Aprendizagem 2024/25

Homework NTT Data – Group 041

(ist106322, ist106157)

Uma tarefa comum em problemas de aprendizagem é a classificação binária, a qual consiste na distinção entre duas classes, tal como no *dataset Breast Cancer Wisconsin*, onde se procura identificar se um determinado tumor é benigno ou maligno. Para abordar estes problemas, opta-se frequentemente pela *Logistic Regression* como modelo base, tendo em conta a sua simplicidade, baixa exigência computacional e eficiência em situações lineares. Porém, este algoritmo é adequado apenas para lidar com dados linearmente separáveis, o que não é o caso de inúmeros *datasets* encontrados em problemas reais. Como tal, tomamos em consideração o seguinte teorema: "Given a set of training data that is not linearly separable, one can with high probability transform it into a training set that is linearly separable by projecting it into a higher-dimensional space via some non-linear transformation Φ." [Cover (1965)]. *Radial Basis Function Networks* é um modelo mais avançado que procura aplicar esta transformação na sua *hidden layer*. Contudo, deparamo-nos com o seguinte obstáculo: "The number of units in the hidden layer is the same number N of units as the size of the training sample" [Wasserman (1993)]. Este problema resulta no fenómeno conhecido como "maldição da dimensionalidade". O aumento do número de unidades na *hidden layer* não só torna o modelo mais suscetível a problemas de *overfitting*, como também acarreta uma maior exigência de recursos computacionais para o processo de treino da *network*. Deste modo, uma solução para mitigar este desafio é a aplicação de técnicas de pré-processamento, em particular de *clustering*. Ao agrupar os dados em *clusters*, é possível identificar subconjuntos com características semelhantes e definir centros representativos (centroides). Estes centroides são então usados como centros das funções de base radial (*RBF*), em alternativa a usar todas as observações, o que permite diminuir substancialmente o número de *hidden units*. No presente documento, serão estudados estes temas, com uma abordagem que consiste em primeiro realizar *Logistic Regression* para estabelecer um ponto de referência, após o qual será efetuado *EM clustering* como etapa de pré-processamento e, por fim, será aplicada a *RBF Network*.

Como referido anteriormente, foi inicialmente aplicado o algoritmo de *Logistic Regression*, com o objetivo de definir um padrão de comparação e assim avaliar a eficácia dos modelos subsequentes e quantificar potenciais melhorias nos desempenhos. Após a divisão dos dados em conjuntos de treino e teste e normalização dos dados, para garantir uniformidade nas escalas das variáveis, foi possível treinar o modelo com uma exatidão de aproximadamente 98.74% no conjunto de treino e 98.25% no conjunto de teste. Para garantir maior confiabilidade nos resultados obtidos, em alternativa ao procedimento de divisão train_test_split, foi realizada também uma validação cruzada antes do *Logistic Regression*, cujo resultado de exatidão foi também cerca de 98.25%. No entanto, para os exercícios seguintes, foi utilizado o procedimento simples de divisão entre treino e teste, visando assegurar uma comparação entre os modelos mais direta e sem a introdução de variáveis adicionais que pudessem afetar a análise.

De seguida, aplicámos o algoritmo *EM* para agrupar as observações em *clusters*, segundo a *Gaussian Mixture assumption*. Ao contrário do *K-means*, este algoritmo não

realiza um *hard assignment* de pontos a *clusters*, mas sim um *soft assignment*, isto é, a cada ponto são atribuídas probabilidades de pertença aos clusters.

Para tal, utilizámos a função *GaussianMixture* da biblioteca *sklearn*, especificando alguns parâmetros importantes relativos à inicialização. O método de inicialização escolhido consiste em selecionar aleatoriamente observações do *data set*, as quais serão os centróides iniciais. Além disso, escolhemos realizar duas inicializações, isto é, correr o algoritmo duas vezes com *seeds* diferentes, sendo que o modelo selecionará a melhor solução. Desta forma, asseguramos que os resultados sejam mais robustos e representem melhor a estrutura dos dados, reduzindo o risco de convergir para soluções subótimas sem aumentar excessivamente o esforço computacional.

