**COORDENADORIA DE AUTOMAÇÃO INDUSTRIAL**

**Disciplina: Reconhecimento de Padrões**

**Período: 2024-1**

**Professor: Daniel Cruz Cavalieri**

EXERCÍCIO 02: SVM, kNN, DECISION TREE e CNN

**Aluno: Marjorie Ariele Pereira**

**Turma: Mestrado Data:** 01/07/ 2024

**Questões relacionadas ao conteúdo da disciplina:**

**Questão 1**. Suponha que você esteja trabalhando na previsão do tempo e use um algoritmo de aprendizado para prever a temperatura de amanhã (em graus Centígrados/Fahrenheit). Você trataria isso como uma classificação ou um problema de regressão?

Para prever a temperatura usa-se um problema de **regressão**. A razão é que a temperatura é uma variável contínua, e a regressão tem o objetivo de prever valores numéricos contínuos.

**Questão 2**. Alguns dos problemas abaixo são mais bem resolvidos usando um algoritmo de aprendizagem supervisionado, e os outros com um algoritmo não supervisionado. Qual dos seguintes você aplicaria aprendizagem **supervisionada**? (Selecione todas que se apliquem.) Em cada caso, suponha que um conjunto de dados adequado está disponível para o seu algoritmo aprender.

1. Examine uma página da Web e classifique se o conteúdo da página deve ser considerado "amigável para crianças" ou "adulto".

Aprendizagem supervisionada : Problema de classificação binária, onde temos dois rótulos (amigável para crianças e adulto), dados rotulados.

1. Dada uma base de dados sobre como 1.000 pacientes médicos respondem a um medicamento experimental (como eficácia do tratamento, efeitos colaterais, etc.), descubra se existem diferentes categorias ou "tipos" de pacientes em termos de como eles respondem ao medicamento e em caso afirmativo, quais são essas categorias.

Não supervisionada**:** Este é um problema de clustering, onde você está tentando descobrir padrões ou grupos dentro dos dados sem rótulos predefinidos.

1. Na agricultura, com dados sobre o rendimento das colheitas nos últimos 50 anos, aprenda a prever os rendimentos das colheitas do próximo ano.

Supervisionada**:** Este é um problema de regressão, onde você está tentando prever um valor contínuo (rendimento das colheitas) com base em dados históricos.

1. Dado um grande conjunto de dados de registros médicos de pacientes que sofrem de doenças cardíacas, tente descobrir se pode haver diferentes grupos desses pacientes para os quais podemos adaptar tratamentos separados.

**Não supervisionada: Este é outro problema de clustering, onde você quer encontrar diferentes grupos dentro dos dados sem rótulos predefinidos.**

**Questão 3.** Supondo que haja uma base de dados suficientemente grande para o treinamento de algum algoritmo. Assim, treinar utilizando uma grande quantidade de dados normalmente resulta em uma boa performance. Marque verdadeiro ou falso nas afirmações e justifique sua resposta:

1. As características contêm informação suficiente para prever a saída corretamente.

**Verdadeiro, com uma base de dados grande, e com o treinamento bem feito, é possível prever a saída corretamente**

1. Não é necessário utilizar regularização.

**Falso. A regularização ainda é necessária para evitar overfitting e garantir que o modelo generalize bem para novos dados.**

**Questão 4.** Imagine que estamos trabalhando em um classificador de *spam*, onde *e-mails* de *spam* são exemplos positivos (saída igual a 1) e *e-mails* não *spam* são exemplos negativos (saída igual a zero). Nós temos uma base de dados de treinamento onde 99% dos *e-mails* não são *spam* e os outros 1% que são. Quais das afirmativas abaixo são verdadeiras?

