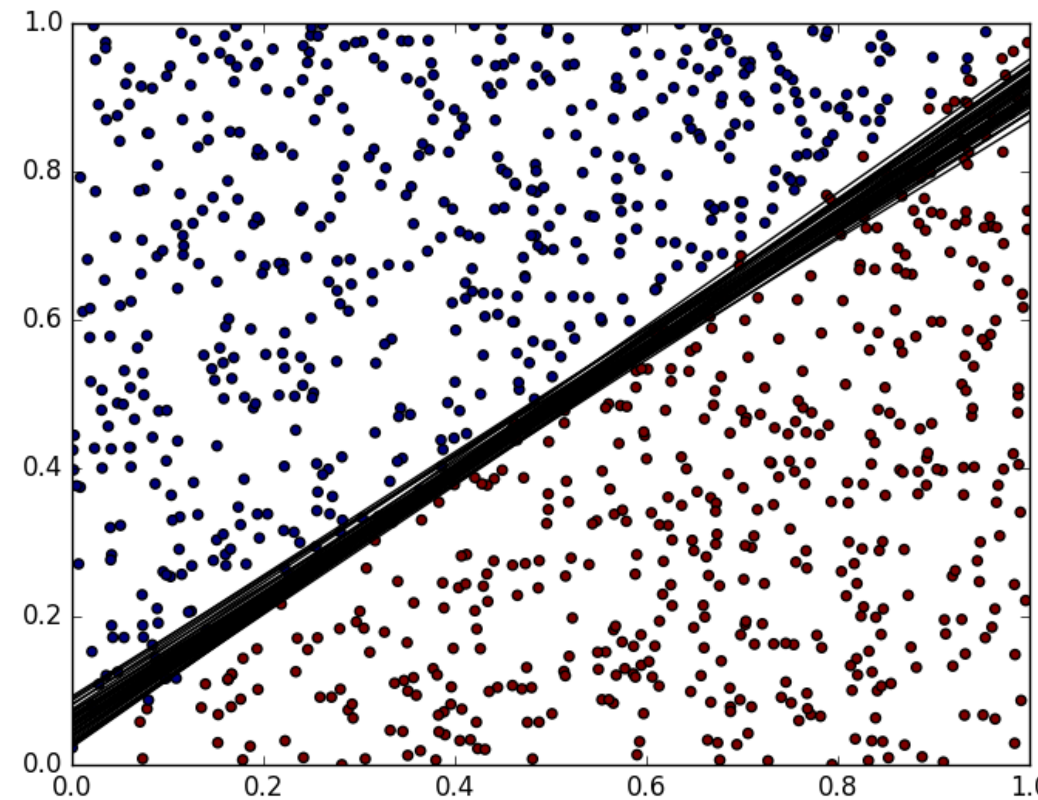
Bootstrapping the Support Vector Machine

