# IA006

# Lista de Exercícios 2

Alunos:

Felicio Harley Garcia de Castro – RA: 180540

Roger Danilo Figlie – RA: 189957

# Sumário

# Exercício 1

## Item a

Nesta parte do exercício fizemos histogramas para todos os dados considerando cada atributo. O primeiro conjunto destes histogramas, exibido na Figura 1, mostra os dados sem a separação entre as classes.

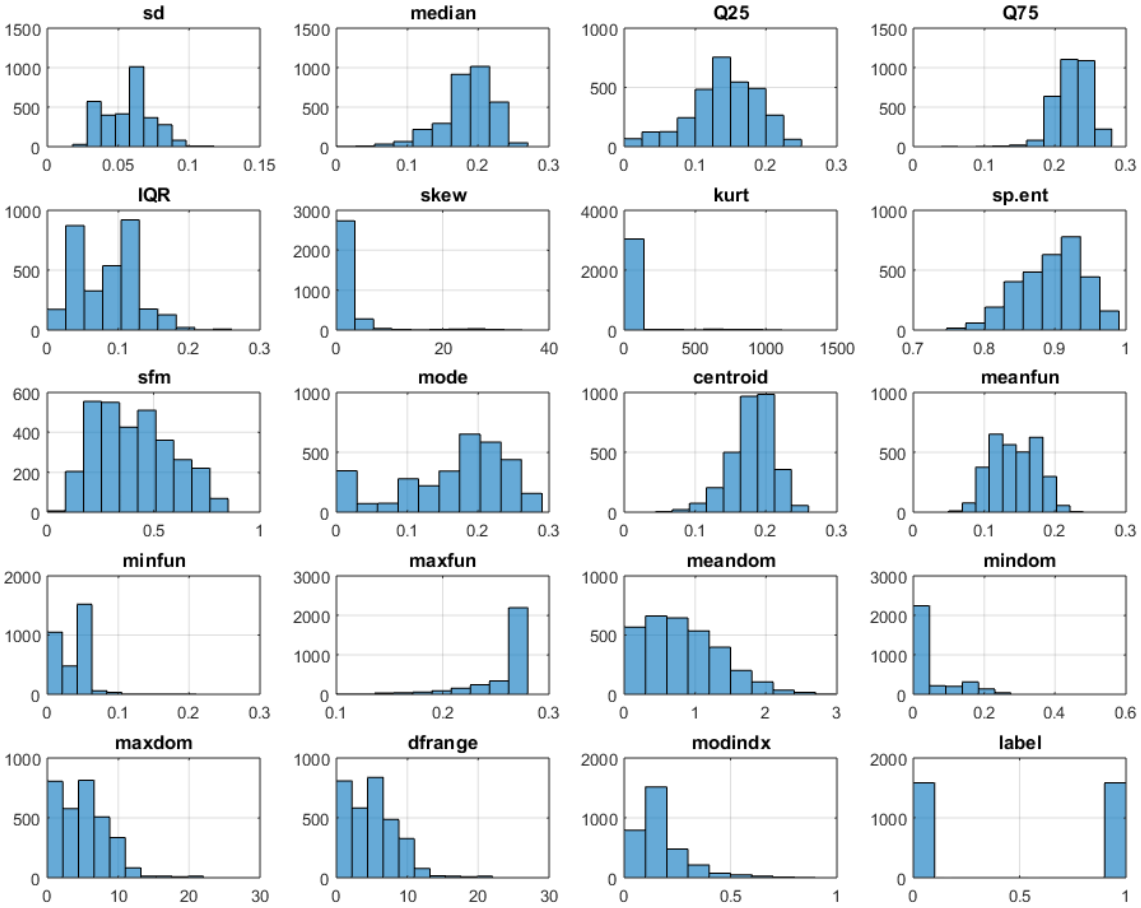


*Figura 1: Histogramas dos atributos sem separação de classes.*

Por este primeiro histograma podemos notar que as variáveis como a ‘Q25’, ‘sp.ent’, etc. seguem uma distribuição aparentemente Gaussiana. Notamos também que os atributos como ‘skew’ e ‘kurt’ possuem valores com baixa frequência e bem distante dos demais valores com alta frequência, o que pode caracterizar a presença de outliers.

Devido à presença destes outliers é indicado a utilização da estandardização (subtração pela média e divisão pelo desvio padrão) ao invés da normalização dos dados. Foi realizada esta ação também para prevenir *overflow* da exponencial na função logística, melhorando a estabilidade numérica do algoritmo. Outra observação importante é que as duas classes estão igualmente distribuídas dentro dos dados. Na Figura 2 temos os histogramas mostrando as distribuições das variáveis separadas por classes,

sendo a classe 1 (homens) representada pela cor azul e a classe 0 (mulheres) pela cor vermelha.

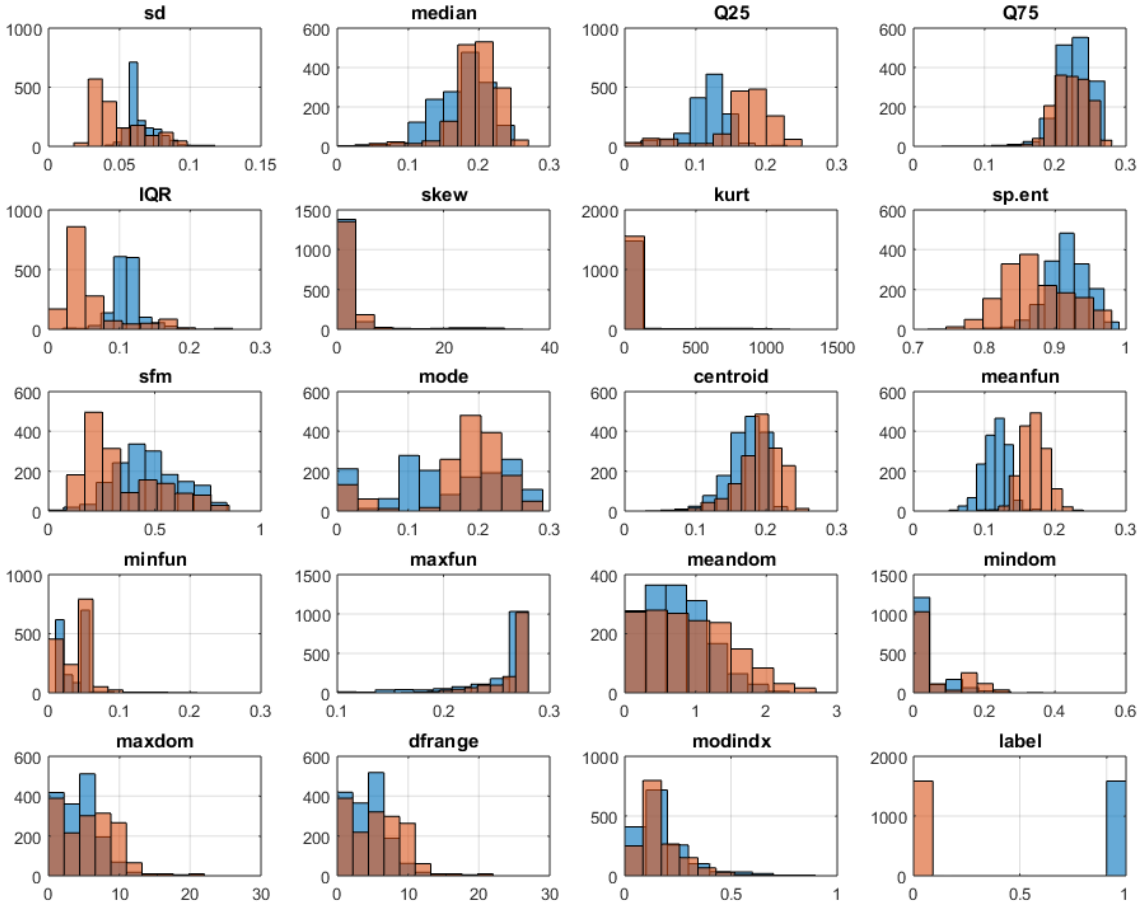


*Figura 2: Histogramas dos atributos com separação de classes.*

Por este histograma vemos que algumas variáveis possuem uma separação natural entre as classes. O atributo ‘meanfun’ por exemplo é a que melhor indica esta separação natural dos dados. Caso tal separação se mostrasse ao avaliarmos o conjunto de treino, poderíamos supor que se trata de uma variável importante para a classificação.

