CENTRO DE ENGENHARIAS E CIÊNCIAS EXATAS

CAMPUS DE FOZ DO IGUAÇU

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E COMPUTAÇÃO

Exame de Qualificação de Mestrado

Isabella Caroline Sachini Lorena

Meta-aprendizado aplicado à Otimização de Algoritmos de Quantificação

Orientador: Prof. Dr. Willian Zalewski

Foz do Iguaçu - Paraná

2025

1 Introdução

A quantificação é uma tarefa supervisionada que tem como objetivo predizer a distribuição das classes em uma amostra de dados. A quantificação possui diversas aplicações como, por exemplo, contagem de células não-viáveis em uma amostra (González et al., 2017; Alaiz-Rodríguez et al. 2008), análise de sentimentos em redes sociais (González et al., 2017; Giachanou and Crestani 2016) e contagem de mosquitos capturados (Maletzke 2019; Silva et al., 2015). Diversos métodos de quantificação foram propostos pela comunidade científica nos últimos anos (Schumacher et al., 2021). Esses métodos foram desenvolvidos com o objetivo de tratar adequadamente diferentes tipos de mudança nas distribuições nos conjuntos de dados de treinamento e de teste (González et al., 2024). Contudo, apesar dos avanços, ainda não existe um método de quantificação capaz de superar consistentemente todos os demais em diferentes cenários (Gomes et al., 2024). Estudos indicaram que a escolha do quantificador mais adequado depende de características intrínsecas de cada conjunto de dados, como o tamanho da amostra, a composição e a prevalência das classes (Maletzke et al., 2021). Diante dessa limitação, algumas pesquisas têm explorado o uso de abordagens baseadas em *meta-learning* para construir sistemas de recomendação de métodos de quantificação. O meta-aprendizado baseia-se na extração de conhecimento do desempenho de modelos em tarefas anteriores, com o intuito de identificar quais algoritmos têm maior probabilidade de obter bons resultados em novos conjuntos de dados (Faceli et al., 2011; Brazdil et al., 2009). Nesse contexto, Gomes et al. (2024) desenvolveram um sistema de recomendação capaz de predizer os quantificadores mais apropriados para um determinado conjunto de dados. De forma semelhante, Maletzke et al. (2021) propuseram o *framework Meta-Learning Quantification* (MLQ), para a recomendação de algoritmos de quantificação para cada amostra de dados. Entretanto, apesar dos esforços, temos que os métodos de quantificação ainda possuem algumas limitações. A avaliação experimental conduzida por González et al. (2023) evidenciou que os métodos de quantificação apresentam variabilidade no erro de suas estimativas quando expostos a amostras com diferentes prevalências, refletindo mudanças no comportamento desses métodos diante de distintas distribuições dos dados. Contudo, a literatura atual não apresenta métodos capazes de ajustar ou controlar o erro de predição dos quantificadores. Assim, considera-se que abordagens de *meta-learning* podem aprimorar o desempenho dos quantificadores ao modelar e corrigir seu comportamento sob diferentes cenários de distribuição.

Hipótese: O erro de quantificação pode ser reduzido quando as estimativas de prevalência são ajustadas de acordo com as características de desempenho de cada quantificador.

Objetivo geral: Utilizar estratégias de meta-aprendizado ajustar/corrigir as predições de quantificadores.

Como objetivos específicos, temos:

- Caracterizar os erros dos quantificadores em diferentes cenários de distribuição;
- Selecionar e construir meta-*features* capazes de capturar padrões sistemáticos de erro;
- Desenvolver um modelo de meta-aprendizado para ajustar as predições dos quantificadores;
- Avaliar experimentalmente o desempenho e a robustez da abordagem.

2 Quantificação

A quantificação é uma tarefa de aprendizado supervisionado (Forman, 2005) semelhante à classificação. Ambas as tarefas recebem de entrada um conjunto de instâncias rotuladas durante o treinamento (Maletzke et al. 2021). No entanto, um quantificador não tem o propósito de predizer a classe de cada instância, mas sim estimar a proporção de cada classe em uma amostra dos dados (*batch*) (González et al., 2017).

Formalmente, temos um conjunto de dados rotulado $D = \{(\overrightarrow{x_1}, y_1), \dots, (\overrightarrow{x_n}, y_n)\}$, onde cada exemplo $\overrightarrow{x_i} \in X$, é um vetor do espaço de características m-dimensional X, e $y_i \in C = \{c_1, \dots, c_k\}$, é o rótulo contendo a classe.

Um quantificador pode ser definido como um modelo que estima a prevalência de cada classe em uma amostra, conforme definido com a equação 1 abaixo:

$$Q : A^X \rightarrow [0,1]^k \quad (1)$$

Onde A^X, representa as possíveis amostras de X. Dada uma amostra $A \in A^X$, o modelo produz um vetor $\hat{\vec{q}}(A) = [\hat{p}_1(A), \dots \hat{p}_k(A)]$, que prediz a probabilidade para cada classe.

Em uma análise superficial, a tarefa de quantificação poderia ser entendida como trivial, e que, portanto, poderia ser solucionada utilizando uma abordagem simples, como o método popular *Classify & Count*. Esse método utiliza um classificador para predição das instâncias e, posteriormente conta a distribuição dessas para cada classe. Os algoritmos de classificação baseiam-se na suposição

de que os dados de treinamento e de teste são amostras independentes e identicamente distribuídas, provenientes da mesma distribuição subjacente. Contudo, na prática, essas condições nem sempre são atendidas, o que pode comprometer o desempenho e a confiabilidade dos modelos quando aplicados a cenários reais. Desse modo, abordagens de quantificação precisam ser capazes de tratar eventuais mudanças na distribuição dos dados. De acordo com (González et al., 2023), as principais mudanças que ocorrem nos conjuntos de dados são i) *Prior probability shift:* Mudanças nas distribuições das classes entre as etapas de treinamento e teste; ii) *covariate shift*: Mudanças na distribuição das covariáveis (atributos); iii) *concept shift*: Mudanças entre a relação entre os atributos e as classes.

