

Classification of Real Estate Pricing in São Paulo

1stGuilherme dos Santos Wisniewski 2nd Pedro Vinicius Guandalini Vicente

Registro Acadêmico: 813319

Universidade Federal de São Carlos

São Carlos, Brazil

guilhermesw@estudante.ufscar.br

Registro Acadêmico: 812124

Universidade Federal de São Carlos

São Carlos, Brazil

pedrovincente@estudante.ufscar.br

3rd Pedro Henrique Ghiotto

Registro Acadêmico: 812115

Universidade Federal de São Carlos

São Carlos, Brazil

pedroghiotto@estudante.ufscar.br

Abstract—Este trabalho tem como objetivos buscarmos um modelo apropriado para a classificação de valores de alugueis na cidade de São Paulo. Por meio desse trabalho, buscamos tanto trazer maior respaldo para aqueles que buscam um imóvel na capital paulista, evitando fraudes ou valores desproporcionais, quanto aqueles que são possuidores de imóvel e pretendem anunciá-lo. Na sua implementação, foi utilizada a técnica de ensemble, combinando modelos em busca de um melhor resultado.

I. INTRODUÇÃO E MOTIVAÇÃO

A classificação de valores de aluguel de residências em grandes centros urbanos, como São Paulo, é uma tarefa complexa e essencial para o planejamento econômico, tanto por parte de proprietários quanto de locatários. A definição de preços justos depende de diversos fatores, incluindo localização, tamanho do imóvel, infraestrutura disponível e características econômicas da região. Modelos de aprendizado de máquina têm se mostrado eficazes para lidar com essa complexidade, oferecendo soluções precisas e baseadas em dados para a predição de valores de aluguel.

O uso de ensemble é especialmente relevante para problemas que envolvem alta variabilidade e múltiplas dimensões, como o mercado imobiliário de São Paulo. Ao combinar as saídas de diferentes algoritmos, é possível reduzir o impacto de erros individuais e obter uma estimativa mais robusta. Além disso, a diversidade dos modelos utilizados contribui para melhorar a generalização das previsões, tornando-as aplicáveis a cenários reais e dinâmicos.

Este trabalho tem como objetivo aplicar métodos de ensemble para classificar o valor de aluguel de residências na cidade de São Paulo. Para isso, um conjunto de dados reais do mercado imobiliário da capital paulista será utilizado, considerando atributos como características dos imóveis, tipos de imóveis e indicadores socioeconômicos das respectivas subprefeituras. A análise busca não apenas atingir alta acurácia nas classificações, mas também entender quais fatores mais influenciam os resultados, oferecendo uma contribuição prática e teórica para a área de aprendizado de máquina e estudos de mercado imobiliário.

II. OBJETIVOS

O objetivo deste presente trabalho é, a partir de uma base de dados reais de imóveis em São Paulo, utilizar a técnicas de Ensemble para classificar as faixas de valores dos imóveis

com precisão e acurácia significativos, de modo que a partir do projeto realizado, possa-se obter uma estimativa satisfatória e sistemática da precificação de novos imóveis que podem se beneficiar dessa base de valores.

III. METODOLOGIA EXPERIMENTAL

A. Ferramenta Utilizada

O ambiente de programação utilizado para a criação dos modelos de classificação foi o Google Colab em virtude da facilidade de uso e do caráter colaborativo da plataforma. Além disso, utilizou-se a linguagem de programação Python em todas as etapas do processo tanto para tratamento dos dados (juntamente a ferramenta de planilhas Google), quanto para treinamento dos modelos e obtenção de resultados. Por fim, deve-se citar também o uso da biblioteca Scikit-learn (sklearn) na aplicação dos modelos, além das bibliotecas Numpy, Pandas e Nominatim para o tratamento dos dados.

B. Dados Analisados

O conjunto bruto de dados, antes de passar pelo tratamento, foi uma base de dados, datada de 1 de Maio de 2023, disponível via Kaggle no formato CSV, possuindo cerca de 736.36kB de tamanho, contando com um total de 12 mil tuplas, além de 8 atributos, que são:

- *Adress: Endereço da Propriedade*
- *District: teoricamente seria o distrito, porém se confunde com os bairros também*
- *Area: Tamanho da propriedade referenciada*
- *Bedrooms: Quantidade de quartos catalogadas no imóvel*
- *Garage: Atributo referente à quantidade de garagens do imóvel*
- *Rent: Atributo referente ao valor do aluguel do imóvel*
- *Type: Tipo do imóvel catalogado*
- *Total: Valor total do imóvel, contando aluguel, condomínio, impostos e demais taxas*

C. Obtenção de Dados e Tratamento

A base de dados bruta corresponde a várias características do imóvel, como sua área, numero de quartos e aluguel. Todavia, diferentemente desses valores, que são numéricos, a localização era no formato string, o que fez com que fosse necessário tratamento desses dados para que auxiliasse na classificação, o exemplo está na figura 1.

