Projeto da Disciplina Ciência de Dados

Relatório de Preparação de Dados e Modelagem

Carlos Eduardo Nascimento Cajado

Análise e Predição de Alunos com Risco de Baixo Desempenho Acadêmico

Histórico de Revisões do Documento

Revisão	Descrição	Modificado por	Status	Data
1.0	Preparação de Dados e Modelagem	Carlos Eduardo Cajado	Aprovado	01/12/2024
2.0	Preparação de Dados e Modelagem atualizado	Carlos Eduardo Cajado	Aprovado	11/12/2024

1 Objetivo Técnico

O objetivo técnico deste relatório foi desenvolver um modelo computacional para análise e Predição de Alunos com Risco de Baixo Desempenho Acadêmico. É utilizado duas técnicas de aprendizado de máquina: redes neurais e árvores de decisão, visando identificar o modelo mais eficiente para o problema apresentado.

2 Preparação dos Dados

Com base nos processos e resultados apresentados no Relatório de Coleta, Descrição, Exploração e Avaliação da Qualidade dos Dados, são necessárias manipulações adicionais para o tratamento do arquivo CSV, a fim de aplicá-lo nos modelos computacionais. Esse processo é detalhado nas subseções: Seleção de Dados, Limpeza, Construção de Dados, Integração de Dados e Formatação dos Dados.

2.1 Seleção de Dados

Para facilitar a análise, algumas colunas com informações irrelevantes ou com baixa correlação para a modelagem, além de colunas no formato de string, serão removidas do arquivo. Entre elas destacam-se: 'Matricula', 'situacaoDisciplina', 'idGrade' e 'Sexo'. As demais colunas foram consideradas essenciais para o processo de aprendizado. A Tabela 1 apresenta a lista completa das colunas selecionadas e excluídas, incluindo suas descrições e os motivos para sua inclusão ou exclusão no conjunto de dados de entrada para os modelos.

Nome Coluna	Descrição	Motivo	Status
anoNascimento	Ano de nascimento do aluno.	potencialmente relevante	
idSerie	Identificador série escolar.	potencialmente relevante	
EscolaridadeResponsavel	Nível de escolaridade do responsável pelo aluno.	potencialmente relevante	
idEstadoCivilResponsavel	Identificador do estado civil do responsável pelo aluno.	potencialmente relevante	
Total_Faltas	Total de faltas acumuladas pelo aluno.	potencialmente relevante	
numeroReprovacoes	Porcentagem de reprovações do aluno.	potencialmente relevante	
AprovacoesComExcelencia	Porcentagem aprovações do aluno com desempenho de excelência	Potencialmente relevante	
Aprovacoes	Porcentagem de aprovações do aluno em relação ao número de disciplinas cursadas.	potencialmente relevante	
sexo_int	Representação numérica do sexo do aluno	potencialmente relevante	
situacaoDisciplina_inteiro	Representação em inteiro da situação do aluno na disciplina (valor de saída desejado).	potencialmente relevante	

Diciplina	Nome da disciplina associada ao registro.	Formato String, incompatível com modelos de redes neurais,	
situacaoDisciplina	situação do aluno na disciplina (valor de saída desejado).	Formato String, incompatível com modelos de redes neurais.	
Matricula	Registro do aluno relacionado ao banco de dados	Irrelevante.	
idGrade	Grarde do Aluno no sistema	Irrelevante. Todos possuem a mesma grade.	

Ainda, se faz necessário balancear os dados, pois uma categoria possui o número de amostra muito superior. É fixado o valor máximo de 4000 para cada categoria, tem-se a nova distribuição:

```
Dados de treino balanceados: (16109, 11) (16109,)
Distribuição das classes pós-balanceamento:
situacaoDisciplina_inteiro
8     4000
10    4000
1     3845
7     3148
2     1008
5      108
Name: count, dtype: int64
```

2.2 Limpeza

Para elevar a qualidade dos dados ao nível exigido pelas técnicas de análise selecionadas, é necessário aplicar processos de limpeza, com destaque para a exclusão de linhas que contenham campos vazios. Esse procedimento foi realizado utilizando as funcionalidades da biblioteca **pandas** do Python. Como resultado, 720 registros foram removidos de um total inicial de 52.770.

