|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dia** | **Aspecto** | **Temperatura** | **Umidade** | **Vento** | **Decisão** |
| **1** | Sol | Quente | Alta | Fraco | Não |
| **2** | Sol | Quente | Alta | Forte | Não |
| **3** | Nublado | Quente | Alta | Fraco | Sim |
| **4** | Chuva | Agradável | Alta | Fraco | Sim |
| **5** | Chuva | Fria | Normal | Fraco | Sim |
| **6** | Chuva | Fria | Normal | Forte | Não |
| **7** | Nublado | Fria | Normal | Forte | Sim |
| **8** | Sol | Agradável | Alta | Fraco | Não |
| **9** | Sol | Fria | Normal | Fraco | Sim |
| **10** | Chuva | Agradável | Normal | Fraco | Sim |
| **11** | Sol | Agradável | Normal | Forte | Sim |
| **12** | Nublado | Agradável | Alta | Forte | Sim |
| **13** | Nublado | Quente | Normal | Fraco | Sim |
| **14** | Chuva | Agradável | Alta | Forte | Não |

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **kg** | **m²** | **Classe** |
| **1** | 4 | 2 | A |
| **2** | 3 | 3.5 | A |
| **3** | 4 | 4 | A |
| **4** | 4 | 5 | A |
| **5** | 5 | 3 | A |
| **6** | 6.5 | 5 | B |
| **7** | 7.5 | 1 | B |
| **8** | 8 | 4 | B |
| **9** | 9 | 4.5 | B |
| **10** | 9.5 | 2 | B |
| **11** | 6 | 4 | ? |

|  |
| --- |
|  |

K=5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **kg** | **m²** | **Classe** | **Distância para a instância #11** |
| **1** | 2 | 4 | A | 4 |
| **2** | 3 | 3.5 | A | 3.04 |
| **3** | 4 | 4 | A | 2 |
| **4** | 4 | 5 | A | 2.23 |
| **5** | 5 | 3 | A | 1.41 |
| **6** | 6.5 | 5 | B | 1.11 |
| **7** | 7.5 | 1 | B | 3.35 |
| **8** | 8 | 4 | B | 2 |
| **9** | 9 | 4.5 | B | 3.08 |
| **10** | 9.5 | 2 | B | 4.03 |
| **11** | 6 | 4 | A |  |

|  |
| --- |
| **Algoritmo 1 – Exemplo de código em Python** |
| **import** sys  **from** PyQt5 **import** uic**,** QtGui  **from** PyQt5**.**QtWidgets **import** QFileDialog**,** QApplication**,** QWidget**,** QLabel**,** QTableWidget**,** QTableWidgetItem**,** QListWidget**,** QListWidgetItem  **import** pandas **as** pd  **from** sklearn **import** preprocessing  **from** sklearn**.**neighbors **import** KNeighborsClassifier  **from** sklearn**.**metrics **import** accuracy\_score  **from** sklearn**.**tree **import** DecisionTreeClassifier  **from** sklearn**.**naive\_bayes **import** GaussianNB  **from** sklearn**.**model\_selection **import** KFold  **import** numpy **as** np  **from** scipy **import** stats  Ui\_MainWindow**,** QtBaseClass **=** uic**.**loadUiType**(**"interface.ui"**)** |

|  |  |
| --- | --- |
| **Cross Validation** | |
| **Classificadores** | **Acurácia** |
| **KNN (K=5)** | 90.18% |
| **C4.5** | 91.13% |
| **Naive Bayes** | 52.33% |

|  |  |  |
| --- | --- | --- |
| **P-VALUES** | | |
| **KNN X C4.5** | **KNN X NB** | **C4,5 X NB** |
| 0.033 | 0.0 | 0.0 |

|  |  |
| --- | --- |
| **Holdout** | |
| **Classificadores** | **Acurácia** |
| **KNN (K=5)** | 89.05% |
| **C4.5** | 90.05% |
| **Naive Bayes** | 52.68% |

