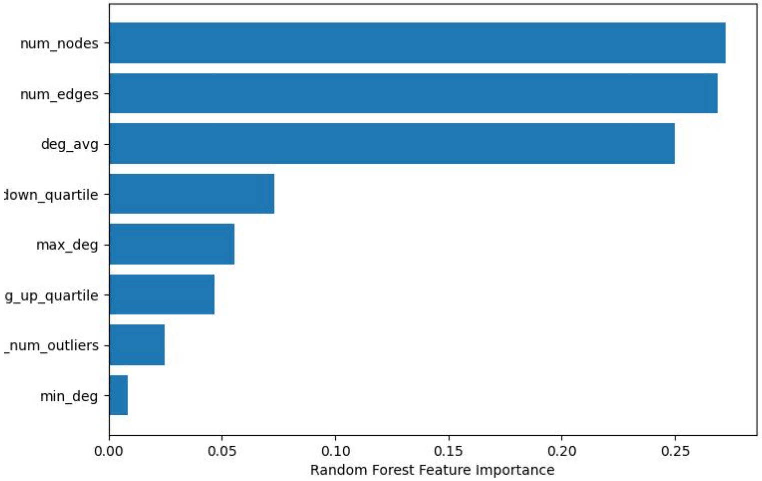
More Baseline Classifiers

- Isolation forest (100 trees, 0.1 contamination)
- Random forest (100 trees)
    - W. feature importance
- Decision tree classifier

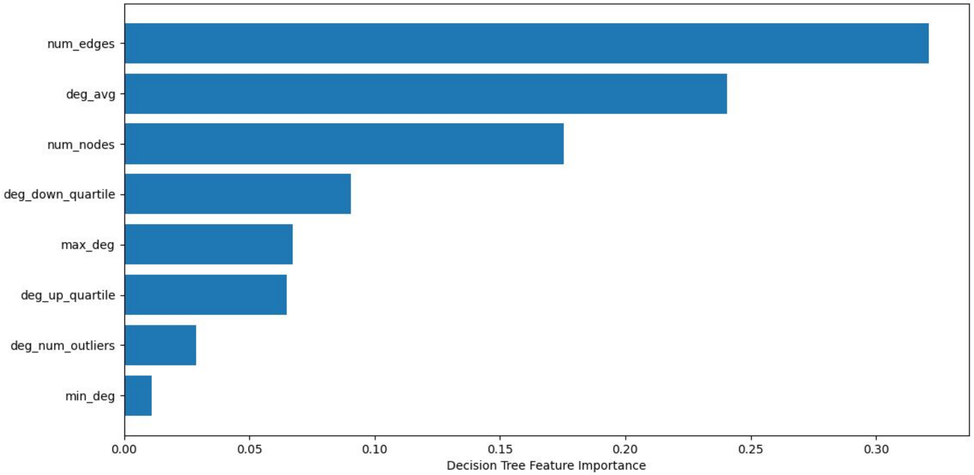
More Baseline Classifiers (cont.)

Datasets	GLADC	IF	RF	DT
MMP	**0.696 ± 0.042**	**0.462 ∓ 0.013**	**0.372 ∓ 0.005**	**0.423 ∓ 5.551**
HSE	**0.618 ± 0.110**	**0.423 ∓ 0.009**	**0.557 ∓ 0.005**	**0.512 ∓ 0.000**
p53	**0.649 ± 0.216**	**0.472 ∓ 0.008**	**0.502 ∓ 0.008**	**0.589 ∓ 0.000**
BZR	**0.715 ± 0.067**			
DHFR	**0.612 ± 0.041**			
COX2	**0.615 ± 0.044**			
ENZYMES	**0.583 ± 0.035**			
IMDB	**0.656 ± 0.023**			
AIDS	0.993 ± 0.005			
NCI1	0.683 ± 0.011	**0.486 ∓ 0.015**	**0.693 ∓ 0.016**	**0.663 ∓ 0.013**

Feature Importance (NCI1)

