CODER HOUSE

CURSO DE DATA SCIENCE DA CODER HOUSE

PAULO SÉRGIO PEREIRA FRANÇA JÚNIOR
THIAGO DE ANDRADE CARVALHO

Análise de Churn de clientes em banco utilizando Machine Learning

São Paulo/SP
2023

Sumário

Introdução

A retenção de clientes é uma preocupação central para qualquer empresa que deseje construir uma base de clientes sustentável a longo prazo. O termo "churn" refere-se à taxa na qual os clientes deixam de fazer negócios com uma empresa durante um período de tempo específico. O cálculo básico para se compreender essa taxa deve ser feito da seguinte maneira:

> *Taxa de Churn = total de clientes que cancelaram / total de clientes ativos no mesmo período*

Essa métrica desempenha um papel fundamental na avaliação da saúde de muitas empresas, uma vez que o crescimento dos negócios não se limita apenas à aquisição de novos clientes, mas também à manutenção daqueles já conquistados. O acompanhamento da taxa de churn também afeta diretamente a previsibilidade de receita e a manutenção da receita recorrente do negócio. Tal compreensão é essencial, pois, caso o churn de clientes não seja gerenciado adequadamente, pode resultar em perdas significativas de receita e impactar negativamente a saúde financeira e reputação da companhia.

Este trabalho busca testar e aplicar técnicas diversas de *machine learning* em uma base de dados genérica em formato *.csv* de um banco multinacional fictício chamado Banco ABC - que foi disponibilizada e retirada do site Kaggle. O objetivo é identificar, pela utilização e manipulação de dados pré-existentes, quais fatores têm maior impacto na taxa de churn desse banco e medir a acurácia de modelos preditivos que possam ser mais eficazes em antecipar tendências e comportamentos futuros.

Na parte inicial deste trabalho, faz-se a análise individual de cada variável disponível na base de dados. São utilizadas análises estatísticas “superficiais”, como a observação da média e mediana, variância, valores mínimos e máximos, dentre outros. Também se utilizam de gráficos de ocorrência para entender cada fator. Em seguida, utilizando-se de um gráfico tipo *pairplot*, faz-se a análise bidimensional da relação entre variáveis em função da taxa de churn. Nesse momento, busca-se antecipar observações e possíveis conclusões acerca de quais variáveis podem ser mais relevantes para a construção do modelo.

Na etapa de desenvolvimento do modelo, aplicamos o modelo de *Random Forest* para identificar o grau de importância de cada variável. Por meio da medição de relevância de cada variável, é possível reduzir a quantidade de variáveis utilizadas, buscando mitigar a possibilidade de *overfitting* ou alta multicolinearidade. Em seguida, testamos a aplicação de cinco modelos de classificação diferentes, rodando cinco vezes cada um e utilizando o método de *cross validation*. Com base na medição de acurácia de cada modelo, é possível selecionar, dentre as diferentes técnicas de *machine learning,* qual adequa-se melhor à criação de um modelo preditivo para compreensão e predição do evento de churn no caso teórico objetivo desta análise.

Após escolhido o modelo, faz-se, finalmente, uma comparação entre o modelo sem otimização de parâmetros e com otimização. Esta etapa ajudaria o time de negócio (também hipotético) a entender se a etapa de otimização faz sentido para a empresa em termos de custo operacional, uma vez que seria uma etapa que exigiria alto esforço computacional em uma situação de aplicação em uma base de dados completa. Ao final, compartilhamos nossas

conclusões e destacamos os desafios e possíveis caminhos a serem buscados em análises futuras.

Objetivos da análise

O objetivo do estudo é utilizar uma base de dados pré-existente e teórica para aplicar técnicas de análise descritiva e *machine learning* a fim de identificar as variáveis de maior relevância para a ocorrência de churn, identificar clientes/casos mais propensos a sofrer churn e avaliar se estes, de fato, ocorreram. Também é objetivo deste trabalho testar a acurácia de diferentes modelos de *machine learning* para identificação e predição dos fatores causadores de churn na base de clientes do banco hipotético. Finalmente, através da interpretação dos dados, busca-se construir um entendimento sobre a perspectiva de negócio sobre o banco hipotético ABC como um caso de aplicação prática das ferramentas teóricas abordadas.

Análise descritiva

Os dados sobre taxa de churn utilizados nesta análise foram extraídos da base de dados *Bank Customer Churn Dataset* disponibilizada no Kaggle[1] e já estão devidamente limpos e formatados para utilização. A base contém 1001 linhas, sendo a primeira os cabeçalhos com os títulos das doze colunas, conforme listadas e classificadas por tipo pela tabela a seguir, sendo *int64* um número inteiro, *float64* um número decimal e *object* uma *string*, ou texto:

```
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customer_id       10000 non-null  int64
 1   credit_score      10000 non-null  int64
 2   country           10000 non-null  object
 3   gender            10000 non-null  object
 4   age               10000 non-null  int64
 5   tenure            10000 non-null  int64
 6   balance           10000 non-null  float64
 7   products_number   10000 non-null  int64
 8   credit_card       10000 non-null  int64
 9   active_member     10000 non-null  int64
 10  estimated_salary  10000 non-null  float64
 11  churn             10000 non-null  int64
dtypes: float64(2), int64(8), object(2)
```

