Esse relatório busca explicar a criação de um classificador para apoio à decisão de aprovação de crédito. A ideia é identificar, dentre os clientes que solicitam um produto de crédito (como um cartão de crédito ou um empréstimo pessoal, por exemplo) e que cumprem os pré-requisitos essenciais para a aprovação do crédito, aqueles que apresentem alto risco de não conseguirem honrar o pagamento, tornando-se inadimplentes.

Bibliotecas fundamentais:

os: Usada para manipulação do sistema de arquivos, como alterar o diretório de trabalho.
pandas: Manipulação de dados estruturados em DataFrames.
numpy: Operações matemáticas e manipulação de arrays.
matplotlib.pyplot: Geração de gráficos estáticos.
warnings: Gerenciamento de mensagens de aviso no Python.

Pré-processamento:

MinMaxScaler: Normalização dos dados numéricos para um intervalo fixo (geralmente entre 0 e 1).
LabelBinarizer: Codificação de variáveis categóricas em uma representação binária.

Modelos e algoritmos:

KNeighborsClassifier: Implementação do algoritmo K-Nearest Neighbors (KNN) para classificação.
LogisticRegression: Regressão logística para tarefas de classificação.
RandomForestClassifier: Modelo de ensemble baseado em árvores de decisão.

Validação de modelos:

cross_val_score: Cálculo de métricas de validação cruzada.
KFold, LeaveOneOut: Estratégias de divisão dos dados para validação cruzada.
HalvingGridSearchCV, GridSearchCV: Busca de melhores hiperparâmetros, com o primeiro sendo uma versão mais eficiente que reduz o número de candidatos progressivamente.

Outras ferramentas:

scipy.stats: Ferramentas estatísticas avançadas, como testes de hipóteses.
Comentários gerais:

Essas importações são para:

Pré-processamento (tratamento e normalização dos dados).
Modelagem (construção e ajuste de modelos preditivos).
Validação (uso de técnicas avançadas para avaliar a performance).

Passo 1: Carregamento dos Dados

Carrega dois conjuntos de dados em formato CSV: um para treino (conjunto_de_treinamento.csv) e outro para teste (conjunto_de_teste.csv).
delimiter=',': Define que os valores estão separados por vírgulas.
decimal='.': Indica que o ponto é usado como separador decimal.

Propósito:
Fornece os dados de entrada para o modelo.
O conjunto de treino é usado para ajustar os modelos, e o conjunto de teste avalia o desempenho em novos dados.

Passo 2: Tratamento de Valores Ausentes no Conjunto de Treino

Identifica valores ausentes (NaN) em colunas específicas:
meses_na_residencia: Tempo de residência do solicitante.
profissao_companheiro: Profissão do companheiro, se aplicável.
tipo_residencia: Tipo de residência (própria, alugada, etc.).
Substitui valores ausentes pela mediana dos valores na coluna correspondente.

Motivação:
A mediana é usada porque não é afetada por outliers, sendo robusta para dados numéricos.

Propósito:
Garante que as colunas não contenham valores ausentes, o que pode causar erros nos modelos preditivos.

Passo 3: Tratamento Similar no Conjunto de Teste

Realiza o mesmo processo para o conjunto de teste, garantindo consistência.
Motivação:

O modelo deve lidar com dados de teste no mesmo formato e padrão do conjunto de treino.

Passo 4: Substituição de Estados por Regiões

Substitui os nomes dos estados por suas respectivas regiões (sudeste, norte, etc.).
Aplica a transformação tanto no conjunto de treino quanto no de teste.

Motivação:
A substituição reduz a cardinalidade (número de categorias únicas), agrupando estados em um número menor de categorias (regiões).
Isso simplifica a modelagem, pois variáveis com menor cardinalidade são mais fáceis de tratar.

Propósito:
Reduzir a complexidade do modelo e melhorar sua capacidade de generalização.

Passo 5: Preenchimento de Valores Ausentes com Zero

Preenche quaisquer valores ausentes nas colunas dos conjuntos de dados com 0.

Motivação:
Evitar erros durante a execução do modelo preditivo, que não aceita valores ausentes.
Valores ausentes podem indicar "ausência de informação", o que, dependendo do contexto, pode ser representado como zero.

