Relatório: Previsão de Compra de Casa

1. Introdução

1.1 Objetivo

Esse projeto tem como objetivo construir um modelo de classificação para prever se um usuário de um site imobiliário irá efetuar a compra de uma casa. A variável alvo é a classe Compra, uma variável binária (sendo 1 para comprador e 0 para não comprador). Partindo disso exploramos o conjunto de dados com várias variáveis a fim de encontrar possíveis padrões que auxiliem nas previsões de decisão de compra.

1.2 Descrição do Dataset

O *dataset* contém informações de usuários que visitaram um site imobiliário. As variáveis disponíveis são:

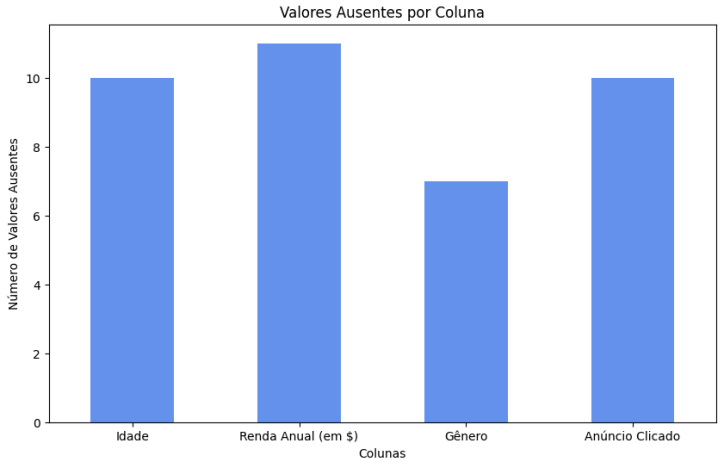
- **Idade**: Idade do usuário.
- **Renda Anual (em $)**: Renda anual do usuário em dólares.
- **Gênero**: Gênero do usuário (Feminino ou Masculino).
- **Tempo no Site (min)**: Tempo que o usuário passou navegando no site (em minutos).
- **Anúncio Clicado**: Indica se o usuário clicou em um anúncio no site (Sim ou Não).
- **Compra**: Indica se o usuário comprou uma casa (1 para sim, 0 para não).

2. Análise Exploratória dos Dados

2.1 *Metadados*

A análise exploratória começa com a obtenção de informações sobre os dados do *dataset*:

- Tipos das variáveis: float, int, object.
- Há 200 amostras.
- Há 6 colunas em que cada uma corresponde a uma variável.
- Colunas com valores nulos e ausentes: Idade (10), Renda Anual (11), Gênero (7), Anúncio Clicado (10).