1. Se predizermos tudo como não *spam*, a acurácia do classificador será 99%

**verdadeira**

1. Se predizermos tudo como não *spam*, nosso classificador terá uma sensibilidade (*recall*) igual a 0%.

**verdadeira**

1. Se predizermos tudo como *spam*, o classificador terá uma sensibilidade de 100% e uma precisão de 1%.

**verdadeira**

1. Se predizermos tudo como *spam*, o classificador terá uma sensibilidade de 0% e uma precisão de 99%.

**Falso**

**Questão 5.** Marque verdadeiro ou falso nas afirmativas abaixo.

1. Utilizar uma base de dados grande para o treinamento pode evitar o *overfitting* do classificador.

**Falso. Mesmo com uma base de dados grande normalizar e regularizar esses dados é importante**

**generalização.**

1. Se ocorre o *underfitting* do algoritmo na base de treinamento, obter mais dados pode ajudar.

**Falso.Para resolver o underfitting, é necessário aumentar a complexidade do modelo, melhorar as características ou ajustar os hiperparâmetros.**

1. O processo de análise manual dos exemplos que o algoritmo errou pode ajudar a sugerir os próximos passos para melhorar a performance do algoritmo (por exemplo, inserir novas características).

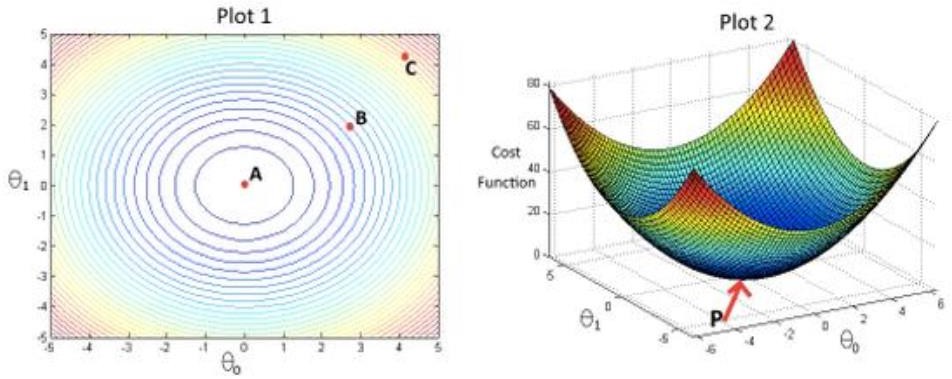
**Verdadeiro**

1. É uma boa ideia gastar bastante tempo na construção de uma base de dados suficientemente grande, antes de construir a primeira versão do algoritmo de aprendizado de máquinas.

Verdadeiro, afinal é fundamental o tratamento dos dados em qualquer problema de machine learning.

**Questão 6.** A figura 1 apresenta a função de custo 𝐽(𝜃0, 𝜃1) (“Plot 2”), assim como o contorno da mesma função (“Plot 1”). Baseado nas figuras, verifique quais das afirmativas abaixo são verdadeiras.

Figura 1 – Plotes da função de custo 𝐽(𝜃0, 𝜃1)



1. Se partimos do ponto B, um algoritmo de gradiente descendente com um valor adequado de taxa de aprendizado ajudará a atingirmos o ponto A, onde a função de custo 𝐽(𝜃0, 𝜃1) possui seu valor máximo.

Resposta: Falso. O Ponto A é o valor mínimo.

1. Se partimos do ponto B, um algoritmo de gradiente descendente com um valor adequado de taxa de aprendizado ajudará a atingirmos o ponto A, onde a função de custo 𝐽(𝜃0, 𝜃1) possui seu valor mínimo.

**Verdadeiro.**

1. O ponto P (mínimo global no Plot 2) corresponde ao ponto A no Plot 1.

**Verdadeiro pois ambas indicam o valor mínimo da função de custo**

1. Se partimos do ponto B, um algoritmo de gradiente descendente com um valor adequado de taxa de aprendizado ajudará a atingirmos o ponto C, onde a função de custo 𝐽(𝜃0, 𝜃1) possui seu valor mínimo.

**Falso. O ponto C é o valor máximo e além disso o algoritmo de gradiente descendente ajudará a alcançar o valor de mínimo erro**

1. O ponto P (mínimo global no Plot 2) corresponde ao ponto C no Plot 1.

**Falso. O ponto P no plot 2 corresponde ao ponto A no plot 1**

**Questão 7.** Você roda um algoritmo de gradiente descendente por 15 iterações com uma taxa de aprendizado 𝛼 = 0.3 e computa a perda 𝐽(𝜃) após cada iteração. Você observa que o valor de 𝐽(𝜃) aumenta ao longo do tempo. Baseado nisso, qual das sentenças abaixo parece mais plausível.