A normalização dos dados não foi necessária na formulação deste modelo, uma vez que o algoritmo *EM* faz uso da distância de *Mahalanobis* no cálculo das *likelihoods*. A distância de *Mahalanobis* mede a distância entre um ponto e uma distribuição, tendo em conta as relações entre variáveis na sua fórmula:

$$D_M = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)}$$

Ao usar a inversa da matriz de covariâncias, esta métrica toma em consideração as escalas de cada *feature*, bem como as suas variâncias e covariâncias, ao contrário da distância Euclidiana. Sendo assim, o *scaling* torna-se irrelevante neste caso. Note-se que, se a escolha para o método de inicialização fosse o algoritmo *K-means* ou *K-means++*, a normalização de variáveis já seria pertinente.

Repetindo o processo para diferentes números de *clusters* (k) e calculando a silhueta para cada modelo, obtivemos um gráfico que nos permite avaliar a relação entre número de *clusters* e qualidade *clustering*. Através da observação do gráfico da Figura 1, verificamos que existe uma tendência decrescente da silhueta à medida que k aumenta, ou seja, quanto maior for o número de *clusters* considerados, pior será a separação e coesão dos mesmos. Por forma a confirmar, recorremos ao cálculo da correlação de Spearman e obtivemos o valor arredondado de -0.66 o que indica uma forte relação inversa entre as variáveis. O correspondente p-value de 0.026 (2.6%), inferior aos valores de significância de 5% e 10%, permite-nos concluir que os resultados obtidos são estatisticamente significativos. Finalmente, inferimos que o valor ótimo de k equivale a 2, pois corresponde ao valor de silhueta mais elevado. Dada esta informação, podemos construir um gráfico (Figura 2) que ilustra em detalhe a silhueta de cada um dos clusters e respetivas observações, possibilitando uma visualização útil dos resultados do *EM clustering* para k=2.

Uma vez aplicado o algoritmo *EM* ao conjunto de dados e analisados os seus resultados, podemos agora explorar as potencialidades da técnica de *clustering*, nomeadamente como transformação que permite obter uma representação alternativa dos dados. Esta transformação será útil futuramente para o treino da *RBFN* na pergunta 5. A motivação para este exercício encontra-se na chamada "maldição da dimensionalidade", que neste caso diz respeito ao tamanho do conjunto de dados, isto é, o número de observações. O *dataset* que tomamos como objeto de estudo não apresenta uma dimensionalidade extraordinariamente elevada, sendo composto por 569 observações. No entanto, à medida que

este número cresce, as consequências podem tornar-se alarmantes, entre as quais se destacam o poder computacional requerido e o risco de *overfitting*, o qual ocorre devido ao facto de o *feature space* ficar maioritariamente vazio e, por consequência, não ser possível obter uma boa capacidade de generalização. Em particular, para *RBF networks*, este passo demonstra ser bastante eficaz e valioso, reduzindo a dimensão da *hidden layer*, a qual corresponde ao número de observações.

De facto, é possível transformar o conjunto de dados através do modelo *EM* obtido anteriormente, mapeando cada observação para um vetor das probabilidades de pertença a cada *cluster*. Ao transformarmos os dados em probabilidades dos *k-clusters*, estamos, efetivamente, a reduzir a dimensionalidade de d para k, onde d é o número de *features* original, que corresponde a 30 neste caso (excluindo a variável *ID*). O resultado traduz-se num conjunto de dados com uma nova representação baseada em probabilidades Gaussianas, o qual pode ser usado, posteriormente, com o propósito de treinar o modelo de classificação de *Logistic Regression*. Este será o procedimento usado na formulação do modelo *RBFN*.

Podemos, então, treinar a Logistic Regression para cada número de *clusters* k, fazendo uso do conjunto de dados de treino transformados. Em seguida, aplicamos o modelo de forma a realizar a classificação dos dados de teste, também estes transformados. Comparando as previsões obtidas com as *labels* reais, é-nos possível analisar a *accuracy* do modelo para cada k e ainda estabelecer relações entre esta, o número de clusters e a qualidade de *clustering*.

