Voter Targeting For Low-Funded Campaigns

Charles Kramer/ISYE 7406/April 2024

Overview

- Campaign Analytics: Great but Expensive
- Answer: Target Voters Using Free Public Data
- Data Overview
- Methods
- Results
- Conclusions

Campaign Analytics:

Great but Expensive

- Campaign analytics help target campaign resources (ads, events, canvassing)
- But they aren’t cheap, and down-ballot campaigns are not always well-funded
- These campaigns, i.e. state legislature, county board matter—education, health, taxes
- Can we devise useful campaign targeting analytics using free public data?

Voter Targeting with Free Public Data

- Need two components:
  - Voting data by precinct: Virginia Department of Elections
  - Demographic data by Census tract: American Community Survey
  - Merge voting to demographics using geospatial join (tract nearest precinct)
- Research question: which precincts are apt to flip parties, based on demographics?
  - -> focus efforts on voters in these precincts
  - Cost: $0

Data

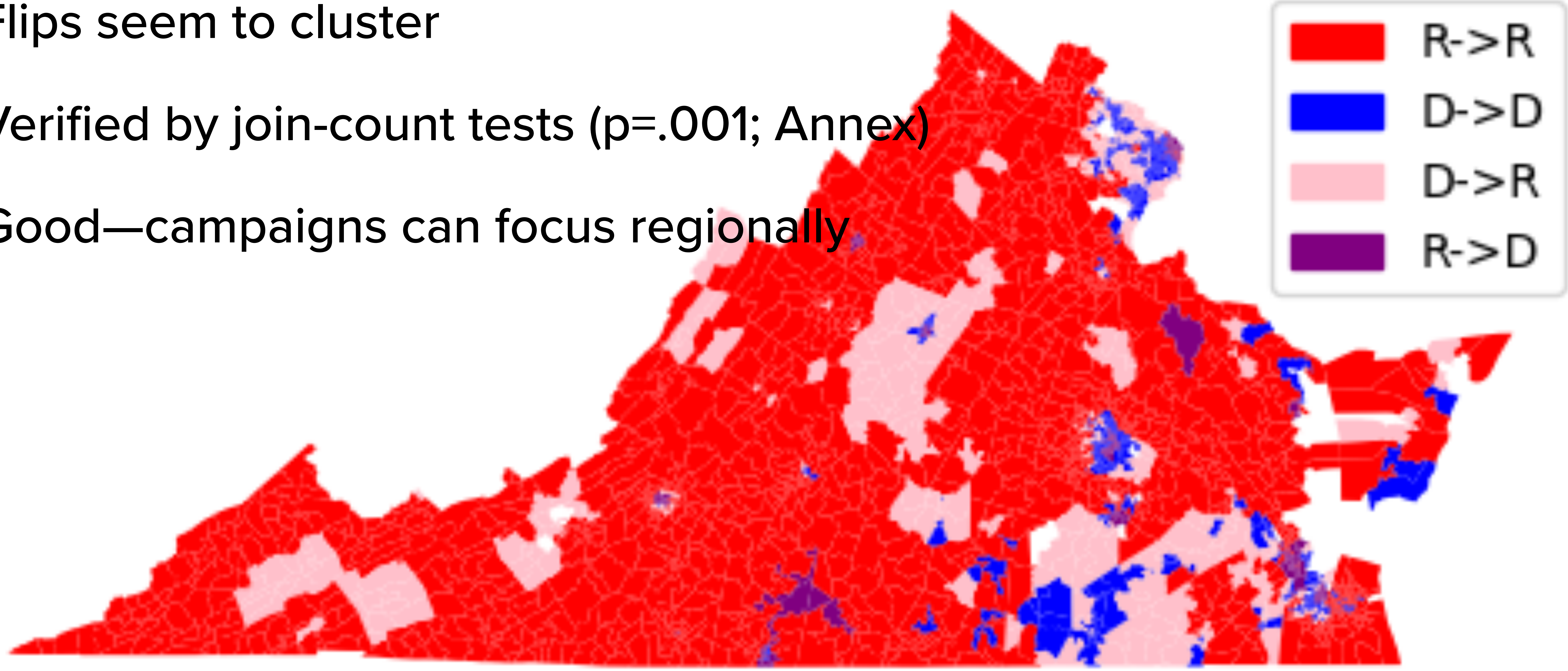
Virginia 2020 General Election

- Virginia: “Purple” state, not dominated by Republican or Democratic voters
- 2020 General Election: high turnout, high-interest election
- Precincts: 2424 precincts; measure party affiliation by total vote majority (R or D)
  - 4 categories: 2 flips: D->R, R->D; 2 not-flips: D->D, R->R
  - This is really 2 problems: Republican campaign wants to ID flippable Democratic precincts, vice versa
- Demographics: concepts shown to correlate with ‘on the fence’ voters (Pew)
  - Median age, % white, % male, % under poverty line, % foreign born, % eligible for Medicaid, % with broadband
- Cleaning: Keep precincts that exist both years, replace missing age with median, standardize demographics

EDA: The Target

Flips from 2019 to 2020, All Precincts

- Flips seem to cluster
- Verified by join-count tests (p=.001; Annex)
- Good—campaigns can focus regionally



Flips are Uncommon

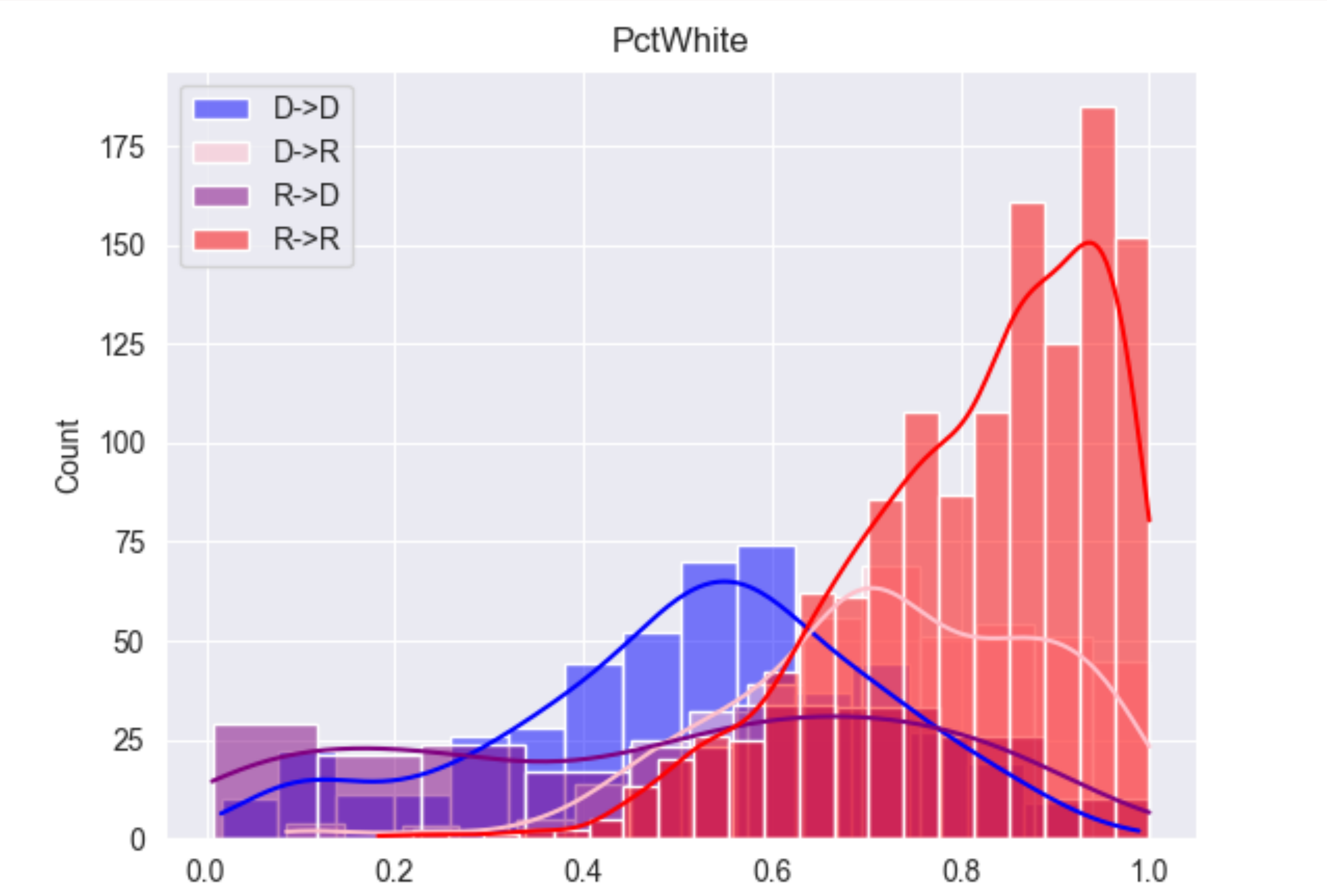
- First panel: P(2020=a & 2019=b)
- Second panel P(2020=a | 2019=b)
- This is the campaign's focus: flippables
- E.g. only 14.5 percent of R districts flipped D
- Unbalanced data; challenging for ML
- => rebalance using SMOTE

		Unconditional P(2020, 2019)	
		2020	
		D	**R**
2019	**D**	20.0	18.5
	R	8.9	52.6
		Conditional P(2020 | 2019)	
		2020	
		D	**R**
2019	**D**	52.0	48.0
	R	14.5	85.5

Demographic Data

See Annex for other variables

- Distributions differ across flip category -> variables can identify category
- Consistent with studies that correlate race, etc with party lean
- Confirmed by K-sample Anderson-Darling tests; rejects equality null (Annex)
- True for all variables