Nesse contexto, diferentes métodos de quantificação têm sido propostos na literatura nos últimos anos (González et al. (2017). A maioria desses métodos utilizam classificadores para produzir estimativas de confiança (*scores*) das predições de uma amostra como uma etapa intermediária para a quantificação. González et al. (2017) apresenta uma taxonomia organizada em três diferentes grupos de algoritmos:

- **Classifica, Conta e Corrige**. métodos que apresentam como base a classificação das instâncias em uma amostra, para estimar através de contagem a prevalência de cada classe. Esse grupo, também inclui métodos que aplicam fatores de correção a contagem. Exemplo, Adjusted *Classify and Count* - ACC (Forman, 2005), *Probabilistic Classify and Count* - PCC (Bella et al., 2010), *Probabilistic Adjusted Classify and Count* - PACC (Bella et al., 2010);
- **Métodos tradicionais de classificação adaptados para quantificação**: Esses métodos são semelhantes aos métodos que classificam, contam e corrigem. Porém, são sutilmente diferentes, pois realizam uma adaptação na estrutura dos classificadores para que se tornem quantificadores. Como, por exemplo, o método *Quantification Trees* - (Milli et al., 2013);
- **Correspondência de distribuição**: Métodos que geram um modelo paramétrico de distribuição dos dados de treino, e posteriormente realizam a busca pelos parâmetros que melhor correspondam às distribuições dos dados de teste. Alguns dos principais métodos pertencentes a esse grupo são *Hellinger Distance Minimization* - HDy

(González et al., 2013), *Distribuition y-Similarity* - DyS (Maletzke et al., 2019), *Sample Mean Matching* - SMM (Hassan et al., 2020).

Apesar da diversidade de métodos de quantificação presentes na literatura, A maioria desses métodos utiliza como etapa intermediária os *scores* provenientes de um classificador binário para representar a confiança em uma classificação positiva. No entanto, no estudo proposto por Tasche (2019) foi demonstrado que a qualidade dos *scores* tem influência significativa nos intervalos de confiança produzidos pelos métodos de quantificação (Maletzke et al., 2019).

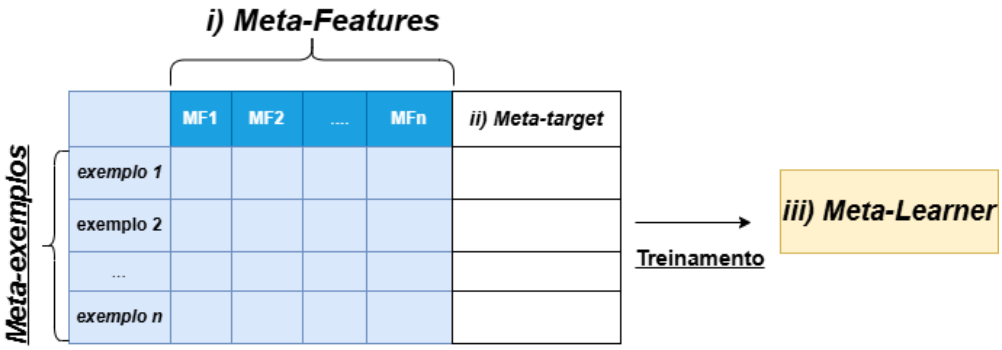
Nesse sentido, temos que estes métodos podem sofrer mudança em seu desempenho de acordo com as seguintes constatações: i) Métodos de Quantificação sofrem mudança em seu desempenho ao variar tamanho e composição dos *batches* de teste (Maletzke et al., 2021); ii) Nenhum método de quantificação é robusto suficiente para lidar com diferentes mudanças de distribuições dos dados de treino e de teste (González et al., 2023). iii) A acurácia de um quantificador possui relação com a qualidade dos *scores* do classificador. Desse modo, temos que a estimativa de um quantificador é dependente das características de cada conjunto de dados, um quantificador não é suficientemente robusto para lidar com os diferentes cenários de distribuições

3 Meta-Aprendizado

O Meta aprendizado (*Meta-Learning*) é uma abordagem que possibilita a seleção e/ou otimização automática de algoritmos por meio da extração de propriedades a partir dos dados e da aplicação de aprendizado de máquina. Trata-se de uma abordagem que utiliza de experiências anteriores para aprender com o próprio processo de aprendizagem (Rivolli et al., 2022). De modo geral, a Meta aprendizagem (MTL), busca associar elementos de tarefas de aprendizagem ao desempenho de algoritmos.

De modo geral, o processo de MTL, pode ser entendido por meio de três dimensões, conforme ilustrado na Figura 1 Primeiramente, recebe como entrada um conjunto de meta-exemplos, que são obtidos a partir de um conjunto de problemas de aprendizagem e um conjunto de algoritmos candidatos. Esses meta-exemplos, guardam as características que descrevem um problema, ou seja, meta-características (i - *Meta-features*), juntamente com um atributo alvo (ii - *Meta-target*) como, por exemplo, o melhor algoritmo candidato para determinado problema. Ao final do processo, um

meta-modelo (iii - *Meta-Learner*) é treinado para predizer o melhor algoritmo para novos problemas, tendo como base as meta-características.



Fonte: A autora (2025).

