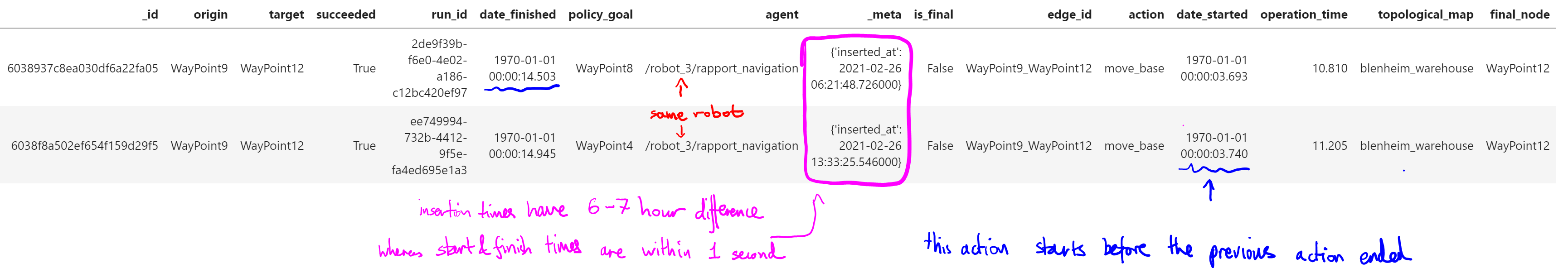
**Wk8 Writeup**

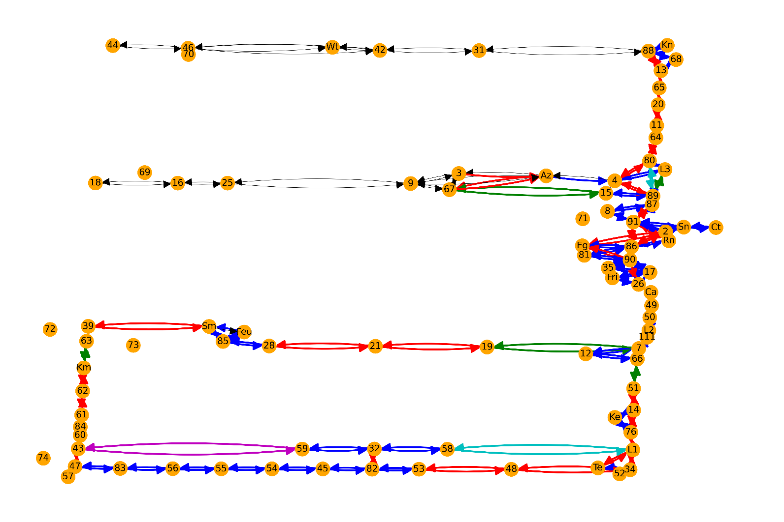
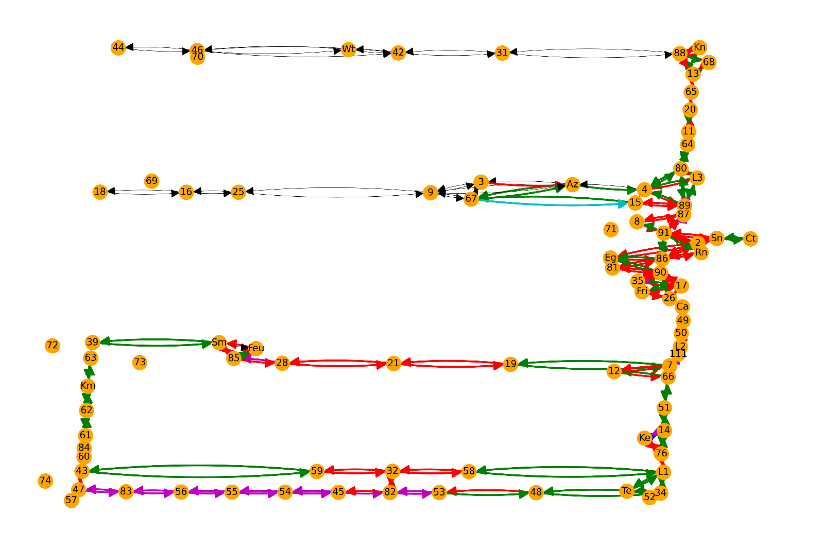
# RECAP: Week 7 Summary

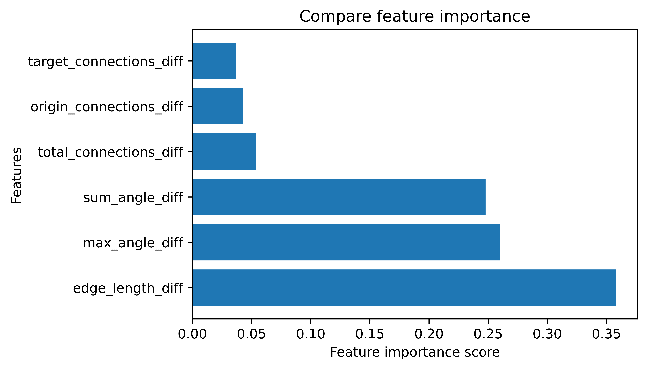
1. **[MON]** – Fix issues from week 6
   1. Used Blenheim\_scalar\_success.yaml data. **Still 1 cluster** according to KS for all levels of congestion
   2. Randomising mergefit makes little difference to performance (but allows for generality argument)
   3. Clarified CVM != integral of square of distances between CDFs. It is multiplied by pdf
   4. Question about which time used for congestion filtering (see below):
      1. Time resets on each run\_id – when checking for congestion, split data out by run\_id

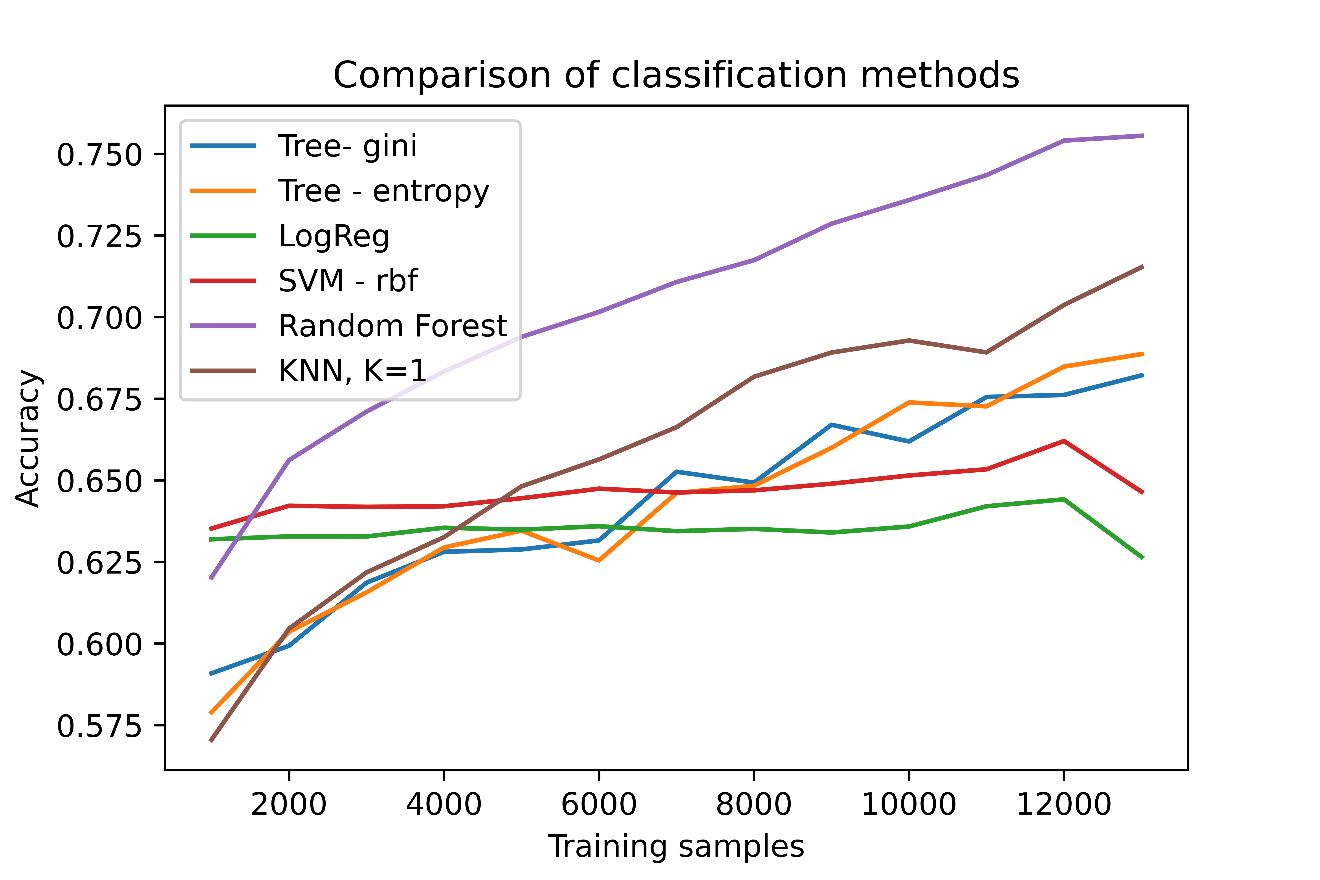


1. **[TUE]** – Fit distributions for STRANDS AAF dataset
   1. Worse performance across the board compared to Walmart
   2. Johnson SU is still the best. Lognormal has KS ~ 0.25 (top 20/88 distributions)
      1. Time-to-waypoint
2. **[WED]** – Cluster edges according to KS & spatial factors
   1. **5 clusters** according to KS score. Silhouette score has a max at 5 clusters.
      1. Check scale of python plot
      2. Try CVM/Square difference
      3. Look at pdf/cdf distributions for 61-62 & 43-59
      4. Reverse directions 48-53 vs 53-48
      5. Should there be more clusters?
   2. KS clustering cannot be wholly explained by edge length

Clusters for KS (left) do not exactly correspond to clusters for edge length difference (right).



1. **[THU, FRI, MON]** – Binary classification based on spatial factors
   1. **Input** = difference in spatial factors
      1. Edge length
      2. Max angle
      3. Sum of angles
      4. Total connections
      5. Origin connections
      6. Target connections
   2. **Output** = Do 2 edges belong to the same cluster?
   3. Tried:
      1. **Logistic regression** (my implementation & library) – 64% accuracy
      2. **Neural network** logistic regression (my implementation & library) – more than 1 hidden layer decreases performance. Performance is similar to standard logistic regression.
      3. **KNN** (library) – k=1 has best performance. Feature normalisation & reducing no. of features decreases performance
      4. **SVM** (library) – Radial basis function has best performance
      5. **Decision Trees** (library) – gini & entropy criterion have similar performance
      6. **Random Forest** (library) – best performance, performance increases with training samples



# PLANS: Week 8

1. **[TUE]** –
   1. Fix congestion filtering
   2. Analyse unexpected observations in KS clustering for AAF
      1. Check scale of python plot
      2. Try CVM/Square difference
      3. Look at pdf/cdf distributions for 61-62 & 43-59
      4. Reverse directions 48-53 vs 53-48
      5. Should there be more clusters?
   3. Any more classification methods & spatial factors.
      1. Context? N\_robots
      2. What new environment would be interesting to try in order to get congestion data
   4. Try using different number of KS clusters before binary classification.
2. **[WED]** – Add in data from previous years.
   1. Check distributions on same edge from different years
   2. Ask Nick
3. **[THU]** – Use RF/KNN classifier on another map: using fit from AAF. Also try fitting classifiers to new map
4. **[FRI]** – Use binary classifier to generate clusters. Visualise.
5. **[EXTRA]** – Try different GOF scores for ranking Scipy distributions. Implement Johnson-SU model.

STRANDS all use same robot (Betty). Walmart/Blenheim use same robot (Jackal)

**Remaining questions:**

* Any other Classification methods to try?
* Any other important spatial factors? (evaluate feature importance using RF)
* What happens if you change the number of KS clusters before Binary Classification?
  + Try plotting performance vs no. of KS clusters in AAF map
* Can you use the AAF data from previous years so we get more data for classifiers?
* Can you use this (RF) classifier for another map? E.g. TSC, Walmart, Blenheim
  + Validate according to KS clustering on other maps
  + Do any other maps use the same robots?
* Use the binary classifier to generate clusters
* Try ranking Scipy (MLE-fitted) distributions according to square\_distance score
* Try fitting Johnson SU distribution using a Bayesian method (no prior 🡺 MCMC)

# MON: Congestion, analyse KS clustering, classification for different KS clusters

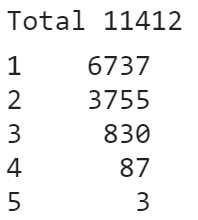
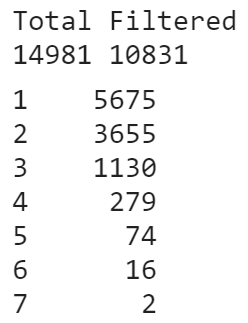
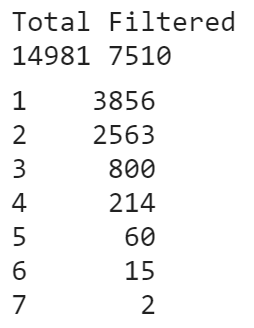
#### Congestion filtering on Blenheim map

**Filter by:**

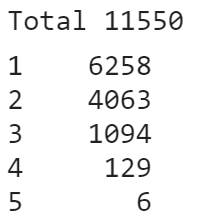
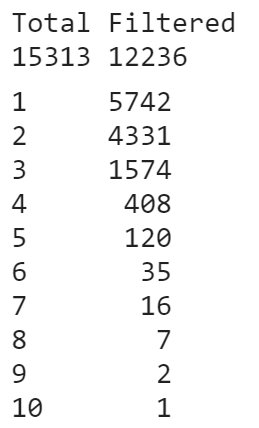
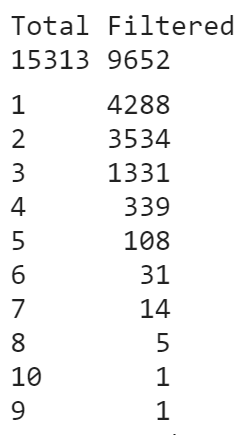
* Run\_id
* Edge id – all edges leading to target node & target\_origin edge
* Present at same time
* Then count all robots (including the robot taking data)

