Optimising Cylindrical Algebraic Decomposition using Machine Learning

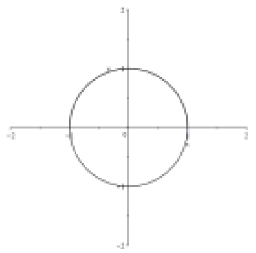
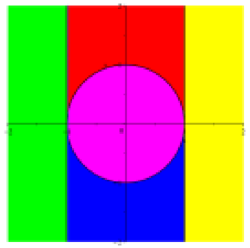
Rohit John & Prof. James Davenport
24/07/2024

Talk content

- Intro to CAD and variable ordering
- Current heuristics for optimising CAD
- Data and feature sets used
- Possible data leakage
- Implementation of ML heuristics
- Results of ML experiments
- Additional improvements & future work

What is Cylindrical Algebraic Decomposition?

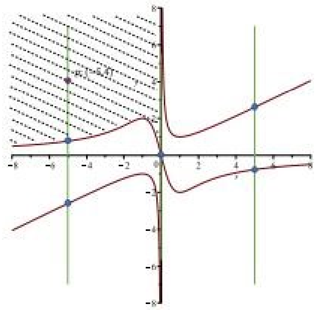
- Used for solving polynomial systems and quantifier elimination.
- Efficiently breaks down multi-dimensional spaces into simpler, sign invariant regions (cells) which can then be checked if the quantified formula holds.
- Applied in various scientific disciplines such epidemics and economics.

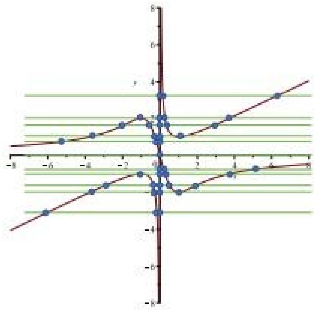


An example of CAD in use

Variable Ordering in CAD

- CAD’s doubly exponential complexity means it quickly becomes intractable as variable size increases.
- However this can be mitigated by the order in which variables are decomposed, which is decided by the mathematician

Example of the difference variable ordering can make for $x^3y + 4x^2 + xy, -x^2 + 2xy - 1$


(b) $x \prec y$


(c) $y \prec x$

Manual heuristics

- Various heuristics have been suggested for variable ordering such as Brown, sotd & ndrr.
- Browns heuristic is based off information derived from the polynomial and works as follows -

Eliminate variables in the following order:

1. it has lower overall degree in the input;

2. it has lower (maximum) total degree of those terms in the input in which it occurs;

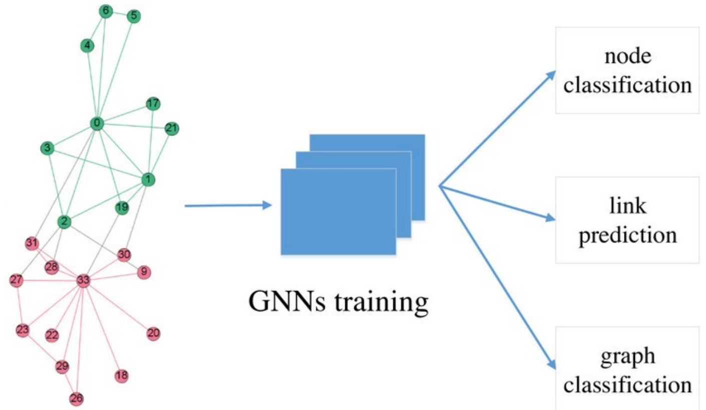
3. there is a smaller number of terms in the input which contain the variable

Machine Learning based heuristics

- Previous literature has explored using ML models to select heuristics and directly select orderings.
- The first implementation used SVM to determine which heuristic to use (Huang et al, 2014).
- Current work focuses on directly selecting optimal orderings.
- Recently Reinforcement Learning with Graph Neural Networks have been used to select orderings for CAD's of varied size (Jia et al, 2024).