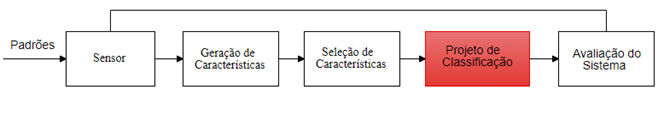


Figura introdução

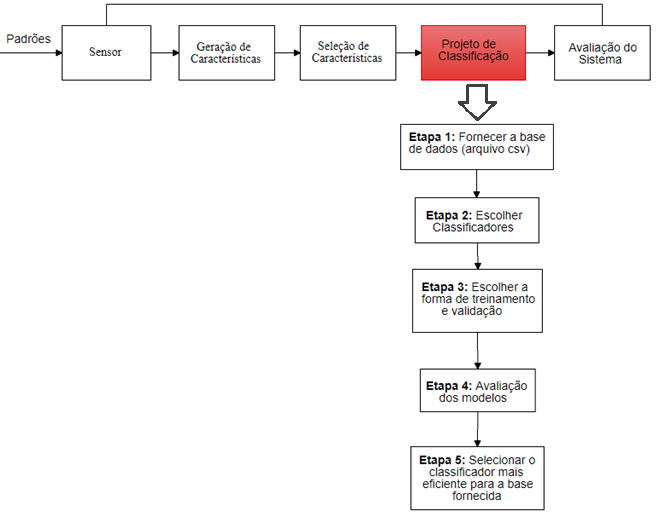


Figura metodologia

|  |
| --- |
| **Algoritmo 2 – Exemplo de código em Python** |
| **for** treino\_index**,** teste\_index **in** kf**.**split**(**x**):**  x\_treino**,** x\_teste **=** x**[**treino\_index**],** x**[**teste\_index**]**  y\_treino**,** y\_teste **=** y**[**treino\_index**],** y**[**teste\_index**]**    #knn escolhido  **if** self**.**cbKnn**.**isChecked**()** **and** self**.**inputK**.**text**()!=**"" **:**  cont\_knn **=** 1  #vizinhos, o usuario escolhe o valor de k  k **=** int**(**self**.**inputK**.**text**())**  classificador\_KNN **=** KNeighborsClassifier**(**n\_neighbors**=**k**)**  #KNN  #cria o modelo do KNN = treinamento  classificador\_KNN**.**fit**(**x\_treino**,** y\_treino**)**  # classifica os dados do x\_teste  y\_predict**=**classificador\_KNN**.**predict**(**x\_teste**)**  #adiciona a acurácia na lista do Knn  list\_knn**.**append**(**accuracy\_score**(**y\_teste**,** y\_predict**,**  normalize**=True))**  **if** self**.**cbArvores**.**isChecked**()** **:**  cont\_arvores **=** 1  classificador\_arvores **=** DecisionTreeClassifier**(**random\_state**=**0**)**  #Arvore  #cria o modelo do Arvore = treinamento  classificador\_arvores**.**fit**(**x\_treino**,** y\_treino**)**  # classifica os dados do x\_teste  y\_predict**=**classificador\_arvores**.**predict**(**x\_teste**)**  #adiciona a acurácia na lista do arvore  list\_arvores**.**append**(**accuracy\_score**(**y\_teste**,** y\_predict**,**  normalize**=True))**    **if** self**.**cbNb**.**isChecked**()** **:**  cont\_nb **=** 1  classificador\_Nb **=** GaussianNB**()**  #NB  #cria o modelo do NB = treinamento  classificador\_Nb**.**fit**(**x\_treino**,** y\_treino**)**  #classifica os dados do x\_teste  y\_predict**=**classificador\_Nb**.**predict**(**x\_teste**)**  #adiciona a acurácia na lista da arvore  list\_nb**.**append**(**accuracy\_score**(**y\_teste**,** y\_predict**,**  normalize**=True))** |