**Compare to blenheim\_scalar\_success.yaml:**

Random (left – YAML data, middle – My implementation, right – my implementation, removing is\_initial):

Targeted (left – YAML data, middle – My implementation, right – my implementation, removing is\_initial):

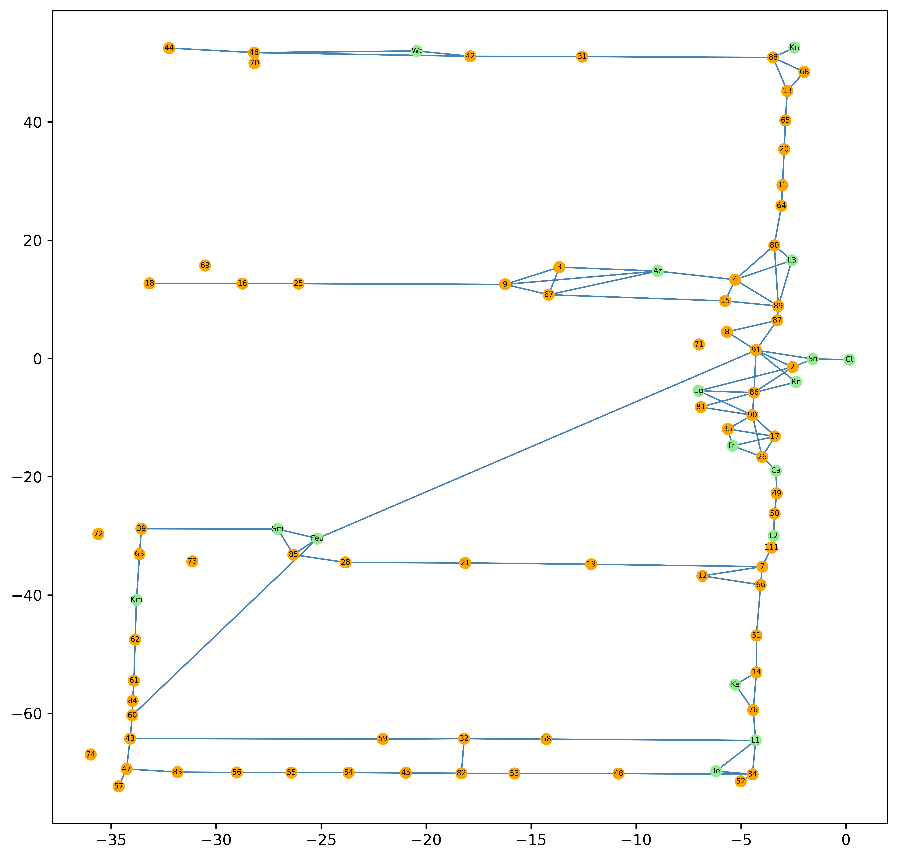
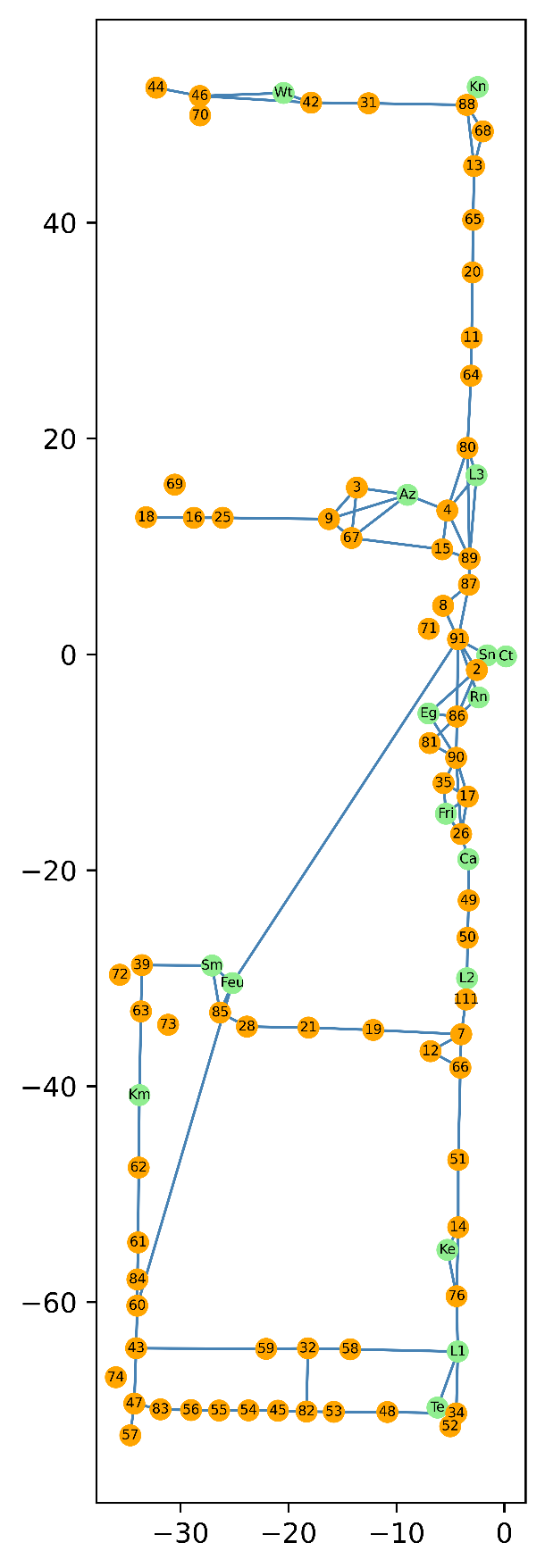
#### Analyse unexpected observations in KS clusters for AAF map

**To check:**

1. Check scale of network visualisation plot (is it artificially square)? YES
2. Edges of diff length in same cluster: Look at pdf/cdf distributions for 61-62 & 43-59
3. Opposite directions of same edge in different clusters: Reverse directions 48-53 vs 53-48
4. Try CVM/Square difference to determine clusters. KS works better because it’s normalised.
5. What does the network map look like with more clusters?

**Scale of network Visualisation Plot**

Left – ticks have equal scales. Right – axes have equal lengths.

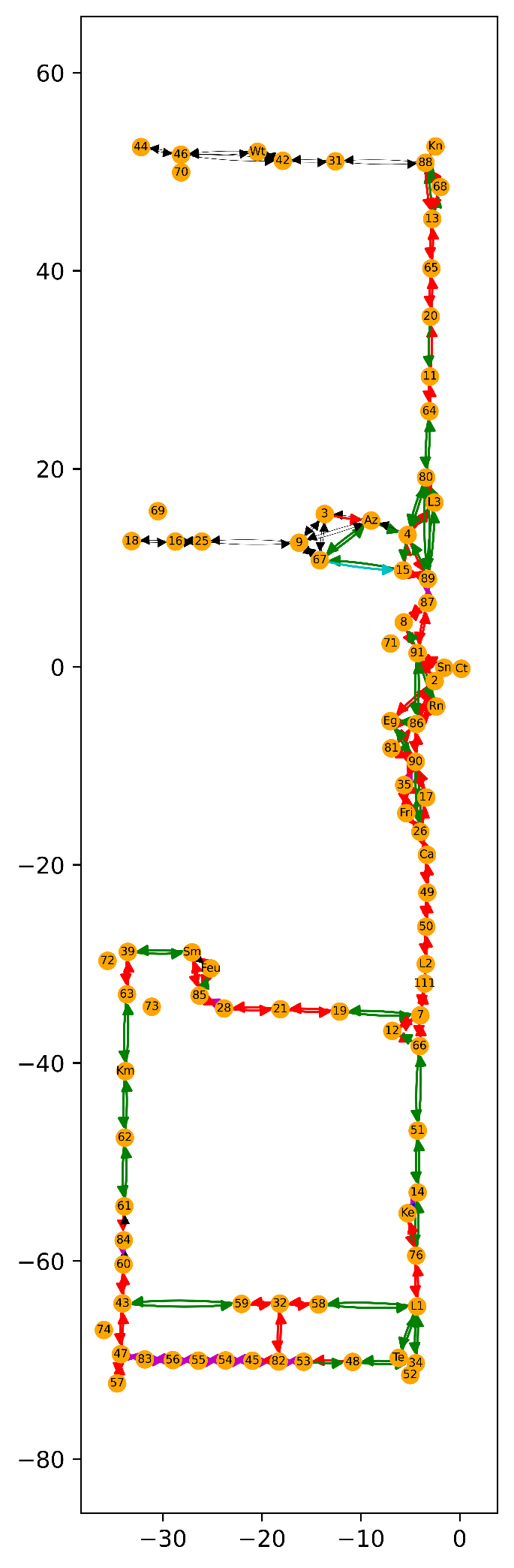
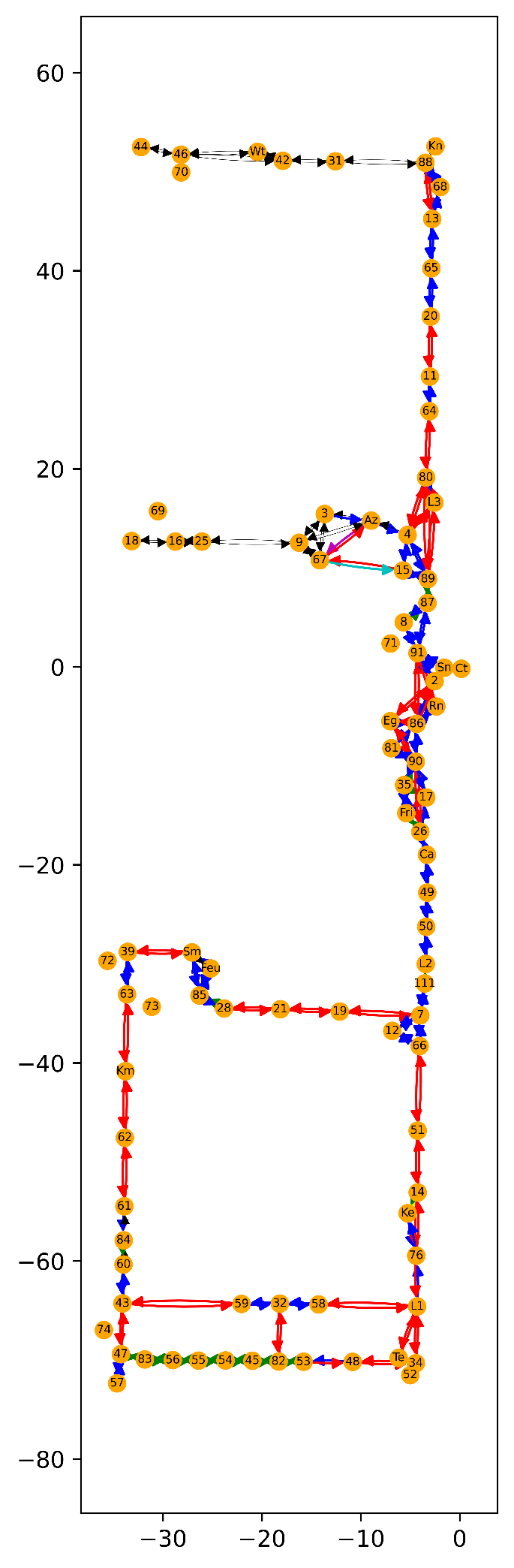


**KS cluster**

Left (by operation\_time 🡺 5 clusters). Right (by operation\_time – time\_to\_waypoint 🡺 6 clusters)

**Comparing 43\_59 vs 43\_47 suggests clustering by operation\_time is better**

**Most clustering discrepancies are due to lack of datapoints (approx. 50). KS score is very high (but presumably better/same compared to other edges)**