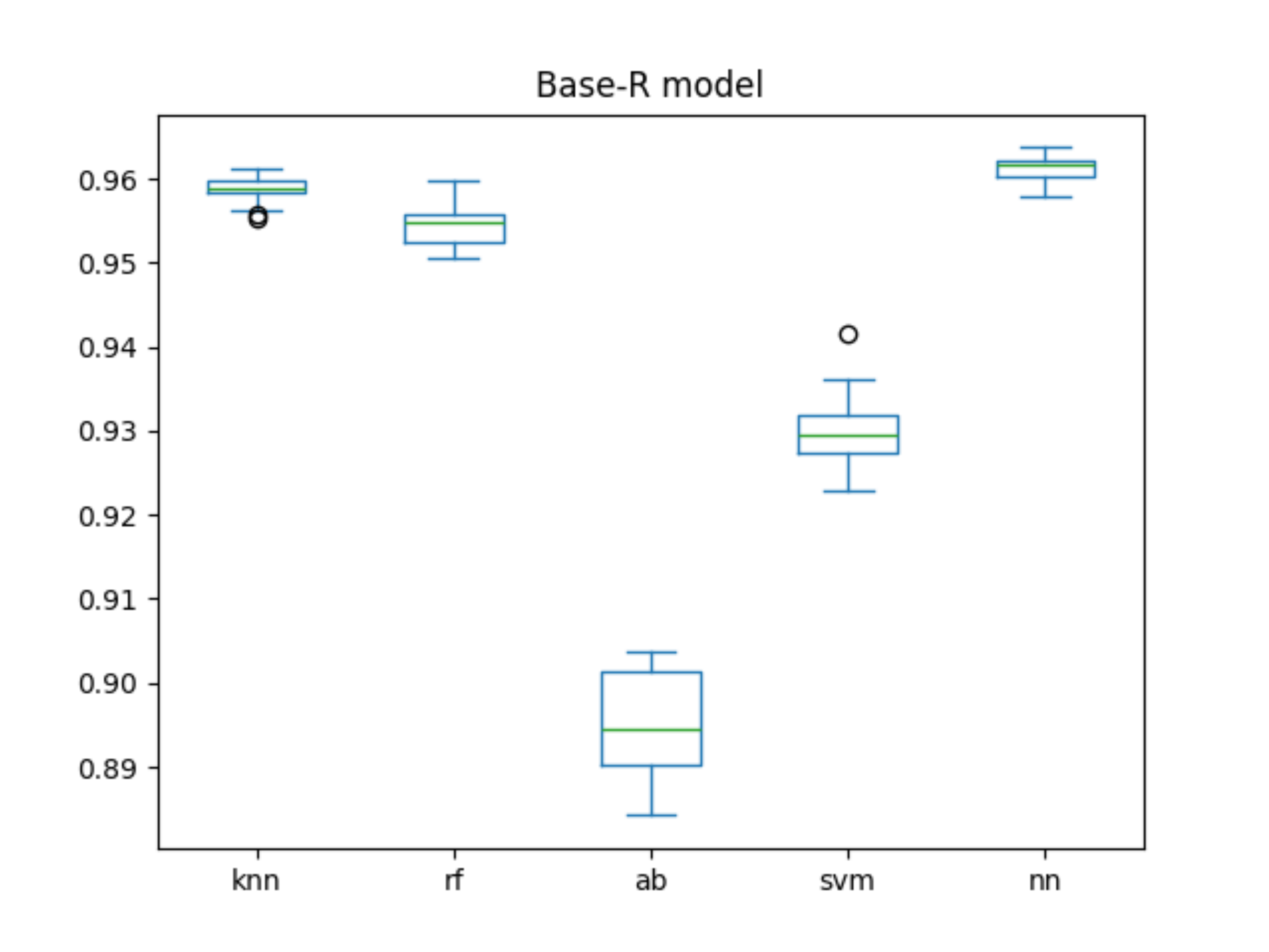
Method

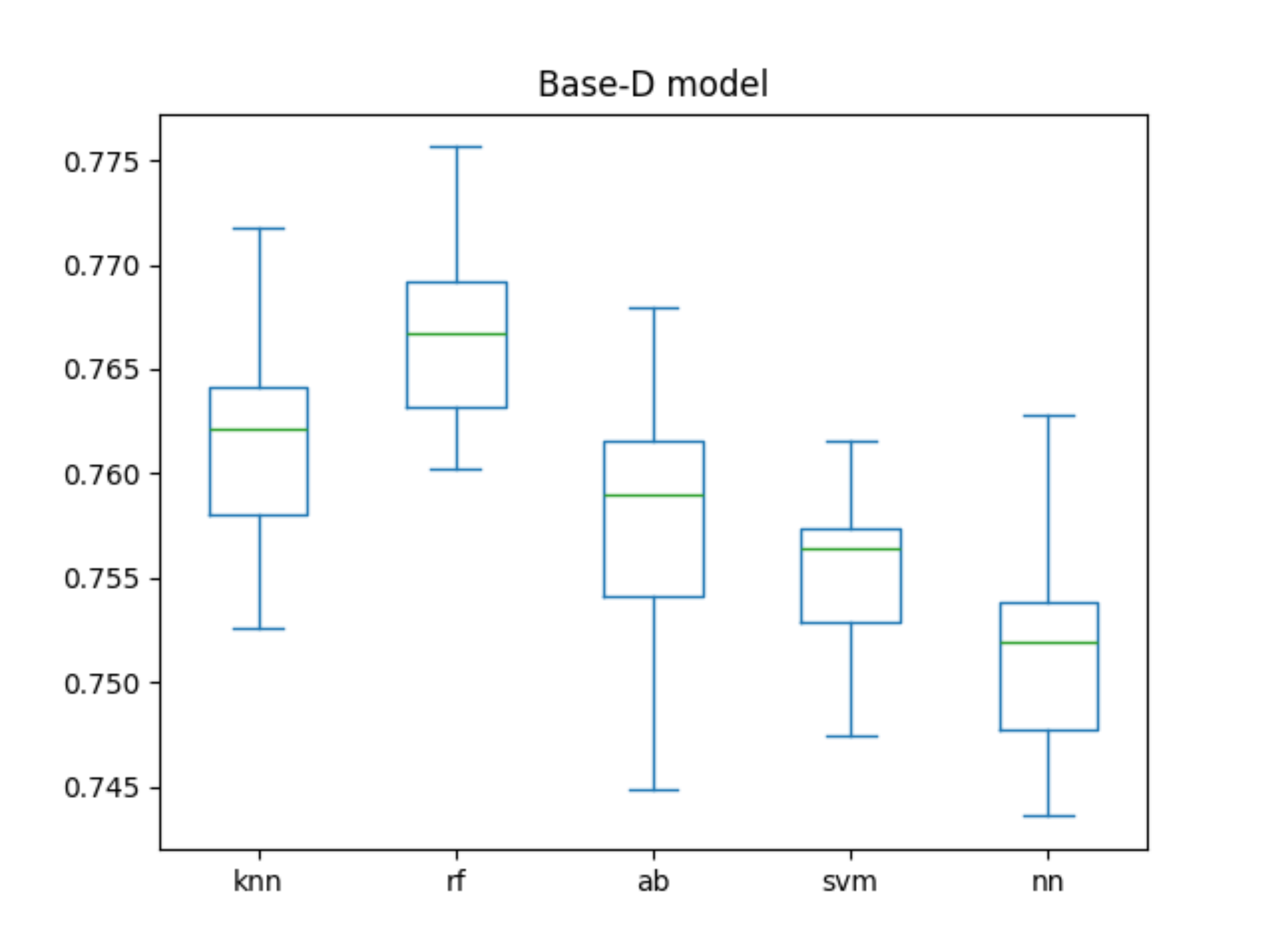
- 2 sets of models: base-R (R in 2019) and base-D (D in 2019)
- For base in {base-D, base-R}:

```
Do 30 times: #Monte Carlo Cross Validation Loop
    Split sample randomly into train/test (80/20)
    Use SMOTE to generate balanced categories for training data

    For model in {K Nearest Neighbors, Random Forest, ADABoost, SVM, Neural Net}:
        Cross-validate parameters on training data (10 fold) (see Annex for parameters)
        Re-fit model with best parameters, generate balanced accuracy on test
        If balanced accuracy > previous results:
            Save results as best model parameters

For model in {K Nearest Neighbors, Random Forest, ADABoost, SVM, Neural Net}: # Final evaluation
    Retrieve parameters obtained from best results of above loop
    Split sample randomly into train/test (80/20)
    Use SMOTE to generate balanced categories for training data
    Re-fit model to training data
    Generate diagnostics (F1, Matthews correlation, balanced accuracy) on test (robust to unbalanced sample)
```

CV Results (Balanced Accuracy)

Best model: Neural Net (Base-R), Random Forest (Base-D)
Confirmed by t-tests for difference in means vs next-best model (Annex)





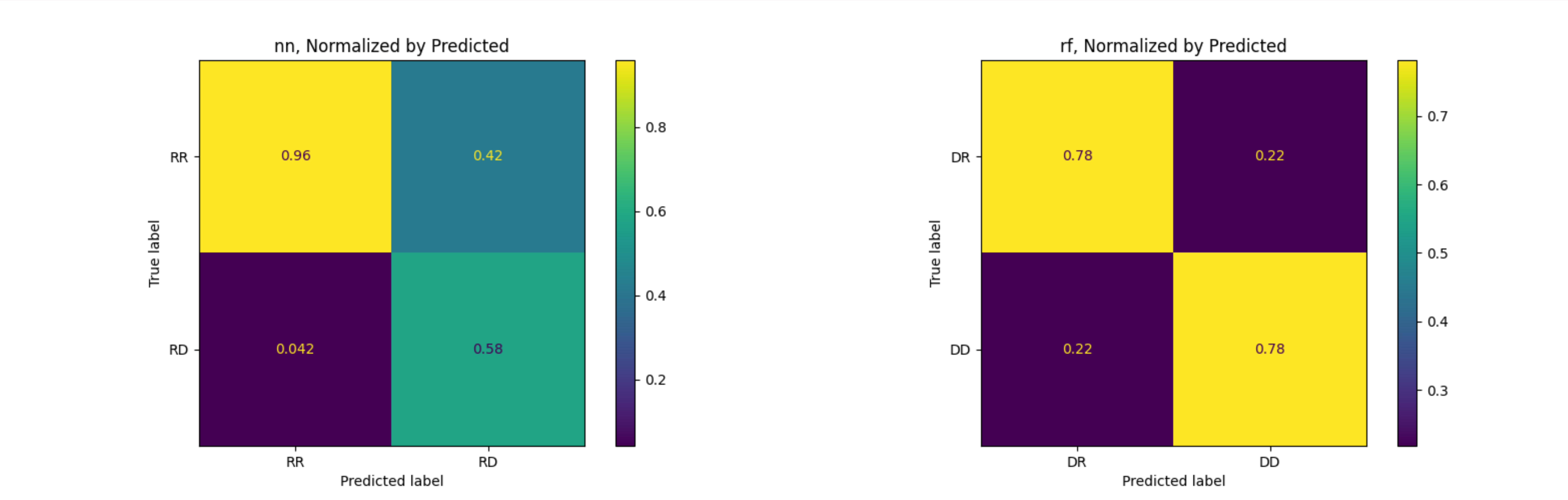
Model Diagnostics

Estimated on New Train/Test Split

Model	F1 score	Matthews	Balanced Accuracy
	Base-D model		
KNN	0.7807	*0.5619*	*0.7809*
Random Forest	*0.7853*	0.5613	0.7806
ADABoost	0.7539	0.4971	0.7485
SVM	0.7789	0.5507	0.7753
NN	0.7254	0.4329	0.7162
	Base-R model		
KNN	0.7059	0.6557	*0.8659*
Random Forest	*0.7253*	*0.6765*	0.8475
ADABoost	0.6666	0.6145	0.8616
SVM	0.6857	0.6331	0.8601
NN	0.6602	0.6006	0.8373
	Best results in each category are italicized.		

Confusion Matrices

Conditional on Base-year Party Majority



Large Gain in Accuracy for D->R

Smaller for R->D

	Raw Data P(flip | 2019)	Modeled P(flip|flip label)	Gain
R->D	15%	58%	43%
D->R	48%	78%	30%

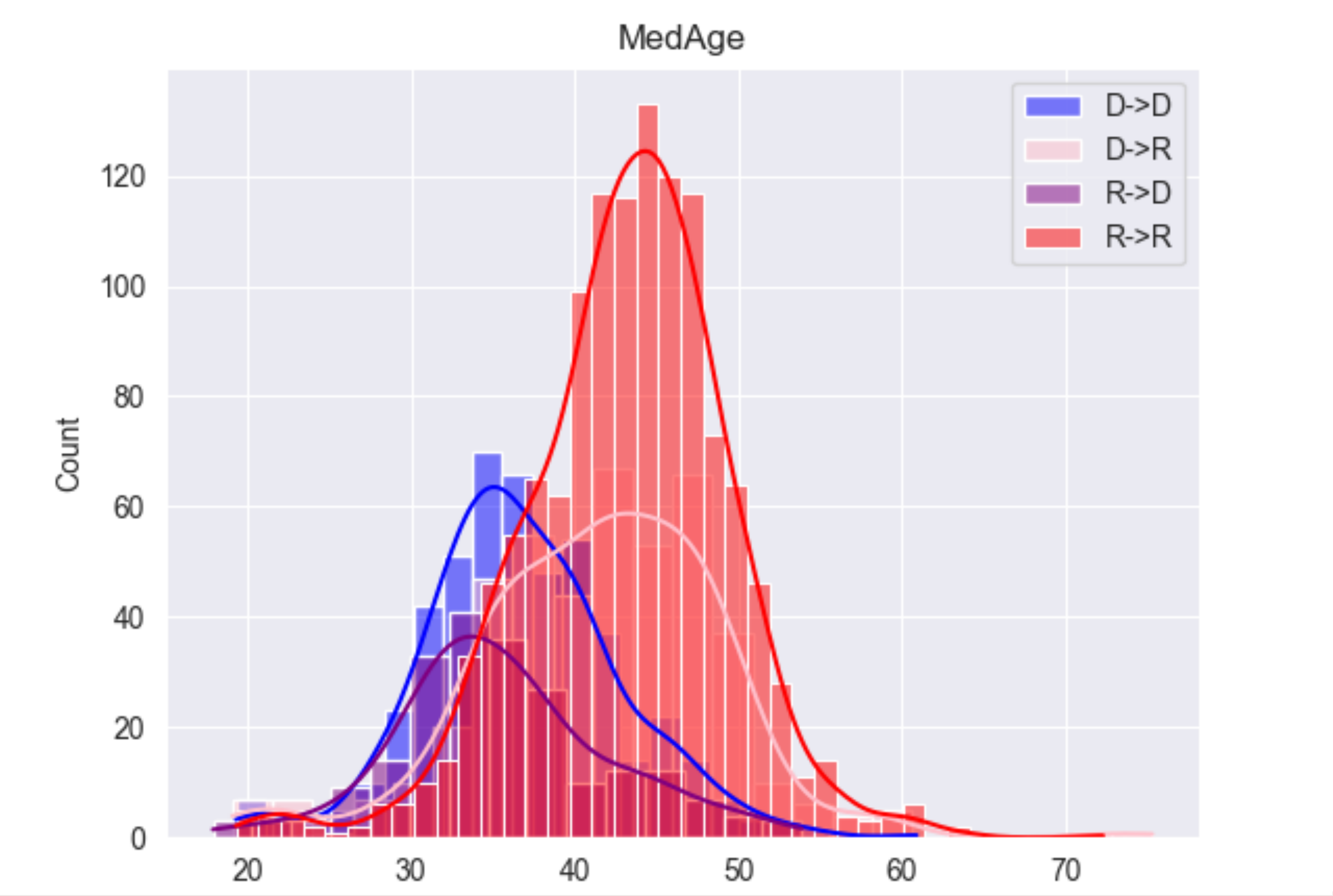
Conclusions

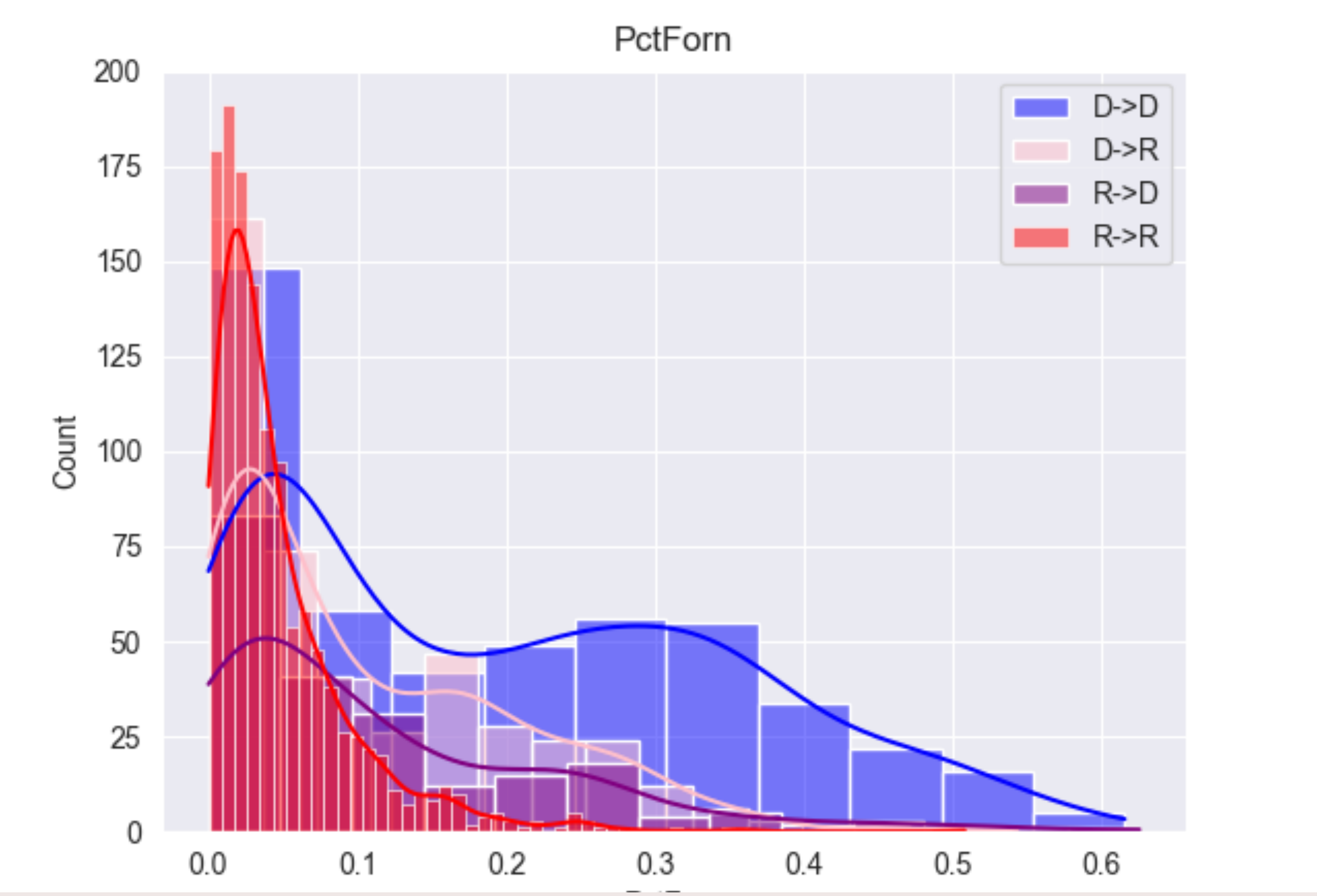
- Substantial improvements in accuracy using public Census data to predict flips:
  - Republican->Democratic precincts: 43 percentage point gain
  - Democratic->Republican precincts: 30 percentage point gain
- Future research:
  - Additional election cycles
  - More focused elections (look at one office vs all offices together)
  - Causal analysis: measure campaign effort
- Code: https://github.com/Charlie-Kramer/precinct_flips