Observando o gráfico resultante (Figura 3), verificamos que a accuracy da logistic regression se mantém estável independentemente do número de clusters considerados e do valor de silhueta respetivo ($\approx$0.92 para k=2). Podemos, então, inferir que não existe uma relação entre a performance deste modelo e a qualidade de clustering, isto é, uma má separação e coesão dos clusters não implicam uma menor capacidade de classificação. Este comportamento pode ser explicado pelo uso dos dados transformados no treino do modelo. Neste caso, as probabilidades assumem o papel de *features* e contêm informações relevantes para a classificação, mesmo quando a silhueta é baixa, proporcionando à *Logistic Regression* uma *accuracy* elevada e estável. Em suma, existe uma relação entre o número de *clusters* e a qualidade de *clustering*, tal como foi averiguado anteriormente, mas não existe uma relação entre estes e a accuracy do modelo de *Logistic Regression* treinado com o conjunto de dados transformado. As probabilidades resultantes do *EM clustering* permitem ao modelo realizar uma boa classificação independente do aumento do número de clusters e do decréscimo da silhueta.

Finalmente, para implementar uma *Radial Basis Function (RBF) Network*, recorre-se a um processo estruturado de transformação dos dados para um espaço de maior dimensionalidade, facilitando a separação linear entre classes. Estas *networks* são constituídas por três camadas: a *input layer*, que inclui as *features* dos dados iniciais; a *hidden layer,* onde ocorre a computação, sendo composta por um número de unidades correspondente à quantidade de dados de treino; a *output layer*, que, formada por apenas uma unidade, utiliza os valores provenientes da camada anterior, valores estes que representam já um problema linearmente separável e aplica ou *Logistic Regression* ou o algoritmo do Percetrão.

Um conceito fundamental das *RBFN* resume-se na proximidade dos pontos em relação a centros específicos. As n observações dos dados, uma para cada unidade da *hidden layer*, representam os centros das funções *RBF*. Nesta camada, é computada a distância euclidiana entre cada ponto e um determinado centro (raio). Seguidamente, esta distância sofre uma transformação não linear proveniente da aplicação de uma função de ativação, denominada neste contexto como uma *radial-basis function* [Powell (1988)]. Tipicamente, a função Gaussiana utilizada para este efeito é a seguinte:

$$\phi(x)=\exp(-\frac{\|x-x_k\|^2}{2\sigma^2})$$

que decai exponencialmente com a distância. Quanto mais próximo um ponto está de um centro, maior a ativação desse neurão da *hidden layer*, o que significa uma maior contribuição na previsão final.

Portanto, sem qualquer etapa de pré-processamento, todos os pontos pertencentes aos dados de treino são utilizados como centros nas unidades da *hidden layer*. Contudo, como referido anteriormente, esta abordagem resulta em elevados requisitos computacionais, uma vez que cada unidade processa distâncias em relação a todas as observações. Complementarmente, ao utilizar todas as amostras como centros, aumenta-se o risco de *overfitting*, dificultando a capacidade do modelo de generalizar para novos dados e comprometendo a eficácia do processo de treino. A aplicação de um algoritmo de *clustering* apresenta-se, então, como solução promissora a esta questão, permitindo tomar os centroides resultantes como os novos centros da *hidden layer*. Estes substituirão os n centros previamente considerados, proporcionando uma redução significativa do número de unidades nesta camada e uma representação mais compacta dos dados.

Assim sendo, foi efetuado *EM clustering*, cuja fórmula indica a seguinte função gaussiana:

$$\phi(x) = exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

Note-se que, uma vez que o mapeamento dos dados envolve a utilização da fórmula da *likelihood* ao realizar o *soft assignment* dos pontos aos *clusters,* e esta é igual à fórmula da função gaussiana para a *RBF* com *EM clustering,* com a exceção do denominador, cujo objetivo é garantir que a integral da função densidade de probabilidade equivale a um. Como tal, os valores obtidos anteriormente das probabilidades de pertencer a cada *cluster* foram utilizados como os resultados da aplicação da função *RBF*.

Tendo em conta o número ótimo de *clusters* obtido, os dados são representados por dois centros, o que resulta na seguinte arquitetura da *network*: m-2-1. Após o treino, a *output layer* desempenha o papel de classificador linear através da *Logistic Regression*, que beneficia da separação realizada pela *hidden layer*.

