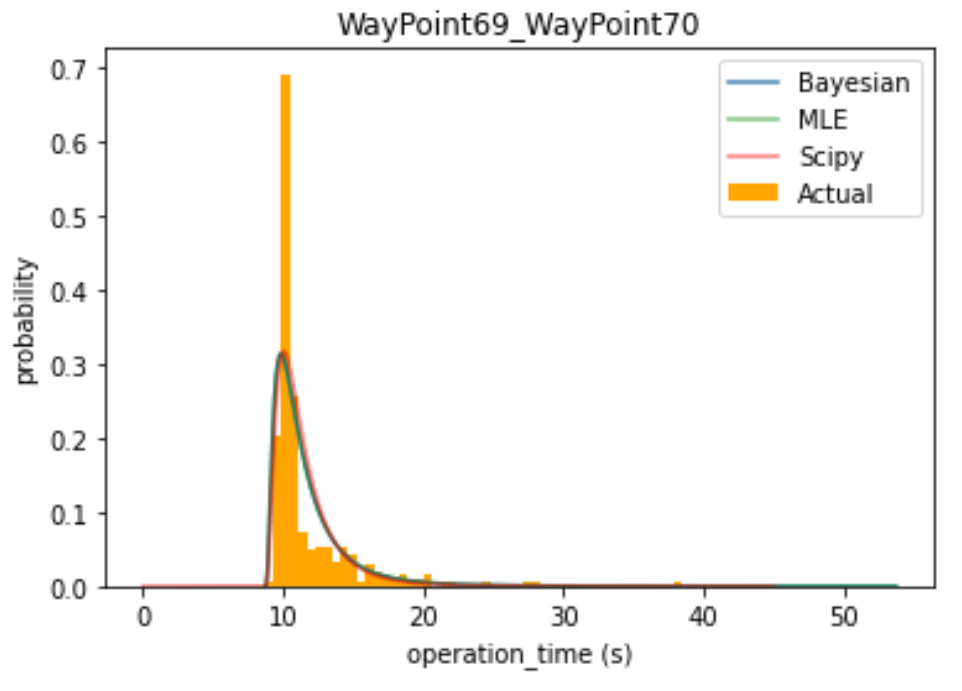
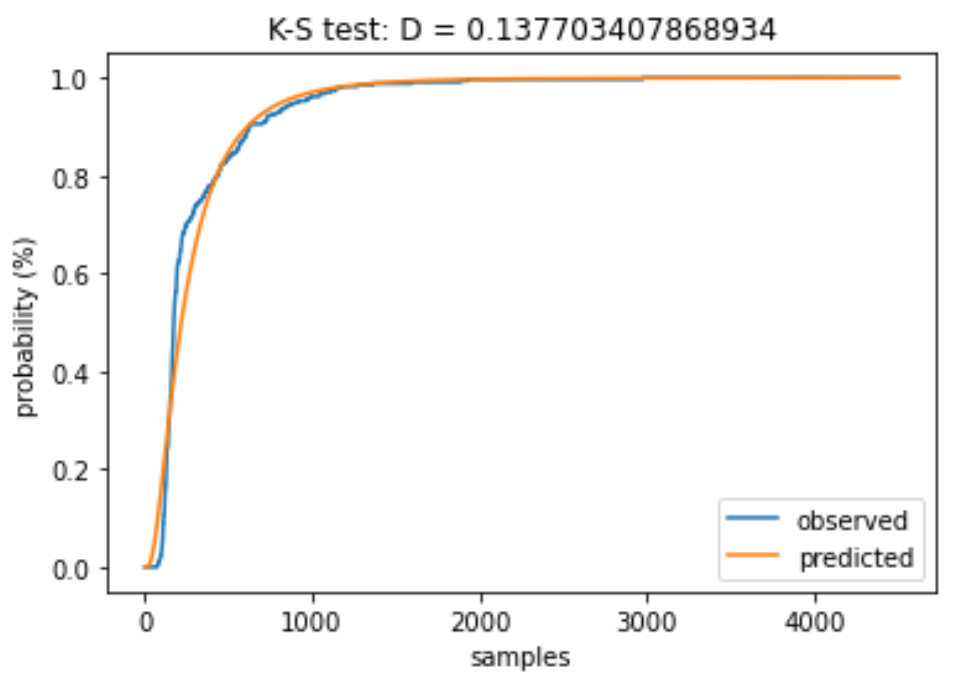
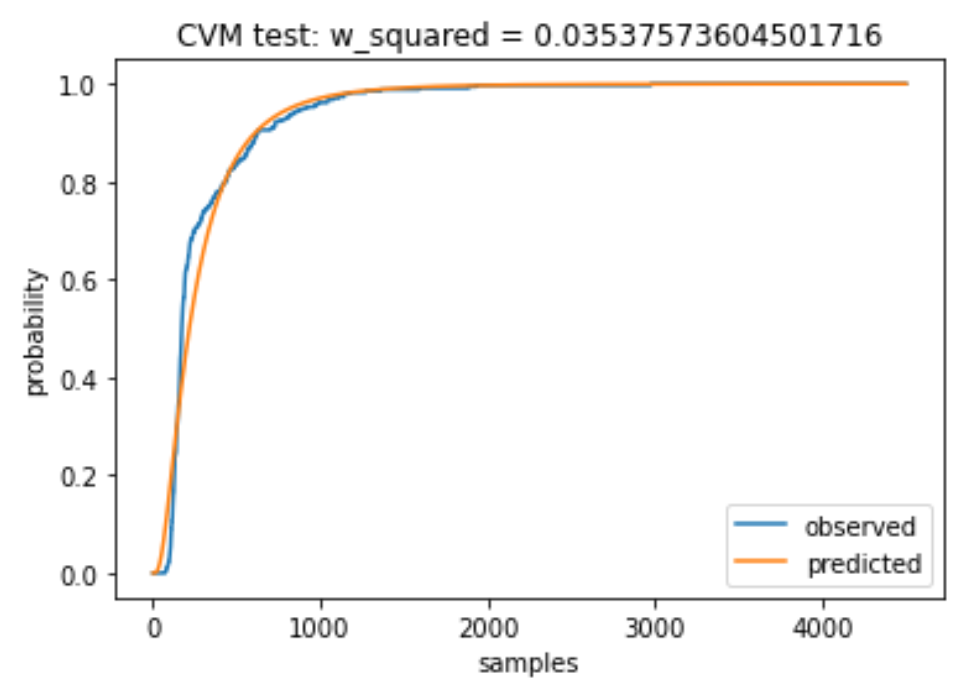
**Week 5 Writeup**

# RECAP: Week 4 Summary

1. **[MON]** – Uploaded Charlie’s data to GitHub. Performed additional filtering to remove instances where the robot starts the first edge of a sequence – since the robot will need to accelerate from stationary. Fixed lognormal model (below)