Graph Neural Networks

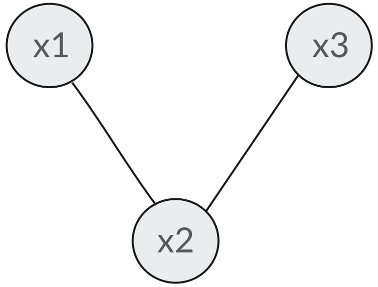
- While effective, general ML models only work for fixed polynomial sizes.
- GNN is a ML model which operates on graphs of varying sizes
- This representation can be used for graph, node and link classification, of which we implemented graph and node prediction.



Representing polynomials as graphs

- Each variable acted as a node, and an edge existed between two nodes if the corresponding variables appeared in the same polynomial.

Graph for the polynomial set $x_1^2+x_2^2+5$, x_2x_3-10



Aims for this project

- Gain an understanding of CAD and the heuristics used for variable ordering.
- Implement efficient and accurate ML models for selecting fixed variable orderings.
- Develop and implement a model which can select orderings for polynomials of different variable sizes.

Datasets

- We explored various datasets to use in our experiments, eventually deciding upon augmented-metitarski, based of the original MetiTarski theorem solver.
- The polynomials were then converted into graph form using the method described previously.
- The dataset had flaws in its labelling of lowest orderings, with 15% of entries having duplicate lowest times and cells,

Data Leakage Exploration

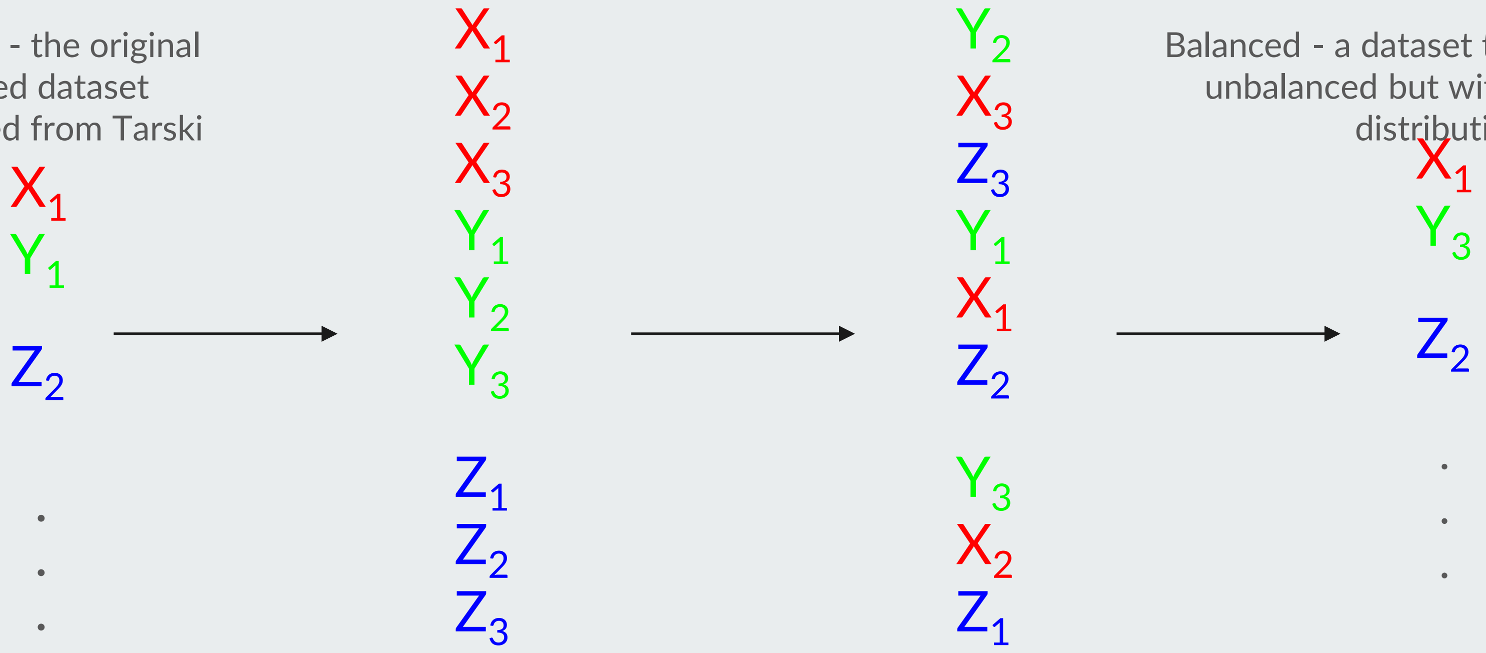
- The augmenting authors permuted the variables in each equation, leading to a perfectly balanced dataset.
- The data was then shuffled before training, leading to permutations of equations appearing in both sets and significant data pollution occuring.
- We experimented with 4 variants of this dataset which we will call “**original**”, “**augmented**”, “**shuffled**” and “**balanced**”.

