Stacking para um problema de classficação

Leonardo Thurler, Rodrigo Veloso
Universidade Federal Fluminense

Introdução

O objetivo desse trabalho é implementar e avaliar o método de Stacking. O método será usado para resolver um problema de classificação binária, ou seja, um problema de aprendizado supervisionado onde os rótulos (classes) possuem apenas dois valores distintos.

É necessário então a escolha de um dataset compatível com este tipo de problema, possuindo um atributo categórico binário para servir de rótulo. O dataset deve ainda atender alguns critérios que serão detalhados na próxima seção.

Todos os métodos utilizados na composição do Stacking também devem ser avaliados individualmente através de um processo de validação cruzada, onde serão obtidas as métricas: acurácia, precision, recall, F-measure, assim como as curvas ROC e matrizes de confusão, essas métricas são clássicas, para maior explicação sobre elas ver Marsland (2009).

O objetivo final do trabalho é comparar as diferentes metodologias de classificação, identificar a mais adequada para a base de dados escolhida e as vantagens e desvantagens de cada abordagem.

Dataset

Escolha do dataset e motivação

A base escolhida para esse trabalho foi a base Credit Card Customers (CCC) postada por Sakshi Goyal (Sakshi Goyal, 2020). A base possui 23 atributos e 10127 exenplos, dentre os atributos, 6 são categóricos e os demais são numéricos.

Essa base contem os dados de clientes de um banco. O objetivo proposto pelo autor da base é prever a partir dos dados quais os clientes com maior probabilidade de cancelar seus serviços de cartão crédito, possibilitando então um atendimento direcionado e melhores serviços destinados à esse grupo de clientes com a expectativa de reverter suas decisões. Note então que se deseja resolver um problema de classificação binária, onde deve-se prever se um cliente irá cancelar seus serviços ou não.

A CCC foi escolhida por ser destinada para a solução de um problema de classificação binária, possuir usabilidade avaliada em 10.0 por usuários do kaggle e por se tratar de uma base de dados de um contexto empresarial, tendo então uma aplicação prática relevante. Um outro ponto positivo é que a base não possui valores faltantes.

Descrição do dataset

A base de dados possui 6 atributos categóricos, dentre esses está o atributo 'Attrition_Flag'. O atributo 'Attrition_Flag' é categórico e binário e será usado como rótulo, este atributo indica se um cliente cancelou ou não seu serviço de cartão de crédito. Apenas 16.07% dos clientes cancelaram o serviço, indicando que as classes são altamente desbalanceadas.

Os demais atributos categóricos são: 'Gender', 'Education_Level', 'Marital_Status', 'Income_Category', 'Card_Category', que representam respectivamente: o gênero, o nível de educação, o estado civil, a categoria de renda e a categoria do cartão de crédito de cada cliente. As distribuições dos atributos categóricos se encontram nas figuras 1, 2, 3, 4 e 5.

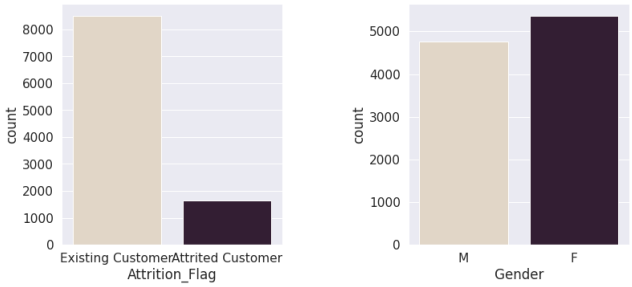

Figure 1: Distribuição do atributo 'Attrition_Flag' (esquerda) e 'Gender' (direita)

Este trabalho foi realizado usando o scikit learn, que não aceita atributos não numéricos. É então necessário transformar os atributos categóricos em atributos numéricos. Cada possível categoria desse tipo de atributo recebeu um número inteiro de 0 ao número total de categorias menos 1. Para exemplificar, considere o atributo gênero, para esse atributo existem duas possíveis categorias: masculino e feminino, neste caso a categoria relativa ao sexo masculino recebeu o número 0 e a relativa ao feminino recebeu o número 1.

