PROJETO DE MACHINE LEARNING - Expansão para outra cidade

Expansão de uma determinada empresa para São Paulo

Objetivos

Atualmente, uma determinada Empresa está presente na cidade do Rio de Janeiro, com 160 lojas, e quer expandir seus negócios na cidade de São Paulo, tendo como público alvo adultos de 25 a 50 anos, das classes A e B. Com dados do Rio de Janeiro e de São Paulo previamente obtidos do time de Engenharia de Dados , traçamos aqui três objetivos que devem ser alcançados para São Paulo na seguinte ordem:

- Estimar o faturamento que uma loja teria em cada um dos bairros;
- Classificar o potencial de cada bairro como Alto, Médio ou Baixo;
- Segmentar os bairros de São Paulo de acordo com a renda e a idade, e indicar aqueles com maior aderência ao público alvo.

Organização da apresentação

Esta apresentação está dividida nos seguintes tópicos gerais:

- Descrição dos dados
- Pré-processamento dos dados
- Construção do Modelo Preditor de faturamento
- Construção do Modelo Preditor de potencial
- Segmentação dos bairros de São Paulo

Descrição dos dados

Os dados estão dispostos, originalmente, numa tabela com colunas informando códigos dos bairros, nomes dos bairros, várias populações por faixa etária, várias contagem de domicílios por classe social e renda média, sendo cada linha representando um bairro. As últimas colunas são as informações que mais nos interessam, pois os bairros do Rio de Janeiro estão com elas preenchidas e os de São Paulo pretendemos preencher. Segue tabela com parte dos dados para nos dar essa noção:

	codigo	nome	cidade	estado	população	popAte9	popDe10a14	popDe15a19	popDe20a24	popDe25a34	...	domiciliosA2	domiciliosB1	domiciliosB2	domiciliosC1	domiciliosC2	domiciliosD	domiciliosE	rendaMedia	faturamento	potencial
0	3.304557e+09	Abolição	Rio de Janeiro	RJ	11676.0	1027.0	483.0	688.0	800.0	1675.0	...	145.0	715.0	1242.0	1093.0	758.0	92.0	304.0	2501.0	932515.0	Médio
1	3.304557e+09	Acari	Rio de Janeiro	RJ	27564.0	5131.0	2188.0	2697.0	2630.0	4810.0	...	0.0	82.0	506.0	2040.0	2490.0	827.0	2506.0	931.0	588833.0	Baixo
2	3.304557e+09	Água Santa	Rio de Janeiro	RJ	9003.0	883.0	399.0	597.0	762.0	1755.0	...	96.0	404.0	652.0	644.0	522.0	77.0	254.0	2391.0	874200.0	Baixo
3	3.304557e+09	Alto Da Boa Vista	Rio de Janeiro	RJ	9606.0	1072.0	538.0	660.0	685.0	1317.0	...	178.0	393.0	517.0	945.0	584.0	137.0	286.0	3727.0	912226.0	Médio
4	3.304557e+09	Anchieta	Rio de Janeiro	RJ	57222.0	7677.0	3774.0	4892.0	4600.0	8660.0	...	0.0	1089.0	2821.0	5110.0	5422.0	1073.0	3261.0	1380.0	553020.0	Médio
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
451	3.550302e+08	Vila Sônia	São Paulo	SP	34061.0	2908.0	1462.0	2253.0	2076.0	4579.0	...	1089.0	2866.0	2219.0	2216.0	1227.0	306.0	1098.0	5285.0	NaN	NaN
452	3.550302e+08	Vila Suzana	São Paulo	SP	35403.0	4127.0	1890.0	2678.0	2433.0	5855.0	...	1522.0	2458.0	1186.0	1166.0	918.0	209.0	3840.0	7418.0	NaN	NaN
453	3.550302e+08	Vila Terezinha	São Paulo	SP	122359.0	18304.0	9304.0	13258.0	9965.0	19248.0	...	0.0	1758.0	4517.0	9450.0	11473.0	3218.0	7540.0	1252.0	NaN	NaN
454	3.550302e+08	Vila Zatt	São Paulo	SP	125864.0	14670.0	7305.0	11225.0	9338.0	18841.0	...	872.0	5093.0	8063.0	10012.0	8082.0	2856.0	6853.0	1936.0	NaN	NaN
455	3.550302e+08	Vista Alegre	São Paulo	SP	106.0	23.0	7.0	15.0	11.0	15.0	...	0.0	0.0	6.0	9.0	8.0	2.0	4.0	1288.0	NaN	NaN

456 rows × 24 columns

Pré-processamento dos dados

Separamos os dados em dois conjuntos: bairros do Rio de Janeiro e bairros de São Paulo. Assim, podemos utilizar somente os do Rio como dados prévios para tentarmos gerar um modelo preditor de valores de faturamento e outro de nível de potencial a serem utilizados para prevermos, com a máxima aproximação, os valores de faturamento e potencial de São Paulo que estão vazios. Segue algumas informações de cada conjunto e soluções para anomalias encontradas:

- Bairros do Rio de Janeiro:
  - São 160 bairros;
  - Campos vazios em 6 bairros na coluna de renda média. Foram preenchidos com o valor médio de todos os outros bairros.
- Bairros de São Paulo:
  - São 296 bairros;
  - Campos vazios em 3 bairros na coluna de renda média. Foram preenchidos com o valor médio de todos os outros bairros.
  - Campos vazios para todos os bairros nas colunas de faturamento e potencial. Serão preenchidos pelos respectivos modelos preditores a serem gerados com os dados do Rio de Janeiro.

Construção do Modelo Preditor de faturamento

No conjunto de dados do Rio de janeiro, as colunas numéricas de **populações por faixa etária**, de **contagens por classe social** e de **renda média** foram separadas para serem nossas variáveis preditoras, mais conhecidas pelo termo em inglês ***features***, enquanto a coluna de **faturamento** foi nossa variável alvo ou rótulo, mais conhecida pelo termo em inglês ***target*** ou ***label***. Como nosso *label* é também numérico contínuo, temos um problema de regressão, no qual isso definirá a escolha dos tipos de algoritmos de aprendizagem de máquina (mais conhecido pelo termo em inglês ***machine learning***).

O desenvolvimento do nosso modelo preditor de faturamento teve quatro momentos:

1. Dados de treino/validação;
    a. Validação cruzada
    b. Métrica utilizada
    c. Engenharia de features
2. Escolha do modelo;
3. Melhoria do modelo;
4. Predição de faturamento dos bairros de SP.

(...) Construção do Modelo Preditor de faturamento

1. Dados de treino/validação

No **treinamento**, todo algoritmo de aprendizagem (modelo preditor) precisa dos dados das *features* e do *label* para ajustar sua estrutura interna. Na **validação**, após esse ajuste, o modelo faz um auto-teste com outros dados, predizendo valores novos de *label* e comparando com o *label* real usando a **métrica adequada** para ter ideia do quanto os resultados estão longe do real. Esses dois processos são executados repetidamente até que se tenha a máxima aproximação do *label* predito com o *label* real. E, para isso, é utilizado dois tipos de conjuntos de dados da mesma base: conjunto de treinamento e conjunto de teste.

