Predição dos danos sofridos pelas construções do Nepal causados pelo terremoto de 2015

Ana Gabriela Faria da Silva
Bruno de Assis Silva
João Phellip de Mello da Rocha
Rodrigo Matheus Rocha de Medeiros

Universidade de São Paulo
Instituto de Matemática e Estatística

27 de Novembro de 2020

Introdução

Introdução

- Em 2015 ocorreu um terremoto com magnitude de 7,8 na escala Richter que atingiu o Nepal, Índia, Bangladesh, Paquistão e China. O Nepal foi o país mais atingido, sendo este o terremoto mais violento a atingir o país em 81 anos.
- Após o terremoto, o governo local realizou uma grande pesquisa domiciliar para avaliar os danos às construções nos distritos afetados pelo terremoto.
- Os dados coletados são formados principalmente por informações sobre a estrutura das construções da região atingida, e do grau de dano sofrido por elas.

Descrição dos dados

Descrição dos dados

- Cada linha representa uma construção específica na região que foi atingida.
- A base de dados possui 260.601 observações com 38 variáveis preditoras, em que as variáveis preditoras se dividem entre 5 quantitativas e 33 qualitativas.
- Optamos por utilizar cerca de 60% da base de dados fornecida para treinar os modelos, e os outros 40% como conjunto de validação.
- O conjunto de treinamento resultante possui 156361 observações.

Descrição dos dados

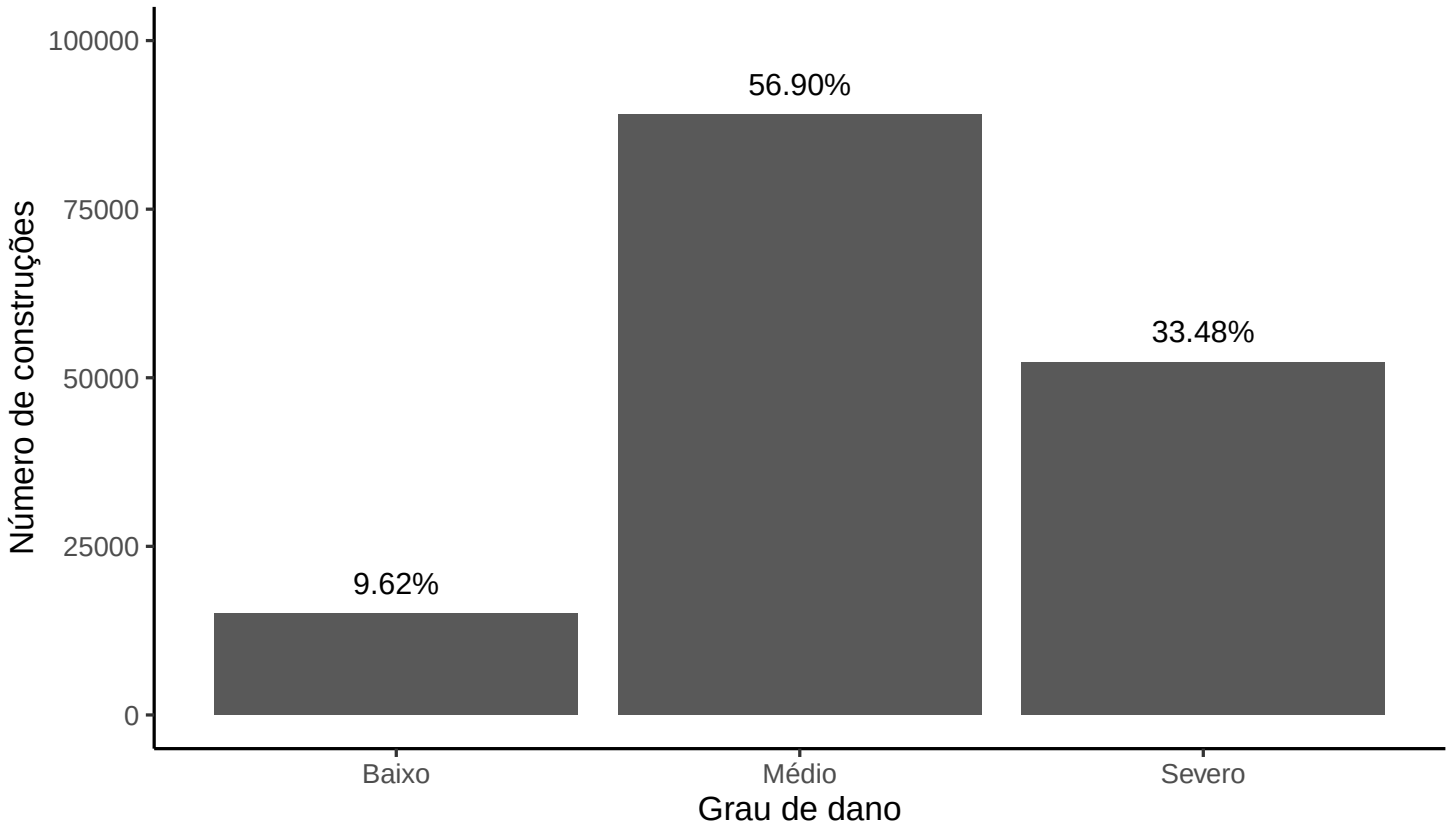
A variável resposta representa o nível de dano à construção causado pelo terremoto. Existem 3 graus de dano:

1 — representa grau baixo de dano;

2 — representa grau médio de dano;

3 — representa grau severo de dano.

