**Relatório <010> - < 10 - Prática: Lidando com Dados do Mundo Real (II) >**

<Kawan Machado>

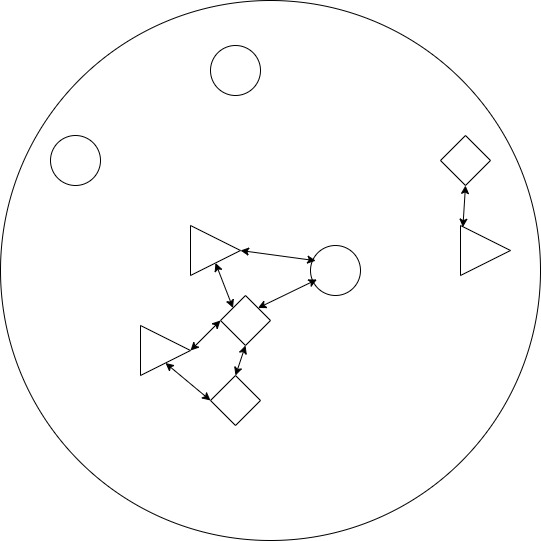
# Descrição da atividade

## More Data Mining and Machine Learning Techniques

### K-Nearest-Neighbors Concepts

Iniciando por esse conceito, fica evidente como as técnicas de aprendizado de máquina podem ser tão simples. A ideia é classificar os pontos dos dados baseado na distância entre si. Para cada nova entrada, o algoritmo classifica a distância dos vizinhos mais próximos e agrupa os baseados em suas distâncias.

FIGURA 1 - K-NEAREST-NEIGHBORS

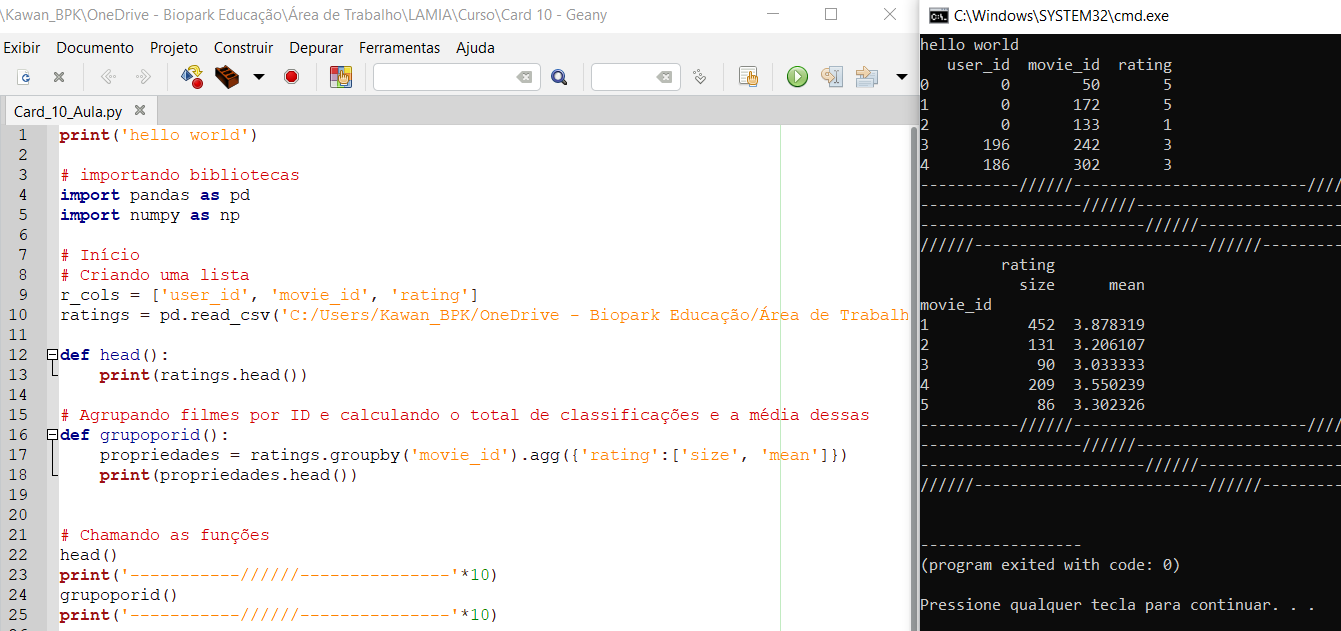
Elaborado pelo próprio autor, 2025.

O KNN é um dos modelos mais simples de aprendizado de máquina, ele é classificado como aprendizado supervisionado. No próximo tópico vamos explorar a classificação de filmes baseado nos seus metadados.