Os demais 17 atributos são numéricos. O 'CLIENTNUM' é o identificador do cliente, sendo a chave primária da

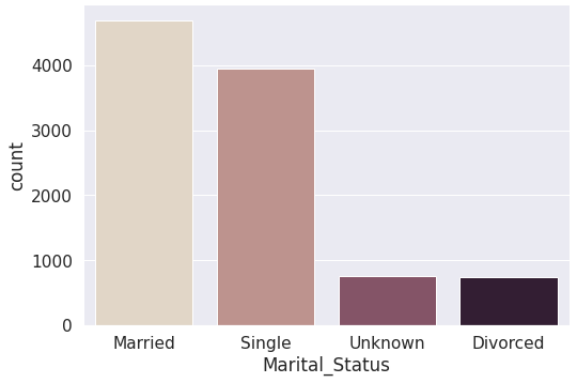

Figure 2: Distribuição do atributo 'Marital_Status'

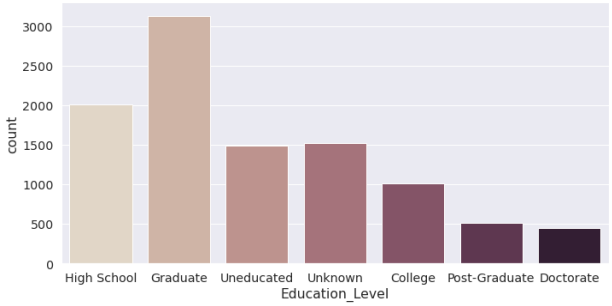

Figure 3: Distribuição do atributo 'Education_Level'

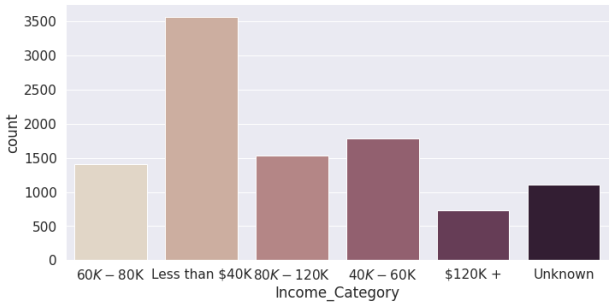

Figure 4: Distribuição do atributo 'Income_Category'

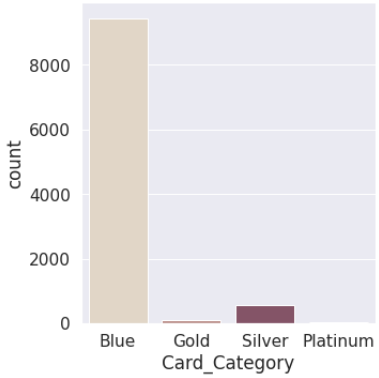

Figure 5: Distribuição do atributo 'Card_Category'

base de dados, este atributo foi removido pois não existe qualquer relação entre o identificador e o cancelamento do serviço, podendo trazer malefícios aos classificadores, como overfitting. 'Customer_Age' indica a idade do cliente. 'Dependent_count' diz quantos dependentes o cliente tem. O tempo total que um cliente consumiu algum dos produtos do banco é dado por 'Months_on_book'. O número total de produtos que o cliente já consumiu é dado por 'Total_Relationship_Count'. 'Months_Inactive_12_mon' é um atributo que informa o número de meses que o cliente ficou inativo durante os últimos 12 meses. 'Contacts_Count_12_mon' é um atributo que informa quantas vezes o cliente entrou em contato com o banco durante os últimos 12 meses. 'Credit_Limit' é o limite do cartão de crédito. 'Total_Revolving_Bal' indica a quantia não paga ao final de um ciclo de pagamento. 'Avg_Open_To_Buy' é um atributo que indica a diferença entre o limite do cartão e o que já foi gasto. O atributo 'Total_Amt_Chng_Q4_Q1' indica a diferença entre a quantia transacionada no último trimestre e a quantia transacionada no primeiro. O atributo 'Total_Trans_Amt' indica a quantia total transacionada nos últimos 12 meses. O atributo 'Total_Trans_Ct' indica o número total de transações feitas nos últimos 12 meses. 'Total_Ct_Chng_Q4_Q1' é relativo a diferença no número de transações entre o último e o primeiro trimestre do ano. 'Avg_Utilization_Ratio' é a taxa média de utilização do cartão de crédito. Os dois últimos atributos da base são relativos à probabilidades calculadas pelo algoritmo Naïve Bayes, o autor da base recomenda a remoção desses atributos e sua recomendação foi seguida. As figuras de número 6 a 12, contem gráficos com as distribuições dos atributos numéricos.