2.3 Construção de Dados

Na construção de dados, como mencionado anteriormente, as colunas `sexo_id` e `situacao_disciplina_id` foram derivadas de outras colunas, mapeando as opções existentes e gerando um correspondente em formato inteiro.

Ademais, durante a fase de seleção dos dados, foi necessário realizar o processo de integração, no qual foram criadas colunas de percentuais, incluindo a de porcentagem de aprovações com excelência. Esses valores foram calculados com base na seguinte razão:

$$(X / N) \times 100$$

onde X representa o número de aprovações com excelência para cada aluno, e N é o total de matrículas de disciplina.

2.4 Integração de Dados

Não tivemos integração dos dados.

2.5 Formatação dos Dados

Normalmente, os registros de um conjunto de dados são inicialmente organizados de forma ordenada, mas algoritmos de modelagem, como redes neurais, requerem que os dados estejam em uma ordem aleatória para um melhor treinamento. visando atender a essa necessidade, utilizou-se o parâmetro random_state

dos próprios modelos, para garantir a aleatoriedade controlada dos dados ao serem aplicados nos algoritmos de aprendizado de máquina.

3 Modelagem

O processo de modelagem foi estruturado em duas abordagens principais: redes neurais e árvores de decisão, ambas implementadas utilizando a biblioteca scikit-learn do Python. A escolha dessas técnicas foi motivada por suas capacidades complementares em lidar com problemas de classificação e predição. Critérios como precisão, recall, f1-score e acurácia serão utilizados para avaliar e comparar o desempenho de cada abordagem.

3.1 Seleção da Técnica de Modelagem

3.1.1 Modelo Redes neurais (MLP)

Partindo do princípio que as redes MLP são sensíveis à escala dos atributos, utilizaremos a normalização dos dados para o intervalo [-1, +1]. Segue:

Normalização:

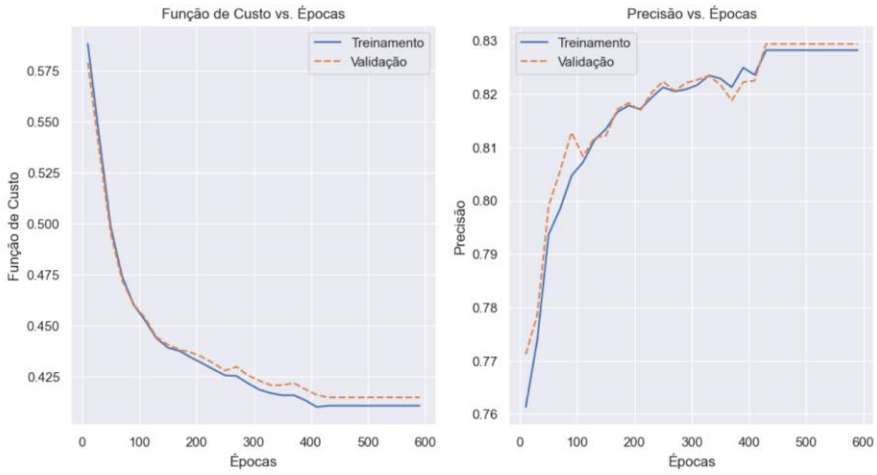
```
def Normalization(x):

    return x/x.abs().max()
```

```
X_treino_Normalizado = Normalization(X_treino)
X_teste_Normalizado = Normalization(X_teste)
```

O modelo de classificação utiliza redes neurais do tipo Multilayer Perceptron (MLP), implementadas com a biblioteca scikit-learn. Foram testados diferentes números de épocas, variando de 10 a 600 em incrementos de 10, para determinar o número ideal de épocas. O parâmetro random_state é fixado em 30 para assegurar a reprodutibilidade dos resultados.

Na figura 1 é apresentados os gráficos que ilustram a relação entre o número de épocas e o erro, bem como a relação entre o número de épocas e a acurácia.

Figura 1: Relação entre o número de épocas e o erro, relação entre o número de épocas e a acurácia.