### Activity Using KNN to predict a rating for a movie

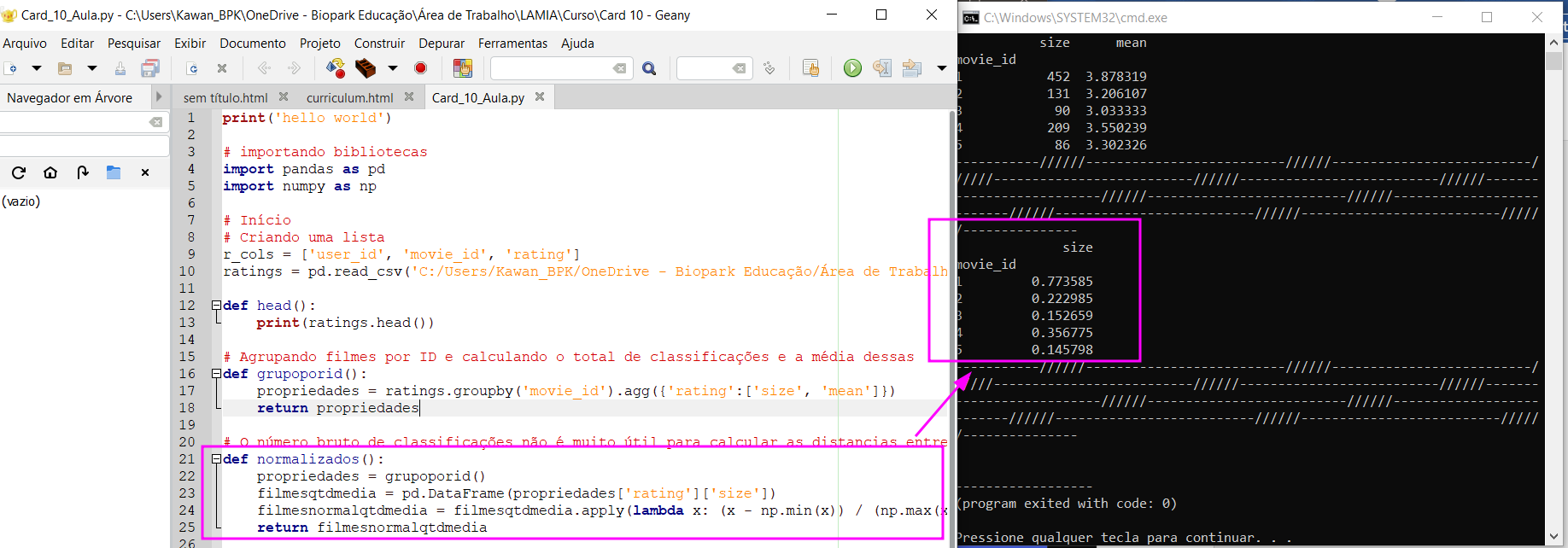
Para esse exemplo vamos tratar o data set ‘MovieLens’, com ele vamos classificar os filmes olhando os 10 mais próximos em termos de gênero e popularidade e com isso vamos conseguir predizer a avaliação média esperada de um filme do mesmo gênero. Primeiro fazemos a importação dos dados e depois calculamos a quantidade e a média de classificação dos filmes.

FIGURA 2 – IMPORTAÇÃO E PRIMEIRO CÁLCULO

Elaborado pelo próprio autor, 2025.

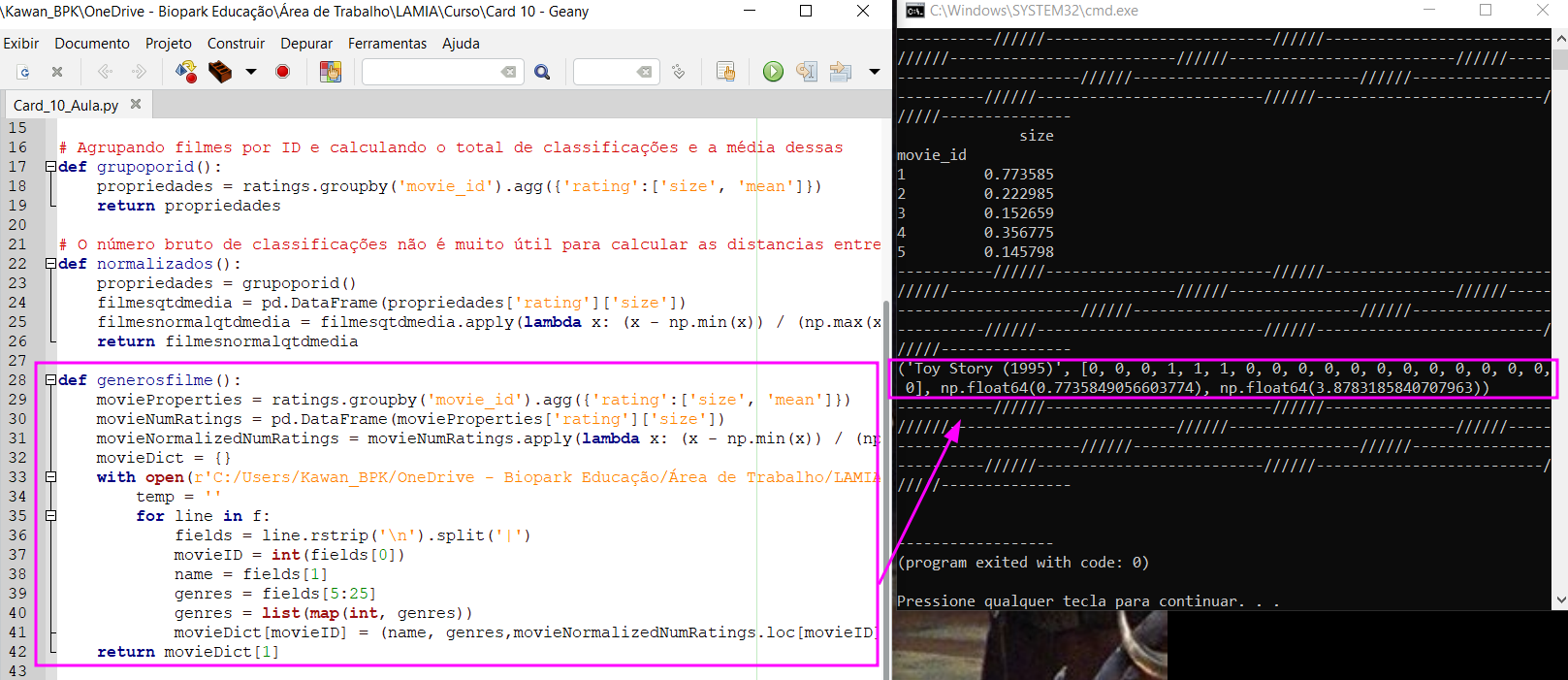
A média bruta de avaliações dos filmes pode conter problemas, pois alguns filmes tem mais avaliações que outros. Para isso efetuamos a normalização dos valores para eles ficarem entre 0 e 1, com isso conseguimos prosseguir a análise.