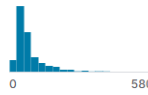

Δ address	Δ district	# area	# bedrooms
The address of the property	The district where the property is located	The area of the property in square meters	The number of bedrooms in the property
5348 unique values	Bela Vista 3% Vila Mariana 2% Other (11073) 95%		
Rua Herval	Belenzinho	21	1
Avenida São Miguel	Vila Marieta	15	1
Rua Oscar Freire	Pinheiros	18	1

Fig. 1: Atributos de localização no Data Set

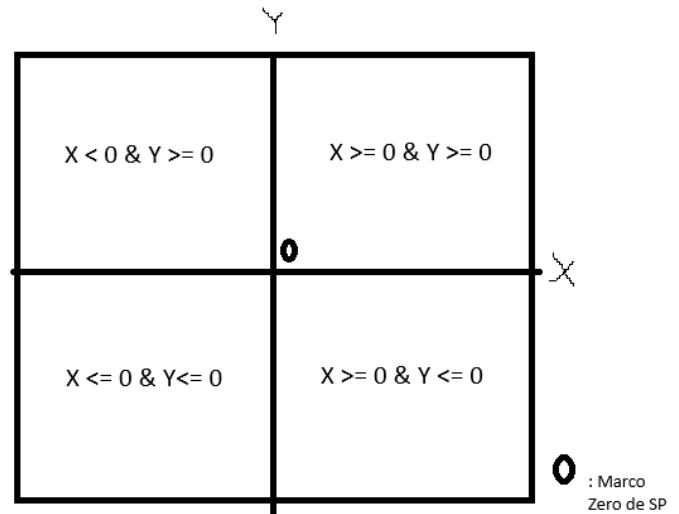


Fig. 2: Gráfico de como seria a conversão com base no Marco Zero

A partir disso, a ideia original foi de converter a string de endereço e de bairro em uma localização em coordenadas latitude e longitude utilizando a API NOMINATIM, inclusive, utilizou-se essa biblioteca, pois a API do google cloud maps era paga a partir de determinado número de solicitações, como faríamos mais de 11 mil solicitações à API, optou-se pela Nominatim.

Outro ponto que foi impactante negativamente na ideia de converter em coordenadas foi na tentativa de utilizar as coordenadas em absoluto como atributos para o modelo. Ao fazer a correlação dos atributos, praticamente não houve correlação entre a classe final e as coordenadas, uma vez que não faz sentido lógico analisá-las em absoluto.

Depois disso, adaptou-se essa ideia para a tentativa de normalização das coordenadas em relação ao Marco Zero de São Paulo (coordenadas 23° 33 01 S, 46° 38 02 O), porém não houve uma melhora na correlação entre o atributo, fazendo com que fosse necessário repensar a ideia de converter em coordenadas.

Devido aos fatos já citados, uma nova tentativa de pré-processamento da string de localização foi a de conversão dos endereços em coordenadas e, posteriormente, em Macro-Região de São Paulo, isto é, zona leste, zona sul, zona oeste, zona norte e Centro. O método como seria aplicado é através dos valores das coordenadas de latitude e longitude a partir do marco zero de São paulo (observar Figura 2). Porém, devido à diversidade de valores ser altíssima mesmo dentro de cada Macro-Região, não foi possível utilizar

Após essa série de tentativas de conversão, concluiu-se que não seria possível utilizar no formato de coordenadas, assim, foi feita uma busca por possíveis fontes em valores numéricos que, intuitivamente, têm relação com a localização dos imóveis, qual seja, o IDH da localidade do imóvel em um processo chamado Feature Transformation.

Para a aplicação dessa técnica de feature transformation, utilizou-se o dado grupo do endereço (rua - string) em conjunto à biblioteca Nominatim, uma vez que essa consegue retornar informações detalhadas sobre o endereço em string como o bairro e o CEP.

Primeiramente, tentou-se uma conversão do endereço em bairro, todavia, não foi encontrada nenhuma base de dados confiável de IDHs por bairro de São Paulo, o que poderia não trazer uma análise justa para o trabalho.

Em seguida, buscou-se no site da Prefeitura de São Paulo fontes de dados que fossem possíveis de conversão para IDH e encontrou-se uma Publicação datada de 2017 com os dados dos IDHs de cada subprefeitura de São Paulo, figura 3.

Com isso, foi estabelecido que seria necessário fazermos a conversão do endereço em SubPrefeituras para que, assim, fossem obtidos os IDHs. Para a realização dessa tarefa, foi usada uma outra Publicação também da Prefeitura de São Paulo como referência, uma vez que teria segurança da fonte. Pode-se ver a tabela encontrada na figura 4:

Subprefeitura *	IDH-M		Variação %
	2000	2010	
Pinheiros	0,910	0,942	3,52
Vila Mariana	0,897	0,938	4,57
Santo Amaro	0,867	0,909	4,84
Lapa	0,849	0,906	6,71
Sé	0,831	0,889	6,98
Mooca	0,811	0,869	7,15
Santana / Tucuruvi	0,811	0,869	7,15
Butantã	0,789	0,859	8,87
Ipiranga	0,759	0,824	8,56
Aricanduva / Vila Formosa	0,762	0,822	7,87
Jabaquara	0,756	0,816	7,94
Penha	0,745	0,804	7,92
Casa Verde	0,732	0,799	9,15
Vila Maria / Vila Guilherme	0,733	0,793	8,19
Pirituba / Jaraguá	0,718	0,787	9,61
Vila Prudente	0,723	0,785	8,58
Campo Limpo	0,699	0,783	12,02
Ermelino Matarazzo	0,707	0,777	9,90
Jaçanã / Tremembé	0,716	0,768	7,26
Freguesia do Ó / Brasilândia	0,677	0,762	12,56
Itaquera	0,691	0,758	9,70
Cidade Ademar	0,662	0,758	14,50
Capela do Socorro	0,656	0,750	14,33
São Miguel Paulista	0,650	0,736	13,23
São Mateus	0,658	0,732	11,25
Perus	0,637	0,731	14,76
Itaim Paulista	0,639	0,725	13,46
M'Boi Mirim	0,638	0,716	12,23
Guaianases	0,621	0,713	14,81
Cidade Tiradentes	0,634	0,708	11,67
Parelheiros	0,593	0,680	14,67

