SMARKIO

2021

# Relatório Final

Douglas Oliveira

# Análise Exploratória

Foram um total de 643 amostras, divididas entre 600 amostras aprovadas e 43 amostras para serem revisionadas. A tabela de dados possui 4 colunas, são elas:

- Pred_class
- probabilidade
- status
- True_class

Na coluna Pred_Class, que é a coluna de inteiros das classes preditas pelo modelo, foram identificadas 80 classes diferentes.

Na coluna True_class, que caracteriza a categorização correta da amostra, possui em grande maioria valores nulos que foram tratados assumindo o valor da classe em Pred_class.

A coluna probabilidade aparenta ser a coluna de confiabilidade do modelo sobre sua predição, foi assim que supus durante a análise. Já a coluna status separa os dados aprovados pelos dados que necessitam de revisão.

## RESUMO

- 643 linhas de dados
- 4 colunas
  - Pred_class
  - probabilidade
  - status
  - True_class
- 600 amostras aprovadas
  - representam 93.31%
- 43 amostras para revisão
  - representam 6.69%
- Possui 462 (71.85%) valores nulos em True_class
- 80 classes identificadas

| | Pred_class | probabilidade | status | True_class |
|---|---|---|---|---|
| 0 | 2 | 0.079892 | approved | 0.0 |
| 1 | 2 | 0.379377 | approved | 74.0 |
| 2 | 2 | 0.379377 | approved | 74.0 |
| 3 | 2 | 0.420930 | approved | 74.0 |
| 4 | 2 | 0.607437 | approved | NaN |
| 5 | 2 | 0.690894 | approved | NaN |
| 6 | 2 | 0.759493 | approved | NaN |
| 7 | 2 | 0.834910 | approved | NaN |
| 8 | 2 | 0.861396 | approved | NaN |
| 9 | 2 | 1.000000 | approved | NaN |

Algumas informações estatísticas das colunas:

- Pred_class:
  - int
  - variáveis qualitativas nominais
- probabilidade:
  - floats
  - coluna de variáveis quantitativas contínuas
- status:
  - strings
  - variáveis qualitativas nominais
  - coluna binária
- True_class:
  - floats
  - variáveis qualitativas nominais
  - possui muitos valores nulos

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **Pred_class** | 643.0 | 52.712286 | 37.602068 | 2.000000 | 12.000000 | 59.000000 | 81.000000 | 118.0 |
| **probabilidade** | 643.0 | 0.622436 | 0.266811 | 0.043858 | 0.408017 | 0.616809 | 0.870083 | 1.0 |
| **True_class** | 643.0 | 48.251944 | 38.542269 | 0.000000 | 3.000000 | 55.000000 | 77.000000 | 118.0 |



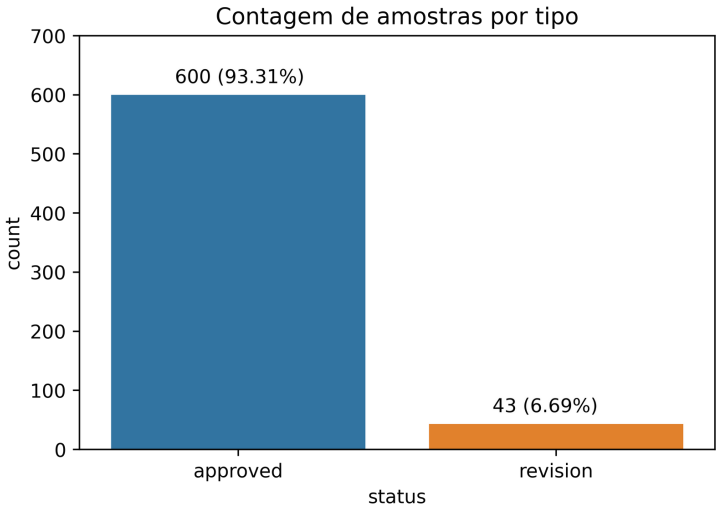
Há algumas informações novas que podemos tomar através desta tabela como a grande quantidade de valores nulos na coluna True_class, valores estes que logo serão filtrados.

