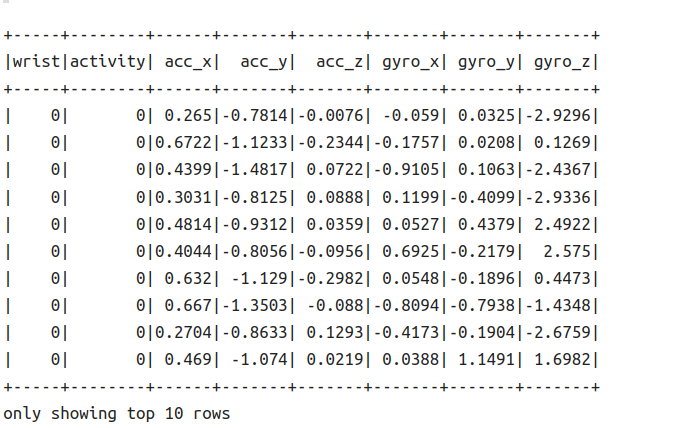
**MoveOn fitness Tracker**

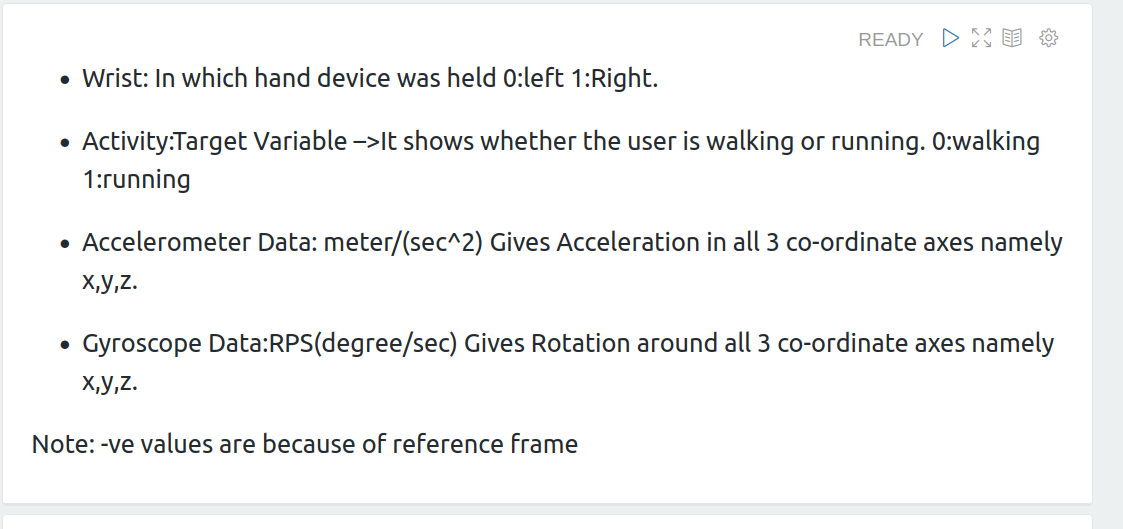
Data of accelerometer and gyroscope from apple user.

**Data link -** [Kinematics Motion Data | Kaggle](https://www.kaggle.com/yasserh/kinematics-motion-data)

**Data:**

****

**Data Description:**

****

* [**Logistic regression**](https://spark.apache.org/docs/latest/ml-classification-regression.html#logistic-regression)

**code-**

%pyspark

#Logistic Regression

from pyspark.ml.classification import LogisticRegression

lr = LogisticRegression(featuresCol='scaledFeatures', labelCol='Target')

# Fit the model

LRModel = lr.fit(train)

# Predictions

predictions=LRModel.transform(X\_test)

**Output-**

****

* [**Decision tree classifier**](https://spark.apache.org/docs/latest/ml-classification-regression.html#decision-tree-classifier)

**Code-**

%pyspark

#Random Forest

from pyspark.ml.classification import DecisionTreeClassifier

tree = DecisionTreeClassifier(featuresCol='scaledFeatures', labelCol='Target', maxDepth=10)

# Fit the model

DTModel = tree.fit(train)

# Predictions

predictions=DTModel.transform(X\_test)

**Output-**

****

* [**Random forest classifier**](https://spark.apache.org/docs/latest/ml-classification-regression.html#random-forest-classifier)

