Mini-project

Britta Wilde (briwil-3), Emil Lundin (lunemi-9)

Agenda

1 Process of choosing datasets

2 Selected datasets

3 Preprocessing of datasets

4 Models

5 Training

6 Results

7 Comparison/Discussion

8 Conclusion

Datasets and preprocessing

Pipeline for selecting and processing the data

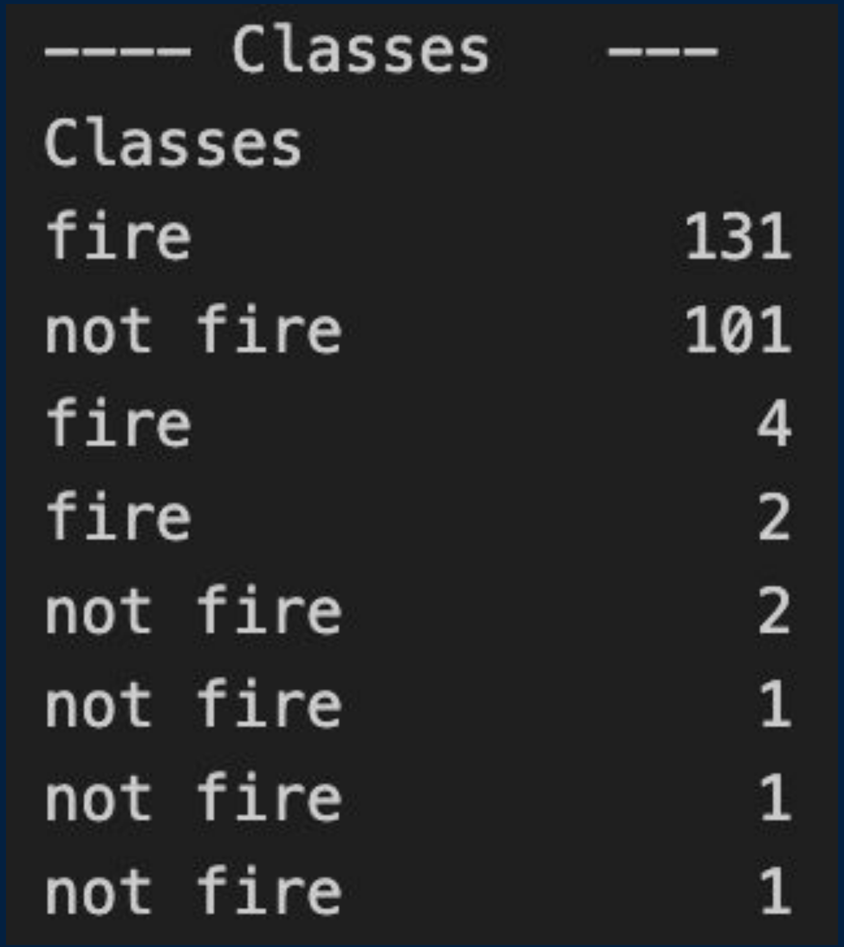
Dataset selection process

Criteria - easy to deal with:

- Based on article
- No special preprocessing
- Limited size

Result: 22 selected datasets

Algerian forest fire:



```
---- Classes   ---
Classes
fire            131
not fire        101
fire              4
fire              2
not fire          2
not fire          1
not fire          1
not fire          1
```

Example of not included dataset.

Selected datasets

Dataset	No. samples	No. features	Missing values	Majority class %	No. classes	UCI id
acute_inflamations	120	7	No	58.3	2	184
balance_scale	625	4	No	46.1	3	12
balloons	**16**	4	No	56.3	2	13
breast_cancer_wisconsin_diagnostic	569	30	No	62.7	2	17
car_evaluation	1728	6	No	70.0	4	19
congress_voting_records	232	16	**Yes**	61.4	2	105
credit_approval	653	15	**Yes**	55.5	2	27
ecoli	336	7	No	42.6	**8**	39

Selected datasets

Dataset	No. samples	No. features	Missing values	Majority class %	No. classes	UCI id
fertility	100	9	No	**88**	2	244
habermans_survival	306	**3**	No	73.5	2	43
hayes_roth	132	4	**Yes**	31.9	4	44
heart_disease	297	13	**Yes**	54.1	5	45
ilpd	579	10	**Yes**	71.4	2	225
iris	150	4	No	33.3	3	53
lenses	24	**3**	No	62.5	3	58
mammographic_mass	830	5	**Yes**	53.7	2	161

Selected datasets

Dataset	No. samples	No. features	Missing values	Majority class %	No. classes	UCI id
mushroom	5644	22	**Yes**	51.8	2	73
spect_heart	267	22	No	79.4	2	95
spectf_heart	267	**44**	No	79.4	2	96
statlog	1000	20	No	70	2	144
wine_quality	**6497**	11	No	43.7	7	186
zoo	101	16	No	40.6	7	111