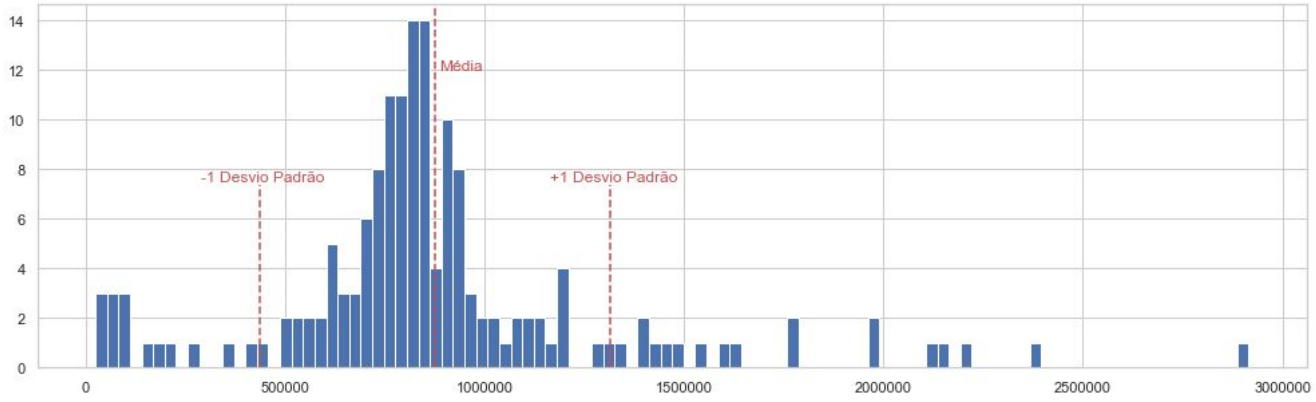
a) Validação Cruzada

Em nosso modelo, usamos a **validação cruzada**, onde dividimos nossa base de dados (bairros do RJ) em três (não mais que isso por causa do tamanho da amostra que é de 160, onde não é aconselhável que as sub-amostras tenham menos que 50). Então, uma das sub-amostras é usada para treino e qualquer outra das duas restantes é usada para teste. Assim é feito para todas as combinações entre as três sub-amostras, **obtendo três pontuações da métrica escolhida, sendo a pontuação final a média entre elas**. O **desvio padrão** delas mostra o quanto nosso modelo está **enviesado** (quanto maior, mais enviesado).

(...) Construção do Modelo Preditor de faturamento

b) Métrica utilizada

Para escolhermos a métrica mais adequada, nos baseamos na distribuição dos dados do *label*, onde fizemos um teste estatístico para sabermos o quanto essa distribuição se aproxima do normal. O resultado está no histograma de faturamento abaixo:




```
Média: 876159.96875
Desvio Padrão: 438210.5955586063
Teste de aproximação da distribuição normal: p-value = 1.0857201332109742e-08, os dados NÃO VIERAM de uma distribuição normal.
```

(...) Construção do Modelo Preditor de faturamento

b) (...) Métrica utilizada

Como a distribuição não se aproxima da normal, segundo o teste, então qualquer métrica baseada em média não seria adequada, pois para dados de uma distribuição desconhecida a mediana é mais usual. Outro fator que contribuiu para a escolha foi o tipo de dados do *label*, onde dispomos de valores numéricos contínuos e de alta magnitude (milhares e milhões). Portanto, escolhemos a **métrica Erro Absoluto Mediano** que também nos dá valores calculados na mesma escala dos valores reais (neste caso, na moeda local).

c) Engenharia de *features*

Para problemas de regressão, pelo qual nosso modelo responde, podemos trabalhar com duas transformações dos dados: **remoção de *outliers*** (dados extremos, muito longe da média que podem interferir negativamente no resultado) e **seleção de features** (selecionar apenas as mais importantes, descartando as demais que podem não causar efeito algum no resultado).

(...) Construção do Modelo Preditor de faturamento

c) (...) Engenharia de *features*

Cada uma dessas transformações podem gerar diferentes resultados para diferentes algoritmos preditores (algoritmos de aprendizado de máquina). Portanto, utilizamos quatro combinações no treino/validação dos modelos:

- SEM remoção de *outliers* e SEM seleção de features;
- COM remoção de *outliers* e SEM seleção de features;
- SEM remoção de *outliers* e COM seleção de features;
- COM remoção de *outliers* e COM seleção de features.

Na remoção de *outliers*, 18 bairros ficam de fora. Na seleção de *features*, somente a contagem de domicílios da classe social A1 fica de fora.

(...) Construção do Modelo Preditor de faturamento

2. Escolha do modelo

Utilizamos dez algoritmos diferentes de aprendizado de máquina para problemas de regressão. São eles: **LinearRegression**, **LGBMRegressor**, **XGBRegressor**, **CatBoostRegressor**, **SGDRegressor**, **KernelRidge**, **ElasticNet**, **BayesianRidge**, **GradientBoostingRegressor** e **SVR** (todos compatíveis com o módulo de aprendizagem de máquina chamado **SKLearn** da linguagem **Python**). Inicialmente, **cada algoritmo** foi treinado/validado com **cada combinação de transformação de dados**, mantendo **valores *defaults* em seus parâmetros de ajustes**, lembrando também que a métrica utilizada foi o **erro absoluto mediano** para todos eles (quanto menor, melhor).

Após todos esses treinos/validações, escolhemos os três melhores modelos, ou seja aqueles que tiveram os menores valores de erro absoluto mediano que foram:

- **BayesianRidge com remoção de outliers e sem seleção de features** (*score* = 5574.52 / desvio padrão entre os *k-folds* = 2846.44);
- **ElasticNet com remoção de outliers e sem seleção de features** (*score* = 5986.56 / desvio padrão entre os *k-folds* = 3290.46);
- **LinearRegression com remoção de outliers e sem seleção de features** (*score* = 6108.64 / desvio padrão entre os *k-folds* = 3315.03)

Onde, *score* é o erro absoluto mediano e *k-fold* são as sub-amostras de alimentação para o teste (no caso, são *3-folds*).

