Understanding the Relationship between Tenure and Total Charges for Telecom Customers

Motivation
To validate the hypothesis that the longer a customer stays in the service, the more he or she tends to spend. Since both features could be important in predicting churn, it might be helpful to first understand the relationship between the two.

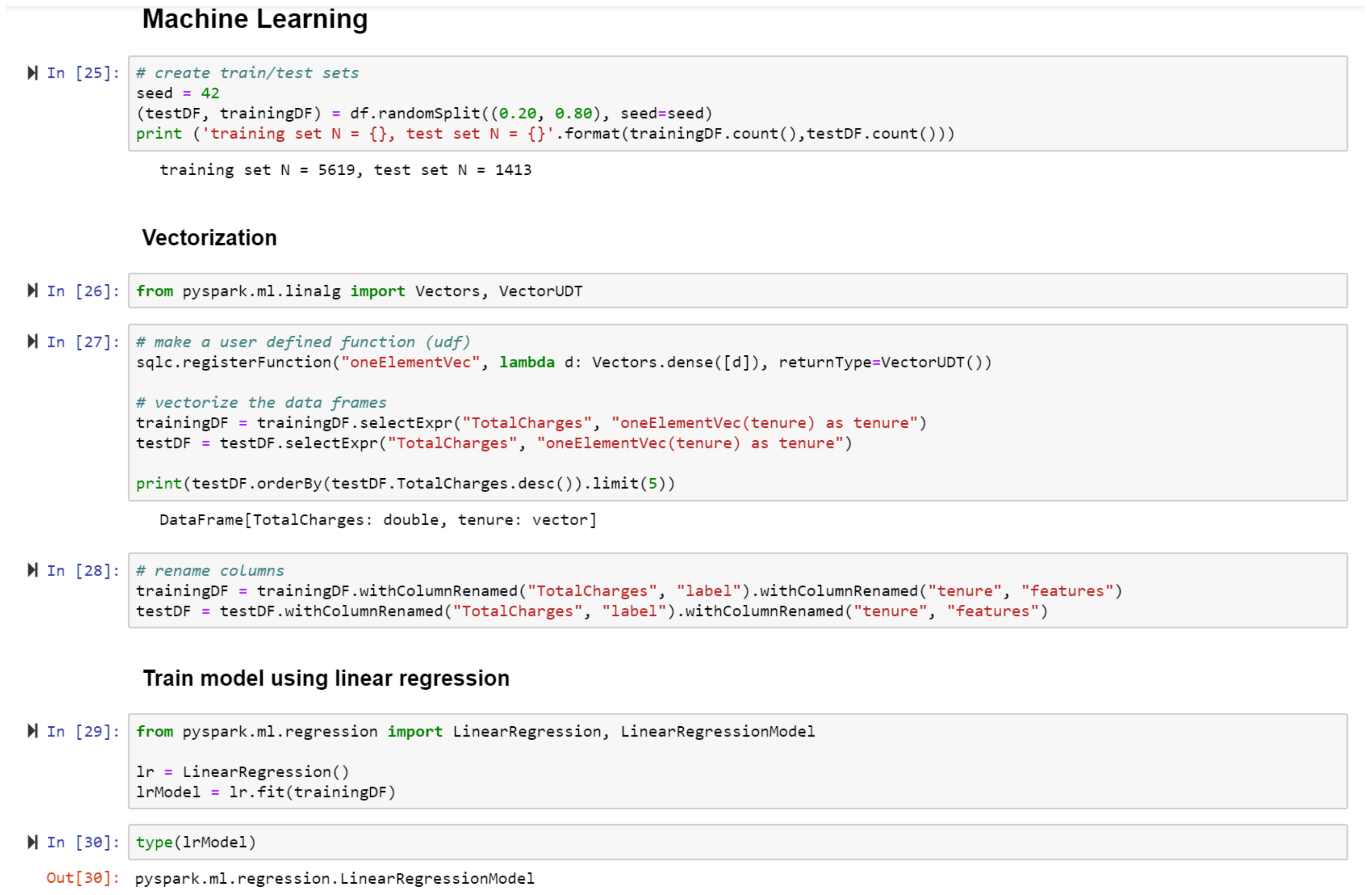
Content of the Dataset
Each row represents a customer, each column contains customer's attributes described on the column Metadata.