FIGURA 3 – NORMALIZAÇÃO DOS DADOS

Elaborado pelo próprio autor, 2025.

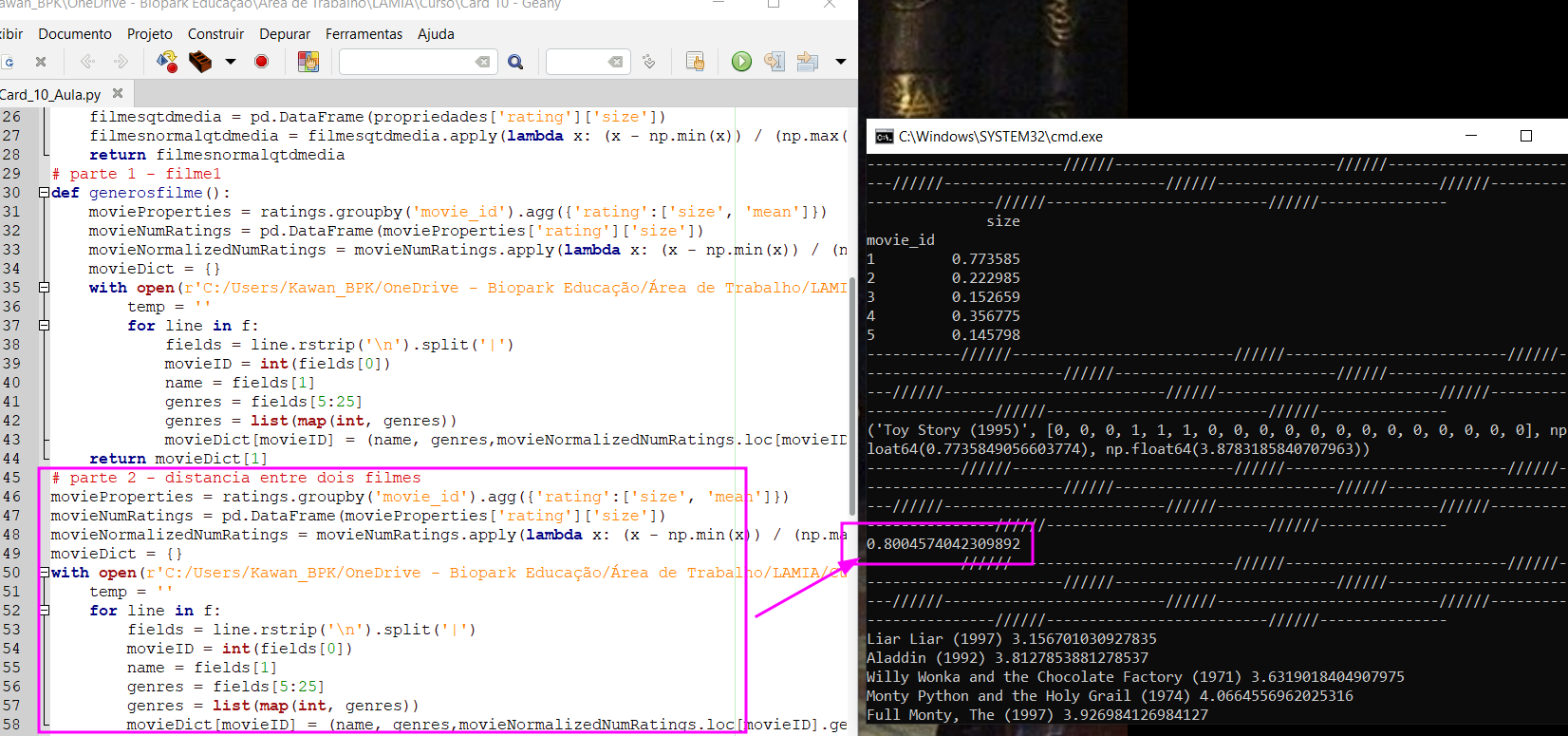
Agora precisamos importar a informação dos gêneros dos filmes que estão contidos no arquivo “u.file” e mesclar com os valores das médias de avaliação e com os valores normalizados das mesmas. Para exemplificar trouxemos os valores do item 1, que corresponde ao filme ‘Toy Story’ que se encaixa em três genêros, tem uma média de avaliações normalizada de 0,77 e uma média bruta de 3,87.