Considerando os dados da base, observa-se que o Banco ABC possui o registro de dez mil clientes únicos com números de identificação distintos (IDs). As colunas são brevemente explicadas pela tabela a seguir posteriormente analisadas através de análise descritiva:

campos	**descrição**
customer_id	Id do cliente
country	país
gender	gênero
age	idade
credit_score	Score de saúde financeira
tenure	tempo em que o cliente está no banco
balance	saldo do cliente
products_number	quantidade de produtos do cliente
credit card	se o cliente tem cartão de crédito
active_member	se é membro ativo no banco

[1] (https://docs.google.com/document/d/14ZgWIHCOF84tRsS2Qykhmw3KsxPfwXGwW6lSBGv2684/edit)

estimated_salary	salário estimado
churn	se cliente ainda é correntista do banco

Tabela 1: campos da tabela de clientes do banco ABC

A análise descritiva das variáveis permite entender melhor a base e escolher quais serão os parâmetros utilizados na fase de modelagem. Também compõe etapa relevante para entender caso hipotético em análise, o *business case* e a interação das variáveis entre si. Cada variável será listada e brevemente descrita a seguir. Depois, faz-se uma análise comparativa de interação de cada uma delas utilizando o *pairplot*.

ID do cliente (*customer_id)*

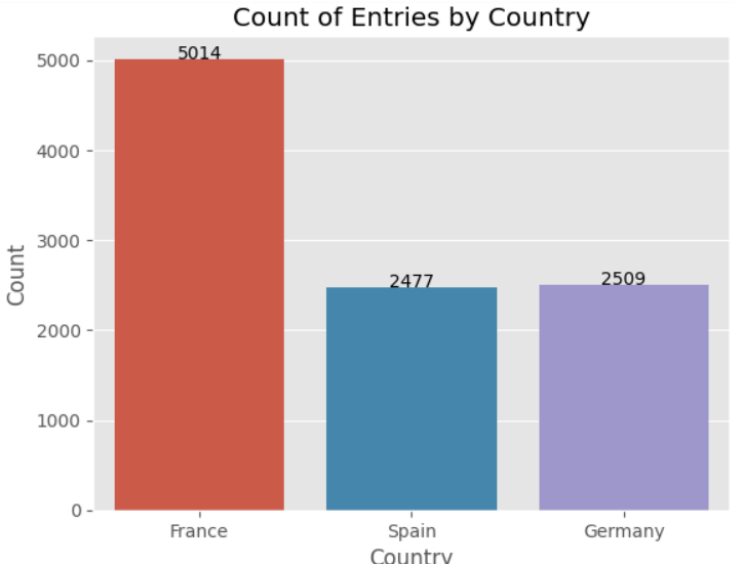
Os dados dos clientes do banco são anonimizados e representados por um número de identificação de oito dígitos. A análise dos dados comprova que tratam-se de clientes únicos.

Pontuação de crédito (*credit_score*)

A pontuação de crédito mede a saúde financeira do crédito dos clientes, usualmente atribuindo valores mais baixos a clientes considerados piores devedores - sob a perspectiva de apresentarem maior risco de não pagarem por seus empréstimos tomados - e valores mais altos a clientes com menos risco de não pagarem seus empréstimos. No Banco ABC, a média da pontuação de crédito dos clientes é de aproximadamente 650, com variação entre 350 (mínimo) e 850 (máximo). A dispersão dos dados é relativamente alta, indicada pelo desvio padrão de 96.65.

País (*country*)

Apenas três países compõem a base de dados analisada: França, Alemanha e Espanha. Aproximadamente metade dos clientes concentram-se na França, enquanto os restantes dividem-se de maneira mais ou menos igual (25%) entre os outros dois países.



Gênero (*gender*)

Um cuidado muito importante que temos que ter ao rodar modelos de machine learning é na checagem de viés dos dados. Como estamos utilizando características pessoais como gênero e idade, é importante tomar cuidado para que o modelo não tome decisões favoráveis a apenas um grupo de pessoas. A análise descritiva vai nos ajudar a entender que aspectos dos clientes possuem relação direta entre si e entre a taxa de churn.

Os clientes do banco são 54,6% homens (5.457) e 45,4% mulheres (4.543). A seguir, observa-se a distribuição dos salários estimados em função do gênero a partir de um gráfico de boxplot:

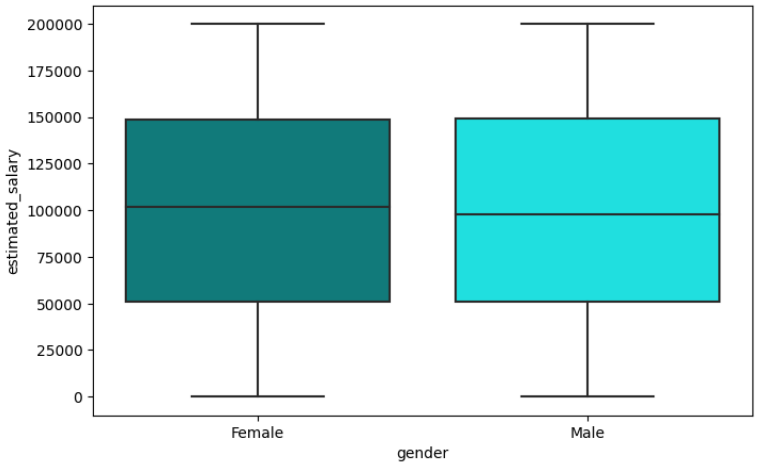
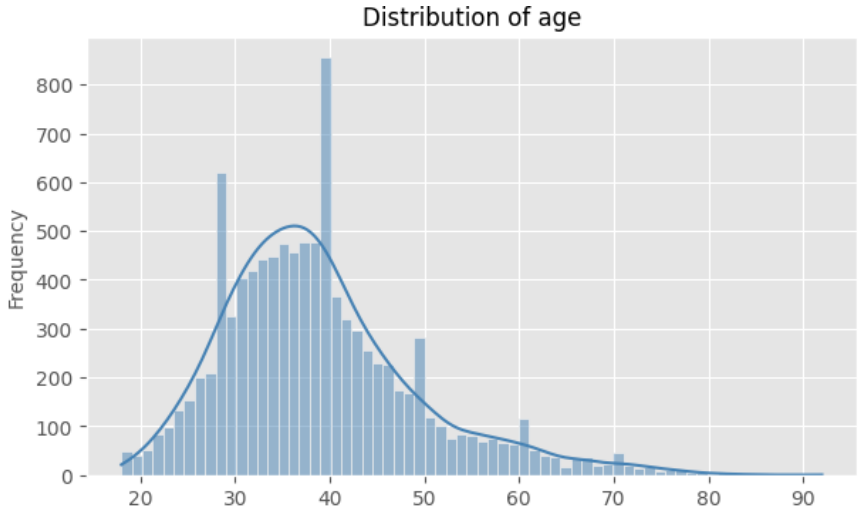