Selected datasets

Dataset	No. samples	No. features	Missing values	UCI id	Our id
acute_inflamations	120	7	No	184	1
balance_scale	625	4	No	12	2
balloons	**16**	4	No	13	3
breast_cancer_wisconsin_diagnostic	569	30	No	17	4
car_evaluation	1728	6	No	19	5
congress_voting_records	232	16	**Yes**	105	6
credit_approval	653	15	**Yes**	27	7
ecoli	336	7	No	39	8

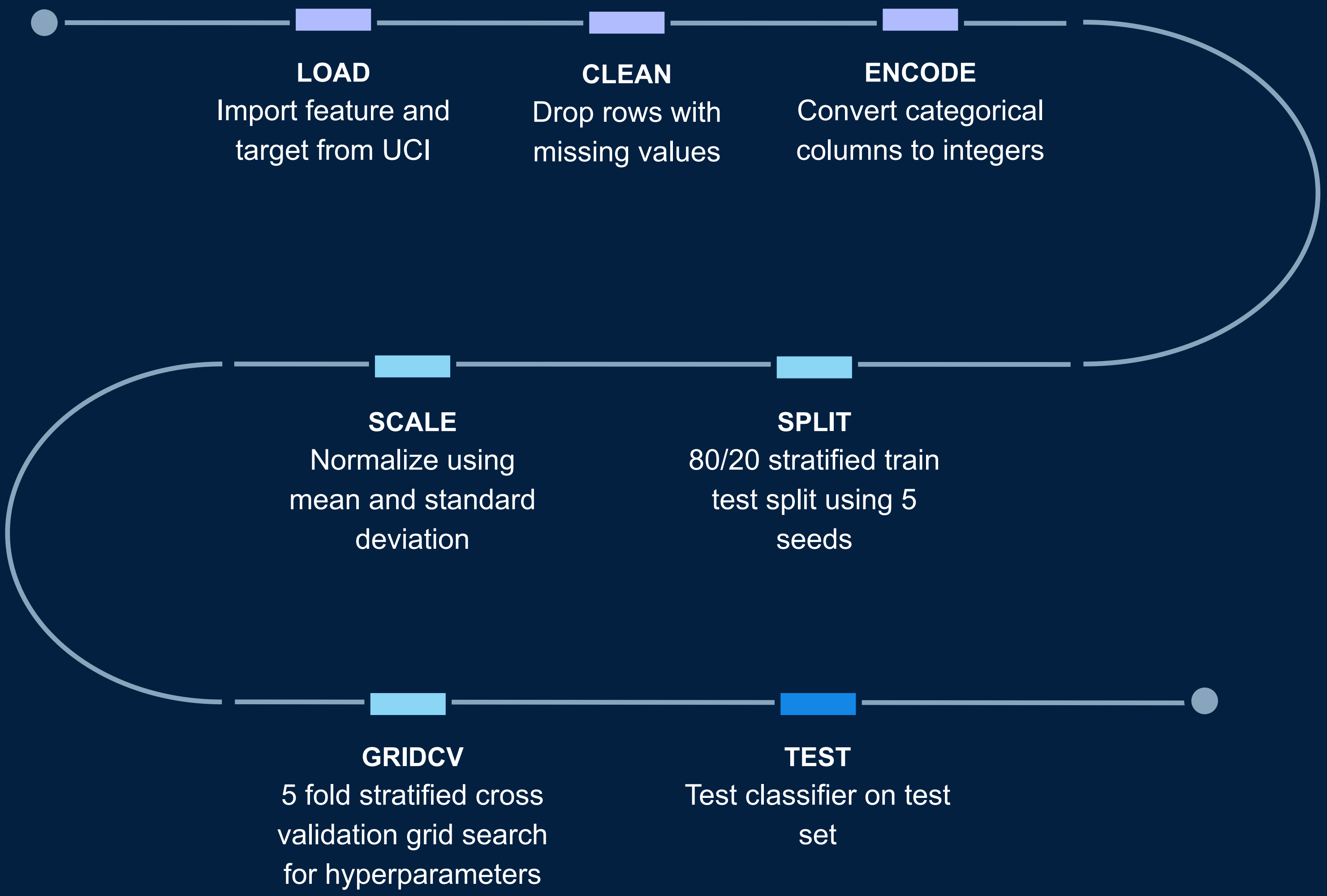
Selected datasets

Dataset	No. samples	No. features	Missing values	UCI id	Our id
fertility	100	9	No	244	9
habermans_survival	306	**3**	No	43	10
hayes_roth	132	4	**Yes**	44	11
heart_disease	297	13	**Yes**	45	12
ilpd	579	10	**Yes**	225	13
iris	150	4	No	53	14
lenses	24	**3**	No	58	15
mammographic_mass	830	5	**Yes**	161	16

