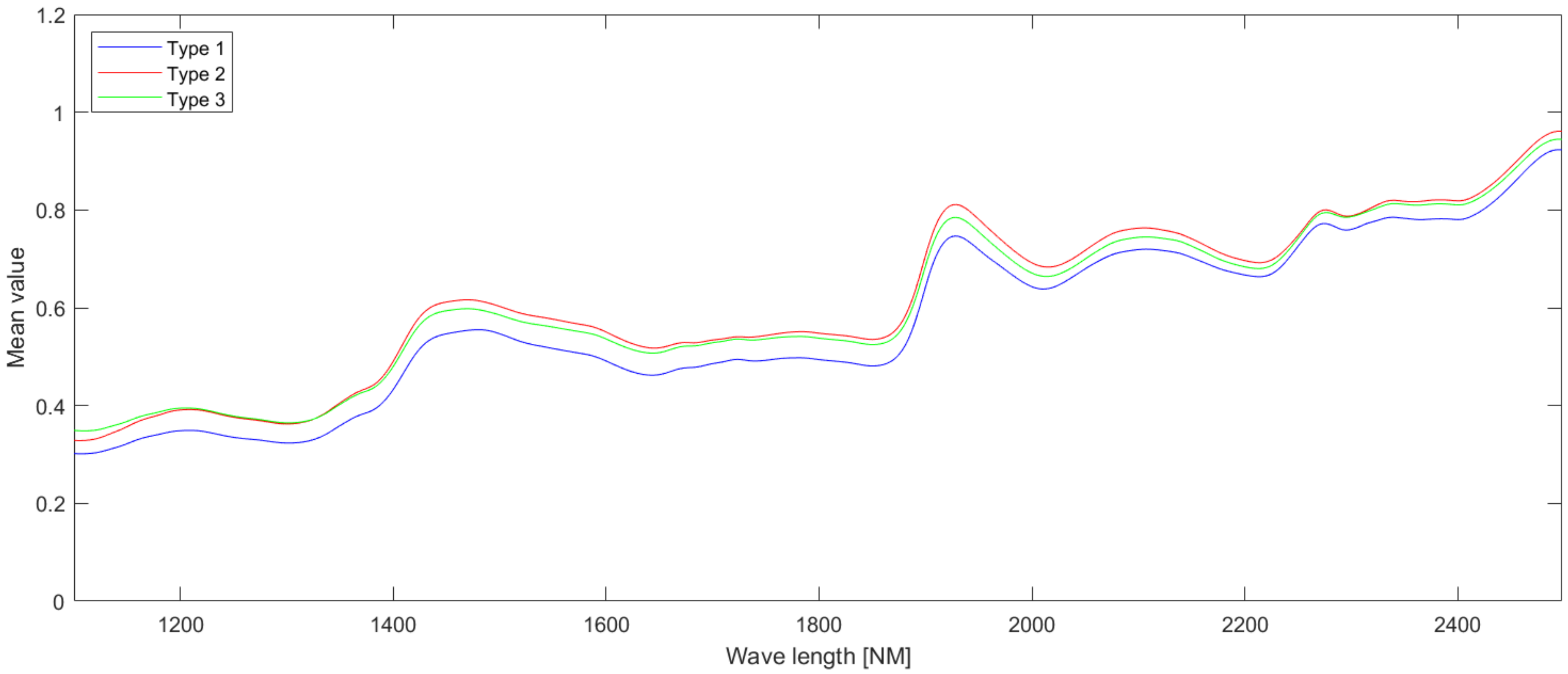
- Non-regular clusters




TSNE plot

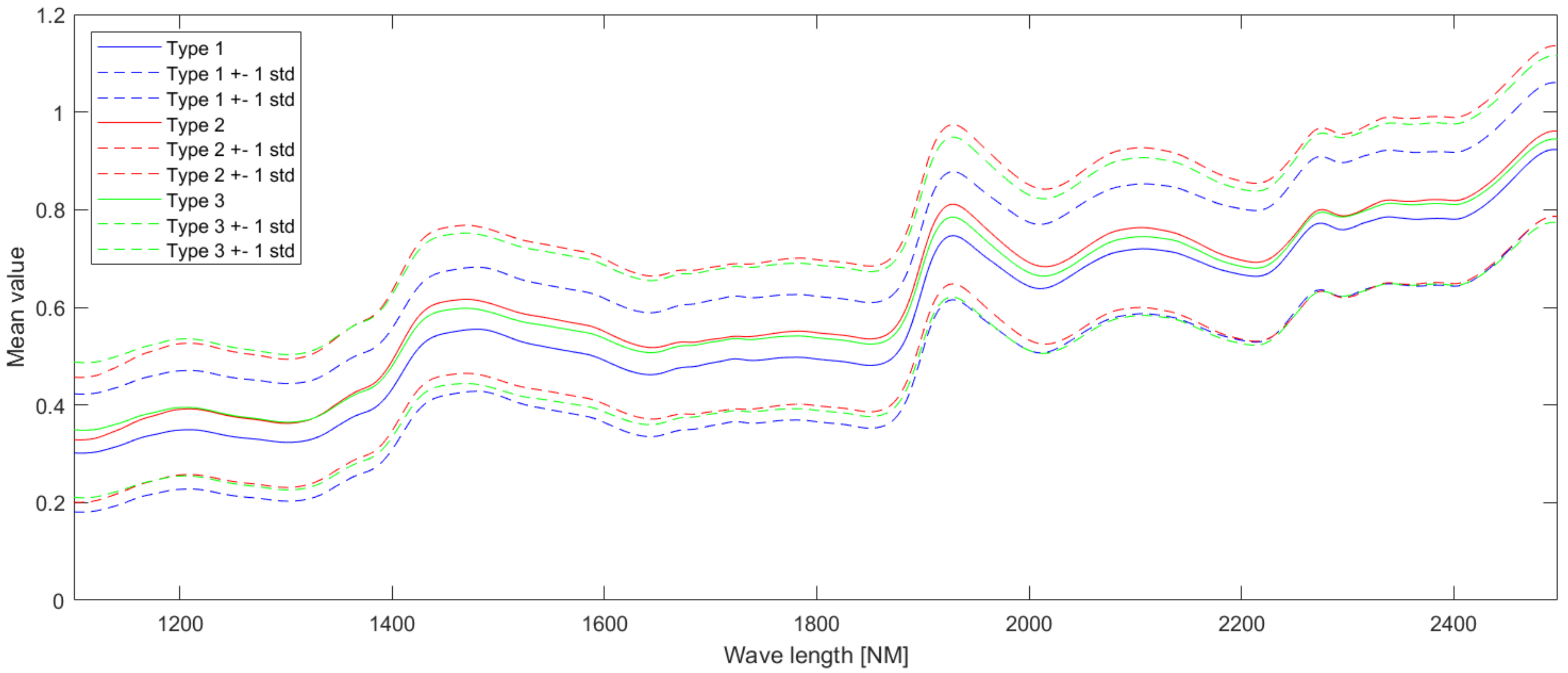
# Using a first order moment model

Aim to find the simplest model, which is still useful



# Using a first order moment model

Aim to find the simplest model, which is still useful



# Using a first order moment model

Aim to find the simplest model, which is still useful

# Solution structure

**Principal component analysis**

- Normalization
- Linear vs. kernel PCA
- Reconstruction of data
- Optimization of number of components

**Least-square support vector machine**

- Min-max transformation
- Choice of kernel
- Tuning of hyper-parameters
- Evaluation of results