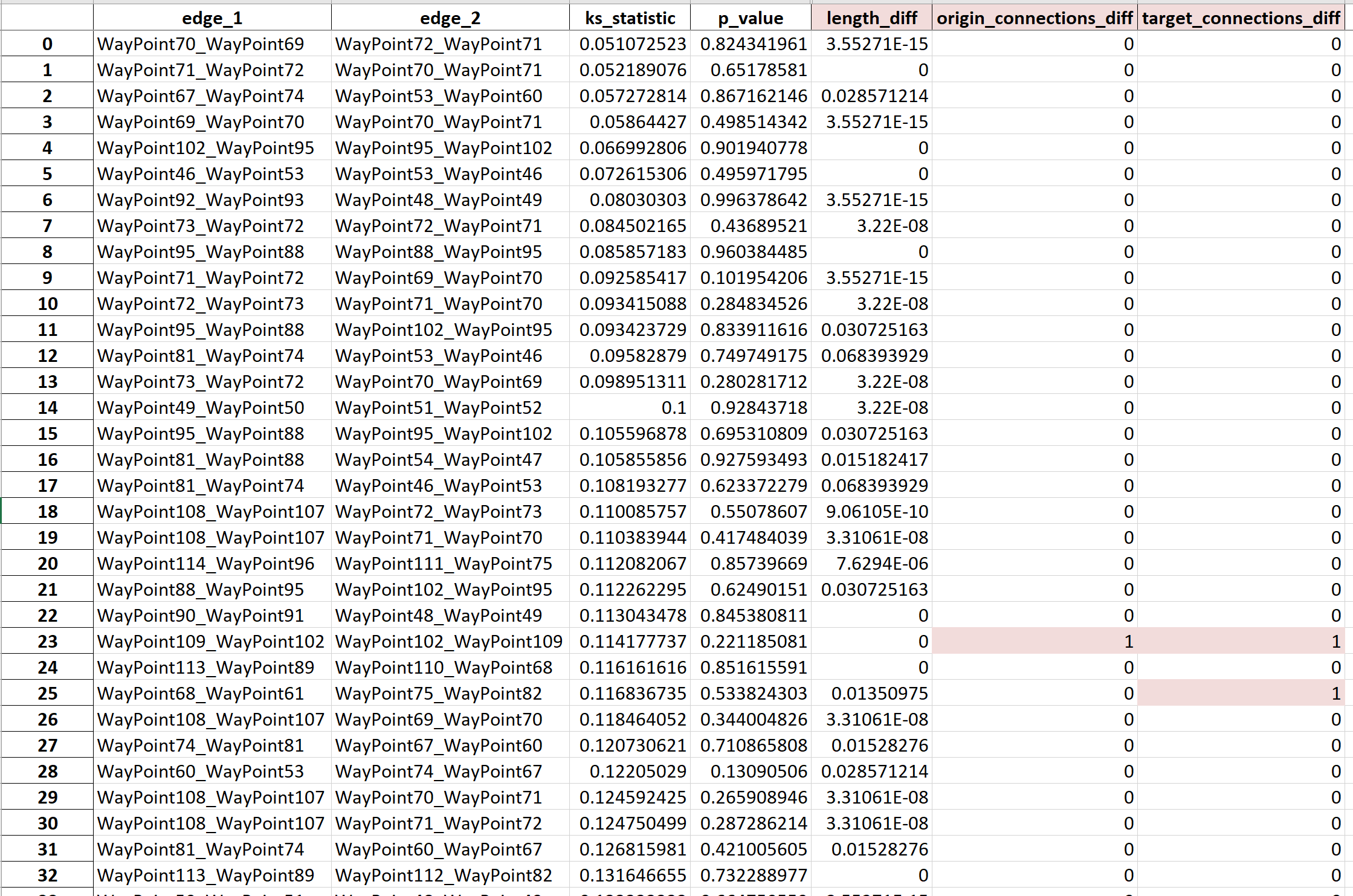
  

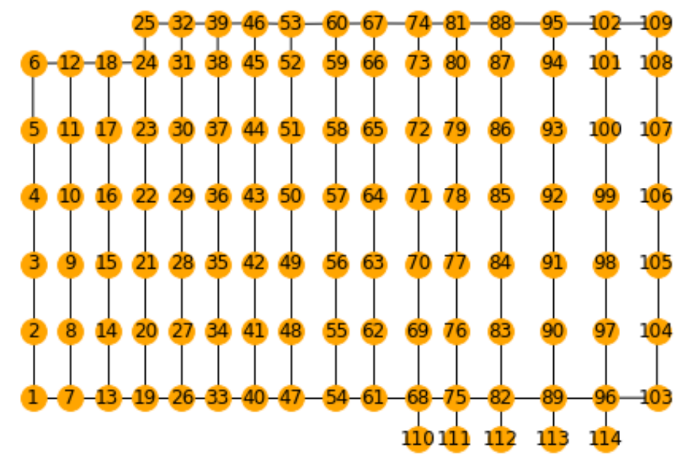
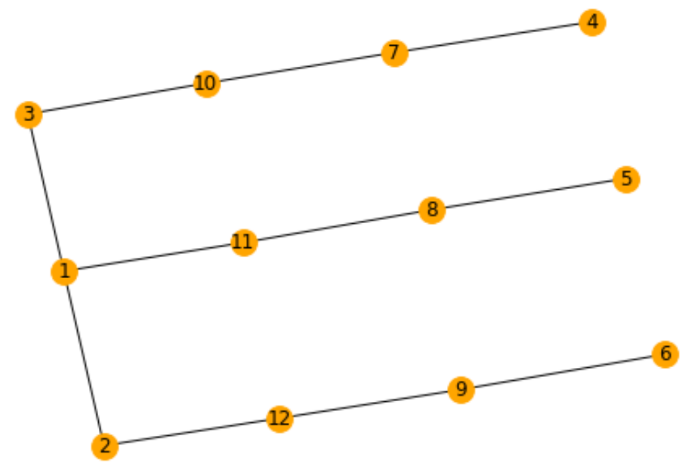
1. **[TUE]** – Inverse Gamma & Inverse Gaussian models
2. **[WED]** – Fixed Inverse Gamma & Inverse Gaussian + implement offsets/cutoffs. Conclusion is that lognormal is more convenient to implement since it can use the same conjugate priors as the normal distribution.

|  |  |  |
| --- | --- | --- |
| **Fitting method** | **KS-statistic** | **CVM-Statistic** |
| Lognormal Bayesian | 0.13770 – (2nd lowest) | 0.03538 – (2nd lowest) |
| Lognormal MLE | 0.13892 | 0.03503 |
| Lognormal Scipy | 0.10549 | 0.04058 |
| Invgamma No Offset | 0.17431 | 0.10897 |
| Invgamma Offset | 0.55566 | 0.37515 |
| Invgamma Cutoff | 0.10861 | 0.13077 |
| Invgamma Scipy | 0.08481 | 0.02981 |
| Invgamma Product of Priors | 0.55564 | 0.37079 |
| Invgauss Offset | 0.16969 | 0.02910 |
| Invgauss Scipy | 0.10426 | 0.04541 |

\* Pink denotes Bayesian methods. Another reason for choosing lognormal is that both KS and CVM statistics are very low (which is good) compared to other Bayesian methods. These are comparable to the statistics for Scipy MLE fit.

1. **[THU]** – KS test between all pairs of edges. Isolate similar edges with a high p-value and dissimilar edges with low p-value.
2. **[FRI]** – Augment KS test data with context from topological maps. Similar edges have similar edge lengths and the same number of connections at origin/target nodes (see table below). Create topological maps





\* Walmart (left),

Blenheim (right)

# Week 5 Plans

**Goal for this week:** How many data points are needed to build a good representation of the same edge? Predict distributions of similar edges.

1. **[MON]** – Weekly meeting. Implement any recommendations for last week’s work. Run model-fitting on multiple edges to get more robust data. Continue with PPT
   1. How to calculate p-value from KS statistic? What does the p-value mean? - Done
2. **[TUE]** – Finish PPT. Present at research update meeting
3. **[WED]** – Choose a probabilistic scoring metric to see how well predictions perform
4. **[THU]** – How many data points are needed to build a good representation of the same edge?
5. **[FRI]** – Generalise between similar edges
6. **[EXTRA]** – Look at STRANDS data and see if there are obvious differences when more noise is present

**Additional questions:**

1. Figure out subsets of edges that are similar

* Clustering algorithms in Scipy – using KS Statistic & distances between edges & number of end nodes
  + Try hierarchical clustering
* Colour edges according to clusters

1. We have 2 edges that are similar. Can we merge the data and use that to predict both edges.
2. Can you also generalise between dissimilar edges? E.g. different edge length, but same number of origin/target nodes.

* Look at difference between robot starting from stationary at a node vs already moving vs turning

1. Given a topological map, find the minimum set of edges to gather data on (and how many observations per edge)

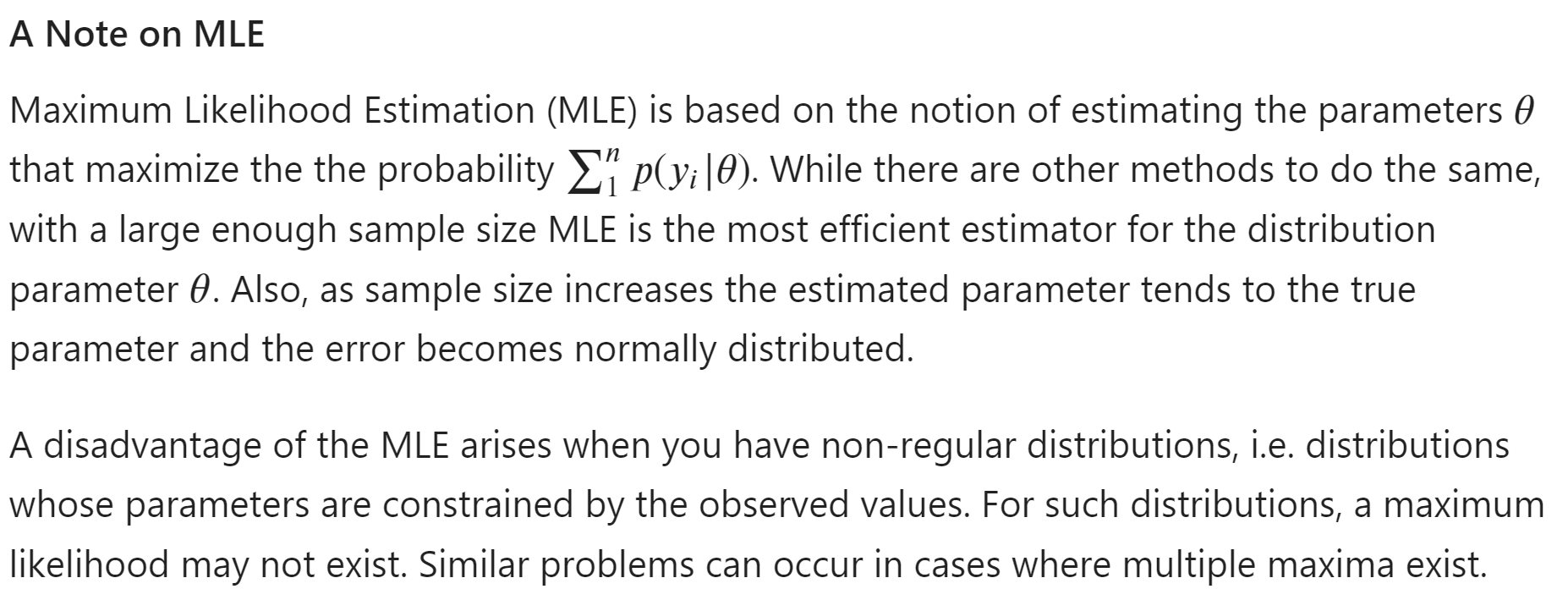
* Dynamic cutoff point – threshold p-val/KS stat
* Given the data for those edges, generate distributions for all edges

1. So far, we have used non-congested data. Will our models still work if there are 2, 3, 4 … robots per edge?

* Much less data available
* How does similarity between edges change as we increase the number of robots on an edge.