(...) Construção do Modelo Preditor de faturamento

2. (...) Escolha do modelo

A tabela abaixo mostra em cada célula o *score* e o desvio padrão dos *k-folds*, respectivamente. Foi dela, tiramos os melhores modelos até agora:

Combinações Algoritmo/Modelo	('Sem remoção de outliers', 'Sem seleção de features')	('Sem remoção de outliers', 'Com seleção de features')	('Com remoção de outliers', 'Sem seleção de features')	('Com remoção de outliers', 'Com seleção de features')
LinearRegression	(47299.52301915737, 18384.54592118613)	(47905.3722782589, 17806.069008516253)	(6108.642664032716, 3315.0293478795893)	(8387.664583551581, 2639.513816218934)
LGBMRegressor	(69268.31248086599, 24965.80421964242)	(63267.3894131263, 17099.655184806834)	(66002.0113671649, 22810.653392020886)	(64153.76508512802, 24206.399604459137)
XGBRegressor	(35513.4375, 6762.195608522181)	(38295.247395833336, 6041.0412579150025)	(41234.083333333336, 10006.231045205519)	(37932.28125, 9293.759357797631)
CatBoostRegressor	(28890.07168762677, 5081.6038009704835)	(36841.787319740455, 13956.172153689628)	(26341.9840948319, 10881.95767421699)	(27818.790652885276, 11847.4333976155)
SGDRegressor	(2.3926010443790157e+18, 1.3187792100560366e+18)	(1.7592379244772444e+18, 9.629800730136462e+17)	(9.658513867579483e+17, 7.054574543196641e+17)	(8.884532719065962e+17, 6.636764413076756e+17)
KernelRidge	(431025.3333333333, 78935.67495180632)	(436898.6666666667, 73816.03090406959)	(340943.4270833333, 26135.773020276338)	(341325.75, 22845.07380181593)
ElasticNet	(40328.09982287484, 7801.05597136278)	(39785.26338614776, 9214.467322320255)	(5986.561037349379, 3290.460703661734)	(8128.487669548388, 2211.3098407562147)
BayesianRidge	(45225.9328044795, 14485.337070351245)	(42272.775866912525, 18863.772104493048)	(5574.519272962294, 2846.445072780905)	(7732.759539340604, 1865.9905922364715)
GradientBoostingRegressor	(45264.93778302472, 8950.430879795043)	(43202.96058866663, 10073.681767367034)	(33934.72572857045, 7288.662073390121)	(33866.92840819901, 10773.284226492224)
SVR	(130368.57941864029, 41224.76558702446)	(130368.57250034098, 41224.764919693494)	(108811.78681238448, 28396.056932375694)	(108811.79382192211, 28396.05491145474)

(...) Construção do Modelo Preditor de faturamento

3. Melhoria do modelo

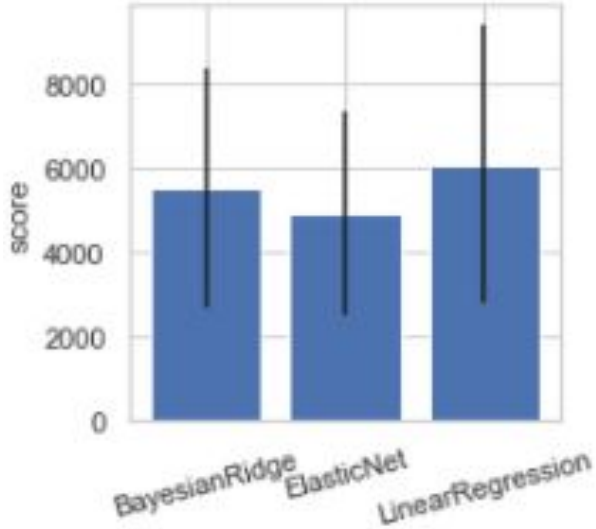
Para escolhermos nosso modelo fina, tivemos que trabalhar na melhoria dos três modelos escolhidos antes na etapa de treino/validação **SEM ajustes de parâmetros**. Obviamente, o treino/validação **COM ajuste de parâmetros** foi nossa próxima etapa. Cada algoritmo tem suas particularidades quando se fala de parâmetros. Portanto, não vamos falar detalhes, mas somente que testamos vários valores para cada um dos parâmetros de cada algoritmo escolhido previamente. Alguns algoritmos não possuem parâmetros para serem ajustados, sendo o caso do **LinearRegression** (mantendo o mesmo *score* e desvio padrão de antes). Após ajustes dos parâmetros (quando coube ajustar), os resultados foram:

- **BayesianRidge com remoção de outliers e sem seleção de features** (*score* = 5570.51 / desvio padrão entre os *k-folds* = 2851.92);
- **ElasticNet com remoção de outliers e sem seleção de features** (*score* = 4948.37 / desvio padrão entre os *k-folds* = 2432.92);
- **LinearRegression com remoção de outliers e sem seleção de features** (*score* = 6108.64 / desvio padrão entre os *k-folds* = 3315.02)

(...) Construção do Modelo Preditor de faturamento

3. (...) Melhoria do modelo

Para escolhermos o melhor entre eles, devemos olhar para o menor *score* e também o menor desvio padrão. Segue um gráfico com os resultados (a linha preta representa a variação / desvio padrão):



(...) Construção do Modelo Preditor de faturamento

3. (...) Melhoria do modelo

Após ajustes, o melhor modelo foi **ElasticNet COM remoção de outliers e SEM seleção de features** por obter o melhor score, em torno de 4948.0, e melhor desvio padrão, em torno de 2433.0, tendo como pior caso a soma deles, em torno de 7381.0 . Isso significa dizer que cada valor de faturamento a ser predito por esse modelo tem um erro máximo aproximado de R$ 7381.00 do valor real.

4. Predição de faturamento dos bairros de SP

Com o modelo vencedor, previamente escolhido e ajustado, bastamos utilizá-lo para prever os valores de faturamento dos bairros de São Paulo que estavam com campos vazios. Não tivemos que remover nenhuma *feature* para fazer essas predições, pois o modelo vencedor não contempla a seleção de *features*.

(...) Construção do Modelo Preditor de faturamento

4. (...) Predição de faturamento dos bairros de SP

Segue tabela mostrando alguns dos bairros de SP com valores de faturamento preenchidos pela predição do modelo vencedor (veja que ainda temos que construir outro modelo preditor para preencher os campos de potencial):

