**Project Assignment**

**Chasing Phishing URLs**

Reconhecimento de Padrões

Vasco Rodrigues Nr.º 2024169097

21/03/2025

Índice

[1. Introdução 4](#_Toc193486279)

[2. Implementação 5](#_Toc193486280)

[2.1. Normalização 5](#_Toc193486281)

[2.1.1 Min-max 5](#_Toc193486282)

[2.1.2 Z-score 5](#_Toc193486283)

[2.2. Kruskal-Wallis 6](#_Toc193486284)

[2.3. PCA 6](#_Toc193486285)

[2.4. Minimum Distance Classifier 7](#_Toc193486286)

[2.4.1 Euclidean distance discriminant 7](#_Toc193486287)

[2.4.2 Mahalanobis distance discriminant 8](#_Toc193486288)

[2.5. Fisher LDA 9](#_Toc193486289)

[2.6. Cross Validation 11](#_Toc193486290)

[3. Resultados e Comparações 12](#_Toc193486291)

[3.1. Minimum Distance classifier 12](#_Toc193486292)

[3.1.1 Euclidean Distance Discriminant 12](#_Toc193486293)

[3.1.2 Mahalanobis Distance Discriminant 12](#_Toc193486294)

[3.2. Fisher LDA 12](#_Toc193486295)

[4. Discussão e Conclusão 13](#_Toc193486296)

[5. Referências 14](#_Toc193486297)

Índice Figuras

[Figura 2‑1 - Demonstração de kaiser criterium e scree test 7](#_Toc193486298)

[Figura 2‑2 - Hiperplano de decisão do Fisher LDA 10](#_Toc193486299)

# Introdução

Atualmente, enquanto navegamos na internet, estamos em constante perigo e sujeitos a tentativas de roubo da nossa informação pessoal sem a nossa permissão. Por esta razão, o trabalho proposto e realizado no âmbito da cadeira de Reconhecimento de Padrões apresenta uma grande importância para assegurar a nossa segurança. O objetivo do mesmo é a identificação de *URLs* maliciosos através de um modelo de classificação.

Para garantir que esta classificação seja a melhor possível, dados os dados disponíveis, serão realizadas diversas operações, como *feature selection*, *cleaning* e *normalization of data*, e *dimensionality reduction*. Além disso, serão utilizados diversos modelos, tendo os seus resultados comparados entre si, de forma a garantir que seja escolhido o modelo que melhor se comporta em dados nunca antes vistos.

Dentro das operações de *feature selection*, serão utilizados testes como o de *Kruskal-Wallis*, que permitirá verificar como as características se relacionam com a variável *target*, podendo assim escolher aquelas que são mais discriminativas e excluir as que menos. Será também verificada a matriz de correlação para identificar dados altamente correlacionados e, consequentemente, remover características redundantes.

Já nas operações de *dimensionality reduction*, será testada a utilização de *PCA* (*Principal Component Analysis*), que procura representar os dados num novo eixo, de forma a obter uma melhor representação e uma mais fácil discriminação dos dados.

Os modelos que serão utilizados são: *Minimum Distance Classifier*, tanto com *Euclidean Distance Discriminant* como com *Mahalanobis Distance Discriminant*, e *Fisher LDA* (*Linear Discriminant Analysis*). Ambos os modelos serão treinados e testados no mesmo *dataset*, utilizando *cross-validation*, para garantir resultados de treino/teste mais coerentes. As métricas utilizadas para avaliar estes modelos serão: *sensitivity*, *specificity*, *F1-score* e *ROC Curves*. A *accuracy* não será considerada, pois, como é referido no enunciado do próprio projeto, este é um *dataset* desequilibrado (*imbalanced dataset*), e a *accuracy*, no seu cálculo, não tem em consideração este facto.

# Implementação

Antes da especificação da implementação dos modelos é importante mencionar que as características categóricas e binárias presentes no *dataset* foram totalmente removidas isto porque os modelos dados até agora apenas lidam com dados contínuos e são próprios para esse tipo de dados.

## Normalização

O processo de normalização consiste na conversão dos valores de dados contínuos para uma escala comum, de forma a garantir que todas as características contribuam de forma equilibrada para o processo de classificação.

Neste trabalho foram considerados dois métodos de normalização: *min-max* e *z-score*.

### Min-max

O método *min-max* transforma os dados de forma que estes se situem no intervalo 0 e 1. Para isso, utiliza o valor mínimo e máximo de cada característica. O processo é o seguinte:

1. Subtrai-se o valor mínimo da característica a cada valor dos dados.
2. Divide-se o resultado pela diferença entre o valor máximo e o mínimo da característica.

A normalização *min-max* é expressa pela fórmula:

Onde x é o valor original, min(X) é o valor mínimo da característica, e max(X) é valor máximo.