FIGURA 4 – DADOS DO FILME 1

Elaborado pelo próprio autor, 2025.

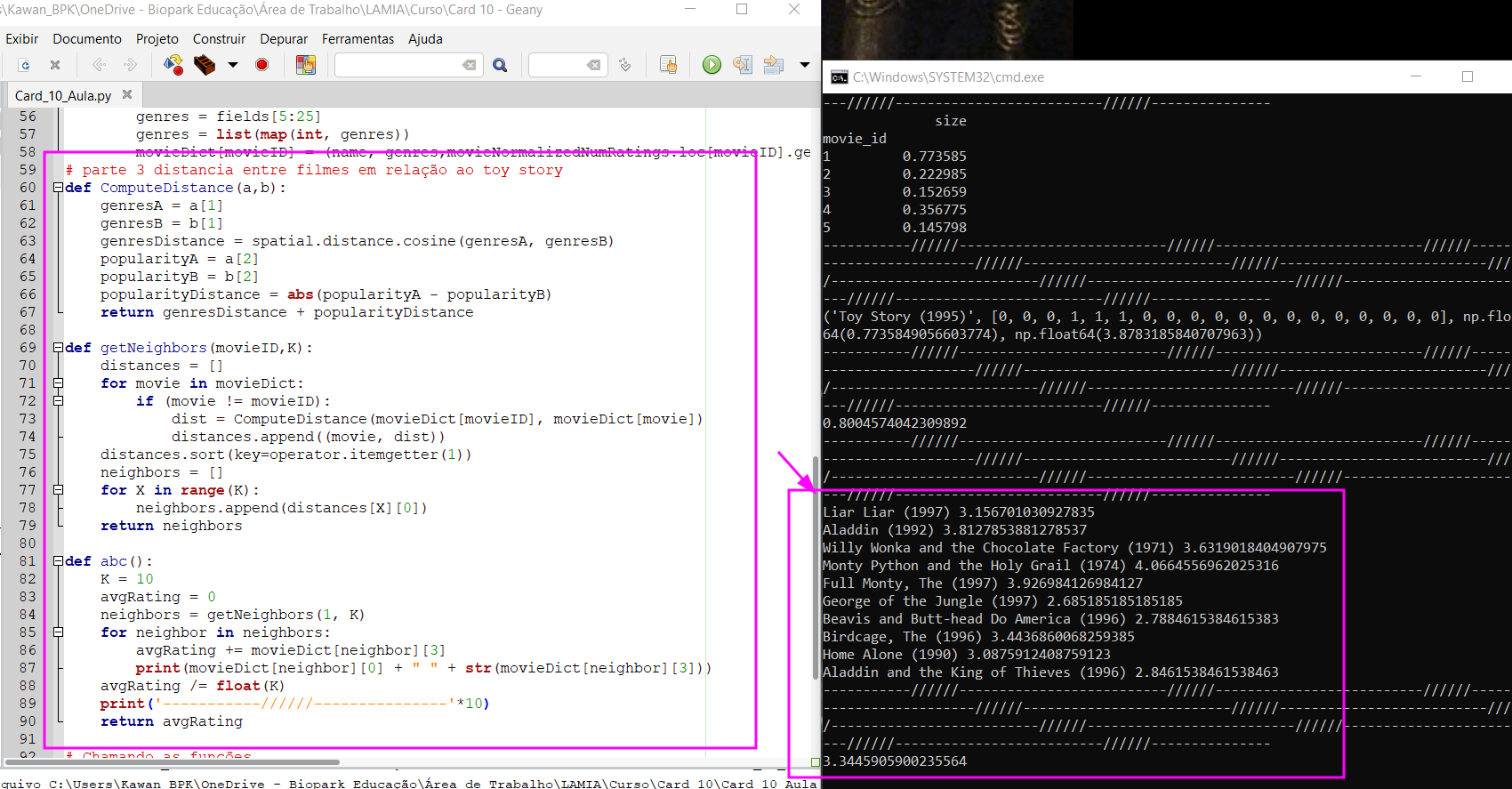
Agora vamos computar a distância entre dois filmes, no caso o filme 2 e o filme 4. O resultado é um valor de 0,80, lembrando que quanto maior menos similar um filme é de outro, nesse caso, 0,80 significa que os filmes não são parecidos.

FIGURA 5 – DISTÂNCIA ENTRE FILME 2 E 4

Elaborado pelo próprio autor, 2025.

Por fim, vamos usar o filme 1 (‘toy sotry’) para agrupar todos os filmes do mesmo gênero e calcular o valor médio de avaliação destes e com isso predizer a média de avaliação esperada para um filme do mesmo grupo.

FIGURA 6 – MÉDIA DE AVALIAÇÃO

 Elaborado pelo próprio autor, 2025.