**Questions from the PPT**

1. Should new data points (e.g. of a new edge) have a greater impact on Bayesian updates than old data points
   1. Each data point count’s multiple times? Decaying to ordinary?
2. Consider the edge velocities
   1. Use edge length and max/expected velocity to predict MAP estimate of mean
   2. Use number of end nodes to predict MAP estimate of variance
   3. Compute a prior that reflects these MAP estimates and your uncertainty
3. What are the alternatives to the Bayesian method

\*Bayesian Inference (Coursera 3a)

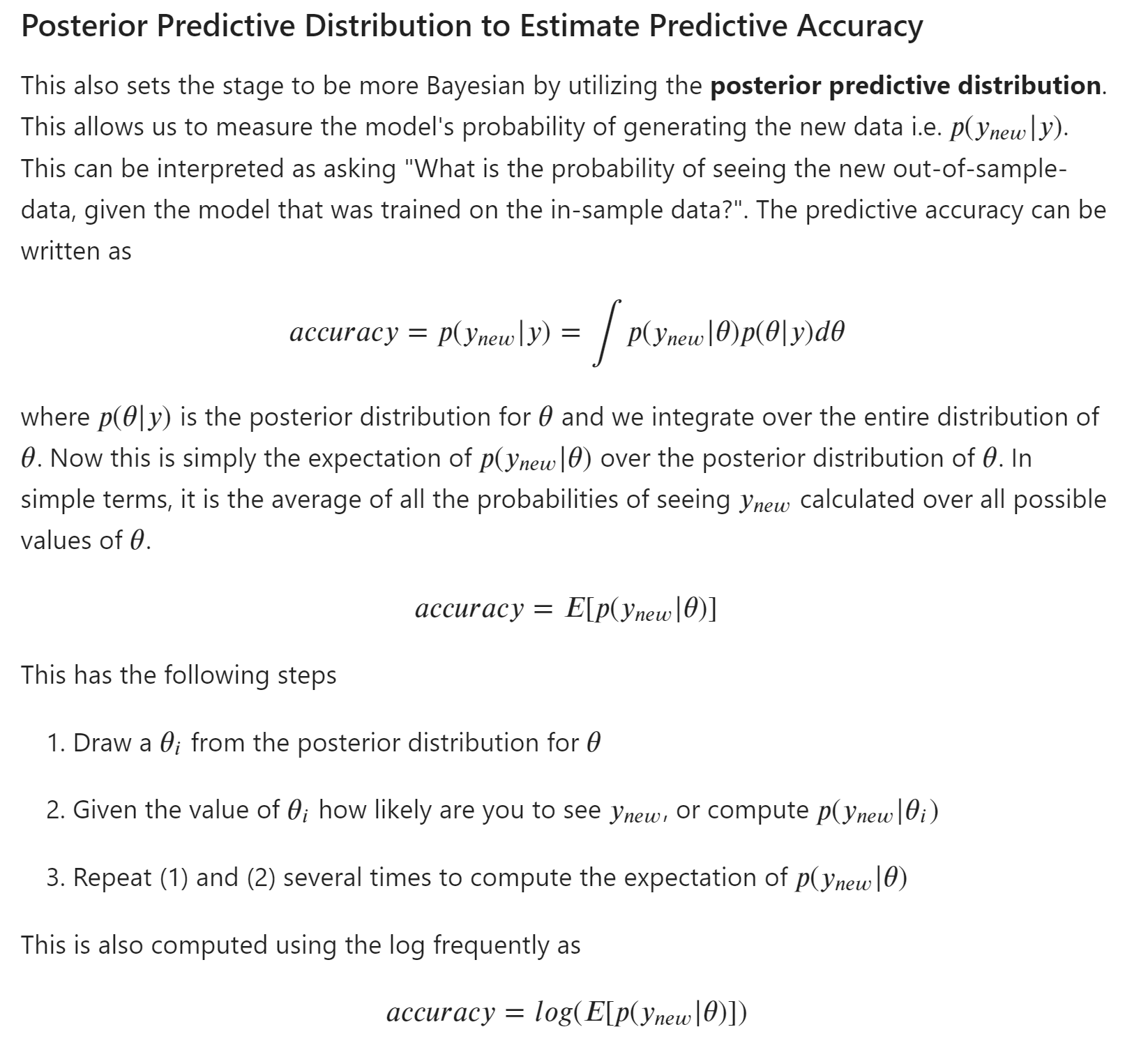
1. Consider using phase-type distributions?

# Measures of Predictive Accuracy

#### Useful definitions

Score function/Scoring rule:

#### Posterior predictive accuracy



[From Scratch: Bayesian Inference, Markov Chain Monte Carlo and Metropolis Hastings, in python | by Joseph Moukarzel | Towards Data Science](https://towardsdatascience.com/from-scratch-bayesian-inference-markov-chain-monte-carlo-and-metropolis-hastings-in-python-ef21a29e25a)

* Implement MCMC from scratch

[Estimating the posterior predictive distribution by sampling - YouTube](https://www.youtube.com/watch?v=TMnXQ6G6E5Y)

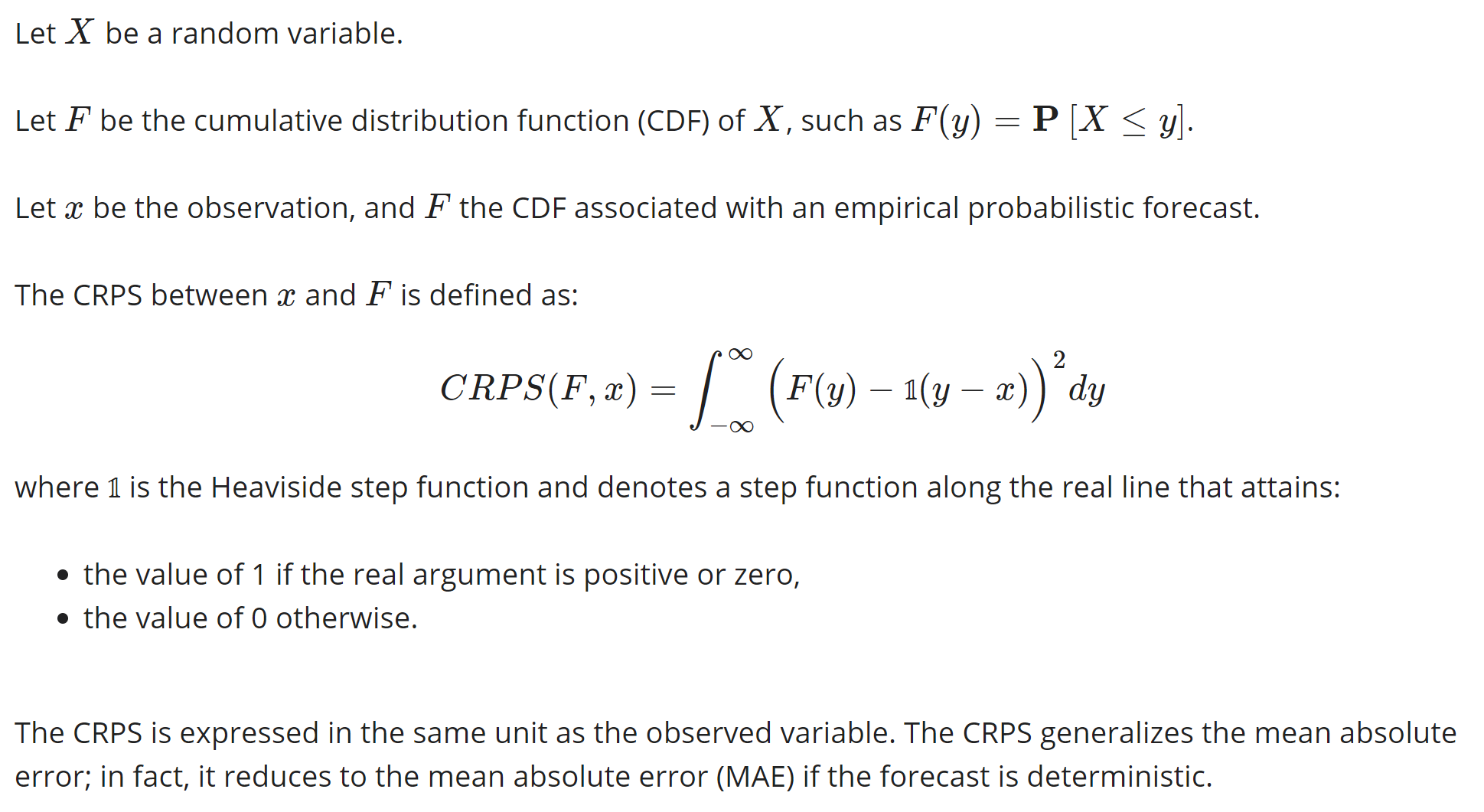
* See here for a non-parametric method of creating a duration distribution