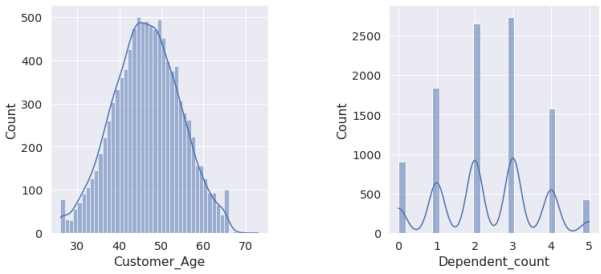

Figure 6: Distribuição dos atributos 'Customer_Age' (esquerda) e 'Dependent_count' (direita)

Análise de componentes principais

Quanto maior o número de dimensões da base de dados (número de atributos), maior o custo computacional da grande maioria dos algoritmos de aprendizado e maior o número de exemplos de treinamento requeridos. Reduzir a dimensão da base pode contribuir para remoção de ruídos e aumentar a performance dos algoritmos de aprendizado.

A análise de componentes principais (PCA) é uma técnica de redução de dimensionalidade. A ideia principal dessa

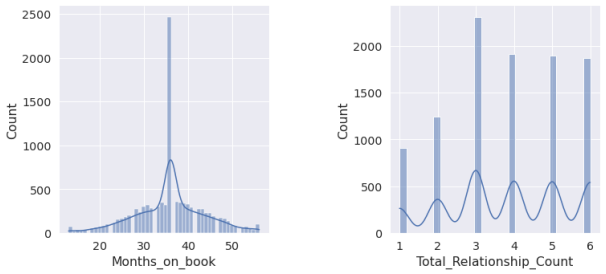

Figure 7: Distribuição dos atributos 'Months_on_book' (esquerda) e 'Total_Relationship_Count' (direita)

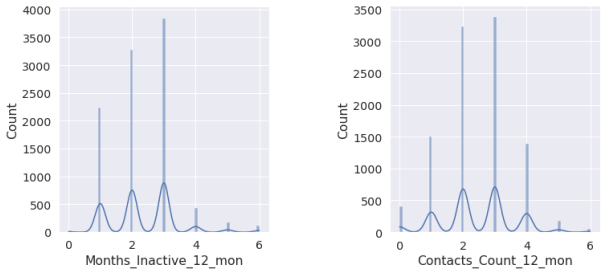

Figure 8: Distribuição dos atributos 'Months_Inactive_12_mon' (esquerda) e 'Contacts_Count_12_mon' (direita)

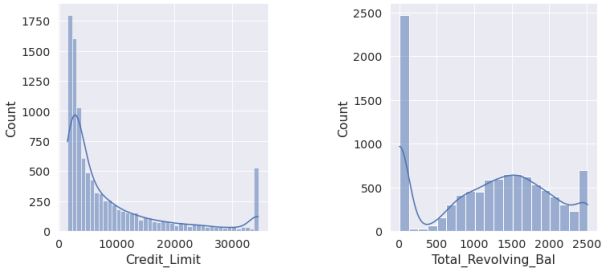

Figure 9: Distribuição dos atributos 'Credit_Limit' (esquerda) e 'Total_Revolving_Bal' (direita)

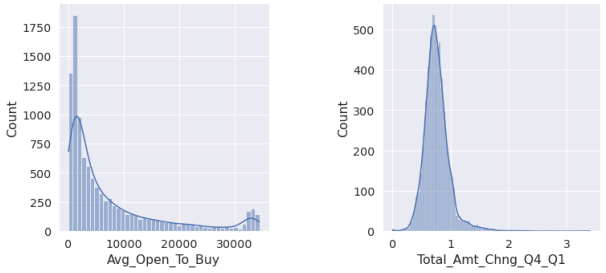

Figure 10: Distribuição dos atributos 'Avg_Open_To_Buy' (esquerda) e 'Total_Amt_Chng_Q4Q1' (direita)

metodologia é identificar as direções que mais contribuem para a variação da base de dados, as componentes principais. O algoritmo escolhe cada componente de cada vez,

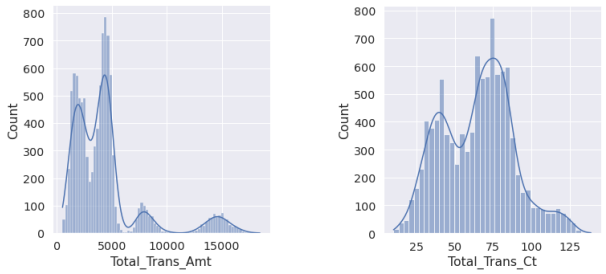

Figure 11: Distribuição dos atributos 'Total_Trans_Amt' (esquerda) e 'Total_Trans_Ct' (direita)