Grau de dano


100000
75000
50000
25000
0
Número de construções
56.90%
33.48%
9.62%
Baixo
Médio
Severo
Grau de dano

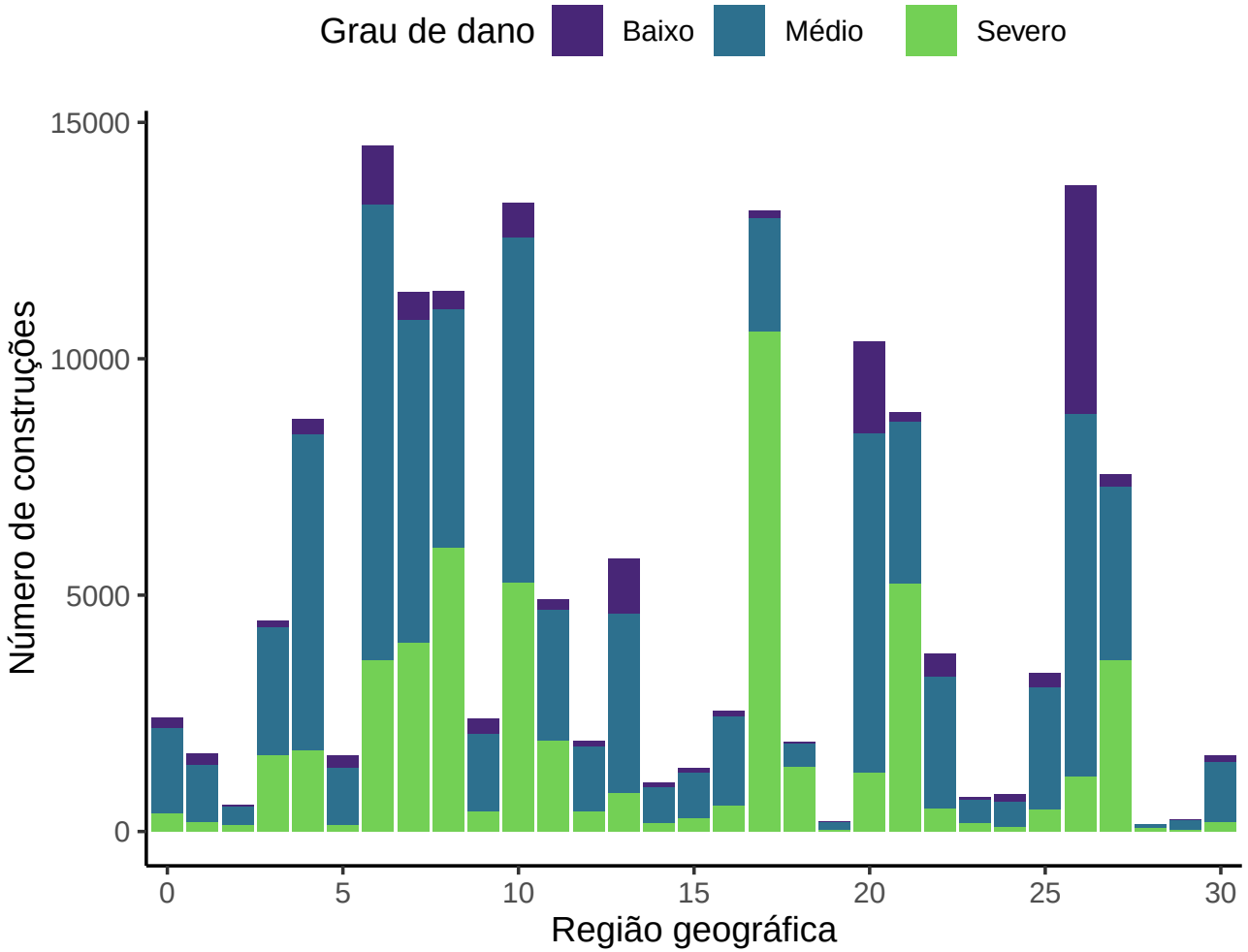
Descrição dos dados

Variáveis geográficas

- **geo_level_1_id**
- **geo_level_2_id**
- **geo_level_3_id**

Região geográfica em que existem construções, do maior agrupamento (nível 1) à sub-região mais específica (nível 3). Possíveis valores: nível 1: 0–30, nível 2: 0–1427, nível 3: 0–12567;