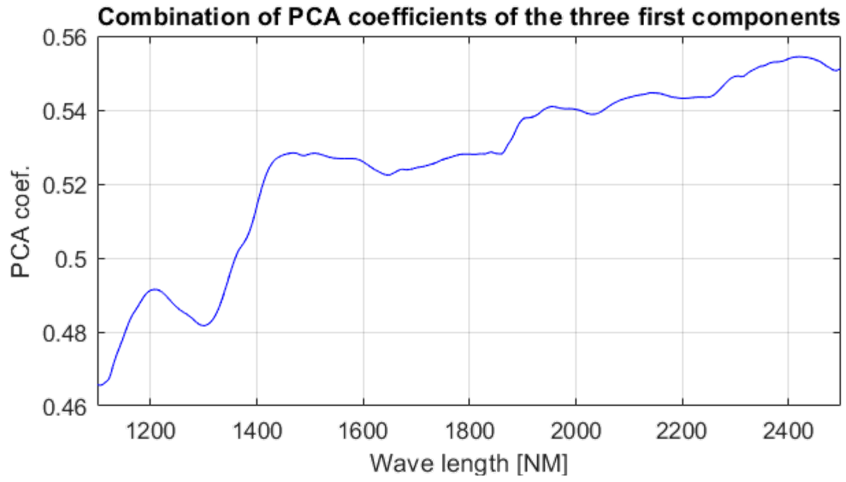
**Correction for prior distribution**

- Computing priors
- Comparison to obtained classifications correcting wrt. to prior knowledge

```python
# %%
from sklearn.preprocessing import StandardScaler
#X_train, X_test, y_train, y_test = train_test_split(data[:,4:], data[:,3], test_size=1/5, random_state=2020)


# Scaling features (from sklearn)
scaler = MinMaxScaler()
scaler.fit(X_train_proj)
X_tr_norm = scaler.transform(X_train_proj)
X_ts_norm = scaler.transform(X_test_proj)


print('Gaussian kernel:')

#gammaVal = [130,150]

#for i in gammaVal:

#    lssvc = LSSVC(gamma=i, kernel='rbf', sigma=1.2) # Class instantiation
#    lssvc.fit(X_tr_norm, Y_train) # Fitting the model
#    y_pred = lssvc.predict(X_ts_norm) # Making predictions with the trained model
#    acc = accuracy_score(dummie2multilabel(Y_val1), dummie2multilabel(y_pred)) # Calculate Accuracy
#    print('acc_test = ', acc, ' for gamma = ', i, '\n')

sigmaVal = [1.2]

gam = 140
sig = 1.2

for i in sigmaVal:

    lssvc = LSSVC(gamma=140, kernel='rbf', sigma=i) # Class instantiation
    lssvc.fit(X_tr_norm, Y_train) # Fitting the model
    y_pred = lssvc.predict(X_ts_norm) # Making predictions with the trained model
    #acc = accuracy_score(dummie2multilabel(Y_test), dummie2multilabel(y_pred)) # Calculate Accuracy
    #print('acc_test = ', acc, ' for sigma = ', i, '\n')


#lssvc = LSSVC(gamma=gam, kernel='rbf', sigma=sig) # Class instantiation
#lssvc.fit(X_tr_norm, Y_train,  sample_weights = [1,1,1]) # Fitting the model
#y_pred = lssvc.predict(X_ts_norm) # Making predictions with the trained model
#acc = accuracy_score(dummie2multilabel(Y_test), dummie2multilabel(y_pred)) # Calculate Accuracy
#print('acc_test = ', acc, ' for sigma = ', '\n')

# %% KPCA

#Applying Kernel PCA
from sklearn.decomposition import KernelPCA
kpca = KernelPCA(n_components = 50, kernel = 'rbf')
X_train = kpca.fit_transform(X_tr_norm)
X_test = kpca.transform(X_ts_norm)




lssvc = LSSVC(gamma=gam, kernel='rbf', sigma=sig) # Class instantiation
lssvc.fit(X_train, Y_train) # Fitting the model


y_pred = lssvc.predict(X_test) # Making predictions with the trained model
acc = accuracy_score(dummie2multilabel(Y_test), dummie2multilabel(y_pred)) # Calculate Accuracy
print('acc_test = ', acc, ' for n_comp = ', '\n')
```

# Principal component analysis

- Normalization
- Linear vs. kernel PCA
  - KPCA showed unstable estimates and worse in terms of MSE
- Reconstruction of data (denoising)
- Optimization of number of components
  - Optimized linear PCA reduced noise and improve MSE
- Small wavelengths have the least explanatory power