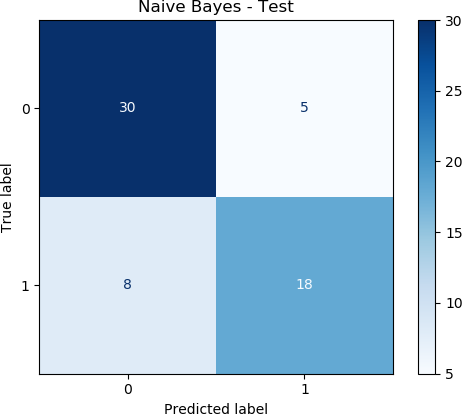
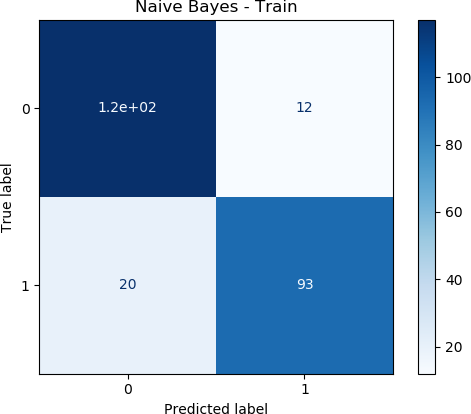
1. 𝛼 = 0.3 é um valor efetivo de taxa de aprendizado.
2. Ao invés de se utilizar o valor corrente de 𝛼, é mais interessante aumentar seu valor (por exemplo, 𝛼 = 1.0).
3. Ao invés de se utilizar o valor corrente de 𝛼, é mais interessante diminuir seu valor (por exemplo, 𝛼 = 0.1).

**Item 3, pois com a taxa de aprendizado menor haverá um menor acúmulo de perdas ao longo das iterações.**

**Código-exemplo para Classificação**

Assim como na lista de exercícios 1, será utilizado o conjunto de dados c*leveland.csv*, onde foi desenvolvido o código *código-exemplo-pca.py* para prever se uma pessoa terá ou não doença do coração. Neste código foi utilizado o algoritmo Naive Bayes sendo realizado um pré- processamento de dados (uma vez que existem dados faltantes). Com os dados carregados e corrigidos, eles foram separados em treinamento (80%) e teste (20%). O resultado do treinamento e teste são apresentados nas matrizes de confusão da figura 2a e 2b, respectivamente.

Figura 2. Matriz de confusão geradas para o treinamento (a) e para o teste (b) do classificador Naive Bayes.

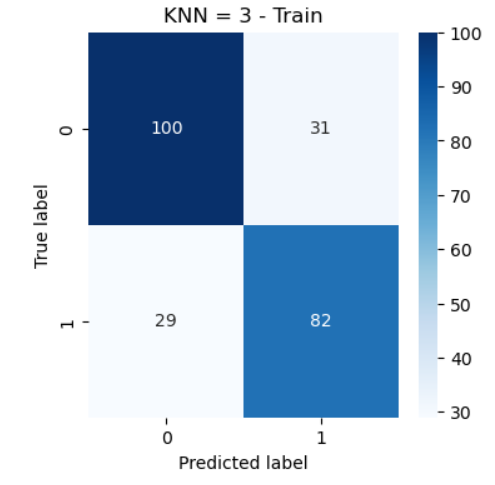
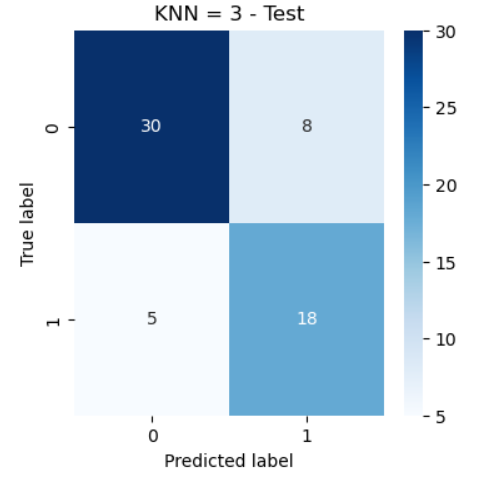