Annex Slides

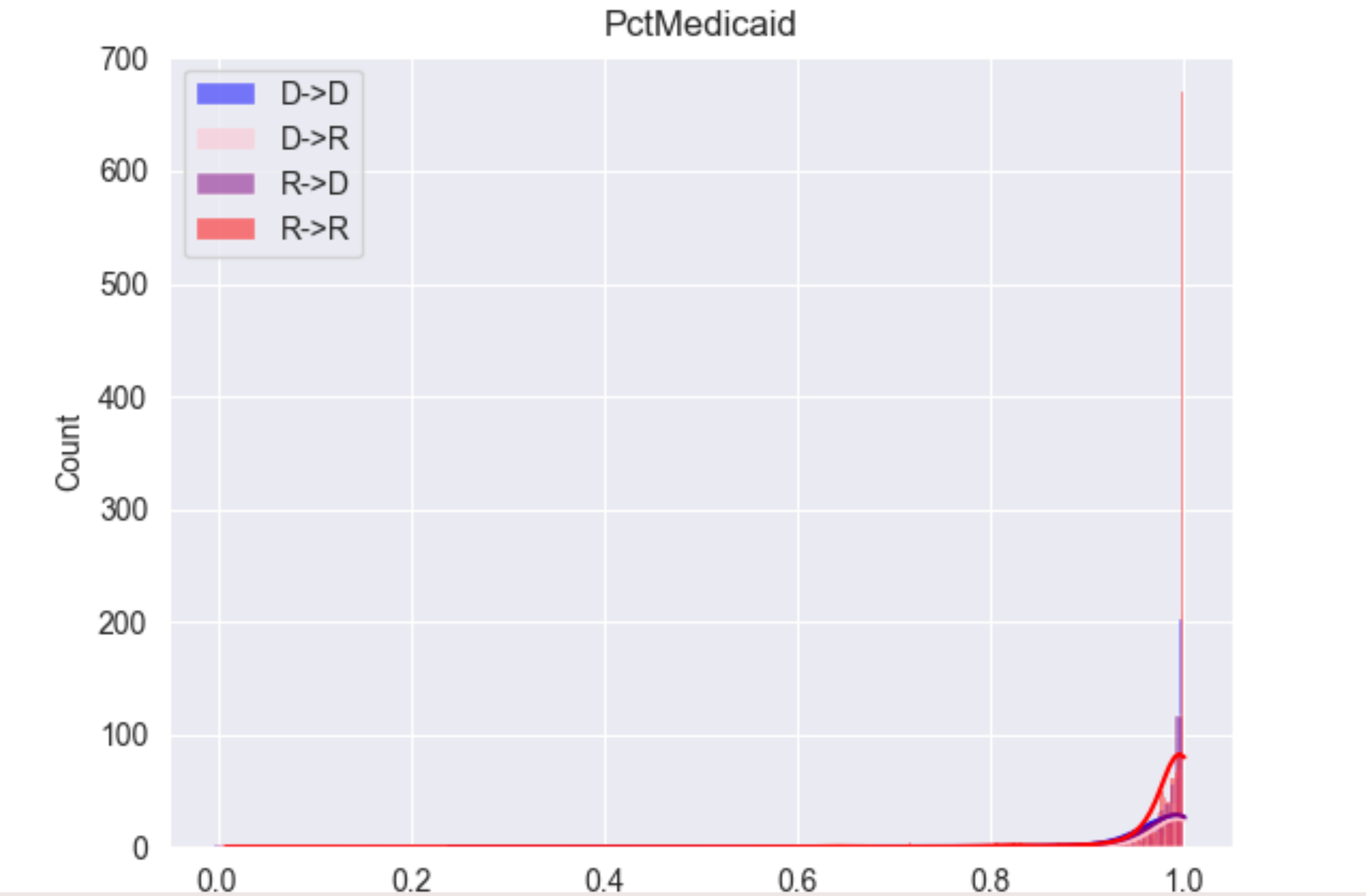
- Distribution of Demographic Variables by Flip Category
- Parameters Chosen by Cross-Validation
- Full Set of Confusion Matrices