Figura 1: Boxplot do salário estimado com base no gênero

Através da análise do gráfico de boxplot, observa-se que ambos os gêneros não possuem diferenças significativas no salário estimado mostrado no exemplo que ambos os gêneros têm salários estimados com a mesma variância.

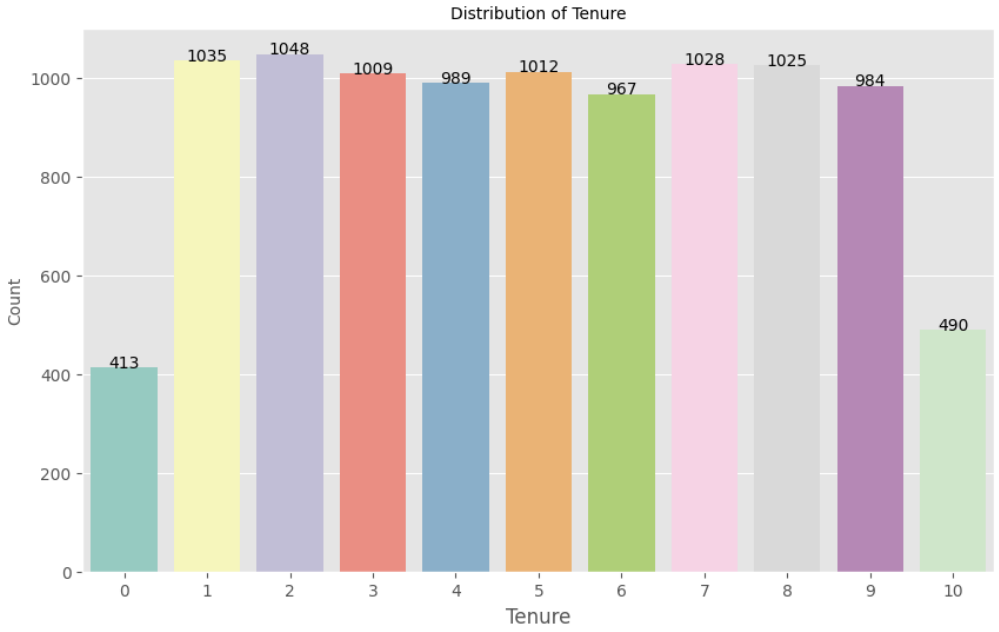
Idade (*age*)

A idade média dos clientes é de aproximadamente 38 anos e varia entre 18 a 92 anos. A maioria dos clientes (75%) têm menos de 44 anos. Observando-se a distribuição da idade na base nota-se uma distribuição próxima à normal com assimetria à direita e com pico de observações com 39-40 anos e outro pico com 28 anos:



Tempo no banco (*tenure)*

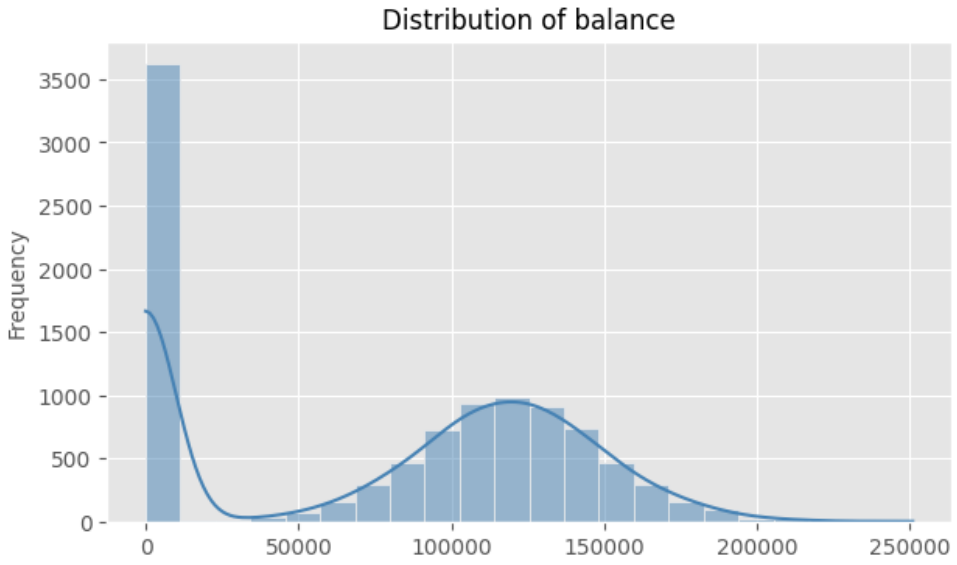
O tempo médio de relacionamento dos clientes com o Banco ABC é de aproximadamente 5 anos, variando de 0 a 10 anos. Clientes não completam um ano de relacionamento com o banco e que chegam a 10 anos somam 903 casos e representam 9%. A maioria dos clientes (50%) possui um relacionamento de até 5 anos, de acordo com a mediana. A análise do gráfico de distribuição de tempo de relacionamento dos clientes mostra que há uma divisão próxima nas faixas de tempo entre um e nove anos, variando de 984 (clientes há nove anos) a 1048 (clientes há dois anos).