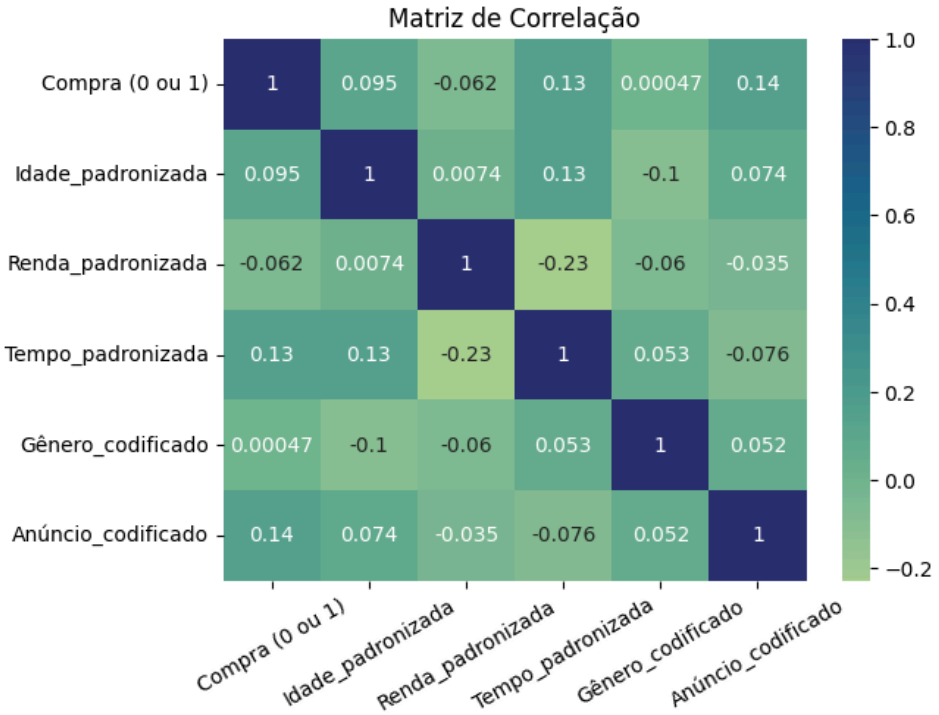


Removendo esses elementos incomuns obtemos uma estatística descritiva mais limpa das variáveis numéricas utilizando a função *describe()*:

	Idade	Renda Anual (em $)	Tempo no Site (min)	Compra (0 ou 1)
count	165.000000	165.000000	165.000000	165.000000
mean	39.254545	59515.151515	17.994546	0.321212
std	12.829394	26221.524872	7.191249	0.468364
min	18.000000	30000.000000	5.052596	0.000000
25%	28.000000	30000.000000	12.616978	0.000000
50%	39.000000	50000.000000	18.312317	0.000000
75%	51.000000	70000.000000	24.047671	1.000000
max	59.000000	100000.000000	29.853484	1.000000

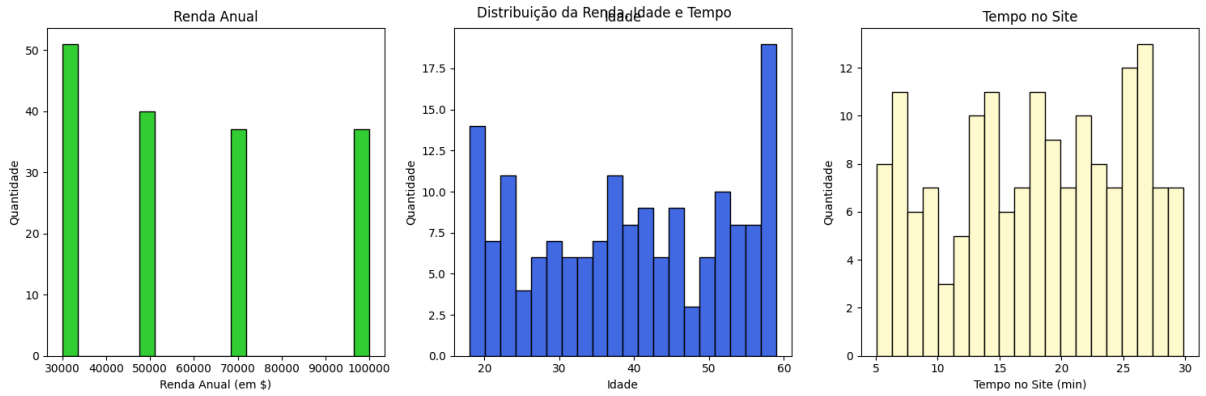
Agora obtendo um conjunto de dados ligeiramente menor que o inicial, podemos observar de cada classe: 165 entradas únicas, proporções diferentes (como Tempo e Renda), suas respectivas médias, desvio padrão, máximos e mínimos, limite inferior (em Idade: 25% dos valores estão abaixo de 28), mediana e limite superior (em Idade: 75% dos valores estão abaixo de 51).