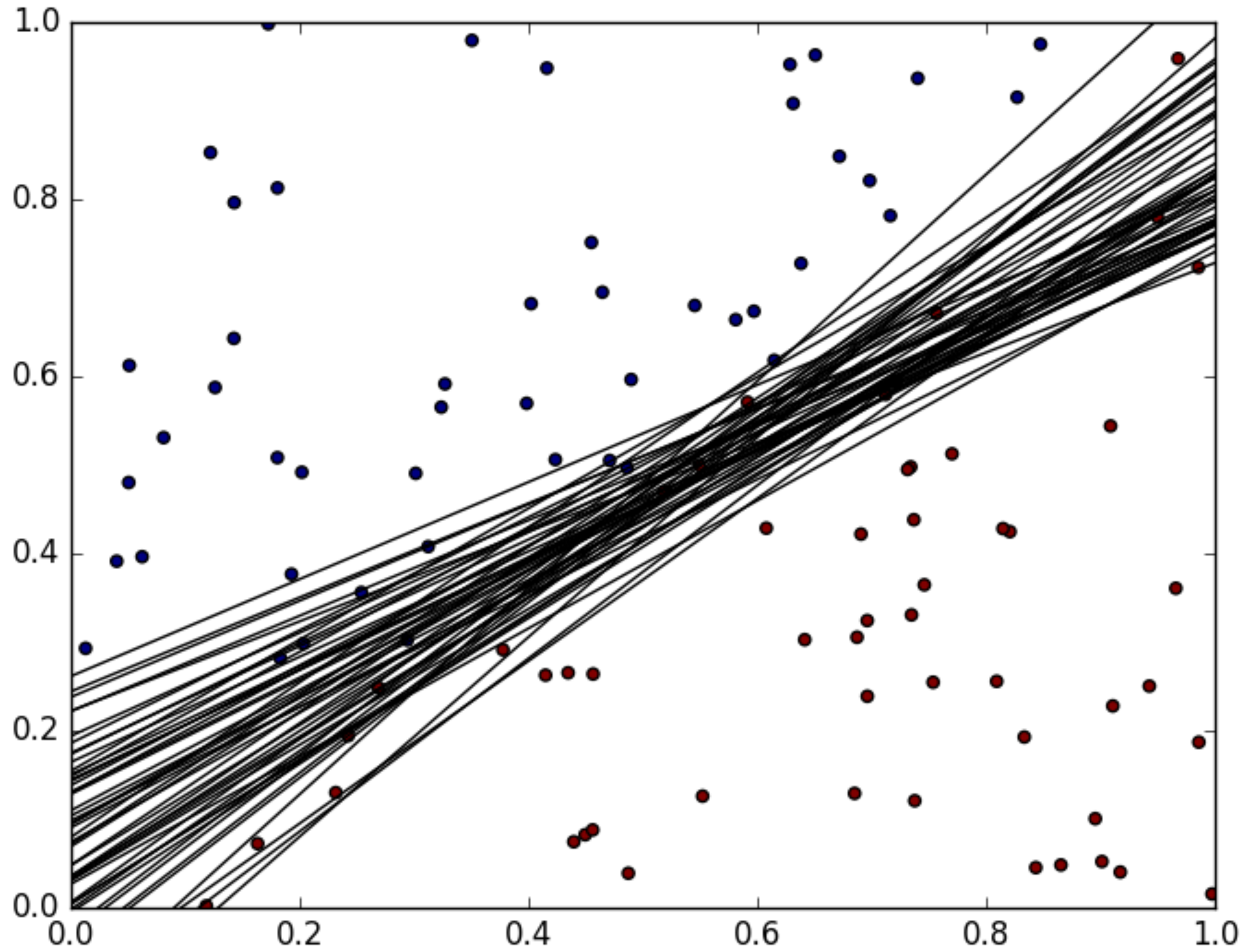
Patricia Craja, Thomas Goerttler, Christian Koopmann

Support Vector Machine

- Uses hyperplane in feature space to seperate data
- Dimension of feature space controlled by choice of kernel function
- Commen Kernels: linear, polynomial, gaussian(rbf)



Why use Bootstrapping ?



- **Goal:** Find influence of tuning parameters, data-distribution and svm-attributes on variances of predictions
- **Problem:** No way to calculate uncertaincy (variance) of predicitons
- **Solution:** Calculate sample variance based on bootstrapping

