

# Data exploration and enrichment for supervised classification

Alexandre Furriel 202307781

Eduarda Neves 202307178

Leonor Carvalho 202306674

# Specification of the work to be performed

Neste trabalho, o nosso objetivo é determinar a sobrevivência de pacientes um ano após o diagnóstico da doença HCC. Para tal seguiremos os seguintes passos:

- Análise e pré-processamento dos dados clínicos fornecidos: identificação da relevância dos dados, eliminação de parâmetros irrelevantes, transformação dos valores em falta, variáveis numéricas/qualitativas e possivelmente novas variáveis.
- Supervised machine learning/Comparação de resultados: divisão dos dados em “data and training sets”, “Decision Trees e KNN algorithms” para desenvolver “classification models”, avaliação do desempenho através da acurácia, precisão, ROC, “recall” e “confusion matrix”, comparação dos resultados obtidos em diferentes testes, utilizando gráficos.

# Tools

- Pandas
- Scikit Learn
- Numpy
- Seaborn
- Matplotlib libraries

# Algorithms

- Decision Trees
- KNN algorithms
- Neural Networks

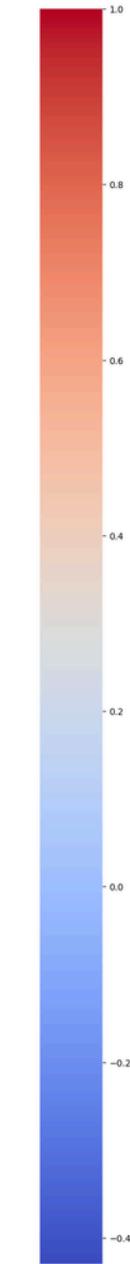
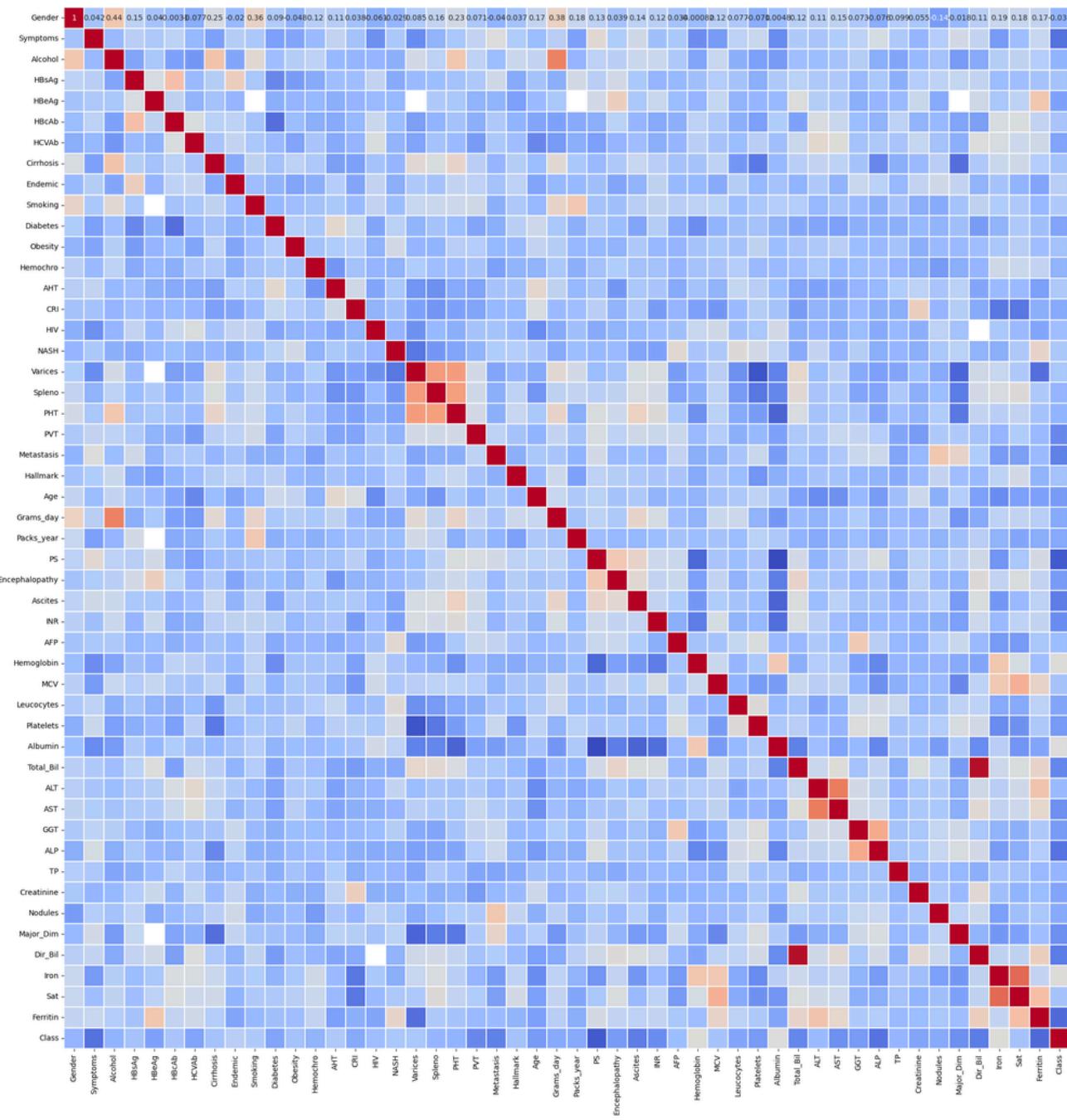
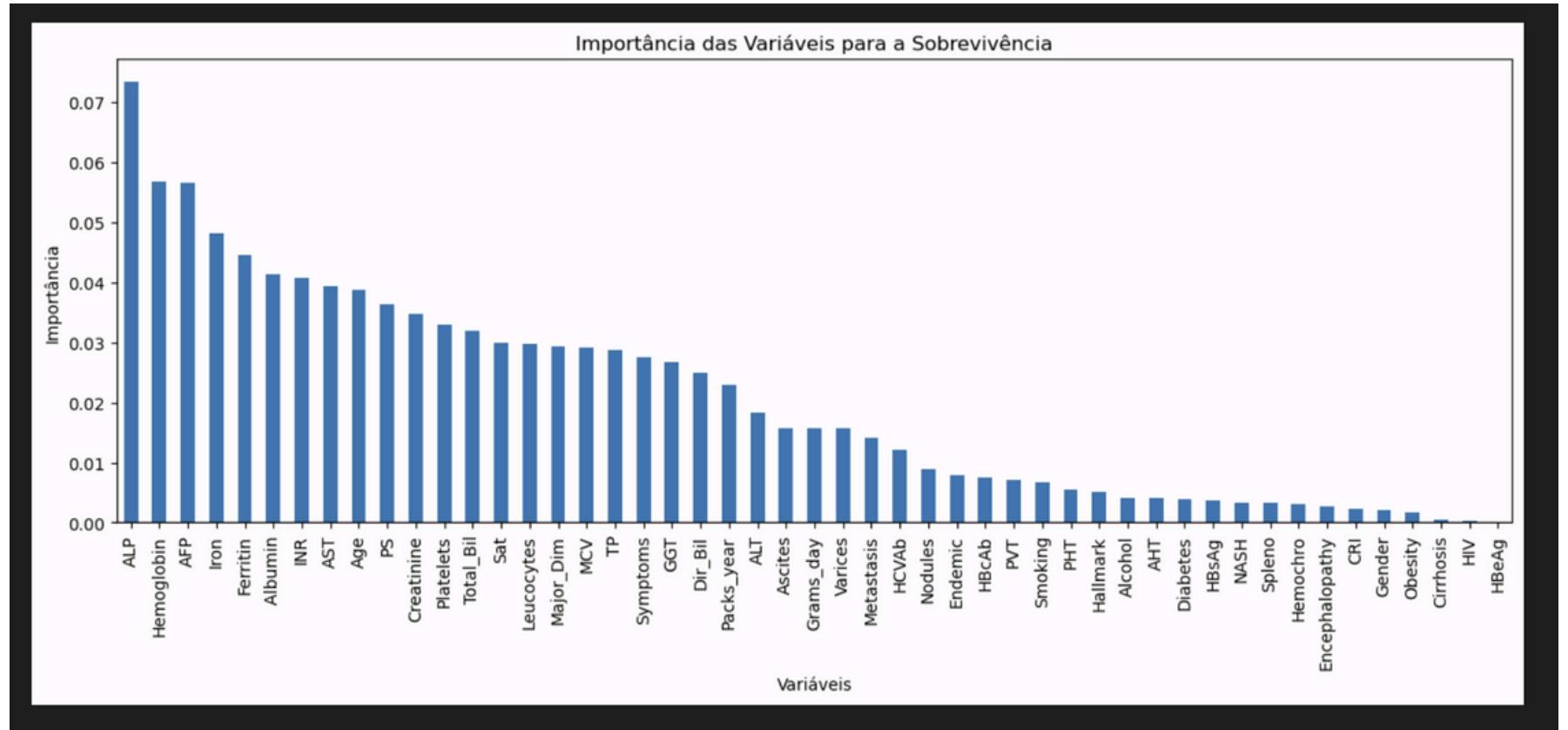
# Referências para a interpretação dos dados