Figura 1: Dimensões de *Meta-learning*

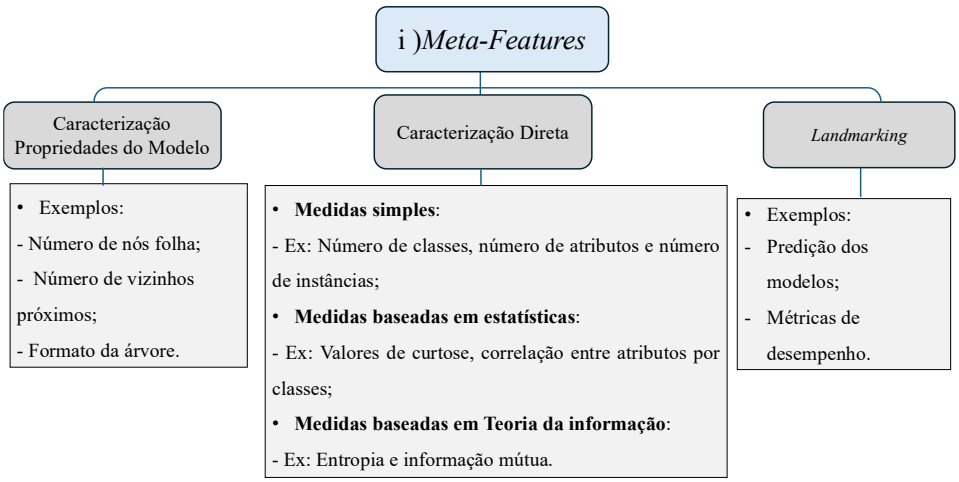
Faceli et al. (2011) apresenta uma organização para as dimensões das *Meta-Features* (Figura 2) e *Meta-Target*:

As *Meta-target* podem ser apresentadas de três formas:

i) Melhor algoritmo, de acordo com alguma medida de avaliação;

ii) Desempenho do algoritmo;

iii) Ranking dos melhores algoritmos.



Fonte: A autora (2025).

Figura 2: Grupos e classes das *Meta-Features*.

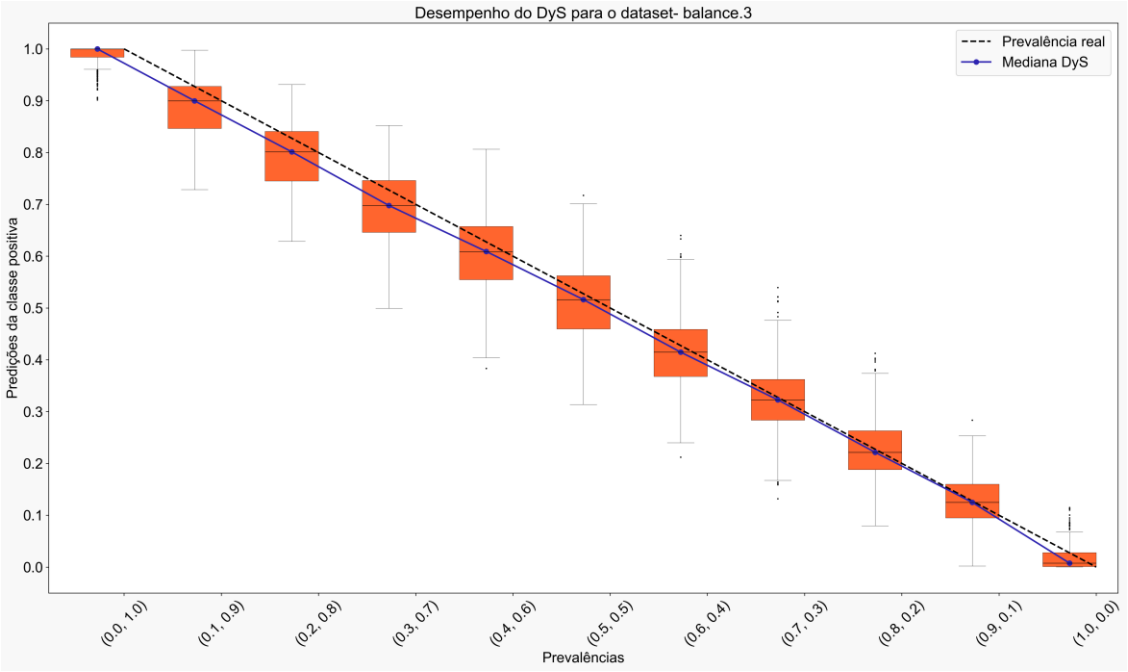
Tarefas de MTL podem apresentar variações de acordo com o objetivo da tarefa que se pretende melhorar, as quais incluem: i) Recomendar algoritmos candidatos para determinado conjunto de dados; ii) Prever a métrica de desempenho de um único método; iii) Recomendar valores para os hiperparâmetros de um algoritmo com base nas características do problema (Lorena et al., 2017).

4 Proposta

Sabemos que a quantificação binária apresenta alguns desafios, entre eles i) Nenhuma abordagem atual de quantificação é suficientemente robusta para lidar com diferentes mudanças presentes nos conjuntos de dados; ii) a qualidade dos *scores* provenientes de um classificador afeta o desempenho de um quantificador; iii) o desempenho dos quantificadores muda conforme a composição, tamanho e prevalência das classes presentes no *batch* de teste.