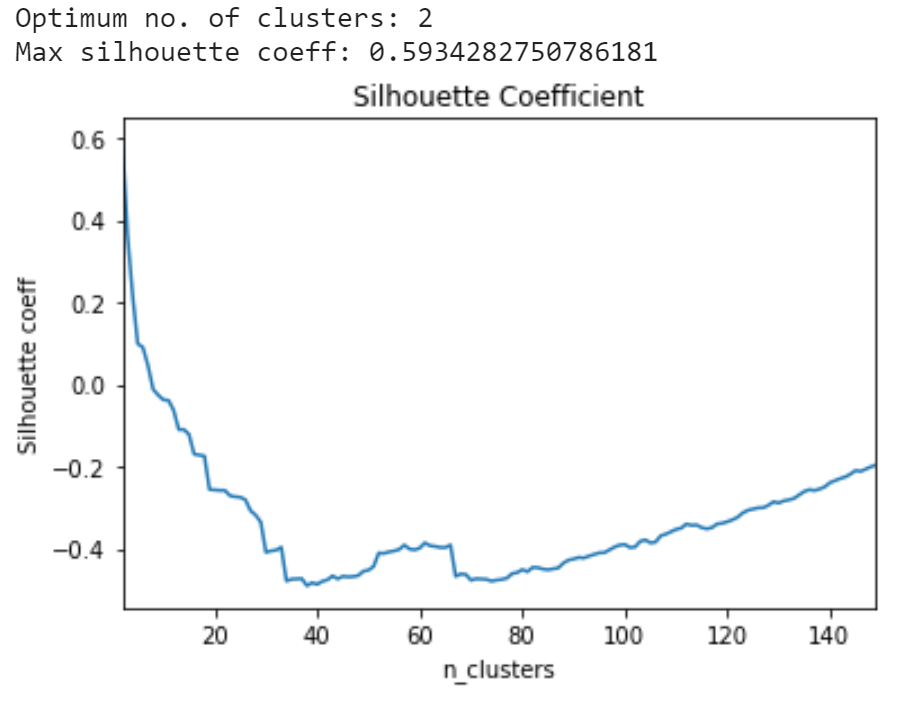
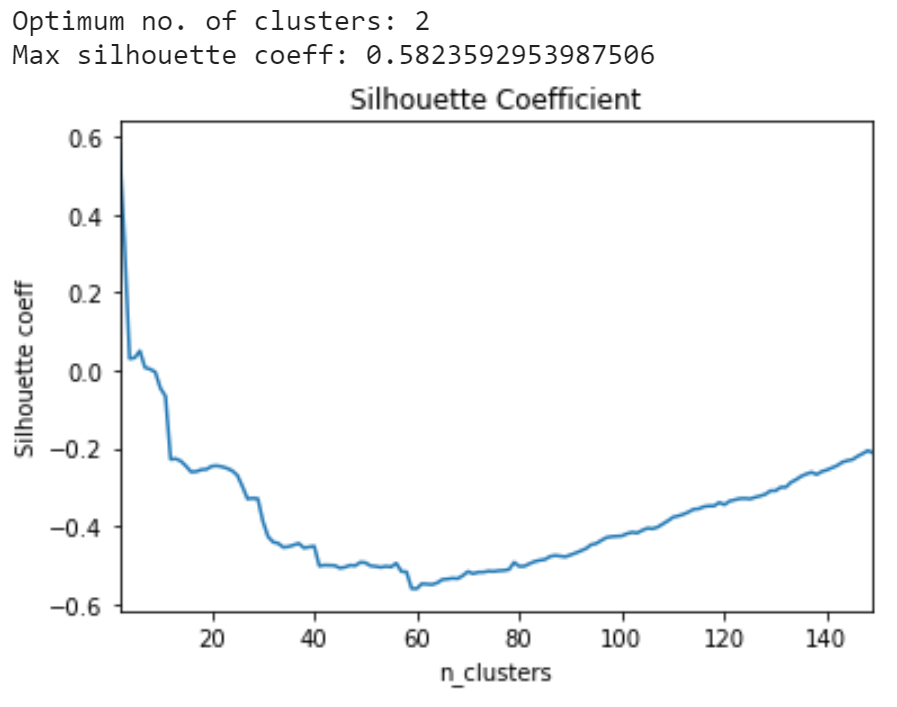
 

**Using other metrics for clustering (CVM & Square distance) – see Wk7STRANDS/STRANDS2.ipynb**

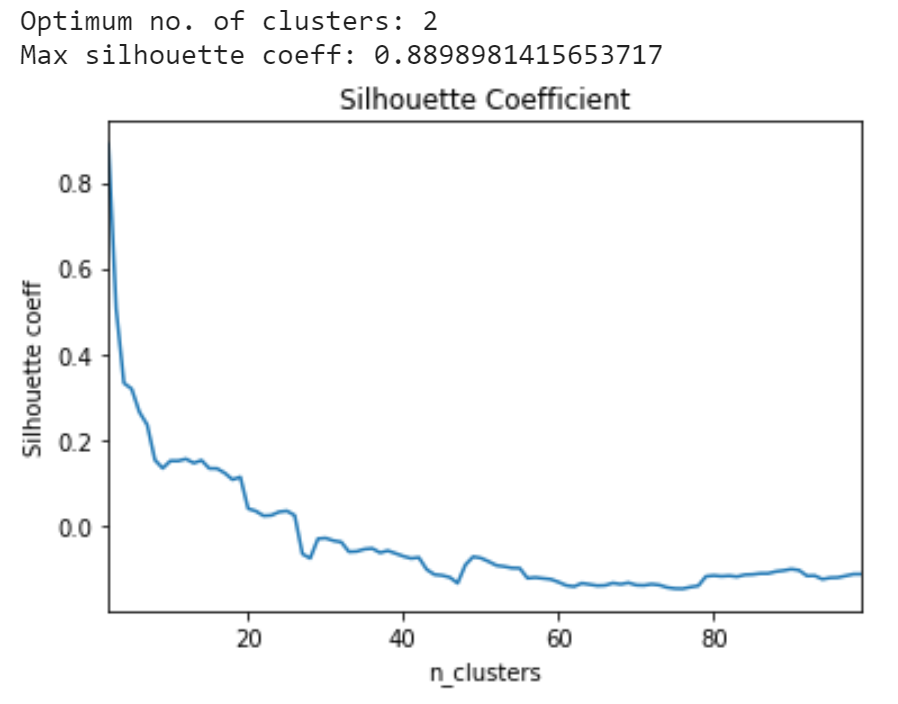
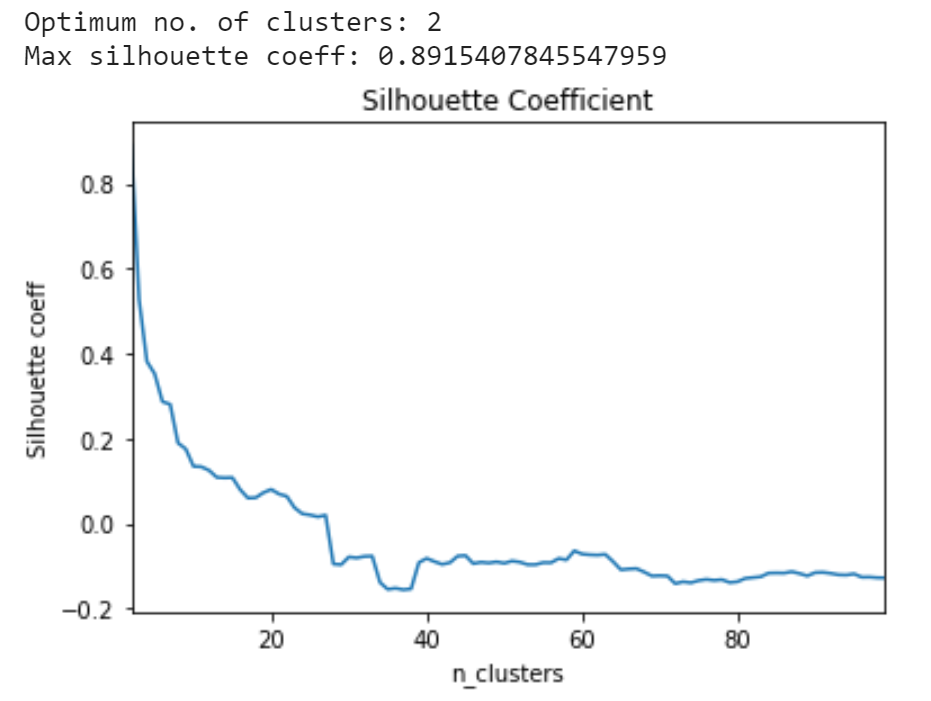
CVM & square distance have either 1 cluster / 110 clusters, so are much less useful than the KS score.

This may be because CVM is sample size dependent and CVM/square difference are not normalised within a range. This makes it more difficult to compare CVM & square difference compared to KS scores, and use this for clustering.

CVM: Left (operation\_time). Right (operation\_time – time\_to\_waypoint)

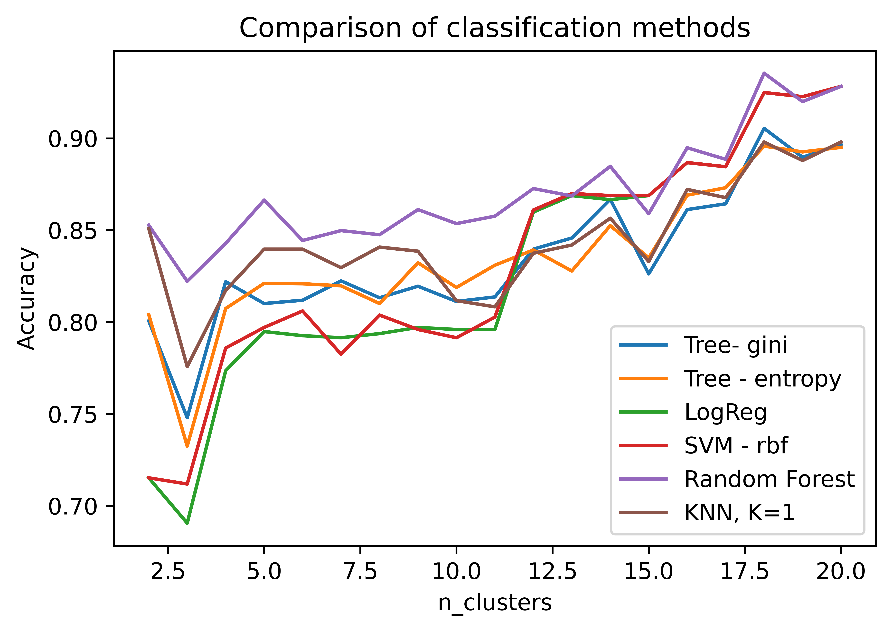
Integral of Squared distance: Left (operation\_time). Right (operation\_time – time\_to\_waypoint)

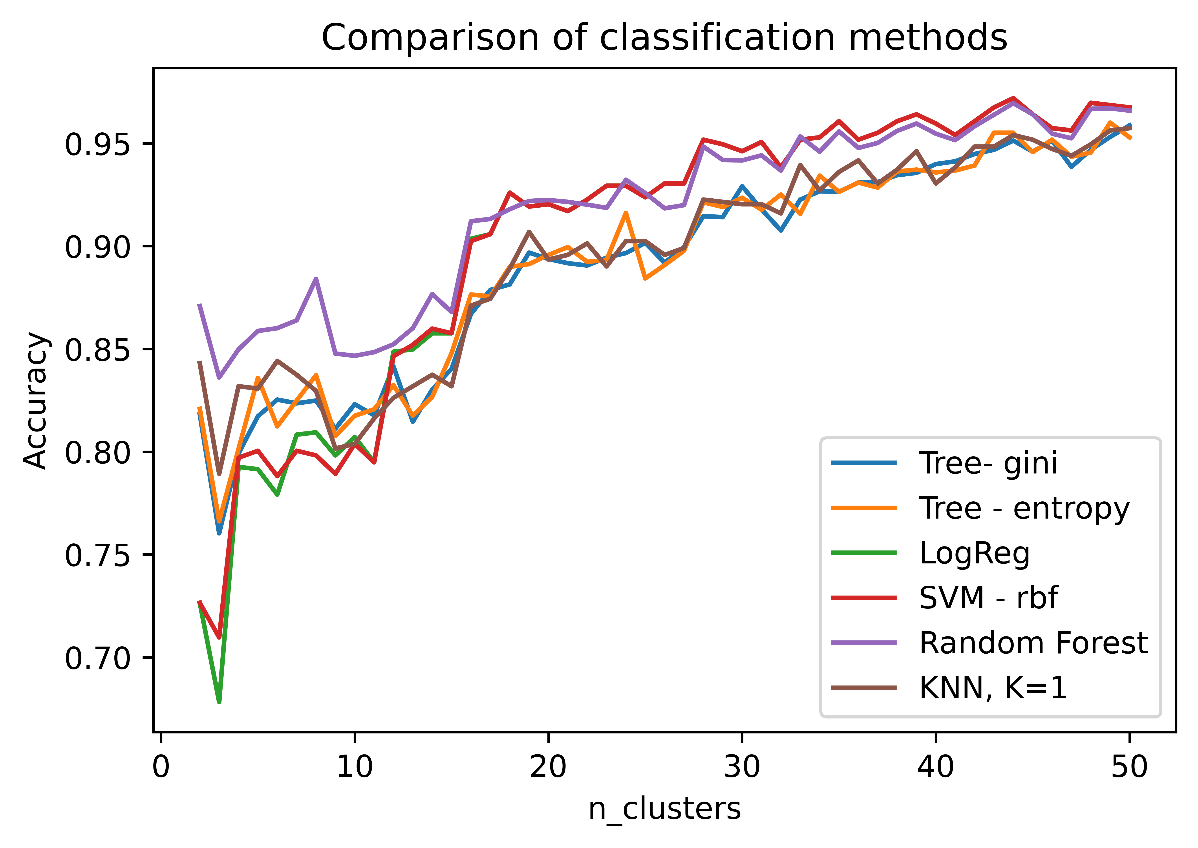
 

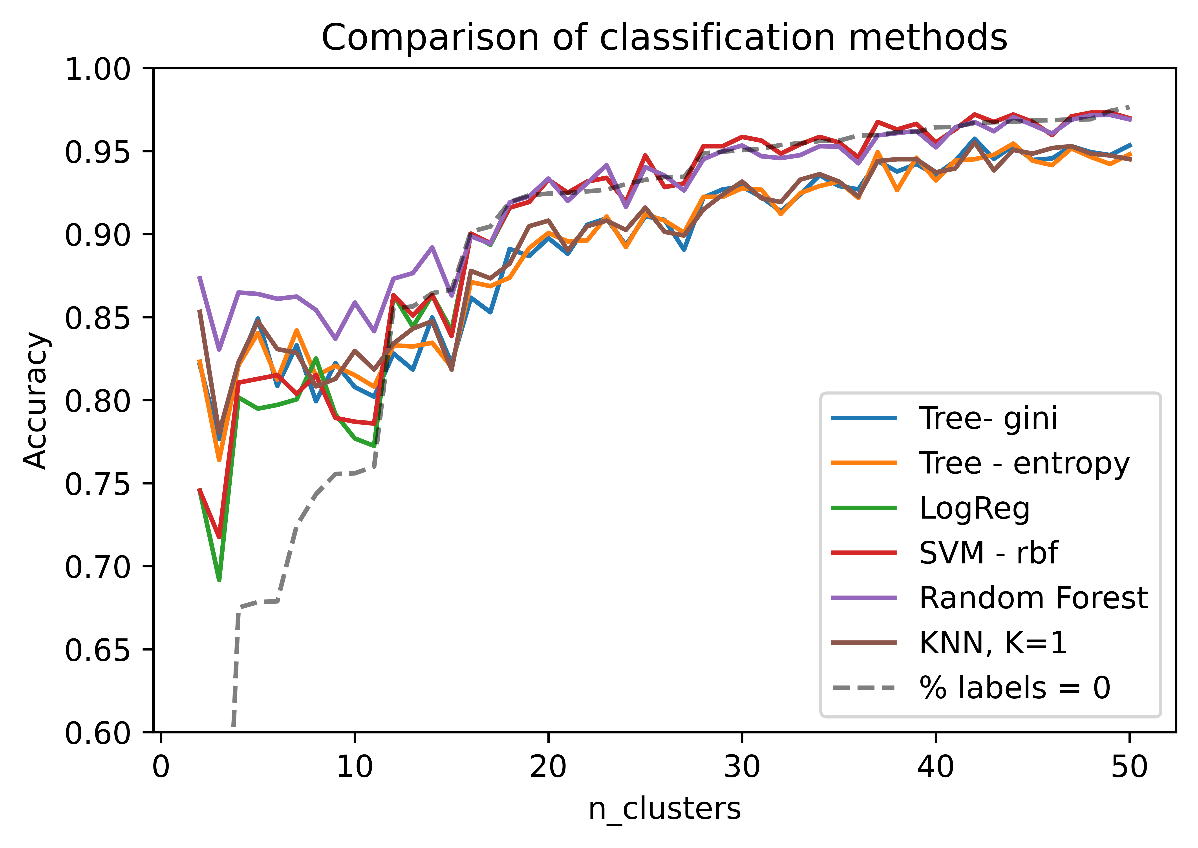
# WED: More cluster & More Years (AAF)