2.2 Correlação



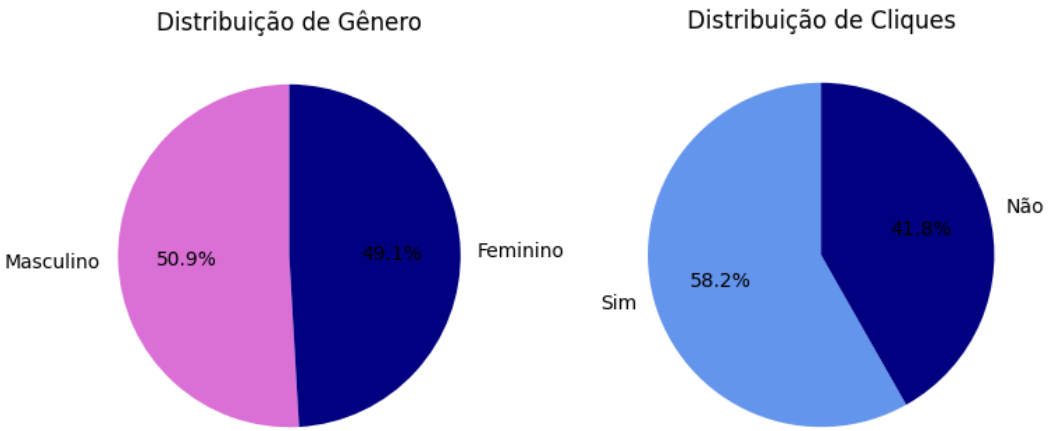
As variáveis não apresentaram uma forte correlação entre si no geral, o que indica que elas são independentes umas das outras, a principal correlação encontrada foi Anúncio Clicado e Compra, e Tempo no Site e Compra, porém com apenas 14% e 13% respectivamente. O restante das variáveis têm correlação praticamente nula entre si, o que sugere utilização de técnicas específicas e mais robustas para estudar a relação entre essas variáveis e criação de novas variáveis ou interações entre as mais correlacionadas com "Compra".

2.3 Distribuição de Variáveis



Os histogramas apresentam as distribuições das variáveis numéricas Renda, Idade e Tempo no Site:

- Renda: os usuários têm rendas anuais pontualmente de 30 mil, 50 mil, 70 mil e 100 mil, se concentrando em certos níveis de renda;
- Idade: a maior quantidade de usuários têm 20 anos e 60 anos;
- Tempo no Site: a distribuição é relativamente equilibrada;

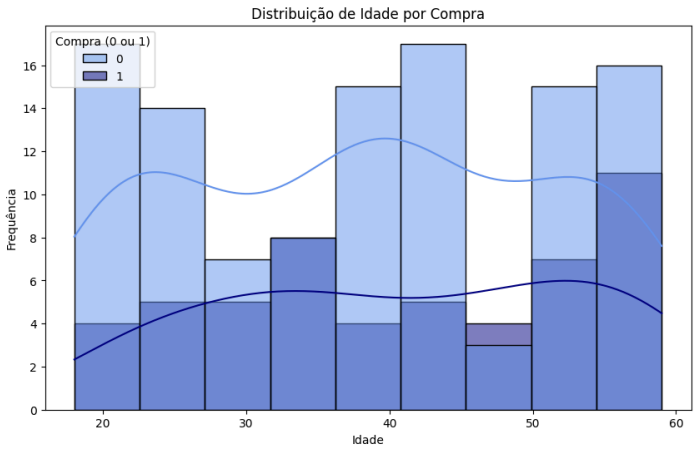


As variáveis categórica e binária, Gênero e Anúncio Clicado, têm distribuições proporcionais.