Fig. 3: Print com dados dos IDHs das SubPrefeituras

Macroregião	Subprefeitura	Distritos	CEP Início	CEP Fim
Centro	Sé	Sé / República	01000-000	01099-999
		Bom Retiro / Barra Funda	01100-000	01199-999
		Santa Cecília / Consolação	01200-000	01299-999
		Consolação / Bela Vista	01300-000	01399-999
		Liberdade / Cambuci	01500-000	01599-999
Zona Norte	Santana - Tucuruvi	Santana	02000-000	02099-999
		Vila Maria - Vila Guilherme	02100-000	02199-999
		Santana - Tucuruvi / Jaçanã - Tremembé	02200-000	02299-999
		Jaçanã - Tremembé	02300-000	02399-999
		Santana	02400-000	02499-999
	Casa Verde	Casa Verde	02500-000	02599-999
		Cachoeirinha	02600-000	02699-999
		Limão	02700-000	02799-999
		Brasilândia	02800-000	02899-999
		Brasilândia - Freguesia	02900-000	02999-999
	Pirituba	Pirituba / São Domingos	05100-000	05199-999
		Pirituba / Perus	05200-000	05299-999
	Mooca	Jaraguá / Perus	03000-000	03099-999
		Belém / Brás / Pati	03100-000	03199-999
		Mooca / Vila Prudente	03200-000	03299-999
		São Lucas / Sapopemba	03300-000	03399-999
		Tatuapé / Aricanduva - Vila Formosa	03400-000	03499-999
Penha	Penha	Vila Matilde / Artur Alvim	03500-000	03599-999
		Penha	03600-000	03699-999

Fig. 4: Print publicação com as SubPrefeituras

Como pode ser visto, havia possibilidade de conversão de Distrito para subprefeitura, porém havia o problema dos dados do atributo District que não estavam corretamente relacionados às ruas, isto é, havia nomes de bairros que não de São Paulo, bairros com erros de escrita, bairros que são de São Paulo, mas que não correspondem à rua, etc. Assim, por meio do CEP de retorno da Nominatim, foi a conversão para a determinada subPrefeitura. De forma geral, a sequência da conversões até o IDH ficou:

A partir do exposto na figura 5, tem-se, ao final de todas as

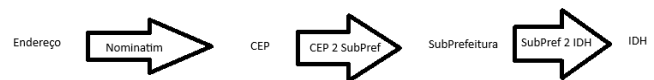


Fig. 5: Gráfico de conversões

conversões, uma coluna só com os IDHs correspondentes de cada subprefeitura dos endereços, o que fez com que os dados passassem por um processo de normalização.

O processo de tratamento dos dados de localização até sua transformada em IDH de subprefeitura está presente no seguinte link.

Os dados estão separados em arquivos separados pois foi feita a conversão de forma "paralelizada" e, depois, concatenada em 1 arquivo apenas. Ademais, foram utilizados apenas os 3 primeiros dígitos do CEP para identificar suas subprefeituras, posto que isso já bastava para referenciar.

Apesar da complexidade para o tratamento da localização, os demais atributos foram de menor complexidade: para o tipo de imóvel, também do tipo string, foi estabelecida uma hierarquia com base nos tipos tendo valores inteiros de 0 a 3: "Studio e kitnet"(valor 0), "Apartamento"(valor 1), "Casa em condomínio"(valor 2) e "Casa"(valor 3).

Após feito esse tratamento no tipo, foi necessário normalizar os dados utilizando a técnica de Min-Max, para o melhor treinamento dos modelos.

Por fim, tendo em vista que o projeto é de classificação, foi feito também o processo de discretização da classe target. Para isso, foi-se utilizada a classe "total" e discretizada da seguinte forma: Até 1850, de 1850 a 3200, de 3200 a 5650 e 5650 ou mais.

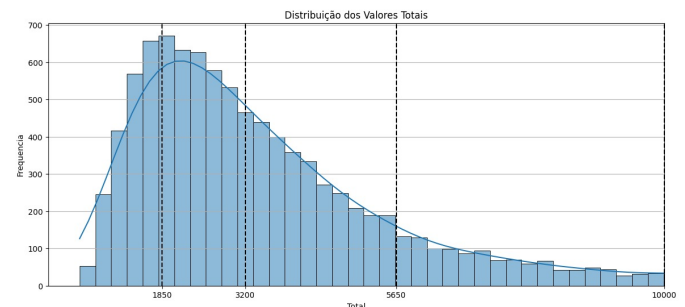


Fig. 6: distribuição dos dados

Em resumo, o processo de tratamento dos dados consistiu no processo de feature transformation do endereço para IDH, do conversão do tipo de imóvel para dados numéricos, além do processo de normalização desses atributos e da discretização da classe target a fim de viabilizar o processo de Classificação.