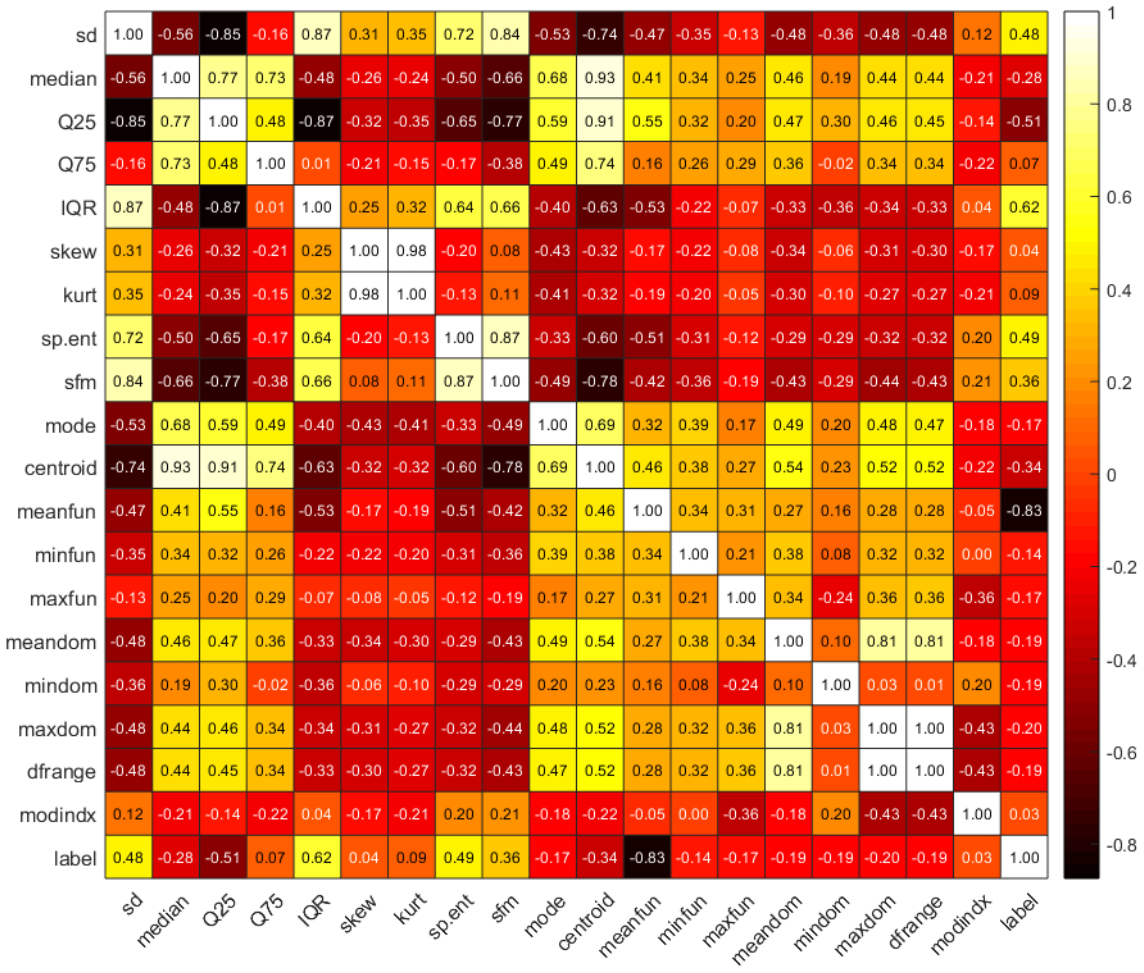


*Figura 3: Mapa de calor da correlação entre os atributos.*

Já através do gráfico de correlação, apresentado na Figura 3, podemos notar duas coisas interessantes. A primeira delas é que algumas das variáveis possuem alta correlação entre si (valores próximos de 1 ou -1), o que indica variáveis possivelmente redundantes como os pares (skew, kurt) e (maxdom, dfrange). A segunda informação interessante é a correlação entre as variáveis e a classe. Notamos que variáveis como a 'meanfun' e a 'IQR' que possuíam uma separação mais distinta entre as classes no histograma possuem também uma forte correlação com a label.

## Item b

Para iniciar esta parte do exercício os dados foram estandardizados e em seguida foi adicionada uma coluna com valores iguais a "1", para gerar o termo independente da regressão. Os dados foram então separados em treino e teste com 80% dos dados para treino.

Prosseguiu-se a realização de uma regressão logística tendo como objetivo a minimização da entropia cruzada e esta minimização foi realizada através do método do gradiente descendente. Utilizamos como critério de parada uma variação no $\nabla J$ menor que $10^{-5} * (\nabla J_{SegundaIteração} -$

$\nabla J_{PrimeiraIteração}$), por 50 iterações consecutivas e, como contingência, um limite de 20000 iterações. Empregamos um passo $\alpha = 1$, tal passo foi selecionado através de tentativa e erro. O vetor $w$ foi inicializado com valores aleatórios segundo uma distribuição uniforme com valores entre -0.5 e 0.5. Através dessa minimização obtiveram-se os seguintes gráficos para a entropia cruzada e a norma do gradiente:

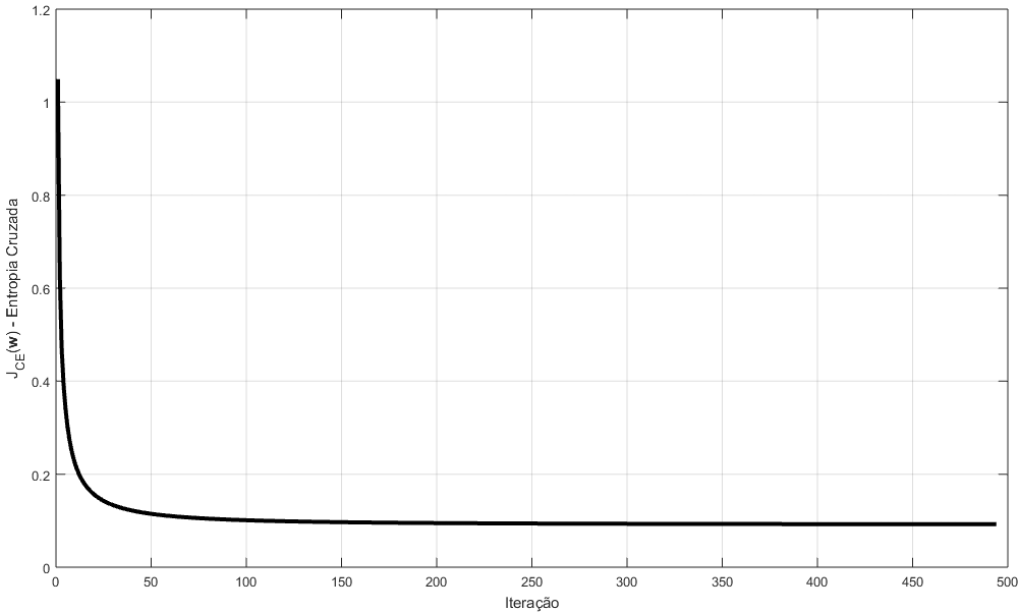


*Figura 4: Variação da entropia cruzada em função das iterações com α = 1.*

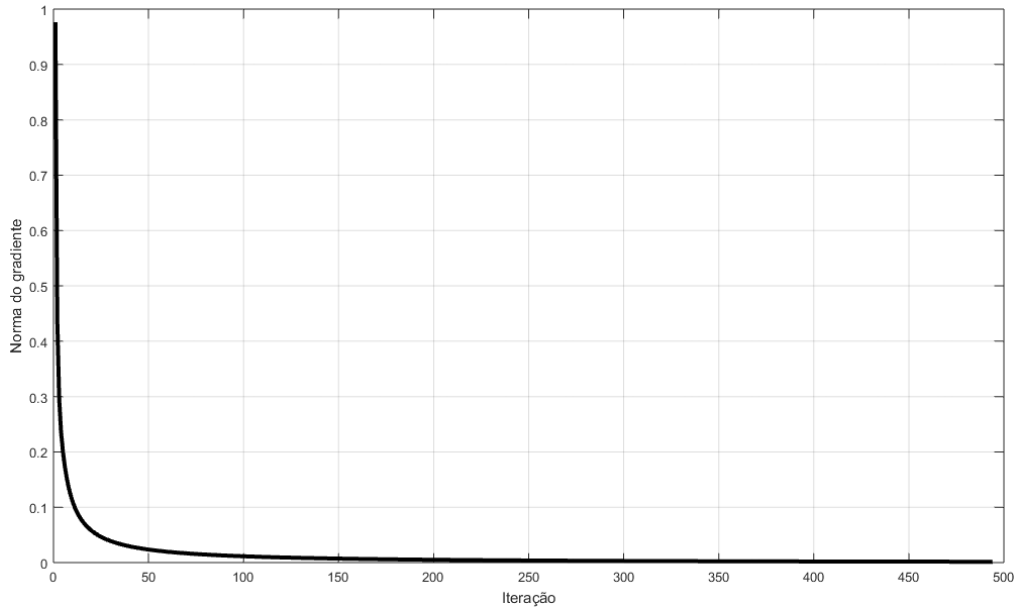


*Figura 5: Variação da norma do gradiente em função das iterações com α = 1.*

O custo final obtido foi de 0.092 e a norma final do gradiente foi 0.0015 com 494 iterações. Assim obtivemos os seguintes pesos para o vetor $w$:

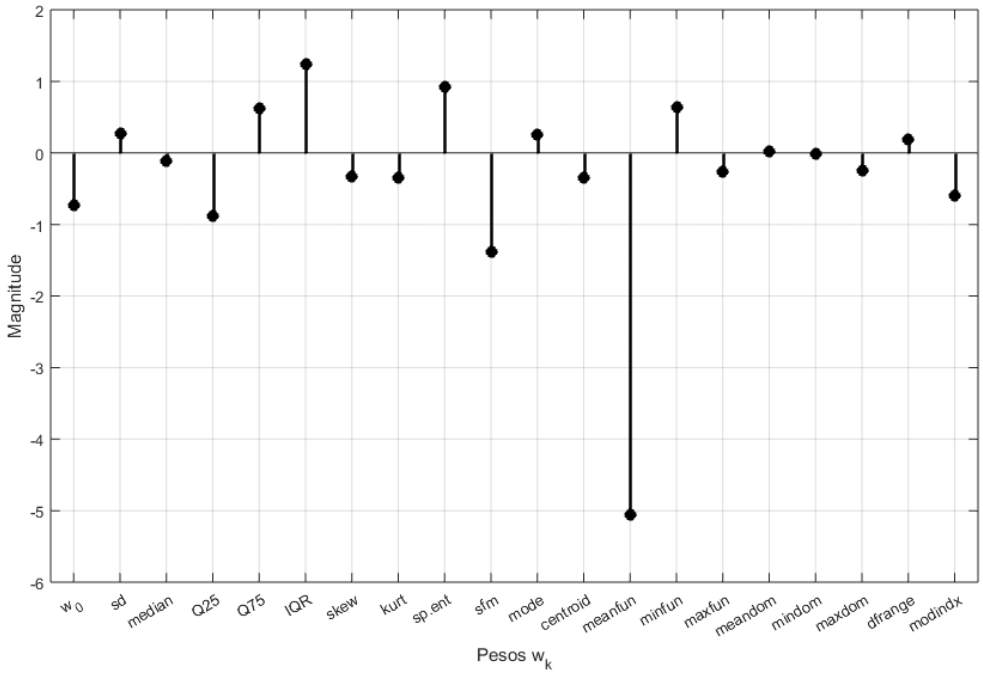


*Figura 6: Pesos calculados do vetor w.*

Aqui também é interessante notar que as variáveis de alta correlação com a classe e grande separação entre os dados nos histogramas, foram as que obtiveram maiores valores de $w$. Visto que os dados estão estandardizados, isso indica uma maior contribuição destas variáveis para a classificação.

Obtivemos também a curva ROC e o gráfico da $F_1 - Medida$ para os dados de teste variando o threshold com um passo 0.001. Estes gráficos podem ser encontrados nas Figuras 7 e 8, respectivamente.

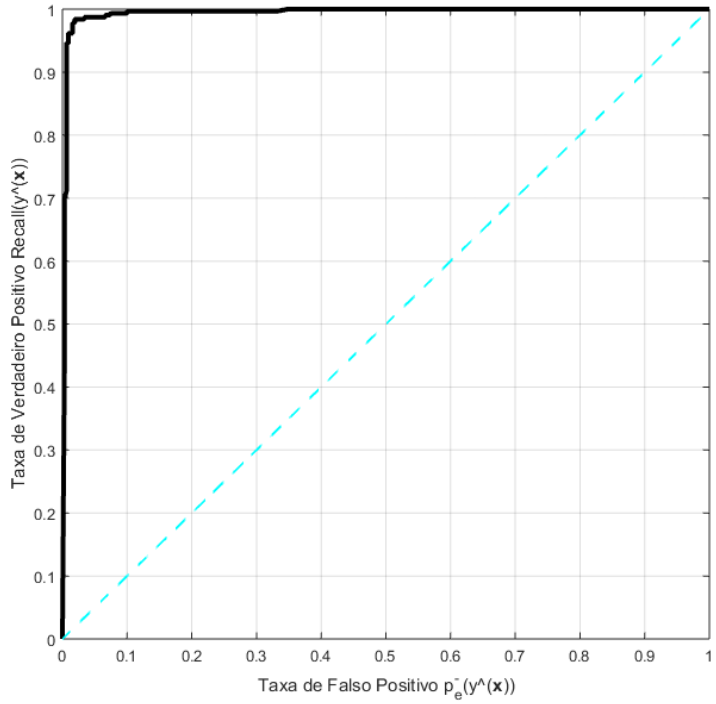


*Figura 7: Curva ROC do classificador.*

Através do gráfico da ROC vemos que se trata de um bom classificador, pois o gráfico se aproxima muito das retas x = 0 e y = 1, se distanciando assim do classificador aleatório denotado pela reta tracejada. Tal desempenho traduz-se também numa área da curva ROC de 0.995, ou seja, um valor próximo de 1, o que também caracteriza um bom classificador.