Implementation of Bootstrap

Input: Training-Data, Test-Data, SVM-Parameters, N = #Bootstrap-Replications

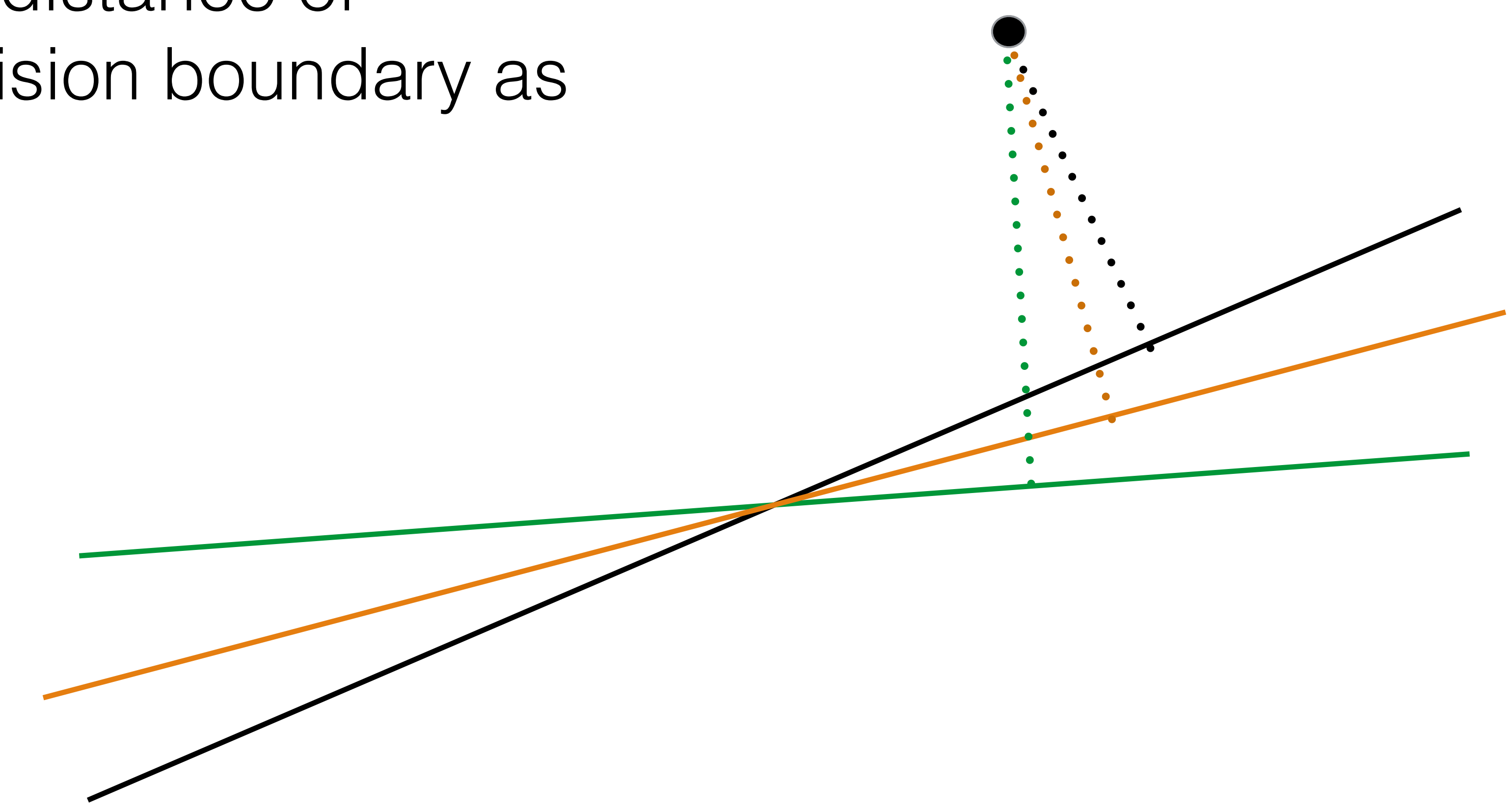
Output: Full-SVM, Variance of TestData-Distances

1. Train SVM on full Training-Data
2. Repeat N-times:
   1. Draw random sample of full size with replacement from Training-Data
   2. Train SVM on sampled Data
   3. Return distance from decision boundary for each point in Test-Data
3. Calculate Variance of distances for each Test-Point and average those Variances

Why use distance?

- **Problem**: Predictions are only binary variables and therefore might be identical across all bootstrap samples

- **Solution**: Use minimal distance of predictionpoint to decision boundary as real valued substitue



Simulation of Data

Hyperplane-Approach

Input:
- Hyperplane-Parameters w
- Constant c
- X-Distribution
- error-distribution

Algorithm:

1. Randomly draw n-observations from X- and error-distribution
2. Calculate Latent variable $y^* = c + w'x + error$
3. Calculate labels as $y = sign(y^*)$

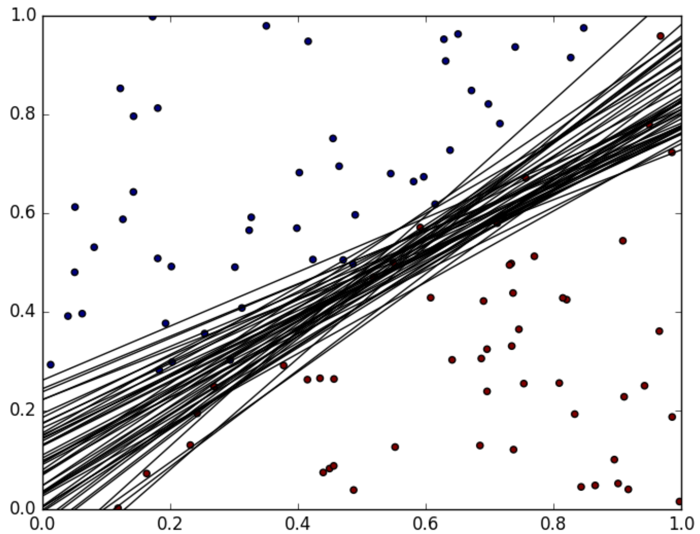
Centroid-Approach

Input:
- Centroid-Locations z_i
- Centroid-Parameters a_i
- Constant c
- X-Distribution
- error-distribution
- Distance function d(a,b)