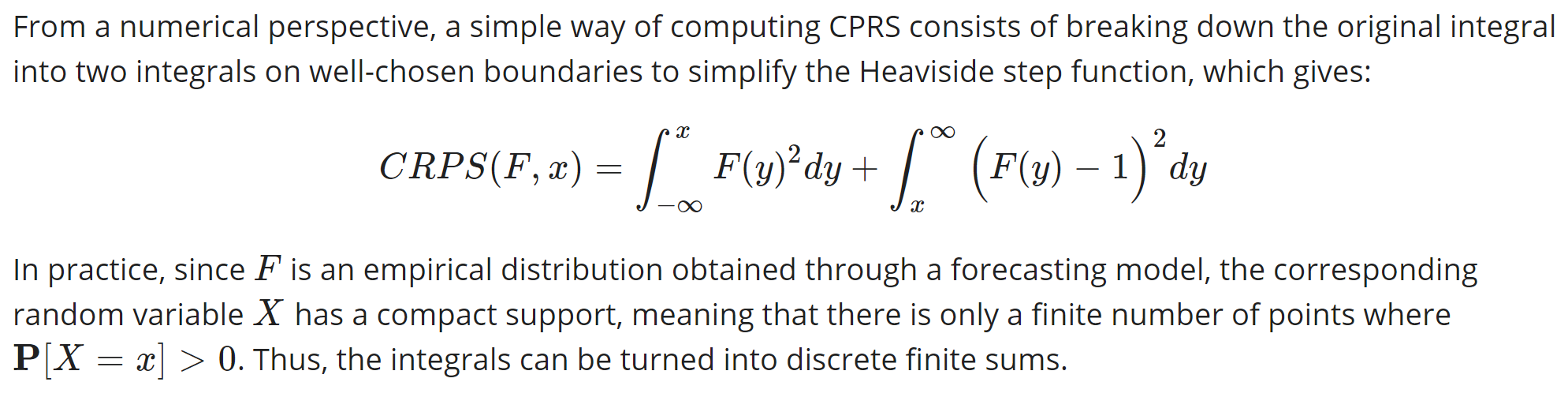
#### Continuous Ranked Probability Score

[Continuous Ranked Probability Score (CRPS) (lokad.com)](https://www.lokad.com/continuous-ranked-probability-score)

[properscoring/\_crps.py at master · TheClimateCorporation/properscoring (github.com)](https://github.com/TheClimateCorporation/properscoring/blob/master/properscoring/_crps.py)

The continuous ranked probability score (CRPS) is a much used measure of performance for probabilistic forecasts of a scalar observation. It is a quadratic measure of the difference between the forecast cumulative distribution function (CDF) and the empirical CDF of the observation



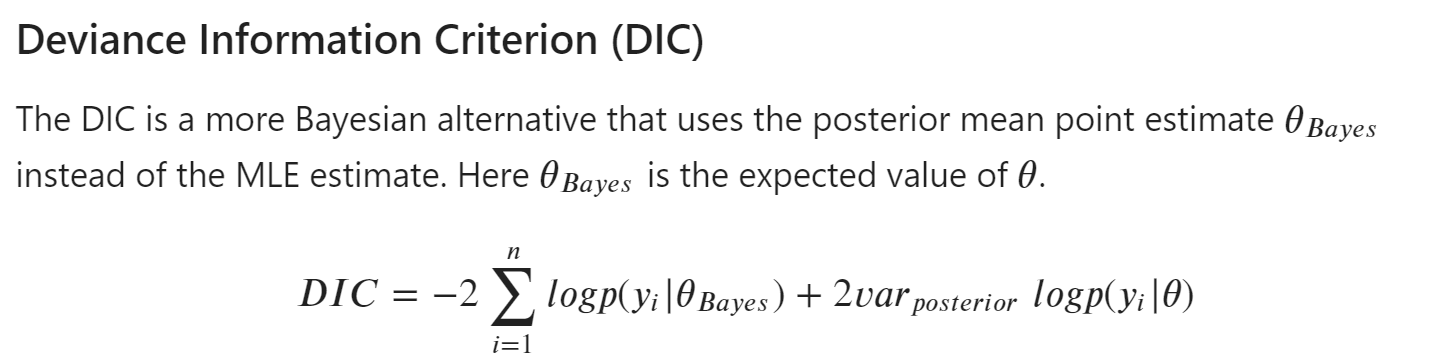


Functions needed

* Integrate pdf to cdf
* Integration function

#### Brier Score

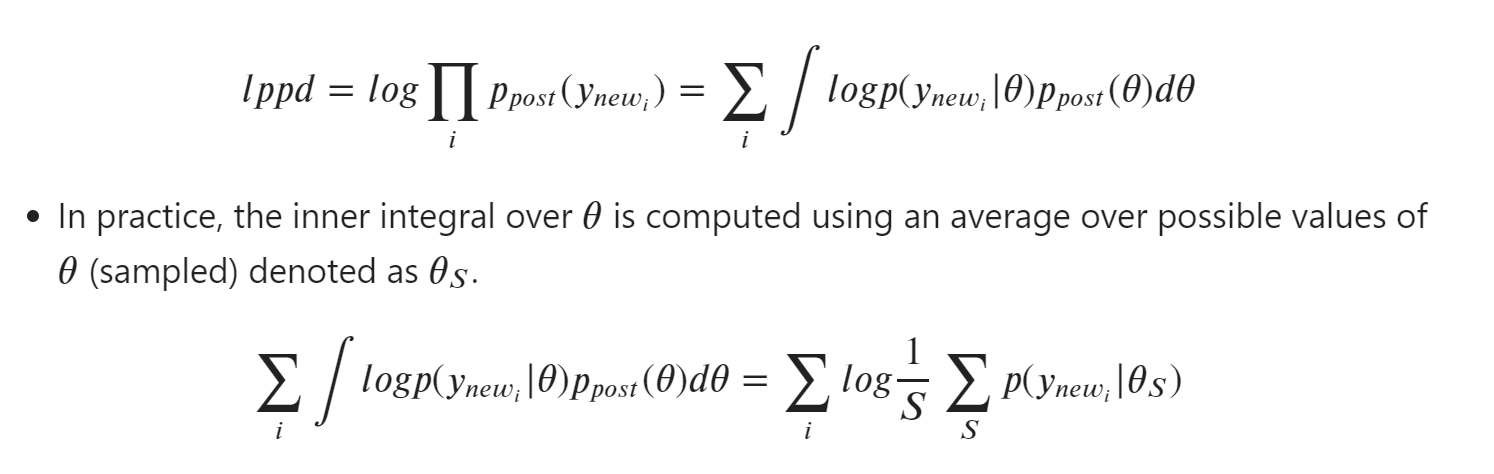
#### DIC – more Bayesian than AIC & BIC



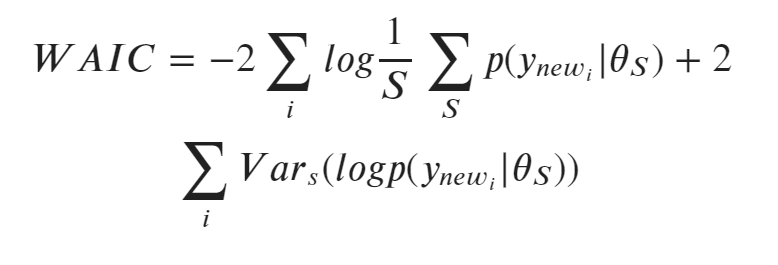
2nd term is variance of log-prob(yi | params) over whole posterior distribution

#### WAIC – most Bayesian

Log Pointwise Predictive Density is similar to the posterior predictive model (above)



WAIC adds a penalisation term to reduce the number of parameters (and therefore overfit)



# Clustering

Used Hierarchical (agglomerative) clustering with distance matrices (ks score, edge\_difference, node\_diff) on “walmart\_random” data.