Saldo (*balance)*

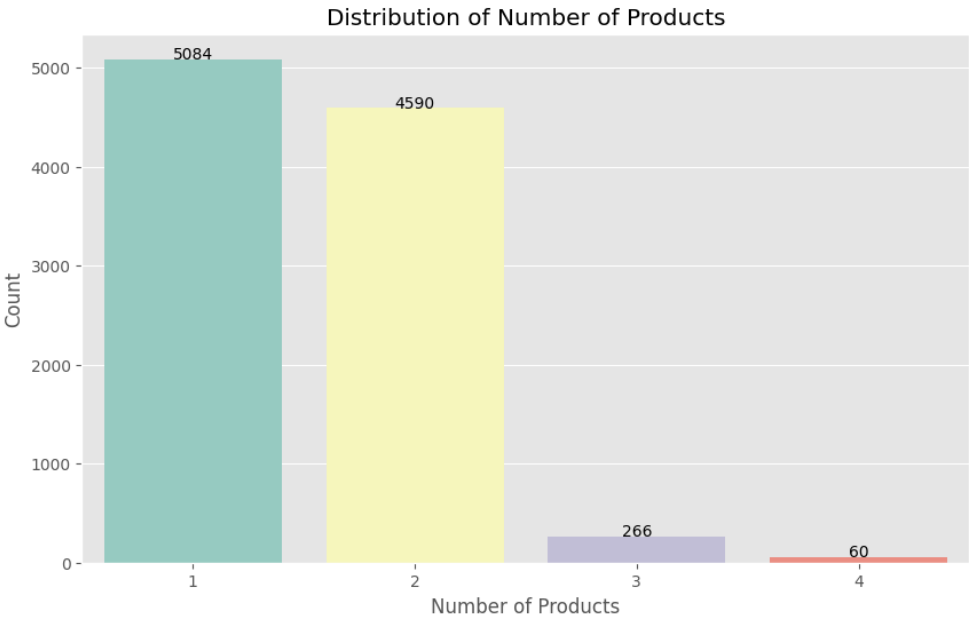
O saldo médio na conta dos clientes do banco é de aproximadamente 76.486 unidades monetárias, com uma variação significativa indicada pelo desvio padrão. Observa-se que há um considerável número de clientes com saldos zerados, o que, do ponto de vista de negócios,

poderia indicar a presença de contas inativas ou recentemente abertas. Excetuando-se as contas zeradas, o restante segue uma distribuição normal.



Número de produtos (*products_number*)

São quatro os produtos oferecidos pelo banco, identificados por numeração de um a quatro, sem detalhamento em relação a suas características. Apenas 60 (0,6%) clientes possuem quatro produtos ofertados pelo banco e 266 (2,7%) possuem dois. Quase 97% dos clientes possuem um ou dois produtos ofertados, sendo 5.084 os que têm apenas um produto e representando aproximadamente 51%.



Cartão de crédito (*credit_card*)

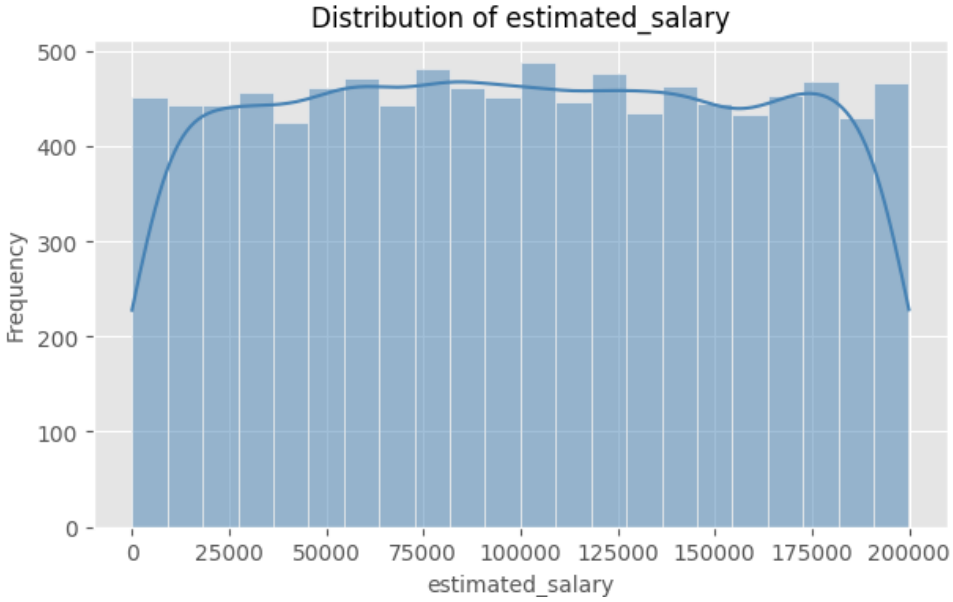
Dentre os clientes do banco, 7.055 (aproximadamente 70,5%) possuem ao menos um cartão de crédito e o restante não possui nenhum cartão de crédito.

