

# Classifying New Particle Formation

Group: John 117

Members: Elias Toukolehto and Joacim Sarén

## Methodology

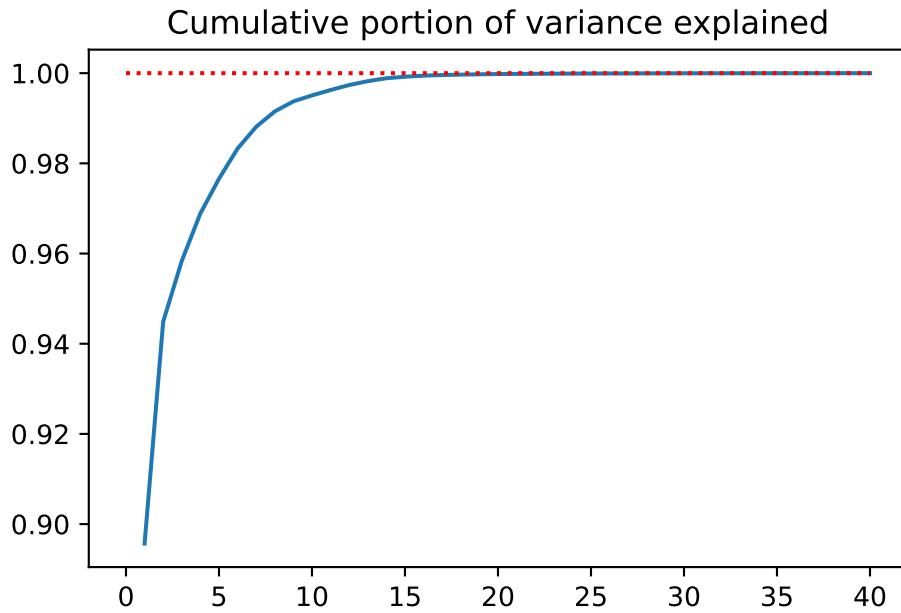
We were interested in comparing the effects principal component analysis and feature selection have on different machine learning models. We decided to rely on cross-validation for estimating accuracy. We wanted to start simple and increase the complexity of our models over time. We began with logistic regression models, and realized that we should not use L1 or L2 penalties with PCA, but decided to use them for feature selection because the performance was better. Once we had logistic regression implemented for both methodologies, we moved on to other models.

Our method for feature selection was Recursive Feature elimination, which required models to rank features in one way or another. Therefore we decided not to implement models that do not have such ranking features in feature selection. Naive Bayes classifier is an example of such model. However, since the principal components are the same for each model, we had no such issues combining naive Bayes with PCA.

At this point we had some models, but we were unhappy with their performance. We decided to split our model into two tiers. The first tier would only do binary classification, while the second tier would only classify between event types. This way if we found a model that performs better for either tier, we can combine them in our model for optimal results. Finally we tried adding more models we have encountered during the course, but their performances were quite low.