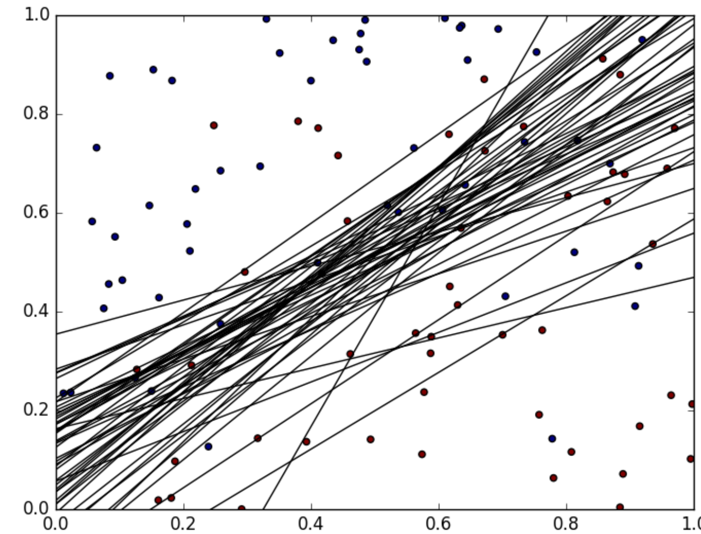
Algorithm:

1. Randomly draw n-observations from X- and error-distribution
2. Calculate Latent variable $y^* = c + a_1{}^*(d(x,z_1))^{-1} .... + error$
3. Calculate labels as $y = sign(y^*)$