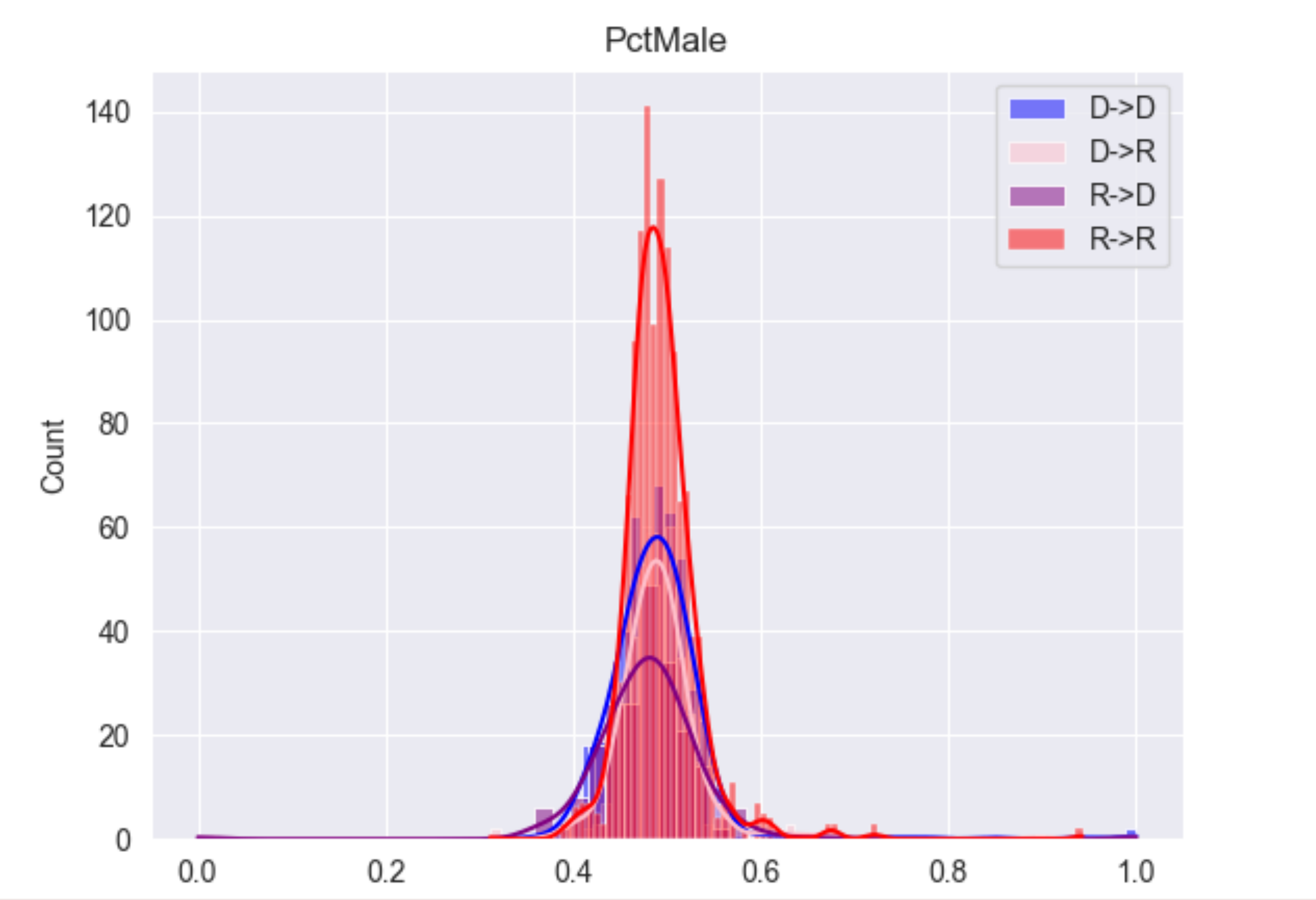
Demographic Variables by Category



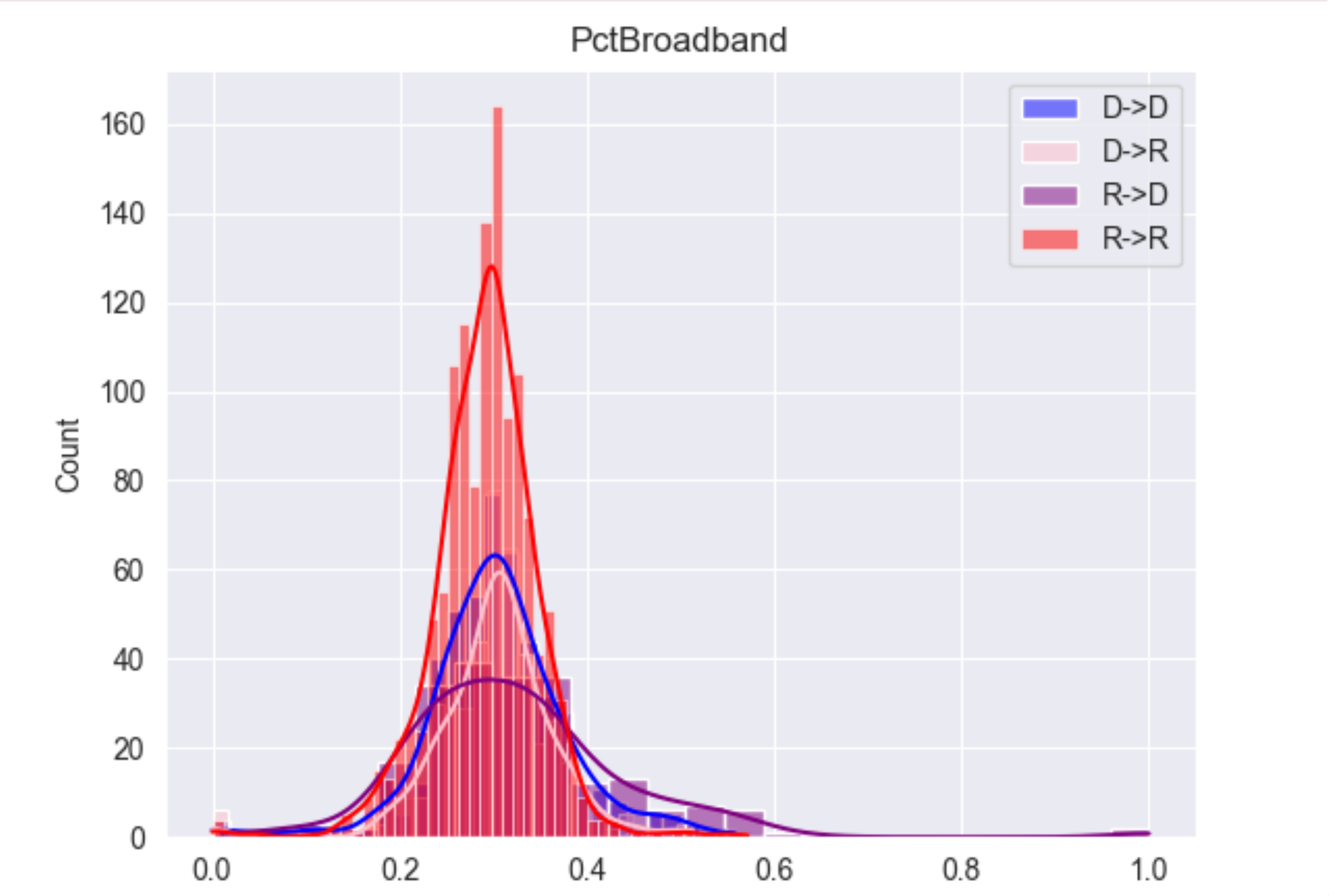


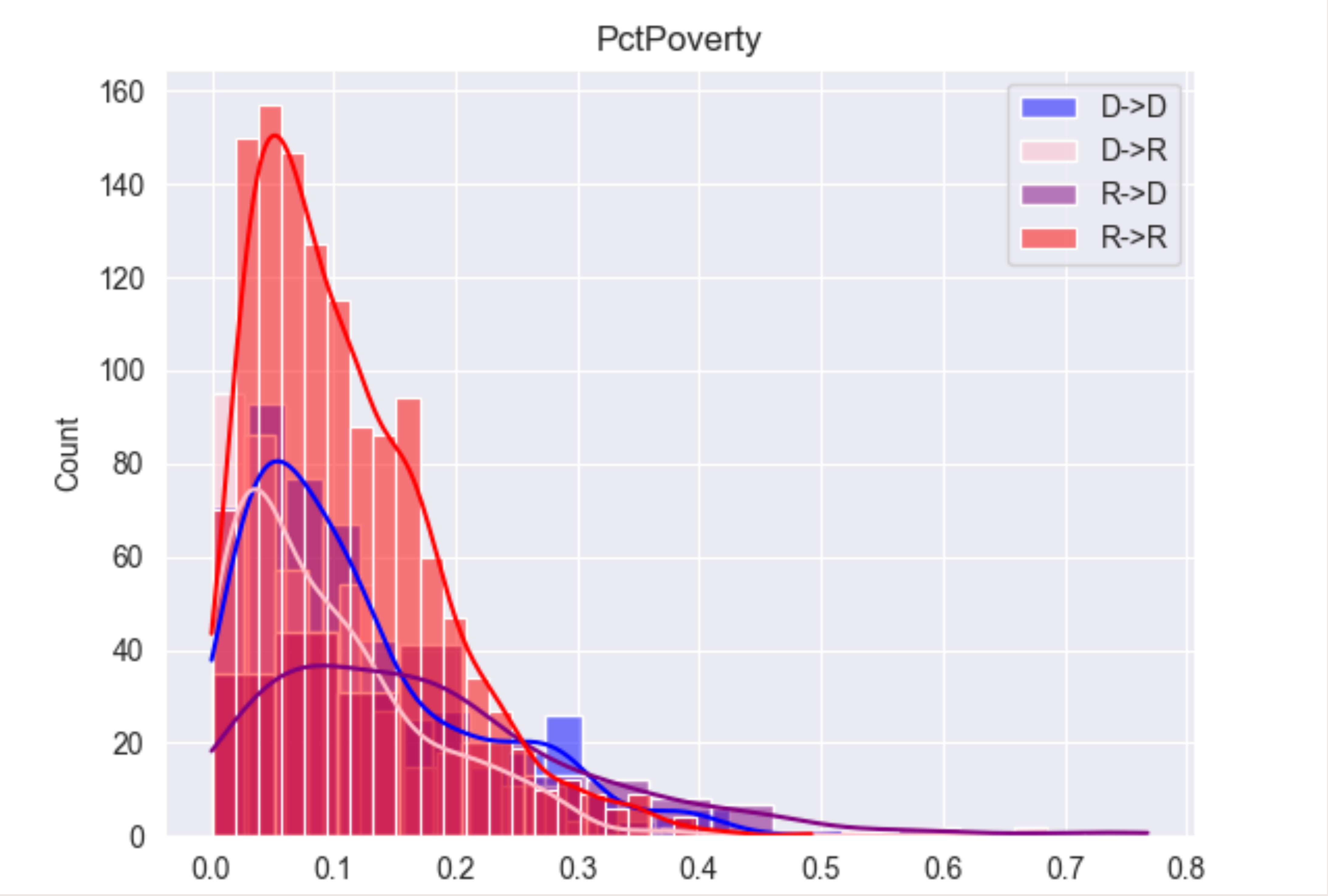
Demographic Variables by Category





Demographic Variables by Category



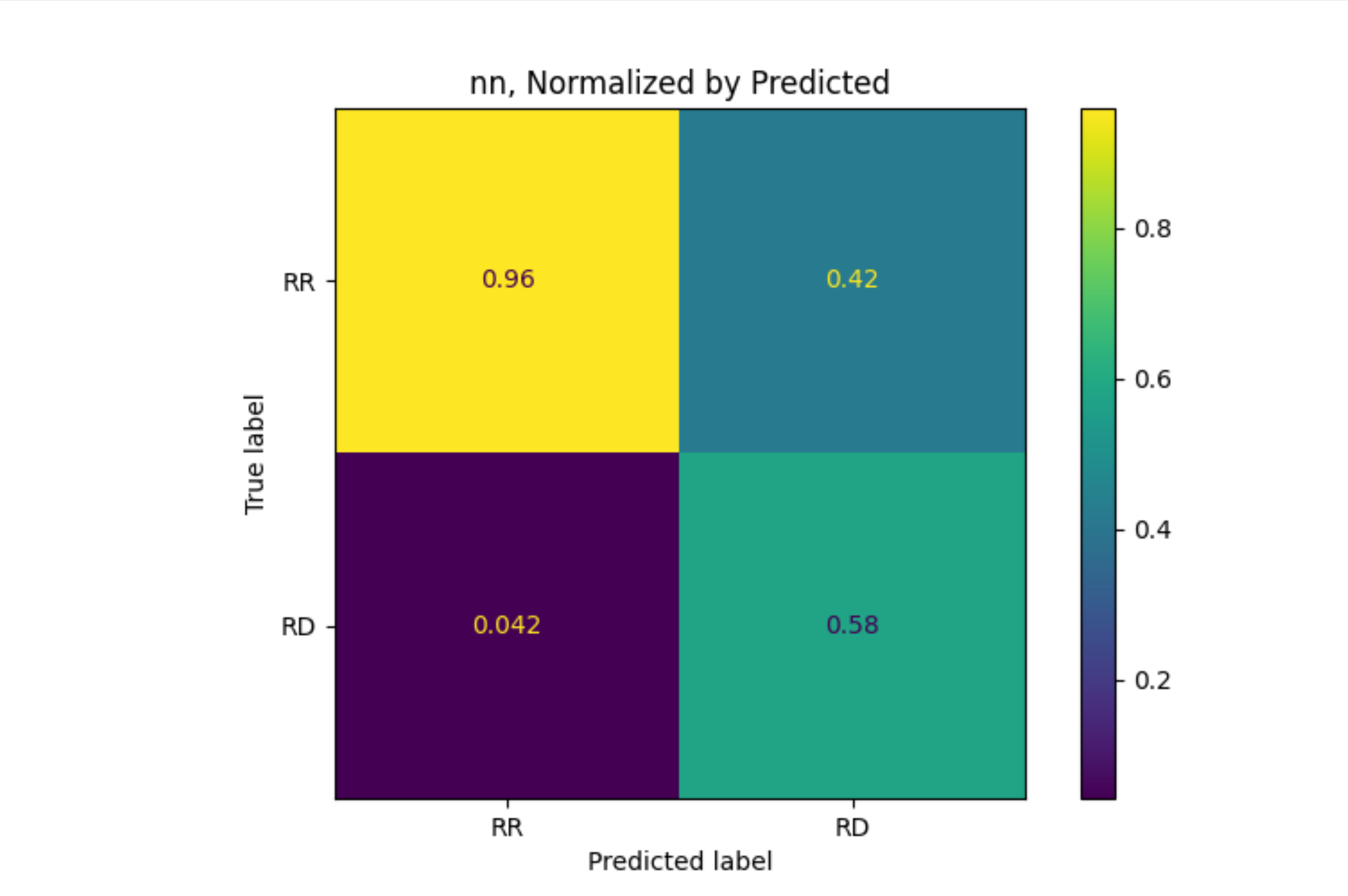


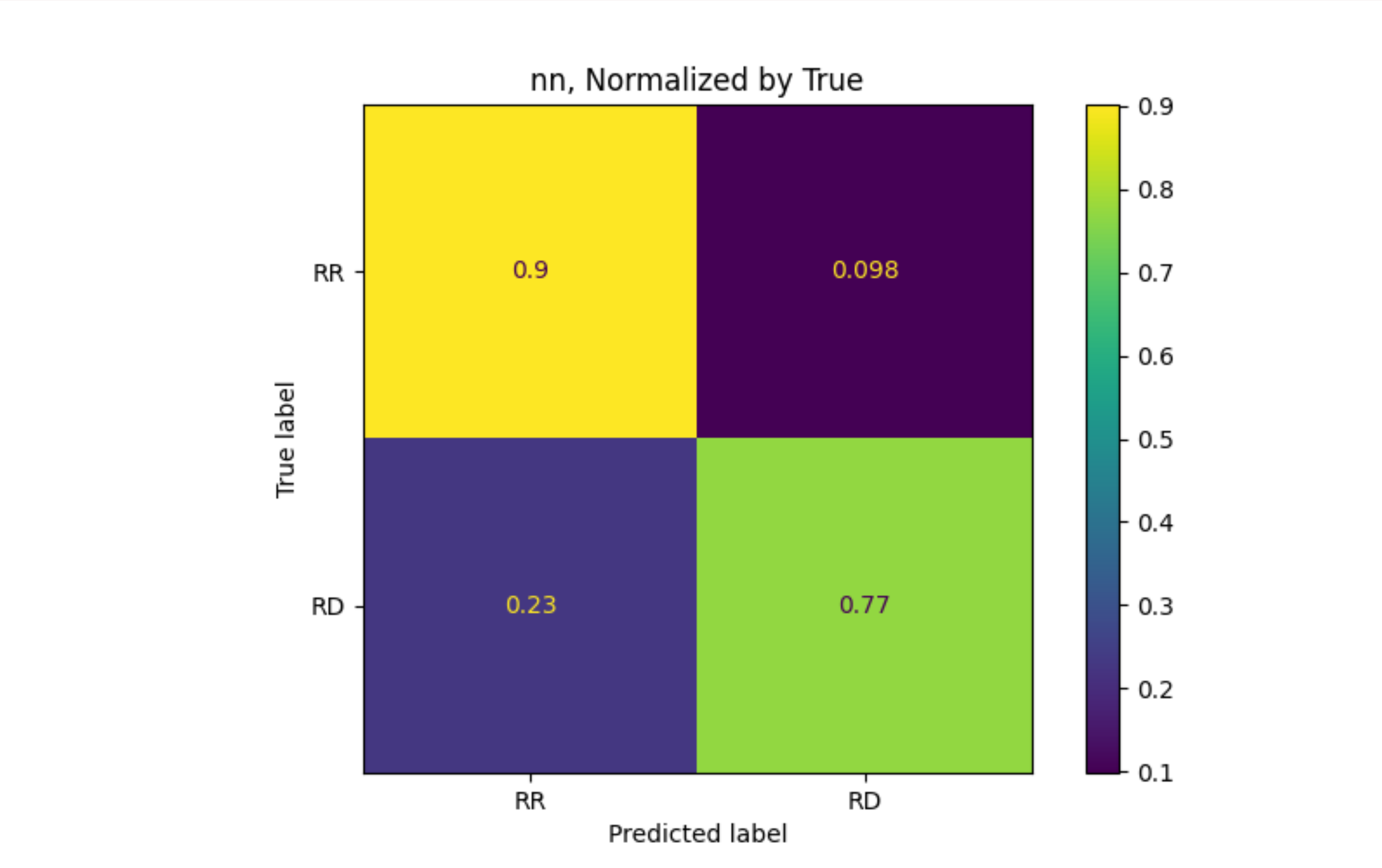
Parameters Chosen by CV

Model	**Key Parameters (CV grid; bold = parameter chosen by 10x10-fold CV)**
Submodel	Base-D model
KNN	Number of neighbors (1-20; **17**); P-exponent on distance metric (**1**, 2); weights (**uniform**, distance)
Random Forest	Number of estimators (20, 30,,..,200; **120**), criterion (gini, **entropy**, log_loss), minimum samples for split (**2**,4,6), minimum samples per leaf (1,3,**5**)
ADABoost	Number of estimators (5, 10, 15,..,100; **55**), learning rate (.25, .75, **1**, 2, 4),
SVM	C (**.5**,1,2), kernel (**linear**, poly, rbf, sigmoid), gamma (**scale**, auto)
NN	Activation(**tanh**, relu),hidden layer sizes(50,**100**,200), learning_rate(**constant**, adaptive)
Submodel	Base-R model
KNN	Number of neighbors (1-20, **4**); P-exponent on distance metric (1, **2**); weights (uniform, **distance**)
Random Forest	Number of estimators (20, 30,..,200; **140**), criterion (gini, **entropy**, log_loss), minimum samples for split (**2**,4,6), minimum samples per leaf (**1**,3,5)
ADABoost	Number of estimators (5, 10,..,100; **90**), learning rate (.25, .75, **1**, 2, 4).
SVM	C (.5,1,**2**), kernel (linear, poly, **rbf**, sigmoid), gamma (scale, **auto**)
NN	Activation(tanh, **relu**),hidden layer sizes(50, **100**, 200), learning_rate(**constant**, adaptive)