**Clustering by ks statistic is exactly the same as the edge length difference**

Metric for choosing the optimal number of clusters:

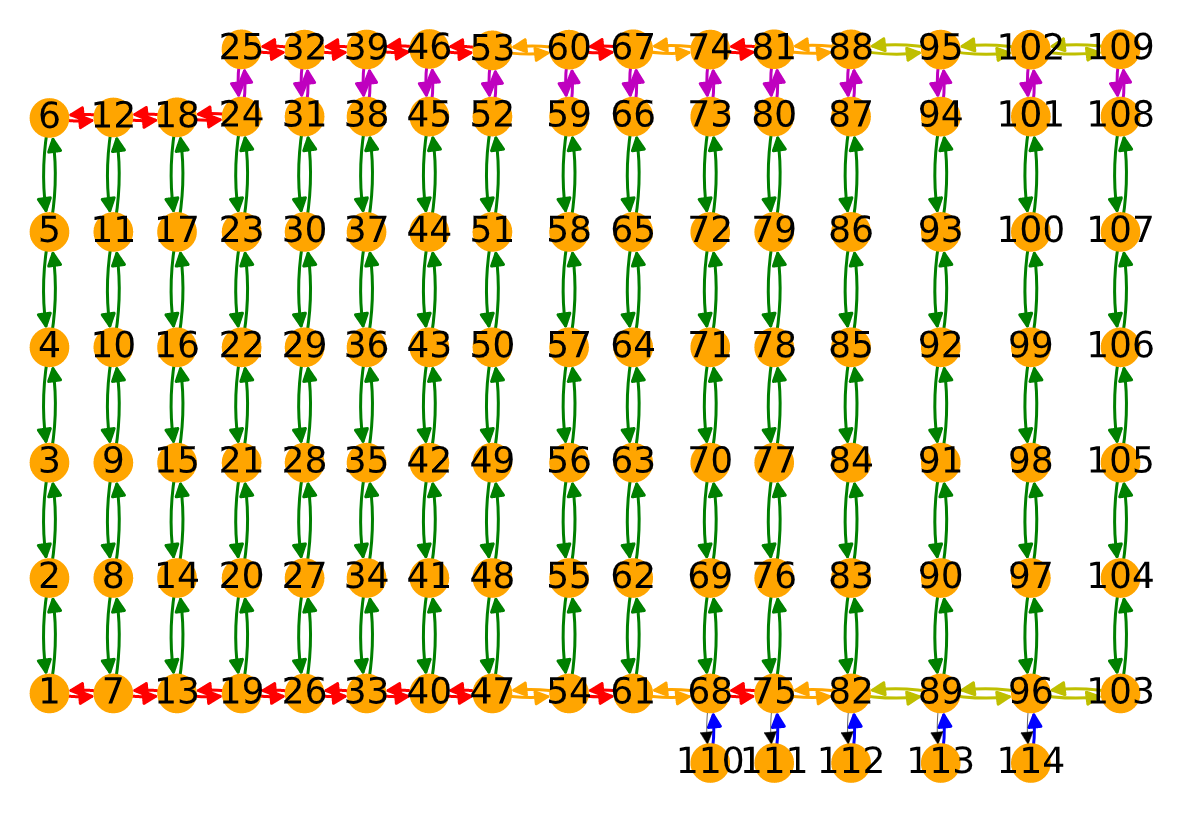
1. By Inspection: increment no. of clusters from 2, stopping when an increase does not create noticeable change in the clusters
2. By Silhouette score ([2.3. Clustering — scikit-learn 0.24.2 documentation](https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation))
   * **A** = Mean distance between 1 sample and all other points in the same cluster
   * **B** = Mean distance between 1 sample and all other points in the **next nearest cluster** (i.e. compute B for all other clusters and choose the minimum)
   * **S\_sample** = (B – A) / max(A,B)
   * **S** = mean(S\_sample)

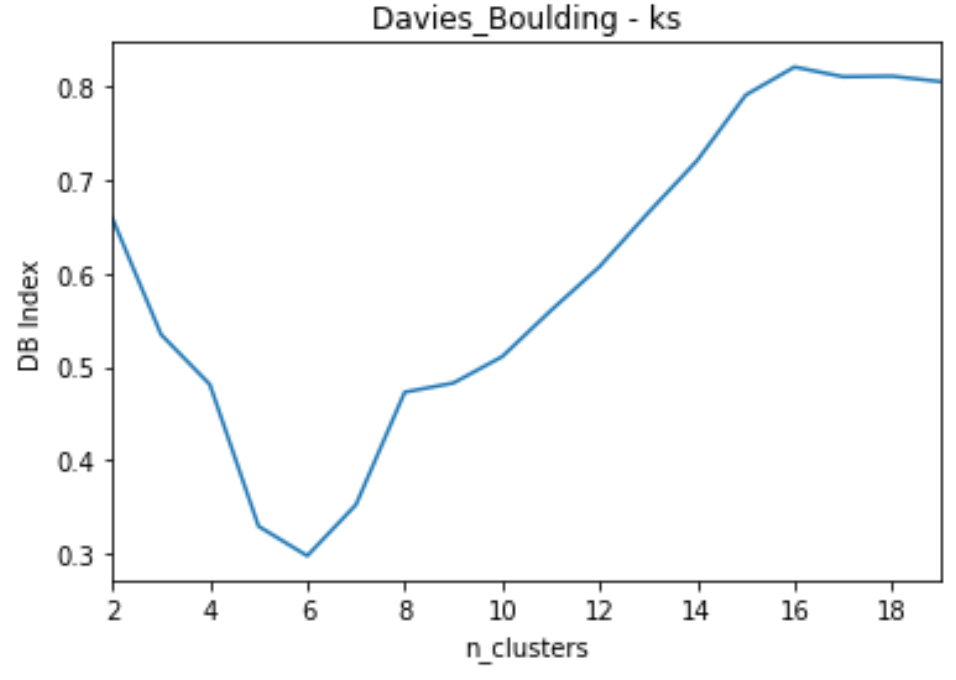
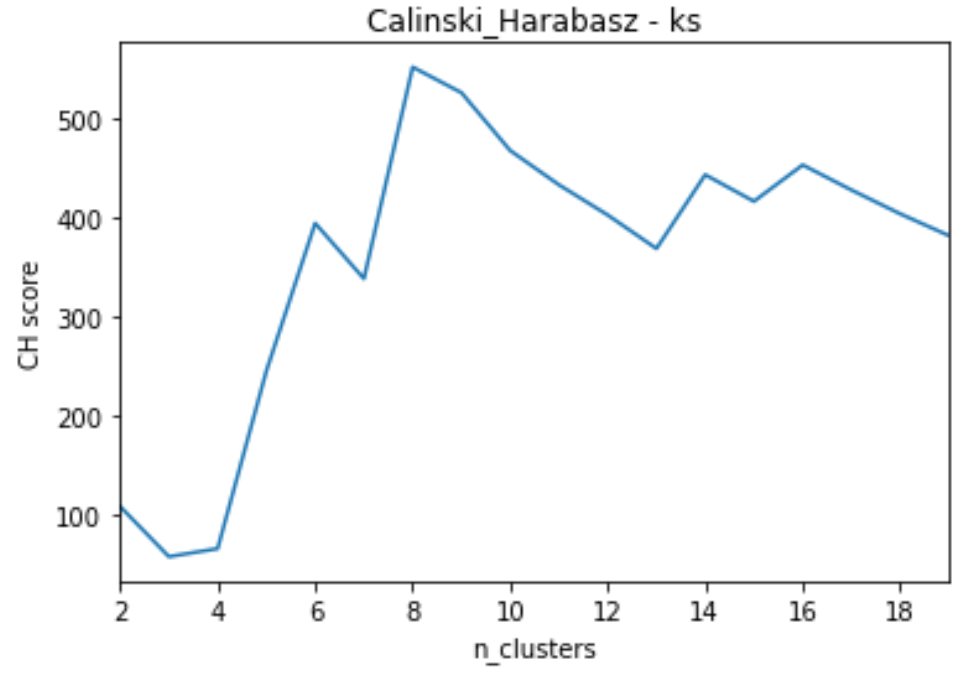
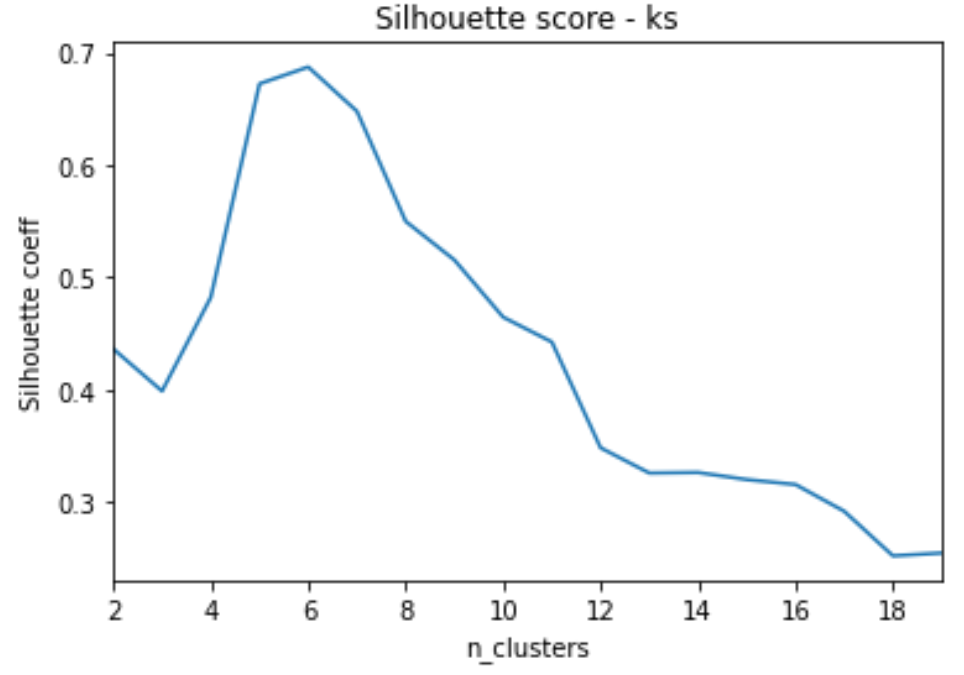
Other metrics for evaluating the density and separation of clusters (i.e. ground truth not known)