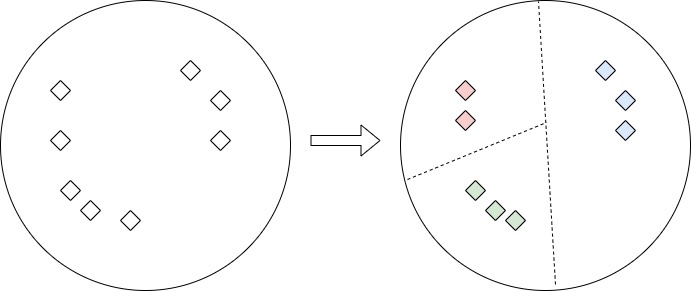
Observando a imagem acima, primeiro temos os filmes que se encaixaram no mesmo grupo do filme ‘toy story’ e abaixo temos a média de avaliações ‘3,34’, com isso podemos dizer que a avaliação média esperada para um filme que se encaixa no mesmo padrão de gênero de ‘toy story’ é de 3,34.

### Dimensionality Reduction; Principal Component Analysis (PCA)

Explorando os temos de dimensionalidades, o primeiro item que chama a atenção é a maldição da dimensionalidade, ou seja, um cálculo matemático que pode gerar infinitas dimensões, por exemplo, em um sistema de recomendações, o vetor de avaliações para cada filme pode representar uma dimensão, ou seja, cada filme possuiria sua própria dimensão, o que pode comprometer recursos computacionais.

Para tratar dessa situação, é feita a redução da dimensionalidade, um dado em grandes dimensões reduzido a uma dimensão menor preservando o máximo de variância que for possível. Uma forma de fazer isso é usando K-Means, se cada filme é a sua própria dimensão, agrupar eles reduzem a dimensão individual para uma dimensão de grupos.

FIGURA 7 – K-MEANS REDUÇÃO DE DIMENSIONALIDADES

 Elaborado pelo próprio autor, 2025.

Outra forma de diminuir a dimensionalidade é através da Análise de Componentes Principais (PCA), ela envolve matemática complexa, mas em termos simples, ela identifica autovetores nos dados em alta dimensão, os quais definem hiperplanos que dividem os dados, preservando ao máximo a variância (diferença) entre eles. Os dados então são projetados nesses hiperplanos que representam dimensões reduzidas.

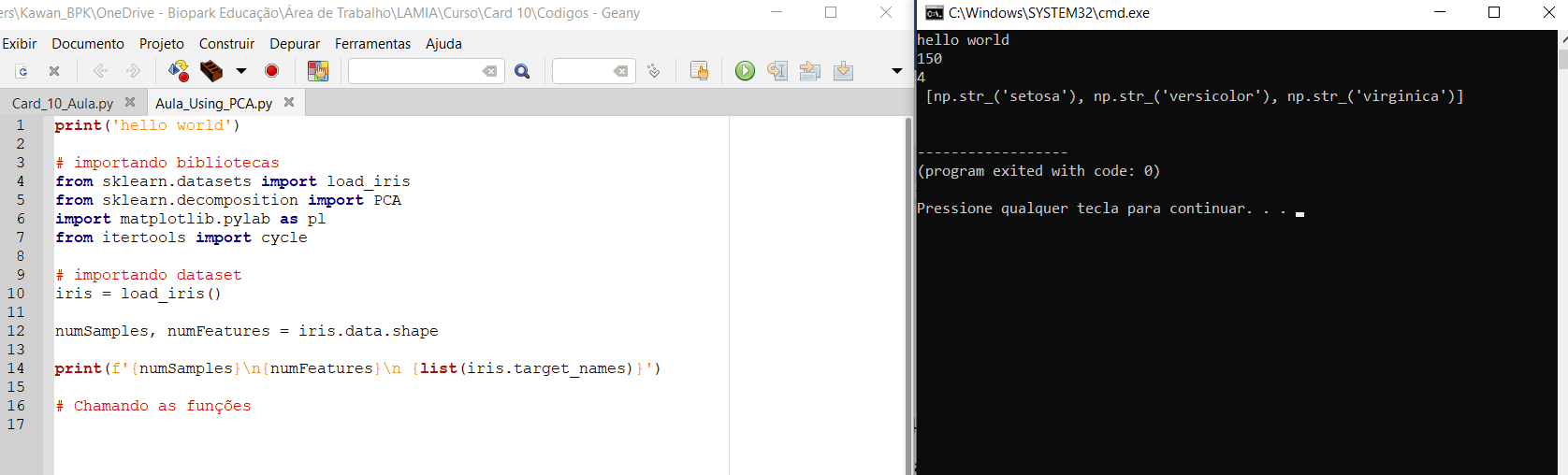
Uma implementação dessa técnica é o algoritmo, Decomposição em Valores Singulares (SVD). Essa técnica é usada para a compressão de imagens e reconhecimento facial. Um exemplo é a visualização de dados do conjunto íris, ela contém dados das pétalas e sépalas de diversas flores de íris.