Cliente ativo (*active_member*)

A variável binária indica se o cliente é ativo (1) ou não ativo (0). Aproximadamente 51,5% dos clientes são ativos.

Salário Estimado (*estimated_salary*)

Buscando observar se existe distribuição diferentes de salário na base, nota-se que o salário estimado é uniformemente distribuído, o que aponta para a não existência de um grupo de clientes predominante de uma determinada classe, conforme é possível observar pelo gráfico a seguir.



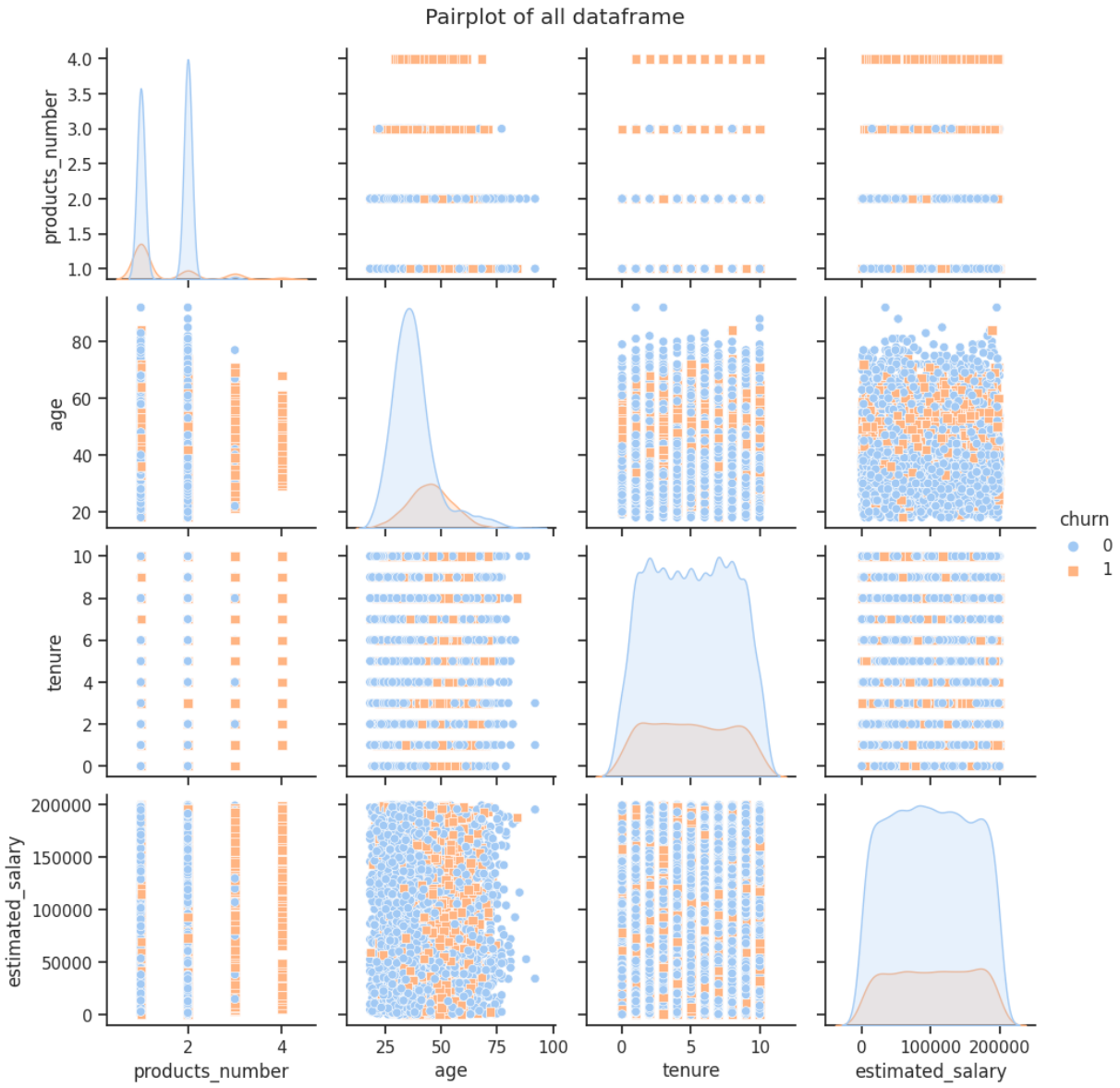
Churn

A taxa média de churn (rotatividade) é de aproximadamente 20,37%, indicando que 2.037 clientes deixaram o banco durante o período de referência dos dados da base. O restante e maior parte dos clientes (75%) não apresentam churn, conforme indicado pelo terceiro quartil.

Análise bidimensional de variáveis

Após a análise individual das variáveis da base, utilizamos a construção de um *pairplot* para analisar as relações entre pares de variáveis. Essa técnica de análise bidimensional permitiu observar, de maneira exploratória a partir da análise dos gráficos gerados, padrões e tendências das relações das variáveis entre si. Também influencia na escolha das variáveis mais relevantes para a correta parametrização do modelo:

A partir da primeira observação, nota-se que algumas das variáveis disponíveis agregam pouca relevância à análise e têm pouca influência no comportamento de churn que este trabalho se propõe a analisar, sendo elas: *customer_id, balance, credit_card, active_member*.