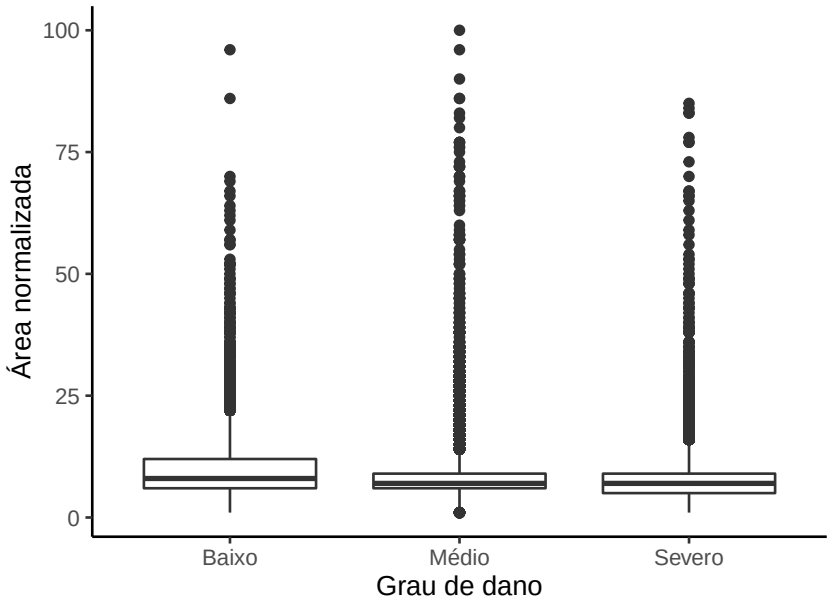
Descrição dos dados

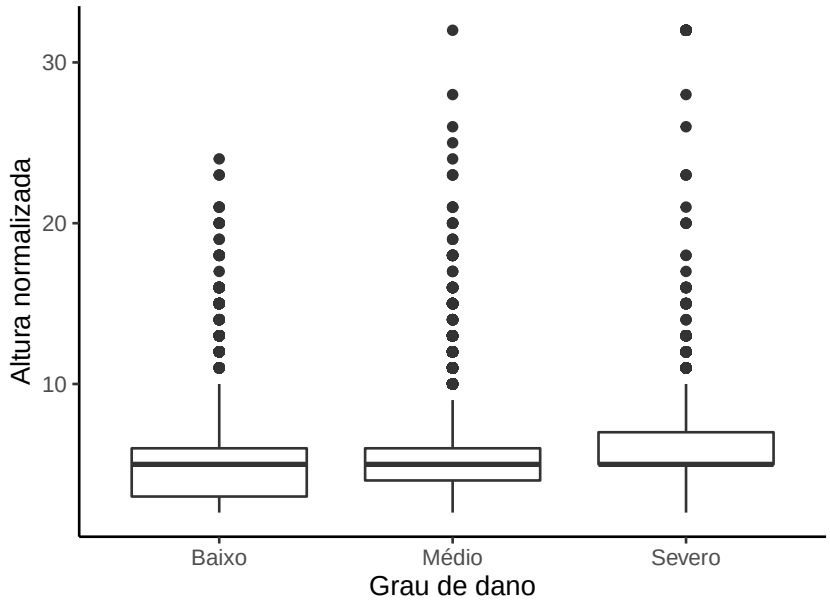


Características da construção

- **land_surface_condition** – Condição da superfície do terreno,
- **area_percentage** – Área (normalizada),
- **height_percentage** – Altura (normalizada),
- **age** – Idade,
- **count_floors_pre_eq** – Número de andares antes do terremoto,
- **foundation_type** – Tipo de fundação,
- **ground_floor_type** – Tipo de andar térreo,
- **other_floor_type** – Tipo de pisos,
- **roof_type** – Tipo de telhado,
- **position** – Posição,
- **plan_configuration** – Configuração do plano de construção,
- **legal_ownership_status** – Status legal,
- **count_families** – Número de famílias.

