Project_MLmodel

June 8, 2024

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from pyspark.sql import SparkSession
from pyspark.ml import Pipeline
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.classification import DecisionTreeClassifier
from pyspark.ml.classification import RandomForestClassifier
from pyspark.ml.classification import GBTClassifier
from sklearn.metrics import accuracy_score
```

0.1 LOADING TRANSFORMED DATA

[ ]:
```
spark = SparkSession.builder.appName('KickStarter_ML').getOrCreate()
df = spark.read.csv('kickstarter_cleaned.csv', header = True, inferSchema =␣
 ↪True)
df.printSchema()

# show features column
df.show(5)
```

```
root
 |-- main_category: integer (nullable = true)
 |-- currency: string (nullable = true)
 |-- deadline: date (nullable = true)
 |-- launched: date (nullable = true)
 |-- state: integer (nullable = true)
 |-- backers: integer (nullable = true)
 |-- country: integer (nullable = true)
 |-- usd_pledged_real: integer (nullable = true)
 |-- usd_goal_real: integer (nullable = true)
 |-- duration in days: integer (nullable = true)

+-------------+--------+----------+----------+-----+-------+-------+------------
----+-------------+----------------+
|main_category|currency|  deadline|
launched|state|backers|country|usd_pledged_real|usd_goal_real|duration in days|
```

```
+-------------+--------+----------+----------+-----+-------+-------+-----------
----+-------------+----------------+
|            6|     USD|2012-08-10|2012-07-07|    0|     12|     21|
296|         4000|              34|
|            6|     USD|2014-01-05|2013-11-21|    1|    148|     21|
25712|        25000|              45|
|           12|     USD|2011-05-16|2011-04-16|    0|      0|     21|
0|          200|              30|
|           13|     USD|2015-06-10|2015-05-11|    1|    298|     21|
23447|        10000|              30|
|            8|     GBP|2013-11-30|2013-10-31|    1|    122|      9|
7660|         3268|              30|
+-------------+--------+----------+----------+-----+-------+-------+-----------
----+-------------+----------------+
only showing top 5 rows
```

0.2 Prepping our pipeline

[ ]:
```
numericCols = ['main_category','country','usd_goal_real','duration in days']
featurizationPipeline = Pipeline(stages =␣
 ↪[VectorAssembler(inputCols=numericCols, outputCol="feature_vector")])
featurizationPipelineModel = featurizationPipeline.fit(df)
df = featurizationPipelineModel.transform(df)
train, test = df.randomSplit([0.7, 0.3], seed = 2018)
```

0.3 Logistic Regression model

[ ]:
```
lr = LogisticRegression(featuresCol = 'feature_vector', labelCol = 'state',␣
 ↪maxIter=10)
lrModel = lr.fit(train)

# Make predictions on the test set.
predictions = lrModel.transform(test)
```

0.3.1 Model Evaluating.

[ ]:
```
true_labels=predictions.select('state')
lr_predictions=predictions.select('prediction')

accuracy = accuracy_score(true_labels.toPandas(), lr_predictions.toPandas())
print("Logistic Regression Accuracy =",accuracy*100,"%")
```

```
Logistic Regression Accuracy = 64.22191810422444 %
```

0.4 Decision tree classifier

```
[ ]: dt = DecisionTreeClassifier(featuresCol = 'feature_vector', labelCol = 'state',␣
      ↪maxDepth = 3)

     dtModel = dt.fit(train)
     predictions = dtModel.transform(test)
```

0.4.1 Evaluation

```
[ ]: true_labels=predictions.select('state')
     dt_predictions=predictions.select('prediction')

     accuracy = accuracy_score(true_labels.toPandas(), dt_predictions.toPandas())
     print("Decision Tree Accuracy =",accuracy*100,"%")
```

```
Decision Tree Accuracy = 65.34735477365217 %
```

0.5 Random Forest Classifier

```
[ ]: rf = RandomForestClassifier(featuresCol = 'feature_vector', labelCol = 'state',␣
      ↪numTrees=10)
     rfModel = rf.fit(train)
     predictions = rfModel.transform(test)
```

0.5.1 Evaluation

```
[ ]: true_labels=predictions.select('state')
     rf_predictions=predictions.select('prediction')

     accuracy = accuracy_score(true_labels.toPandas(), rf_predictions.toPandas())
     print("Random Forest Accuracy =",accuracy*100,"%")
```

```
Random Forest Accuracy = 65.98210105520941 %
```

0.6 Gradient Boosted tree classifier

```
[ ]: gbt = GBTClassifier(featuresCol = 'feature_vector', labelCol = 'state',␣
      ↪maxIter=10)
     gbtModel = gbt.fit(train)
     predictions = gbtModel.transform(test)
```

0.6.1 Evaluation

```
[ ]: true_labels=predictions.select('state')
     gbt_predictions=predictions.select('prediction')

     accuracy = accuracy_score(true_labels.toPandas(), gbt_predictions.toPandas())
     print("Gradient Boosted Tree Accuracy =",accuracy*100,"%")
```

```
Gradient Boosted Tree Accuracy = 67.28490654373897 %
```

0.6.2 It is understandable why the logistic regression model performed slightly worse than its peers , due the the high number of outliers across our data set , however 79% is considered acceptable