Selected datasets

Dataset	No. samples	No. features	Missing values	UCI id	Our id
mushroom	5644	22	**Yes**	73	17
spect_heart	267	22	No	95	18
spectf_heart	267	**44**	No	96	19
statlog	1000	20	No	144	20
wine_quality	**6497**	11	No	186	21
zoo	101	16	No	111	22

Overview: pipeline



LOAD
Import feature and target from UCI

CLEAN
Drop rows with missing values

ENCODE
Convert categorical columns to integers

SCALE
Normalize using mean and standard deviation

SPLIT
80/20 stratified train test split using 5 seeds

GRIDCV
5 fold stratified cross validation grid search for hyperparameters

TEST
Test classifier on test set

Preprocessing

Cleaning

Remove rows with missing values.

Encoding

Convert categorical features into numerical.

Splitting

Stratified 80-20 train-test split.

Scaling

Normalize using mean and standard deviation.

Based on article

Models and Training

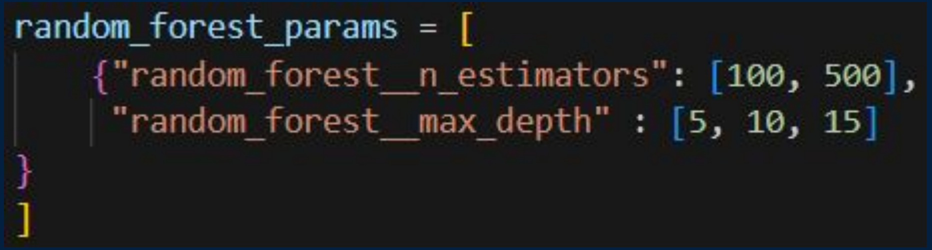
For supervised and unsupervised classification

Models: supervised

1. Random forest
2. Support vector machine
3. Logistic regression
4. K-nearest neighbors
5. Gaussian naive bayes
6.

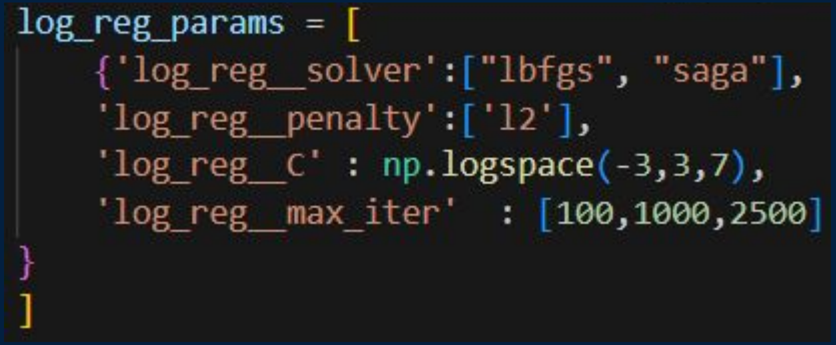
Hyperparameter optimization

Random Forest



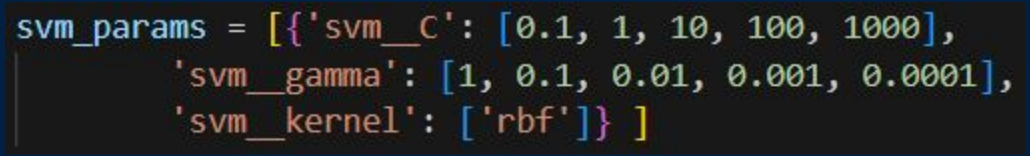
```
random_forest_params = [
    {"random_forest__n_estimators": [100, 500],
     "random_forest__max_depth" : [5, 10, 15]
}
]
```

Logistic Regression



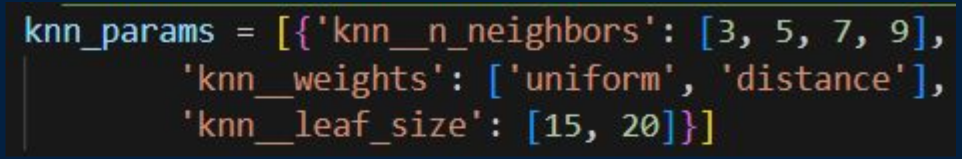
```
log_reg_params = [
    {'log_reg__solver':["lbfgs", "saga"],
    'log_reg__penalty':['l2'],
    'log_reg__C' : np.logspace(-3,3,7),
    'log_reg__max_iter'  : [100,1000,2500]
}
]
```