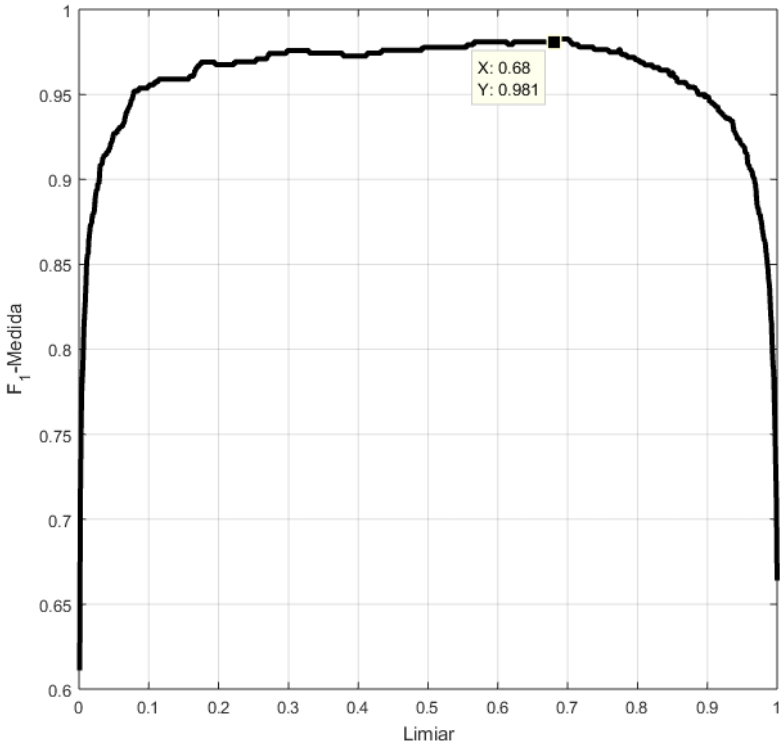


*Figura 8: $F_1 - Medida$ do classificador em função do threshold.*

Já através do gráfico da $F_1 - Medida$ vemos que conforme aumentamos o threshold temos um valor máximo de 0.981 para o threshold de 0.68.

## Item c

Utilizando como threshold o valor de 0.68, que obteve o maior valor da $F_1 - Medida$, geramos a matriz de confusão do classificador, que pode ser observada na Figura 9.

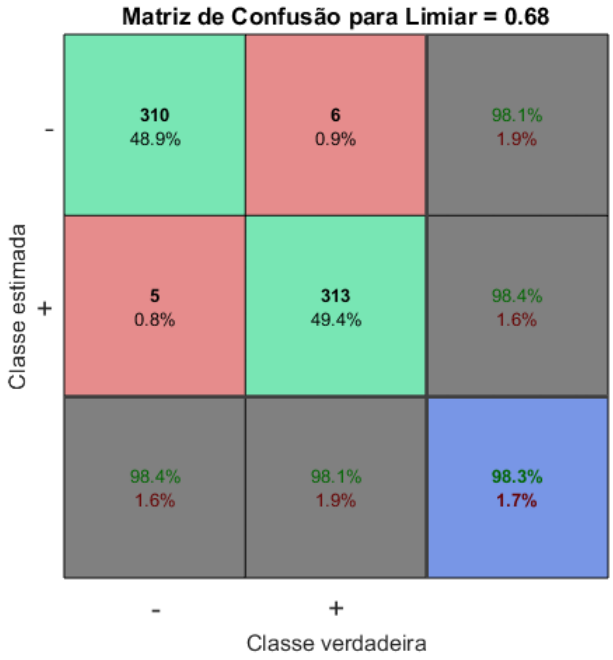


*Figura 9: Matriz de confusão para limiar = 0.68.*

Tal classificador obteve uma acurácia de 98.3% e conseguiu classificar bem as duas classes (uma alta taxa de TP e TN). É interessante notar que caso utilizássemos o threshold de 0.5, que parecia ser uma escolha natural, obteríamos um resultado pior, conforme apresentado na matriz de confusão da Figura 10.

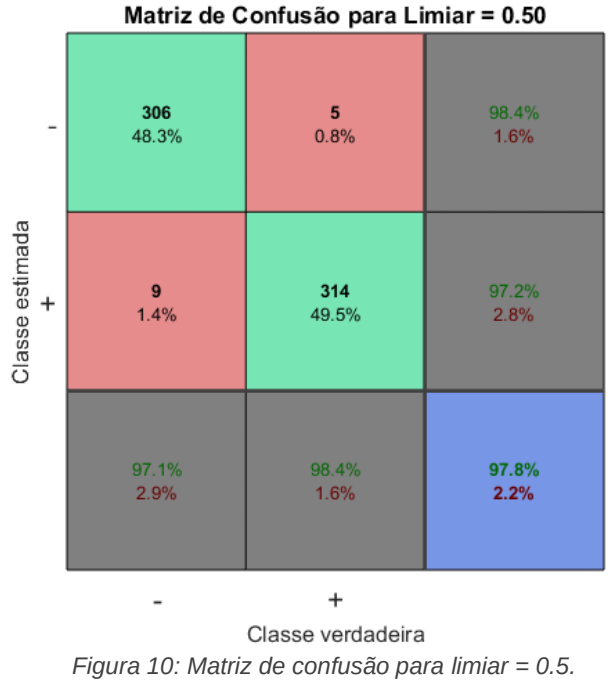


*Figura 10: Matriz de confusão para limiar = 0.5.*

Outro aspecto interessante é que, como a variável ‘meanfun’ aparenta mostrar uma separação natural para os dados, decidimos fazer dois novos treinamentos, sendo o primeiro somente com a variável ‘meanfun’ e o outro com todas as variáveis, exceto a ‘meanfun’. Para o treinamento apenas com a variável ‘meanfun’ obtivemos a curva de $F_1 - Medida$ da Figura 11.

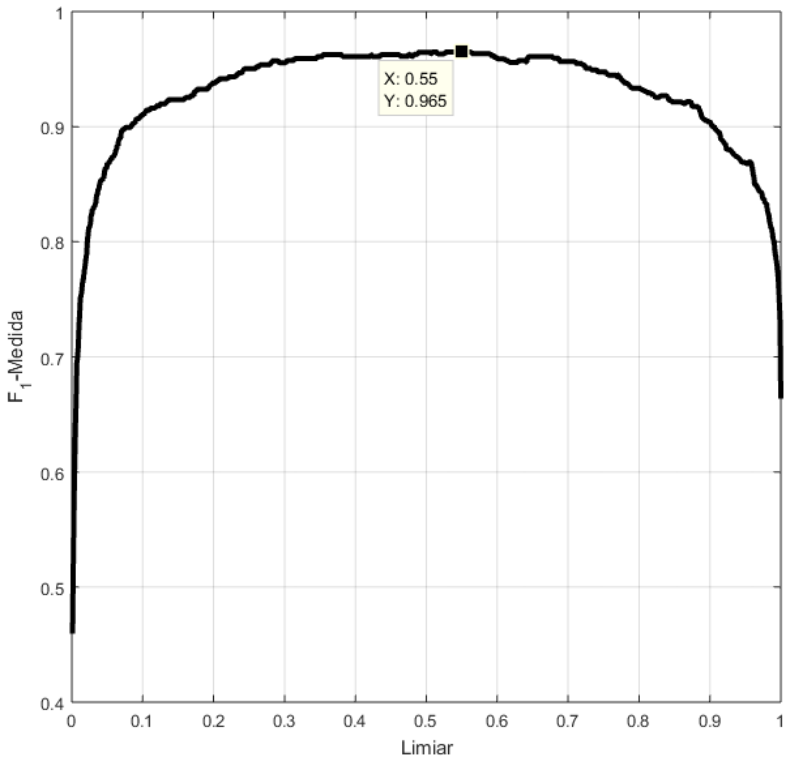


*Figura 11: $F_1 - Medida$ do classificador treinado somente com o atributo "meanfun".*