Original - the original biased dataset extracted from Tarski

Augmented - an expanded dataset where every polynomial permutation is included.

Shuffled - the currently existing dataset used in augmented-metitarski.

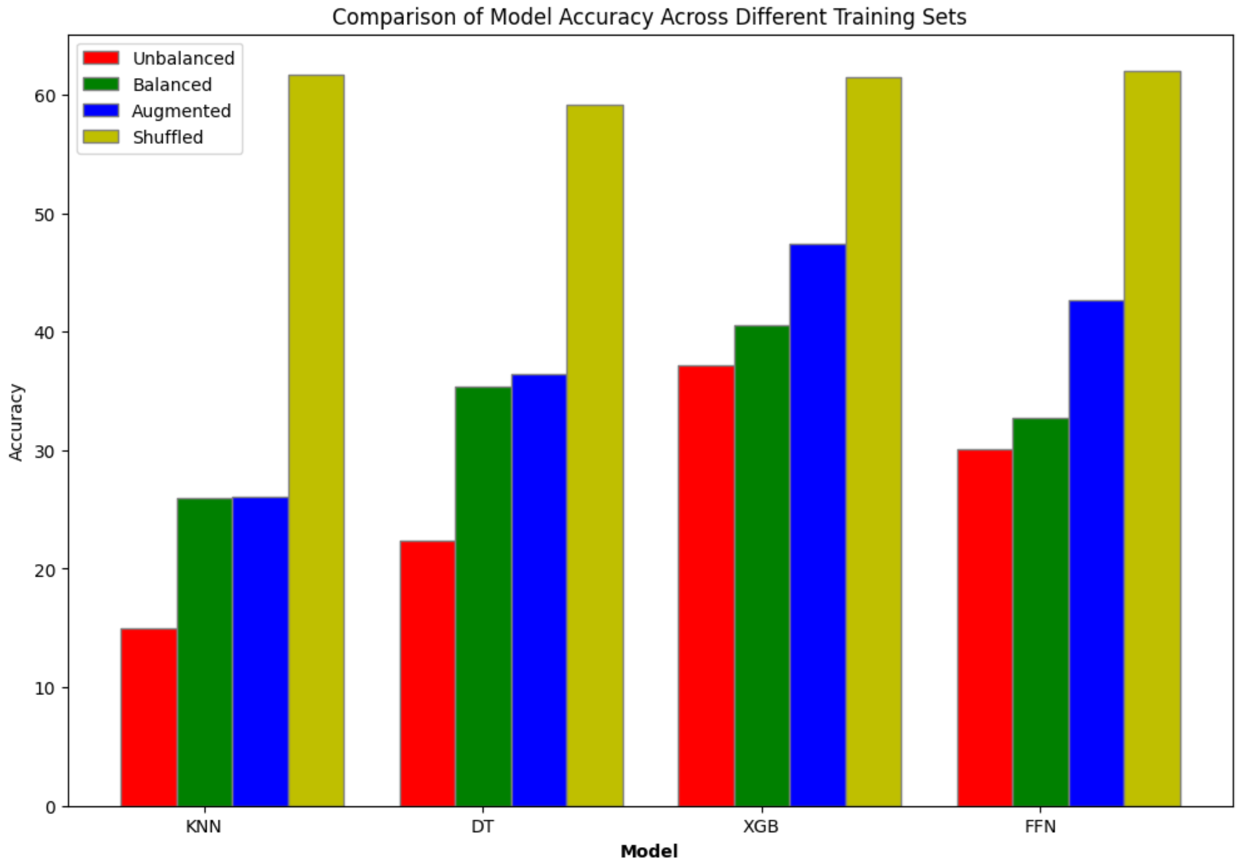
Balanced - a dataset the same size as unbalanced but with equal label distribution.