**Code-**

%pyspark

#Random Forest

from pyspark.ml.classification import RandomForestClassifier

rf = RandomForestClassifier(featuresCol='scaledFeatures', labelCol='Target', maxDepth=10)

# Fit the model

rfModel = rf.fit(train)

# Predictions

predictions=rfModel.transform(X\_test)

**Output-**

****

* [**Gradient-boosted tree classifier**](https://spark.apache.org/docs/latest/ml-classification-regression.html#gradient-boosted-tree-classifier)

**Code-**

%pyspark

#Gradient Boost

from pyspark.ml.classification import GBTClassifier

gb = GBTClassifier(featuresCol='scaledFeatures', labelCol='Target')

# Fit the model

GBModel = gb.fit(train)

# Predictions

predictions=GBModel.transform(X\_test)

**Output-**

****

* [**Linear Support Vector Machine**](https://spark.apache.org/docs/latest/ml-classification-regression.html#linear-support-vector-machine)

**Code-**

%pyspark

#SVC

from pyspark.ml.classification import LinearSVC

svc = LinearSVC(featuresCol='scaledFeatures', labelCol='Target')

# Fit the model

SVCModel = svc.fit(train)

# Predictions

predictions=SVCModel.transform(X\_test)

**Output-**

****

* [**Naive Bayes**](https://spark.apache.org/docs/latest/ml-classification-regression.html#naive-bayes)

**Code-**

%pyspark

#Naive Bayes

from pyspark.ml.classification import NaiveBayes

nb = NaiveBayes(featuresCol='scaledFeatures', labelCol='Target',smoothing=1.0)

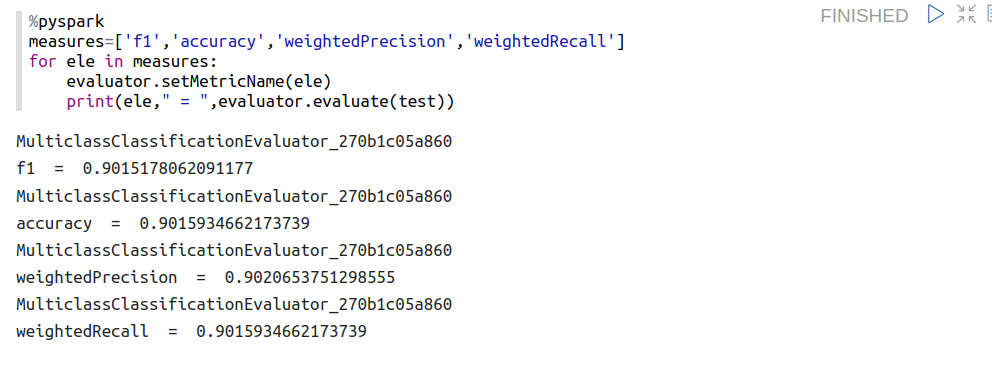
# Fit the model

NBModel = nb.fit(train)

# Predictions

predictions=NBModel.transform(X\_test)

**Output-**



**Comparison of models: 80-20 split**

| Model Name | Accuracy | Precision | Recall | F1-Score |
| --- | --- | --- | --- | --- |
| [Logistic regression](https://spark.apache.org/docs/latest/ml-classification-regression.html#logistic-regression) | 0.8648 | 0.8674 | 0.8648 | 0.8644 |
| [Decision tree classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#decision-tree-classifier) | 0.9805 | 0.9807 | 0.9805 | 0.9805 |
| [Random forest classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#random-forest-classifier) | 0.9842 | 0.9842 | 0.9842 | 0.9842 |
| [Gradient-boosted tree classifier](https://spark.apache.org/docs/latest/ml-classification-regression.html#gradient-boosted-tree-classifier) | 0.9783 | 0.9784 | 0.9783 | 0.9783 |
| [Linear Support Vector Machine](https://spark.apache.org/docs/latest/ml-classification-regression.html#linear-support-vector-machine) | 0.8717 | 0.8823 | 0.8717 | 0.8707 |
| [Naive Bayes](https://spark.apache.org/docs/latest/ml-classification-regression.html#naive-bayes) | 0.5574 | 0.5573 | 0.5574 | 0.5571 |

**Conclusion:**

Random Forest classifies data more accurately but gradient boost also has nearly same accuracy, we will go ahead with the gradient boost classifier here in order to work on boosting algorithm.