Autor: próprio

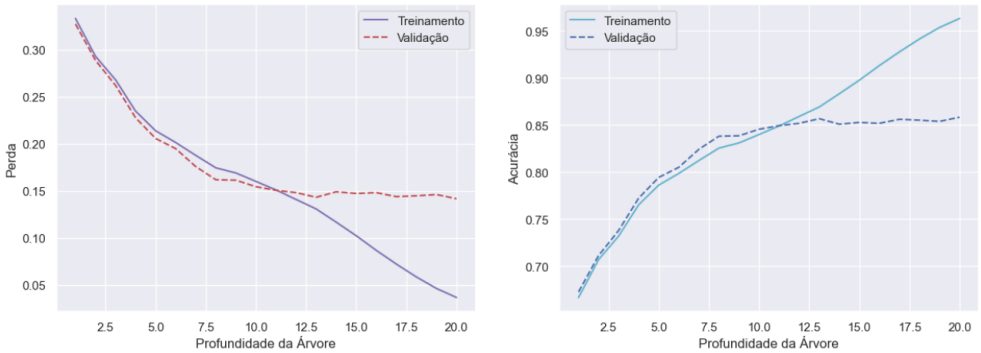
A partir da época 400, a precisão na validação e treinamento se mantém constante Indicando que o modelo está começando a sofrer de overfitting, pois além desse ponto, o aprendizado adicional não está se traduzindo em uma melhor generalização para novos dados.

3.1.2 Modelo em Árvore

Nos experimentos com árvores de decisão, diferentes profundidades foram avaliadas para medir o erro de generalização. Profundidade da árvore foi definida para variar de 1 a 20 níveis. A ideal é definida na etapa de construção e treinamento do modelo, descrita na Seção 3.4. Assim como no modelo MLP, o parâmetro ‘random_state’ é fixado em 30.

A figura 2 apresenta os resultados para escolha da melhor profundidade.

Figura 2: Relação entre a perda e a profundidade, relação entre a acurácia e a profundidade.



Autor: próprio

Analisando os gráficos é notório que para profundidades de árvores superiores a 12, entram em estado de *overfitting* onde, basicamente, não temos mais a influência da profundidade da árvore para o aumento da acurácia.

3.2 Design Experimental

Para o design experimental, inicialmente temos a Divisão dados. O *Database* será dividido, inicialmente, nas partes:

- Treino: 75%
- Teste: 25%

Posteriormente serão separados 15% dos dados de treino para validação, pode-se verificar a distribuição das instâncias nas figuras 3 e 4.

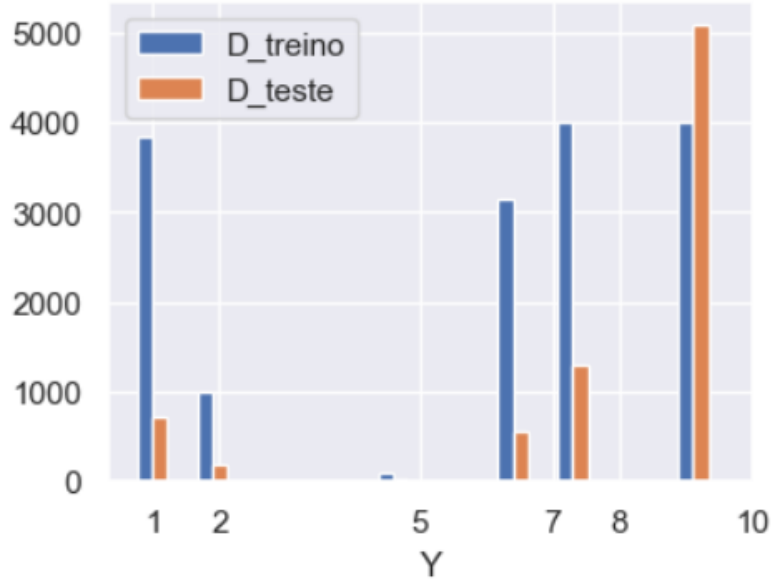
Figura 3: distribuição das instâncias (treino + validação) e teste.

```
X_treino = D_treino.iloc[:, 0:11]
Y_treino = D_treino.iloc[:, -1]
print("D_treino e validação: ", (X_treino.shape, Y_treino.shape))

X_teste = D_teste.iloc[:, 0:11]
Y_teste = D_teste.iloc[:, -1]
print("D_teste: ", (X_teste.shape, Y_teste.shape))
```

```
D_treino e validação:  ((44548, 11), (44548,))
D_teste:  ((7862, 11), (7862,))
```