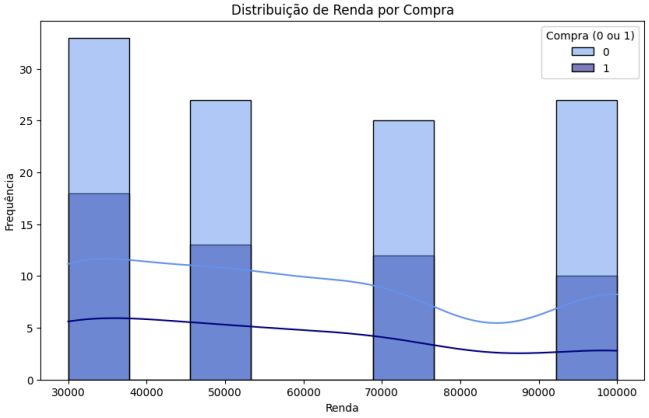
2.4 Relação entre as Variáveis Independentes e a Variável Alvo (Compra)

Foi realizada uma análise para entender como as variáveis independentes estão relacionadas com a variável alvo:

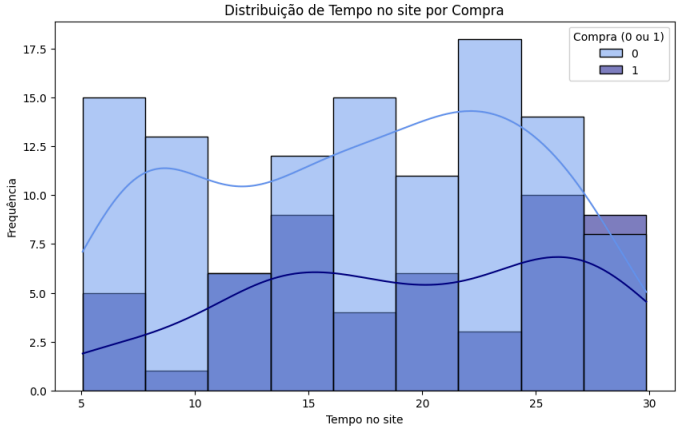
- Idade: A relação entre a idade e a compra foi verificada por meio de histograma onde foi possível verificar que os compradores mais novos (50-60 anos) têm uma frequência de compra maior, enquanto a proporção de compradores mais novos é menos dos 20 aos 40 anos.



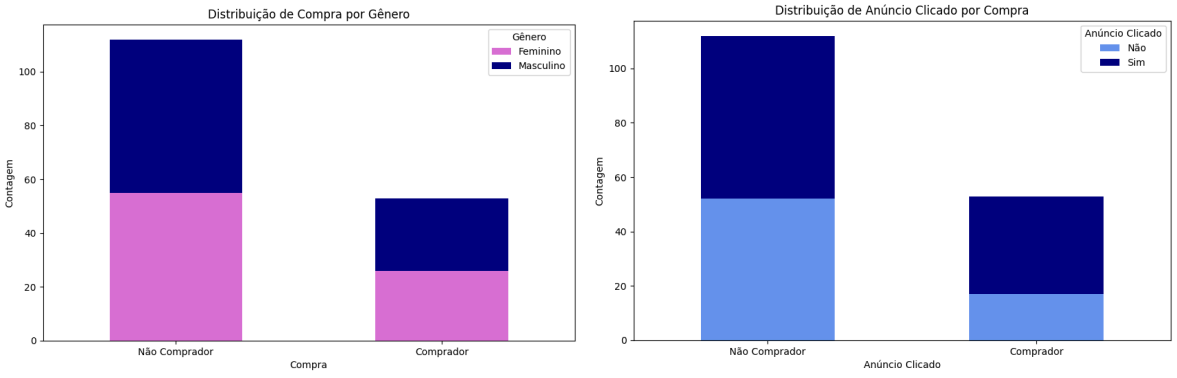
- Renda Anual: Foi analisada a correlação entre a renda e a compra utilizando histograma também: o público com maior renda (70 e 100 mil) são os que compram com menos frequência.



- Tempo no Site: A relação entre o tempo gasto no site e a probabilidade de compra foi analisada com histograma que indicou que quando o cliente passa mais de 15 minutos a probabilidade de que realize uma compra é considerável e maior se for depois dos 25 minutos.



- Gênero e Anúncio Clicado: A relação entre as variáveis categóricas e a compra foi analisada por gráficos de barras.