Uma informação importante é definirmos os quartis, os quartis dividem nossos dados em quatro partes iguais que são usados para entendermos melhor a distribuição dos nossos dados, o valor de Q1 indica que 25% dos dados estão abaixo de Q1 e 75% deles acima de Q1, o Q2 indica que 50% dos dados estão abaixo e 50% acima, intuitivamente sabemos sobre Q3 (75%). É interessante esclarecer que Q2 trata-se justamente da mediana dos valores.

Na coluna Pred_class, o Q1 tem valor 12.0, o Q2 tem 59 e Q3 tem valor 81 e que o maior valor é 118. Já na coluna das classes verdadeiras, depois de filtrado os valores nulos, temos uma leve queda nos valores dos quartis. Para probabilidade é feito a mesma análise. Podemos ver a média e o desvio padrão de todas as colunas, mas vale ressaltar que, neste caso, não faa muito sentido calcularmos a média e o desvio padrão nas colunas categóricas, mas elas estão ali expostas.

Abaixo seguiremos plotando alguns gráficos para visulizarmos nossos dados seguido de uma breve explicação. **Caso deseje ter acesso aos códigos que geraram estes gráficos, todos eles estão no Jupyter Notebook enviado junto com este documento.**

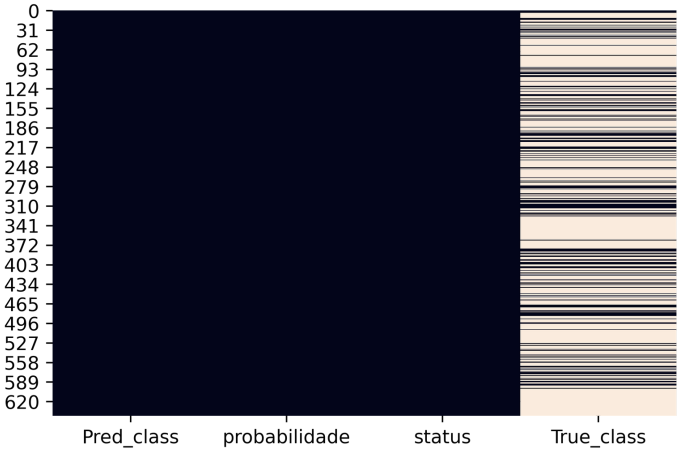
Iniciaremos a análise exploratória criando um gráfico de contagem comparativo entre as amostras separando elas pelo status.