#### Binary classification when different no. of KS clusters used

Use 8911 datapoints for C = 1,2,3,4… 20 clusters

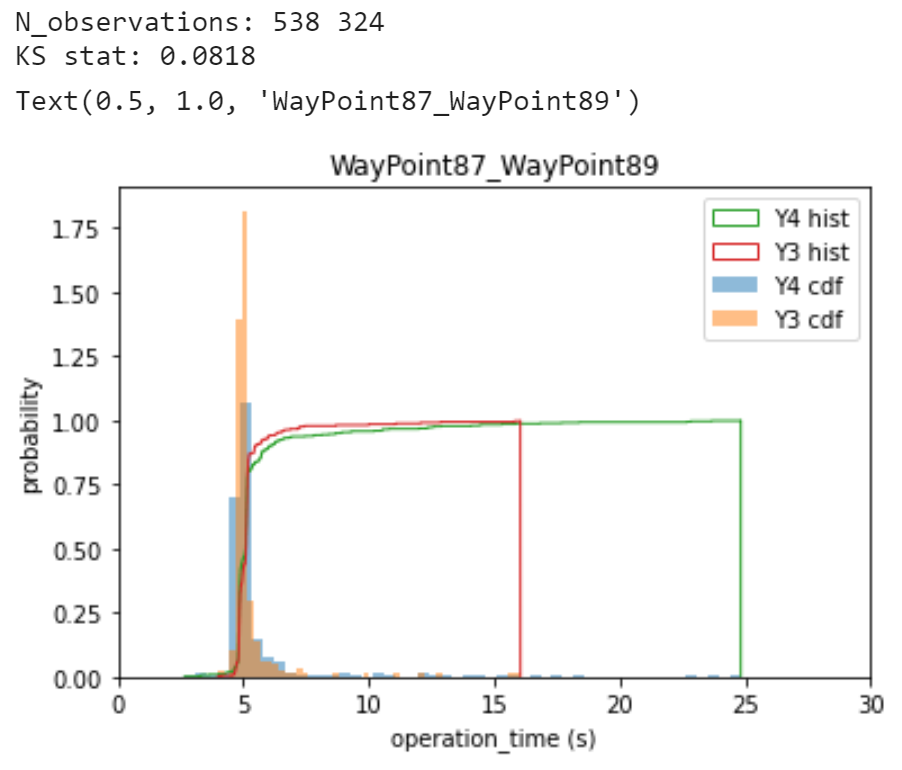
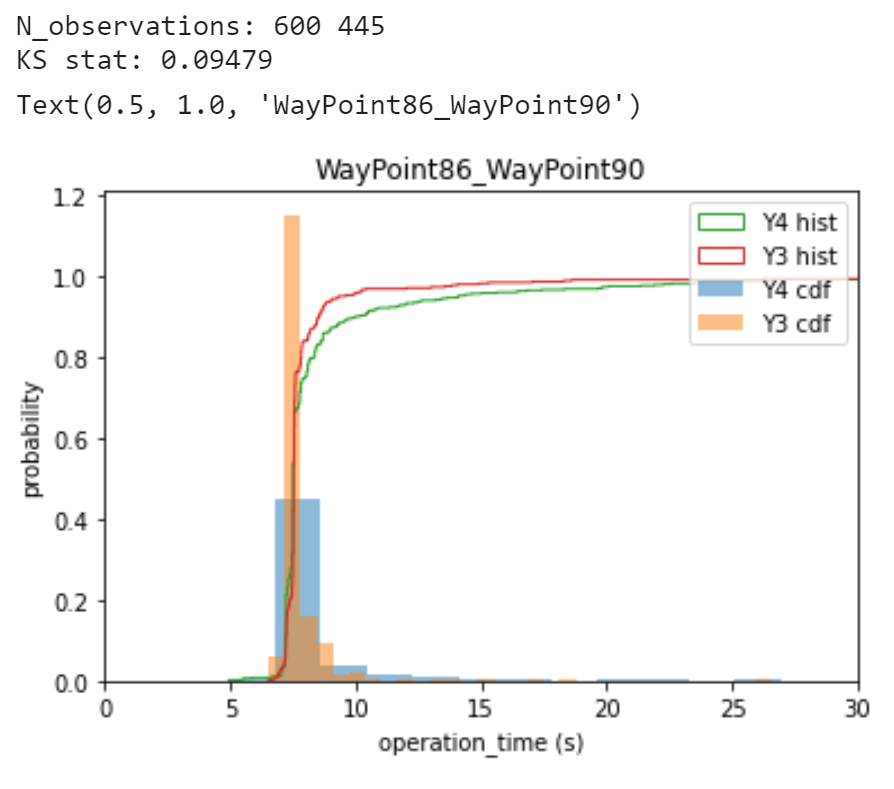






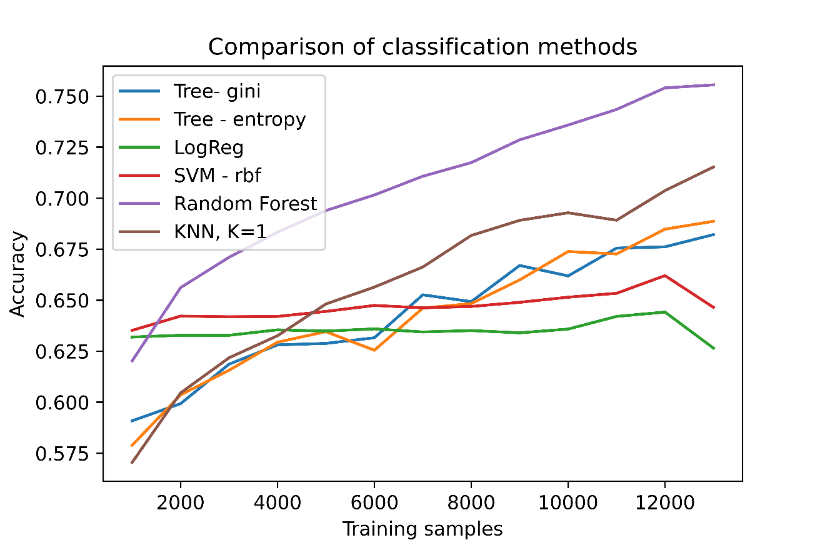
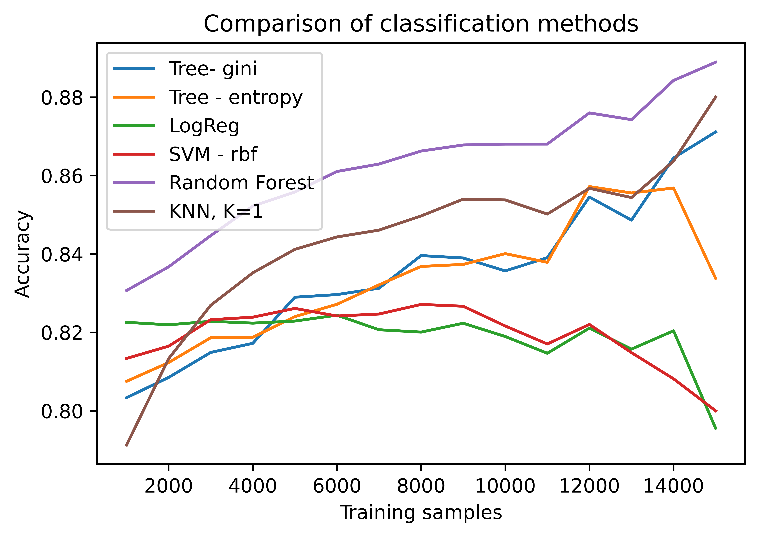
#### Can we merge Y2, Y3, Y4 data?

For Y4 vs Y3 data: compare pdf/cdf/ks of the same edge



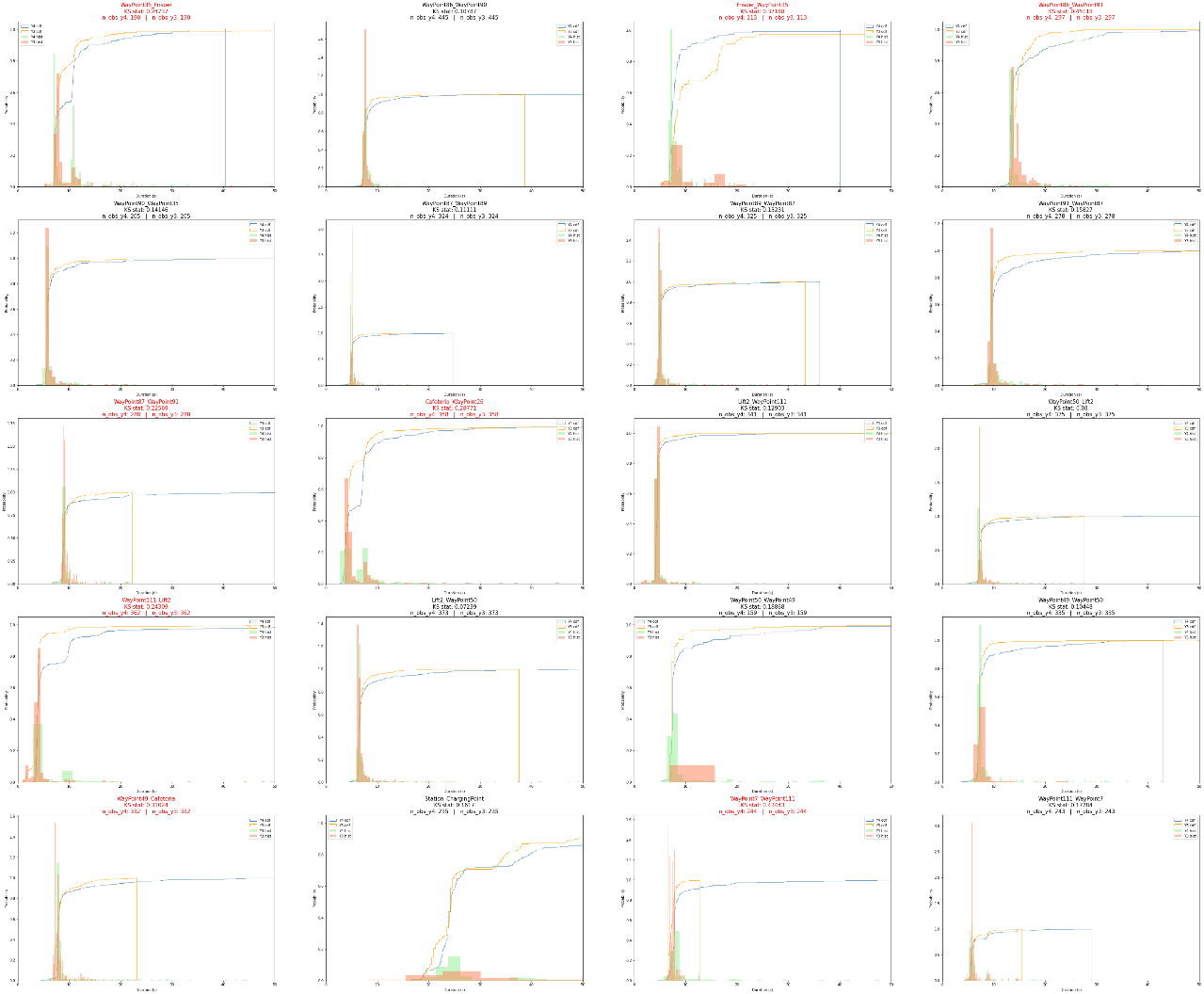
#### Use merged data for classification

Better performance (merged data – left, original – right)