The data set includes information about:

- *Customers who left within the last month – the column is called Churn*
- *Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies*
- *Customer account information – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges*
- *Demographic info about customers – gender, age range, and if they have partners and dependents*

Bochao Li, bl2fh

Code Snippet and Explanation



Machine Learning

```
In [25]: # create train/test sets
         seed = 42
         (testDF, trainingDF) = df.randomSplit((0.20, 0.80), seed=seed)
         print ('training set N = {}, test set N = {}'.format(trainingDF.count(),testDF.count()))

         training set N = 5619, test set N = 1413
```

Vectorization

```
In [26]: from pyspark.ml.linalg import Vectors, VectorUDT

In [27]: # make a user defined function (udf)
         sqlc.registerFunction("oneElementVec", lambda d: Vectors.dense([d]), returnType=VectorUDT())

         # vectorize the data frames
         trainingDF = trainingDF.selectExpr("TotalCharges", "oneElementVec(tenure) as tenure")
         testDF = testDF.selectExpr("TotalCharges", "oneElementVec(tenure) as tenure")

         print(testDF.orderBy(testDF.TotalCharges.desc()).limit(5))

         DataFrame[TotalCharges: double, tenure: vector]

In [28]: # rename columns
         trainingDF = trainingDF.withColumnRenamed("TotalCharges", "label").withColumnRenamed("tenure", "features")
         testDF = testDF.withColumnRenamed("TotalCharges", "label").withColumnRenamed("tenure", "features")
```

Train model using linear regression

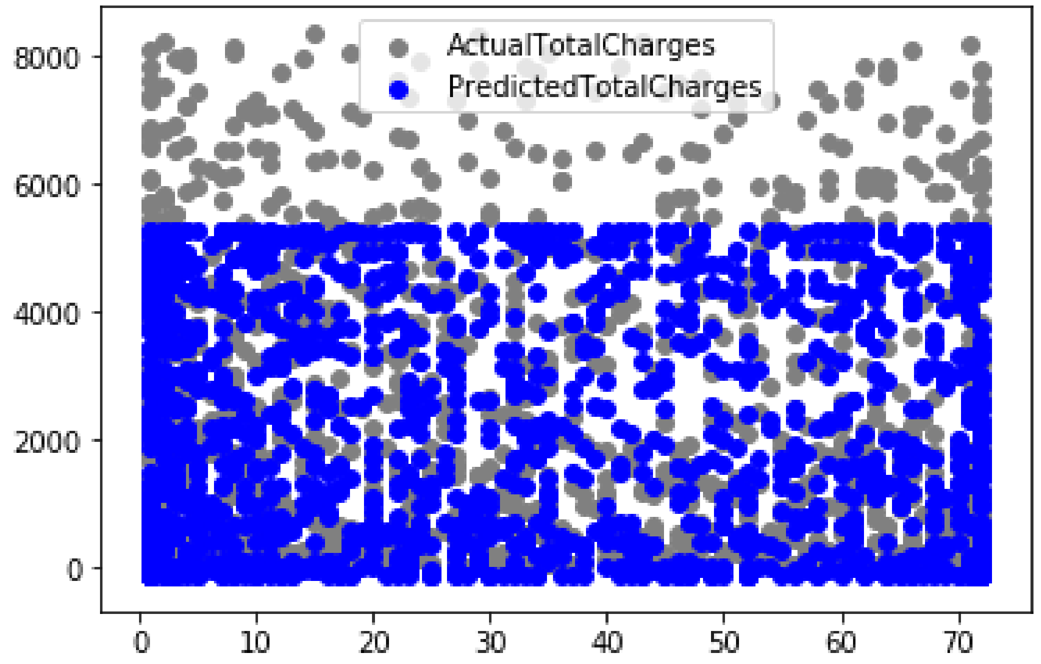
```
In [29]: from pyspark.ml.regression import LinearRegression, LinearRegressionModel

         lr = LinearRegression()
         lrModel = lr.fit(trainingDF)

In [30]: type(lrModel)

Out[30]: pyspark.ml.regression.LinearRegressionModel
```

Methodology

1. Set up environment
   1. *Create Spark context*
   2. *Download dataset from Kaggle and read csv into s3*
   3. *Write parquet to s3 and write to spark data frame from parquet*
2. MLlib Analysis
   1. *Set up sample data frame*
   2. *Explore features*
   3. *Format target features*
3. Machine Learning
   1. *Separate into train and test datasets*
   2. *Vectorize features*
   3. *Train model using linear regression*
   4. *Evaluate model*
4. Visualize regression result

Bochao Li, bl2fh

Visualization



Result
Weird as it looks, the predicted values seem to be capped within thresholds, although overall the scatter points are distributed similarly to the actual values.

Bochao Li, bl2fh