Inteligênica Artificial – PCS3438
Lista de Exercícios – Professor Eduardo

Arthur Font Gouveia – 12036152

1) Trecho principal do script em python:

```
----------------------------------------------------------------------------------------------------------------
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()                                  # algoritmo Naive Bayes Gaussiano
nb.fit(data.iloc[0:350,], target.iloc[0:350])      # 350 primeiras linhas utilizadas para treino

train_accuracy = sk.metrics.accuracy_score(target.iloc[0:350], nb.predict(data.iloc[0:350]))
test_accuracy = sk.metrics.accuracy_score(target.iloc[350:1000], nb.predict(data.iloc[350:1000]))
----------------------------------------------------------------------------------------------------------------
```

Resultados:

- Train Accuracy = 0.76
- Test Accuracy = 0.627

2) Trecho principal do script em python:

```
----------------------------------------------------------------------------------------------------------------
from sklearn.neighbors import KNeighborsClassifier
folds5 = Kfold(n_splits=5)              # 5 pastas
train_accuracy = []
test_accuracy = []

for train_index, test_index in folds5.split(table): # Validação cruzada k-fold com 5 pastas
    knn = KNeighborsClassifier(n_neighbors=10)      # KNN com k=10
    knn.fit(data.iloc[train_index], target.iloc[train_index])

    train_accuracy.append(sk.metrics.accuracy_score(target.iloc[train_index],
                                knn.predict(data.iloc[train_index])))
    test_accuracy.append(sk.metrics.accuracy_score(target.iloc[test_index],
                                knn.predict(data.iloc[test_index])))
----------------------------------------------------------------------------------------------------------------
```

Resultados:

- Average train accuracy: 0.866
- Average test accuracy: 0.838

3) Trecho principal do script em python:

```
----------------------------------------------------------------------------------------------------------------
from sklearn.linear_model import Lasso
folds1000 = Kfold(n_splits=1000)     # 1000 pastas
train_accuracy = []
test_accuracy = []

for train_index, test_index in folds1000.split(table): # Leave-one-out com 1000 pastas
    lasso = Lasso()                                             # LASSO com α = 1
    lasso.fit(data.iloc[train_index], target.iloc[train_index])

    train_accuracy.append(np.sqrt(sk.metrics.mean_squared_error(target.iloc[train_index],
                                        lasso.predict(data.iloc[train_index]))))
    test_accuracy.append(np.sqrt(sk.metrics.mean_squared_error(target.iloc[test_index],
                                        lasso.predict(data.iloc[test_index]))))
----------------------------------------------------------------------------------------------------------------
```

Resultados:

- Root Mean Square Error (**RMSE**) train: 19.220
- Root Mean Square Error (**RMSE**) test: 15.465

4) Trecho principal do script em python:

```
--------------------------------------------------------------------------------------------------------------------------------
from sklearn.tree import DecisionTreeRegressor
folds5 = Kfold(n_splits=5)      # 5 pastas
train_accuracy = []
test_accuracy = []

for train_index, test_index in folds5.split(table):
    tree = DecisionTreeRegressor()                          # Árvore de Regressão
    tree.fit(data.iloc[train_index], target.iloc[train_index])

    train_accuracy.append(sk.metrics.mean_absolute_error(target.iloc[train_index],
tree.predict(data.iloc[train_index])))
    test_accuracy.append(sk.metrics.mean_absolute_error(target.iloc[test_index],
tree.predict(data.iloc[test_index])))
--------------------------------------------------------------------------------------------------------------------------------
```

Resultados:

- Mean Square Error (**MSE**) train: 0.0
- Mean Square Error (**MSE**) test: 44.322