A primeira coisa a se notar se compararmos essa curva com a curva utilizando todas as variáveis, é que essa curva da $F_1 - Medida$ é menos "estável", ou seja, uma pequena variação no threshold pode levar a uma maior variação na qualidade do classificador. Desta forma é mais importante uma correta seleção do threshold neste caso. Com esse gráfico selecionamos o valor de threshold de 0.55, $F_1 - Medida$ igual a 0.965 e geramos a matriz de confusão para os dados de teste, exibida na Figura 12.

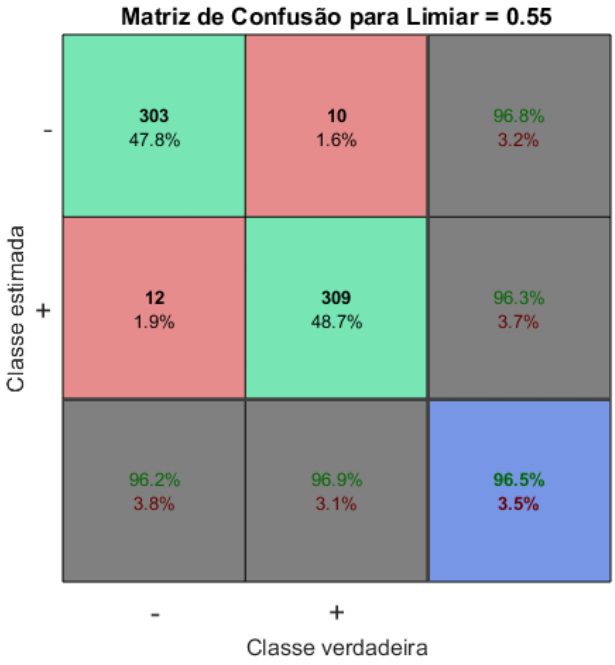


*Figura 12: Matriz de confusão para limiar = 0.55 (treinamento somente com "meanfun").*

Assim notamos que apenas com a variável ‘meanfun’ obtivemos uma acurácia de 96.5% muito próxima da acurácia de teste utilizando todas as variáveis que foi de 98.3%. Já treinando sem a variável ‘meanfun’, obtivemos o gráfico de $F_1 - Medida$ e a matriz de confusão apresentados nas Figuras 13 e 14, respectivamente. Em tais figuras é possível observar uma sensível piora do classificador, corroborando com a ideia de que este atributo tem bastante importância na classificação.

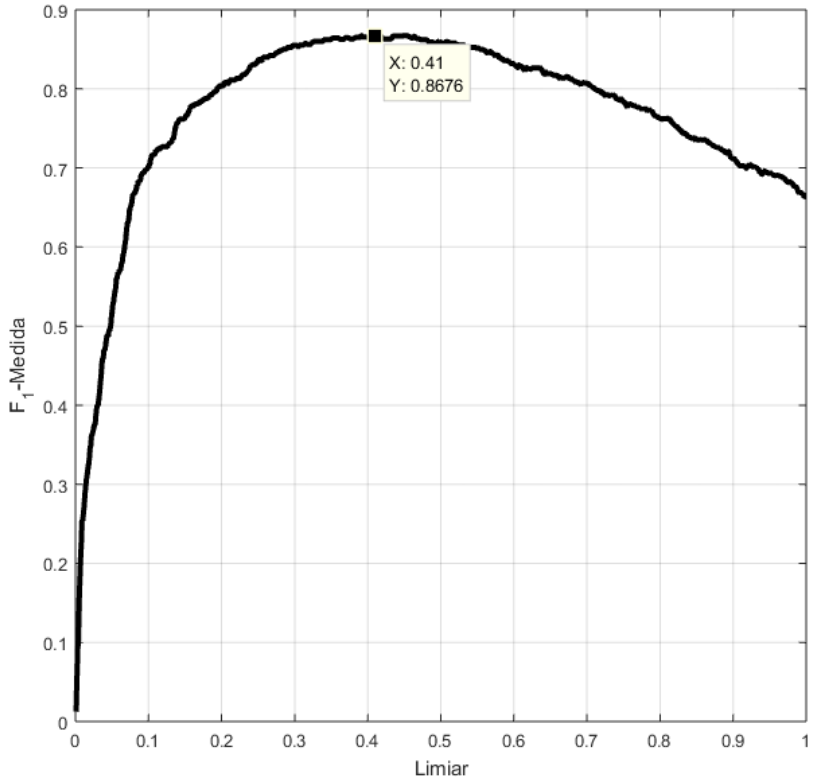


*Figura 13: $F_1 - Medida$ do classificador treinado com todos os atributos, exceto o “meanfun”.*

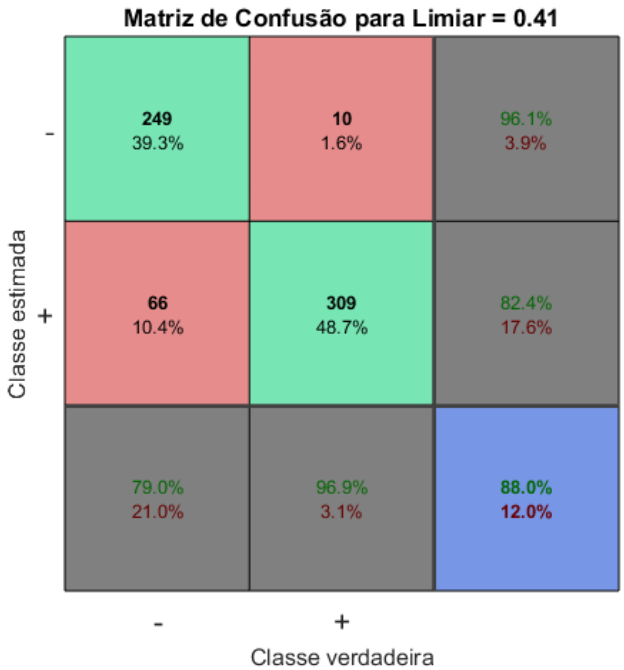


*Figura 14: Matriz de confusão para limiar = 0.41 (treinamento sem “meanfun”).*

# Exercício 2

## Item a

Este problema foi atacado com duas abordagens diferentes. Na primeira delas foram utilizados 6 classificadores no modelo um contra todos e na segunda foi utilizado o softmax. Em ambas as implementações o critério de escolha da classe foi o maior valor de saída dos classificadores e os critérios de parada para o treino foram os mesmos do exercício 1. Para a abordagem um contra todos podemos observar as convergências da entropia cruzada e da norma do gradiente com um passo α = 0.03 para cada classificador nas Figuras 15 e 16, respectivamente.