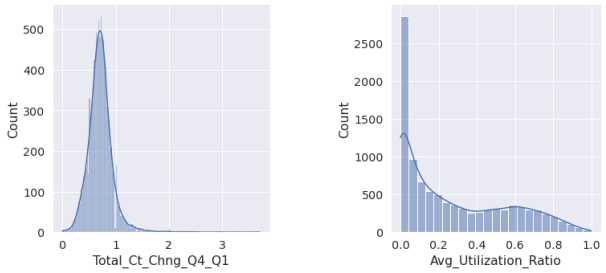

Figure 12: Distribuição dos atributos 'Total_Ct_Chng_Q4_Q1' (esquerda) e 'Avg_Utilization_Ratio' (direita)

priorizando as direções que possuem a maior variância, de forma que cada componente é normal à anterior. Para mais detalhes sobre PCA ver Marsland (2009).

A PCA foi aplicada à base CCC. Um gráfico mostrando a variação em função do número de componentes se encontra na figura 13. Note que com poucas componentes já se consegue obter a variação total da base de dados original. Com apenas 3 componentes principais se é obtido 0.999 da variação da base original.

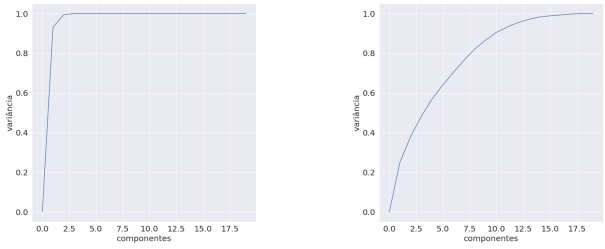

Figure 13: Variação em função do número de componentes principais para a base original (esquerda) e base normalizada (direita)

A PCA também foi aplicada à base CCC normalizada, a normalização foi feita utilizando a equação 1,

$$v' = \frac{v - min_{Atr}}{max_{Atr} - min_{Atr}} \quad (1)$$

Onde v' é um valor do atributo Atr normalizado, v é o valor original, min_{Atr} é o menor valor de Atr e max_{Atr} é o maior. Um gráfico mostrando a variação em função do número de componentes se encontra na figura . Note que o número de componentes necessárias para recuperar a variação total é bem superior em relação a base não normalizada.

Stacking e seus Pipelines

Stacking é um método de ensemble e agregação de resultados, em que a saída de um método de aprendizado de máquina se torna um atributo para a criação de uma nova base de dados. Assim, os atributos dessa nova base de dados serão as respostas dos métodos executados anteriormente. Essa nova base é então utilizada para treinar um meta-modelo, que irá fazer uma previsão com base nas previsões de cada metodologia individual.

Essa metodologia de ensemble é interessante pois é flexível, podendo utilizar metodologias de aprendizado heterogêneas ou homogêneas (pipelines). Como todo ensemble, se as metodologias individuais foram escolhidas de forma adequada, os resultados obtidos através do Stacking podem superar os resultados dos pipelines utilizados em sua composição (Pavlyshenko, 2018). Os pipelines devem ser escolhidos de modo a se complementarem.

Neste trabalho serão utilizados 10 pipelines distintos para a composição de um modelo de stacking. Os pipelines foram escolhidos primeiramente considerando algoritmos de classificação de diferentes paradigmas. Os algoritmos escolhidos foram árvore de decisão, Naïve Bayes (NB) e k Nearest Neighbor (kNN). As próximas seções do trabalho possuem um breve comentário sobre cada um deles, mas por serem algoritmos clássicos não serão explicados em detalhes. Para uma explicação detalhada sobres esses algoritmos ver Han, Kamber, and Pei (2011).

Árvore de decisão

A Árvore de decisão (AD) é um algoritmo que busca utilizar um conjunto de atributos de uma instância, para predizer o valor da sua classe (rótulo). Na etapa de treino, a árvore de decisão busca identificar e mapear os atributos que melhor separam as classes. Esse mapeamento é armazenado em uma estrutura de dados semelhante a uma árvore. Onde cada folha representa uma classe, os nós intermediários representam atributos que foram considerados na divisão e as arestas possuem os valores desses atributos. A etapa de teste consiste em verificar os atributos da instância de teste e percorrer a árvore da raiz até chegar a uma folha para classificar a tupla. Um exemplo de modelo de árvore de decisão se encontra na figura 14.