SVM



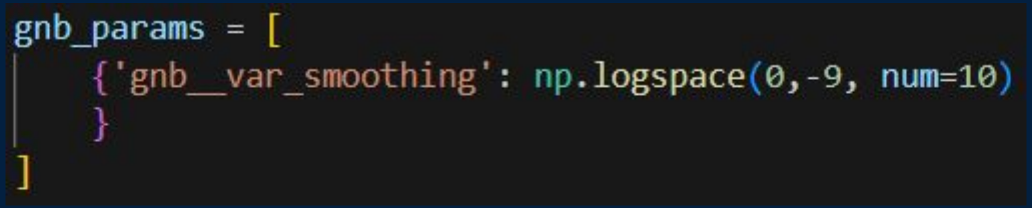
```
svm_params = [{'svm__C': [0.1, 1, 10, 100, 1000],
        'svm__gamma': [1, 0.1, 0.01, 0.001, 0.0001],
        'svm__kernel': ['rbf']} ]
```

KNN



```
knn_params = [{'knn__n_neighbors': [3, 5, 7, 9],
        'knn__weights': ['uniform', 'distance'],
        'knn__leaf_size': [15, 20]}]
```

Gaussian naive bayes



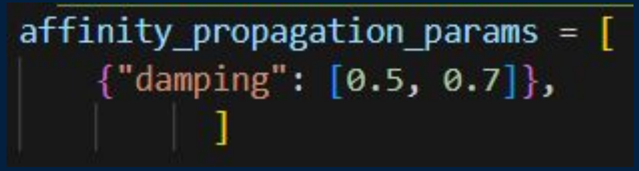
```
gnb_params = [
    {'gnb__var_smoothing': np.logspace(0,-9, num=10)
    }
]
```

Models: unsupervised

1. Agglomerative clustering
2. Affinity propagation
3. K-means
4.

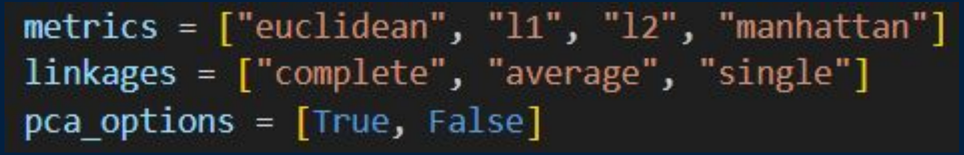
Hyperparameter optimization

Affinity Propagation

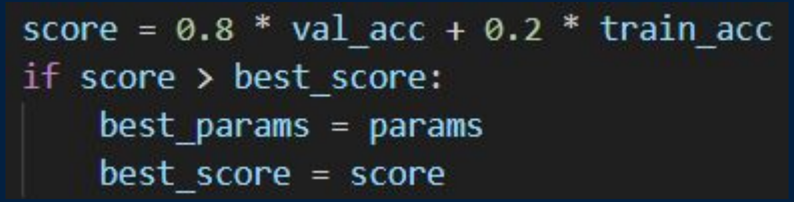


```
affinity_propagation_params = [
    {"damping": [0.5, 0.7]},
          ]
```

Agglomerative Clustering

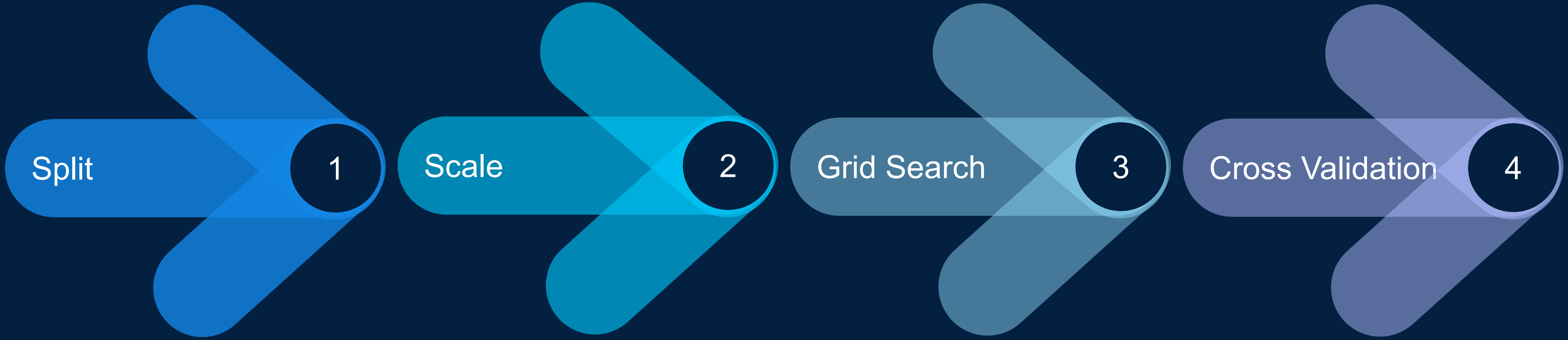


```
metrics = ["euclidean", "l1", "l2", "manhattan"]
linkages = ["complete", "average", "single"]
pca_options = [True, False]
```



```
score = 0.8 * val_acc + 0.2 * train_acc
if score > best_score:
    best_params = params
    best_score = score
```