IV. RESULTADOS

A. Modelos Utilizados

O nosso conjunto de dados é complexo e multiclasse, sem divisões claras entre as classes criadas, para isso optamos por

explorar método ensemble do tipo bagging para tentar aprimorar os resultados obtidos por outros métodos tradicionais. Os métodos clássicos de classificação utilizados foram:

- Knn [B]
- Random Forest [C]
- Decision Tree [D]
- SVM [E]
- Gradient Boosting [F]
- Naive Bayes [G]
- Multi Layer Perceptron [H]

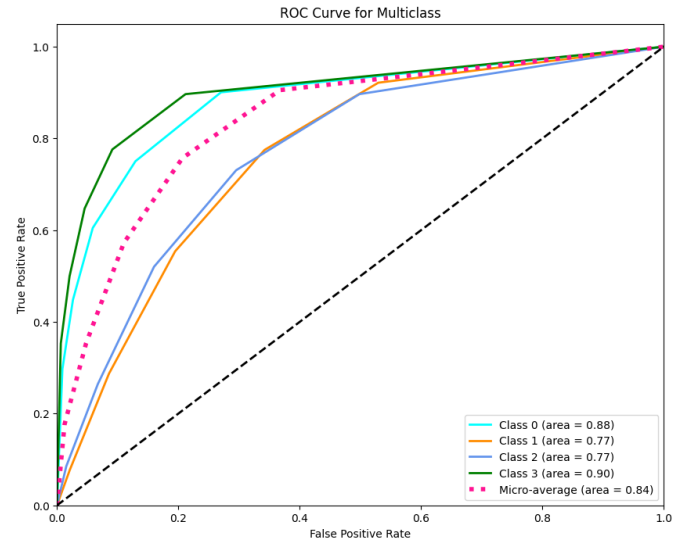
Para a análise e verificação de cada um dos modelos foi realizada a validação cruzada com 10 folds estratificados, onde em cada uma das 10 iterações o modelo foi treinado com 90% dos dados e testados com os 10% restantes. Assim mantivemos a uniformidade na quantidade dos dados no modelo de avaliação para que eles possam ser utilizados e comparados no ensemble final com votação simples e com pesos.

Como mostrado pelas imagens abaixo, foram coletadas e analisadas a Curva ROC[a], a Matrix de Confusão[b] e o Relatório de Classificação[c] contendo dados como acurácia, precisão, medida-F e Recall de todos os modelos usados, para que à partir de nossas observações, pudéssemos decidir como ficaria a estrutura e pesos de cada modelo nos comitês de votação dos nossos ensembles.

Dessa forma, notamos melhores performances com o SVM e o Gradient Boosting (modelo que por si só utiliza-se de ensemble), obtendo acurácias de 0.60 e 0.64 respectivamente, além de desempenhos muito acima do “chute aleatório”, que foi ilustrado pelos gráficos da Curva HOC.

Outra observação importante é que o Naive Bayes teve o pior desempenho, muito abaixo do esperado como mostrado por sua matriz de confusão com pouquíssimos valores na diagonal principal e sua incapacidade em prever dados da classe 0 (como mostrado por sua precisão para essa categoria). E por esse motivo colocamos a sua participação com peso negativo no ensemble de votação ponderada e, no método por votação simples ele foi excluído completamente para garantir que ele não atrapalhasse nossas predições com suas inconsistências.

B. KNN



(a) Curva ROC do KNN. Apresenta classificação adequada, acima da previsão aleatória.

Confusion Matrix				
Actual	0	1	2	3
	1372	668	60	3
	543	1941	624	40
	100	853	1560	332
	13	95	529	1172
Predicted				
	0	1	2	3

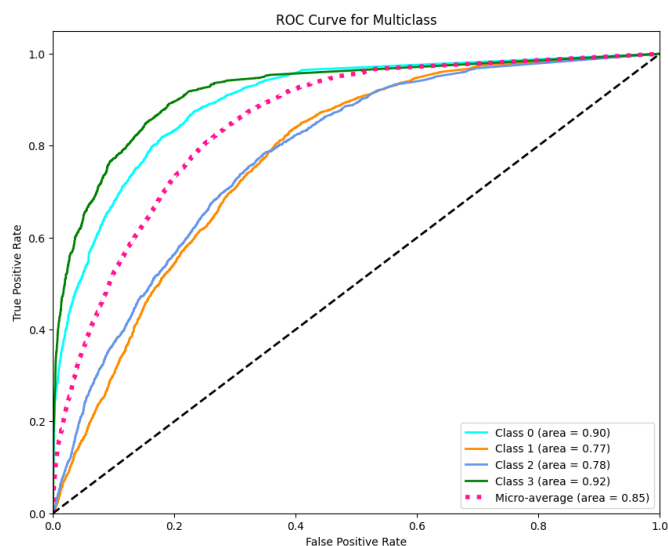
(b) Matriz de Confusão, previsões adequadas na diagonal principal