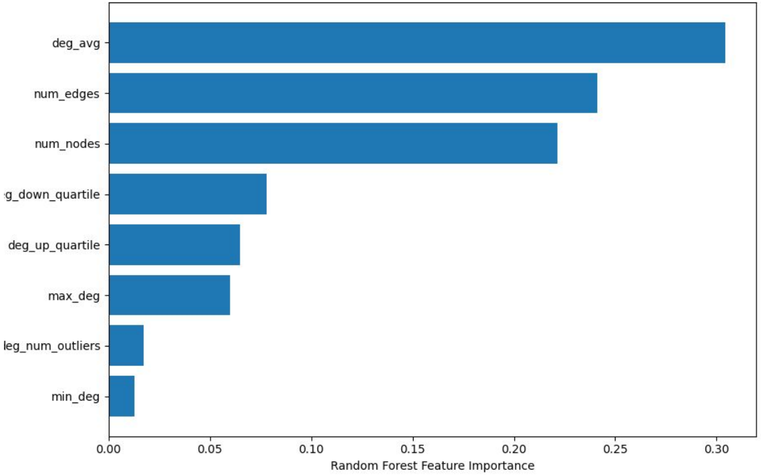


RF

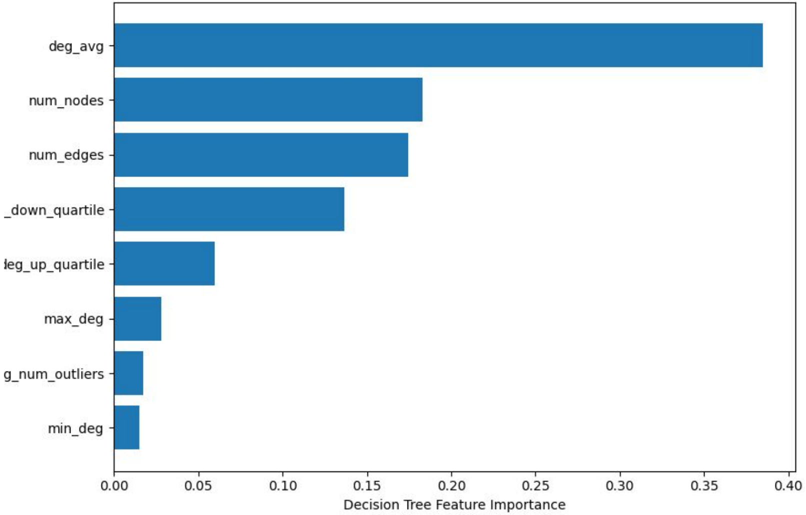


DT

Feature Importance (p53)