	codigo	nome	cidade	estado	população	popAte9	popDe10a14	...	domiciliosC2	domiciliosD	domiciliosE	rendaMedia	faturamento	potencial
160	355030251.0	A. E. Carvalho	São Paulo	SP	94034.0	12668.0	6853.0	...	7011.0	2247.0	5670.0	1501.0	3.869801e+05	NaN
161	35503020.0	Aclimação	São Paulo	SP	32791.0	2297.0	1017.0	...	827.0	291.0	1617.0	5920.0	1.425917e+06	NaN
162	355030285.0	Adventista	São Paulo	SP	104193.0	15070.0	7343.0	...	10082.0	3111.0	5776.0	1284.0	4.027942e+04	NaN
163	35503088.0	Água Branca	São Paulo	SP	12721.0	953.0	343.0	...	361.0	84.0	404.0	6278.0	1.118423e+06	NaN
164	35503066.0	Água Funda	São Paulo	SP	48417.0	5078.0	2396.0	...	2836.0	1104.0	2553.0	1905.0	8.622022e+05	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
451	355030213.0	Vila Sônia	São Paulo	SP	34061.0	2908.0	1462.0	...	1227.0	306.0	1098.0	5285.0	1.275856e+06	NaN
452	355030207.0	Vila Suzana	São Paulo	SP	35403.0	4127.0	1890.0	...	918.0	209.0	3840.0	7418.0	1.294612e+06	NaN
453	355030162.0	Vila Terezinha	São Paulo	SP	122359.0	18304.0	9304.0	...	11473.0	3218.0	7540.0	1252.0	-1.242247e+05	NaN
454	355030157.0	Vila Zatt	São Paulo	SP	125864.0	14670.0	7305.0	...	8082.0	2856.0	6853.0	1936.0	8.135220e+05	NaN
455	355030164.0	Vista Alegre	São Paulo	SP	106.0	23.0	7.0	...	8.0	2.0	4.0	1288.0	8.446202e+05	NaN

296 rows × 24 columns

(...) Construção do Modelo Preditor de faturamento

4. (...) Predição de faturamento dos bairros de SP

Os 15 bairros de São Paulo que mais fatururiam e os que dariam prejuízos segundo essa predição são mostrados nas tabelas abaixo:

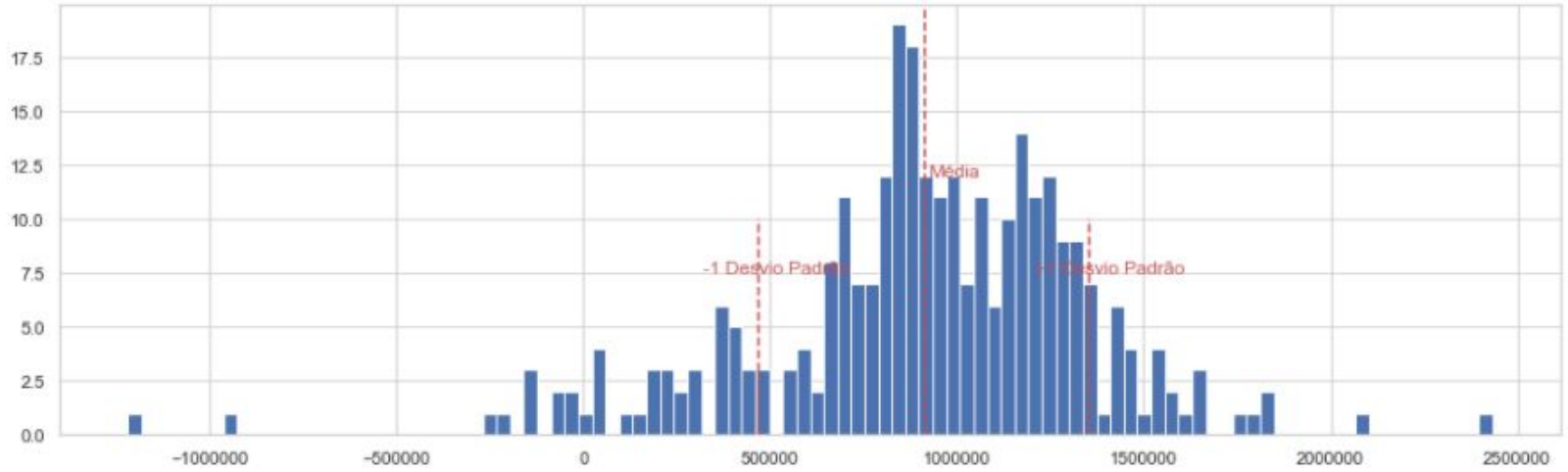
nome	faturamento
Moema	2432663.586151
Perdizes	2084683.368790
Vila Mariana	1834717.809247
Trianon	1829816.103885
Saúde	1789779.432175
Chácara Itaim	1749623.387331
Tatuapé	1656255.681506
Pamplona	1655702.854914
Paraíso	1637120.894908
Alfredo Pujol	1619215.182981
Brooklin	1568164.052744
Mirandópolis	1567158.657872
Jardim São Paulo	1554178.600789
Viera De Morais	1552727.833512
Marechal Deodoro	1528687.189636

nome	faturamento
Cocaia	-1219952.057492
Cidade Tiradentes	-951139.394129
Jardim Capela	-247806.864573
Iguatemi	-217202.261220
Parque Fernanda	-132244.156474
Parada De Taipas	-131784.823984
Vila Terezinha	-124224.727285
Centro Empresarial	-55532.661134
Fazenda Itaim	-51682.094639
Grajaú	-46823.077194
Jardim Miriam	-44156.936126
M'Boi Mirim	-1706.963284

(...) Construção do Modelo Preditor de faturamento

4. (...) Predição de faturamento dos bairros de SP

A distribuição dos dados de faturamento em São Paulo ficou assim:




```
Média: 911661.6456704597
Desvio Padrão: 442376.1491038962
Teste de aproximação da distribuição normal: p-value = 1.9173614362354498e-07, os dados NÃO VIERAM de uma distribuição normal.
```

Construção do Modelo Preditor de potencial

Para esse próximo modelo, as mesmas *features* utilizadas no preditor de faturamento também foram utilizadas para preverem o potencial. Porém, a própria coluna de faturamento também virou *feature*, pois já dispomos do faturamento preenchido para os bairros de São Paulo. Para esse novo modelo, nosso *label* foi, obviamente, a coluna de potencial.

Assim como no preditor de faturamento, esse novo preditor teve os mesmos quatro momentos. Vamos passar por cada um.

1. Dados de treino/validação

a) Validação Cruzada

Em nosso modelo, utilizamos a **validação cruzada** da mesma forma que no preditor de faturamento.

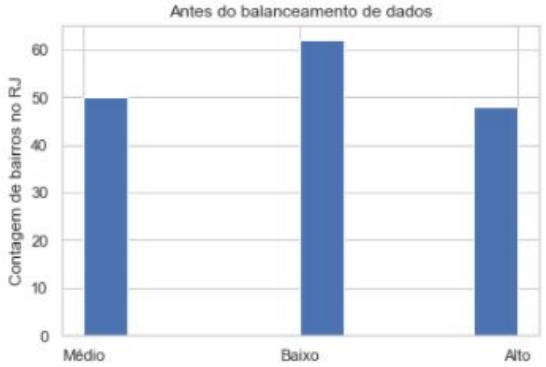
b) Métrica utilizada

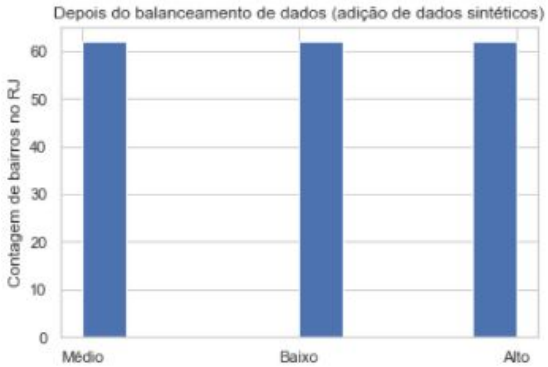
Sendo um problema de classificação (valores do *label* são categóricos), onde as classes de potencial são **Alto**, **Médio** e **Baixo** , utilizamos a métrica mais simples, considerada universal: a **acurácia** (conhecida popularmente e erroneamente por precisão, pois, tecnicamente, esta última tem outro conceito). A acurácia mostra o percentual de acertos de valores preditos em relação aos valores reais.