Neste método *outliers* podem ter uma grande influência, pois eles afetam diretamente os valores de mínimo e máximo

### Z-score

O método *z-score*, transforma os dados de forma que tenham uma média de 0 e um desvio padrão de 1. Para isso, utiliza a média e o desvio padrão de cada característica. O processo é o seguinte:

1. Subtrai-se a média da característica a cada valor dos dados.
2. Divide-se o resultado pelo desvio padrão da característica.

A normalização *z-score* é expressa pela fórmula:

Onde x é o valor original, μ é a média da característica e σ é o desvio padrão.

Este método é mais robusto à presença de *outliers*, pois a média e o desvio padrão são menos sensíveis a valores extremos do que o mínimo e o máximo. No entanto, assume que os dados seguem uma distribuição aproximadamente normal, o que nem sempre é verdade.

## Kruskal-Wallis

Este teste permite verificar quais características são mais discriminativas em relação a uma dada classe, ou seja, quais *features* apresentam diferenças significativas nas suas distribuições entre as diferentes classes.

Para a sua implementação foi utilizada o módulo *kruskal* da biblioteca *scipy* para o cálculo do *H-value*, no entanto o seu cálculo é feito da seguinte forma:

1. É selecionado uma *feature;*
2. Para cada classe do target são atribuídos *ranks* por ordem crescente em relação aos valores da *feature* atual;
3. É calculada a média do *rank* para cada classe da variável target*;*
4. É calculada a média de todos os *ranks*;
5. É calculado o *H-value* que tem a seguinte formula

Onde n = número total de observações, c é o número de classes da variável target e ni é o tamanho de cada grupo.

A *feature* que obtiver o *H-value* maior é aquela que melhor discrimina os dados. Isto pode então ser utilizado para *feature selection*.

Como este teste apenas verifica o poder discriminativo de uma característica é necessário descartar *features* que sejam altamente correlacionadas para isso foi feito o uso da matriz de correlação e serão removidas aquelas que apresentam valores muito altos entre si novamente, *feature selection*

## PCA

O *Principal Component Analysis* (*PCA*) tem como objetivo principal descobrir um novo conjunto reduzido de *features* com menos redundância, com perda mínima de informação, esta preservação de informação é medida em termos de variância. Ele projeta os dados em direções onde os dados apresentam maior variação.

Para isto o PCA segue os seguintes Passos:

1. Normalização dos dados para ter média zero e desvio padrão de 1. Para garantir que a variância seja comparável entre as características
2. Cálculo da matriz de covariação.
3. Cálculo dos *eigenvectors* e *eigenvalues*, pois estes representam as direções no espaço dos dados e a quantidade de variância capturada por cada componente principal, respetivamente.
4. Ordenação dos *eigenvectors* e *eigenvalues* em ordem decrescente
5. Seleção dos componentes principais, foram implementados dois métodos:
   1. *Kaiser criterium*: Selecionar o eigenvalue antes do plot cair para baixo de 1, *Figura 2‑1*
   2. *Scree test*: Selecionar o eigenvalue que estabilizou a curva, *Figura 2‑1*

*Uma imagem com diagrama, file, Gráfico, captura de ecrã

Os conteúdos gerados por IA poderão estar incorretos.*

Figura 2‑1 - Demonstração de kaiser criterium e scree test

Desta forma serão selecionadas um novo conjunto reduzido de *feature* que se insere no âmbito da *feature reduction*/*dimensionality reduction*.

## Minimum Distance Classifier

Neste classificador uma amostra é classificada como a classe cuja distribuição conhecida ou estimada mais se assemelha com amostra em questão. Esta medida de semelhança é a distância no espaço de distribuição, ou seja, calcula-se a distância entre a amostra e as representações das diferentes classes (usualmente a médias dos dados delas), atribuindo-se a amostra à classe que estiver mais "próxima".

Este tipo de classificador baseia-se na ideia de que as amostras de uma mesma classe tendem a estar próximas umas das outras no espaço de características, enquanto amostras de classes diferentes tendem a estar mais afastadas. Para isto, é necessário definir métricas de distância. Entre várias existentes as mais comuns são *Euclidean Distance* e *Mahalanobis Distance*:

* *Euclidean Distance*: Mede a distância em “linha reta” entre dois pontos no espaço. E por consequência considera que as distribuições das classes são esféricas.
* *Mahalanobis Distance*: Leva em consideração a correlação entre as características e as variações nas diferentes direções do espaço. Esta é particularmente útil quando as distribuições das classes não são esféricas ou têm escalas diferentes

### Euclidean distance discriminant

Como mencionado anteriormente este tipo de discriminante considera que as classes e as suas distribuições são esféricas pela forma como é calculada a sua distância. A distância euclidiana mede a distância "em linha reta" entre dois pontos no espaço de características, sem considerar a correlação entre as variáveis ou as diferenças de escala.

Esta abordagem pode não ser a melhor opção quando as distribuições das classes são alongadas ou têm orientações diferentes, pois ignora a estrutura de covariância dos dados.