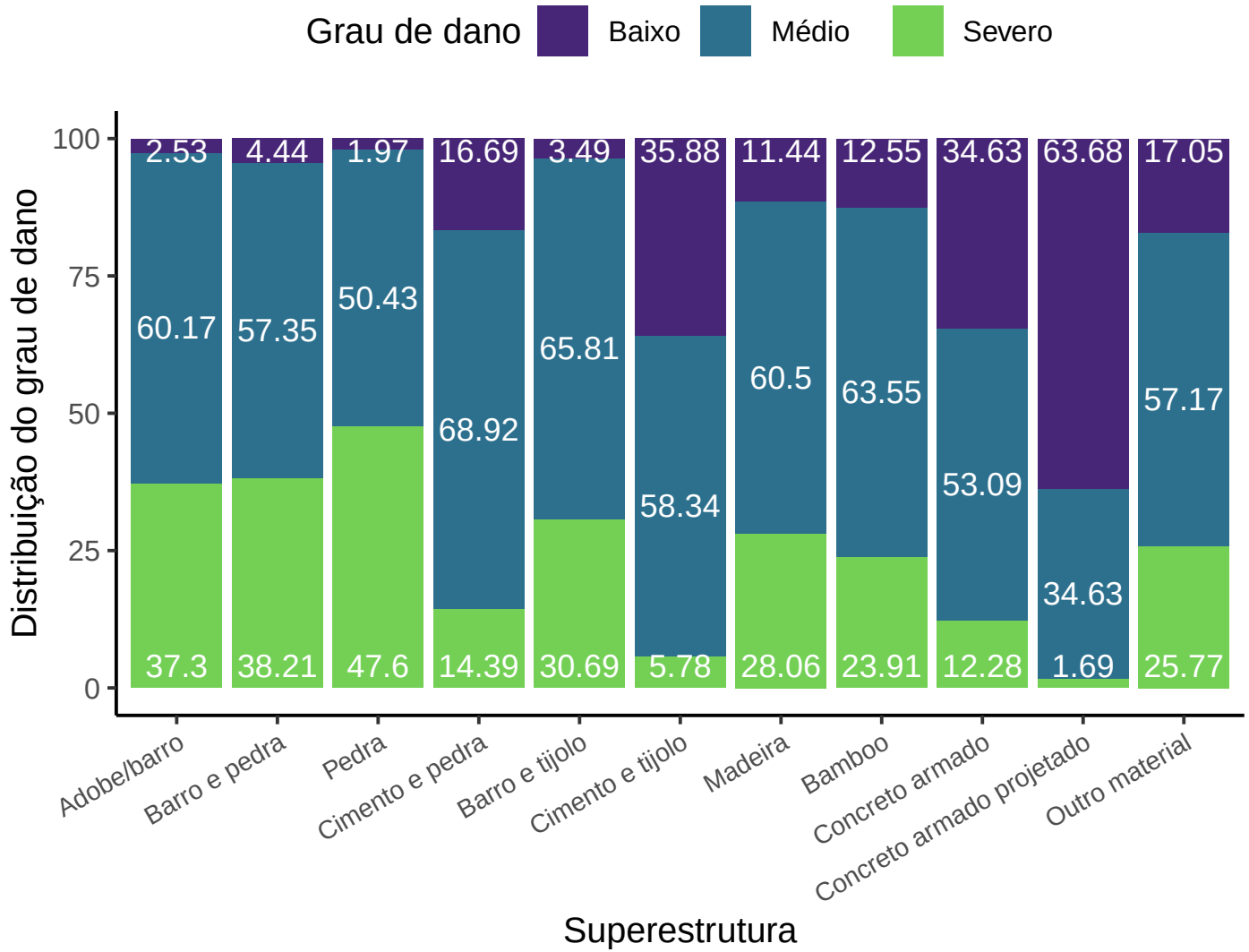
Descrição dos dados





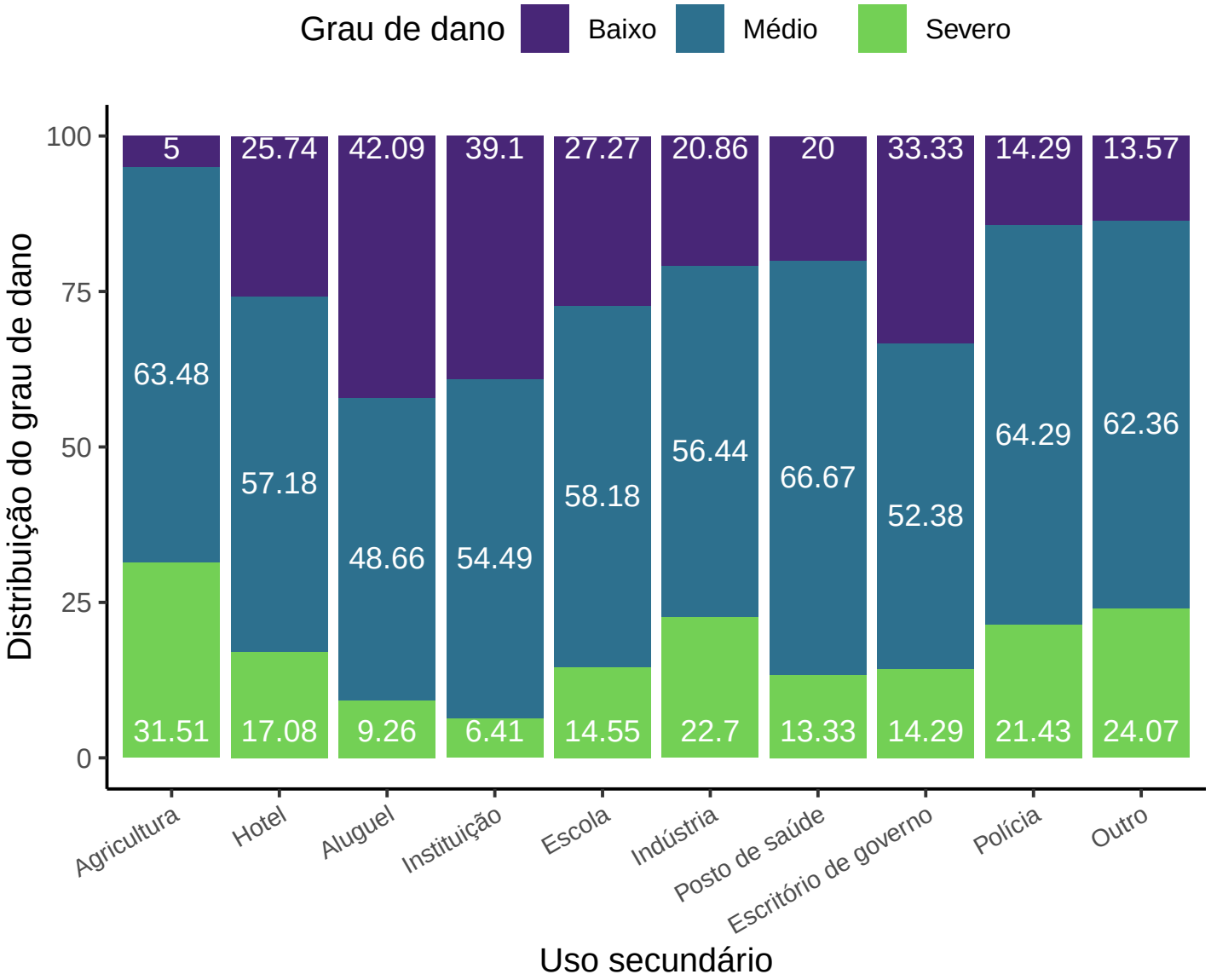
Superestrutura

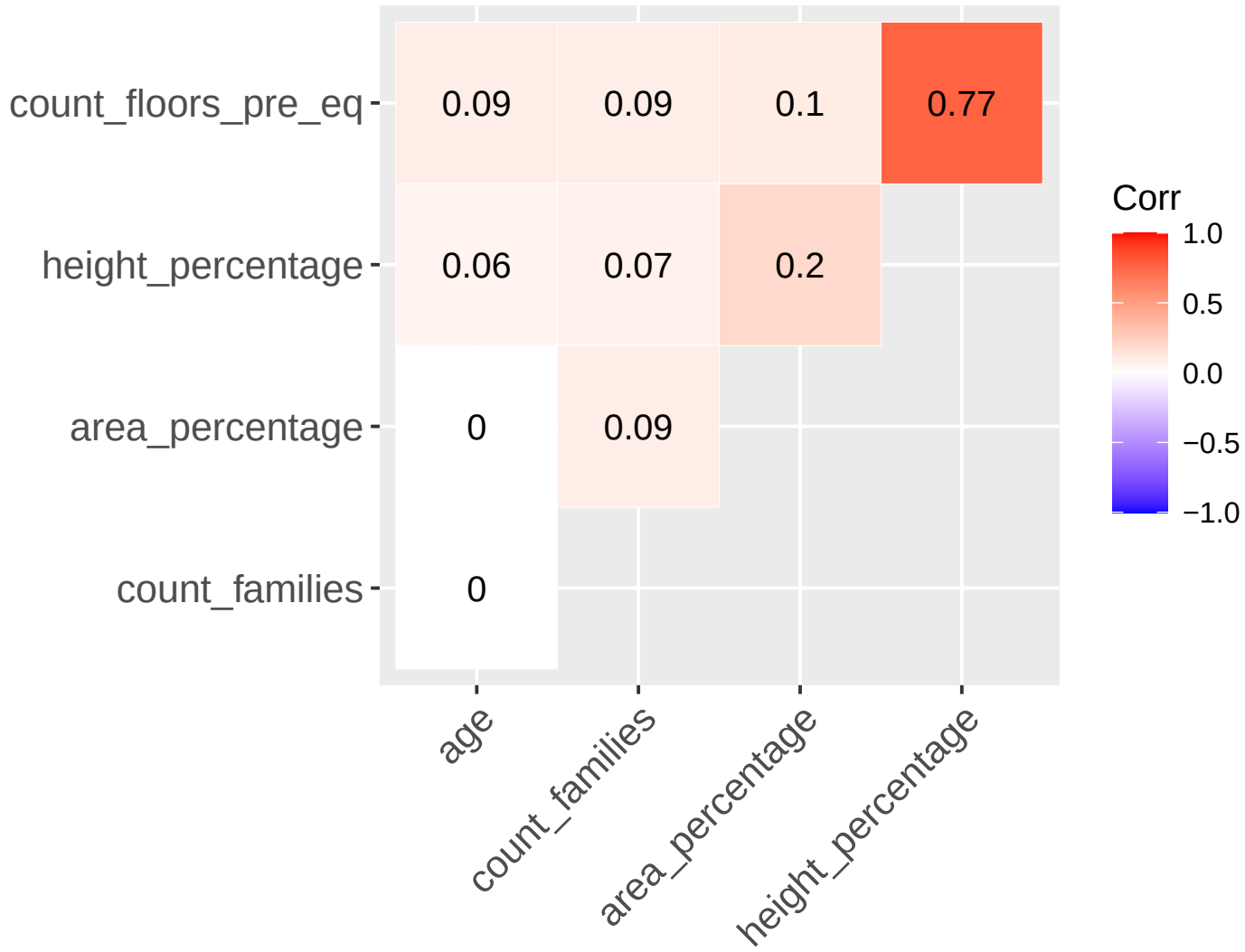
- **has_superstructure_adobe_mud** – Feita de adobe/barro,
- **has_superstructure_mud_mortar_stone** barro e pedra,
- **has_superstructure_stone_flag** – Feita de pedra,
- **has_superstructure_cement_mortar_stone** – Feita de cimento e pedra,
- **has_superstructure_mud_mortar_brick** – Feita de barro e tijolo,
- **has_superstructure_cement_mortar_brick** – Feita de cimento e tijolo,
- **has_superstructure_timber** – Feita de madeira,
- **has_superstructure_bamboo** – Feita de bamboo,
- **has_superstructure_rc_non_engineered** – Feita de concreto armado sem engenharia,
- **has_superstructure_rc_engineered** – Feita de concreto armado projetado,
- **has_superstructure_other** – Feita de outro material.