- Nestas tabelas encontram-se os ranges de valores para estas condições médicas com valores numéricos assim como a média obtida através da dataset.
- Número de pacientes: 165

	Range	Média Obtida
Age	20-93 anos	74 anos
Grams-day	0-500 g/d	~70.1 g/d
Packs-year	0-510 packs	19 packs
INR	0.84-4.82	~1.62
AFP	1.2-1810346.0	99883.8
Hemoglobin	5.0-18.7	~9.5
MCV	69.5-119.6	~85.1
Leucocytes	2.2-44.6	~4702.9
Platelets	1.71-459000.00	142993.67
Albumin	1.9-4.9	~3.1
Total_Bil	0.3-40.5	~2.9
ALT	11-420	35

	Range	Média Obtida
AST	17-553	50
GGT	23-1575	~166
ALP	1.28-980.00	~137.51
TP	3.9-102.0	~7.2
Creatinine	0.2-6.1	0.7
Nodules	0-5	~4
Major_Dim	1.5-22.0	~7.0
Dir_Bil	0.1-29.3	~1.3
Iron	0-224	~106
Sat	0-126	~38
Ferritin	0-2230	~795

# Data Preprocessing



	Python																
	Gender	Symptoms	Alcohol	HBsAg	HBeAg	HBCAb	HCVAb	Cirrhosis	Endemic	Smoking	...	ALP	TP	Creatinine	Nodules	Major_Dim	
count	165.000000	165.000000	165.000000	165.000000	165.000000	165.000000	165.000000	165.000000	165.000000	165.000000	...	165.000000	165.000000	165.000000	165.000000	165.000000	16
unique	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN									
top	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN									
freq	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN									
mean	0.806061	0.678788	0.739394	0.096970	0.006061	0.230303	0.206061	0.903030	0.060606	0.630303	...	212.211605	8.961039	1.127089	2.736196	6.851172	
std	0.396586	0.468364	0.440302	0.296817	0.077850	0.422308	0.405706	0.296817	0.239333	0.484192	...	166.400389	11.328570	0.935235	1.786904	4.774271	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	1.280000	3.900000	0.200000	0.000000	1.500000	
25%	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	...	109.000000	6.400000	0.710000	1.000000	3.000000	
50%	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	1.000000	...	163.000000	7.100000	0.860000	2.000000	6.000000	
75%	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	1.000000	...	260.000000	7.700000	1.127089	5.000000	8.500000	
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	...	980.000000	102.000000	7.600000	5.000000	22.000000	2

- Agrupamento dos dados, distinção da informação importante.
- Substituição dos valores em falta.
- Substituição de valores categóricos por numéricos.
- Uso de vários tipos de gráficos para retirar conclusões sobre os dados.

# Machine Learning

- Na implementação de Machine Learning neste projeto, optámos por usar algoritmos como KNN, Decision tree e Neural Networks.
- Usámos métodos de partição de dados entre treinar o modelo 70% dos dados e testar com 30%. O Leave One Out e o Cross Validation também tiveram a mesma função sendo que o primeiro faz os testes com cada exemplo individual demorando mais tempo e o segundo divide o conjunto de dados em k partes. Se todos os dados previstos coincidissem com os reais a accuracy seria de 1, se nenhum coincidisse seria 0.

## KNN

```
# Convertendo para NumPy Arrays explicitamente
X_train = np.array(X_train)
X_test = np.array(X_test)
y_train = np.array(y_train)
y_test = np.array(y_test)

# Cria o modelo KNN
knn = KNeighborsClassifier(n_neighbors=1) # Exemplo com k=5, você pode ajustar conforme necessário

# Medição do tempo
start_time = time.time()

# Treina o modelo
knn.fit(X_train, y_train)

# Faz previsões
y_pred = knn.predict(X_test)

# Medição do tempo
end_time = time.time()
execution_time = end_time - start_time

# Calcula a precisão
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

# Imprime o relatório de classificação
print(classification_report(y_test, y_pred))
print('Tempo que demorou', execution_time)
```