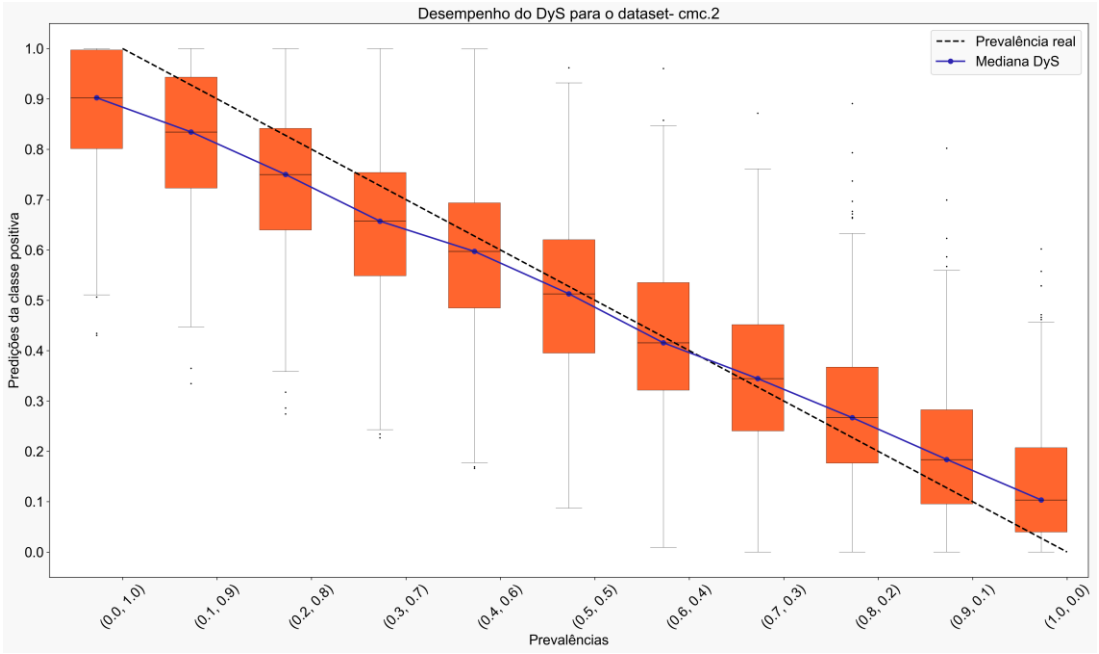
Neste contexto, para melhor compreender a variabilidade das predições de um quantificador em função das mudanças presentes no conjunto de dados, neste trabalho, foi realizado um estudo experimental prévio, utilizando quantificação binária. Para isso foi selecionado o método *Distribution y-Similarity* (DyS) proposto por Maletzke et al., 2019, uma vez que é um *benchmark* dos quantificadores binários. Nos experimentos realizados. foi adotado o protocolo APP (*Artificial Prevalence Protocol*) para geração de *batches.* Onde foram gerados 100 *batches*, cada um com 100 instâncias, para cada uma das 11 prevalências do conjunto: {0.0, 0.1, 0.2, …, 0.9, 1.0}, totalizando 1100 *batches*.

Por meio dos resultados desse estudo exploratório foi possível verificar que o DyS apresentou uma notável variabilidade nas predições considerando as prevalências avaliadas. Como exemplo, nas Figuras 3 e 4 são ilustradas essas variabilidades de predição para dois conjuntos de dados analisados: balance.3 e cmc.2. Pela análise desses resultados foi possível constatar um deslocamento da mediana das predições em relação aos valores reais de cada prevalência.



Fonte: A autora (2025).

Figura 3: Desempenho do DyS para o *dataset* Balance.3

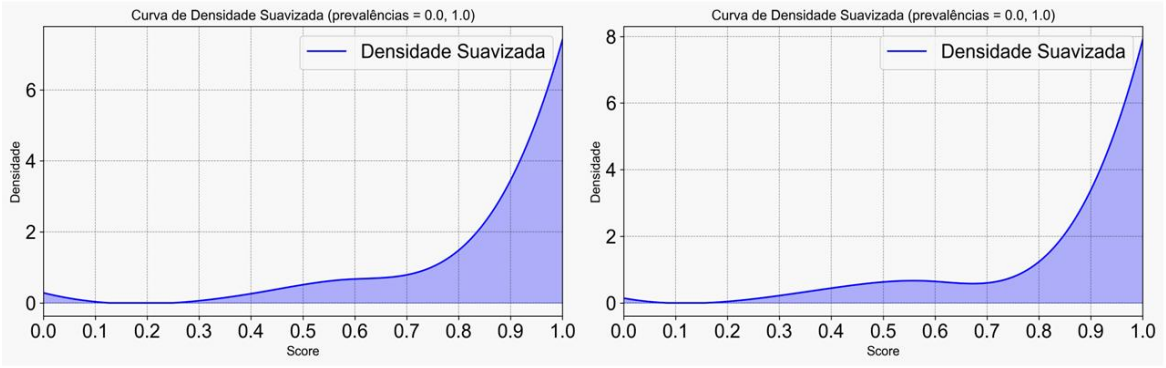


Fonte: A autora (2025).

Figura 4: Desempenho do DyS para o *dataset* cmc.2.