Propósito:
Garantir consistência nos dados e evitar problemas técnicos durante a modelagem.

Passo 6: Identificação de Variáveis Categóricas

Identifica colunas categóricas no conjunto de dados (aquelas com tipo object).
Para cada variável categórica:
Conta o número de categorias únicas.
Exibe as categorias no console.

Motivação:
Ajuda a identificar variáveis com alta cardinalidade, que podem ser problemáticas para os modelos.
Fornece uma visão geral da natureza dos dados categóricos.

Propósito:
Informar decisões sobre quais variáveis manter ou descartar com base na cardinalidade.

Passo 7: Descartar Variáveis de Alta Cardinalidade

Remove colunas específicas tanto do conjunto de treino quanto do de teste. Exemplos:
codigo_area_telefone_residencial, codigo_area_telefone_trabalho: Alta cardinalidade (muitos valores únicos).
id_solicitante: Identificador único, irrelevante para o modelo.
estado_onde_trabalha, estado_onde_nasceu: Já substituídos por regiões.

Motivação:
Variáveis com alta cardinalidade podem dificultar o aprendizado do modelo.
Identificadores únicos não fornecem informação preditiva.

Propósito:
Simplificar os dados, removendo variáveis que podem causar overfitting ou não são úteis para a predição.

Passo 8: Substituir Valores Específicos

Substitui valores de espaço em branco (' ') na coluna sexo por 'N'.

Motivação:
Espaços em branco não são interpretados corretamente pelos modelos.
'N' pode representar um valor ausente ou desconhecido de maneira mais explícita.

Propósito:
Garantir consistência nos dados e melhorar a qualidade das informações.

Essas etapas representam um pré-processamento importante que inclui:

Agrupamento: Reduz a cardinalidade substituindo estados por regiões.
Tratamento de valores ausentes: Substituição por valores padrão como 0 ou 'N'.
Identificação e exclusão de variáveis: Remoção de colunas irrelevantes ou problemáticas.
Preparação para modelagem: Assegura que os dados estejam no formato esperado pelos modelos.

Passo 9: Codificação de Variáveis Categóricas com pd.get_dummies

Transforma variáveis categóricas (forma_envio_solicitacao, estado_onde_reside, sexo) em variáveis dummy (representação binária).
Cada categoria única em uma coluna se torna uma nova coluna no DataFrame, com valores 0 ou 1 indicando a presença daquela categoria.

Motivação:
Muitos algoritmos de machine learning, como Regressão Logística e Random Forest, não trabalham diretamente com variáveis categóricas.
A codificação transforma essas variáveis em um formato numérico compreensível.

Propósito:
Garantir que as variáveis categóricas possam ser usadas nos modelos preditivos.

Passo 10: Harmonização entre Conjuntos de Dados

Adiciona colunas ausentes no conjunto de teste para garantir que ele tenha as mesmas colunas que o conjunto de treino.
Reordena as colunas do conjunto de teste para seguir a mesma ordem do conjunto de treino.

Motivação:
Após a criação de variáveis dummy, pode haver diferenças nas colunas entre treino e teste (algumas categorias podem não estar presentes em ambos os conjuntos).
Os modelos exigem que os conjuntos de treino e teste tenham o mesmo número e ordem de colunas.

Propósito:
Evitar erros durante a predição devido a inconsistências nos conjuntos de dados.

Passo 11: Binarização de Variáveis

Converte variáveis categóricas com duas categorias (vinculo_formal_com_empresa, etc.) em valores binários (0 ou 1).
Usa LabelBinarizer para realizar a conversão.

Motivação:
Reduz a complexidade dessas variáveis, já que possuem apenas duas categorias, eliminando a necessidade de criar múltiplas colunas dummy.

Propósito:
Representar variáveis de maneira eficiente para o modelo.

Passo 12: Listagem de Colunas Selecionadas

Define as colunas (atributos) a serem usadas para o modelo.
Define a variável-alvo (inadimplente) como o rótulo para predição.

Motivação:
Nem todas as colunas são relevantes para o modelo. Esta seleção foca nos atributos mais informativos.

Propósito:
Reduzir a dimensionalidade do problema e melhorar a performance do modelo.

Passo 13: Embaralhamento dos Dados

Embaralha os conjuntos de treino e teste, preservando a distribuição original.
Usa random_state para garantir que o embaralhamento seja reprodutível.