Foi utilizada a árvore de decisão implementada no pacote scikit learn. Essa implementação possui um hiperparâmetro principal, a profundidade máxima da árvore (max_depth). Para estimar o valor ótimo para esse hiperparâmetro, foi realizada uma validação cruzada estratificada para max_depth entre 1 e 100. Essa estratégia busca dividir a base em K partições, após isso são realizadas K iterações onde em cada iteração uma partição é utilizada para a base de teste e as

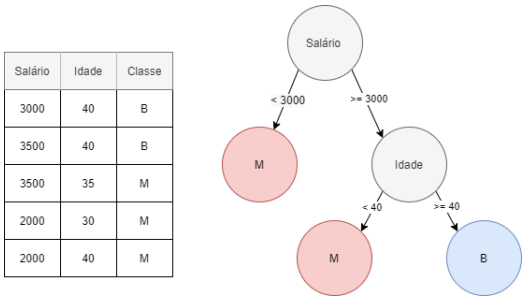

Figure 14: Exemplo de árvore de decisão.

outras são utilizadas para a base de treino. A divisão estratificada, indica que cada uma das partições geradas, buscará manter a proporção de elementos de cada classe encontrada na base original. Uma exemplificação desse processo se encontra na figura 15. A validação cruzada estratificada foi executada na base em seu formato padrão, sem nenhum tipo de tratamento, na base após uma transformação dada pela PCA com 5 componentes principais e também na base após normalização. Os resultados desses processos se encontram na figura 16.

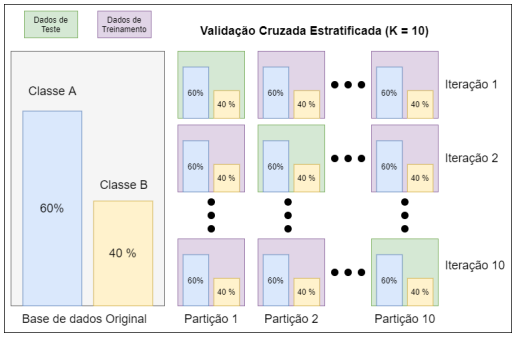

Figure 15: Exemplificação do processo de validação cruzada estratificada para o K=10.

Os melhores resultados foram obtidos para as bases padrão e normalizada, com ligeira vantagem para a base normalizada. Foram escolhidos então max_depth=7 para ambas as bases padrão (AD1) e normalizada (AD2) como pipelines para o Stacking pois apresentaram os melhores resultados. Foram escolhidos também como pipelines max_depth=2 (AD3) e max_depth=20 (AD4) para a base normalizada. Os últimos dois pipelines foram escolhidos para garantir uma maior variação entre as metodologias.

Naïve Bayes

O classificador Naïve Bayes (NB) é um classificador estatístico. Como todo classificador Bayesiano, é baseado no teorema de Bayes Han, Kamber, and Pei (2011). NB assumi que os valores de um atributo da base de dados são independentes dos valores de outros atributos da base. Essa hipótese é feita para simplificar os cálculos necessários, por isso o algoritmo é considerado Naïve (ingênuo).

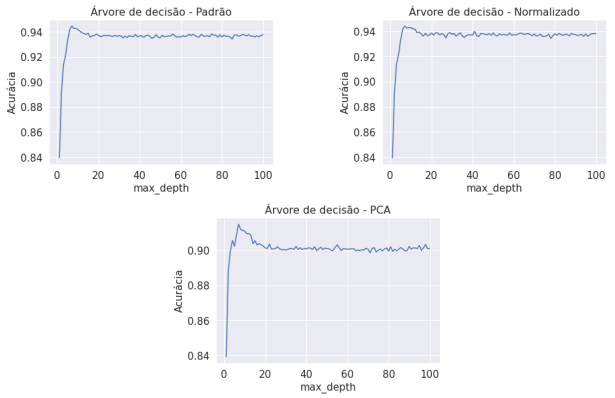

Figure 16: Acurácia obtida utilizando árvore de decisão em função de max_depth para diferentes formatos da base de dados.

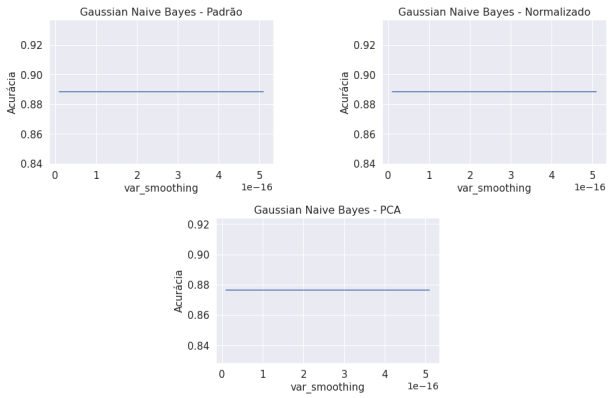

Figure 17: Acurácia obtida utilizando NB em função de var_smoothing para diferentes formatos da base de dados.