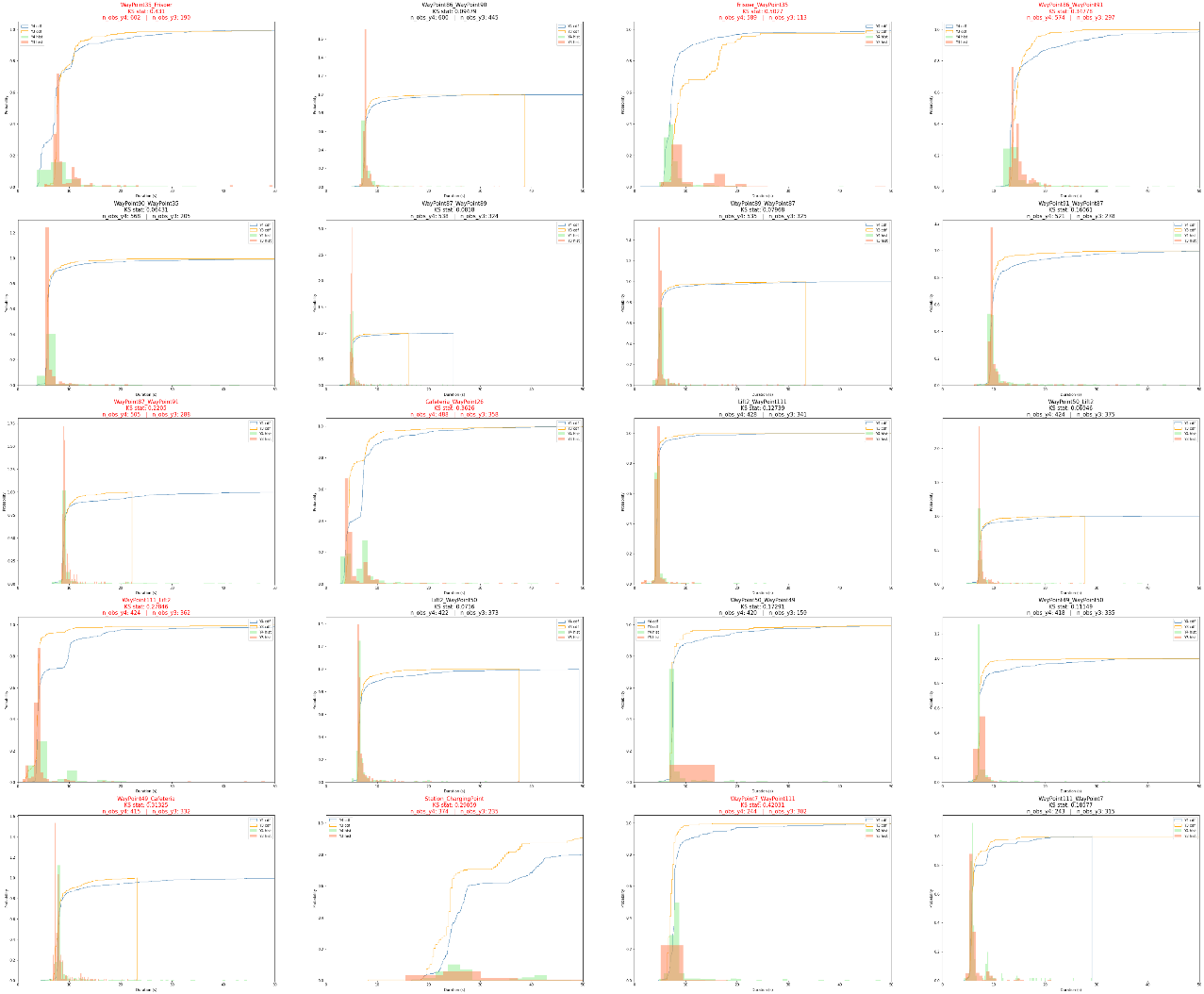


# THU: Fix issues with clustering

#### Same number of datapoints when comparing edges in Y3 vs Y4 of AAF (balanced – top, unbalanced – bottom)



With the full number of datapoints from each year:



Not a huge difference. Visually inspecting the pdfs/cdfs 🡺 can see that observations occur in similar positions.

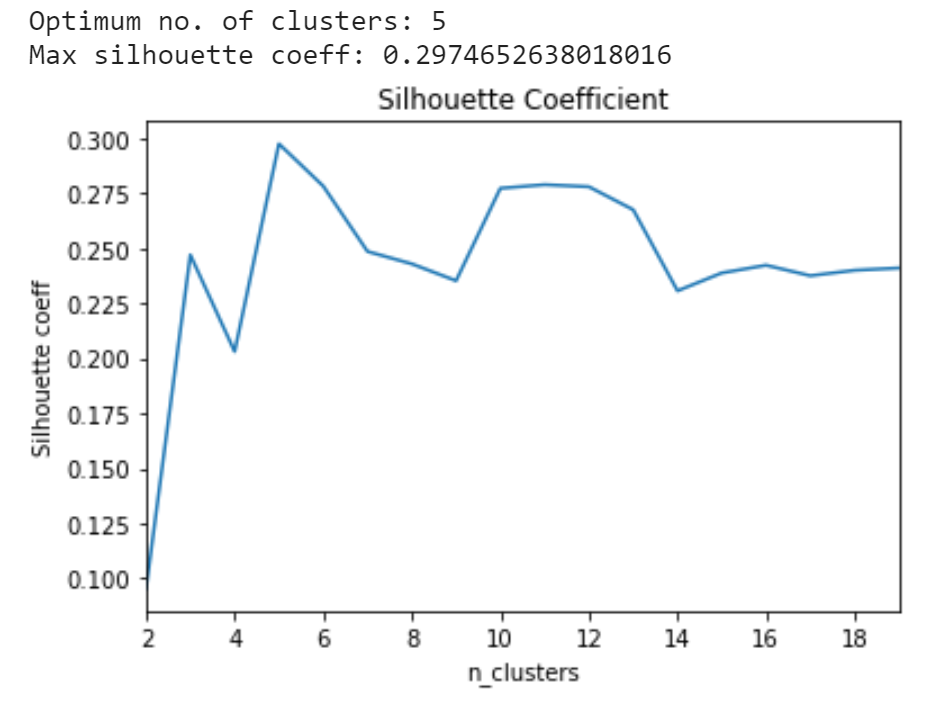
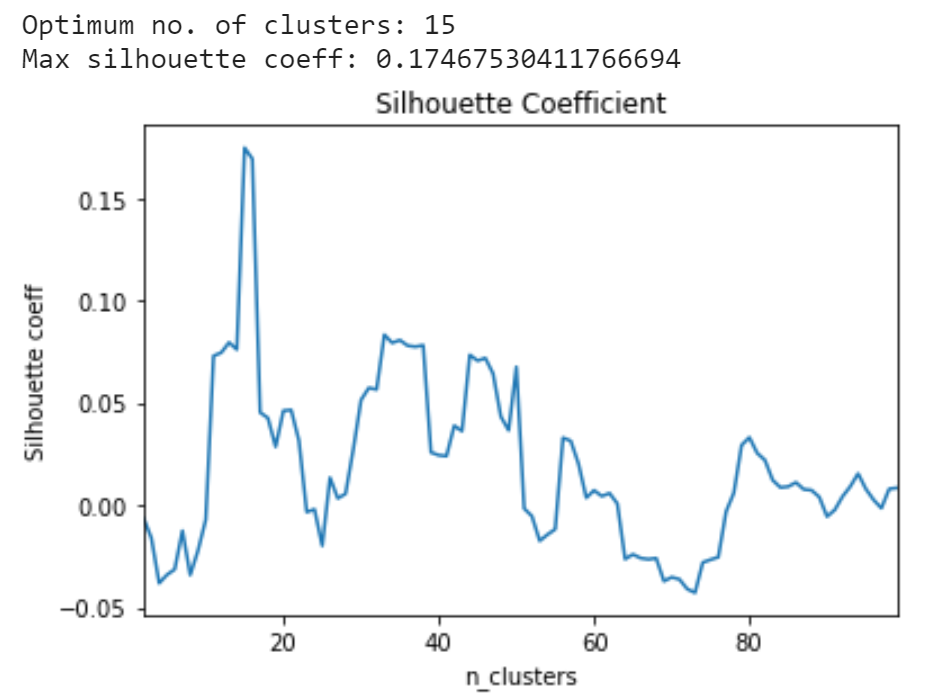
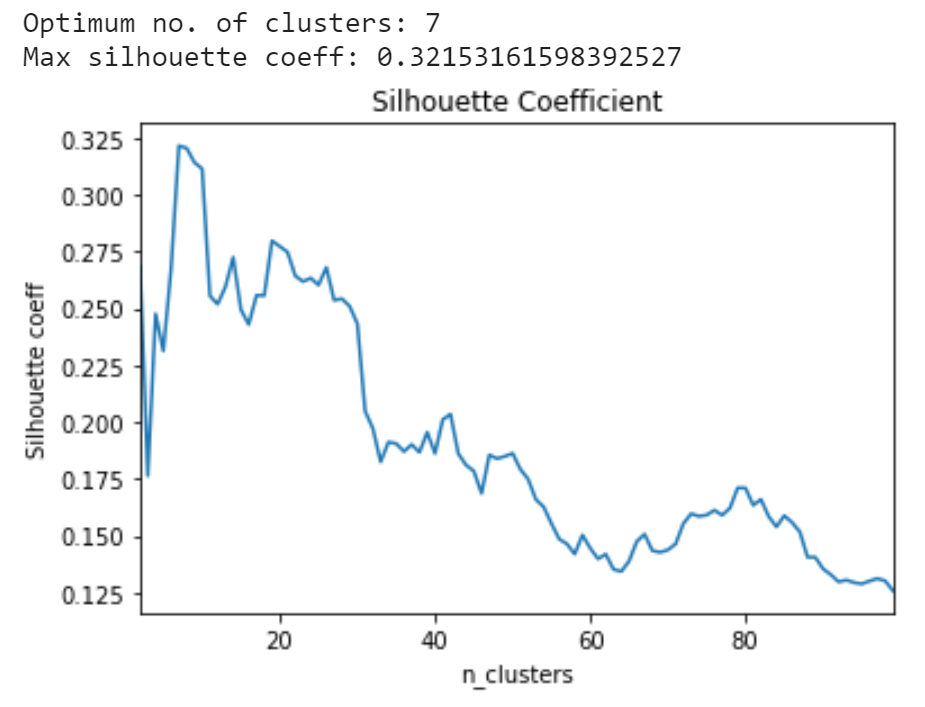
#### Clustering linkage metric (AAF) – average, single or complete

Possible linkage metrics:

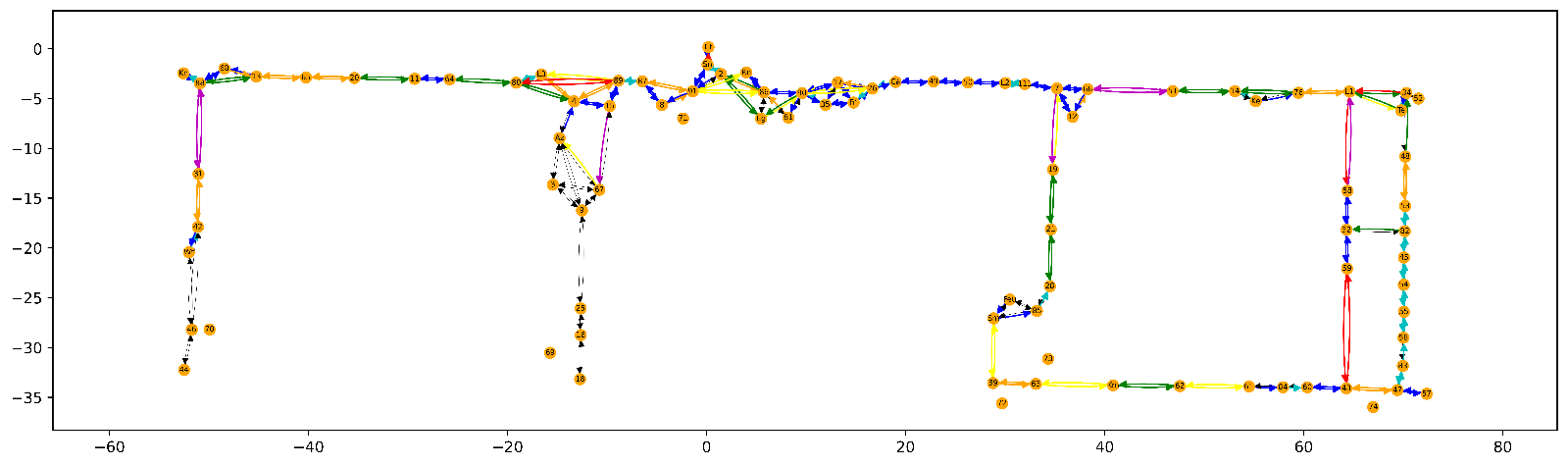
* "average” - average distance between each point in one cluster to every point in the other cluster.
* "single" - shortest distance between a point from each cluster.
* "complete" - longest distance between a point from each cluster
* "ward" - sum of square distances within all clusters.

Ward wasn't available for clustering with a precomputed distance matrix and I chose "average" since this gave more reasonable clusters in the Walmart case.