X_1 Y_1 Z_2 . . .
X_1 X_2 X_3 Y_1 Y_2 Y_3 Z_1 Z_2 Z_3
Y_2 X_3 Z_3 Y_1 X_1 Z_2 Y_3 X_2 Z_1
X_1 Y_3 Z_2 . . .

X_1 - X represents a polynomial set, 1 represents ordering x1 -> x2 ->x3 being the optimal variable ordering.

Data leakage results



Original - the original biased dataset extracted from Tarski

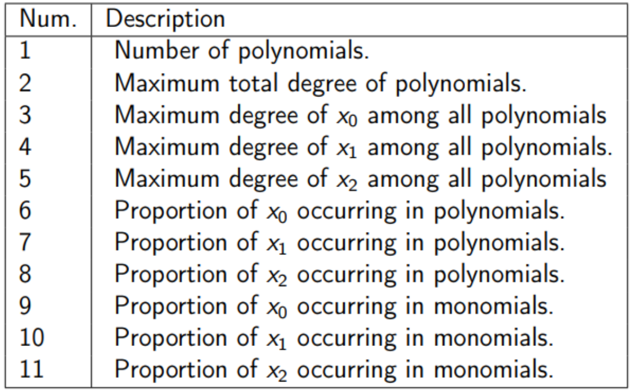
Balanced - a dataset the same size as unbalanced but with equal label distribution.

Augmented - an expanded dataset where every polynomial permutation is included.

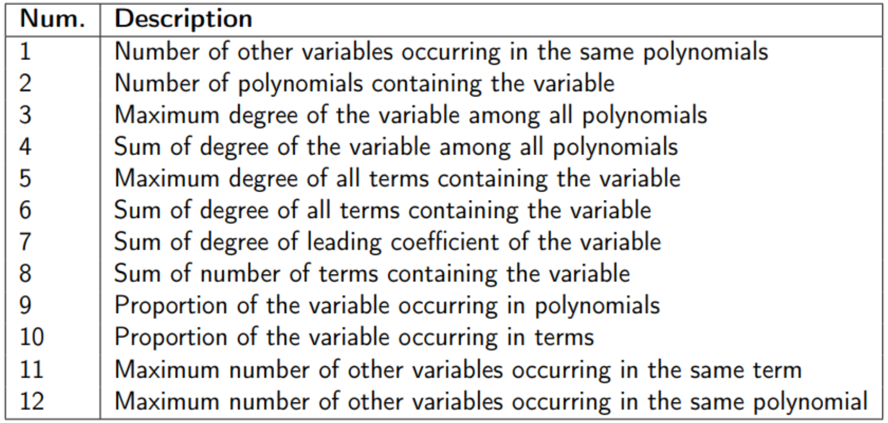
Shuffled - the currently existing dataset used in augmented-metitarski.

Feature Sets

- The initial set consists of 11 features and has been used in various experiments, starting with Huang.
- GNN requires a feature set for each variable which was implemented using Jia’s feature set.