A ideia principal é usar o teorema de Bayes para calcular a probabilidade ($P(y_i|X)$) de uma tupla X da base de dados pertencer a cada uma das classes y_i. O classificador decide que X é da classe y_i se $P(y_i|X) > P(y_j|X)$ para todo j diferente de i, ou seja, escolhe a classe com a maior probabilidade de X pertencer. Calcular essas probabilidades de maneira exata não é uma tarefa simples, o NB não calcula as probabilidades de forma exata, mas de forma precisa o suficiente para ser capaz de rankear as probabilidades e selecionar a maior.

Estudos apontam que classificadores do tipo NB se comparam com árvore de decisão em termos de performance e algumas arquiteturas de redes neurais, apresentando também alta acurácia e baixo tempo de processamento quando aplicados à grandes bases de dados Han, Kamber, and Pei (2011). Sendo assim é um ótimo candidato para ser aplicado à base de dados CCC, pois é comparável à arvore de decisão e ao mesmo tempo é de um paradigma diferente.

Foi utilizada o NB implementado no pacote scikit learn. Essa implementação possui um hiperparâmetro principal, var_smoothing. Para estimar o valor ótimo para esse hiperparâmetro, foi realizada uma validação cruzada estratificada para var_smoothing entre 0.1×10^{-16} e 5×10^{-16} e o resultado desse processo se encontra na figura 17.

Note que a acurácia não depende de var_smoothing, e não há diferença significativa entre a acurácia obtida com a base padrão e a base normalizada. Foram então escolhidos o NB com var_smoothing = 1×10^{-9}, por ser o valor padrão para este hiperparâmetro. Esta configuração foi utilizada na base padrão (NB1) e na base transformada por PCA (NB2) como pipelines para o Stacking.

k Nearest Neighbor

Os métodos discutidos anteriormente são considerados classificadores do tipo eager. Classificadores eager geram um modelo a partir da base de dados antes de classificar novas tuplas.

O kNN é um método lazy, ou seja, ele só executa uma generalização com base nos dados de treino quando recebe um nova tupla para avaliar. Dessa forma o kNN realiza pouco ou nenhum trabalho na fase de treino, a fase mais custosa é a de predição. Classificar muitas tuplas pode ser uma tarefa computacionalmente cara Han, Kamber, and Pei (2011).

É um algoritmo de um paradigma diferente dos anteriores, se baseando no aprendizado por analogia. O kNN classifica um tupla X considerando uma votação simples entre os k vizinhos mais próximos de X que pertencem à base de treino. A noção de proximidade se dá utilizando uma métrica de distância, sendo a principal métrica utilizada a distância Euclidiana.

Foi utilizada o kNN implementado no pacote scikit learn. Essa implementação possui um hiperparâmetro principal, k. Para estimar o valor ótimo para esse hiperparâmetro, foi realizada uma validação cruzada estratificada para k entre 1 e 29 e o resultado desse processo se encontra na figura 18.

Figure 18: Acurácia obtida utilizando kNN em função de k para diferentes formatos da base de dados.

A base transformada por PCA obteve os melhores resultados em termos de acurácia obtendo um valor de 0.8952 contra 0.8951 da base padrão e 0.8902 da normalizada, mas a base normalizada obteve maior precision chegando ao valor de 0.8169 contra 0.7174 da base padrão e 0.7176 da base

transformada por PCA. Foram escolhidos como pipelines k = 9 (kNN3) para a base PCA e k = 11 (kNN1) para a base normalizada por causa do melhor desempenho e termos de acurácia e precision respectivamente. Foram também escolhidos k = 1 (kNN2) e k = 20 (kNN4) utilizando a base PCA para garantir maior variação entre os pipelines.

Resultados

Para aplicar o Stacking, as previsões de cada pipeline foram usadas como atributos de um novo dataset. É necessário então treinar um meta modelo utilizando está nova base, que irá fazer uma previsão com base nas previsões de cada metodologia individual. O meta modelo pode ser gerado através de uma metodologia de classificação qualquer. Foram avaliadas, utilizando o processo de validação cruzada estratificada, as 3 metodologias comentadas anteriormente para gerar o meta modelo: AD, NB e kNN. Os resultados dessa avaliação se encontram na tabela 1, os algoritmos foram utilizados com seus hiperparâmetros definidos pelo default da implementação do scikit learn. É notório que o kNN gerou o melhor meta modelo de classificação, por isso foi escolhido. Para facilitar a reprodução dos resultados, o k default é igual à 5.