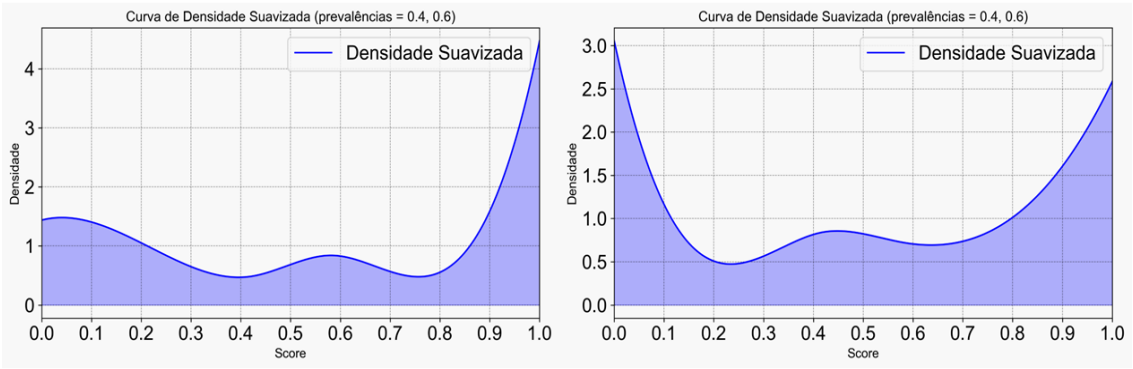
Com o intuito de compreender a variabilidade das predições do quantificador DyS nos cenários mencionados, foi realizada uma análise das distribuições dos *scores* presentes em diferentes *batches*

com as mesmas prevalências A partir dessa análise, foi possível constatar diferenças na distribuição dos *scores* mesmo em *batches* de igual prevalência. Como exemplo, na Figura 5 são ilustradas duas distribuições de *scores* semelhantes na mesma prevalência; por outro lado, na Figura 6 é ilustrado um cenário no qual as distribuições de *scores* são significativamente distintas.



Fonte: A autora (2025).

Figura 5 :Curvas de densidade dos *scores* para *batches* de p{+1} = 1.0 do *Dataset* Balance.3.



Fonte: A autora (2025).

Figura 6: Curvas de densidade dos *scores* para *batches* de p{+1} = 0.6 do *Dataset* Balance.3

Considerando os cenários avaliados neste estudo exploratório, observamos que quantificadores podem apresentar diferentes níveis de variabilidade nas predições de distintas prevalências e um deslocamento da mediana das predições em relação aos valores reais de cada prevalência. Além disso, verificou-se a existência de batches com diferentes distribuições dos scores nas mesmas prevalências.

Conforme visto em (Maletzke et al., 2021), abordagens de *meta-learning* possibilitam encontrar padrões entre diferentes *batches*. Nesse sentido, argumentamos que algoritmos de quantificação

podem ter sua robustez ajustada utilizando meta-conhecimento. Neste trabalho, propomos explorar meta-aprendizagem possa auxiliar na otimização de algoritmos de quantificação.

4.1 Arquitetura Geral do Método proposto

Para atingir os objetivos apresentados nesta proposta, é apresentado um método que pode ser estruturado em 5 etapas principais, conforme ilustrado na Figura 7 (a):

Etapa 1 - Divisão do conjunto de dados: nesta etapa o conjunto de treinamento é particionado em conjunto de treino (30% dos dados), para treinar o quantificador, e conjunto de validação (70% dos dados);

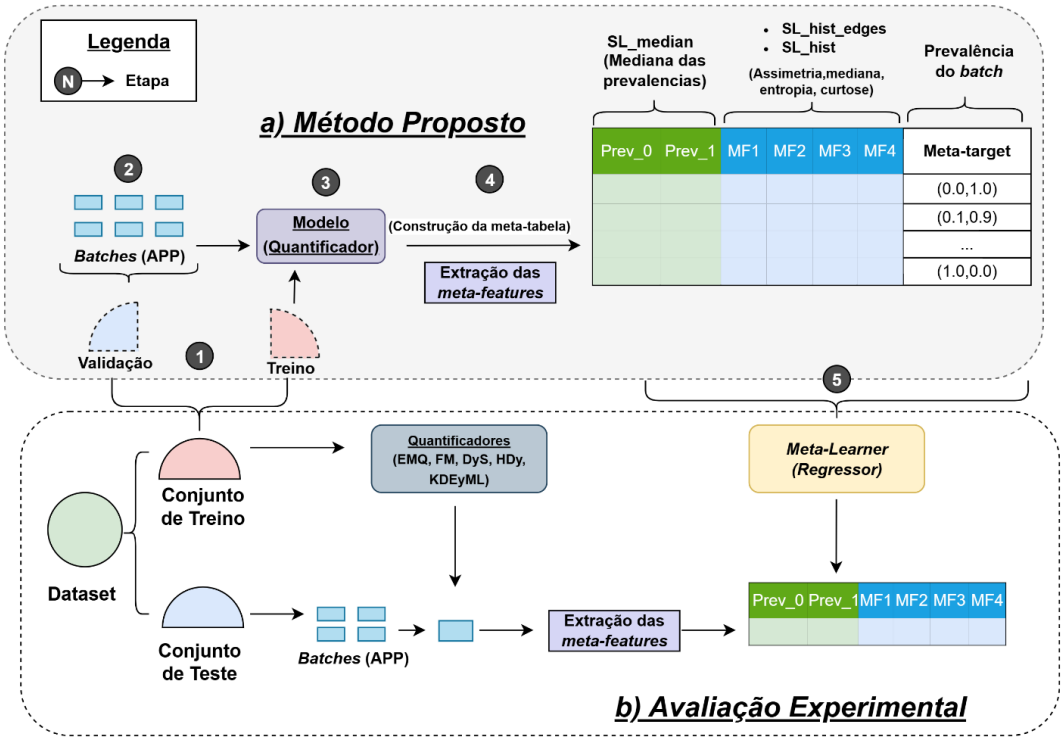
Etapa 2 - Geração de *batches*: usando os dados do conjunto de validação são gerados *batches* para um conjunto de prevalências. Neste estudo foram gerados 100 *batches*, cada um com 100 instâncias, para cada uma das 11 prevalências do conjunto: {0.0, 0.1, 0.2, …, 0.9, 1.0}, totalizando 1100 *batches*.