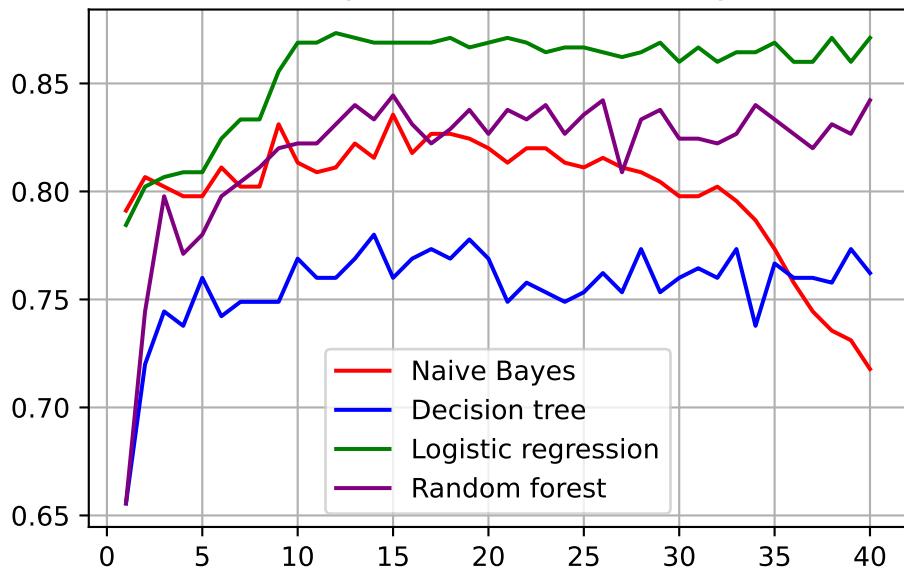
## Principal component analysis

For dimensionality reduction, We decided to use PCA for it's relative simplicity. First we looked at cumulative proportion of variance explained by the principal components. We used L2 normalization in order to keep all predictors in consideration when running PCA. Despite PVE rising very quickly, our models benefitted from more features than could be expected based on a cumulative PVE, so we did not focus on the PVE result much at all.

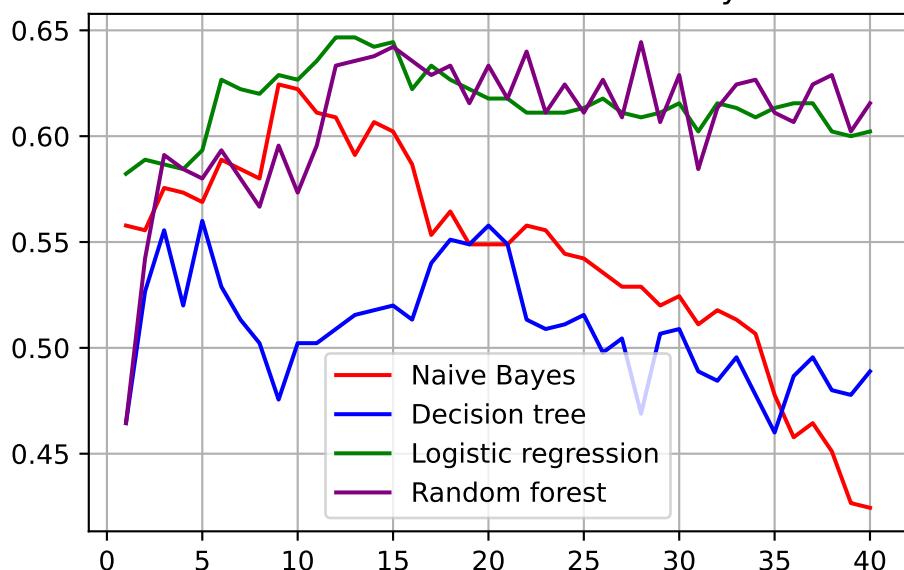


We looked at the behaviour of PCA and some models up to 100 components, but settled on a maximum of 40 for comparisons. The models we did test with more didn't show interesting behaviour beyond 40 and training random forest takes a long time with many components. Then we looked at the accuracy of logistic regression, decision tree and gaussian naive Bayes classifiers with different numbers of principal components for binary and 4-class classification using 10-fold cross-validation. The accuracies with different principal component counts are shown below.