Grau de dano
Baixo
Médio
Severo
Distribuição do grau de dano
100
75
50
25
0
2.53
4.44
1.97
16.69
3.49
35.88
11.44
12.55
34.63
63.68
17.05
60.17
57.35
50.43
68.92
65.81
58.34
60.5
63.55
53.09
34.63
57.17
37.3
38.21
47.6
14.39
30.69
5.78
28.06
23.91
12.28
1.69
25.77
Adobe/barro
Barro e pedra
Pedra
Cimento e pedra
Barro e tijolo
Cimento e tijolo
Madeira
Bamboo
Concreto armado
Concreto armado projetado
Outro material
Superestrutura

Uso secundário

- **has_secondary_use_agriculture** – Utilizada para fins agrícolas.
- **has_secondary_use_hotel** – Utilizada como um hotel.
- **has_secondary_use_rental** – Utilizada para aluguel.
- **has_secondary_use_institution** – Utilizada como uma instituição.
- **has_secondary_use_school** – Utilizada como uma escola.
- **has_secondary_use_industry** – Utilizada para fins industriais.
- **has_secondary_use_health_post** – Utilizada como um posto de saúde.
- **has_secondary_use_gov_office** – Utilizada como um escritório do governo.
- **has_secondary_use_use_police** – Utilizada como uma delegacia de polícia.
- **has_secondary_use_other** – Utilizada secundariamente para outros fins.


Grau de dano
Baixo
Médio
Severo
Distribuição do grau de dano
100
75
50
25
0
5
25.74
42.09
39.1
27.27
20.86
20
33.33
14.29
13.57
63.48
57.18
48.66
54.49
58.18
56.44
66.67
52.38
64.29
62.36
31.51
17.08
9.26
6.41
14.55
22.7
13.33
14.29
21.43
24.07
Agricultura
Hotel
Aluguel
Instituição
Escola
Indústria
Posto de saúde
Escritório de governo
Polícia
Outro
Uso secundário


count_floors_pre_eq
height_percentage
area_percentage
count_families
0.09
0.09
0.1
0.77
0.06
0.07
0.2
0
0.09
0
age
count_families
area_percentage
height_percentage
Corr
1.0
0.5
0.0
−0.5
−1.0

Modelos

Modelos

- Regressão logística multinomial
- Regressão logística ordinal
- Classificação por árvores
- Máquinas de vetores de suporte (SVM)

Regressão logística multinomial

Regressão logística multinomial

Para a i-ésima construção do conjunto de treinamento, assuma que

$$\log\left\{\frac{\Pr(G_i = k|x = x_i)}{\Pr(G_i = 1|x = x_i)}\right\} = \beta_{k0} + \beta_k^\top x_i, \quad k = 2, 3,$$

em que G_i é o grau de dano sofrido pela i-ésima construção em sua codificação original, $\beta_{20}, \beta_{30} \in \mathbb{R}$, e $\beta_1 = (\beta_{11}, \beta_{12}, \ldots, \beta_{189})^\top \in \mathbb{R}^{89}$ e $\beta_2 = (\beta_{21}, \beta_{22}, \ldots, \beta_{289})^\top \in \mathbb{R}^{89}$ são os coeficientes associados as variáveis explicativas.