Etapa 3 - Treinamento do quantificador: nesta etapa um quantificador é treinado utilizando o conjunto de treino particionado na Etapa 1. Neste estudo utilizamos o quantificador DyS com parâmetros *default* da biblioteca *MlQuantify*[1] e com o algoritmo *RandomForest* como classificador base.

Etapa 4 - Extração de meta-*features*: nesta etapa são extraídos atributos para caracterizar as predições do quantificador e dos *batches* gerados. Para este estudo são propostas três abordagens para extração de meta-features: *SL median, SL_hist_edges e SL_hist.* Ao final desta etapa, é produzida uma meta-tabela, a qual descreve as meta-*features* de cada *batch* em função da meta-*target*, que é a prevalência do *batch*.

Etapa 5 - Construção do meta-*learner*: nesta etapa, a meta-tabela é utilizada para a indução de um do meta-*learner*. Neste estudo foi utilizado o algoritmo *Random Forest Regressor* com parâmetros *default* da biblioteca *scikit-learn.*[2]

[1] https://pypi.org/project/mlquantify/
[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html



Fonte: A autora (2025).

Figura 7:Arquitetura geral do método

Como mencionado, neste trabalho, foram propostas três abordagens para a extração de meta-*features*, os quais são brevemente descritos a seguir:

***SL median* (SL_A):** A ideia dessa abordagem baseia-se na suposição de que o desempenho do quantificador pode ser ajustado ao utilizar a informação da mediana das predições de diferentes *batches*. Conforme observado nas Figuras 3 e 4, a mediana das predições de cada prevalência apresenta um deslocamento em relação ao desempenho ideal do quantificador. Desse modo, nessa estratégia, cada uma das prevalências geradas na Etapa 2 é considerada como um meta-exemplo e a mediana das predições do quantificador (Etapa 3) é utilizada como meta-feature. Com o intuito de obter valores mais representativos da variabilidade das predições, o processo das Etapas 1, 2 e 3 é repetido cinco vezes, totalizando 500 predições para cada prevalência.

***SL_hist_edges* (SL_HE) *e SL_hist* (SL_H):** A proposição dessas duas abordagens baseia-se na ideia de que o desempenho do quantificador pode ser ajustado ao utilizar as informações das predições em diferentes *batches*, juntamente com características extraídas a partir da distribuição de densidade dos *scores*. Essa suposição é baseada na observação de que *batches* de mesma prevalência

podem apresentar variação significativa na distribuição dos *scores*, conforme ilustrado nas Figuras 5 e 6. Nestas propostas, buscamos fornecer informações que possibilitem a diferenciação dos *batches* com prevalências semelhantes, mas com distribuições internas diferentes. O processo de ambas as estratégias consiste em gerar histogramas para representar a distribuição de *scores* de cada *batch* gerado na Etapa 2. Na abordagem *SL_hist,* os histogramas têm um número fixo de intervalos (*n_bin*s = 30) para todo conjunto de dados avaliado. Na abordagem *SL_hist_edge* é utilizado o método *Freedman Diaconis Estimator*[3] para determinar o número de intervalos do histograma em cada conjunto de dados. A partir dos histogramas gerados, foram extraídas quatro *Meta-features* estatísticas: Assimetria, mediana, entropia e curtose. A utilização desses descritores fundamenta-se pela suposição de que essas informações possam auxiliar a diferenciar *batches* com a mesma prevalência, mas com distribuições de *scores* diferentes. Como resultado de ambas as estratégias, as meta-*features* estatísticas são concatenadas com a predição do quantificador (prevalências estimadas do Dys + 4 Meta-*Features*).

4.2 Avaliação Experimental

Com o intuito de comparar o desempenho da proposta com os diferentes métodos de quantificação, neste trabalho, selecionamos os métodos EMQ, FM, DyS, HDy, presentes na biblioteca MlQuantify[4], e o método KDEyML, pertencente a biblioteca Quapy[5]. Para esses quantificadores foi utilizado o algoritmo *RandomForest* como classificador base e os demais parâmetros foram mantidos com o *default* de cada biblioteca. Nesta avaliação, foram selecionados 17 conjuntos de dados binários da UCI *Machine Learning Repository*. Para a avaliação dos métodos foi utilizada a métrica *Mean Absolute Error* (MAE). Na Figura 7 (b) é ilustrado o fluxo da avaliação experimental realizada neste estudo. Cada conjunto de dados foi particionado, aleatoriamente, em conjunto de treinamento (70%) e conjunto de teste (30%). O conjunto de treinamento foi utilizado para o processo de aprendizado dos quantificadores avaliados. O conjunto de teste foi utilizado para a geração de *batches*. Para cada conjunto de teste, foram gerados 1100 *batches*, cada um com 100 instâncias, resultante de 100 *batches* para cada prevalência do conjunto: {0.0, 0.1, 0.2, …, 0.9, 1.0}. No processo de extração de características da avaliação experimental foi utilizado o mesmo quantificador e classificador base

[3] https://numpy.org/devdocs/reference/generated/numpy.histogram_bin_edges.html
[4] https://pypi.org/project/mlquantify/
[5] https://hlt-isti.github.io/QuaPy/index.html

induzidos na Etapa 3 do método. Todo o processo experimental foi repetido 30 vezes para cada conjunto de dados, a fim de observar a variabilidade do desempenho de cada quantificador em relação ao método proposto. Os experimentos foram realizados em um computador Intel Core i9-14900 com 32 núcleos (24 núcleos físicos) e 64GB de memória RAM.