Training



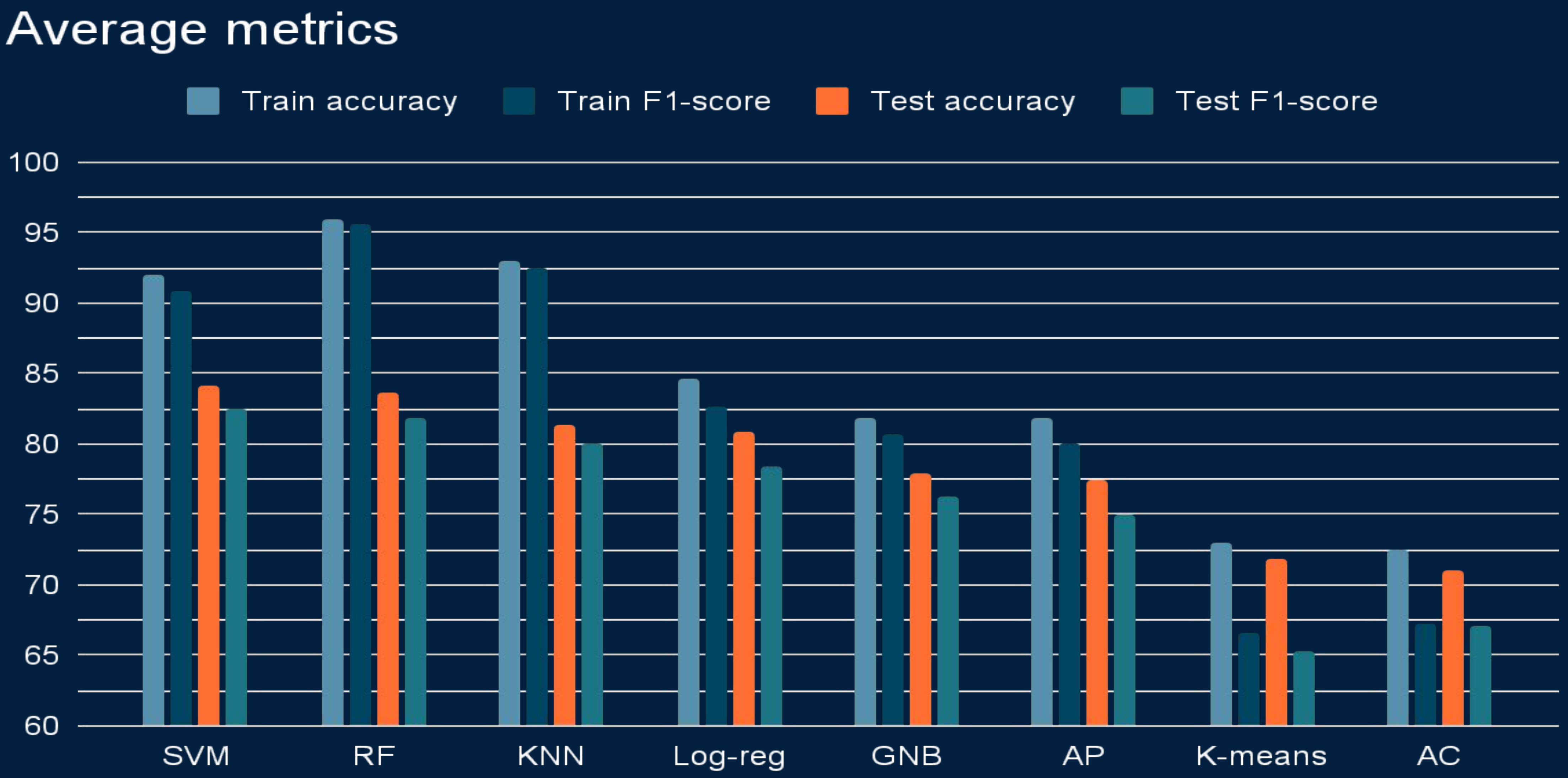
For K-means and affinity propagation: mode mapping