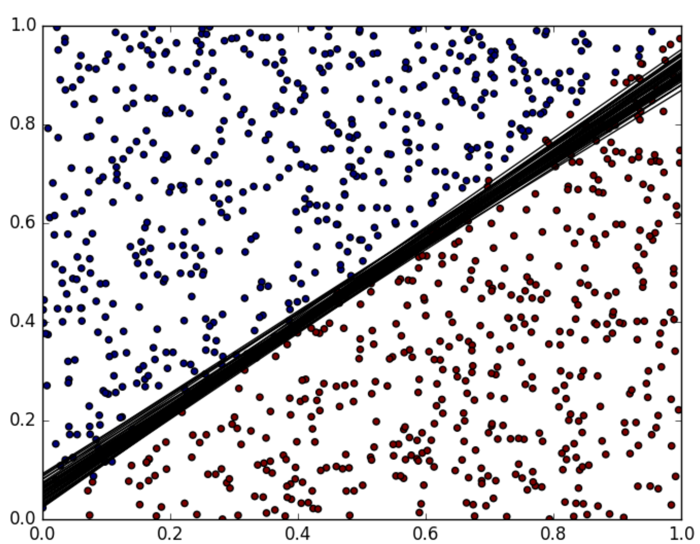
Example



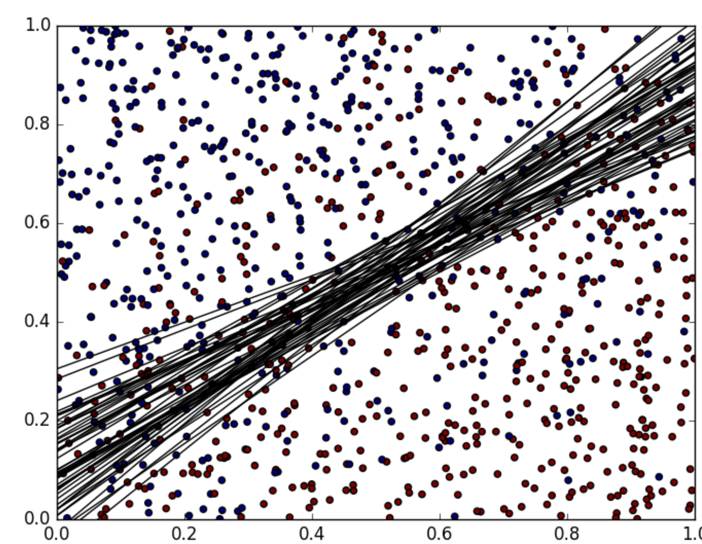
0.20950



0.26697



0.12684



0.15181

Results

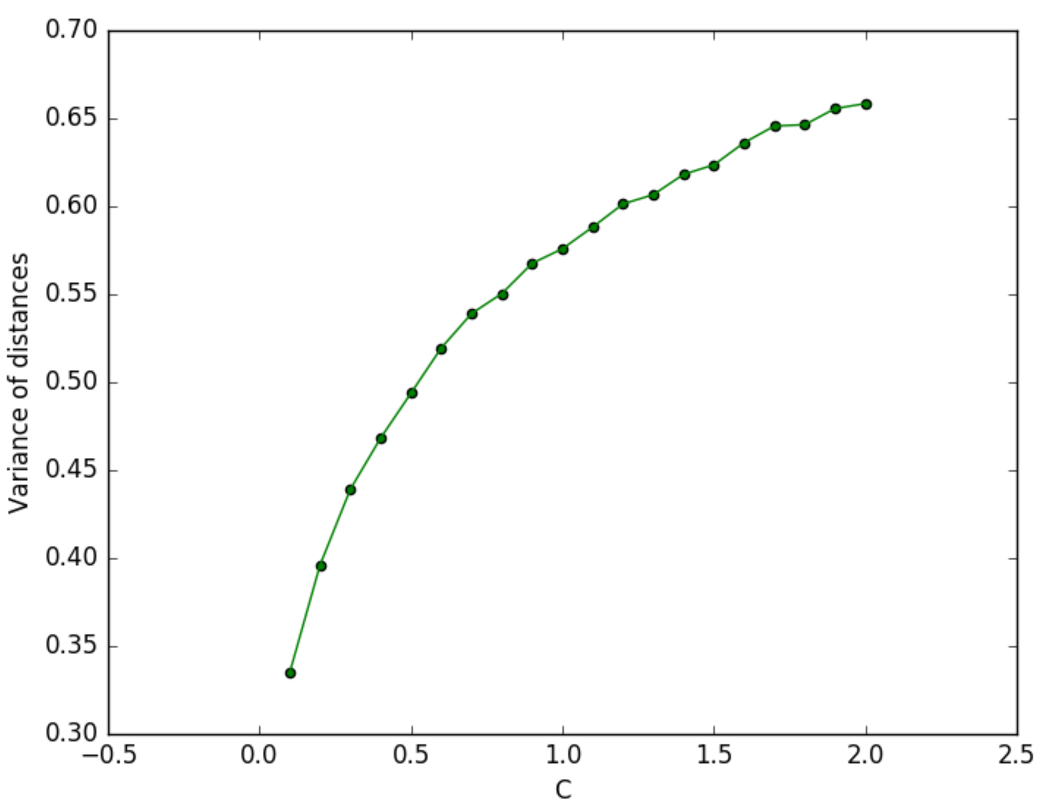
We analysed impact of following factors on prediction variance for both linear and gaussian kernel SVMs:

1. Choice of C-Parameter
2. Balance of the Training-Data (relative Proportion of 1 Label-category)
3. Number of support vectors of the original SVM