A partir de uma segunda análise, mais direcionada, nota-se que alguns fatores já aparecem como possíveis variáveis de atenção para a criação do modelo. Nota-se que o número de produtos é um aspecto relevante para o churn. Para os clientes com três e/ou quatro produtos, nota-se grande ocorrência de churn quando se observa os resultados em função da idade, tempo de no banco (tenure) e também pelo salário estimado.

A idade também aparece como um elemento relevante, uma vez que se observa algum grau de concentração da ocorrência de churn entre a faixa de 40 a 60 anos. Esse padrão de concentração se mantém e se repete (parcialmente) quando se observa a relação entre idade e salário estimado e entre idade e tenure.

Sobre o Banco ABC

A análise possível sobre o banco ABC a partir dos dados disponíveis aponta para um banco com presença em três países: França, Espanha e Alemanha. A presença de clientes na

França responde por 50% da base de clientes do banco e o restante divide-se de maneira mais ou menos igual entre os dois outros países. A divisão entre homens e mulheres também é equilibrada e não apresenta outros vieses de impacto a outras variáveis, como salário e saldo bancário.

A idade média dos clientes do banco é de aproximadamente 38 anos, sendo portanto um público mais "maduro". O tempo estimado de relacionamento dos clientes com o banco é de aproximadamente cinco anos e varia entre zero e dez anos. A distribuição do tempo de relacionamento dos clientes com o banco também é equilibrado se apresenta de maneira mais ou menos "constante", sem indicar de maneira clara um possível tempo de relacionamento crítico que possa justificar o churn.

A maioria dos clientes possuem apenas um produto ofertado pelo banco e praticamente todos os clientes (97%) têm até no máximo dois produtos. A quantidade de clientes que possuem três ou quatro produtos são ínfimos em relação ao total. A média de slário estimado é de cerca de 100 mil unidades e não é possível saber exatamente a correspondência real desse valor. Imagina-se que a unidade monetária seria o Euro se considerados os três países que aparecem na base, mas é impossível afirmar. A distribuição do salário é também mais ou menos homogênea em relação às várias faixas salariais.

Por fim, observa-se que o churn para o banco é de cerca de 20%. A taxa é alta para clientes com três e quatro produtos, com idade entre 40 e 60 anos, com mais tempo de relacionamento com o banco e em função de salários mais altos.

Pela análise inicial das variáveis, parece ser possível inferir, sob o ponto de vista do negócio, que o churn do Banco ABC ocorre desde o início da relação dos clientes com a instituição - como se observa nos gráficos que envolvem a variável *tenure.* Também parece semelhante ao padrão constante de churns independentemente da estimativa de salário. Ou seja, é algo constante para a instituição e está presente em toda a jornada do cliente - do início ao fim do relacionamento. Pela observação da ocorrência de churn em função da idade, parece razoável acusar um grau de insatisfação maior e específica entre clientes com idade entre 40 e 60 anos.

Desenvolvimento do modelo

Após a análise descritiva do modelo, nesta etapa são realizadas uma análise de redução de variáveis, a escolha do modelo com melhor acurácia e a otimização de hiperparâmetros. Essas escolhas visam a garantir que tenhamos um modelo capaz de ser o "vencedor" dentre os vários tipos de modelos possíveis. Também garante que os hiperparâmetros sejam otimizados, garantindo assim, um bom primeiro modelo de classificação para o banco de dados.

Redução de Variáveis

Para reduzir a quantidade de variáveis, optou-se por uma avaliação de *feature importances* do modelo extraído do algoritmo de classificação *Random Forest*. Com isso, é possível reduzir a quantidade de variáveis aplicadas. A imagem a seguir mostra a aplicação prática e compila os resultados obtidos. E baseando-se nestes resultados, também optou-se pela adoção de *features* que representem mais de 0,010 de importância e pela consequente remoção das variáveis de *gender* e *balance* do desenvolvimento do modelo.

```
      rfc = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)
      rfc.fit(X_train, y_train)
      importances = rfc.feature_importances_

      indices = np.argsort(importances)[::-1]

      print("Feature ranking:")

      for f in range(X_train.shape[1]):
          print("%d. feature %d (%f) ---> %s" % (f + 1, indices[f],
importances[indices[f]], df.columns[indices[f]]))
```

Resultado:

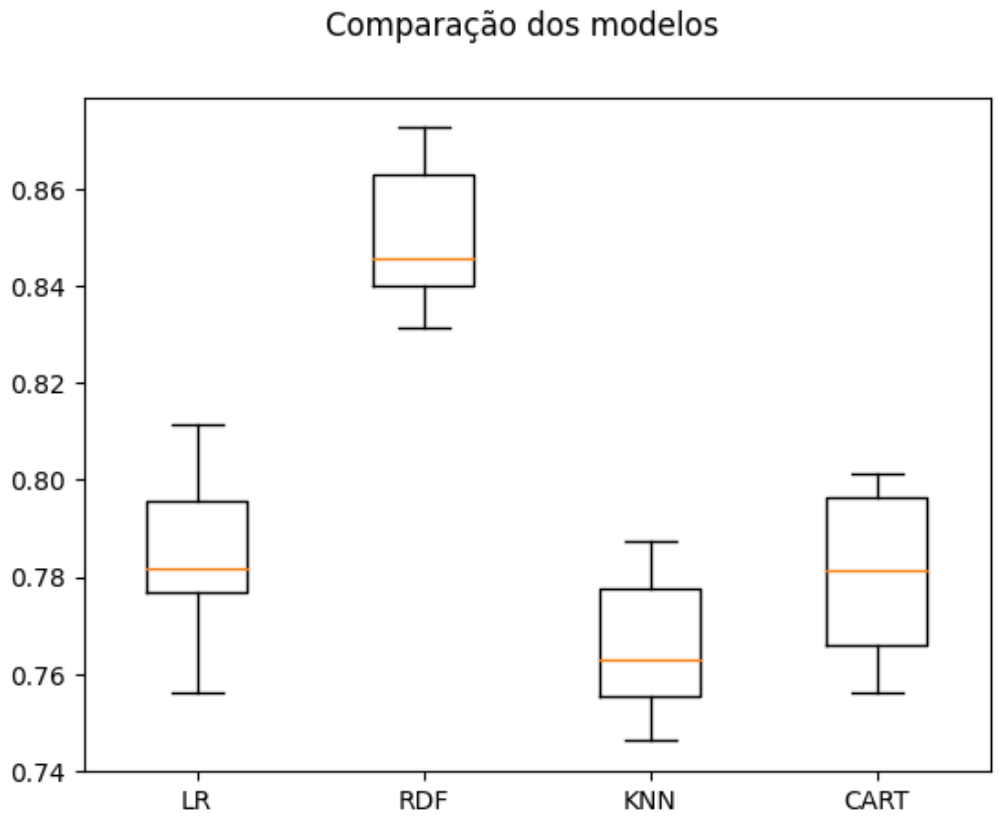
```
Feature ranking:
1. feature 2 (0.438578) ---> country
2. feature 5 (0.348334) ---> tenure
3. feature 7 (0.108098) ---> products_number
4. feature 4 (0.049455) ---> age
5. feature 0 (0.018763) ---> customer_id
6. feature 1 (0.016495) ---> credit_score
7. feature 8 (0.012672) ---> credit_card
8. feature 3 (0.006019) ---> gender
9. feature 6 (0.001586) ---> balance
```

Definição do melhor modelo de Classificação

Como ponto de partida, optou-se por analisar e testar quatro modelos de classificação diferentes. Após o teste de cada um, escolheu-se aquele que apresentou maior eficácia observada dentre todos. A seguir consta a lista dos modelos de classificação considerados:

- Regressão logística
- KNN
- Árvore de decisão
- Random Forest

Para uma melhor avaliação, realizou-se a definição de teste e controle por meio do kfold. Assim, cada modelo contou com informações aleatórias de treino e teste para avaliar o melhor comportamento da variável. A quantidade de partes divididas de todos os modelos testados é de 10 (n_splits=10). Não há avaliação de hiperparâmetros dos modelos neste teste inicial. Os resultados da acuracidade de cada modelo constam no *box plot* abaixo:



Pela análise dos gráficos e dispersão dos resultados encontrados em todos os modelos, é possível afirmar que o modelo com maior variação no resultado é o CART (modelo de árvore de decisão). O que apresentou o pior resultado foi o KNN. Já o modelo com o melhor resultado de acurácia foi o Random Forest.

Random Forest

Random Forest (Floresta Aleatória) é um algoritmo de aprendizado de máquina que pertence à categoria de métodos de *ensemble*. Ensemble refere-se à técnica de combinar vários modelos para criar um modelo mais robusto e preciso. A Random Forest é composta por um conjunto de árvores de decisão e cada árvore é treinada independentemente com uma amostra aleatória do conjunto de dados e, muitas vezes, com uma seleção aleatória de características em cada divisão. Durante o treinamento de cada árvore, uma amostra aleatória (com reposição) dos dados é utilizada introduzindo variabilidade nos conjuntos de treinamento de cada árvore, o que contribui para a diversidade do *ensemble*.

Em cada nó de divisão de uma árvore, uma seleção aleatória de características é considerada para determinar a melhor divisão. Isso ajuda a evitar que as árvores se especializem em características específicas do conjunto de dados.

Treinando Random Forest

Para critério comparativo realizamos dois treinamentos. Um com os hiperparâmetros e outro após a otimização de parâmetros. O código aplicado consta na imagem a seguir. A melhor acurácia encontrada no modelo sem a otimização dos hiperparâmetros foi de 85,65%.

```
      #### Treinando Random Forest com Kfold

      # Definir o número de folds para o K-Fold cross-validation
      n_folds = 5

      # Inicializar o KFold
      kf = model_selection.KFold(n_splits=n_folds, shuffle=True, random_state=42)

      # Inicializar o modelo Random Forest
      rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

      # Listas para armazenar os resultados de cada fold

      cross_val_scores = model_selection.cross_val_score(rf_model, X, y,
cv=n_folds, scoring='accuracy')

      # Exibir as pontuações de validação cruzada para cada fold
      print(f'Accuracy for each fold: {cross_val_scores}')
      print(f'Mean accuracy: {cross_val_scores.mean()}')
```

Otimização de parâmetros

GridSearch é uma técnica de otimização de hiperparâmetros que ajuda a encontrar a combinação ideal de valores de hiperparâmetros para um modelo de *machine learning*, como o Random Forest.

Para a nossa aplicação otimizamos os hiperparâmetros do modelo Random Forest, como o número de árvores, a profundidade máxima das árvores, o número mínimo de amostras para dividir um nó e o número mínimo de amostras em uma folha. O GridSearchCV treinou e avaliou o modelo com diferentes combinações desses hiperparâmetros e encontrou a combinação que produz o melhor desempenho de acordo com a métrica de avaliação especificada (neste caso, a acurácia).