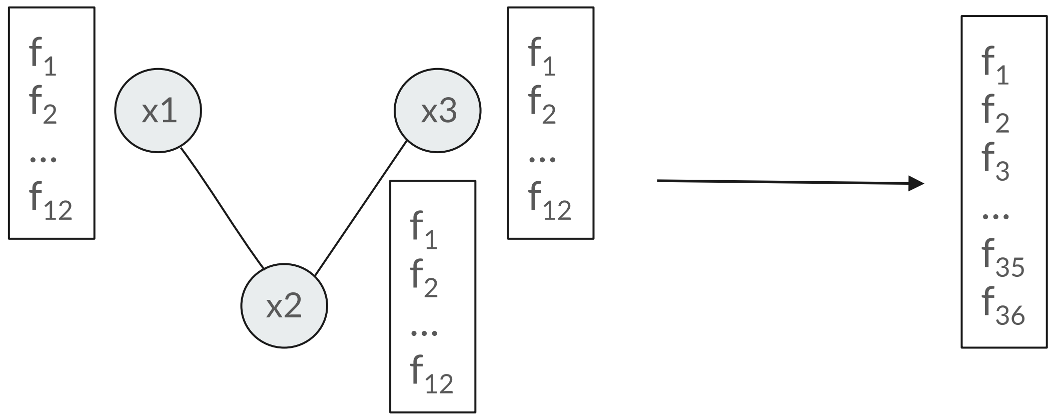
Num.	Description
1	Number of polynomials.
2	Maximum total degree of polynomials.
3	Maximum degree of x_0 among all polynomials
4	Maximum degree of x_1 among all polynomials.
5	Maximum degree of x_2 among all polynomials
6	Proportion of x_0 occurring in polynomials.
7	Proportion of x_1 occurring in polynomials.
8	Proportion of x_2 occurring in polynomials.
9	Proportion of x_0 occurring in monomials.
10	Proportion of x_1 occurring in monomials.
11	Proportion of x_2 occurring in monomials.



Num.	**Description**
1	Number of other variables occurring in the same polynomials
2	Number of polynomials containing the variable
3	Maximum degree of the variable among all polynomials
4	Sum of degree of the variable among all polynomials
5	Maximum degree of all terms containing the variable
6	Sum of degree of all terms containing the variable
7	Sum of degree of leading coefficient of the variable
8	Sum of number of terms containing the variable
9	Proportion of the variable occurring in polynomials
10	Proportion of the variable occurring in terms
11	Maximum number of other variables occurring in the same term
12	Maximum number of other variables occurring in the same polynomial

Extended feature set

- We additionally experimented with an extended feature set to compare with the original.
- This was generated by merging the 12 features of each variable in our GNN feature set to create 36 total features.

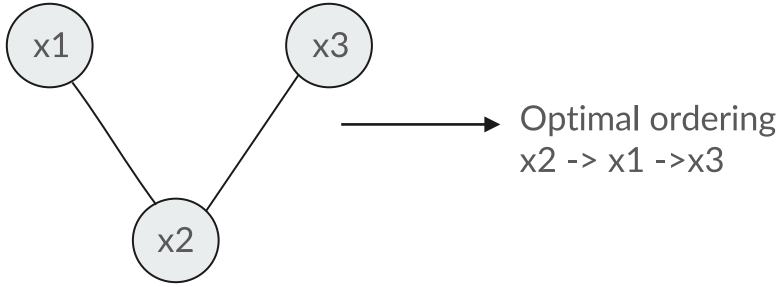


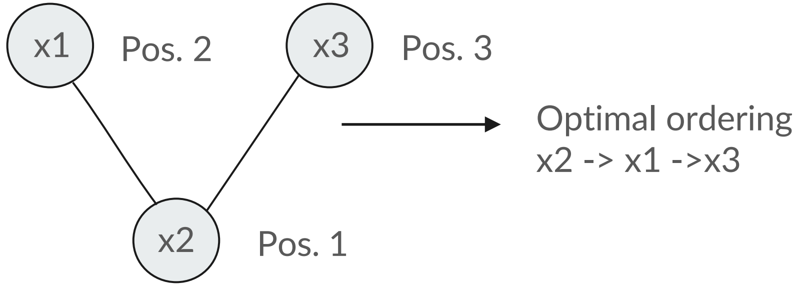
Implementing ML models

- Implemented common ML models (LR, KNN, DT etc.)
- SVM were initially implemented but were later removed due to long training times.
- Classification and Regression FFN models were also implemented.

GNN implementation

- GNN implementation began by classifying each graph into one of 6 possible orderings, however this model did not greatly utilise the graph representation.
- The model was then trained to assign a position in the orderings to each node in the graph.