Table 1: Resultados preliminares do Stacking para diferentes geradores de meta modelo.

	acurácia	desvio padrão
AD	0.925	0.005
NB	0.922	0.009
kNN	0.948	0.01

Com o Stacking totalmente definido, uma avaliação final foi feita considerando as métricas acurácia, precision, recall e F-measure, assim como matrizes de confusão e curvas ROC para cada um dos pipelines e para o Stacking. Os valores de acurácia, precision, recall e F-measure se encontram na tabela 2, as matrizes de confusão nas figura 20, 21, 22 e 23, já as curvas ROC podem ser observadas na figura 19. A classe positiva (P) foi definida como a classe 'Existing Customer' e a classe negativa (N) como 'Attrited Customer', ou seja, P para clientes que não cancelaram seus serviços e N para os que cancelaram.

Como desejado, o Stacking obteve um resultado melhor que todos os pipelines individualmente em todas as métricas de avaliação.

Conclusões

A base de dados CCC foi escolhida para aplicação do método de aprendizado de máquina Stacking. A escolha se deve ao fato da base conter número de atributos e exemplos suficientes, possuir atributos categóricos e numéricos, possuir usabilidade avaliada em 10.0 por usuários do kaggle e por se tratar de uma base de dados com aplicação prática relevante. A base foi utilizada para resolver um problema de classificação binária, onde se deseja prever se um cliente irá cancelar ou não o seu serviço de cartão de crédito.

Table 2: Acurácia, precision, recall e F-measure para os pipelines do Stacking e Stacking.

	acurácia	precision	recall	F-measure
AD1	0.943	0.842	0.800	0.820
AD2	0.944	0.843	0.803	0.823
AD3	0.891	0.735	0.510	0.602
AD4	0.937	0.805	0.803	0.804
NB1	0.888	0.652	0.652	0.652
NB2	0.876	0.662	0.467	0.547
kNN1	0.890	0.816	0.408	0.544
KNN2	0.871	0.603	0.578	0.590
kNN3	0.895	0.717	0.573	0.637
kNN4	0.891	0.735	0.504	0.598
Stacking	0.946	0.855	0.805	0.829

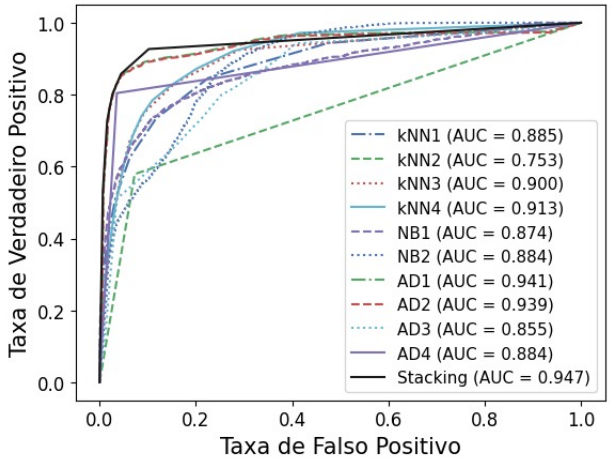

Figure 19: Curvas ROC para todos os pipelines e para o Stacking, AUC é a área abaixo da curva.

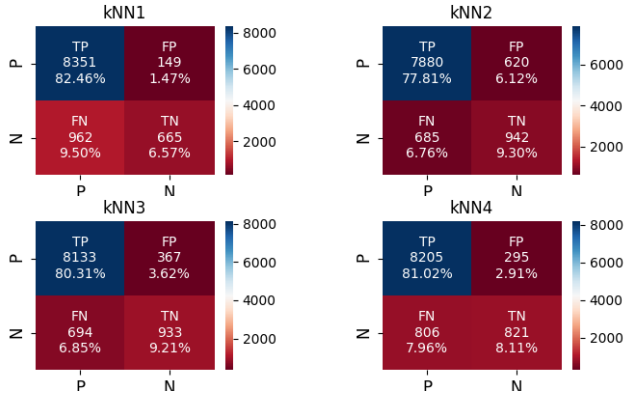

Figure 20: Matrizes de confusão para pipelines com kNN, TP é referente aos verdadeiros positivos, FP aos falsos positivos, FN aos falsos negativos e TN aos verdadeiros negativos.