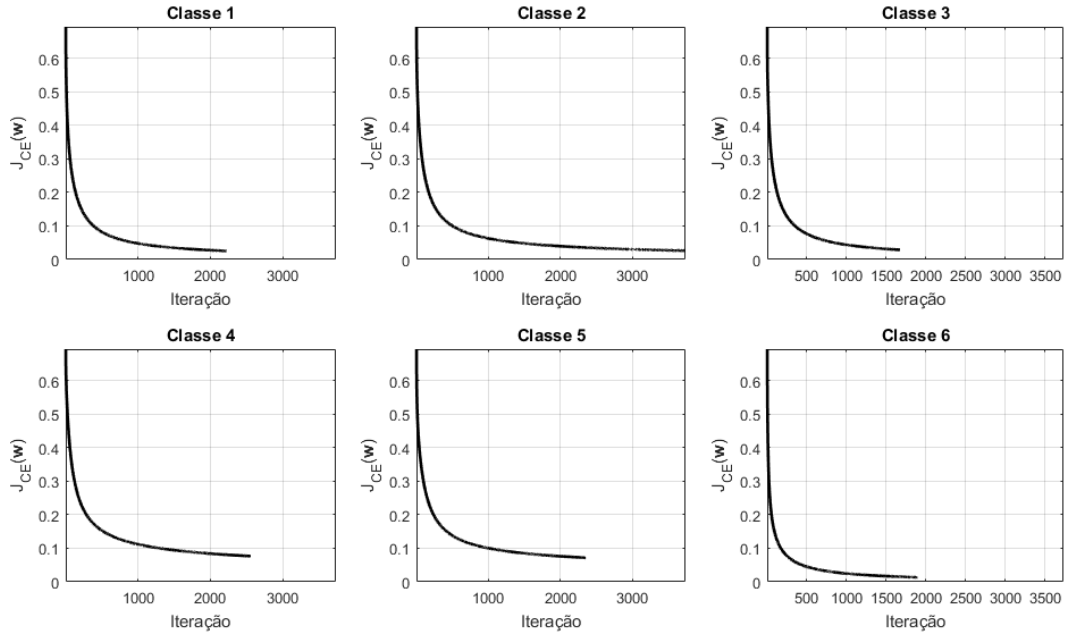


*Figura 15: Variação da entropia cruzada com passo α = 0.03 para cada classificador.*

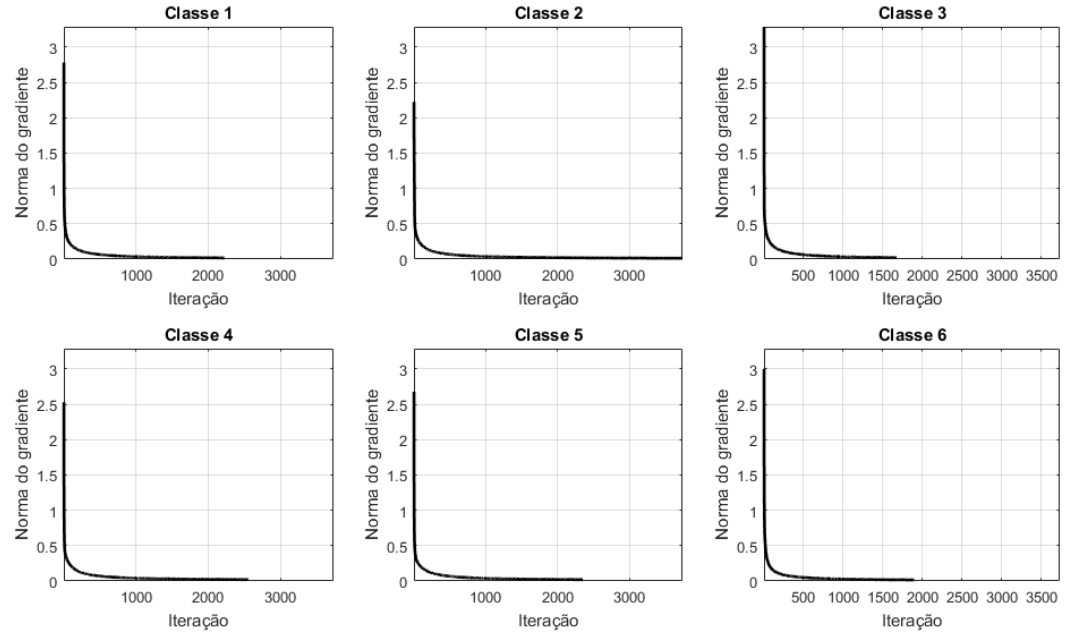


*Figura 16: Variação da norma do gradiente com passo α = 0.03 para cada classificador.*

A matriz de confusão para o classificador um contra todos utilizado nos dados de teste é apresentada na Figura 17.

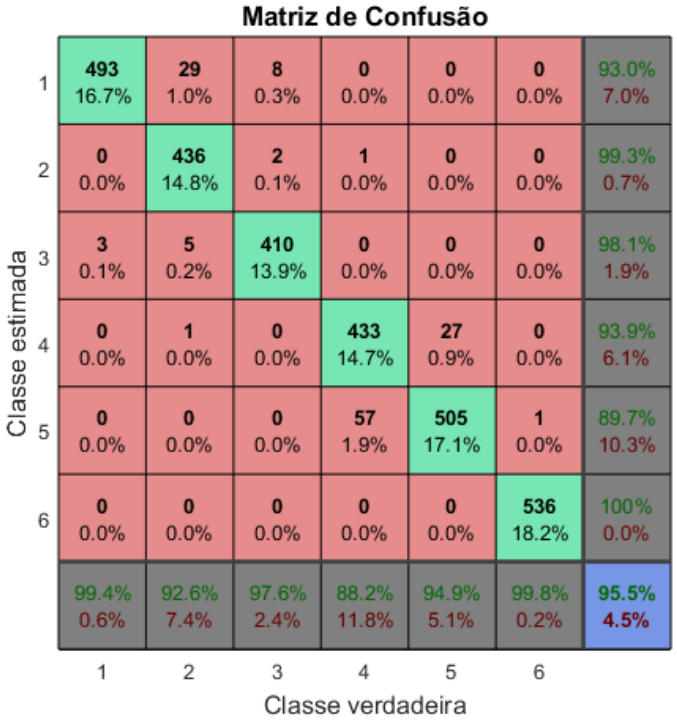


*Figura 17: Matriz de confusão do classificador um contra todos.*

Esta abordagem teve maior dificuldade em classificar a classe 4 e teve maior facilidade em classificar a classe 6.

A abordagem softmax gerou os resultados de entropia cruzada e da norma do gradiente também com um passo α = 0.03 exibidos nas Figuras 18 e 19, respectivamente.