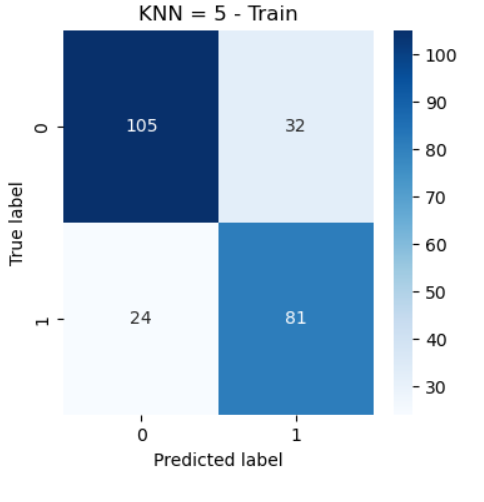
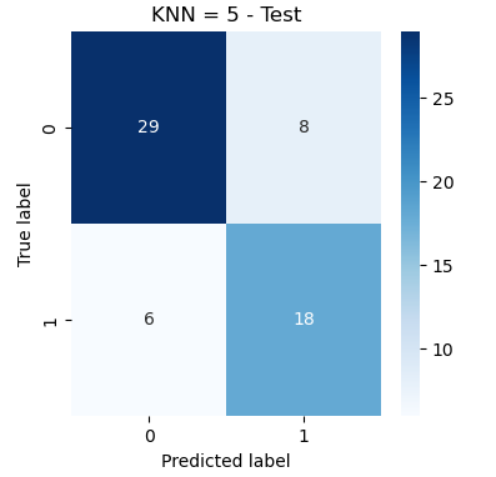
(a) (b)

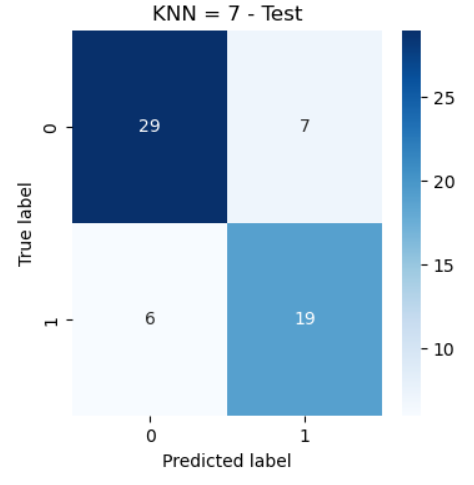
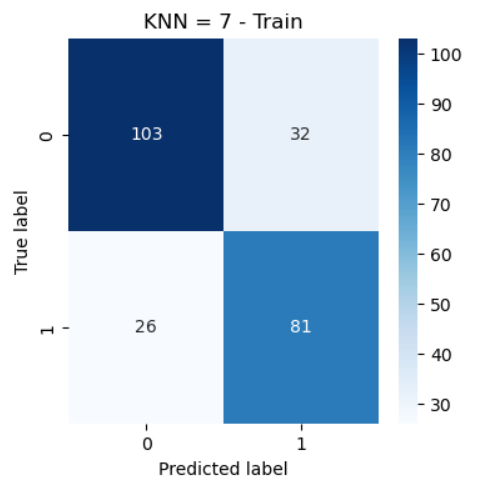
1. 𝒌**NN**

Com o pré-processamento realizado, adapte o código-exemplo inserindo agora o algoritmo

𝑘NN e gere os resultados (matrizes de confusão) considerando 𝑘 = 3, 5 e 7. Utilize a distância euclidiana como métrica de distância.





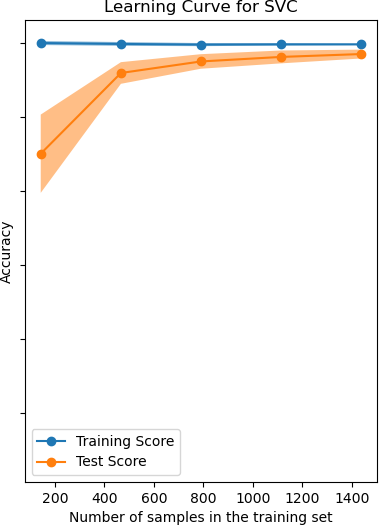
1. **Decision Tree**

Adapte novamente o código-exemplo inserindo agora o algoritmo para a criação de uma árvore de decisão. Apresente os resultados do treinamento e teste seguindo o exemplo dado na figura 2.