Nesses dados temos o comprimento e a largura das pétalas e o comprimento e a largura das sépalas. Isso significam 4 dimensões, se quisermos plotar isso em um gráfico 2D ou 3D, precisamos diminuir as dimensões. O PCA reduz as 4 dimensões para 2 dimensões preservando ao máximo a variação, permitindo plotar os dados em um gráfico 2D, mantendo a distinção entre as espécies.

### Activity PCA Example with the Iris data set

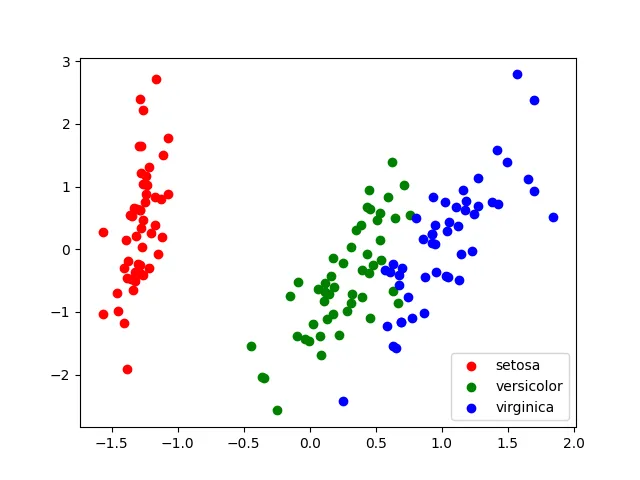
Vamos reduzir a dimensionalidade do dataset ‘iris’ lembrando que temos dados de quatro dimensões, o comprimento e a largura de sépalas e pétalas para quatro tipo de Iris (setosa, versicolor e verginica), com 150 observações.

FIGURA 8 – DATASET IRIS

 Elaborado pelo próprio autor, 2025.

Reduziremos a dimensionalidade dos dados para duas dimensões. Após isso precisamos ver o quanto de variância foi preservada para PC1 e PC2, nesse caso, 0.92 e 0.05, resultando numa variância total preservada de 0.97, isso significa que 97% da informação de quatro dimensões foi preservada em duas dimensões. Se a variância for muito baixa os dados vão apresentar muitos ruídos. Depois plotamos tudo em um gráfico.

FIGURA 9 – GRÁFICO COM DIMENSÕES REDUZIDAS

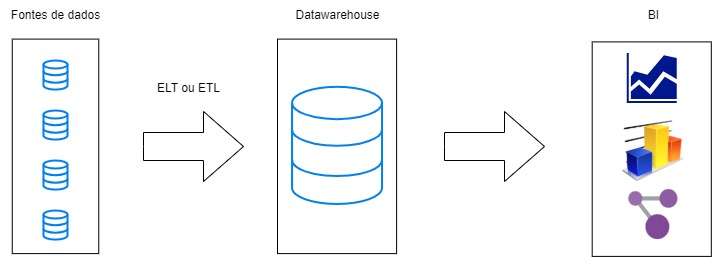
 Elaborado pelo próprio autor, 2025.

### Data Warehousing Overview ETL and ELT

Primeiro precisamos abordar o que é o DW (Data Warehouse), se trata de uma base de dados que centraliza dados de muitas fontes diferentes. Ele não centraliza todos os dados, mas apenas dados que são importantes para uma empresa ou organização. É muito utilizado pelo analista de BI (Business Inteligence) para criar relatórios. É geralmente construído através da linguagem SQL.

Departamentos inteiros são dedicados para manter o DW, primeiro porque é muito difícil manter os dados normalizados, por exemplo, como um dado se relaciona com o outro? Quais conjuntos de dados as pessoas precisam e por que? Segundo manter o fluxo de dados é um desafio contínuo, pois as fontes de dados podem mudar, ou a conexão com a internet pode cair, ou o site ou link pode mudar, etc. Terceiro, é muito difícil e custoso aumentar a escala.

FIGURA 10 – DATA WAREHOUSE

 Elaborado pelo próprio autor, 2025.

Antes de chegarmos no ‘Data Warehouse, precisamos explorar a forma como é feita a importação dos dados das diferentes fontes de dados. Existem duas formas, ETL (Extract, Transform and Load) ou ELT (Extract, Load and Transform). A primeira forma é a mais tradicional, consiste em extrair os dados de forma periódica, depois transformar os dados no esquema necessário para o DW e finalmente carregar esses dados já estruturados. Em grandes conjuntos de dados, a etapa de transformar pode ser um grande problema, com isso surge uma pergunta, como lidar com o ‘big data’?

Para isso usamos a segunda forma, o ELT consiste em primeiro extrair, depois carregar os dados para o DW e aí sim realizarmos a etapa de transformação. Existem ferramentas que ajudam no processamento desses dados contidos no DW, o ‘Hadoop’ é uma delas, ele permite o processamento paralelo para massivos volumes de dados diretamente no DW.

### Reinforcement Learning

Nós temos um agente que explora um espaço, esse agente vai aprender com os valores dos diferentes estágios e condições. Esses valores guiam o comportamento futuro do agente. A isso denominamos aprendizado por reforço. Uma implementação específica desse modo de aprendizado é o algoritmo ‘Q-learning’.