Motivação:
Evitar que a ordem dos dados influencie o aprendizado do modelo.

Propósito:
Garantir que o modelo não seja influenciado por padrões na ordem original dos dados.

Passo 14: Separação de Features e Alvo

Divide os conjuntos de dados em:
x: Features (atributos preditivos).
y: Alvo (inadimplente).
Realiza o mesmo para os dados de teste.

Motivação:
Separar os atributos do rótulo facilita o uso das bibliotecas de machine learning.

Propósito:
Preparar os dados para o treinamento e a validação do modelo.

Essas etapas representam os estágios finais de preparação dos dados para os modelos:

Codificação: Transforma variáveis categóricas em dummies ou binárias.
Harmonização: Garante consistência entre treino e teste.
Seleção de atributos: Define as colunas mais relevantes para a modelagem.
Separação e embaralhamento: Organiza os dados para o uso em algoritmos preditivos.

Passo 15: Separação e Normalização dos Dados

Motivação:
Divisão dos Dados: Os dados são divididos em conjuntos de treino e teste para evitar que o modelo aprenda padrões específicos dos dados de teste.
x_treino e x_teste contêm as variáveis preditivas (features) sem a última coluna.
y_treino e y_teste contêm os rótulos (variável-alvo), representando a inadimplência.
Normalização: A normalização através do MinMaxScaler ajusta as features para o intervalo [0,1], tornando os dados comparáveis e melhorando o desempenho dos modelos.

Propósito:
Garantir que o modelo generalize bem para novos dados.
Prevenir problemas causados por escalas diferentes nas variáveis.
Criar uma base consistente para o treinamento e a avaliação do modelo

Passo 16: Avaliação com K-Nearest Neighbors (KNN)

Motivação:
Testar diferentes valores de k (vizinhos) para encontrar o mais adequado para o modelo.
Avaliar a precisão (acurácia) nos conjuntos de treino e teste.

Propósito:
Comparar o desempenho do modelo com diferentes configurações.
Identificar possíveis problemas de overfitting (alta acurácia no treino, mas baixa no teste) ou underfitting (baixa acurácia em ambos).

Passo 17: Regressão Logística com Hiperparâmetro C

Motivação:
Explorar diferentes valores do hiperparâmetro C, que controla o peso da regularização:
Valores altos de C reduzem a regularização, ajustando melhor os dados.
Valores baixos de C aumentam a regularização, prevenindo overfitting.

Propósito:
Identificar o valor ideal de C que balanceia desempenho e regularização.
Garantir a robustez do modelo ao evitar ajuste excessivo.

Passo 18: Treinamento Final

Motivação:
Após testar os hiperparâmetros e algoritmos, utiliza-se o modelo otimizado (C=1) para treinar com todos os dados de treino disponíveis (x_treino_final e y_treino_final).

Propósito:
Construir o modelo final, pronto para aplicação em dados de produção ou análise detalhada.

Passo 19: Definição do Grid de Hiperparâmetros

Motivação:
Configurar os possíveis valores dos hiperparâmetros para o modelo RandomForestClassifier.
Explorar diferentes combinações de parâmetros para otimizar o desempenho.

Propósito:
Garantir que o modelo de floresta aleatória seja ajustado de maneira eficiente aos dados, maximizando a acurácia.

Passo 20: Busca de Hiperparâmetros com HalvingGridSearchCV

Motivação:
Utilizar HalvingGridSearchCV, uma estratégia eficiente para busca de hiperparâmetros, testando combinações em um espaço de busca reduzido.

Propósito:
Identificar os melhores valores de hiperparâmetros para o modelo, reduzindo o custo computacional.

Passo 21: Validação Cruzada K-Fold

Motivação:
Dividir os dados de treino em múltiplos subconjuntos (folds) para avaliar o modelo de forma consistente.

Propósito:
Medir a estabilidade do modelo e a média de acurácia em diferentes partições.

Passo 21: Treinamento e Predição Final com Random Forest

Motivação:
Treinar o modelo RandomForestClassifier com os melhores hiperparâmetros encontrados.
Fazer predições no conjunto de teste final.

Propósito:
Obter as respostas finais para o conjunto de teste.

Passo 22: Exportação do Arquivo de Respostas