3. Pré-processamento dos Dados

3.1 Normalização e Padronização

As variáveis numéricas Idade, Renda Anual, e Tempo no Site, foram padronizadas, conforme necessário, para garantir que todas as variáveis estivessem na mesma escala e evitassem a dominância de variáveis de maior magnitude utilizando a função *StandardScaler()*. Foi adicionado no dataset original as colunas com as variáveis numéricas padronizadas.

3.2 Codificação de Variáveis Categóricas

As variáveis categóricas, como Gênero e Anúncio Clicado, foram convertidas em variáveis numéricas utilizando *LabelEncoding()*, conforme o caso, para que pudessem ser utilizadas nos modelos de machine learning. Ficaram organizadas como:

- Gênero - 0: Feminino; 1: Masculino;
- Anúncio clicado - 0: Não, 1: Sim.

3.3 Divisão dos Dados

Os dados foram divididos, usando *train_test_split()*, em dois conjuntos: treinamento (70%) e teste (30%). O conjunto de treinamento foi utilizado para ajustar o modelo, enquanto o conjunto de teste foi utilizado para avaliar o desempenho do modelo.

4. Construção do Modelo de Classificação

4.1 Seleção do Modelo

Foram avaliados três modelos: Random Forest, Árvores de Decisão e Regressão Logística. Esses modelos se diferem quando falamos de abordagem e métodos, sendo respectivamente um ensemble, um baseado em regras e outro linear.

A seleção foi feita seguindo o algoritmo:

1. Criar o dicionário de modelos simples sem ajuste de hiperparâmetros.
2. Criar uma lista para guardar os resultados.
3. Iteramos aplicando o modelo nos dados de treino e gerando as previsões para os dados de teste.
4. Logo após, antes de finalizar a iteração, são calculadas as métricas de acurácia, precisão, recall, F1-Score e ROC AUC.
5. Os resultados das métricas são armazenados na lista de resultados dos modelos.
6. Voltamos ao passo 3 até que todos os modelos sejam testados.
7. Um *dataframe* é criado com os resultados a fim de comparar os valores obtidos.
8. Por fim é realizada plotagem comparativa dos resultados de cada modelo para cada métrica.

As métricas utilizadas fornecem uma visão abrangente do desempenho, principalmente em situações em que temos classes desbalanceadas, onde o F1-Score e ROC AUC são indicadores importantes.

Modelo	**Acurácia**	**Precisão**	**Recall**	**F1-Score**	**ROC AUC**
Random Forest	**66,67%**	60%	25%	**35,29%**	51,59%
Regressão Logística	63,64%	0%	0%	0%	55,95%
Árvores de Decisão	51,52%	33,33%	33,33%	33,33%	47,62%

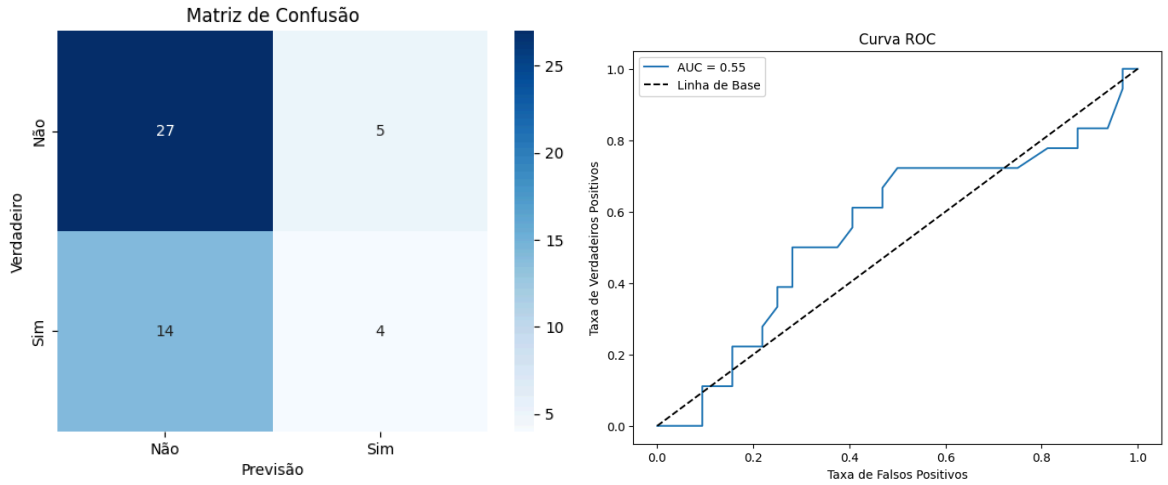
O Random Forest apresentou a melhor performance em geral em termos de acurácia e F1-Score, Árvores de Decisão teve um desempenho intermediário enquanto a Regressão Logística teve resultados insatisfatórios. No geral o desempenho de todos os modelos não foi tão satisfatório, mas devido as métricas o Random Forest foi selecionado para aprimoramento.