Confusion Matrices

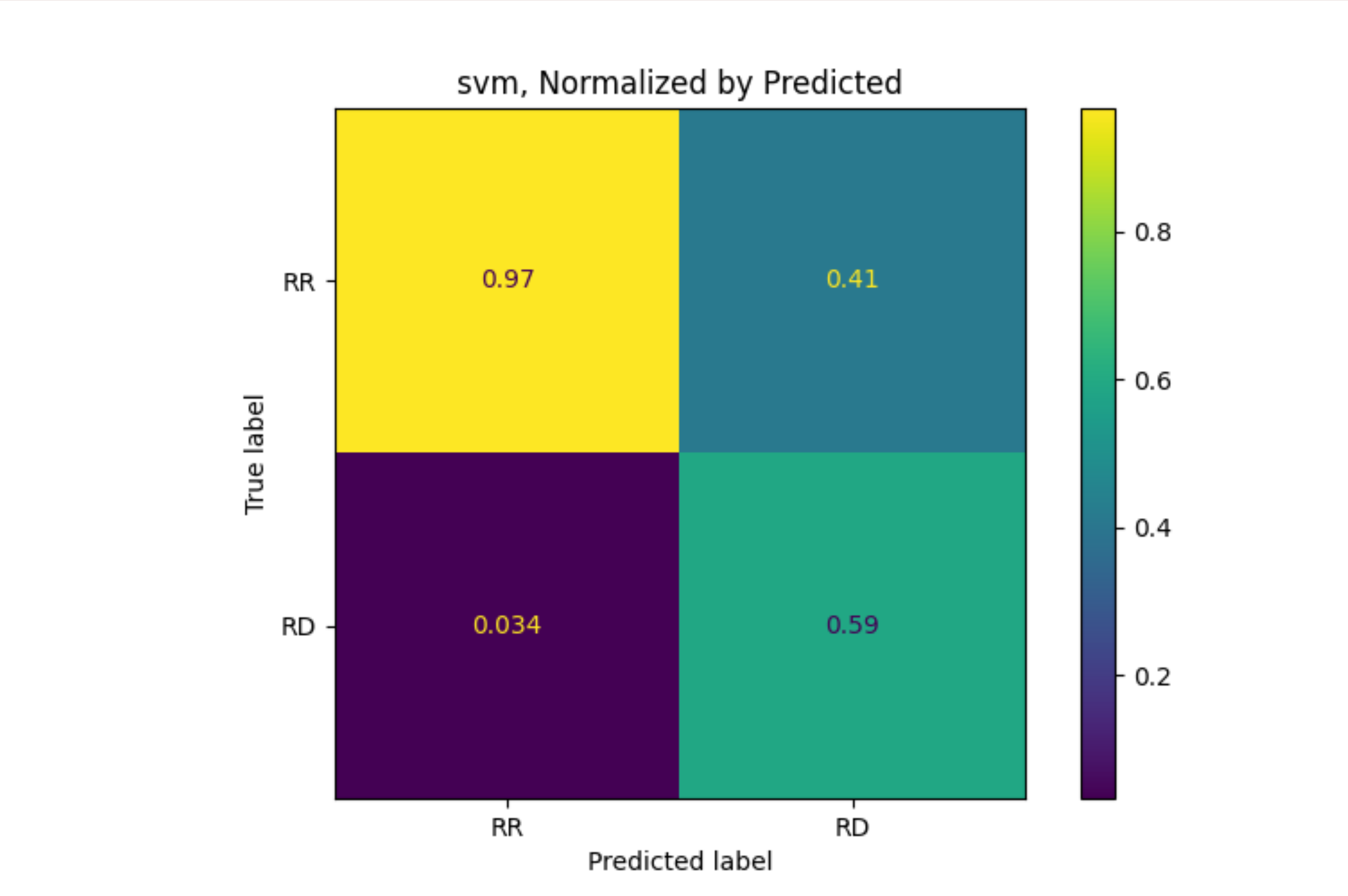
Base-R: Neural Net

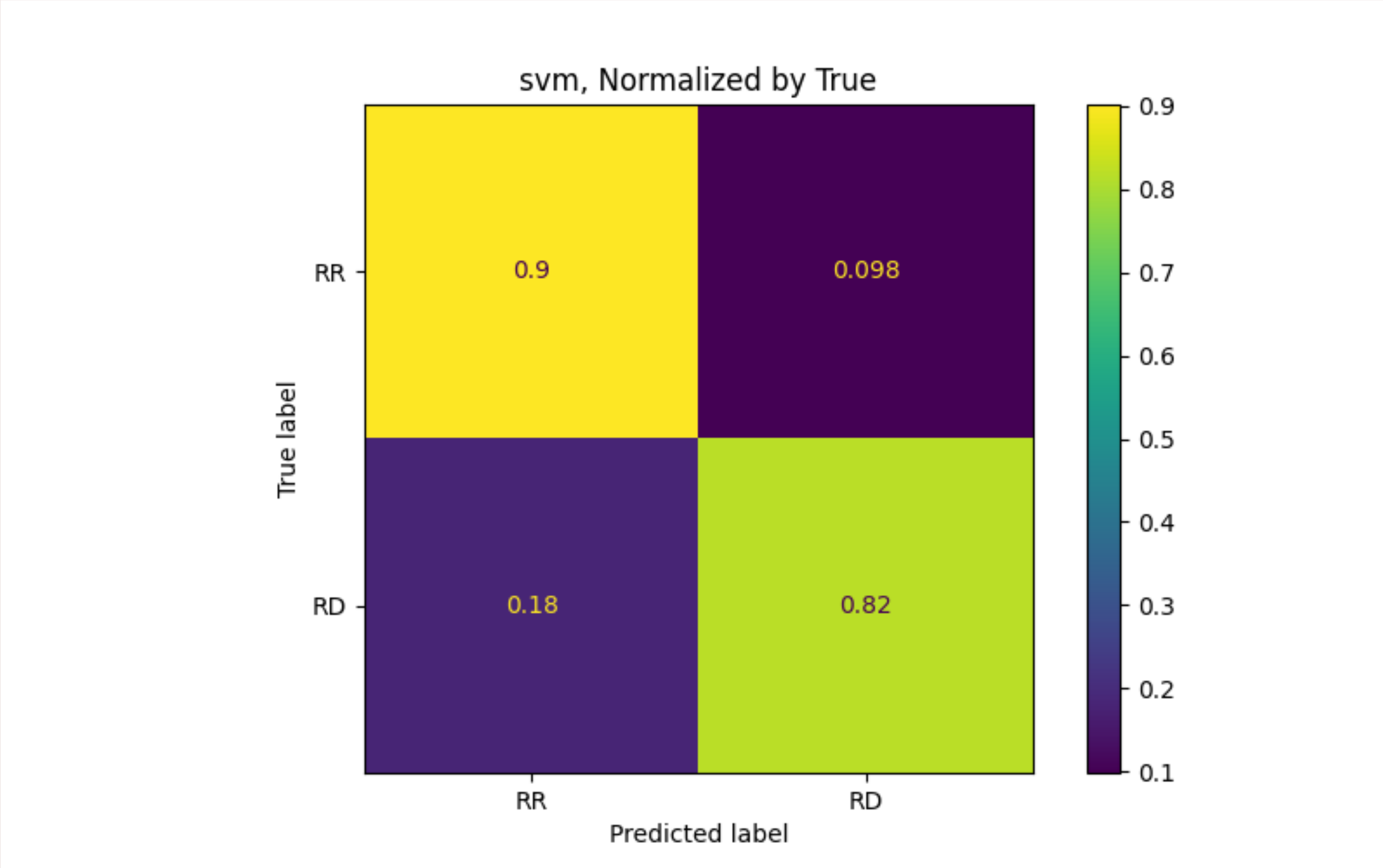




Confusion Matrices

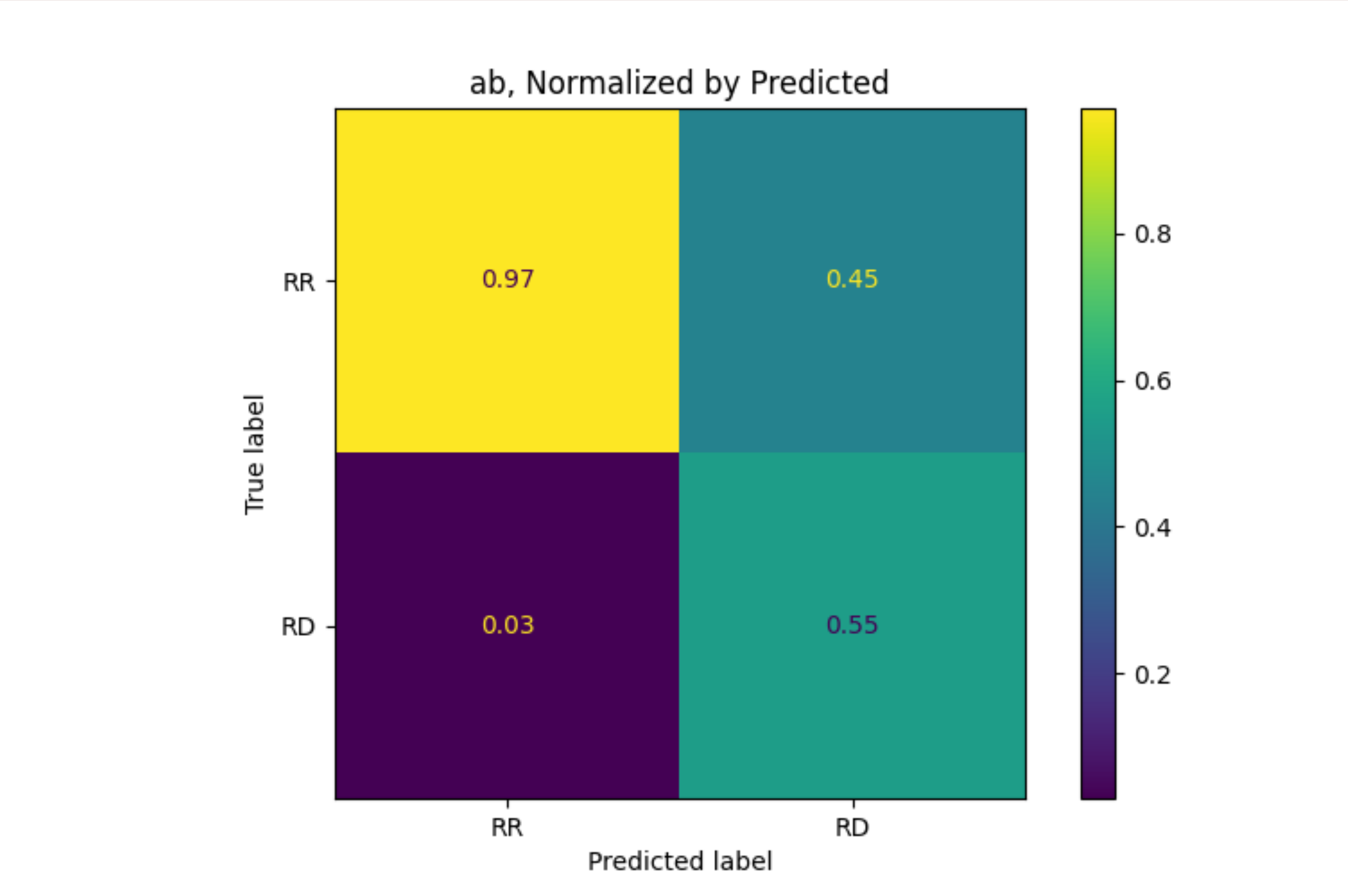
Base-R: SVM

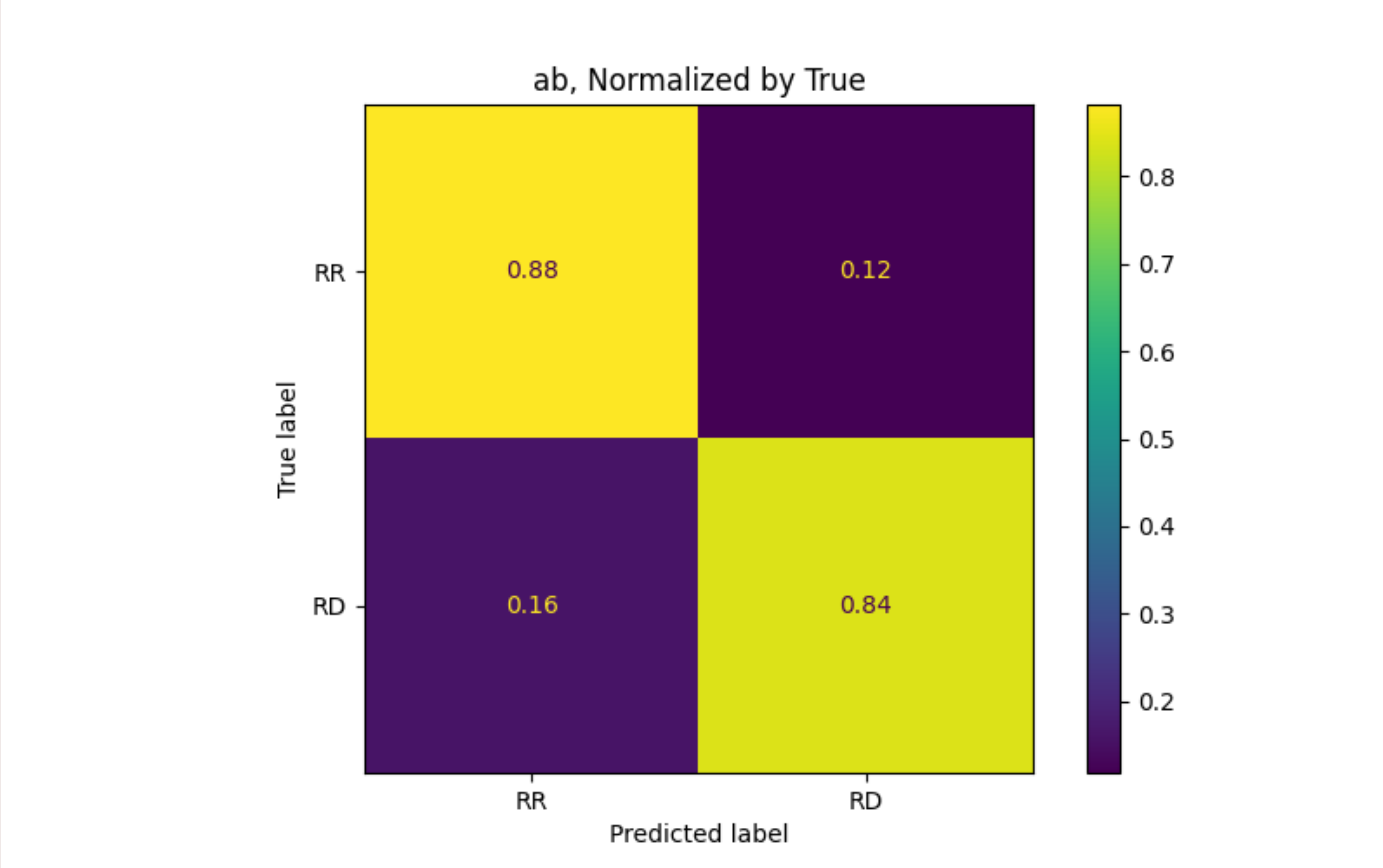




Confusion Matrices

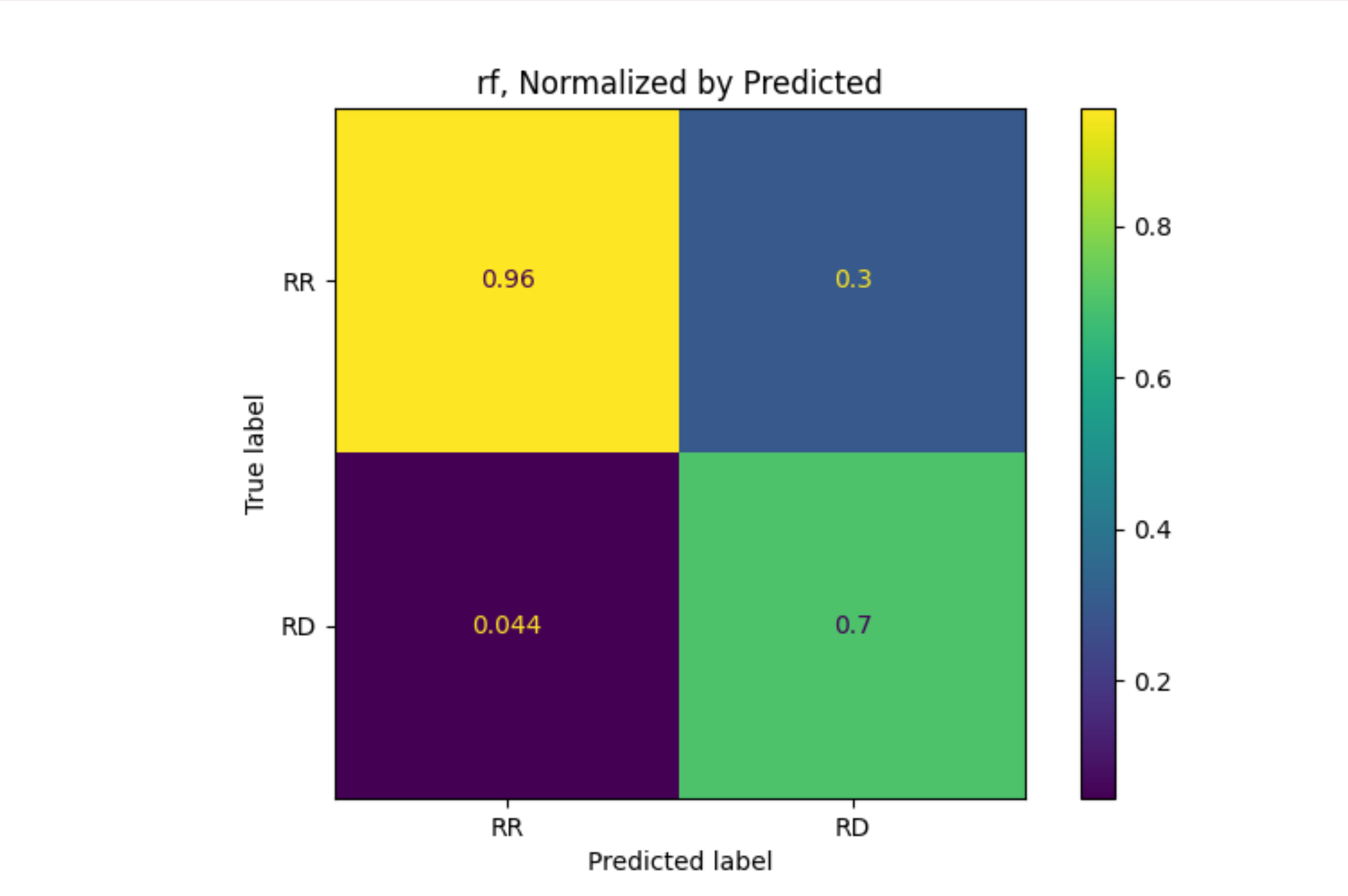
Base-R: ADABoost

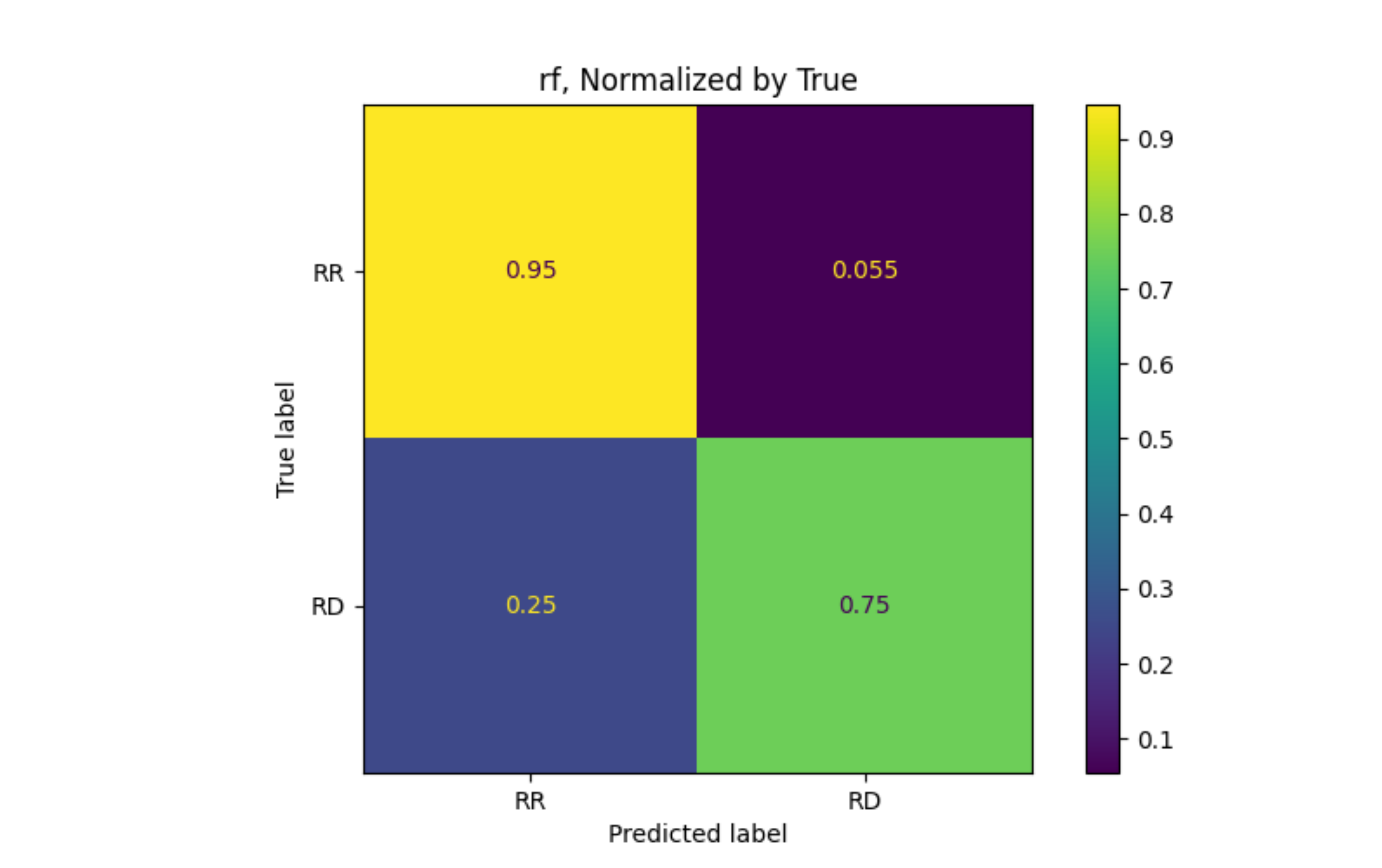




Confusion Matrices

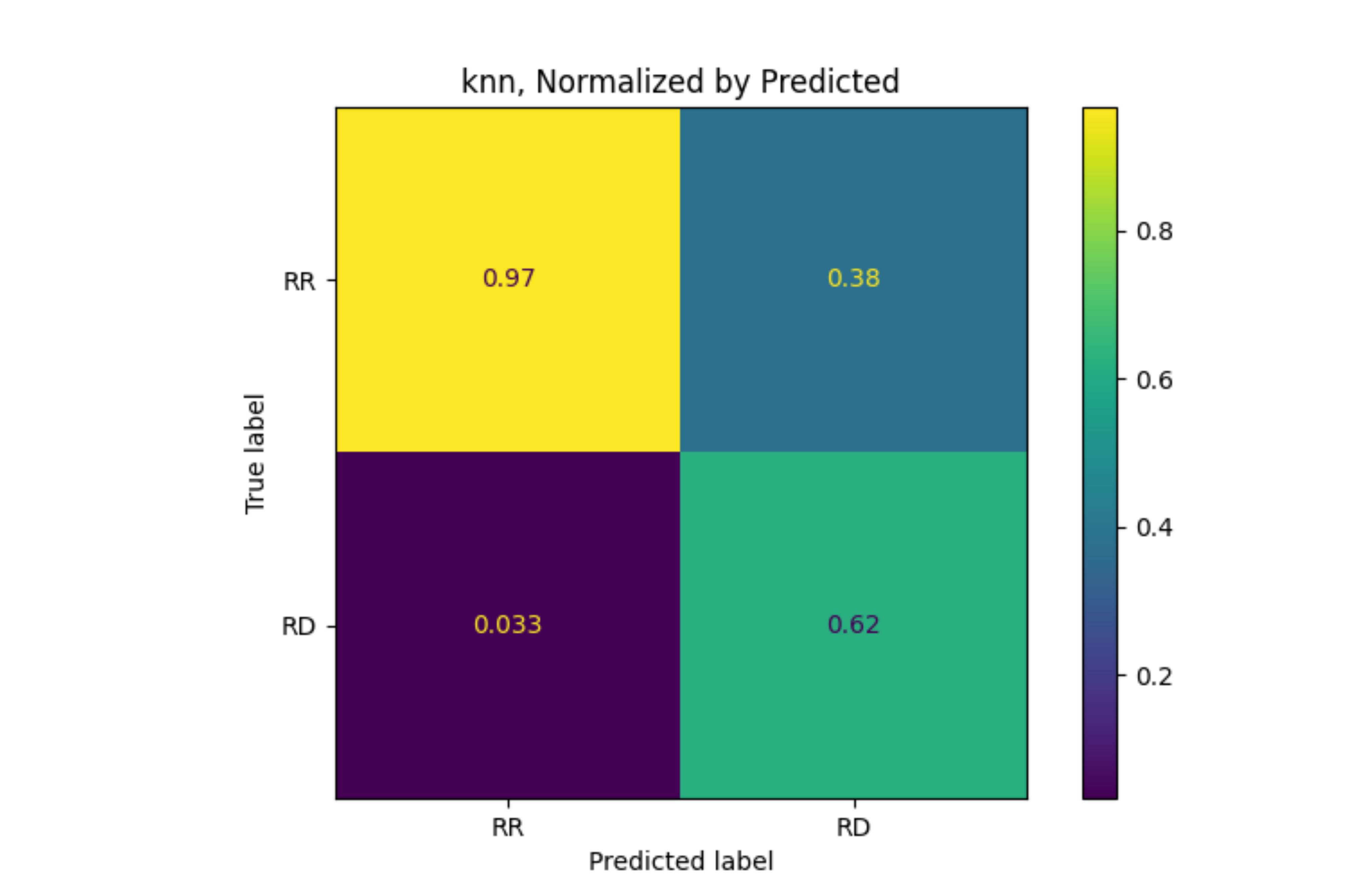