Relatório de Classificação:				
	precision	recall	f1-score	support
0	0.68	0.65	0.66	2103
1	0.55	0.62	0.58	3148
2	0.56	0.55	0.56	2845
3	0.76	0.65	0.70	1809
accuracy			0.61	9905
macro avg	0.64	0.62	0.62	9905
weighted avg	0.62	0.61	0.61	9905
Balanced Accuracy Score: 0.6162963611991366				

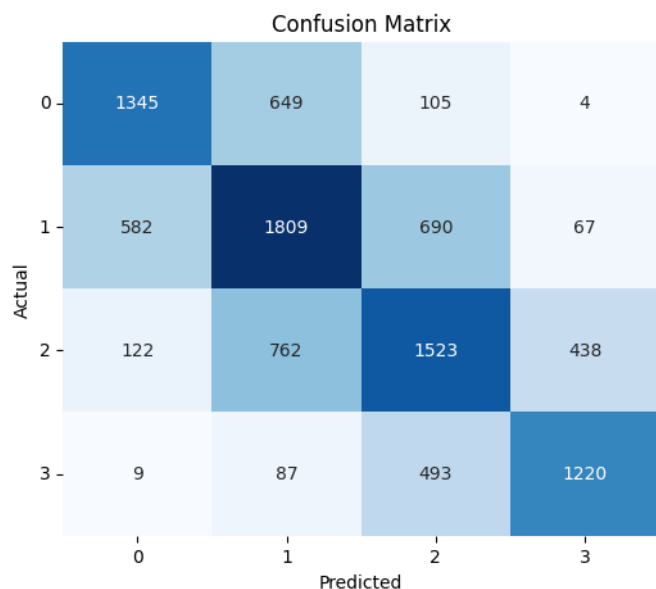
(c) Relatório de Classificação, valores adequados

Fig. 7: KNN

C. Random Forest



(a) Curva ROC do Random Forest. Apresenta classificação adequada, acima da previsão aleatória.



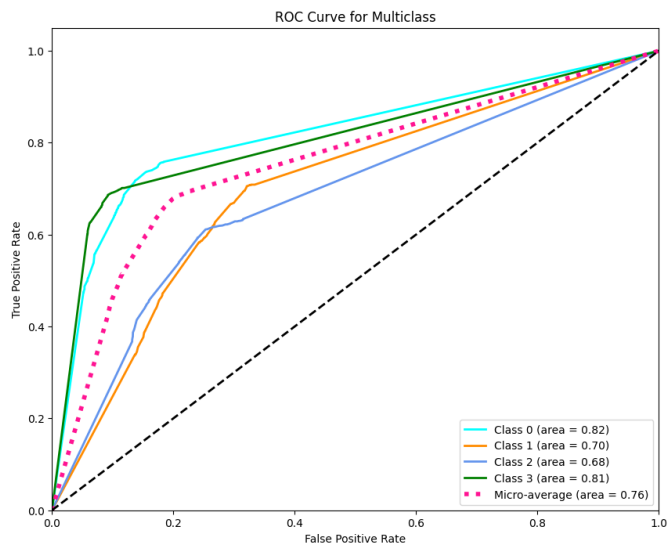
(b) Matriz de Confusão, previsões adequadas na diagonal principal

Relatório de Classificação:				
	precision	recall	f1-score	support
0	0.65	0.64	0.65	2103
1	0.55	0.57	0.56	3148
2	0.54	0.54	0.54	2845
3	0.71	0.67	0.69	1809
accuracy			0.60	9905
macro avg	0.61	0.61	0.61	9905
weighted avg	0.60	0.60	0.60	9905
Balanced Accuracy Score: 0.6059859955884676				

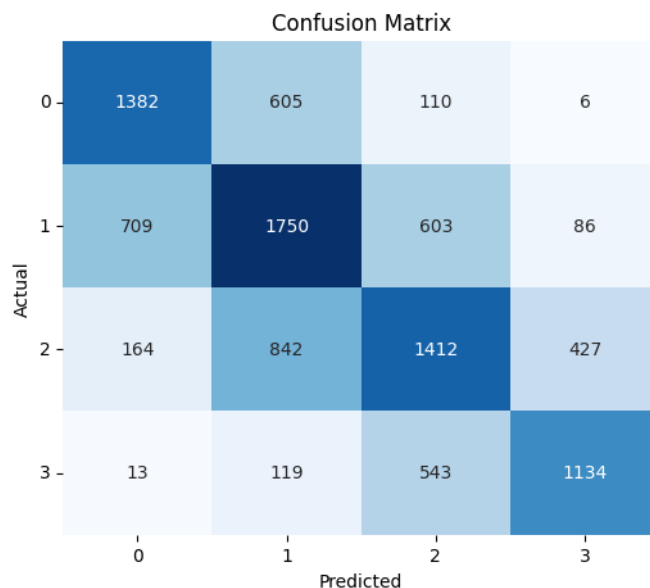
(c) Relatório de Classificação, valores adequados

Fig. 8: Random Forest

D. Decision Tree



(a) Curva ROC do Decision Tree. Apresenta classificação adequada, acima da previsão aleatória.