4.2 Avaliação dos Modelos

O modelo foi avaliado utilizando as seguintes métricas:

- **Acurácia**: A porcentagem de previsões corretas.
- **Precisão**: A capacidade do modelo de não classificar incorretamente exemplos negativos como positivos.
- **Recall**: A capacidade do modelo de capturar todos os exemplos positivos.
- **F1-Score**: A média harmônica entre precisão e recall.
- **ROC AUC**: Capacidade de fazer a separação corretamente das variáveis.
- **Matriz de Confusão**: Para visualizar o desempenho do modelo em termos de verdadeiros positivos, falsos positivos, verdadeiros negativos e falsos negativos.

4.3 *Random Forest*



O Random Forest foi testado novamente sem hiperparâmetros definidos e mostrou uma acurácia de 62% e AUC de 55%. O relatório com as principais métricas de avaliação utilizando o *classification_report()*:

	precision	recall	f1-score	support
Não Comprador	0.66	0.84	0.74	32
Comprador	0.44	0.22	0.30	18
accuracy			0.62	50
macro avg	0.55	0.53	0.52	50
weighted avg	0.58	0.62	0.58	50

A Precisão diz respeito a porcentagem de previsões corretas, o Recall indica a porcentagem de exemplos positivos indicados corretamente e F1-Score é a média harmônica entre a precisão e o recall.

4.4 Ajuste de Hiperparâmetros

Foi realizada uma busca por hiperparâmetros utilizando GridSearchCV para encontrar a melhor combinação de parâmetros do modelo: número de árvores (*n_estimators*), profundidade máxima (*max_depth*), número mínimo de amostras para dividir um nó (*min_samples_split*), número mínimo de amostras em cada folha (*min_samples_leaf*), número de recursos a serem considerados para dividir um nó (*max_features*) e penalização para lidar com desbalanceamento de classes (*class_weight*).

O *make_scorer()* foi utilizado para otimizar o F1-Score com base na classe Comprador e a Acurácia, isto é, avaliar qual combinação tem melhor desempenho para essa métrica. Assim, definimos o *grid_search()* com um cross-validation de 5 folds.

