```
from pyspark.ml.linalg import Vectors
denseVec = Vectors.dense(1.0, 2.0, 3.0)
size = 3
idx = [1, 2] # locations of non-zero elements in vector
values = [2.0, 3.0]
sparseVec = Vectors.sparse(size, idx, values)


# COMMAND ----------

df = spark.read.json("/data/simple-ml")
df.orderBy("value2").show()


# COMMAND ----------

from pyspark.ml.feature import RFormula
supervised = RFormula(formula="lab ~ . + color:value1 + color:value2")


# COMMAND ----------

fittedRF = supervised.fit(df)
preparedDF = fittedRF.transform(df)
preparedDF.show()


# COMMAND ----------

train, test = preparedDF.randomSplit([0.7, 0.3])


# COMMAND ----------

from pyspark.ml.classification import LogisticRegression
lr = LogisticRegression(labelCol="label",featuresCol="features")


# COMMAND ----------

print lr.explainParams()


# COMMAND ----------

fittedLR = lr.fit(train)


# COMMAND ----------

train, test = df.randomSplit([0.7, 0.3])


# COMMAND ----------

rForm = RFormula()
lr = LogisticRegression().setLabelCol("label").setFeaturesCol("features")
```

```
# COMMAND ----------

from pyspark.ml import Pipeline
stages = [rForm, lr]
pipeline = Pipeline().setStages(stages)


# COMMAND ----------

from pyspark.ml.tuning import ParamGridBuilder
params = ParamGridBuilder()\
  .addGrid(rForm.formula, [
    "lab ~ . + color:value1",
    "lab ~ . + color:value1 + color:value2"])\
  .addGrid(lr.elasticNetParam, [0.0, 0.5, 1.0])\
  .addGrid(lr.regParam, [0.1, 2.0])\
  .build()


# COMMAND ----------

from pyspark.ml.evaluation import BinaryClassificationEvaluator
evaluator = BinaryClassificationEvaluator()\
  .setMetricName("areaUnderROC")\
  .setRawPredictionCol("prediction")\
  .setLabelCol("label")


# COMMAND ----------

from pyspark.ml.tuning import TrainValidationSplit
tvs = TrainValidationSplit()\
  .setTrainRatio(0.75)\
  .setEstimatorParamMaps(params)\
  .setEstimator(pipeline)\
  .setEvaluator(evaluator)


# COMMAND ----------

tvsFitted = tvs.fit(train)


# COMMAND ----------
```