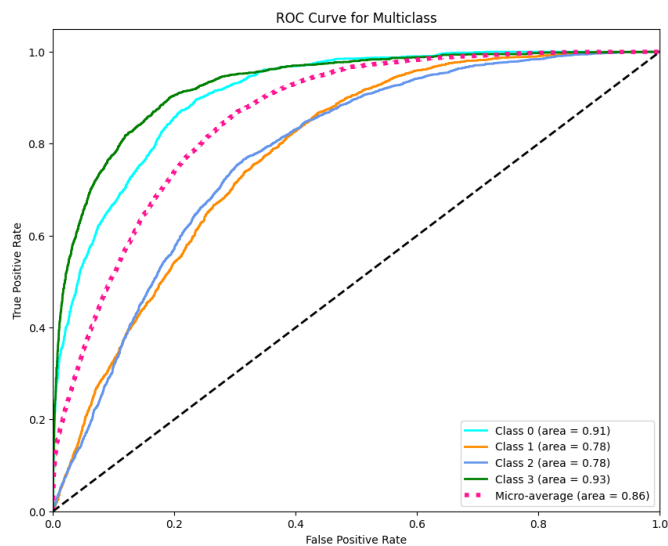
(b) Matriz de Confusão, previsões adequadas na diagonal principal

Relatório de Classificação:				
	precision	recall	f1-score	support
0	0.61	0.66	0.63	2103
1	0.53	0.56	0.54	3148
2	0.53	0.50	0.51	2845
3	0.69	0.63	0.66	1809
accuracy			0.57	9905
macro avg	0.59	0.58	0.59	9905
weighted avg	0.57	0.57	0.57	9905
Balanced Accuracy Score: 0.5840599856867512				

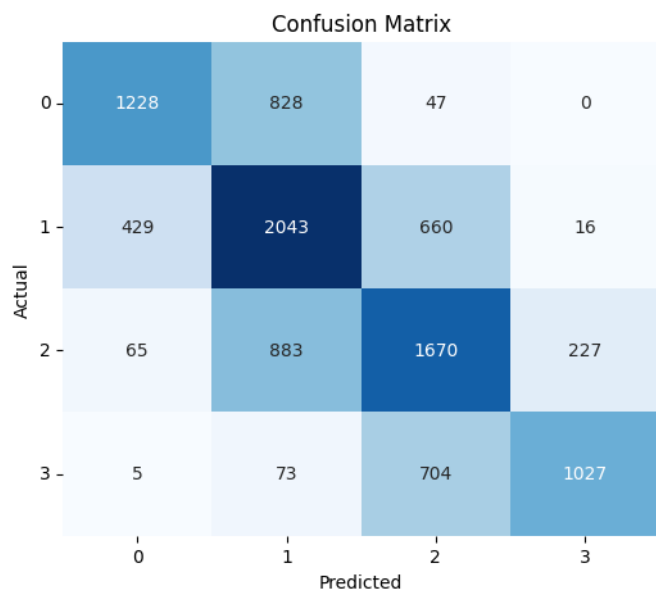
(c) Relatório de Classificação, valores adequados

Fig. 9: Decision Tree

E. SVM



(a) Curva ROC do SVM. Apresenta classificação adequada, acima da previsão aleatória.



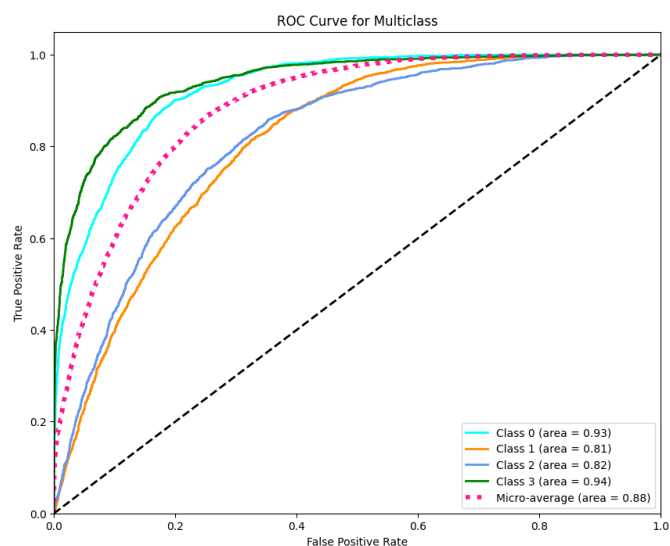
(b) Matriz de Confusão, previsões adequadas na diagonal principal

Relatório de Classificação:				
	precision	recall	f1-score	support
0	0.71	0.58	0.64	2103
1	0.53	0.65	0.59	3148
2	0.54	0.59	0.56	2845
3	0.81	0.57	0.67	1809
accuracy			0.60	9905
macro avg	0.65	0.60	0.61	9905
weighted avg	0.62	0.60	0.61	9905
Balanced Accuracy Score: 0.5969057255428475				