Base-R: Random Forest

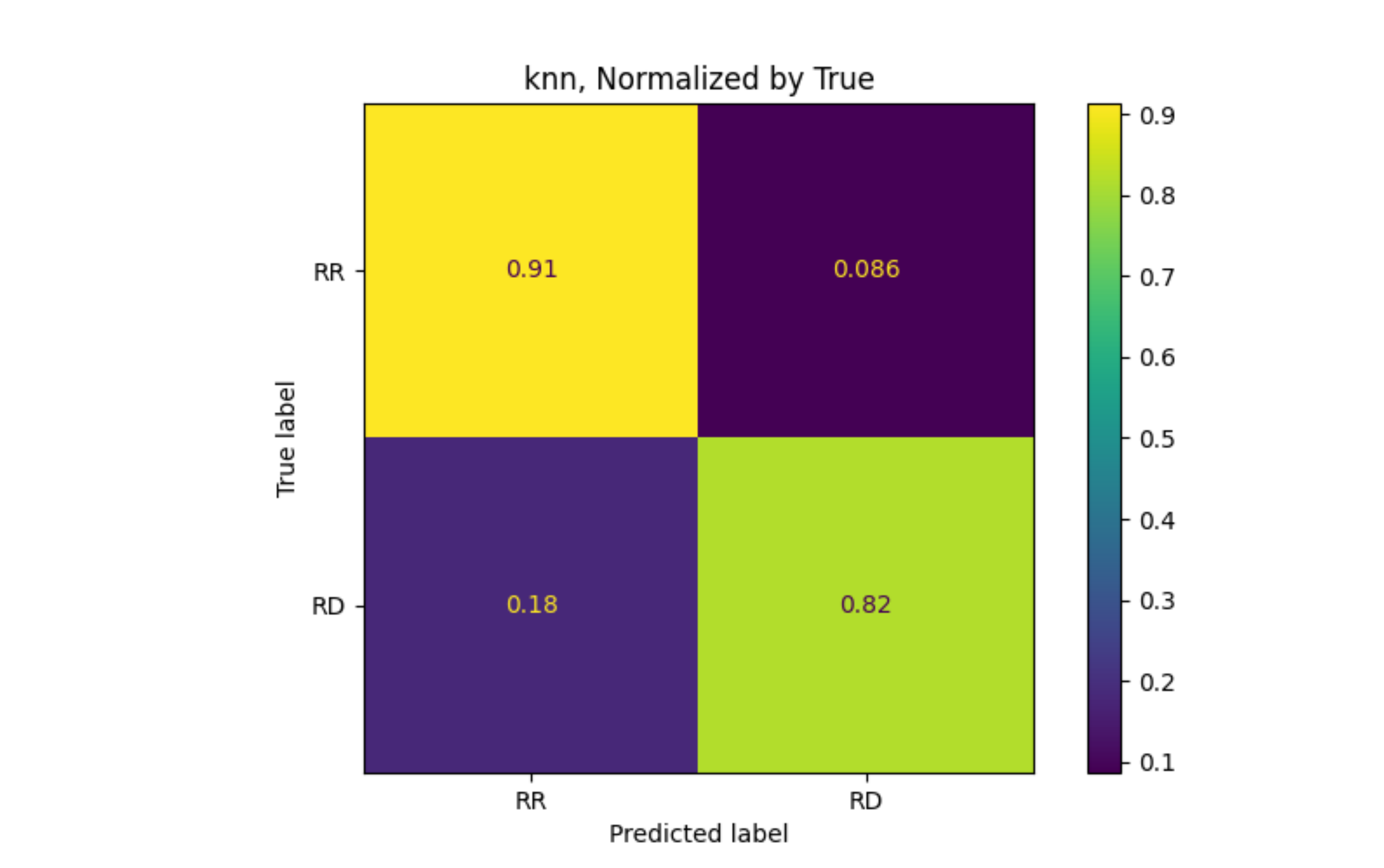




Confusion Matrices

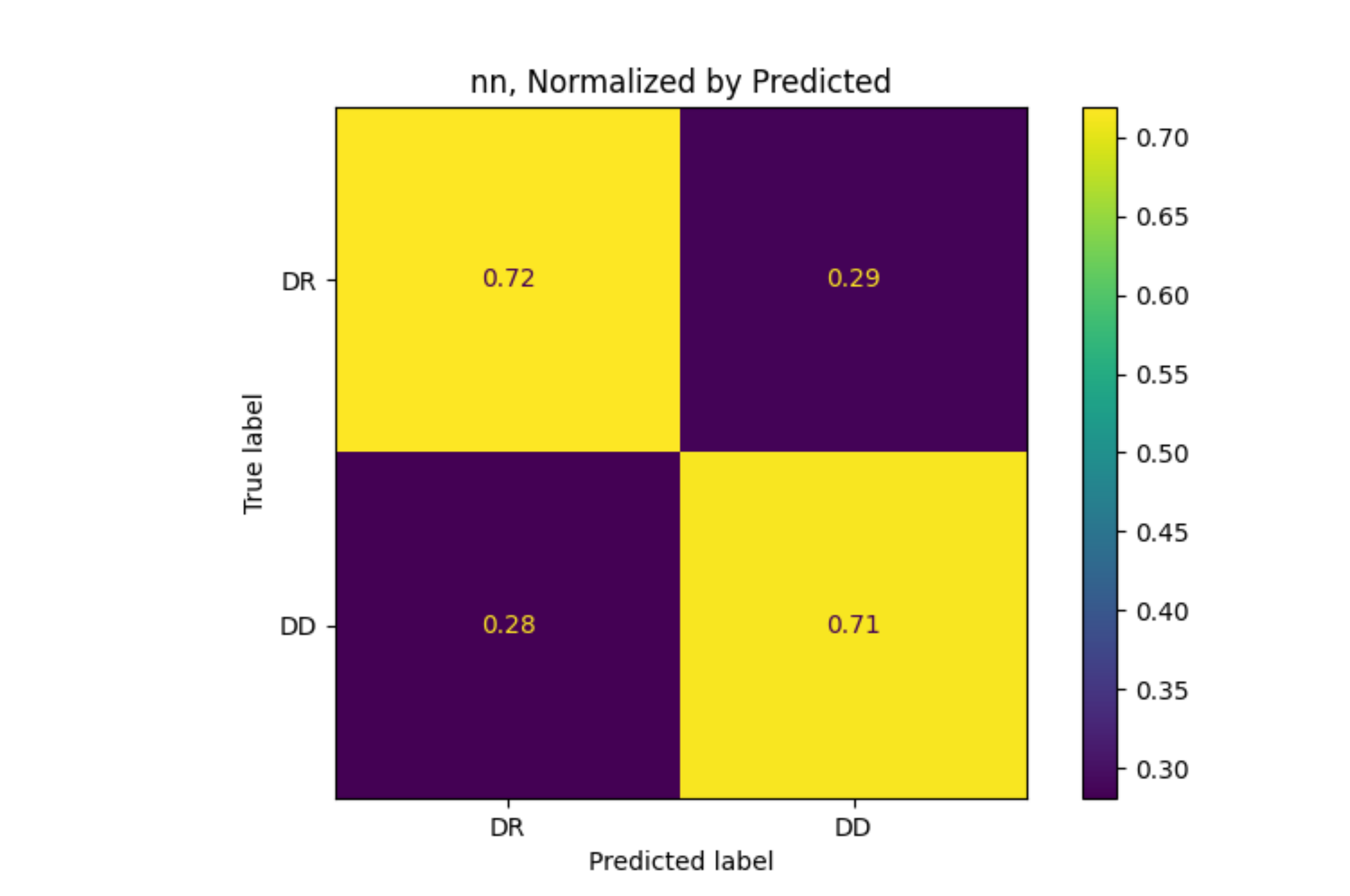
Base-R: KNN

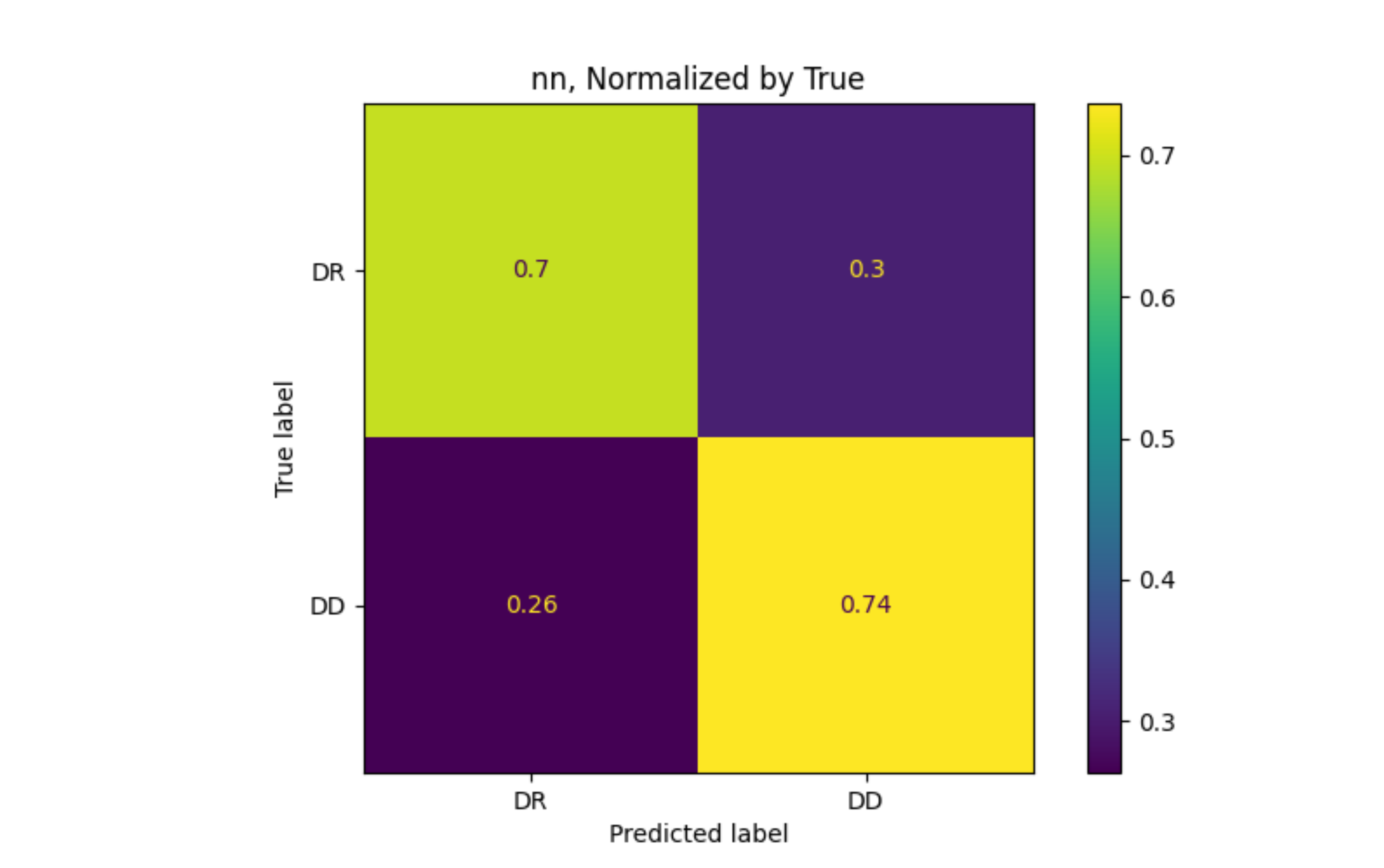




Confusion Matrices

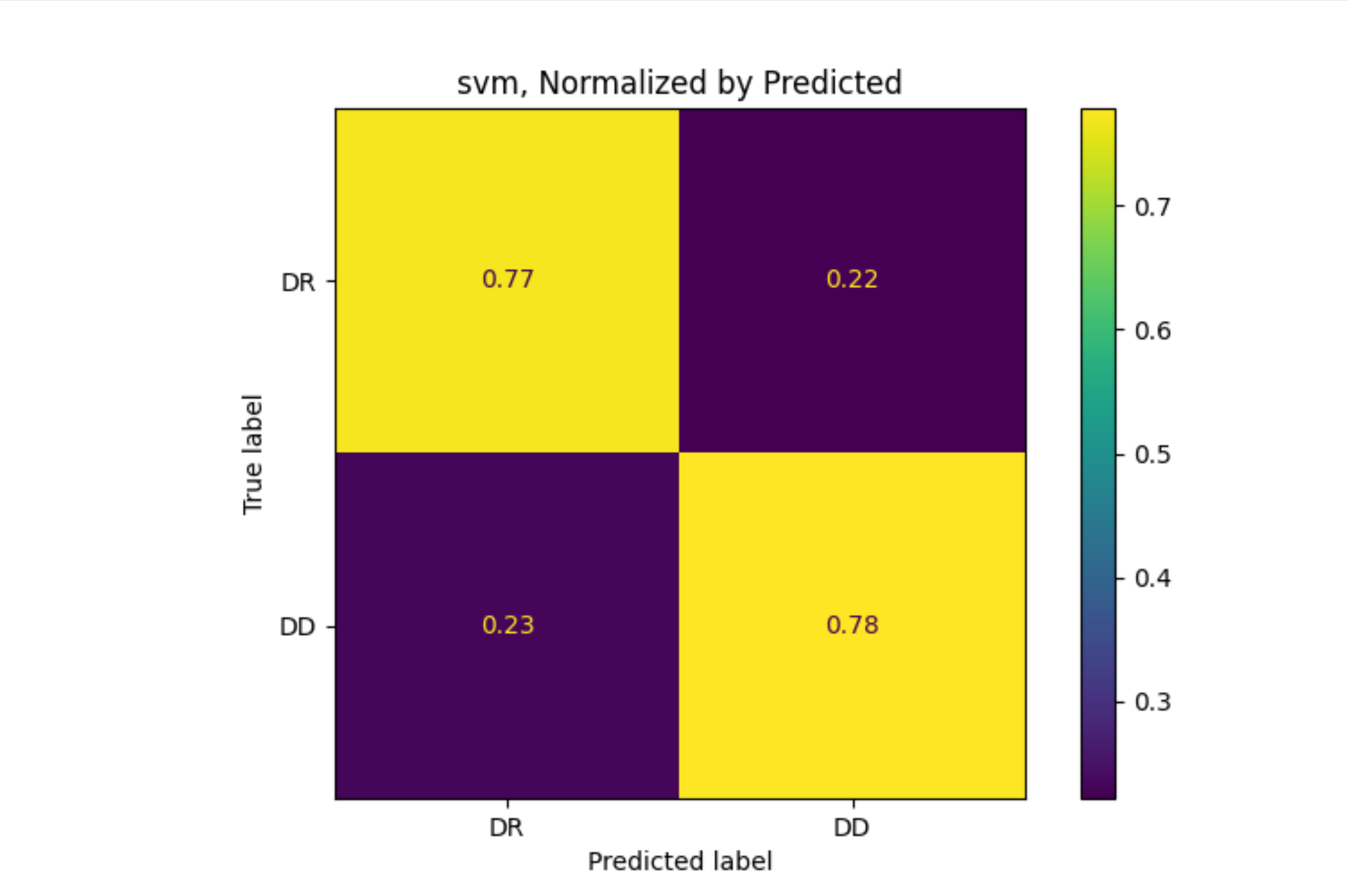
Base-D: Neural Net

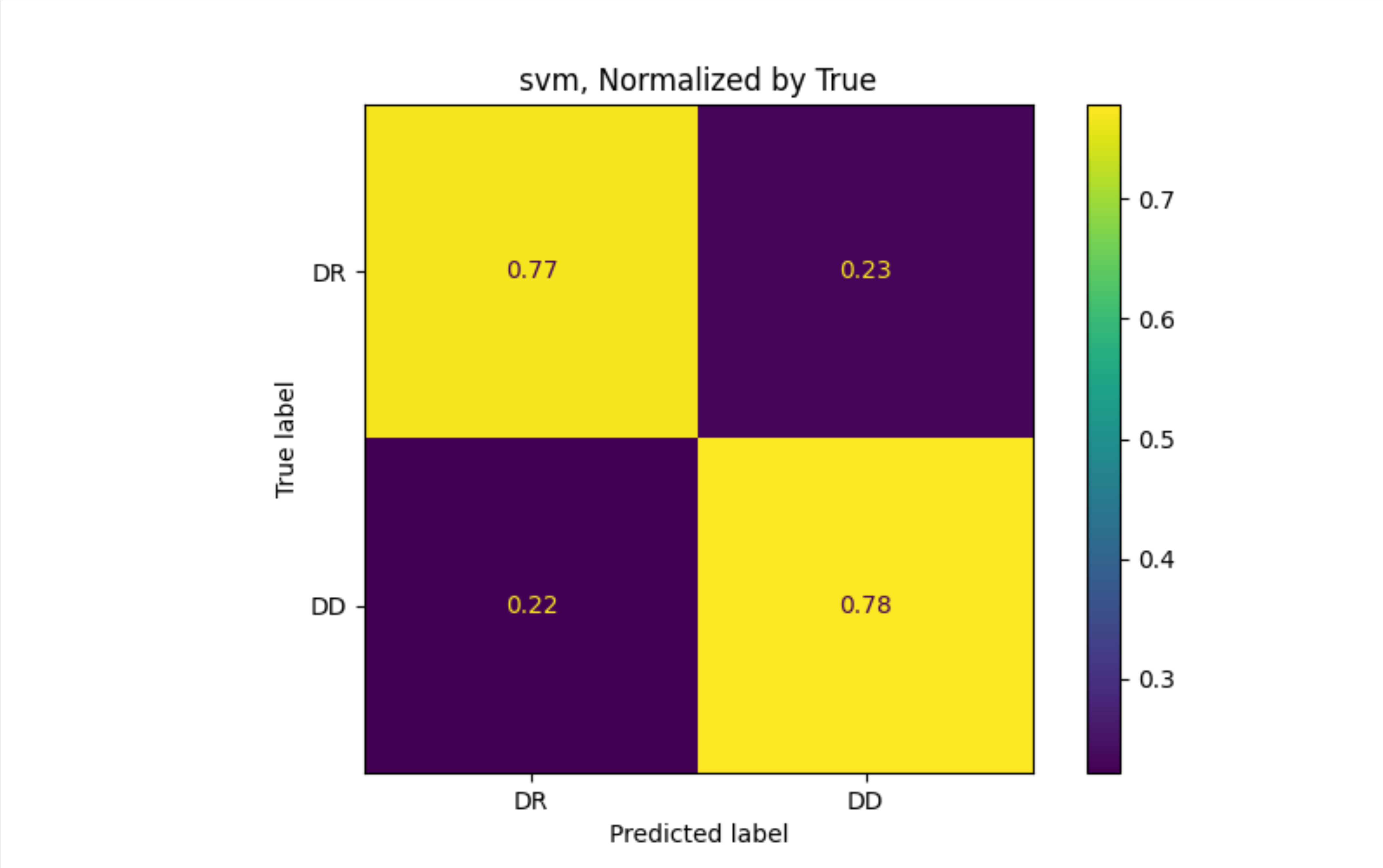




Confusion Matrices

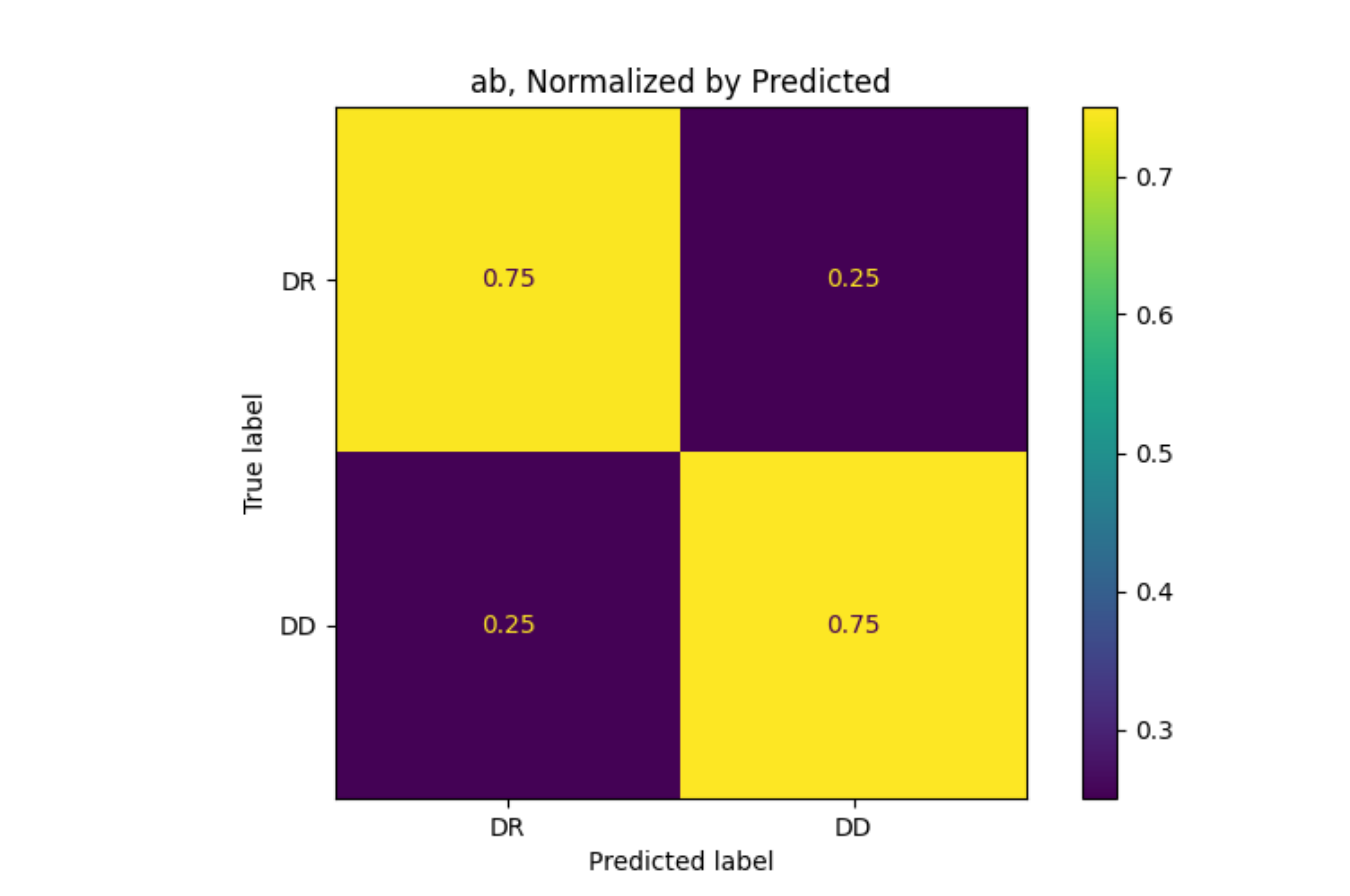
Base-D: SVM

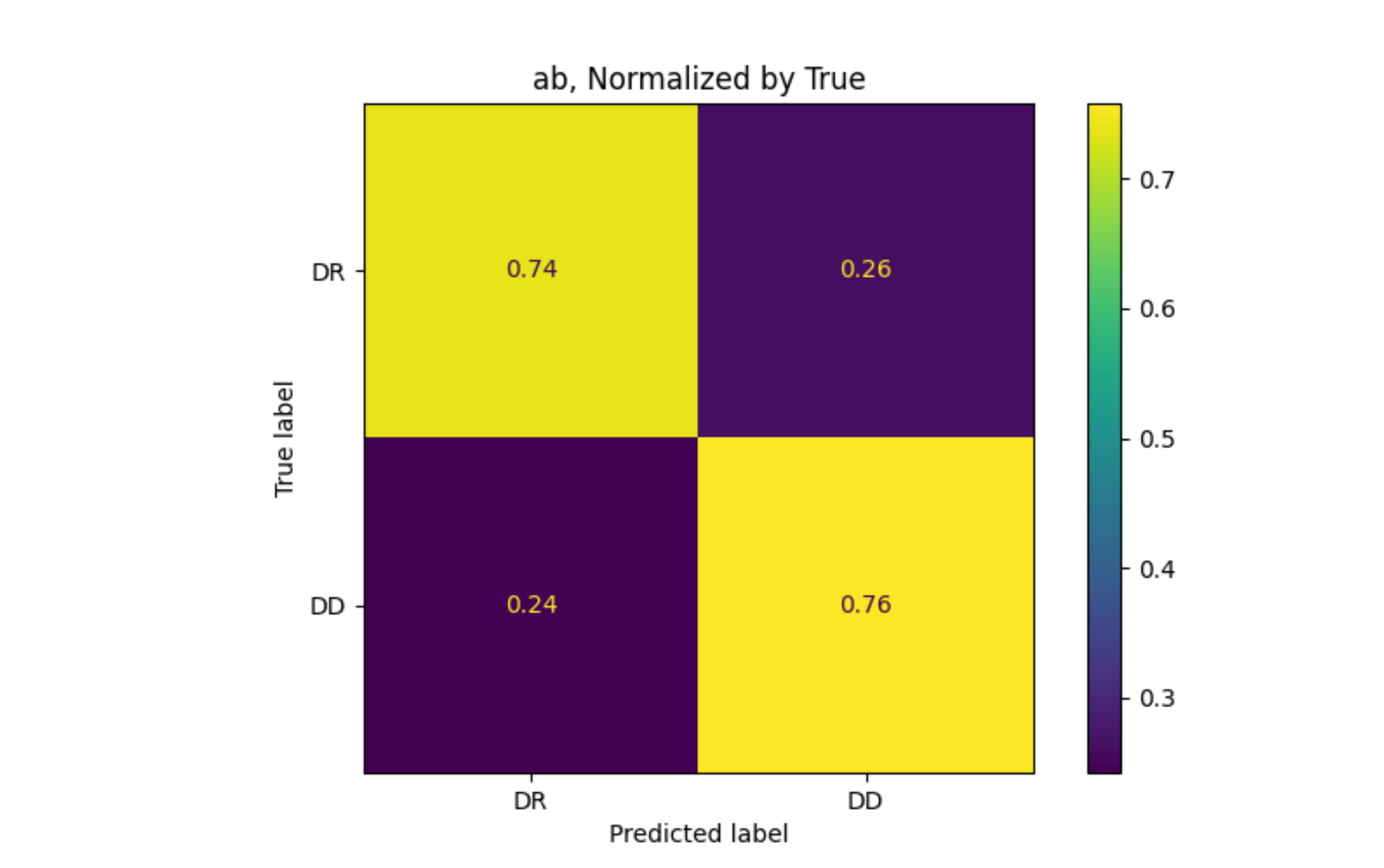