(...) Construção do Modelo Preditor de potencial

c) **Engenharia de *features***

Para problemas de classificação, pelo qual esse novo modelo responde, podemos trabalhar com três transformações dos dados: **balanceamento de dados** (equilíbrio entre as contagens de bairros para cada nível de potencial), **remoção de *outliers*** e **seleção de features.** A exemplo do preditor de faturamento, fizemos as combinações entre essas transformações de dados no treino/validação dos modelos. Falando mais sobre o balanceamento de dados, utilizamos um algoritmo baseado em um método chamado *SMOTE* para gerar dados sintéticos (bairros e seus dados gerados de forma fictícia, mas usando os dados dos bairros reais como referência), e equilibrar os níveis de potenciais. Os gráficos a seguir mostram antes e depois do balanceamento, respectivamente:





(...) Construção do Modelo Preditor de potencial

2. Escolha do modelo

Utilizamos onze algoritmos diferentes de aprendizado de máquina para problemas de classificação. São eles: **LogisticRegression**, **SVC**, **GaussianNB**, **MultinomialNB**, **SGDClassifier**, **KNeighborsClassifier**, **DecisionTreeClassifier**, **RandomForestClassifier**, **GradientBoostingClassifier**, **LGBMClassifier**, **XGBClassifier** (todos também compatíveis com o módulo **SKLearn** da linguagem **Python**). Seguimos o mesmo processo do preditor de faturamento, com exceção da métrica utilizada que dessa vez foi a **acurácia** (valores entre 0 e 1, ou entre 0% e 100%, quanto maior, melhor). Após todos os treinos/validações, escolhemos os quatro melhores modelos:

- **RandomForestClassifier_1 com balanceamento de dados, sem remoção de outliers e com seleção de features** (score = 0.8978 / desvio padrão entre os k-folds = 0.0331);
- **RandomForestClassifier_2 com balanceamento de dados, sem remoção de outliers e sem seleção de features** (score = 0.8924 / desvio padrão entre os k-folds = 0.0076);
- **RandomForestClassifier_3 sem balanceamento de dados, sem remoção de outliers e com seleção de features** (score = 0.8876 / desvio padrão entre os k-folds = 0.0261);
- **RandomForestClassifier_4 sem balanceamento de dados, sem remoção de outliers e sem seleção de features** (score = 0.8874 / desvio padrão entre os k-folds = 0.0271).

Observe que o algoritmo foi o mesmo para todos os modelos, diferenciando na combinação de transformações dos dados. Para não confundir, resolvemos numerar o nome do algoritmo de cada modelo.

(...) Construção do Modelo Preditor de potencial

2. (...) Escolha do modelo

A tabela abaixo mostra em cada célula o *score* e o desvio padrão dos *k-folds*, respectivamente, de onde tiramos os melhores modelos:

	('Sem balanceamento de dados', 'Sem remoção de outliers', 'Sem seleção de features')	('Sem balanceamento de dados', 'Sem remoção de outliers', 'Com seleção de features')	('Sem balanceamento de dados', 'Com remoção de outliers', 'Sem seleção de features')	('Sem balanceamento de dados', 'Com remoção de outliers', 'Com seleção de features')	('Com balanceamento de dados', 'Sem remoção de outliers', 'Sem seleção de features')	('Com balanceamento de dados', 'Sem remoção de outliers', 'Com seleção de features')	('Com balanceamento de dados', 'Com remoção de outliers', 'Sem seleção de features')	('Com balanceamento de dados', 'Com remoção de outliers', 'Com seleção de features')
LogisticRegression	(0.7874446773817843, 0.039019264200233314)	(0.7811553692056837, 0.03261829812538992)	(0.75177304964539, 0.03616325896165096)	(0.75177304964539, 0.026536577211162704)	(0.8225806451612904, 0.047482585302837894)	(0.8225806451612904, 0.04561979233461598)	(0.8147186147186147, 0.043532599692601905)	(0.8147186147186147, 0.043532599692601905)
SVC	(0.5687165152573957, 0.0310432159135015)	(0.5687165152573957, 0.0310432159135015)	(0.524822695035461, 0.043719248248006944)	(0.524822695035461, 0.043719248248006944)	(0.5698924731182795, 0.030413194889744005)	(0.5698924731182795, 0.030413194889744005)	(0.5570346320346321, 0.018980474926625156)	(0.5570346320346321, 0.018980474926625156)
GaussianNB	(0.6183321686466341, 0.04971916425439687)	(0.6182156999767062, 0.06093793963217427)	(0.6737588652482268, 0.053073154422325394)	(0.673758865248227, 0.0610093990570399)	(0.6559139784946237, 0.06496261276126113)	(0.6451612903225806, 0.06968538385384798)	(0.7547619047619047, 0.04328490229213775)	(0.7308441558441557, 0.04216031295579943)
MultinomialNB	(0.5313300722105754, 0.0248206731981784)	(0.5313300722105754, 0.0248206731981784)	(0.5177304964539007, 0.020059766842171565)	(0.5177304964539008, 0.03616325896165096)	(0.521505376344086, 0.015206597444871977)	(0.521505376344086, 0.015206597444871977)	(0.5271645021645021, 0.026800085785585216)	(0.5091991341991342, 0.026652836268451822)
SGDClassifier	(0.29990682506405775, 0.013055924908074805)	(0.29990682506405775, 0.013055924908074805)	(0.39716312056737585, 0.08569536151485511)	(0.39716312056737585, 0.08569536151485511)	(0.3440860215053763, 0.020116437563300768)	(0.3440860215053763, 0.020116437563300768)	(0.3170995670995671, 0.03137595024745505)	(0.3170995670995671, 0.03137595024745505)
KNeighborsClassifier	(0.6688795713952946, 0.048485950655425304)	(0.6688795713952946, 0.048485950655425304)	(0.6595744680851063, 0.05211680303793999)	(0.6595744680851063, 0.05211680303793999)	(0.6881720430107526, 0.020116437563300792)	(0.6881720430107526, 0.020116437563300792)	(0.6706709956709958, 0.02191872405494208)	(0.6706709956709958, 0.02191872405494208)
DecisionTreeClassifier	(0.7812718378756115, 0.008089333964213997)	(0.79979035639413, 0.033114400417046774)	(0.7588652482269503, 0.01002988342108579 5)	(0.75177304964539, 0.053073154422325394)	(0.8440860215053764, 0.06223568227306572)	(0.8172043010752689, 0.06496261276126113)	(0.8265151515151515, 0.041491028024987976)	(0.8324675324675325, 0.04644332705310684)
RandomForestClassifier	(0.8873747961798276, 0.027104439125468387)	(0.8876077335196833, 0.026125872390777464)	(0.8723404255319149, 0.03008965026325733)	(0.8439716312056738, 0.026536577211162746)	(0.8924731182795699, 0.007603298722435987)	(0.8978494623655914, 0.03314201076865038)	(0.8862554112554113, 0.007998413946710775)	(0.8743506493506494, 0.013792652099054907)
GradientBoostingClassifier	(0.8749126484975541, 0.01830300608484805)	(0.8685068716515257, 0.03152710302344811)	(0.8368794326241135, 0.0501494171054289 2)	(0.8297872340425533, 0.06263660188884998)	(0.8494623655913979, 0.054828166812825645)	(0.8494623655913979, 0.054828166812825645)	(0.8563852813852814, 0.05232300648068234)	(0.8504329004329004, 0.0604142372476697)
LGBMClassifier	(0.8624505008152807, 0.03891901292310161)	(0.8310039599347775, 0.06761786213237252)	(0.8439716312056738, 0.026536577211162746)	(0.851063829787234, 0.017372267679313345)	(0.8655913978494624, 0.042333375666730184)	(0.8602150537634409, 0.049858163954277 99)	(0.8624458874458875, 0.054856573374214)	(0.8564935064935065, 0.03798100000510161)
XGBClassifier	(0.8746797111576986, 0.0475648693896118)	(0.8746797111576986, 0.0475648693896118)	(0.851063829787234, 0.017372267679313345)	(0.8368794326241135, 0.020059766842171559)	(0.8548387096774194, 0.057403646516297376)	(0.8655913978494624, 0.054828166812825645)	(0.8623376623376623, 0.044353548991556434)	(0.8623376623376623, 0.044353548991556434)