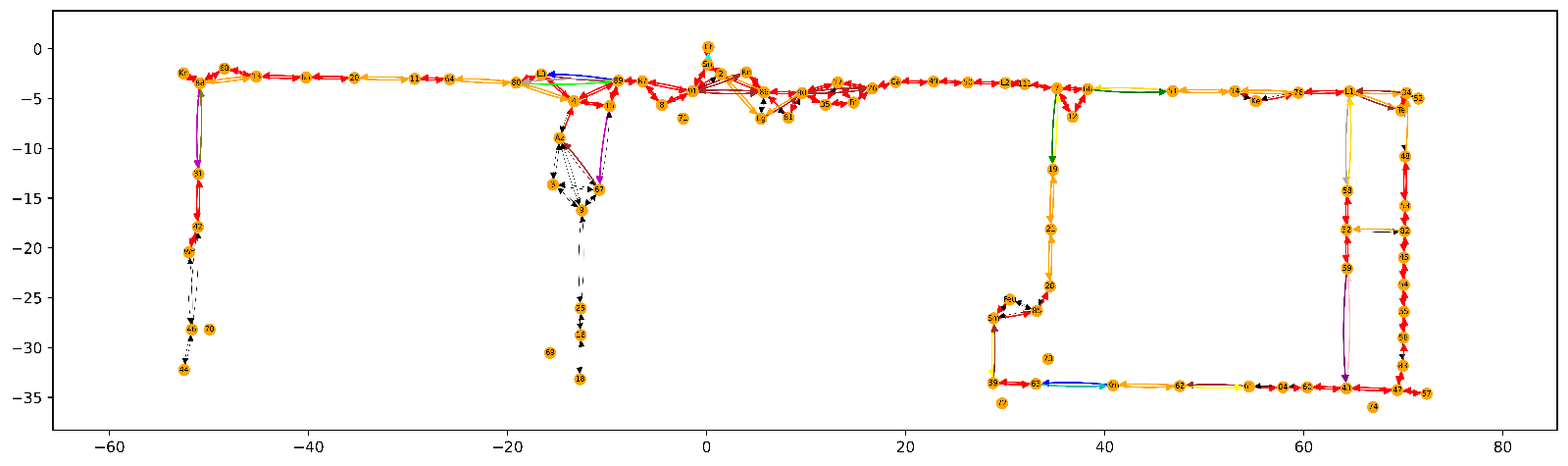
**N\_clusters (average – left, single – middle, complete – right):**



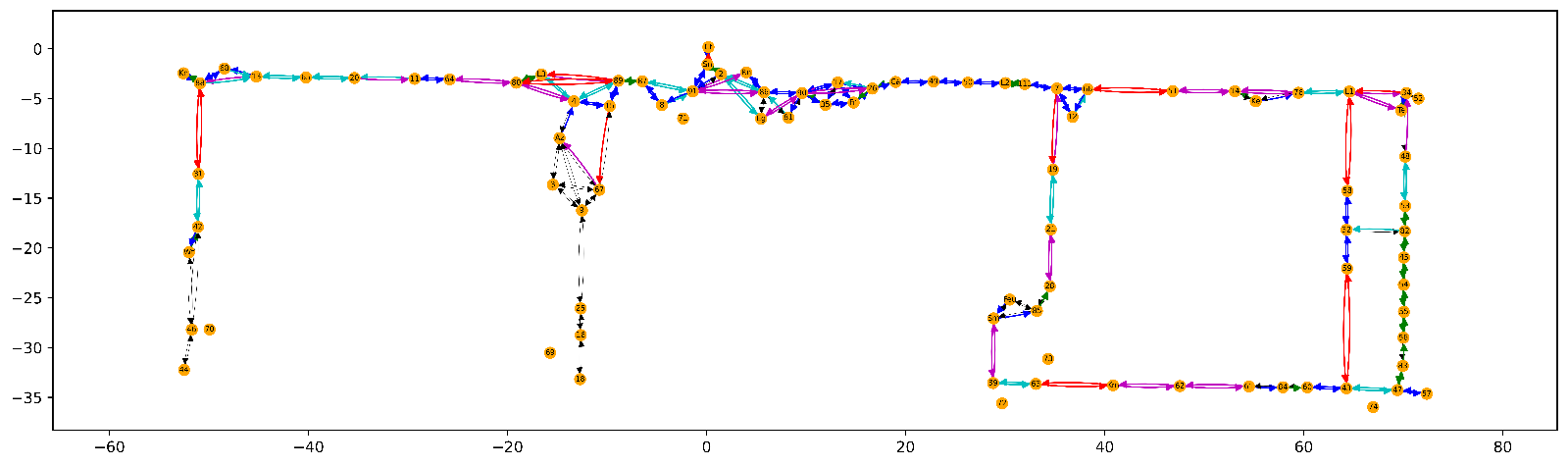
**Average (7 clusters)**



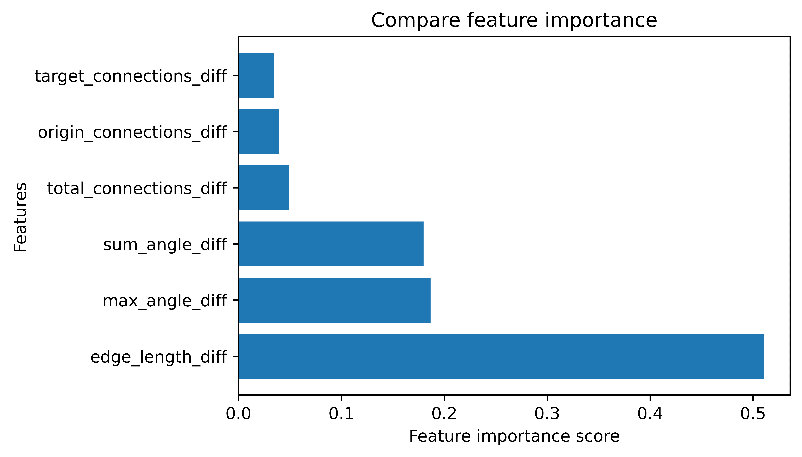
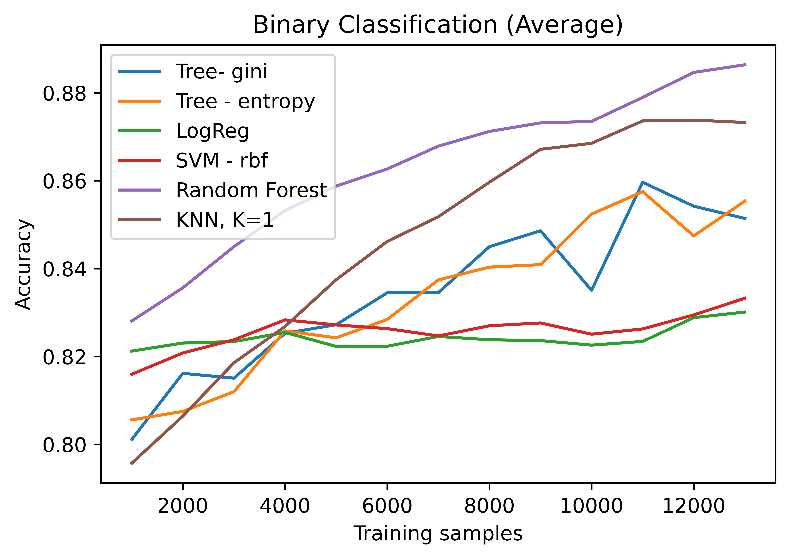
**Single (15 clusters):**

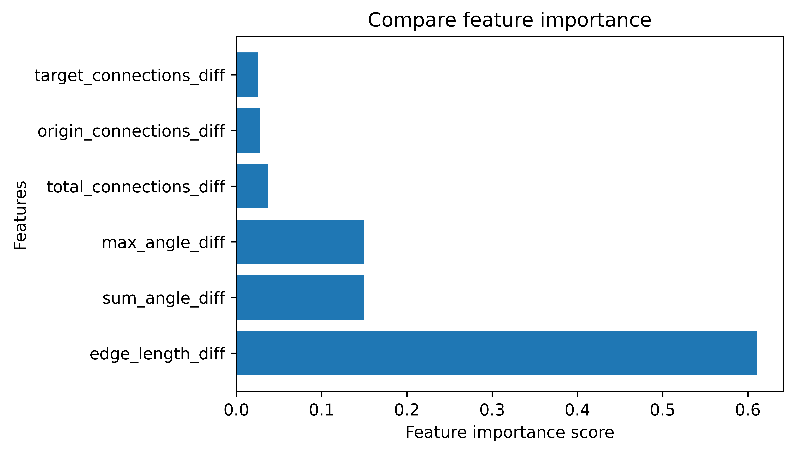
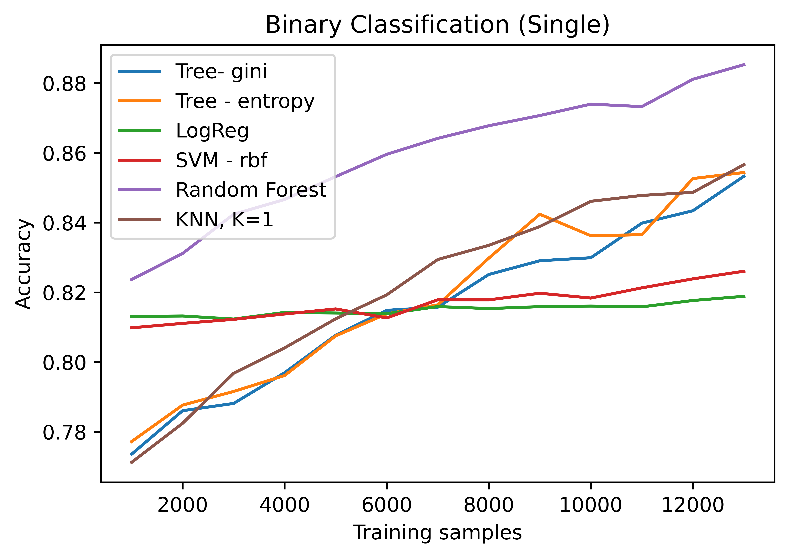


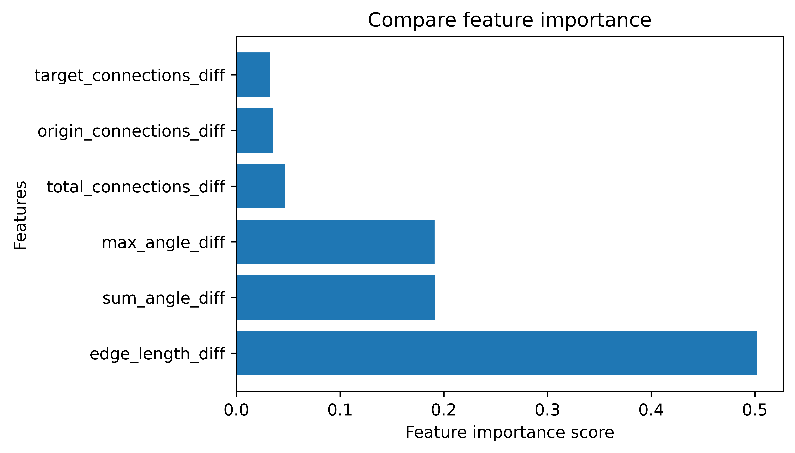
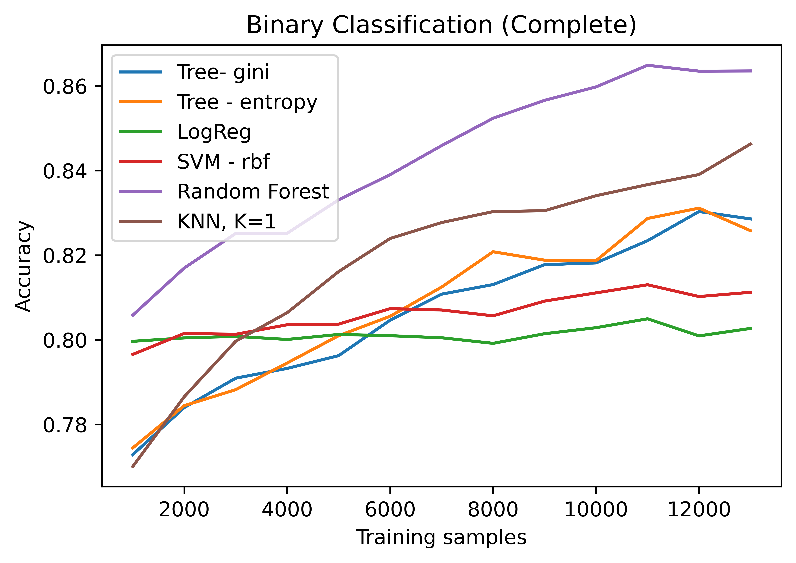
**Complete (5 clusters):**



**Compare classification performance:**All have good performance. However, dependence on edge length is high for “single”

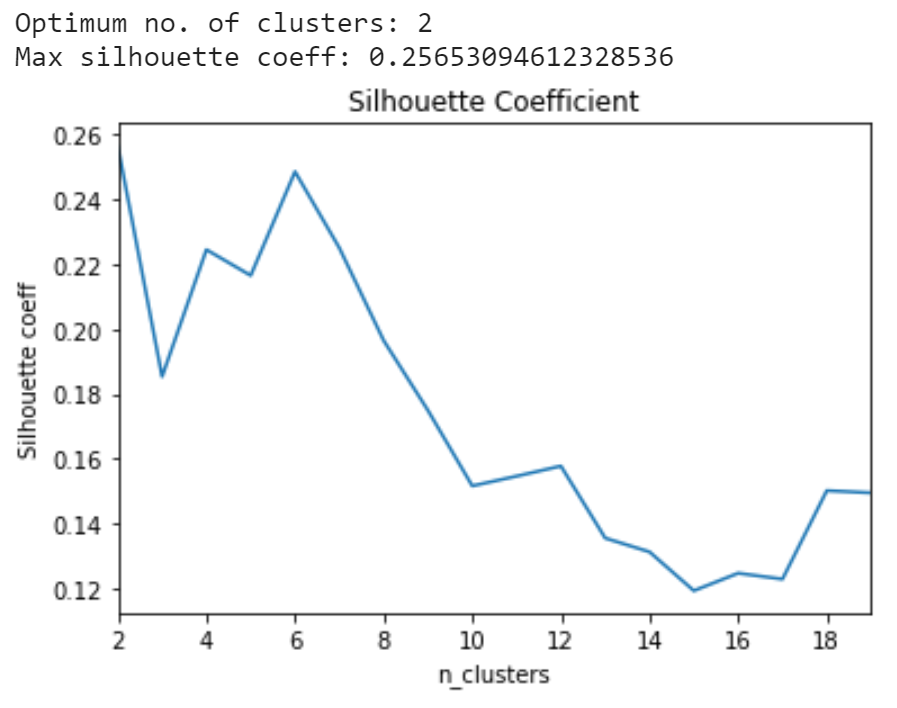
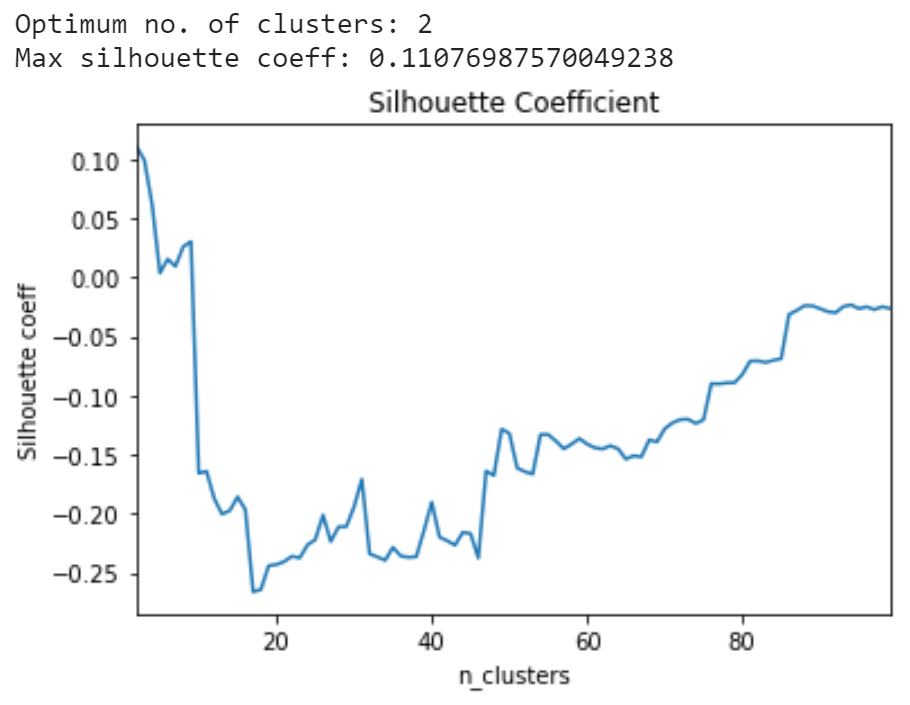
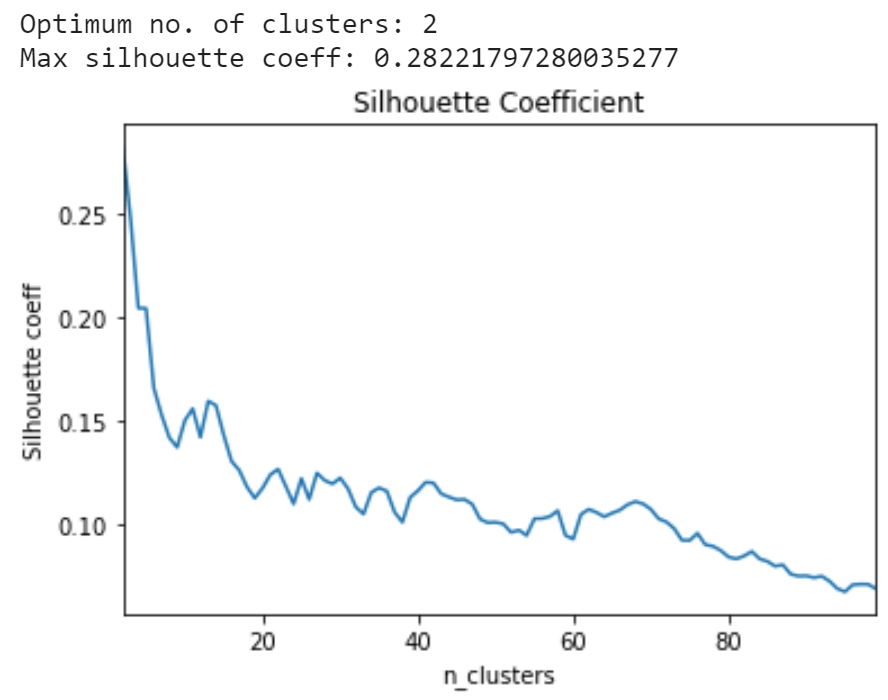