Regressão logística multinomial

O modelo é especificado em termos das transformações *logit* em relação a classe 1, de menor dano. Com esta especificação as probabilidades a posteriori são dadas por

$$\Pr(G_i = k|x = x_i) = \frac{\exp\{\beta_{k0} + \beta_k^\top x_i\}}{1 + \sum_{l=2}^{3} \exp\{\beta_{l0} + \beta_l^\top x_i\}}, \quad k = 2, 3$$

e

$$\Pr(G_i = 1|x = x_i) = \frac{1}{1 + \sum_{l=2}^{3} \exp\{\beta_{l0} + \beta_l^\top x_i\}}$$

Regressão logística multinomial

Utilizamos a função `multinom()` do pacote do R `nnet` para ajustar o modelo aos dados do conjunto de treinamento.

```
fit <- multinom(damage_grade ~ ., data = damage)
```

Regressão logística multinomial

A tabela a seguir mostra a matriz de confusão resultante da classificação no conjunto de validação.

		Predição		
		Baixo	Médio	Severo
Real	Baixo	3637	6248	199
	Médio	2376	47963	8943
	Severo	173	16651	18050

Taxa de acertos: 66,82%.

Regressão logística multinomial

A tabela a seguir mostra o percentual de erros por classe no conjunto de validação.

Table: Percentual de erros por classe (%)

Baixo	Médio	Severo
63,93	19,09	48,24

Regressão logística ordinal

Regressão logística ordinal

Aplicamos o modelo ordinal quando o número de categorias da variável é maior que dois e elas são ordenadas. O modelo de logito cumulativo é definido como:

$$logito[\mathrm{P}(Y_i \leq j|x_i)] = log\frac{\mathrm{P}(Y_i \leq j|x_i)}{1-\mathrm{P}(Y_i \leq j|x_i)}, \quad j = 1, ..., c-1$$

Supondo que as variáveis explicativas tenha diferentes efeitos temos o modelo de logito cumulativo sem chances proporcionais:

$$logito[\mathrm{P}(Y_i \leq j|x_i)] = \alpha_j + \boldsymbol{\beta_j'x_i}$$

Regressão logística ordinal

Utilizamos a função `vglm()` do pacote do R `VGAM` para ajustar o modelo aos dados do conjunto de treinamento.

```
fit <- vglm(damage_grade ~ ., data = damage)
```

Regressão logística ordinal

Para o conjunto de validação foi obtida a seguinte matriz de confusão:

		Predição		
		Baixo	Médio	Severo
Real	Baixo	5261	4693	130
	Médio	4357	46771	8154
	Severo	338	17501	17035

Taxa de acertos: 66,17%.

Regressão logística ordinal

O percentual de erros por classe no conjunto de validação.

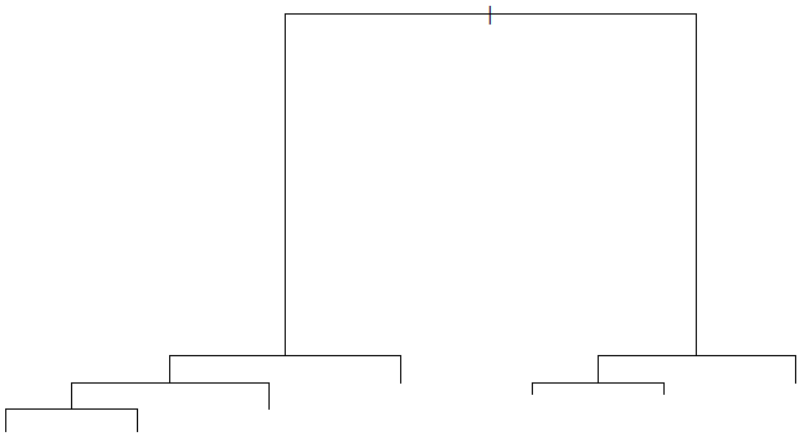
Table: Percentual de erros por classe (%)

Baixo	Médio	Severo
47,82	21,10	51,15

Classificação por árvores

Classificação por árvores

Modelos baseados em árvores têm como ideia central a segmentação do espaço de predição em regiões mais homogêneas, de acordo com a resposta.



Classificação por árvores

Vantagens:

- Podem ser aplicadas em problemas de regressão/classificação;
- Não precisam de variáveis dummy para lidar com preditores qualitativos;
- São fáceis de interpretar; entre outras.