Autor: próprio

Figura 4: Plot (treino + validação) e teste



Autor: próprio

3.3 Construção/Treinamento de Modelos

- Modelo Redes neurais (MLP)

  Analisando os resultados obtidos com os teste da dessão 3.1.1é utilizado 400 épocas como o modelo ideal. Tem-se o processo de construção, figura 5 e treinamento do modelo figura 6:

Figura 5: processo de construção ideal do modelo.

```
#Criando o modelo ideal com 400 epocas
from sklearn import neural_network
modelo = neural_network.MLPClassifier(random_state=30, max_iter=400)
```

Figura 6: processo de treinamento do modelo.

```
# Dividindo em  dados de treino e validação 15%
X_train, X_val, y_train, y_val = train_test_split(X_treino_Normalizado, Y_treino, test_size=0.15, random_state=30)
```

Autor: próprio

- Modelo usando Árvore

Para a árvore de decisão, foi utilizada uma profundidade de 12,efeito de poda, conforme ilustrado na Figura 7. A representação do modelo para essa configuração está apresentada na Figura 8.

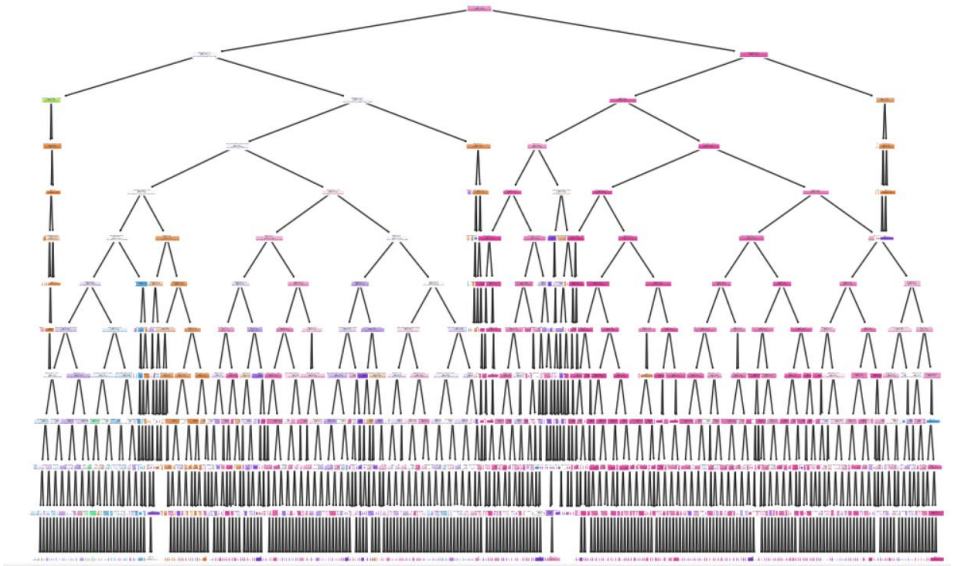
Figura 7: processo construção e treinamento do modelo de árvore.

```
model = tree.DecisionTreeClassifier(max_depth=12, random_state=30)
model.fit(X_train, y_train)

# Plotar a árvore IDEAL
plt.figure(figsize=(12, 8))
tree.plot_tree(model, filled=True, feature_names=X_train.columns)
plt.show()
```

```
# Predições no conjunto de treino e validação
y_pred_train = M.predict(X_train)
y_pred_val = M.predict(X_val)
```

Autor: próprio

Figura 8: Representação do de árvore modelo criado.