O cuidado que se deve ter em relação ao algoritmo em grade é que esta pode ser computacionalmente intensiva, especialmente quando você tem muitas combinações de hiperparâmetros e um grande conjunto de dados. Portanto, é possível ajustar os hiperparâmetros do GridSearchCV para equilibrar o desempenho e o tempo de execução, como o número de dobras de validação cruzada (cv) e o número de núcleos de CPU a serem usados (n_jobs). Na imagem abaixo segue o trecho do código utilizado:

```
      # Defina os hiperparâmetros que você deseja otimizar
      param_grid = {
          'n_estimators': [10, 50, 100, 200], # Número de árvores na floresta
          'max_depth': [None, 10, 20, 30],    # Profundidade máxima de cada árvore
          'min_samples_split': [2, 5, 10],    # Número mínimo de amostras para
dividir um nó
          'min_samples_leaf': [1, 2, 4]       # Número mínimo de amostras em uma
folha
      }

      # Crie um objeto Random Forest Classifier
      rf = RandomForestClassifier()

      # Crie um objeto GridSearchCV
      grid_search = GridSearchCV(rf, param_grid, cv=5, scoring='accuracy')

      # Ajuste o modelo aos dados para encontrar a melhor combinação de parâmetros
      grid_search.fit(X_train, y_train)  # Substitua X e y pelos seus dados de
treinamento

      # Visualize os melhores parâmetros encontrados
      print("Melhores parâmetros encontrados: ", grid_search.best_params_)

      # Obtenha o melhor modelo treinado
      best_rf_model = grid_search.best_estimator_
```

Rodando a validação, obteve-se os seguintes valores de parâmetros otimizados para o algoritmo random forest:

```
Melhores parâmetros encontrados:  {'max_depth': 10,
'min_samples_leaf': 4, 'min_samples_split': 2, 'n_estimators': 50}
```

Logo, o número máximo de árvores foi de 200 e não 100, como utilizado no primeiro modelo. A profundidade máxima é de 10 a quantidade mínima de observações em cada folha é de 1. A divisão das folhas deve ocorrer ao menos com 5 indivíduos. Com base nesses resultados de saída, antecipou-se a complexidade da árvore para retornar a árvore de melhor acurácia. Como foi utilizado o kfold de 5, obteve-se um argumento de que, apesar da complexidade, não há overfitting ao prever churn com futuras bases de dados.

Em seguida, alterando os modelo com os parâmetros ótimos:

```
      #### Treinando Random Forest com Kfold para parâmetros otimizados

      # Definir o número de folds para o K-Fold cross-validation
      n_folds = 5

      # Inicializar o KFold
      kf = model_selection.KFold(n_splits=n_folds, shuffle=True, random_state=42)

      # Inicializar o modelo Random Forest
      rf_model = RandomForestClassifier(n_estimators=50, random_state=0, max_depth
= 10, min_samples_leaf = 4, min_samples_split = 2)

      # Listas para armazenar os resultados de cada fold

      cross_val_scores = model_selection.cross_val_score(rf_model, X, y,
cv=n_folds, scoring='accuracy')

      # Exibir as pontuações de validação cruzada para cada fold
      print(f'Accuracy for each fold: {cross_val_scores}')
      print(f'Mean accuracy: {cross_val_scores.mean()}')
```

Rodando o modelo com a acurácia encontrada do melhor dos cinco modelos testados foi de 86,55% de acurácia. E como resultado entre o valor predito e o y_teste:

	precision	recall	f1-score	support
0	0.87	0.97	0.92	1607
1	0.78	0.42	0.54	393
accuracy			0.86	2000
macro avg	0.83	0.69	0.73	2000
weighted avg	0.85	0.86	0.85	2000

Conclusão

São muitos os modelos de classificação disponíveis. Devido ao constante aumento de capacidade de processamento de dados, torna-se cada vez mais acessível e possível a comparação de eficiência de vários desses modelos. Para nossa amostra de dez mil clientes, fomos capazes de rodar os modelos usando *cross validation* e o utilizar *grid search* para identificar os melhores hiperparâmetros. O *grid search* foi a etapa de desenvolvimento do modelo que levou mais tempo de processamento no Google collab.

Quando comparamos o *random forest* otimizado e o random forest não otimizado tivemos uma diferença na acuracidade de 0,9%. Quando vemos os hiperparâmetros da árvore que utilizou o *grid search* vimos que não limitava muito o crescimento da árvore. apenas o parâmetros de árvores geradas foi menor que a default utilizada(100 árvores) contra 50 árvores do modelo otimizado. As demais sugestões relacionadas a quantidade mínima de dados na folha (4 observações) e a quantidade de dados para divisão (2 observações) do nó um valor baixo. Um modelo de machine learning muito complexo perde a característica explicativa caso deseja-se, mas como esse não é o objetivo principal do modelo Random Forest isso não é um problema relevante.

Tendo considerado os principais desafios encontrados no modelo proposto, entende-se como aceitável a acurácia encontrada de 86,55%. O *random forest* apresentou um desempenho ‘superior’ ao dos outros modelos testados em suas cinco amostras que passaram por validação cruzada. Portanto esse foi o algoritmo mais adequado a ser escolhido.

Tivemos um f1 score bom para para prever o não churn porém um f1 não tão bom na precisão do churn obtendo o valor de 0,54 com precisão de 0,78. Uma análise do corte de classificação do churn possa aumentar a precisão para casos de churn de clientes que o modelo desenvolvido não é tão capaz de identificar.