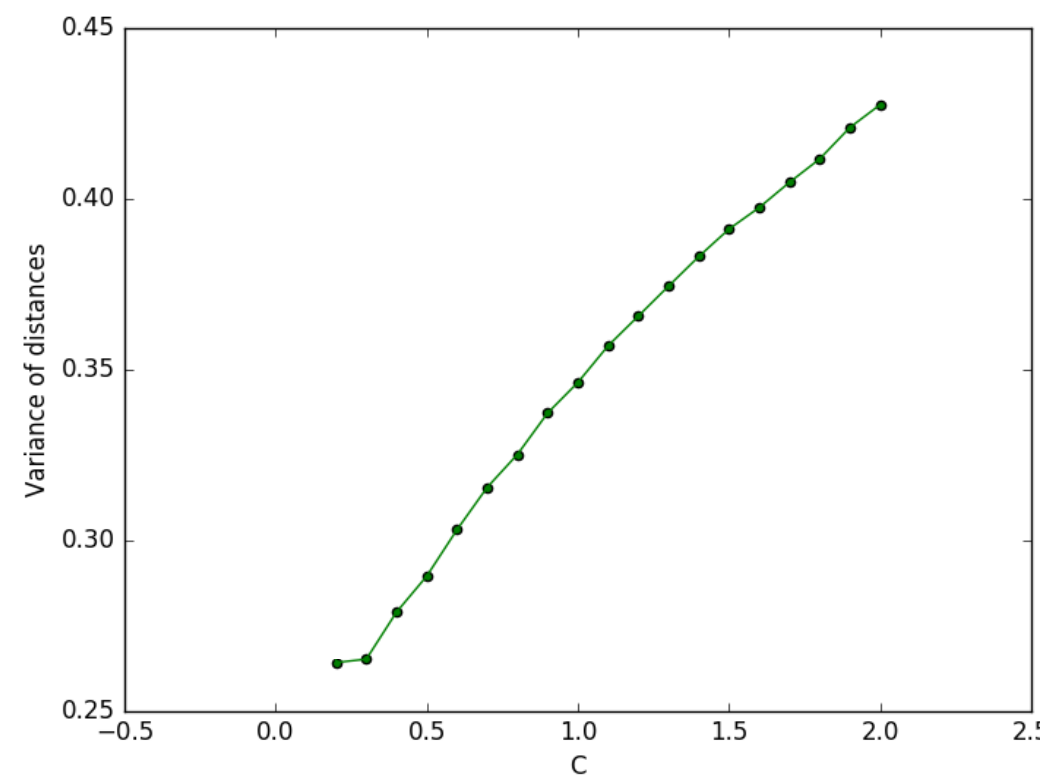
- Following slides contain results from data simulated according to the "Hyperplane" approach. With
    - Observations: 1000, Bootstrap-Replications: 100, X-Dimension: 4