* Calinski-Harabasz (CH) Index – higher is better
  + Sum of square distances between all clusters / sum of square distances within all clusters
* Davies-Bouldin (DB) Index – close to zero is best, faster to compute than silhouette
  + Si = cluster diameter = mean distance between centroid and points in cluster i
  + Dij = distance between centroids of clusters i & j
  + Rij = similarity = (Si + Sj)/ Dij
  + DB = (1/n) \* SUM( max(Rij) where i != j ) for n total clusters

#### By KS stat (no data for black edges):

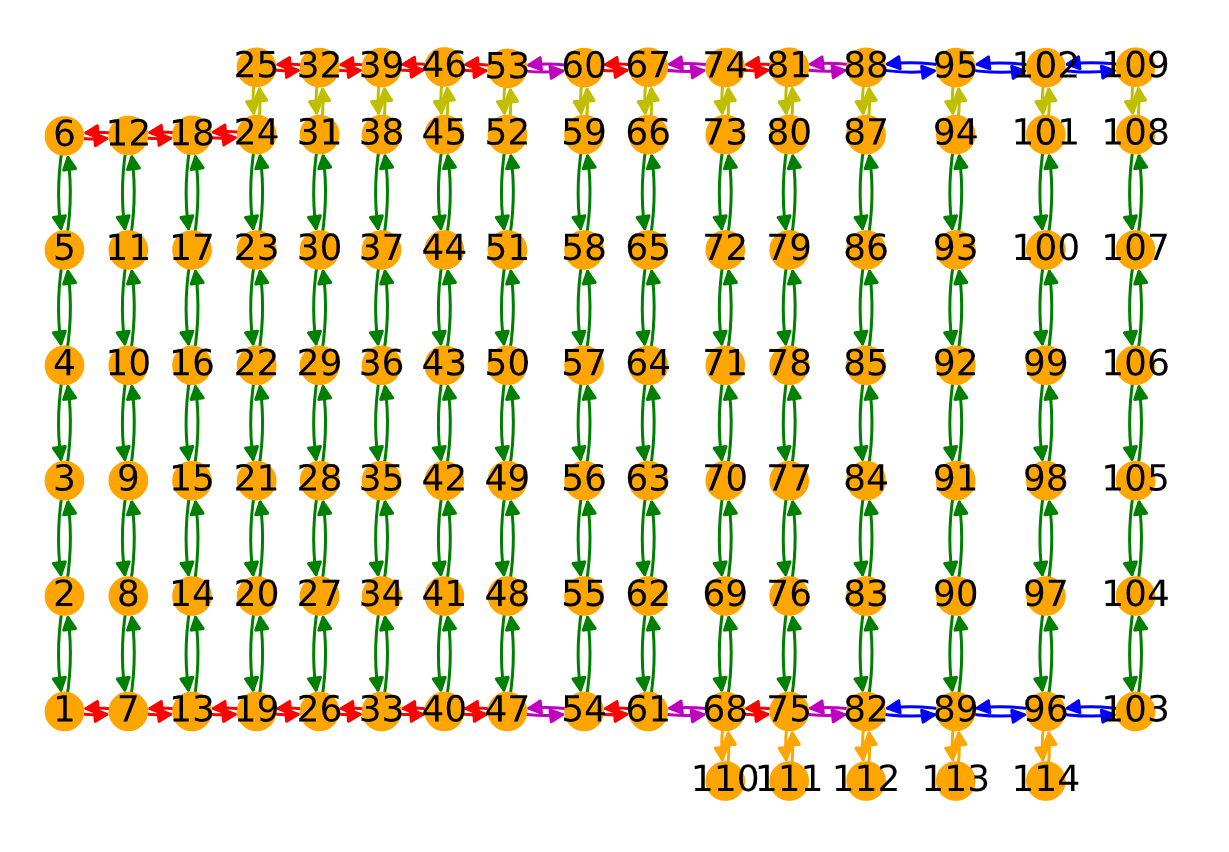
* 6 clusters has highest silhouette score of 0.687
* Agrees with lowest DB score of 0.298

