Predictive Modeling using Spark

Spark DS 6003-001

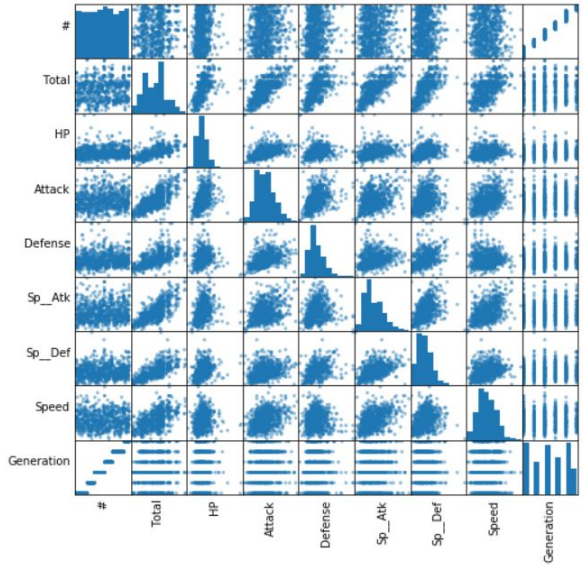
Murugesan Ramakrishnan

Motivation

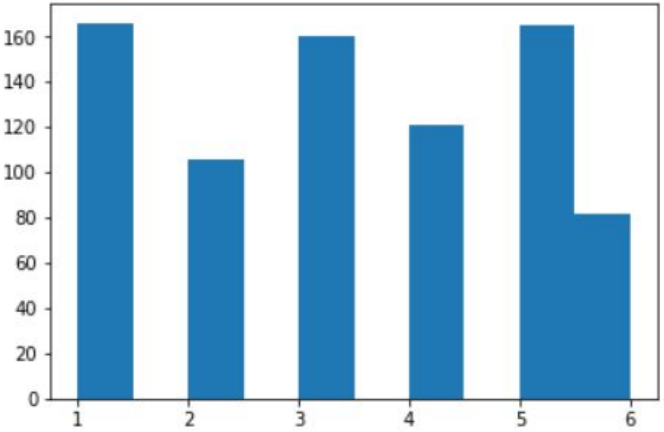
- There are 1000+ Pokemons - fictional characters - each with different strengths, weaknesses belonging to different generation
- The objective is to classify a Pokemon to its corresponding generation (1 - Nascent stage, 6 - Developed stage)
- Spark coupled with S3 storage has been used to store and retrieve data
- Turns out the result is not acceptable which shows that the selected features are not enough to predict the Generation of a Pokemon

Visualization

Correlation Analysis



Distribution of the Response Variable



Code Snippets

Vectorization

In [31]:
```
from pyspark.ml.linalg import Vectors
from pyspark.ml.feature import VectorAssembler
```

In [32]:
```
# vectorize the data frame features
assembler = VectorAssembler(
    inputCols=df_model.columns[:7],
    outputCol="features")

trainingVDF = assembler.transform(trainingDF)
testVDF = assembler.transform(testDF)
```

In [33]:
```
# define label columns
trainingVDF = trainingVDF.withColumnRenamed("Generatic
testVDF = testVDF.withColumnRenamed("Generation", "lab
```

Multinomial Logistic Regression

```
#from pyspark.ml.regression import LinearRegression, LinearRegress
from pyspark.ml.classification import LogisticRegression
```

```
lr = LogisticRegression(maxIter=10, regParam=0.3,\
                        elasticNetParam=0.8,family="multinomial")

# Fit the model
lrModel = lr.fit(trainingVDF)
```

```
#### Predicting the values for test dataset
predictionsAndLabelsDF = lrModel.transform(testVDF)

print(predictionsAndLabelsDF.\
      orderBy(predictionsAndLabelsDF.label.desc()).take(1))
```