C parameter

Linear

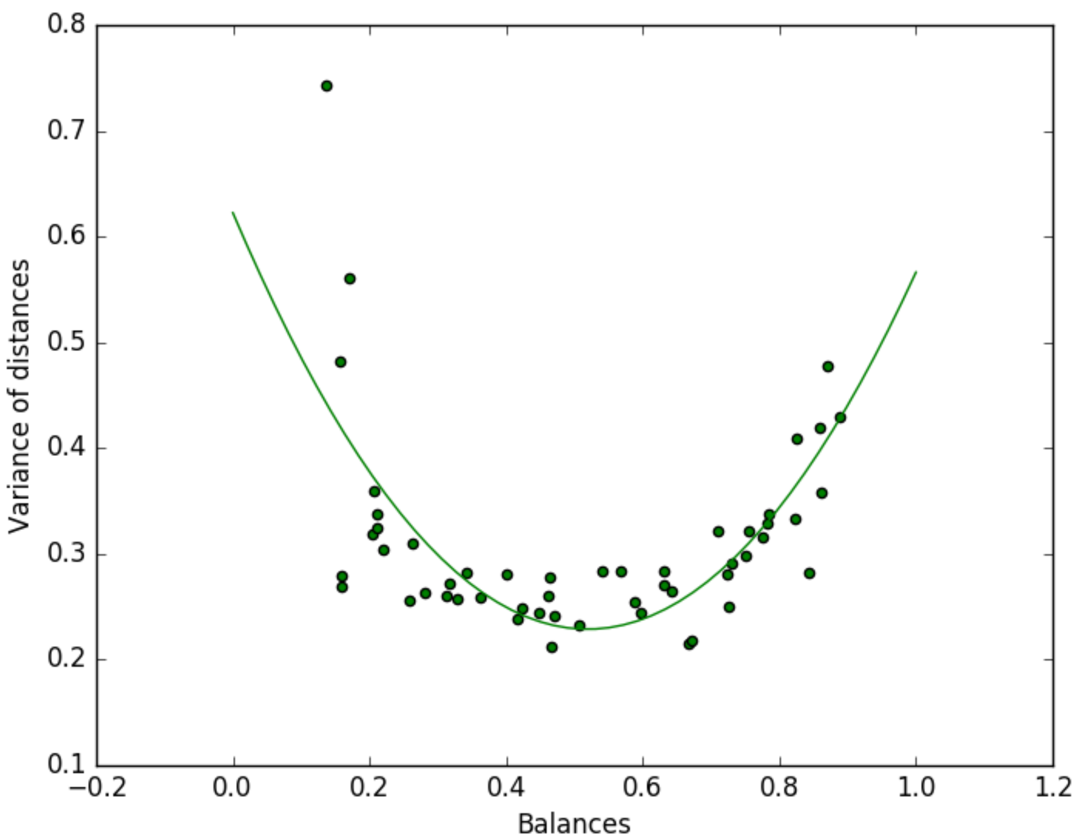


Gaussian

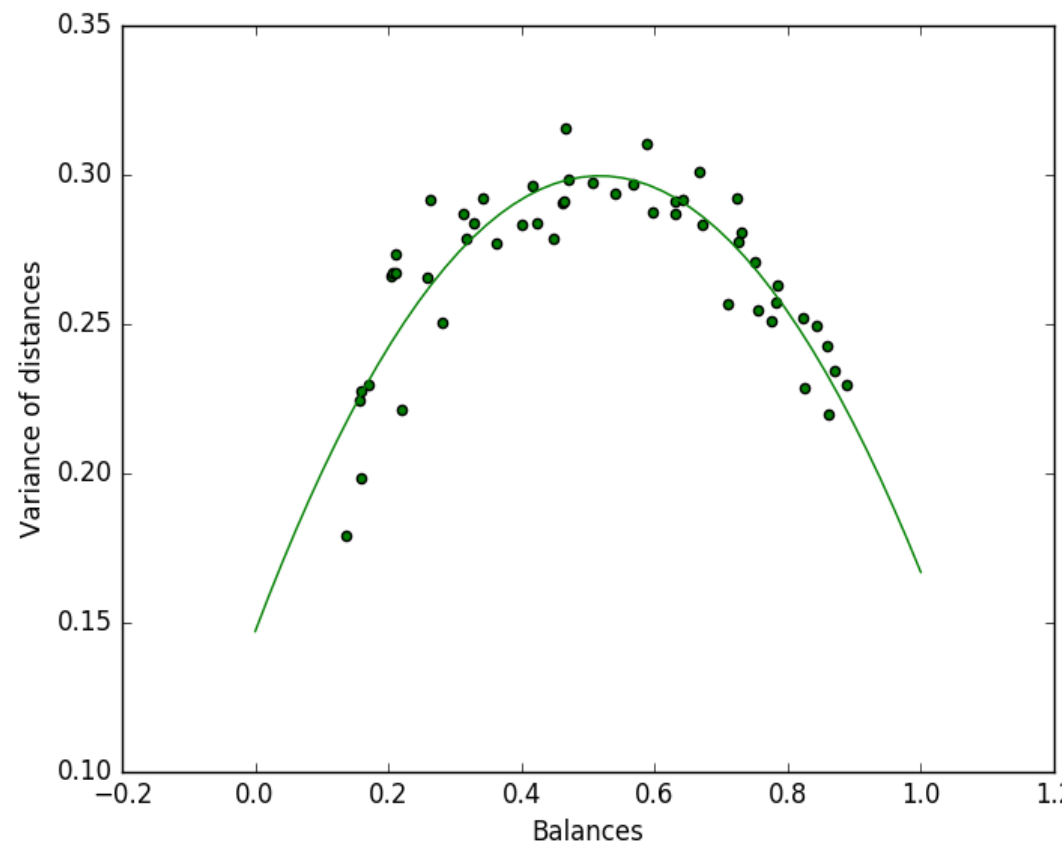


Balance of Data

Linear

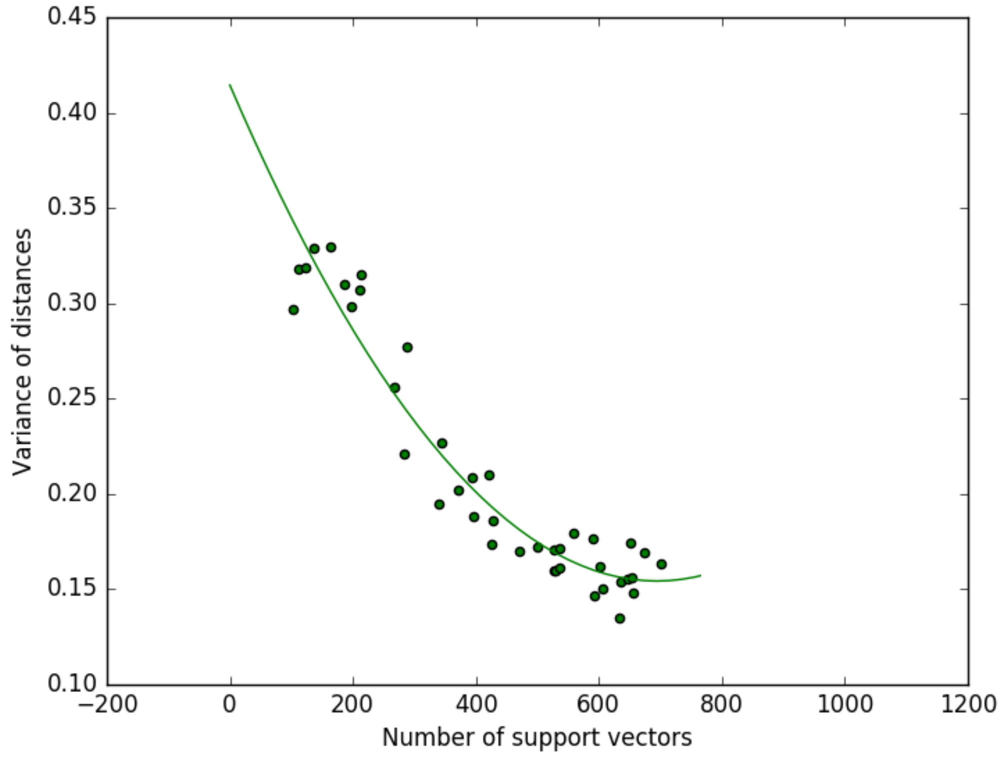


Gaussian

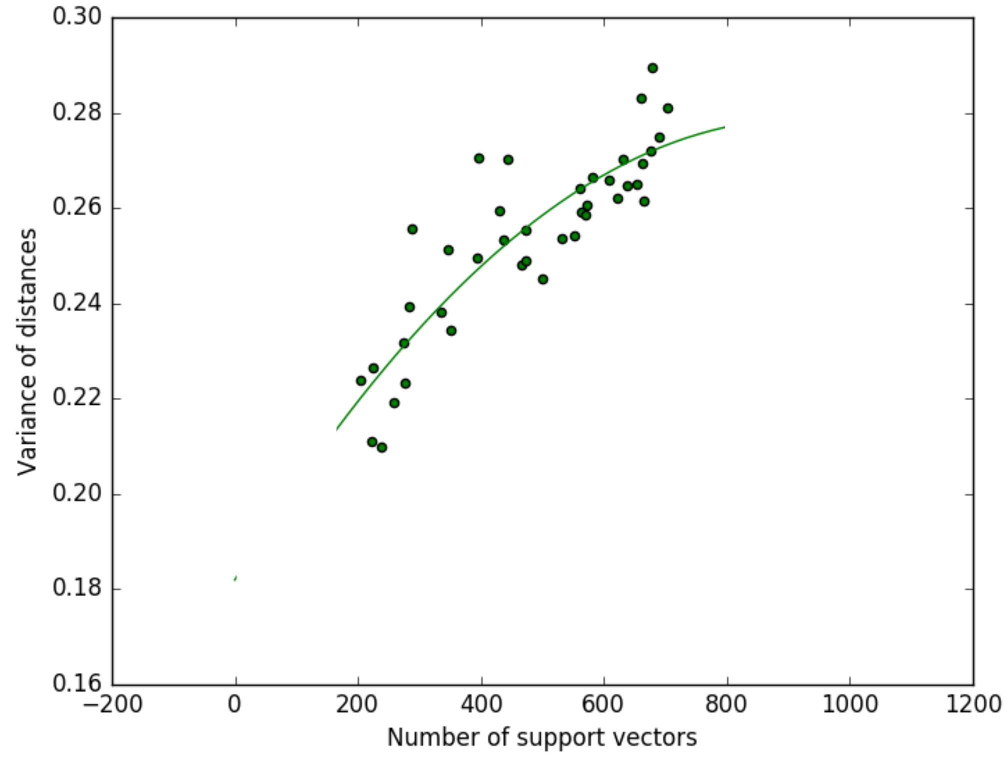


Number of support vectors

Linear



Gaussian



Conclusion

Influence of different aspects on variance

- **C-Parameter**: Positive but decreasing influence on variance
- **Balance of Data**: Positive Quadratic influence for Linear and negative quadratic influence for Gaussian SVM. Both with extremum around 0.5 (perfect balance).
- **Number of support vectors:** Positive influence for Gaussian and negative influence for linear SVMs

Outlook

Further analysis of potential interest:

- Analyse influence of dimensionality on variances
- Analyse Data simulated from more "exotic" distributions
- Analyse influence of other tuning parameters e.g. "Gamma" of rbf-kernel, degree of polynomial kernel etc.

„Kernels are powerfull!“

Herr Einstein