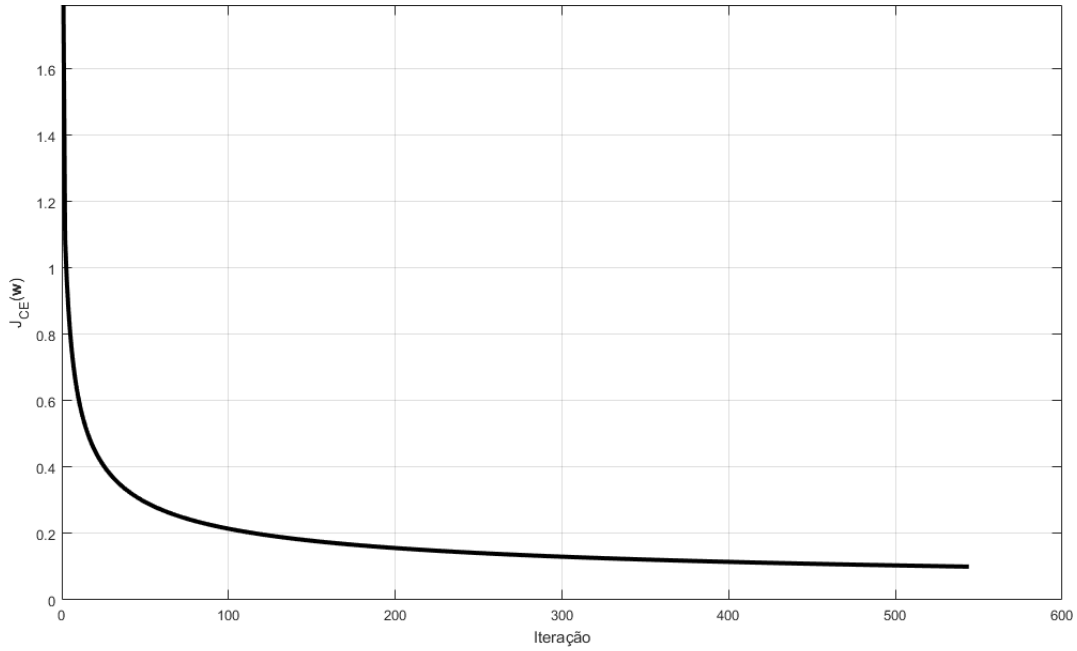


*Figura 18: Variação da entropia cruzada com passo α = 0.03 para o classificador softmax.*

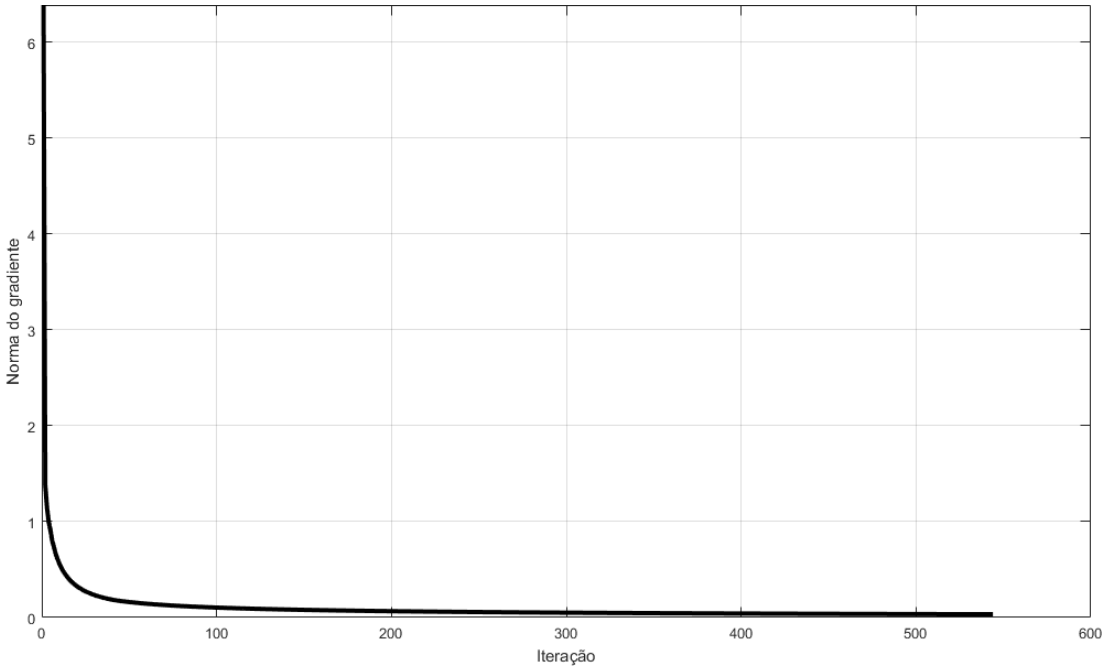


*Figura 19: Variação da norma do gradiente com passo α = 0.03 para o classificador softmax*

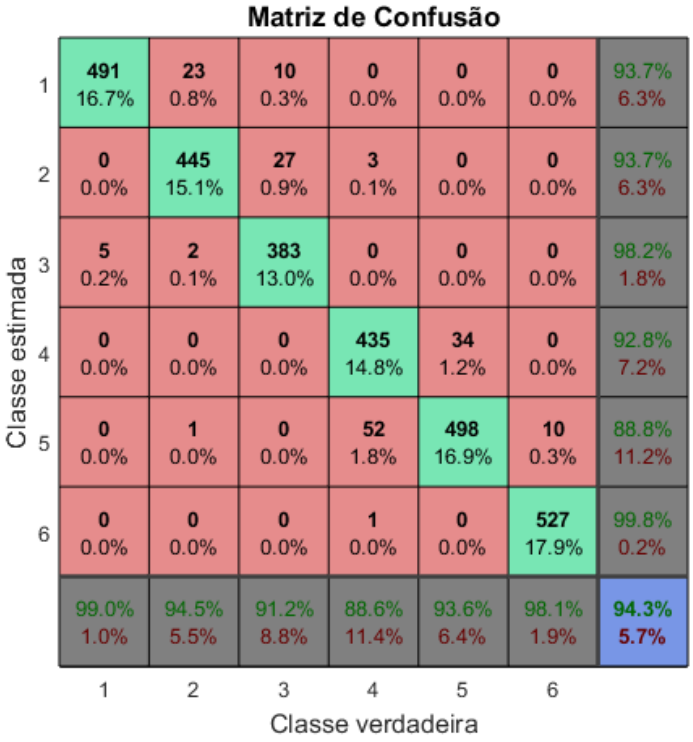


*Figura 20: Matriz de confusão do classificador softmax.*

Podemos observar na matriz de confusão da Figura 20 que a abordagem por softmax exibiu as mesmas facilidades e dificuldades em classificar determinadas classes que a abordagem um contra todos, porém obtendo uma acurácia ligeiramente menor. Para avaliar o desempenho destes classificadores escolhemos a $F_1 - Medida$ com macro-média conforme exposto no artigo. Utilizamos a macro-média devido esta considerar igualmente todas as classes. É digno de nota que percebemos em testes que a $F_1 - Medida$ com micro-média para multiclasses, utilizando todas as classes na média, é igual a

precisão, o recall e a acurácia. Durante a implementação notou-se ainda uma diferença no cálculo da $F_1 - Medida$ para multiclasse com macro-média, entre a fórmula do artigo e a fórmula utilizada pelo pacote *sklearn*. A fórmula do artigo calcula a precisão e o recall médios antes de calcular a $F_1 - Medida$, já o pacote *sklearn* calcula a $F_1 - Medida$ para cada classe e depois realiza a média. A $F_1 - Medida$ foi escolhida por possibilitar uma análise que leva em conta tanto o recall quanto a precisão, obtendo os seguintes resultados:

| | $F_1 - Medida$ | Tempo Execução (s) |
|---|---|---|
| Um contra todos | 0.9555 | 432 |
| Softmax | 0.9433 | 19 |

Comparando as duas abordagens tanto em tempo quanto em número de iterações o softmax foi mais rápido, sendo que em tempo ele foi quase 20 vezes mais rápido. Em termos de acurácia e $F_1 - Medida$ as duas abordagens apresentaram desempenhos similares.