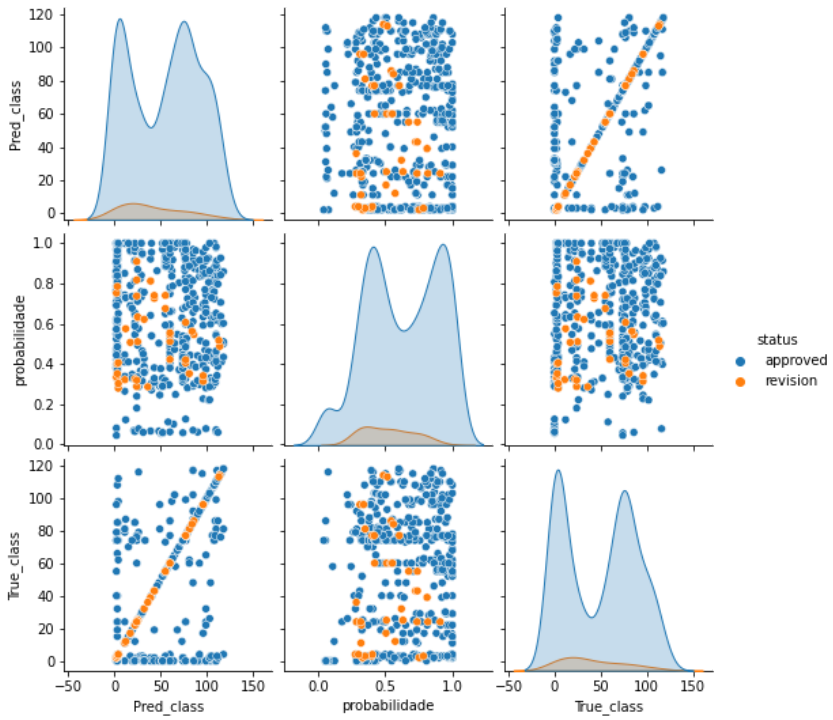

Podemos ver que o dataset tem uma grande quantidade de predições que foram aprovadas (93.31%), contrário ao número de amostras que necessitam de revisão que representa apenas 6.69% do nosso conjunto de dados.

---

Possuímos 462 valores nulos na coluna True_class, ou seja, devemos filtrar estes 462 valores nulos com os respectivos valores da coluna Pred_class, como fora especificado. Para melhor visualizarmos o problema, plotei uma matriz onde os valores nulos estão em branco, este HeatMap nos ajuda a perceber que os valores nulos se distribuiem de forma uniforme pela coluna.



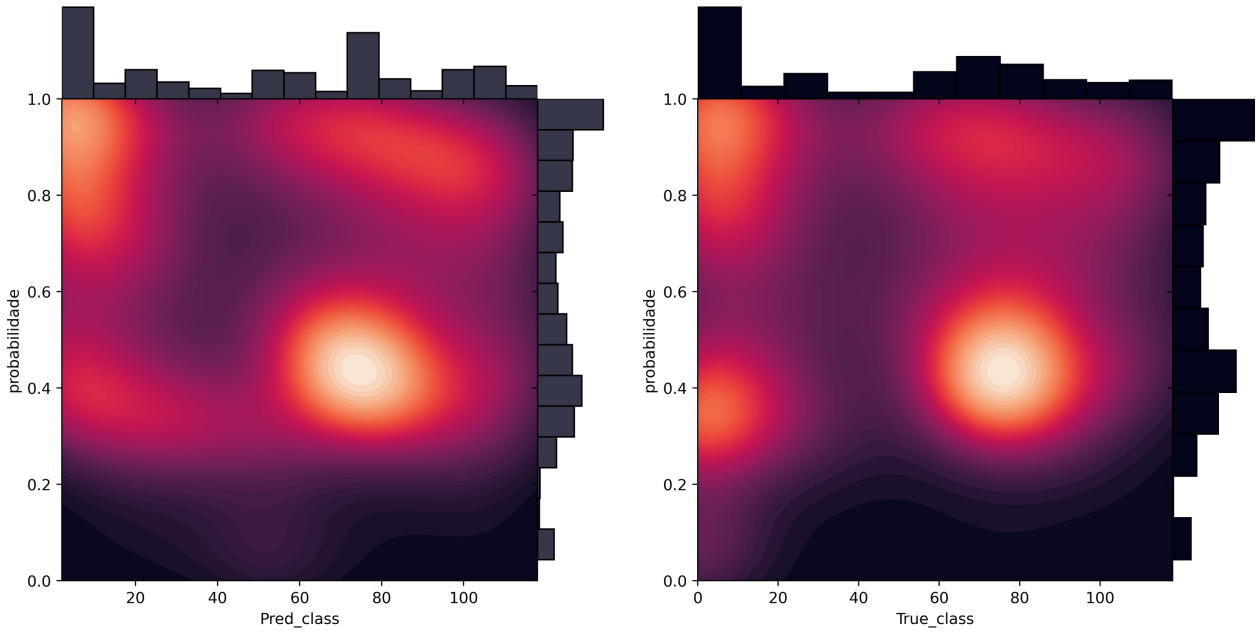
Abaixo plotei uma matriz de gráficos relacionando cada coluna com as outras, a ideia é entender melhor como é o comportamente dos dados quando uma variável é comparada a outra. Separei utilizando a coluna status.