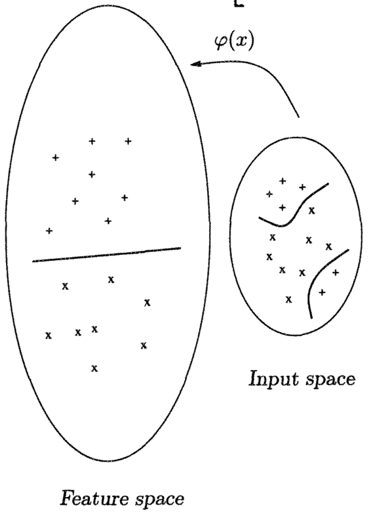


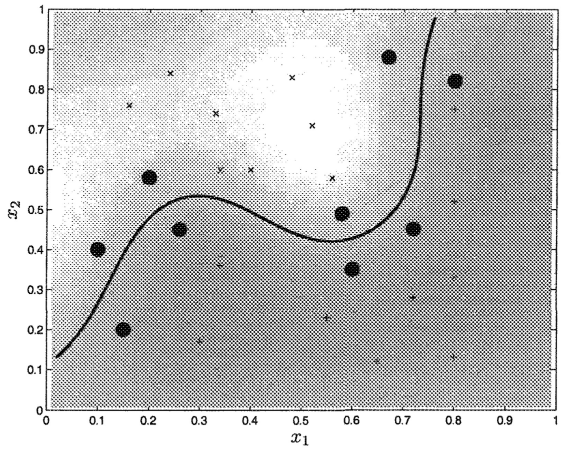
# Least-square support vector machine

- Min-max standardization
- Construction of optimal hyperplane separating classes
- Constrained quadratic optimization
  - Construction of Lagrangian
  - Free of charge change between primal and dual space
- Choice of kernel
  - Gaussian or epanechnikov (for improved performance)
- Tuning of hyper-parameters
  - Sparcity knob and kernel bandwidth
- Construction of validation and test set and evaluation of results

$$\left[\begin{array}{lll} \boxed{\mathrm{P}}: & \min_{w,b,\xi} J_{\mathrm{P}}(w,\xi) = & \frac{1}{2}w^T w + c\sum_{k=1}^{N}\xi_k \\ & \text{such that} & y_k[w^T x_k + b] \geq 1-\xi_k, \quad k=1,\ldots,N \\ & & \xi_k \geq 0, \quad k=1,\ldots,N \end{array}\right]$$

$$\left[\begin{array}{lll} \boxed{\mathrm{D}}: & \max_{\alpha} J_{\mathrm{D}}(\alpha) = & -\frac{1}{2}\sum_{k,l=1}^{N} y_k y_l K(x_k,x_l)\,\alpha_k\alpha_l + \sum_{k=1}^{N}\alpha_k \\ & \text{such that} & \sum_{k=1}^{N}\alpha_k y_k = 0 \\ & & 0 \leq \alpha_k \leq c, \; k=1,\ldots,N. \end{array}\right]$$




Source: Johan Suykens, Least-square support vector machines, 2002

# Correction for prior distribution

- Exploit the problem structure in the classification
  - 3 replications x 32 scans
- Choosing a decision heuristic to classify samples
- Given the data collection the model is biased

| | |
|---|---|
| Treated Wood | 51.7% |
| Virgin Wood | 32.6% |
| Disposal Wood | 15.7% |

Source: Åsmund Rinnan

- **Decision heuristic**: For a sample, compute the fractional split from the LS-SVM predictions. Assign the sample to the class with the largest increase with respect to the priors.

# Correction for prior distribution

- Example of decision heuristic

| Wood type | Number of classifications | Percentage | Percentage points above prior |
|---|---|---|---|
| Treated Wood | 53 | 55.2% | 3.5 |
| Virgin Wood | 10 | 10.5% | -22 |
| Disposal Wood | 33 | 34.3% | 19 |

Source: Åsmund Rinnan

# Questions

**Albert Skovgaard Bisgaard**
Mathematical Modelling and Computation

**Magnus Hamann Poulsen**
Studying Mathematical Modelling and Computation at Technical University of Den...