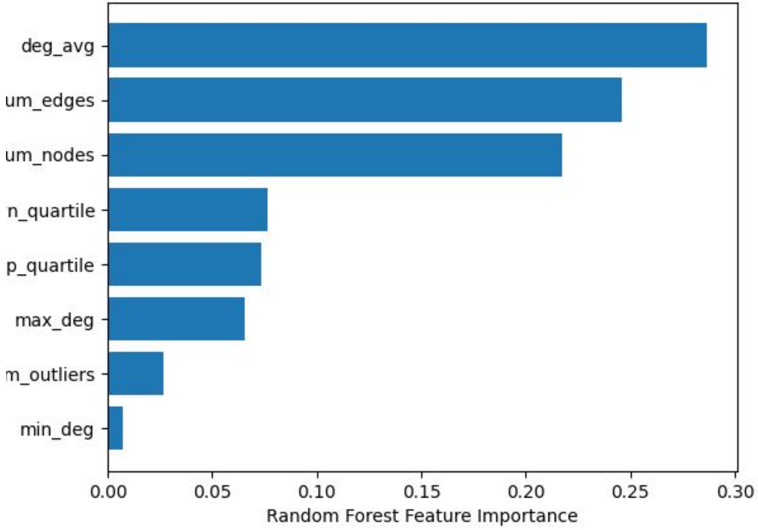


RF

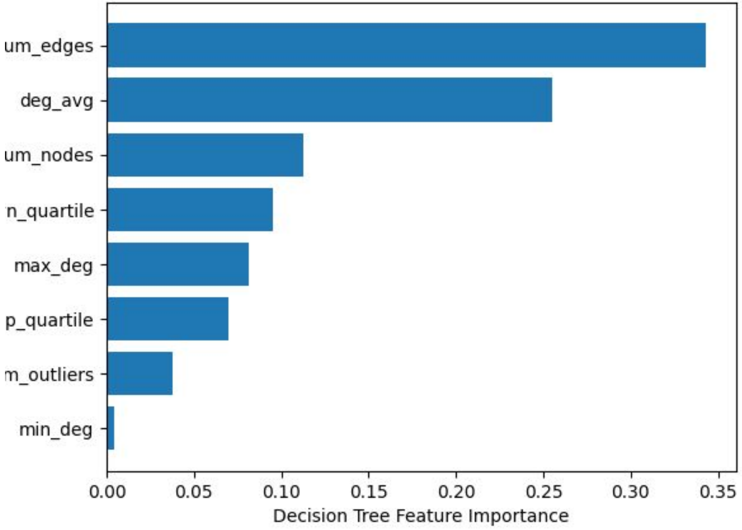


DT

Feature Importance (MMP)

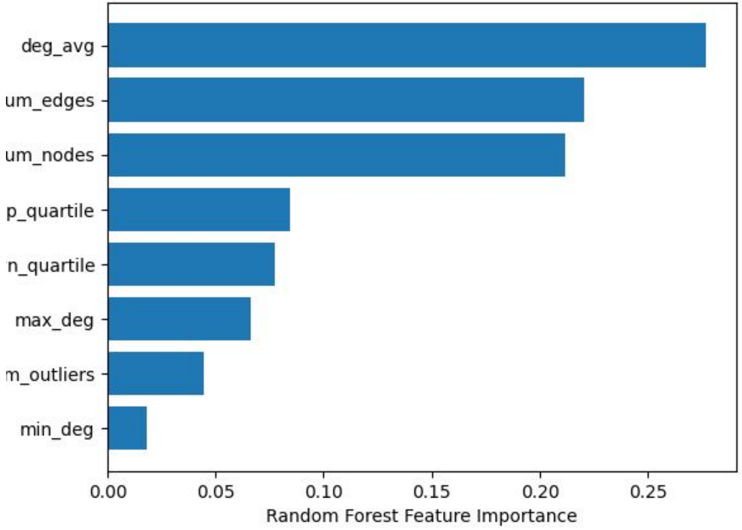


RF

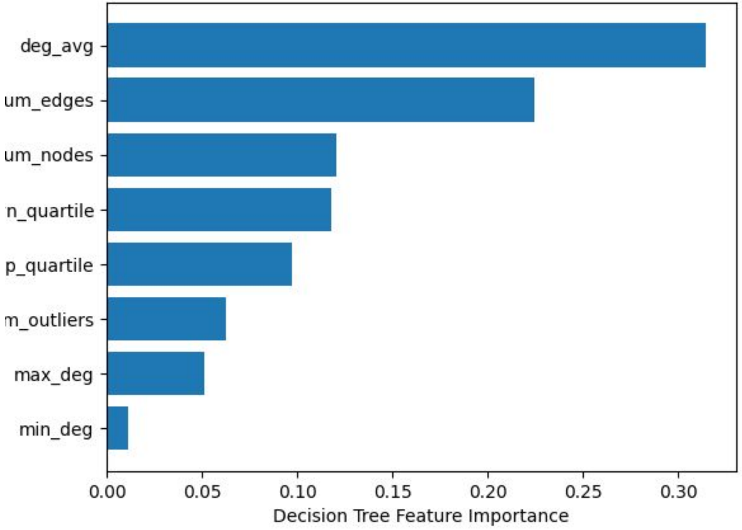


DT

Feature Importance (HSE)



RF



DT

GANs (Brainstorm)

- Existing implementations:
  - MolGAN: https://arxiv.org/abs/1805.11973
    - https://github.com/yongqyu/MolGAN-pytorch
  - GraphGAN: https://ojs.aaai.org/index.php/AAAI/article/view/11872
- Different alternatives:
  - Simply replace graph encoder module from GLADC with custom GAN module and keep the rest intact.
    - We could also avoid computing l2 and l3 losses (latent and contrastive) to keep things simple at first (since we determined that their presence is not significant to the results)
  - After the fake graph is generated, the graph anomaly detection module uses a simple formula to evaluate if the loss is exceptionally higher.
    - Can we use a ML approach for this last step (perhaps using another network).
    - We have a Discriminator from previous module that has learnt to distinguish between real and fake graphs, will that be of help?

GANs (Brainstorm)