Conseguimos tirar algumas informações que possam nos ajudar a encontrar insights ou nos indicar algum sinal deles, veja algumas abaixo:

- As probabilidades das classes preditas tendem a serem maiores que 30%.
- As amostras que precisam de revisão aparentemente se limitam acima de 30% apenas nas classes preditas, mas isso pode ser explicado pela alta densidade de pontos nessa região, a quantidade de amostras em revisão é diretamente proporcional a quantidade de pontos. Precisaríamos de mais dados para afirmarmos melhor.
- A densidade das colunas Pred_class, probabilidade e True_class têm aspectos muito semelhantes, podemos ver isto nos gráficos na diagonal principal. Talvez exista alguma relação entre a probabilidade e o valor das classes.
- As probabilidades tendem a serem maiores para as classes maiores, vide a densidade de pontos de aprovados no primeiro quadrante do gráfico Pred_class x probabilidade, o mesmo parece ocorrer no gráfico True_class pela probabilidade. Isto pode não ser uma regra geral, são necessários mais dados para termos melhor clareza.
- Os pontos concentrados na diagonal secundária no gráfico da Pred_class pela True_class é devido a estratégia de filtragem adotada. Podemos ver que nem sempre valores de Pred_class e True_class diferentes indicam que a instância precisa de revisão.

Vamos tentar entender melhor a relação entre a Pred_class com a probabilidade. Para realizar tal tarefa, vamos plotar um gráfico chamado JointGrid, ele nos mostra a relação entre duas variáveis em um espaço bidimensional levando em conta a intensidade do cruzamento entre as duas variáveis.

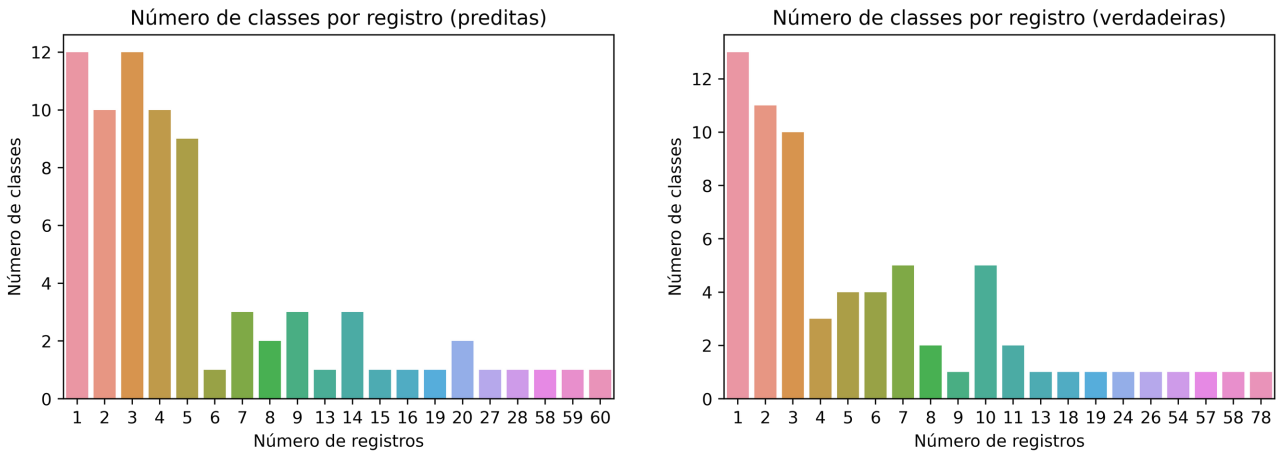



O histograma na parte superior de cada gráfico é a distribuição da variável do eixo x, o histograma na lateral direita de cada gráfico é a distribuição da variável representada no eixo y. Olhando os gráficos acima podemos perceber como se distribui a relação da classes, preditas e verdadeiras, com a probabilidade, veja que as amostras que pertencem à classes entre 60 e 90 tendem a terem probabilidades entre 30% e 60%, já as classes com valores mais baixos tendem a ter uma probabilidade próxima de 100%, mas com uma intensidade menor comparada à intensidade expressa anteriormente.