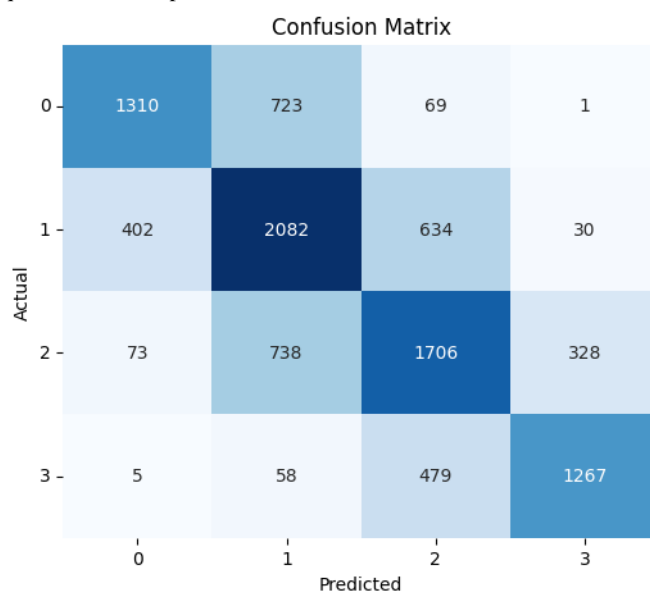
(c) Relatório de Classificação, valores adequados

Fig. 10: SVM

F. Gradient Boosting



(a) Curva ROC do Gradient Boosting. Apresenta classificação adequada, acima da previsão aleatória.



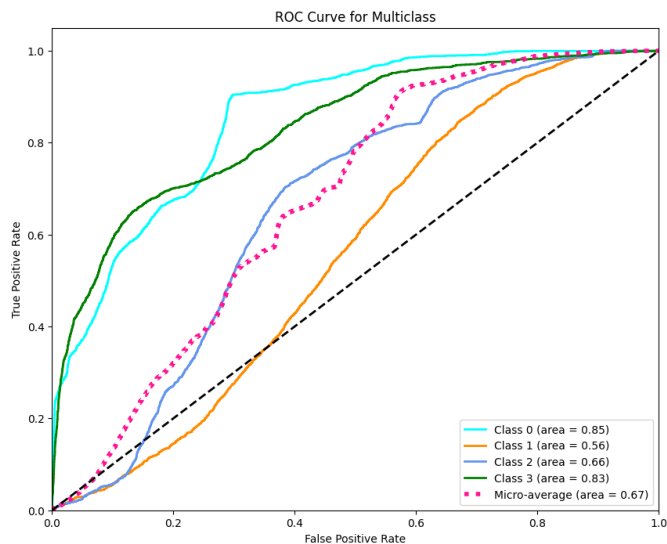
(b) Matriz de Confusão, previsões adequadas na diagonal principal

Relatório de Classificação:				
	precision	recall	f1-score	support
0	0.73	0.62	0.67	2103
1	0.58	0.66	0.62	3148
2	0.59	0.60	0.60	2845
3	0.78	0.70	0.74	1809
accuracy			0.64	9905
macro avg	0.67	0.65	0.66	9905
weighted avg	0.65	0.64	0.64	9905
Balanced Accuracy Score: 0.6460818496884706				

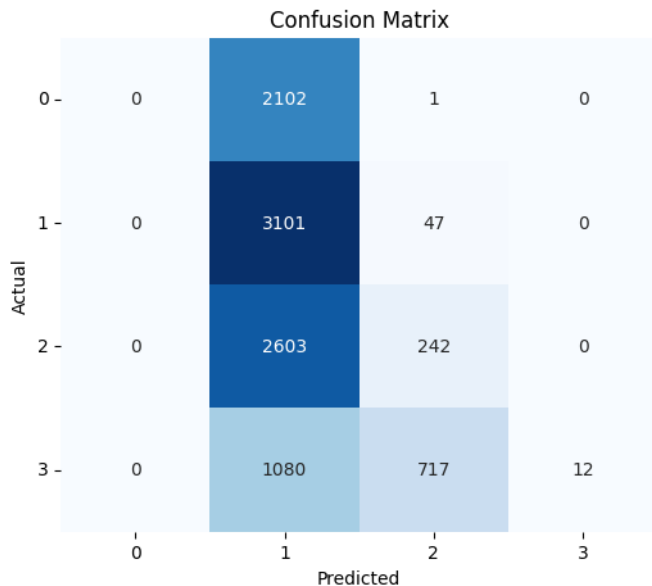
(c) Relatório de Classificação, valores adequados

Fig. 11: Gradient Boosting

G. Naive Bayes



(a) Curva ROC do Naive Bayes. Apresenta péssima classificação, abaixo da previsão aleatória.



(b) Matriz de Confusão, previsões inconsistentes

Relatório de Classificação:

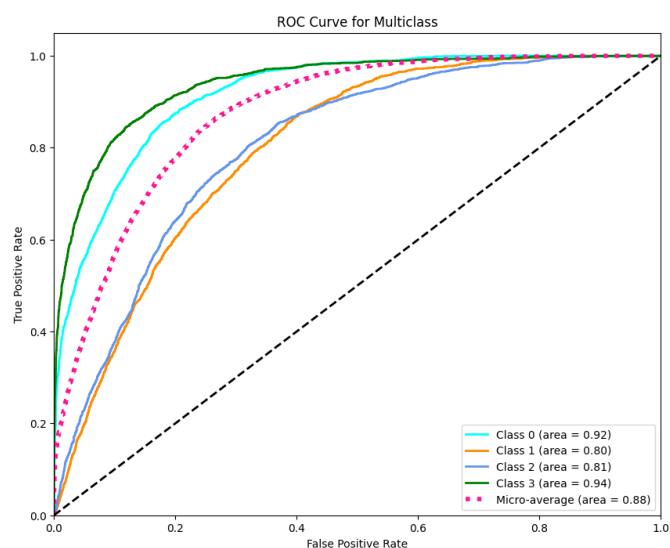
	precision	recall	f1-score	support
0	0.00	0.00	0.00	2103
1	0.35	0.99	0.52	3148
2	0.24	0.09	0.13	2845
3	1.00	0.01	0.01	1809
accuracy			0.34	9905
macro avg	0.40	0.27	0.16	9905
weighted avg	0.36	0.34	0.20	9905