Binary classification accuracy



4-class classification accuracy



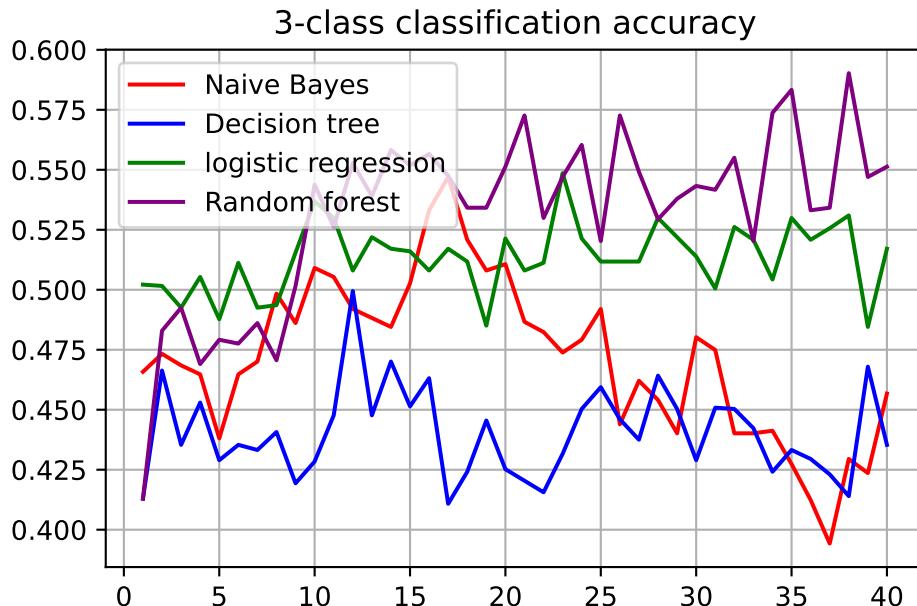
Maximum accuracies

Model	Binary	4-class
Linear regression:	0.87333	0.64667
Naive Bayes:	0.83556	0.62444
Decision tree:	0.78	0.56

Random forest: 0.84444 0.84444

Logistic regression achieves the highest accuracy in both binary and 4-class classification. Because the task focuses on binary classification, which is also easier than multiclass classification, we wanted to explore a 2-tier classifier. First tier separates events and non-events, and the 2nd classifies events to the specific event classes. To choose the model for the 2nd tier, we looked at the 3-class classification accuracy of the same classification methods trained on data only containing events.

The data with 10-fold cross-validation was inconclusive as accuracy for logistic regression jumped up and down in around the same range with 15 or less components. We suspect the main reason to be the small population of the smallest class. Out of the 225 events in the training data, only 26 are in class Ia. To smooth out the result, we decided to use 26-fold CV to match this population size. Our cross validation method maintains class sizes within folds, so the validation set always includes one Ia event.



#### Maximum accuracies

Linear regression:	0.54861
Naive Bayes:	0.54701
Decision tree:	0.49947
Random forest:	0.59028

The results still vary at higher component counts, but we can pick some well performing models from the lower component count ones. Random forest with 37 components has the

highest accuracy, but the variance in this component count range is high. Clear improvement stops at 13 components. Logistic regression reaches its peak accuracy at 22 components, but is almost as accurate with just 9 components. Naive Bayes with 16 components is between them, and clearly the best result for NB. We used 10-fold cross validation to determine the mean accuracy of these models.

## PCA-based 2-tier classifier

Based on results shown in the previous chapter, we used logistic regression with 12 principal components to separate events and non-events. For event classification we tested LR, RF and NB with 9, 13 and 16 components respectively. Combining a linear regression binary classifier with the other 3-class models got us the following 4-class classification accuracies:

Linear regression: 0.534

Naive Bayes: 0.547

Random forest: 0.542

We did not manage to improve accuracy beyond what the best individual 4-class classifiers achieved. We did not look at other performance metrics due to time constraints, but we made kaggle submissions with various models. We tested logistic regression with 12 principal components and Naive Bayes with 9 components. Logistic regression alone performed poorly, only getting a score of 0.46718. Despite having a lower accuracy, Naive Bayes got a score of 0.57045. Our 2-tier models got the following scores:

Linear regression: 0.56990

Naive Bayes: 0.57638

Random forest: 0.58070

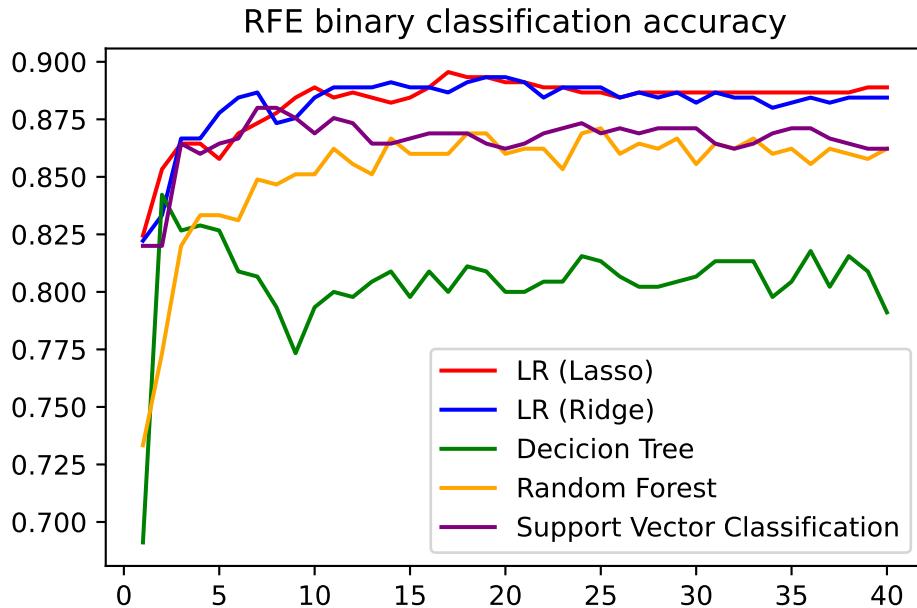
While the scores are close, there is an improvement over the single 4-class models.