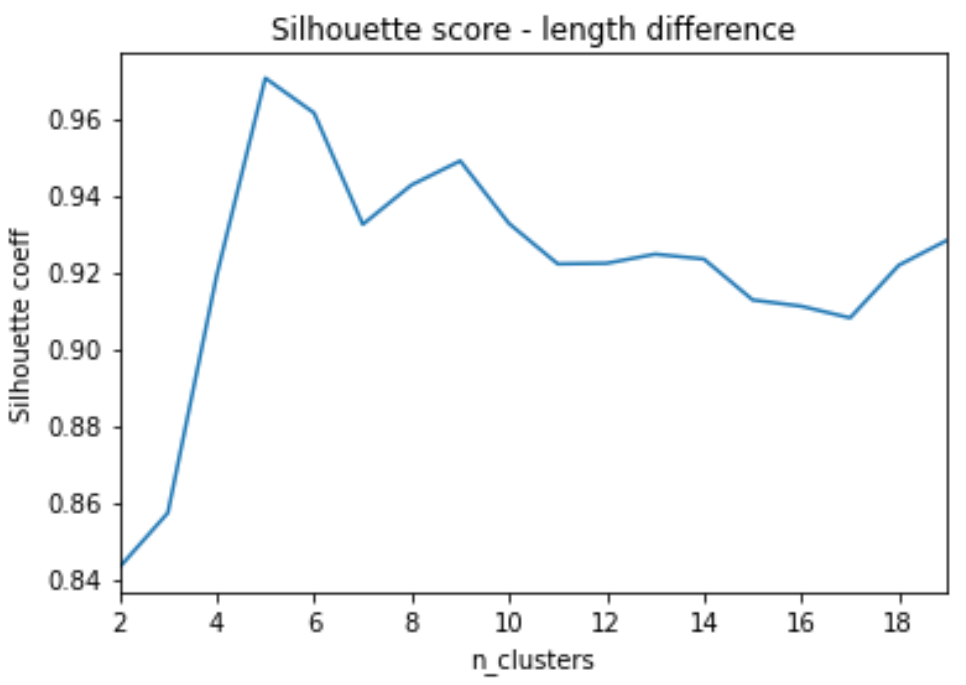




#### By edge\_diff:

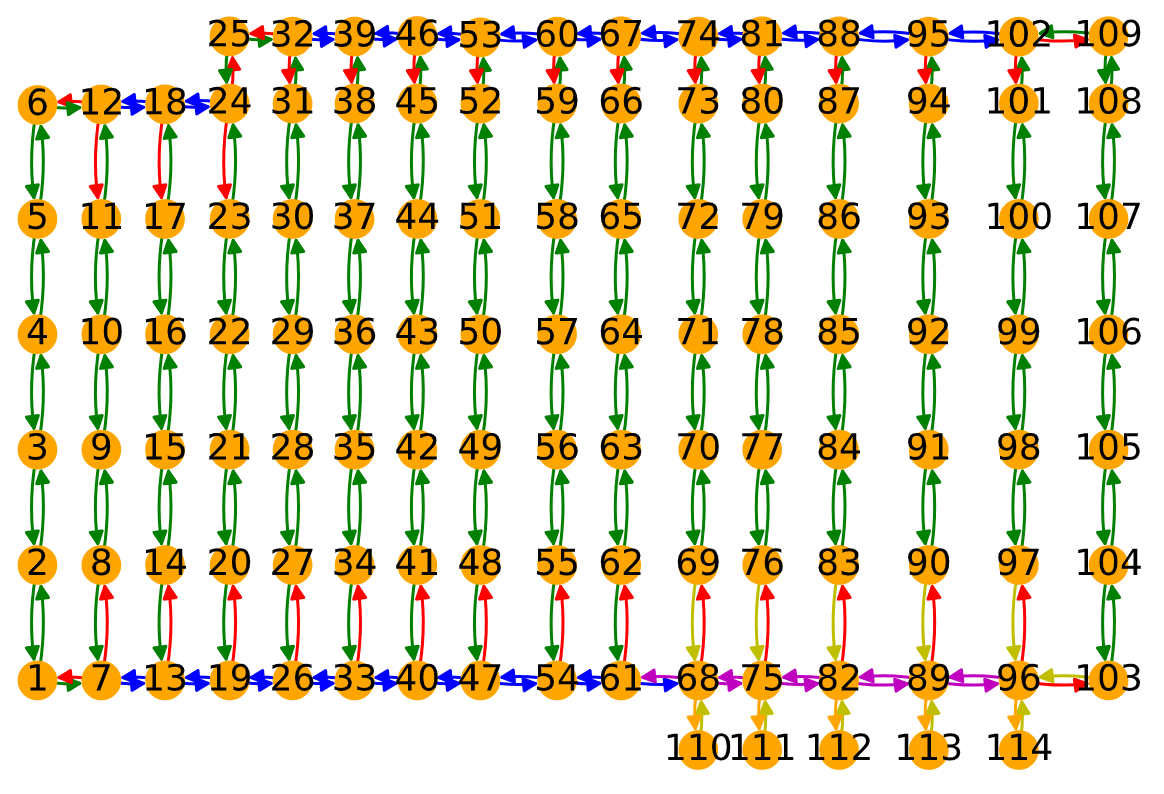
* 5 clusters has highest silhouette score of 0.971

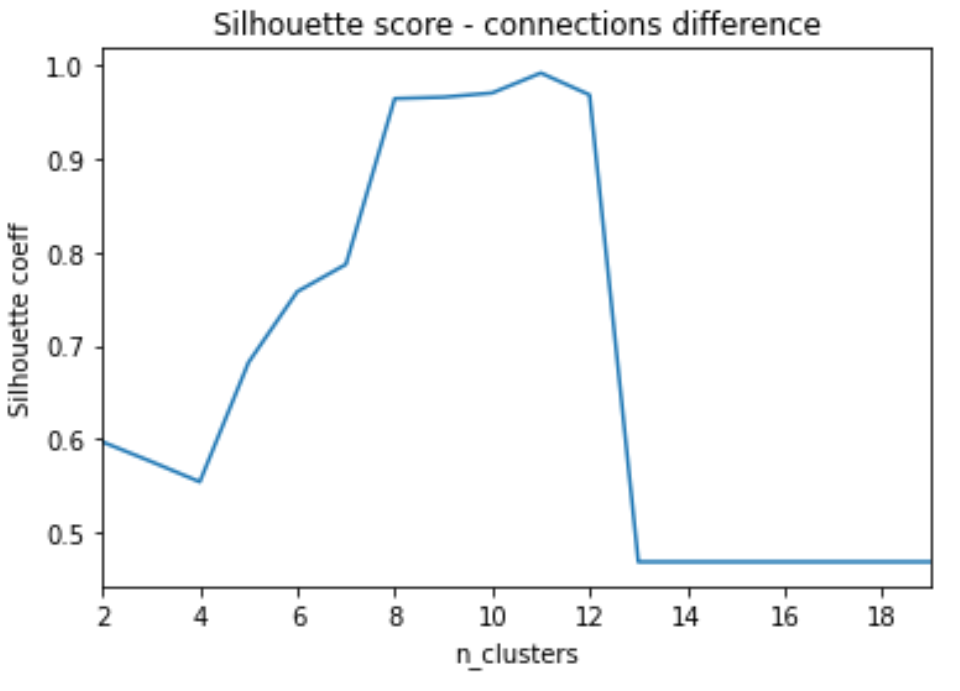




#### By connections\_diff:

* 11 clusters has highest silhouette score of 0.992
* However, 8 clusters has very comparable silhouette score of 0.966



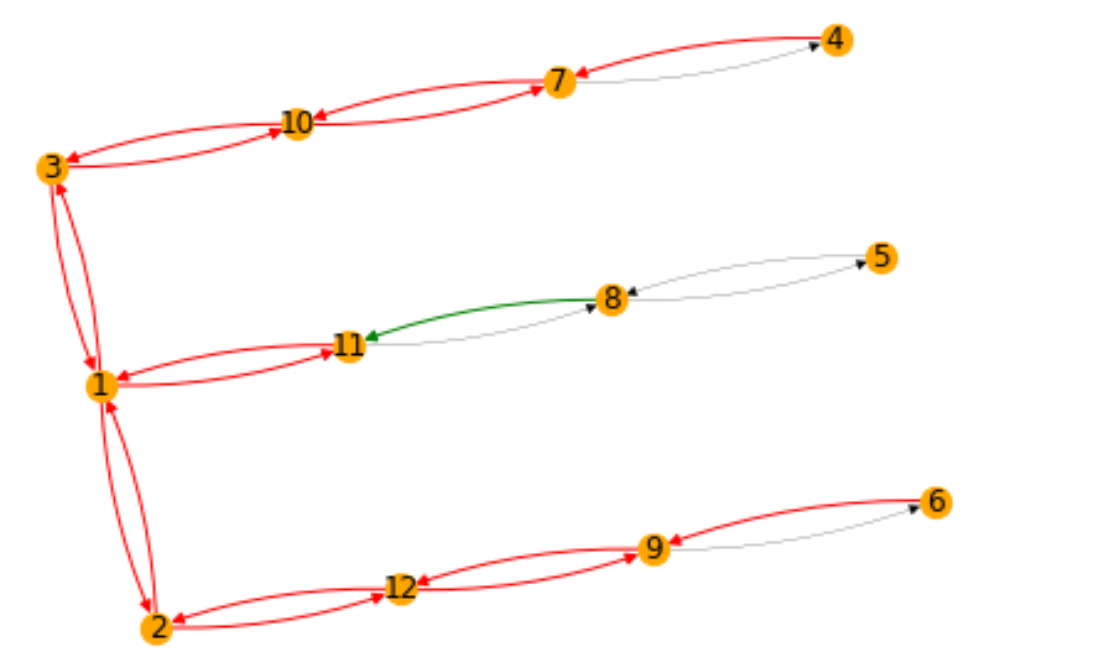


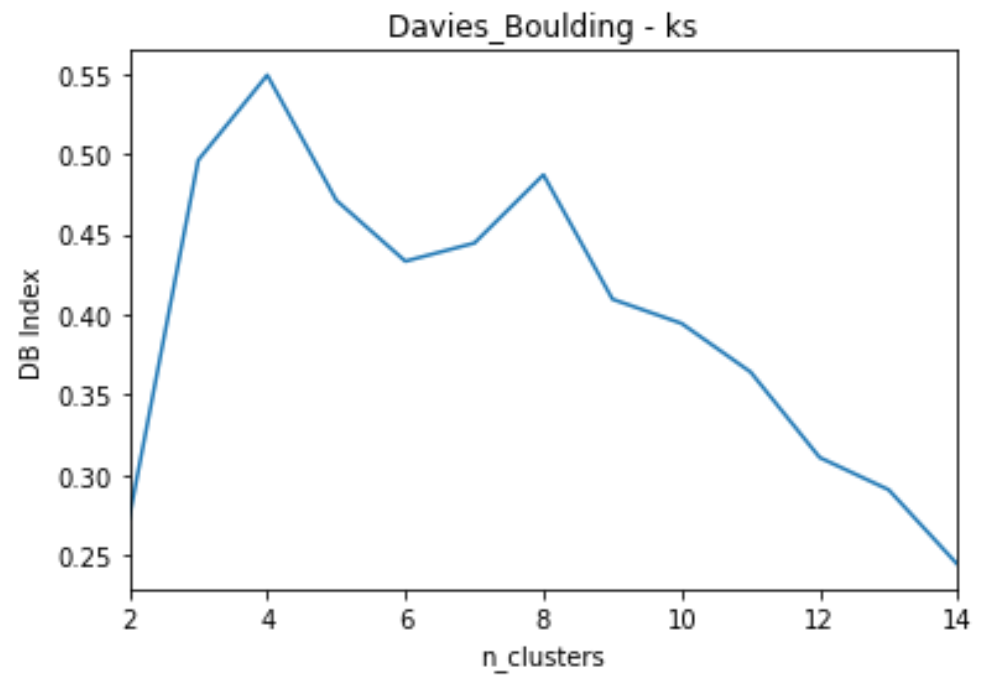
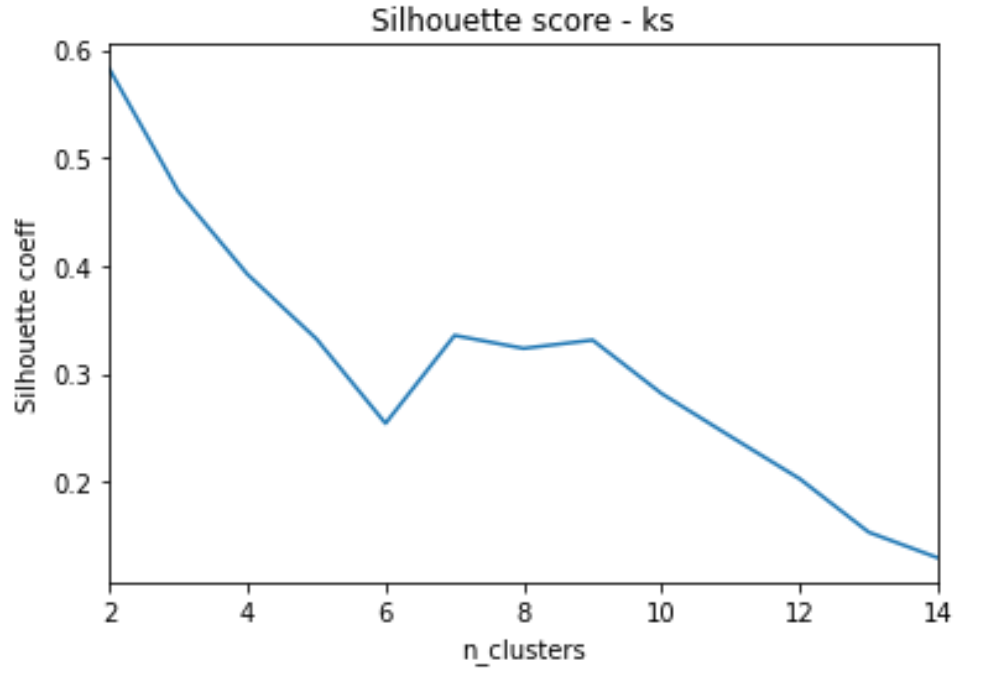
# Greater Congestion