Balanced Accuracy Score: 0.26919122405900997

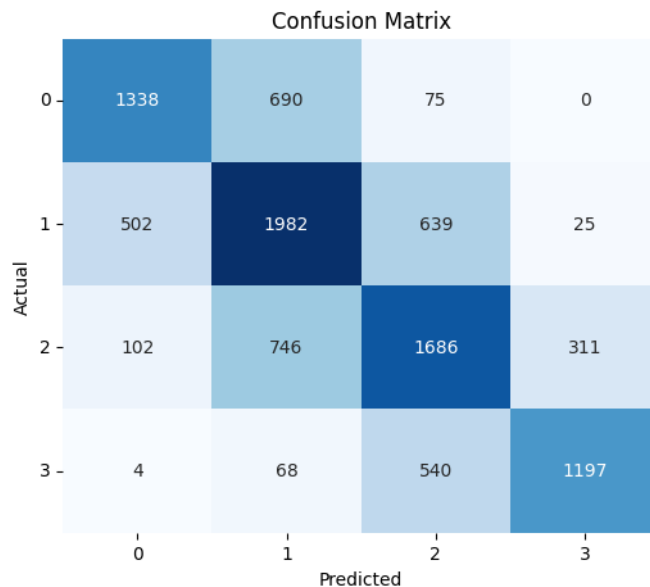
(c) Relatório de Classificação, valores muito abaixo do esperado

Fig. 12: Naive Bayes

H. Multi Layer Perceptron



(a) Curva ROC do MLP. Apresenta classificação adequada, acima da previsão aleatória.



(b) Matriz de Confusão, previsões adequadas na diagonal principal

Relatório de Classificação:

	precision	recall	f1-score	support
0	0.69	0.64	0.66	2103
1	0.57	0.63	0.60	3148
2	0.57	0.59	0.58	2845
3	0.78	0.66	0.72	1809
accuracy			0.63	9905
macro avg	0.65	0.63	0.64	9905
weighted avg	0.63	0.63	0.63	9905

Balanced Accuracy Score: 0.6300375555177383

(c) Relatório de Classificação, valores adequados

Fig. 13: MLP

I. Ensemble

1) *Votação Simples*: O método de votação simples sem o naive bayes teve o resultado: Balanced Accuracy Score (Voting): 0.6376370393036643, o que foi inferior ao resultado do Gradient Boosting.

2) *Votação Ponderada*: Os pesos utilizados na votação ponderada foram:

- Knn = 1
- Random Forest = 1
- Decision Tree = 1
- SVM = 2
- Gradient Boosting = 4
- Naive Bayes = -3
- Multi Layer Perceptron = 2

O método de votação ponderada teve o seguinte resultado: Balanced Accuracy Score (Voting): 0.6473826594286775, o que foi levemente (0.001) melhor que o Gradient Boosting.

V. CONCLUSÕES

Este trabalho explorou o uso de 9 diferentes modelos em aprendizado de máquina, incluindo Árvores de Decisão, Random Forest, KNN, Naive Bayes, SVM, Gradient Boosting e Multilayer Perceptron. Os modelos foram desenvolvidos e avaliados a partir de métricas de desempenho uniformes para decidir sobre seu impacto nos ensembles finais.

O resultado final do ensemble embora singelo, atendeu as expectativas do grupo, se demonstrando mais sólido e acurado do que qualquer um dos resultados que já haviam sido obtidos com os modelos individuais. Assim, aumentando a confiabilidade dos nossos resultados e sua robustez devido à grande variedade de técnicas e estratégias para a obtenção da predição de cada dado com o comitê de votação bem ajustado. Desse modo, possíveis distorções ou imprecisões que algum modelo possa ter na rotulação de um dado com características específicas é mitigado.

O resultado de 0.647 de acurácia pode parecer modesto à primeira vista. Com quatro classes, o resultado aleatório se aproxima de 0.25. No entanto, considerando que o problema envolve quatro classes sem uma distinção abrupta entre elas e que, na maioria das vezes, os erros dos modelos ocorrem em classes adjacentes, é possível observar que os modelos conseguiram extrair uma quantidade significativa de informações mesmo com poucos atributos disponíveis.

REFERENCES

- [1] Colab Final: <https://colab.research.google.com/drive/16YogCSsRVrsxt4mcqZ5giRPtEUjp3uqa?usp=sharing>
- [2] Colab para obter CEP: https://colab.research.google.com/drive/1VwJRIIS-ZOYHT2Xt7ABtaEWiJB0_bIHMD?usp=sharing
- [3] Colab para obter IDH: <https://colab.research.google.com/drive/1DkzJG1zLZuFeXsvzX9IZWmxDaL9xETtv?usp=sharing>
- [4] Link do Kaggle: <https://www.kaggle.com/datasets/renatosn/sao-paulo-housing-prices/data>
- [5] API de Geolocalização: <https://nominatim.org/>