## Feature selection

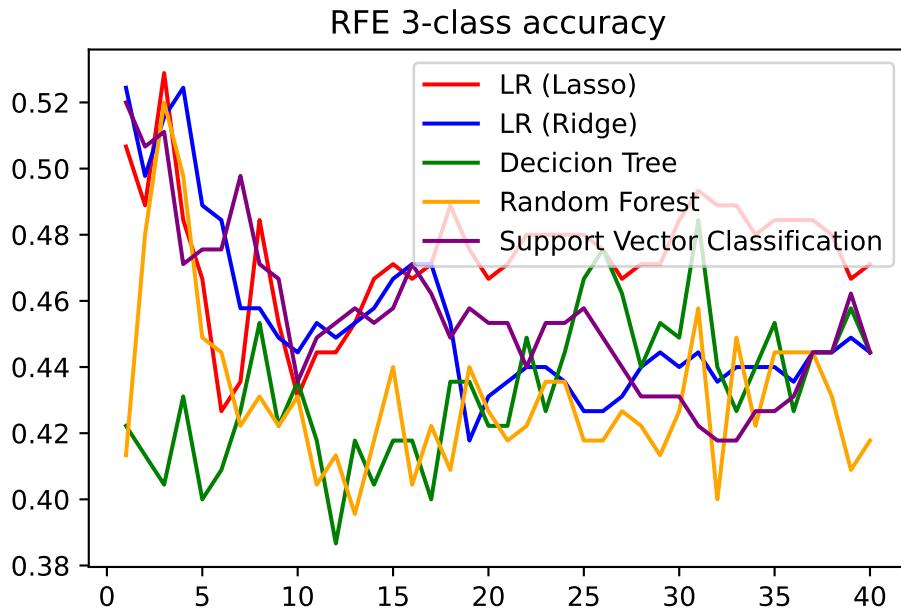
Another preprocessing method is feature selection, for which we have used Recursive Feature elimination using 5-fold cross-validation, which is a form of backward selection. With RFE we get the following results for binary, and 3-class classification.

	Model	Optimal n of features	Best accuracy	Accuracy without RFE
0	LR L1	17	0.895556	0.891111
1	LR L2	19	0.893333	0.886667
2	DT	2	0.842222	0.817778
3	RF	25	0.871111	0.866667
4	SVC	7	0.880000	0.866667
	Model	Optimal n of features	Best accuracy	Accuracy without RFE
0	LR L1	3	0.528889	0.466667
1	LR L2	1	0.524444	0.475556
2	DT	31	0.484444	0.417778
3	RF	3	0.520000	0.453333
4	SVC	1	0.520000	0.457778

Our best model for binary classification ends up being Logistic regression with L1 penalty, using 17 features reaching accuracy of 89.5%. This is a vey slight increase to the accuracy we recieve without feature selection (89.4%). The increase in model performance is more significant in the 3-class classification, being around 6% for most models. Once again linear regressiion is the best performing model, and with surprisingly few features.