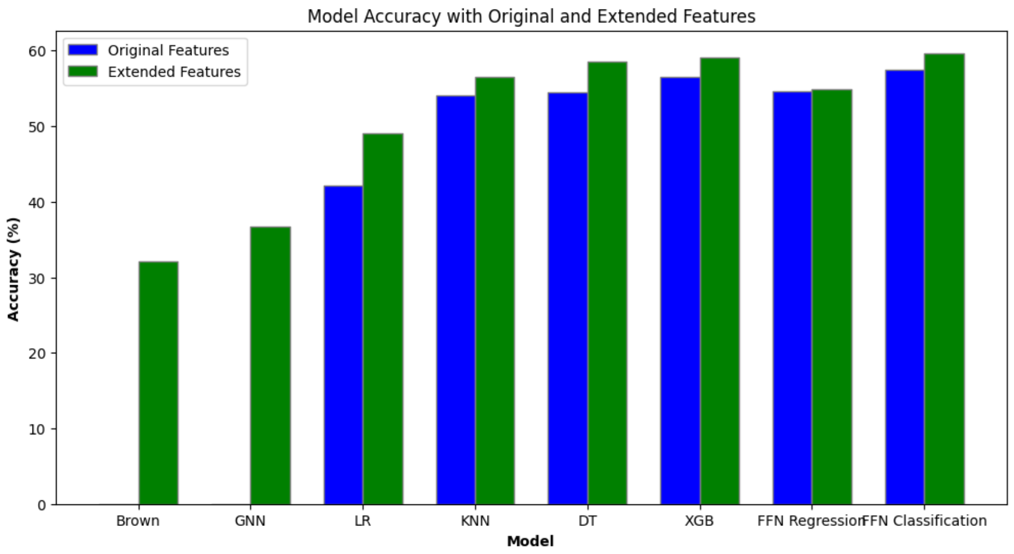




ML Results using accuracy

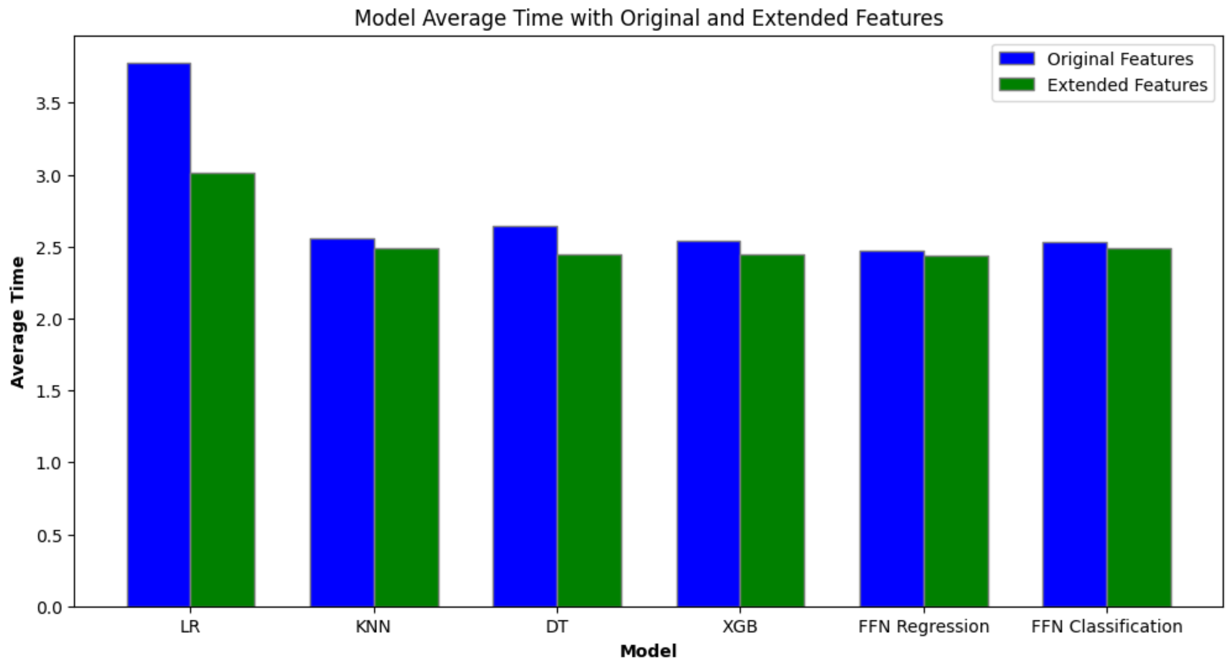
- FFN classification was the most effective.
- Using extended features has a small but consistent advantage.
- GNN were less effective than other models, but more effective than Brown.
- When measuring accuracy Regression performed worse than expected.