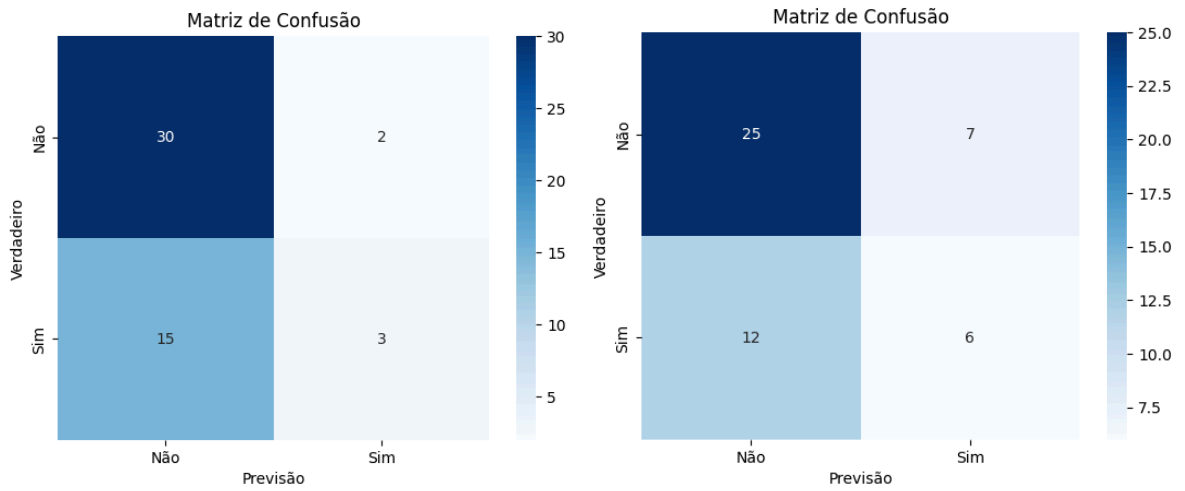
Para o teste otimizando o F1-Score obtivemos:

```
Melhores hiperparâmetros: {'bootstrap': False, 'class_weight': {0: 1, 1: 2}, 'max_depth': None, 'max_features': 'log2', 'min_samples_leaf': 2, 'min_samples_split': 10, 'n_estimators': 50}

Melhor F1-Score (validação cruzada): 0.36897651368239603
```

E para o teste otimizando a Acurácia obtivemos:

```
Melhores hiperparâmetros: {'bootstrap': True, 'class_weight': None, 'max_depth': None, 'max_features': 'log2', 'min_samples_leaf': 4, 'min_samples_split': 5, 'n_estimators': 200}

Melhor Acurácia (validação cruzada): 0.7304347826086955
```

Ambos apresentaram as seguintes matrizes de confusão para Acurácia e F1-Score , respectivamente:

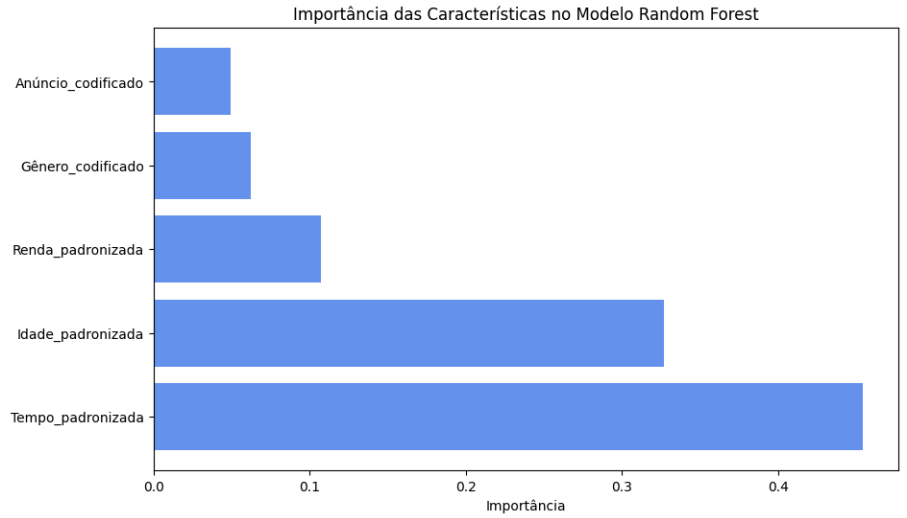


Podemos notar que de fato os hiperparâmetros definidos com base no F1-Score (direita) apresentaram um desempenho melhor (100%) em determinar compradores, mas em comparação a matriz dos hiperparâmetros definidos com base na acurácia o número de Falsos Positivos foi maior também. Como a maior quantidade de acertos foi no geral pelos hiperparâmetros de acurácia, esses serão os hiperparâmetros escolhidos.