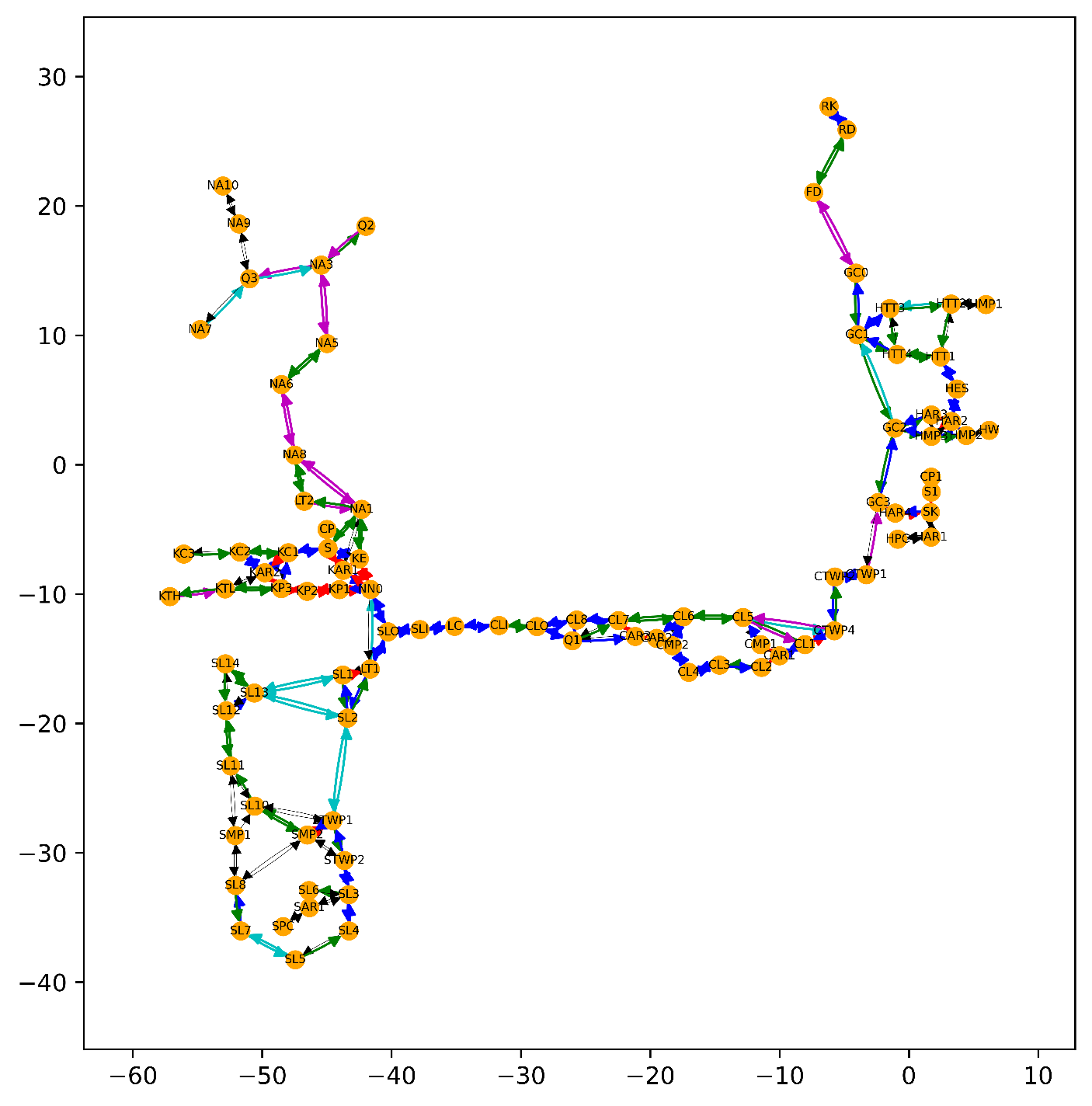


#### Clustering linkage metric (TSC) – average, single or complete

Average: Single: Complete:

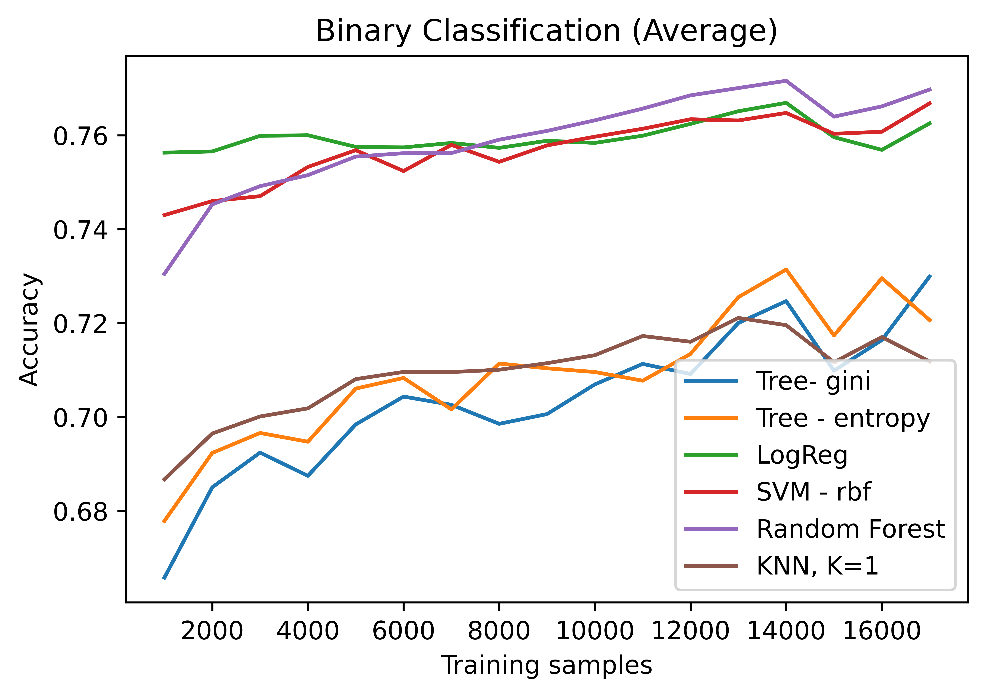
  

**Average is best!** Predicted KS clusters for this linkage are as follows:



**Compare classification methods:**

Random forest still the best. Acceptable accuracy – lower than AAF, but we have fewer datapoints

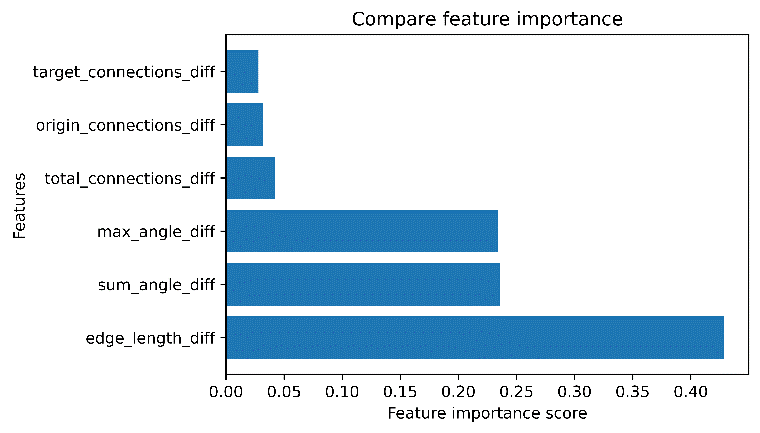
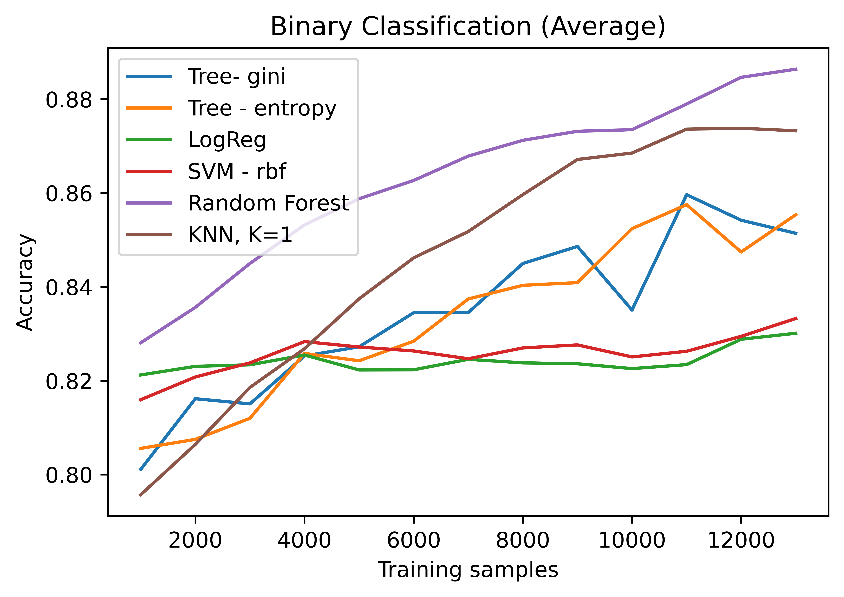