(...) Construção do Modelo Preditor de potencial

3. Melhoria do modelo

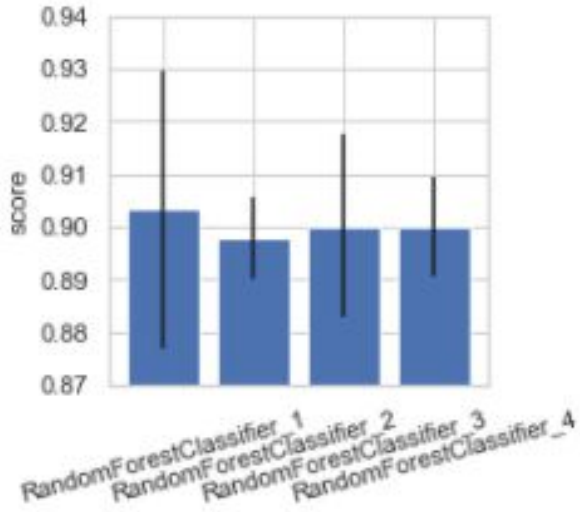
Também seguimos os passos do preditor de faturamento, ajustando os parâmetros dos quatro modelos escolhidos na etapa anterior. Os resultados foram:

- **RandomForestClassifier_1 com balanceamento de dados, sem remoção de outliers e com seleção de features** (score = 0.9032 / desvio padrão entre os k-folds = 0.0263);
- **RandomForestClassifier_2 com balanceamento de dados, sem remoção de outliers e sem seleção de features** (score = 0.8978 / desvio padrão entre os k-folds = 0.0076);
- **RandomForestClassifier_3 sem balanceamento de dados, sem remoção de outliers e com seleção de features** (score = 0.9000 / desvio padrão entre os k-folds = 0.0173);
- **RandomForestClassifier_4 sem balanceamento de dados, sem remoção de outliers e sem seleção de features** (score = 0.8999 / desvio padrão entre os k-folds = 0.0093).

(...) Construção do Modelo Preditor de potencial

3. (...) Melhoria do modelo

Para escolhermos o melhor entre eles, devemos olhar para o maior *score* e também o menor desvio padrão. Segue um gráfico com os resultados:



(...) Construção do Modelo Preditor de potencial

3. (...) Melhoria do modelo

Após ajustes, o modelo RandomForestClassifier_1 teve a maior acurácia com 0.9032, mas um desvio padrão de 0.0263, sendo o mais alto e não desejado. Isso significa que no pior caso esse modelo pode prever incorretamente 12.31% dos casos (1 - acurácia + desvio padrão), enquanto que RandomForestClassifier_2 e RandomForestClassifier_4 podem prever nos piores caso 10.98% e 10.94%, respectivamente, de valores preditos incorretamente. São praticamente iguais para o pior caso, mas o RandomForestClassifier_4 pode acertar mais no melhor caso, sendo 90.54% e 90.92% (acurácia + desvio padrão), respectivamente, de valores preditos corretamente. O vencedor fica com **RandomForestClassifier_4 sem balanceamento de dados, sem remoção de outliers e sem seleção de features**.

Observe que um modelo preditor é mais que simplesmente um algoritmo treinado. É o algoritmo mais as ações de engenharia de *features*.

(...) Construção do Modelo Preditor de potencial

4. Predição de potencial dos bairros de SP

Com o modelo vencedor, bastamos utilizar para prever os níveis de potencial dos bairros de São Paulo que estavam com campos vazios. Também não tivemos que remover *features* (por causa da seleção de *features*) para fazer essas predições, pois o modelo vencedor não contempla nenhuma das transformações de dados. Segue tabela mostrando alguns dos dados dos bairros de SP com níveis de potencial preenchidos pela predição do modelo vencedor:

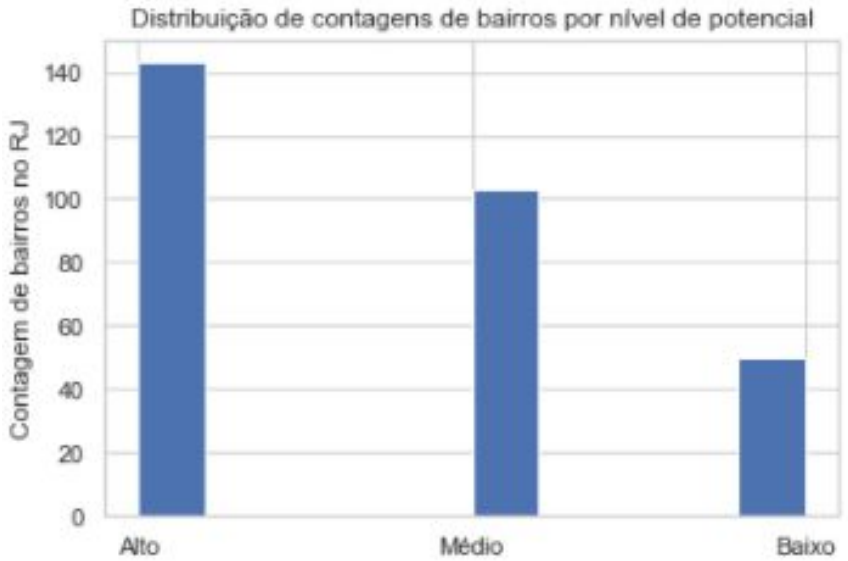
	codigo	nome	cidade	estado	população	popAte9	popDe10a14	popDe15a19	popDe20a24	popDe25a34	...	domiciliosA2	domiciliosB1	domiciliosB2	domiciliosC1	domiciliosC2	domiciliosD	domiciliosE	rendaMedia	faturamento	potencial
160	355030251.0	A. E. Carvalho	São Paulo	SP	94034.0	12668.0	6853.0	9836.0	7487.0	14535.0	...	253.0	2197.0	4368.0	6681.0	7011.0	2247.0	5670.0	1501.0	3.869801e+05	Alto
161	35503020.0	Aclimação	São Paulo	SP	32791.0	2297.0	1017.0	2096.0	2197.0	5341.0	...	1734.0	3704.0	2351.0	1946.0	827.0	291.0	1617.0	5920.0	1.425917e+06	Alto
162	355030285.0	Adventista	São Paulo	SP	104193.0	15070.0	7343.0	10631.0	8657.0	17749.0	...	0.0	1423.0	4875.0	8595.0	10082.0	3111.0	5776.0	1284.0	4.027942e+04	Médio
163	35503088.0	Água Branca	São Paulo	SP	12721.0	953.0	343.0	627.0	819.0	2142.0	...	667.0	1558.0	1032.0	915.0	361.0	84.0	404.0	6278.0	1.118423e+06	Alto
164	35503066.0	Água Funda	São Paulo	SP	48417.0	5078.0	2396.0	4018.0	3571.0	7388.0	...	303.0	1794.0	2986.0	4489.0	2836.0	1104.0	2553.0	1905.0	8.622022e+05	Alto
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
451	355030213.0	Vila Sônia	São Paulo	SP	34061.0	2908.0	1462.0	2253.0	2076.0	4579.0	...	1089.0	2866.0	2219.0	2216.0	1227.0	306.0	1098.0	5285.0	1.275856e+06	Alto
452	355030207.0	Vila Suzana	São Paulo	SP	35403.0	4127.0	1890.0	2678.0	2433.0	5855.0	...	1522.0	2458.0	1186.0	1166.0	918.0	209.0	3840.0	7418.0	1.294612e+06	Alto
453	355030162.0	Vila Terezinha	São Paulo	SP	122359.0	18304.0	9304.0	13258.0	9965.0	19248.0	...	0.0	1758.0	4517.0	9450.0	11473.0	3218.0	7540.0	1252.0	-1.242247e+05	Médio
454	355030157.0	Vila Zatt	São Paulo	SP	125864.0	14670.0	7305.0	11225.0	9338.0	18841.0	...	872.0	5093.0	8063.0	10012.0	8082.0	2856.0	6853.0	1936.0	8.135220e+05	Alto
455	355030164.0	Vista Alegre	São Paulo	SP	106.0	23.0	7.0	15.0	11.0	15.0	...	0.0	0.0	6.0	9.0	8.0	2.0	4.0	1288.0	8.446202e+05	Baixo

296 rows × 24 columns

(...) Construção do Modelo Preditor de potencial

4. (...) Predição de potencial dos bairros de SP

A distribuição dos dados de potencial preditos em São Paulo ficou assim:



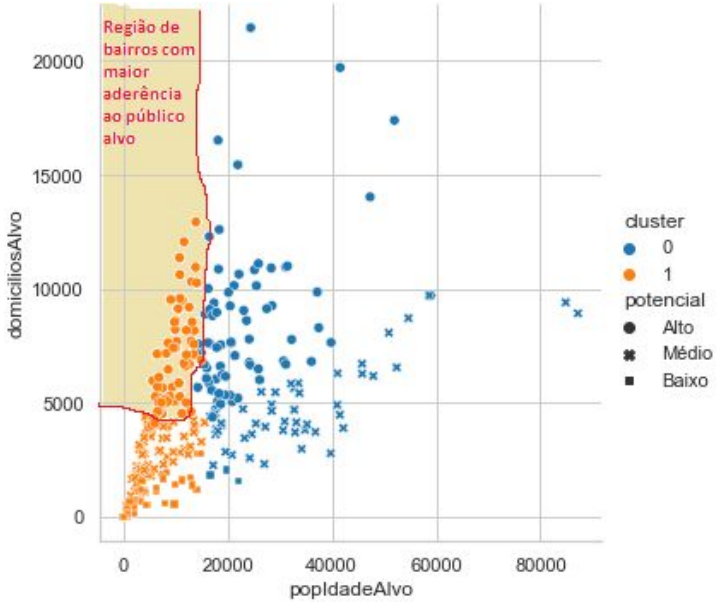
Segmentação dos bairros de São Paulo

Tivemos que segmentar de acordo com a renda e a idade, indicando aqueles com maior aderência ao público alvo que são adultos de 25 a 50 anos das classes A (rendas A1 e A2) e B (rendas B1 e B2). O público alvo de idades está separado nas colunas de populações e fizemos simplesmente somar aquelas das idades alvos (**popDe25a34+popDe35a49=popIdadeAlvo**), gerando uma nova coluna com esses dados, fazendo o mesmo com as contagens de domicílios alvos (**domiciliosA1+domiciliosA2'+domiciliosB1+domiciliosB2=domiciliosAlvo**). Como devemos incluir outras features, como a renda média do bairro, então vamos agrupá-los em clusters utilizando um modelo de Machine Learning do tipo não-supervisionado (não rotulado). Aproveitamos para incluir também nesse agrupamento a feature potencial, além das referentes ao público alvo. Esse modelo não-supervisionado tenta separar os dados usando uma métrica de distanciamento entre grupos e há uma forma técnica de saber quantos grupos é a melhor opção. Neste caso, apenas dois grupos (grupos 0 e 1).