4.3 Resultados Parciais

Os resultados obtidos na avaliação experimental estão descritos na Tabela. Para cada *dataset*, reporta-se a mediana do desempenho dos algoritmos de quantificação considerando as 30 repetições realizadas. Além disso, apresenta-se o *rank* correspondente de cada algoritmo em cada *dataset*. Os resultados são expressos no formato mediana (*rank*).

Tabela 1: Desempenho dos métodos.

Dataset	EMQ	FM	HDy	KDE	DyS	SL_A	SL_HE	SL_H
balance.1	0,01809(5)	0,03195(8)	0,02000(7)	0,01646(3)	0,01649(4)	**2,6E-16(1)**	0,00600(2)	0,01899(6)
balance.3	0,01824(6)	0,03262(8)	0,02000(7)	0,01662(3)	0,01686(4)	**2,6E-16(1)**	0,00900(2)	0,01700(5)
breast-cancer	0,01085(5)	0,01334(7)	0,02500(8)	0,01049(4)	0,01177(6)	**2,7E-17(1,5)**	8,3E-17(3)	**2,7E-17(1,5)**
cmc.1	0,07992(5)	0,09529(6)	0,07000(3)	0,06933(2)	**0,06731(1)**	0,09000(8)	0,07899(4)	0,09699(7)
cmc.2	0,12106(6)	0,11913(5)	0,22000(8)	0,10134(3)	**0,09996(1)**	0,10000(2)	0,11800(4)	0,14300(7)
cmc.3	0,13559(5)	0,13912(6)	0,13000(4)	0,10726(3)	0,10593(2)	**0,10000(1)**	0,14000(7)	0,16200(8)
ctg.1	0,01577(6)	0,01857(7)	0,04000(8)	0,01276(4)	0,01387(5)	**2,7E-17(1)**	8,8E-16(2)	0,00100(3)
ctg.2	0,02082(5)	0,02360(7)	0,08500(8)	0,01854(3)	0,01988(4)	**2,5E-16(1)**	0,01399(2)	0,02200(6)
ctg.3	0,01269(7)	0,01264(6)	0,07500(8)	0,00959(4)	0,01085(5)	**2,7E-17(1)**	2,6E-16(3)	2,4E-16(2)
german	**0,06528(1)**	0,07905(4)	0,09000(6)	0,06865(3)	0,06724(2)	0,09000(8)	0,08299(5)	0,09900(7)
mammographic	0,06056(7)	0,05738(6)	0,04500(3)	0,04373(2)	**0,04350(1)**	0,09000(8)	0,04900(4)	0,04999(5)
spambase	0,01682(7)	0,01509(6)	0,01900(8)	0,01155(4)	0,01199(5)	**2,7E-17(1)**	2,4E-16(2)	9,9E16(3)
tictactoe	0,05562(8)	0,00923(5)	0,02500(7)	0,00479(2)	0,00697(4)	**2,6E-16(1)**	0,00499(3)	0,01700(6)
transfusion	0,15772(4)	0,15536(3)	0,30000(8)	**0,11636(1)**	0,14938(2)	0,20000(7)	0,17500(6)	0,16400(5)
wine-q-red	0,03899(2)	0,05089(5)	0,04400(4)	**0,03865(1)**	0,03908(3)	0,09000(8)	0,05699(6)	0,06100(7)
wine-q-white	0,03204(4)	0,03993(5)	0,04900(7)	0,03051(2)	0,03138(3)	**9,9E-16(1)**	0,04700(6)	0,05399(8)
yeast	**0,05286(1)**	0,07472(5)	0,07900(6)	0,05997(3)	0,05631(2)	0,09000(8)	0,07399(4)	0,08399(7)
Mediana do Rank*	5	6	7	3	3	1	4	6

Fonte: A autora (2025).

Nota:
*: Foi calculado a mediana das posições no *rank* para os métodos;
- Desempenho avaliado utilizando MAE;
- Negrito: Melhor desempenho no *rank*.

Na tabela 2 é reportado o número de vezes que os métodos propostos ajustaram com sucesso o desempenho do modelo. A partir da análise dos resultados é possível notar que para 10 *datasets* o método proposto, SL_A alcançou o objetivo geral de otimizar o desempenho do método DyS. Porém,

apesar de promissor, é possível observar que há também um aumento no erro de predição para alguns *datasets*. Os métodos SL_HE e SL_H alcançaram um desempenho inferior quando comparado com o método SL_A.

Tabela 2: Comparativo geral dos métodos de ajuste propostos *versus* DyS.

–	SL_A	SL_HE	SL_H
DYS	10	8	3

Fonte: A autora (2025).

A suposição para esses resultados é que a mediana das estimativas do quantificador seja uma informação relevante para a acurácia do método, uma vez que as meta-*features* utilizadas por estes não caracterizaram os diferentes cenários conforme visto na Figura 6. Além disso, ressalta-se que estes métodos utilizam menos batches quando comparados com SL_A. Neste sentido, o método proposto necessita de ajustes que possibilitem a identificação das correções necessárias para estes cenários.

4.4 Expansão do projeto

Apesar dos resultados promissores para maioria dos *datasets* avaliados, constatou-se que para alguns destes a utilização do método proposto aumenta o erro de predição. Diante deste cenário, como próximos passos para a melhoria e expansão do método, elencamos as seguintes tarefas:

i) Identificar os cenários em que a aplicação do método é vantajosa;
ii) Avaliar o método em diferentes *datasets* binários;
iii) Explorar a utilização do método em diferentes algoritmos de quantificação;
iv) Explorar o método no cenário multiclasse;
v) Avaliar o desempenho de outros algoritmos como *meta-learner*;
vi) Explorar outras *meta-features* como, por exemplo, o tamanho do *batch*;
vii) Avaliação dos parâmetros do método proposto;
viii) Expandir da faixa de prevalências do método.