Por forma a averiguar o impacto desta etapa de pré-processamento, foi comparado o desempenho de uma *Radial Basis Function Network (RBFN)* com e sem a utilização de *clustering*, cujos resultados se encontram visíveis no gráfico da Figura 4. Na abordagem sem *clustering,* foi obtida uma exatidão de 99,246% no conjunto de treino e 67,251% no conjunto

de teste, sendo esta discrepância indicativa de *overfitting*, uma vez que o modelo tem um bom desempenho relativamente aos dados de treino mas para os dados de teste tem um desempenho indesejável. Com a aplicação de clustering, os valores alteraram para 87.688% no conjunto de treino e 92.398% no conjunto de teste. Estes resultados confirmam como o pré-processamento dos dados, através da seleção otimizada de centros via *clustering*, contribui drasticamente para a melhoria da capacidade de generalização do modelo.

Em suma, a utilização de funções de base radial *(RBFs)* permite transformar problemas não lineares em espaços onde as classes se tornam linearmente separáveis. Isto ocorre porque as *RBFs* projetam os dados para um espaço de alta dimensionalidade, onde pontos que são não linearmente separáveis no espaço original podem ser separados por hiperplanos simples no espaço transformado. Para além disso, usar os centroides como *hidden units* permite que a *network* se concentre em padrões globais em vez de detalhes específicos de cada amostra e simultaneamente reduz o número de unidades na *hidden layer*. Assim, o pré-processamento com clustering melhora não só a generalização do modelo, mitigando o risco de *overfitting* e atenuando os efeitos da "maldição da dimensionalidade", como torna o treino computacionalmente mais eficiente.

ANEXO:

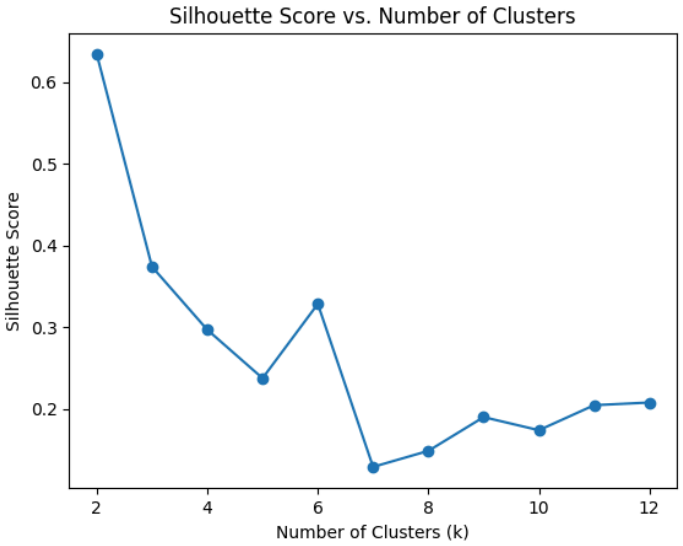

Figura 1: Silhouette Scores vs. Number of Clusters

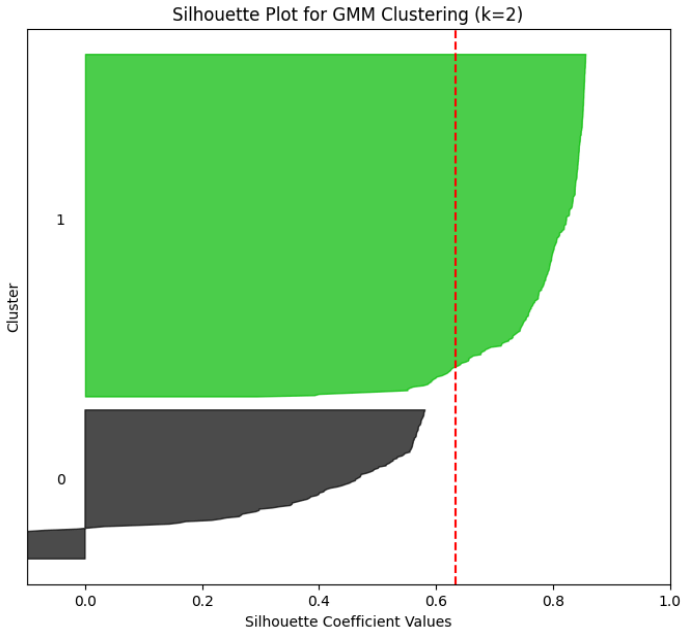

Figura 2: Silhouette Plot for GMM Clustering (k=2)