# FRI: Train, test on different maps

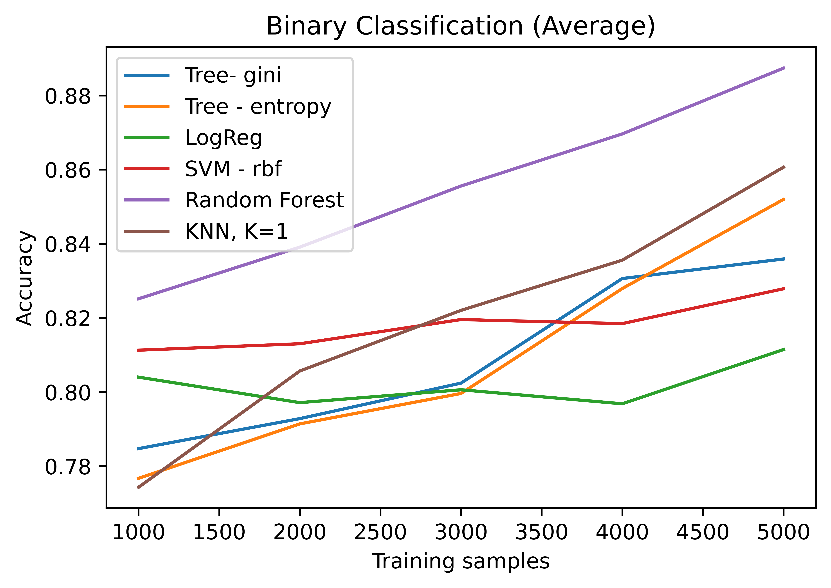
#### Train on AAF, Test on AAF

Cutoff below 20 observations, 7 clusters, average linkage

**Unbalanced dataset:**



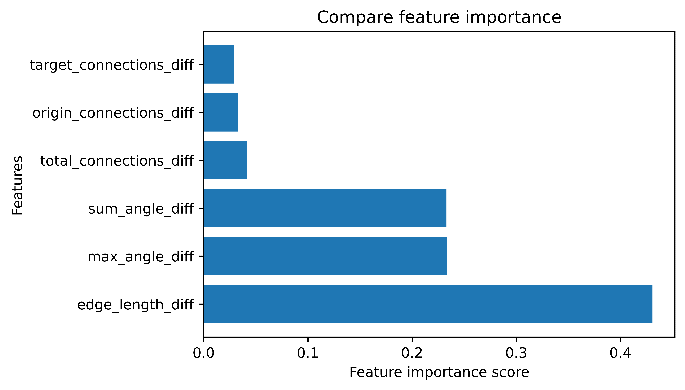
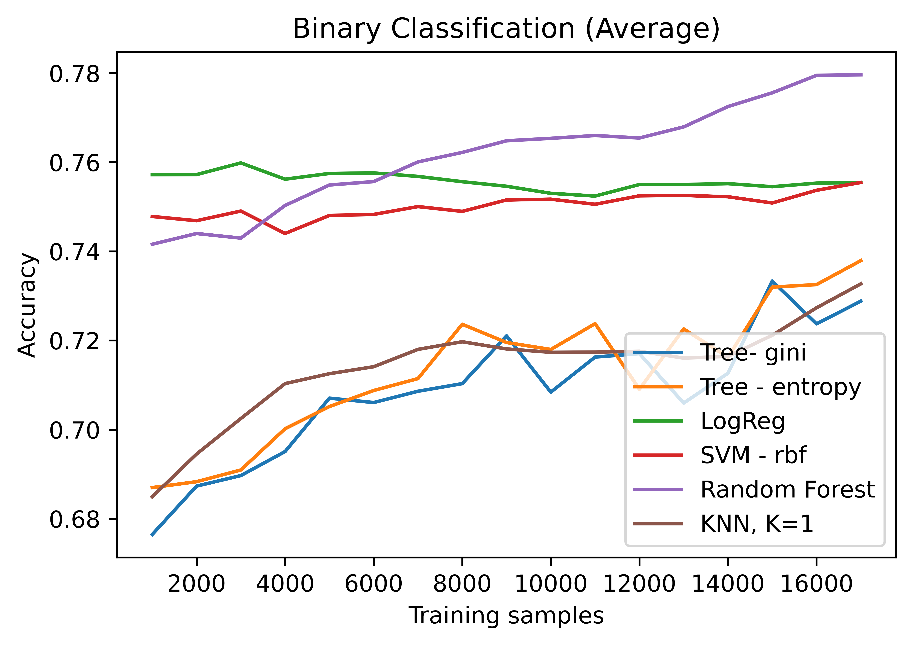
**Balanced dataset:**



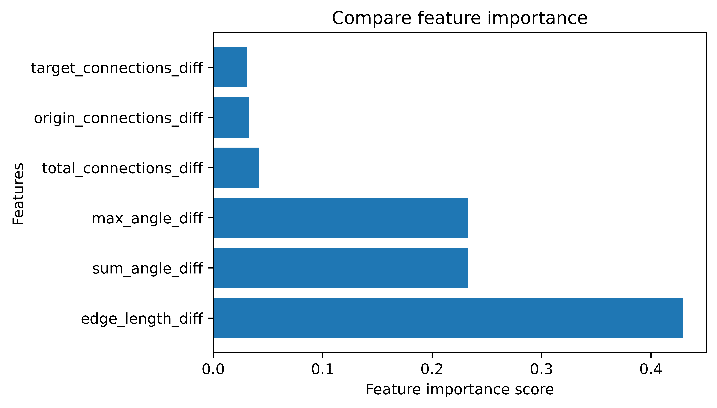
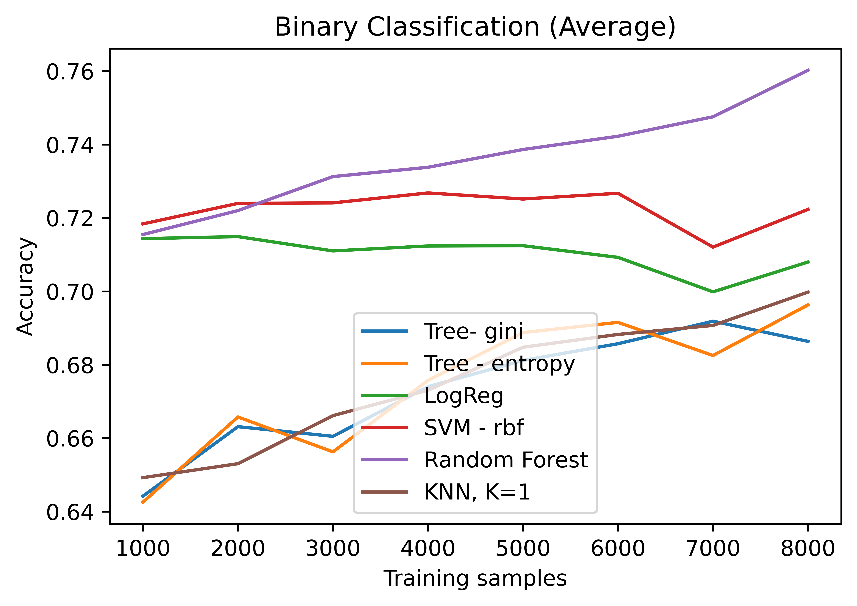
#### Train on TSC, Test on TSC

Cutoff below 20 observations, 6 clusters, Average linkage

**Unbalanced dataset:**



**Balanced dataset:**



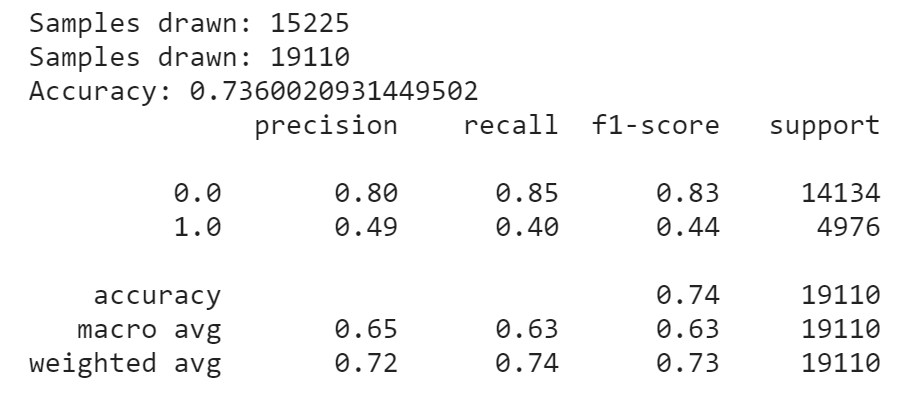
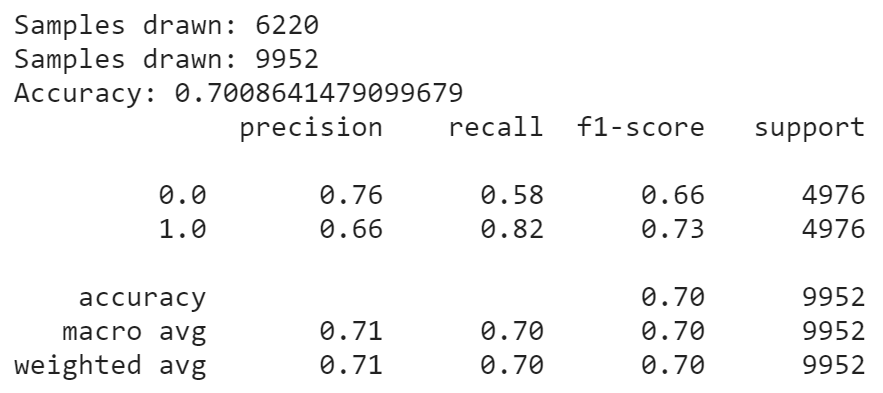
#### Train on AAF, test on TSC

Cut-off at 20 observations, uses “average” linkage

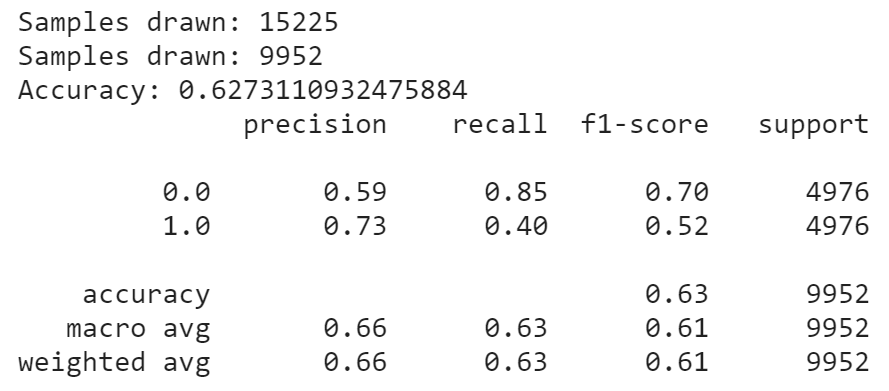
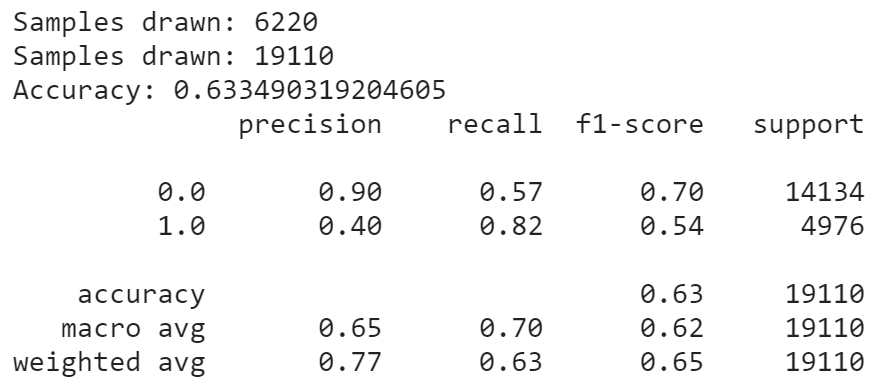
7 clusters for AAF, 6 clusters at TSC

**Point tests:**

AAF unbalanced, TSC unbalanced: AAF balanced, TSC balanced:

AAF unbalanced, TSC balanced: AAF balanced, TSC unbalanced:

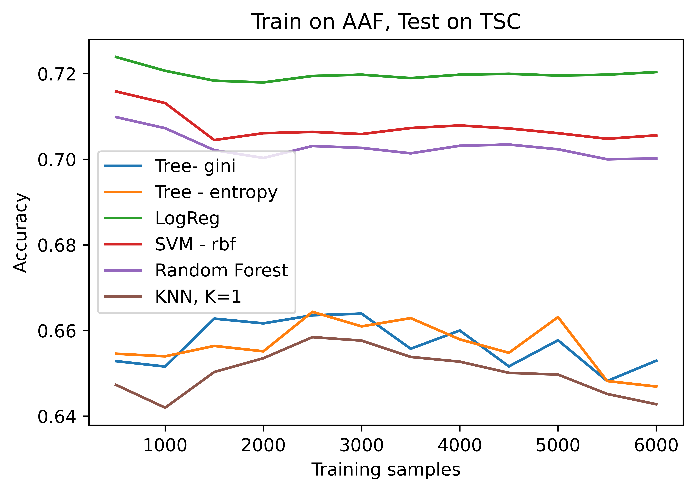
 

Both unbalanced case has higher accuracy, but low f1 score for class 1. This is also not better than the naïve classifier.

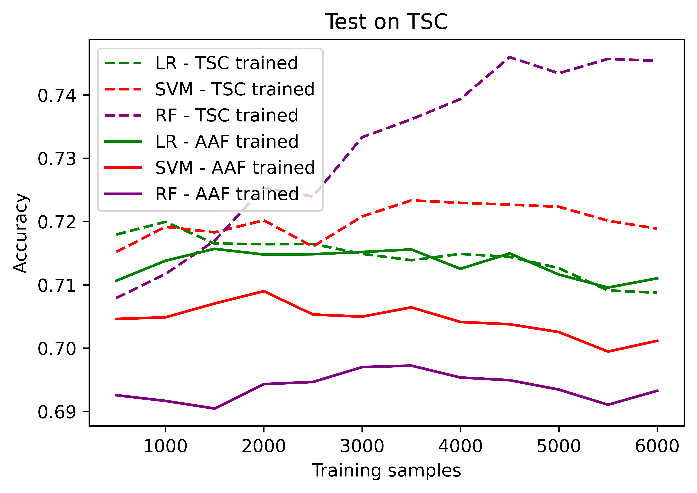
Best is **both balanced** 🡺 70% accuracy & f1 scores. Significantly better than naïve classifier

**Compare all classifiers**

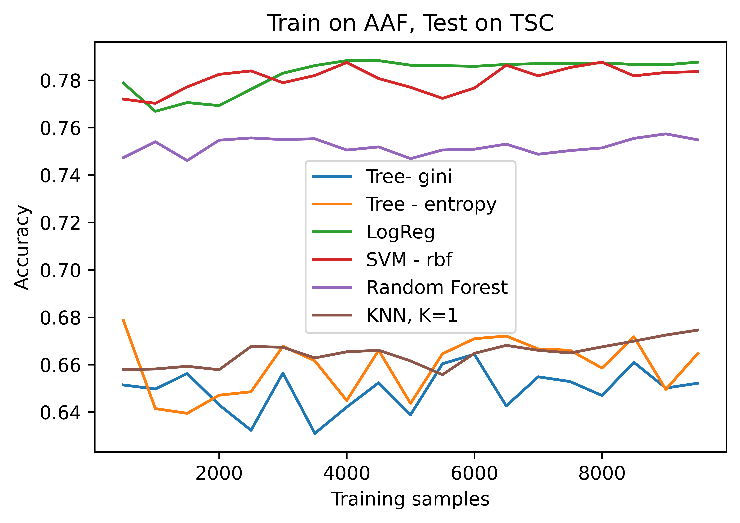
Both balanced



**Compare classifiers trained on AAF vs classifier trained on TSC**



#### Train on TSC, Test on AAF



**Compare classifiers trained on AAF vs classifier trained on TSC**