Esta mesma distribuição espacial da frequência ocorre nas classes corretas, mas com uma intensidade ainda mais forte. Isto nos leva a crer que as classes entre 60 e 90 são as mais díficeis de predizer por algum motivo ou que é onde o classificador possui maior dificuldade em confiar no resultado que forneceu, este último pode ter diversas explicações, uma delas é que talvez o modelo fora treinado com poucos dados rotulados, distribuição mais uniforme das classes no banco de treinamento deste modelo pudesse resolver o problema ou aliviar a intensidade.

---

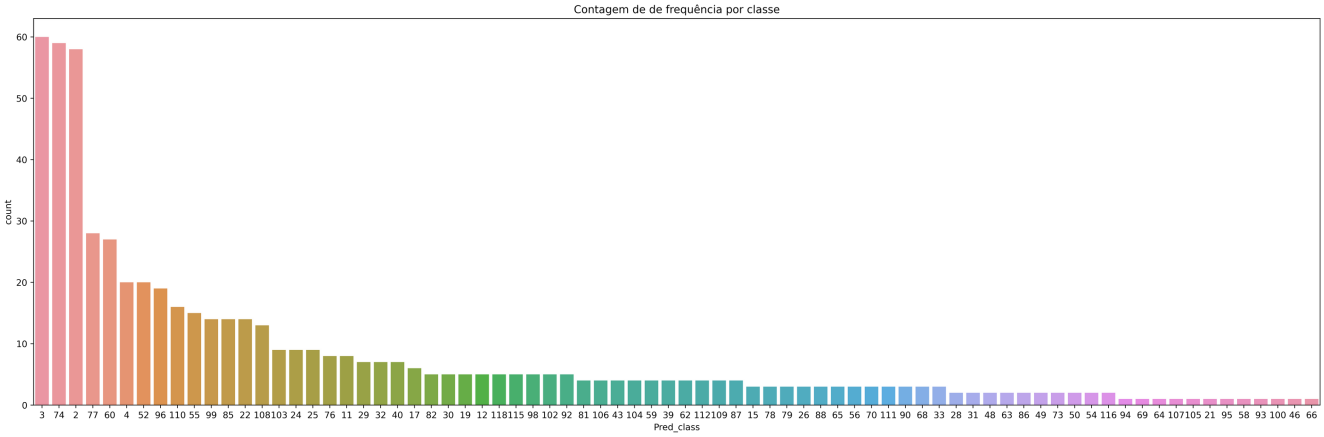
Agora iremos checar a distribuição do número de classes baseadas na quantidade de amostras registradas.




Acima temos um gráfico que relaciona o número de registros por número de classes. Por exemplo, no gráfico à esquerda, temos que 12 classes possui apenas 1 instância classificada, 10 classes possuem 2 instâncias, 12 classes possuem 3 instâncias, assim sucessivamente. Isto esclarecido, podemos afirmar que são poucas as classes que possuem muitas instâncias pertencentes a elas.