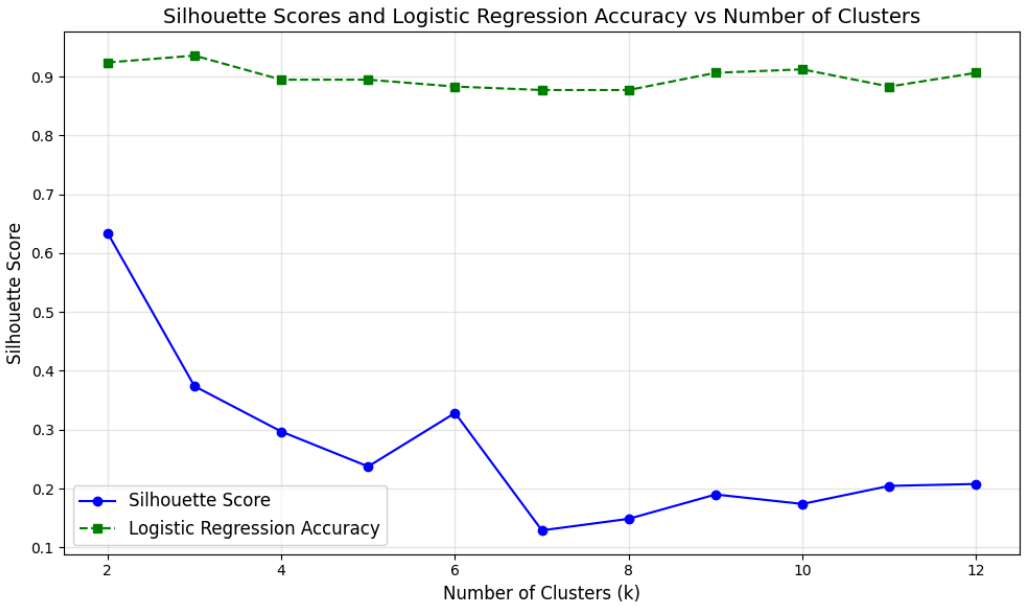

Figura 3: Silhouette Scores and Logistic Regression Accuracy vs Number of Clusters

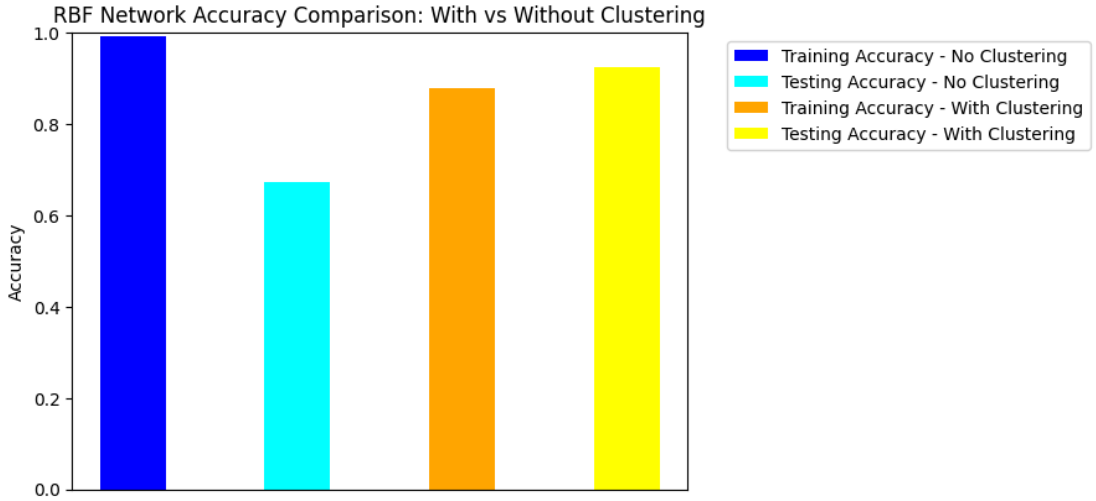

Figura 4: RBF Network Accuracy Comparison: With vs Without Clustering