Autor: próprio

3.4 Avaliação do(s) Modelo(s)

- **Modelo Redes neurais (MLP)**

  No processo de avaliação tem-se os resultados, figura 9 :

Figura 9: Avaliação MLP

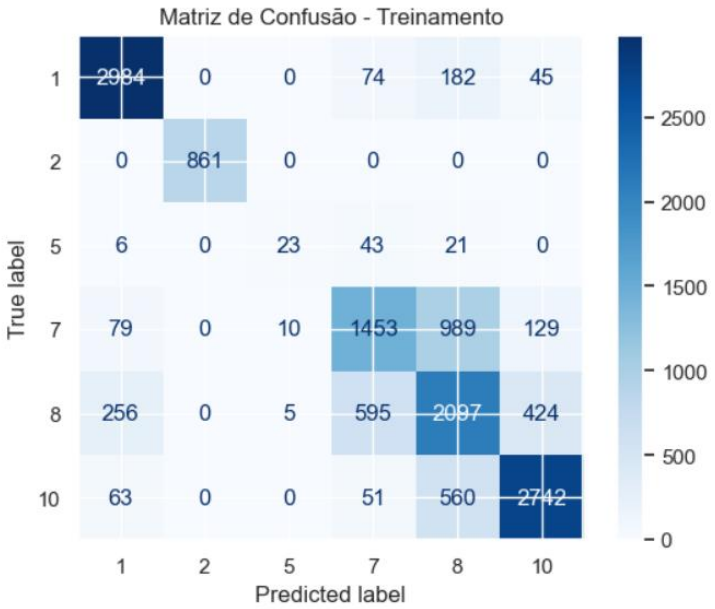
```
#Avaliando o modelo:
E_trn, E_tst = validacao(modelo, (X_train, y_train), (X_teste_Normalizado, Y_teste), zero_one_loss)
print("Erro de treinamento:", E_trn)
print("Erro de teste :", E_tst)
```

```
Erro de treinamento: 0.25796085305287764
Erro de teste : 0.24713813279063856
```

Autor: próprio

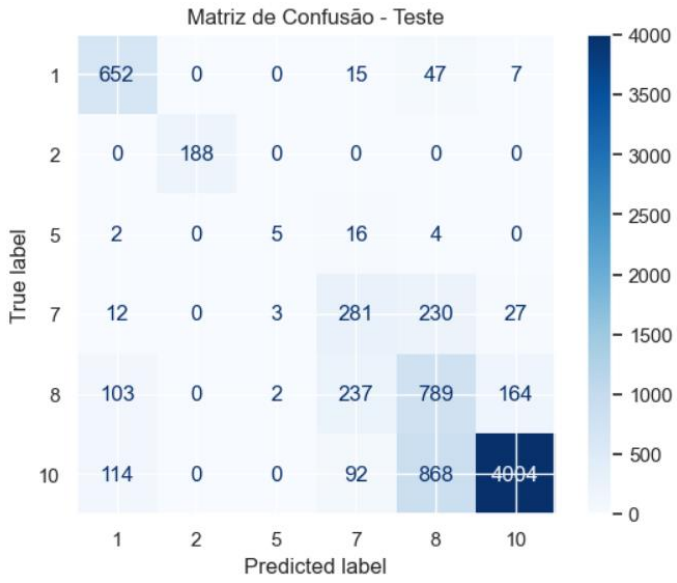
Ou seja, foi identificado um erro de 25,79% durante o treinamento e 24,71% ao utilizar os dados de teste, o que representa bons resultados, considerando os critérios de sucesso técnico previamente estabelecidos. Para uma análise mais detalhada, a Figura 10 apresenta a matriz de confusão do treinamento, a Figura 11 mostra a matriz de confusão do teste.

Figura 10: plote Matriz de confusão Treino MLP



Autor: próprio

Figura 11: plote Matriz de confusão Teste MLP



Autor: próprio

A Figura 12 exibe o relatório de avaliação do modelo de redes neurais, apresentando os resultados para cada classe do problema.