Measuring accuracy

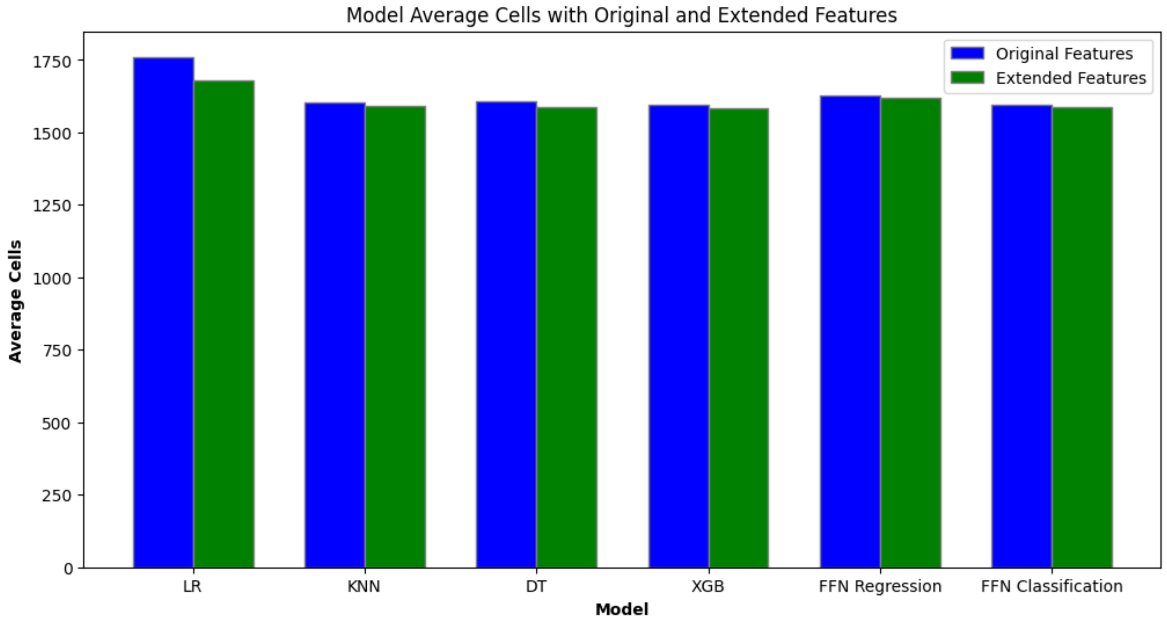


Results using average time

- Linear regression performs poorly but is significantly improved by using extended features.
- Regression performs the best when using average time metrics.



Results using average cells



What I’ve learnt

- Datasets matter - I initially chose my dataset based off recency and without thorough checking which might have prevented the described problems.
- Data leakage and best practices in ML - sometimes common practices in data science (shuffling datasets before splitting) can lead to misleading results and should not be applied blindly.
- Metrics are difficult to compare - our results can be interpreted differently when certain metrics are used, and so metrics should always take into account

Final notes

- The updated dataset and code used for these experiments can be found at https://github.com/rohitpj/New_ML_Models_for_CAD
- We used additional metrics of average time and average cell count in our experiments, details of which can be found in my dissertation available above.
- However more complex polynomials can greatly outweigh others so we suggest using normalised time differences between lowest and suggested ordering.