...	codigo	nome	cidade	estado	população	popAte9	...	domiciliosE	rendaMedia	faturamento	potencial	popIdadeAlvo	domiciliosAlvo	cluster
161	35503020.0	Aclimação	São Paulo	SP	32791.0	2297.0	...	1617.0	5920.0	1.425917e+06	Alto	12622.0	9202.0	1
163	35503088.0	Água Branca	São Paulo	SP	12721.0	953.0	...	404.0	6278.0	1.118423e+06	Alto	4975.0	3881.0	1
165	35503052.0	Água Rasa	São Paulo	SP	26134.0	2102.0	...	1510.0	5907.0	1.237846e+06	Alto	9462.0	5698.0	1
167	35503097.0	Alfredo Pujol	São Paulo	SP	39310.0	2720.0	...	1561.0	6586.0	1.619215e+06	Alto	13888.0	10954.0	1
168	355030146.0	Alto Da Lapa	São Paulo	SP	15551.0	1160.0	...	379.0	8303.0	1.178428e+06	Alto	5524.0	4083.0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

(...) Segmentação dos bairros de São Paulo

Pudemos, então, plotar um gráfico para observarmos a dispersão dos bairros segregados por população de idades alvos, contagem de domicílios alvos, agrupamento incluindo estes e mais a renda média e nível de potencial. Incluímos o potencial para obtermos informações adicionais sobre o público alvo.



Plotamos e observamos o gráfico disperso e segregado ao lado, sendo cada ponto um bairro, onde conseguimos separar visualmente uma região que nos mostra os mais propensos a terem maiores faturamentos com alto potencial e dentro do público alvo. É a região compreendida pelo cluster 1 (cor laranja) e ao mesmo tempo pelo potencial Alto (pontos em formato de círculo). Alguns dados se mostraram interessantes somente olhando para esse gráfico, como o mínimo de contagem de domicílios alvo em torno de 5000 e o mínimo de população com idade alvo também em torno de 5000.

(...) Segmentação dos bairros de São Paulo

Para obtermos a lista dos bairros com maior aderência ao público alvo, basta filtrar da nossa tabela mostrada anteriormente os bairros do cluster 1 e também com nível Alto de potencial, resultando na tabela abaixo com 72 bairros, mostrando alguns deles:

	codigo	nome	cidade	estado	população	popAte9	popDe10a14	popDe15a19	popDe20a24	popDe25a34	...	domiciliosE	rendaMedia	faturamento	potencial	potencial_num	...
161	35503020.0	Aclimação	São Paulo	SP	32791.0	2297.0	1017.0	2096.0	2197.0	5341.0	...	1617.0	5920.0	1.425917e+06	Alto	3	...
163	35503088.0	Água Branca	São Paulo	SP	12721.0	953.0	343.0	627.0	819.0	2142.0	...	404.0	6278.0	1.118423e+06	Alto	3	...
165	35503052.0	Água Rasa	São Paulo	SP	26134.0	2102.0	996.0	1694.0	1553.0	3625.0	...	1510.0	5907.0	1.237846e+06	Alto	3	...
167	35503097.0	Alfredo Pujol	São Paulo	SP	39310.0	2720.0	1317.0	2422.0	2351.0	5431.0	...	1561.0	6586.0	1.619215e+06	Alto	3	...
168	355030146.0	Alto Da Lapa	São Paulo	SP	15551.0	1160.0	507.0	911.0	757.0	1967.0	...	379.0	8303.0	1.178428e+06	Alto	3	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

72 rows × 30 columns

(...) Segmentação dos bairros de São Paulo

Abaixo, segue os 15 bairros **mais aderentes ao público alvo com maiores** potenciais e faturamentos preditos e os 15 bairros **mais aderentes ao público alvo com menores** potenciais e faturamentos preditos, respectivamente:

nome	rendaMedia	potencial
Morumbi	14504.000000	Alto
Viera De Morais	13650.000000	Alto
Chácara Klabin	13218.000000	Alto
Real Parque	12706.000000	Alto
Trianon	12550.000000	Alto
Chácara Itaim	12424.000000	Alto
Paraíso	11686.000000	Alto
Chácara Flora	11500.000000	Alto
Joaquim Nabuco	11335.000000	Alto
Vila Mariana	10575.000000	Alto
Puc	10064.000000	Alto
Pamplona	9843.000000	Alto
Boçava	9800.000000	Alto
Vila Leopoldina	9781.000000	Alto
Vila Beatriz	9406.000000	Alto

nome	rendaMedia	potencial
Penha	2121.000000	Alto
São Lucas	2222.000000	Alto
Ladeira Da Memória	2360.000000	Alto
Santa Clara	2419.000000	Alto
Ipiranga	3427.000000	Alto
Oratório	3694.000000	Alto
Casa Verde	3862.000000	Alto
Mooca	4111.000000	Alto
Tucuruvi	4198.000000	Alto
Parque Continental	4276.000000	Alto
Jardim Anália Franco	4295.000000	Alto
Tietê	4331.000000	Alto
Belém	4366.000000	Alto
Vila Prudente	4497.000000	Alto
Nazaré - Alto Do Ipiranga	4675.000000	Alto

Conclusão

Criamos dois modelos preditores baseados nos algoritmos **ElasticNet** e **RandomForestClassifier** de aprendizado de máquina supervisionado (com *label*), para prevermos faturamento e nível de potencial, respectivamente, dos bairros de São Paulo tomando como referência dados dos bairros do Rio de Janeiro.

Para se chegar nesses modelos, passamos por vários outros, onde pudemos testar cada um deles e escolher o que teve melhor desempenho segundo as métricas que melhor convinha com os tipos de problemas, sendo a métrica **erro absoluto mediano** para a predição de faturamento e a métrica **acurácia** para o problema de predição de nível de potencial. Para este último, utilizamos os dados preditos de faturamento dos bairros de São Paulo uma vez que já tínhamos pronto o modelo preditor para tal.

Aplicamos três tipos de transformação de dados para serem testadas pelos modelos: **balanceamento de dados** (somente para problemas de classificação, ou seja, o preditor de potencial), r**emoção de *outliers***, e **seleção de *features***, onde foi feito combinação entre eles para diversificar os treinos/validações.

O **erro absoluto mediano** para o modelo escolhido de **predição de faturamento** (ElasticNet) foi de **R$ 4948.37** no teste/validação e a **acurácia** para o modelo escolhido de **predição de nível de potencial** (RandomForestClassifier) foi de **89.99%** no teste/validação. A decisão de escolha desses modelos não foi baseada somente nessas métricas, mas também pelo menor desvio padrão obtido das pontuações dessas métricas em cada sub-amostra no processo de validação cruzada.

Conseguimos **segmentar os bairros de São Paulo pelo público alvo** de população com idades entre 25 e 50 anos e domicílios das classes A e B. Porém, tivemos que **criar dois clusters (grupos) segregados pelas features do público alvo, renda média e potencial** com o objetivo de indicar os **bairros com maior aderência ao público alvo**, resultando numa tabela de **72 bairros onde a empresa pode começar a investir com mais segurança financeira**.

Contato

- E-mail: erivelton.mc@gmail.com
- Fone: (98)98809-1227.

Obrigado!