---

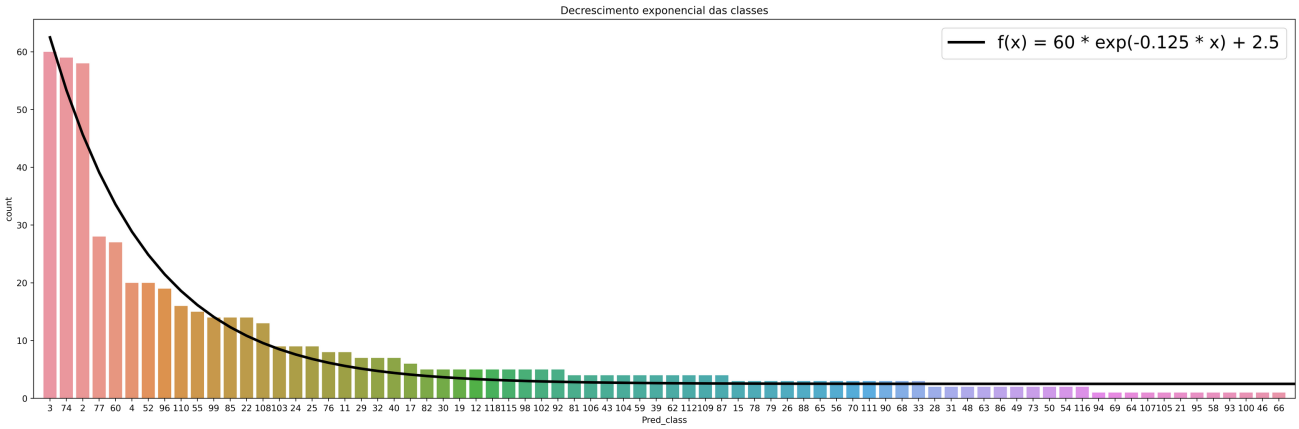
Percebemos pelo gráfico de distribuição abaixo que neste dataset possuímos uma maior frequência das classes 3, 74 e 2, 60 59 e 58 instâncias, respectivamente, e um pouco menos frequente se encontram as classes 77, 60, 4, 52, 96 e 110, com 28, 27, 20, 20, 19 e 16 instâncias catalogadas. A distribuição das classes continua decrescente até se limitar ao mínimo de uma ocorrência em determinadas classes.




Podemos perceber também que a distribuição das classes descresce exponencialmente, para obtermos a função exponencial utilizei apenas ajuste de parâmetros de modo a encaixar a função ao gráfico. Depois de algumas sessões de ajustes manuais, constatei que a função é

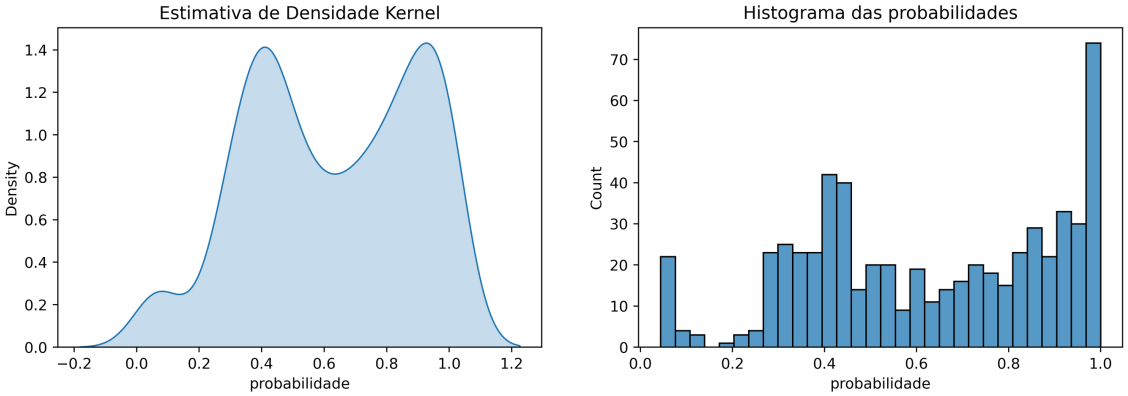
$$f(x) = 60 * \exp(-0.125 * x) + 2.5, \text{ onde } x \in \mathbb{Z}$$

Onde x = 1 representa a classe com maior frequência, x = 2 a segunda, etc.




Os gráficos abaixo expressam como estão distribuídas as probabilidades, veja que a maioria das probabilidades se concentram em valores próximos a 40/45% e acima de 94%, mas com uma leve queda por volta dos 60%. Isto nos indica que o modelo aplicado para predição das classes tende a acertar muito bem, mas que em alguns casos ele se encontra razoavelmente confuso quando a confiabilide de sua predição, talvez por conta da dificuldade para predizer esta classe.

Talvez o gráfico de densidade gere um pouco mais de confusão pois a interpretação do eixo y não é óbvia, este plot é, sumariamente falando, um gráfico cuja a área rachurada tem soma igual a 1, desta forma, supondo que eu queira saber a porcentagem das probabilidades que vão de 20% a 40%, basta calcular a área entre 0.2 e 0.4 no eixo x delimitada pela função de densidade, efetuando o cálculo da área usando integral definida, chegamos ao valor de aproximadamente 0.165, ou seja, 16,5% das probabilidades estão entre 20% e 40%.