Uses “Blenheim\_targeted” data & congestion also considers cases where a robot is on an adjacent edge

#### N\_robots = 1 (no congestion)

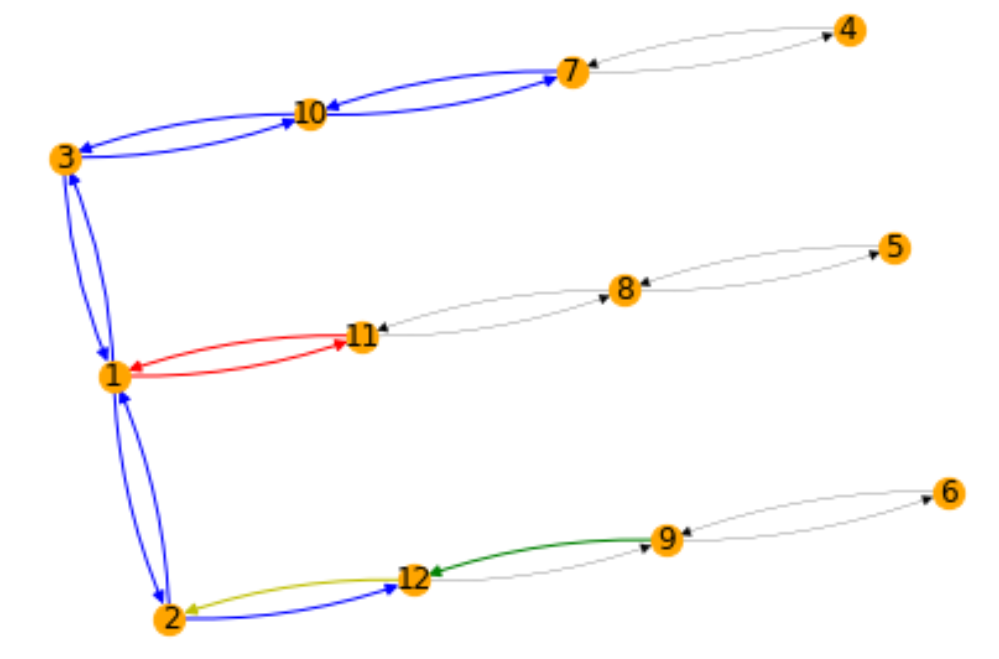
Single cluster

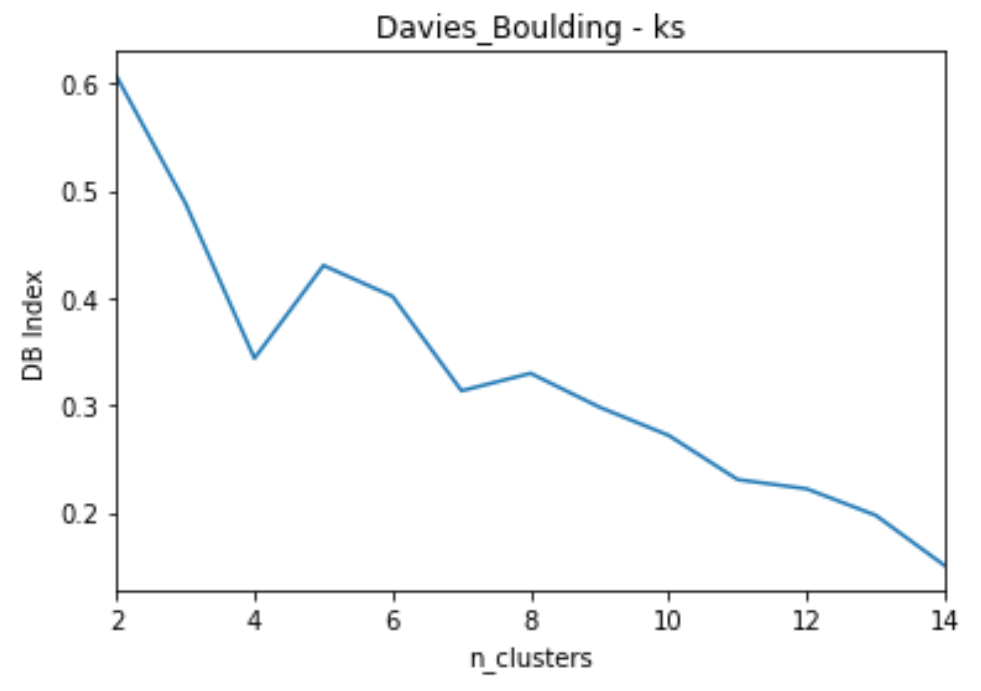
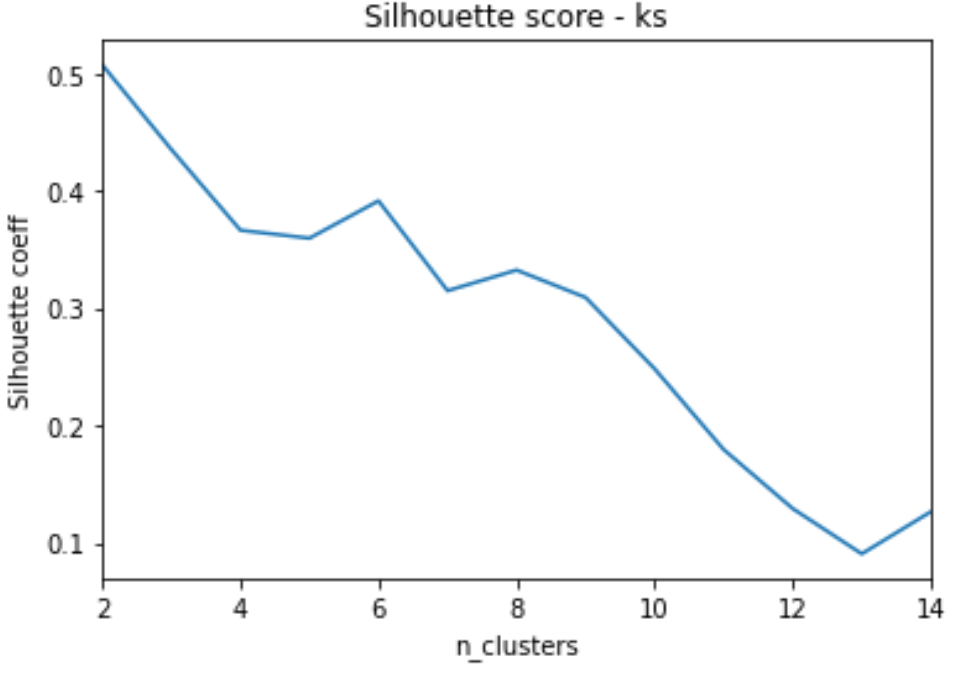




### N\_robots = 2

Silhouette score still suggests no clustering. DB score suggests 4 could be possible





### N\_robots = 3

Silhouette & DB scores suggest 4-5 clusters