Figura 12: Relatório de avaliação MLP.

treino:

	precision	recall	f1-score	support
1	0.88	0.91	0.89	3285
2	1.00	1.00	1.00	861
5	0.61	0.25	0.35	93
7	0.66	0.55	0.60	2660
8	0.54	0.62	0.58	3377
10	0.82	0.80	0.81	3416
accuracy			0.74	13692
macro avg	0.75	0.69	0.71	13692
weighted avg	0.74	0.74	0.74	13692

teste :

	precision	recall	f1-score	support
1	0.74	0.90	0.81	721
2	1.00	1.00	1.00	188
5	0.50	0.19	0.27	27
7	0.44	0.51	0.47	553
8	0.41	0.61	0.49	1295
10	0.95	0.79	0.86	5078
accuracy			0.75	7862
macro avg	0.67	0.67	0.65	7862
weighted avg	0.81	0.75	0.77	7862

Autor: próprio

É notório que os maiores problemas na avaliação estão relacionados à precisão da categoria 5 (APROVADO PELO CONSELHO), para os dados de teste, apresentou uma precisão de 50%, por de ter a menor quantidade de registros, conforme indicado pela matriz de confusão. Por outro lado, a categoria 8 (BOM/APROVADO) enfrentou dificuldades devido à proximidade das categorias 7 (APROVADA) e 10 (MUITO BOM APROVADA), resultando em uma precisão de 41%. As categorias 1 (REPROVADO), 2 e 10, por sua vez, apresentaram as melhores precisões, com 74%, 100% e 95%, respectivamente.

- **Modelo usando Árvore**

  No processo de avaliação tem-se os resultados, figura 13 :

Figura 12: Relatório de avaliação

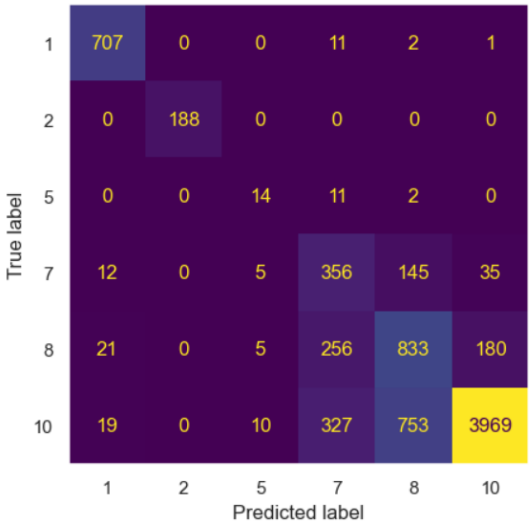
```
# Calcular o erro
e = zero_one_loss(Y_teste, Y_pred)
print("Erro no conjunto de teste: {:.2f}%".format(e * 100))
```

```
Erro no conjunto de teste: 22.83%
```

Autor: próprio

Com um erro nos testes de 22,83%, a Figura 14 apresenta a matriz de confusão.

Figura 14: matrix de confusão (Árvore de decisão com profundidade 12).



Autor: próprio

Apresentamos o relatório de avaliação correspondente há árvore de decisão, figura 15, temos:

Figura 15: Relatório de avaliação (Árvore de decisão com profundidade 12).

	precision	recall	f1-score	support
1	0.93	0.98	0.96	721
2	1.00	1.00	1.00	188
5	0.41	0.52	0.46	27
7	0.37	0.64	0.47	553
8	0.48	0.64	0.55	1295
10	0.95	0.78	0.86	5078
accuracy			0.77	7862
macro avg	0.69	0.76	0.72	7862
weighted avg	0.83	0.77	0.79	7862

Autor: próprio

Algumas considerações podem ser feitas com base nos resultados: a categoria 5 apresentou uma precisão de 41%, conforme indicado pela matriz de confusão; a categoria 8 teve uma precisão de 48%, enquanto a categoria 7 resultou em uma precisão de 37%. As categorias 1 (REPROVADO), 2 e 10, por sua vez, destacaram-se com as melhores precisões, alcançando 93%, 100% e 95%, respectivamente.

Com base nos resultados obtidos, o modelo de árvore de decisão, nas configurações analisadas, apresentou desempenho ligeiramente superior ao da MLP. Portanto, o modelo de árvore de decisão é recomendado como a melhor abordagem para resolver a problemática.