Finalizamos aqui nossa análise exploratória, partiremos agora para o relatório das avaliações/métricas que foram selecionadas, seguido da implementação dos classificadores.

# Métricas

A primeira métrica trata-se da **matriz de confusão**, esta métrica gera uma matriz cuja qual relaciona os acertos e erros do modelo. Decidi colocá-la em primeiro pois esta métrica é a origem de outras métricas utilizadas em problemas de classificação, tais como as duas seguintes Recall e Acurácia.

| | Positive | Negative | |
|---|---|---|---|
| **Positive** | True Positive (TP) | False Negative (FN) **Type II Error** | **Sensitivity** $\frac{TP}{(TP+FN)}$ |
| **Negative** | False Positive (FP) **Type I Error** | True Negative (TN) | **Specificity** $\frac{TN}{(TN+FP)}$ |
| | **Precision** $\frac{TP}{(TP+FP)}$ | **Negative Predictive Value** $\frac{TN}{(TN+FN)}$ | **Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$ |

Suponha que tenhamos uma classificação binária, a classe positiva são as amostras catalogadas como 1 e na classe negativa as amostras categorizadas como 0. Em VP teríamos as amostras que foram classificadas como 1 e que de fato eram 1, analogamente com VN, predições 0 que de fato eram 0. Por outro lado, nos quadrantes falsos que é onde ocorrem os erros, temos FN que representa os registros classificados como 0 mas que eram 1 e em FP as instâncias que foram classificadas como 1 mas que, na verdade, pertenciam à classe 0.

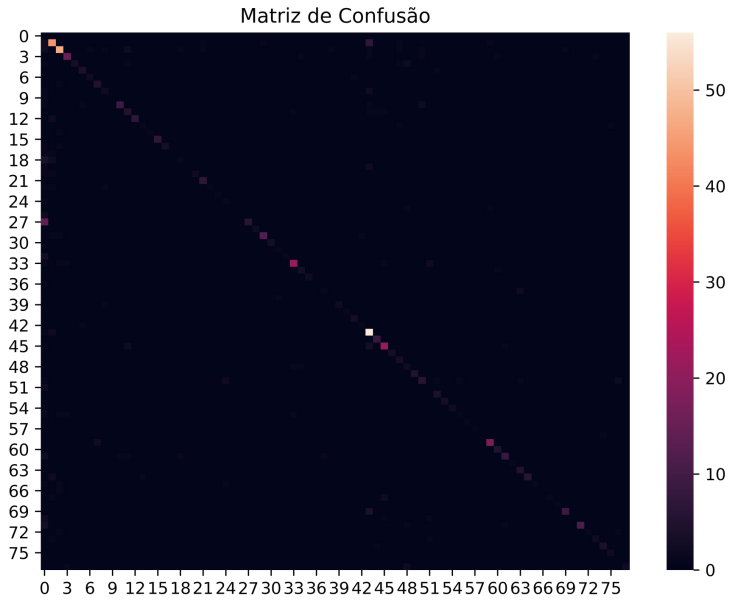
## METRICAS ESCOLHIDAS

- Matriz de Confusão
- Pontuação de Acurácia
- Recall Score

## RESUMO DAS PONTUAÇÕES

- Pontuação de Acurácia
  - 0.6983%
- Recall Score
  - 0.6983%

Mas temos um problema, esta matriz pode ficar bem maior para problemas de classificação multiclasse. Para melhor visualização, observe o HeatMap abaixo.




Os pontos mais claros representam os valores maiores. A diagonal principal da matriz representa os acertos, ou seja, quanto mais claro e denso esta diagonal, melhor é o modelo.

