Modelos de Machine Learning - PySpark e Python

Preparação dos dados

Substituir
```
dataset = df.replace('null', None).dropna(how='any')
```

Mostrar os dados
```
dataset.show()
```

Preparação dos dados para o Modelo
```
required_features = [...]
from pyspark.ml.feature import VectorAssembler
assembler = VectorAssembler(inputCols=required_features,
outputCol='features')
transformed_data = assembler.transform(dataset)
```

Separação em treino e validação
```
(training_data, test_data) = transformed_data.randomSplit([0.8,0.2])
```

Modelos

1) Random Forest Classifier
Definindo os parâmetros
```
from pyspark.ml.classification import RandomForestClassifier
rf = RandomForestClassifier(labelCol='Outcome', featuresCol='features',
maxDepth=5)
```

Ajustando o modelo
```
model = rf.fit(training_data)
```

Scorando a base
```
rf_predictions = model.transform(test_data)
```

Avaliando o modelo
```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
multi_evaluator = MulticlassClassificationEvaluator(labelCol =
'Outcome', metricName = 'accuracy')
print('Random Forest classifier Accuracy:',
multi_evaluator.evaluate(rf_predictions))
```

2) Decision Tree Classifier
Definindo os parâmetros
```
from pyspark.ml.classification import DecisionTreeClassifier
dt = DecisionTreeClassifier(featuresCol = 'features', labelCol = 'Outcome', maxDepth = 3)
```

Ajustando o modelo

```
dtModel = dt.fit(training_data)
```

Scorando a base

```
dt_predictions = dtModel.transform(test_data)
```

Avaliando o modelo

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
multi_evaluator = MulticlassClassificationEvaluator(labelCol = 
'Outcome', metricName = 'accuracy')
print('Decision Tree Accuracy:', 
multi_evaluator.evaluate(dt_predictions))
```

3) Regressão Logística
Definindo os parâmetros

```
from pyspark.ml.classification import LogisticRegression
lr = LogisticRegression(featuresCol = 'features', labelCol = 'Outcome', 
maxIter=10)
```

Ajustando o modelo

```
lrModel = lr.fit(training_data)
```

Scorando a base

```
lr_predictions = dtModel.transform(test_data)
```

Avaliando o modelo

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
multi_evaluator = MulticlassClassificationEvaluator(labelCol = 
'Outcome', metricName = 'accuracy')
print('Logistic Regression Accuracy:', 
multi_evaluator.evaluate(lr_predictions))
```

4) Modelo Gradient Boosting
Definindo os parâmetros

```
from pyspark.ml.classification import GBTClassifier
gb = GBTClassifier(labelCol = 'Outcome', featuresCol = 'features')
```

Ajustando o modelo

```
gbModel = gb.fit(training_data)
```

Scorando a base

```
gb_predictions = gbModel.transform(test_data)
```

Avaliando o modelo

```
print('Gradient-boosted Trees Accuracy:', 
multi_evaluator.evaluate(gb_predictions))
```

Referências

`https://github.com/harunurrashid97/Machine-learning-with-PySpark/blob/master/First%20PySpark%20ml%20model/First%20ML%20models%20using%20PySpark.ipynb`