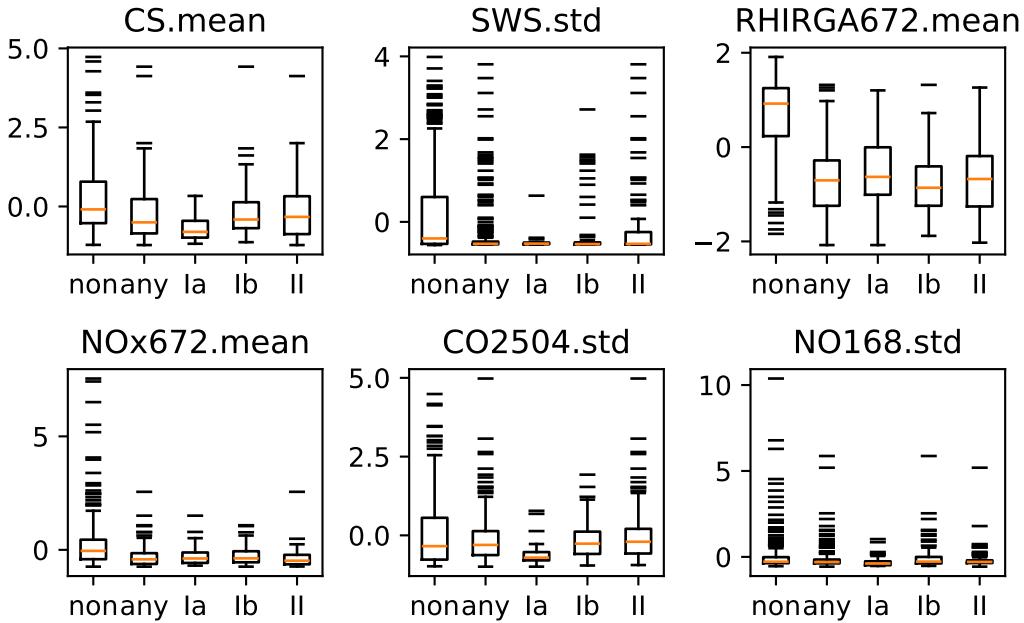
We can see that increasing features results in better accuracy, usually having the best accuracy at 10-20 features. Decision treehas it's best accuracy at only two features, while random forest has it all the way at 25. The accuracy does not increace or decrease much after 15 features for most models



Looking at the graph for accuracy per number of features for 3-class RFE, we can see that the differences each feature makes are lot more significant, even with 20+ features. The accuracy is quite poor, and for most models it's best with only very few features.

## Feature analysis

Analyzing coefficients allowed us to rank some features by usefulness. The ones for the plot below were defined by performance with logistic regression using 5-fold cross-validation.



Here are boxplots of some of the most useful (top row), and least useful (bottom row) features for binary classification in the dataset. For the reliable features we can see that the quartile ranges don't have much overlap, as for the unreliable features there is a lot of overlap between classes. CS.mean is an outlier, where models find it very useful even though it has surprisingly similar boxplots between classes. Some features that are good with binary classification, are less useful in 3-class classification. RHIRGA672.mean is an example of such feature. This is one of the strengths of using two tiers with our model for feature selection.

## Conclusions

In the end our best performing model was the 2-tier linear regression + random forest model with PCA. Improving over simple models turned out to be surprisingly difficult.

We had many improvement ideas in mind, many of which were implemented in the models seen in the presentations. One obvious idea was to increase our model from 2-tier model into a 3-tier model. This would let us improve between stages once more. The second idea was to add engineered features to the data. The data is the means and variances of different units, and by fitting them into some equation, we could have features that make classifying the data easier. We also could have used other performance metrics besides accuracy to improve our submission. Some domain knowledge or further data analysis would probably have been beneficial too. In addition, further collaboration would allow for more model combinations. It's possible that using feature selection for one tier and PCA for another would've produced a better classifier.

We were still able to test plenty of different models, and compare their accuracy scores by adjusting our features. Having two tiers in our models, allows us to pick the best amount of features for both stages, slightly improving our accuracy

## **Grading suggestion**

We give ourselves a grade of 3. We explain our approach well and our goals were clear from the start. The report stays on topic and our performance measures are consistent. There are obvious shortcomings which is why we would not grade this higher. In short, our project lacks the depth to be considered very good. We did not use any advanced techniques and we only looked at model accuracy as a metric because it was convenient.