## Item b

Aqui implementamos o algoritmo do k-nearest neighbors utilizando a distância Euclidiana e variando o k entre 1 e 200. Como critério de desempate foi utilizado o vizinho de menor distância dentre as classes do empate. Na Figura 21, podemos ver também a evolução do número de amostras de teste que geraram empates em função do número de vizinhos k, onde é interessante observar uma certa tendência para mais empates quando k é par, mesmo quando se usa um grande número de vizinhos.

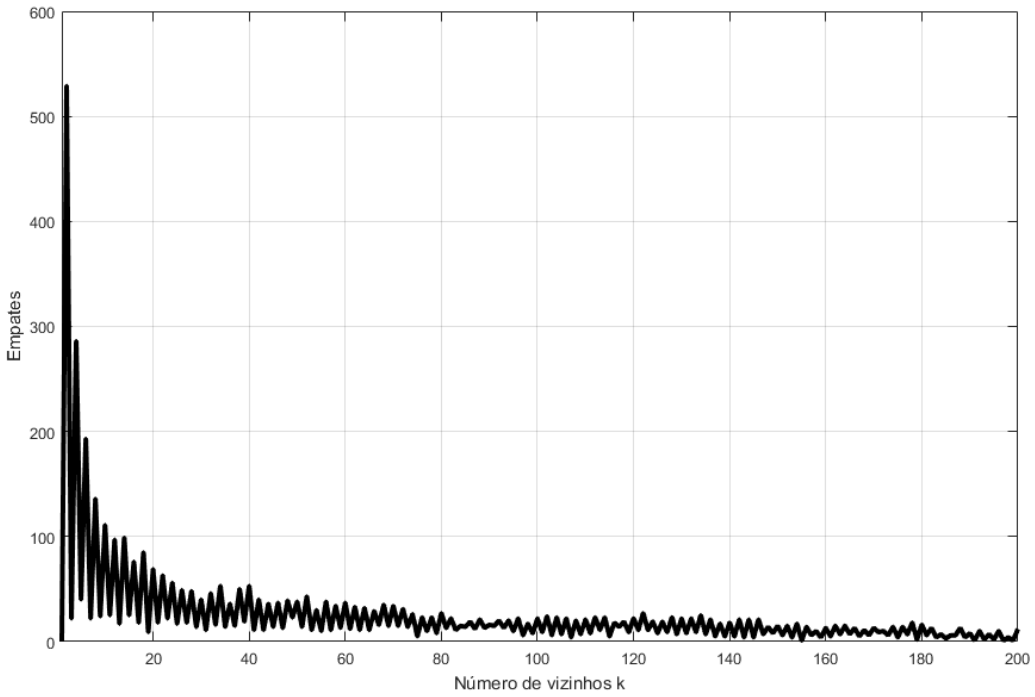


*Figura 21: Gráfico do número de empates em função do número de vizinhos.*

A métrica de avaliação foi a mesma $F_1 - Medida$ utilizada no item “a” e o gráfico deste resultado para cada valor de k pode ser visto na Figura 22.

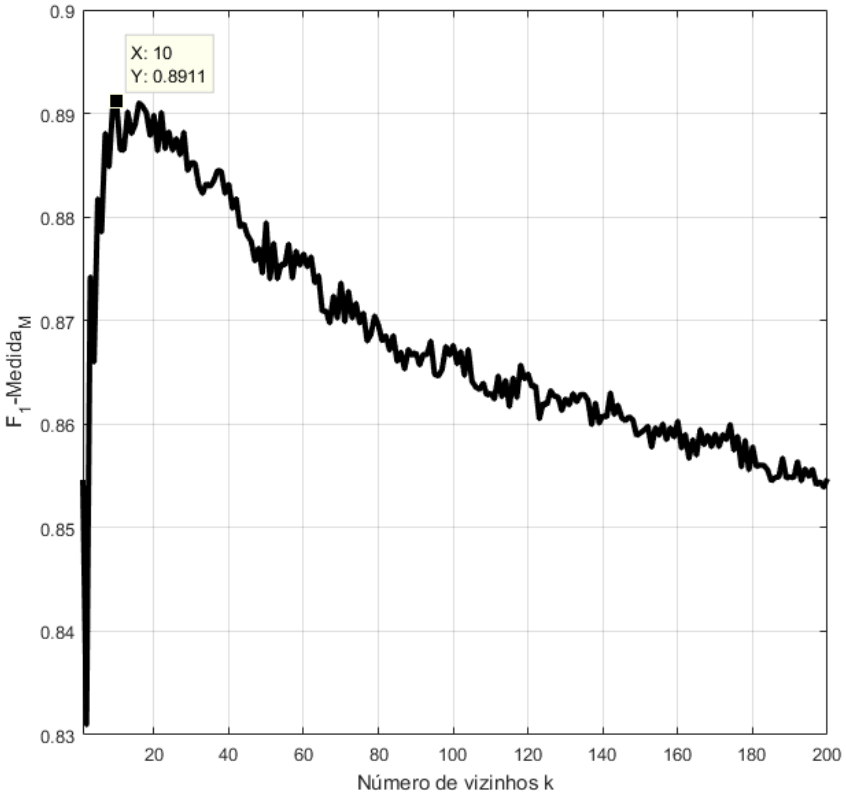


*Figura 22: Gráfico da $F_1 - Medida$ em função do número de vizinhos.*

O melhor valor da $F_1 - Medida$ foi de 0.8911 com k = 10, e para este k obtêm-se a seguinte matriz de confusão:

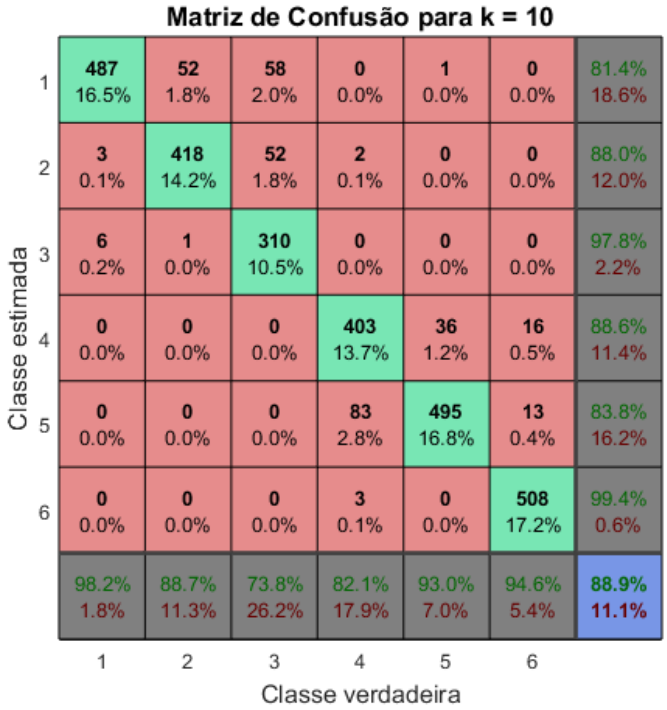


*Figura 23: Matriz de confusão do k-nearest neighbors para k = 10.*

Vemos que este classificador teve uma maior dificuldade em classificar a classe 3, uma dificuldade diferente da obtida pelos métodos do item “a”. Verificamos também que, tanto a $F_1 - Medida$ quanto a acurácia do k-nearest neighbors foram significativamente piores do que as duas abordagens do item “a”. Entretanto, considerando que este é um método sem aprendizado, os valores de acurácia e da $F_1 - Medida$ obtidos ainda podem ser considerados altos.