For agglomerative clustering: Hungarian algorithm mapping, no single prediction, no cross validation (80/20 train/validation split)

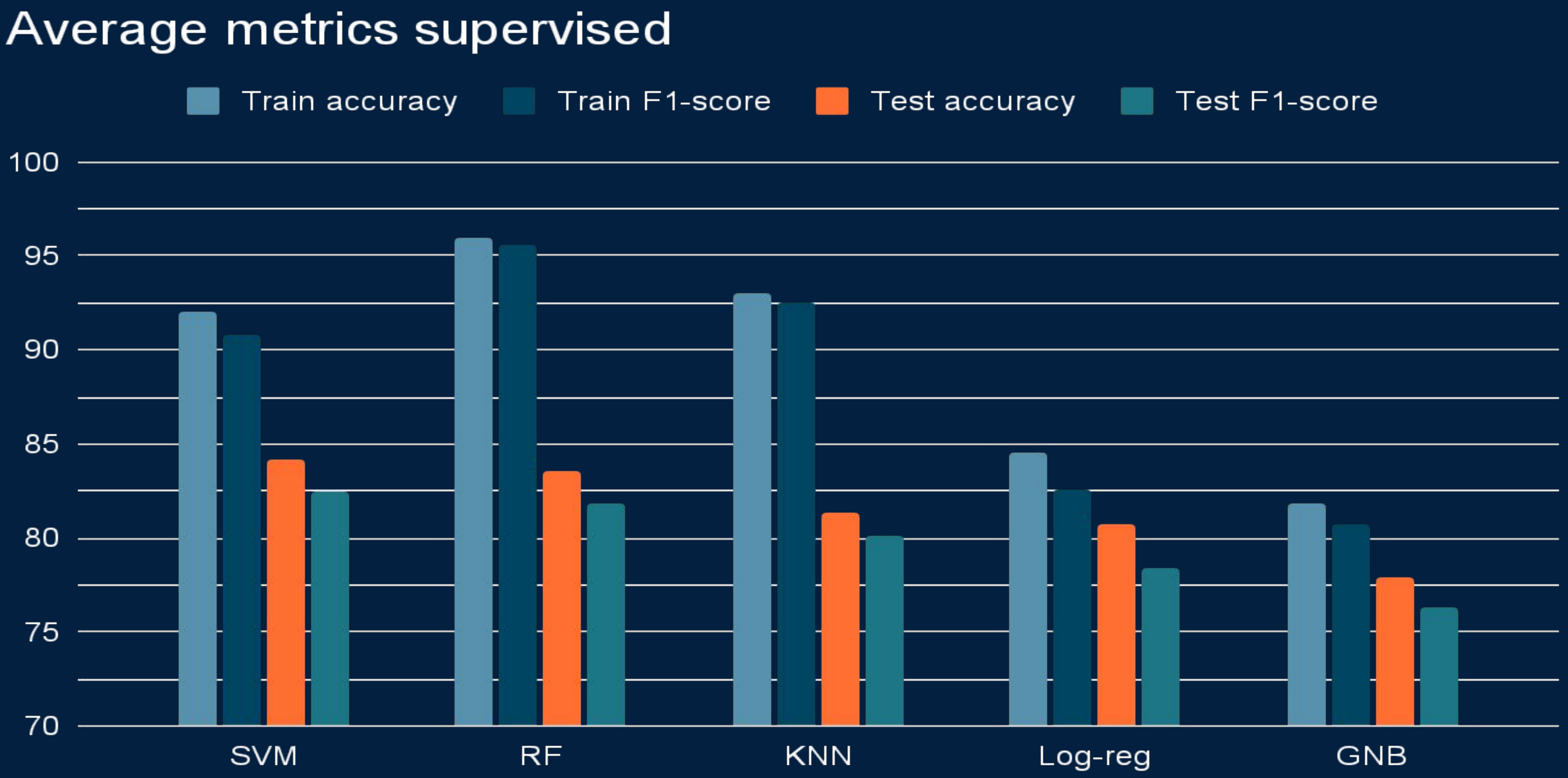
Results and Discussion

Our results in comparison to the article

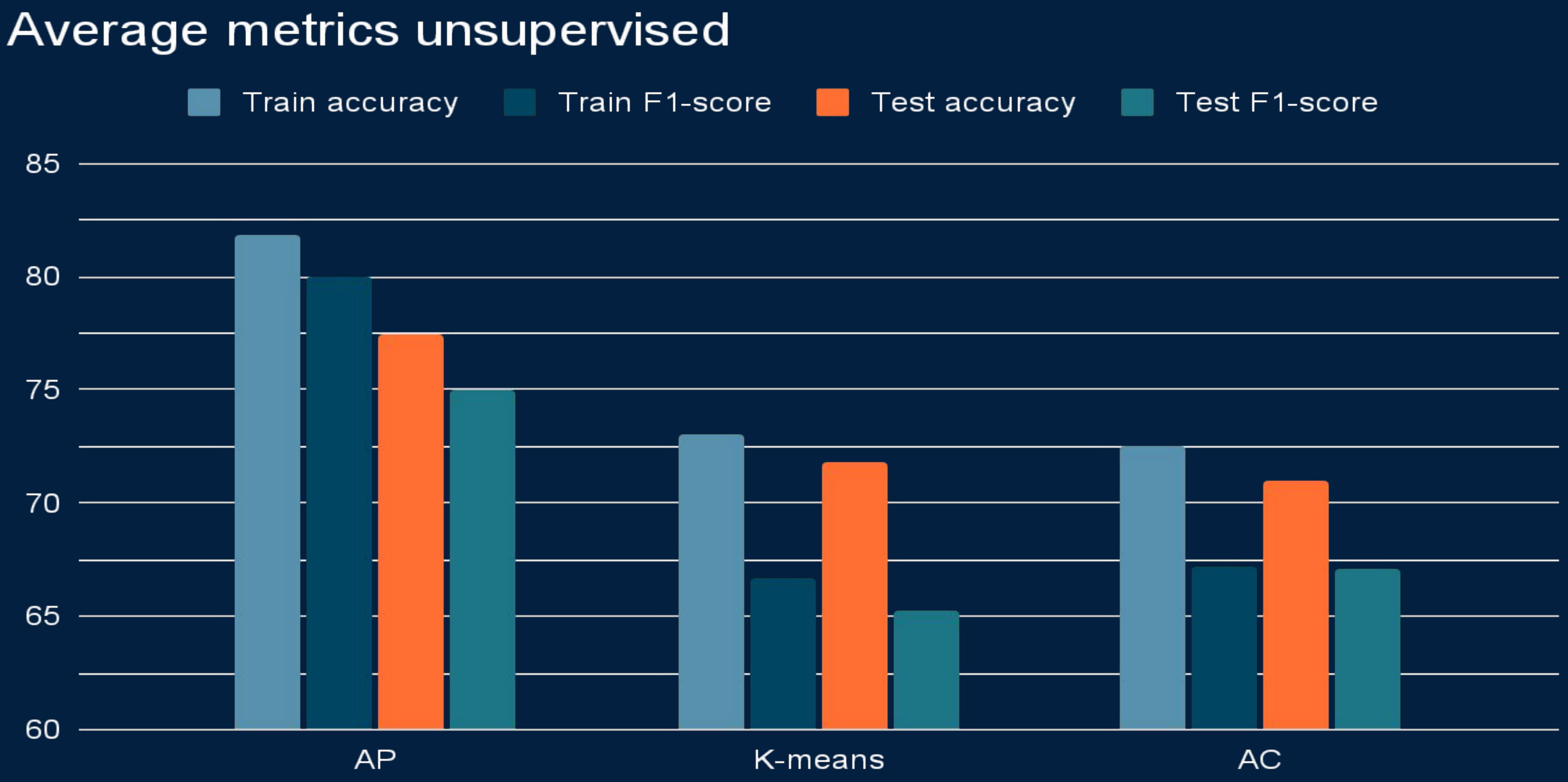
Results


Average metrics
Train accuracy
Train F1-score
Test accuracy
Test F1-score
100
95
90
85
80
75
70
65
60
SVM
RF
KNN
Log-reg
GNB
AP
K-means
AC

Results: supervised



Results: unsupervised



Results

To see all results:
We saved all results in a tensorboard logger

Our runs are saved in our GitHub

Command to start (given tensorboard installed, replace runs/ with path to folder, might have to try different port number)

```
tensorboard --logdir runs/ --port 6006 --samples_per_plugin images=22
```