A segunda é a métrica de **acurácia**, para calcular fazemos utilizamos os valores da matriz de confusão:

$$\frac{TP + TN}{(TP + TN + FP + FN)}$$

Onde,

- TP = Verdadeiro Positivo
- TN = Verdadeiro Negativo
- FP = Falso Positivo
- FN = Falso Negativo

A terceira métrica é chamada **Recall Score**, esta métrica é calculada como o número de amostras da classe X que foram classificadas corretamente dividido por todas as amostras que foram classificadas como X, sejam elas acertadas ou não. Em outras palavras, é a capacidade que o classificador tem de identificar todas as amostras corretas (sensibilidade).

$$\frac{TP}{(TP + FN)}$$

Onde,

- TP = Verdadeiro Positivo
- FN = Falso Negativo

**O desempenho destas métricas encontram-se no resumo das pontuações no início desta seção, no canto direito.**

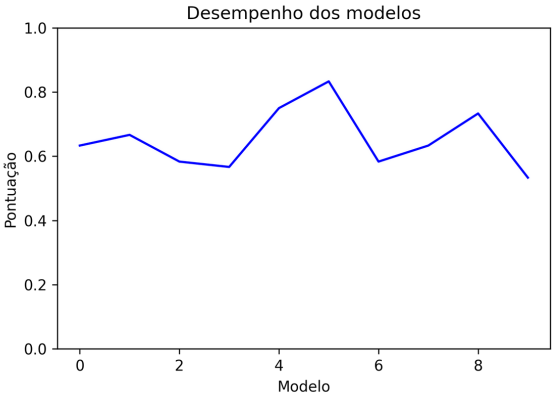
# Classificador

Decidi pelo algoritmo Decision Tree, uma vez que os dados não são complexos e que, aparentemente, obedecem a algumas regras específicas. O classificador de árvore de decisão é "simples" e excelente para encontrar estes padrões que regem o dataset.

Para o método de Cross-Validation resolvi seguir a sugestão, utilizei o KFold já que é um método que tenho costume de aplicar em meus projetos pessoais e que confio bastante.

Treinei o meu modelo 10 vezes e tive uma acurácia média de 65.17% nos bancos de teste.

Utilizando este modelo para corrigir os dados com status de revisão, tivemos 35 de acertos e 8 erros, dando 81.40% de êxito.




Por algum motivo o modelo 5 que treinei foi o melhor, isso pode ser explicado devido a qualidade dos dados de treino na sessão 5 ou pela facilidade de predição dos dados de teste da mesma sessão de treino.

## CLASSIFICADOR

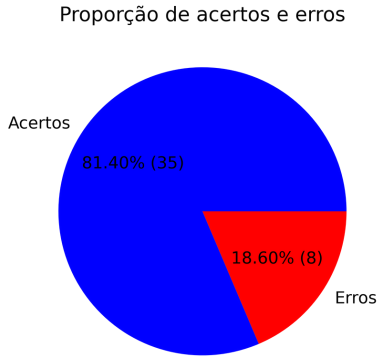
- Árvore de Decisão
  - DecisionTreeClassifier

## MEU MODELO

- Acurácia: 65.17% nos bancos de teste

## MODELO PRINCIPAL

- Acertos: 35
- Erros: 8
- Acurácia: 81.40% na correção final



Para o problema de Processamento de Linguagem Natural proposto, utilizei o MultinomialNB() da bilbioteca Scikit-Learn. Para o pré-processamento dos textos utilizei o CountVectorizer(), também da mesma bilbioteca.

Não é possível trabalhar com textos quando estamos lidando com algoritmos de Machine Learning, então precisamos buscar alguma representação matemática para os textos e uma maneira de fazer isso é descrita na implementação do método CountVectorizer(). Este método associará a cada palavra um valor inteiro, este procedimento é chamado de Bag-of-Words, ou BoW, em seguida ele contabiliza a frequência de cada uma destas palavras presentes no texto gerando como resultado uma matriz sparse com os valores de frequência de cada palavra e com dimensão de número total de palavras encontradas no dataset.

## CLASSIFICADOR NAIVE-BAYES

- MultinomialNB

## PONTUAÇÃO

- Acertos: 42
- Erros: 10
- Acurácia: 80.77%