Tendo em conta esta informação para a sua implementação é necessário:

1. É calculada a média de cada classe da variável *target* com base nos dados
2. É feita a classificação de novos pontos calculando a distância euclidiana entre as médias da classe. A distância euclidiana é dada pela fórmula:

### Mahalanobis distance discriminant

Ao contrário do discriminante anterior, o *Mahalanobis distance discriminant* leva em consideração as variações nas diferentes direções do espaço. Isto é feito incorporando a matriz de covariação no cálculo da distância, o que permite ajustar a métrica à forma e à orientação da distribuição dos dados.

Esta característica torna o *Mahalanobis distance discriminant* mais flexível e robusto, especialmente quando as classes têm distribuições elípticas ou quando as características estão correlacionadas. No entanto, o cálculo da matriz de covariação pode ser computacionalmente mais exigente.

Tendo em conta esta informação para a sua implementação é necessário:

1. É calculada a média de cada classe da variável *target* com base nos dados
2. É calculada a inversa da matriz de covariação para cada classe
3. É feita a classificação de novos pontos calculando a distância *mahalanobis* entre cada ponto e a média das classes, a distância mahalanobis é dada pela fórmula:

Onde p é o vetor que representa o ponto no espaço, μ é o vetor que representa a média das classes da variável target, C é a matriz de covariação.

## Fisher LDA

Enquanto técnicas como o *PCA* (*Principal Component Analysis*) se focam em encontrar uma representação dos dados num subespaço linear de dimensão reduzida que captura as maiores variâncias nos dados, o *Fisher's Linear Discriminant Analysis* (*LDA*) tem como objetivo maximizar a separabilidade entre classes.

O *PCA* não tem em conta a informação das classes, ou seja, não leva em consideração as *labels* dos dados ao determinar as direções de máxima variância. Embora isso seja útil para representação dos dados, as direções de maior variância nem sempre são as melhores para discriminar duas classes distintas.

O *LDA*, por outro lado, procura encontrar uma projeção linear dos dados num subespaço que maximize a separação entre as classes. Para isso, utiliza um critério de otimização baseado na razão de *Fisher (Fisher's ratio)*, que mede a relação entre a variância entre classes (*inter-class*) e a variância dentro de cada classe (*intra-class*). Ao maximizar esta razão, o *LDA* garante que as projeções dos dados no novo subespaço mantenham as classes tão distantes quanto possível, enquanto minimiza a dispersão dentro de cada classe.

Por esta razão, não faz sentido aplicar métodos como a *PCA* antes de utilizar o classificador *Fisher LDA*. Isto porque a *PCA* representa os dados num novo eixo que maximiza a variância, mas ignora a informação das classes. Como resultado, esta transformação pode dificultar a tarefa do *Fisher LDA*, que procura encontrar um eixo que maximize a separação entre as classes. Por outras palavras a projeção inicial feita pelo PCA pode destruir as diferenças que existem entre as classes, reduzindo a eficácia do Fisher LDA na discriminação dos dados.

Para a sua implementação foi utilizado o módulo *LDA* da biblioteca *scipy*, no entanto este é implementado da seguinte forma:

1. Calcular as médias das classes da variável target
2. Calcular a matriz de dispersão dentro das classes, que é dada pela seguinte formula:
3. Calcular o vetor de projeção w, dado pela formula:
4. Projetar os dados no novo espaço através de w
5. Calcular o Hiperplano de decisão no novo espaço
6. Classificar dados segundo este hiperplano, *Figura 2‑2*

Uma imagem com texto, captura de ecrã, Gráfico, diagrama

Os conteúdos gerados por IA poderão estar incorretos.

Figura 2‑2 - Hiperplano de decisão do Fisher LDA

## Cross Validation

Para os obter dados de avaliação dos modelos mais coerentes foi utilizado *cross-validation*, que é uma técnica que parte o *dataset* em k partições, onde 1 delas é escolhida para teste e as restantes k-1 são utilizadas para treino, este processo é repetido k vezes. Isto permite obter resultados médios e portanto mais confiáveis.

# Resultados e Comparações

Tendo todas as implementações feitas basta agora fazer os testes para verificar qual combinação de classificadores se melhor comporta.

## Minimum Distance classifier

### Euclidean Distance Discriminant

### Mahalanobis Distance Discriminant

## Fisher LDA

# Discussão e Conclusão

# Referências

1. Wacker, A G, and D A Landgrebe. ‘Minimum Distance Classification in Remote Sensing’, n.d.
2. ‘Fisher Linear Discriminant - an Overview | ScienceDirect Topics’. Accessed 21 March 2025. <https://www.sciencedirect.com/topics/engineering/fisher-linear-discriminant>.
3. Alexander. ‘Mahalanobis Distance: Simple Definition, Examples’. Statistics How To, 8 May 2024. <https://www.statisticshowto.com/mahalanobis-distance/>.
4. GeeksforGeeks. ‘Euclidean Distance | Formula, Derivation & Solved Examples’, 18:50:03+00:00. <https://www.geeksforgeeks.org/euclidean-distance/>.