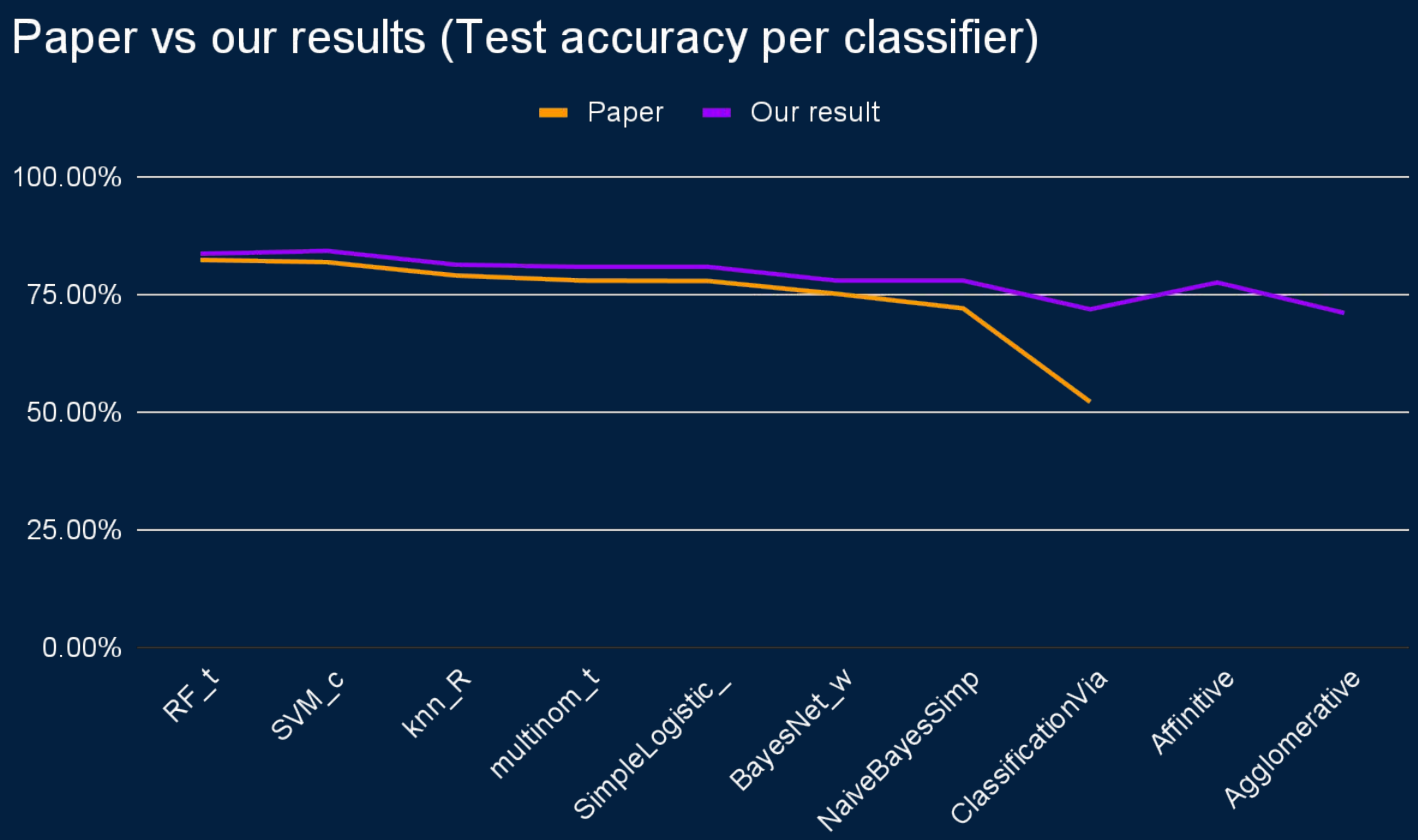
Regex for colours:

```
(knn|kmeans|svm|random|gnb|log_reg|agglo|affini)
```

Comparison/discussion

	Classifier name	Paper (average accuracy)	Our result
Random Forest	RF_t	**82.3 %**	83.6 %
Support Vector Machine	SVM_c	81.8 %	**84.2 %**
K-Nearest Neighbor	knn_R	79.0 %	81.3 %
Logistic Regression	multinom_t	77.9 %	80.8 %
	SimpleLogistic_w	77.8 %	
Naive Bayes	BayesNet_w	75.1 %	77.9 %
	NaiveBayesSimple_w	72.0 %	
Clustering (K-Means)	ClassificationViaClustering_w	**52.1%**	71.8 %
Affinity Propagation	/	/	**77.5 %**
Agglomerative Clustering	/	/	71

Comparison/discussion



Conclusion

- implemented 8 classifiers for 21 datasets, common preprocessing and CV
- reach similar but slightly better results than the paper
- Difficulties with imbalanced datasets => stratified sampling, F1-score
- Supervised classifiers were easier to implement with the pipeline that is provided by sklearn
- learned about different techniques how to remap labels for unsupervised classifiers
- Future additions:
  - removing highly correlated columns
  - adding more classifiers
  - adding more possible combinations of hyperparameters
  - adding more complex datasets

Thank you for your attention!

Kontakt

Britta Wilde - briwil-3
Emil Lundin - lunemi-9
Github: https://github.com/Dropptimus/D7041E_Mini_project
=> full_pipeline_fixed_metrics_multiple_seed.ipynb

LULEÅ
TEKNISKA
UNIVERSITET