5. Interpretação dos Resultados

5.1 Importância das Variáveis

Utilizando o método *feature_importances_* do Random Forest, identificamos as variáveis que mais influenciam na decisão do modelo. As variáveis com maior importância foram Idade, Renda Anual, e Tempo no Site. Podemos ver no gráfico abaixo:



Em modelos como Random Forest e Árvores de Decisão já é realizada uma seleção implícita com base nas variáveis mais relevantes ao construir as árvores, reduzir o número de features, pode não ter tanto impacto.

5.2 Desempenho do Modelo

O modelo testado com os hiperparâmetros obtidos no GridSearchCV utilizando a acurácia como otimizador tiveram os seguintes resultados:

- O valor médio da validação cruzada foi de 70,4% o que indica uma performance geral acima da média, porém a variação entre os folds indica uma instabilidade do modelo a depender dos dados usados.
- O modelo tem um ótimo desempenho para a classe Não Comprador com 30 previsões corretas, porém teve dificuldades em determinar a classe Comprador acertando apenas 3 e apresentando 15 Falsos Negativos.
- A Precisão para Não Comprador e Comprador foi respectivamente 67% e 60%.
- O Recall para verdadeiros positivos foi bem alto 94%, enquanto para verdadeiros positivos foi de 17%.
- O F1-Score para Comprador foi de apenas 26% e enquanto para Não comprador foi 78%.

- A curva ROC com AUC foi de 0.60 significando que o modelo tem dificuldade de separação das classes sendo apenas ligeiramente melhor que o acaso (0.5).

5.3 Possíveis Melhorias

As variáveis parecem se relacionar muito melhor com a classe Não Comprador do que com a classe Comprador, além de não terem tanta relação entre si, para isso pode ser interessante fazer uma engenharia de features, isto é, removendo correlações fracas, transformar o dado caso a padronização não seja satisfatória ou criar novas variáveis a partir das existentes.

É válido também, analisar outros algoritmos de classificação como:

- KNN - usa a classe de vizinhos dominante para classificar com base em similaridade;
- Support Vector Machine - mais robusto que utiliza um hiperplano que melhor cubra os dados;
- Naive Bayes - utiliza probabilidade condicional com base em conhecimento prévio para calcular a probabilidade de uma classe;

É interessante fazer uma análise mais profunda das features como por exemplo utilizar o SHAP e fazer um tratamento das features mais impactantes, além de verificar quais se relacionam mais com a classe que não apresentou resultados bons. Um caminho possível também é aplicar técnicas de balanceamento como undersampling ou oversampling para melhorar a representatividade da classe ‘Comprador’.

6. Conclusão

Neste trabalho, analisamos e construímos diferentes modelos de classificação para prever o comportamento de compra de uma casa com base em variáveis como idade, renda anual, tempo no site, gênero e interação com anúncios. O modelo Random Forest obteve o melhor desempenho geral com uma Acurácia média de 70% e uma área sob a curva ROC (AUC) de 0.60. Porém, observamos dificuldades de todos os modelos em predizer corretamente a classe ‘Comprador’ o que resultou num recall de 17% para essa categoria.

Os resultados indicaram que o modelo teve uma boa capacidade de predição para identificar os usuários que não realizam compras (‘Não Comprador’). A dificuldade em prever compradores pode estar relacionada ao desbalanceamento das classes ou a falta de variáveis altamente correlacionadas com o comportamento da compra.

O principal desafio encontrado foi a baixa correlação entre as variáveis preditoras e o alvo, que mesmo após os ajustes e utilizando o GridSearchCV apresentou um desempenho distante do que seria ideal, existindo a possibilidade dos dados disponíveis não serem o suficiente para transpor os fatores relevantes.

Como próximo passo, é interessante coletar mais variáveis que expliquem melhor o comportamento de compra e também aplicar técnicas de balanceamento para mitigar os efeitos da falta de balanceamento, assim como explorar técnicas que ajudem a explorar relações mais complexas e menos lineares.

7. Código-Fonte

https://github.com/xndrxssx/HouseBuyers