Confusion Matrices

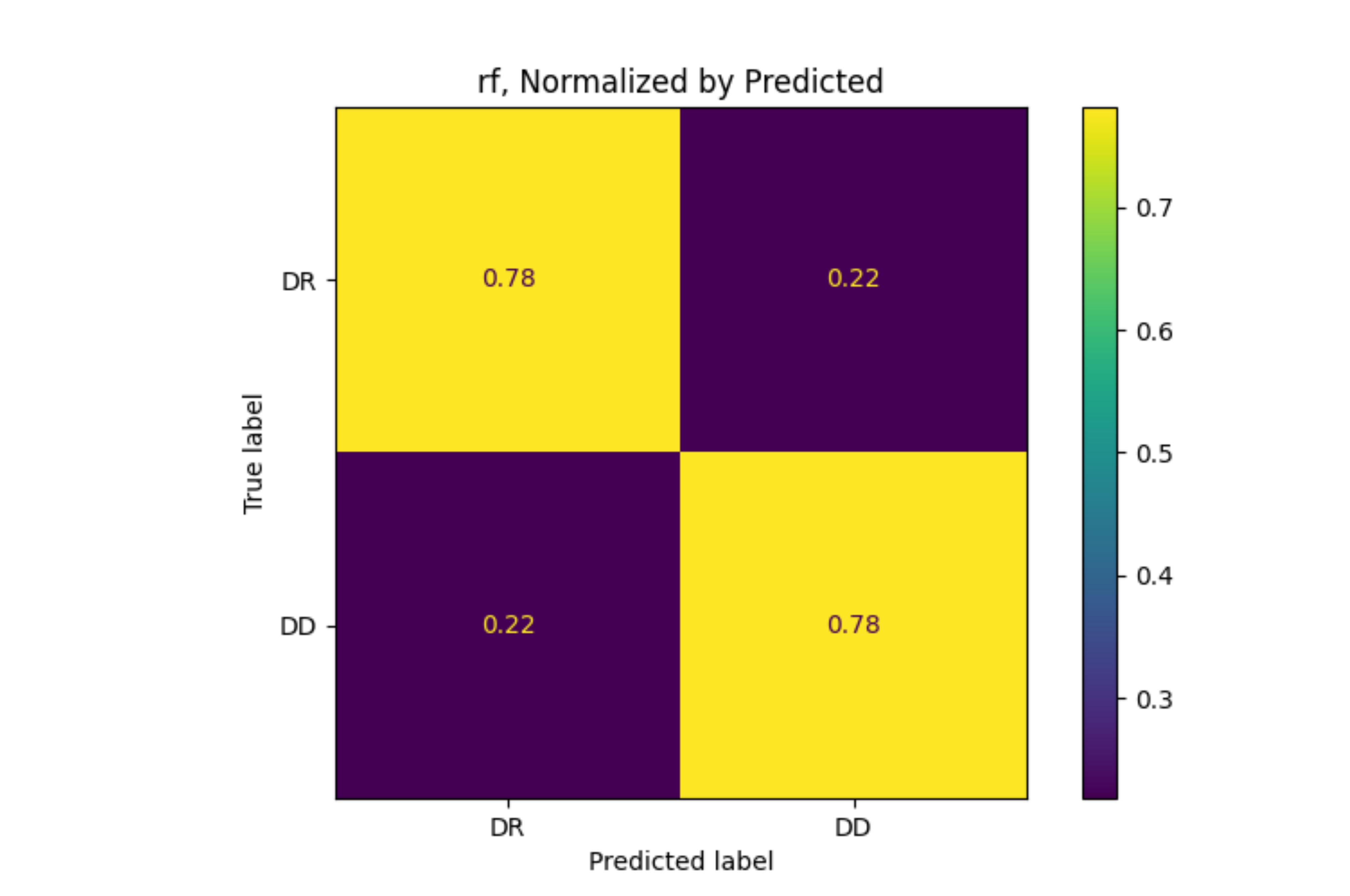
Base-D: ADABoost

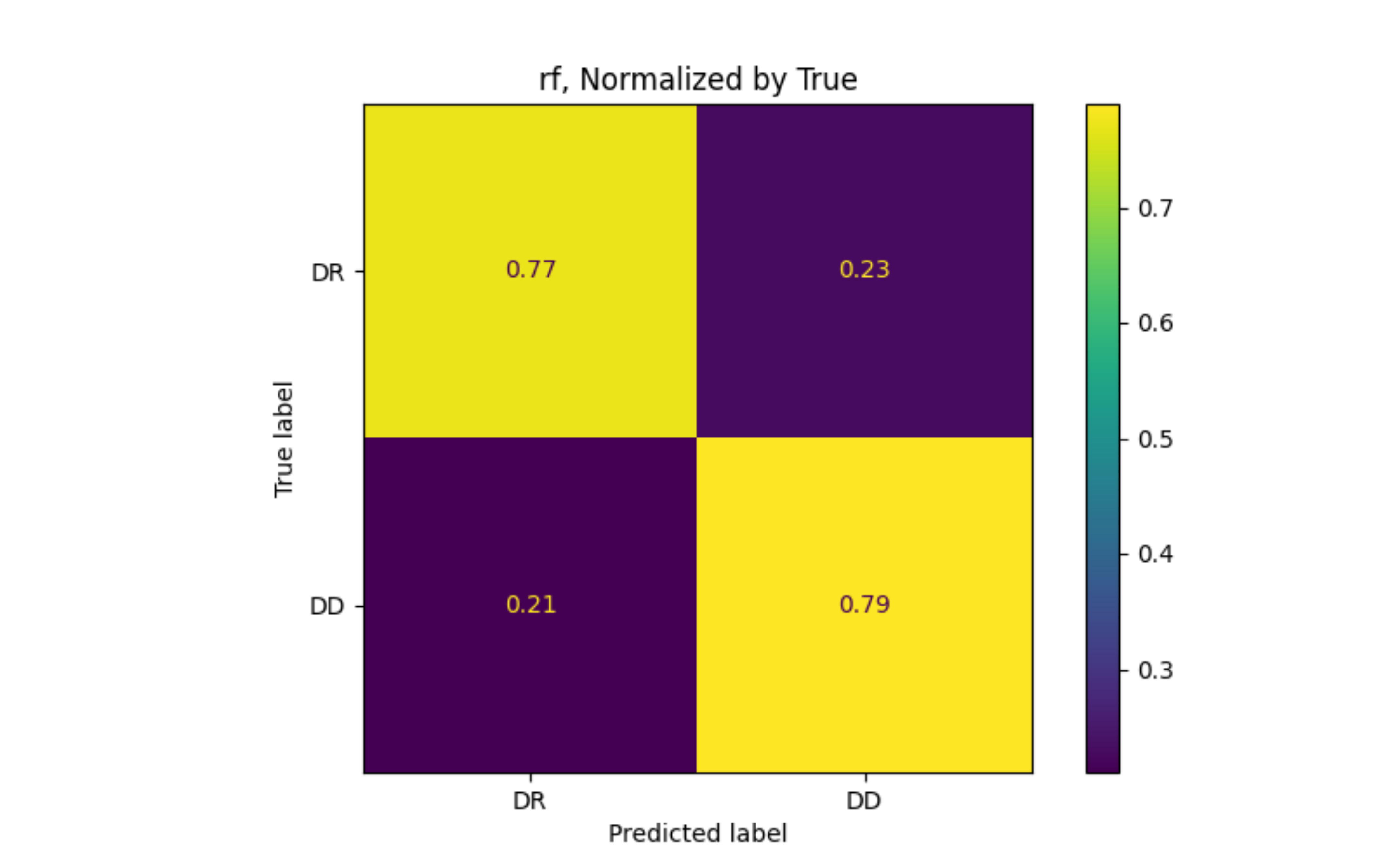




Confusion Matrices

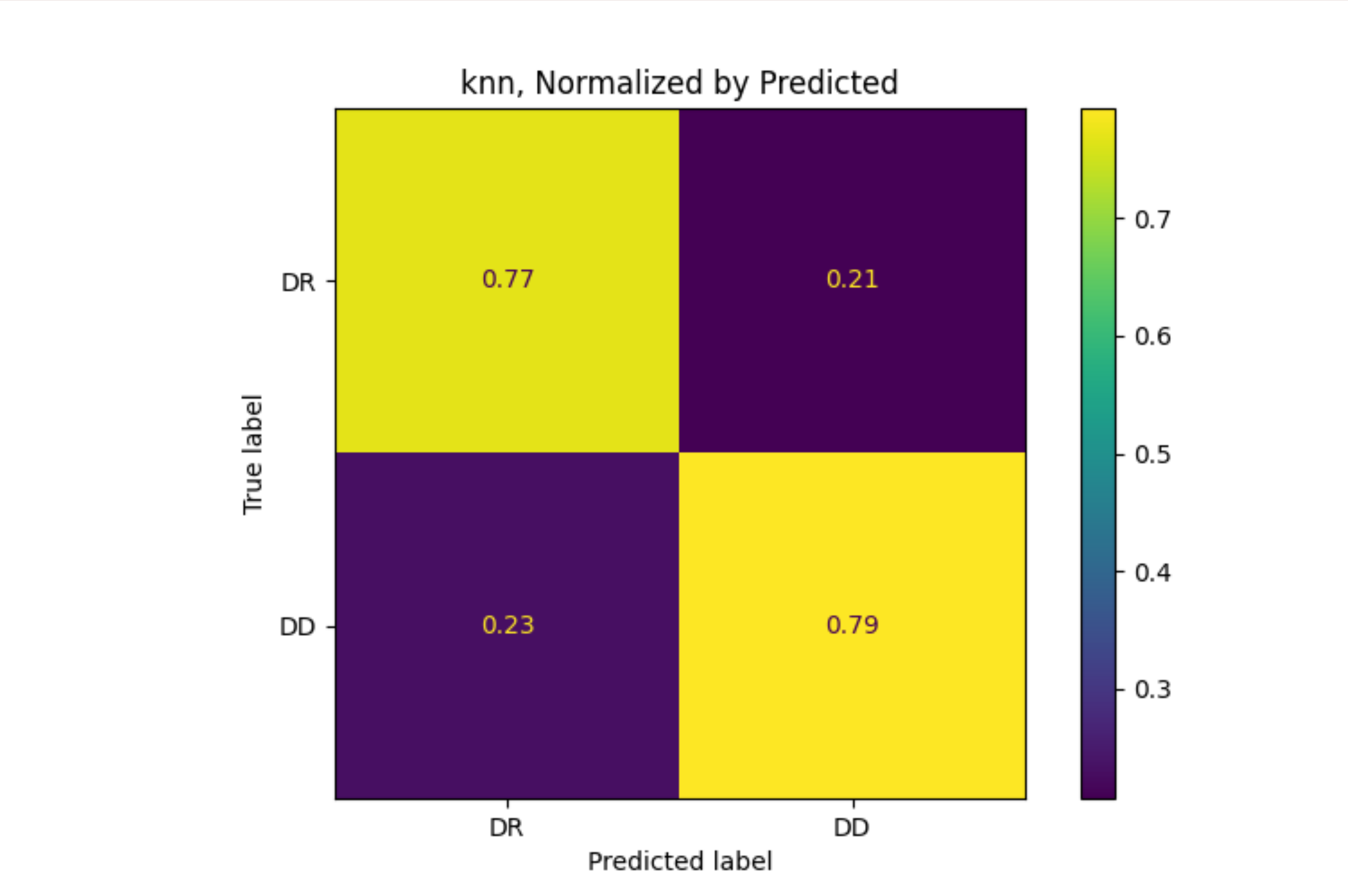
Base-D: Random Forest

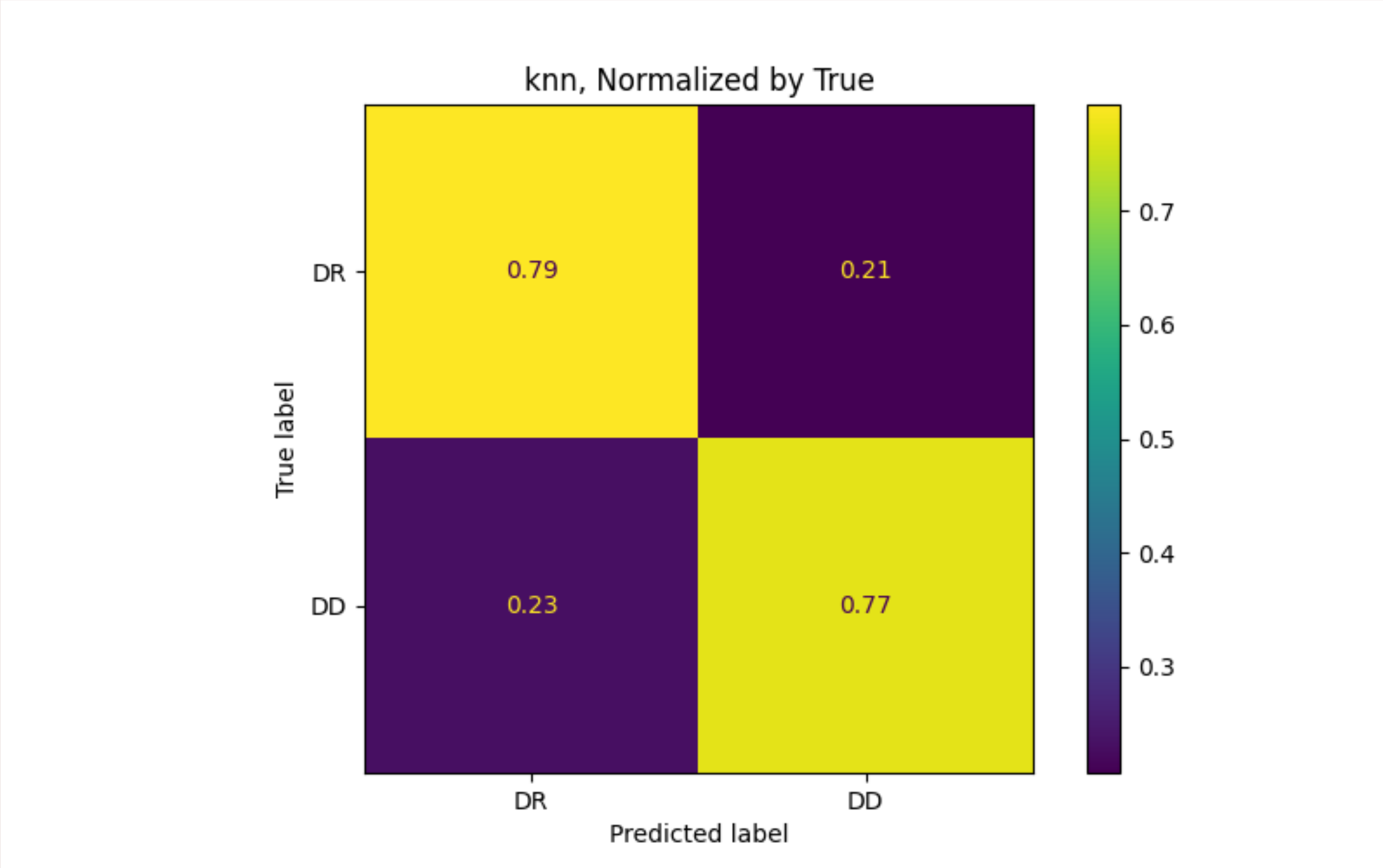




Confusion Matrices

Base-D: KNN





Miscellaneous Statistical Tests

	Join count tests	
	“BB”	“BW”
base-D	0.001	1.0
base-R	0.001	1.0

“BB” tests the null that the number of similarly labeled neighbors is not statistically different from random assignment (“BW” for differently labeled).

CV: equality of means tests

Base D: RF-KNN	4.112	0.0001
Base R: NN-RF	13.113	0.0000

	K-sample Anderson-Darling Tests for Similarity of Distributions		
Variable	Statistic	P-value	
% Male	14.9	0.001	
% White	336.0	0.001	
% Foreign	190.0	0.001	
% Poverty	38.7	0.001	
%Broadband	26.8	0.001	
% Medicaid	24.1	0.001	