5 Cronograma

Espera-se que o desenvolvimento deste trabalho cumpra as seguintes etapas:

- Etapa 1 – Revisão da Literatura;
- Etapa 2 – Realização de experimentos

- Etapa 3 – Elaboração da qualificação
- Etapa 4 – Expansão do método proposto;
- Etapa 5 – Escrita e submissão de artigos em eventos científicos;
- Etapa 6 – Elaboração da dissertação;
- Etapa 7 – Apresentação e defesa da dissertação.

Tabela 3: Cronograma das atividades.

-	Meses																							
Etapas	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
1																								
2																								
3																								
4																								
5																								
6																								
7																								

Fonte: A autora (2025).

Referências Bibliográfica

Alaiz-Rodríguez, R., Alegre-Gutiérrez, E., González-Castro, V., & Sánchez, L. (2008). Quantifying the proportion of damaged sperm cells based on image analysis and neural networks. In *Proceedings of the WSEAS International Conference on Simulation, Modelling and Optimization (SMO'08)* (pp. 383–388). WSEAS Press

Bella, A., Ferri, C., Hernández-Orallo, J., & Ramírez-Quintana, M. J. (2010). Quantification via probability estimators. In *Proceedings of the IEEE International Conference on Data Mining*. https://doi.org/10.1109/ICDM.2010.75

Brazdil, P., Giraud-Carrier, C., Soares, C., & Vilalta, R. (2009). *Meta-learning: Applications to data mining*. Springer Science and Business Media.

Faceli, K., Lorena, A. C., Gama, J., & Carvalho, A. C. P. de L. F. (2011). *Inteligência artificial: Uma abordagem de aprendizado de máquina*. LTC.

Forman, G. (2005). Counting positives accurately despite inaccurate classification. In *Lecture Notes in Computer Science* (pp. 564–575). https://doi.org/10.1007/11564096_55

Giachanou, A., & Crestani, F. (2016). Like it or not: A survey of Twitter sentiment analysis methods. *ACM Computing Surveys, 49*(2), 28:1–28:41.

Gomes, G. B., Zalewski, W., & Maletzke, A. G. (2024). Enhancing quantification through meta-learning. In *Proceedings of the Learning to Quantify Workshop (LeQua 2024)* (pp. 35–50)

González, P., Castaño, A., Chawla, N. V., & Del Coz, J. J. (2017). A review on quantification learning. *ACM Computing Surveys, 50*(5), Article 74. https://doi.org/10.1145/3117807

González, P., Moreo, A., & Sebastiani, F. (2024). Binary quantification and dataset shift: An experimental investigation. *Data Mining and Knowledge Discovery, 38*, 1670–1712. https://doi.org/10.1007/s10618-024-01014-1

González-Castro, V., Alaiz-Rodríguez, R., & Alegre, E. (2013). Class distribution estimation based on the Hellinger distance. *Information Sciences, 218*, 146–164. https://doi.org/10.1016/j.ins.2012.05.028

Hassan, W., Maletzke, A., & Batista, G. (2020). Accurately quantifying a billion instances per second. In *Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics* (DSAA). https://doi.org/10.1109/DSAA49011.2020.00012

Kelly, M., Longjohn, R., & Nottingham, K. (n.d.). The UCI Machine Learning Repository. https://archive.ics.uci.edu/

Lorena, A. C., Maciel, A. I., de Miranda, P. B. C., & Costa, I. G. (2018). Data complexity meta-features for regression problems. *Machine Learning, 107*, 209–246. https://doi.org/10.1007/s10994-017-5681-1

Maletzke, A., Reis, D. D., Cherman, E., & Batista, G. (2019). DYS: A framework for mixture models in quantification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1), 4552–4560. https://doi.org/10.1609/aaai.v33i01.33014552

Maletzke, A. G. (2019). *Quantificação binária em cenários não estacionários* (Tese de doutorado). Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP.

Maletzke, A., Hassan, W., Dos Reis, D., & Batista, G. (2021). The importance of the test set size in quantification assessment. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)* (pp. 2640–2646).

Milli, L., Monreale, A., Rossetti, G., Giannotti, F., Pedreschi, D., & Sebastiani, F. (2013). Quantification trees. In *Proceedings of the 2013 IEEE 13th International Conference on Data Mining* (ICDM). https://doi.org/10.1109/ICDM.2013.122

Moreo, A., Esuli, A., & Sebastiani, F. (2021). QuaPy: A Python-based framework for quantification. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (pp. 4534–4543). https://doi.org/10.1145/3459637.3482053

mlquantify. (n.d.). *PyPI: mlquantify*. https://pypi.org/project/mlquantify/

Rivolli, A., Garcia, L. P., Soares, C., Vanschoren, J., & De Carvalho, A. C. (2022). Meta-features for meta-learning. Knowledge-based Systems, 240, 108101. https://doi.org/10.1016/j.knosys.2021.108101

Schumacher, T., Strohmaier, M., & Lemmerich, F. (2021). A comparative evaluation of quantification methods. *arXiv preprint arXiv:2103.03223*. https://arxiv.org/abs/2103.03223

Silva, D. F., Souza, V. M., Ellis, D. P., Keogh, E. J., & Batista, G. E. (2015). Exploring low cost laser sensors to identify flying insect species. *Journal of Intelligent & Robotic Systems, 80*(1), 313–330.

Tasche, D. (2019). Confidence intervals for class prevalences under prior probability shift. *arXiv preprint arXiv:1906.04119*. https://doi.org/10.48550/arXiv.1906.04119.