De forma simples, ele funciona da seguinte maneira, temos um conjunto de estados do ambiente (s), um conjunto de ações possíveis nesses estados (a) e um valor associado a cada par estado/ação (Q). Iniciamos com valores Q zerados, ao explorar o espaço, quando algo ruim acontece o valor de Q reduz, se algo bom acontecer o valor de Q aumenta.

Para o exemplo do jogo Pac-man, ele tem as seguintes opções dentro do jogo, ele pode bater na parede, pode morrer se encostar nos inimigos, ou caso nenhuma dessas opções aconteça, ele continua ‘vivo’. Uma vez que o ambiente do jogo é dinâmico, não calculamos um passo de cada vez, temo que calcular todas as possibilidades antes, para isso usamos a seguinte fórmula ‘Q(s,a) += discount \* (reward(s,a) + max(Q(s’)) – Q(s,a))’, onde s é o estado anterior e s’ é o estado atual.

Com isso atingimos o problema da exploração, ele consiste em descobrir qual é a forma mais eficiente de explorar os possíveis estados. Uma abordagem simples é sempre escolher a ação com a maior recompensa, porém isso pode ser muito ineficiente, pois pode ignorar atalhos ou possibilidades de caminhos por não estarem com os valores mais altos na hora da decisão. Uma outra abordagem é inserir um termo épsilon, se um número aleatório for menor, não seguir o maior Q, mas sim uma ação aleatória, dessa forma a exploração nunca acontece, porém encontrar o valor de épsilon pode ser difícil.

Essa exploração tem um termo técnico, Processo de Decisão de Markov (Markov Decision Process - MDP), elas fornecem uma estrutura matemática para modelar a tomada de decisão em situações onde os resultados são parcialmente aleatórios e parcialmente controláveis pelo agente. Elas são um processo estocástico de controle em tempo discreto. Elas são uma forma matemática de descrever o que o algoritmo de ‘Q-learning’ faz.

Programação dinâmica, é o método de resolver problemas complexos quebrando em uma coleção de subproblemas simples, resolvendo cada um desses subproblemas e armazenando seus resultados. Da próxima vez que um subproblema acontece, ao invés de recalcular tudo novamente, apenas olhamos a solução computada, salvando tempo de computação em troca de um pouco de gasto com armazenamento.

No caso do pac-man, para criar um pac-man inteligente, primeiro precisamos explorar diferentes opções de forma semi-aleatória (ações), com diferentes condições (estados). Rastrear as recompensas e aprendizados associados a cada uma das opções (Q). Usar as informações armazenadas em futuras decisões.

### Activity Reinforcement Learning & Q-Learning with Gym

Para praticar o aprendizado do algoritmos Q-learning, vamos explorar a biblioteca ‘gym’ nela conseguimos simula a seguinte situação, um táxi autônomo que precisa buscar passageiros em um local e entregar em outro local.

FIGURA 11 – TÁXI AUTÔNOMO

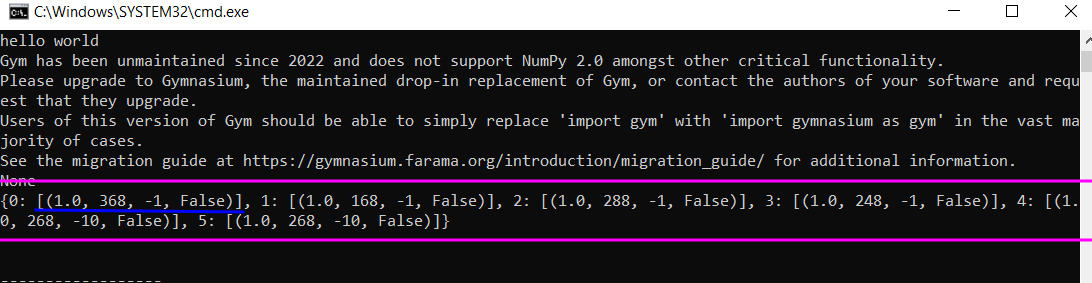
Elaborado pelo próprio autor, 2025.

Nesse caso, os pontos, amarelo, vermelho, verde e azul são onde pode haver ‘pessoas’ que o táxi precisa buscar ou locais que ele precisa deixar uma pessoa. A quantidade de possibilidade é calculada através do grid de 5x5, 4 possibilidades de destino e 5 de passageiros (4 nos pontos e 1 dentro do táxi), calculando, 5x5x4x5 = 500 possibilidades.

Para cada estado temos 6 posiveis ações, mover entre 4 direções, pegar um passageiro ou deixar um passageiro. O Q-learning, vai ter as seguintes opções deixar o passageiro no local correto (20 pontos), cada movimento com um passageiro dentro (-1 pontos) e buscar ou entregar em local incorreto (-10 pontos). Além disso, movimentos através da parede são ilegais.

Primeiro vamos definir a localização do táxi (2,3). O passageiro estará no ponto 2 e o destino no ponto 0. Após isso, imprimimos a lista de possíveis ações (5), ela é composta de 4 valores, a probabilidade de escolher uma opção, o próximo estado resultado daquela ação, a recompensa dessa ação e o indicativo de sucesso para deixar o passageiro.

FIGURA 12 – TÁXI AUTÔNOMO

 Elaborado pelo próprio autor, 2025.