Para implementar o Stacking é preciso primeiro definir 10 pipelines diferentes. Os pipelines foram escolhidos uti-

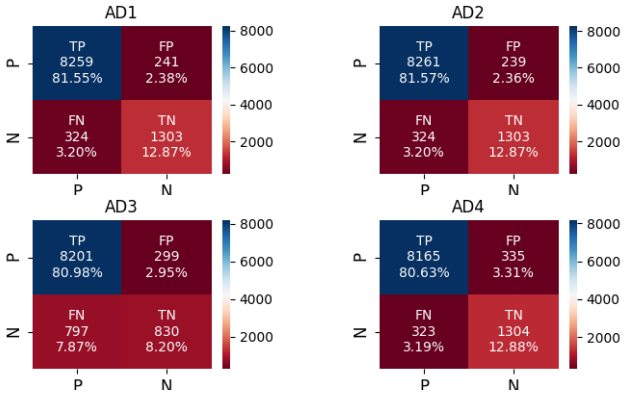

Figure 21: Matrizes de confusão para pipelines com AD, TP é referente aos verdadeiros positivos, FP aos falsos positivos, FN aos falsos negativos e TN aos verdadeiros negativos.

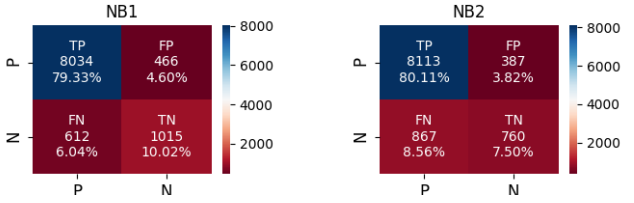

Figure 22: Matrizes de confusão para pipelines com NB, TP é referente aos verdadeiros positivos, FP aos falsos positivos, FN aos falsos negativos e TN aos verdadeiros negativos.

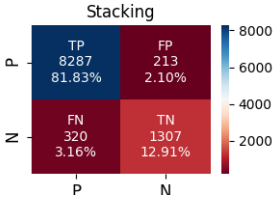

Figure 23: Matrizes de confusão para o Stacking, TP é referente aos verdadeiros positivos, FP aos falsos positivos, FN aos falsos negativos e TN aos verdadeiros negativos.

lizando como base os algoritmos de classificação AD, NB e kNN, considerando valores de hiperparâmetros diferentes e diferentes configurações da base de dados (normalização, PCA). Após um processo de validação cruzada estratificada, foram escolhidos os pipelines com maior acurácia. Foram também escolhidos escolhidos alguns pipelines para garantir uma maior variação entre os mesmos, é preciso combinar bons modelos individuais mas que sejam diferentes entre si.

Os resultados dos pipelines foram combinados em um novo dataset, que foi usado para treinar um meta modelo. O algoritmo escolhido para gerar o modelo foi o kNN com k=5, pois apresentou desempenho maior que os obtidos pelo NB e AD, em relação a acurácia.

Os pipelines individuais e o Stacking foram então avaliados utilizando como métricas: acurácia, precision, recall, F-measure, matriz de confusão e curva ROC (assim como AUC). Os resultados dessa avaliação mostram que o Stacking obteve maior desempenho em todas as métrica de avaliação utilizadas, tendo um desempenho um pouco acima dos obtidos pelas AD2 e AD1. O Stacking superou os pipelines individuais.

Esse resultado era desejado mas não era garantido. Meta modelos gerados pelo NB e AD, por exemplo, geram um resultado pior que os pipelines AD2 e AD1. A escolha do algoritmo para gerar o meta modelo e sua calibração são fases importantes da contrução do Stacking e não podem ser negligenciadas. A escolha adequada dos pipelines também é fundamental para garantir um bom desempenho, os pipelines precisam ser bons individualmente mas também devem ter boa sinergia, o que é difícil de se determinar apiori.

Em termos gerais, o Stacking apresentou um excelente resultado, obtendo uma acurácia acima de 94%.

Referências

Han, J.; Kamber, M.; and Pei, J. 2011. *Data Mining Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems, 3rd edition.

Marsland, S. 2009. *Machine Learning - An Algorithmic Perspective*. Chapman and Hall / CRC machine learning and pattern recognition series. CRC Press.

Pavlyshenko, B. 2018. Using stacking approaches for machine learning models. In *2018 IEEE Second International Conference on Data Stream Mining Processing (DSMP)*, 255–258.

Sakshi Goyal. 2020. Credit card customers. Disponível em: `https://www.kaggle.com/sakshigoyal7/credit-card-customers`. Acesso em: 2 dezembro 2020.