Desvantagens:

- Árvores não são muito robustas, ou seja, uma pequena mudança nos dados pode causar uma grande mudança na estimativa final da árvore.

Possíveis soluções:

- *Bagging*
- *Random Forests*
- *Boosting*

Classificação por árvores - Random Forests

Utilizamos a função `randomForest()` do pacote do R `RandomForest` para ajustar o modelo aos dados.

```
fit.flor <- randomForest(damage_grade ~ .,
data = damage_RF, ntree = 200,
mtry = 8,importance = TRUE)
```

Para tratar o desbalanceamento dos dados: strata e sampsize.

Classificação por árvores - Random Forests

A tabela a seguir mostra a matriz de confusão resultante da classificação no conjunto de validação.

Table: Classificação predita versus valores reais.

		Valores reais					
		Desbalanceado			Balanceado		
		Baixo	Médio	Severo	Baixo	Médio	Severo
Preditos	Baixo	4447	1740	157	7983	8122	1078
	Médio	5494	50792	13727	1921	38991	9103
	Severo	143	6750	20990	180	12169	24693

Taxa de acertos: Desbalanceado = 73,13% e Balanceado = 68,75%.

Classificação por árvores - Random Forests

A tabela a seguir mostra o percentual de erros por classe no conjunto de validação.

Table: Percentual de erros por classe (%)

	Baixo	Médio	Severo
RF: Desbalanceado	55,90	14,32	39,81
RF: Balanceado	20,83	34,23	29,19

Classificação por árvores - Boosting

Utilizamos a função `xgboost()` do pacote do R `XGBoost` para ajustar o modelo aos dados.

XGBoost, que significa Extreme Gradient Boosting, é uma implementação específica do método Gradient Boosting para encontrar o melhor modelo de árvore. Ele tem se mostrado eficiente na solução de diversos problemas e é muito utilizado em competições.

```
xgb.fit <- xgboost(data = features_train,
label = response_train,
eta=0.2, max_depth = 10, min_child_weight = 5, gamma = 1,
subsample = 0.8, colsample_bytree = 0.6,
nrounds = 149, objective = "multi:softmax",
num_class = 4, verbose = 0,)
```

Artigo dos autores originais: https://arxiv.org/abs/1603.02754

Classificação por árvores - Boosting

A tabela a seguir mostra a matriz de confusão resultante da classificação no conjunto de validação.

Table: Classificação predita versus valores reais.

		Predição		
		Baixo	Médio	Severo
Real	Baixo	4563	5388	133
	Médio	1834	50309	7139
	Severo	153	13528	21193

Taxa de acertos:72,97%.

Classificação por árvores - Boosting

A tabela a seguir mostra o percentual de erros por classe no conjunto de validação.

Table: Percentual de erros por classe (%)

Baixo	Médio	Severo
54,75	15,14	39,23

Máquinas de vetores de suporte (SVM)

SVM

Utilizamos a implementação do pacote `e1071` para treinar uma máquina de vetores de suporte sobre os dados.

Durante a etapa de pré processamento, as variáveis de entrada númericas foram normalizadas linearmente no intervalo [0,1] e as variáveis preditoras do tipo categoricas foram transformadas em dummy; para o treinamento, utilizamos um Kernel do tipo `radial basis function` e uma estratégia de multiclassificação por votos: treinamos simultaneamente 3 problemas de classificação binários e a classificação final foi feita para a classe com maior incidencia na resposta.

Em função da complexidade temporal do algoritmo, a primeira etapa de grid-search dos hiperparâmetros foi feita sobre um subconjunto de 10K; em seguida, uma busca refinada foi feita sobre um conjunto maior de 20K amostras. Utilizou-se um estratégia de validação cruzada de 5-fold para ambos.

A tabela a seguir mostra a matriz de confusão resultante da classificação no conjunto de validação.

		Predição		
		Baixo	Médio	Severo
Real	Baixo	4024	5742	162
	Médio	2295	48172	8922
	Severo	184	15049	19690

Taxa de acertos: 68,96%.

Desempenho dos modelos na competição

Desempenho dos modelos na competição

A tabela a seguir mostra o desempenho de cada modelo na competição.

Table: Resultados por método aplicado.

Modelos	Conjunto de Teste - Competição(%)
Regressão Logística Multinomial	66,90
Regressão Logística Ordinal	66,22
Random Forests: Desbalanceado	72,90
Random Forests: Balanceado	65,87
Gradient Boosting	72,26
SVM Kernel Radial	68,82

- O melhor desempenho foi obtido, sem tratar o desbalanceamento dos dados, utilizando Random Forests.
- A competição conta com 3563 participantes, nós estamos na posição 456.
- O líder atual obteve uma acurácia de 75,58%.

Obrigado!