1. ***Support Vector Machines***

Adapte o código-exemplo e insira o algoritmo de máquina de vetores suporte. Considere, neste caso os kernels linear e rbf e apresente os resultados de treinamento e teste como na figura 2. Com a matriz de entrada reduzida pelo PCA (somente os dois primeiros *scores*), plote a curva de aprendizado do algoritmo treinado, como aparece na figura 3 ([https://scikit-](https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html) [learn.org/stable/auto\_examples/model\_selection/plot\_learning\_curve.html](https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html)). Neste caso utilize somente os dados de treinamento e o algoritmo com *kernel* RBF.

Figura 3. Exemplo de curva de aprendizado de um algoritmo do tipo SVM.



1. **Comparação de resultados**

A partir das matrizes de confusão de teste de cada algoritmo (Naive Bayes, kNN e Decision Tree), construa uma tabela apresentando os resultados de acurácia de cada um e responda:

Tabela com resultados de acurácia dos modelos:

|  |  |  |
| --- | --- | --- |
| Modelo | Acuracia Train | Acuracia Test |
| Naive Bayes | 0.8677685950413223 | 0.7868852459016393 |
| KNN K=3 | 0.7520661157024794 | 0.7868852459016393 |
| KNN K=5 | 0.768595041322314 | 0.7868852459016393 |
| KNN K=7 | 0.7603305785123967 | 0.7868852459016393 |
| D. Tree | 1.0 | 0.7540983606557377 |

* + Neste problema, a acurácia é uma boa métrica para comparar cada algoritmo? Se não, qual medida poderia ser utilizada?

A acurácia foi similar para todos algoritmos, exceto o Decision Tree, devido a isso não vejo como uma boa métrica de comparação dos modelos

* + Repita os resultados de cada algoritmo, porém, utilizando validação cruzada (𝑐𝑜𝑚 𝑘 = 10). Discuta os resultados comparando-os com os resultados utilizando a partição 80- 20.

|  |  |  |
| --- | --- | --- |
| Cross Validation | | |
| Modelo | Média de acurácia | Desvio padrão |
| Naive Bayes | 0.84 | 0.06 |
| KNN | 0.64 | 0.08 |
| DT | 0.73 | 0.09 |

A validação cruzada obteve melhor resultado aplicando no modelo naive bayes

1. **CNN**

Fashion-MNIST é um conjunto de dados de imagens de artigos de vestuário da empresa Zalando

- consistindo em um conjunto de treinamento de 60.000 exemplos e um conjunto de teste de

10.000 exemplos. Cada exemplo é uma imagem em tons de cinza de 28𝑥28, associada a um rótulo de 10 classes:

* T-shirt/top
* Trouser
* Pullover
* Dress
* Coat
* Sandal
* Shirt
* Sneaker
* Bag
* Ankle boot

O Fashion-MNIST é utilizado como um substituto direto para o conjunto de dados MNIST original para benchmarking de algoritmos de aprendizado de máquina. Ele compartilha o mesmo tamanho de imagem e estrutura de divisões de treinamento e teste. Um tutorial oficial utilizando o Keras e o Fashion-MNIST pode ser encontrado no seguinte link: <https://www.tensorflow.org/tutorials/keras/classification>

Dado esse conjunto de imagens, projete uma rede neural convolucional para classificação dos objetos. O modelo deverá ser desenhado do zero seguindo o tutorial e deverá ser constituído de camadas convolutivas, camadas de *pooling*, diferentes funções de ativação e uma *fully- connected* layer na camada de saída. Escolha a função custo *cross entropy* e avalie a acurácia na base de teste. Disponibilize o código-fonte gerado no Github e apresente o *link*.

Relu, Selu e swish, obteve-se uma acurácia de 0.880299985408783.

Substituindo a camada swish pela softplus, a acuracia foi 0.8803 e perda de: 0.5236

CODIGOS: [marjorieariele/Reconhecimento-de-padroes (github.com)](https://github.com/marjorieariele/Reconhecimento-de-padroes)