Python

## Decision Tree

```
# Cria o modelo Decision Tree
tree=DecisionTreeClassifier(criterion='gini', max_depth=10, min_samples_leaf=10, min_samples_split=10, random_state=42)

# Medição do tempo
start_time = time.time()

# Treina o modelo
tree.fit(X_train, y_train)

# Testing the classifier, faz as previsões
y_pred = tree.predict(X_test)

# Medição do tempo
end_time = time.time()
execution_time = end_time - start_time

accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')

print(classification_report(y_test, y_pred))
print('Tempo que demorou', execution_time)
```

Python

## Neural Networks

```
# Normalizar os dados
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Treinar o modelo de rede neural
mlp = MLPClassifier(hidden_layer_sizes=(100,), max_iter=300, random_state=42)
mlp.fit(X_train, y_train)

# Fazer previsões
y_pred = mlp.predict(X_test)

media_nn_recall= recall_score(y_test, y_pred, average='weighted')
media_nn_f1 = f1_score(y_test, y_pred, average='weighted')
media_nn_precision= precision_score(y_test, y_pred, average='weighted')

#Acurácia
media_accuracy_nn = accuracy_score(y_test, y_pred)
# Gerar a matriz de confusão
media_conf_matrix_nn = confusion_matrix(y_test, y_pred)
# Fazer previsões de probabilidade no conjunto de teste
y_prob = mlp.predict_proba(X_test)[:, 1]
# Calcular a curva ROC
media_fpr_nn, media_tpr_nn, thresholds = roc_curve(y_test, y_prob)
# Calcular a AUC
media_roc_auc_nn = auc(media_fpr_nn, media_tpr_nn)
report_nn = classification_report(y_test, y_pred)
print(report_nn)
```

Python

# Testes Realizados:

## Teste 1:

- O primeiro teste a ser realizado foi usar o dataset completo, substituindo apenas os valores em falta com o uso da média e da moda e do KNN. Algoritmos: Decision Tree, KNN e Neural Networks.
- Comparando os dois métodos concluímos que para uma amostra de grandes dimensões como esta (165x50) a substituição pelo knn era a melhor opção.

## Teste 2:

- Neste teste foram retiradas todas as categorias com valores em falta.
- A performance do Neural Networks destaca-se pela positiva, ainda que em pequena escala, em relação aos restantes métodos.
- Não se verifica novamente uma grande discrepância de resultados.

## Teste 3:

- Neste teste foram retiradas as categorias cuja relevância fosse menor que 0.01, assim como 3 categorias, que apesar da sua elevada relevância, de um ponto, não serão relevantes para o prognóstico dado o valor de correlação.

- Uso dos métodos de partição Leave One Out e Cross Validation
  - Leave One out não é dos melhores métodos a utilizar num dataset grande dado ao peso e tempo de duração.

- A alteração do data set em estudo não provocou grandes diferenças.

## Teste 4:

- Neste teste mantivemos apenas as colunas com valores numéricos, reduzindo a informação em cerca de 50% (26 colunas).
- Neural Networks apresenta uma elevada accuracy.

# Comparação de Modelos

-Ao analisarmos os gráficos podemos concluir que o Neural Networks é o modelo mais eficaz para os testes realizados. Este modelo destacou-se pela maior pontuação em cada um dos parâmetros de avaliação usados, como podemos ver na tabela, em proporção com o menor valor de Falsos Positivos. Este parâmetro é tido em conta cautelosamente dada a sua significância em situações médicas.

Accuracy	Decision Tree	KNN	Neural Networks	Cross Validation/Leave One Out
Teste 1	64% / 60%	62% / 60%	68% / 68%	-
Teste 2	64%	62%	72%	-
Teste 3	56%	60%	66%	52% - 58% - 65% / 64% - 56% - 65%
Teste 4	100%	62%	94%	100% - 58% - 62%

# Webgrafia

<https://www.geeksforgeeks.org/introduction-to-data-processing/>

[https://www.youtube.com/watch?v=bDhvCp3\\_IYw](https://www.youtube.com/watch?v=bDhvCp3_IYw)

<https://www.youtube.com/watch?v=OY4eQrekQvs>

<https://towardsdatascience.com/a-comprehensive-guide-to-a-classification-project-data-cleaning-and-exploration-88edd5617ce2>

## Github:

<https://github.com/FreddieAlvin/Artificial-Intelligence-in-Data-Processing.git>