Precisamos agora treinar o modelo, para isso definimos uma taxa de aprendizado (0,1), um fator de desconto (0,6), fator de exploração (0,1) e a quantidade de épocas de treinamento (10000). Com isso podemos rodar nossa simulação e ver nosso modelo autônomo de taxi funcionar.

### Understanding a Confusion Matrix

Uma matriz de confusão surge pois nem sempre a acurácia conta toda a ‘história’. Em um teste para doenças raras que possua acurácia de 99,9%, conseguiu isso ao ‘advinhar’ ‘não’ a maior parte das vezes em que foi testado. Isso implica que precisamos entender verdadeiros/falsos positivos e negativos. Uma matriz de confusão consegue nos mostrar isso.

FIGURA 13 - MATRIZ DE CONFUSÃO

|  |  |  |
| --- | --- | --- |
|  | Resultado Sim | Resultado Não |
| Predito Sim | Verdadeiro Positivo | Falso Positivo |
| Predito Não | Falso Negativo | Verdadeiro Negativo |

Elaborado pelo próprio autor, 2025.

Se o resultado de um teste for ‘sim’ e o valor predito for ‘sim’ então é um ‘verdadeiro positivo’, se o valor predito for ‘não’ então será um ‘falso negativo’. O mesmo vale para o contrário, se o resultado de um teste for ‘não’ e o valor predito for ‘sim’ então será um ‘falso positivo’, se o valor predito for ‘não’ então será um ‘verdadeiro negativo’.

### Measuring Classifiers (Precision, Recall, F1, ROC, AUC)

Baseado em nossa matriz de confusão, nós temos alguns indicadores que conseguimos medir, o primeiro que vamos abordar é o ‘Recall’, é composto da seguinte equação:

Ele possui outros nomes como ‘aka sensitive’, ‘true positive rate’ e ‘completeness’. Esse indicador é muito bom para quando se importa muito com falsos negativos (exemplo, detecção de fraude).

FIGURA 13 – DETECÇÃO DE FRAUDE

|  |  |  |
| --- | --- | --- |
|  | Fraude | Não é fraude |
| Predito ‘É fraude’ | 5 | 20 |
| Predito ‘Não é fraude’ | 10 | 100 |

Elaborado pelo próprio autor, 2025.

A segunda equação é a ‘precission’, muito parecida com a anterior, também é chamada de ‘aka correct positives’, representa a porcentagem de resultados relevantes. É uma boa escolha quando a métrica mais importante são os falsos positivos (exemplo, teste de drogas, detecção de doenças, etc).

Usando o mesmo exemplo anterior:

Outras métricas, temos:

OU

Outro indicador é a ‘ROC Curve’, Curva Característica de Operação do Receptor, Gráfico da taxa de verdadeiros positivos (recall) vs. taxa de falsos positivos em várias configurações de limiar. Pontos acima da diagonal representam boa classificação (melhor que aleatória). A curva ideal seria apenas um ponto no canto superior esquerdo. Quanto mais "dobrada" em direção ao canto superior esquerdo, melhor.

FIGURA 14 – ROC CURVE

Elaborado pelo próprio autor, 2025.

Também temos o indicador AUC, que é a área sob a curva Equivalente à probabilidade de que o classificador ranqueie uma instância positiva aleatória acima de uma negativa aleatória. AUC-ROC de 0,5 indica um classificador inútil (equivalente a aleatoriedade) e 1,0 representa perfeição. É uma Métrica comum para comparação de classificadores.

## Dealing with Real-World Data

### Bias Variance Tradeoff

Quando tratamos de treinamento de máquina, dois conceitos são muito importantes, o viés e variância. O primeiro refere-se a distância da média dos seus valores preditos comparado com a resposta "real". O segundo, é o quão disperso os valores preditos estão da resposta "real".

FIGURA X – EXEMPLO VIÉS, VARIÂNCIA

Elaborado pelo próprio autor, 2025.

Nesse exemplo, temos baixa variância e alto viés nesse caso podemos dizer que os dados são underfitting, ou seja, o modelo é muito simples não com seguindo capturar a relação entre os dados de entrada e saída. O resultado disso, é um desempenho ruim do modelo.

FIGURA X – EXEMPLO VIÉS, VARIÂNCIA

Elaborado pelo próprio autor, 2025.

Nesse exemplo, temos alta variância e baixo viés nesse caso podemos dizer que o modelo está overfitting, ou seja, o modelo é muito complexo se ajustando demais aos dados de treinamento, incluindo o ‘erro’. O resultado disso, é um desempenho ruim do modelo com os dados dos testes.

O que realmente importa é que tanto o viés como a variância contribuem para o erro do modelo, e o independente da complexidade do modelo, estamos sempre tentando diminuir o erro. A fórmula matemática para isso é:

### Activity K-Fold Cross-Validation to avoid overfitting

### Data Cleaning and Normalization

### Activity Cleaning web log data

### Normalizing numerical data

### Activity Detecting outliers

### Feature Engineering and the Curse of Dimensionality

### Imputation Techniques for Missing Data

### Handling Unbalanced Data Oversampling, Undersampling, and SMOTE

### Binning, Transforming, Encoding, Scaling, and Shuffling

# Conclusões

# Referencias

As referências foram apenas os vídeos da atividade.